



DEPARTMENT FÜR INFORMATIK
SYSTEMSOFTWARE UND VERTEILTE SYSTEME

Konzeption und Implementierung einer Touristen-App für die Gemeinde Hude

Masterarbeit

30. September 2013

Björn-Hendrik von der Linde
Friedensheimer Weg 2
26419 Schortens

Erstprüfer
Zweitprüfer

Prof. Dr.-Ing. Oliver Theel
Robert Schadek, M.Sc. Inf.

Erklärung zur Urheberschaft

Hiermit versichere ich, dass ich diese Arbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel und Quellen benutzt habe.

Oldenburg, den 30. September 2013

Björn-Hendrik von der Linde

Der binnenländische Tourismus in Deutschland floriert zusehends, ebenso wie die Verbreitung und Nutzung von Smartphones und Apps. In Kombination beider Einflüsse ist eine vermehrte Nutzung von Apps im Tourismussektor deutlich wahrnehmbar. Im Jahr 2011 wurde für die Gemeinde Hude (Niedersachsen) ein Tourismuskonzept entwickelt, das vorsieht, den lokalen Tourismus kreativ und innovativ zu fördern - unter anderem durch den Einsatz neuer Technologien. Die aus diesem Grund initiierte Realisierung einer Tourismus-App für die Gemeinde Hude ist Thema der vorliegenden Arbeit. Durch den Einsatz von Webtechnologien und des PhoneGap-Frameworks konnte eine für Touristen ansprechende hybride App geschaffen werden, die eine leichte Wartbarkeit garantiert und eine Portierung auf nahezu jedes mobile Betriebssystem ermöglicht. Das gemeinsam mit Partnern aus der Tourismusbranche entwickelte Produkt ist auf ganzer Linie erfolgreich ausgefallen.

The inland tourism in Germany is flourishing rapidly, as well as the distribution and use of smartphones and apps. A result of both factors is an increased use of apps in the tourism sector. In 2011, a tourism concept was developed for the community Hude (Lower Saxony), which aims at creative and innovative promotion of local tourism - including the use of new technologies. The implementation of a tourism app for the community Hude, which was planned on this account, is the subject of this work. Based on the use of web technologies and the PhoneGap-Framework an attractive tourist hybrid app could be created, that ensures easy maintenance and allows a port to almost any mobile operating system. The product, that was developed in cooperation with partners from the tourism industry, turned out successfully all along the line.

Inhaltsverzeichnis

1. Einleitung	2
2. Grundlagen	5
2.1. Smartphones	5
2.2. Betriebssysteme	5
2.2.1. Android	5
2.2.2. iOS	6
2.2.3. Windows Phone	7
2.3. Mobile Applikation - App	8
2.3.1. Native App	8
2.3.2. WebApp	8
2.3.3. Hybride App	9
2.3.4. Übersichtsdarstellung der Vor- und Nachteile verschiedener App-Arten	10
2.4. Frameworks	10
2.5. Hybertext Markup Language (HTML)	10
3. Nutzeranforderungen	12
3.1. Treffen in Hude	12
3.1.1. Hude 12. April 2013	12
3.1.2. Hude 19. April 2013	12
3.1.3. Hude 8. Mai 2013	12
3.1.4. Hude 21. Juni 2013	13
3.1.5. Hude 4. Juli 2013	13
3.1.6. Hude 1. August 2013	13
3.1.7. Hude 15. August 2013	14
3.2. Personas	14
3.2.1. Persona I - Zielgruppe: Best Ager 50+	15
3.2.2. Persona II - Zielgruppe: junge Familie mit Kind	15
3.3. Anforderungen	16
3.3.1. Funktionale Anforderungen	16
3.3.2. Optionale Anforderungen	17
3.3.3. Nicht-funktionale Anforderungen	17
4. Anwendungsfälle	19
4.1. Anwendungsfalldiagramm	19
4.2. Anwendungsfallmodell	19
4.2.1. Akteure	20
4.2.2. Anwendungsfälle Anwender	20
5. Entwurf	30
5.1. Architektur	30
5.2. Datenstruktur	31
5.3. Methodenübersicht	32
5.4. Sequenzdiagramme	34

5.5. Prototypen	40
5.5.1. Prototyp I	40
5.5.2. Prototyp II	43
5.5.3. Prototyp III	45
5.5.4. Prototyp IV	46
5.5.5. Prototyp V	46
5.5.6. Prototyp VI	48
5.5.7. Prototyp VII	48
6. Implementierung	51
6.1. Frameworks	51
6.1.1. jQuery	51
6.1.2. Foundation	53
6.1.3. Phonegap	53
6.2. Installation	55
6.2.1. SDK	56
6.2.2. Build PhoneGap	56
6.3. App-Entwicklung	59
6.3.1. Design	59
6.3.2. Seiten	62
6.3.3. Funktionalitäten	73
7. Evaluation	100
7.1. Brauchbarkeit und Nutzbarkeit	100
7.1.1. Shneiderman	100
7.1.2. Test der Benutzbarkeit	103
7.2. Erfüllung der Anforderungen	105
7.3. Richtigkeit	108
8. Fazit	109
Glossar	111
Literatur	113
Anhang	117
A. Zertifikate	117
B. GitHub	119
C. Übermittlung an die App Stores	120
D. QR-Codes	122

Abbildungsverzeichnis

1.1. Anwendung von Apps in den unterschiedlichen Reisephasen. [Mei]	3
2.1. Verteilung der Marktanteile von mobilen Betriebssystemen	6
2.2. Entwicklungsschritte einer nativen und einer hybriden App [Iff12]	9
2.3. Übersicht der Vor- und Nachteile der unterschiedlichen App-Arten[Bec]	10
2.4. Grundstruktur HTML5	11
2.5. HTML-Elemente zur Definition eines Dokuments	11
4.1. Anwendungsfalldiagramm für die Touristen-App von Hude	19
4.2. Anwendungsfall: ZurückZurStartseite	20
4.3. Anwendungsfall: QR-Code-Scannen	20
4.4. Anwendungsfall: PositionAnzeigen	21
4.5. Anwendungsfall: ImpressumAnzeigen	21
4.6. Anwendungsfall: EinstellungenAnzeigen	22
4.7. Anwendungsfall: ZielgruppeEinstellen	22
4.8. Anwendungsfall: SeiteZurück	23
4.9. Anwendungsfall: InformationenÜberHudeAnzeigen	23
4.10. Anwendungsfall: UnterkünfteAnzeigen	23
4.11. Anwendungsfall: UnterkunftAnzeigen	24
4.12. Anwendungsfall: SehenswürdigkeitenÜbersicht	24
4.13. Anwendungsfall: SehenswürdigkeitAnzeigen	25
4.14. Anwendungsfall: GastronomienAnzeigen	25
4.15. Anwendungsfall: ECAutomatenAnzeigen	26
4.16. Anwendungsfall: EinkaufsmöglichkeitenAnzeigen	26
4.17. Anwendungsfall: ParkmöglichkeitenAnzeigen	27
4.18. Anwendungsfall: GesundheitseinrichtungenAnzeigen	27
4.19. Anwendungsfall: VeranstaltungenAnzeigen	28
4.20. Anwendungsfall: SpieleUndRätselAnzeigen	28
4.21. Anwendungsfall: Fotojagd	28
4.22. Anwendungsfall: Planetenabitur	29
4.23. Anwendungsfall: HudePerfekt	29
5.1. Architektur einer mit PhoneGap realisierten App[pho11]	30
5.2. Struktur der App	31
5.3. Erste Ebene des Dateisystems der App	31
5.4. Sequenzdiagramm: App-Start	34
5.5. Sequenzdiagramm: Navigation	35
5.6. Sequenzdiagramm: Download der Audio-Inhalte	35
5.7. Sequenzdiagramm: Abspielen von Audio-Inhalten	36
5.8. Sequenzdiagramm: Kartenmaterial	37
5.9. Sequenzdiagramm: QR-Code	38
5.10. Sequenzdiagramm: Fotojagd	39
5.11. Sequenzdiagramm: Planetenabitur	40
5.12. Prototyp I: Startansicht	41

5.13. Prototyp I: Offenes Panel	41
5.14. Prototyp I: Übersicht von Sehenswürdigkeiten	41
5.15. Prototyp I: Oberer Teil der Detailansicht	42
5.16. Prototyp I: Unterer Teil der Detailansicht	42
5.17. Prototyp I: QR-Code-Scanner	43
5.18. Prototyp I: Karte mit aktuellen Standort	43
5.19. Prototyp II: Startansicht mit Icons	43
5.20. Prototyp II: Übersicht Sehenswürdigkeiten	43
5.21. Prototyp II: Ohne Klappmenü	44
5.22. Prototyp II: Ohne Klappmenü II	44
5.23. Prototyp II: Klappmenü geschlossen	44
5.24. Prototyp II: Klappmenü geöffnet	44
5.25. Prototyp II: Panel	45
5.26. Prototyp II: Pop-Up Kontakt	45
5.27. Prototyp III: Inhalte mit Klappmenüs	45
5.28. Prototyp III: Anpassung an das Corporate Design	45
5.29. Prototyp IV: Listviews innerhalb des Klappmenüs	46
5.30. Prototyp IV: Anpassung von Überschrift, Footer und Übersichten	46
5.31. Prototyp V: Neue Einstellungen	47
5.32. Prototyp V: Auswertung des Planetenabiturs	47
5.33. Prototyp V: Fehlerkennzeichnung	47
5.34. Prototyp V: Alte Listviews	48
5.35. Prototyp V: Neue Listviews	48
5.36. Prototyp VII: Foundation Zurb	49
5.37. Prototyp I-VI: jQuery Mobile	49
5.38. Prototyp VII: Hude-Perfekt	49
5.39. Prototyp VII: Karte im Kontakt	49
5.40. Prototyp VII: Panel-Ebene I	50
5.41. Prototyp VII: Panel-Ebene II	50
5.42. Prototyp VII: Einstellungen Sprache	50
5.43. Prototyp VII: Einstellungen Schriftgröße	50
6.1. Geöffnete fancyBox[fan]	52
6.2. Google-Maps-Karte - eingebunden per Plugin[joh12]	53
6.3. Benötigte Pfade des PhoneGap-Frameworks zum Erzeugen von Projekten einzelner Apps	54
6.4. Übersicht über die unterstützte Geräte-Hardware in PhoneGap [Ado]	54
6.5. Erzeugung der Projekte für die SDKs	56
6.6. PhoneGap-Build config.xml für Hude	59
6.7. Icon und Startgrafik	60
6.8. Anpassung der CSS für den Header und das Panel	60
6.9. Anpassung der CSS für die Klappmenüs	62
6.10. <head>-Bereich	63
6.11. <body>-Bereich	65
6.12. Ausschnitt der Klappmenüs aus der Gesundheits-Seite	67
6.13. Resultat des Codes aus der Abbildung 6.12	67
6.14. Ausblenden von Inhalte auf bestimmten Systemen	73
6.15. JS-Funktion, die die Bilder in der Lightbox öffnet	74
6.16. Lightbox ohne Galerie-Funktion	75

6.17. JS-Funktion, die Bilder in der Lightbox öffnet	75
6.18. Lightbox mit Galerie-Funktion	75
6.19. Google Maps im HTML	76
6.20. Google-Maps-Implementierung im JS	78
6.21. Zielgruppeneinstellungen Realisierung in HTML	79
6.22. Zielgruppeneinstellungen	80
6.23. Schriftgrößen-Implementierung	81
6.24. Funktionen zum Speichern der Sprachen	81
6.25. Multilinguales HTML	82
6.26. Multilinguales JS	83
6.27. Download der Audio-Dateien mittels JS	85
6.28. Ausschnitt einer Inhaltsseite mit Audio-Player	86
6.29. Überprüfung ob eine Audio-Datei vorhanden ist	87
6.30. Funktionen der Kontrollelemente des Audio-Players	88
6.31. Navigation in der App mittels Anker-Tag	89
6.32. Hashchange-Event und Aufruf der neuen Seite	89
6.33. Start des BarcodeScanners und Verarbeitung des gescannten QR-Code	90
6.34. Ausschnitt der HTML zur Implementierung des Planetenabiturs A	91
6.35. Resultat des Planetenabiturs	92
6.36. Auswertung des Planetenabiturs	92
6.37. Ausschnitt Fotojagd „Hude“	93
6.38. JS-Implementierung zur Fotojagd	96
6.39. Abfrage in der Zukunft liegender Veranstaltungen	98
6.40. Aufbau der Seite für Veranstaltungen	98
6.41. Skript zum Einfügen der Veranstaltungen	99
7.1. Eight Golden Rules of Interface Design [Shn01]	100
7.2. Testpersonen und gefundene Fehler [Nie00]	103
7.3. SUS-Fragebogen	104
7.4. Auswertung SUS-Fragebögen	105
7.5. Umsetzung der funktionalen Anforderungen	106
7.6. Umsetzung der optionalen Anforderungen	107
7.7. Umsetzung der nicht-funktionalen Anforderungen	108
A.1. Einstellungen im App-Identifizier	117
A.2. Zertifizierungsanfrage	117
A.3. iOS Provisioning Profile	118
B.1. GitHub-Benutzerdaten	119
C.1. FTP-Verbindungsdaten	121
D.1. EC-Automaten	122
D.2. Einkaufen	122
D.3. Gastronomie	122
D.4. Gesundheit	122
D.5. Hude-Info	122
D.6. Hude-Perfekt	122
D.7. Parkmöglichkeiten	122
D.8. Fotojagd: Hude	122
D.9. Fotojagd: Klosterbezirk	122
D.10. Fotojagd: Skulpturenufer	123
D.11. Planeten-Abitur A	123

D.12. Planeten-Abitur B	123
D.13. Planeten-Abitur C	123
D.14. Artesischer Brunnen	123
D.15. Hasbruch	123
D.16. Klosterbezirk: Klosterruine	123
D.17. Klosterbezirk: Klosterschänke	123
D.18. Klosterbezirk: Museum	123
D.19. Klosterbezirk	124
D.20. Klosterbezirk: Torkapelle	124
D.21. Klosterbezirk: Wassermühle	124
D.22. Planetenlehrpfad: Erde	124
D.23. Planetenlehrpfad: Jupiter	124
D.24. Planetenlehrpfad: Mars	124
D.25. Planetenlehrpfad: Merkur	124
D.26. Planetenlehrpfad: Neptun	124
D.27. Planetenlehrpfad: Pluto	124
D.28. Planetenlehrpfad: Saturn	125
D.29. Planetenlehrpfad: Sonne	125
D.30. Planetenlehrpfad: Uranus	125
D.31. Planetenlehrpfad: Venus	125
D.32. Sehenswürdigkeiten	125
D.33. Skulpturenufer	125
D.34. Sonnenuhr	125
D.35. Megalithkultur	125
D.36. Vielstedter Bauernhaus	125
D.37. Wanderweg Huder Bach	126
D.38. Wittemoor	126
D.39. Zeitstrahl 2000	126
D.40. Ziehbrunnen	126
D.41. Spielplätze	126
D.42. Ferienhaus Aschenbeck	126
D.43. Ferienhaus Hude	126
D.44. Ferienwohnung Alte Bäckerei	126
D.45. Ferienwohnung Aschenbeck	126
D.46. Ferienwohnung Burgmannshof	127
D.47. Ferienwohnung Dat Lüttje Huus	127
D.48. Ferienwohnung Eichkatz	127
D.49. Ferienwohnung F. Hattendorf	127
D.50. Ferienwohnung Forsthaus Hasbruch	127
D.51. Ferienwohnung Haus Lichtblick	127
D.52. Ferienwohnung Hof Stolle	127
D.53. Ferienwohnung Klosterschänke	127
D.54. Ferienwohnung M. Kallisch	127
D.55. Ferienwohnung Seekermann	128
D.56. Ferienwohnung U. Kallisch	128
D.57. Gästehaus Wobig	128
D.58. Gästehaus Würdemann	128
D.59. Hotel Burgdorf	128
D.60. Hotel Garni	128
D.61. Hotel Klosterschänke	128

D.62. Jugendherberge	128
D.63. Pension Hof Heinemann	128
D.64. Pension W. Paulus	129
D.65. Unterkünfte	129
D.66. Wohnmobilstellplatz	129
D.67. Veranstaltungen	129

Sprachliche Anmerkungen

Zur Erhöhung der Verständlichkeit und der Vermeidung von Redundanz wird bei Personenbezeichnungen die maskuline Form gewählt, ohne implizit nur männliche Personen anzusprechen. Ebenso werden zugunsten der Verständlichkeit Fachbegriffe aus dem Englischen nicht ins Deutsche übersetzt, wenn die Fachbegriffe im deutschen ebenso geläufig sind. Dazu zählen beispielsweise „App“, „App Store“ oder „open source“. Bei einer Wortzusammensetzung werden die deutschen Regeln zur Wortkomposition angewendet.

1. Einleitung

Tourismus hat sich in Deutschland in der vergangenen Zeit zu einem der wichtigsten volkswirtschaftlichen Bereiche entwickelt. Die Studie „Wirtschaftsfaktor Tourismus“ [BMW] belegte die enorme Bedeutung der Tourismuswirtschaft innerhalb der deutschen Gesamtwirtschaft. Schon jetzt hat der Zweig der Tourismuswirtschaft eine höhere Bruttowertschöpfung als die Kfz-Industrie, der Maschinenbau oder die Bankwirtschaft. Zugleich machte die Studie deutlich, dass der Tourismus eine sehr arbeitsintensive Branche ist und damit deutlich zur Schaffung und zum Erhalt von Arbeitsplätzen beiträgt. Speziell Niedersachsen schaffte es, sich als beliebtestes Reiseziel Norddeutschlands zu etablieren. Die Landesregierung sieht den Tourismus in den kommenden Jahren als einen zentralen Wirtschaftsfaktor für Niedersachsen an [Bud12].

Die Gemeinde Hude (Oldb) ist eine knapp 16.000 Einwohner umfassende Kleinstadt zwischen Oldenburg (Oldb) und Bremen in Niedersachsen. Auch in Hude wird vermehrt Augenmerk auf den Bereich des Tourismus gelegt. So erhielt der Verein „Touristik-Palette-Hude e.V.“ durch die Gemeinde Hude den Auftrag, den Wirtschaftsfaktor „Tourismus“ zu analysieren und weiterzuentwickeln. Dabei kam es zu einem Austausch zwischen politischen Gremien der Gemeinde Hude und des Touristik-Palette-Hude e.V.. Einer Gäste- und Gastgeberbefragung im Jahr 2009 folgte eine Stärken- und Schwächeanalyse im Expertengespräch im Jahr 2010. Daraus resultierten Aufschlüsse über den aktuellen Stand des Tourismus in Hude und Bereiche, in denen Potentiale zur Weiterentwicklung bestehen. Die im Konzept vorgeschlagenen Maßnahmen gemäß dem Motto „Bleib dem Alten geöffnet, aber verschließ‘ Dich nicht dem Neuen“ sollen zu einer Verbesserung der Positionierung am binnenländischen Tourismusmarkt führen. Beeindruckend ist bereits die Feststellung, dass die Zahl der Übernachtungen in gewerblichen Betrieben in Hude im Jahr 2010 mit knapp 23.000 so hoch wie nie gewesen ist und 16 Prozent höher als im Jahr 2009 ausfällt. Diese Entwicklung soll weiterhin gefördert werden. Das vom Touristik-Palette-Hude e.V. vorgelegte „Tourismus-Entwicklungskonzept 2011“ gibt hierzu entsprechende Orientierungshinweise. Unter anderem sieht das Konzept vor, den Tourismus durch den Einsatz innovativer Technik weiterzuentwickeln [TPH11]. Ein wichtiges Element der innovativen Technik im Tourismusbereich sind die sogenannten mobilen Applikationen. Laut Meier [Mei] gibt es bereits 30.000 Reiseanwendungen für die Planung, Durchführung und Nachbereitung einer Reise für das iOS-Betriebssystem der Firma Apple. Mobile Applikationen beeinflussen somit alle Reisephasen, in denen Reisende Informationen suchen und anfragen.

So wird aus der Übersichtsgrafik („Anwendung von Apps in den unterschiedlichen Reisephasen. [Mei]“ auf Seite 3) ersichtlich, dass mobile Applikationen in der Vorbereitungsphase einer Reise, bei der Reisezielsuche aber auch bei der Planung und Buchung von Tickets genutzt werden. Während einer Reise können sie hilfreich sein, um beispielsweise Transporte zu organisieren und den Aufenthalt mit Aktivitäten zu füllen. Nach einer Reise kommen mobile Applikationen zum Beispiel dann zum Einsatz, wenn Fotos und Urlaubserinnerungen über soziale Netzwerke geteilt werden. Laut Meier [Mei] wird es für den Erfolg von Unternehmen auch im Bereich des Tourismus immer wichtiger werden, eine klare Strategie ihrer mobilen Dienste zu entwerfen. Entscheidend sei, bei Applikationen nicht nur den benötigten Service zu stellen, sondern unerwartete und qualitativ hochwertige Erlebnisse zu bieten.

Auch für die Gemeinde Hude kam die Idee auf, dieser Notwendigkeit mit der Entwicklung und Bereitstellung einer mobilen Tourismus-Applikation zu begegnen.

Laut Statistischem Bundesamt [TPH11, S. 7] ist die größte Gruppe der Reisenden die der sogenannten „Best Ager 50+“ (45- bis 64-Jährige) mit 34,3 Millionen, gefolgt von der Gruppe der 25- bis 44-Jährigen mit 31,2 Millionen. Beide Gruppen recherchieren ihre Informationen zu einem Reisegebiet zum Großteil im Internet. Daraus lässt sich ableiten, dass sie mit dem Umgang mit neuen Technologien vertraut sind.



Abbildung 1.1.: Anwendung von Apps in den unterschiedlichen Reisephasen. [Mei]

Zudem kann davon ausgegangen werden, dass mittlerweile ein Großteil der Personen in beiden Gruppen im Besitz eines Smartphones ist. Auch der Verein „Tourismus-Palette-Hude e.V.“ bestätigt, die Hauptzielgruppen des Huder Tourismus seien weiterhin Familien mit Kindern, die Generation 50plus sowie Gruppenreisende [TPH11].

Ausgehend von dieser Annahme und anknüpfend an das „Tourismus-Entwicklungskonzept 2011“ entwickelte sich die Fragestellung dieser Masterarbeit. Ziel ist es, eine mobile App zu entwickeln, mit der es Besuchern und Urlaubern in Hude möglich ist, jederzeit Informationen über Hude, seine Gastronomie, Sehenswürdigkeiten und Freizeitangebote abzurufen.

Zunächst werden zu diesem Zwecke in der Arbeit Grundlagen vorgestellt, die für den weiteren Verlauf der Arbeit benötigt sind. Darunter fallen die Definition eines Smartphones, eines Betriebssystems (beinhaltend die verschiedenen Arten von Betriebssystemen) und die Definition einer App. Anschließend wird mit der Analyse der Nutzeranforderungen fortgefahren. Diese wurden zum einen durch regelmäßige Gespräche mit dem Kunden, der Touristik-Palette-Hude herausgestellt, weitere Nutzeranforderungen ergaben sich aus dem „Tourismus-Entwicklungskonzept 2011 für den staatlich anerkannten Erholungsort Hude“, sowie aus der Analyse bereits bestehender Tourismus-Apps. Beschrieben werden die Anforderungen zunächst durch Personas (siehe Abschnitt 3.2 auf Seite 14). Eine Aufschlüsselung in funktionale Anforderungen, optionale Anforderungen und nicht-funktionale Anforderungen erfolgt übersichtshalber tabellarisch. Aus den herausgestellten Nutzeranforderungen werden Anwendungsfälle abgeleitet und beschrieben. Im Kapitel zur Entwurfsdarstellung wird die Implementierung geplant. So findet die Definition der Architektur und einer Datenstruktur statt. Eine Methodenübersicht stellt einen Überblick über die zu implementierenden Funktionalitäten her und Sequenzdiagramme verdeutlichen notwendige Abläufe. Durch die Beschreibung der einzelnen Prototypen ist eine prototypische evolutionäre Entwicklung der App nachvollziehbar. Im Kapitel der Implementierung werden zunächst die eingesetzten Frameworks und deren Zweck der Verwendung vorgestellt. Die Implementierung umfasst zudem die Installation der Entwicklungsumgebungen sowie die

1. Einleitung

eigentliche App-Entwicklung. Anschließend erfolgt die Evaluierung durch die Analyse der Benutzbarkeit, Erfüllung der Anforderungen sowie der Richtigkeit der App (letzteres durch die Touristik-Palette-Hude). In einem Fazit wird ein Rückblick auf die Arbeit gegeben, ein Ausblick auf eine mögliche Weiterentwicklung eröffnet und ein persönliches Resümee gezogen.

2. Grundlagen

Im Kapitel Grundlagen werden zunächst die Grundlagen vorgestellt, die für den weiteren Verlauf der Arbeit benötigt werden. Darunter fallen die Definition eines Smartphones, die Definition eines Betriebssystems (beinhaltend die Unterscheidung verschiedener Arten von Betriebssystemen) sowie die Definition einer App und verschiedener App-Arten.

2.1. Smartphones

Als Smartphone bezeichnet man die Klasse der Mobiltelefone, im Gegensatz zu den herkömmlichen Mobiltelefonen wiesen diese einen deutlich höheren Funktionsumfang auf. Dieser beschränkt sich nicht auf die Telefonie und SMS¹, sondern bietet weitere Funktionen wie z.B.² den Internetzugang mit E-Mail und Surfen, Terminkalender oder die Aufnahme und Wiedergabe von Audio- und Video-Aufzeichnungen mittels Mikrofon oder Kamera. Diese Funktionalität lässt sich durch sogenannte App³s (siehe Kapitel 2.3 Mobile Applikation - App auf Seite 8) erweitern [Gab].

2.2. Betriebssysteme

Um ein Smartphone (siehe Kapitel 2.1 Smartphones auf Seite 5) und die verbauten Komponenten überhaupt nutzen zu können, sowie weitere Apps auf dem Gerät installieren zu können, sind mobile Betriebssysteme notwendig. Diese unterscheiden sich in ihrem Funktionsumfang und ihrer Geschwindigkeit von herkömmlichen Betriebssystemen. Mobile Betriebssysteme können Apps nicht parallel ausführen. So ist es beispielsweise nicht möglich, zwei Fenster gleichzeitig geöffnet zu halten. Zudem werden Apps, um Speicherkapazität zu sparen und wieder freizugeben, nach einer gewissen Zeit in Inaktivität beendet. Auf dem Weltmarkt existieren verschiedenste mobile Betriebssysteme für Smartphones. Die sechs meist verbreiteten sind Android, bada, BlackBerry, iOS, Windows Mobile und Windows Phone. Die folgenden Abschnitte stellen die drei am häufigsten genutzten mobilen Betriebssysteme vor. Dazu zählen Googles „Android“ mit einem Marktanteil von 69,7 % im vierten Quartal 2012, Apples „iOS“, mit einem Marktanteil von 20,9 % im vierten Quartal 2012 und Microsofts „Windows Phone“ mit einem Marktanteil von 3,0 % im vierten Quartal 2012 [van13].

2.2.1. Android

Im Jahr 2005 erwarb Google Inc.⁴ die Android Inc. für 50 Millionen US-Dollar. Im November 2007 gründete Google die „Open Handset Alliance“, in der 34 (Gründungs-) Unternehmen aus den Bereichen der Handyhersteller, Mobilfunknetzbetreiber, Halbleiterproduzenten, Softwarefirmen und Marketingspezialisten offene Standards für Mobilgeräte schaffen. Das erste Android Phone wurde im Herbst 2008 veröffentlicht. In Deutschland gab es das erste Android Smartphone (siehe Kapitel 2.1 Smartphones auf Seite 5) mit der Android Version 1.1 auf dem T-Mobile G1. Das T-Mobile G1 wurde noch mit einer ausklappbaren

¹Short Message Service (SMS) sind kurze Text-Nachrichten von maximal 160 Zeichen, die über das Mobilfunknetz übertragen werden.

²zum Beispiel

³Applikationen (Apps) bezeichnen Anwendungen, die auf einem Betriebssystem lauffähig sind. Erst durch die große Verbreitung von mobilen Endgeräten (Smartphones & Tablets) setzte sich für diese der Begriff „App“ durch.

⁴Google Inc. ist eine US-amerikanische Unternehmung. Bekannt ist Google Inc. besonders durch die weltgrößte gleichnamige Suchmaschine Google. Neben dem Betrieb der Suchmaschine Google hat Google Inc. seine Geschäftsfelder im Internet weiter ausgedehnt.

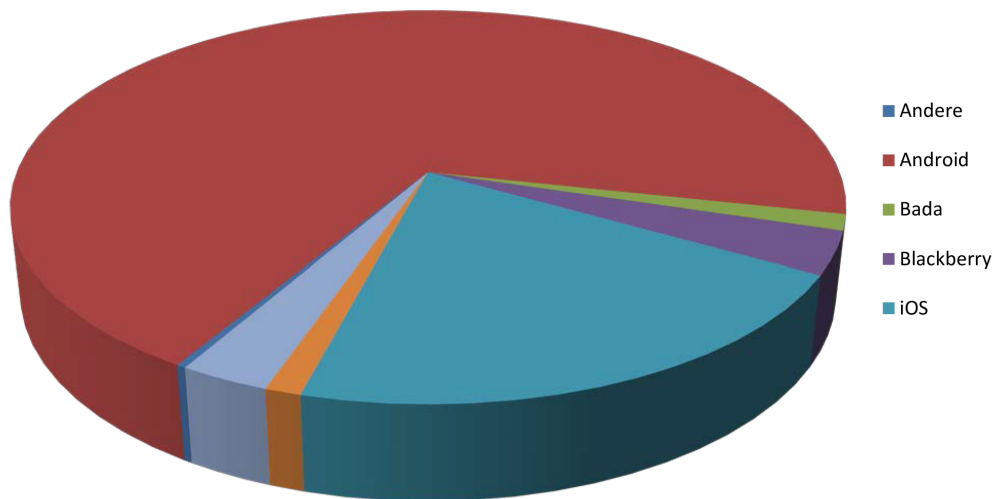


Abbildung 2.1.: Verteilung der Marktanteile von mobilen Betriebssystemen

Tastatur bedient. Zu diesem Zeitpunkt gab es rund 2000 Apps (siehe Kapitel 2.3 Mobile Applikation - App auf Seite 8) im „Android Market“ (heute „Google Play Store“), die das G1 erweitern konnten. In den kommenden Jahren wurde das Android-Betriebssystem weiterentwickelt. Daraus resultieren die Versionen „Cupcake“ (1.5), „Donut“ (1.6), „Eclair“ (2.0/2.1), „Froyo“ (2.2.x), „Gingerbread“ (2.3.x), „Honeycomb“ (3.x.x), „Ice Cream Sandwich“ (4.0.x) bis zur aktuellen Version „Jelly Bean“ (4.1.x/4.2.x). Google plant im Mai 2013 die Veröffentlichung der Version „Key Lime Prime“ (5.0). Die Version „Cup Cake“ benötigte auch keine Tastatur mehr, sondern wurde allein über den Bildschirm gesteuert. Nicht nur das Betriebssystem hat sich weiter entwickelt, auch die Anzahl der Apps ist ansteigend. Ausgehend von 2000 Apps sind es mittlerweile 900.000 - jede fünfte davon ist ein Spiel. Da Android ein Open-Source-System auf JAVA-Basis ist, besteht die Möglichkeit, eigene Android-Versionen zu entwickeln. Daraus entstanden, neben den Versionen der einzelnen Smartphone-Hersteller, auch verschiedene Custom Roms, die sich zum Teil großer Beliebtheit erfreuen. So ist es kaum verwunderlich, dass viele verschiedene Geräte unterschiedlicher Hersteller mit dem Android-Betriebssystem ausgestattet sind [con13].

2.2.2. iOS

Im Jahr 2007 wurde das erste Betriebssystem für das „iPhone 2G“ und den „iPod touch“ unter dem Namen „iPhone OS“ eingeführt. Dieser Name blieb bis zum Juni 2010 bestehen, bis eine Umbenennung auf „iOS“ erfolgte. Im Gegensatz zu Android besaß das von Apple entwickelte „iPhone OS“ weniger Features. Der Nutzer war zunächst von den 3G-Standards abgeschnitten. So fehlten die Funktionen: „Copy&Paste“, „MMS“⁵, „Push-Mail von Exchange-Servern“, „Tethering“⁶ und der „Zugriff auf das Dateisystem“. Allerdings wurden die Geräte von Apple mit einem kapazitiven Display⁷ ausgestattet und die physischen Tasten auf ein Minimum reduziert. Das „iPhone 2G“, sowie der „iPod touch“ besaßen nur noch 5 Tasten. Dazu zählten die „Home-Taste“ unter dem Display, zwei Tasten zur Lautstärkeregelung an der Außenseite, sowie eine Taste zum Stummschalten und eine Taste zum Aktivieren des Standby-Modus [Boh11]. Zum gleichen Zeitpunkt wollte Steve Jobs, Gründer und langjähriger Geschäftsführer von Apple, auf dem iPhone die Web- Apps (siehe Kapitel 2.3.2 WebApp auf Seite 8) als Standard etablieren. Bereits ein halbes Jahr

⁵Multi Message Service (MMS) ist die Weiterentwicklung des „Short Message Service“ (SMS). Im Gegensatz zu SMS können via MMS neben Textnachrichten auch Medieninhalte (Bilder, Sprachnachrichten, etc.), über das Mobilfunknetz versendet werden.

⁶Als Tethering bezeichnet man die Möglichkeit, die Datenverbindung des Smartphones über andere Geräte mitnutzen zu können.

⁷Ein kapazitives Display reagiert auf Berührung. Dabei ist für die Interaktion nicht der ausgeübte Druck entscheidend, sondern die Veränderungen der Leitfähigkeit des Displays durch die Berührung.

später wurde jedoch ein SDK⁸ veröffentlicht, um native Apps (siehe Kapitel 2.3.1 Native App auf Seite 8) zu entwickeln [FI12]. Die Entwicklung der Apps erfolgt in der Programmiersprache „Objective-C“. Zeitgleich gab es im Juli 2008 ein Update für die Apple-Geräte, mit dem der „App Store“ integriert wurde. Allerdings wurden von Apple Apps, die im „Android Market“ zugelassen wurden, zurückgewiesen, da sie nicht den Apple-Guidelines [App] entsprachen. Im Juni 2009 veröffentlichte Apple dann das „iPhone OS3“ mit dem die fehlenden Features beinahe alle eingearbeitet waren. Ein Jahr später, im Juni 2010, folgte die Version „iPhone OS4“ mit dem auch das fehlende Feature des Tethering ermöglicht wurde [Gil]. Mit der Veröffentlichung einer neuen Systemversion benannte Apple sein „iPhone OS“ in „iOS“ um. Die erste „iOS“-Version wurde mit der fortlaufenden Nummer des „iPhones OS“ herausgebracht. Im Oktober 2011 erschien somit „iOS 5“ in Verbindung mit dem „iPhone 4S“ und dem großen Feature „Siri“, Apples Sprachsteuerung. Mit der Version „iOS 6“, die im September 2012 veröffentlicht wurde, wollte sich Apple vollständig von Google lösen. Aus diesem Grund wurden Apps wie zum Beispiel „Google Maps“⁹ von den Geräten entfernt und durch Eigenentwicklungen ersetzt. Apple machte dies inzwischen jedoch wieder rückgängig, so dass alle Google-Dienste durch Apps wieder nutzbar wurden [Boh11]. Apples größter Vorteil ist jedoch, dass das Unternehmen wenige Modelle produziert und somit bei jedem „iOS“-Update nahezu jedes Gerät damit versorgen kann - eine Anpassung durch die verschiedenen Hersteller, wie bei einer neuen Version von Android, ist nicht nötig.

2.2.3. Windows Phone

Im November 2010 veröffentlichte Microsoft sein neues Betriebssystem für Mobilgeräte mit dem Namen „Windows Phone 7“. „Windows Phone 7“ ist der Nachfolger des „Windows CE“, das vorzugsweise auf Handheld-Geräten, Vorgängern der heutigen Tabs, installiert war. „Windows Phone 7“ fehlten jedoch Features wie „Hintergrundprozesse“, „Bluetooth-Dateiübertragung“, „Copy&Paste“ oder die „Synchronisation mit Outlook“. Durch ein Update („NoDo“-Update), welches 6 Monate später veröffentlicht wurde, wurden einige dieser und andere fehlende Features integriert. Microsoft gab an die Hersteller von Handys Hardwarevoraussetzungen heraus, um allen Anwendern eine gleiche Benutzbarkeit zu gewährleisten [Gil11]. Microsoft entwickelte sein „Windows Phone“ für kapazitive Displays, so dass das System über die „Metro“-Oberfläche gesteuert wird [Sac11]. Im Februar 2011 schlossen sich Microsoft und Nokia zusammen, um den Verlust von Marktanteilen der Nokia-Smartphones zu stoppen [Ber11]. Im September 2011 [Wik13b] veröffentlichte Microsoft das „Windows Phone 7.5“. Laut Microsoft sind in diesem Update viele neue und verbesserte Features integriert worden, die von den Anwendern gewünscht wurden [Mica]. Dazu zählen z.B. „eigene Klingeltöne“, „Multitasking“, „Unterstützung versteckter WLAN¹⁰s“ und die Facebook-Anbindung, etc.¹¹ [Wik13b]. Im Oktober 2011 präsentierte Nokia mit dem „Lumia 800“ das erste Smartphone mit einem Windows-Phone-Betriebssystem [n-t11]. 15 Monate nach dem Update auf „Windows Phone 7.5“ veröffentlichte Microsoft im Januar 2013 das Update zur „Windows Phone 7.8“-Version [Wik13b]. Trotz der Vorgabe von Mindest-Hardwareanforderungen war dieses Update nicht für alle Smartphones mit dem „Windows Phone“-Betriebssystem verfügbar, denn Microsoft setzte auf neue Mindestanforderungen [Micb]. Im September 2012 präsentierte Microsoft die derzeit aktuellste Version seines Betriebssystems, das „Windows Phone 8“ [Wik13c]. Mit dieser Version sollte die große Lücke zwischen den Windows Phone-Geräten und den vielen Android-, sowie iOS-Geräten geschlossen werden [Her13]. Trotz der Kooperation zwischen Microsoft und Nokia und der damit verbundenen höheren Verbreitung der Windows Phone-Geräte, gibt es nur um die 150.000 Apps [dpa13] im „Windows Phone Store“. Die Apps, die im Windows Phone Store zu finden sind, werden im „Windows Phone SDK“ entwickelt. In

⁸Software Development Kit (SDK) ist eine Entwicklungsumgebung für eine Anwendung. Ein SDK kann sich aus verschiedenen Komponenten zusammensetzen. Beispielsweise kann dies ein Compiler oder der Editor (IDE) sein.

⁹Kartendienst von Google

¹⁰Wireless Local Area Network (WLAN) bezeichnet die Verbindung zu einem Netzwerk per Funk.

¹¹et cetera

2. Grundlagen

diesem ist eine eigenständige Version von „Visual Studio 2012“ [Micc] integriert. Die Entwicklung erfolgt dabei in den Sprachen C# oder VB.NET [Wik13b].

2.3. Mobile Applikation - App

Eine mobile Applikation (App) ist ein Anwendungsprogramm auf dem Smartphone bzw.¹² Mobilgerät (siehe Kapitel 2.1 Smartphones auf Seite 5), das auf jenes Gerät zusätzlich heruntergeladen und auf ihm ausgeführt werden kann [Bib]. Dabei unterscheidet man zwei Arten von Apps. Zum einem existieren „native Apps“ (siehe Kapitel 2.3.1 Native App auf Seite 8) und zum anderen die „Web- Apps“ (siehe Kapitel 2.3.2 WebApp auf Seite 8). Apps sind jedoch keine Programme im herkömmlichen Sinn, wie beispielsweise bei PC¹³s, sondern werden mehr als Dienste oder Services verstanden, die dem Nutzer den Alltag erleichtern sollen bzw. ihn bei Alltagsaufgaben unterstützen sollen.

2.3.1. Native App

Eine native App ist in der Programmiersprache (JAVA, Objective-C, C/C++ etc.) des jeweiligen Betriebssystems (Android, iOS, etc.) geschrieben und kompiliert. So ist ein Zugriff auf die gesamte Hardware und Schnittstellen des mobilen Geräts (siehe Kapitel 2.1 Smartphones auf Seite 5) möglich. Aus diesem Grund ist die native App nicht durch das Internet an ihre Funktionalität gebunden. Im Gegensatz zur Web-App (siehe Kapitel 2.3.2 WebApp auf Seite 8), können Offline-Funktionalitäten auch ohne den Zugriff auf das Internet genutzt werden oder im Hintergrund ablaufen - wie z.B. ein Radio, während im Internet gesurft wird [FI12, S. 23,24]. Zur Verbreitung, ob kostenlos oder kostenpflichtig, werden die vom Betriebssystemhersteller bereitgestellten Märkte genutzt. Dazu zählen z.B. der „Google Play Store“ von Android oder der „App Store“ von Apples iOS.

2.3.2. WebApp

Die Bezeichnung „Web-App“ setzt sich aus den Wörtern „Web“ (Englisch: Netz) und „App“ zusammen und umfasst somit Webseiten, die für den Aufruf durch ein mobiles Gerät (siehe Kapitel 2.1 Smartphones auf Seite 5) optimiert sind. Im Gegensatz zu normalen Websites erwarten den Benutzer - wie in nativen Apps - Symboleisten, Buttons, Dialoge und Eingabefelder. Da Web-Apps über den Browser aufgerufen werden, müssen diese nicht auf dem Gerät installiert werden und keine Updates durch den Nutzer durchgeführt werden. Ebenso sind sie, durch den Aufruf im Browser, plattformunabhängig und können nahezu mit jedem mobilen Gerät genutzt werden, das einen Browser und Internetzugang hat [FI12, S. 22]. Da Web-Apps über den Browser aufgerufen werden, ist ihre Funktionalität durch die Webstandards (HTML¹⁴, CSS¹⁵, JS¹⁶, PHP¹⁷ und viele andere mehr) definiert und begrenzt. Aus diesem Grund sind Web-Apps in ihrer Funktionalität beschränkt. Durch den Aufruf über den Browser und die client-seitige Verarbeitung der Eingaben, die nur durch JS realisierbar ist, ist aus Sicherheitsgründen nur ein begrenzter Zugriff auf Hardwarefunktionen möglich. Der Zugriff beschränkt sich, zum aktuellen Zeitpunkt, auf die Positionsbestimmung.

¹²beziehungsweise

¹³Personal Computer

¹⁴Eine detaillierte Erklärung des Begriffs Hypertext Markup Language (HTML) wird in Abschnitt 2.5 auf Seite 10 gegeben.

¹⁵Cascading Style Sheets (CSS) ermöglichen die Anpassung und Definition der Darstellung von HTML-Elementen.

¹⁶JavaScript (JS) ist eine Skriptsprache, die zur Interaktion mit dem Anwender und zur Veränderung von CSS- und HTML-Elementen entwickelt wurde. Die Ausführung eines JavaScripts erfolgt auf Seite des Anwenders.

¹⁷Hypertext Preprocessor (PHP) ist eine skriptbasierte, serverseitige Programmiersprache.

2.3.3. Hybride App

Als hybride App bezeichnet man die Symbiose der zwei App-Typen der nativen App (siehe Kapitel 2.3.1 Native App auf Seite 8) und der Web-App (siehe Kapitel 2.3.2 WebApp auf Seite 8). Eine hybride App ist mit den Webstandards HTML, JS und CSS entwickelt und somit eine vollwertige Web-App. Diese wird in einen nativen Container des jeweiligen Smartphones (siehe Kapitel 2.1 Smartphones auf Seite 5) eingebettet. Dieser native Container nutzt das „WebKit“ (einen HTML-Rendering-Engine), um die Daten im Vollbildmodus einer „Web View“, keinem Browser, darzustellen. Durch das WebKit entstehen, im Vergleich zur nativen App, Performanz-Einbußen. Diese Einbußen sind jedoch zu vernachlässigen, da diese hauptsächlich bei Entwicklungen von Spielen zum Tragen kommen, aber normale Geschäftsprozesse sich hervorragend abbilden lassen. Der große Vorteil im Vergleich zu einer Web-App (siehe Kapitel 2.3.2 WebApp auf Seite 8) liegt darin, dass über ein JS API¹⁸, das durch ein Framework (siehe Kapitel 2.4 Frameworks auf Seite 10) bereitgestellt wird, ein Zugriff auf das API des jeweiligen mobilen Betriebssystems ermöglicht wird. Per JS besteht dann die Möglichkeit z.B. auf die Kamera oder das Telefonbuch zuzugreifen. Für jedes mobile Betriebssystem kann, mittels des Frameworks, auf Basis der Web-App eine eigene native App generiert werden. Diese kann, wie eine eigens entwickelte native App (siehe Kapitel 2.3.1 Native App auf Seite 8), im betriebssystemeigenen Markt zur Verbreitung angeboten werden.

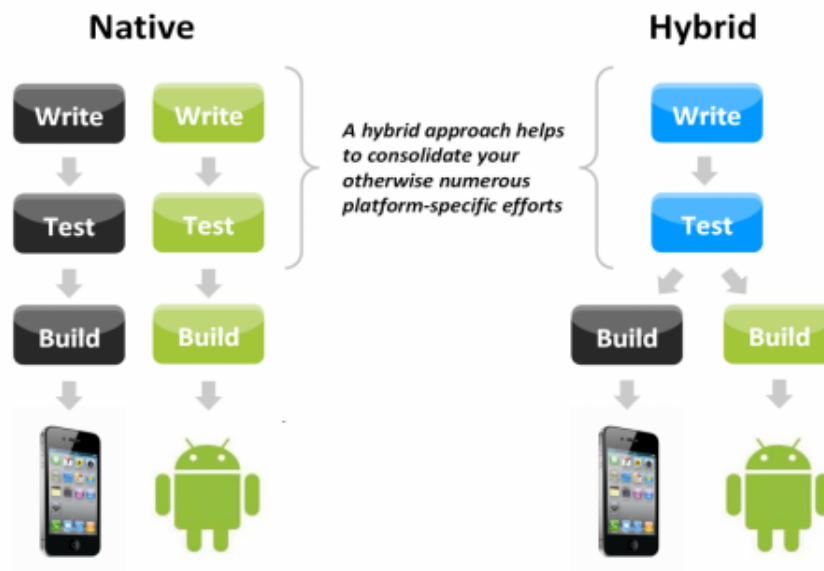


Abbildung 2.2.: Entwicklungsschritte einer nativen und einer hybriden App [Iff12]

¹⁸Application programming interface (API) bezeichnet eine Schnittstelle zwischen zwei unterschiedlichen Systemen oder Programmen innerhalb des Source Codes, die es ermöglicht, eine Verbindung untereinander herstellen zu können.

