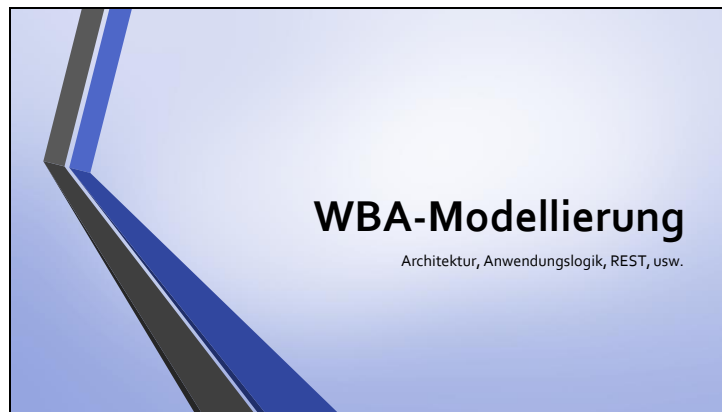
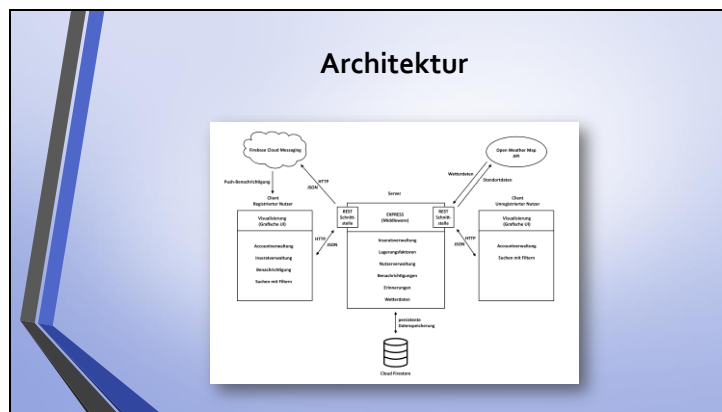




Diese Präsentation wird in die beiden Hauptbereiche WBA und MCI unterteilt. Für beide Schwerpunkte wurden verschiedene Artefakte erarbeitet und sollen hiermit präsentiert werden. Für eine detaillierte Projektbegründung möchten wir auf unser Github verweisen, wo die Artefakte alle nochmal individuell als PDF vorliegen.



Um den Bereich der WBA-Modellierung umfassend zu bearbeiten, wurde zunächst die angestrebte Architektur strukturiert und modelliert. Daraufhin die Anwendungslogik beschrieben und Risiken für das angestrebte System analysiert. Weiter wurden verschiedene APIs betrachtet und die für unser Projekt am sinnvollsten nutzbare ausgewählt. Um die spätere Implementierung zu erleichtern, wurden bereits Ressourcen in REST erarbeitet und eine Datenstrukturierung modelliert. Zuletzt wurden die wichtigsten Punkte der Implementierung mit Proof of Concepts überprüft und als Rapid Prototype umgesetzt, um sicherzustellen, dass das Projekt umsetzbar ist. Diese wurden zunächst spezifiziert und anschließend dokumentiert.



Als erstes wird analysiert welche Kommunikationsmethode für das System verwendet werden sollte. Anschließend wird ermittelt welche Komponenten benötigt werden und das Ganze in einem Diagramm zur besseren Veranschaulichung dargestellt.

Für die Kommunikationsmethode wurde zwischen Peer To Peer und Server / Client entschieden. Wir haben das Modell des Server / Client gewählt. Es überwiegen die positiven Aspekte zum Beispiel die flexible Erweiterungsmöglichkeit, wie auch die Möglichkeit zur Einhaltung moderner Sicherheitsstandards.

Unser angestrebtes System besteht dementsprechend aus folgenden Komponenten:

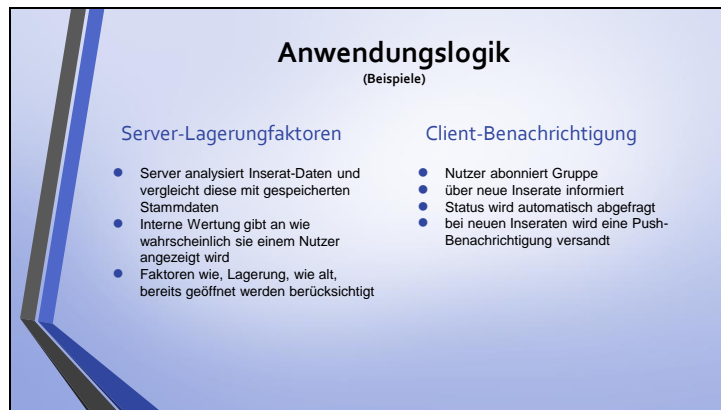
Server - Der Server wird für die Speicherung, mittels Datenbank, und Anreicherung von Daten genutzt. Der Server soll mit NodeJS arbeiten.

