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 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 Peet 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. Die P2P Kommunikation verläuft folgendermaßen:

1. Nutzer melden sich über firebase an (oder registrieren sich)
2. Nutzer wählen unter Freunde hinzufügen ihre Freunde aus
3. Nutzer gehen auf Explore und aktualisieren die peers
4. Rezepte der online verfügbaren Freunde werden geladen und angezeigt

Was passiert unter der Haube? Im angemeldeten Zustand wird eine Verbindung zum Peer2Peer Broker aufgebaut und eine PeerId erhalten. Diese wird im Firebase Realtime Database hinterlegt zusammen mit den allgemeinen Nutzerdaten. Wird man nun als Freund hinzugefügt wird auch die aktuellste peerId gesichert. Werden nun die Rezepte der Freunde abgefragt, wird eine p2p Verbindung zu dem aktuell genutzten Gerät des Freundes aufgebaut und es wird nach neuen Rezepten gefragt. Der Freund antwortet mit dem Array der Rezepte und der Empfänger speichert diese unter Rezepten seiner Freunde. Geht einer Offline (Verbindung zum Freund wird unterbrochen) - so ist die p2p Verbindung geschlossen und es werden keine Rezepte mehr ausgetauscht.

Was funktioniert noch nicht? Rezepte werden nur temporär im Vuex Store gesichert. Wir haben mit 4 VueJs Plugins experimentiert um die gespeicherten Rezepte auch persistent, also auch nach Schließen der Anwendung, zu sichern, leider ohne Erfolg.

Welche Exit- und Fail-Kriterien sind eingetroffen

- Nutzer können sich ein uns ausloggen und haben dann Zugriff auf ihre Rezepte.
- Nutzer können anderen Peers Rezepte schicken
- Nutzer haben keinen Zugriff auf ihre Rezepte nach schließen der Anwendung
- Geräte lassen sich nicht synchronisieren
- Nutzer können Rezepte drucken
- Nutzer können die Rezepte nicht exportieren und privat sichern.

(Optional:) Welche Gründe gab es für den Misserfolg

Es scheint so als würden die Nutzerobjekte von Firebase als auch die Peer Objekte von PeerJS circular JSONs sein, was dazu führt, dass die Plugins Fehler werfen und auch die JSON Files zu parsen oder zu stringlifyen hat nichts geholfen. Die Erweiterung Flatten hat leider auch keinen Erfolg gebracht.

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

Wie bereits bei der WebRTC PoC Dokumentation beschrieben, ist zwar das Anlegen, temporär speichern sowie der Austausch möglich, jedoch nicht das persistente Speichern der Rezepte auch nach Schließen der Anwendung oder Abmelden des Nutzers. Das Errechnen der Präferenzen der Nutzer funtkioniert, wird alledings auch nicht dauerhaft gespeichert. Wird ein Rezept eines Freundes gespeichert oder man selber fügt ein Rezept seiner Sammlung hinzu so wird die Kategorie, der Schwierigkeitsgrad und die verwendeten Zutaten ausgelesen und das Vorkommen innerhalb des Arrays der Nutzer-Präferenzen inkrementiert. Wird jetzt ein Request nach Vorschlägen ausgeführt, so prüft die App ob ein oder mehrere Rezepte der Freunde/Familie den Präferenzen des Nutzers gerecht wird und gibt dieses unter dem Pfad Explore als Vorschlag aus.

Welche Exit- und Fail-Kriterien sind eingetroffen

- Rezepte können angelegt werden
- Rezepte können (temporär) gespeichert werden
- Rezepte können versendet und empfangen werden
- Rezepte werden ausgewertet
- Präferenzen werden für jeden aktiven Nutzer errechnet
- Die Ausgabe von Vorschlägen funktioniert
- Rezepte lassen sich in voller Größe anzeigen
- Rezepte könnnen gedruckt werden
- Die Ansicht oder Vorschau von Rezepten ist dynamisch und wenn ein Rezept gespeichert oder hinzugefügt wird, aktualisiert sich die gesamte Ansicht der App.

(Optional:) Welche Gründe gab es für den Misserfolg

Die Anwendungslogik ist unserer Meinung nach noch nicht komplex genug um präzise auf die Nutzerpräferenzen einzugehen, allerdings haben wir sie nach besten Gewissen und Erfahrung programmiert und fertiggestellt. Verbesserungsmöglichkeiten sind nicht auszuschließen, sondern eher gewünscht, die Anwendungslogik soll mit der Zeit iteriert werden und sich mit Sammeln der Nutzererfahrungen dynamisch weiterentwickeln.

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

Gibt ein Nutzer Alternative Zutaten beim erstellen eines Rezepts an, so werden diese dem Rezept und der Zutat zugeordnet. Auch bei anderen Familienmitgliedern werden diese dann als Vorschlag angezeigt.