2.3.4. Übersichtsdarstellung der Vor- und Nachteile verschiedener App-Arten

	Native App	Web-App	Hybride App
Vorteile	<ul style="list-style-type: none"> - Offline nutzbar - Performance - Hardwarefunktionen - Vertrieb über Betriebssystem Märkte 	<ul style="list-style-type: none"> - Plattformübergreifend - Aufwand - Keine Prüfung durch Marktplatz-Betreiber - Keine Provision an Marktplatz-Betreiber 	<ul style="list-style-type: none"> - Plattformübergreifend - Offline nutzbar - Zugriff auf Hardware - Vertrieb über Betriebssystem-Märkte
Neutral			- Performance
Nachteile	<ul style="list-style-type: none"> - nicht Plattformübergreifend - hoher Entwicklungs- und Wartungsaufwand 	<ul style="list-style-type: none"> - Inhalte müssen online zunächst geladen werden - geringe Performance - Zugriff auf Hardware nicht gegeben - Vertrieb nicht über Betriebssystem-Märkte möglich 	

Abbildung 2.3.: Übersicht der Vor- und Nachteile der unterschiedlichen App-Arten[Bec]

2.4. Frameworks

Als Framework bezeichnet man eine Sammlung von Klassen, die zu einem bestimmten Anwendungsbereich oder Problem eine gewisse Anzahl an Funktionen bereitstellt, um die Aufgabe oder das Problem zu lösen. Das Framework und dessen Klassen sind alleine nicht lauffähig, sondern müssen vom Entwickler in eine ausführbare Umgebung eingebettet werden. Dabei kann der Entwickler die Klassen einbinden, anpassen oder auch ihre Funktionalität erweitern. Ziel eines Frameworks ist die Wiederverwendung von Source Codes, Architekturen, Entwurfsmustern und Verhaltensschemata. Die im Rahmen der entwickelten App eingesetzten Frameworks werden im Kapitel Implementierung auf Seite 51 genauer erläutert.

2.5. Hypertext Markup Language (HTML)

HTML ist eine beschreibende Definitionssprache, die innerhalb ihrer Beschreibung „Hyperlinks“ (Links) enthalten kann um mit anderen Dokumenten verknüpft zu werden. Dies ist seit der Entwicklung von HTML das herausstechende Merkmal dieser Sprache. Innerhalb eines HTML-Dokuments werden Definitionen an Inhalten vorgenommen. Durch umschließende „Tags“ werden einem Browser, oder - im Fall dieser App - der WebView, mitgeteilt wie die Inhalte interpretiert werden müssen. Um die Interpretation zu vervollständigen, können den HTML-Tags Attribute gegeben werden, die die Interpretation des Elements beeinflussen. Die am häufigsten verwendeten und für die meisten Elemente nutzbaren Attribute sind „id“, „class“, „href“ und „src“. Dabei ist bereits bei der Definition des HTML-Dokuments auf die HTML-Version zu

achten. Die App nutzt den neuesten Standard, HTML5. HTML-Dateien werden mit der Dateieindung .html oder .htm abgespeichert.

```

1 <!Doctype html>
  <html>
3   <head>
      <title>Titel des Dokuments</title>
5     <script src="javascript.js"></script>
      <link rel="stylesheet" type="text/css" href="stylesheet.css">
7   </head>
  <body>
9     <h1>Überschrift</h1>
      <p>Ein einfacher Absatz</p>
11    <a href="seite2.html">Link</a>
      <p>
13      <h1>Überschrift</h1>
      Ein einfacher <a href="seite2.html">Link</a> in einem Absatz
15    </p>
  </body>
17 </html>

```

Abbildung 2.4.: Grundstruktur HTML5

In Zeile 1 wird dem Interpreter mitgeteilt, dass es sich um ein HTML5-Dokument handelt. Zeile 2 und 13 definieren, durch das Umschließen der Elemente in Zeile 3-12, dass es sich um eine HTML-Seite handelt. Innerhalb der Zeilen 3 und 7 werden Definitionen vorgenommen, die zunächst nicht mit den Inhalten dargestellt werden. Ebenso werden in diesem Bereich Verknüpfungen zu JavaScript- (Zeile 5) oder CSS-Dateien (Zeile 6) vorgenommen. Innerhalb der body-Tags (Zeile 8 und 16) befinden sich die Inhalte, die dem Anwender angezeigt werden. Dazu zählt die Überschrift in Zeile 9, die von den Tags h1-h6 vordefiniert ist. Ein Absatz wird in Zeile 10 definiert, Zeile 11 stellt einen Link zu einem anderen Dokument her. Diese Inhalte können auch verschachtelt dargestellt werden, so wie es die Zeilen 12-15 anzeigen. In einem Absatz (Zeile 12-15) können sich eine Überschrift (Zeile 13) und ein Text befinden, der einen zu einem anderen Dokument führenden Link enthält. Weitere mögliche und vornehmlich genutzte Elemente zur Definition von HTML-Dokumenten zeigt die Grafik 2.5 auf Seite 11.

 	Zeilenumbruch
<div></div>	DIV-Container
	Bild-Datei
	Definition eines Bereichs
	Ungeordnete Liste

Abbildung 2.5.: HTML-Elemente zur Definition eines Dokuments

3. Nutzeranforderungen

Im Kapitel „Nutzeranforderungen“ sollen die Anforderungen an das System herausgestellt werden, die durch potentielle Anwender gewünscht werden könnten bzw. von den Verantwortlichen gefordert werden. Aus diesem Grund werden die Treffen mit den Verantwortlichen der „Touristik-Palette-Hude e.V.“ beschrieben und die daraus resultierenden Ergebnisse präsentiert. Außerdem werden, um das Nutzungsumfeld besser zu verstehen und darzustellen, Personas kreiert.

3.1. Treffen in Hude

Die folgenden Abschnitte bilden die Verläufe der in Hude stattgefundenen Treffen ab.

3.1.1. Hude 12. April 2013

Im Rahmen eines ersten Treffens mit den verantwortlichen Ansprechpartnerinnen Dipl.-Ing. Martina Weisensee und Dipl.-Betw. Kerstin Sonka in Hude stand zunächst die persönliche Vorstellung im Mittelpunkt. Um einen Gesamteindruck der Potenziale von touristischen Apps zu vermitteln, wurden Beispiele aus anderen Städten präsentiert sowie erste selbstentworfene Screenshots einer beispielhaften Oberflächengestaltung gezeigt. Die Anordnung der Icons war hierbei aus Gründen der Übersichtlichkeit an der Gliederung der Homepage der Gemeinde Hude orientiert. Im Mittelpunkt des Treffens stand im weiteren Verlauf die Diskussion erster Ideen zu möglichen Inhalten einer Tourismus-App für Hude. So ist es denkbar, Geocaching-Punkte ebenso wie einen QR-Scanner als Komponenten einzubinden. Neben einer Vermittlung der Inhalte über Texte ist weiterhin eine Text-to-Speech-Wiedergabe denkbar, außerdem sollten textergänzende Videos und Bilder verwendet werden. Berücksichtigt werden sollte für das Design der App das Corporate Design der Gemeinde Hude. Der Kontakt zur Grafikerin und zum Webmaster der Homepage wird von den verantwortlichen Kontaktpersonen hergestellt. Wünschenswert ist des Weiteren ein zielgruppenorientierter Filter, der die Anwendung für bestimmte Zielgruppen - wie etwa Familien mit Kind oder Best Ager - erleichtert. Grundlegend für die weitere Gestaltung der App ist zudem die Information über eine schlechte Datenverbindung in Hude, die die Entscheidung auf eine native App anstelle einer Web-App fallen lässt.

3.1.2. Hude 19. April 2013

Auf Basis der festgehaltenen Wünsche aus dem ersten Austauschtreffen wurde ein erster klickbarer Prototyp einer App entwickelt, den Verantwortlichen vorgestellt und in gemeinsamer Diskussion weiterentwickelt. Inhaltlich ergänzt werden soll eine Übersicht über Parkmöglichkeiten. Die Idee vom 12. April, betreffend die Einbindung von Geocaching-Punkten, wurde fixiert und mit dem Titel „Schnitzeljagd“ in die geplanten Menüpunkte aufgenommen. Auch die Idee der Text-to-Speech-Komponenten wird als fester Bestandteil beschlossen. Betreffend das Design soll Übersichtlichkeit durch Klappenmenüs erreicht werden. Unter einer Grafik zuzüglich eines Einleitungstextes sollen in der App Klappenmenüs mit weiteren Informationen zur Verfügung gestellt werden. Das Hauptmenü mit der Iconübersicht soll drei- bis vier-spaltig aufgebaut sein, um ebenfalls Übersichtlichkeit zu garantieren.

3.1.3. Hude 8. Mai 2013

Entsprechend den Wünschen der verantwortlichen Ansprechpartnerinnen wurde der erste Prototyp weiterentwickelt und im Rahmen eines erneuten Treffens vorgestellt und diskutiert. Beschlossen wurde in

diesem Treffen eine Zentrierung der Icons im Footer. Eine weitere Idee bestand in der Aufnahme des jeweilig aktuellen „Tipp des Tages“ in das freie Feld im Homescreen. Weiterhin soll in den Einstellungen die Schriftgröße definiert und das Design randlos, ohne einen weißen Rahmen für den Inhalt, umgesetzt werden. Die bisher global eingesetzte Überschrift „Hude“ soll durch die jeweiligen Bereichsüberschriften ersetzt werden. Der Bereich der „Sehenswürdigkeiten“ soll in der Form überarbeitet werden, dass die Inhalte „Haus Kimmen“ und „Goldart“ entfallen dafür aber die Inhalte „Klosterbezirk“, „Hasbruch“ und „Wittenmoor“ ergänzt werden. Der bisherige Punkt der „Klosterruine“ entfällt ebenfalls und wird dem Abschnitt „Klosterbezirk“ zugeordnet. Weiterhin sollen sich unter „Klosterbezirk“ die Unterpunkte „Wassermühle“, „Schenke und Remise“, „Torkapelle“ sowie „Museum“ wiederfinden. Beschlossen wird außerdem die Einbindung von Karten, ungeklärt bleibt im Treffen ob es sich hierbei um Open-Streets-Maps oder Fotos von Karten handeln wird. Die beim vergangenen Treffen beschlossene Schnitzeljagd wird ebenfalls weiterentwickelt. So soll per GPS-Rückmeldung eine Bestätigung ermöglicht werden, bestimmte Sehenswürdigkeiten besucht zu haben. Außerdem soll, analog zum Rätsel für den „Planetenlehrpfad“, ein Rätsel für den Besuch Huder Sehenswürdigkeiten eingebunden werden.

3.1.4. Hude 21. Juni 2013

In diesem Treffen wurden die aktuellen Fortschritte und Weiterentwicklungen der App den verantwortlichen Ansprechpartnerinnen erneut vorgestellt. Fortschritte bestanden unter anderem in einer Anpassung der Bilder in den Listviews. Zudem wurden die Sehenswürdigkeiten „Kulturpfad“ und „Klosterbezirk“ in ihrer Reihenfolge vertauscht. Dem Abschnitt zum Kulturpfad wurde weiterhin ein Einleitungstext hinzugefügt. Den Darstellungen der Planeten im Planetenlehrpfad wurden tatsächliche Bilder der jeweiligen Planeten aus dem Archiv der NASA zugeordnet. Diskutiert wurde die Option, die in Hude verteilten „Zeitstrahl-Kacheln“ zu fotografieren und in die App aufzunehmen. Auch der Abschnitt zu den Einkaufsmöglichkeiten soll weiter angepasst werden. Ein letzter Diskussionspunkt bestand darin, das Ergebnis des Planetenabiturs um ein Ergebnis-Ranking als Einschätzungsmöglichkeit des eigenen Abschneidens zu ergänzen.

3.1.5. Hude 4. Juli 2013

Im Rahmen des Treffens am 4. Juli wurde erneut über die Integration der Zeitstrahl-Kacheln in die App diskutiert. Aufgrund der Tatsache, dass sie nicht als bedeutende Tourismus-Attraktion eingeschätzt wurden, wurde die Idee gemeinsam kritisch bewertet, obwohl an diesem Tag bereits die Kacheln fotografisch festgehalten wurden. Soll eine ausführliche Version des Zeitstrahls dargestellt werden oder nur die Informationen aus der Broschüre dargestellt werden? Eine abschließende Entscheidung wurde auf das Treffen am 1. August vertagt. Überlegungen wurden ebenfalls dahingehend angestellt, welche Parkplätze in die App aufgenommen werden sollen und in welcher Form die Lokalisation der aufgeführten Parkplätze erfolgen soll. Vorstellbar wäre eine Beschreibung anhand der Nennung von Straßenkreuzungen bzw. Straßen. Beschlossen wurde im Treffen weiterhin, dass in der Darstellung des Planetenabiturs die Listview so gestaltet sein soll, dass alle Bilder einheitlich sind und die Listenzeile ohne weißen Rand komplett ausfüllen. Weiterhin sollte bereits auf dem Bild ausgewiesen werden, ob es sich um die Version A, B oder C handelt. Überlegt wurde auch, ob der „Hude-Padd“ in die Bilder-Schnitzeljagd integriert werden sollte. Eine weitere Idee bestand darin, verschiedene Schwierigkeitsstufen und Einstellungen für die Bilder-Schnitzeljagd zu erlauben. So könnte beispielsweise ein bestimmter Radius als Gebietsbegrenzung eingestellt werden oder nur Objekte in eigener Laufrichtung angezeigt werden. Außerdem wäre eine Hilfsfunktion wünschenswert, wenn Objekte nicht gefunden werden. Auch bei der Bilder-Schnitzeljagd bestehen Überlegungen, das eigene Ergebnis in einem Ranking darzustellen, ähnlich dem des Planetenabiturs.

3.1.6. Hude 1. August 2013

Im Rahmen des Treffens am 1. August wurden zunächst die Neuigkeiten der App-Entwicklung vorgestellt. Die Foto-Schnitzeljagd wurde mittlerweile implementiert, zunächst für den Klosterbezirk und „Hude ge-

3. Nutzeranforderungen

samt“. Implementiert wurde auch die Funktion sich die eigene Position auf einer Karte anzeigen zu lassen. Neu ist weiterhin die Fertigstellung der Audiodarstellung. Hier stehen nun neben der Download- und Abspielfunktion die Buttons „Play“ und „Pause“ zur Verfügung. Neben den Neuentwicklungen wurden weitere Anpassungen diskutiert. Die Darstellungen des Planetenabiturs, die aufzunehmenden Einkaufsmöglichkeiten und EC-Automaten, ebenso wie Stationen im Bereich „Gesundheit“ und die eigene Positionsanzeige wurden thematisiert und gemeinsam eine Einigung gefunden. Beschlossen wurde weiterhin, dass in die Foto-Schnitzeljagd Fotos des Partnerschaftsschildes am Rathaus und der Moorheiligen aufgenommen werden. Das Skulpturenufer soll die bisher vorhandenen Bezirke „Klosterbezirk“ und „Hude gesamt“ ergänzen. Außerdem wurden zwei inhaltliche Korrekturen gewünscht („Bio Wingang“ wird zu „Bio Weingang“, „Traute“ entfällt). Eine entscheidende Änderung wurde zudem beschlossen um dem begrenzten Datenvolumen pro Nutzer zu begegnen: Bilder, die bislang immer von Google bei Abruf heruntergeladen wurden, sollen zukünftig fest implementiert sein. Als neuer inhaltlicher Aspekt sollen zudem Spielplätze in den frei gebliebenen Platz auf dem Homescreen integriert werden. Auch hier gilt: Ist die Datenverbindung gut, werden die Bilder per Live-Google-Karte heruntergeladen. Ein vorhandenes, bereits heruntergeladenes Bild erscheint, wenn die Datenverbindung schlecht ist. Abschließend wurde die Frage der Mehrsprachigkeit diskutiert. Die englischsprachige Version der Informationstexte wird durch die Touristik-Palette Hude gestellt. Eine Übersetzung ins Plattdeutsche wird bis zum 8. August intern in der Touristik-Palette Hude geklärt.

3.1.7. Hude 15. August 2013

Im Rahmen des Treffens am 15. August wurde zunächst der Umbau der App in ein neues Framework vorgestellt. Von jQuery Mobile wurde auf Foundation umgestellt, da Foundation schneller und sauberer arbeitet und eine leichtere Wartung möglich macht. Weiterhin wurde der „Huder Padd“ in „Hude Perfekt“ aufgenommen. Die dort befindlichen Sehenswürdigkeiten werden, zusätzlich zur Darstellung des Huder Padd selbst, durch Marker hervorgehoben. Die Touristik-Palette Hude teilte zudem die Entscheidung betreffend die Übersetzung der Informationstexte ins Plattdeutsche mit. Aus Zeitgründen wird eine Übersetzung nicht stattfinden. Viel Zuspruch fand im Treffen die Präsentation des neuen Designs. Änderungen, die noch integriert werden müssten sind folgende: Der Absatz zur Beschreibung des Skulpturenhauses muss entfallen (aufgrund der Schließung des Skulpturenhauses), das Ferienhaus „Ingridruh“ ist ebenfalls nicht mehr aktuell. Inhaltlich muss die „Klosterschenke“ zur „Klosterschänke“ korrigiert werden und zwei weitere Bilder (Schild des Wanderweges am Huder Bach und Bild des Ziebrunnens) aufgenommen werden. Betreffend den Zielgruppen-Filter für „Hude Perfekt“ besteht Unschlüssigkeit, welche Filterkriterien für die einzelnen Zielgruppen herangezogen werden sollten. Beschlossen wird, dass die Einstellung der Zielgruppen bestehen bleibt, weiterhin aber jeder Zielgruppe alle Inhalt angezeigt werden.

3.2. Personas

Durch eine Persona wird ein hypothetisch existierender Anwender, abgeleitet aus der Benutzergruppe des Systems, beschrieben. Die Beschreibung erfolgt dabei möglichst detailliert und enthält Eigenschaften, Vorlieben, sowie mögliches Nutzerverhalten. Personas werden meist durch Fragebögen, Interviews oder Beobachtungen der Nutzer abgeleitet [Jen10]. Im vorliegenden Fall müssen diese Beobachtungen nicht durchgeführt werden, denn die Gemeinde Hude erteilte der Touristik-Palette-Hude e.V bereits den Auftrag, den Wirtschaftsfaktor „Tourismus“ zu analysieren und weiterzuentwickeln. Durch eine Gäste- und Gastgeberbefragung, Expertengespräche und den Austausch mit politischen Gremien, sowie durch Zahlen des Statistischen Bundesamtes, wurden die Benutzergruppen bereits identifiziert. Die zwei wesentlichen Benutzergruppen sind auf der einen Seite die „Best-Ager“ und auf der anderen Seite die „jungen Familien mit Kind“ [TPH11]. Beide Gruppen sollen mittels Personas in den folgenden Abschnitten beschrieben werden.

3.2.1. Persona I - Zielgruppe: Best Ager 50+

Anna Schütte ist 54 Jahre alt, ledig und lebt in Oldenburg (Oldb). Sie hat eine Tochter, die 25 Jahre alt ist. Sie ist studierte Literaturwissenschaftlerin, schloss das Studium 1985 mit dem Diplom ab und arbeitet seitdem als freiberufliche Autorin für Jugendbücher. In ihrer Arbeit ist sie autodidaktisch tätig und arbeitet im Home Office am eigenen Notebook (Dell Latitude). Die Arbeit am Computer beschränkt sich auf die reine Schreibtätigkeit – Layout und Satz werden vom Lektorat des jeweiligen Verlages übernommen. Die Ideen zu den jeweiligen Jugendbüchern entstehen aus Inspirationen des Alltags. Das Internet wird selten zur Ideenfindung genutzt, wohl aber zum Versand der jeweiligen Entwürfe an die entsprechenden Verleger. Die Arbeit an einem Buch nimmt in den meisten Fällen – je nach Länge des Buches – vier bis zwölf Monate in Anspruch. Anna Schütte besitzt ein Smartphone, das sie vor allem zum Telefonieren, in seltenen Fällen auch zum Schreiben von SMS an ihre Tochter benutzt. Auf die Internetfähigkeit ihres Smartphones greift sie bislang vor allem während Reisen zurück, bei denen sie beispielsweise die kommenden Zugverbindungen überprüft.

Anna Schütte findet Inspiration häufig auch auf Reisen, die sie mindestens vierteljährlich unternimmt. Besonders gern reist sie innerhalb Deutschlands und nutzt dabei immer den öffentlichen Personennahverkehr, in der Regel die Deutsche Bahn. Auf Reisen zieht Anna Schütte Unterkünfte mit Komplettverpflegung vor, um sich ganz auf die Erkundung des jeweiligen Reiseziels konzentrieren zu können. Sie liebt Spaziergänge und erschließt sich vor allem zu Fuß den Urlaubsort. Kulturelle Angebote vor Ort nutzt sie ebenfalls gern, um den Charakter des Urlaubsortes hautnah zu erfahren.

Bislang nutzte Anna Schütte noch keine Apps auf ihrem Smartphone, sondern griff auf Internetnutzungsangebote der jeweiligen Unterkunft zurück um ihre täglichen Urlaubsaktivitäten zu planen. Bei der Nutzung einer App würde sie vor allem im ersten Schritt erwarten, dass das Vorhandensein und das Vorgehen bei der Installation der App gut vermarktet bzw. präsentiert wird. Die App sollte für ihre Bedürfnisse vor allem einen schnellen Zugang zu aktuellen kulturellen Angeboten des Zielortes bieten. Auch der öffentliche Personennahverkehr sollte repräsentiert sein, um Ausflüge zu benachbarten Gemeinden unternehmen zu können. Schnelligkeit ist nicht das vorstechende Kriterium, das Anna Schütte für zentral hält, Übersichtlichkeit hingegen schon.

3.2.2. Persona II - Zielgruppe: junge Familie mit Kind

Familie Müller ist eine vierköpfige Familie aus Dortmund. Sebastian Müller (35 Jahre alt) ist Physiotherapeut, Stefanie Müller (33 Jahre alt) ist Hausfrau. Sebastian Müller ist angestellt in einer physiotherapeutischen Praxis, seine Frau Stefanie ist gelernte Kosmetikerin, kümmert sich aktuell aber um die beiden Kinder des Paares Tochter Emilie und des Sohnes Max (sieben und vier Jahre alt).

Sebastian Müller fährt einmal jährlich mit seiner Familie in den Urlaub, den er langfristig planen muss. Da Tochter Emilie bereits in die Schule geht, ist die Familie auf den Zeitraum der Sommerferien beschränkt. Die Familie verfügt über keine großen finanziellen Mittel, weshalb Reiseziele häufig innerhalb Deutschlands liegen und die Reisen stets mit dem eigenen PKW unternommen werden. Eine Reiseunterkunft mit Verpflegung wäre für die Familie Müller aufgrund der Alleinverdienerrolle des Mannes zu teuer, weshalb der Familienvater Unterkünfte mit Selbstverpflegungsmöglichkeit vorzieht. Auch die Aktivitäten im jeweiligen Urlaubsort müssen finanziell überschaubar sein. Sebastian Müller ist - wie die ganze Familie - beispielsweise sportlich sehr aktiv und nimmt seit Jahren an Triathlonveranstaltungen teil. Auch im Urlaub legt Sebastian Müller viel Wert auf sportliche Unternehmungen für die ganze Familie. Die Kinder Emilie und Max freuen sich immer sehr auf die Urlaube der Familie. Besonders gefallen ihnen die Unterhaltungsprogramme für Kinder, die das jeweilige Reiseziel zu bieten hat. Emilie findet besonders Gefallen an Aktivitäten, die mit Tieren zu tun haben, wie etwa Bauernhofbesuche oder Reitausflüge. Max wiederum liebt Abenteuer und Abenteuerspielplätze, auf denen es etwas zu entdecken gibt und Wissen spielerisch vermittelt wird.

Sebastian Müller besitzt ein Smartphone der neuesten Generation. Er ist sehr technikaffin und im Umgang

3. Nutzeranforderungen

mit Apps sehr erfahren. Sowohl die Urlaubsplanung (Routenplanung und Unterkunftssuche) geschieht unter Zuhilfenahme des Internets als auch die Erkundung der Gegend während der Reise. Sebastian Müller ist eine schnelle App wichtig, die alle Bereiche des Urlaubsalltags bedient. So sind für ihn sportliche Angebote aber auch Aktivitäten und Einrichtungen für Kinder wichtig. Schlechtwettervarianten sollten die sommerlichen Angebote ergänzen. Entscheidend sind auch medizinische Angebote vor Ort, da der Sohn Max unter chronischem Asthma leidet, das während Sommerzeit und Pollenflug zu akuten Atemnotanfällen führen kann.

3.3. Anforderungen

Im Abschnitt Anforderungen sollen die zu erfüllenden Funktionen dargestellt werden, die in die App implementiert werden sollen.

3.3.1. Funktionale Anforderungen

FA 1	Die App muss dem Anwender jederzeit die Möglichkeit bieten, zur Startseite zurückzukehren.	Abbildung: 4.2
FA 2	Die App muss dem Anwender jederzeit die Möglichkeit bieten, einen QR-Code zu scannen.	Abbildung: 4.3
FA 3	Die App muss dem Anwender jederzeit die Möglichkeit bieten, sich seinen Standort anzeigen zu lassen.	Abbildung: 4.4
FA 4	Die App muss dem Anwender jederzeit die Möglichkeit bieten, das Impressum anzuzeigen.	Abbildung: 4.5
FA 5	Die App muss dem Anwender jederzeit die Möglichkeit bieten, Einstellungen anzuzeigen.	Abbildung: 4.6
FA 6	Die App muss dem Anwender die Möglichkeit bieten, eine Zielgruppe einzustellen.	Abbildung: 4.7
FA 7	Die App muss dem Anwender die Möglichkeit bieten, eine Seite zurückzugehen.	Abbildung: 4.8
FA 8	Die App muss dem Anwender Informationen über Hude präsentieren.	Abbildung: 4.9
FA 9	Die App muss dem Anwender eine Übersicht der Sehenswürdigkeiten bieten.	Abbildung: 4.12
FA 10	Die App muss dem Anwender die Sehenswürdigkeiten informativ präsentieren.	Abbildung: 4.12, 4.13
FA 11	Die App muss dem Anwender eine Übersicht der Unterkünfte bieten.	Abbildung: 4.10
FA 12	Die App muss dem Anwender eine Unterkunft informativ präsentieren.	Abbildung: 4.10, 4.11
FA 13	Die App muss dem Anwender eine Übersicht der Gastronomien bieten.	Abbildung: 4.14
FA 14	Die App muss dem Anwender eine Übersicht der EC-Automaten in Hude anzeigen.	Abbildung: 4.15
FA 15	Die App muss dem Anwender Einkaufsmöglichkeiten in Hude anzeigen.	Abbildung 4.16
FA 16	Die App muss dem Anwender Parkmöglichkeiten in Hude anzeigen.	Abbildung 4.17
FA 17	Die App muss dem Anwender Gesundheitseinrichtungen in Hude anzeigen.	Abbildung 4.18

FA 18	Die App muss dem Anwender die bevorstehenden Veranstaltungen in Hude anzeigen.	Abbildung 4.19
FA 19	Die App muss dem Anwender Spiele und Rätsel anbieten.	Abbildung 4.20
FA 20	Die App muss den Anwender eine Fotojagd durchführen lassen.	Abbildung 4.20, 4.21
FA 21	Die App muss den Anwender ein Planetenabitur lösen lassen.	Abbildung 4.20, 4.22
FA 22	Die App muss dem Nutzer, für eine unterschiedliche Dauer an Tagen und eingestellte Zielgruppen, Vorschläge für Unternehmungen anzeigen.	Abbildungen 4.23
FA 23	Die App muss dem Anwender Standorte von Örtlichkeiten auf einer Karte anzeigen.	Abbildungen: 4.13, 4.11, 4.14, 4.15, 4.16, 4.17, 4.18

3.3.2. Optionale Anforderungen

OA 1	Die App soll verschiedene Schriftgrößen unterstützen.
OA 2	Die App soll verschiedene Sprachen in der Ausgabe der Texte unterstützen.
OA 3	Die App soll über eine Sprachausgabe verfügen.
OA 4	Die App soll verschiedene Sprachen in der Sprachausgabe unterstützen.
OA 5	Die App soll dem Nutzer ein Navigationspanel zur Navigation innerhalb der App bieten.
OA 6	Die App soll dem Nutzer Spielplätze auf einer Karte anzeigen.
OA 7	Die App soll bei einer nicht ausreichenden Datenverbindung einen Kartenausschnitt, anstatt einer Google Maps Karte, als Grafik bereitstellen.

3.3.3. Nicht-funktionale Anforderungen

NFA 1	Die App muss ohne Registrierung nutzbar sein.
NFA 2	Die App muss leicht wartbar sein.
NFA 3	Die App muss leicht modifizierbar sein.
NFA 4	Die App muss portierbar sein.
NFA 5	Die App muss zuverlässig sein.
NFA 6	Die App muss korrekt funktionieren.
NFA 7	Die App muss auf den gängigen Smartphones lauffähig sein.
NFA 8	Die App muss der Corporate Identity des Touristik-Palette Hude e.V. entsprechen.
NFA 9	Die App muss mindestens die Unternehmen, die Mitglied des Touristik-Palette Hude e.V. sind, präsentieren.
NFA 10	Die App muss für jüngere und ältere Nutzergruppen bedienbar sein.
NFA 11	Die App muss intuitiv bedienbar sein.
NFA 12	Die App muss rechtskonform sein.
NFA 13	Die App muss über die App Stores der mobilen Betriebssysteme vertrieben und bezogen werden.
NFA 14	Die App muss als Web-App aufrufbar sein.
NFA 15	Die App muss ein Header-Element haben.
NFA 16	Die App muss ein Footer-Element haben.
NFA 17	Die App muss ein Panel-Element haben.
NFA 18	Die App muss das GPS-Modul nutzen.

3. Nutzeranforderungen

NFA 19	Die App muss das Kamera-Modul nutzen.
NFA 20	Die App muss einen QR-Code-Scanner integriert haben.
NFA 21	Die App muss über QR-Codes Inhalte direkt öffnen.
NFA 22	Die App muss ohne Internetzugang nutzbar sein.

4. Anwendungsfälle

Aufbauend auf den Anforderungen (siehe Kapitel 3 „Nutzeranforderungen“ auf Seite 12) werden in diesem Kapitel die resultierenden Anwendungsfälle definiert und dargestellt.

4.1. Anwendungsfalldiagramm

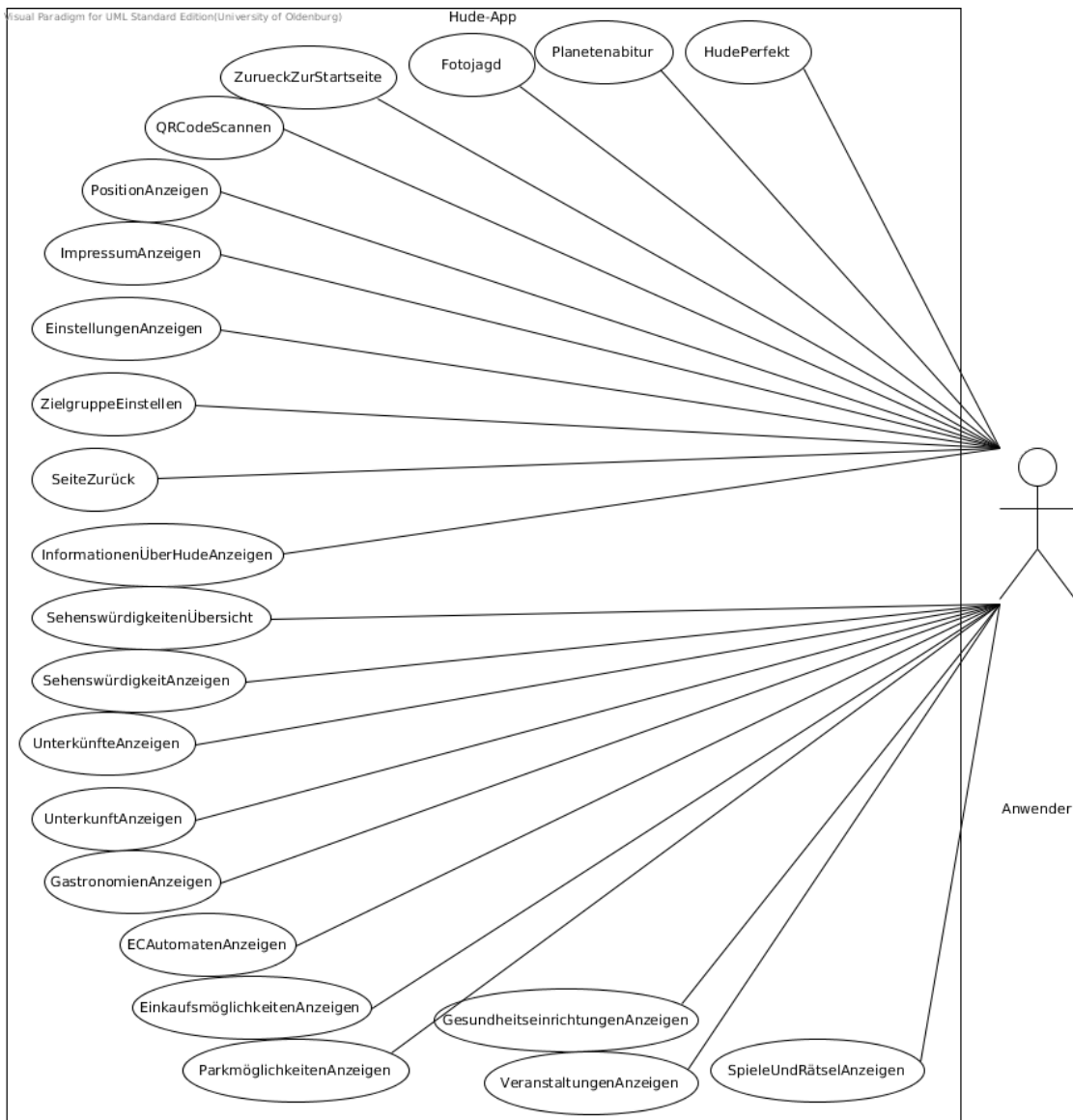


Abbildung 4.1.: Anwendungsfalldiagramm für die Touristen-App von Hude

4.2. Anwendungsfallmodell

Im folgenden Abschnitt zum Anwendungsfallmodell werden die Akteure, sowie die einzelnen Anwendungsfälle, detailliert vorgestellt. Zur Präsentation der Anwendungsfälle zählen die Angabe des Namens,

4. Anwendungsfälle

der involvierten Akteure, des Verlaufs, der Anfangs- und Abschlussbedingungen, sowie möglicher Qualitätsanforderungen an den Anwendungsfall.

4.2.1. Akteure

Anwender	Jede Person, die die App (native App oder Web-App, auf dem Smartphone nutzt, ist ein Anwender.
----------	--

4.2.2. Anwendungsfälle Anwender

Name	ZurückZurStartseite
Akteur	Anwender
Ablauf	<ol style="list-style-type: none">1. Der Anwender drückt den Home-Button im Footer.2. <i>Die App öffnet die Startseite.</i>
Anfangsbedingung	Es ist eine Seite, abgesehen von der Startseite, geöffnet.
Abschlussbedingung	Die Startseite der App wird angezeigt.
Qualitätsanforderung	

Abbildung 4.2.: Anwendungsfall: ZurückZurStartseite

Name	QRCodeScannen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none">1. Der Anwender drückt auf den QR-Code-Button im Footer.2. <i>Die App startet das QR-Code-Scanner-Plugin.</i>3. Der Anwender scannt mit dem QR-Code-Plugin einen QR-Code ein.4. <i>Die App öffnet die entsprechende Seite.</i>
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Die App öffnet die Seite mit den passenden Inhalten zum QR-Code.
Qualitätsanforderung	Bei einem nicht zur App gehörenden QR-Code wird der Anwender auf eine Fehlerseite weitergeleitet.

Abbildung 4.3.: Anwendungsfall: QR-Code-Scannen

Name	PositionAnzeigen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender drückt den Positions-Button im Footer. 2. <i>Die App ruft über das GPS¹-Modul des Smartphones die aktuelle Position des Anwenders ab und stellt diese auf einer Google-Maps-Karte dar.</i>
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Die Position des Anwenders wird angezeigt.
Qualitätsanforderung	Ist die Datenverbindung nicht ausreichend, wird die Position mittels der GPS-Koordinaten, Höhe und Richtung dargestellt.

Abbildung 4.4.: Anwendungsfall: PositionAnzeigen

Name	ImpressumAnzeigen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender drückt den Einstellungen-Button . 2. <i>Die App öffnet die Seite mit den Einstellungen.</i> 3. Der Anwender drückt auf die Überschrift „Impressum“. 4. <i>Die App öffnet das Klappmenu und zeigt das Impressum an.</i>
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Dem Anwender wird das Impressum angezeigt.
Qualitätsanforderung	

Abbildung 4.5.: Anwendungsfall: ImpressumAnzeigen

¹Global Positioning System (GPS) ist ein satellitengestütztes System zur Positionsbestimmung.

4. Anwendungsfälle

Name	EinstellungenAnzeigen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender klickt auf den Einstellungen-Button. 2. <i>Die App öffnet die Einstellungenseite.</i> 3. Der Anwender klickt auf die Überschrift einer Einstellung. 4. <i>Die App öffnet das Klappmenu der Einstellung.</i>
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Dem Anwender werden die Einstellungen angezeigt.
Qualitätsanforderung	

Abbildung 4.6.: Anwendungsfall: EinstellungenAnzeigen

Name	ZielgruppeEinstellen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender wählt über den Einstellungen-Button die Einstellungen. 2. <i>Die App zeigt dem Anwender die Einstellungen an.</i> 3. Der Anwender drückt die Überschrift „Zielgruppeneinstellungen“. 4. <i>Die App öffnet das Klappmenu zur Einstellung der Zielgruppe.</i> 5. Der Anwender wählt eine Zielgruppe aus.
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Die App speichert die Zielgruppeneinstellung.
Qualitätsanforderung	

Abbildung 4.7.: Anwendungsfall: ZielgruppeEinstellen

Name	SeiteZurück
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender drückt die Zurück-Taste auf dem Smartphone (Android) oder den Zurück-Button im Header (iOS). 2. <i>Die App öffnet die zuvor geöffnete Seite.</i>
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Es wird die zuvor geöffnete Seite angezeigt.
Qualitätsanforderung	

Abbildung 4.8.: Anwendungsfall: SeiteZurück

Name	InformationenÜberHudeAnzeigen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender drückt auf „Hude-Info“ auf dem Startbildschirm. 2. <i>Die App öffnet die Seite mit den Informationen über Hude.</i>
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Dem Anwender werden die Informationen über Hude angezeigt.
Qualitätsanforderung	

Abbildung 4.9.: Anwendungsfall: InformationenÜberHudeAnzeigen

Name	UnterkünfteAnzeigen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender klickt auf „Unterkünfte“. 2. <i>Die App öffnet eine Seite mit der Übersicht der Unterkünfte in Hude.</i>
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Die App zeigt eine Übersicht der Unterkünfte.
Qualitätsanforderung	

Abbildung 4.10.: Anwendungsfall: UnterkünfteAnzeigen

4. Anwendungsfälle

Name	UnterkunftAnzeigen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender drückt auf eine Unterkunfts-kategorie in der Unterkunftsübersicht. 2. <i>Die App öffnet das Klappmenu der Unterkunfts-kategorie und zeigt deren Unterkünfte an.</i> 3. Der Anwender wählt eine angezeigte Unterkunft aus. 4. <i>Die App öffnet die Detailansicht der ausgewählten Unterkunft.</i>
Anfangsbedingung	Die Unterkunftsübersicht ist geöffnet.
Abschlussbedingung	Eine ausgewählte Unterkunft wird im Detail (Beschreibung, Preise, Kontakt und Bilder) angezeigt.
Qualitätsanforderung	

Abbildung 4.11.: Anwendungsfall: UnterkunftAnzeigen

Name	SehenswürdigkeitenÜbersicht
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender drückt auf der Startseite auf „Sehenswürdigkeiten“. 2. <i>Die App öffnet eine Übersicht der Sehenswürdigkeiten in Hude.</i>
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Dem Anwender wird eine Übersicht der Sehenswürdigkeiten präsentiert.
Qualitätsanforderung	

Abbildung 4.12.: Anwendungsfall: SehenswürdigkeitenÜbersicht

Name	SehenswürdigkeitAnzeigen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender wählt eine Sehenswürdigkeit aus der Übersicht aus. 2. <i>Die App öffnet die Seite der Sehenswürdigkeit.</i>
Anfangsbedingung	Die Seite mit der Übersicht der Sehenswürdigkeiten ist geöffnet.
Abschlussbedingung	Eine detaillierte Ansicht der Sehenswürdigkeit wird angezeigt.
Qualitätsanforderung	

Abbildung 4.13.: Anwendungsfall: SehenswürdigkeitAnzeigen

Name	GastronomienAnzeigen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender klickt auf „Gastronomien“. 2. <i>Die App öffnet die Seite der Gastronomien in Hude.</i> 3. Der Anwender drückt die Überschrift „Gastronomie“. 4. <i>Die App zeigt eine Übersicht der Gastronomien in Hude.</i> 5. Der Anwender drückt auf eine Gastronomie. 6. <i>Die App öffnet das Klappmenu der ausgewählten Gastronomie.</i>
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Die App zeigt die Details (Kontakt und Öffnungszeiten) der ausgewählten Gastronomie aus.
Qualitätsanforderung	

Abbildung 4.14.: Anwendungsfall: GastronomienAnzeigen

4. Anwendungsfälle

Name	ECAutomatenAnzeigen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender klickt auf „EC-Automaten“. 2. <i>Die App öffnet die Seite der EC-Automaten in Hude.</i> 3. Der Anwender drückt eine der Überschriften „Banken & Sparkassen“ oder „Post“. 4. <i>Die App zeigt eine Übersicht der EC-Automaten der Kategorie in Hude an.</i> 5. Der Anwender drückt auf die Überschrift einer Filiale. 6. <i>Die App öffnet das Klappmenu der ausgewählten Filiale.</i>
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Die App zeigt die Details der ausgewählten Filiale an.
Qualitätsanforderung	

Abbildung 4.15.: Anwendungsfall: ECAutomatenAnzeigen

Name	EinkaufsmöglichkeitenAnzeigen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender klickt auf „Einkaufsmöglichkeiten“. 2. <i>Die App öffnet die Seite der Einkaufsmöglichkeiten in Hude.</i> 3. Der Anwender drückt auf eine der Kategorien von Einkaufsmöglichkeiten. 4. <i>Die App zeigt die Übersicht der Einkaufsmöglichkeiten der Kategorie an.</i> 5. Der Anwender drückt auf die Überschrift einer Einkaufsmöglichkeit. 6. <i>Die App öffnet das Klappmenu der ausgewählten Einkaufsmöglichkeit.</i>
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Die App zeigt die Details der ausgewählten Einkaufsmöglichkeit (Kontakt und Standort) an.
Qualitätsanforderung	

Abbildung 4.16.: Anwendungsfall: EinkaufsmöglichkeitenAnzeigen

Name	ParkmöglichkeitenAnzeigen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender klickt auf „Parkmöglichkeiten“. 2. <i>Die App öffnet die Seite der Parkmöglichkeiten in Hude.</i>
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Die App zeigt die Parkmöglichkeiten, Kartenausschnitte, in denen die Parkmöglichkeiten vermerkt sind, und bei ausreichender Datenverbindung außerdem die Parkmöglichkeiten in einer Google-Maps-Karte an.
Qualitätsanforderung	

Abbildung 4.17.: Anwendungsfall: ParkmöglichkeitenAnzeigen

Name	GesundheitseinrichtungenAnzeigen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender klickt auf „Gesundheit“. 2. <i>Die App öffnet die Seite der Gesundheitseinrichtungen in Hude.</i> 3. Der Anwender drückt auf eine der Kategorien von Gesundheitseinrichtungen. 4. <i>Die App zeigt eine Übersicht der Gesundheitseinrichtungen der Kategorie</i> 5. Der Anwender drückt auf die Überschrift einer Gesundheitseinrichtung. 6. <i>Die App öffnet das Klappmenu der ausgewählten Gesundheitseinrichtung.</i>
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Die App zeigt die Details der ausgewählten Gesundheitseinrichtung (Kontakt und Standort) an, sowie einen Anruf-Button.
Qualitätsanforderung	

Abbildung 4.18.: Anwendungsfall: GesundheitseinrichtungenAnzeigen

4. Anwendungsfälle

Name	VeranstaltungenAnzeigen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender drückt auf „Veranstaltungen“. 2. <i>Die App öffnet die Seite der Veranstaltungen.</i>
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Die App zeigt dem Anwender die bevorstehenden Veranstaltungen an.
Qualitätsanforderung	

Abbildung 4.19.: Anwendungsfall: VeranstaltungenAnzeigen

Name	SpieleUndRätselAnzeigen
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender drückt auf „Schnitzeljagd“. 2. <i>Die App öffnet die Seite der Schnitzeljagd.</i>
Anfangsbedingung	Die App ist geöffnet.
Abschlussbedingung	Die App zeigt dem Anwender mögliche Spiele und Rätsel an (Fotojagd und Planetenabitur).
Qualitätsanforderung	

Abbildung 4.20.: Anwendungsfall: SpieleUndRätselAnzeigen

Name	Fotojagd
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender drückt auf die Kategorie „Fotojagd“. 2. <i>Die App öffnet das Klappmenu der Fotojagd und zeigt die Bereiche der Fotojagd an.</i> 3. Der Anwender drückt auf einen Bereich der Fotojagd. 4. <i>Die App öffnet die Ansicht mit den zu suchenden Objekten.</i>
Anfangsbedingung	Die Seite „Schnitzeljagd“ ist geöffnet.
Abschlussbedingung	Der Anwender kann die Objekte suchen und als gefunden markieren.
Qualitätsanforderung	

Abbildung 4.21.: Anwendungsfall: Fotojagd

Name	Planetenabitur
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender drückt auf die Kategorie „Planetenabitur“. 2. <i>Die App öffnet das Klappmenu des Planetenabiturs und zeigt die unterschiedlichen Versionen des Planetenabiturs an.</i> 3. Der Anwender drückt auf ein Planetenabitur. 4. <i>Die App öffnet die Ansicht des ausgewählten Planetenabiturs.</i>
Anfangsbedingung	Die Seite „Schnitzeljagd“ ist geöffnet.
Abschlussbedingung	Der Anwender kann mit der Lösung des Planetenabiturs beginnen.
Qualitätsanforderung	

Abbildung 4.22.: Anwendungsfall: Planetenabitur

Name	HudePerfekt
Akteur	Anwender
Ablauf	<ol style="list-style-type: none"> 1. Der Anwender drückt auf „Hude-Perfekt“. 2. <i>Die App öffnet die Seite „Hude-Perfekt“.</i>
Anfangsbedingung	Die App ist gestartet und die Startseite ist geöffnet.
Abschlussbedingung	Dem Anwender werden Vorschläge, den gesetzten Einstellungen entsprechend, für einen Aufenthalt in Hude gemacht.
Qualitätsanforderung	

Abbildung 4.23.: Anwendungsfall: HudePerfekt

5. Entwurf

Das Kapitel Entwurf gibt einen Blick auf die Architektur, die Datenstruktur, die zu implementierenden Methoden und die Abläufe innerhalb der App, sowie eine Übersicht des evolutionären Prototypen.

