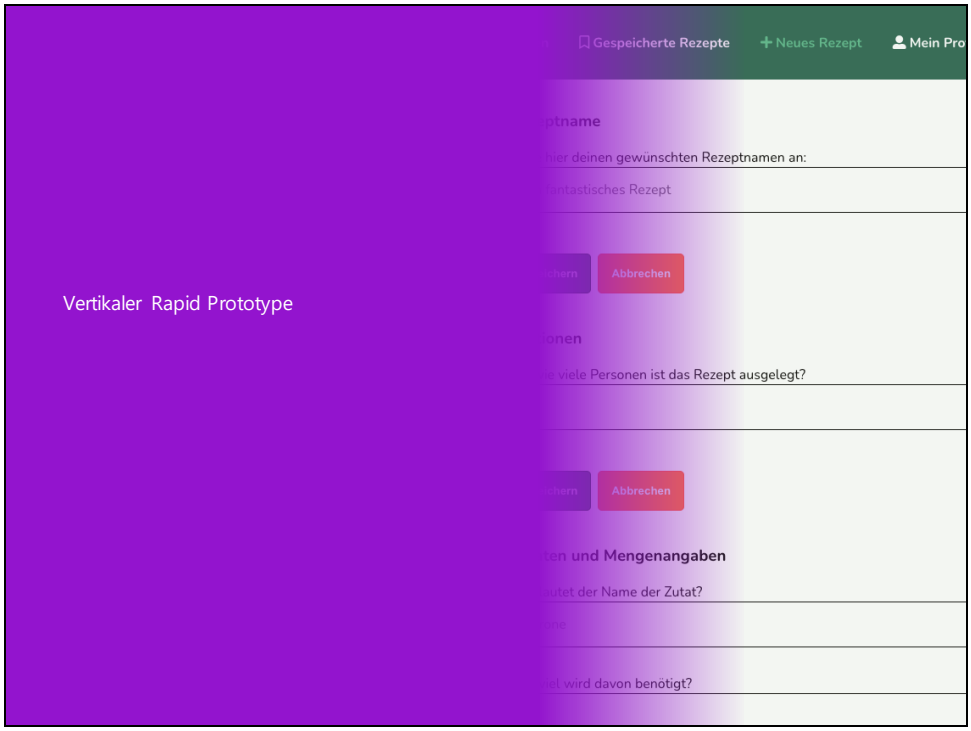
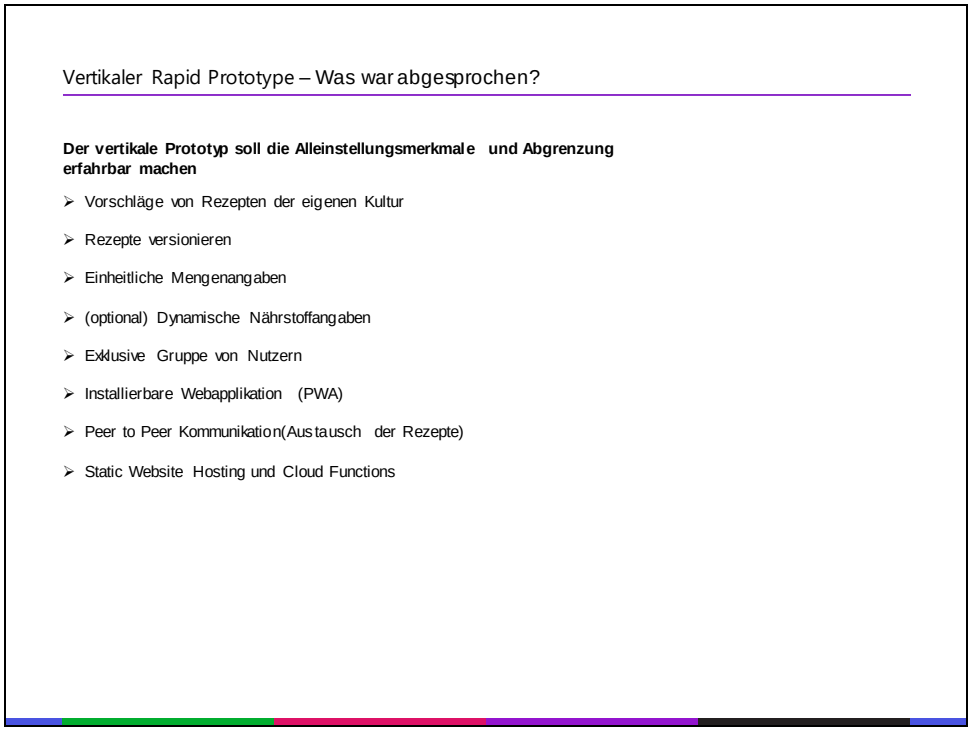
Audit Drei – EPWS2020CobanMai

Artefakte: Erster vertikaler Rapid Prototype, Durchgeführte PoCs, Iterierte Modellierungen und Modellierungsbegründungen, Modellierung der Anwendungslogik in Pseudocode, Deliverables für den 4. Audit (Projektplan)



Mit unserem vertikalen Prototyp haben wir ausgewählte Funktionen unseres Zielsystems
durch Systemschichten hindurch imple mentieren können.
Diese Technik ist daher sehr geeignet, weil Funktionalitäts-
und Implementierungsoptionen somit g eklärt werden können.
Ein vertikaler Prototyp ermöglicht es, e

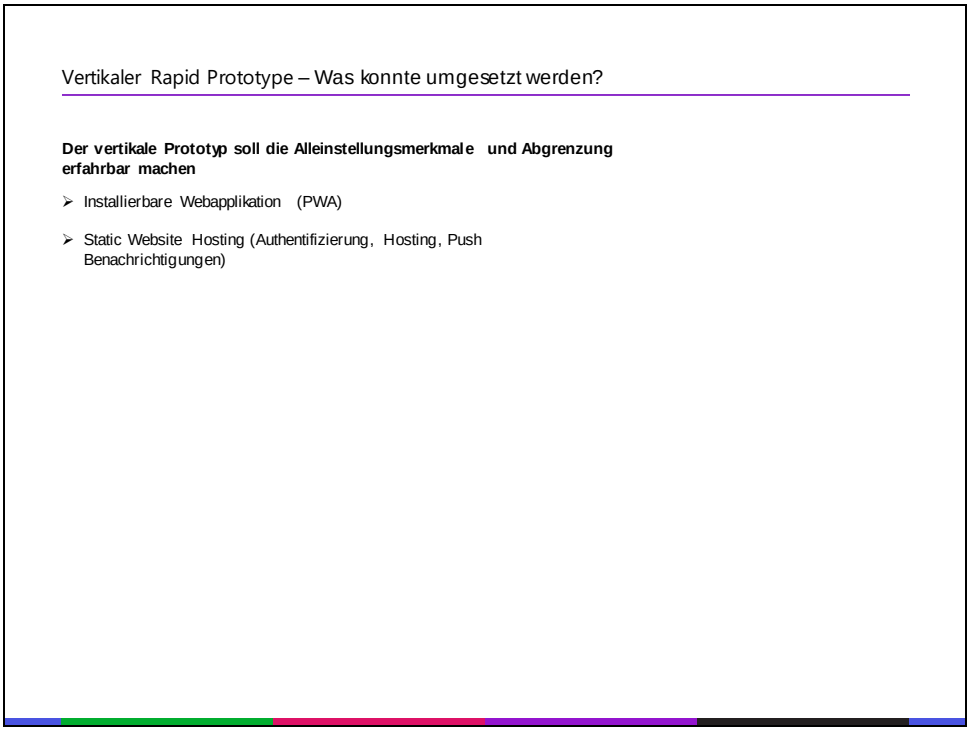
inzelne Anwendungsfälle oder eine Zieltechnologie auszuprobieren oder die generelle Umsetzbarkeit zu testen.