Clients - Werden in diesem Projekt durch eine mobile Anwendung realisiert, da hier der Vorteil für das Projekt gesehen wird, durch die leichte Einbindung der Standortdaten und Kamerafunktion.

Middleware - Durch die Verwendung von NodeJS beim Server, wurde das Framework "express" angestrebt, um einen Austausch über REST zu ermöglichen.

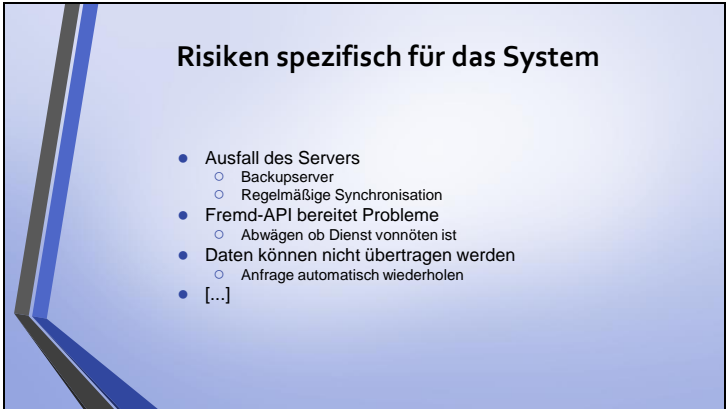
Weitere Dienste - Um den Nutzern mehr Komfort zu bieten, werden zusätzlich Wetterdaten empfangen und die Möglichkeit Gruppen zu abonnieren über Cloud Messaging.

Für die Realisierung der Datenbank wird NoSQL verwendet, die Struktur der angedachten Datenbank unterstützt dieses, zudem ist NoSQL flexibel und schnell in der Verarbeitung.



Die wichtigste Anwendungslogik wird auf dem Server ausgeführt. So werden erstellte Inserate beim Speichern in die Datenbank analysiert und mit Stammdaten abgeglichen, um so eine interne Produktbewertung zu erreichen. Anhand dieser Bewertung können Warnhinweise zur Haltbarkeit gemacht werden, aber auch dem Nutzer die besten Produkte zuerst angezeigt werden. Ebenfalls wird berücksichtigt wie nah die Anzeige zum Nutzer ist. Herrschen extreme Wetterbedingung bei dem Standort der Anzeige, wird der Nutzer darauf hingewiesen und nähere Inserate bevorzugt. Inserate werden außerdem regelmäßig überprüft, ob diese noch haltbar sind, wenn die Frist näher rückt, werden dem Nutzer automatisch Warnungen geschickt, und das Inserat nach Ablauf der Frist automatisch gelöscht.

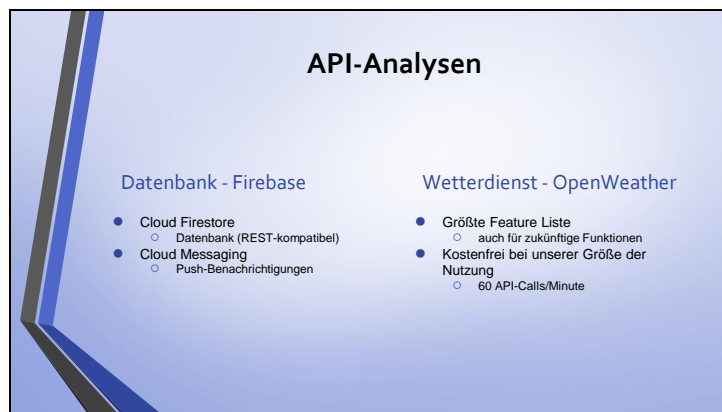
Weitere Funktion sind unter anderem die Nutzer- und Accountverwaltung, die Möglichkeit bestimmten Gruppen beizutreten und über neue Inserate innerhalb dieser Gruppe benachrichtigt zu werden und die Möglichkeit gezielt zu Suchen mit Filterungseinstellungen



Risiken spezifisch für das System

- Ausfall des Servers
 - Backupserver
 - Regelmäßige Synchronisation
- Fremd-API bereitet Probleme
 - Abwägen ob Dienst vonnöten ist
- Daten können nicht übertragen werden
 - Anfrage automatisch wiederholen
- [...]

Bei den spezifischen Risiken wird sich noch einmal verstärkt auf Risiken bezogen, die bei der Entwicklung des Systems auftreten können. Sie sind weniger allgemein und eher technischer Natur.