5.1. Architektur

Die Architektur der App setzt sich aus verschiedenen Komponenten zusammen (siehe Abbildung 5.1), wobei hier zwischen den App-Komponenten und den Geräte-Komponenten zu unterscheiden ist. Letztere gelten als Basis, da auf dem Smartphone die App ausgeführt wird und mögliche Hardware-Komponenten wie z.B. Sensoren oder die Kamera, mittels des API angesprochen und genutzt werden können. Die App selbst besteht aus drei Hauptkomponenten, die in sich weitere Komponenten enthalten. Die erste Komponente ist das PhoneGap-Framework (Abschnitt 6.1.3 auf Seite 53 beschreibt das Framework detailliert), welches mit einem API auf die Hardwarekomponenten des Gerätes zugreifen kann und ebenso auf PlugIn¹s, die sich zusätzlich einbinden lassen und zur Erweiterung der App genutzt werden können. Die zweite Komponente ist die WebView, ein Vollbildbrowser ohne Adresszeile. In der WebView ist es zudem möglich, Daten im Hintergrund zu speichern. Dies geschieht im sogenannten „Local Storage“, der Daten als „Key-Value-Paare“ speichert. Vergleichbar ist dies mit einem assoziativen Array. Die im Local Storage gespeicherten Daten stehen auch ohne Datenverbindung zur Verfügung. Eine Web-App kann somit auch anteilig offline genutzt werden. Die gespeicherten Daten sind zudem nur über die eine Domain verfügbar. Der Zugriff auf den Local Storage erfolgt über JS. Zur Nutzung des Local Storage werden fünf Funktionen bereitgestellt. Mit diesen ist es möglich Daten zu speichern, abzurufen, einzelne Daten zu löschen, alle Daten zu löschen oder die Anzahl der gespeicherten Daten abzurufen [Pot12]. Die dritte Komponente ist die Web App selbst. Diese Komponente setzt sich aus den Webstandards HTML, CSS, JS und weiteren Elementen wie z.B. Grafiken zusammen. Durch den Einsatz von JS kann durch das JS-PhoneGap-API, über die WebView, das API von PhoneGap angesprochen werden. Dadurch wird ein Zugriff auf die PlugIns von PhoneGap ermöglicht, sowie ein Zugriff auf die Geräte-Hardware.

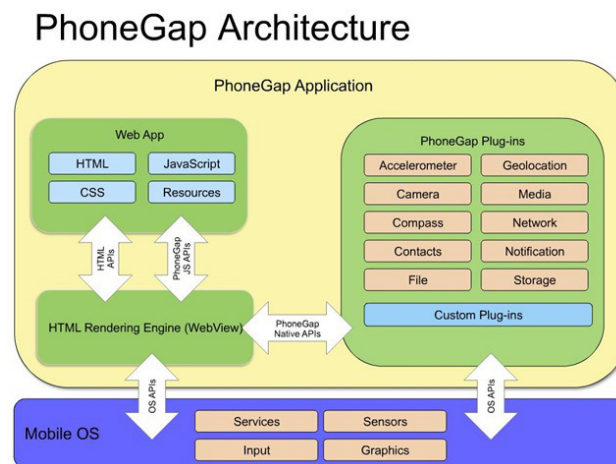


Abbildung 5.1.: Architektur einer mit PhoneGap realisierten App[pho11]

¹Ein PlugIn ist eine Erweiterung für ein bereits bestehendes Programm, das auf eine Erweiterung der Funktionalität dieses Programms abzielt.

5.2. Datenstruktur

Die Datenstruktur der App entspricht dabei zwei Baumstrukturen. Die erste Baumstruktur ist innerhalb von strukturierten HTML-Dokumenten eingebettet und wird als „Document Object Model“ (DOM) bezeichnet. Wie die Daten für die App der Touristik-Palette-Hude in HTML-Dokumenten gespeichert sind, wird in Kapitel 6 „Implementierung“ in Abschnitt 6.3.2 und 6.3.2 erläutert. Die Abbildung 5.2 auf Seite 31 zeigt die einzelnen HTML-Dokumente (blaue Punkte), die Datensätze der App enthalten.

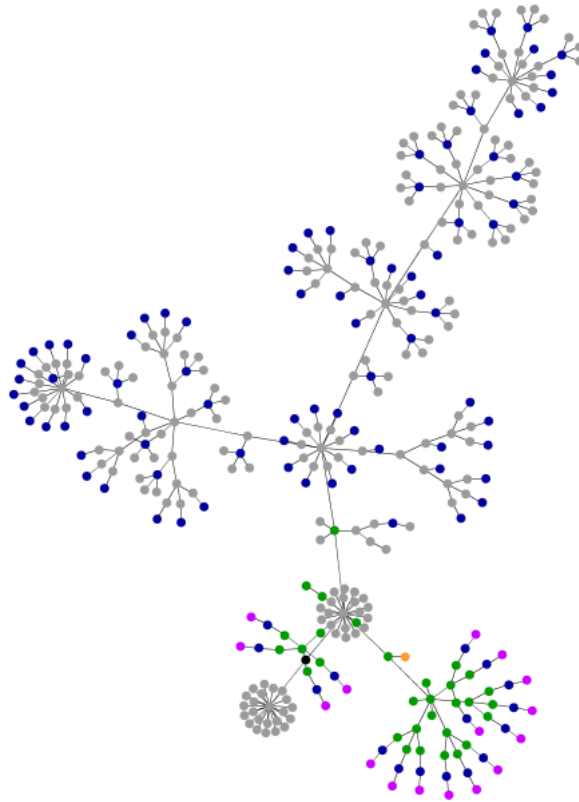


Abbildung 5.2.: Struktur der App

Die zweite Baumstruktur befindet sich auf dem internen Speicher (Dateisystem) des Smartphones und wird durch verschiedene Ordner repräsentiert. Innerhalb dieser befinden sich die einzelnen HTML-Dokumente und zusätzliche Daten, wie z.B. Grafiken, die in die HTML-Dokumente eingebunden werden. Um die Daten persistent auf dem Smartphone zu speichern und diese jederzeit mit der App aufrufen zu können, wird das Dateisystem des Smartphones genutzt. Auf diesem befinden sich neben den Datensätzen ebenfalls die JavaScript-Dateien, die Cascading-Style-Sheets-Dateien, sowie die Grafiken und Bilder.

Die Abbildung 5.3 auf Seite 31 zeigt die erste Ebene, des zur App gehörenden Dateisystems. Innerhalb des Ordners „www“ befinden sich die Unterordner „css“, „images“, und „js“, sowie die einzelnen Datensätze in HTML-Dateien. Mittels dieser Struktur kann mit den HTML-Dokumenten auf die, sich in der Struktur tiefer befindlichen, Elemente einfach zugegriffen werden.

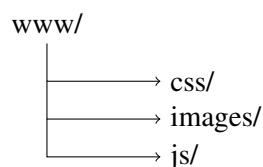


Abbildung 5.3.: Erste Ebene des Dateisystems der App

5.3. Methodenübersicht

tphStartApp() wird beim Start der App ausgeführt und überprüft, ob diese zum ersten Mal startet. Ist dies der Fall, werden die Grundeinstellungen im Local Storage des Gerätes gesetzt.

tphAudioAbspielen(file) startet die Wiedergabe einer Audio-Datei. Der Pfad zur Audio-Datei wird mittels Parameter übergeben.

tphAudioDateien() liefert ein Array mit den Pfaden der Audio-Dateien, die sich auf dem Server befinden, zurück.

tphAudioPause() pausiert die Wiedergabe der abgespielten Audio-Datei.

tphAudioStoppen() stoppt die Wiedergabe der abgespielten Audio-Datei.

tphConvertDMStoDec(dmsArray) berechnet aus Grad, Minuten und Sekunden den Dezimalwert. Übergeben wird ein Array mit drei Elementen [0] - Grad, [1] - Minuten, [2] - Sekunden.

tphDateisystem(option, dateiname) ermöglicht den Zugriff auf das Dateisystem. Der Parameter „option“ gibt dabei die auszuführende Operation an und kann folgende Werte übergeben bekommen:

erstellen

auslesenHude

auslesenAudio

loeschen

download

audioVorhanden

Der Parameter „dateiname“, ist optional und wird nur bei der Operation „audioVorhanden“ benötigt. Er enthält einen Dateinamen und wird zur Prüfung der Existenz übergeben.

tphDownloadPfad(url) liefert den Downloadpfad einer Audio-Datei zurück, wenn die übergebene URL einer bestimmten Domain entspricht.

tphEinstellungenSetzen() setzt die im Local Storage gespeicherten Einstellungen auf der Einstellungsseite.

tphEinstellungenUeberwacher() überwacht, ob Einstellungen vorgenommen bzw. geändert werden, und gibt die Änderungen an die zuständigen Funktionen weiter.

tphGPSAbstand(lat1, lon1, lat2, lon2) berechnet den Abstand zwischen zwei dezimalen GPS-Koordinaten.

tphGoogleMapsBildMitMarker(lat, lon, zoom, elementID) hängt einen Ausschnitt (Zoomlevel) eines Gebietes anhand von GPS-Koordinaten (lat, lon) an ein Element (elementID) an.

tphHeaderUberschriftAendern(neueUeberschrift) setzt die Überschrift auf einer Seite.

tphHoleAudioDateienHeruntergeladen() lädt aus dem Local Storage die Anzahl der bereits heruntergeladenen Audio-Dateien.

tphHoleFotojagdBilderAusLocal Storage() lädt die Anzahl und den Status (gefunden/nicht gefunden) der Bilder einer Fotojagd aus dem Local Storage.

tphHoleGPSAusBild(imgID) liest die GPS-Koordinaten aus einem Bild heraus. Dazu wird die ID des Bildes übergeben.

- tphHoleOS()** liest das Betriebssystem des Gerätes aus.
- tphHoleSchriftgroesse()** liest die im Local Storage aktuell gesetzte/gespeicherte Schriftgröße aus.
- tphHoleSchriftgroesseNormal()** liest die standardmäßige Schriftgröße aus.
- tphHoleSprache()** liest die für den Text eingestellte Sprache aus.
- tphHoleSpracheAudio()** liest die für die Audio-Dateien eingestellte Sprache aus.
- tphHoleZielgruppe()** liest die eingestellte Zielgruppe aus.
- tphHuderPadd()** enthält die GPS-Koordinaten des Huder Padd.
- tphLadeLocal Storage()** erstellt eine Instanz für den Zugriff auf den Local Storage.
- tphLadeVeranstaltungen()** lädt die gespeicherten Veranstaltungen vom Server.
- tphNutzeGPS(option)** ermöglicht die Abfrage der Position über das GPS-Modul und stellt mit unterschiedlichen Optionen (optionen) Google-Maps-Karten bereit.
- tphParkplaetze()** liefert ein Array mit den GPS-Koordinaten der Parkplätze in Hude.
- tphQRCodeScan()** ruft das PhoneGap-BarcodeScanner-Plugin auf.
- tphSchnitzeljagdPlanetenlehrpfadAbiturA()** wertet das Planetenabitur A aus und gibt das Ergebnis in der View aus.
- tphSchnitzeljagdPlanetenlehrpfadAbiturB()** wertet das Planetenabitur B aus und gibt das Ergebnis in der View aus.
- tphSchnitzeljagdPlanetenlehrpfadAbiturC()** wertet das Planetenabitur C aus und gibt das Ergebnis in der View aus.
- tphSetzeEinstellungenAufSeite()** setzt die gespeicherten Einstellungen aus dem Local Storage auf den einzelnen Seiten.
- tphSetzeFotojagdBilderAufNichtGefunden()** startet die geöffnete Fotojagd neu, indem alle Bilder im Local Storage auf false gesetzt werden.
- tphSpeicherDateienHeruntergeladen(anzahl)** speichert die Anzahl der heruntergeladenen Audio-Dateien im Local Storage.
- tphSpeicherFotojagdBilderImLocal Storage()** speichert die Bilder beim ersten Aufruf einer Fotojagd als nicht gefunden im Local Storage.
- tphSpeicherSchriftgroesse(tphSchriftgroesse)** speichert die übergebene Schriftgröße im Local Storage.
- tphSpeicherSprache(tphSprache)** speichert die für den Text übergebene Sprache im Local Storage.
- tphSpeicherSpracheAudio(tphSpracheAudio)** speichert die für die Audio-Dateien übergebene Sprache im Local Storage.
- tphSpeicherZielgruppe(tphZielgruppe)** speichert die übergebene Zielgruppe im Local Storage.
- tphSpielplaetze()** gibt ein Array mit den GPS-Koordinaten der schönsten Spielplätze zurück.

tphSplitURL(url) separiert aus der URL den aufzurufenden Teil für den integrierten BarcodeScanner.

tphUeberpruefeMultipleChoice(name) überprüft die übergebene Antwort auf eine Multiple-Choice-Frage auf ihre Richtigkeit.

tphVersteckeOS() blendet für ein Betriebssystem den Inhalt aus, wenn eine Versteck-Klasse für das Betriebssystem gesetzt ist.

5.4. Sequenzdiagramme

In diesem Abschnitt werden durch Sequenzdiagramme bestimmte Interaktionen zwischen dem Anwender und der App dargestellt.

Das Sequenzdiagramm in Abbildung 5.4 auf Seite 34 beschreibt den Startvorgang der App. Durch den Aufruf der App wird diese gestartet. Dem Anwender wird während des Startvorgangs eine Startgrafik angezeigt. Nach Abschluss des Startvorgangs wird diese ausgeblendet und die App kann vom Anwender genutzt werden.

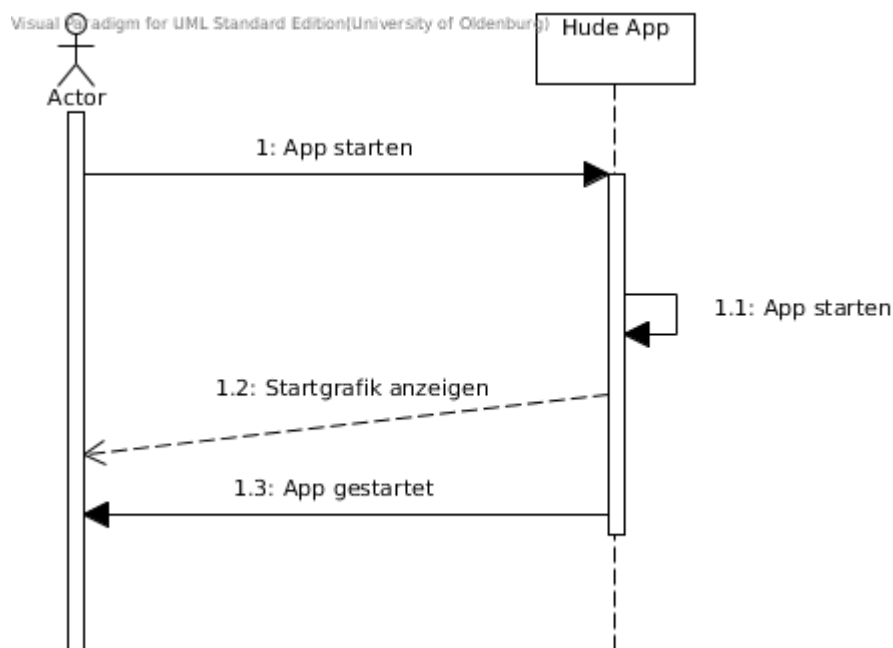


Abbildung 5.4.: Sequenzdiagramm: App-Start

Der Verlauf der Navigation wird in Abbildung 5.5 auf Seite 35 dargestellt. So wird ein Seitenaufruf angestoßen und asynchron behandelt. Die App zeigt dem Anwender nach dem Laden die Seite an.

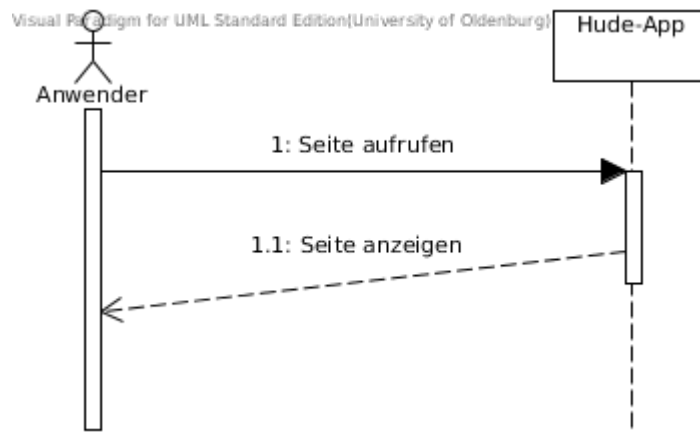


Abbildung 5.5.: Sequenzdiagramm: Navigation

Damit es der App möglich ist Audio-Inhalte wiederzugeben ist ein Download dieser notwendig. Der Ablauf des Downloads wird in Abbildung 5.6 auf Seite 35 dargestellt. Durch den Anwender wird der App mitgeteilt, dass der Download der Audio-Dateien gestartet werden soll (1). Die App stellt eine Verbindung zum Server her (1.1), auf dem die Audio-Dateien hinterlegt sind. Nun erfolgt eine Abfrage, sowie der Download aller Audio-Dateien (1.2). Der Anwender erhält nach Abschluss des Downloads darüber eine Benachrichtigung (1.3). Ist es der App nicht möglich eine Verbindung zum Server herzustellen, so wird der Anwender ebenfalls darüber informiert (1.4).

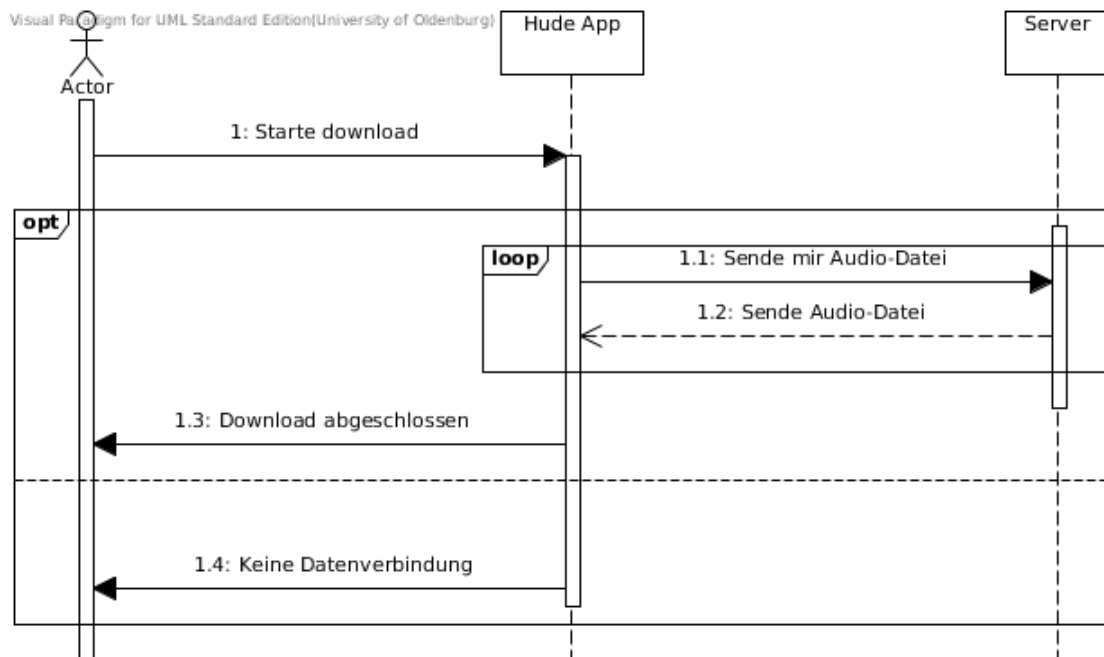


Abbildung 5.6.: Sequenzdiagramm: Download der Audio-Inhalte

Nach dem Download der Audio-Datei (Abbildung 5.6, Seite 35) ist auch das Abspielen dieser möglich. Der Verlauf des Abspielens wird im Sequenzdiagramm in der Abbildung 5.7 auf Seite 36 dargestellt. Liegen keine Audio-Dateien vor, so wird dem Anwender dies mitgeteilt (6). Sind Audio-Dateien vorhanden, so kann der Anwender die Wiedergabe starten (1) und die Wiedergabe erfolgt (2). Während der Wiedergabe kann diese pausiert (2) oder gestoppt (5) werden. Wird eine Wiedergabe pausiert, so kann diese fortgesetzt werden (3) oder gestoppt werden (4). Wurde eine Wiedergabe gestoppt, muss die Audio-Datei neu abgespielt werden (1).

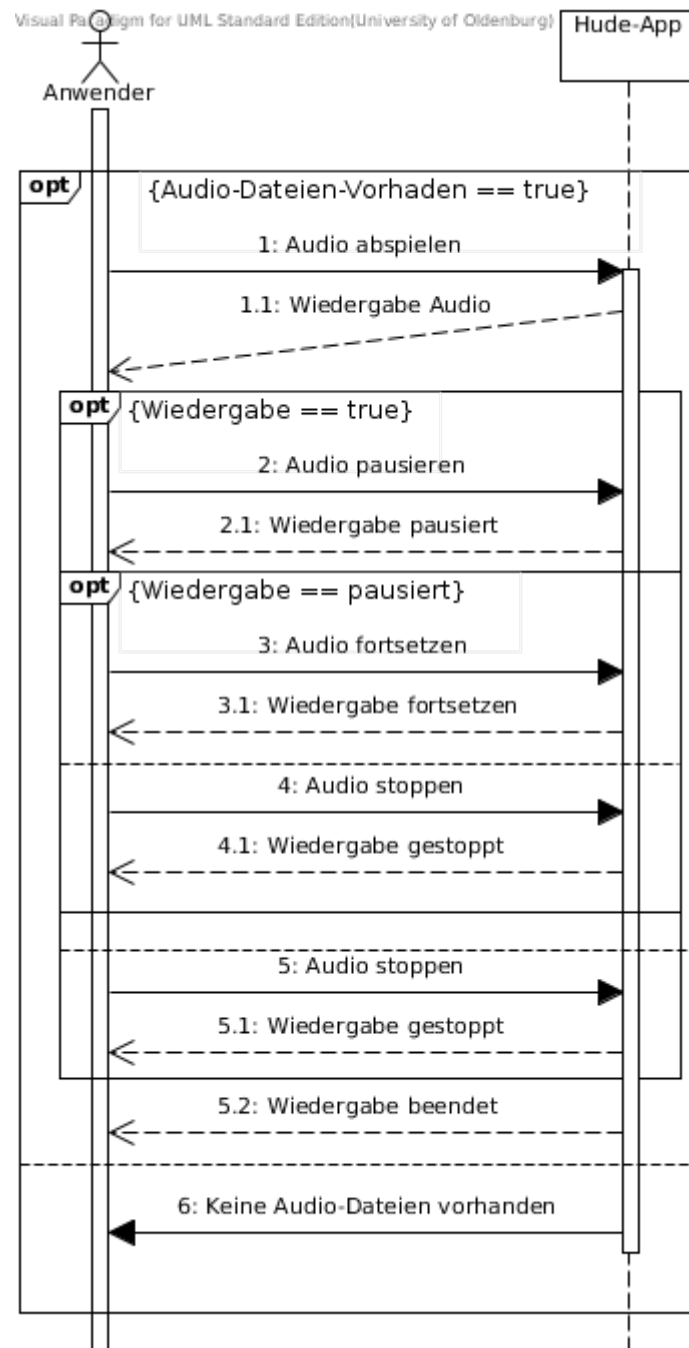


Abbildung 5.7.: Sequenzdiagramm: Abspielen von Audio-Inhalten

Die Darstellung von Kartenmaterial wird im Sequenzdiagramm der Abbildung 5.8 auf Seite 37 dargestellt. Der Anwender ruft eine Seite auf, die Kartenmaterial enthalten soll (1). Die App bestimmt zunächst die eigene Position (1.1) und die Art der Datenverbindung (2). Liegt eine ausreichend gute Datenverbindung vor, so wird das Kartenmaterial vom Google-Maps-Server angefordert (3). Das Material wird an die App gesendet (3.1) und dem Anwender dargestellt (4). Liegt keine ausreichende Datenverbindung vor, so werden dem Anwender die Koordinaten seiner Position angezeigt (5) bzw. passende Kartenausschnitte, die auf dem Smartphone gespeichert sind.

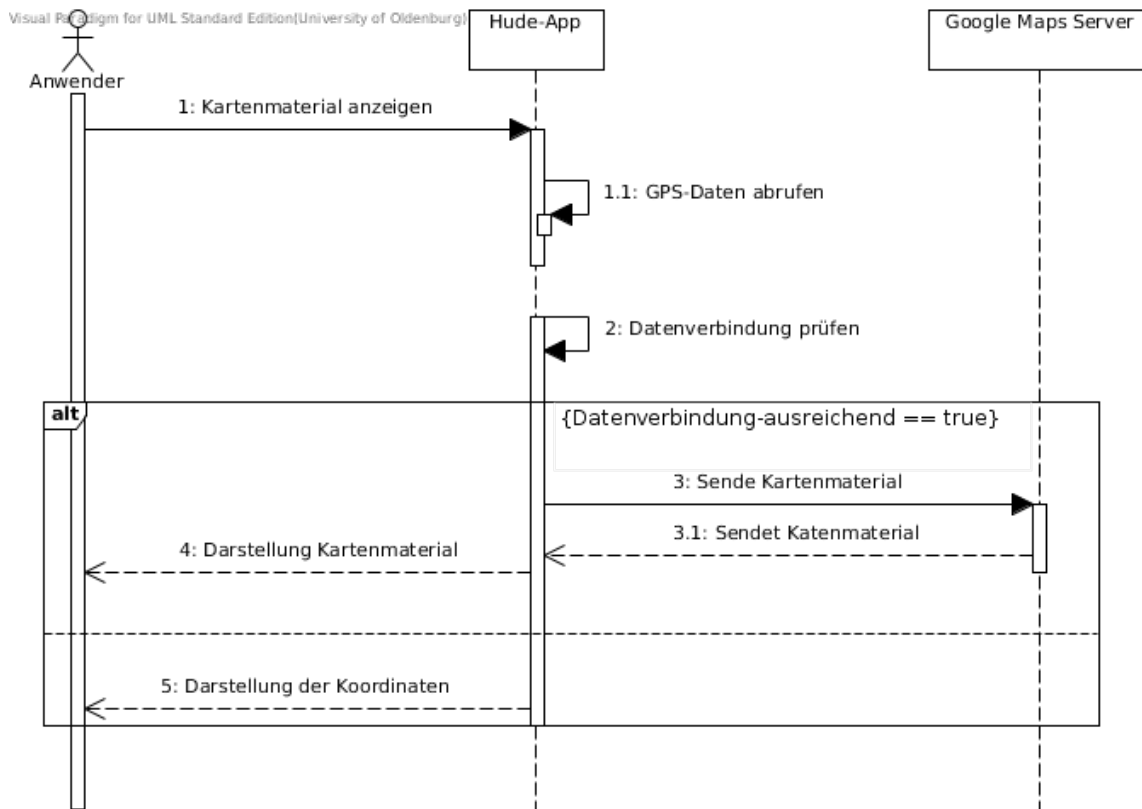


Abbildung 5.8.: Sequenzdiagramm: Kartenmaterial

Der Ablauf um einen QR-Code zu scannen wird durch das Sequenzdiagramm in Abbildung 5.9 auf Seite 38 dargestellt. Der Anwender ruft den QR-Code-Scanner auf (1). Die App startet diesen und stellt ihn dem Nutzer bereit (1.1). Der Nutzer übermittelt der App durch den QR-Code-Scanner einen QR-Code (2). Ist dieser gültig, wird die entsprechende Seite aufgerufen (2.1), ist dies nicht der Fall, wird der Anwender über die Ungültigkeit des QR-Codes benachrichtigt (2.2).

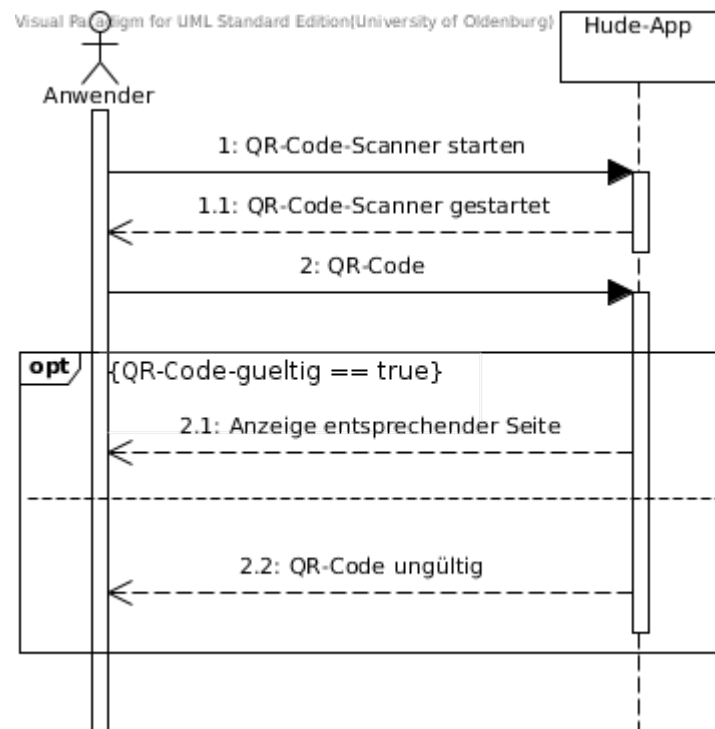


Abbildung 5.9.: Sequenzdiagramm: QR-Code

Im Sequenzdiagramm der Abbildung 5.10 auf Seite 39 wird der Ablauf der Fotojagd dargestellt. Der Anwender wählt ein Foto aus (1), das ihm durch die App in einer fancybox (siehe Abschnitt 6.1.1 auf Seite 51) geöffnet wird (1.1). In dieser kann er der App mitteilen, dass das Objekt auf dem Bild gefunden wurde (2). Daraufhin ruft die App die eigene Position ab (2.1), liest die GPS-Koordinaten aus dem Bild aus (2.2) und berechnet den Abstand zwischen diesen. Ist der Abstand gering genug, wird das Objekt als gefunden gewertet (3), ist dies nicht der Fall, wird das Bild weiterhin als nicht gefunden gewertet (4). Jederzeit kann die Anzeige des Bildes geschlossen werden (5).

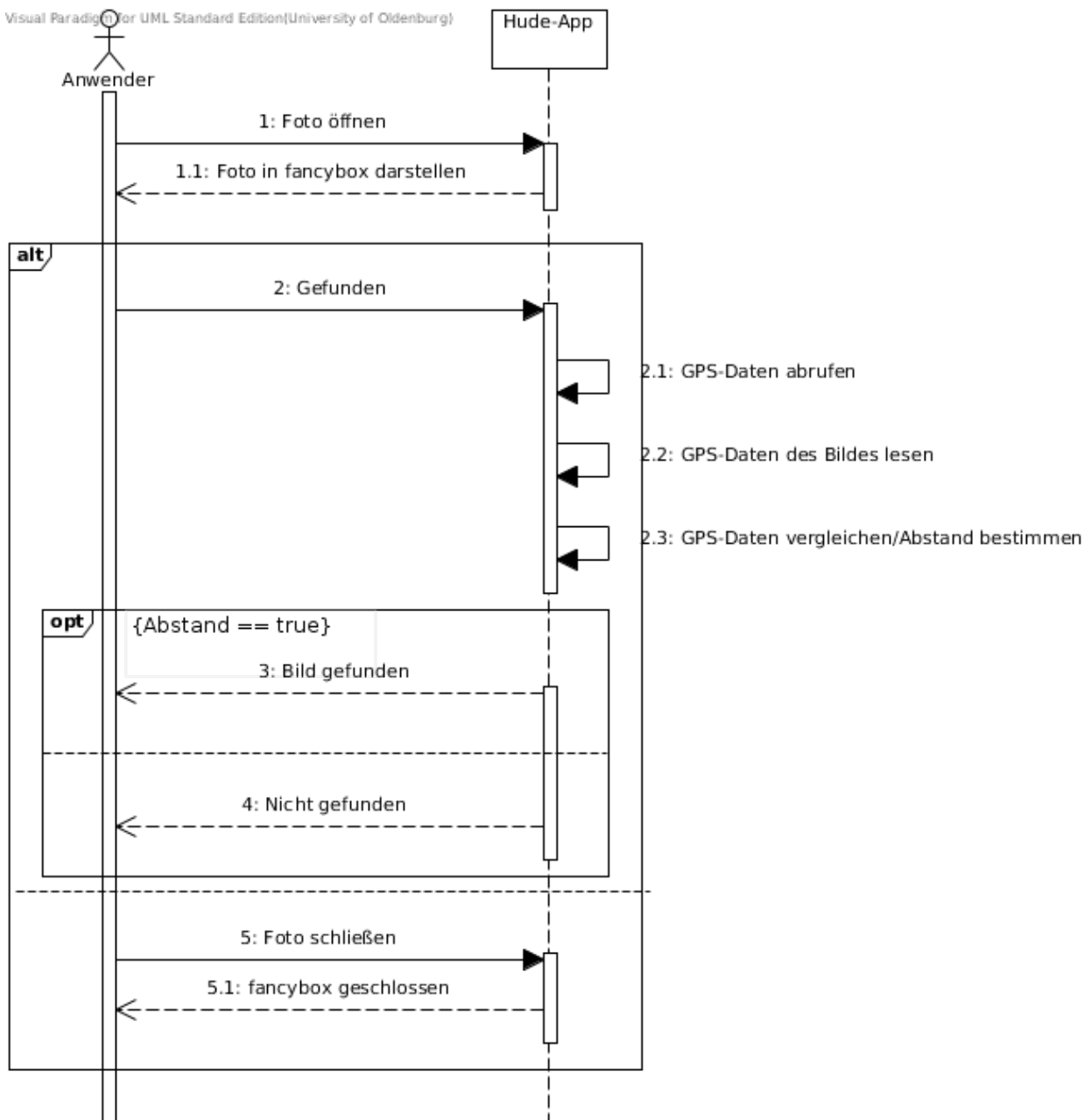


Abbildung 5.10.: Sequenzdiagramm: Fotojagd

Das Planetenabitur wird durch das Sequenzdiagramm in Abbildung 5.11 auf Seite 40 beschrieben. Der Anwender beantwortet zunächst die Fragen (1). Anschließend werden die Antworten an die App zur Auswertung übergeben (2). Das Ergebnis der Auswertung wird dem Anwender mitgeteilt (2.1).

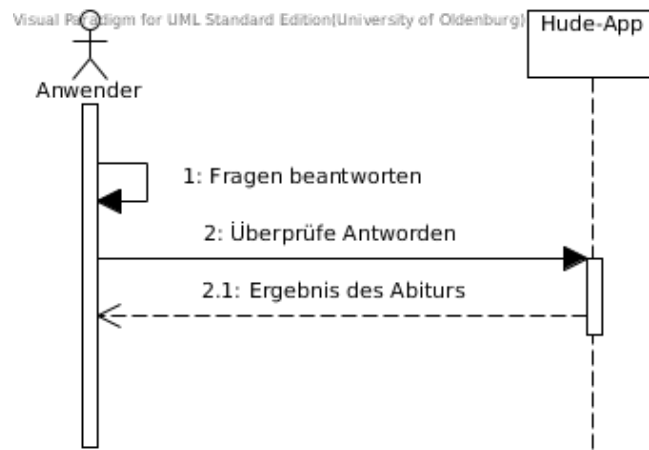


Abbildung 5.11.: Sequenzdiagramm: Planetenabitur

5.5. Prototypen

Zur Entwicklung der App hat ein evolutionärer Prototyp beigetragen. Im ersten Schritt wurden zunächst Apps aus anderen Städten, Orten und Gemeinden auf ihre Funktionalität hin untersucht. Dabei stellte sich heraus, dass die Funktionalität meist auf die Auflistung von Einrichtungen, wie z.B. Restaurants und Krankenhäuser, sowie die Anzeige von Veranstaltungen vor Ort begrenzt ist. Eine App allein hätte zwar bereits dem Ziel des Tourismus-Entwicklungskonzept entsprochen, innovative/neue Medien zu nutzen[TPH11], doch bereits in den gemeinsamen Vorüberlegungen wurde eine größere, umfangreichere Funktionalität angestrebt, als sie in den soeben beschriebenen Praxisbeispielen vorlag.

5.5.1. Prototyp I

Zum ersten Treffen mit der Touristik-Platte-Hude am 12. April 2013, wurde ein konzeptioneller Papier-Prototyp erstellt. Dieser sollte das mögliche Aussehen und mögliche Funktionalitäten bereits darstellen. Aus den Vorüberlegungen ging hervor, dass die App einen ständig sichtbaren Header-Bereich (Kopfbereich) haben soll, der die aktuelle Überschrift der aufgerufenen Ansicht enthält, einen Button der einen QR-Code-Scanner startet, sowie einen Button der ein Seitenmenu (Panel) zur Navigation in der App öffnet. In diesem Panel sollten zunächst nur die Navigationspunkte der Startansicht eingebettet sein, so dass zur Navigation kein Sprung zur Startansicht notwendig ist. In einem Footer-Bereich (Fußbereich) sollten Buttons angeordnet werden, die einem mit einem Klick die Startansicht, die Kontaktdaten zur Touristik-Palette-Hude, das Impressum oder den aktuellen Standort auf einer Karte aufrufen. In der Startansicht sollten in vier Spalten diverse Icons zur Navigation eingebunden werden, um in tiefere Ebenen der App zu navigieren. Dazu zählen zum Beispiel „Sehenswürdigkeiten“ oder „Hotels“.



Abbildung 5.12.: Prototyp I: Startansicht



Abbildung 5.13.: Prototyp I: Offenes Panel

Die weiterführenden Ansichten, wie Abbildung 5.14 auf Seite 41 zeigt, sollten einen gleichbleibenden Aufbau haben. Dies sollte eine Interaktion mit der App erleichtern. So sollten der Header- und Footer-Bereich weiterhin immer angezeigt werden. Die Übersicht für folgende Inhalte sollte zunächst eine Überschrift haben, die passend die nächste Ansicht beschreibt. Unter der Überschrift ist zusätzlich eine Grafik eingebunden, die die folgende Ansicht beschreibt. Darunter wiederum ist ein kurzer Einführungstext nachlesbar, der den Nutzer neugierig auf die folgende Ansicht machen soll, sowie ein Button, der die nächste Ansicht öffnet.



Abbildung 5.14.: Prototyp I: Übersicht von Sehenswürdigkeiten

5. Entwurf

Die Detaildarstellung einer Ansicht sollte wieder den Header- und Footer-Bereich enthalten. Um einen Wiedererkennungswert zu schaffen, sollte ebenfalls die Überschrift und die Grafik aus der Übersichtsseite angezeigt werden. Unter diesen beiden Elementen sollte ein Player eingerichtet sein, mittels dessen der Text der Seite als Audio-Datei abgespielt werden kann. Unter dem Audio-Player sollte sich der Text mit Informationen befinden. Unterhalb des Textes sollte ein Video-Player integriert sein. Dieser sollte, sofern vorhanden, ein Video zur aktuellen Ansicht enthalten. Unter dem Video-Player, ebenfalls sofern vorhanden, sollte sich eine Galerie mit Bildern, passend zur Ansicht, befinden.



Abbildung 5.15.: Prototyp I: Oberer Teil der Detailansicht
Abbildung 5.16.: Prototyp I: Unterer Teil der Detailansicht

Über den Button im Header-Bereich sollte ein QR-Code-Scanner geöffnet werden. Würde ein QR-Code der Touristik-Palette-Hude gescannt werden, so sollte die App den dazugehörigen Inhalt öffnen. Diese QR-Codes könnten z.B. an Sehenswürdigkeiten angebracht werden können, um ohne Navigation innerhalb der App, die Informationen über die jeweilige Sehenswürdigkeit abrufen zu können.

Durch einen Klick auf den Button „Standort“ im Footer-Bereich sollte dem Nutzer eine Karte angezeigt werden, auf dem sein aktueller Standort zur eigenen Orientierung vermerkt ist.

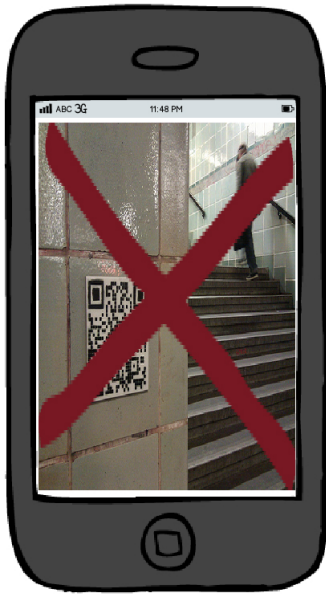


Abbildung 5.17.: Prototyp I: QR-Code-Scanner



Abbildung 5.18.: Prototyp I: Karte mit aktuellen Standort

5.5.2. Prototyp II

Anhand der gesammelten Ideen, Vorschläge und Wünsche, die sich während des ersten Treffens am 12. April 2013 in Hude herausstellten, wurde zum nächsten Treffen am 19. April 2013 ein erster klickbarer Prototyp entwickelt. Dieser enthielt zunächst die Startansicht mit Icons, die Beispiel-Seite „Hude-Info“ mit Klappmenüs, die Beispielseite „Klosterruine“, die Übersichtsseite „Sehenswürdigkeiten“ und das Seiten-Panel. Außerdem befand sich das „Impressum“ als Pop-Up im Prototyp.



Abbildung 5.19.: Prototyp II: Startansicht mit Icons

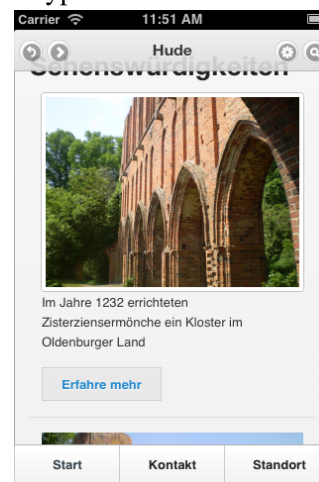


Abbildung 5.20.: Prototyp II: Übersicht Sehenswürdigkeiten

Die Abbildung 5.5.2 auf Seite 43 zeigt die Startansicht der App. Über Icons aus selbsterklärenden Piktogrammen sollte eine Navigation zu den unterschiedlichen Bereichen ermöglicht werden. Um einen Überblick über Inhalte eines Bereichs zu erhalten, sollte es Ansichten zur Übersicht geben. Abbildung 5.5.2 auf Seite 43 zeigt diese Übersicht anhand des Beispiels „Sehenswürdigkeiten“.

Zur Repräsentation von Informationen in den Ansichten, wurden zwei Optionen im Prototypen implemen-

5. Entwurf

tiert. Die eine Möglichkeit bestand darin, Informationen sowie die Funktionalität (Audio-Player, Video-Player, Text etc.) auf der Ansicht gut arrangiert untereinander anzuzeigen, so wie es Abbildung 5.5.2 und Abbildung 5.5.2 auf Seite 44 zeigt.

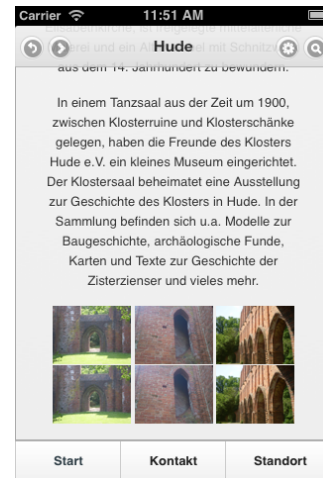
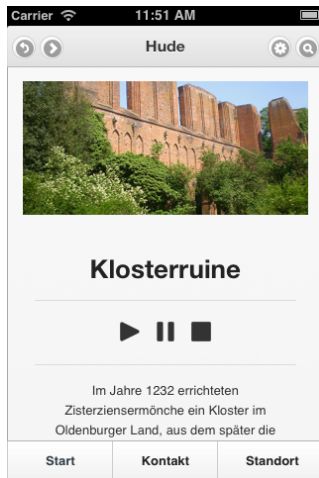


Abbildung 5.21.: Prototyp II: Ohne Klappmenüs Abbildung 5.22.: Prototyp II: Ohne Klappmenüs II

Die andere Möglichkeit, so zeigen es Abbildung 5.5.2 und Abbildung 5.5.2 auf Seite 44, besteht darin, die Funktionalitäten und Informationen in Klappmenüs zu verankern, so dass eine Übersicht über alle Inhalte gegeben ist.

