Sinn der App / Vision

Die App soll Nutzerinnen und Nutzern helfen, ihre tägliche Kalorienaufnahme einfach und mobil zu erfassen. Der Fokus liegt auf einer schnellen und intuitiven Bedienung, zum Beispiel durch das Scannen von Barcodes und das automatische Übernehmen von Nährwertdaten.

Ziel ist eine Kalorientracking-App, die vollständig lokal auf dem Gerät funktioniert, kein Benutzerkonto benötigt und alle wichtigen Funktionen für den Alltag bietet.

Zielgruppe

Die App richtet sich an Personen, die ihre Ernährung bewusster kontrollieren möchten. Zum Beispiel:

- Menschen, die Kalorien zählen (z.B. zum Abnehmen oder Gewichtserhalt)
- Sportlich aktive Nutzer, die ihre Ernährung überwachen möchten

Anforderungen

Funktionale Anforderungen

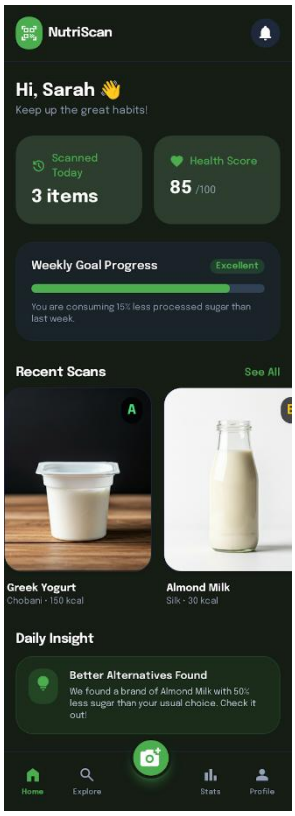
- Lebensmittel können per Barcode gescannt werden
- Lebensmittel können über eine Suche gefunden werden
- Manuelle Eingabe von Lebensmitteln ist möglich
- Tagesübersicht der Nutri-Score
- Lokale Speicherung gescannte Produkte

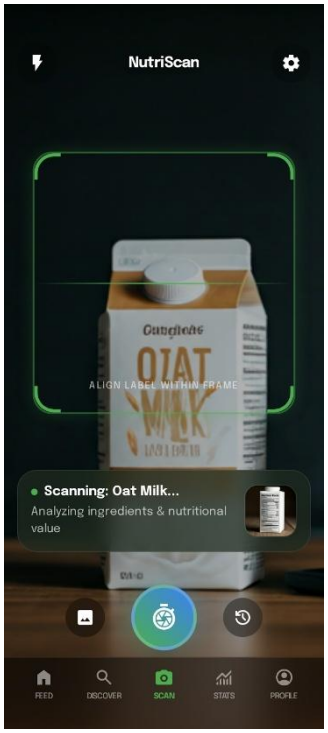
Nicht-funktionale Anforderungen

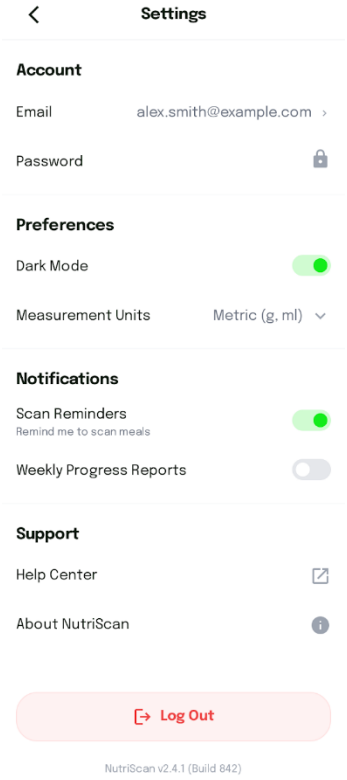
- Hohe Benutzerfreundlichkeit und intuitive Bedienung
- Schnelle Reaktionszeiten durch lokalen Cache
- Plattformunabhängigkeit (Android, optional Web)
- Offline-Nutzung der App

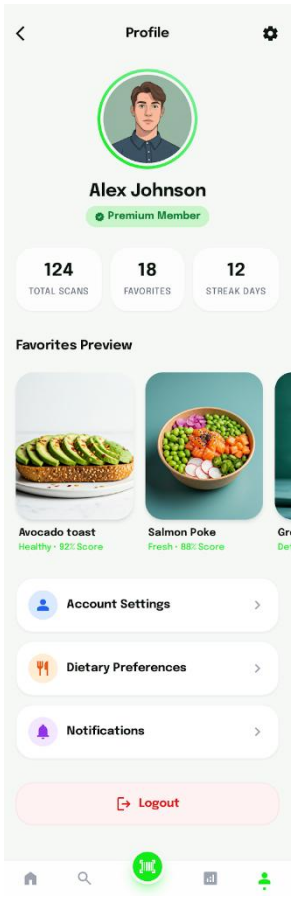
Mockups

Google Stitch:
https://stitch.withgoogle.com/projects/2880159095978237677









Funktionen der App

- Barcode-Scan von Lebensmitteln
- Abruf von Produkt- und Nährwertdaten über die Open Food Facts API
- Anzeige der Nutri-Score
- Lokale Speicherung und Wiederverwendung bereits gescannter Produkte

Architektur der App

Die App verwendet eine MVVM-Architektur (Model-View-ViewModel). Das Frontend ist mit Ionic und Vue umgesetzt und stellt die Benutzeroberfläche dar. Die Views kommunizieren nicht direkt mit der Datenquelle, sondern über ViewModels in Form von Pinia-Stores.