Um herauszufinden welcher Dienst am besten zu uns passt und uns am meisten Nutzen bringt wurden verschiedene Dienstleister in der jeweiligen Kategorie verglichen.

Die Datenbank stellt ein großes Stück unseres Projektes dar und muss daher gut überdacht sein. Die Datenbanken, die wir uns angeschaut haben sind Firebase von Google und MongoDB. Beide sind großartige Cloud-Datenbank Dienstleister und bieten eine Vielzahl an Funktionen an.

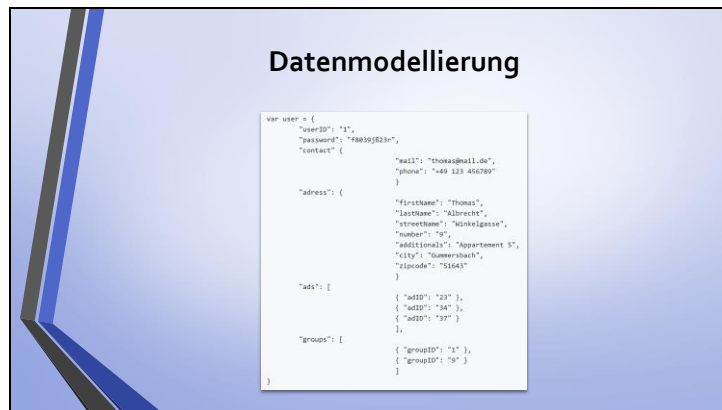
Entschieden haben wir uns dann für Firebase aufgrund der hohen Anzahl an Möglichkeiten, die es uns bietet, nicht nur in Richtung Datenbank, sondern auch Cloud Messaging.

Um herauszufinden welcher Wetterdienst zu uns am besten passt haben wir sie auf Funktionen, die wir nutzen können und Kosten verglichen. Schlussendlich haben wir uns jetzt für OpenWeatherMap entschieden, da dieser alles bietet was wir an Funktionen benötigen, noch einige zusätzliche Features, die man in Zukunft einbauen kann und dazu noch bei unserem Kontingent völlig kostenfrei zur Verfügung gestellt wird.

REST-Ressourcen

| Nutzerverwaltung | | | | |
|------------------|-----------|---|------|-------------------------|
| Ressource | HTTP-Verb | Semantik | Code | Fehlercode |
| /user | GET | Listet alle Nutzer auf | 200 | 404, 500, 503 |
| /user/login | POST | Überprüft ob der Nutzer registriert ist und ruft seine Daten ab | 200 | 400, 500, 503 |
| /user/userID | GET | Gibt Daten eines bestimmten Nutzers aus | 201 | 400, 500, 503 |
| /user/userID | DELETE | Löscht einen Nutzer vollständig | 204 | 401, 403, 404, 500, 503 |
| /user/register | POST | Erstellt neues Nutzerkonto | 200 | 400, 500, 503 |
| /user/edit | PUT | Lässt Nutzer private Daten bearbeiten | 204 | 400, 500, 503 |

In den Tabellen werden alle bereits beschriebenen Funktionen des Systems gelistet. Auf die Ressourcen werden lediglich von dem Server und den Entwicklern zugegriffen, reguläre Nutzer haben keine Berechtigung hierzu. Es wird auf Funktionen verzichtet die automatisiert ablaufen, die weder Server noch Entwickler ansprechen können. Diese Tabelle ist ebenfalls iterativ und wird im Verlauf der Implementierung bei Bedarf angepasst und erweitert.

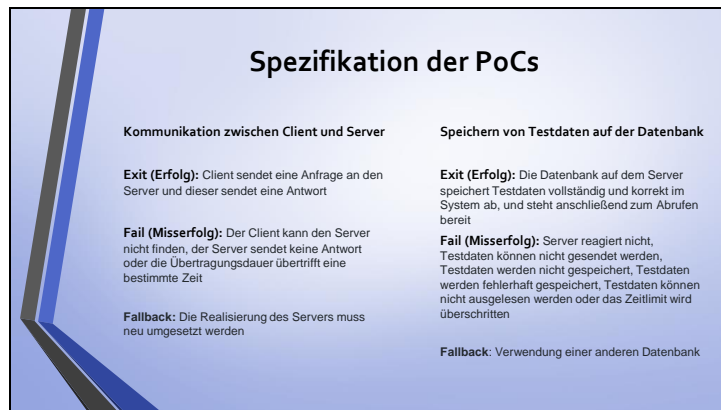


Wir haben uns bereits Gedanken gemacht, wie die Daten für das System in der Datenbank am Besten zu speichern sind. Dabei haben sich 3 Hauptpunkte herausgestellt. Nutzer, Inserate und Gruppen.