Im vertikalen Prototype kriegt man unter dem Button "Entdecken" Vorschläge von Rezepten der eigenen Kultur angezeigt.
Die bereits hochgeladene Rezepte kann man durch persönliche Änderungen versionisieren.
Die einheitliche Mengenangaben gehören ebenfalls zu unseren Alleinstellungsmerkmalen.
Die dynamische Nährstoffangaben, wie

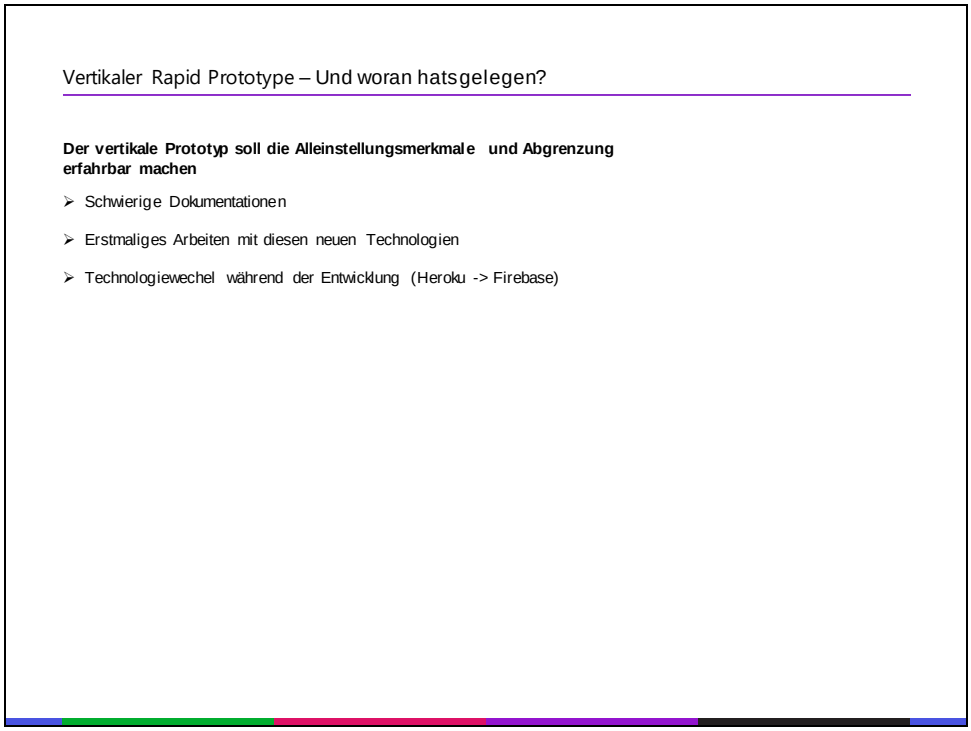
zum Beispiel wie viele Kalorien dieses Rezept hat, sind optional anzugeben.
Die PWA hat eine exklusive Gruppe von Nutzern, wie z.B. nur ersten
grades Familienmitglieder.
Unser Projekt wird eine installierbare Webapplikation(PWA), welche auch aus einer responsiven Webseite und einer App beschrieben werden können.
Der Austausch der Rezepte erfolgt über die Peer to Peer Kommunikation, das heißt, dass die Nutzer direkt miteinander verknüpft sind (von Computer zu Computer)
Zuletzt gehört zu unseren Alleinstellungsmerkmalen das Static Website Hosting, die die HTML-basierte Website mit festem Inhalt unterstützt, die allen Nutzern die gleiche Informationen angeben und die Cloud Functions, mit der wir die einfache

Einzweckfunktionen schreiben, die an Ereignisse angehängt sind, die von ihrer Cloud Infrastruktur und ihren Cloud Diensten ausgegeben werden.



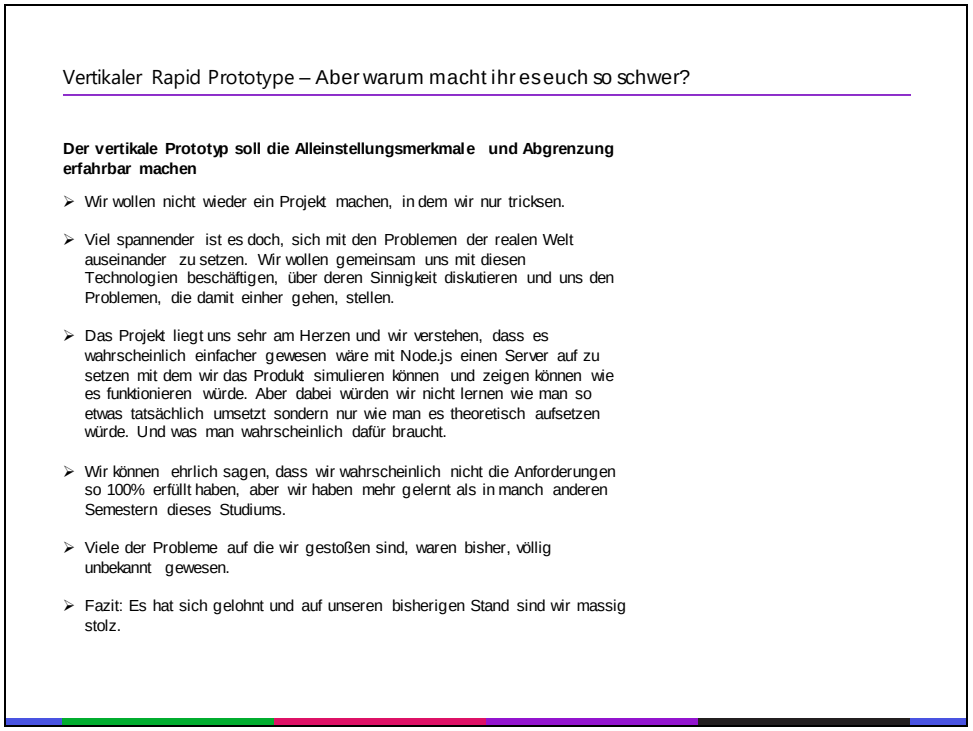
Was wir jedoch von unseren Alleinstellungsmerkmalen bis jetzt umsetzen konnten, ist die installierbare Webapplikation (Progressiv Web App). Mit dem Static Website Hosting konnten wir bisher in Erfahrung bringen, dass man sich schon Authentifizieren, Ein-und Ausloggen kann, das Hosting klappt(Adresse ist über Internet erreichbar) und dass die Push

Benachrichtigungen soweit funktionieren.