Die Kommunikation mit dem Backend erfolgt indirekt über eine Service- und Repository-Schicht. Diese ist für den Zugriff auf die Open Food Facts API sowie für die lokale Datenspeicherung in IndexedDB zuständig. Dadurch sind API-Zugriffe, Caching und lokale Speicherung klar vom Frontend getrennt.

Die MVVM-Architektur wurde gewählt, da sie eine saubere Trennung von Benutzeroberfläche, Anwendungslogik und Datenzugriff ermöglicht. Sie ist gut für moderne Frontend-Frameworks geeignet, verbessert die Wartbarkeit des Codes und unterstützt eine klare und übersichtliche Projektstruktur.

Verwendete Technologien

Frontend

- Vue 3 als JavaScript-Framework
- Vite für schnelles Development und Build-Prozess
- Vue Router für die Navigation zwischen Seiten
- Pinia für State Management

Backend

- Node.js
- Hono
- REST API over HTTPS

Cloud Database

- PostgreSQL

Mobile-Plattform

- Capacitor zum Verpacken der Vue-App als native Android-App
- Barcode-Scanner-Plugin für den Zugriff auf die Kamera

Datenspeicherung

- (IndexedDB als lokale Hauptdatenbank)
- In-Memory Cache für schnellen Zugriff während der App-Nutzung
  Alle Daten bleiben ausschließlich auf dem Gerät gespeichert.

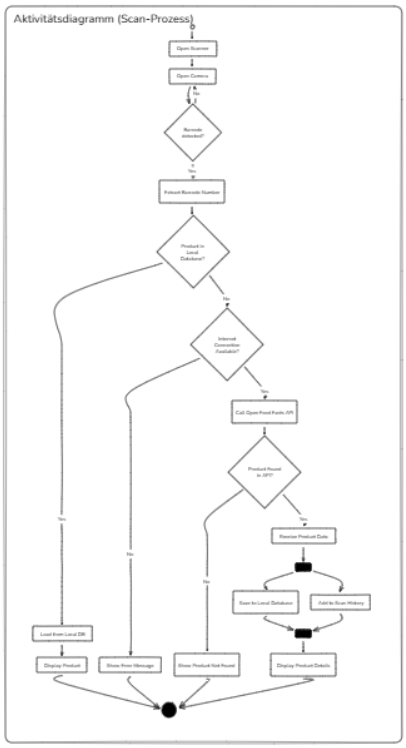
Externe API

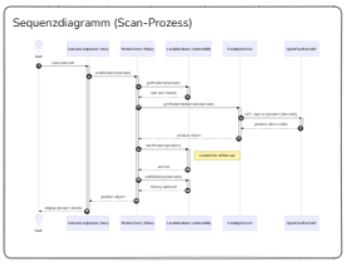
- Open Food Facts API zum Abrufen von Produktinformationen
  Nach dem ersten Abruf werden die Daten lokal gespeichert, um weitere API-Anfragen zu vermeiden.

Diagramme

Excalidraw:

https://excalidraw.com/#room=46ab9c069ca60d1db1b7,wlz9OZJwyORvhbV77RAvmA





Testfall 1: Lebensmittel per Barcode scannen

Vorbedingungen

- App ist installiert und gestartet
- Internetverbindung ist aktiv
- Kamera-Zugriff wurde erlaubt

Vorgehen beim Test

1. App starten
2. Auf den Button „Barcode scannen“ tippen
3. Kamera auf den Barcode eines Lebensmittels richten
4. Warten, bis der Barcode erkannt wird
5. Produktdetails anzeigen lassen
6. Menge in Gramm eingeben
7. Auf „Hinzufügen“ tippen

Erwartetes Resultat

- Das gescannte Produkt wird korrekt erkannt
- Die Nährwertdaten werden angezeigt
- Die Kalorien werden entsprechend der eingegebenen Menge berechnet
- Das Lebensmittel erscheint im Tagesprotokoll

Testfall 2: Profil und Statistiken anzeigen

Vorbedingungen

- App ist installiert und gestartet
- Es sind bereits Kaloriendaten gespeichert

Vorgehen beim Test

1. App starten
2. In der Navigationsleiste auf den Button „Profil“ tippen
3. Profilansicht wird geöffnet
4. Statistikbereich in der Profilansicht anzeigen lassen

Erwartetes Resultat

- Die Profilseite wird korrekt angezeigt
- Die persönlichen Profildaten sind sichtbar
- Die Statistiken (z.B. tägliche Kalorienübersicht) werden korrekt dargestellt
- Die angezeigten Werte entsprechen den gespeicherten Daten