Für den Nutzer werden alle relevanten Informationen zu seinem individuellen Profil gespeichert. Um ihn identifizieren zu können, bekommt jeder Nutzer eine eindeutige ID zugewiesen. Da für jeden registrierten Nutzer die Möglichkeit besteht mehrere Inserate zu erstellen und Gruppen beizutreten, werden die jeweiligen IDs dieser Strukturen für jeden Nutzer individuell gespeichert. Diese Datenstruktur stellt den wichtigsten Datensatz dar, da sie auch für die anderen Strukturen gebraucht wird.

Wie beim Nutzer werden auch für Inserate eindeutige IDs für jedes einzelne Inserat erstellt. Die Datenstruktur enthält ebenfalls die NutzerID um diesen zuordnen zu können. Um den korrekten Ablauf der Anwendungslogik zu realisieren, werden die relevanten Daten individuell abgespeichert. Die Längen und Breitengrade werden für die Berechnung der Entfernung und Ermittlung der zugehörigen Stadt benötigt. Wenn Inserate nicht öffentlich sichtbar sein sollen, können diese privat eingestellt und der entsprechenden Gruppe zugeordnet werden.

Es soll jedem Nutzer die Möglichkeit geboten werden eine Gruppe zu erstellen, um seine Inserate nur einem bestimmten Personenkreis zur Verfügung zu stellen. Diese Gruppen haben eine eindeutige ID und einen Namen. Dieser Gruppe können User beitreten und Inserate nur für diese Gruppen verfügbar machen. Hierzu werden alle relevanten IDs von Nutzern und Inseraten gespeichert.



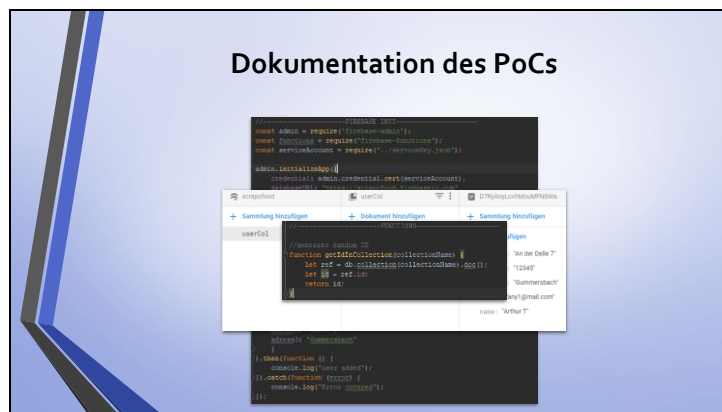
Für die Implementierung und Umsetzung der Architektur müssen kritische Punkte zunächst überprüft werden. Dies betrifft vor allem die Verwendung von externen Diensten, aber auch die Umsetzung der Datenbank. Für die Spezifikation werden jeweils Exit und Fail-Kriterien angegeben, zudem jeweils ein Fallback, sollte die Umsetzung scheitern.

Für die Proof of Concepts wurden folgende Fälle berücksichtigt:

- Kommunikation zwischen Client und Server
- Speichern von Testdaten auf der Firestore Datenbank
- Wetterdaten über externen Dienst abfragen und speichern

Angedacht vor der Implementierung wäre noch eine Umsetzung von:

- Standortdaten vom Client mit Städten abgleichen
- Push-Benachrichtigung



Im Beispiel wird hier das Speichern von Testdaten auf der Firestore Datenbank vorgeführt:
Firestore initialisieren

In dem Modul 'firebase-admin' befinden sich alle notwendigen Funktionalitäten, um sich mit der Datenbank zu verbinden und auch Daten zu speichern oder abzurufen. Danach wird die Verbindung initialisiert und mit einem ServiceKey eine sichere Verbindung sichergestellt. Wenn die Verbindung erfolgreich hergestellt wurde kann man auf die 'Sammlungen' und 'Dokumente' zugreifen und im JSON Format die Daten unter den 'Dokumenten' ablegen. Daten hinzufügen

Die User werden unter der Sammlung 'userCol' mit allen notwendigen Informationen abgelegt. Unter der Sammlung eines jeden Nutzer wird zukünftig ein Verweis der Gruppen, denen ein einzelner Nutzer zugehört sowie die verschiedenen Inserate die dieser erstellt oder abonniert hat abgelegt.

Generieren der UserIDs

Firebase bietet eine Möglichkeit userIDs automatisch generieren zu lassen sodass es keine ID ein zweites Mal geben wird und somit keine Fehler entstehen können.