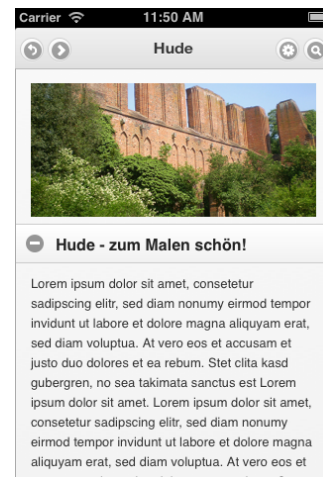
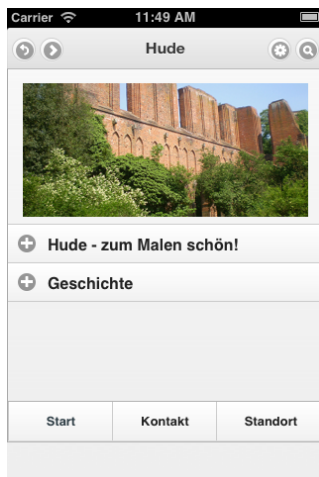


Abbildung 5.23.: Prototyp II: Klappmenüs geschlossen Abbildung 5.24.: Prototyp II: Klappmenü geöffnet

Durch das Panel, welches in jeder Ansicht über den Header-Bereich zu öffnen sein sollte, war eine Navigation innerhalb der App möglich. Innerhalb des Panels befanden sich die Navigationspunkte aus der Startansicht (siehe Abbildung 5.5.2 auf Seite 45).

Um schnell einen Kontakt zu einem Ansprechpartner in Hude herstellen zu können, wurde bereits ein Pop-Up (siehe Abbildung 5.5.2 auf Seite 45) eingebaut, das die Kontaktdaten der Touristik-Palette-Hude enthielt.

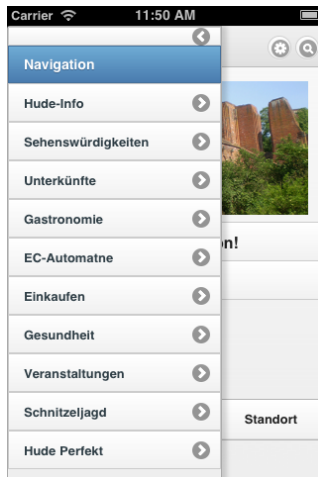


Abbildung 5.25.: Prototyp II: Panel

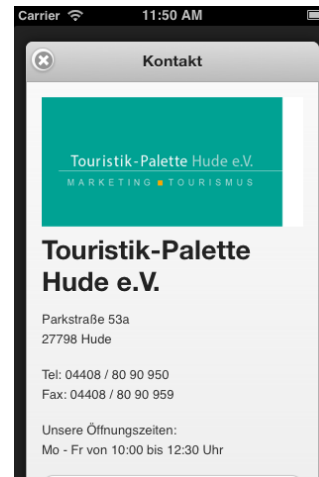


Abbildung 5.26.: Prototyp II: Pop-Up Kontakt

5.5.3. Prototyp III

Zum nächsten Treffen bei der Touristik-Palette-Hude (8. Mai 2013), wurde der Prototyp um Inhalte und Design weiterentwickelt, sowie den Wünschen der Touristik-Palette-Hude angepasst. Zunächst erfolgte eine Anpassung an das Corporate Design, wie es in Abbildung 5.5.3 auf Seite 45 deutlich wird. Durch den Beschluss beim vergangenen Treffen wurden die Inhalte und die Funktionalitäten in Klappmenüs integriert und ein Zurück-Button auf den Ansichten implementiert, der die zuvor angezeigte Ansicht aufruft (siehe Abbildung 5.5.3 auf Seite 45). Seiten, die der Übersicht dienen, behielten ihre Überschrift, ein Bild, sowie einen kurzen Einleitungstext.

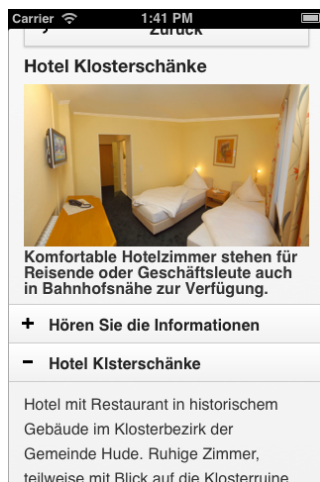


Abbildung 5.27.: Prototyp III: Inhalte mit Klappmenüs

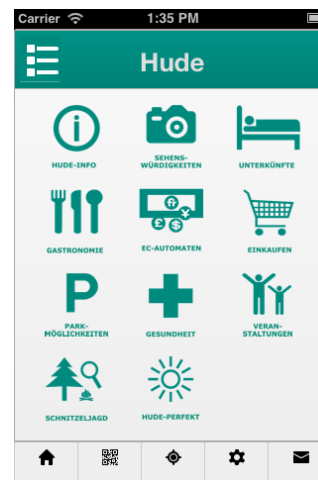


Abbildung 5.28.: Prototyp III: Anpassung an das Corporate Design

5.5.4. Prototyp IV

Für das nächste Treffen (30. Mai 2013) wurde der Prototyp um Inhalte erweitert und im Design angepasst. Im Header-Bereich wechselte die Überschrift nun passend zur angezeigten Ansicht und blieb nicht wie bisher bei der statischen Überschrift „Hude“. Ebenso wurde im Footer-Bereich die Toolbar angepasst. Die Icons wurden im Vergleich zum vorangegangenen Prototypen zentriert. Außerdem wurde die Übersichtsansicht in der Form angepasst, dass die graue Trennlinie entfernt und der kleine Button gegen einen klick-freundlicheren ausgetauscht wurde (siehe Abbildung 5.5.4 auf Seite 46).

Die bereits bestehenden Inhalte „Haus Kimmen“ und „Goldart“ wurden entfernt. Neu hinzu kamen die Inhalte „Klosterbezirk“, „Hasbruch und „Wittemoor“, „Skulpturenufer und -haus“, „Galerie am Stall“ und „Vielstedter Bauernhaus“. Der Inhalt „Klosterruine“ wurde dem neu angelegten „Klosterbezirk“ zugeordnet, sowie die neuen Inhalte „Klosterschänke & Remise“, „Wassermühle“, „Torkapelle“, und „Museum“ aufgenommen. Erreichbar sind diese Unterinhalte nun über Links in Listviews im übergeordneten Bereich innerhalb der Klappmenüs. Die einzelnen Elemente der Listviews erhielten Grafiken, die zur nächsten Ansicht passen (siehe Abbildung 5.5.4 auf Seite 46).

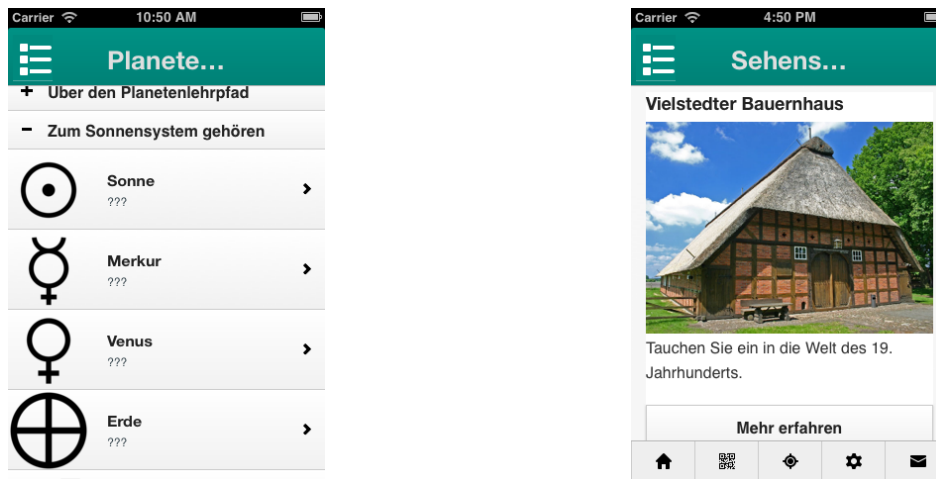


Abbildung 5.29.: Prototyp IV: Listviews innerhalb des Klappmenüs

Abbildung 5.30.: Prototyp IV: Anpassung von Überschrift, Footer und Übersichten

5.5.5. Prototyp V

Zum nächsten Treffen am 4. Juli 2013 in Hude wurde der Prototyp um zusätzliche Inhalte und um Funktionalitäten erweitert. Im Bereich der Funktionalitäten kamen die Zielgruppeneinstellungen, die Spracheinstellungen für Audio Inhalte (Abbildung 5.31 auf Seite 47), die Implementierung des QR-Code-Scanners, die aktuelle Positionsbestimmung und die damit verbundene Anzeige dieser Position auf einer Karte, die Auswertung (Abbildung 5.5.5 auf Seite 47) der drei Planetenabitur-Rätsel und die Fehlerkennzeichnung (Abbildung 5.5.5 auf Seite 47), sowie die ersten Zugriffe auf das Dateisystem hinzu. Zu letzterem gehörten das Erstellen, Auslesen und Löschen von Ordnern.

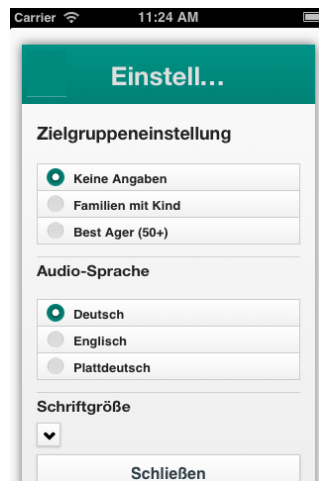


Abbildung 5.31.: Prototyp V: Neue Einstellungen

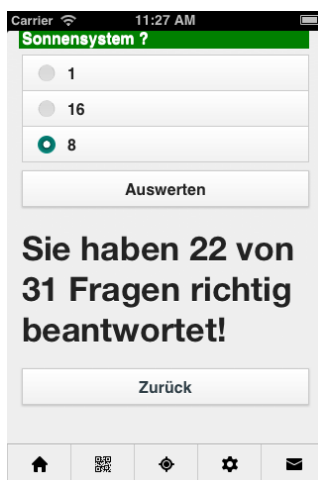


Abbildung 5.32.: Prototyp V: Auswertung des Planetenabiturs

Abbildung 5.33.: Prototyp V: Fehlerkennzeichnung

Die Erweiterung der Inhalte erfolgte in den Bereichen „Sehenswürdigkeiten“, „Unterkünfte“, „Gastronomie“, „EC-Automaten“, „Einkaufen“, „Gesundheit“ und „Schnitzeljagd“.

Ebenso wurden in den vorhandenen Listviews (Abbildung 5.5.5 auf Seite 48) die Bilder und Grafiken so angepasst, dass diese nun optisch mit dem klickbaren Bereich abschlossen (Abbildung 5.5.5 auf Seite 48).

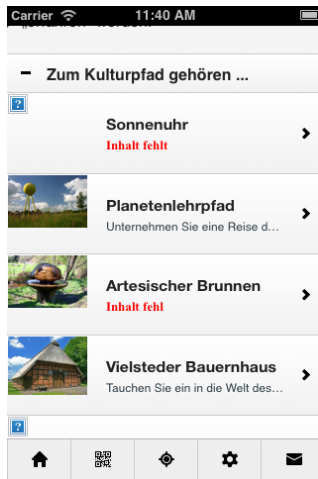


Abbildung 5.34.: Prototyp V: Alte Listviews

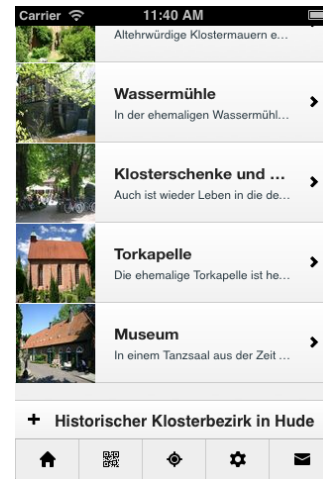


Abbildung 5.35.: Prototyp V: Neue Listviews

5.5.6. Prototyp VI

Zum nächsten Treffen in Hude am 1. August 2013 wurde der Prototyp wieder um Inhalte und Funktionalitäten erweitert. Zu diesem Zeitpunkt waren bereits über 95% der Inhalte und Funktionen implementiert, so dass dieser Prototyp - nach einer Revision der Inhalte - die Freigabe zur Veröffentlichung in den App Stores hätte erhalten können.

Die Funktionalität wurde um den Download von Audio-Dateien erweitert. Somit konnten nun die Inhalte in den unterschiedlichen Ansichten in der eingestellten Sprache wiedergegeben werden. Ebenso wurden Kartenausschnitte, mit Angabe von GPS-Koordinaten, online abgerufen und den Inhalten und Kontaktdaten hinzugefügt. Mit dieser Implementierung war nun eine Orientierung möglich, wo sich z.B. ein gesuchter Gastronomiebetrieb befindet. Weiterhin haben die Kontakt-Daten einen „Anrufen“-Button erhalten, der die Telefon-App des Telefon startet, so dass ein direkter Anruf ermöglicht wurde. Neben der Schnitzeljagd „Planetenabitur A“, „Planetenabitur B“ und „Planetenabitur C“, wurde ein weiteres Spiel implementiert - die „Fotojagd“. Bei diesem Spiel sollte der Nutzer den Standort eines Objektes auf einem Foto wiedererkennen und diesen aufsuchen. Hat sich der Nutzer dem Objekt auf eine vordefinierte Distanz genähert, so wurde das Bild nach Drücken des Buttons „Gefunden“ als gefunden gewertet. Zunächst waren die Bereiche „Klosterbezirk“ und „Hude“ als Spielflächen implementiert.

Parkmöglichkeiten wurden, wenn eine ausreichende Datenverbindung bestand, auf einer Google-Maps-Karte dargestellt. Bei einer nicht ausreichenden Datenverbindung wurden die Parkplätze in einer Liste mit einer kurzen Beschreibung dargestellt.

5.5.7. Prototyp VII

Da alle Prototypen im Simulator präsentiert wurden und nur kurze Funktions-Tests auf einem Hardware-Gerät durchgeführt wurden, wurde erst beim Treffen am 1. August 2013 festgestellt, dass das bisher genutzte Framework „jQuery Mobile“ sich zu inperformant verhält. Dieses inperformante Verhalten wurde selbst durch diverse Optimierungen nicht eliminiert. Aus diesem Grund musste der bisher bestehende Prototyp, auf Basis eines neuen Frameworks („Foundation Zurb“), umgebaut werden.

Durch den Wechsel des Frameworks verlor die App ihr bisheriges Design (Prototyp I - VI), welches sich stark am iOS-Design orientierte (Abbildung 5.5.7 auf Seite 49). Die App erhielt mit dem neuen Prototypen ein eigenes, innovatives Design.



Abbildung 5.36.: Prototyp VII: Foundation Zurb

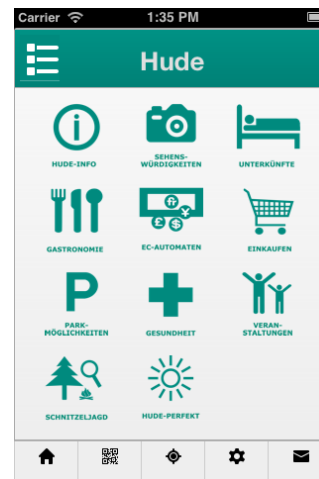


Abbildung 5.37.: Prototyp I-VI:jQuery Mobile

Inhaltlich wurde unter „Hude-Perfekt“ der „Huder Padd“ hinzugefügt. Dieser wurde durch einen kurzen Text beschrieben und per Google Maps angezeigt. Auf dem Weg wurden Sehenswürdigkeiten auf dem Padd als einzelne Punkte markiert (Abbildung 5.5.7 auf Seite 49).

Inhaltlich und funktionell wurden darüber hinaus die Veranstaltungen implementiert. Diese wurden vom Server heruntergeladen und in die Ansicht dynamisch eingefügt. Entfernt wurde die Funktion, dynamisch Google-Maps-Bilder in die Ansichten einzufügen. Bei einer schlechten Datenverbindung würde es zu lange dauern, bis diese angezeigt würden. Aus diesem Grund wurden die Kartenausschnitte erzeugt, ins Dateisystem integriert und fest eingebunden (Abbildung 5.5.7 auf Seite 49).

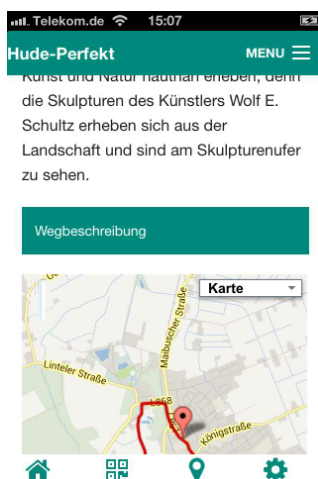


Abbildung 5.38.: Prototyp VII:Hude-Perfekt

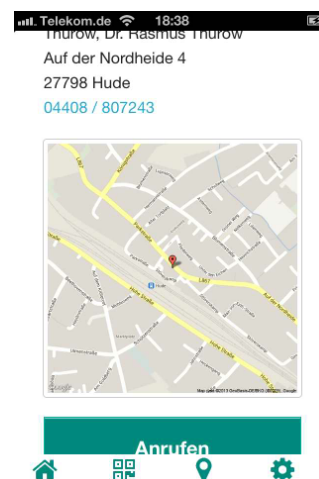


Abbildung 5.39.: Prototyp VII: Karte im Kontakt

Das Seitenpanel wurde von der linken Seite nach oben umgebaut. In diesem befanden sich nun nicht nur die Navigationsmöglichkeiten der Startansicht (Abbildung 5.5.7 auf Seite 50), sondern ebenfalls die Möglichkeit zur direkten Navigation auf die unterste Ebene in den Ansichten (Abbildung 5.5.7 auf Seite 50).

5. Entwurf

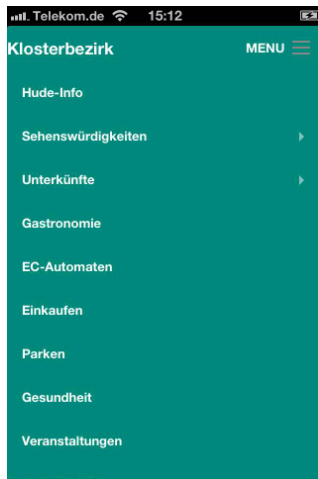


Abbildung 5.40.: Prototyp VII: Panel-Ebene I

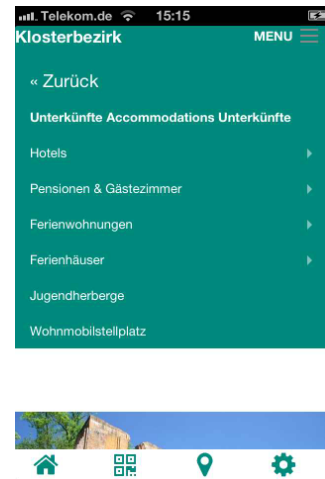


Abbildung 5.41.: Prototyp VII: Panel-Ebene II

Funktionell wurde darüber hinaus die Einstellung der Text-Sprache implementiert, so dass verschiedene Sprachen für Text und Audio eingestellt werden konnten. In gleicher Weise wurde auch die Einstellung für die Schriftgröße innerhalb der App implementiert.

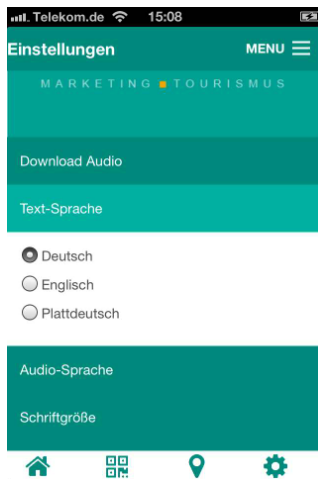


Abbildung 5.42.: Prototyp VII: Einstellungen Sprache

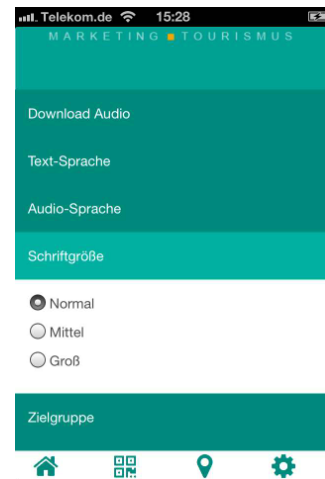


Abbildung 5.43.: Prototyp VII: Einstellungen Schriftgroesse

6. Implementierung

Im folgenden Kapitel werden die zur Implementierung notwendigen Schritte und Vorgehensweisen erklärt. Nach einem Überblick über Frameworks und Installationsvarianten wird die Implementierung der App in allen Einzelschritten dargestellt.

6.1. Frameworks

Im Abschnitt „Frameworks“ wird herausgestellt, welche Frameworks genutzt werden und warum ebendiese Verwendung in der App erhalten.

6.1.1. jQuery

jQuery ist ein von John Resig programmiertes JavaScript-Framework, das erstmalig 2006 vorgestellt wurde und mittlerweile in der Version 2.0.0¹ verfügbar ist. Dieses Framework stellt Funktionen zur einfachen DOM-Manipulation sowie Navigation bereit. Im Vergleich zum regulären JavaScript erfährt der Entwickler viel Unterstützung bei der Programmierung. So gibt es Funktionen zur Behandlung von AJAX, dem Event-Handling und Animationen.

In dieser Arbeit wird nicht die aktuellste Version (2.0.0) des Frameworks verwendet, sondern auf die Vorgänger-Version 1.9.1 zurückgegriffen, da die meisten Plugins inkompatibel zur neusten Version sind.

Wie jQuery-Funktionen in der App eingesetzt wurden, wird beginnend mit Abschnitt 6.3 genauer erläutert.

Plugins

fancybox FancyBox ist ein Plugin auf Basis von jQuery, das die Darstellung von Grafiken, Texten oder anderen multimedialen Inhalten auf einer Seite innerhalb einer sogenannten „Lightbox“ ermöglicht. Diese Lightbox enthält einen gesetzten Inhalt, der über den bereits angezeigten Inhalt platziert wird. Dabei verhält sich die geöffnete Lightbox responsive², so dass eine optimale Darstellung der Inhalte ermöglicht wird. Durch fancyBox wird ebenso ein Galerverhalten ermöglicht, so dass nicht für jedes einzelne Bild eine neue Lightbox geöffnet wird, sondern die Inhalte durch Vor- und Zurück-Buttons ausgetauscht werden können[an]. Eine ausführliche Dokumentation und eine Vielzahl an Beispielen ist auf der Entwicklerseite³ einzusehen. Wie fancyBox für diese App eingesetzt wurde, wird in den Abschnitten 6.3.3 und 6.3.3 genauer erläutert.

Google maps v3 plugin for jQuery and jQuery Mobile Zur Anzeige von Kartenmaterial wird auf Google Maps zurückgegriffen und kein Kartenmaterial auf dem Smartphone installiert und aufgerufen. Um jedoch nicht jegliche API-Aufrufe mittels HTTP-Request senden und empfangen zu müssen, wie es das Google-Maps-API⁴ beschreibt, wird das „Google-maps-v3-plugin for jQuery and jQuery Mobile“ verwendet. Zu diesem Plugin existiert online eine umfangreiche Dokumentation mit Beispielen⁵.

¹Stand: 18.05.2013

²Als „responsives“-Verhalten wird die automatische Anpassung an die Displaygröße des genutzten Gerätes und die möglichst optimale Skalierung der auf dem Display dargestellten Objekte bezeichnet.

³<http://fancyapps.com/fancybox/>

⁴<https://developers.google.com/maps/documentation/javascript/reference?hl=de> (letzter Abruf: 21. August 2013)

⁵<https://code.google.com/p/jquery-ui-map/>

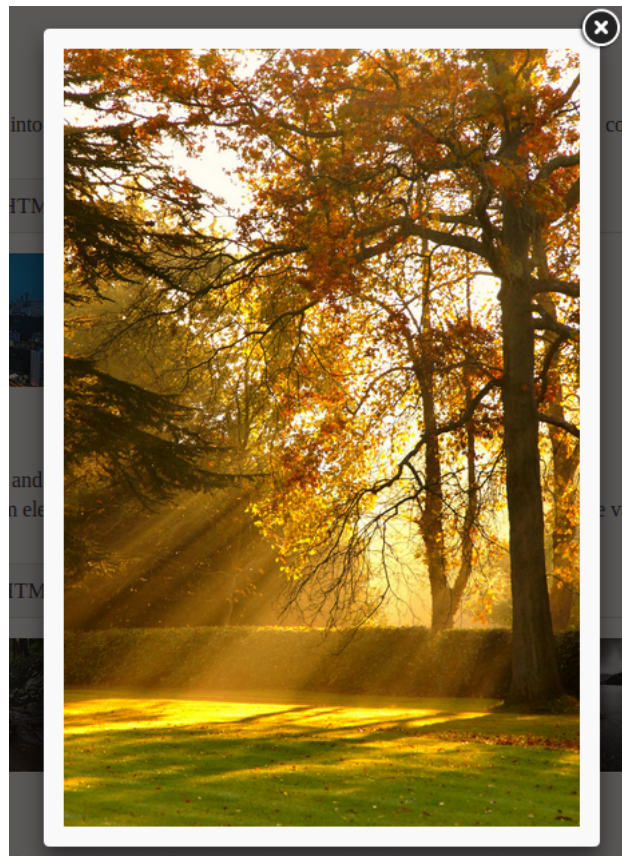


Abbildung 6.1.: Geöffnete fancyBox[fan]

Nach dem Einbinden der benötigten JS-Skripte ist mit wenigen zu setzenden Parametern eine Karte innerhalb eines `<div>`-Elements, mit einer gesetzten ID oder Klasse, anzeigbar. Wie dies geschieht wird in Abschnitt 6.3.3 detaillierter erläutert.

jQuery hashchange Das „jQuery hashchange“-Plugin ist das kleinste eingesetzte Erweiterungs-Plugin für jQuery. Die Aufgabe des Plugin besteht darin, den Wechsel des Hashes in der Domain zu registrieren und dementsprechend zu reagieren [Ben10]. Der Hash ist Teil einer URI⁶, wird durch das Doppelkreuz # gekennzeichnet und ist die darauf folgende Zeichenkette. Die URI: „`http://www.meinedomain.de/index.html#ich`“, trägt den Hash „ich“. Wie die Reaktion des Plugins gestaltet ist, wird im Abschnitt 6.3.3 beschrieben.

Exif Reader Exif repräsentiert das „Exchangeable Image File Format“ und ist ein Format, um Metadaten in digitalen Bildern abzuspeichern. Ein Großteil der Metadaten wird bei der Aufnahme in den Header der Bilddatei geschrieben. Mögliche Metadaten sind beispielsweise das Modell der Kamera, Zoomeinstellungen, Ausrichtung, Aufnahmedatum, Belichtungszeit [Wik13a] und GPS-Daten.

Das Exif-Reader-Plugin besteht aus zwei Plugin-Dateien, die zwei unterschiedliche Aufgaben bewältigen. Zum einen die „binaryajax.js“, die den Zugriff auf die Binär-Daten des Bildes ermöglicht, folglich die binären Exif-Metadaten aus dem Header des Bildes ausliest. Auf der anderen Seite die „exif.js“, die die ausgelesenen Binärdaten in lesbare Exif-Daten umwandelt [Sei08]. Wie und in welcher Form das Plugin genutzt wird, wird in Abschnitt 6.3.3 genauer erläutert.

⁶Uniform Resource Identifier (URI) ist eine Zeichenfolge, die ein Objekt im Internet eindeutig identifiziert. Bei Domains (Aufruf von Webseiten) spricht man vom Uniform Resource Locator (URL).

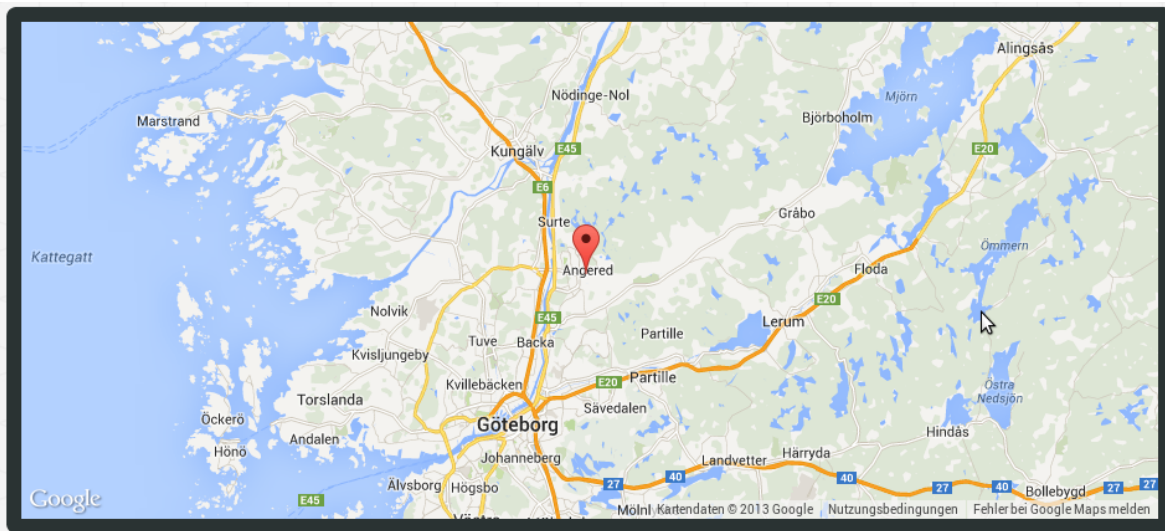


Abbildung 6.2.: Google-Maps-Karte - eingebunden per Plugin[joh12]

6.1.2. Foundation

ZURB⁷ beschreibt sein Foundation-Framework selbst mit den Worten „the most advanced responsive front-end framework in the world“[ZUR]. Foundation ist ein Framework zur schnellen Entwicklung von Benutzeroberflächen im Bereich des Webdesigns, welches ein stark responsives-Verhalten aufweist. Neben einem 12-Spalten-Design, das sich der Auflösung des Gerätes anpasst, bringt Foundation bereits eine Vielzahl an vordefinierten Elementen, wie Navigation, Buttons, Tabs oder Formularen mit. So wird ein schnelles Design der Oberfläche ermöglicht, welches sich anschließend - durch Veränderung der CSS - leicht an die jeweiligen Gegebenheiten anpassen lässt.

Neben vordefinierten Styles bringt Foundation ebenso eine Vielzahl an Plugins mit. Einige von diesen werden eingesetzt, um die Interaktion des Anwenders mit der App zu vereinfachen. Welches Plugin wo und wie eingesetzt wird, wird beginnend mit Abschnitt 6.3 genauer erläutert.

6.1.3. Phonegap

Um nicht für jedes Smartphone-Betriebssystem eine eigene App programmieren zu müssen, wird ein Framework eingesetzt, mit dem es möglich ist, parallel und mit wenigen Änderungen für jedes Betriebssystem eine App zu portieren (Hybrid App). Dies wird erst dadurch möglich, dass PhoneGap den WebView-Container, der auf jedem modernen Smartphone vorhanden ist, und die dazugehörige WebKit-Rendering-Engine nutzt. Der WebView-Container ist im Grunde ein Vollbildbrowser ohne Adresszeile. Aus diesem Grund wird die Benutzeroberfläche der App mit HTML5 sowie CSS gestaltet und die Logik durch JavaScript realisiert. Mittels eines API ist ein Zugriff auf die Hardware des Gerätes möglich, die über das JavaScript angesprochen wird. So können zum Beispiel die Kamera, der Speicher oder die Bewegungssensoren genutzt werden. Anwendungen, die allerdings leistungs- bzw. grafiklastig sind (zum Beispiel Spiele), stoßen durch die Beschränkung des Browsers an ihre Grenzen und sollten somit nicht mit PhoneGap realisiert werden.

⁷Zurb ist eine Produkt-Design-Unternehmung aus Campbell, California USA[ZUR]. <http://zurb.com/>

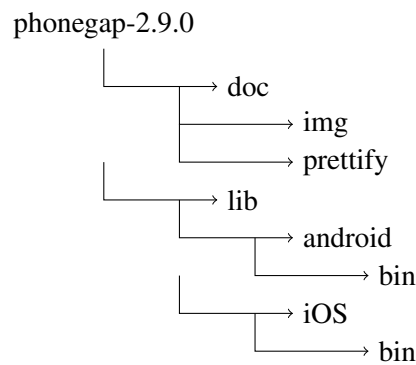


Abbildung 6.3.: Benötigte Pfade des PhoneGap-Frameworks zum Erzeugen von Projekten einzelner Apps

Unterstützte Geräte-Hardware

Die folgende Tabelle zeigt die für jedes Smartphone-Betriebssystem von PhoneGap unterstützte Geräte-Hardware, die über das API angesteuert werden kann.

	Android	Bada	iOS	Tizen	webOS	Windows Phone 7/8
Bewegungssensor	✓	✓	✓	✓	✓	✓
Kamera	✓	✓	✓	✓	✓	✓
Kompass	✓	✓	✓	✓	✓	✓
Kontakte	✓	✓	✓	✗	✓	✓
Daten	✓	✗	✓	✗	✓	✓
Geolocation	✓	✓	✓	✓	✓	✓
Medien	✓	✗	✓	✓	✓	✗
Netzwerk	✓	✓	✓	✓	✓	✓
Notification (Alarm)	✓	✓	✓	✓	✓	✓
Notification (Vibration)	✓	✓	✓	✓	✓	✓
Speicher	✓	✗	✓	✗	✓	✗

Abbildung 6.4.: Übersicht über die unterstützte Geräte-Hardware in PhoneGap [Ado]

Um bei der Verwendung der App keine Fehler hervorzurufen, wenn es darum geht die Hardware aus Abbildung 6.4 auf Seite 54 zu nutzen, ist eine „Device Ready“-Funktion in das PhoneGap-API implementiert. Erst wenn diese meldet, dass das Smartphone und die Hardware bereit sind, ist ein Zugriff auf die Geräte-Hardware möglich.

Plugins

Um die App um zusätzliche Funktionalitäten zu erweitern, werden Plugins eingesetzt. Diese sind bereits programmierte, eigenständige Anwendungen, die in die App eingebunden werden können und ihre Funktionalität an diese weitergeben. In dieser App wird das Plugin „BarcodeScanner“ eingebunden, um später mittels QR-Codes in der App navigieren zu können und direkt zu Inhalten zu gelangen.

BarcodeScanner Das BarcodeScanner-Plugin ist eines von 16 Plugins[pho], das über PhoneGap Build (siehe dazu Abschnitt 6.2.2) nutzbar ist. Dieses Plugin befindet sich in der Version 0.6.0⁸ und ist mit den Betriebssystemen iOS und Android und der PhoneGap-Version 2.0.0 (oder einer neueren Version) nutzbar [Kan].

Um einen Code zu auszulesen, werden für beide Betriebssysteme (iOS & Android) folgende Barcode-Formate unterstützt [Wil]:

- QR_CODE
- DATA_MATRIX
- UPC_E
- UPC_A
- EAN_8
- EAN_13
- CODE_128
- CODE_39
- ITF

Für den Einsatz in der App selbst ist jedoch nur das Format QR_CODE relevant. Wie diese Funktionalität eingesetzt wird, wird in Abschnitt 6.3.3 genauer erläutert.

Ebenso ist es möglich, mittels BarcodeScanner-Plugins QR-Codes zu erzeugen. Zur Erzeugung werden die folgenden Typen unterstützt [Wil]:

- TEXT_TYPE
- EMAIL_TYPE
- PHONE_TYPE
- SMS_TYPE

Da diese Funktionalität jedoch nicht genutzt wird, ist die genauere Betrachtung an dieser Stelle nicht notwendig. Der Entwickler [Wil] gibt eine detaillierte Beschreibung zum Einsatz dieser Funktionalität.

6.2. Installation

Zur Erzeugung einer App mit dem PhoneGap-Framework gibt es zwei Varianten. Die eine Variante besteht darin, das SDK des jeweiligen mobilen Betriebssystems zu nutzen, für das die App erzeugt werden soll, und die PhoneGap-Bibliotheken einzubinden. Die andere Variante besteht darin, PhoneGap Build zu nutzen. PhoneGap Build bezeichnet einen Webservice von PhoneGap, um plattformunabhängig für jedes Smartphone-Betriebssystem eine App zu erzeugen. Für beide Varianten wird im Vorfeld eine Web-App mittels Webtechnologien erstellt. Wie beide Varianten eingerichtet und genutzt werden können, wird in den Abschnitten 6.2.1 und 6.2.2 genauer erläutert.

⁸Stand: 23. August 2013

6.2.1. SDK

Um PhoneGap innerhalb der dem Betriebssystem eigenen SDKs nutzen zu können, muss für jedes Betriebssystem ein Projekt mittels des „Command Line Tools“, welches in jeden Download des PhoneGap-Frameworks integriert ist, erzeugt werden. Im „Command Line Tool“ müssen dazu die folgenden drei Parameter gesetzt werden:

- Ort:** Dies ist der Ort, an dem das Projekt und die dazugehörigen Dateien abgespeichert werden.
- Paketname:** Der Paketname ist der eindeutig identifizierende Bezeichner für das Projekt im App Store. Damit dieser einmalig ist, wird dazu eine Domain, die dem Entwickler gehört, rückwärts genutzt. Der Paketname muss im Apple „Provisioning Portal“ des „iOS Developer Account“ angegeben sein.
- Projektname:** Der Projektname entspricht dem Namen der App.

Grafik 6.5 zeigt das Vorgehen zur Erzeugung der Projekte für Android und iOS.

```
1      # Navigation zum Command Line Tool für iOS
      cd Downloads/phonegap-2.9.0/lib/iOS/bin
3      # Erzeugung des Projektes für iOS
      # ./create Ort Paketname Projektname
5      ./create /Users/ImpY/Documents/Hude/iOS de.gastinhude.m Hude
      # Navigation zum Command Line Tool für Android
7      cd
      cd Downloads/phonegap-2.9.0/lib/android/bin
9      # Erzeugung des Projektes für Android
      # ./create Ort Paketname Projektname
11     ./create /Users/ImpY/Documents/Hude/android de.gastinhude.m Hude
```

Abbildung 6.5.: Erzeugung der Projekte für die SDKs

Nach dem Erzeugen der Projekte können diese in die entsprechenden SDKs importiert und geöffnet werden. Führt man direkt nach dem Erzeugen den Simulator (Xcode) bzw. den Emulator (Android Development Tool) aus, wird die PhoneGap-Beispiel-App gestartet. Um die entwickelte Web App als native App zu starten, müssen die Daten in die jeweiligen „www“-Ordner des Projekts kopiert werden. Im Android-Projekt ist der „www“-Ordner im Ordner „assets“ zu finden. Im Xcode-Projekt befindet sich dieser im Root-Verzeichnis.

Durch das Starten des Simulators bzw. Emulators werden jeweils die Apps erzeugt. Für iOS ist dies die „Hude.ipa“ und für Android die „Hude.apk“. Diese Dateien sind im jeweiligen Projekt-Ordner zu finden und müssen an den entsprechenden App Store übermittelt werden, damit die App über den App Store bezogen werden kann.

6.2.2. Build PhoneGap

Neben der lokalen Entwicklung über die SDKs (Android Developer Tools, Xcode usw.) der einzelnen mobilen Betriebssysteme, besteht die Möglichkeit über einen Build-Service auf der PhoneGap-Website eine App für alle mobilen Betriebssysteme zu erzeugen. Um diesen Build-Service nutzen zu können ist das Anlegen eines Benutzeraccounts über die Website⁹ notwendig. Alternativ können auch die Benutzeraccounts

⁹<https://build.phonegap.com/plans> (letzter Abruf: 19.05.2013)

von Adobe¹⁰ oder GitHub¹¹ zur Erstellung von Apps genutzt werden. Für eine einzelne App ist dieser Service kostenlos.

Für die Hude-App wurde ein GitHub-Account angelegt, so dass der Build-Service auf das Git¹²-Repository und damit auf die Daten der App zugreifen kann und daraus die Apps erzeugen kann. GitHub bietet auch einen weiteren, besonderen Vorteil. So befinden sich alle, zur App gehörigen Dateien in einem Repository. Von dort werden sie an den PhoneGap-Build-Service übertragen. Die Größe aller Dateien zusammengekommen darf zur Übertragung an PhoneGap Build mehr als 15 MB betragen. Nutzt man den PhoneGap-Build-Service ohne GitHub und übermittelt die Daten durch eine ZIP-Datei über den Web-Upload an PhoneGap Build, darf dieses Datenpaket nicht größer als 15 MB sein.

In das Git-Repository müssen die Dateien der Web-App gepushed werden, einschließlich einer „config.xml“. In dieser Datei werden die Spezifikationen und Metadaten - die App betreffend - gesetzt. Diese sind Standards der W3C [W3C12] und werden von PhoneGap-Build zusätzlich erweitert, um Plugins (wie zum Beispiel einen Barcodescanner) einbinden und nutzen zu können. Bei der Erstellung ist zu beachten, dass es notwendige und optionale Entitäten gibt[Ado13]. Zu den notwendigen Entitäten und deren Attributen gehören folgende:

widget Der <widget>-Tag ist die Root-Entität. Der Tag trägt die folgenden notwendigen Attribute:

id ist der eindeutig identifizierende Bezeichner der App und sollte der rückwärts gelesenen Domain des Entwicklers entsprechen.

version beschreibt die Versionsnummer und muss zu jedem Update im App Store inkrementiert werden.

name trägt den Namen der App, wie er auch im App Store angezeigt wird.

description trägt die Beschreibung der App wie sie im App Store angezeigt wird.

Die für die Hude-App erstellte „config.xml“ zeigt die Abbildung 6.6 auf Seite 59. Weitere Konfigurationsmöglichkeiten werden in der PhoneGap-Dokumentation¹³ gezeigt.

Nach dem Abrufen des aktuellen Standes des Git-Repository wird durch den PhoneGap Build-Service eine App für alle mobilen Betriebssysteme erstellt. Abgesehen von der App für iOS-Systeme funktioniert dieses direkt. Damit PhoneGap Build iOS-Apps erzeugen kann, müssen zusätzliche Zertifikate hinzugefügt werden. Das Anlegen der Zertifikate wird im Anhang behandelt.

```

1  <?xml version="1.0" encoding="UTF-8" ?>
  <widget
3    xmlns="http://www.w3.org/ns/widgets"
    xmlns:gap="http://phonegap.com/ns/1.0"
5    id="de.gastinhude.m"
    version="1.0.0">
7    <name>Hude</name>
    <description>
9      Informationen immer dabei - für Ihren Besuch in Hude und Umgebung!
    </description>
11   <author href="http://touristik-palette-hude.de"
        email="info@touristik-palette-hude.de">
        Touristik-Palette Hude e.V.
13   </author>
    <!-- Allgemein -->
15   <preference name="phonegap-version" value="2.9.0" />

```

¹⁰<https://build.phonegap.com/plans/free-adobeid> (letzter Abruf: 14.08.2013)

¹¹<https://github.com/signup/free> (letzter Abruf: 14.08.2013)

¹²Git ist eine Software zur Versionsverwaltung unterschiedlicher Dokumente. Innerhalb eines sogenannten Repositories werden die, durch die Modifizierung unterschiedlichen, Versionen von Dateien gespeichert. Mitnutzer eines Git erfahren so ebenfalls die Änderungen und können Veränderungen nachverfolgen oder andere Versionen wiederherstellen.