Welche Exit- und Fail-Kriterien sind eingetroffen

- Das Anlegen alternativer Zutaten funktioniert
- Das Ausgeben alternativer Zutaten funktioniert
- Das Anwenden alternativer Zutaten funktioniert nicht

(Optional:) Welche Gründe gab es für den Misserfolg

Zeitmangel, um es kurz zu machen. Sicherlich wäre die Implementierung nicht all zu kompliziert, aber mit unter all den Anderen Features mussten wir Prioritäten setzen und Abstriche machen. Des Weiteren müssen wir uns Gedanken dazu machen wie wir das Anwenden umsetzen wollen, denn ganz simpel ist das mit der Versionierung kombiniert nicht.

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 harmonieren 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

- [FAILED] Das System soll Nutzern die Möglichkeit bieten, Zutaten zu priorisieren (Wichtig für das Rezept).
    - [FAILED] Bei dem angeben einer Zutat sollen Nutzer die Möglichkeit haben diese zu Priorisieren/die Wichtigkeit hervorzuheben.
    - [FAILED] Auch nach der Angabe sollen diese Erweiterungen noch getätigt werden können.
- [ERFOLG] Das System soll Nutzern die Möglichkeit bieten, die Herkunft/Verfügbarkeit über einer Zutat anzugeben/anzuzeigen.
    - [ERFOLG] Bei dem angeben einer Zutat sollen Nutzer die Möglichkeit haben die Herkunft/Verfügbarkeit über einer Zutat anzugeben/anzuzeigen.
    - [FAILED] Auch nach der Angabe sollen diese Erweiterungen noch getätigt werden können.
- [FAILED] Das System soll Nutzern die Möglichkeit bieten, Rezepte offline anzulegen und erst später zu synchronisieren/teilen.
    - [FAILED] Die PWA muss die Möglichkeit haben Rezepte lokal zwischen zu speichern/Cachen und dann nach Bedarf zu teilen.
- [FAILED] Das System soll Nutzern die Möglichkeit bieten, Mengenangaben zu vereinheitlichen und leichter umsetzbar zu machen.
    - [FAILED] Das System muss über die verschiedenen Mengeneinheiten Datensätze haben und die Umrechnung bestenfalls lokal effektiv durchführen.
- [ERFOLG] Das System soll Nutzern die Möglichkeit bieten, benötigte Haushaltsgeräte anzugeben/anzuzeigen.
    - [ERFOLG] Bei dem angeben eines Rezepts sollen Nutzer die Möglichkeit haben benötigte Haushaltsgeräte anzugeben/anzuzeigen.
    - [FAILED] Auch nach der Angabe sollen diese Erweiterungen noch getätigt werden können.
- [ERFOLG] Das System soll Nutzern die Möglichkeit bieten, Rezepte um Kommentare und Tipps sowie Feedback zu erweitern.
    - [FAILED] Das System muss Nutzern an den passenden Stellen die Möglichkeit einer Texteingabe sowie das Teilen dieser anbieten.
- [FAILED] Das System soll Nutzern die Möglichkeit bieten, häufig genutzte Zutaten als "Vorratskammer" anzuzeigen. *[ERFOLG] Das System muss die Zutaten in geteilten Rezepten auslesen und deren Häufigkeit berechnen sowie einen Schwellwert errechnen können.
    - [FAILED] Das System braucht einen Bereich in dem es die Erkenntnisse mit den Nutzern teilen kann.
- [FAILED] Das System soll Nutzern die Möglichkeit bieten, die Farben der PWA selbst zu gestalten (Personalisierung). *[ERFOLG] Die PWA muss in der Lage sein sich selbst zu aktualisieren um die

Benutzer Bedürfnisse adäquat zu stillen.
- [FAILED] Das System soll Nutzern die Möglichkeit bieten, Rezepte übersetzen zu lassen.
    - [FAILED] Das System benötigt die Integration eines zuverlässigen Übersetzers und muss die zu Übersetzende Sprache als auch die Ausgabe Sprache erkennen/übermittelt bekommen.

(Optional:) Welche Gründe gab es für den Misserfolg

Die meisten der Failkriterien kommen auf Grund von Zeitmangel zustande. Die Implementierung der P2P Kommunikation und die scheinbar fehlende Kooperation der Firebase Dienste oder VueJs Plugins hat die Implementierung nicht wirklich erleichtert. Abschließend können wir festhalten, dass dieses Entwicklungsprojekt uns vieles gelehrt hat, aber wir gerne auch mehr noch gemacht und gelernt hätten.

Trotz der nicht vollständigen abarbeitung unserer Featureliste sind wir sehr stolz auf unseren Vertikalen Prototypen, der trotz der brutal langen Implementierungszeiten und vielen Nachtsessions steht und tatsächlich die oben genannten Exit Kriterien erfüllt.

Mit Stolz präsentieren wir "Mein Kochbuch", eine PWA auf Basis von dem dynamischen Framework VueJS, die P2P Kommunikation nutzt um den Rezept austausch zu ermöglichen und modernes Static Website Hosting nutzt um sie für Nutzer erreichbar zu machen, sowie die Sicherheit der Authentifizierung durch etablierte Paradigmen von Google.