Um den Bereich der MCI-Modellierung umfassend zu bearbeiten, wurden zunächst anhand der vorherigen Recherche und Analyse User Profils erstellt, aus diesen wurden Persona gewonnen und für diese Persona Szenarien erstellt. Um das Verhalten ebenfalls modellieren zu können wurden dann Concrete Use Cases angefertigt. Basierend auf diesen konnte ein Feature-Set entworfen werden. Ebenfalls wurden die Anforderungen noch einmal überarbeitet und erweitert.

Mit dieser Vorarbeit konnte sich dann um die Umsetzung der Prototypen gekümmert werden. So wurde aus einem einfachen paper based prototype eine low fidelity digitale Version. Mit diesen beiden Versionen konnten bereits erste Evaluation vorgenommen werden. Mit dem daraus resultierenden Feedback wurde dann eine high fidelity digitale Version gefertigt, die ebenfalls nochmal evaluiert wurde.


| Benutzermodellierung - User Profils | |
|-------------------------------------|---|
| Nutzer 2 - Suchend | |
| Art | Beschreibung |
| Motivation | Möchte Verschwendung reduzieren |
| Menge der Lebensmittel | Sucht gezielt einzelne Produkte |
| Lebensmittel Lagerung | Kümmert sich um anständige Lagerungsbedingung |
| Domänenspezifisches Fachwissen | Hat sich informiert |
| Bereitschaft zu suchen | Sobald Bedarf besteht |
| Erreichbarkeit | Rund um die Uhr |
| Nutzer 4 - Anbietend | |
| Art | Beschreibung |
| Motivation | Möchte mit Freunden und Bekannten teilen |
| Menge der Lebensmittel | unregelmäßig ein wenig |
| Lebensmittel Lagerung | meist frisch zubereitet und nicht gelagert |
| Domänenspezifisches Fachwissen | Kümmert sich nicht unbedingt um Lagerung |
| Bereitschaft anzubieten | Im privaten Kreis gern, öffentlich ungern |
| Erreichbarkeit | Zum Feierabend |

Bei den User Profiles handelt es sich um eine abstrakte Beschreibung für Nutzer des Systems. Die Informationen sind kompakt formuliert und sind an das System angepasst. Sie wurden durch die Recherchen aus dem Konzept ermittelt.

Es wurden möglichst alle für unser System relevanten Nutzer berücksichtigt und versucht als User Profil darzustellen.

Benutzermodellierung - Persona


Persona 4: Heidrun Muck



| Alt | Beschreibung |
|--------------------------------------|-------------------------------------|
| Alter | 38 |
| Geschlecht | weiblich |
| Berufsaufstellung | Techniker |
| Lebensstand | verheiratet, 2 Kinder, 1 Elternteil |
| Wohnort | Stadt mit 100.000 Einwohnern |
| Interesse | Auto und Technik |
| Ziele | Wissen aufbauen |
| Technische Skills | Sehr gut |
| Allgemeiner Bildungsgrad | Sehr gut |
| Technische Fähigkeiten/Problemlösung | Sehr gut |
| Interesse an neuen Technologien | Sehr gut |

Beschreibung
Heidrun Muck ist eine Frau und arbeitet als Technikerin. Sie ist sehr technisch versiert und hat eine hohe Motivation, ihr Wissen zu erweitern. Sie ist sehr engagiert und arbeitet sehr hart. Sie ist sehr kreativ und hat eine hohe Motivation, ihr Wissen zu erweitern. Sie ist sehr engagiert und arbeitet sehr hart. Sie ist sehr kreativ und hat eine hohe Motivation, ihr Wissen zu erweitern.

Persona 5: Detlef Berger



| Alt | Beschreibung |
|--------------------------------------|-------------------------------------|
| Alter | 45 |
| Geschlecht | weiblich |
| Berufsaufstellung | Techniker |
| Lebensstand | verheiratet, 2 Kinder, 1 Elternteil |
| Wohnort | Stadt mit 100.000 Einwohnern |
| Interesse | Auto und Technik |
| Ziele | Wissen aufbauen |
| Technische Skills | Sehr gut |
| Allgemeiner Bildungsgrad | Sehr gut |
| Technische Fähigkeiten/Problemlösung | Sehr gut |
| Interesse an neuen Technologien | Sehr gut |

Beschreibung
Detlef Berger ist ein Mann und arbeitet als Techniker. Er ist sehr technisch versiert und hat eine hohe Motivation, sein Wissen zu erweitern. Er ist sehr engagiert und arbeitet sehr hart. Er ist sehr kreativ und hat eine hohe Motivation, sein Wissen zu erweitern. Er ist sehr engagiert und arbeitet sehr hart. Er ist sehr kreativ und hat eine hohe Motivation, sein Wissen zu erweitern.

