

Entwicklungsprojekt interaktive Systeme

WS 18/19

Modellierung

PlaceToBe

Dozenten

Prof. Dr. Gerhard Hartmann

Prof. Dr. Kristian Fischer

Betreuer

Robert Gabriel

Projekt

von

Elena Correll

Mike Klement

Entwicklungsprojekt interaktive Systeme	1
Modellierung	1
Dozenten	1
Betreuer	1
Projekt von	1
1. Verstehen und Festlegen des Nutzungskontextes	6
1.1 Stakeholderanalyse	7
1.2 Umfrage	9
1.3 User Profiles	9
Benutzergruppe Studenten	9
Benutzergruppe Familie	10
Benutzergruppe Senioren	11
1.4 Personas	11
1.5 Nutzungsumgebung	15
1.6 Use Case	15
Use Case 1: bestimmten Ort recherchieren	16
Use Case 2: Wohnort-Finder anwenden	17
Use Case 3: Informationen einsehen	18
Use Case 4: Ort anreichern	19
1.7 Szenario	20
2. Festlegen der Nutzungsanforderungen	21
2.1 funktionale Anforderungen	21
2.2 qualitative Anforderungen	23
2.3 organisatorische Anforderungen	24
3. Erarbeiten von Gestaltungslösungen zur Erfüllung der Nutzungsanforderungen	25
3.1 Gestaltungslösung der einzelnen Arbeitsaufgaben	25
3.2 Allgemeine Designentscheidungen	32
4. Evaluation von Gestaltungslösungen	33
4.1 Christine Correll	33
4.2 Bastian Schmalbach	34
4.3 Zusammenfassung und Auswertung der Evaluation	35
5. Iteration der Schritte 2-5	35
5.1 Anforderungen	35
5.2 Verfeinerung der Gestaltungslösungen	36
1.Systemarchitektur	39

1.1 Der Server	40
1.2 Der Client	40
1.3 Verwendete Protokolle	40
1.4 Verwendete Cloud	40
1.5 asynchrone Kommunikation	40
1.6 synchrone Kommunikation	41
1.7 Datenbank	41
2. Begründung der verwendeten API's und Betrachtung von Alternativen	42
3. Anwendungslogik	43
3.1 Clientseitige Anwendungslogik	43
Suchanfrage: Ein Benutzer priorisiert Kriterien, die ihm bei der Wohnortwahl (Kultur, Infrastruktur, Wetter) besonders wichtig sind. Der Client berechnet daraus eine Abfrage, die er synchron an den Server sendet. Als Antwort werden ein Ort und dessen Informationen ausgegeben.	43
3.2 Serverseitige Anwendungslogik	43
3.2.1 Zuweisungsalgorithmus	43
3.2.2 Wetterwertberechnung	44
Server (Aus dem Konzept)	44
4. Datenstruktur	44
4.1. Wohnungsort	45
4.2. Kulturanzeige	45
4.3. WetterAPI	45
4.4. GoogleMapsAPI	45
4.5. Infrastrukturwerte	45
4.6. Zuweisungsalgorithmus	46
4.7. WikidataDB	46
5. Datenhaltung in der DB	46
5.1 Vorteile und Nachteile der dokument basierten Datenhaltung	47
6. Programmiersprachen	47
6.1 Clientseitige Programmiersprache	48
6.2 Serverseitige Programmiersprache	48
7. Entwicklungsumgebung	48
7.1 Clientseitige Umgebung	49
7.2 Serverseitig Umgebung	49
7.3 Zusammenspiel der beiden Umgebungen	49
8. Vielfalt der Endgeräte	50
8.1 Begründung für die Benutzung von Android	50

9.Reflektion des POCs	50
10.Fazit	51
Literaturverzeichnis	52
Anhang	52

I Einleitung

In der Modellierungsphase haben wir unserem Projekt endlich einen Namen gegeben: PlaceToBe. Der Name soll aussagen, dass unser System fähig ist den perfekten Wohnort für einen zu finden, an dem man sich wohlfühlen wird.

Das Projekt PlaceToBe beschäftigt sich mit dem allgemeinen Problem der Landflucht in Deutschland und verknüpft dieses mit dem Entscheidungsproblem bei der Wohnortsuche. Dazu soll ein System entwickelt werden, das Orte Deutschlands repräsentiert, Benutzer bei der Wahl des Wohnortes unterstützt und dabei die Aspekte der Landflucht beachtet.

In diesem Dokument werden unter Kapitel II Methodischer Rahmen der Mensch Computer Interaktion die Designlösungen erarbeitet und unter Kapitel III Modellierung der webbasierten Anwendung Entscheidungen zur Architektur und dem technischen Hintergrund getroffen.

II Methodischer Rahmen der MCI

Im Konzept haben wir uns für das Vorgehen nach dem Modell ISO 9240-210 entschieden. Dies ist ein menschenzentriertes Vorgehen, was bedeutet, dass die Anforderungen und Gestaltungslösungen an den Erfordernissen der Benutzer orientiert sind. Sie sind auch in die Evaluation miteinbezogen, die iterativ durchgeführt wird.

Die ISO gibt folgende Schritte vor, die in diesem Kapitel erarbeitet werden:

1. Planen des menschenzentrierten Gestaltungsprozesses (siehe Konzept)
2. Verstehen und Festlegen der Nutzungskontexte
3. Festlegen der Nutzungsanforderungen
4. Erarbeiten von Gestaltungslösungen zur Erfüllung der Nutzungsanforderungen
5. Evaluation von Gestaltungslösungen anhand der Anforderungen an den Nutzer und die Nutzung in den Nutzungskontexten
6. Iteration der Schritte 2-5, soweit die Evaluationsergebnisse Bedarf hierfür anzeigen

1. Verstehen und Festlegen des Nutzungskontextes

Zu einem umfangreichen Verständnis des Nutzungskontextes gehört die Analyse der Benutzer sowie anderer Interessengruppen, deren Ziele und Arbeitsaufgaben, sowie die Umgebung in der das System laufen wird.

Dazu wird eine Stakeholderanalyse durchgeführt, User Profiles und Personae erstellt. Die dadurch repräsentierten Benutzer werden später eine entscheidende Rolle bei der Festlegung von Nutzungsanforderungen und bei der Beantwortung von Gestaltungsfragen spielen.

Die Aufgaben der Benutzer werden durch Use Cases festgelegt und durch Szenarien narrativ beschrieben.

1.1 Stakeholderanalyse

Eine erste Identifizierung der Stakeholder im Konzept wird hier erweitert.

Bezeichnung	Bezug zum System	Objektbereich	Erfordernis/Erwartungen
Wohnort-Suchender	Interesse	gesamtes System	Finden eines geeigneten Wohnortes, der seine persönlichen Kriterien erfüllt
	Interesse	Daten	Interesse an Informationen zu einem Ort: Lage, Verkehrsanbindung, durchschnittlicher Mietpreis, Wetter, Einkaufsmöglichkeiten, Freizeitaktivitäten
	Anrecht		keine Verletzung des Datenschutzes, Möglichkeit zur Datenabfrage
	Anspruch		Ausgabe korrekter und aktueller Informationen

Amt Ort	Interesse Anspruch	gesamtes System Daten	Verbesserung der finanziellen Lage des Ortes korrekte, faire Beurteilung und Darstellung des Ortes guter Umgang mit städte-spezifischen Daten
Regierung	Interesse	gesamtes System	Verbesserung der Landflucht Quote
Verein	Interesse Anrecht	gesamtes System Daten	Gewinnung von interessierten Mitgliedern korrekter Umgang mit persönlichen Daten
Grundstücksb esitzer	Interesse Anspruch	gesamtes System Daten	Verkauf/Verpachtung von Grundstücken korrekter Umgang mit persönlichen Daten
Schnittstellen- Besitzer	Interesse Anteil	gesamtes System	sinnvolle Nutzung der Schnittstelle an eingebundenen Funktionen bei gewerblichem Gebrauch

1.2 Umfrage

Um die Benutzer besser verstehen zu können wurde eine Umfrage durchgeführt mit dem Titel „Ausziehen Umziehen Rumziehen“ (siehe Anhang). Diese haben wir an 40 Personen von unterschiedlichem Alter, in unterschiedlichen Lebenssituationen und mit unterschiedlichen Werten durchgeführt.

Die drei wichtigsten Erkenntnisse, die daraus gezogen werden konnten sind folgende:

- für den Benutzer besonders relevante Aspekte bei der Wohnortwahl: Einkaufsmöglichkeiten, Freunde in der Umgebung, Freizeitangebot, die “Stimmung” und Art der Leute in der Nachbarschaft und allgemein im Ort
- Nicht die Quantität von Bars, Restaurants/Freizeitangeboten ist entscheidend, sondern die Qualität

Besonders der zweite Punkt ist wichtig für unser System. Die Freizeitaktivitäten, die der Benutzer hat und Bars/Restaurants/Cafes, die seinem Geschmack entsprechen, können dann jeweils hervorgehoben werden, falls sie in einem Ort existieren. So können auch Dörfer und kleinere Städte attraktiv werden, was das strategische Ziel der Landflucht-Minimierung dienen kann.

1.3 User Profiles

Die Daten aus der Umfrage, Gesprächen und Beobachtungen konnten verschiedene Benutzergruppen und relevante Eigenschaften erörtert werden. Die User Profiles, die daraus entstanden sind wurden von mit potentiellen Benutzern evaluiert und überarbeitet.

Die Auswahl der User Profiles ist nicht vollständig, repräsentiert jedoch den Großteil der Benutzer.