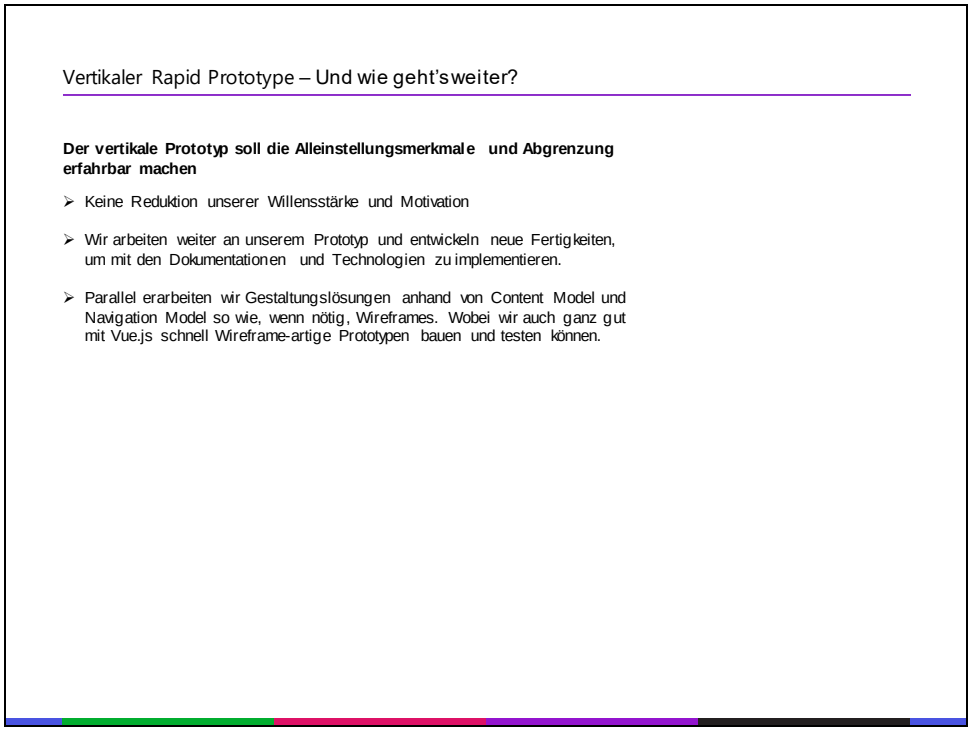
Bei der Erstellung unseres Prototypes sind wir ein bisschen in Zeitdruck gewesen. Trotz des täglichen Arbeits, hatten wir das Gefühl, für all das, was wir uns vorgenommen haben, mehr Zeit zu benötigen. Das Dokumentieren war auch sehr schwierig. Wir haben mit diesen neuen Technologien das erste mal befasst, daher mussten wir uns zuerst einmal informieren und reinarbeiten. Des

weiteren hatten wir einen Technologiewechsel während der Entwicklung. Wir sind von Heroku zu Firebase gewechselt. Da es bei Heroku zu mehreren Nachteilen kam, wie zum Beispiel, dass die Konfigurierbarkeit relativ eingeschränkt ist und dies könnte zu Probleme führen, beispielsweise der Skalierbarkeit so wie die portierung neuerer Versionen aber auch das Testen neuer Änderungenen an der Zielgruppe. Daher haben wir uns für firebase entschieden. Hier kann man Hosten, Pushbenachrichtigungen senden, Authentifizieren über etablierte sichere Apis und AB Testing betreiben sowie vieles mehr an das wir momentan noch gar nicht denken.



Wir wollen unbedingt auf die von uns vorgenommenen Punkte eingehen, denn der Wille ein wirklich gutes Projektes abzugeben, ist für uns sehr wichtig. Daher setzen wir uns mit Problemen aus der realen Welt auseinander und beschäftigen uns mit den Technologien und über deren Sinnigkeit diskutieren und uns den Problemen, die damit einher gehen, stellen.

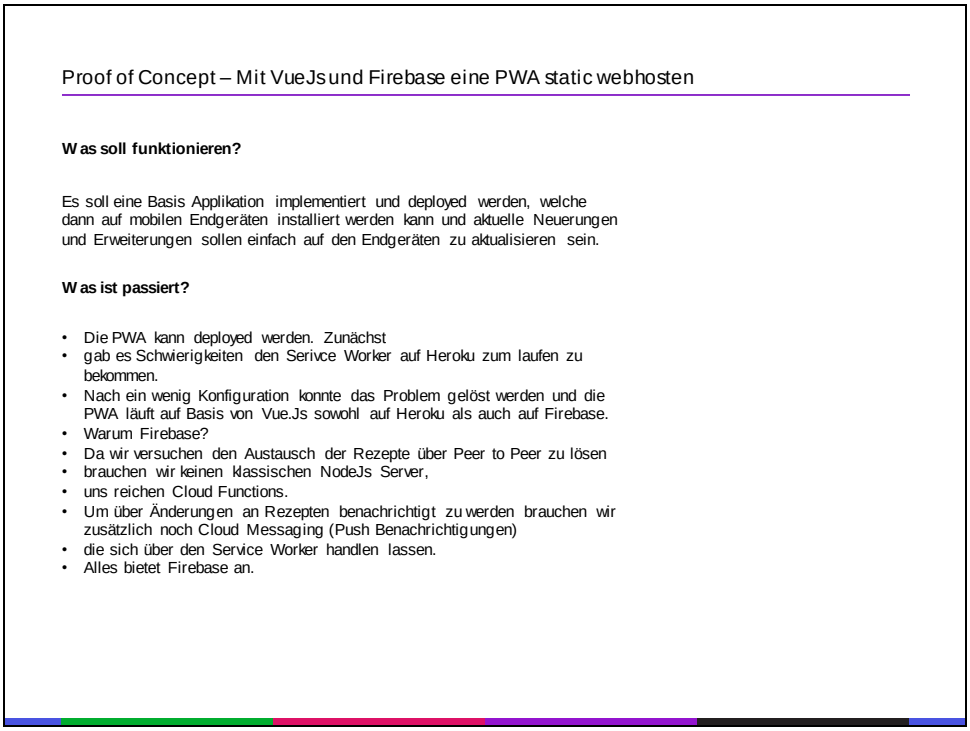
Unser Ziel ist es, bei diesem Projekt wirklich etwas zu lernen. Denn nur das Arbeiten mit dem node.js würde uns zeigen, wie man es theoretisch aufsetzen würde und was man alles dafür braucht. Wir wollen wissen, wie man so ein Projekt tatsächlich umsetzen kann.
Uns ist auch bewusst, dass wir die Anforderungen für den Audit 3 nicht zu 100% erfüllt haben, wobei wir mehr gelernt haben als in manch anderen Semestern dieses Studiums.
Viele der Probleme, auf die wir währenddessen gestoßen sind, waren uns bisher völlig unbekannt gewesen.
Aus diesem Grund, sind wir mit unserem aktuellen Wissensstand, welchen wir in so einer kurzen Zeit erarbeitet haben, sehr glücklich und stolz darauf.