Die Persona wurden aus den User Profiles gewonnen. Sie beschreiben nicht nur eine kompakte, abstrakte Form von Nutzern sondern enthalten persönliche Informationen und beschreiben die relevantesten Informationen zu der Person. Um das Ziel des User-Centered-Design zu erreichen, werden Personen beschrieben deren Ziele und Motive nachvollzogen werden können und damit die Entwicklung des Systems begleiten sollen. Die beschriebenen Persona basieren auf echten Menschen, die aus Datenschutzgründen jedoch leicht angepasst wurden. Anstelle von Stockfotos, oder tatsächlichen Bildern der realen Personen, wurde deren Beine und Schuhe fotografiert. Dies erlaubt es immer noch sich ein Bild von der Person zu machen, ohne zu viel von ihren tatsächlichen Daten preiszugeben.

Benutzermodellierung - Szenarien

Szenario 7: Das ist mir zu kompliziert!

Detlefs Frau berichtet ihm von einer Reportage die sie letzts im Fernsehen gesehen hat, in der es um eine brandneue Anwendung ging, die schon viele Nutzen würden, um ihre Sachen zum Verkauf anzubieten. Detlef der selbst gerne Online-Auktionsportale nutzt ist interessiert. Er lädt die Anwendung auf seinem Smartphone herunter. "Und jetzt? Wie nutz ich die jetzt?", fragt er seine Frau. "Keine Ahnung, du bist doch der Techniker unter uns. Ansonsten warte bis die Kinder vorbei kommen, die können dir das bestimmt erklären!". Detlef verdreht die Augen. Der letzte Besuch der Kinder ist schon eine ganze Weile her.

Zu allen beschriebenen Persona wurden Szenarien verfasst. Diese beschreiben wie die Personen das angestrebte System nutzen werden. Ebenfalls sollen hier durch die Anforderungen und die Benutzbarkeit verdeutlicht werden. Beschriebene Szenarien ähneln eher eine Geschichte als einem technischen Vorgehen, hierdurch wird, wie bei den Persona, ermöglicht das Beschriebene nachzuempfinden und fördert das User-Centered-Design.

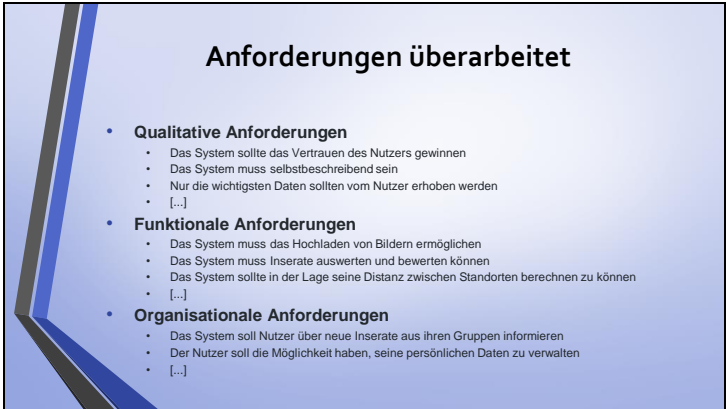
| Verhaltensmodellierung - Use Cases | |
|------------------------------------|---|
| Profilinformationen aktualisieren | |
| USE CASE | Profil aktualisieren |
| Scope & Level | Nutzerverwaltung, Accountverwaltung, Primary Task |
| Preconditions | Account ist im System vorhanden, Nutzer ist authentifiziert |
| Success End Condition | Nutzer hat erfolgreich sein Profil aktualisiert |
| Failed End Condition | Nutzer kann sein Profil nicht aktualisieren |
| Actors | Nutzer |
| Trigger | Nutzer möchte seine persönlichen Informationen aktualisieren |
| DESCRIPTION | <ol style="list-style-type: none"> 1. Nutzer navigiert zu seinem Profil 2. Nutzer wählt "Profil bearbeiten" aus 3. Nutzer bearbeitet gewünschte Information 4. Nutzer bestätigt die Eingabe |
| EXTENSIONS | <ol style="list-style-type: none"> 3a. Eingabe entspricht nicht der geforderten Vorgabe -> Nutzer wird informiert 4a. Server nicht erreichbar -> Nutzer wird informiert |

Bei den von uns verwendeten Use Cases handelt es sich um Concrete Use Cases. Es werden gezielt und konkret bestimmte Funktion und Eigenschaften adressiert und beschrieben. Im Gegensatz zu den Essential Use Cases sind diese also nicht technologiefrei und implementierungsunabhängig. Für die Umsetzung wurde das Template von Alistair Cockburn verwendet.