Benutzergruppe Studenten

Alter	18 - 28
Beruf	Schüler

Wohnort	Land/Stadt
Kinder	nein
Einkommen	600€ - 1000€
verfügbare Technologie	Smartphone
Motiv für einen Umzug in einen anderen Wohnort	von zuhause ausziehen, Ausbildungs/Studienplatz ist in einem anderen Ort
Vorteile eines Wohnortes in ländlicher Region	weniger Ausgaben für Miete, familiäre Gegend
Nachteile eines Wohnortes in ländlicher Region	ggf. größere Entfernung vom Studien/Ausbildungsplatz, weniger Ausgelmöglichkeiten

Benutzergruppe Familie

Alter	25-45
Beruf	
Wohnort	Stadt
Kinder	1-5
Einkommen	3000 - 5000 €
verfügbare Technologie	Smartphone, Tablet, Laptop
Motiv für einen Umzug in einen anderen Wohnort	zu teuer/laut/eng in der Stadt
Vorteile eines Wohnortes in ländlicher Region	weniger Ausgaben für Miete, familiäre Gegend, Natur, Möglichkeit ein Haus zu bauen, mehr Platz für Kinder

Nachteile eines Wohnortes in ländlicher Region	ggf. größere Entfernung vom Arbeitsplatz, weniger Ausgelmöglichkeiten
--	---

Benutzergruppe Senioren

Alter	60 - 100
Beruf	/
Wohnort	Land/Stadt
Kinder	nein
Einkommen	600€ - 1000€
verfügbare Technologie	Smartphone
Motiv für einen Umzug in einen anderen Wohnort	in die Nähe der Familie ziehen
Vorteile eines Wohnortes in ländlicher Region	Natur, Ruhe
Nachteile eines Wohnortes in ländlicher Region	medizinische Versorgung evtl. schlechter

1.4 Personas

Repräsentativ für die Benutzergruppen "Studenten", "Familie" und "Senioren" wurden die folgenden drei Personae erstellt. Dies wird als sinnvoll erachtet um sich besser in die Benutzer hineinversetzen zu können und im Gestaltungsprozess aus den Augen dieser argumentieren zu können, da man sich durch Personae ein lebendiges narratives Bild der zukünftigen Aufgabenbewältigung der Benutzer machen kann.



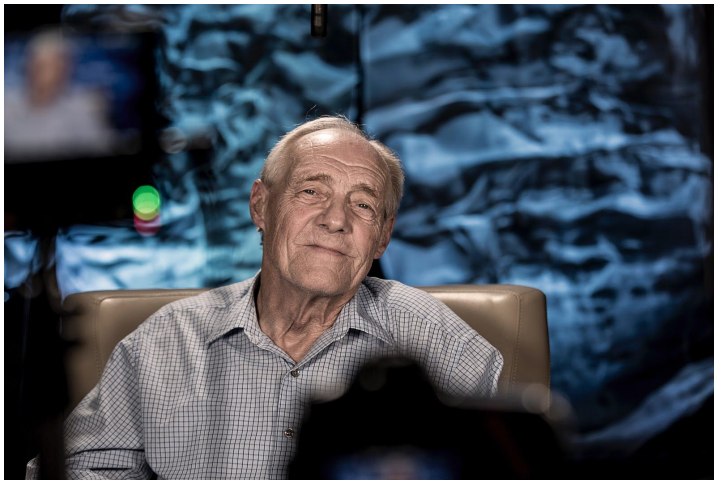
Name: Kay Gugelhupf
Age: 25
Job: Wirtschaftsingenieur Student (Master)
Wohnort: Nürnberg
Budget: 900€ / Monat
verfügbare Technik: Smartphone, Laptop
physische Einschränkungen: /
Kinder: /
Hobbies: skaten, reisen
Ziele: erfolgreicher Ingenieur werden
Erwartungen von PlaceToBe: Einen Ort in der Nähe von München finden, den er sich leisten kann

Kay ist 25. In seiner Freizeit geht er gerne skaten oder spielt mit seinen Jungs online Computerspiele. Kay studiert an der FH Nürnberg Wirtschaftsingenieurwesen und möchte nächstes Semester seine Masterarbeit in einem Unternehmen schreiben. Dazu hat er sich mehrmals in München beworben und wurde schließlich bei Siemens genommen, worüber er sich sehr gefreut hat. Als er jedoch die Mietpreise für Einzimmerwohnungen in München gesehen hat, hat er sich nicht mehr so gefreut. Er weiß einfach nicht wie er sich das bloß leisten soll.



Name: Pia Padowski
Alter: 41
Job: Journalistin
Wohnort: Frankfurt
Budget: 7000 € (zusammen mit ihrem Ehemann Dima)
Technik: Smartphone, PC, Tablet
physische Einschränkungen: /
Kinder: 2 (Marie (5) und Jonas (8))
Hobbies: lesen, reiten
Ziele: ein eigenes Haus haben
Erwartungen von PlaceToBe: einen Ort finden, an dem Sie sich ein Haus kaufen können, der Schulen hat, wo sie ihren Interessen nachgehen kann und nicht zu lang nach Frankfurt braucht

Pia lebt mit ihrem Mann Dima und ihren gemeinsamen zwei Kindern in Frankfurt. Pia ist dort Journalistin für die örtliche Zeitung. Sie haben ein gutes Einkommen und sind auch sehr zufrieden in Frankfurt in ihrer 4 - Zimmer Wohnung. Pia und ihr Mann hatten schon seit sie geheiratet haben den Traum ein eigenes Haus außerhalb der Stadt zu kaufen und haben darauf hin gespart. Da ihre jüngere Tochter bald in die Schule kommt ist es Ihnen wichtig einen Ort mit Schule zu finden. Pia möchte außerdem ihren Job als Journalistin nicht aufgeben. Der Job erfordert es Interviews in und rund um Frankfurt zu führen, zu Veranstaltungen zu gehen.



Name: Ralf Ranke
ALter: 73
Job: Koch (in Rente)
Wohnort: Halver (NRW)
Budget: 5000€/Monat
verfügbare Technik: Laptop
physische Einschränkungen: Weitsichtig (2 Dioptrin)
Kinder: 1 (Elisa (43))
Hobbies: angeln
Ziele: in Ruhe alt werden, ein Kochbuch schreiben
Erwartungen von PlaceToBe: einen Ort finden, der näher an seiner Familie ist und in dem er angeln gehen kann

Ralf ist 73 und wohnt in einem kleinen Dorf namens Halver. Seit seine Frau vor einem Jahr verstorben ist fühlt er sich einsam. Seine Tochter Elisa wohnt in Köln gezogen. Seit dem Tod ihrer Mutter versucht Sie Ralf dazu zu überreden zu ihr in die Nähe zu ziehen, da er so auch öfter seine Enkelkinder sehen könnte, aber Ralf will sein gewohntes Umfeld nicht zurücklassen. Vor Allem den See an dem er jeden Sonntag angeln geht wird er vermissen. Er hat schon immer auf dem Land gelebt und sagt von sich selbst, dass er kein Stadtmensch ist. In Halver leben immer mehr ältere Menschen und es sieht so aus, als könnte dieser Ort aussterben. Das macht Ralf traurig, denn er ist dort aufgewachsen.

1.5 Nutzungsumgebung

Die Umgebung in der die Benutzer das System verwenden werden kann überall sein, wo Sie mit dem Smartphone sind und eine Internet Verbindung haben. Es kann gut sein, dass sie die Orte zusammen mit Freunden oder der Familie anschauen wollen, daher wäre es sinnvoll PlaceToBe auch auf einem größeren Screen zu verwenden.

1.6 Use Case

Die Aufgaben der Benutzer sind:

- Recherche zu einem Wohnort (use case: bestimmten Ort recherchieren)
- Finden eines geeigneten Wohnortes (use case: Wohnort-Finder anwenden)
- Bewerben und bewerten von Wohnorten (use case: Ort anreichern)
- Informationen zu einem Ort durchsehen (use case: Informationen einsehen)

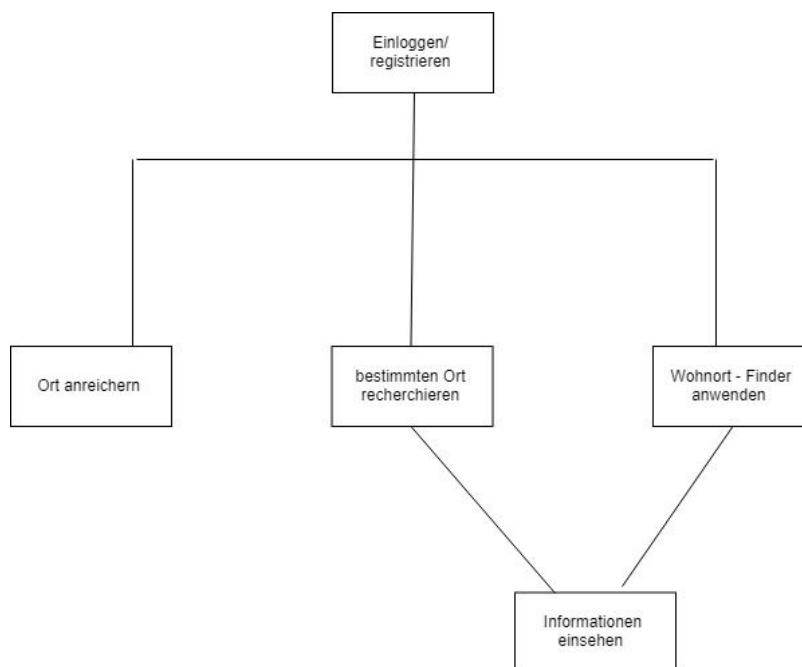


Diagramm 1: Use Case Map

Im Folgenden möchten wir näher auf die einzelnen Use cases eingehen. (bis auf Use Cas “Einloggen/Registrieren”)

Use Case 1: bestimmten Ort recherchieren

Name	bestimmten Ort recherchieren	
Auslösender Aktor	Benutzer	
Weitere Aktoren		
Auslöser	Benutzer interessiert sich für einen bestimmten Ort	
Vorbedingung	Der Benutzer kennt den Namen/ die Lage des Ortes	
Nachbedingung	Der Benutzer hat den Ort gefunden	
Haupt-Szenario	<i>Benutzer</i>	<i>System</i>
	Der Benutzer gibt den Ort in die Suchleiste ein	
		Das System gibt Orte an die zu dieser Suche im System registriert sind
	Der Benutzer wählt den gesuchten Ort aus	
		Die Übersicht des Ortes wird ausgegeben
Alternativ-Szenario	Der Benutzer wählt ein Bundesland aus der Deutschland Karte	

		Das System zeigt das Bundesland an
	Der Benutzer wählt den gesuchten Landkreis	
		Das System zeigt den Landkreis an
	Der Benutzer wählt einen Ort	
		Die Übersicht des Ortes wird ausgegeben

Use Case 2: Wohnort-Finder anwenden

Name	Wohnort-Finder anwenden	
Auslösender Akteur	Wohnort-Suchender	
Weitere Akteure		
Auslöser	Benutzer ist auf der Suche nach einem neuen Wohnort	
Vorbedingung	Der Benutzer sucht einen Wohnort in einem Gebiet, dass im System ist	
Nachbedingung	Das System hat dem Benutzer Vorschläge zu Wohnorten ausgegeben	
Haupt-Szenario	<i>Benutzer</i>	<i>System</i>
	Benutzer gibt an, dass er zum Wohnort Finder möchte	