Natürlich geht es für uns nach dem dritten Audit weiter und möchten das Projekt komplett und erfolgreich zu Ende bringen und daher gibt es für uns keine Reduktion unserer Willensstärke und Motivation.
Wir arbeiten an unserem Prototyp weiter und entwickeln neue Fertigkeiten, um mit den Dokumentationen die Technologien zu implementieren.
Parallel erarbeiten wir die

Gestaltungslösungen anhand von Content Model und Navigation Model so wie, wenn nötig, Wireframes.
Wir können auch rasch mit Vue.js Wireframe-artige Prototypen bauen und
diese auch testen.

Unsere durchgeführten Proof of Concepts (PoC) dienen der Überprüfung der Umsetzbarkeit unserer Idee in der Realität.
Das Proof of Concept soll die Machbarkeit unseres Projektes beweisen oder zeigen, ob unser Projekt wie geplant funktioniert.



PWA aus Vue.js Komponenten

Aus welchem Projektrisiko leitet sich der PoC ab?

Einfaches Finden und Installieren der PWA

Mit Vue.js sollte es einfach sein eine PWA mit Manifest anzulegen welche einfach instand zu halten ist.

Eine PWA ist modern und auf fast allen mobilen Geräten zu installieren,

selbst wenn nicht, über den Browser nutzbar.

Was wird wie umgesetzt?

Es soll eine Basis Applikation implementiert und deployed werden, welche dann auf mobilen Endgeräten installiert werden kann und aktuelle Neuerungen und Erweiterungen sollen einfach auf den Endgeräten zu aktualisieren sein.

Exit-Kriterien - Erfolgsfall

Die Basis Applikation lässt sich:

installieren,

Updates werden im besten Fall automatisch installiert beim online Aufruf der Seite/App.

Fail-Kriterien - Misserfolg

Die PWA lässt sich nicht installieren.

Updates werden nicht erkannt und

installiert.

Fallbacks - alternative Lösungsansätze

Die Updates müssen manuell installiert werden und der Nutzer auf das Update hingewiesen werden.

Dokumentation

Welche Exit- und Fail-Kriterien sind eingetroffen

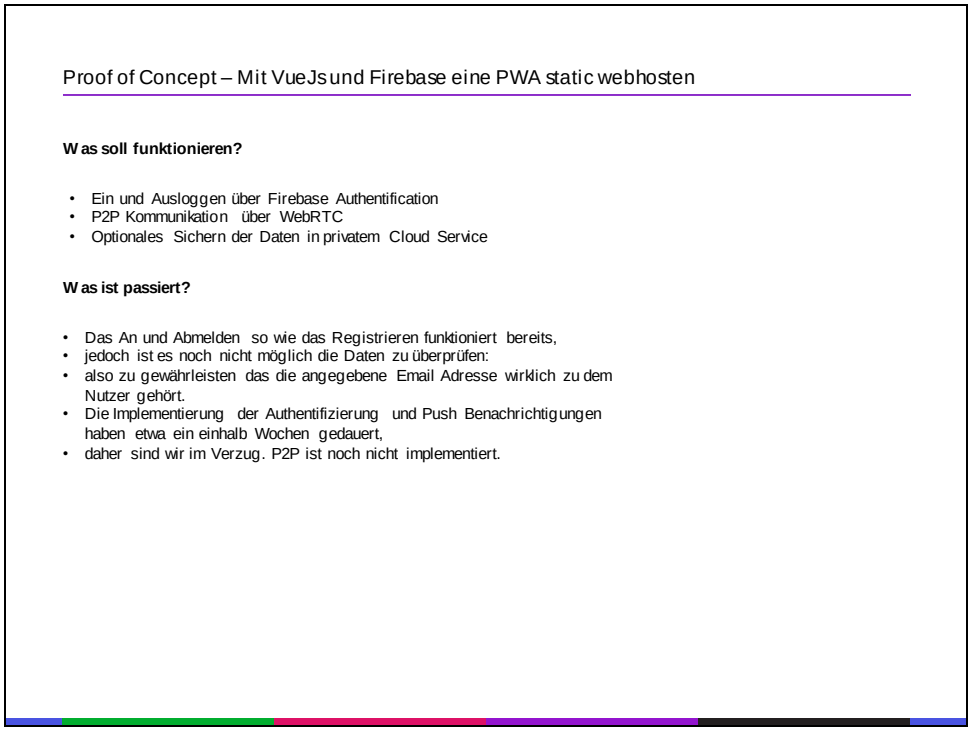
Good News Everyone. Die PWA kann deployed werden. Zunächst gab es Schwierigkeiten den Serivce Worker auf Heroku zum laufen zu bekommen. Nach ein wenig Konfiguration konnte das Problem gelöst werden und die PWA läuft auf Basis von Vue.Js sowohl auf Heroku als auch auf Firebase. Warum Firebase? Da wir versuchen den Austausch der Rezepte über Peer to Peer zu lösen brauchen wir keinen klassischen NodeJs

Server, uns reichen Cloud Functions. Um über Änderungen an Rezepten benachrichtigt zu werden brauchen wir zusätzlich noch Cloud Messaging (Push Benachrichtigungen) die sich über den Service Worker handlen lassen. Alles bietet Firebase an. Ich hab also für unser Projekt dort meine Zahlungsdaten hinterlegt und wir sind nun in der Lage (Theoretisch) all die angenehmen Lösungen von Firebase zu nutzen. Das Deployment auf Firebase war um einiges Reibungsloser als bei Heroku. Einfach über NPM das Package installiert und über die CLI und JSON konfiguriert. VueJs ist mir von der Arbeit her bekannt, daher eine angenehmere Alternative als alles über HTML und CSS und JS selbst zu handeln. Leider hatte Derya ein paar

Einstiegsschwierigkeiten, was verständlich ist, da sie Vue oder ähnliche Frameworks bisher nicht kannte. Wir versuchen daher auch sie eher in der Entwicklung der notwendigen Komponenten einzusetzen und mir die Konfiguration und Recherche der Lösungswege zu überlassen.

(Optional:) Welche Gründe gab es für den Misserfolg

Es kam leider hin und wieder vor das das manuelle leeren des Caches notwendig war, um Updates zu installieren. Der Hintergrund dies bezüglich ist mir noch nicht bekannt.



Peer to Peer Kommunikation über WebRTC

Aus welchem Projektrisiko leitet sich der PoC ab?