¹³<https://build.phonegap.com/docs/config-xml> (letzter Abruf: 14.08.2013)

6. Implementierung

```
17 <preference name="orientation" value="default" />
18 <preference name="fullscreen" value="false" />
19 <preference name="webviewbounce" value="true" />
20 <preference name="load-url-timeout" value="30000" />
21 <preference name="splash-screen-duration" value="5000" />
22
23 <icon src="images/system/icon/icon_android_36px_ldpi.png" gap:platform="android"
24     gap:density="ldpi" />
25 <icon src="images/system/icon/icon_android_48px_mdpi.png" gap:platform="android"
26     gap:density="mdpi" />
27 <icon src="images/system/icon/icon_android_72px_hdpi.png" gap:platform="android"
28     gap:density="hdpi" />
29 <icon src="images/system/icon/icon_android_96px_xhdpi.png" gap:platform="android"
30     gap:density="xhdpi" />
31
32 <gap:splash src="images/system/splash/splash_android_720x1280px_portrait_xhdpi.png"
33     gap:platform="android" gap:density="xhdpi" />
34 <gap:splash src="images/system/splash/splash_android_480x800px_portrait_hdpi.png"
35     gap:platform="android" gap:density="hdpi" />
36 <gap:splash src="images/system/splash/splash_android_320x480px_portrait_mdpi.png"
37     gap:platform="android" gap:density="mdpi" />
38 <gap:splash src="images/system/splash/splash_android_200x320px_portrait_ldpi.png"
39     gap:platform="android" gap:density="ldpi" />
40 <gap:splash src="images/system/splash/splash_android_1280x720px_landscape_xhdpi.png"
41     gap:platform="android" gap:density="xhdpi" width="1280" height="720" />
42 <gap:splash src="images/system/splash/splash_android_800x480px_landscape_hdpi.png"
43     gap:platform="android" gap:density="hdpi" width="800" height="480" />
44 <gap:splash src="images/system/splash/splash_android_480x320px_landscape_mdpi.png"
45     gap:platform="android" gap:density="mdpi" width="480" height="320" />
46 <gap:splash src="images/system/splash/splash_android_320x200px_landscape_ldpi.png"
47     gap:platform="android" gap:density="ldpi" width="320" height="200" />
48
49 <preference name="android-minSdkVersion" value="11" />
50 <preference name="android-maxSdkVersion" value="17" />
51
52 <!-- iOS -->
53 <icon src="images/system/icon/icon_ios_57px_iphone.png" gap:platform="ios" width="57"
54     height="57" /> -->
55 <icon src="images/system/icon/icon_ios_72px_ipad.png" gap:platform="ios" width="72"
56     height="72" />
57 <icon src="images/system/icon/icon_ios_114px_iphone4_retina.png" gap:platform="ios"
58     width="114" height="114" />
59 <icon src="images/system/icon/icon_ios_144px_ipad3_retina.png" gap:platform="ios"
60     width="144" height="144" />
61
62 <gap:splash src="images/system/splash/splash_ios_320x480px_iphone_portrait.png"
63     gap:platform="ios" width="320" height="480" />
64 <gap:splash
65     src="images/system/splash/splash_ios_640x960px_iphone_retina_portrait.png"
66     gap:platform="ios" width="640" height="960" />
67 <gap:splash
68     src="images/system/splash/splash_ios_640x1136px_iphone_2x_retina_portrait.png"
69     gap:platform="ios" width="640" height="1136" />
70 <gap:splash src="images/system/splash/splash_ios_1024x748px_ipad_landscape.png"
71     gap:platform="ios" width="1024" height="768" />
72 <gap:splash src="images/system/splash/splash_ios_768x1004px_ipad_portrait.png"
73     gap:platform="ios" width="768" height="1024" />
74 <gap:splash
75     src="images/system/splash/splash_ios_2048x1496px_ipad_2x_retina_landscape.png"
76     gap:platform="ios" width="2048" height="1496" />
77 <gap:splash
78     src="images/system/splash/splash_ios_1536x2048px_ipad_2x_retina_portrait.png"
79     gap:platform="ios" width="1536" height="2048" />
80
81 <preference name="webviewbounce" value="true" />
82 <preference name="stay-in-webview" value="false" />
83 <preference name="detect-data-types" value="true" />
84 <preference name="exit-on-suspend" value="false" />
85
86 <!-- Windows Phone
87 <icon src="icons/winphone/icon.png" gap:platform="winphone" />
88
```



```

        <icon src="icons/winphone/tileicon.png" gap:platform="winphone" gap:role="background"
        />
61 <gap:splash src="winphoneSplash.jpg" gap:platform="winphone" />
    -->
63 <!-- Features -->
    <feature name="http://api.phonegap.com/1.0/camera"/>
65 <feature name="http://api.phonegap.com/1.0/file"/>
    <feature name="http://api.phonegap.com/1.0/geolocation"/>
67 <feature name="http://api.phonegap.com/1.0/media"/>
    <feature name="http://api.phonegap.com/1.0/network" />
69 <!-- Plugins -->
    <gap:plugin name="com.phonegap.plugins.barcodescanner" />
71 </widget>

```

Abbildung 6.6.: PhoneGap-Build config.xml für Hude

6.3. App-Entwicklung

In diesem Abschnitt werden die Schritte beschrieben, die notwendig waren, um die App zu realisieren - begonnen beim Design, über den Aufbau der Seiten bis hin zur Implementierung der einzelnen Funktionalitäten.

6.3.1. Design

Das Design der App wurde durch die Kombination von Grafiken und CSS realisiert, so dass eine identische Darstellung des Designs auf jedem Smartphone gesichert ist.

Grafiken

Die verwendeten Grafiken der Startseite sind Vektor-Grafiken und stammen von „all-free-download.com“¹⁴. Die Erstellung der Icons wurde mit Adobe Illustrator durchgeführt. Dabei wurden unter anderem auch einzelne Vektor-Grafiken zu einem neuen Icon kombiniert. Die Steuerelemente des Audio-Players stammen aus den Vektor-Grafiken von „roamdesign.co.uk“¹⁵ und wurden dem Design angepasst. Zur Anordnung der Grafiken wurde das Raster, das das Foundation-Framework (siehe Abschnitt 6.1.2 auf Seite 53) bietet, genutzt. Dieses Raster verhält sich „responsive“ und skaliert automatisch die enthaltenen Elemente. Auf der Startseite wird ein Raster der Größe 3x4 verwendet. Im Audio-Player findet ein 1x3-Raster Anwendung. Zur Darstellung der Bilder, z.B. einer Sehenswürdigkeit, wurde ein Raster mit zwei Spalten und Zeilen entsprechend der Anzahl der Bilder gewählt.

Die Startgrafik, sowie das Icon (Abbildung 6.7) für den Homescreen des Smartphones, wurden aus dem Logo der Touristik-Palette-Hude erstellt. Für jede Ausrichtung und Displaygröße, Portrait oder Landscape, bei Android (ldpi, mdpi, hdpi und xhdp) und iOS (normal, retina und 2xretina) wurde dazu eine eigene Startgrafik und ein eigenes Icon erstellt.

Das Bildmaterial über den Inhalten stammt von der Website¹⁶ der Touristik-Palette-Hude und findet dort in den unterschiedlichen Bereichen Verwendung.

CSS

Damit das verwendete Foundation-Framework (Abschnitt 6.1.2 auf Seite 53) und dessen Elemente dem Corporate Design der Touristik-Palette-Hude entsprechen, mussten einige Veränderungen an den CSS-Einstellungen vorgenommen werden. Hauptsächlich mussten das Header-Element und das darin integrierte

¹⁴http://de.all-free-download.com/frei-vektoren/vektoren-ikone/free_vector_icons_piktogramm_145355.html (letzter Abruf: 5. September 2013)

¹⁵<http://blog.roamdesign.co.uk/?p=272> (letzter Abruf: 5. September 2013)

¹⁶<http://www.gastinhude.de>



Abbildung 6.7.: Icon und Startgrafik

Panel-Element angepasst werden (Abbildung 6.8), sowie die Einstellungen für die Klappmenüs (Abbildung 6.9). Die Anpassung für das Header- und Panel-Element bestand in der Anpassung der Farbdefinitionen des Hintergrundes und der Überschrift.

```
1 .top-bar {  
    background-color: #008A7E;  
3 }  
    .top-bar-section ul {  
5     background-color: #008A7E;  
    }  
7 .top-bar-section li a:not(.button) {  
    background-color: #008A7E;  
9 }  
    .top-bar-section ul li.hover > a {  
11    background: none repeat scroll 0 0 #008A7E;  
    color: white;  
13    font-weight: 500;  
    }  
15 .top-bar.expanded .toggle-topbar a {  
    color: #fff;  
17 }
```

Abbildung 6.8.: Anpassung der CSS für den Header und das Panel

Bei der Anpassung der Klappmenüs (Abbildung 6.9) wurden Anpassungen an die Farbdefinition des Hintergrundes, des Rahmens und der Schriftart vorgenommen. Ebenso mussten die Abmaße angepasst werden, so dass die gesamte Bildschirmbreite ausgenutzt wird.

```
1 .section-container.auto > section > .title a, .section-container.auto > .section >  
    .title a, .section-container.vertical-tabs > section > .title a,  
    .section-container.vertical-tabs > .section > .title a,  
    .section-container.vertical-nav > section > .title a,  
    .section-container.vertical-nav > .section > .title a,  
    .section-container.horizontal-nav > section > .title a,  
    .section-container.horizontal-nav > .section > .title a,  
    .section-container.accordion > section > .title a, .section-container.accordion >  
    .section > .title a {  
    color: #fff;  
3 }
```

```

5  .section-container.auto > section > .title, .section-container.auto > .section > .title,
   .section-container.vertical-tabs > section > .title,
   .section-container.vertical-tabs > .section > .title,
   .section-container.vertical-nav > section > .title, .section-container.vertical-nav
   > .section > .title, .section-container.horizontal-nav > section > .title,
   .section-container.horizontal-nav > .section > .title, .section-container.accordion
   > section > .title, .section-container.accordion > .section > .title {
6  background-color: #008a7e;
7  cursor: pointer;
8  border-top-width: 0px;
9  border-right-width-value: 0px;
   border-bottom-width: 0px;
10 border-left-width-value: 0px;
11 border-top-style: solid;
12 border-right-style-value: solid;
13 border-bottom-style: solid;
14 border-left-style-value: solid;
15 border-top-color: #008a7e;
16 border-right-color-value: #008a7e;
17 border-bottom-color: #008a7e;
18 border-left-color-value: #008a7e;
19 border-image-source: none;
20 border-image-slice: 100% 100% 100% 100%;
21 border-image-width: 1 1 1 1;
22 border-image-outset: 0 0 0 0;
23 border-image-repeat: stretch stretch;
24 }
25
26
27 .section-container.auto > section.active > .title, .section-container.auto >
   .section.active > .title, .section-container.vertical-tabs > section.active >
   .title, .section-container.vertical-tabs > .section.active > .title,
   .section-container.vertical-nav > section.active > .title,
   .section-container.vertical-nav > .section.active > .title,
   .section-container.horizontal-nav > section.active > .title,
   .section-container.horizontal-nav > .section.active > .title,
   .section-container.accordion > section.active > .title, .section-container.accordion
   > .section.active > .title {
28 background-color: #02b0a2;
29 }
30
31 .section-container.auto > section > .title:hover, .section-container.auto > .section >
   .title:hover, .section-container.vertical-tabs > section > .title:hover,
   .section-container.vertical-tabs > .section > .title:hover,
   .section-container.vertical-nav > section > .title:hover,
   .section-container.vertical-nav > .section > .title:hover,
   .section-container.horizontal-nav > section > .title:hover,
   .section-container.horizontal-nav > .section > .title:hover,
   .section-container.accordion > section > .title:hover, .section-container.accordion
   > .section > .title:hover {
32 background-color: #02b0a2;
33 }
34
35 container.horizontal-nav, .section-container.accordion {
36 border-top: 0px solid #CCCCCC;
37 }
38
39 .section-container.auto > section > .content, .section-container.auto > .section >
   .content, .section-container.vertical-tabs > section > .content,
   .section-container.vertical-tabs > .section > .content,
   .section-container.vertical-nav > section > .content,
   .section-container.vertical-nav > .section > .content,
   .section-container.horizontal-nav > section > .content,
   .section-container.horizontal-nav > .section > .content,
   .section-container.accordion > section > .content, .section-container.accordion >
   .section > .content {
40 background-color: white;
41 border: 0px solid #CCCCCC;
42 padding: 0.9375em;
43 }

```

6. Implementierung

```
45 .section-container.auto > section.active > .title a, .section-container.auto >
    .section.active > .title a, .section-container.vertical-tabs > section.active >
    .title a, .section-container.vertical-tabs > .section.active > .title a,
    .section-container.vertical-nav > section.active > .title a,
    .section-container.vertical-nav > .section.active > .title a,
    .section-container.horizontal-nav > section.active > .title a,
    .section-container.horizontal-nav > .section.active > .title a,
    .section-container.accordion > section.active > .title a,
    .section-container.accordion > .section.active > .title a {
    color: #fff;
47 }

49 .section-container.auto > section > .title, .section-container.auto > .section > .title,
    .section-container.vertical-tabs > section > .title,
    .section-container.vertical-tabs > .section > .title,
    .section-container.vertical-nav > section > .title, .section-container.vertical-nav
    > .section > .title, .section-container.horizontal-nav > section > .title,
    .section-container.horizontal-nav > .section > .title, .section-container.accordion
    > section > .title, .section-container.accordion > .section > .title {
    background-color: #008A7E;
51    border: 0 solid #008A7E;
    cursor: pointer;
53 }
```

Abbildung 6.9.: Anpassung der CSS für die Klappmenüs

6.3.2. Seiten

Zur Darstellung der unterschiedlichen Inhalte werden diverse HTML-Seiten eingesetzt. Der Aufbau der darstellenden Seiten setzt sich aus zwei Elementen zusammen. Zum einen ist es die Hauptseite (Seite 62), zum anderen die Inhaltsseiten (Seite 65). Wie die Symbiose beider Seiten ermöglicht wird, wird im Folgenden genauer erläutert.

Hauptseite

Die Hauptseite ist das Gerüst für die gesamte App und ist in HTML realisiert. In ihr werden alle benötigten Framework-, Plugin-, und Logikdateien eingebunden, so dass ein Zugriff auf diese und deren Funktionalität von jeder Inhaltsseite möglich ist. In ihr befindet sich ein DIV-Container, in den der Inhalt der anzuzeigenden Inhaltsseite geladen wird.

Innerhalb des <head>-Tags (Abbildung 6.10) werden eine Vielzahl an Einbindungen zur Darstellung und Verarbeitung von allen Daten vorgenommen. Dabei enthalten die Zeilen 8-11 alle CSS-Dateien, die die Darstellung der Daten beeinflussen. In Zeile 11 werden die CSS-Definitionen eingebunden, die das Aussehen der App dem Corporate Design der Touristik-Palette-Hude geben.

In den Zeilen 12-30 werden alle benötigten Logik-Dateien eingebunden. Die Zeilen 13 und 14 binden die Logik des PhoneGap-Frameworks (Abschnitt 6.1.3) ein, sowie das dazugehörige BarcodeScanner-Plugin (Abschnitt 6.1.3). Zeile 16 bindet das Foundation-Framework ein (Abschnitt 6.1.2) und Zeile 18 die Logik zur Darstellung von Kartenmaterial mittels Google Maps. Die Zeilen 20-27 binden das jQuery-Framework (Abschnitt 6.1.1) sowie die dazugehörigen Plugins ein.

Das in Zeile 29 eingebundene JS enthält die entwickelte Logik für die App. Die implementierten Funktionen sind in Abschnitt 5.3 auf Seite 32 zu sehen und werden im Abschnitt 6.3.3 genauer erläutert.

Zur Unterstützung bei der Entwicklung und späteren Weiterentwicklung wurde das JS in Zeile 30 geschrieben und eingebunden. Dieses Script gibt Variablen direkt in einer lesbaren Form aus, so dass z.B. zur Anzeige von Arrays keine Schleifen durchlaufen werden müssen.

Das Skript zwischen den Zeilen 31 und 40 ist abhängig von zwei Faktoren. Zum einen ist die Einbindung des PhoneGap-Frameworks notwendig (Zeile 13), zum anderen ist Voraussetzung, dass die App als native App auf einem Smartphone gestartet wird. Erst dadurch kann der Event-Listener (Zeile 32) das Event

abfangen, dass das Gerät bereit ist, und die Ausführung des Codes ab Zeile 34 einleiten. Der Event-Listener in Zeile 36 ist speziell für Smartphones mit einem Hardware-Zurück-Button. Wird dieser gedrückt, gelangt der Nutzer bei jedem getätigten Druck um eine Ansicht zurück.

Wird die App per Browser aufgerufen, wird dieser Code ignoriert, kann aber weiterhin genutzt werden.

```

1  <!DOCTYPE html>
2  <!--[if IE 8]> <html class="no-js lt-ie9" lang="en" > <![endif]-->
3  <!--[if gt IE 8]><!--> <html class="no-js" lang="en" > <!--<![endif]-->
4  <head>
5      <meta charset="utf-8">
6      <meta name="viewport" content="width=device-width">
7      <!-- Cascading Stylesheet Definitionen -->
8      <link rel="stylesheet" href="css/foundation.css">
9      <link rel="stylesheet" href="css/jquery.fancybox.css">
10     <link rel="stylesheet" href="css/jquery.fancybox-buttons.css">
11     <link rel="stylesheet" href="css/hude.css">
12     <!-- Phonegap Abhängigkeiten -->
13     <script src="phonegap.js"></script>
14     <script src="barcodescanner.js"></script>
15     <!-- Foundation Framework Abhängigkeiten -->
16     <script src="js/vendor/custom.modernizr.js"></script>
17     <!-- Google Maps Abhängigkeit-->
18     <script src="http://maps.google.com/maps/api/js?sensor=true"
19         type="text/javascript"></script>
20     <!-- jQuery Framework und PlugIns -->
21     <script src="js/jquery-1.9.1.min.js"></script>
22     <script src="js/jquery.ui.map.full.min.js" type="text/javascript"></script>
23     <script src="js/jquery.ba-hashchange.min.js" type="text/javascript"></script>
24     <script src="js/jquery.exif.js" type="text/javascript"></script>
25     <script src="js/jquery.fancybox.js" type="text/javascript"></script>
26     <script src="js/jquery.fancybox-buttons.js" type="text/javascript"></script>
27     <script src="js/binaryajax.js"></script>
28     <script src="js/exif.js"></script>
29     <!-- Logik der App -->
30     <script src="js/hude.js"></script>
31     <script src="js/print_r.js"></script>
32     <script>
33         document.addEventListener("deviceready", onDeviceReady, false);
34         function onDeviceReady() {
35             tphAppStart();
36             // Zurück-Button auf Android-Geräten mit der Zurück-Funktion belgen
37             document.addEventListener("backbutton", function(event) {
38                 parent.history.back();
39             }, false);
40         }
41     </script>
42 </head>

```

Abbildung 6.10.: <head>-Bereich

Im <body>-Tag (Grafik 6.11) befinden sich vier Bereiche der App. Der erste Bereich ist der Header. In diesem befindet sich der Titel der aktuell angezeigten Seite (Zeile 6), sowie die Möglichkeit, das Navigations-Panel (Zeile 8-12) zu öffnen. Das Navigations-Panel wird über ungeordnete Listen () realisiert. Die Zeilen 14-41 geben dazu einen kleinen Ausschnitt und zeigen dabei ebenfalls die tiefe Verschachtelung (Zeile 18-35) innerhalb des Navigations-Panels. Im gegebenen Ausschnitt ist es möglich, über den Navigationspunkt „Unterkünfte“ (Zeile 20-22), „Hotels“ (Zeile 26) auszuwählen und im Anschluss eines der drei in Hude angesiedelten Hotels (Zeile 28-30) auszuwählen.

Der zweite Bereich ist der Content (Zeile 42). In diesem werden alle aufgerufenen Inhalte dargestellt. Wie dies ermöglicht wird, wird in Abschnitt ?? genauer erläutert.

Der dritte Bereich ist der Footer (Zeile 43-50). In diesem sind vier Buttons integriert, über die mit einem Klick die Startansicht, der QR-Code-Scanner, die aktuelle Position, sowie die Einstellungen aufgerufen werden können.

6. Implementierung

Unterhalb des Footers befindet sich der vierte Bereich (Zeile 51-91). In diesem werden zusätzliche Skripte eingebunden, u.a. Plugins, die das Foundation-Framework benötigt (Zeile 51-72), und aufgerufen.

```
1 <div id="tphHeader" class="fixed">
  <nav class="top-bar"
    data-options="custom_back_text:true;back_text:Zurück;is_hover:false;">
3 <ul class="title-area">
  <!-- Title Area -->
5 <li class="name">
  <h1 id="tphHeaderUeberschrift">Hude</h1>
7 </li>
  <li class="toggle-topbar menu-icon">
9 <a id="tphMenu" href="#">
  <span>Menu</span>
11 </a>
  </li>
13 </ul>
  <section class="top-bar-section">
15 <ul class="left">
  <li><a href="#tphSeiteHudeInfo" >Hude-Info</a></li>
17 [...]
  <li class="has-dropdown">
19 <a href="#tphSeiteUnterkuenfte">
  <span class="tphSpracheDE">Unterkünfte</span> <!-- auf deutsch -->
21 <span class="tphSpracheEN">Accommodations</span> <!-- auf englisch -->
  <span class="tphSprachePD">Unterkünfte</span> <!-- auf plattdeutsch -->
23 </a>
  <ul class="dropdown">
25 <li class="has-dropdown">
  <a>Hotels</a>
27 <ul class="dropdown">
  <li><a href="#tphSeiteUnterkuenfteHotelBurgdorf">Hotel Burgdorf</a></li>
29 <li><a href="#tphSeiteUnterkuenfteHotelGarni">Hotel Garni</a></li>
  <li><a href="#tphSeiteUnterkuenfteHotelKlosterschaenke">Hotel
    Klosterschänke</a></li>
31 </ul>
  </li>
33 [...]
  </ul>
  </li>
35 [...]
  <li><a href="#tphSeiteSpielplaetze">Spielplätze</a></li>
37 </ul>
39 </section>
  </nav>
41 </div>
  <div class="tphContent"></div>
43 <div class="tphFooter">
  <div class="row tphFooter">
45 <div class="large-3 small-3 columns"><a href="#tphDashboard"></a></div>
  <div class="large-3 small-3 columns"><a href="#" onclick="tphQRCodeScan();"></a></div>
47 <div class="large-3 small-3 columns"><a href="#tphSeitePosition"></a></div>
  <div class="large-3 small-3 columns"><a href="#tphSeiteEinstellungen"></a></div>
49 </div>
  </div>
51 <script>
  document.write('<script src=' +
53 ('__proto__' in {}) ? 'js/vendor/zepto' : 'js/vendor/jquery') +
  '.js></script>')
55 </script>
  <script src="js/foundation.min.js"></script>
57 <script src="js/foundation/foundation.js"></script>
  <script src="js/foundation/foundation.alerts.js"></script>
59 <script src="js/foundation/foundation.clearing.js"></script>
  <script src="js/foundation/foundation.cookie.js"></script>
61 <script src="js/foundation/foundation.dropdown.js"></script>
```

```

<script src="js/foundation/foundation.forms.js"></script>
63 <script src="js/foundation/foundation.joyride.js"></script>
<script src="js/foundation/foundation.magellan.js"></script>
65 <script src="js/foundation/foundation.orbit.js"></script>
<script src="js/foundation/foundation.reveal.js"></script>
67 <script src="js/foundation/foundation.section.js"></script>
<script src="js/foundation/foundation.tooltips.js"></script>
69 <script src="js/foundation/foundation.topbar.js"></script>
<script src="js/foundation/foundation.interchange.js"></script>
71 <script src="js/foundation/foundation.placeholder.js"></script>
<script src="js/foundation/foundation.abide.js"></script>
73 <script>
  $(document).foundation();
75  $(function() {
    tphAppStart();
77    $(window).hashchange(function() {
      var hash = location.hash;
79      if (hash !== '#tphSeiteIndex') {
        hash = (hash.replace(/^#/ , \grqq{ }) || 'blank') + '.';
81        var datei = hash + 'html';
        if (hash !== 'blank.') {
83          tphAudioStoppen();
        }
85        $(' .tphContent').load(datei);
      }
87    });
    $(window).hashchange();
89    tphSetzeEinstellungenAufSeite();
  });
91 </script>

```

Abbildung 6.11.: <body>-Bereich

Inhaltsseiten

Für die anzuzeigenden Inhalte wurden jeweils einzelne Seiten angelegt. Ihre Dateinamen wurden nach einem einheitlichen Muster vergeben. Alle Seiten tragen „tphSeite“ als führendes Element im Namen. Darauf folgend wird entweder die übergeordnete Kategorie der Seite angegeben, sollte diese existieren (z.B. „Sehenswürdigkeiten“), oder der Name der Seite (z.B. „Einstellungen“). Ist die aufzurufende Seite einer weiteren Kategorie untergeordnet (z.B. „Planetenlehrpfad“ als Unterkategorie der „Sehenswürdigkeiten“), so werden alle Kategorien ebenfalls angegeben. Kategorien können auch aufgerufen werden und geben, wie im Folgenden dargestellt, einen Überblick über die enthaltenen Seiten.

Kategorien und deren Unterkategorien sind:

- Sehenswürdigkeiten
 - Klosterbezirk
 - Kulturpfad
 - Planetenlehrpfad
- Unterkünfte
 - Hotels
 - Pensionen & Gästezimmer
 - Ferienwohnungen
 - Ferienhäuser

6. Implementierung

- Schnitzeljagd
 - Planetenabitur
 - Fotojagd

Durch das Foundation-Framework (Abschnitt 6.1.2) wird bereits die Darstellung der Inhalte für die jeweilige Größe des Displays optimiert. Eine weitere Optimierung erfolgt durch die Nutzung von Klappmenüs. Innerhalb dieser ist der Inhalt zunächst ausgeblendet, erst durch Klicken auf die Überschrift wird er eingeblendet. Durch das Klicken auf eine andere Überschrift wird der bereits sichtbare Inhalt wieder ausgeblendet und der neue Inhalte eingeblendet.

```
1 <div class="section-container accordion" data-section="accordion">
  <section>
3     <p class="title" data-section-title>
      <a href="#">Allgemeinmedizin</a>
5     </p>
      <div class="content" data-section-content>
7         <p>Inhalt Allgemeinmedizin ...</p>
      </div>
9  </section>
  <section>
11     <p class="title" data-section-title>
      <a href="#">Apotheken</a>
13     </p>
      <div class="content" data-section-content>
15         <p>
17             <div class="section-container accordion" data-section="accordion">
                <section>
19                     <p class="title" data-section-title>
                        <a href="#">Alte Raben-Apotheke</a>
21                     </p>
                        <div class="content" data-section-content>
23                             <h4>
                                <span class="tphSpracheDE">Kontakt</span> <!-- auf deutsch -->
                                <span class="tphSpracheEN">Contact</span> <!-- auf englisch -->
                                <span class="tphSprachePD">Kontakt</span> <!-- auf plattdeutsch -->
25                             </h4>
                                <p>
27                                     Apotheke am Schützenlatz<br/>
29                                     Inh. Birgit Halfter<br/>
                                        Schützenstraße 10<br/>
31                                     27798 Hude<br/>
                                        04408 / 92110<br/>
33                                 </p>
                                <p>
35                                     <a class="th radius tphGalerie"
                                        href="images/gesundheit/tphKarteSchuetzenlatz.png">
                                        
37                                     </a>
                                </p>
39                                 <p>
                                    <!-- Button der die Telefon-App aufruft -->
41                                     <a href="tel:0440892110" class="large button expand">
                                        <span class="tphSpracheDE">Anrufen</span> <!-- auf deutsch -->
43                                     <span class="tphSpracheEN">Call</span> <!-- auf englisch -->
                                        <span class="tphSprachePD">Anroepen</span> <!-- auf plattdeutsch -->
45                                     </a>
                                </p>
                                </div>
                </section>
            </div>
          </div>
        </div>
      <div class="content" data-section-content>
49         <p>
51             <a href="#">Flores Apotheke</a>
          </p>
          <div class="content" data-section-content>
53              <p>Inhalt Flores Apotheke</p>
          </div>
55         </div>
      </div>
  </div>
```



```

57     </section>
58     </div>
59     </p>
60     </div>
61 </section>
62 </div>

```

Abbildung 6.12.: Ausschnitt der Klappmenus aus der Gesundheits-Seite

Die Abbildung 6.12 zeigt die Implementierung zweier Klappmenus, „Allgemeinmedizin“ und „Apotheken“. Im Klappmenu „Apotheken“ sind drei weitere Klappmenus implementiert, die Informationen enthalten. Die Abbildung 6.13 zeigt das Resultat der Abbildung 6.12.

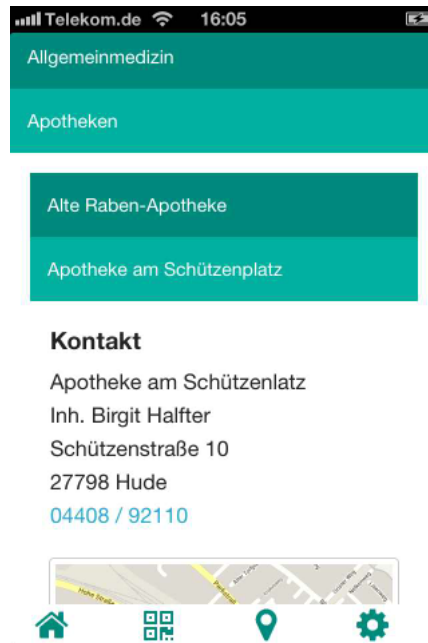


Abbildung 6.13.: Resultat des Codes aus der Abbildung 6.12

Die Inhalte der einzelnen Seiten stammen aus den vorhandenen Flyern der Touristik-Palette-Hude oder von deren Website¹⁷. Für die App wurden zur Darstellung der Inhalte folgende Seiten angelegt:

tphDashboard.html Das Dashboard enthält die Navigation mittels Icons auf der Startseite.

tphSeiteECAutomaten.html Die Seite enthält die EC-Automaten. Eingebettet in Klappmenus befinden sich Kontaktdaten, sowie ein Kartenausschnitt, der die Position des EC-Automaten angibt.

tphSeiteEinkaufen.html Die Seite enthält die Einkaufsmöglichkeiten. Eingebettet in Klappmenus befinden sich Kontaktdaten mit direkter Anruf-Funktion (mittels Button), sowie ein Kartenausschnitt, der die Position des EC-Automaten angibt.

tphSeiteEinstellungen.html Auf der Seite können einzelne Einstellungen gesetzt werden, die die Funktion der App beeinflussen. Dazu zählen:

- Download von Audio-Dateien
- Einstellen der Text-Sprache

¹⁷<http://gastinhude.de>

6. Implementierung

- Einstellen der Audio-Sprache
- Einstellen der Schriftgröße
- Einstellen der Zielgruppe

tphSeiteGastronomie.html Auf dieser Seite sind die unterschiedlichen gastronomischen Betriebe aufrufbar. Eingebettet in Klappmenüs befinden sich Kontaktdaten mit direkter Anruf-Funktion (mittels Button), Öffnungszeiten, sowie ein Kartenausschnitt, der die Position des Betriebes anzeigt.

tphSeiteGesundheit.html Auf dieser Seite sind die unterschiedlichen Einrichtungen, die Gesundheitsbelange betreffen, aufrufbar. Nach Fachgebieten geordnet befinden sich, in Klappmenüs eingebettet, Kontaktdaten mit direkter Anruf-Funktion (mittels Button), sowie ein Kartenausschnitt, der die Position der Einrichtung anzeigt.

tphSeiteHudelInfo.html Auf der Seite sind, in Klappmenüs eingebettet, ein Audio-Player (falls die Audio-Daten heruntergeladen wurden), ein Begrüßungstext und ein kurzer geschichtlicher Abriss Hude betreffend.

tphSeiteHudePerfekt.html Die Seite bietet, in Klappmenüs eingebettet, Vorschläge für den perfekten Tag oder die perfekten drei/sieben Tage in Hude.

tphSeiteIndex.html Diese Seite ist die Startseite und somit die Hauptseite (Abschnitt 6.3.2).

tphSeiteParkmoeglichkeiten.html Auf dieser Seite werden die Parkmöglichkeiten in Hude angezeigt - bei einer ausreichenden Datenverbindung auf einer Google-Maps-Karte, sonst auf Ausschnitten einer Google-Maps-Karte.

tphSeitePosition.html Bei Aufruf der Seite wird die aktuelle Position per GPS abgerufen und in einer Google-Maps-Karte angezeigt.

tphSeiteQRCodeFehler.html Diese Seite wird aufgerufen, falls ein Fehler beim Aufruf des BarcodeScanner-Plugin geschieht. Es erscheint eine entsprechende Fehlermeldung.

tphSeiteQRCodeUnguelstig.html Diese Seite wird aufgerufen, wenn mit der App ein QR-Code gescannt wird, der keinen URI mit entsprechendem Muster enthält.

tphSeiteSchnitzeljagdFotojagdHude.html Auf dieser Seite befinden sich die Fotos der Fotojagd „Hude“.

tphSeiteSchnitzeljagdFotojagdKlosterbezirk.html Auf dieser Seite befinden sich die Fotos der Fotojagd „Klosterbezirk“.

tphSeiteSchnitzeljagdFotojagdSkulpturenufer.html Auf dieser Seite befinden sich die Fotos der Fotojagd „Skulpturenufer“.

tphSeiteSchnitzeljagd.html Auf dieser Seite wird ein Überblick über die Spiele der „Fotojagd“ und des „Planetenabitur“ gegeben. Zudem können die Spiele aufgerufen werden.

tphSeiteSchnitzeljagdPlanetenabiturA.html Auf dieser Seite kann das Planetenabitur A durchgeführt werden.

tphSeiteSchnitzeljagdPlanetenabiturB.html Auf dieser Seite kann das Planetenabitur B durchgeführt werden.

tphSeiteSchnitzeljagdPlanetenabiturC.html Auf dieser Seite kann das Planetenabitur C durchgeführt werden.

tphSeiteSehenswuerdigkeitenArtesischerBrunnen.html Auf der Seite sind, in Klappmenüs eingebettet, ein Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über den Artesischen Brunnen und Bilder einsehbar.

tphSeiteSehenswuerdigkeitenHasbruch.html Auf der Seite sind, in Klappmenüs eingebettet, ein Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über den Hasbruch und Bilder einsehbar.

tphSeiteSehenswuerdigkeiten.html Diese Seite bietet einen Überblick über die unterschiedlichen Sehenswürdigkeiten in Hude.

tphSeiteSehenswuerdigkeitenKlosterbezirk.html Auf der Seite sind, in Klappmenüs eingebettet, ein Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über den Klosterbezirk, geschichtliche Hintergründe, eine Karte mit der Position und Bilder des Klosterbezirks einsehbar.

tphSeiteSehenswuerdigkeitenKlosterbezirkKlosterruine.html Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über die Klosterruine und Bilder zu ebendieser.

tphSeiteSehenswuerdigkeitenKlosterbezirkKlosterschenke.html Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über die Klosterschänke, einen Kartenausschnitt mit ihrer Position und Bilder zu ebendieser.

tphSeiteSehenswuerdigkeitenKlosterbezirkMuseum.html Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über das Museum, einen Kartenausschnitt mit der Position und Bilder zu ebendiesem.

tphSeiteSehenswuerdigkeitenKlosterbezirkTorkapelle.html Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über die Torkapelle, einen Kartenausschnitt mit der Position und Bilder.

tphSeiteSehenswuerdigkeitenKlosterbezirkWassermuehle.html Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über die Wassermühle, einen Kartenausschnitt mit der Position und Bilder.

tphSeiteSehenswuerdigkeitenKulturpfad.html Auf der Seite des Kulturpfads werden die zum Kulturpfad gehörenden Sehenswürdigkeiten aufgezeigt und können in einer separaten Seite aufgerufen werden.

tphSeiteSehenswuerdigkeitenPlanetenlehrpfadErde.html Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über den Planeten Erde, sowie Bilder der Erde.

tphSeiteSehenswuerdigkeitenPlanetenlehrpfad.html Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Übersicht über die zum Planetenlehrpfad gehörenden Planeten (die dann auf separaten Seiten aufgerufen werden können) und Bilder der Planeten.

tphSeiteSehenswuerdigkeitenPlanetenlehrpfadJupiter.html Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über den Planeten Jupiter, sowie Bilder des Jupiter.

tphSeiteSehenswuerdigkeitenPlanetenlehrpfadMars.html Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über den Planeten Mars, sowie Bilder des Mars.

- tphSeiteSehenswuerdigkeitenPlanetenlehrpfadMerkur.html** Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über den Planeten Merkur, sowie Bilder des Merkur.
- tphSeiteSehenswuerdigkeitenPlanetenlehrpfadNeptun.html** Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über den Planeten Neptun, sowie Bilder des Neptun.
- tphSeiteSehenswuerdigkeitenPlanetenlehrpfadPluto.html** Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über den Planeten Pluto, sowie Bilder des Pluto.
- tphSeiteSehenswuerdigkeitenPlanetenlehrpfadSaturn.html** Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über den Planeten Saturn, sowie Bilder des Saturn.
- tphSeiteSehenswuerdigkeitenPlanetenlehrpfadSonne.html** Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über die Sonne, sowie Bilder der Sonne.
- tphSeiteSehenswuerdigkeitenPlanetenlehrpfadUranus.html** Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über den Planeten Uranus, sowie Bilder des Uranus.
- tphSeiteSehenswuerdigkeitenPlanetenlehrpfadVenus.html** Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über den Planeten Venus, sowie Bilder der Venus.
- tphSeiteSehenswuerdigkeitenSkulpturenufer.html** Die Seite des Skulpturenufers enthält einen Audio-Player (sofern die Audio-Daten heruntergeladen wurden) und Informationen über das Skulpturenufer.
- tphSeiteSehenswuerdigkeitenSonnenuhr.html** Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über die Sonnenuhr, einen Kartenausschnitt mit ihrer Position und Bilder zu ebendieser.
- tphSeiteSehenswuerdigkeitenStrasseDerMegalithkultur.html** Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über die Straße der Megalithkultur und Bilder zu dieser.
- tphSeiteSehenswuerdigkeitenVielstedterBauernhaus.html** Die Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über das Vielstedter-Bauernhaus, Kontakt-Daten, sowie einen Kartenausschnitt der Position und Bilder.
- tphSeiteSehenswuerdigkeitenWanderwegHuderBach.html** Die Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über den Wanderweg „Huder Bach“ und Bilder.
- tphSeiteSehenswuerdigkeitenWittemoor.html** Auf der Seite sind, in Klappmenüs eingebettet, ein Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über das Wittemoor und Bilder verfügbar.
- tphSeiteSehenswuerdigkeitenZeitstrahl2000.html** In Klappmenüs eingebettet, sind auf dieser Seite ein Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über den Zeitstrahl 2000 und Bilder verfügbar.

tphSeiteSehenswuerdigkeitenZiehbrunnen.html Diese Seite enthält einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), Informationen über den Ziehbrunnen, einen Kartenausschnitt mit der Position und Bilder.

tphSeiteSpielplaetze.html Zeigt eine Auswahl an Spielplätzen auf einer Google-Maps-Karte an, sowie die eigene Position.

tphSeiteUnterkuenfteFerienhausAschenbeck.html Die Seite enthält einen Audio-Player zum Ferienhaus Aschenbeck (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung des Hauses, Preisinformationen und Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteFerienhausHude.html Die Seite enthält zum Ferienhaus Hude einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung des Hauses, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteFerienwohnungAlteBaeckerei.html Die Seite enthält zur Ferienwohnung Alte Bäckerei einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Ferienwohnung, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteFerienwohnungAschenbeck.html Die Seite enthält zur Ferienwohnung Aschenbeck einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Ferienwohnung, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteFerienwohnungBurgmannshof.html Die Seite enthält zur Ferienwohnung Burgmannshof einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Ferienwohnung, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteFerienwohnungDatLuettjeHuus.html Die Seite enthält zur Ferienwohnung Dat Lüttje Huus einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Ferienwohnung, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteFerienwohnungEichkatz.html Die Seite enthält zur Ferienwohnung Eichkatz einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Ferienwohnung, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteFerienwohnungFHattendorf.html Die Seite enthält zur Ferienwohnung F. Hattendorf einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Ferienwohnung, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteFerienwohnungForsthausHasbruch.html Die Seite enthält zur Ferienwohnung Forsthaus Hasbruch einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Ferienwohnung, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteFerienwohnungHausLichtblick.html Die Seite enthält zur Ferienwohnung Haus Lichtblick einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Ferienwohnung, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteFerienwohnungHofStolle.html Die Seite enthält zur Ferienwohnung Hof Stolle einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Ferienwohnung, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteFerienwohnungKlosterschaenke.html Die Seite enthält zur Ferienwohnung Klosterschänke einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Ferienwohnung, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteFerienwohnungMKallisch.html Die Seite enthält zur Ferienwohnung M. Kallisch einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Ferienwohnung, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteFerienwohnungSeekermann.html Die Seite enthält zur Ferienwohnung Seekermann einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Ferienwohnung, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteFerienwohnungUKallisch.html Die Seite enthält zur Ferienwohnung U. Kallisch einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Ferienwohnung, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteGaestehausWobig.html Die Seite enthält zum Gästehaus Wobig einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung des Gästehauses, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteGaestehausWuerdemann.html Die Seite enthält zum Gästehaus Würdemann einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung des Gästehauses, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteHotelBurgdorf.html Die Seite enthält zum Hotel Burgdorf einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung des Hotels, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteHotelGarni.html Die Seite enthält zum Hotel Garni einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung des Hotels, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteHotelKlosterschaenke.html Die Seite enthält zum Hotel Klosterschänke einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung des Hotels, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfte.html Diese Seite bietet einen Überblick über die verschiedenen in Hude ansässigen Unterkünfte.

tphSeiteUnterkuenfteJugendherberge.html Die Seite enthält zur Jugendherberge einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Herberge, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenftePensionHofHeinemann.html Die Seite enthält zur Pension Hof Heinemann einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Pension, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenftePensionWPaulus.html Die Seite enthält zur Pension W. Paulus einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung der Pension, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Hauses enthalten, sowie Bilder.

tphSeiteUnterkuenfteWohnmobilstellplatz.html Die Seite enthält zum Wohnmobilstellplatz einen Audio-Player (falls die Audio-Daten heruntergeladen wurden), eine Beschreibung des Platzes, Preisinformationen, Kontaktinformationen, die auch einen Kartenausschnitt mit der Position des Platzes enthalten, sowie Bilder.

tphSeiteVeranstaltungen.html Auf der Seite werden die bevorstehenden Veranstaltungen angezeigt, die zuvor von einem Server abgerufen wurden.

6.3.3. Funktionalitäten

Limitierung von Inhalten

Um Inhalte auf einem mobilen Betriebssystem (siehe Abschnitt 2.2 auf Seite 5) nicht anzuzeigen (beispielsweise aufgrund nicht-unterstützter Funktionalität), muss dem nicht-anzuzeigenden HTML-Element eine Klasse zugewiesen werden. In Abbildung 6.14 wird dargestellt, wie welche Klasse zum Ausblenden in einem Android- (Zeile 2), iOS- (Zeile 5) und Windowsbetriebssystem (Zeile 8) gesetzt werden muss. Diese Klassen können kombiniert werden, so dass eine Anzeige nur auf einem Betriebssystem stattfindet (Zeile 11).

```

1  [...]
2  <div class="tphVersteckeAndroid">
3    Dieser Inhalt wird auf iOS- und Windows-Smartphones angezeigt.
4  </div>
5  <div class="tphVersteckeiOS">
6    Dieser Inhalt wird auf Android- und Windows-Smartphones angezeigt.
7  </div>
8  <div class="tphVersteckeWindows">
9    Dieser Inhalt wird auf Android- und iOS-Smartphones angezeigt.
10 </div>
11 <div class="tphVersteckeAndroid tphVersteckeWindows">
12   Dieser Inhalt wird nur auf iOS-Smartphones angezeigt.
13 </div>
14 [...]
```

Abbildung 6.14.: Ausblenden von Inhalte auf bestimmten Systemen

6. Implementierung

Zum Ausblenden der Inhalte wurde die Funktion „tphVersteckeOS()“ (Zeile 16-26), dargestellt in Abbildung 6.15 implementiert. Mit Hilfe der Funktion „tphHoleOS()“ (Zeile 2-12) wurde zunächst das Betriebssystem des Smartphones ausgelesen (Zeile 17). Anhand des Rückgabewertes wurden die Inhalte ausgeblendet, die die entsprechende Klasse im HTML-Gerüst gesetzt haben (18-25).

```
// Holt das Betriebssystem des Geräts
2  function tphHoleOS() {
    var OSName = "Unknown OS";
4   if (navigator.appVersion.indexOf("Win") !== -1)
        OSName = "Windows";
6   if (navigator.appVersion.indexOf("Mac") !== -1)
        OSName = "MacOS";
8   if (navigator.appVersion.indexOf("X11") !== -1)
        OSName = "UNIX";
10  if (navigator.appVersion.indexOf("Linux") !== -1)
        OSName = "Linux";
12  return OSName;
    }
14
// Versteckt die Ansicht für ein bestimmtes Betriebssystem
16  function tphVersteckeOS() {
    var tphOS = tphHoleOS();
18  if (tphOS === 'Linux')
        $('tphVersteckeAndroid').hide();
20  if (tphOS === 'MacOS')
        $('tphVersteckeiOS').hide();
22  if (tphOS === 'Windows')
        $('tphVersteckeWindows').hide();
24  if (tphOS === 'X11')
        $('tphVersteckeUnix').hide();
26  }
```

Abbildung 6.15.: JS-Funktion, die die Bilder in der Lightbox öffnet

Galerien

Nahezu jede/s in der App eingebundene Bild/Grafik öffnet sich innerhalb einer Lightbox (siehe Abschnitt 6.1.1 auf Seite 6.1.1). Damit sich die Bilder/Grafiken innerhalb einer Lightbox öffnen, müssen Sie nach dem Schema (Abbildung 6.16) eingebunden werden. Der <div>-Container mit den Klassen „small-6“ und „columns“ gibt an, dass für das Bild 50% der verfügbaren Breite, ohne geöffnete Lightbox, zur Darstellung genutzt werden. So werden, sofern vorhanden, immer zwei Bilder nebeneinander angezeigt. In den <a>-Tags gibt die Klasse „th“ an, dass es sich um ein Thumbnail handelt, „radius“ erzeugt runde Ecken. Durch die Klasse „tphGalerie“ wird das Bild beim Öffnen in einer Lightbox dargestellt. In der JS-Funktion „tphSetzeEinstellungenAufSeite()“ ist dieser Vorgang realisiert. Der dazu benötigte Code-Ausschnitt ist in Abbildung 6.17 dargestellt. Das Bild wird über den -Tag mit dem Attribut „src“ eingebunden. Das Attribut „src“ erhält als Wert den relativen Pfad des Bildes.

```
[...]
2  <div class="small-6 columns">
    <a class="th radius tphGalerie"
        href="images/sehenswuerdigkeiten/klosterbezirk/klosterbezirk_hude_01.jpg">
4   
    </a>
6  </div>
    <div class="small-6 columns">
8   <a class="th radius tphGalerie"
        href="images/sehenswuerdigkeiten/klosterbezirk/klosterbezirk_hude_02.jpg">
```



```

    
10 </a>
    </div>
12 [...]