	Benutzer füllt erforderliche Fragen aus	
		System sucht nach passenden Orten
		System gibt Übersicht aller passenden Orte aus
	Benutzer klickt einen Ort an	
		System gibt Übersicht des Ortes aus

Use Case 3: Informationen einsehen

Name	Informationen einsehen	
Auslösender Aktor	Benutzer	
Weitere Aktoren		
Auslöser	Benutzer interessiert sich für einen Ort	
Vorbedingung	Benutzer hat den Ort bereits aufgerufen	
Nachbedingung	Der Benutzer hat Informationen zum Ort erhalten	
Haupt-Szenario	<i>Benutzer</i>	<i>System</i>
		Das System zeigt die Übersicht des Ortes mit den Informations-Kategorien an
	Benutzer wählt eine Kategorie aus	

		die jeweiligen Informationen werden angezeigt
	Benutzer klickt auf "zur Galerie"	
		System zeigt Galerie Seite an

Use Case 4: Ort anreichern

Name	Wohnort anreichern	
Auslösender Aktor	Benutzer	
Weitere Akteure		
Auslöser	Benutzer will etwas zu einem Wohnort hochladen	
Vorbedingung	Benutzer wohnt in diesem Ort/ war schon einmal sem Ort	
Nachbedingung	Benutzer hat Video/ Bild/ Kommentar hochgeladen	
Haupt-Szenario	<i>Benutzer</i>	<i>System</i>
	Benuter gibt an, dass er etwas hochladen möchte	
		System gibt Seite aus, in der der Benutzer den Ort auswählen kann
	Benutzer sucht nach einem Ort	

		Ort wird ausgegeben mit der Funktion ein Bild hochzuladen
	Benutzer lädt ein Bild/Kommentar/Video hoch	

1.7 Szenario

Kay weiß nicht ob er lachen oder weinen soll. vor ein paar Tagen hat eine Zusage von Siemens bekommen: er kann dort seine Masterarbeit absolvieren. Allerdings ist das in München und das ist ja bekanntermaßen eine verdammt teure Stadt. Die Miete für eine Einzimmerwohnung dort würde sein Budget sprengen. Er hat schon zahlreiche Wohnungsanbieter Seiten durchforstet, aber ohne Erfolg.

Sein Kumpel Chris hat ihm empfohlen die App PlaceToBe herunterzuladen um zu sehen ob es einen geeigneten Vorort für ihn gibt. Kay ist skeptisch. Er denkt dass er zu hohe Ansprüche für einen kleinen Ort hat, denn er möchte unbedingt skaten und ohne Auto schnell in die Arbeit gelangen.

Er öffnet die App und sieht eine Karte von deutschland vor sich. Darunter ist ein Button "wohnort finder". Es erscheinen verschiedene Fragen, die er beantwortet. Als er damit fertig ist wird ihm eine Karte angezeigt: Im Mittelpunkt der Karte ist Siemens, außenrum werden ihm 3 Orte vorgeschlagen. Kay ist immer noch skeptisch, aber seine Neugierde wurde geweckt. Er klickt auf die erste stadt "Erding" und eine Übersicht mit allgemeinen Informationen erscheint.

Kay sieht sich auch die anderen Orte an und merkt, dass man nicht unbedingt in der Stadt leben muss um einen Skateplatz zu Fuß erreichen zu können.

Kay ist zufrieden mit dem System.

2. Festlegen der Nutzungsanforderungen

Auf Basis des definierten Nutzungskontextes in Kapitel 2.8, sowie aus Erkenntnissen der Ergonomie und Mensch Computer Interaktion werden die Anforderungen an das System gestellt.

2.1 funktionale Anforderungen

Diese Anforderungen spezifizieren, welche Funktionen das Systemen haben muss, um den Bedürfnissen der Benutzer gerecht zu werden.

F10 geographische Daten

Das System muss die geographische Lage von Orten kennen.

F11 Darstellung geographischer Daten

Das System soll die Möglichkeit bieten alle Orte Deutschlands Übersichtlich darzustellen (bundesweit, bundeslandweit, landkreisweit).

F20 ortsspezifische Daten

Das System muss fähig sein allgemeine Informationen zu einem Ort (Name, Bundesland, Region, Einwohnerzahl, Mietpreisspiegel) zu kennen.

F30 Landflucht

Das System soll fähig sein anzugeben, ob und in welchem Maß der Ort von Landflucht betroffen ist.

F40 Wetterdaten

Das System soll Wetterinformationen zu einem Ort kennen.

F50 Wetter Durchschnitt

Das System soll fähig sein Wetterdurchschnittswerte zu einem bestimmten Ort zu berechnen (z.B. In Köln regnet es jeden fünften Tag.)

F60 Katastrophen

Das System soll fähig sein Naturkatastrophen die in einem Ort auftreten zu kennen.

F70 Entfernung

Das System muss fähig sein die Entfernung von einer Stadt zur nächsten Großstadt zu berechnen, sowie die Entfernung zum Arbeits/Studien/Ausbildungsplatz des Benutzers.

F80 Infrastruktur

Das System muss fähig sein Informationen zur Infrastruktur (öffentliche Verkehrsmittel, Einkaufsmöglichkeiten, Schulen/Studienangebote, medizinische Versorgung) eines Ortes zur Verfügung zu stellen.

F90 Freizeit

Das System soll fähig sein Freizeitangebote in einem Ort auszugeben.

F100 Entscheidungen

Das System muss fähig sein den Benutzer bei der Auswahl eines geeigneten Wohnortes zu unterstützen.

F110 kulturelle Daten

Das System soll fähig sein kulturellen Kriterien zu einem Ort aussagekräftig zu repräsentieren.

F120 Benutzerinteraktion

Das System muss dem Benutzer die Möglichkeit bieten seine Meinung zu einem Ort abzugeben.

F130 multimediale Repräsentation

Das System muss die Möglichkeit bieten Daten (Bilder, Audio, Video) hochzuladen.

F140 Daten

Das System muss fähig sein zwischen relevanten und irrelevanten Daten zu unterscheiden.

F150 Kategorisierung

Das System muss in der Lage sein alle Informationen zu einem Ort zu verknüpfen.

F160 Eingabe Bestätigung

Das System muss zur Fehlervermeidung auf eine Eingabe Bestätigung des Benutzers warten.

F170 Undo

Das System soll dem Benutzer die Möglichkeit bieten einfach einen seiner Schritte rückgängig zu machen.

F180 Übersichtlichkeit

Das System soll dem Benutzer einen Überblick über alle möglichen Funktionen geben können.

F190 Login

Das System soll in fähig sein Benutzerdaten zu speichern.

F200 Suche

Das System soll die Möglichkeit bieten nach einem Ort zu suchen.

F210 Bewertung

Das System soll dem Benutzer die Möglichkeit bieten Informationen zu einem Ort (Bilder, Videos) zu bewerten.

F220 Bewerben

Das System muss fähig sein Anreize für ein Leben in ländlichen Regionen zu geben.

F230 Suchradius

Das System soll fähig sein herauszufinden, in welchem Radius der Benutzer nach einer Wohngegend suchen will.

F240 Wohnort Ausgabe

Das System muss fähig sein passende Wohnorte im gesuchten Radius auszugeben.

F250 individuelle Ausgabe

Das System soll fähig sein die Informationen, die einen Benutzer interessieren individuell anzuzeigen.

F260 vorteilhafte Ausgabe

Das System muss fähig sein die individuellen Vorteile eines Ortes hervorzuheben.

2.2 qualitative Anforderungen

Q10 Zuverlässigkeit

Das System soll immer erreichbar sein, d.h. die Wahrscheinlichkeit der Nichtverfügbarkeit soll bei 1% liegen.

Q20 Aktualität

Die Informationen zu einem Ort müssen jährlich aktualisiert werden.

Q30 Geschwindigkeit

Das System soll auf eine Benutzereingabe innerhalb von 30 Sekunden reagieren.

Q40 Speicher

Das System soll einen Speicherbedarf von 100 MB nicht übersteigen.

Q50 Sicherheit

Das System muss sicherstellen, dass der Datenschutz eines Benutzers nicht verletzt wird.

Q60 Ergonomie

Die Systemsprache ist deutsch und englisch..

Q70 Verständlichkeit

Die Eingabeaufforderungen und Interaktionselemente müssen verständlich geschrieben werden.

Q80 Feedback

Das System soll auf Feedback der Community reagieren.

Q90 Datenspeicherung

Das System muss fähig sein Daten persistent zu speichern.

Q100 Kompatibilität

Das System soll auf andere Länder anwendbar/übertragbar sein.

Q110 Portabilität

Das System soll auf allen Endgeräte laufen können.

Q120 Fehler

Das System soll Fehler von Benutzern abfangen können.

Q130 Wartbarkeit

Das System muss Änderungen zulassen können.

Q140 Hilfe

Das System soll die Möglichkeit bieten Hilfe anzubieten. Online Dokumentation und Hilfe

Q150 Metaphern

Das System soll Metaphern aus der Domäne der Wohnort-Suchenden verwenden.

Q160 Vibration

Das System soll bei wichtigen Mitteilungen eine Vibration auslösen.

Q170 Navigation

Das System soll die Möglichkeit bieten die Navigation mit der Tastatur durchzuführen.

Q180 Sound

Das System soll ein Signal mit einer Frequenz von 1000 bis 4000 Hz ausgeben.

Q190 Text

Das System muss Texte angemessen groß und mit Abständen darstellen.

Q200 Farbe

Die Farben, die im System verwendet werden, müssen so gewählt sein, dass der Kontrast bei Farbenblindheit hoch genug ist.

Q210 Symbole

Das System muss die Bedeutung von Symbolen dokumentieren.

2.3 organisatorische Anforderungen

O10 Dokumentation

Die Dokumentation des Projektes soll in deutsch erfolgen.

O20 Testen

Das System muss regelmäßig und sinnvoll getestet werden.

O30 Zeitplanung

Zwei Use Cases sollen bis zum 20.Januar implementiert sein.

3. Erarbeiten von Gestaltungslösungen zur Erfüllung der Nutzungsanforderungen

Für den Gestaltungsprozess wird die Interaktion zwischen dem Nutzer und dem System nach anerkannten Normen und Richtlinien, sowie Paradigmen und Metaphern gestaltet. Anschließend wird die Gestaltungslösung der Benutzungsschnittstelle durch benutzerzentrierte Evaluation verfeinert.

