Systemarchitektur

1. Schnittstelle codecheck.info

Codecheck ist sowohl eine mobile Applikation als auch eine Webseite, die es ermöglicht Nährwerte und Inhaltsstoffe eines Produktes anzuzeigen. Die API ist ein wichtiger Bestandteil des zu entwickelnden Systems, da durch dessen Datenbank neben der EAN auch auf weitere Informationen zugegriffen werden kann. Für den geplanten Barcode Scanner ist die EAN von hoher Bedeutung, um die gesamte Anwendungslogik innerhalb des Systems zu realisieren.

2. Firebase Cloud Messaging

FCM ist für die Umsetzung der Benachrichtigungen des Servers an den Client mittels „Push-Notifications“ vorgesehen.
Über das Software Development Kit des Herstellers ist es möglich, Benachrichtigungen basierend auf dem Publish-Subscribe Pattern oder Single Device Messaging, per Token, via http APIs durchzuführen.

3. Google Maps

Für das Anzeigen von den zu vermittelnden Produkten innerhalb des vom Benutzer einzustellenden Umkreises, wird die Google Maps API verwendet. Durch die Benutzung einer Karte wird durch die grafische Darstellung ein besseres Verständnis für die Distanz zwischen den einzelnen Benutzern vermittelt. Außerdem wird das Individualisieren der Kartenelemente ermöglicht.

4. Datenformat JSON

Die Wahl des Datenformats ergab sich aus der Kommunikation zwischen Client und der Schnittstelle. Innerhalb dieser Kommunikation werden die zu beziehenden Daten als JSON-Objekte übermittelt. Des Weiteren würde das System von der Einfachheit und Schlankheit von JSON profitieren, da es nicht nur leicht zu erlernen und gut zu lesen ist, sondern sich auch, im Gegensatz zu XML, die ressourcenschonende Verarbeitung der Daten für die Applikation eignet.

5. Client

Der Client ist in drei Schichten unterteilt: „Datenspeicher“, „Präsentationslogik“ und „Anwendungslogik“. Während die Präsentationslogik lediglich als Benutzungsschnittstelle dient, dient die Anwendungslogik sowohl der Steuerung der Verarbeitung von eingehenden Informationen als auch zur Berechnung und Benachrichtigung des Nutzers. Der Datenspeicher gibt den Nutzern die Möglichkeit ihre bereits im System eingepflegten Daten jederzeit einzusehen, ohne über einen Internetzugriff zu verfügen.
Für den Client wird eine mobile Applikation als Benutzungsschnittstelle genutzt. Für die Entwicklung wird das Betriebssystem Android verwendet, da dies die einzig verfügbare Möglichkeit zurzeit ist.

Kommunikation

Der Client kommuniziert sowohl mit dem Server als auch mit dem API.

Client -> Server

Der Client kontaktiert den zu implementierenden REST-Server via http-Schnittstelle. Diese Kommunikation findet sowohl asynchron als auch synchron statt.

Asynchron

Innerhalb der asynchronen Kommunikation mit dem REST-Server soll basierend auf den eingehenden Informationen, in Form von Produkten innerhalb von Einkaufslisten oder Inventaren, des Benutzers innerhalb des Systems gefiltert und Statistiken erstellt werden. Durch die einzelnen Prozessstufen innerhalb des REST-Servers wird dem Client jeweils eine Benachrichtigung geschickt. Diese Benachrichtigung wird über eine „Push-Notification“ gesendet.

Synchron
Anders als bei der asynchronen Kommunikation wird das Durchlaufen der Prozessstufen so beeinflusst, dass der REST-Server dem Client die gewünschten Informationen sofort zur Verfügung stellt.

Client -> codecheck

Der Client bezieht die Informationen über die Schnittstelle von codecheck. Über die bereitgestellten Informationen verschiedener Produkte werden mittels der EAN, sowohl die Inventarverwaltung als auch die Erstellung von Einkaufslisten realisiert. Des Weiteren ist es ein wichtiger Bestandteil, um eine Erfassung von Produkten über den Barcode Scanner zu ermöglichen.

6. Server

Der Server hat im Wesentlichen die Aufgabe, Anfragen zu bearbeiten. Die Anwendungslogik ist für die automatisierten Suchvorgänge, basierend auf den vom Client eingespeisten Informationen, zuständig.
Für die Umsetzung des REST-Servers wird die Plattform NodeJS genutzt, da NodeJS eine ressourcensparende Architektur bietet, die eine besonders große Anzahl gleichzeitig bestehender Netzwerkverbindungen ermöglicht.
Für das Erstellen von Application Programming Interface, wird das für NodeJS entwickelte Web-Framework Express verwendet. Dies gestaltet das Erstellen einer leistungsfähigen API mithilfe von http-Methoden und Middleware-Funktionen, laut den Informationen der Webseite schnell und einfach.