Verfügbarkeit des Services -> Server offline -> keine Synchronisierung unter Nutzern

Da die Nutzer über Peer to Peer Daten austauschen, braucht es keinen Server der die Kommunikation leitet,

Nutzer sollten einfach über Push Benachrichtigungen eine Peer to Peer Verbindung über WebRTC aufbauen können und dann Rezepte unabhängig von uns austauschen können.
Schwere Implementierung -> Als zweier Team sollte die Integration einzelner Komponenten für uns möglichst einfach sein

WebRTC ist noch recht neu und die Dokumentation unzureichend. Wir werden wohl viel Zeit brauchen um uns reinzuarbeiten

Wie genau wir die Peers verbinden ist noch nicht klar
Anforderungen wurden nicht richtig ermittelt -> unzureichende Nutzung der Plattform

P2P ist sicher eine gute Lösung um

Vertrauen in den Service zu schaffen, die Frage ist ob das P2P Nutzer nicht eher überfordert...

Sicherer Austausch der Daten

Da die Rezepte nur bei den Peers liegen und nicht irgendwo gesichert (optional wäre das private sichern in einem Cloud Service unabhängig von uns) ist der Austausch und dessen Sicherheit abhängig von WebRTC

Zugang zu Accounts sichern

Nutzer sollten sich ein und ausloggen können und so die Sicherheit ihrer Daten gewähren.

Was wird wie umgesetzt?

Ein und Ausloggen über Firebase Authentification

P2P Kommunikation über WebRTC

Optionales Sichern der Daten in privatem

Cloud Service

Exit-Kriterien - Erfolgsfall

Nutzer können sich ein uns ausloggen und haben dann Zugriff auf ihre Rezepte.

Nutzer können ihre Geräte synchronisieren und anderen Peers Rezepte schicken

Nutzer können ihre Rezepte in einem geeigneten Service privat sichern.

Fail-Kriterien - Misserfolg

Das Ein und Ausloggen funktioniert nicht

Nutzer haben keinen Zugriff auf ihre Rezepte

Geräte lassen sich nicht synchronisieren

Nutzer können nicht über P2P Rezepte versenden

Nutzer können die Rezepte nicht exportieren und privat sichern.

Fallbacks - alternative Lösungsansätze

Nutzer können ihre Rezepte in unserem Service sichern und teilen
Nutzer können sich über alternativen ein und ausloggen

Dokumentation

Das An und Abmelden so wie das Registrieren funktioniert bereits, jedoch ist es noch nicht möglich die Daten zu überprüfen: also zu gewährleisten das die angegebene Email Adresse wirklich zu dem Nutzer gehört. Die Implementierung der Authentifizierung und Push Benachrichtigungen haben etwa ein einhalb Wochen gedauert, daher sind wir im Verzug. P2P ist noch nicht implementiert.

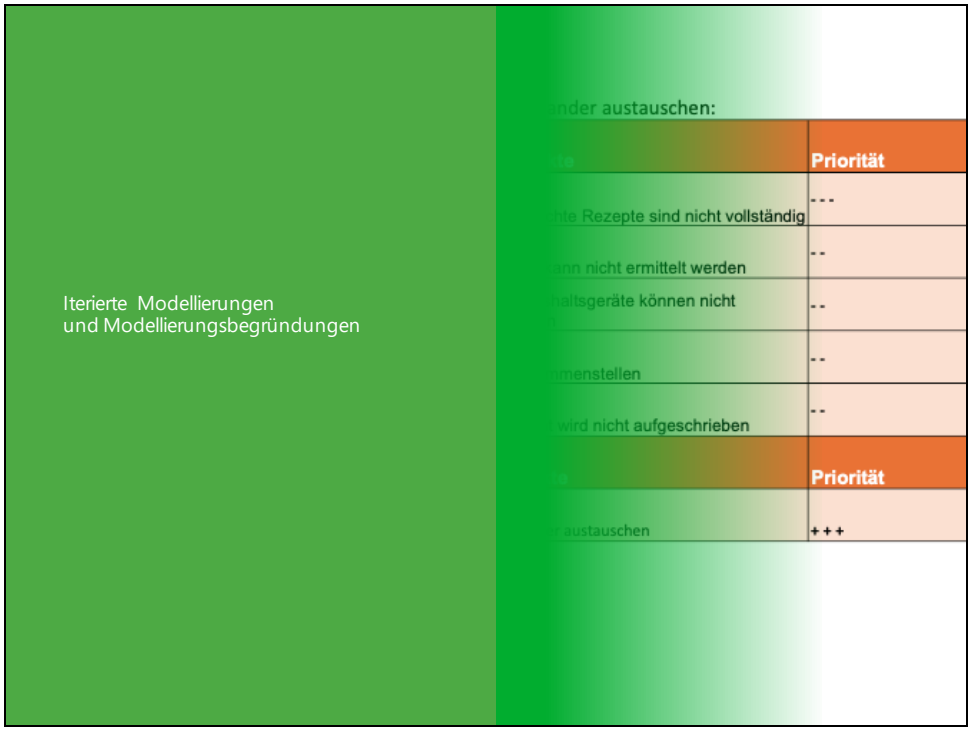
Welche Exit- und Fail-Kriterien sind eingetroffen

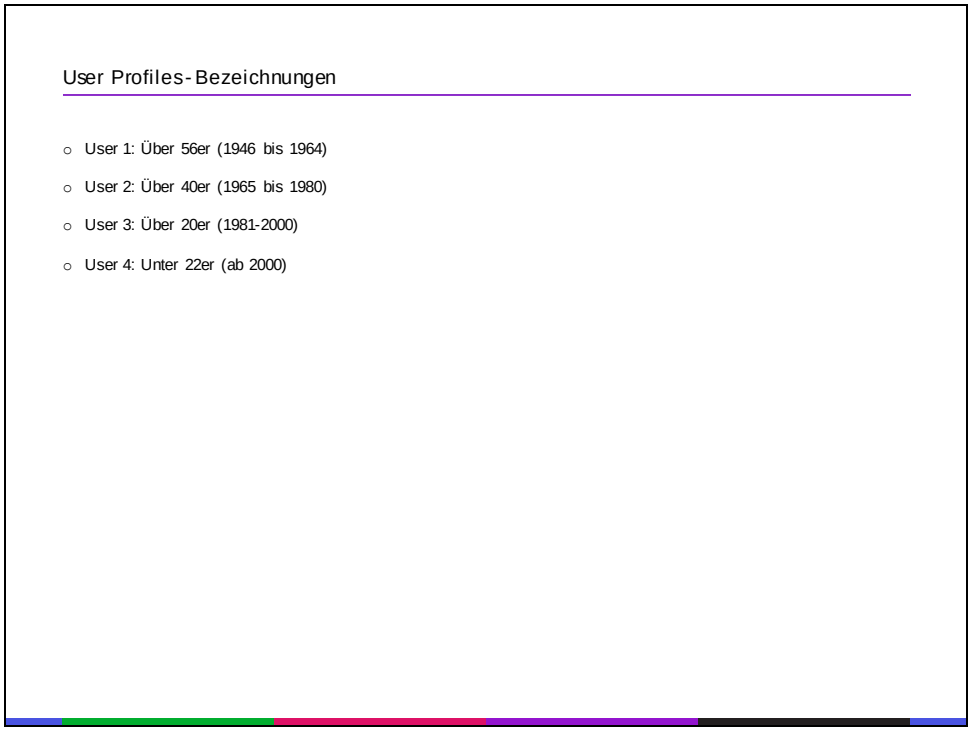
(Optional:) Welche Gründe gab es für

den Misserfolg

P2P ist recht aufwendig für das Web und wir sind noch nicht soweit gekommen

Iterierte Modellierungen
und Modellierungsbegründungen





Wir haben die User Profiles wie im Audit 2 besprochen, umgeändert und an HCI angepasst.

