

## Proof of Concept Bezeichnung

**Aus welchem Projektrisiko leitet sich der PoC ab?**

**Was wird wie umgesetzt?**

**Exit-Kriterien - Erfolgsfall**

**Fail-Kriterien - Misserfolg**

**Fallbacks - alternative Lösungsansätze**

**Dokumentation**

**Welche Exit- und Fail-Kriterien sind eingetroffen**

**(Optional:) Welche Gründe gab es für den Misserfolg**

---

## 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 Service 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 Node.js 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 handeln 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. Vue.js 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 Authentication
- P2P Kommunikation über WebRTC
- Optionales Sichern der Daten in privatem Cloud Service

### **Exit-Kriterien - Erfolgsfall**

- Nutzer können sich ein und 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

---

## **Rezeptvorschläge**

### **Aus welchem Projektrisiko leitet sich der PoC ab?**

- unzureichende Nutzung der Plattform
  - Wenn Nutzer selbst nicht aktiv nach ihrer Familienkultur suchen, besteht die Gefahr, dass das Interesse unzureichend ist und der Service stirbt. Um das zu verhindern sollen Nutzer Vorschläge

basieren auf ihrem Verhalten erhalten die ihnen gefallen könnten.

### **Was wird wie umgesetzt?**

- Nutzer müssen die Möglichkeit haben:
  - Rezepte anlegen zu können
  - Rezepte bearbeiten zu können
  - Rezepte speichern zu können
  - Rezepte kochen zu können
- Das System muss die Zutaten und Kategorien sowie andere für den Algorithmus relevanten Metadaten auswerten können
- Das System muss eine Komponente besitzen die aus diesen Daten und dem Algorithmus Vorschläge generieren kann
- Das System muss dem Nutzer Vorschläge präsentieren können.

### **Exit-Kriterien - Erfolgsfall**

- Alle Anforderungen werden erfüllt
- Nutzer nutzen die Vorschläge wie erwartet

### **Fail-Kriterien - Misserfolg**

- Nicht alle Anforderungen werden erfüllt
- Nutzer nutzen die Vorschläge nicht

### **Fallbacks - alternative Lösungsansätze**

- Die Anforderungen für den PoC werden angepasst
- Es wurden alternative Wege gefunden Nutzer zur Interaktion mit dem System zu motivieren.

### **Dokumentation**

Welche Exit- und Fail-Kriterien sind eingetroffen

(Optional:) Welche Gründe gab es für den Misserfolg

---

## **Alternative Zutatenvorschläge**

### **Aus welchem Projektrisiko leitet sich der PoC ab?**

- Barrierefreiheit
  - Nicht jeder Nutzer hat zwingend alle Zutaten verfügbar, es wäre sinnvoll Alternative Zutaten vorzuschlagen

### **Was wird wie umgesetzt?**

- Das System muss dem Nutzer die Möglichkeit bieten, für Zutaten alternative Zutaten vorzuschlagen.

- (Optional) Das System soll häufig vorgeschlagene Alternativ Zutaten zu einer Zutat erkennen und selbstständig für noch nicht eingetragene Alternativ Zutaten in Rezepten ergänzen.

### **Exit-Kriterien - Erfolgsfall**

- Die Alternativen Zutaten sind für Nutzer sichtbar und auswählbar,
- Die Alternativen Zutaten sind bearbeitbar
- (Optional) Die Alternativen Zutaten werden automatisch basierend auf den Aktivitäten der Nutzer vom System ergänzt/vorgeschlagen.

### **Fail-Kriterien - Misserfolg**

- Die Alternativen Zutaten sind nicht sichtbar/auswählbar.
- Die Alternativen Zutaten sind nicht mehr bearbeitbar.
- (Optional) Die Alternativen Zutaten werden nicht automatisch ergänzt.

### **Fallbacks - alternative Lösungsansätze**

- Die Anforderungen an Alternative Zutaten müssen überarbeitet werden.