3.1 Gestaltungslösung der einzelnen Arbeitsaufgaben

Die Interaktion zwischen Benutzer und System wurden für ein Smartphone entworfen. Dabei wurden MockUps erstellt, die später noch evaluiert werden und daher nicht detailliert ausgeführt wurden. In Abbildung 2 sieht man drei Skizzen der Arbeitsaufgabe, in der ein Benutzer nach einem beliebigen Wohnort suchen kann. Abbildung 3 und 4 zeigen den ersten Prototyp davon.

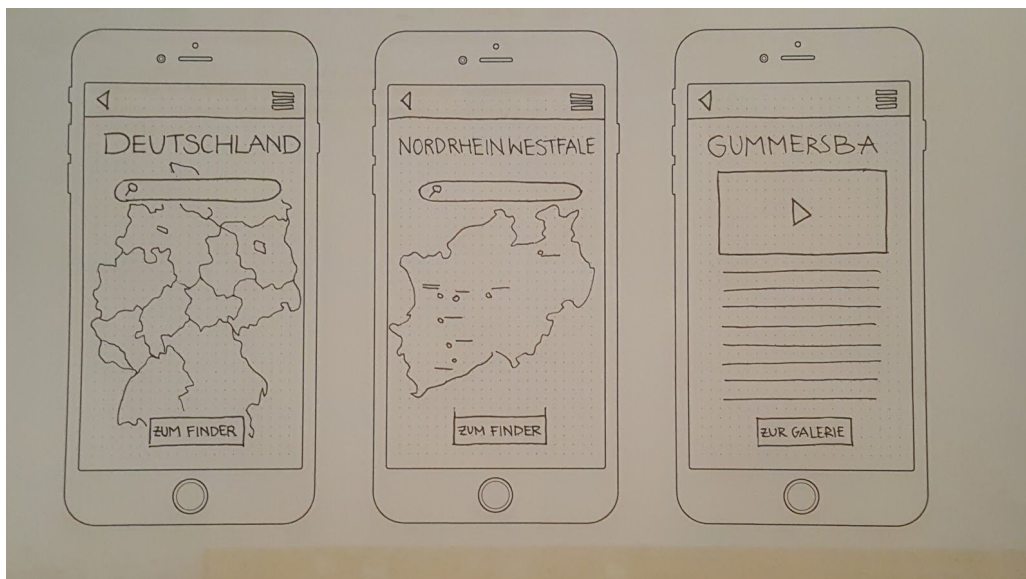


Abbildung 2: Skizzen "Wohnort suchen"



Abbildung 3: MockUp Deutschland



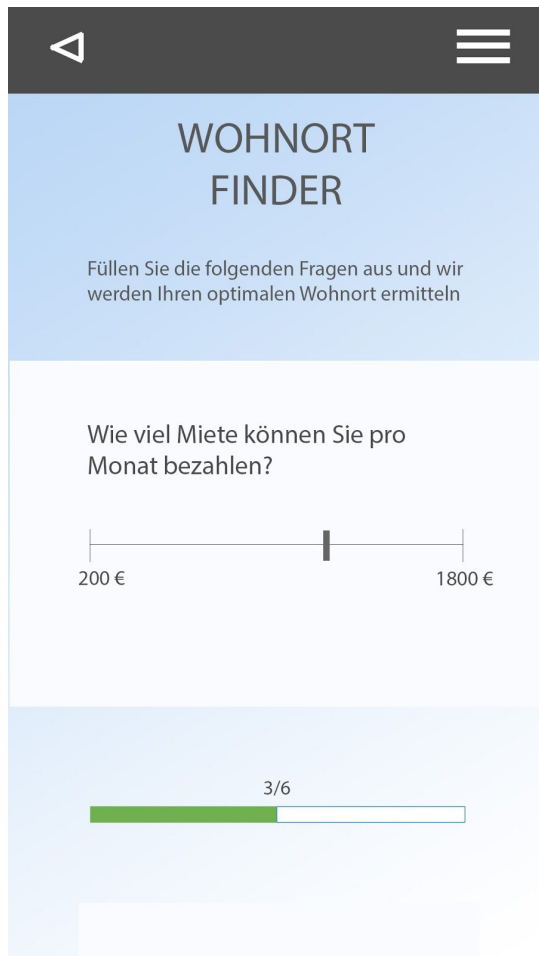
Abbildung 4: MockUp Bundesland

Um **F11 Darstellung geographischer Daten** zu erfüllen, wurde die Auswahl des Bundeslandes und der Region über eine Karte realisiert. Dadurch wird auch das Paradigma der direkten Manipulation verwendet. Alternativ dazu kann man über ein Textfeld nach einem Ort suchen. Dies wäre der effizientere Weg, er bietet jedoch kein visuelles Bild der geographischen Lage.

Wählt der User den Weg der direkten Manipulation wird er zu einer Karte des jeweiligen Bundeslandes und darauf zu der Karte der jeweiligen Region weitergeleitet. Auf der Karte soll der Name der einzelnen Felder mit Hover-Effekt angezeigt werden. Die 5-10 wichtigsten Städte des Bundeslandes oder der Region sollen eingezeichnet sein.

Wird ein spezieller Ort über die Karte ausgewählt oder über die Suchleiste aufgerufen, wird eine Übersicht mit Informationen ausgegeben.

Hat man keinen speziellen Ort, nach dem man sucht und möchte man Wohnorte vorgeschlagen bekommen, so ruft man den Wohnort Finder auf. Dies erfolgt über den Button “zum Wohnort-Finder” oder das Menü.



Der Wohnort Finder funktioniert, indem man verschiedene Fragen beantwortet. Aus den Antworten wird eine Suchanfrage an den Server erstellt. Das Ziel ist es, einen Ort für den Benutzer zu finden, der seinen individuellen Bedürfnissen entspricht.

Die Fragen dafür sind in zwei verschiedenen Formen aufgebaut. Entweder ein Texteingabefeld oder ein skalierbarer Regler (siehe Abbildung 5). Die Fragen werden nacheinander angezeigt und über einen Fortschrittsbalken wird visualisiert, wie viele Fragen der Benutzer noch zu beantworten hat.

Abbildung 5: MockUp Wohnort-Finder

Im Folgenden werden wir die 6 erforderlichen Fragen aufschreiben und begründen.

- Im Radius welchen Ortes wollen Sie einen Wohnort finden?

Diese Frage soll über ein Texteingabefeld beantwortet werden. Aus der Benutzermodellierung hat sich ergeben, dass die meisten Leute in der Nähe eines Studien/Ausbildungs/Arbeitsplatzes leben möchten oder ihre Wohnortwahl anhand der Lage des Wohnortes von Freunden/Familien festmachen. Der Ausgangspunkt kann eine Stadt oder ein Dorf sein.

- Wie weit sollte ihr Wohnort höchstens davon entfernt sein?

Diese Frage wird über einen Regler beantwortet, der von 5km zu 70km reicht. 70km wurden gewählt um auch abgelegenen Orten eine Chance zu geben im System aufgenommen zu werden.

- Wie viel Miete können Sie pro Monat bezahlen?

Auch diese Frage wird über einen Regler beantwortet, der von 200 bis 1800 Euro reicht. Diese Frage wird als relevant betrachtet, da die Benutzer wahrscheinlich eher weniger bezahlen möchten und somit eher Orte in ländlichen Regionen ausgegeben werden. So kann das strategische Ziel der Landflucht Minimierung besser erreicht werden.

- Wie wichtig ist Ihnen Natur?

Auch durch diese Frage soll das strategische Ziel besser erreicht werden. Der Regler wird von sehr wichtig zu unwichtig reichen. Je wichtiger desto mehr wird auf die Grünfläche/Quadratmeter in einem Ort geachtet. Außerdem werden die Attraktionen, die es in der Natur gibt bei der Übersicht eines Wohnortes hervorgehoben.

- Welche Freizeitaktivitäten üben Sie aus?

Diese Frage wird über ein Eingabefeld beantwortet. Auch diese Eingabe wird wieder auf Korrektheit geprüft. Hat der Benutzer z.B das Hobby Handball, kann nach Orten gesucht werden, in denen es einen Handballverein gibt. Die Vorschläge, die man im Anschluss erhalten wird, werden so weiter eingegrenzt.

- Wie wichtig ist Ihnen ein vielfältiges Freizeitangebot?

Hier wird wieder ein Regler angezeigt, der von irrelevant zu sehr relevant reichen wird. Durch diese Frage möchten wir herausfinden, was für Themen den Benutzer interessieren. Ist ihm beispielsweise ein vielfältiges Freizeitangebot wichtig, so werden bei der Städteübersicht, die Informationen und Attraktionen, die mit Freizeit zu tun haben priorisiert und eher angezeigt.

Die Fragen können noch erweitert werden, z.B ist für manche Benutzer die medizinische Versorgung wichtig oder dass es Schulen im Ort gibt. Aus Zeitgründen werden wir solche Aspekte jedoch nicht weiter ausführen.

Nachdem der Benutzer alle Fragen beantwortet hat sollen ihm Vorschläge ausgegeben werden. Auch hier haben wir zwei Design Alternativen. In der einen (Abb. 5) bekommt er direkt einen Wohnort mit den Informationen angezeigt und

kann zum nächsten Vorschlag über das Touchpad switchen, in der anderen (Abb. 6) erhält er eine Karte auf der die Lage der Orte und die Übereinstimmung angezeigt werden. Aus Sicht der Benutzer haben wir uns für die zweite Lösung entschieden. Für Pia (siehe Personae) ist es zum Beispiel sehr relevant, wie weit der Wohnort von Frankfurt entfernt sein wird. Über die Karte kann sie dies mit einem Blick erfassen.



Abbildung 5: Skizze "Vorschläge"



Abb. 6: MockUp Vorschläge

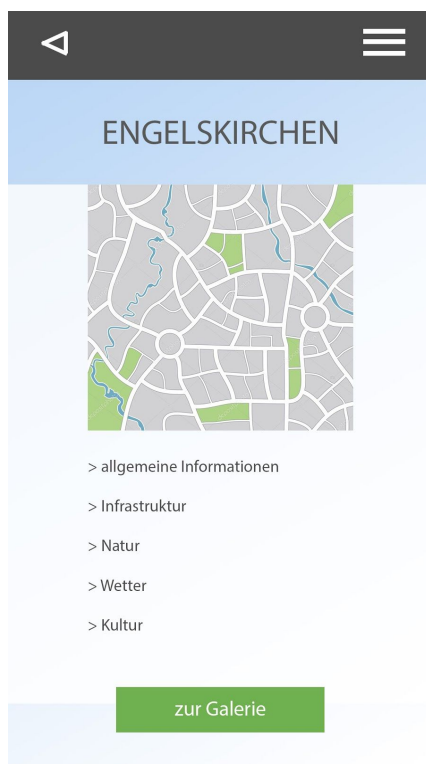
Abbildung 6 zeigt diese Karte mit Vorschlägen. In der Mitte der Karte ist jeweils der Ausgangsort eingetragen. Darum ist ein Kreis, der die gewünscht Höchst-Entfernung angeben soll. Die Orte, die nach dem System für den Benutzer geeignet werden sind mit der jeweiligen Übereinstimmung eingetragen.