```

Abbildung 6.16.: Lightbox ohne Galerie-Funktion

```

function tphSetzeEinstellungenAufSeite() {
2   [...]

4   $(".tphGalerie").fancybox({
        openEffect: 'none',
6       closeEffect: 'none'
    });
8
    [...]
10 }

```

Abbildung 6.17.: JS-Funktion, die Bilder in der Lightbox öffnet

Mit dem zuvor beschriebenen Verfahren wird ein Bild in einer Lightbox geöffnet und muss zunächst wieder geschlossen werden, um ein neues zu öffnen. Mit einer kleinen Änderung bei der Einbindung der Bilder, kann innerhalb der Lightbox ein Bilderwechsel durchgeführt werden. Dazu muss jedem `<a>`-Tag das „rel“-Attribut mit gleichem Wert hinzugefügt werden. Einen beispielhaften Ausschnitt zeigt Abbildung 6.18.

```

[...]
```

```

2 <div class="small-6 columns">
    <a class="th radius tphGalerie"
        href="images/sehenswuerdigkeiten/klosterbezirk/klosterbezirk_hude_01.jpg"
        rel="galerie">
4     
    </a>
6 </div>
<div class="small-6 columns">
8     <a class="th radius tphGalerie"
        href="images/sehenswuerdigkeiten/klosterbezirk/klosterbezirk_hude_02.jpg"
        rel="galerie">
        
10    </a>
    </div>
12 [...]
```

Abbildung 6.18.: Lightbox mit Galerie-Funktion

Kartenmaterial

Zur Einbindung von Kartenmaterial, stammend von Google Maps, wird das Plugin „Google-maps-v3-plugin for jQuery and jQuery Mobile“ genutzt (siehe Abschnitt 6.1.1 auf Seite 51). Mit der Funktion „tph-NutzeGPS(option)“ können Google-Maps-Karten auf einer Seite eingebunden werden. Dazu muss auf der Seite ein `<div>`-Container, mit der Klasse „tphGoogleMapsKarte“, vorhanden sein. Aufgrund der schlechten Datenverbindung mancher Telekommunikationsanbieter in Hude und Umgebung, wird das Kartenmaterial nur bei einer ausreichend guten Datenverbindung dargestellt. Als ausreichend gute Datenverbindung gilt 3G oder 4G. Ist die Datenverbindung nicht ausreichend, werden dem Anwender alternativ seine GPS-Koordinaten, Höhe und Richtung angezeigt.

Durch den optionalen Parameter „option“ kann bestimmt werden, welche zusätzliche Daten auf der Karte angezeigt werden. Der Parameter kann dabei drei Werte annehmen:

6. Implementierung

- `tphHudePadd` Dem Anwender wird seine Position und die Route des Huder Padd angezeigt, sowie Marker von Sehenswürdigkeiten, die sich auf dem Padd befinden.
- `tphParklaetzeHude` Dem Anwender werden (mit Markern im Parkplatz-Design) alle Parkplätze in Hude angezeigt und (mit einem Marker-Pin) die Position des Anwenders.
- `tphSpielplätze` Dm Anwender wird (mit Markern im Spielplatz-Design) eine Auswahl von fünf Spielplätzen in Hude angezeigt und (mit einem Marker-Pin) die Position des Anwenders.

In Abbildung 6.19 wird die beispielhafte Einbindung einer Karte innerhalb einer Seite dargestellt. In den `<div>`-Container in Zeile 1 wird die Karte eingefügt. Der Aufruf der Funktion „`tphNutzeGPS()`“ in Zeile 5 startet den Abruf der eigenen Position und das Einfügen der Karte.

```


---

<div class="tphGoogleMapsKarte"></div>  
2 <script>  
   $(function() {  
4     tphHeaderUberschriftAendern('Aktuelle Position');  
     tphNutzeGPS();  
6 });  
</script>

---


```

Abbildung 6.19.: Google Maps im HTML

Abbildung 6.20 zeigt die Implementierung von Google Maps mittels JS. Der Abruf der eigenen Position erfolgt in Zeile 4. Wird diese erfolgreich bestimmt, wird die Datenverbindung geprüft (Zeile 8). Entspricht diese einer LAN-, WIFI-, 3G- oder 4G-Verbindung wird der `<div>`-Container auf eine Breite und Weite von 90% der Displaygröße vergrößert (Zeile 9-10). Anschließend wird die Karte eingefügt und auf den GPS-Koordinaten des Anwenders zentriert. An dieser Stelle wird ebenfalls ein Marker-Pin gesetzt um die Position zu visualisieren. Der Radius von 15 Metern wird, zur Orientierung, zusätzlich um den Pin angezeigt (Zeile 12-22).

Bei einer nicht ausreichenden Datenverbindung erhält der Anwender darüber eine Information. Ihm werden anstatt einer Karte seine GPS-Koordinaten, Höhe und Richtung angezeigt (Zeile 23-25).

Durch Setzen eines Parameters „`option`“, mit einem der Werte „`tphParkplaetzeHude`“, „`tphSpielplätze`“ oder „`tphHudePadd`“, werden zusätzliche Daten zur Karte hinzugefügt.

Mit dem Parameter „`tphParkplaetzeHude`“ (Zeile 28) werden alle öffentlichen Parkplätze in Hude der Karte hinzugefügt. Dazu besitzt die Funktion „`tphParkplaetze()`“ (Zeile 69-86 zeigt nur einen Ausschnitt der Funktion) als Rückgabewert ein Array mit den GPS-Koordinaten der Parkplätze. Diese Koordinaten werden mit einem „Parkplatz“-Marker (Zeile 30) nacheinander der Karte hinzugefügt (Zeile 31-36).

Mit dem Parameter „`tphSpielplätze`“ (Zeile 38) werden die fünf schönsten Spielplätze Hudes der Karte hinzugefügt. Die Funktion „`tphSpielplaetze()`“ (Zeile 102-110) hat als Rückgabewert ein Array mit den GPS-Koordinaten der Spielplätze. Diese Koordinaten werden mit einem „Spielplatz“-Marker (Zeile 40) nacheinander der Karte hinzugefügt (Zeile 41-47).

Durch den Parameter „`tphHudePadd`“ (Zeile 49) wird der Huder Padd der Karte hinzugefügt. Ein Array mit den GPS-Koordinaten des Huder Padd besitzt die Funktion „`tphHudePadd`“ (Zeile 88-110 zeigt einen Ausschnitt der Funktion) als Rückgabewert. Die einzelnen Koordinaten des Padd werden der Karte hinzugefügt (Zeile 52-54) und anschließend als Pfad verbunden (Zeile 55-59).

```


---

1 function tphNutzeGPS(option) {  
   var lat;  
3   var lon;  
   navigator.geolocation.getCurrentPosition(tphNutzeGPSSuccess, tphNutzeGPSError,  
     {maximumAge: 0, timeout: 15000, enableHighAccuracy: true});  
5   function tphNutzeGPSSuccess(position) {  
     lat = position.coords.latitude;  


---


```

```

7     lon = position.coords.longitude;
8     if (Connection.ETHERNET || Connection.WIFI || Connection.CELL_3G ||
9         Connection.CELL_4G) {
10        $('.tphGoogleMapsKarte').css('height', $(window).height() * 0.9);
11        $('.tphGoogleMapsKarte').css('width', $(window).width() * 0.9);
12        var aktuellePosition = new google.maps.LatLng(lat, lon);
13        $('.tphGoogleMapsKarte').gmap({'center': aktuellePosition});
14        $('.tphGoogleMapsKarte').gmap('option', 'zoom', 15);
15        $('.tphGoogleMapsKarte').gmap('addMarker', {'id': 'aktuellePosition', 'position':
16            aktuellePosition, 'bounds': false});
17        $('.tphGoogleMapsKarte').gmap('addShape', 'Circle', {
18            'strokeWeight': 0,
19            'fillColor': "#008595",
20            'fillOpacity': 0.25,
21            'center': aktuellePosition,
22            'radius': 15,
23            'clickable': false
24        });
25    } else {
26        $('.tphGoogleMapsKarte').html('<div id="tphGoogleMapsKarte"><p>KEINE AUSREICHENDE
27            DATENVERBINDUNG</p><p>' + print_r(navigator) + '</p></div>');
28    }
29
30    switch (option) {
31        case 'tphParklaetzeHude':
32            var parkplaetze = tphParkplaetze();
33            var icon = new
34                google.maps.MarkerImage("http://m.touristik-palette-hude.de/download/image/parking.png");
35            for (var i = 0; i < parkplaetze.length; i++) {
36                /*
37                 * bounds: true richtet die Karte so aus, dass alle Marker zu sehen sind.
38                 * bounds: false fügt alle Marker lediglich der Karte hinzu
39                 */
40                $('.tphGoogleMapsKarte').gmap('addMarker', {'id': 'tphParkplatz-' + i,
41                    'position': parkplaetze[i], 'bounds': true, 'icon': icon});
42            }
43            break;
44        case 'tphSpielplätze':
45            var spielplaetze = tphSpielplaetze();
46            var icon = new
47                google.maps.MarkerImage("http://m.touristik-palette-hude.de/download/image/playground.png");
48            for (var i = 0; i < spielplaetze.length; i++) {
49                /*
50                 * bounds: true richtet die Karte so aus, dass alle Marker zu sehen sind.
51                 * bounds: false fügt alle Marker lediglich der Karte hinzu
52                 */
53                $('.tphGoogleMapsKarte').gmap('addMarker', {'id': 'tphSpielplatz-' + i,
54                    'position': spielplaetze[i], 'bounds': true, 'icon': icon});
55            }
56            break;
57        case 'tphHudePadd':
58            var hudePadd = tphHuderPadd();
59            var koordinaten = new Array();
60            for (var i = 0; i < hudePadd.length; i++) {
61                koordinaten.push(new google.maps.LatLng(hudePadd[i][0], hudePadd[i][1]));
62            }
63            $('.tphGoogleMapsKarte').gmap('addShape', 'Polyline', {
64                'path': koordinaten,
65                'strokeColor': '#c00',
66                'strokeThickness': 5
67            });
68            break;
69    }
70
71    function tphNutzeGPSError() {
72        console.log('fehler');
73    }
74
75    function tphParkplaetze() {
76        var parkplaetze = new Array();

```

6. Implementierung

```
71 // Kirchstraße/Friedhof/St. Elisabeth Kirche
    parkplaetze.push('53.118029,8.449199');
73 // Waldbad
    parkplaetze.push('53.119665,8.445948');
75 // Sport- und Freizeitzentrum
    parkplaetze.push('53.111627,8.445061');
77
    [...]
79
    // Torfplatz
81    parkplaetze.push('53.111456,8.459861');
    // An der Weide
83    parkplaetze.push('53.117967,8.455893');
    parkplaetze.sort();
85    return parkplaetze;
}
87
function tphHuderPadd() {
89    var wegpunkte = new Array();
    wegpunkte.push(['53.110073', '8.460763']);
91    wegpunkte.push(['53.110516', '8.460983']);
    wegpunkte.push(['53.111004', '8.460409']);
93
    [...]
95
    wegpunkte.push(['53.108761', '8.461651']);
97    wegpunkte.push(['53.109131', '8.460797']);
    wegpunkte.push(['53.110080', '8.460760']);
99    return wegpunkte;
}
101
function tphSpielplaetze() {
103    var spielplaetze = new Array();
    spielplaetze.push('53.118085,8.454319');
105    spielplaetze.push('53.115497,8.459709');
    spielplaetze.push('53.110063,8.449881');
107    spielplaetze.push('53.095256,8.464326');
    spielplaetze.push('53.097363,8.484035');
109    return spielplaetze;
}
```

Abbildung 6.20.: Google-Maps-Implementierung im JS

Zielgruppeneinstellungen

Durch die Zielgruppeneinstellungen sollen bestimmte Inhalte für ausgewählte Zielgruppen hervorgehoben werden. Es kann aus folgenden drei Zielgruppen gewählt werden: „keine“, „Familie mit Kind“ und „Best Ager“. Zum aktuellen Zeitpunkt kann eine Zielgruppeneinstellung vorgenommen werden, die allerdings hat keine Auswirkungen auf die App hat. Die Begründung ist in Abschnitt 3.1.7 auf Seite 14 zu finden.

Um für die verschiedenen Zielgruppen Inhalte in unterschiedlicher Form dazustellen, ist es notwendig, dass den HTML-Elementen Klassen zugewiesen werden. Abbildung 6.21 zeigt für jede Zielgruppe einen eigenen <div>-Container. Innerhalb jedes <div>-Containers können die Inhalte für jede Zielgruppe unterschiedlich aufgearbeitet werden. Dem Anwender wird jedoch nur der Inhalt der eingestellten Zielgruppe angezeigt.

```
<div class="tphZielgruppeKeine">
2   Inhalt für keine Zielgruppe
</div>
4
<div class="tphZielgruppeFamilie">
```

```

6   Inhalt für die Zielgruppe 'Familie mit Kind'
   </div>
8
   <div class="tphZielgruppeBestager">
10  Inhalt für die Zielgruppe 'Best Ager'
   </div>

```

Abbildung 6.21.: Zielgruppeneinstellungen Realisierung in HTML

Abbildung 6.22 zeigt die Implementierung der Zielgruppeneinstellung. Beim ersten Start der App wird durch die Funktion „tphAppStart“ (Zeile 1-11 zeigt einen Ausschnitt der Funktion) die Zielgruppe zunächst auf „keine“ gesetzt (Zeile 8), so dass alle Inhalte ohne Präferenz dargestellt werden würden.

Die Einstellung der Zielgruppe wird mit der Funktion „tphSpeicherZielgruppe(tphZielgruppe)“ (Zeile 14-18), im localStorage als Key-Value-Paar gespeichert. Der Parameter „tphZielgruppe“ bekommt dazu den Wert „keine“, „familie“ oder „bestager“ übergeben. Durch die Funktion „tphHoleZielgruppe()“ (Zeile 14-18) wird die eingestellte Sprache aus dem localStorage ausgelesen.

Zeile 26-46 zeigt einen Ausschnitt der Funktion „tphSetzeEinstellungenAufSeite()“. Innerhalb dieser ist die Anzeige der Zielgruppeneinstellungen realisiert (Zeile 30-44). Dies geschieht innerhalb dieser Funktion, da sie bei jedem Seitenaufruf ausgeführt wird. Je nach eingestellter Zielgruppe wird nur der Inhalt für die entsprechende Zielgruppe (Abbildung 6.21) dargestellt.

```

1  function tphAppStart() {
   var tphStorage = tphLadeLocalStorage();
3  // Enthält den Zeitstempel des ersten Starts
   var tphErsterStart = tphStorage.getItem('tphErsterStart');
5  // Überprüfen, ob die App zum ersten Mal startet
   if (tphErsterStart === null) {
7     [...]
     tphSpeicherZielgruppe('keine');
9     [...]
   }
11  [...]
  }

13
   function tphSpeicherZielgruppe(tphZielgruppe) {
15  // keine, familie, bestager
   var tphStorage = tphLadeLocalStorage();
17  tphStorage.setItem('tphZielgruppe', tphZielgruppe);
   }

19
   // Lädt die Zielgruppe aus dem localStorage
21  function tphHoleZielgruppe() {
   var tphStorage = tphLadeLocalStorage();
23  return tphStorage.getItem('tphZielgruppe');
   }

25
   function tphSetzeEinstellungenAufSeite() {
27  [...]
   var tphZielgruppe = tphHoleZielgruppe();
29  [...]
   if (tphZielgruppe === 'keine') {
31     $('.tphZielgruppeKeine').show();
     $('.tphZielgruppeFamilie').hide();
33     $('.tphZielgruppeBestager').hide();
   }
35  if (tphZielgruppe === 'familie') {
     $('.tphZielgruppeKeine').hide();
37     $('.tphZielgruppeFamilie').show();
   }

```

6. Implementierung

```
    $('.tphZielgruppeBestager').hide();
39 }
    if (tphZielgruppe === 'bestager') {
41     $('.tphZielgruppeKeine').hide();
        $('.tphZielgruppeFamilie').hide();
43     $('.tphZielgruppeBestager').show();
    }
45     [...]
}
```

Abbildung 6.22.: Zielgruppeneinstellungen

Schriftgröße

Zur Verbesserung der Nutzerfreundlichkeit kann in den Einstellungen die Schriftgröße angepasst werden. Dazu stehen drei Schriftgrößen zur Verfügung: „normal“, „mittel“ und „gross“. Die Implementierung wird in Abbildung 6.23 dargestellt. Beim ersten Start der App wird die Schriftgröße durch die Funktion „tphSpeicherSchriftgroesse(tphSchriftgroesse)“ (Zeile 49-54) innerhalb „tphStart()“ auf „normal“ eingestellt (Zeile 9). Der Grund dafür ist, dass zu diesem Zeitpunkt die optimale Schriftgröße für die Displaygröße durch das Foundation Framework (siehe Abschnitt 6.1.2 auf Seite 6.1.2) eingestellt ist. Der Pixelwert dieser optimalen Darstellung wird ebenfalls im localStorage gespeichert (Zeile 11). Durch die Funktion „tphSetzeEinstellungenAufSeite()“ (Zeile 16-35 zeigt einen Ausschnitt) wird die Schriftgröße auf der Seite eingestellt. Zur Einstellung der Schriftgrößen wird zunächst die eingestellte Schriftgröße ausgelesen (Zeile 18). Im Weiteren wird dann der Pixelwert der Schriftgröße „normal“ ausgelesen (Zeile 19). Für die Schriftgrößen „mittel“ wird zu diesem Wert „2“ hinzu addiert (Zeile 26), für die Schriftgröße „gross“ wird zu diesem Wert „4“ hinzu addiert. Das Setzen der Schriftgröße erfolgt nach deren Berechnung (Zeile 23, 28 und 33).

```
function tphAppStart() {
2  var tphStorage = tphLadeLocalStorage();
    // Enthält den Zeitstempel des ersten Starts
4  var tphErsterStart = tphStorage.getItem('tphErsterStart');
    // Überprüfen, ob die App zum ersten Mal startet
6  if (tphErsterStart === null) {
    [...]
8     // Schriftgroesse auf normal setzen
    tphSpeicherSchriftgroesse('normal');
10    // Größe der Schriftgröße 'normal'
    var tphSchriftgroesseNormal = parseInt($('.tphContent').css('font-size'));
12    tphStorage.setItem('tphSchriftgroesseNormal', tphSchriftgroesseNormal);
    [...]
14 }

16 function tphSetzeEinstellungenAufSeite() {
    [...]
18    var tphSchriftgroesse = tphHoleSchriftgroesse();
    var tphSchriftgroesseNormal = tphHoleSchriftgroesseNormal();
20    [...]
    if (tphSchriftgroesse === 'normal') {
22        tphSchriftgroesseNormal += 'px';
        $('.tphContent').css('font-size', tphSchriftgroesseNormal);
24    }
    if (tphSchriftgroesse === 'mittel') {
26        var tphSchriftgroesseMittel = parseInt(tphSchriftgroesseNormal) + 2;
        tphSchriftgroesseMittel += 'px';
28        $('.tphContent').css('font-size', tphSchriftgroesseMittel);
    }
30    if (tphSchriftgroesse === 'gross') {
        var tphSchriftgroesseGross = parseInt(tphSchriftgroesseNormal) + 4;
32        tphSchriftgroesseGross += 'px';
        $('.tphContent').css('font-size', tphSchriftgroesseGross);
34    }
}
```

```

36 }
37 // Lädt die Schriftgröße aus dem localStorage
38 function tphHoleSchriftgroesse() {
39     var tphStorage = tphLadeLocalStorage();
40     return tphStorage.getItem('tphSchriftgroesse');
41 }
42 // Lädt die normale Schriftgröße vom ersten Start
43 function tphHoleSchriftgroesseNormal() {
44     var tphStorage = tphLadeLocalStorage();
45     return tphStorage.getItem('tphSchriftgroesseNormal');
46 }
47
48 function tphSpeicherSchriftgroesse(tphSchriftgroesse) {
49     // normal, mittel, gross
50     var tphStorage = tphLadeLocalStorage();
51     tphStorage.setItem('tphSchriftgroesse', tphSchriftgroesse);
52 }

```

Abbildung 6.23.: Schriftgrößen-Implementierung

Multilingual

Die multilinguale Unterstützung ist neben der visuellen Darstellung auch auditiv implementiert. Zum jetzigen Zeitpunkt könnten drei Sprachen eingestellt werden: Deutsch, Englisch und Plattdeutsch. In den Einstellungen kann jedoch nur zwischen den Sprachen Deutsch und Englisch gewählt werden. Die Begründung ist in Abschnitt 3.1.7 auf Seite 14 zu finden. Beim initialen Start der App wird für visuelle als auch auditive Inhalte die Sprache auf „Deutsch“ eingestellt. Die Einstellung der Sprache wird mittels der Abkürzungen de, en oder pd (Deutsch, Englisch oder Plattdeutsch) im localStorage als Key-Value-Paar abgespeichert. Für visuelle Inhalte ist der Key „tphSprache“ und für auditive Inhalte „tphSpracheAudio“. Zum Speichern der Spracheinstellungen sind die Funktionen „tphSpeicherSprache(tphSprache)“ und „tphSpeicherSpracheAudio(tphSpracheAudio)“ realisiert. Beide Funktionen bekommen die zu speichernde Sprache als Abkürzung, als Parameter übergeben (Abbildung 6.24).

```

1 // Speichert die Sprache im localStorage
2 function tphSpeicherSprache(tphSprache) {
3     var tphStorage = tphLadeLocalStorage();
4     tphStorage.setItem('tphSprache', tphSprache);
5 }
6
7 // Speichert die Sprache für Audio-Dateien im localStorage
8 function tphSpeicherSpracheAudio(tphSpracheAudio) {
9     var tphStorage = tphLadeLocalStorage();
10    tphStorage.setItem('tphSpracheAudio', tphSpracheAudio);
11 }

```

Abbildung 6.24.: Funktionen zum Speichern der Sprachen

Die Sprache kann in den Einstellungen separat (audio/visuell) eingestellt werden.

Text

Zur Realisierung der unterschiedlichen Sprache wurden folgende drei Klassen definiert:

tphSpracheDE Klasse für die deutsche Sprache.

tphSpracheEN Klasse für die englische Sprache.

tphSprachePD Klasse für die plattdeutsche Sprache.

Diese Klassen können jedem HTML-Element zugewiesen werden. Der Inhalt, den die HTML-Elemente mit der gesetzten Klasse umschließen, entspricht der Sprache der gesetzten Klasse. Alle Sprachen, die auf einer Seite verfügbar sind, werden auch auf dieser Seite implementiert (Abbildung 6.25). Die Anzeige der eingestellten Sprache führt das JS-Skript (Abbildung 6.26) aus.

```
1 <div class="tphSpracheDE">
2   Eine schöne Tasse Tee
3 </div>
4 <div class="tphSpracheEN">
5   A nice cup of tea
6 </div>
7 <div class="tphSprachePD">
8   Een schön Tass Tee
9 </div>
```

Abbildung 6.25.: Multilinguales HTML

Zur Darstellung der eingestellten Sprache ist die Funktion „tphSetzeEinstellungenAufSeite()“ zuständig. Mit dieser werden ebenso weitere Einstellungen auf den einzelnen anzuzeigenden Seiten gesetzt, so dass die Abbildung 6.26 nur den Ausschnitt zeigt, der für die Darstellung der visuellen Inhalte benötigt wird. In Zeile 3 wird die eingestellte Sprache aus dem localStorage geladen und in der Variable „tphSprache“ gespeichert. So kann die Variable die Werte „de“, „en“ oder „pd“ annehmen. Je nach Wert wird die eingestellte Sprache eingeblendet (Zeile 9, 17 oder 25) und die anderen Sprachen ausgeblendet (Zeile 10 und 11, 16 und 18 oder 23 und 25).

```
1 function tphSetzeEinstellungenAufSeite() {
2   // Lädt die im localStorage gespeicherte Spracheinstellung
3   var tphSprache = tphHoleSprache();
4
5   [...]
6
7   // Zeigt nur den Text auf Deutsch an
8   if (tphSprache === 'de') {
9     $('.tphSpracheDE').show();
10    $('.tphSpracheEN').hide();
11    $('.tphSprachePD').hide();
12  }
13
14  // Zeigt nur den Text auf Englisch an
15  if (tphSprache === 'en') {
16    $('.tphSpracheDE').hide();
17    $('.tphSpracheEN').show();
18    $('.tphSprachePD').hide();
19  }
```



```

21 // Zeigt nur den Text auf Plattdeutsch an
    if (tphSprache === 'pd') {
23     $('.tphSpracheDE').hide();
        $('.tphSpracheEN').hide();
25     $('.tphSprachePD').show();
    }
27
    [...]
29
}

```

Abbildung 6.26.: Multilinguales JS

Audio

Zur Realisierung der multilingualen auditiven Inhalte mussten zwei zusätzliche Ressourcen eingesetzt werden. Die eine Ressource ist die Stimmen-Demonstrations-Website der Acapella Group¹⁸ und die andere „Audacity“.

Audacity ist eine, freie und plattformunabhängige, Software zur Aufnahme und Bearbeitung von Audio-Dateien[Aud13], die sich zur Zeit in der Version 2.0.3¹⁹ befindet.

Acapella Group ist nach eigener Aussage der führende Experte im Bereich der Übersetzung von Texten in Sprache (TTS²⁰), mit einer Erfahrungen von 25 Jahren. Dazu werden verschiedene Lösungen angeboten. Im Repertoire befindet sich eine Auswahl von 100 unterschiedlichen, natürlich klingenden synthetisierten Stimmen, die auf 30 Sprachen verteilt sind[Aca]. Die Acapella Group bietet auf ihrer Website einen Stimmen-Demo-Service¹⁸ an. An diesen können 300 Zeichen zur probeweisen Wiedergabe und zum Testhören übergeben werden.

Als Sprach-Einstellungen für die Wiedergabe wurde für die deutschen Texte „Laura“ als Stimme gewählt und für die englischen „Tracy“. Die Wiedergabe der 300 Zeichen wurde mit dem Programm Audacity aufgezeichnet. Dieser Vorgang wurde wiederholt, bis ein Text vollständig auditiv vorhanden war. Anschließend wurde der Text mit Audacity zusammengeschnitten, so dass für jeden Text eine einzelne Audio-Datei verfügbar gemacht wurde.

Die Audio-Dateien wurden auf den Server der Touristik-Palette-Hude hochgeladen und können über die Einstellungen, innerhalb der App, als zusätzlicher Inhalt heruntergeladen werden. Dazu wurde ein Array mit den URIs der einzelnen Audio-Dateien angelegt (Abbildung 6.27, Zeile 3). Der Download der Dateien wird mit der Funktion „tphDateisystem(option, dateiname)“ realisiert. Der Parameter „option“ gibt dabei an, welche Aktion mit dem Dateisystem durchgeführt werden soll, der Parameter „dateiname“ ist optional und wird für den Download nicht benötigt.

Die Abbildung 6.27 zeigt zwischen Zeile 7 und 74 die Implementierung der Download-Funktion. Mit dem Aufruf der Funktion wird zunächst der Zugriff auf das Dateisystem hergestellt (Zeile 8). Ist dieser Zugriff nicht möglich, wird eine Fehlermeldung ausgegeben (Zeile 71-74). Wenn der Zugriff auf das Dateisystem möglich ist, wird mittels Switch-Case-Vergleich (Zeile 9) entschieden, welche Funktionalität der Funktion ausgeführt wird. Für den Download der Audio-Dateien wird in den Ordner „Hude“ gewechselt, existiert dieser nicht, wird er automatisch erstellt (Zeile 12) und anschließend (Zeile 13) eine „FileTransfer“-Instanz des PhoneGap-Frameworks erzeugt. Über diese wird der Download im Folgenden ausgeführt.

Die Dateien, die heruntergeladen werden sollen, müssen einen absoluten Pfad auf dem Dateisystem zugewiesen bekommen. Der absolute Pfad zum Ordner „Hude“ wird in Zeile 21 erzeugt. Für jede Sprache wird der Downloadpfad (Zeile 33/34) angepasst, so dass für jede Sprache ein Unterordner im Ordner Audio existiert, der die Audio-Dateien enthält. Von der „<http://m.touristik-palette-hude.de/download/audio/de/erde.mp3>“

¹⁸<http://www.acapella-group.com/text-to-speech-interactive-demo.html>

¹⁹Stand: 27. August 2013

²⁰Text-To-Speech (TTS) ist ein Verfahren zur Konvertierung eines Fließtextes in eine Sprachausgabe. Die Sprachausgabe erfolgt durch synthetisierte Stimmen - bekannt zum Beispiel aus Navigationssystemen.

6. Implementierung

bleibt nur „audio/de/erde.mp3“, das dem absoluten Pfad hinzugefügt wird. Für jede Sprache muss die Datei denselben Namen tragen. Die Unterscheidung der Sprache wird über den Ordner vorgenommen.

Über die „FileTransfer“-Instanz wird mit der Funktion „download“ und den Parametern „uri“ (Quelle) und „downloadPfad“ (Ziel) der Download gestartet (Zeile 56). Beim erfolgreichen Download wird der Anwender darüber informiert, dass der Download abgeschlossen wurde (Zeile 58). Tritt ein Fehler während des Downloads auf, wird der Anwender ebenfalls darüber informiert (Zeile 61). Während des Downloads erhält der Anwender, durch die Funktion „onprogress“ der FileTransfer-Instanz, Informationen über die Durchführung (Zeile 38) und über den Status (Zeile 42) des Downloads. Parallel wird dem Anwender angezeigt, wie viele Dateien bereits heruntergeladen wurden (Zeile 50). Diese Anzahl wird ebenfalls im localStorage gespeichert, so dass der Anwender beim erneuten Abruf der Einstellungen eine Information darüber erhält.

```
// Enthält die Liste der zur downloadenden Dateien
2 function tphAudioDateien() {
    var dateien = new
      Array('http://m.touristik-palette-hude.de/download/audio/de/erde.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/neptun.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/unterkuenfte.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/gastronomie.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/planetenlehrpfad.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/uranus.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/hude-info.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/pluto.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/urwaldhasbruch.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/jupiter.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/saturn.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/venus.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/klosterbezirk.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/schenke_und_remise.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/vielstedter_bauernhaus.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/kulturpfad.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/skulpturenufer-und-haus.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/wassermuehle.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/mars.mp3', [...],
        'http://m.touristik-palette-hude.de/download/audio/de/sonne.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/wittemoor.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/merkur.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/strasseder-megalithkultur.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/zeitstrahl2000.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/museum.mp3',
        'http://m.touristik-palette-hude.de/download/audio/de/torkapelle.mp3',
        'http://m.touristik-palette-hude.de/download/audio/en/hude-info.mp3');
4   return dateien;
  }
6
function tphDateisystem(option, dateiname) {
8   window.requestFileSystem(LocalFileSystem.PERSISTENT, 0, function(filesystem) {
    switch (option) {
10      [...]
      case 'download':
12        filesystem.root.getDirectory('Hude', {create: true, exclusive: false},
          function(directory) {
14          var ft = new FileTransfer();
            // Erhält die URI eines Downloads
            var uri;
16          // Enthält den Dateinamen
            var file;
18          // Erhält den Downloadpfad der Datei
            var downloadPfad;
20          // Enthält der Pfad zum Ordner \grqq{}Hude\grqq{} auf dem Dateisystem
            var pfad = directory.fullPath;
22          // Enthält alle URIs der Audio-Dateien
            var dateien = tphAudioDateien();
24          // Anzahl der Audio-Dateien
            var anzahlDateienGesamt = dateien.length;
26          var tphStorage = tphLadeLocalStorage();
```

```

// Zurücksetzen der Anzeige in den Einstellungen von bereits heruntergeladenen
// Dateien und den zu ladenden Dateien
28 $(' #tphAnzahlAudioBereitsGeladen').html('<span
    id="tphAnzahlAudioBereitsGeladen">0</span>');
$(' #tphAnzahlAudioInsgesamt').html('<span id="tphAnzahlAudioInsgesamt">' +
    dateien.length + '</span>');
30 // Download der Dateien starten
for (var i = 0; i < dateien.length; i++) {
32     uri = encodeURIComponent(dateien[i]);
    file = tphDownloadPfad(uri);
34     downloadPfad = pfad + '/' + file;
    // Funktion, die den Fortschritt während des Downloads der Datei angibt
36     ft.onprogress = function(progressEvent) {
        // Feedback für den Anwender, dass der Download ausgeführt wird.
38         $(' #tphDownloadStatus').html('<p id="tphDownloadStatus">Download wird
            ausgeführt ...</p>');
        if (progressEvent.lengthComputable) {
40             var perc = Math.floor(progressEvent.loaded / progressEvent.total * 100);
            //console.log(perc + '% geladen' + downloadPfad);
42             $(' #tphDownloadStatus').html('<p><strong>' +
                (Math.floor(progressEvent.loaded / progressEvent.total * 100)) +
                '</strong></p>');
            if (progressEvent.loaded === progressEvent.total) {
44                 var anzahlHeruntergeladen =
                    tphStorage.getItem('tphAudioDateienHeruntergeladen');
                    var anzahlDateien = tphAudioDateien().length;
                    //console.log(anzahlHeruntergeladen);
                    if (anzahlHeruntergeladen < anzahlDateien) {
46                         anzahlHeruntergeladen++;
                        tphSpeicherDateienHeruntergeladen(anzahlHeruntergeladen);
50                         $(' #tphAnzahlAudioHeruntergeladen').text(anzahlHeruntergeladen);
                    }
                    }
52             }
        }
54     };
    // Funktion die den Download startet.
56     ft.download(uri, downloadPfad,
        function(entry) {
58         $(' #tphDownloadStatus').html('<p id="tphDownloadStatus">Download
            abgeschlossen</p>');
        },
60         function(error) {
            $(' #tphDownloadStatus').html('<p id="tphDownloadStatus"><strong
                style="color: red;">Fehler beim Download: ' + error.code +
                '</strong></p>');
62         });
        }
64     }, function(e) {
        console.log('Fehler Dateisystem ' + print_r(e));
66         $(' #tphDownloadStatus').append('<p>Fehler beim Zugriff auf das Dateisystem:
            <br/>' + print_r(e) + '</p>');
        });
68     break;
    [...]
70 }
}, function(e) {
72     console.log('Fehler: kein Zugriff auf das Dateisystem: ' + print_r(e));
    $(' #tphDownloadStatus').append('<p>Fehler kein Zugriff auf das Dateisystem: <br/>' +
        print_r(e) + '</p>');
74 });

```

Abbildung 6.27.: Download der Audio-Dateien mittels JS

Zur Wiedergabe von Audio-Dateien ist ein Audio-Player (Abbildung 6.28) innerhalb der Inhaltsseiten integriert, der die Kontrollelemente „Play“, „Pause“ und „Stop“ besitzt (Zeile 13-36). Sollte keine Audio-Datei mit der Seite verknüpft sein, so werden die Kontrollelemente nicht angezeigt und stattdessen ein Hinweis-text präsentiert (Zeile 37-39). Die Verknüpfung einer Audio-Datei mit einer Seite erfolgt in Zeile 51.

```

1   [...]
2
3   <section>
4     <p class="title" data-section-title>
5       <a href="#">
6         <span class="tphSpracheDE">Hören Sie die Informationen</span>
7         <span class="tphSpracheEN">Listen to the information</span>
8         <span class="tphSprachePD">Lustern tau de Toon</span>
9       </a>
10    </p>
11    <div class="content" data-section-content>
12      <!-- Audio-Player -->
13      <div id="tphPlayer">
14        <!-- Kontrollelemente des Players -->
15        <div class="tphPlayerControl" style="display: none">
16          <div class="row">
17            <!-- Kontrollelement: Abspielen -->
18            <div class="small-4 columns">
19              <a id="tphPlayButton" href="#">
20                
21              </a>
22            </div>
23            <!-- Kontrollelement: Pause -->
24            <div class="small-4 columns">
25              <a id="tphPauseButton" href="#" onclick="tphAudioPause();">
26                
27              </a>
28            </div>
29            <!-- Kontrollelement: Stop -->
30            <div id="tphStopButton" class="small-4 columns">
31              <a href="#" onclick="tphAudioStoppen();">
32                
33              </a>
34            </div>
35          </div>
36        </div>
37        <div class="tphPlayerKeineDateien">
38          <p>Sie haben keine Audio-Inhalte heruntergeladen oder die von Ihnen gewählte
39            Audio-Sprache ist noch nicht verfügbar. Haben Sie noch keine Audio-Inhalte
40            heruntergeladen, können Sie dies in den Einstellungen nachholen. Dazu klicken
41            Sie bitte <a href="#tphSeiteEinstellungen">hier</a> und wählen "Download
42            starten".</p>
43        </div>
44      </div>
45    </div>
46  </section>
47
48  [...]
49
50  <script>
51    $(function() {
52      // Überschrift der Seite setzen
53      tphHeaderÜberschriftAendern('Hude-Info');
54      // Verknüpfung einer Audio-Datei mit der Seite
55      tphDateisystem('audioVorhanden', 'hude-info.mp3');
56    });
57  </script>

```

Abbildung 6.28.: Ausschnitt einer Inhaltsseite mit Audio-Player

Die Logik des Audio-Players (Abbildung 6.28) wird in Abbildung 6.29 dargestellt. Die Verknüpfung einer Audio-Datei mit einer Seite erfolgt wie bereits in Abbildung 6.28 Zeile 51 dargestellt. Die Funktion „tph-Dateisystem“ stellt zunächst den Zugriff auf das Dateisystem her (Zeile 3). Durch den Parameter „option“ mit dem Wert „audioVorhanden“ wird die eingestellte Sprache für die Audio-Dateien ausgelesen (Zeile 8) und der dazugehörige Pfad erstellt (Zeile 12). Da die Datei noch nicht gefunden wurde, werden die Kon-

trollelemente zunächst ausgeblendet (Zeile 15) und der Hinweistext, dass keine Audio-Dateien vorhanden sind/heruntergeladen wurden, eingeblendet (Zeile 16).

Mit dem Parameter „dateiname“ wurde die verknüpfte Datei übergeben. Diese wird nun im Ordner (Zeile 12) der eingestellten Sprache gesucht (Zeile 17 bis 35). Mit einem „directoryReader“ (Zeile 20) werden die Inhalte des Ordner gelesen. Zur Suche werden die Dateinamen der heruntergeladenen Dateien mit dem Parameter „dateiname“ verglichen (Zeile 23-35). Zusätzlich wird dem Kontrollelement „Play“ das Attribut „onclick“ mit dem Wert „tphAudioAbspielen(tphAudioPfad)“ hinzugefügt (Zeile 30). Der Parameter „tphAudioPfad“ enthält dabei den absoluten Pfad zur Audio-Datei (Zeile 28). Daraufhin wird der Hinweistext ausgeblendet und alle Kontrollelemente eingeblendet (Zeile 31 und 32).

```

1  [...]
2  function tphDateisystem(option, dateiname) {
3      window.requestFileSystem(LocalFileSystem.PERSISTENT, 0, function(filesystem) {
4          switch (option) {
5              [...]
6              case 'audioVorhanden':
7                  // Holt die eingestellte Sprache für Audio-Dateien
8                  var tphSpracheAudio = tphHoleSpracheAudio();
9                  // Annahme: Audio-Datei ist nicht vorhanden
10                 var tphAudioVorhanden = false;
11                 // Variable für den Pfad zur Datei
12                 var tphAudioPfad = 'Hude/audio/' + tphSpracheAudio + '/';
13                 //alert('audioVorhanden ' + dateiname);
14                 // Überprüfen, ob Datei im Dateisystem vorhanden ist
15                 $('tphPlayerControl').hide();
16                 $('tphPlayerKeineDateien').show();
17                 filesystem.root.getDirectory(tphAudioPfad, {create: false, exclusive: false},
18                     function(directory) {
19                     console.log('Ordner Hude auslesen: ' + tphAudioPfad);
20                     // Directory reader initialisieren
21                     var directoryReader = directory.createReader();
22                     // Liste aller Einträge im Ordner ausgeben:
23                     directoryReader.readEntries(function(entries) {
24                         for (var i = 0; i < entries.length; i++) {
25                             // Audio Datei vorhanden
26                             if (entries[i].name === dateiname) {
27                                 // setzen der Variable auf true
28                                 tphAudioVorhanden = 'true';
29                                 tphAudioPfad = entries[i].fullPath;
30                                 // Play-Button hinzufügen
31                                 $('tphPlayButton').html('<a id="tphPlayButton" href="#" "
32                                     onclick="tphAudioAbspielen(\\grqq{ } + tphAudioPfad + '\\)"> </a>');
34                                 $('tphPlayerControl').show();
35                                 $('tphPlayerKeineDateien').hide();
36                                 //alert('Gefunden');
37                             }
38                         }
39                     }, function() {
40                         $('#consolelog').append('<p>Fehler Directory Reader</p>');
41                     });
42                     if (tphAudioVorhanden === true || tphAudioVorhanden === 'true') {
43                         console.log(tphAudioPfad);
44                         $('#consolelog').append('<p>' + tphAudioPfad + '</p>');
45                     }
46                 }, function() {
47                     console.log('Fehler beim Auslesen von "Hude/audio"');
48                     $('#consolelog').append('<p>Hude/audio Zugriff fehlgeschlagen</p>');
49                 });
50                 break;
51             }
52         }
53     }
54 }
55 [...]

```

Abbildung 6.29.: Überprüfung ob eine Audio-Datei vorhanden ist

6. Implementierung

Die Implementierung der Kontrollelement-Funktionen wird in Abbildung 6.30 dargestellt. Zur Wiedergabe einer Audio-Datei muss der Funktion „tphAudioAbspielen(file)“ als Parameter der Pfad zur Datei übergeben werden. Die Audio-Datei wird als Medien-Datei einer Variable zugewiesen (Zeile 8) und wird mit der Funktion „play()“ der Medien-Klasse abgespielt (Zeile 8). Die Zeilen 13-27 geben die aktuelle Position der Wiedergabe aus. Die Wiedergabe einer Audio-Datei kann mit der Funktion „tphAudioPause“ (Zeile 38-42) pausiert oder durch die Funktion „tphAudioStoppen“ (Zeile 45-53) abgebrochen werden.

```
1  [...]
   var audio = null;
3  var audioTimer = null;
   var pausePos = 0;
5
   function tphAudioAbspielen(file) {
7     console.log(file);
     audio = new Media(file, function() {
9         console.log("playAudio():Audio Success");
     }, function(error) {
11        console.log('code: ' + error.code + ' fehler: ' + error.message);
     });
13    // Aktuelle Position der Wiedergabe
     var duration = audio.getDuration();
15    //$('#tphAudioDauer').html('<span id="tphAudioDauer">' + duration + '</span>');
     // Abspielen der Audio-Datei
17    audio.play();
     audio.seekTo(pausePos * 1000);
19    // Sekündliche Abfrage der Position der Audio-Datei
     if (audioTimer === null) {
21        audioTimer = setInterval(function() {
           // Abfrage der Position
23            audio.getCurrentPosition(
               function(position) {
25                if (position > -1) {
                   console.log('Position: ' + position);
27                }
               }, function(e) {
29                    console.log("Error: " + e);
                   //setAudioPosition(duration);
31                }
               );
33            }, 1000);
     }
35 }

37 // Pausiert die gestartete Audio-Datei
   function tphAudioPause() {
39     if (audio) {
         audio.pause();
41     }
     }
43
   // Stoppt die gestartete Audio-Datei
45   function tphAudioStoppen() {
       if (audio) {
47           audio.stop();
           audio.release();
49       }
       clearInterval(audioTimer);
51       audioTimer = null;
       pausePos = 0;
53   }

55 [...]
```

Abbildung 6.30.: Funktionen der Kontrollelemente des Audio-Players

Navigation

Zur Navigation innerhalb der App wird zum einen das jQuery-Plugin „Hashchange“ (Abschnitt 6.1.1 auf Seite 52) genutzt, zum anderen HTML-Anker-Tags (<a>). Dem Anker-Tag wird das Attribut „href“ hinzugefügt. Diesem Attribut wird als Wert ein Hash zugewiesen, der die spätere Zielseite repräsentiert. Der Hash trägt immer den Dateinamen der Zielseite abzüglich der Endung „.html“.

Wird beispielsweise der Anker in Abbildung 6.31 gedrückt, registriert das Hashchange-Plugin den Wechsel des Hashes von „index.html“ auf „index.html#tphSeiteHudeInfo“. Dadurch wird das JS-Skript (Abbildung 6.32), aufgerufen, welches sich, da es die Hauptseite ist (siehe dazu Abschnitt 6.3.2 auf Seite 62), in der index.html befindet.

```
1 <a href="#tphSeiteHudeInfo">Hude-Info</a>
```

Abbildung 6.31.: Navigation in der App mittels Anker-Tag

Das JS-Skript in Abbildung 6.32 entfernt (Zeile 3) das Doppelkreuz (#) vor dem Hash und fügt dem Hash einen Punkt („.“) an. Aus dem modifizierten Hash wird in Zeile 4 ein Dateiname mit Endung „.html“ erzeugt. Im dargestellten Beispiel lautet der Dateiname „tphSeiteHudeInfo.html“. Diese Datei wird in Zeile 8 aufgerufen und in den <div>-Container mit der Klasse „tphContent“ eingefügt, welcher sich auf der Hauptseite (Zeile 42, Abbildung 6.11 auf Seite 65) befindet. Beim Anzeigen der neuen Seite wird, falls eine Audio-Datei abgespielt wird, diese gestoppt (Zeile 6 Abbildung 6.32).

```
1 $(window).hashchange(function() {
2     var hash = location.hash;
3     hash = (hash.replace(/^#/,\grqq{}) || 'blank') + '.';
4     var datei = hash + 'html';
5     if (hash !== 'blank.') {
6         tphAudioStoppen();
7     }
8     $('tphContent').load(datei);
9 });
```

Abbildung 6.32.: Hashchange-Event und Aufruf der neuen Seite

QR-Code-Scans

Um den QR-Code-Scanner nutzen zu können, muss das BarcodeScanner-Plugin (siehe Abschnitt 6.1.3 auf Seite 54) zuvor über die „config.xml“ eingebunden werden (siehe Zeile 70 der Abbildung 6.6 auf Seite 59). Ein Aufruf des QR-Code-Scanners erfolgt über die JS-Funktion „tphQRCodeScan()“, die auf das API des PhoneGap-Frameworks (siehe Abschnitt 6.1.3 auf Seite 53) und des BarcodeScanner-Plugins zugreift. Die Funktion wird in Zeile 1-17 der Abbildung 6.33 dargestellt.

In einem try-catch-Block wird versucht den QR-Code-Scanner aufzurufen. Der QR-Code-Scanner wird in Zeile 3 instanziiert. Durch den Aufruf der „scan()“-Funktion in Zeile 4 wird versucht, den QR-Code-Scanner zu starten. Gelingt dies nicht, wird der Anwender auf eine Fehlerseite umgeleitet (Zeile 13-16). Nach dem erfolgreichen Starten ist der BarcodeScanner bereit QR-Codes einzulesen. Wird ein QR-Code erfolgreich gelesen, wird dieser an die Funktion „tphSplitURL(url)“ (Zeile 19-30) übergeben. Tritt ein Fehler beim Lesen auf, wird der Anwender ebenfalls auf eine Fehlerseite umgeleitet.