### **Dokumentation**

Welche Exit- und Fail-Kriterien sind eingetroffen

(Optional:) Welche Gründe gab es für den Misserfolg

---

## **Erweiterte Funktionalität und Abgrenzung zu anderen Systemen**

### **Aus welchem Projektrisiko leitet sich der PoC ab?**

- Anforderungen wurden nicht richtig ermittelt
- unzureichende Nutzung der Plattform
- mangelnder persönlicher Austausch der Nutzer untereinander
- Barrierefreiheit

### **Was wird wie umgesetzt?**

- Das System soll Nutzern die Möglichkeit bieten, Zutaten zu priorisieren (Wichtig für das Rezept).
  - Bei dem angeben einer Zutat sollen Nutzer die Möglichkeit haben diese zu Priorisieren/die Wichtigkeit hervorzuheben.
  - Auch nach der Angabe sollen diese Erweiterungen noch getätigt werden können.
- Das System soll Nutzern die Möglichkeit bieten, die Herkunft/Verfügbarkeit über einer Zutat anzugeben/anzuzeigen.
  - Bei dem angeben einer Zutat sollen Nutzer die Möglichkeit haben die Herkunft/Verfügbarkeit über einer Zutat anzugeben/anzuzeigen.
  - Auch nach der Angabe sollen diese Erweiterungen noch getätigt werden können.

- Das System soll Nutzern die Möglichkeit bieten, Rezepte offline anzulegen und erst später zu synchronisieren/teilen.
  - Die PWA muss die Möglichkeit haben Rezepte lokal zwischen zu speichern/Cachen und dann nach Bedarf zu teilen.
- Das System soll Nutzern die Möglichkeit bieten, Mengenangaben zu vereinheitlichen und leichter umsetzbar zu machen.
  - Das System muss über die verschiedenen Mengeneinheiten Datensätze haben und die Umrechnung bestenfalls lokal effektiv durchführen.
- Das System soll Nutzern die Möglichkeit bieten, benötigte Haushaltsgeräte anzugeben/anzuzeigen.
  - Bei dem angeben eines Rezepts sollen Nutzer die Möglichkeit haben benötigte Haushaltsgeräte anzugeben/anzuzeigen.
  - Auch nach der Angabe sollen diese Erweiterungen noch getätigt werden können.
- Das System soll Nutzern die Möglichkeit bieten, Rezepte um Kommentare und Tipps sowie Feedback zu erweitern.
  - Das System muss Nutzern an den passenden Stellen die Möglichkeit einer Texteingabe sowie das Teilen dieser anbieten.
- Das System soll Nutzern die Möglichkeit bieten, häufig genutzte Zutaten als "Vorratskammer" anzuzeigen.
  - Das System muss die Zutaten in geteilten Rezepten auslesen und deren Häufigkeit berechnen sowie einen Schwellwert errechnen können.
  - Das System braucht einen Bereich in dem es die Erkenntnisse mit den Nutzern teilen kann.
- Das System soll Nutzern die Möglichkeit bieten, die Farben der PWA selbst zu gestalten (Personalisierung).
  - Die PWA muss in der Lage sein sich selbst zu aktualisieren um die Benutzer Bedürfnisse adäquat zu stillen.
- Das System soll Nutzern die Möglichkeit bieten, Rezepte übersetzen zu lassen.
  - Das System benötigt die Integration eines zuverlässigen Übersetzers und muss die zu Übersetzende Sprache als auch die Ausgabe Sprache erkennen/übermittelt bekommen.

### **Exit-Kriterien - Erfolgsfall**

- Die Erweiterungen harmonisieren miteinander.
- Nutzer nutzen die Erweiterungen.

### **Fail-Kriterien - Misserfolg**

- Die Erweiterungen funktionieren nicht wie erwartet.
- Nutzer nutzen die Erweiterungen nicht wie erwartet.

### **Fallbacks - alternative Lösungsansätze**

- Die Erweiterungen müssen überarbeitet/überdacht werden.

### **Dokumentation**

**Welche Exit- und Fail-Kriterien sind eingetroffen**

**(Optional:) Welche Gründe gab es für den Misserfolg**