Über den Wohnort-Button soll der Benutzer zurück zu den Fragen gelangen und diese eventuell anpassen.

Angenommen Pia erhält auf ihrer Karte um Frankfurt drei Vorschläge. Sie muss so keine Recherche mehr zu der Umgebung von Frankfurt machen, einzelne Orte raus suchen und zu diesen nochmals Informationen sammeln. Das System vereint diese

Teilaufgaben und kann sogar entscheiden welche Orte passend sind. Obwohl Pia Journalistin ist, soll unser System den Zeitaufwand für die normalerweise erforderlichen Arbeitsschritte halbieren.

Wenn ihr die Vorschläge ausgegeben werden sieht Sie wie viel Prozent ihren Angaben jeweils übereinstimmt und kann für nähere Informationen auf einen Ort klicken. So gelangt Sie zu der Übersicht mit Informationen des jeweiligen Ortes (Abbildung 7).



Die Darstellung dieser Informationen sieht man in Abbildung 7. Da Pia die Natur liebt, werden ihr zunächst die Attraktionen, die mit Natur zu tun haben angezeigt. Die Idee dahinter ist, dass dem Benutzer erst die Informationen angezeigt werden, die ihn wirklich interessieren. Der Benutzer wird sich so angesprochenener fühlen und die Anforderung **F250 individuelle Ausgabe** wird somit erfüllt.

Das Element das als erstes ins Auge fällt ist die Karte, die einen Überblick über den jeweiligen Ort aufzeigen wird. Man kann so direkt sehen, wie das Verhältnis von bebauter Fläche zu Grünfläche ist. Zudem werden wir mit verschiedenen Farben anzeigen wo sich Lokale/Restaurant, Bars/Clubs und Sportplätze/Sportvereine befinden. So kann der Benutzer direkt einen Eindruck vom Ort vermittelt werden

Abbildung 7: MockUp Übersicht

Die Rubriken und die darin enthaltenen Informationen, die angezeigt werden wurden aus den Anforderungen abgeleitet und wie folgt zusammengesetzt:

Informationen in der Städte Übersicht

Allgemeine Informationen	Bundesland Region Einwohnerzahl Bevölkerungsdichte Mietspiegel
Infrastruktur	öffentliche Verkehrsmittel

	Einkaufsmöglichkeiten
Natur	Fauna rund um den Ort Attraktionen/Sehenswürdigkeiten in der Natur
Wetter	aktuelles Wetter wie oft regnet es/scheint die Sonne im Durchschnitt pro Woche
Kultur	Sehenswürdigkeiten Ausgelmöglichkeiten (Cafés, Restaurants, Bars, Clubs)
Aktivitäten	Freizeitangebote (Skateplatz, Schwimmbad, Volleyballverein, ..)

Über den “zur Galerie” Button gelangt man zu einer Übersicht, in der man als erstes das Video, das zum Ort erstellt wurde angezeigt wird. Darunter findet man Bilder/Videos, die Benutzer zu diesem Ort hochgeladen haben. Man kann den Bildern/Video ein Like da lassen. Je mehr Likes ein Bild/Video hat, desto höher ist die Wahrscheinlichkeit, dass es in das Hauptvideo des Ortes gelangt. Über den grünen Button (siehe Abbildung 8) kann man selbst ein Bild/Video hochladen.



Abbildung 8: Mock Up Galerie

3.2 Allgemeine Designentscheidungen

In jedem Screen ist ein Header vorhanden, der das Element des Hamburger Icons beinhaltet um zum Menü zu gelangen und einem Dreieck, das als zurück Button dient. Dies erfüllt die Anforderungen **F170 Undo** und **F180 Übersichtlichkeit**.

Die Kombination der Farben wurde so gewählt, dass **Q200 Farbe** erfüllt wird, also auch Leute mit Farbenblindheit, alles anhand der Kontraste erkennen können.

Farben

hellblau: Der Hintergrund zeigt einen Verlauf von hell-blau zu weiß. Da blau für Freiheit steht unter anderem für Ruhe und Freiheit. Die Farbe wurde gewählt um dem Benutzer ein Gefühl von Entscheidungsfreiheit zu geben. Auch wenn versucht wird die ländlichen Regionen attraktiver zu machen, soll sich der Benutzer nicht bedrängt fühlen.

grün: Der Button und die Bestätigung sind in grün gehalten. Grün steht u.a für Hoffnung und Natur.

Die Verbindung zur Natur kann außerdem zur Erfüllung der Anforderung F20 Landflucht geben.

dunkelgrau: Schrift und Header sind in dunkelgrau. Diese Farbe ist sehr neutral und gut auf dem hellen Hintergrund lesbar.

orang: diese Farbe steht für Lebensfreude und Neugier. Die Elemente Orte sowie die prozentuale Übereinstimmung sind in dieser Farbe gewählt. Der Nutzer soll so neugierig auf die vorgeschlagenen Orte sein.

Schrift und Buttons

Aus Gründen der Integrität ist der Button und die Überschrift immer an der gleichen Stelle und in der gleichen Größe. Die Schrift wurde so gewählt, dass auch Personen mit leichter Seh-Einschränkung die Texte lesen können.

4. Evaluation von Gestaltungslösungen

Es wurde eine benutzerzentrierte Evaluation durchgeführt. Dazu wurden den Probanden die Gestaltungslösungen aus Kapitel 4 ausgedruckt auf Papier vorgelegt und beobachtet, wie sie mit dem System interagieren würden. Die Benutzer wurden dazu aufgefordert dabei laut mitzudenken. Im Folgenden werden die wichtigsten Aussagen zusammengefasst.

4.1 Christine Correll

Zur Person. Christine ist 39 hat 2 Kinder und lebt in Frankfurt. Sie ist gelernt Grafik Designerin. Sie übt diesen Beruf selbstständig aus und studiert nebenbei Lehramt für Informatik und Kunst.

Das System wäre für Sie interessant, da das Leben in Frankfurt immer teurer wird und Sie eine größere Wohnung bräuchte. Sie ist neugierig, ob es für Sie einen Ort geben kann, der noch ihren geliebten Stadt Flair hat und trotzdem bezahlbar ist.

Beim think-aloud sind folgende Schwierigkeiten und Vorschläge aufgekommen:

Mock Up Name	Schwierigkeiten (S), Anmerkungen(A), Vorschläge (V)
Deutschland	S: man versteht den Button "Wohnort Finder" nicht, da man denkt, die Karte ist der Wohnort Finder
Bundesland	S: Orientierung fehlt A: "hier man muss sich schon wirklich sehr gut mit Erdkunde auskennen" V: größere Städte einzeichnen
Wohnort-Finder	
Vorschläge	A: fühlt sich bevormundet durch die doppelte Information, dass Engelskirchen 92% Übereinstimmung hat
Übersicht	A: Die Informationen sind nicht spannend V: mehr das Attraktive von Ort hervorheben (bei

	Gummersbach z.B. die Aggertalsperre oder die Konditorei Hecker)
Galerie	S: Versteht nicht, warum hier der Button "Bild/Video hinzufügen" ist, wenn man doch diesen Ort gerade erst entdeckt
Allgemein	A: Die Idee ist gut und hat Potential A: Das Design ist nicht hip, sondern eher steril

4.2 Bastian Schmalbach

Feedback Bastian Schmalbach, 22, studiert Medieninformatik im 5. Semester. Wir haben ihn für das Feedback ausgewählt, da er für seine sehr ehrliche Kritik bekannt ist.

Mock Up Name	Schwierigkeiten (S), Anmerkungen(A), Vorschläge (V)
Deutschland	S: man versteht den Button "Wohnort Finder" nicht, da man denkt, die Karte ist der Wohnort Finder
Bundesland	/
Wohnort-Finder	/
Vorschläge	V: anstatt der Karte eine Liste der Wohnorte anzeig V: In der Karte die Entfernung in Fahrtzeit angeben
Übersicht	A: es ist ineffektiv, wenn man für jede Kategorie tippen muss um neue Informationen zu erhalten
Galerie	A: Die Anordnung der Bilder und Videos ist unattraktiv

4.3 Zusammenfassung und Auswertung der Evaluation

Problem Name	Erklärung	Lösung
Übersicht der Funktionen	man versteht beim Mock Up Deutschland nicht, dass es sich hier nicht um den Wohnort-Finder handelt	Einen Text einfügen, der dies deutlich macht
Die Übersicht der Städte	hat einen Interaktionsschritt zu viel, die Informationen sind nicht attraktiv	Die Informationen sollen direkt angezeigt werden, eventuell andere Darstellung der Informationen
Bild/Video hochladen	Der Button "Bild/Video hinzufügen" auf der Galerie Seite ist unlogisch	Das Hinzufügen eines Bildes oder eines Videos soll über das Menü erfolgen
Farbgebung	Die Farben machen das Design bieder	knallige Farben verwenden

5. Iteration der Schritte 2-5

Anhand der Evaluationen wurden Lösungen für die auftretenden Probleme entwickelt. Demnach werden die Anforderungen und Gestaltungslösungen nochmals überarbeitet.

5.1 Anforderungen

F270 Funktionsdarstellung

Das System muss ersichtlich machen, welche Funktionen es gibt und diese klar voneinander trennen.

F280 Name des Systems

Das System muss klar machen, was der Name des Systems ist.