Der hier gezeigte Use Case stellt nur ein Beispiel dar, in den Artefakten sind weitere zu finden.



Um die spätere Entwicklung und Implementierung zu beschleunigen und erleichtern, wurde ein Feature Set entworfen. Dies kann als Anhaltspunkt dienen und darstellen welche Features noch umgesetzt werden sollten. Ebenfalls dient es bei der Evaluation, die wichtigsten Punkte mit den Probanden durchzuarbeiten, um Schwächen und Stärken des Systems zu überprüfen und bewerten. Das Feature Set wurde aus den Anforderungen, der geplanten Architektur und Use Cases gewonnen.



Anforderungen überarbeitet

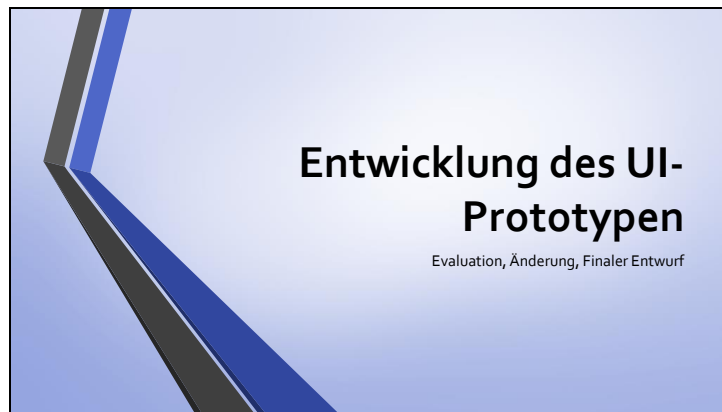
- **Qualitative Anforderungen**
 - Das System sollte das Vertrauen des Nutzers gewinnen
 - Das System muss selbstbeschreibend sein
 - Nur die wichtigsten Daten sollten vom Nutzer erhoben werden
 - [...]
- **Funktionale Anforderungen**
 - Das System muss das Hochladen von Bildern ermöglichen
 - Das System muss Inserate auswerten und bewerten können
 - Das System sollte in der Lage seine Distanz zwischen Standorten berechnen zu können
 - [...]
- **Organisationale Anforderungen**
 - Das System soll Nutzer über neue Inserate aus ihren Gruppen informieren
 - Der Nutzer soll die Möglichkeit haben, seine persönlichen Daten zu verwalten
 - [...]

Die Anforderungen wurden mit den neuen Erkenntnissen erweitert und spezifischer ans System angepasst. Es gilt immer noch dem Nutzer das bestmögliche System zu bieten. Die hier gezeigten Anforderungen stellen nur Beispiele dar und nicht die gesamte Liste.

Die funktionalen Anforderungen beschreiben einzelne Funktion, die unser System können muss, um einwandfrei zu funktionieren und das angestrebte Ziel zu erreichen.

Im Bereich der organisationalen Anforderung befindet sich vor allem der Schwerpunkt der Verwaltung und Aktualisierung von Daten im Bereich Profil und Inserat.

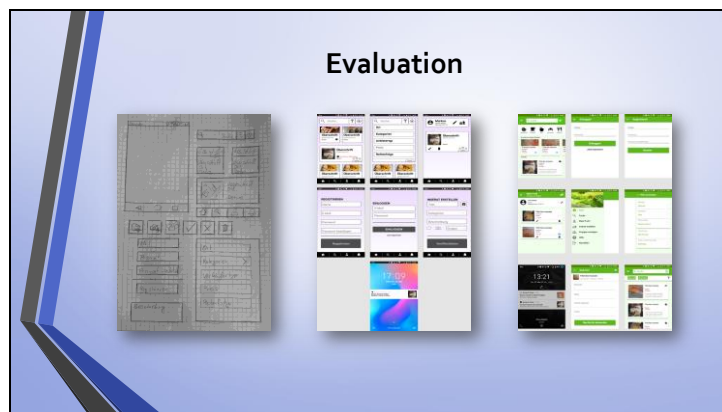
Bei den qualitativen Anforderungen muss darauf geachtet werden, dem Nutzer einen hohen Standard anbieten zu können. Das System muss vor Allem vertrauenswürdig sein und leicht zugänglich für jegliche Art an Nutzer sein.



Um das bestmögliche Ergebnis für den Nutzer zu erstellen, wurden mehrere Testdurchläufe mit verschiedenen Prototypen durchgeführt. Es wurden anhand der Benutzer- und Verhaltensmodellierung die wichtigsten Funktionen getestet und überprüft. Um ein möglichst breites Spektrum an Meinungen und Kritik zu bekommen, wurden Probanden aus verschiedenen Kreisen gewählt. So wurden unter anderem Kommilitonen, junge UX-Designer aus Softwareunternehmen gefragt, aber auch ältere Menschen, die weniger geübt im Umgang mit moderner Technik sind.