Die QR-Codes, die für die Touristik-Palette-Hude erstellt wurden, erhalten lediglich einen URI in Verbindung mit einem Hash (siehe dazu Abschnitt 6.3.3 auf Seite 89). Dieser wird der Funktion „tphSplitURL(url)“ als Parameter übergeben und in dieser verarbeitet. Enthält der gescannte QR-Code nicht die Adresse „http://m.touristik-palette-hude.de/tphSeiteIndex.html“, wird er als ungültig gewertet. Der Nutzer wird daraufhin auf eine Seite mit dem Hinweis umgeleitet, dass ein ungültiger QR-Code gescannt wurde

6. Implementierung

(Zeile 26-29). Dies wird zum Zweck der Sicherheit eingehalten, um zu gewährleisten, dass am Gerät keine Fehler/Beschädigungen durch schadhafte QR-Codes entstehen können. Enthält der gescannte QR-Code die Adresse „http://m.touristik-palette-hude.de/tphSeiteIndex.html“ und zusätzlich einen Hash, so wird die Adresse und das # des Hashes entfernt (Zeile 22-23). Dem Hash wird „.html“ angefügt (Zeile 24) und die Seite aufgerufen (Zeile 25).

```
1  function tphQRCodeScan() {
   try {
3     var scanner = cordova.require("cordova/plugin/BarcodeScanner");
       scanner.scan(
5         function(result) {
           tphSplitURL(result.text);
7         },
           function(error) {
9             var datei = 'tphSeiteQRCodeFehler.html';
               $(' .tphContent').load(datei);
11          }
           );
13 } catch (exception) {
       var datei = 'tphSeiteQRCodeFehler.html';
15     $(' .tphContent').load(datei);
       }
17 }

19 function tphSplitURL(url) {
   var domain = 'http://m.touristik-palette-hude.de/tphSeiteIndex.html';
21   if (url.indexOf(domain) !== -1) {
       url = url.replace(domain, \grqq{}).trim();
23     var hash = (url.replace(/^#/ , \grqq{})) || 'blank' + '.';
       var datei = hash + 'html';
25     $(' .tphContent').load(datei);
   } else {
27     var datei = 'tphSeiteQRCodeUnguelteig.html';
       $(' .tphContent').load(datei);
29   }
   }
```

Abbildung 6.33.: Start des BarcodeScanners und Verarbeitung des gescannten QR-Code

Schnitzeljagd

Besonders für Kinder wurde die Schnitzeljagd implementiert. In dieser sind zwei unterschiedliche Spiele zu finden. Zum einen das „Planetenabitur“, das Fragen rund um den Planetenlehrpfad - in drei verschiedenen Schwierigkeitsstufen („Aufgepasst“, „Nachgelesen“ und „Gewusst“) - bereit hält. Zum anderen wurde die „Fotojagd“ implementiert, bei der, in drei verschiedenen Bereichen in Hude, auf einem Foto abgebildete Objekte gefunden werden müssen. Ob ein Objekt gefunden wurde, wird durch den Abgleich der GPS-Koordinaten des eigenen Standorts und der GPS-Koordinaten des Objekts bestimmt.

Planetenabitur

Das Planetenabitur, in den Varianten A, B und C, ist die Digitalisierung des bereits bestehenden Planetenabitur aus dem Flyer „Planeten-Lehr-Pfad“. Die Abbildung 6.34 zeigt einen Ausschnitt, wie Fragen auf der Seite implementiert wurden. Die Abbildung 6.35 zeigt das Resultat des Planetenabiturs.

Für jede Frage wurde ein Fieldset mit dem Attribut „data-role“ und dem Wert „controlgroup“ (Zeile 1-15) erzeugt. Dies fasst die Frage sowie die Antworten zu einem Block zusammen.

Die Frage wird vom <legend>-Tag umschlossen und wird dadurch dem Rahmen des <fieldset> hinzugefügt. Die ID des <legend>-Tag identifiziert die Frage eindeutig. Für das Planetenabitur baut sich diese ID aus folgenden Zeichenketten auf:

- plpAbi
- Festlegung des Abiturs (A, B oder C)
- Nummer der Frage (01, 02, ...)

Für das Beispiel in Abbildung 6.34 entsteht somit die ID und der Name „plpAbiA01“.

Die Antworten werden jeweils innerhalb eines <label>-Tags definiert, das das „for“-Attribut trägt. Des- sen Wert entspricht einer ID eines <input>-Tag und wird dadurch diesem zugewiesen. Dieses <input>-Tag trägt zum einen das Attribut „name“ das den Wert der ID vom <legend>-Tag annimmt und dadurch die Antwortmöglichkeit der Frage zuweist. Zur eindeutigen Identifizierung einer Antwortmöglichkeit, wird die <legend>-Tag-ID für jede Antwort um ein Zeichen aus dem Alphabet erweitert (a, b, c, ...) und als ID dem <input>-Tag zugewiesen. Das „value“-Attribut im <input>-Tag trägt das Ergebnis einer Antwortmög- lichkeit. Diese kann „falsch“ oder „richtig“ sein. Mit dem Attribut „type“ und dem Wert „radio“ ist nur eine Antwortmöglichkeit pro Frage auswählbar.

Dem Absatz in Zeile 19 wird, nach Abschluss der Auswertung aller Fragen, das Ergebnis angehängt. Dabei wird angegeben wie viele Frage von allen insgesamt gestellten Fragen des jeweiligen Planetenabiturs, richtig beantwortet wurden. Falsch beantwortete Fragen werden rot hinterlegt, richtig beantwortete Fragen grün.

```

<fieldset data-role="controlgroup">
2  <legend id="plpAbiA01">Welchen Durchmesser hat die Sonnenkugel auf dem
    Lehr-Pfad?</legend>
    <label for="plpAbiA01a">
4     <input name="plpAbiA01" id="plpAbiA01a" value="falsch" type="radio">
        13 cm
6    </label>
    <label for="plpAbiA01b">
8     <input name="plpAbiA01" id="plpAbiA01b" value="richtig" type="radio">
        139 cm
10   </label>
    <label for="plpAbiA01c">
12    <input name="plpAbiA01" id="plpAbiA01c" value="falsch" type="radio">
        13 mm
14   </label>
</fieldset>
16
[... ]
18
<p id="tphSchnitzeljagdPlanetenlehrpfadAbiturAErgebnis"></p>
20 <a onclick="tphSchnitzeljagdPlanetenlehrpfadAbiturA();" class="large button
    expand">Auswerten</a>

```

Abbildung 6.34.: Ausschnitt der HTML zur Implementierung des Planetenabiturs A

Die Auswertung des Abiturs erfolgt durch das Drücken des Buttons in Zeile 20 und das damit verbundene JS-Skript in Abbildung 6.36.

Zur Auswertung einer einzelnen Frage wurde die Funktion „tphUeberpruefeMultipleChoice(name)“ (Zeile 1) geschrieben, die die ID bzw. den Namen (identisch) der Frage als Parameter übergeben bekommt. Die Schriftfarbe der Frage wird zunächst auf weiß geändert (Zeile 2). Anschließend wird überprüft ob das „value“-Attribut der ausgewählten Antwortmöglichkeit den Wert „richtig“ trägt. Ist dies der Fall, so wird die Frage grün hinterlegt (Zeile 4) und 1 als Wert zurückgegeben (Zeile 5). Entspricht der Wert nicht „richtig“ wird die Frage rot hinterlegt (Zeile 7) und eine 0 zurückgegeben (Zeile 8).

Für jedes Planetenabitur wurde zur Rückgabe des Ergebnis’ eine eigene Funktion implementiert. Zunächst wird die Variable „richtig“ mit „0“ initialisiert (Zeile 13), so dass keine Frage als richtig beantwortet gilt. Eine Frage wird durch die Funktion „tphUeberpruefeMultipleChoice(name)“ zur Auswertung hinzugefügt (Zeile 14 und 18). Der Parameter „name“ entspricht der ID bzw. dem Namen der Frage (identisch). Der

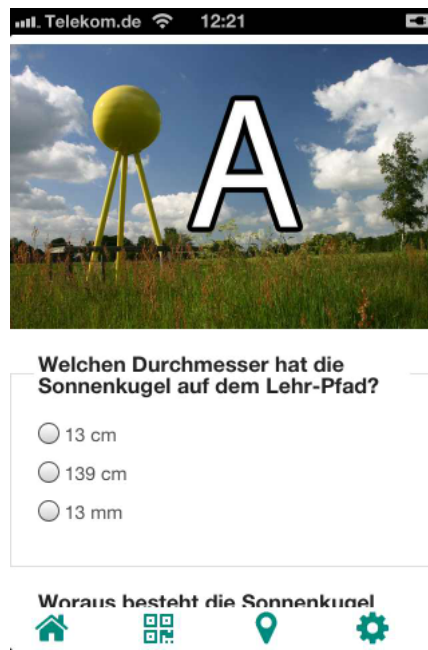


Abbildung 6.35.: Resultat des Planetenabiturs

Rückgabewert der Funktion erhöht die Variable „richtig“ entweder um 0 oder 1. Nach der Auswertung aller Fragen wird das Ergebnis, Wert der Variable „richtig“, im Verhältnis zu den gestellten Fragen (Zeile 19), dem Absatz in Zeile 19 der Abbildung 6.34 hinzugefügt.

```

function tphUeberpruefeMultipleChoice(name) {
2   $('#' + name).css('color', 'white');
   if ($('#input[name=' + name + ']:checked').val() === 'richtig') {
4     $('#' + name).css('background-color', 'green');
       return 1;
6   } else {
       $('#' + name).css('background-color', 'red');
8     return 0;
   }
10 }

12 function tphSchnitzeljagdPlanetenlehrpfadAbiturA() {
   var richtig = 0;
14   richtig += tphUeberpruefeMultipleChoice('plpAbiA01');

16   [...]

18   richtig += tphUeberpruefeMultipleChoice('plpAbiA31');
   $('#tphSchnitzeljagdPlanetenlehrpfadAbiturAErgebnis').html('<p
       id="#tphSchnitzeljagdPlanetenlehrpfadAbiturAErgebnis"><h1>Sie haben ' + richtig +
       ' von 31 Fragen richtig beantwortet!</h1></p>');
20 }

```

Abbildung 6.36.: Auswertung des Planetenabiturs

Fotojagd

Ziel der Fotojagd ist es, ein vorgegebenes, fotografiertes Objekt (Sehenswürdigkeit, Skulptur, Tafel, usw.) während des Besuches in Hude zu finden bzw. wiederzuerkennen und den Standort des Objektes aufzusuchen. Meint man das Objekt gefunden zu haben, markiert man das Bild als „gefunden“. Ob das Objekt

wirklich gefunden wurde, wird anhand des Abstands zum Objekt errechnet. Benötigt werden dazu die GPS-Koordinaten des eigenen Standorts und die GPS-Koordinaten des Objektes, als Exif²¹-Datum innerhalb des Bildes. Für die Fotojagd wurden die drei Bereiche „Hude (insgesamt)“, „Klosterbezirk“ und „Skulpturenufer“ ausgewählt.

Für jeden Bereich wurde eine eigene Seite eingerichtet. Abbildung 6.37 zeigt einen Ausschnitt der Seite für den Bereich „Hude“.

Die Einbindung jedes Bildes erfolgt nach dem Muster in Zeile 2-6, sowie Zeile 11-15. Ein `<div>`-Container umgibt einen `<a>`-Tag, der einen ``-Tag enthält.

Der `<div>`-Container enthält die Klassen „large-6“, „small-6“ und „columns“, was zur Folge hat, dass immer zwei Bilder nebeneinander platziert werden.

Der `<a>`-Tag trägt die Attribute „href“ mit dem Wert des relativen Pfades zum Bild und „class“ mit den Klassen „fancybox“ und „fotojagd“. Durch die Zuweisung der Klasse „fancybox“ wird das Bild in einer Lightbox geöffnet (siehe Abschnitt 6.1.1 auf Seite 51), in der das Bild als gefunden markiert werden kann. Die Klasse „fotojagd“ identifiziert das Bild als zugehöriges Bild einer Fotojagd.

Der ``-Tag erhält die Attribute „id“ und „src“ mit dem Wert des relativen Pfades zum Bild. Die „id“ jedes Bildes muss in allen Fotojagden einmalig sein. Aus diesem Grund sind die Ids nach dem folgenden Muster aufgebaut:

- tphFotojagd
- Name des Bereiches (Hude, Klosterbezirk oder Skulpturenufer)
- Nummer des Bildes.

In der Abbildung 6.37 sind dies „tphFotojagdHude1“ (Zeile 4) und „tphFotojagdHude10“ (Zeile 13).

```

[...]
```

```

2 <div class="large-6 small-6 columns">
  <a href="images/schnitzeljagd/fotojagd/hude/schnitzeljagd_fotojagd_hude_01.jpg"
    class="fancybox fotojagd">
4   
    </a>
6 </div>

8 [...]
```

```

10 </div>
<div class="large-6 small-6 columns">
12 <a href="images/schnitzeljagd/fotojagd/hude/schnitzeljagd_fotojagd_hude_10.jpg"
    class="fancybox fotojagd">
    
14 </a>
</div>
16 <script>
$(function() {
18   tphHeaderUberschriftAendern('Fotojagd Hude');
  });
20 </script>
```

Abbildung 6.37.: Ausschnitt Fotojagd „Hude“

Jeder Fotojagd-Seite wird mit dem JS-Skript „tphSpeicherFotojagdBilderImLocalStorage()“ (Abbildung 6.38) durchsucht. Diese Funktion untersucht alle `<a>`-Tags, die einen ``-Tag umschließen (Zeile 6)

²¹Exchangeable Image File Format (Exif) ist ein Datenformat zum Speichern von Metadaten im Header eines Bildes. Diese Metadaten können zum Beispiel die Belichtungszeit, das Kameramodell oder die Positionsdaten umfassen.

und die Klasse „fotojagd“ tragen (Zeile 8). Die id des -Tag wird, sofern diese noch nicht vorhanden ist, im localStorage gespeichert und auf „false“ (nicht gefunden) gesetzt (Zeile 15).

Um ein Objekt als „Gefunden“ zu markieren muss das Bild auf der Fotojagd-Seite berührt werden. Dadurch öffnet sich eine durch fancyBox generierte (Zeile x) Lightbox. Diese enthält das Bild, sowie einen Button mit der Aufschrift „Gefunden!“. Der Button ruft beim Drücken die Funktion „tphHoleGPSAusBild(imgID)“ auf. Der zu übergebende Parameter ist die Id des Bildes, die in Zeile 30 automatisch ausgelesen wird und in Zeile 31 dem Button übergeben wird.

Die Funktion „tphHoleGPSAusBild(imgID)“ nutzt das Plugin „Exif Reader“ (siehe Abschnitt 6.1.1 auf Seite 52) und liest die GPS-Koordinaten aus dem Bild aus (Zeile 44 und 45). Da diese Daten in einem 2-dimensionalen Array, Format „Grad, Minuten, Sekunde“, zurückgegeben werden, müssen diese Daten mit der implementierten Funktion „tphConvertDMStoDec(dmsArray)“ (Zeile 79-91) ins Dezimal-Format umgewandelt werden (Zeile 47-48).

Zeile 49 ruft das GPS-Modul des Gerätes auf. Darüber wird die aktuelle Position im Dezimalformat bestimmt (Latitude (Zeile 50) und Longitude (Zeile 51)). Durch den Aufruf (Zeile 52) der Funktion „tphGPS-Abstand(latGPS, lonGPS, latFotojagd, lonFotojagd)“ [Pon09] (Zeile 92-119), wird der Abstand zwischen der eigenen Position und der Position des sich auf dem Foto befindlichen Objektes bestimmt. Ist der Abstand als gering genug berechnet worden, wird die geöffnete Lightbox geschlossen (Zeile 57), das Bild auf der Seite der Fotojagd ausgeblendet (Zeile 61) und im localStorage anhand der Id auf „true“ (gefunden) gesetzt (Zeile 63-64). Anschließend wird eine neue Lightbox geöffnet, die das Bild noch einmal anzeigt. Die Bildunterschrift zeigt „BILD GEFUNDEN“ an (Zeile 66-48). Die Bewertung der bereits gefundenen und aller Bilder der Fotojagd wird um „1“ erhöht (Zeile 69-70).

Eine Fotojagd kann jederzeit neu gestartet werden. Dazu werden alle Ids eines Fotojagds-Bereichs durch die Funktion tphSetzeFotojagdBilderAufNichtGefunden() auf „false“ gesetzt. Die Funktion untersucht zu diesem Zweck alle <a>-Tags die einen -Tag umschließen und die Klasse „fotojagd“ tragen (Zeile 131-134). Die Id dieser Bilder wird innerhalb eines Arrays gespeichert (Zeile 137) und auf der Seite wieder angezeigt (Zeile 139). Nachdem alle Ids dem Array hinzugefügt wurden, werden diese Ids im localStorage auf „false“ (nicht gefunden) gesetzt. Das Ergebnis auf der Seite wird ebenso zurückgesetzt, so dass 0 Bilder als gefunden angezeigt werden.

```

1  function tphSpeicherFotojagdBilderImLocalStorage() {
2    console.log('tphSpeicherFotojagdBilderImLocalStorage');
3    // localStorage initialisieren
4    var tphStorage = tphLadeLocalStorage();
5    // id des Bildes als Index für Fotojagd benutzen
6    $('a:has(img)').each(function() {
7      // Wenn der Anker die Klasse 'fotojagd' enthält
8      if ($(this).hasClass('fotojagd')) {
9        // Und diese ID noch nicht im localStorage gesetzt ist
10       if (tphStorage.getItem($(this).find('img').attr('id')) === null) {
11         /*
12          * Wird diese ID im localStorage als false (nicht gefunden)
13          * gespeichert
14          */
15         tphStorage.setItem($(this).find('img').attr('id'), false);
16       }
17     }
18   });
19 }
20
21 function tphSetzeEinstellungenAufSeite() {
22   [...]
23
24   // Fotojagd initialisieren
25   $("a.fotojagd").each(function() {
26     $(this).fancybox({
27       openEffect: 'none',
28       closeEffect: 'none',
29       title: function() {

```

```

30     var imgID = $(this).find('img').attr('id');
31     return '<a onclick="tphHoleGPSAusBild(\`grqq\` + imgID + '\`')">Gefunden!</a>';
32   }
33   });
34 });
35
36 [...]
37
38 }
39
40 function tphHoleGPSAusBild(imgID) {
41   var image = document.getElementById(imgID);
42   EXIF.getData(image, function() {
43     // GPS-Daten aus dem Bild auslesen
44     var latFotojagd = EXIF.getTag(this, "GPSLatitude");
45     var lonFotojagd = EXIF.getTag(this, "GPSLongitude");
46     // GPS-Daten von Grad, Minute, Sekunde ins Dezimale umrechnen
47     latFotojagd = tphConvertDMStoDec(latFotojagd);
48     lonFotojagd = tphConvertDMStoDec(lonFotojagd);
49     navigator.geolocation.getCurrentPosition(function(position) {
50       var latGPS = position.coords.latitude;
51       var lonGPS = position.coords.longitude;
52       var gpsAbstand = tphGPSAbstand(latGPS, lonGPS, latFotojagd, lonFotojagd);
53       // Abstand zum Objekt ist ok
54       if (gpsAbstand === true) {
55         var img = $('#'+ imgID);
56         // Schließen der Bildanzeige
57         $.fancybox.close();
58         // Pfad des Bildes holen
59         var imgSrc = img.attr('src');
60         // Bild ausblenden
61         img.fadeOut();
62         // Pfad im localStorage als gefunden setzen
63         var tphStorage = tphLadeLocalStorage();
64         tphStorage.setItem(imgID, true);
65         // Box mit Bild gefunden
66         $.fancybox.open(
67           {href: imgSrc, title: '<h4 style="color: white">BILD GEFUNDEN!</h4>'}
68         );
69         var anzahlBilderGefunden = parseInt($('#tphAnzahlBilderGefunden').text() + 1);
70         $('#tphAnzahlBilderGefunden').text(anzahlBilderGefunden);
71       }
72     }, function(error) {
73       console.log('KEIN GPS');
74     }, {maximumAge: 0, timeout: 15000, enableHighAccuracy: true});
75   });
76 }
77
78 function tphConvertDMStoDec(dmsArray) {
79   var DEG = dmsArray[0]['numerator'];
80   var MIN = dmsArray[1]['numerator'];
81   var SEC = dmsArray[2]['numerator'];
82   var DEC = \grqq{};
83   if (DEG < 0) {
84     DEC = (Math.ceil(((DEG * 1) - (MIN / 60) - (SEC / 3600)) * 10000)) / 10000;
85   }
86   else {
87     DEC = (Math.floor(((DEG * 1) + (MIN / 60) + (SEC / 3600)) * 10000)) / 10000;
88   }
89   return DEC;
90 }
91
92 function tphGPSAbstand(lat1, lon1, lat2, lon2) {
93   var R = 6371;
94   var dLat = (lat2 - lat1) * Math.PI / 180;
95   var dLon = (lon2 - lon1) * Math.PI / 180;
96   var a = Math.sin(dLat / 2) * Math.sin(dLat / 2) + Math.cos(lat1 * Math.PI / 180) *
97     Math.cos(lat2 * Math.PI / 180) * Math.sin(dLon / 2) * Math.sin(dLon / 2);
98   var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));
99   var d = R * c;

```

6. Implementierung

```
100     if (d > 1) {
101         //return false; // Entfernung = Kilometer
102         return Math.round(d) + "km";
103     }
104
105     if (d <= 1) {
106         // Meter berechnen
107         d = Math.round(d * 1000);
108         // Erlaubte Distanz zum Objekt
109         var tphErlaubteDistanz = 100;
110         // Objekt gefunden wenn die Distanz (d) zum Objekt kleiner oder gleich
111         // der erlaubten Distanz (tphErlaubteDistanz) ist.
112         if (d <= tphErlaubteDistanz) {
113             return true;
114         } else {
115             // Entfernung vom Objekt größer als die erlaubte Distanz
116             return d;
117         }
118     }
119 }
120
121 function tphSetzeFotojagdBilderAufNichtGefunden() {
122     // Bilder die als Parent einen Link mit der Klasse 'fotojagd' werden geladen
123     // LocalStorage initialisieren
124     var tphStorage = tphLadeLocalStorage();
125     /* Bilder der Schnitzeljagd auf Seite heraussuchen, dazu nur Bilder
126      * verwenden, die Kind eines Ankers mit der Klasse 'fotojagd'
127      */
128     var anzahlBilderInsgesamt = 0;
129     var anzahlBilderGefunden = 0;
130     var imgIDsZumLoeschen = new Array();
131     $('a:has(img)').each(function() {
132         if ($(this).hasClass('fotojagd')) {
133             anzahlBilderInsgesamt++;
134             var imgID = $(this).find('img').attr('id');
135             // ID dem Array hinzufügen, das im Anschluss die IDs im localStorage
136             // zurücksetzt
137             imgIDsZumLoeschen.push(imgID);
138             // Bilder wieder anzeigen
139             $('#' + imgID).show();
140         }
141     });
142     // IDs der Bilder im localStorage false setzen
143     for (var i = 0; i < imgIDsZumLoeschen.length; i++) {
144         tphStorage.setItem(imgIDsZumLoeschen[i], false);
145     }
146     $('#tphSchnitzeljagdFotojagdErgebnis').html('<div
147         id="tphSchnitzeljagdFotojagdErgebnis"><p><br/>Bereits gefunden <span
148         id="tphAnzahlBilderGefunden">' + anzahlBilderGefunden + '</span><span
149         id="tphAnzahlBilderInsgesamt">' + anzahlBilderInsgesamt + '</span> Bildern und
150         deren Position!</p></div>');
151 }
```

Abbildung 6.38.: JS-Implementierung zur Fotojagd

Veranstaltungen

Die Veranstaltungen werden aus der Datenbank des Veranstaltungskalenders der Website <http://gastinhude.de> ausgelesen. Zu diesem Zweck verbindet sich das PHP-Skript (siehe Abbildung 6.39) mit der Datenbank und liest die in der Zukunft liegenden Veranstaltungen aus. Anschließend werden die ausgelesenen Veranstaltungen in ein JSONP²²-Objekt umwandelt. Dieses Objekt wird per AJAX²³ abgerufen und nach der

²²JSON with padding (JSONP) ist ein Verfahren zur Datenübertragung mittels JavaScript mit der sogenannten „JavaScript Object Notation“ (JSON). Durch dieses Verfahren wird der Zugriff auf eine externe bzw. andere Domain ermöglicht.

²³Asynchronous JavaScript and XML (AJAX) bezeichnet ein Verfahren zur asynchronen Datenverarbeitung und -übertragung. So können beispielsweise Seiten aufgerufen werden, ohne eine gesamte Seite neu abrufen zu müssen.

Aufbereitung in der App mit jQuery eingefügt.

Server

```

1  <?php
function tph_connect() {
3  $host = 'm.touristik-palette-hude.de';
   $user = 'app';
5  $pass = '7D10C1A3';
   $db = 'tp';
7
   try {
9     $connect = mysql_connect($host, $user, $pass);
       if ($connect) {
11        mysql_select_db($db) or die();
       }
13    } catch (Exception $exc) {
       echo $exc->getTraceAsString();
15    }
   }

17 function tph_close() {
19     mysql_close();
   }

21 try {
23     // Verbindung zur Datenbank herstellen
   tph_connect();
25     // Aktuelles Datum bestimmen
   $timestamp = time();
27     $aktuelleDatum = date('Y-m-d', $timestamp);
   /*
29     * Anfrage an die Datenbank.
   * Abgefragt nur Veranstaltungen Datum >= heutiges Datum
   */
31     $query = "SELECT tp_eventlist_events.dates, tp_eventlist_events.enddates,
       tp_eventlist_events.times, tp_eventlist_events.title,
       tp_eventlist_events.datdescription FROM tp.tp_eventlist_events WHERE
       tp_eventlist_events.published = '1' AND dates >= '" . $aktuelleDatum . "' ORDER BY
       tp_eventlist_events.dates ASC";
33     $result = mysql_query($query) or die("Fehler: " . mysql_error());
   $veranstaltungen = array();
35     // Verarbeitung der gelieferten Datensätze
   while ($row = mysql_fetch_assoc($result)) {
37         if (isset($row['dates'])) {
           $datumStart = $row['dates'];
39           $datumEnd = $row['dates'];
       }
41         if (isset($row['enddates']) && $row['enddates'] != \grqq{}) {
           $datumEnd = $row['enddates'];
43         }
       // Entfernen von Links in der Beschreibung
45         $dataDescription = preg_replace("/\<a(.*)\>(.*?)\</a\>/iU", "$2",
           $row['datdescription']);
       // Entfernen von 'Weitere Infos ...'
47         $dataDescription1 = str_replace("Weitere Infos...", "", $dataDescription);
       // Euro-Zeichen ersetzen
49         $dataDescription2 = str_replace("E", "&euro;", $dataDescription1);
       // Unsichtbare Leerzeichen entfernen
51         $dataDescription3 = str_replace("&nbsp;", "", $dataDescription2);
       // Füllen des Arrays für die JSONP-Rückgabe
53         $veranstaltungen[] = array(
           'datumStart' => utf8_encode(date("d.m.Y", strtotime($datumStart))),
55           'datumEnd' => utf8_encode(date("d.m.Y", strtotime($datumEnd))),
           'uhrzeit' => utf8_encode($row['times']),
57           'titel' => utf8_encode($row['title']),
           'beschreibung' => utf8_encode($dataDescription2),
59         );
   }

```

6. Implementierung

```
61 // Rückgabe der Datenbankabfrage via JSONP
    echo $_GET['jsonp_callback'] . '(' . json_encode($veranstaltungen) . ');';
63 // Trennen der Datenbankverbindung
    tph_close();
65 } catch (Exception $exc) {
    echo $exc->getTraceAsString();
67 }
?>
```

Abbildung 6.39.: Abfrage in der Zukunft liegender Veranstaltungen

App

Zur Darstellung der aus der Datenbank ausgelesenen Daten werden die Seite „tphSeiteVeranstaltungen.html“ (Abbildung 6.40) und das JS-Skript tphLadeVeranstaltungen() (Abbildung 6.41) benötigt. Das JS-Skript ruft mittels AJAX das PHP-Skript (Abbildung 6.39 auf Seite 98) auf dem Server der Touristik-Palette-Hude auf. Ist der Aufruf erfolgreich und die Daten sind als JSONP-Objekt zurückgeliefert, werden die einzelnen Datensätze in HTML-Elemente eingefügt, die zunächst alle in einer Variable gespeichert werden. Sind alle JSONP-Objekt-Elemente durchlaufen, werden die HTML-Elemente, die sich in der Variable „append“ befinden, dem Absatz in Zeile 2 der Abbildung 6.40 hinzugefügt und sind sichtbar.

```

2 <p id="tphVeranstaltungen"><p>
  <script>
4 $(function() {
    tphHeaderUberschriftAendern('Veranstaltungen');
6    tphLadeVeranstaltungen();
    tphSetzeEinstellungenAufSeite();
8  });
  </script>
```

Abbildung 6.40.: Aufbau der Seite für Veranstaltungen

```
1 function tphLadeVeranstaltungen() {
    $.ajax({
3      // Datentyp festlegen
      dataType: 'jsonp',
5      // Callback Name festlegen
      jsonp: 'jsonp_callback',
7      // Server festlegen der abgefragt werden soll
      url: 'http://m.touristik-palette-hude.de/tph/tphserver.php',
9      // Erfolgreiche Abfrage:
      success: function(data) {
11     // Variable die mit HTML-Elementen gefüllt wird
      var append = \grqq{};
13     // Durchlauf durch das JSONP-Objekt
      for (var i = 0; i < data.length; i++) {
15       // Container für jede Veranstaltung erzeugen
      append += '<div class="row sehenswuerdigkeit">';
17       append += '<div class="large-12 small-12 columns">';
      append += '<em>';
19       // Zeitraum der Veranstaltung einfügen
      // Veranstaltungsstart == Ende
21       if (data[i]['datumEnd'] === data[i]['datumStart']) {
      append += data[i]['datumStart'];
23     } else {
      // Veranstaltungsstart != Ende
25       append += data[i]['datumStart'] + ' - ' + data[i]['datumEnd'];
```



```

27     }
    // Einfügen wenn Uhrzeit angegeben wurde
    if (data[i]['uhrzeit'] !== \grqq{}) {
29         append += ', ' + data[i]['uhrzeit'];
    }
31     append += '</em><br/>';
    // Titel der Veranstaltung einfügen
33     append += '<strong>' + data[i]['titel'] + '</strong>';
    // Beschreibung der Veranstaltung einfügen
35     append += '<p>' + data[i]['beschreibung'].replace(/p.P/g, "&euro;
        p.P").replace(/&nbsp;/g, "") + '</p>';
    append += '</div>';
37     append += '</div>';
    }
39     // Aufbereitete Daten in den Absatz \grqq{}#tphVeranstaltungen\grqq{} einfügen
    $(' #tphVeranstaltungen' ).append(append);
41 },
    // Keine erfolgreiche Abfrage
43 error: function(XHR, textStatus, errorThrown) {
    console.log("ERREUR: " + textStatus);
45     console.log("ERREUR: " + errorThrown);
    }
47 });
    $(' &nbsp;' ).remove();
49 }

```

Abbildung 6.41.: Skript zum Einfügen der Veranstaltungen

7. Evaluation

Im Kapitel Evaluation werden die vier Kriterien Brauchbarkeit, Nutzbarkeit, Erfüllung der Anforderungen sowie Richtigkeit überprüft um eine Evaluation der App durchzuführen. Zunächst wurde die Brauchbarkeit und Nutzbarkeit evaluiert. Anschließend die Erfüllung der Anforderung und abschließend die Evaluation der Richtigkeit.

7.1. Brauchbarkeit und Nutzbarkeit

Zur Überprüfung der Brauchbarkeit und Nutzbarkeit wurde die App auf die „Eight Golden Rules of Interface Design“ von Ben Shneidermann hin untersucht, sowie durch den SuS-Score bewertet. Zu diesem Zweck wurden Anwendertests vorgenommen und die Anwendern entsprechende Fragebögen vorgelegt.

7.1.1. Shneiderman

Ben Shneidermann ist ein amerikanischer Informatiker, der - zusammen mit Isaac Nassi - die „Nassi-Shneidermann-Diagramme“ (DIN 66261) zur grafischen Beschreibung von Abläufen innerhalb von Programmen entwickelte. Zudem leistete er im Bereich der Konzipierung von Schnittstellen zur Mensch-Maschine-Interaktion grundlegende Arbeiten [Wik05]. Aus seinen Entwicklungen gingen auch die „Eight Golden Rules of Interface Design“ [Shn01] hervor (Abbildung 7.1).

- 1. Konsistenz** Die Benutzeroberfläche soll ein möglichst konsistentes Aussehen aufweisen. Funktionen, die miteinander verwandt sind (z.B. „weiter“ und „zurück“), sollten im gesamten System vorhanden sein, immer gleich heißen und dieselbe Aktion ausführen.
- 2. Abkürzungen** Dem erfahrenen Benutzer soll mit Abkürzungen die Interaktion erleichtert werden. Dazu zählen Shortcuts (wie z.B. Strg + A), Funktionstasten (z.B. die „Home“-Taste auf dem Smartphone) oder die Erstellungen von Makros.
- 3. Feedback** Für jede gemachte Eingabe soll das System eine Rückmeldung geben. Bei häufig gemachten Eingaben kann die Rückmeldung kleiner ausfallen als bei seltenen Eingaben.
- 4. Dialoge** Aktionen, die durchgeführt werden sollen, sollten in Gruppen organisiert sein. Ein Feedback nach dem Abschluss einer Interaktion gibt ein Gefühl einer ordnungsgemäßen Durchführung, so dass mit gutem Gewissen weitere Interaktionen durchgeführt werden können.
- 5. Fehlerbehandlung** Das System soll so gestaltet sein, dass keine Fehler produziert werden können. Wird trotzdem ein Fehler generiert, soll eine Behebung durch den Anwender möglich sein.
- 6. Stornierung** Jede Aktion, die durchgeführt werden kann, soll auch rückgängig gemacht werden können.
- 7. Benutzerkontrolle** Anwender sollen das Gefühl der „Macht“ über das System haben. Das System soll dem Anwender keine Vorschriften machen, sondern auf die Eingaben des Anwenders reagieren.
- 8. Kurzzeitgedächtnis** Das Kurzzeitgedächtnis eines Anwenders ist zeitlich begrenzt. Deshalb sollten Informationen die Merkkapazität benötigen, sowie Interaktionen über mehrere Fenster und Seiten vermieden werden.

Abbildung 7.1.: Eight Golden Rules of Interface Design [Shn01]

Anhand der von Shneidermann entworfenen Regeln wird eine Evaluation der Oberfläche durchgeführt und eine Bewertung abgegeben.

Konsistenz

Durch eine, für die gesamte App gültige, CSS-Datei, die Trennung von der Hauptseite (siehe Abschnitt 6.3.2 auf Seite 62) als Gerüst und durch die Inhaltsseiten ist ein konsistentes Design gegeben. So befindet sich auf jeder Seite der Header, der die Überschrift der aktuellen Seite enthält und die Möglichkeit bietet, ein Panel zu öffnen, mit dem die Navigation innerhalb der App möglich ist. Auf einem iOS-Device, welches keinen Hardware-Zurück-Button enthält, ist dort ebenfalls ein Button mit der „Zurück“-Funktion implementiert. Innerhalb des Footers, der auf jeder Seite verfügbar ist, kann zur Startseite oder den Einstellungen navigiert werden. Ebenso kann die aktuelle Position abgerufen werden, sowie der QR-Code-Scanner gestartet werden.

In nahezu jeder Inhaltsseite sind die Inhalte innerhalb von Klappmenüs implementiert. Ausnahme bilden zwei Seiten. Zum einen ist dies die Seite „tphSeiteSehenswuerdigkeiten“, die eine Übersicht der Sehenswürdigkeiten in Hude enthält. Mit Bildern und einem kurzen Teaser-Text werde diese eingeführt, bevor dem Anwender über den Button „Mehr erfahren“ eine Seite mit Details zur Sehenswürdigkeit angezeigt wird. Die andere Ausnahme ist die Seite „tphSeitePosition“. Auf dieser wird, bei einer ausreichend guten Datenverbindung, eine Google-Maps-Karte eingeblendet, die die aktuelle Position darstellt oder, bei einer nicht ausreichenden Datenverbindung, die GPS-Koordinaten, Höhe und Richtung des Anwenders ausgibt.

Abkürzungen

Abkürzungen (Shortcuts) ermöglichen ein einfacheres und schnelleres Navigieren innerhalb eines Programms und vereinfachen den Umgang mit diesem. Auf einem Computer werden Shortcuts mit der Alt-Taste bzw. Strg-Taste in Verbindung mit einem weiteren Buchstaben aufgerufen. Beispielsweise kann „Alt + D“ das Datei-Menü des aktuellen Programms öffnen oder „Strg + C“ die aktuell geöffnete Datei speichern. Innerhalb der App werden jedoch keine Tastatur-Shortcuts unterstützt, da zu keinem Zeitpunkt die Tastatur aufgerufen wird und somit keine Interaktion mit dieser erfolgen kann.

Jedoch bietet die App zwei alternative Shortcuts. Zum einen kann jede Seite über einen QR-Code (siehe Abschnitt 6.3.3 auf Seite 89) direkt aufgerufen werden, so dass keine Navigation innerhalb der App notwendig ist. Ein Nachteil der QR-Codes ist jedoch, dass sie nur an bestimmten Plätzen angebracht und zu nutzen sind. Zum anderen ist ein weiterer Shortcut in Verbindung mit Telefonnummern implementiert. So ist es nicht notwendig, dass eine angegebene Telefonnummer kopiert und anschließend in die Telefon-App eingefügt werden muss, sondern diese direkt durch einen Klick aufgerufen wird und eine Verbindung hergestellt werden kann.

Feedback

Innerhalb der App nimmt das Feedback einen geringen Umfang ein, denn durch den Anwender selbst werden nahezu keine Eingaben getätigt. Eingaben erfolgen nur innerhalb der Einstellungen, des QR-Code-Scans und der einzelnen Schnitzeljagden.

In den Einstellungen erfolgt ein Feedback beim Download und Setzen von Einstellungen. Während des Downloads der Audio-Dateien wird dem Anwender angezeigt, dass der Download durchgeführt wird, und im Falle eines Abschlusses, dass der Download beendet wurde. Beim Setzen der Einstellungen erhält der Anwender Feedback, indem ihm angezeigt wird, welche Einstellungen gesetzt wurden.

Im Bereich der QR-Codes werden dem Anwender auf zwei Arten Feedback gegeben. Wurde ein QR-Code der Touristik-Palette-Hude eingescannt, wird dem Anwender Feedback durch das Öffnen des gescannten Inhalts gegeben. Wird ein fremder, nicht verifizierter QR-Code gescannt, wird der Anwender auf eine Fehler-Seite umgeleitet, auf der ein Hinweis über den nicht verifizierten QR-Code enthalten ist.

7. Evaluation

Bei den Schnitzeljagden des Planetenabiturs erhält der Anwender nach der Auswertung seiner Antworten Feedback. In diesem Feedback ist zunächst enthalten, wie viele Fragen richtig beantwortet wurden. Weiterhin sind die Fragen, die korrekt beantwortet wurden, grün hinterlegt und die Fragen, die falsch beantwortet wurden, rot hinterlegt. So kann der Anwender mittels Feedback nachvollziehen, welche Fragen richtig bzw. falsch beantwortet wurden.

Innerhalb der Schnitzeljagd „Fotojagd“ gibt es auf mehrere Weisen ein Feedback für den Anwender. Zunächst wird ihm angezeigt, wie viele Objekte er bereits gefunden hat. Bereits gefundene Objekte werden darüber hinaus ausgeblendet, so dass diese nicht zu Verwirrung führen. Bei der Suche nach Objekten wird dem Anwender ebenso Feedback gegeben. Drückt er auf „Gefunden“, obwohl das Objekt nicht im erlaubten Radius liegt, so wird ihm der Hinweis gegeben, dass das Objekt sich nicht in seiner Nähe befindet. Gefindet sich im Gegensatz dazu das Objekt im erlaubten Radius, so erhält der Anwender eine Benachrichtigung darüber und das Bild wird ausgeblendet.

Dialoge

Jegliche Eingaben innerhalb der App, werden in einzelnen Gruppen organisiert. Im Bereich der Einstellungen ist jede Einstellungsmöglichkeit in einem Klappmenu implementiert. Innerhalb der Schnitzeljagden des Planetenabiturs werden die Fragen in jeweils eigenen <fieldset>-Tags und somit Abschnitten behandelt. Um ein Objekt innerhalb der Fotojagd als gefunden zu markieren, wird ein eigener Dialog geöffnet. Das Feedback der einzelnen Gruppen wurde bereits in Abschnitt 7.1.1 behandelt.

Fehlerbehandlung

Die Fehlerbehandlung wurde in der App ebenfalls berücksichtigt. Eine zu Fehlern führende Eingabe kann jedoch nur im Bereich der QR-Codes zustande kommen. Dies geschieht, wenn der Anwender einen QR-Code gescannt hat, der nicht durch die App validiert wird. Der Anwender wird in diesem Fall darauf hingewiesen, dass der QR-Code ungültig ist und keine weitere Aktion mit diesem QR-Code vorgenommen wird. Mit diesem Verhalten wird das Smartphone vor Beschädigungen durch schädliche QR-Codes geschützt. Dem Anwender ist es jedoch jederzeit möglich, den QR-Code-Scanner erneut zu starten und einen weiteren QR-Code einzuscannen.

Stornierung

Eine Stornierung jeglicher Eingaben ist ebenfalls in der App ermöglicht. Die Eingaben, die in den Bereichen der Einstellungen gemacht werden, können durch die Auswahl einer neuen bzw. anderen Einstellung jederzeit geändert werden. Eingaben, die durch den QR-Code-Scanner erfolgten, können durch den „Zurück“-Button rückgängig gemacht werden oder durch das Einscannen eines anderen QR-Codes.

Im Planetenabitur können gesetzte Lösungen zu jedem Zeitpunkt verändert werden. Dies ist selbst nach der Auswertung möglich, so dass nach einer erneuten Auswertung das Ergebnis verbessert werden kann, wenn eine Lösung falsch war. Erst durch den erneuten Aufruf des Planetenabiturs werden alle Lösungen zurückgesetzt und das Planetenabitur von vorn gestartet.

Um die Eingaben der gefundenen Objekte in einer Fotojagd rückgängig machen zu können, muss die gesamte Fotojagd in dem Bereich neu gestartet werden. Ein einzelnes Objekt ist nicht in die Suche zurücksetzbar.

Benutzerkontrolle

Der Anwender wird zu keinem Zeitpunkt in seiner Interaktion durch die App beschränkt.

Kurzzeitgedächtnis

Es werden keine Anforderungen an das Kurzzeitgedächtnis gestellt. Jegliche Interaktion und Eingabe beschränkt sich auf eine Seite innerhalb der App.

Bewertung

Zusammenfassend kann festgehalten werden, dass die Oberfläche der App alle der acht Regeln von Ben Shneidermann (Abbildung 7.1, Seite 100) erfüllt. Somit wurde eine leicht erlernbare und einfache Oberfläche geschaffen. Der Anwender wird bei seinen Interaktionen und Eingaben durch das System unterstützt, verliert dabei aber nicht die Kontrolle über die App. Für fehlerhafte Eingaben und Interaktionen gibt es ebenso Toleranzen. Diese können durch Hinweise vom Anwender leicht korrigiert werden.

7.1.2. Test der Benutzbarkeit

Zum Test der Benutzbarkeit wurden fünf Personen ausgewählt, da diese Anzahl folgend einer Feststellung von Nielsen völlig ausreichend ist: „Elaborate usability tests are a waste of resources. The best results come from testing no more than 5 users and running as many small tests as you can afford [Nie00].“ Auch aus Abbildung 7.2 ist diese Annahme ableitbar.

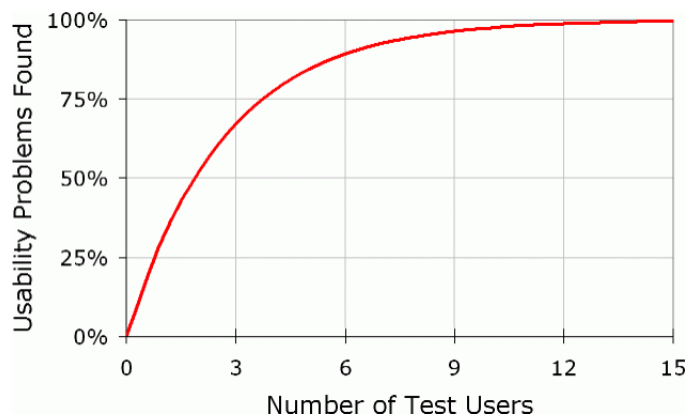


Abbildung 7.2.: Testpersonen und gefundene Fehler [Nie00]

Jede dieser Testpersonen erhielt eine App für ihr Smartphone. Daraufhin sollten sich die Testpersonen mit der App über Hude informieren und die Gemeinde erkunden. Um ein Ergebnis über die Benutzbarkeit zu erhalten, wurde ein quantitatives Auswertungsverfahren gewählt, die von John Brooke im Jahr 1986 entwickelte „System Usability Scale“ (SUS, siehe dazu Abbildung 7.3 auf Seite 104). Bei diesem Verfahren beantworten Testpersonen Aussagen bezüglich der Anwendung. Daraus resultiert ein quantitatives Ergebnis, das sich auswerten und interpretieren lässt. Der Fragebogen besteht aus 10 Aussagen mit jeweils 5 Antwortmöglichkeiten (Likert-Skalen), die beschreiben, wie benutzerfreundlich die App wahrgenommen wurde. Die Antwortmöglichkeiten werden von 0 bis 4 kodiert. Dabei erhält die beste Antwortmöglichkeit einer positiv formulierten Aussage die Kodierung 4 und die schlechteste Antwortmöglichkeit die Kodierung 0. Bei negativ formulierten Items wird die Kodierung umgekehrt gewählt. Die Auswertung des Bogens erfolgt anhand der Kodierung, so dass alle Antworten addiert werden. So kann aus allen 10 Fragen ein Wert zwischen 0-40 resultieren. Dieser Wert wird mit 2,5 multipliziert und ergibt als Resultat den SUS-Score (zwischen 0-100) eines Fragebogens. Aus allen Fragebögen wird das arithmetische Mittel des SUS-Score berechnet, um eine Aussage über die Benutzbarkeit treffen zu können. Die Aussage des SUS-Score lässt sich in die folgenden vier Bereiche einteilen [Rau11]:

7. Evaluation

100% Ein perfektes System ohne jegliche Probleme in der Benutzbarkeit.