Q220 Individualisierbarkeit

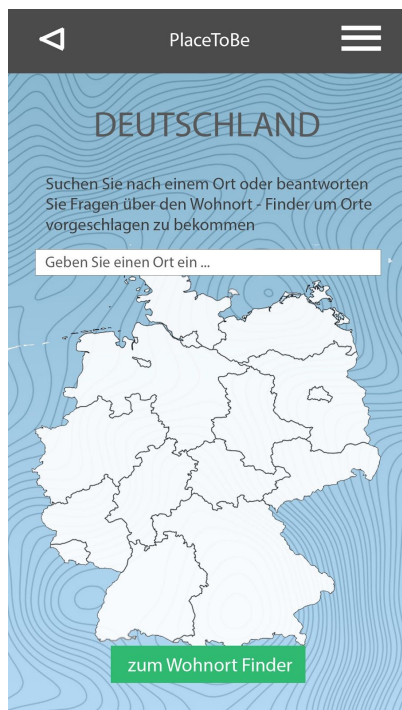
Das Design des Systems soll an die Zielgruppe anpassbar sein.

5.2 Verfeinerung der Gestaltungslösungen

Das Menü, dass über das Hamburger-Icon aufrufbar ist, wird folgende Punkt enthalten:

- Startseite
- Wohnort-Finder
- Bewerbe deinen Orte
- über PlaceToBe
- Hilfe
- Impressum

Durch die Anforderungen, die hinzugekommen sind wird der erste Screen wie folgt verändert.



Der Hintergrund hat einen dunkleren Blauton erhalten und der Button einen kräftigeren Grünton. Im Hintergrund ist eine Höhenkarte, die gut zum Geographie Thema unseres Systems passt.

Über die Suchleiste wurde ein Text hinzugefügt, der die beiden Funktionen "bestimmten Ort suchen" und "Wohnort - Finder ausführen" von einander unterscheidet.

Abb. 9: MockUp Deutschland 2



Dieser Screen soll aufzeigen, dass die Informationen, die ein Benutzer sehen möchte direkt angezeigt werden (man muss nicht mehr auf die Kategorie klicken um die Informationen zu erhalten).

Abb. 10: MockUp Übersicht 2

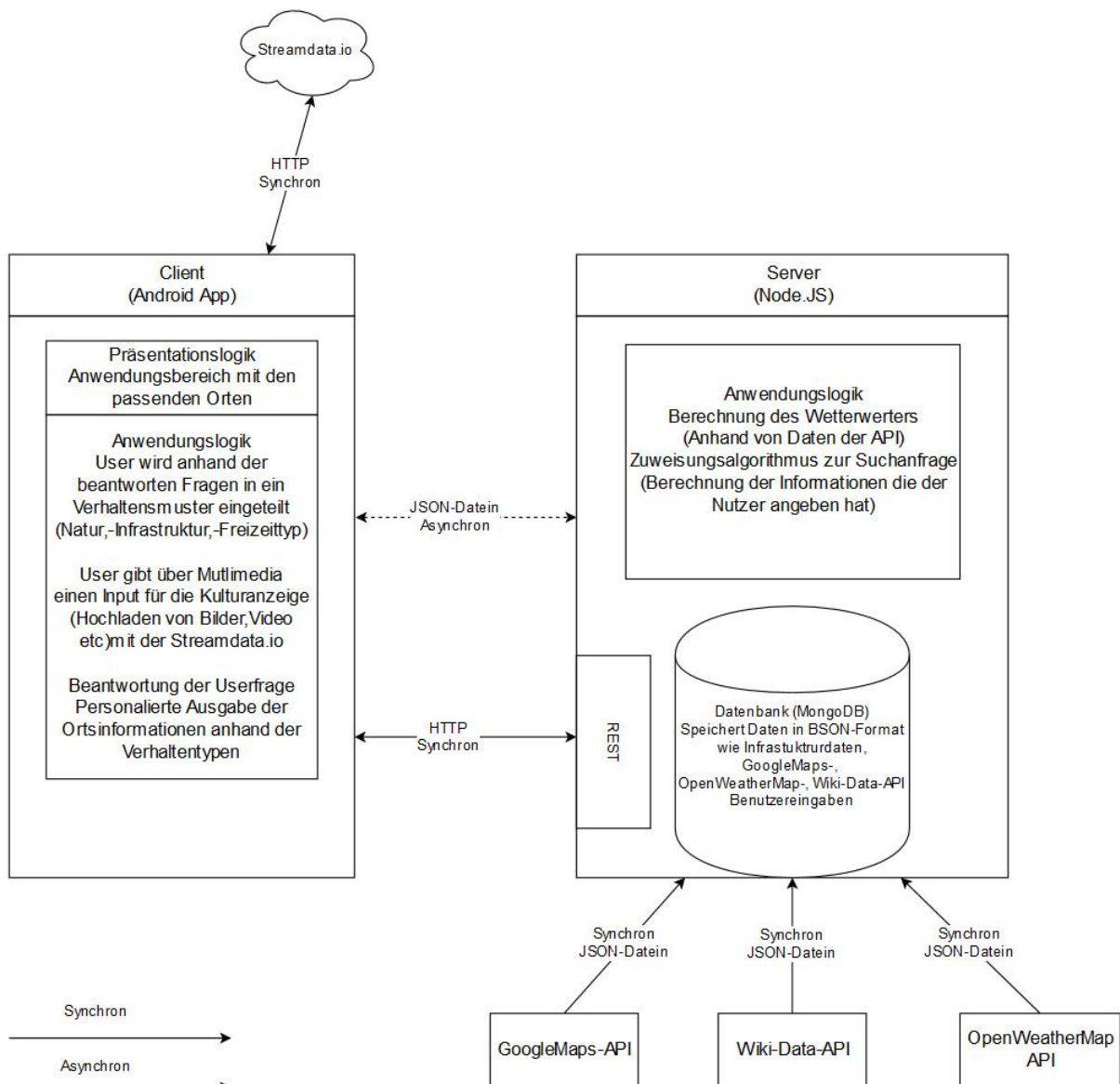


Bei der Galerie wurde die Anordnung und Größe der Element verändert. So steht das Video, das für den Ort erhoben wird im Vordergrund.

Abb. 11: MockUp Galerie

III Modellierung der webbasierten Anwendung

1. Systemarchitektur



1.1 Der Server

Es wurde entschieden den Server über Node.JS laufen zu lassen weil, man damit eine unkomplizierte und wandelbare Programmiersprache besitzt, die einfach zu verstehen ist. Im Server werden die Daten von den API's Synchron empfangen und verarbeitet. Die OpenWeatherMapApi Daten werden zu der Berechnung des Wetterwertes genutzt. Ebenso berechnet der Server den Zuweisungsalgorithmus. Weiterhin sitzt auch die Rest-Schnittstelle im Server und die Datenbank die alle Daten ineinander bringt. In den ersten Schritten der Implementierung wird der Server lokal genutzt und später auf Heroku gehostet.

1.2 Der Client

Über den Client kommuniziert der Benutzer mit dem System über die Präsentationslogik. Hierbei stellt der Client bestimmte Fragen die der Nutzer beantworten. Außerdem werden die Daten von der Wohnungssuche über den Client gestellt und dort auch beantwortet beziehungsweise ausgegeben. Der Nutzer benötigt keine Interaktion mit dem Server. Im Grunde kann man zusammenfassend sagen, dass auf dem Client alle wichtigen Informationen eingegeben werden und zum Server weitergeleitet und verarbeitet werden. Danach gibt der Client die verarbeiteten Daten dem Nutzer aus.

1.3 Verwendete Protokolle

Die Protokolle zur Kommunikation zwischen den Systemkomponenten basiert auf HTTP-Protokollen für die REST Spezifikationen.

1.4 Verwendete Cloud

Die Cloud wird dafür genutzt um die Multimedia-Daten in einer Streamcloud abzuspeichern und wiederzugeben. Die einzige Änderung besteht darin, dass die HTTP-Anforderung jetzt lange läuft, und dass jedes Update als JSON-Patch ausgeführt wird, der die erste mit jeder Änderung erhaltene JSON-Antwort verstärkt. Außerdem müssen Sie die Anzeige von Daten, Inhalt und Änderungen mit jeder inkrementellen Aktualisierung, die über Ihre neuen Datenströme eingeht, überarbeiten.

Die Clients werden zwar weiterhin HTTP als Transportmittel verwendet, aber das Streaming zwingt uns dazu, über die Veröffentlichung von Daten und Inhalten in der Clientbenutzeroberfläche anders zu denken.

1.5 asynchrone Kommunikation

Für die Verteilung von Nachrichten wurde ein Asynchrones Kommunikationsparadigma verwendet, um auf Veränderungen eine schnelle Reaktionszeit zu ermöglichen. Das System

soll auf plötzlichen Schwankungen der Wetterdaten oder Straßendaten reagieren können, damit der Server immer richtige und passende Daten benutzt um den Benutzer immer genaue Informationen bereit zu stellen.

1.6 synchrone Kommunikation

Eindeutig identifizierte Ressourcen sind von Vorteil für einen simplen Abruf zum Beispiel von Einwohnerzahl, die in einem bestimmten Zeitraum erstellt wurden. Das kann mit dem HTTP-Protokoll GET bewerkstelligt werden.

1.7 Datenbank

Für die Datenbank wird MongoDB verwendet. Das ist eine Document basierte Datenbank und es lassen sich beliebig viele Collections erstellen, sodass die Arbeit mit MongoDB einfacher gemacht wird. Bei relationalen Datenbanksystemen wird durch Datenbanksperren und Konsistenzbedingungen oft Geschwindigkeit eingebüßt. MongoDB stellt an sich den Anspruch, den Funktionsumfang von SQL zu bieten, aber gleichzeitig die Geschwindigkeit und Skalierbarkeit von Key-Value-Stores zu erreichen. Dies wird in der Abbildung verdeutlicht.



Neben schnellen Antwortzeiten soll auch die Speicherkapazität des Systems im Online-Betrieb leicht zu erweitern sein. Als letzter Punkt bleibt einfache Benutzbarkeit. Ein weiterer Vorteil ist, dass es einfach und ohne Training zu erlernen ist und es ist überall möglich eine MongoDB-Instanz laufen zu lassen, sei es auf eigenen Servern, in einem CloudService oder auf dem privaten Rechner. Das dokumentorientierte Datenmodell kommt den meisten Programmiersprachen sehr entgegen, da die Dokumente fast direkt dem Datentyp des Objekts entsprechen. Folglich wird in der MongoDB auch die Javascript-Objekt-Notation zur Dokument-Repräsentation verwendet. Darüber hinaus verwendet MongoDB Javascript für serverseitige Skripte und als Programmiersprache für Map-Reduce-Anfragen.