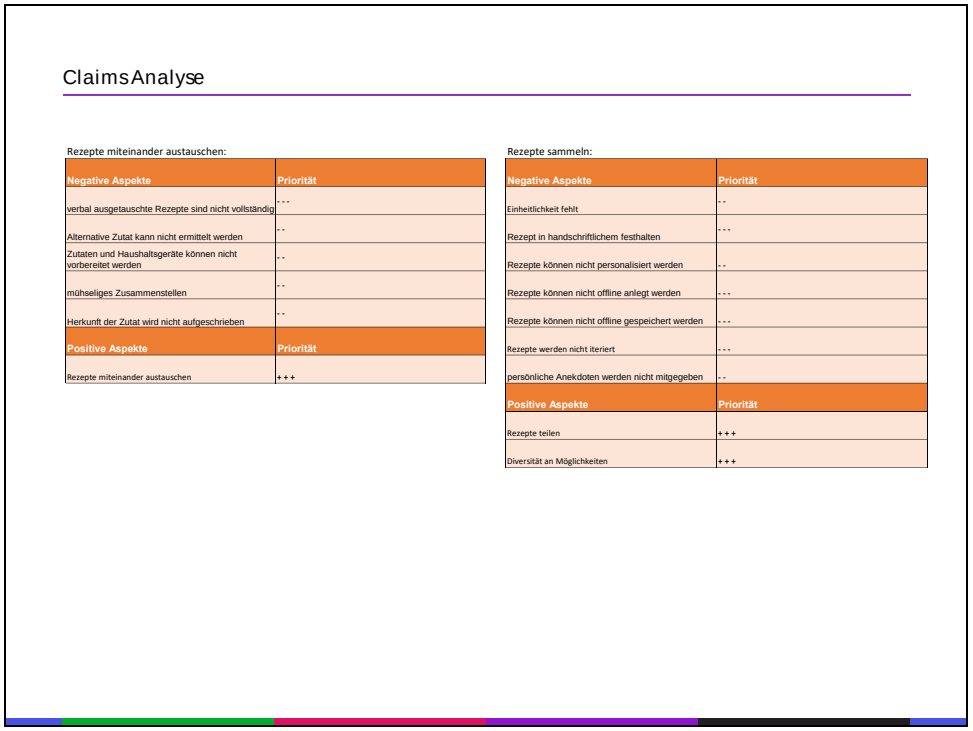
Aus Baby Boomer haben wir Über 56er gemacht.

Aus Generation X haben wir Über 40er gemacht.

Aus Generation Y haben wir Über 20er gemacht und

Aus Generation Z haben wir Unter 22er

gemacht.



ClaimsAnalyse

Rezepte miteinander austauschen:

Negative Aspekte	Priorität
verbal ausgetauschte Rezepte sind nicht vollständig	---
Alternative Zutat kann nicht ermittelt werden	--
Zutaten und Haushaltsgeräte können nicht vorbereitet werden	--
mühseliges Zusammenstellen	--
Herkunft der Zutat wird nicht aufgeschrieben	--
Positive Aspekte	**Priorität**
Rezepte miteinander austauschen	+++

Rezepte sammeln:

Negative Aspekte	Priorität
Einheitlichkeit fehlt	--
Rezept in handschriftlichem festhalten	---
Rezepte können nicht personalisiert werden	--
Rezepte können nicht offline anlegt werden	---
Rezepte können nicht offline gespeichert werden	---
Rezepte werden nicht iteriert	---
persönliche Anekdoten werden nicht mitgegeben	--
Positive Aspekte	**Priorität**
Rezepte teilen	+++
Diversität an Möglichkeiten	+++

Die Claims Analyse ist eine Technik zur Untersuchung der positiven und negativen Folgen von Konstruktionsmerkmalen, die in aktuellen oder zukünftigen Nutzungsszenarien beschrieben werden. Hierzu haben wir für alle einzelnen Problemszenarien eine Claims Analyse erstellt.

Erst wurden die negativen Aspekte aufgelistet und mit "-", "--" oder "---"

wurden die Prioritäten vergeben.
Danach haben wir die positiven Aspekte aufgelistet und hier wurden die Prioritäten mit "+", "++" oder "+++" vergeben.

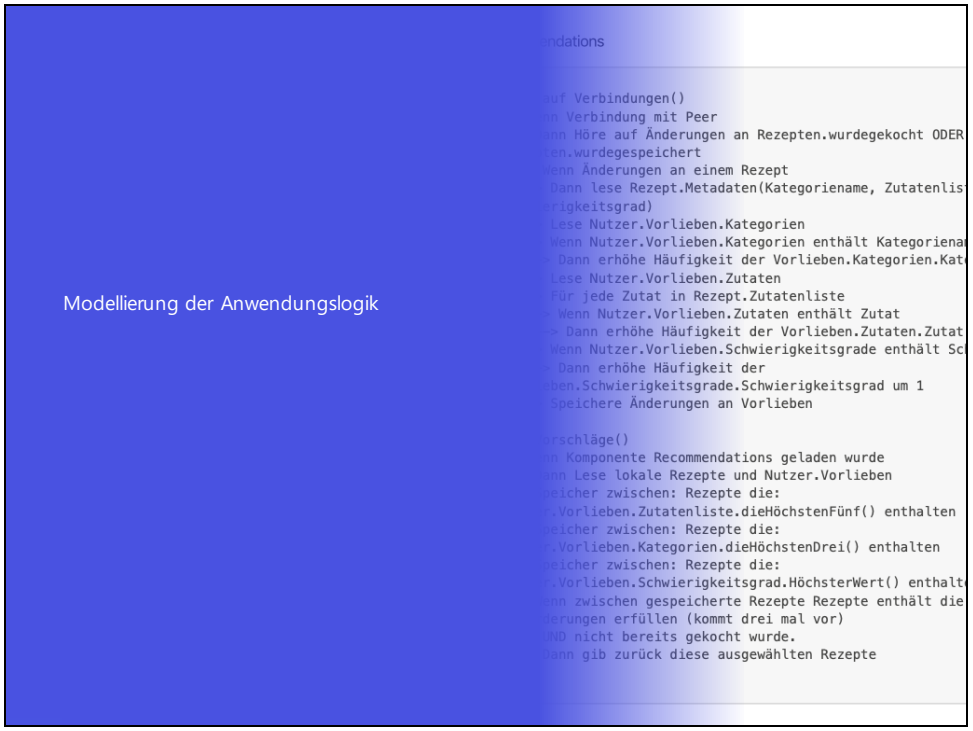
ClaimsAnalyse

Rezepte nachkochen:

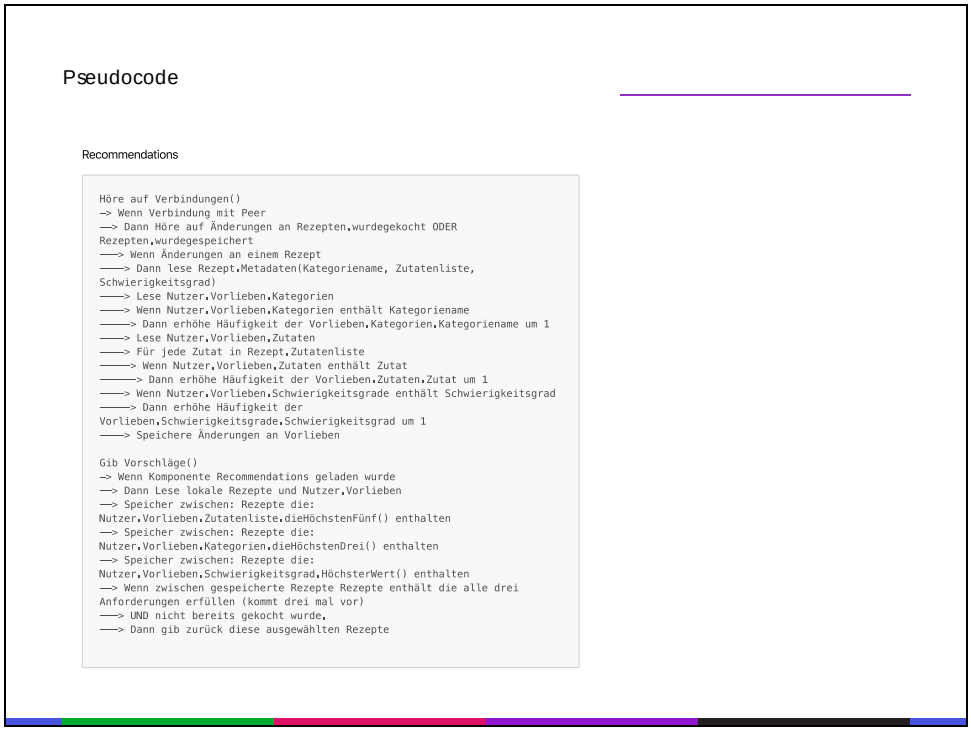
Negative Aspekte	Priorität
mögliche Handgriffe stehen nicht im Rezept	- -
keine genaue Mengenangabe	- - -
Rezepte werden nicht übersetzt	- - -
besondere Zutaten werden nicht markiert	- -
benötigte Haushaltsgeräte werden nicht mitgeschrieben	- -
Tpps und Tricks können nicht langfristig angewendet werden	- -
Positive Aspekte	**Priorität**
Geschichtliche Hinweise mitgeben	+ + +

Eigenes Kochbuch erstellen:

Negative Aspekte	Priorität
Probleme beim Kochvorgang können nicht direkt angesprochen werden	- -
Rezeptaustausch ist zeitaufwendig	- - -
zu viele Rezepte können nicht sortiert werden	- - -
Rezepte können nicht vereinheitlicht werden	- - -
Positive Aspekte	**Priorität**
Verbundenheit der Familie wird gestärkt	+ + +
Kochbuch ist bedeutsam	+ + +
kann personalisiert werden	+ +



Use Case Diagramme visualisieren Anwendungsfälle und Akteure mit ihren jeweiligen Beziehungen. Sie liefern einen guten Überblick über das Gesamtsystem, beschreiben aber im Gegensatz zu den Use Case Spezifikationen keine Abläufe, sondern die Zusammenhänge zwischen einer Menge von Anwendungsfällen und den daran beteiligten Akteuren.



Was sind unsere Alleinstellungsmerkmale?

- [x] Besondere/Wichtige Zutaten als solche markieren
  - [x] Alternative Zutaten angeben
  - [x] Herkunft der Zutat (erhältlich von/über)
- [x] Offline speichern von Rezepten
- [x] Die eigene Kultur entdecken
- [x] Vereinheitlichung von Rezepten

- [x] Benötigte Haushaltsgeräte
- [x] Tipps und Tricks
- [x] Geschichtliche hinweise zu Rezepten
- [x] Vorratskammer & Register
- [x] Verweise auf Autor und Versionierung von Rezepten
- [x] Personalisierung der PWA (Farben)
- [x] Übersetzen von Rezepten

- Was sind triviale UseCases?

Pseudocode

Alternative Ingredient

```
gibAlternativen()
-> prüfe ob Alternative im aktuellen Rezept gefunden wurde
——> Wenn ja: Gib diese aus
——> Wenn nein: prüfe ob in der Sammlung alternativen Zutaten eine gefunden
wurde
———> Gib aus "Es wurden leider keine Alternativen eingetragen, aber vlt
passen dir ja diese:"
———> Gib aus andereAlternativen
```

Projektrisiken

Projektrisiken

Welche gesellschaftlichen Gefahren gibt es bei unserem Projekt?

- unzureichende Nutzung der Plattform -> kein erhalten und stärken von Kultur un
- mangelnder persönlicher Austausch der Nutzer untereinander

Auf welche technischen Probleme könnten wir stoßen?

- Anforderungen wurden nicht richtig ermittelt -> unzureichende Nutzung der Pla
- Verfügbarkeit des Services -> Server offline -> keine Synchronisierung unter Nu
- Schwere Implementierung -> Als zweier Team sollte die Integration einzelner Ko möglichst einfach sein
- Einfaches Finden und Installieren der PWA
- Client Server Architektur:
  - kann nicht beliebig skaliert werden
  - Netzausfall (z.b. wenn der Server überbelastet ist)
  - es wird eine große Bandbreite benötigt
  - möglicher Kontrollverlust über bereitgestellte Daten
- Peer to Peer Architektur:
  - kaum Sicherheit (nur bei geringer Anzahl von Computer geeignet)
  - bei Aktualisierungen müssen alle beteiligte Endgeräte reagieren
  - Verschlüsselung der Daten

Wie handeln wir, wenn sich die Anforderungen dramatisch ändern?

- Anforderungen neu ermitteln, System abgleichen und Release planen

Welche Kommunikationsprobleme könnten auftreten?