Die jüngeren und geschulten Probanden hatten keine Probleme mit dem Cognitive Walkthrough, ebenfalls von sich aus bewusst die Think Aloud Methode verwendet. Bei den unerfahrenen Nutzern wurde das geplante Vorgehen zunächst erklärt, ihr Testverlauf aufgezeichnet und anschließend analysiert und ausgewertet.

Im Verlauf der Implementierung werden weitere kleinere Tests durchgeführt, um Details zu evaluieren. Hierfür müssen jedoch keine Cognitive Walkthroughs mehr verwendet werden, sondern Think Aloud Methoden sind vollkommen ausreichend.

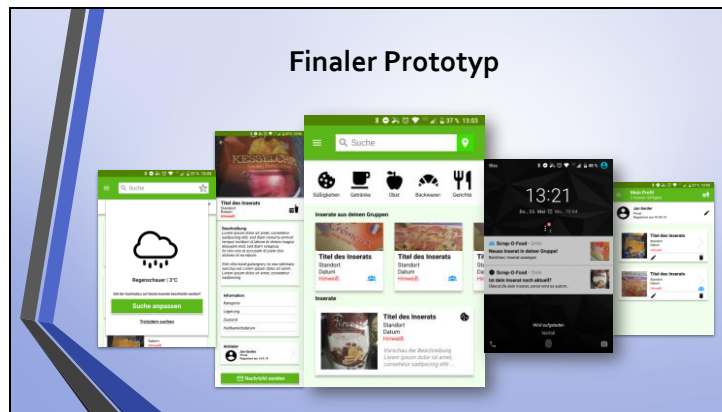


Ziel der Umsetzung ist es, ein System zu entwerfen was für alle Nutzer intuitiv und einfach zu verwenden ist. Die Probanden mit weniger Erfahrung fanden sich zunächst nur schwer zurecht, weil ihnen der Entwurf zu abstrakt gestaltet war. Laut eigener Aussagen benutzen sie nicht viele Anwendung auf ihren mobilen Geräten, aber die die sie verwenden, ähneln einander in der grafischen Oberfläche, so dass sie sich dort zumindest zurecht finden konnten. Als wir sie nach der Entwicklung des High Fidelity Prototype den Testlauf nochmal durchlaufen ließen, fanden sie sich viel schneller zurecht und konnten erklären, was sie tun wollen oder gesucht haben.

Die UX-Designer haben einige Gestaltungsentscheidungen kritisiert und als nicht sinnvoll für diese Art Projekt empfunden. So wurde die ursprünglich angedachte "Homebar" mit 4 Reitern kritisiert. Nutzer, die kein Interesse an der Accounterstellung haben und die Anwendung nur zum Suchen nutzen möchten, können diese als störend empfinden. Ebenfalls sollte überlegt werden, ob die Push-Benachrichtigung wirklich sinnvoll als eigener Tab wäre. Um eine bessere Übersicht zu bieten, welcher Personenkreis das Inserat erstellt hat - gewerblich, privat oder in einer Gruppe, sollten diese markiert und verschieden hervorgehoben werden.



Aus der Grundversion des Paper Based Prototype wurde eine einfach gehaltene Low Fidelity digitale Version erstellt. Durch Testläufe mit diesen beiden Versionen konnte dann eine High Fidelity digitale Version entwickelt werden. Diese ist in ihrer Form viel moderner gestaltet und angepasst an heutige Standards. Mit dieser wurden finale Testdurchläufe durchgeführt.



Mit der Entwicklung des User-Interface Prototype wurde eine intuitive und verständliche Oberfläche zur Verwendung des Systems angestrebt. Der allgemeine Aufbau ist für jeden Nutzer identisch, wird jedoch für registrierte Anwender erweitert. Gestaltungsideen kamen durch moderne und bereits etablierte Anwendungen, um dem Nutzer ein vertrautes Umfeld zu generieren. Ebenfalls wurde das Feedback und die Kritik der Evaluation berücksichtigt und das System an die Bedürfnisse und Wünsche der potentiellen Nutzer angepasst. Farblich wurde ein frisches grün gewählt, um den positiven Charakter der Anwendung in Bezug auf die Reduzierung der Lebensmittelverschwendung zu unterstreichen. Akzente und Aktionsfelder sind in der gleichen Farbe dargestellt, um das Verständnis für diese zu fördern. Zur einfacheren Differenzierung wird neben dem weißen Hintergrund zusätzlich eine schwächere Variante des verwendeten Grüntons genutzt.