2. Begründung der verwendeten API's und Betrachtung von Alternativen

Die GoogleMapsAPI wird zur Veranschaulichung der gesamten Kartenumgebung genutzt und ist daher ein wichtiger Bestandteil des Systems. Der Grund warum wir uns für Google Maps entschieden haben ist, dass wir diese API schon einmal in einem anderen Projekt eingebunden und uns daher mit der Umgebung auskennen. Weitere Vorteil ist das die API über die meisten Informationen verfügt und immer aktuell ist. Außerdem empfiehlt GoogleMaps die Verwendung von unseren nächsten API und zwar die OpenWeatherAPI.

Über diese bekommen wir die Wetterwerte. Ein Konkurrenzprodukt hierbei ist die OpenStreetMap, jedoch gibt es klare Vorteil von Google Maps. Die Google Maps API for Work ist eine auf die Anforderungen von Unternehmen zugeschnittene Sammlung von APIs (Programmier Schnittstellen), mit denen Daten als Overlay in eine benutzerdefinierte Google-Karte eingefügt werden. Die Kartenplattform von Google lässt durch die attraktive Anwendungen und Apps mit Satellitenbildern, ein großer Unterschied zu OpenStreetMap entstehen. Die letzte API die wir verwenden ist die WIKI-DB-API hierbei nutzen wir die große Bandbreite von Wikipedia und lassen uns von den einzelnen Orten die in Wikipedia eintragen sind eine Basisinformation ausgeben (Einwohnerzahl etc.). Damit stellen wir sich das in jeden Ort eine Grundinformation ausgeben kann und dies wiederrum hilft den Zuweisungsalgorithmus. Außerdem erspart es uns eine riesige Datenerhebung. Bis jetzt gibt es kein Programm, das uns so viele Daten ausgeben kann wie die Wiki-DB-API, weshalb wir auch diese nutzen.

3. Anwendungslogik

3.1 Clientseitige Anwendungslogik

Durch den Fehlschlag des POC's musste eine neue Anwendungslogik am Client überlegt werden. Nach einigen Überlegungen wurde sich für eine personalisierte Anzeige festgelegt. Der Nutzer bewertet und beantwortet die Fragen, die ihm vom Client gestellt wurden, daher erhält der Client Auskunft wie das Verhaltensmuster des Nutzer ist. Daraufhin weist der Client ihn in 3 verschiedene Typen ein, entweder in Natur-,Freizeit,- oder Infrastrukturtyp. Je nachdem werden im dann die für ihn relevanten Inhalte des Ortes als erstes angezeigt.

Client (Aus dem Konzept)

Kulturwert: Der Benutzer gibt Bewertungen über verschiedenen kulturellen und sozialen Kriterien zu einem Ort an. Diese werden mit der jeweiligen Priorisierung verrechnet und es entsteht ein repräsentativer Kulturwert. Dieser wird asynchron an den Server gesendet.

Suchanfrage: Ein Benutzer priorisiert Kriterien, die ihm bei der Wohnortwahl (Kultur, Infrastruktur, Wetter) besonders wichtig sind. Der Client berechnet daraus eine Abfrage, die er synchron an den Server sendet. Als Antwort werden ein Ort und dessen Informationen ausgegeben.

3.2 Serverseitige Anwendungslogik

Nach weiteren Evaluationen wurde sich dazu entschlossen vom Kulturwert, als auch vom Bevölkerungswert zu trennen. Der geographische Wert wird geändert.

3.2.1 Zuweisungsalgorithmus

Hierbei wird aus dem Infrastrukturstrukturwert der sich aus der Erhebung von der Wiki-Data-API und der eigenen erarbeiteten Datenerhebung. Der Wetterdaten die von der OpenWeatherMap-API und der Ergebnissen der Beantwortungen der Client-Fragen die dem Nutzer gestellt wurden eine personalisierte Auswahl der Orte mit dem passenden Werten errechnet und ausgegeben.

3.2.2 Wetterwertberechnung

Die API sendet die Wetterdaten der einzelnen Standorte und diese werden beim Server analysiert und ausgewertet. Diese Daten werden dann in den Zuweisungsalgorithmus gesendet um zu berechnung des richtigen Ortes zu helfen.

Server (Aus dem Konzept)

Wetterwert: mit Informationen aus der Wetter API berechnet der Server mehrere Werte, die die Wetterlage repräsentieren sollen.

Bevölkerungswert: In der Datenbank liegen Informationen zu der Bevölkerungsentwicklung in Deutschland. Es wird ein Wert berechnet, der den Grad der Landflucht verdeutlicht.

geographischer Wert: über die Karten-Schnittstelle werden Daten, die dieser zur Verfügung stell angefragt und Werte werden berechnet und benannt. z.B Verkehrsanbindung, Einkaufsmöglichkeiten, Höhenlage.

Kulturwert: Der Server erhält Kulturwerte von verschiedenen Clients und wird den Mittelwert aus diesen bestimmen.

Zuweisungs-Algorithmus: der Server analysiert die Suchanfrage des Clients und gibt ihm mittels eines Zuweisungs-Algorithmus einen geeigneten Ort aus.

4.Datenstruktur

Die Ressourcen sind in JSON Format, da dies gut Gestaltet ist, unkomplizierte serverseitige Verarbeitung ermöglicht und JSON kann direkt ausgeführt werden und in BSON-Datei umgewandelt werden. Was die Speicherkapazität und die Rechenlaufzeit verringert.Das wichtigste Punkt ist das JSON auch von nicht Programmierer verstanden werden kann, das hilft uns dann für die bessere Evaluation, wenn wir mit den Usern über unser System reden, die es dann nutzen werden.

4.1. Wohnungsort

Der User starte die App und besitzt die Funktion bestimmte Orte zu suchen. Mit der Eingabe von ausgewählten Informationen bekommt man durch einen programmierten Zuweisungsalgorithmus die passenden Orte ausgegeben. Dies wird anhand der Präsentationslogik dem Wohnungssuchenden angezeigt.

4.2. Kulturanzeige

wird mit Hilfe von Multimedia erworben, durch das einbringen von Bilder, Video's und Audio wird für jeden Ort ein bestimmtes Profil angelegt. Es wird eine Schnittstelle mit dem System verbunden wie ein Media Player der diese Daten abspielen und verbinden kann. Durch ein Upvotesystem werden nur bestimmte Daten(Bilder, Sound und Video) in das sogenannte Profilvideo(Imagevideo) des einzelnen Ortes gebracht. Dadurch wird eine objektive Repräsentation geboten.

4.3. WetterAPI

ist eine API, mit dieser kann man Wetterinformationen auf der ganzen Welt aufrufen. Das besondere an diese API ist das die URI mit unterschiedlichen Parameter ergänzt werden kann. Es können Informationen für verschiedene Orte und Städte mit der JSON – Datenstruktur aufrufen über eine ID, Name der Stadt oder durch Längen und Breitengrad um die Position genauer zu bestimmen. Fürs Projekt Relevant sind hierbei die Informationen: welche Stadt und Wetter (Regen, Schnee, Sonne...) --und die Stärke des jeweiligen Wetters z.B. Regenstärke Nieselt es oder ist es ein starker Regen--

4.4. GoogleMapsAPI

hat man die Möglichkeit auf den Standort der User zuzugreifen um herauszufinden wo diese sich zurzeit befinden um ihnen dann bestimmte Informationen zuzuschicken oder um Anfragen zu senden. Es wird in Android implementiert.

4.5. Infrastrukturwerte

werden durch die Administratoren eingebunden. Die Daten wie Mietpreisen, Einkaufsmöglichkeiten etc. werden in einer BSON-Datei in der Datenbank gespeichert und bei der Berechnung des Zuweisungsalgorithmus hinzugezogen.

4.6. Zuweisungsalgorithmus

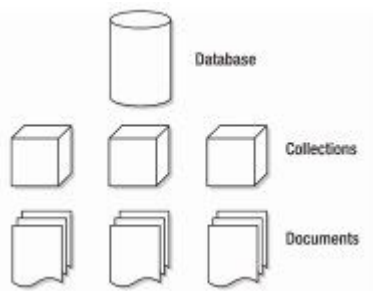
Der Server wertet die Eingabe vom Benutzer aus und zieht Daten von der Infrastruktur und den Wetter-Daten die in der Datenbank liegen hinzu. Die Werte werden dann analysiert und zusammengerechnet und der Server gibt die passenden Orte für die Benutzeranfrage aus.

4.7. WikidataDB

Wikipedia bietet eine API an wo man bestimmte Informationen aus Artikel entnehmen kann. Interessant hierbei ist für unsere System das wir bestimmte Grundinformationen in unsere System speisen können und damit für jeden Ort eine Grundbasis besitzen mit dem wir den Start der App erheblich erleichtern. Man kann auf die Daten über ein einzelnes Datenobjekt oder durch Dumps auf alle Daten gesammelt zugreifen. Wikidata bietet seine Daten frei und ohne Verpflichtung zur Nennung des Urhebers unter CC-0 an. Trotzdem würden wir es sehr begrüßen, wenn Wikidata als Quelle der Daten genannt wird. Dies stellt sicher, dass das Projekt langfristig erhalten bleibt und aktuelle und hochwertige Daten zur Verfügung stellen kann.

5. Datenhaltung in der DB

Die Dokumenten-Datenbank ist ein zentralen Bestandteil des Systemes, weil dort alle Informationen gespeichert sind. In unserer MongoDB werden Dokumente als BSON-Objekte dargestellt, also mithilfe einer speichereffizienten, binären Variante von JSON. Eine sehr übersichtliche Grammatik der JSON-Beschreibungssprache. Eine Collection ist eine benannte Menge von Dokumenten. Jede Collection ist in einer Datenbank gespeichert und jedes MongoDB-System enthält eine Menge von Datenbanken. Dadurch ergibt sich eine gewisse Ähnlichkeit zu der Strukturierung in relationalen Datenbankensystemen. Auch in letzteren sind die Daten in Datenbanken gespeichert, die wiederum Tabellen enthalten. Diese Tabellen entsprechen in etwa den Collections in MongoDB. Innerhalb einer Tabelle sind die Datensätze als Zeilen gespeichert, wobei jede Zeile ein Tupel aus einem oder mehreren Werten ist.