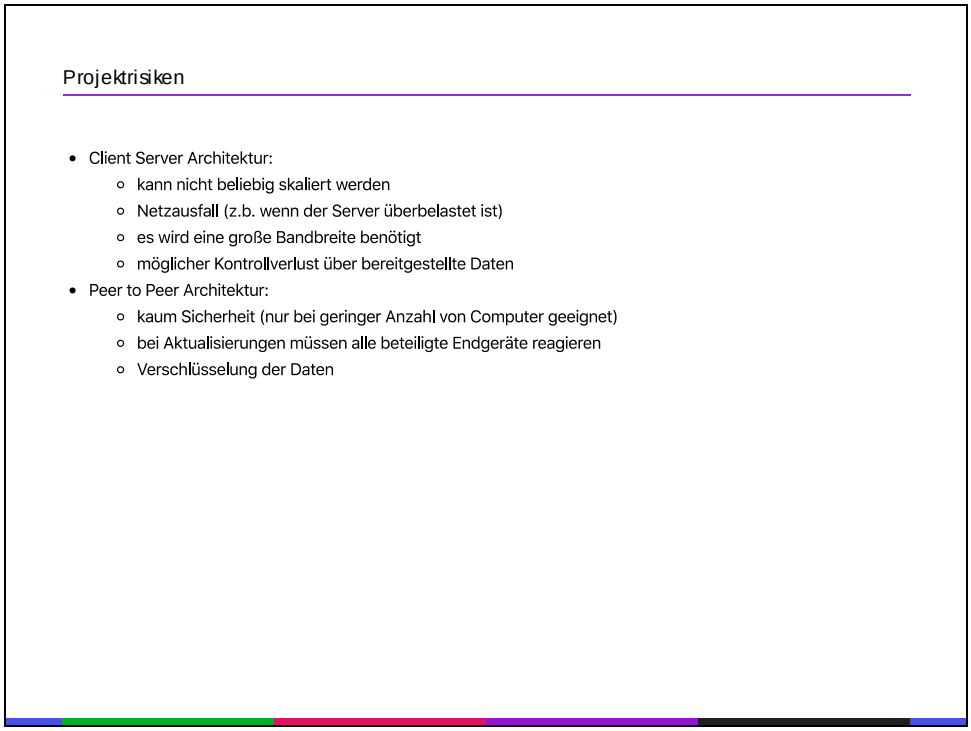
- Projektanforderungen wurden nicht richtig verstanden -> Zeitverlust für Projekta

Welche Zeitmanagementprobleme könnten auftreten?

- Keine überschneidenen Zeiträume der Teammitglieder -> Keine Absprache -> P wurden nicht richtig verstanden

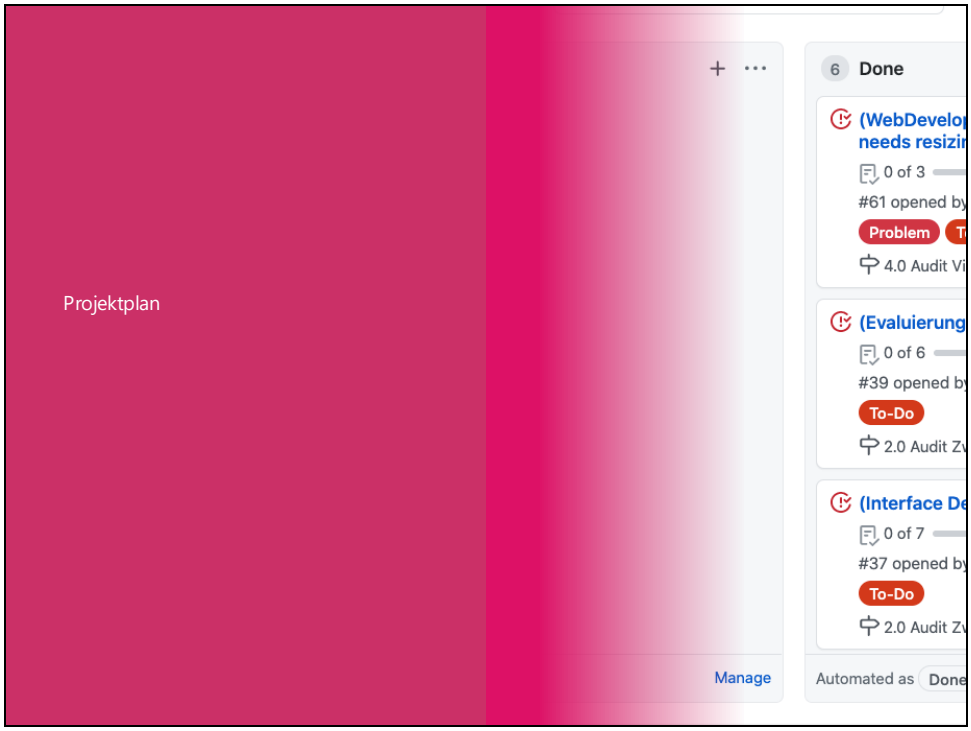
Welche Systemrisiken müssen wir abdecken (Sicherheit)?

- Verschlüsselung der Daten
- Sicherer Austausch der Daten
- Zugang zu Accounts sichern
- Ausfallsicherheit des Systems für Fehler minimieren



Hier haben wir die Projektrisiken zu den jeweiligen Architekturen aufgelistet.
Es geht um das Client Server Architektur, der nicht beliebig skaliert werden kann, es kann zu einem Netzausfall kommen, wenn beispielsweise der Server überbelastet ist, es wird eine große Bandbreite benötigt und es kann zu einem mögichen Kontrollverlust über bereitgestellte Daten kommen.

Dann gibt es noch die Peer to Peer Architektur, bei der es kaum Sicherheit gibt und nur bei geringer Anzahl von Computer geeiget ist, bei Aktualisierungen müssen alle beteiligte Endgeräte reagieren und ein weiteres Risiko besteht darin, dass die Daten verschlüsselt werden können.


Projektplan
6 Done
0 of 3
Problem
4.0 Audit Vi
(Evaluierung
0 of 6
#39 opened by
To-Do
2.0 Audit Zu
(Interface De
0 of 7
#37 opened by
To-Do
2.0 Audit Zu
Manage
Automated as Done

Projektplan und Deliverabels

- ✓ 11.1.2021 - Audit 3
- ✓ 17.1.2021 – WebRTC Implementierung / Rapid Prototype
- ✓ 24.1.2021 – Content Model & Vue-JS Komponenten & Firebase Functions
- ✓ 31.1.2021 – User List getter und setter & Anwendungslogik implementiert
- ✓ 7.2.2021 - Navigation Model & Wireframes/Frontend Rebuild
- ✓ 14.2.2021 – First fully functional Vertical Prototype
- ✓ 22.2.2021 Audit 4
  - Content Model
  - Navigation Model
  - High Fidelity Prototyp (Deployed and Set Up)
  - Fazit und kritisch reflektiertes Prozessassessment

Vielen Dank

Derya Coban und Joel Mai