über 80% Eine gute bis sehr gute Benutzbarkeit liegt vor.

zwischen 60-80% Eine grenzwertige bis gute Benutzbarkeit liegt vor.

unter 60% Deutliche Hinweise über Schwächen in der Benutzbarkeit liegen vor.

Aussage	Stimme gar nicht zu				Stimme voll zu
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.					
Ich empfinde das System als unnötig komplex.					
Ich empfinde das System als einfach zu nutzen.					
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.					
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.					
Ich finde, dass es im System zu viele Inkonsistenzen gibt.					
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					
Ich empfinde die Bedienung als sehr umständlich.					
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.					
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.					

Abbildung 7.3.: SUS-Fragebogen

Die Auswertung der Fragebögen, in Abbildung 7.4 auf Seite 105 dargestellt, ergibt einen Gesamt-SUS-Score von 82% und deutet somit darauf hin, dass in diesem Fall eine gute bis sehr gute Benutzbarkeit der App vorliegt.

Antwortmöglichkeit	Person I	Person II	Person III	Person IV	Person V
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.	3	4	3	3	2
Ich empfinde das System als unnötig komplex.	4	4	3	4	3
Ich empfinde das System als einfach zu nutzen.	3	2	3	3	4
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	4	4	3	3	3
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.	3	3	3	2	3
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	4	4	3	3	3
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.	3	4	2	4	3
Ich empfinde die Bedienung als sehr umständlich.	3	4	3	3	4
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.	4	4	3	3	3
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	4	3	3	4	4
Summe	35	36	29	32	32
Persönlicher SUS-Score	87,5%	90%	72,5%	80%	80%
SUS-Score			82%		

Abbildung 7.4.: Auswertung SUS-Fragebögen

7.2. Erfüllung der Anforderungen

Ein weiterer Faktor, der zur Evaluation herangezogen wird, ist die Implementierung der funktionalen Anforderungen (siehe Abschnitt 3.3.1, Seite 16), optionalen Anforderungen (siehe Abschnitt 3.3.2, Seite 17) sowie den nicht-funktionalen Anforderungen (siehe Abschnitt 3.3.3, Seite 17). Insgesamt galt es, 23 funktionale Anforderungen, 22 nicht-funktionale Anforderungen, sowie sieben optionale Anforderungen zu implementieren.

Anforderung	iOS	Android	Web App
Die App muss dem Anwender jederzeit die Möglichkeit bieten, zur Startseite zurückzukehren.	✓	✓	✓
Die App muss dem Anwender jederzeit die Möglichkeit bieten, einen QR-Code zu scannen.	✓	✓	✗
Die App muss dem Anwender jederzeit die Möglichkeit bieten, seinen Standort anzuzeigen.	✓	✓	✓

7. Evaluation

Anforderung	iOS	Android	Web App
Die App muss dem Anwender jederzeit die Möglichkeit bieten, das Impressum anzuzeigen.	✓	✓	✓
Die App muss dem Anwender jederzeit die Möglichkeit bieten, Einstellungen anzuzeigen.	✓	✓	✓
Die App muss dem Anwender die Möglichkeit bieten, eine Zielgruppe einzustellen.	✓	✓	✓
Die App muss dem Anwender die Möglichkeit bieten, eine Seite zurückzugehen.	✓	✓	✓
Die App muss dem Anwender Informationen über Hude präsentieren.	✓	✓	✓
Die App muss dem Anwender eine Übersicht der Sehenswürdigkeiten bieten.	✓	✓	✓
Die App muss dem Anwender die Sehenswürdigkeiten informativ präsentieren.	✓	✓	✓
Die App muss dem Anwender eine Übersicht der Unterkünfte bieten	✓	✓	✓
Die App muss dem Anwender eine Unterkunft informativ präsentieren.	✓	✓	✓
Die App muss dem Anwender eine Übersicht der Gastronomien bieten.	✓	✓	✓
Die App muss dem Anwender eine Übersicht der EC-Automaten in Hude anzeigen.	✓	✓	✓
Die App muss dem Anwender Einkaufsmöglichkeiten in Hude anzeigen.	✓	✓	✓
Die App muss dem Anwender Parkmöglichkeiten in Hude anzeigen.	✓	✓	✓
Die App muss dem Anwender Gesundheitseinrichtungen in Hude anzeigen.	✓	✓	✓
Die App muss dem Anwender die bevorstehenden Veranstaltungen in Hude anzeigen.	✓	✓	✓
Die App muss dem Anwender muss dem Anwender Spiele und Rätsel anbieten.	✓	✓	✓
Die App muss den Anwender eine Fotojagd durchführen lassen.	✓	✗	✓/✗
Die App muss den Anwender ein Planetenabitur lösen lassen	✓	✓	✓
Die App muss dem Nutzer, für eine unterschiedliche Dauer an Tagen und jeweils eingestellte Zielgruppen, Vorschläge für Unternehmungen anzeigen.	✓	✓	✓
Die App muss dem Anwender Standorte von Örtlichkeiten auf einer Karte anzeigen.	✓	✓	✓

Abbildung 7.5.: Umsetzung der funktionalen Anforderungen

Wie aus Abbildung 7.5 ersichtlich, ist die Implementierung der 23 funktionalen Anforderungen zu 96,38% gelungen. Zum einen war aus technischen Gründen die Implementierung des QR-Code-Scanners (1,449%) in die Web App nicht möglich. Der Grund dafür liegt darin, dass das API des PhoneGap-Framework (sie-

he Abschnitt 6.1.3 auf Seite 53) nicht über einen Browser anzusteuern ist, und das BarcodeScanner-Plugin (siehe Abschnitt 6.1.3 auf Seite 54) nicht eingebunden werden kann. Weiterhin ist aus unerklärlichen Gründen das Auslesen der GPS-Koordinaten, durch das Exif-Reader-Plugin (siehe Abschnitt 6.1.1 auf Seite 52), auf Android-Geräte nicht möglich gewesen. Dieses Phänomen war auch in der Web App nicht kontrollierbar. Aus diesem Grund wurde die Fotojagd auf Android-Geräten zwar implementiert, aber ist nicht aufrufbar (2,173%).

Neben den funktionalen Anforderungen die es zu erfüllen galt, gab es zusätzlich sieben optionale Anforderungen. Diese waren ein sog. „nice to have“, wurden jedoch - soweit möglich war - implementiert. Inwieweit diese optionalen Anforderungen implementiert wurden, zeigt Abbildung 7.6.

Anforderung	iOS	Android	Web App
Die App soll verschiedene Schriftgrößen unterstützen.	✓	✓	✓
Die App soll verschiedene Sprachen in der Ausgabe der Texte unterstützen.	✓	✓	✓
Die App soll eine Sprachausgabe haben.	✓	✓	✗
Die App soll verschiedene Sprachen in der Sprachausgabe unterstützen.	✓	✓	✗
Die App soll dem Nutzer ein Navigationspanel zur Navigation innerhalb der App bieten.	✓	✓	✓
Die App soll dem Nutzer Spielplätze auf einer Karte anzeigen.	✓	✓	✓
Die App soll bei einer nicht ausreichenden Datenverbindung einen Kartenausschnitt, anstatt einer Google-Maps-Karte, als Grafik bereitstellen.	✓	✓	✓

Abbildung 7.6.: Umsetzung der optionalen Anforderungen

Wie in Abbildung 7.6 dargestellt, sind 90,48% der optionalen Anforderungen in allen drei Apps verfügbar. Die übrigen, nicht verfügbaren, 9,52% der Implementierungen, liegen in der Web-App. Der Grund hierfür ist darin zu finden, dass die Audio-Inhalte nicht in Verbindung mit dem PhoneGap-Framework und einem Browser auf dem Dateisystem gespeichert werden können. Dadurch ist eine Wiedergabe jeglicher Audio-Inhalte nicht möglich.

Abbildung 7.7 zeigt, dass die nicht-funktionalen Anforderungen zu 100% erfüllt wurden.

Die App muss ohne Registrierung nutzbar sein.	✓
Die App muss leicht wartbar sein.	✓
Die App muss leicht modifizierbar sein.	✓
Die App muss portierbar sein.	✓
Die App muss zuverlässig sein.	✓
Die App muss korrekt funktionieren.	✓
Die App muss auf den gängigen Smartphones lauffähig sein.	✓
Die App muss der Corporate Identity der Touristik-Palette Hude e.V. entsprechen.	✓
Die App muss mindestens die Unternehmen, die Mitglied des Touristik-Palette Hude e.V. sind, präsentieren.	✓
Die App muss für junge und ältere Anwender nutzbar sein.	✓
Die App muss intuitiv bedienbar sein.	✓

7. Evaluation

Die App muss rechtskonform sein.	✓
Die App muss über die App Stores der mobilen Betriebssysteme vertrieben und bezogen werden.	✓
Die App muss als Web-App aufrufbar sein.	✓
Die App muss ein Header-Element haben.	✓
Die App muss ein Footer-Element haben.	✓
Die App muss ein Panel-Element haben.	✓
Die App muss das GPS-Modul nutzen.	✓
Die App muss das Kamera-Modul nutzen.	✓
Die App muss einen QR-Code-Scanner integriert haben.	✓
Die App muss über QR-Codes Inhalte direkt öffnen.	✓
Die App muss ohne Internetzugang nutzbar sein.	✓

Abbildung 7.7.: Umsetzung der nicht-funktionalen Anforderungen

7.3. Richtigkeit

Die Evaluation der Richtigkeit der Inhalte wurde durch die Touristik-Palette-Hude vorgenommen. Dazu wurde der Touristik-Palette-Hude während der Entwicklungsphase Zugriff auf die Web-App gewährt. Über diesen Zugang konnten die Mitarbeiter stetig die Inhalte einsehen und überprüfen.

Am 20. August 2013 wurde eine E-Mail mit den zu verbessernden Inhalten versandt. Diese Inhalte wurden umgehend in die App eingearbeitet, so dass die Inhalte sich zum aktuellen Zeitpunkt auf einem korrekten Stand befinden.

Die deutschen Inhalte wurden von Leon Stelzner, dem Sohn einer Mitarbeiterin der Touristik-Palette-Hude und angehendem Anglistik-Studenten, ins Englische übersetzt. Diese Übersetzungen wurden als richtig angenommen und ohne eine genauere Prüfung in die App eingepflegt.

8. Fazit

Auf Grundlage des Wachstums und der damit steigenden Bedeutung des binnenländischen Tourismusmarktes, wurde das Tourismus-Konzept als Wegweiser für aktuelle und zukünftige touristische Entwicklungen der Gemeinde Hude und Umgebung erstellt.

Ziel der vorliegenden Arbeit „Konzeption und Implementierung einer Touristen-App für die Gemeinde Hude“ war es, eine Smartphone-Applikation für die Touristik-Palette-Hude e.V. zu entwickeln, die die Umsetzung des Tourismus-Entwicklungskonzeptes für den staatlich anerkannten Erholungsort Hude unterstützt.

Angestrebt wird vor allem eine kreative und innovative Förderung des Tourismus. Zu diesem Zweck wurden in einer „Status-Quo-Analyse“ bestehende Touristik-Apps, z.B. über App Store -Recherche, identifiziert und die Apps der Städte Konstanz, Köln, München und Wernigerode als Basis für einen ersten Prototyp herangezogen. In mehreren Analysetreffen mit der Touristik-Palette-Hude e.V. konnte gezeigt werden, dass das prototypische Vorgehen sehr gut geeignet ist, um den Funktionsumfang und dessen Visualisierung abzustimmen. In der Evaluationsphase konnte zwar anhand der „acht goldenen Regeln zum Design der Benutzeroberfläche“ von Ben Shneidermann und dem Einsatz von 5 Testusern gezeigt werden, dass die prototypische Vorgehensweise eine gute Erfüllung der acht goldenen Regeln zum Design der Benutzeroberfläche und eine sehr hohe Akzeptanz beim Testanwender erzielt. Die Eindrücke der Testpersonen wurden durch das SUS-Verfahren ausgewertet. Die Relevanz der Akzeptanzkriterien der Testpersonen ist aber nur unter der Einschränkung zulässig, dass das Testumfeld und die Tester möglichst wirklichkeitsnah sind. Deshalb wurden zusätzlich aus dem zugrundeliegenden Konzept die Zielgruppen extrahiert und zwei Personas erstellt.

Durch die Verwendung der Personas konnte die prototypische Vorgehensweise sinnvoll unterstützt werden, da bereits an den Prototypen die für die Personas relevanten Anwendungsfälle beurteilt und angepasst werden konnten.

Andererseits musste festgestellt werden, dass insbesondere die nicht-funktionalen Anforderungen beim prototypischen Vorgehen ggf. erst in späteren Implementierungs- bzw. Evaluationsphasen überprüft werden können. Neben dem Hauptziel, der Entwicklung der Touristen-App, konnte zusätzlich gezeigt werden, dass wichtige nicht-funktionale Anforderungen bereits vor der eigentlichen Entwicklungsphase über eine Checkliste ermittelt und überprüft werden sollten. Insbesondere betrifft dieses die Wartbarkeit, die Portierbarkeit, und die Performanz (z.B. Online- versus Offline-Nutzung) der Applikation. In der Arbeit wurde gezeigt, dass diese größtenteils nicht-funktionalen Anforderungen sowohl die Art der Applikation (hybride, Web- oder native App), als auch die zu verwendende Entwicklungsumgebung erheblich beeinflussen können. Was die Entwicklungsumgebung betrifft konnte gezeigt werden, dass nur durch eine genaue Analyse der vorhandenen Frameworks überflüssige mehrfache Implementierungen vermieden werden können. Bei der Analyse der Performance ergab sich, dass die App auch ohne eine Datenverbindung in Hude nutzbar sein muss. Die wichtigsten nicht-funktionalen Anforderungen der Touristik-Palette-Hude e.V. waren die Portierbarkeit auf verschiedene mobile Betriebssysteme und die Möglichkeit einer Wartung durch die Touristik-Palette-Hude e.V. selbst.

Im dritten Schritt der Evaluierung wurde die Umsetzung der funktionalen, optionalen und nicht-funktionalen Anforderungen überprüft. Die vierte und letzte Evaluierung wurde durch die Touristik-Palette-Hude selbst durchgeführt, indem die Inhalte auf ihre Richtigkeit überprüft wurden.

Was die Notwendigkeit der frühen Berücksichtigung der nicht-funktionalen Anforderungen betrifft, so konnte anhand der Evaluationsphase durch Tester gezeigt werden, dass in der wirklichkeitsnahen Evaluation insbesondere diese Anforderungen nachgebessert werden mussten. Des Weiteren ist gezeigt worden, dass die Nutzbarkeit einiger Funktionen im Praxiseinsatz durch örtliche Gegebenheiten erschwert (z.B.

8. Fazit

ungenauere GPS-Lokalisierung) oder unmöglich gemacht werden kann (z.B. Netzverfügbarkeit). Im Verlauf der Arbeit konnte zudem gezeigt werden, dass schon in frühen Entwicklungsphasen eine Evaluierung auf einem aussagekräftigen Subset der Endgerätepalette erfolgen sollte, damit spezifische Framework-Constraints, wie zum Beispiel das stetige Zuweisen neuer Klassen- und Style-Definitionen und das Nachladen, sowie Cachen von Inhalten beim jQuery Mobile Framework, bereits im Vorfeld identifiziert werden und ein Frameworkwechsel – wie im Verlauf dieser Arbeit auf das Foundation Framework von Zurb – vermieden werden kann.

Insgesamt kann festgestellt werden, dass das Ziel der vorliegenden Arbeit nicht nur durch die erfolgreiche Umsetzung der Touristen-App erreicht wurde, sondern dass im Zuge der Arbeit auch wichtige Grundlagen hinsichtlich der Vorgehensweise für die Erstellung solcher Applikationen erarbeitet wurden.

Insofern steht zu hoffen, dass in weiteren Arbeiten die Vorgehensweise der App-Erstellung noch weiter grundlagentechnisch erweitert werden kann. Auch die Erweiterbarkeit der geschaffenen Touristen-App ist in vielen Kapiteln bereits ersichtlich und angerissen. Für zukünftige Erweiterungen sollte nicht nur die entwickelte Vorgehensweise angewendet werden, sondern auch die Bereitstellung des Contents frühzeitig geplant und realisiert werden, damit geplante Anforderungen auch mit realem Content überprüft werden können.

Glossar

- AJAX** Asynchronous JavaScript and XML (AJAX) bezeichnet ein Verfahren zur asynchronen Datenverarbeitung und -übertragung. So können beispielsweise Seiten aufgerufen werden, ohne eine gesamte Seite neu abrufen zu müssen.
- API** Application programming interface (API) bezeichnet eine Schnittstelle zwischen zwei unterschiedlichen Systemen oder Programmen innerhalb des Source Codes, die es ermöglicht, eine Verbindung untereinander herstellen zu können.
- App** Applikationen (Apps) bezeichnen Anwendungen, die auf einem Betriebssystem lauffähig sind. Erst durch die große Verbreitung von mobilen Endgeräten (Smartphones & Tablets) setzte sich für diese der Begriff „App“ durch.
- AWF** Anwendungsfall (AWF) Durch einen Anwendungsfall wird ein bestimmtes Szenario beschrieben, das ein potentieller Akteur in einer zu entwickelnden Anwendung erleben könnte.
- bzw.** beziehungsweise
- CSS** Cascading Style Sheets (CSS) ermöglichen die Anpassung und Definition der Darstellung von HTML-Elementen.
- etc.** et cetera
- Exif** Exchangeable Image File Format (Exif) ist ein Datenformat zum Speichern von Metadaten im Header eines Bildes. Diese Metadaten können zum Beispiel die Belichtungszeit, das Kameramodell oder die Positionsdaten umfassen.
- Git** Git ist eine Software zur Versionsverwaltung unterschiedlicher Dokumente. Innerhalb eines sogenannten Repositories werden die, durch die Modifizierung unterschiedlichen, Versionen von Dateien gespeichert. Mitnutzer eines Git erfahren so ebenfalls die Änderungen und können Veränderungen nachverfolgen oder andere Versionen wiederherstellen.
- Google Inc.** Google Inc. ist eine US-amerikanische Unternehmung. Bekannt ist Google Inc. besonders durch die weltgrößte gleichnamige Suchmaschine Google. Neben dem Betrieb der Suchmaschine Google hat Google Inc. seine Geschäftsfelder im Internet weiter ausgedehnt.
- GPS** Global Positioning System (GPS) ist ein satellitengestütztes System zur Positionsbestimmung.
- HTML** Eine detaillierte Erklärung des Begriffs Hypertext Markup Language (HTML) wird in Abschnitt 2.5 auf Seite 10 gegeben.
- JS** JavaScript (JS) ist eine Skriptsprache, die zur Interaktion mit dem Anwender und zur Veränderung von CSS- und HTML-Elementen entwickelt wurde. Die Ausführung eines JavaScripts erfolgt auf Seite des Anwenders.
- JSONP** JSON with padding (JSONP) ist ein Verfahren zur Datenübertragung mittels JavaScript mit der sogenannten „JavaScript Object Notation“ (JSON). Durch dieses Verfahren wird der Zugriff auf eine externe bzw. andere Domain ermöglicht.

kapazitiven Display Ein kapazitives Display reagiert auf Berührung. Dabei ist für die Interaktion nicht der ausgeübte Druck entscheidend, sondern die Veränderungen der Leitfähigkeit des Displays durch die Berührung.

MMS Multi Message Service (MMS) ist die Weiterentwicklung des „Short Message Service“ (SMS). Im Gegensatz zu SMS können via MMS neben Textnachrichten auch Medieninhalte (Bilder, Sprachnachrichten, etc.), über das Mobilfunknetz versendet werden.

PC Personal Computer

PHP Hypertext Preprocessor (PHP) ist eine skriptbasierte, serverseitige Programmiersprache.

PlugIn Ein PlugIn ist eine Erweiterung für ein bereits bestehendes Programm, das auf eine Erweiterung der Funktionalität dieses Programms abzielt.

responsive Als „responsives“-Verhalten wird die automatische Anpassung an die Displaygröße des genutzten Gerätes und die möglichst optimale Skalierung der auf dem Display dargestellten Objekte bezeichnet.

SDK Software Development Kit (SDK) ist eine Entwicklungsumgebung für eine Anwendung. Ein SDK kann sich aus verschiedenen Komponenten zusammensetzen. Beispielsweise kann dies ein Compiler oder der Editor (IDE) sein.

SMS Short Message Service (SMS) sind kurze Text-Nachrichten von maximal 160 Zeichen, die über das Mobilfunknetz übertragen werden.

Tethering Als Tethering bezeichnet man die Möglichkeit, die Datenverbindung des Smartphones über andere Geräte mitnutzen zu können.

TTS Text-To-Speech (TTS) ist ein Verfahren zur Konvertierung eines Fließtextes in eine Sprachausgabe. Die Sprachausgabe erfolgt durch synthetisierte Stimmen - bekannt zum Beispiel aus Navigationssystemen.

URI Uniform Resource Identifier (URI) ist eine Zeichenfolge, die ein Objekt im Internet eindeutig identifiziert. Bei Domains (Aufruf von Webseiten) spricht man vom Uniform Resource Locator (URL).

uvam. und viele andere mehr

WLAN Wireless Local Area Network (WLAN) bezeichnet die Verbindung zu einem Netzwerk per Funk.

XML Extensible Markup Language (XML) ist eine Auszeichnungssprache. In dieser werden innerhalb eines „Document Object Models“ (DOM) Daten hierarchisch in einer Baumstruktur gespeichert. Aufgrund des Nichtvorhandenseins vordefinierter Elemente (Tags) ist diese frei erweiterbar, wodurch auch keine Definition der Darstellung wie bei der HTML erfolgt.

z.B. zum Beispiel

Literatur

- [Aca] ACAPELA GROUP: *Your Voice expert.* – <http://www.acapela-group.com/company-information-acapela-group.html> (letzter Abruf: 27. August 2013)
- [Ado] ADOBE SYSTEMS INC.: *PhoneGap Documentation.* – Erreichbar unter: <http://docs.phonegap.com/en/2.7.0/index.html> (letzter Abruf: 18. Mai 2013)
- [Ado13] ADOBE SYSTEMS INC.: *Using config.xml.* 2013. – Erreichbar unter: <https://build.phonegap.com/docs/config-xml> (letzter Abruf: 23. Mai 2013)
- [App] APPLE INC.: *Guidelines.* – <https://developer.apple.com/appstore/guidelines.html> (letzter Abruf: 8. April 2013)
- [Aud13] AUDACITY ENTWICKLERTEAM: *Audacity.* Januar 2013. – <http://audacity.sourceforge.net/?lang=de> (letzter Abruf: 27. August 2013)
- [Bec] BECKERT, THOMAS: *Mobile Webseite oder Native App » mit Hybrid Apps in die Zukunft.* – Erreichbar unter: <http://www.sitepoint.de/mobile-webseiten/> (letzter Abruf: 9. April 2013)
- [Ben10] BEN ALMAN: *jQuery hashchange event.* Juli 2010. – <http://benalman.com/projects/jquery-hashchange-plugin/> (letzter Abruf: 21. August 2013)
- [Ber11] BERGERT, DENISE: *Handy-Riese Nokia verbündet sich mit Microsoft.* Februar 2011. – <http://www.welt.de/wirtschaft/webwelt/article12506678/Handy-Riese-Nokia-verbuedet-sich-mit-Microsoft.html> (letzter Abruf: 8. April 2013)
- [Bib] BIBLIOGRAPHISCHES INSTITUT GMBH: *App.* – <http://www.duden.de/rechtschreibung/App> (letzter Abruf: 8. April 2013)
- [BMW] BMWI: *Entwicklung der Tourismusbranche.* – Erreichbar unter: <http://www.bmwi.de/DE/Themen/Tourismus/Branche-im-Ueberblick/entwicklung-der-tourismusbranche.html> (letzter Abruf: 17. April 2013)
- [Boh11] BOHN, DIETER: *iOS: A visual history.* Dezember 2011. – <http://www.theverge.com/2011/12/13/2612736/ios-history-iphone-ipad> (letzter Abruf: 8. April 2013)
- [Bud12] BUDDE, CHRISTIAN: *Welche Bedeutung hat das Tourismusgewerbe für Niedersachsen?* Juli 2012. – Erreichbar unter: http://www.mw.niedersachsen.de/portal/live.php?navigation_id=5459&article_id=107663&psmand=18 (letzter Abruf: 17. April 2013)
- [con13] CONNECT.DE: *Android: Die Geschichte des Erfolgs.* April 2013. – <http://www.connect.de/ratgeber/android-geschichte-des-erfolgs-1491130.html> (letzter Abruf: 8. April 2013)
- [dpa13] DPA/REUTERS/CAT/LW: *Windows Phone SDK 8.0.* Januar 2013. – http://www.pcwelt.de/news/Windows_Phone_Store_erreicht_150.000_Apps-Meilenstein-7334650.html (letzter Abruf: 8. April 2013)

- [fan] FANCYAPPS.COM: *fancyBox*. – <http://fancyapps.com/fancybox/> (letzter Abruf: 21. August 2013)
- [FI12] FRANKE, Florian ; IPPEN, Johannes: *Apps mit HTML5 und CSS3 - für iPad, iPhone und Android*. 1. Aufl. Bonn : Galileo Press GmbH, 2012. – ISBN 978-3-836-21848-1
- [Gab] GABLER WIRTSCHAFTSLEXIKON: *Smartphone*. – <http://wirtschaftslexikon.gabler.de/Archiv/569824/smartphone-v1.html> (letzter Abruf: 8. April 2013)
- [Gil] GILLY: *Endlich Wi-Fi-Tethering für das iPhone 4: Der Personal Hotspot im Video vorgeführt*. – <http://goo.gl/zMaRM> (letzter Abruf: 8. April 2013)
- [Gil11] GILLY: *Meine Windows Mobile Geschichte und die Enttäuschung über das Windows Phone 7 Update*. Mai 2011. – <http://www.connect.de/ratgeber/android-geschichte-des-erfolgs-1491130.html> (letzter Abruf: 8. April 2013)
- [Her13] HERKNER, LUTZ: *Windows Phone ist endlich ebenbürtig*. April 2013. – Erreichbar unter: <http://www.zeit.de/digital/mobil/2012-10/windows-phone-8>, (letzter Abruf: 8. April 2013)
- [Iff12] IFF, STEFAN: *Mobile Business Blog-Serie – Übersicht über die wichtigsten Varianten mobiler Anwendungen*. Septemberr 2012. – http://res.sys-con.com/author/4963/hybriddesignapproach_0.png (letzter Abruf: 8. April 2013)
- [Jen10] JENDRYSCHIK, MICHAEL: *Mit Personas Projekte menschlich und motivierend gestalten*. Dezember 2010. – Erreichbar unter: <http://jendryschik.de/weblog/2010/12/28/mit-personas-projekte-menschlich-und-motivierend-gestalten/> (letzter Abruf: 15. April 2013)
- [joh12] JOHANSALLLARSSON: *basic example*. März 2012. – <http://jquery-ui-map.googlecode.com/svn/trunk/demos/jquery-google-maps-basic-example.html> (letzter Abruf: 22. August 2013)
- [Kan] KANE, Matt: *BarcodeScanner*. – Erreichbar unter: <https://build.phonegap.com/plugins/13> (letzter Abruf: 23. August 2013)
- [Mei] MEIER, SONJA: *Tourismus-Apps*. – Erreichbar unter: <http://www.tourismus-it.de/tourismus-apps> (letzter Abruf: 17. April 2013)
- [Mica] MICROSOFT: *Microsoft Windows Phone 7*. – <http://www.windowsphone.com/en-us/how-to/wp7/basics/update-history> (letzter Abruf: 8. April 2013)
- [Micb] MICROSOFT: *Windows Phone 7.8*. – <http://www.windowsphone.com/en-us/how-to/wp7/start/whats-new-in-windows-phone> (letzter Abruf: 8. April 2013)
- [Micc] MICROSOFT: *Windows Phone SDK 8.0*. – <http://www.microsoft.com/de-de/download/details.aspx?id=35471> (letzter Abruf: 8. April 2013)
- [n-t11] N-TV: *Nokias erstes Windows-Phone-Modell Lumia 800 soll angreifen*. November 2011. – <http://www.n-tv.de/technik/Lumia-800-soll-angreifen-article4745201.html> (letzter Abruf: 8. April 2013)
- [Nie00] NIELSEN, JAKOB: *Why You Only Need to Test with 5 Users*. 2000. – <http://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/> (letzter Abruf: 30. August 2013)

- [pho] *Plugins*. – Erreichbar unter: <https://build.phonegap.com/plugins> (letzter Abruf: 23. August 2013)
- [pho11] *PhoneGap: avagy a mobil fejlesztés leghatékonyabb HTML5/JavaScript-ben?* Oktober 2011. – Erreichbar unter: <http://html5hu.wordpress.com/2011/10/05/a-mobil-fejlesztés-leghatékonyabb-html5javascript-ben/> (letzter Abruf: 21. August 2013)
- [Pon09] PONS, Giulio: *Decimal Degrees conversion and distance of two points on google map*. Dezember 2009. – Erreichbar unter: <http://www.barattalo.it/2009/12/26/decimal-degrees-conversion-and-distance-of-two-points-on-google-map/> (letzter Abruf: 22. Juli 2013)
- [Pot12] POTSCHIEN, DENIS: *HTML5 und Local Storage: Daten für Offlinebetrieb speichern*. Dezember 2012. – <http://t3n.de/news/html5-local-storage-daten-offlinebetrieb-speichern-350314/> (letzter Abruf: 8. April 2013)
- [Rau11] RAUER, MATTHIAS: *Quantitative Usability-Analysen mit der System Usability Scale (SUS)*. April 2011. – <http://blog.seibert-media.net/2011/04/11/usability-analysen-system-usability-scale-sus/> (letzter Abruf: 2. September 2013)
- [Sac11] SACHDEVA, CHIRAG: *History and Key Features of the Windows phone 7*. Januar 2011. – <http://www.techulator.com/resources/4399-History-Key-Features-Windows-phone.aspx> (letzter Abruf: 8. April 2013)
- [Sei08] SEIDELIN, Jacob: *Javascript EXIF Reader*. Mai 2008. – <http://blog.nihilogic.dk/2008/05/reading-exif-data-with-javascript.html> (letzter Abruf: 22. August 2013)
- [Shn01] SHNEIDERMAN, BEN: *Shneiderman's 'Eight Golden Rules of Interface Design'*. 2001. – <http://faculty.washington.edu/jtenenbg/courses/360/f04/sessions/schneidermanGoldenRules.html> (letzter Abruf: 30. August 2013)
- [TPH11] TOURISTIK-PALETTE-HUDE: *Tourismus - Entwicklungskonzept 2011 für den staatl. anerkannten Erholungsort Hude*. (2011), S. 36
- [van13] VAN DER MEULEN, ROB AND RIVERA, JANESSA: *Gartner Says Worldwide Mobile Phone Sales Declined 1.7 Percent in 2012*. Februar 2013. – Erreichbar unter: <http://www.gartner.com/newsroom/id/2335616> (letzter Abruf: 12. April 2013)
- [W3C12] W3C: *Packaged Web Apps (Widgets) - Packaging and XML Configuration (Second Edition)*. November 2012. – Erreichbar unter: <http://www.w3.org/TR/widgets/> (letzter Abruf: 23. Mai 2013)
- [Wik05] WIKIPEDIA: *Ben Shneiderman*. 2005. – http://de.wikipedia.org/wiki/Ben_Shneiderman (letzter Abruf: 30. August 2013)
- [Wik13a] WIKIPEDIA: *Exchangeable Image File Format*. Juli 2013. – http://de.wikipedia.org/wiki/Exchangeable_Image_File_Format (letzter Abruf: 22. August 2013)
- [Wik13b] WIKIPEDIA: *Microsoft Windows Phone 7*. März 2013. – http://de.wikipedia.org/wiki/Microsoft_Windows_Phone_7 (letzter Abruf: 8. April 2013)

Literatur

- [Wik13c] WIKIPEDIA: *Microsoft Windows Phone 8*. April 2013. – http://de.wikipedia.org/wiki/Microsoft_Windows_Phone_8 (letzter Abruf: 8. April 2013)
- [Wil] WILLOUGHBY, Ryan: *BarcodeScanner*. – Erreichbar unter: <https://github.com/phonegap-build/BarcodeScanner/blob/master/README.md> (letzter Abruf: 23. August 2013)
- [ZUR] ZURB: *Foundation*. – <http://foundation.zurb.com/> (letzter Abruf: 22. August 2013)

Anhang

A. Zertifikate

iPhone Developer Zertifikate

Damit die App durch Phonegap Build für iOS-Geräte erstellt werden kann, ist es notwendig, dass die Apple-Entwicklerzertifikate in Phonegap Build eingebunden werden. Diese werden im Apple Developer Center, sowie auf dem OS X des Entwicklers erstellt.

Vorgehen

Nach der Anmeldung im Apple Developer Center wird unter dem Punkt „Identifiers“ über das „+“ eine neue ID für die App erstellt. Diese muss auch in der config.xml verwendet werden.

Name:	Touristik Palette Hude
Explicit App ID, Bundle ID:	de.gastinhude.m

Abbildung A.1.: Einstellungen im App-Identifizier

Im Anschluss daran wird unter „iOS Apps“ „Certificates“ ausgewählt. Nun werden dem Nutzer alle bereits vorhandenen Zertifikate angezeigt. Über das „+“ lassen sich neue Zertifikate erstellen. Für die Erstellung der App durch Phonegap Build wird das „iOS-Development“-Zertifikat benötigt, zur Übermittlung an den Apple App Store das iOS-Deployment-Zertifikat. Zur Erstellung der Zertifikate muss auf dem OS X-System die „Schlüsselbundverwaltung“ geöffnet werden und ein Zertifikat einer Zertifizierungsinstanz angefordert werden.

„Schlüsselbundverwaltung -> Zertifikatsassistent -> Zertifikat einer Zertifizierungsinstanz anfordern“
In den Dialog wurden dazu die Daten aus Abbildung A.2 eingetragen und die Zertifikatsanforderungsdatei auf der Festplatte gespeichert.

E-Mail des Benutzers:	app@touristik-palette-hude.de
Allgemeiner Name:	Touristik Palette Hude
E-Mail der Zert.-Instanz:	
Anfrage:	Auf der Festplatte sichern

Abbildung A.2.: Zertifizierungsanfrage

Die Zertifikatsanforderungsdatei wurde im Anschluss zum Apple Developer Center hochgeladen. Aus dieser Datei wurde dann das Development- und Deployment Zertifikat generiert und heruntergeladen. Diese Zertifikate werden der Schlüsselbundverwaltung auf dem OS X-System hinzugefügt. Nachdem die Zertifikate hinzugefügt wurden, müssen diese jeweils als „Personal Information Exchange“-Datei (*.p12) exportiert werden.

Im Anschluss daran muss im Apple Developer Center unter „Device“ ein Testgerät aus dem Hause „Apple“ hinzugefügt werden. Das „+“-Symbol öffnet einen Dialog zum Hinzufügen eines Gerätes. Dieser Dialog nimmt zwei Daten des Gerätes auf, zum einen den Namen (frei wählbar) zum anderen die „UDID“. Um die erstellte App auf einem Gerät testen zu können, muss ein „iOS Provisioning Profile“ angelegt werden. Dies geschieht über das Menü „Provisioning Profiles“. Das „+“-Symbol öffnet einen Dialog zum Erstellen eines „iOS Provisioning Profile“. Die Abbildung A.3 zeigt die getätigten Angaben.

Type:	iOS App Development
AppID:	de.gastinhude.m
Certificate:	Björn-Hendrik von der Linde
Device:	Bjoern iPhone
Profilname:	appTPH

Abbildung A.3.: iOS Provisioning Profile

Die Development- und Deployment-Zertifikate und die dazugehörigen Provisioning Profiles werden zur Erzeugung der App an PhoneGap Build übermittelt.

Android KeyStore-Zertifikat

Zunächst ist ein Download des „Android KeyStore Tool“¹ notwendig. Anschließend müssen im Tool der Pfad zum Java-JDK gesetzt sowie der Speicherort der KeyStore-Datei festgelegt werden. Zum Zweck der Zertifizierung der App müssen weiterhin Angaben beispielsweise zum Urheber der App und seinen Kontaktdaten getätigt werden. Entscheidend ist, dass in diesem Schritt neben einem „Store Password“ ein „KeyStore Password“ und ein „Alias“ gewählt werden, um eine Entsperrung des Zertifikats und der KeyStore-Datei im späteren Verlauf in PhoneGap Build zu ermöglichen.

¹<http://community.clickteam.com/threads/80249-KeyStore-Generator-Tool> (letzter Abruf: 17.09.2013)

B. GitHub

Benutzername: HudeApp
E-Mail: app@touristik-palette-hude.de
Passwort: Motzgui3

Abbildung B.1.: GitHub-Benutzerdaten

Zunächst erfolgt der Download des Clients für Windows² und dessen Installation. Nach der Installation müssen die Zugangsdaten des GitHub-Accounts eingegeben werden. Durch einen Klick auf „HudeApp“ unter dem Menüpunkt „github“ werden die Repositories angezeigt. Durch einen Klick auf den Pfeil des Repositories „HudeApp/app“ erfolgt die Übertragung auf den Rechner.

Werden Dateien im lokalen Repository verändert, müssen diese mittels des Clients an GitHub übertragen werden. Nach Auswahl des Repository im Client werden die veränderten Dateien angezeigt. Dort kann nun die „Commit-Nachricht“ festgelegt werden. Anschließend kann die Synchronisation mittels des Buttons „sync“ durchgeführt werden.

²<http://github-windows.s3.amazonaws.com/GitHubSetup.exe> (letzter Abruf: 14.08.2013)

C. Übermittlung an die App Stores

Im Anschluss an die Erstellung der App können die jeweiligen Dateien (HudeApp.ipa, HudeApp.apk) vom Build-Service heruntergeladen werden. Diese müssen nun den App Stores zugespielt werden.

App Store

Um die App im App Store veröffentlichen zu können, wird zunächst die Seite "iTunes Connect" aufgerufen und der Menüpunkt "Manage Your Apps" ausgewählt. Durch die Funktion "Add New App" werden Metadaten zur App definiert. Anzugebende Metadaten bestehen unter anderem in der Sprachauswahl ("Default Language: German"), dem App-Namen, der SKU Number und der Bundle ID. Nach Bestätigung der Angaben durch "Continue" wird das Veröffentlichungsdatum und der Preis der App gewählt. Anschließend müssen Informationen über die App festgelegt werden. Dazu zählen die Versionsnummer, Angaben zum Copyright, Angaben zu einer Kategorisierung der App, sowie eine Multiple-Choice-Beschreibung der Inhalte und die Festlegung eines Icons. Weiterhin werden Metadaten gesetzt wie etwa der App-Name, eine Beschreibung der App, entsprechende Keywords (unter denen die App im Store zu finden ist) und Screenshots der App im Betrieb. Auch Kontaktinformationen werden an dieser Stelle hinterlegt. Nun muss der Application Loader von iTunes Connect heruntergeladen und installiert werden. Im Anschluss wird die *.ipa-Datei, die mit dem Deployment-Zertifikat erstellt wurde, von PhoneGap Build heruntergeladen. Darauf folgt der Start des Application Loaders und die Verbindung mit iTunes Connect über denselbigen. Die in iTunes Connect erstellte App muss nun ausgewählt und die *.ipa-App-Datei hochgeladen werden. Nach einer Prüfung durch Apple wird die App freigeschaltet und ist im App Store verfügbar.

Google Play

Um die App in Google Play veröffentlichen zu können, müssen entsprechende, im Folgenden näher beschriebene, Schritte vorgenommen werden.

Die erzeugte KeyStore-Datei wird - unter Eingabe des zuvor definierten Alias - an PhoneGap Build übertragen. Um die KeyStore-Datei für PhoneGap Build nutzbar zu machen, ist es an dieser Stelle notwendig, die Datei anhand der beiden zuvor gewählten Passwörter zu entsperren. Ist die Datei nun entsperrt, muss die App durch PhoneGap Build neu erstellt werden. Dadurch wechselt sie von der Developer-Version („Debug“) in die veröffentlichbare Version („Release“). Diese Datei muss im Folgenden heruntergeladen und auf einem Computer gespeichert werden.

Über die „Developer-Console“³ von Google wird die App Google Play hinzugefügt. Im Einzelnen umfasst dieses Vorgehen das Hochladen der App (Hude.apk) in die Developer-Console und das Setzen des „Store-Eintrages“ mit den App-Details. Diese Details beinhalten die Festlegung einer Standardsprache (in diesem Falle „Deutsch-de-DE“), eines Titels und einer Beschreibung, Screenshots, ein Logo sowie die Einordnung der App in eine Kategorie. Ebenso müssen Kontaktinformationen und Datenschutzbestimmungen angegeben werden. Im Reiter „Preisgestaltung und Vertrieb“ wird weiterhin festgelegt, dass die App kostenfrei und weltweit zu beziehen ist. Abschließend muss eine Einverständniserklärung zu den Google-Play-Richtlinien abgegeben werden.

Web App

Um die Web App zu verbreiten, ist lediglich ein Upload aller Dateien aus dem Ordner „WebApp“ via FTP in den Ordner „httpdocs“ auf den Server notwendig. Die Verbindung zum Server wird mit den folgenden Daten hergestellt:

³<https://play.google.com/apps/publish/> (letzter Abruf: 25.09.2013)

Benutzername: m.tph
Passwort: 7D10C1A3
Server: touristik-palette-hude.de
Transfermodus: Aktiv

Abbildung C.1.: FTP-Verbindungsdaten

Der Aufruf der App erfolgt dann über <http://m.gastinhude.de> oder <http://m.touristik-palette-hude.de>.

D. QR-Codes

Die QR-Codes wurden mit Hilfe der Website „<http://www.qrcode-monkey.de/>“ erstellt. Dazu wurden die URLs im Feld „Gib eine URL ein:“ eingetragen und zusätzlich die Option „Logo in QR-Code einbinden“ ausgewählt. Eingebunden wurde das Logo der Gemeinde Hude - „Hude - Zum Malen schön“.



Abbildung D.1.: EC-Automaten



Abbildung D.2.: Einkaufen



Abbildung D.3.: Gastronomie



Abbildung D.4.: Gesundheit



Abbildung D.5.: Hude-Info



Abbildung D.6.: Hude-Perfekt



Abbildung D.7.: Parkmöglichkeiten



Abbildung D.8.: Fotojagd: Hude



Abbildung D.9.: Fotojagd:
Klosterbezirk



Abbildung D.10.: Fotojagd:
Skulpturenufer



Abbildung D.11.: Planeten-Abitur
A



Abbildung D.12.: Planeten-Abitur
B



Abbildung D.13.: Planeten-Abitur
C



Abbildung D.14.: Artesischer
Brunnen



Abbildung D.15.: Hasbruch



Abbildung D.16.: Klosterbezirk:
Klosterruine



Abbildung D.17.: Klosterbezirk:
Klosterschänke



Abbildung D.18.: Klosterbezirk:
Museum



Abbildung D.19.: Klosterbezirk



Abbildung D.20.: Klosterbezirk:
Torkapelle



Abbildung D.21.: Klosterbezirk:
Wassermühle



Abbildung D.22.: Planetenlehrpfad:
Erde



Abbildung D.23.: Planetenlehrpfad:
Jupiter



Abbildung D.24.: Planetenlehrpfad:
Mars



Abbildung D.25.: Planetenlehrpfad:
Merkur



Abbildung D.26.: Planetenlehrpfad:
Neptun



Abbildung D.27.: Planetenlehrpfad:
Pluto



Abbildung D.28.: Planetenlehrpfad: Saturn
Abbildung D.29.: Planetenlehrpfad: Sonne
Abbildung D.30.: Planetenlehrpfad: Uranus



Abbildung D.31.: Planetenlehrpfad: Venus
Abbildung D.32.: Sehenswürdigkeiten: Venus
Abbildung D.33.: Skulpturenufer: Venus



Abbildung D.34.: Sonnenuhr
Abbildung D.35.: Megalithkultur
Abbildung D.36.: Vielstedter Bauernhaus



Abbildung D.37.: Wanderweg Huder Bach



Abbildung D.38.: Wittemoor



Abbildung D.39.: Zeitstrahl 2000



Abbildung D.40.: Ziehbrunnen



Abbildung D.41.: Spielplätze



Abbildung D.42.: Ferienhaus Aschenbeck



Abbildung D.43.: Ferienhaus Hude



Abbildung D.44.: Ferienwohnung Alte Bäckerei



Abbildung D.45.: Ferienwohnung Aschenbeck



Abbildung D.46.: Ferienwohnung Burgmannshof



Abbildung D.47.: Ferienwohnung Dat Lüttje Huus



Abbildung D.48.: Ferienwohnung Eichkatz



Abbildung D.49.: Ferienwohnung F. Hattendorf



Abbildung D.50.: Ferienwohnung Forsthaus Hasbruch



Abbildung D.51.: Ferienwohnung Haus Lichtblick



Abbildung D.52.: Ferienwohnung Hof Stolle



Abbildung D.53.: Ferienwohnung Klosterschänke



Abbildung D.54.: Ferienwohnung M. Kallisch



Abbildung D.55.: Ferienwohnung Seekermann



Abbildung D.56.: Ferienwohnung U. Kallisch



Abbildung D.57.: Gästehaus Wobig



Abbildung D.58.: Gästehaus Würdemann



Abbildung D.59.: Hotel Burgdorf



Abbildung D.60.: Hotel Garni



Abbildung D.61.: Hotel Klosterschänke



Abbildung D.62.: Jugendherberge



Abbildung D.63.: Pension Hof Heinemann



Abbildung D.64.: Pension Paulus



Abbildung D.65.: Unterkünfte



Abbildung D.66.: Wohnmobilstellplatz



Abbildung D.67.: Veranstaltungen