5.1 Vorteile und Nachteile der dokument basierten Datenhaltung

In einem RDBMS (Relational Database Management System) unterliegt aber die Tabelle einem Schema, sodass alle Zeilen der Tabelle den gleichen Aufbau haben, d.h. die gleiche Anzahl von Werten mit jeweils den gleichen Datentypen. In MongoDB unterliegen die Dokumente keinem festen Schema, sondern jedes Dokument hat seine eigene Strukturierung. Die Dokumente eine Collection können Ähnlichkeiten aufweisen, müssen es aber nicht. Ein weitere Vorteil der Dokumentbasierten Datenhaltung ist die Schemalosigkeit ist aber größere Flexibilität möglich, wenn sich Datensätze oder Typen von Datensätzen nur in kleinen Details auf der Schema-Ebene unterscheiden. In RDBMS sind oft größere Verrenkungen nötig, z.B. für Vererbungshierarchien. In der Dokumenten-Datenbank können die Daten dagegen einfach flach gespeichert werden. Je nachdem, welche Eigenschaften ein bestimmter Datensatz abhängig von seiner Klasse oder seinem Typ hat, enthält das Dokument unterschiedliche Felder. Ein Problem der Dokumenten-Datenbank ist referenzielle Integrität. So etwas wie Fremdschlüsselbeziehungen gibt es nicht. MongoDB verwendet stattdessen das Konzept von Einbettung und Verlinkung. Ein Link ist ein Verweis auf ein anderes Dokument. Er muss auf der Ebene der Anwendungsprogramme interpretiert und das referenzierte Dokument in einer zusätzlichen Abfrage eingelesen werden. Bei der Einbettung werden die referenzierten Datensätze als Unterdokument in das referenzierende Dokument eingebettet.

6. Programmiersprachen

Die Auswahl der Programmiersprache ist ein wichtiger Schritt bei der Konzeption eines interaktiven Systems. Hier gibt es mehrere Kriterien, die für die Auswahl ausschlaggebend sind, zu beachten. Zunächst sollte geprüft werden, welche Programmiersprache bereits bekannt ist und ob die geforderten Systemkomponenten mit dieser umsetzbar sind. Hier

ergeben sich die Vorteile durch Kompaktheit des Codes und bereits vorhandener Frameworks oder Libraries, die für die Umsetzung genutzt werden können. Die Erfahrung der Entwickler spielt bei der Wahl der Programmiersprache auch eine große Rolle. Die Einarbeitung in eine unbekannte Sprache kann aufgrund des Syntax und der Kompaktheit viel Zeit in Anspruch nehmen. JSON steht für JavaScript Object Notation. Es ist strukturiert, leicht, lesbar und leicht zu analysieren. Dies ist die beste Alternative zu XML, wenn unsere Android-App Daten vom Server austauschen muss. XML- Parsing ist im Vergleich zu JSON- Parsing sehr komplex was ein weitere Vorteil der Verwendung ist.

6.1 Clientseitige Programmiersprache

Der für den Client mobile Anwendung in Form einer Android App implementiert wird, ist die Wahl der Programmiersprache stark eingegrenzt. Hier ist die Unterstützung von JSON mit am größten. In den Modulen Webbasierte Anwendung 1 und 2 wurde der Umgang mit der objektorientierten Entwicklung mit JSON gelernt und in Projekten vertieft. Auch in den weiteren Modulen wurde mit JSON und deren und dazu importierten Libraries und Frameworks gearbeitet, weshalb die 2 Sprachen am besten geeignet sind um dieses Projekte zu realisieren.

6.2 Serverseitige Programmiersprache

Serverseitig wird die Programmiersprache JavaScript mit dem Framework NodeJS genutzt. Die Wahl ist auf diese Kombination gefallen, da das Framework NodeJS eine besonders einfachen und kompakten Aufbau für Implementierung eines Servers bietet. NodeJS wurde vor allem auf Performance und erweiterbare Module ausgelegt. Die Basis ist schlank und kann mit den benötigten Modulen so erweitert werden, dass der Umfang des Codes klein gehalten werden kann. Zudem wurde auch hier im Modul Web-basierte Anwendungen 2 bereits ein Server mit einer API implementiert. Außerdem wurde bei diesem Projekt bereits mit einer Anbindung wie OpenWeatherMap als API gearbeitet, sodass auch hier Erfahrung mit dem Syntax und möglichen Problemstellungen in NodeJS gesammelt wurde.

7. Entwicklungsumgebung

Aufbauend auf der Wahl der Programmiersprache wird die Entwicklungsumgebung ausgewählt. Auch hier gibt es wie bei der Programmiersprache mehrere Kriterien, die zu beachten sind. Auf der einen Seite stehen die Kriterien der Umgebung selbst. Ein Kriterium kann die Funktionalität mit der gewählten Programmiersprache genannt werden. Die IDE

sollte zum Beispiel Code-Autovervollständigung und ein gutes Syntax-Highlighting für die Programmiersprache besitzen. Mit diesen Hilfsmitteln wird es für den Entwickler leichter den Code zu schreiben und zu verstehen. Dadurch wird auch die andere Seite der Kriterien klar. Hier stehen die Erfahrungen und Vorlieben der Entwickler mit der IDE selbst.

7.1 Clientseitige Umgebung

Unser Client besteht aus einer Android App, die mit Node.JS entwickelt wird und für das Layout der Interaktionselemente //verwendet. Google bietet für ihre Android App eine offizielle IDE Namens Android Studio an. Betreffend der im oberen Abschnitt genannten Kriterien bietet es eine sehr gute Code-Autovervollständigung und ein gutes Syntax-Highlighting an. Da wir außerdem für die Datenhaltung Mongo DB nutzen, dass gut mit Google auskommt, ist auch hier die Integration gut umgesetzt. Ein weiterer Vorteil von Android Studio ist die Möglichkeit, die Layouts des User Interfaces per Drag-and-Drop zusammenstellen zu können, sodass eine erste rudimentäre Code-Struktur für das Layout ohne viel Aufwand aufbaubar ist. Des weiteren sprechen die Kriterien der Entwickler für den Einsatz dieser IDE.

7.2 Serverseitig Umgebung

Da Serverseitig zur Entwicklung NodeJS genutzt wird, wird eine IDE benötigt, die Javascript-fähig ist. Zudem gibt es für Node.JS Plugins, die genutzt werden können, um den kompletten Entwicklungsprozess über diese IDE handhaben zu können. Auch hier sprechen die Kriterien der Entwickler für Node.JS. Im Modul Web-basierte Anwendungen 2 wurde im Rahmes eines Projektes zum Aufsetzen eines Server Node.JS gearbeitet. Die Vorgänge zum Kompilieren und Ausführen des Servers sind also bereits bekannt.

7.3 Zusammenspiel der beiden Umgebungen

Bei Android Studio spielen die beiden Umgebungen sehr gut zusammen und weisen ein sehr ähnliches Interface auf. Eine Umstellung bei der Entwicklung von Server und Client ist also nicht nötig. Auch die Integration von Git ist in beiden standardmäßig vorhanden und verhält sich identisch, sodass es zu keinen Konflikten bei der Versionsverwaltung kommt.

8. Vielfalt der Endgeräte

Da PlaceToBe ein System ist, das rein über eine Android App benutzt wird, ist der Umfang der Gerätevarianten begrenzt. Es gibt weder eine Webanwendung, die über den Browser benutzt wird, noch gibt es einen Desktop Client für Windows oder Apple Rechner.

8.1 Begründung für die Benutzung von Android

Bei den mobilen Geräten ist Android das am meisten genutzte Betriebssystem. Der Markt für Android Apps ist somit der größte und bietet daher die besten Voraussetzungen für die Vermarktung der App. Des Weiteren ist die Entwicklung von Android Apps im Gegensatz zu iOS Apps nicht an das Betriebssystem gebunden, dass zur Entwicklung eingesetzt wird. Android Studio gibt es sowohl für Windows als auch für Apple Rechner. Das für iOS Apps konzipierte Xcode gibt es jedoch nur für das Apple Betriebssystem MacOS.

9. Reflektion des POCs

Unser POC hat sich mit der Berechnung des Kulturwertes befasst und diesen in eine Zahlenwert definiert. Doch dabei ergab sich ein großes Problem denn, die Kultur wird von jedem Menschen unterschiedlich bewertet. Daher ergibt es kein Sinn den Wert mit einer Zahl zu bestimmen.

Um es verständlicher zu sagen der POC ist damit fehlgeschlagen.

Weshalb eine komplette Umstrukturierung der Kulturwertes nötig war.

Der neue Kulturwert wird über Multimedia angezeigt und erhoben die Nutzer können sich Bilder, Audio und Video-Dateien von einzelnen Orten ansehen und sogar selber interaktiv auch hochladen und bewerten. Das System merkt sich den Feedback des Nutzers und aus den besten hochgeladenen Dateien wird für den Ort ein personalisiertes Profilvideo erstellt um die Kultur des Ortes noch besser dazu stellen. Somit kann jeder Nutzer sich ein eigenes Bild der Kultur machen und das Risiko der Erhebung des Kulturwertes würde drastisch verringert werden.

10.Fazit

Am Ende können wir sagen das wir zufrieden mit unseren Modellierung sind. Jedoch haben wir auch bemerkt dass, wir ein sehr großes System haben. Es werden ganze 3 API genutzt und 2 Programmiersprachen und einer Cloud. Wir haben eine große Datenerhebung vor uns die arbeit mit MongoDB wir sicherlich nicht einfach werden. Aber wir halten an unserem System fest, weil wir versuchen wollen ein echter reales Problem der heutigen Zeit zu lösen oder zumindest geben die Landflucht etwas zu unternehmen. Jetzt stellen sich noch die Fragen ob wir es schaffen auch das System umzusetzen ? Ob es klappt alle Daten mit der Datenbank zu verbinden und die API alle funktionsbereit mit dem System zu implementieren. Die Dateien in BSON umzuwandeln und den Zuweisungsalgorithmus richtig mit allen Daten zu speisen.

Literaturverzeichnis

Ergonomics of human-system interaction - Part 210: Human-centred design for interactive systems (ISO 9241-210:2010)

<https://vollzeitblogger.de/mongoose-js-mongodb-odm-fuer-node-js/>

<http://bsonspec.org/>

<https://streamdata.io/>

<http://www.mongodb.org/display/DOCS/Philosophy>

<http://www.mongodb.org/display/DOCS/Tutorial>

<http://www.mongodb.org/display/DOCS/Overview+-+The+MongoDB+Interactive+Shell>

<http://www.mongodb.org/display/DOCS/Drivers>

<http://www.mongodb.org/display/DOCS/Data+Types+and+Conventions>

<http://json.org/>

Anhang

Die Umfrage "Ausziehen-Umziehen-Rumziehen" finden Sie unter folgendem Link:

<https://www.umfrageonline.com/s/ab8cb1f>