Kommunikation

Der REST-Server kommuniziert mit mobilen Clients, der Datenbank und dem Firebase Cloud Messaging.

Server -> FCM (Firebase Cloud Messaging)
Um über eine asynchrone Kommunikation dem Client die Ergebnisse der zuvor erwähnten Suchvorgänge mitzuteilen, wird FCM verwendet. Realisiert wird dies über sogenannte „Push-Notifications“. Jedoch benötigt der REST-Server eine Middleware um den Datenaustausch mit dem FCM durchzuführen. Für diese Middleware wird das Node Modul „fcm-node“ vorgesehen.

Server -> Google Maps
Um den Benutzern die grafische Darstellung von den Standorten der zu vermittelnden Produkte zu ermöglichen, werden aus den Resultaten der Suchvorgänge die Koordinaten der Personen entnommen und mittels des Google Maps API sowohl eine Karte zur Verfügung gestellt, als auch die zugehörigen individualisierbaren „Location-Marker“ erstellt.

7. Datenbank

Für die Umsetzung der persistenten Datenspeicherung wird die Benutzung von MySQL vorgesehen. Da aufgrund der von der Schnittstelle bereitgestellten Daten, Daten in Form von JSON Objekten bezogen werden, würde sich eine Umsetzung mittels MongoDB lohnen, jedoch setzt die Datenspeicherung keine eindeutige Identifizierung von Datensätzen voraus, diese müsste im Nachhinein programmiertechnisch gelöst werden. Anhand der EAN könnte eine eindeutige Identifizierung von Datensätzen realisiert werden, welche in MySQL bei der Erstellung von Tabellen, in Form von Primärschlüsseln, vorausgesetzt wird. MySQL bietet eine klare Struktur der Datenbank, die Möglichkeit über JOIN-Operatoren verschiedene Datensätze miteinander zu verknüpfen und den Aspekt, dass MySQL nicht nur über mehr Sicherheit verfügt, sondern auch ausgereifte Lösungen bereitstellt.

8. Anwendungslogik Client

Die Anwendungslogik des Clients dient sowohl der Steuerung der Verarbeitung von eingehenden Informationen als auch zur Berechnung und Benachrichtigung des Clients. Dies beschränkt sich auf die Einkaufslisten, die vom Benutzer zu erstellen sind.
Die Berechnung eines optimalen Einkaufs- und Wegwerfverhaltens wird anhand der zu dokumentierenden Daten durchgeführt. Das System dokumentiert sowohl ausgetragene Lebensmittel aus den Inventaren einzelner Benutzer als auch die Lebensmittel, die über das System an andere Benutzer vermittelt wurden. Für die Berechnung ist eine SQL-Abfrage nötig, um die bereits dokumentierten Informationen beziehen zu können und anschließend den Durchschnitt dieser zu berechnen.
Dadurch wird eine optimale Anzahl pro Lebensmittel berechnet und der Benutzer wird beim Erstellen einer Einkaufsliste dementsprechend hingewiesen. Dem Benutzer ist es selbst überlassen, ob er eine Optimierung vornehmen lassen will oder das Risiko eingeht Lebensmittel zu verschwenden.

9. Anwendungslogik Server

Die Anwendungslogik des Servers ist für die automatischen Suchvorgänge, basierend auf den eingehenden Informationen, zuständig. Diese werden in Form von SQL-Abfragen entweder automatisch oder durch das gezielte Auslösen innerhalb der Anwendung ausgelöst. Die automatische Abfrage erfolgt dadurch, dass der Server in einem bestimmten Intervall die Daten der Benutzer ausliest und diese mit den vermeintlich ablaufenden Lebensmitteln in den Inventaren anderer Benutzer innerhalb des Systems abgleicht. Stimmen die EANs mit den Produkten der Einkaufsliste eines Benutzers überein, wird der Benutzer über die Schnittstelle des FCM mittels „Push-Notification“ benachrichtigt.

10. Verteiltheit der Anwendungslogik

Bei dieser Verteilung der Anwendungslogik beeinflusst das Schonen von Ressourcen die Performance des Clients positiv. Die Anwendungslogik wäre aufgrund der Menge der zu vergleichenden Daten für den Client sehr aufwändig. Des Weiteren müsste der Client für jede Überprüfung mit dem Server kommunizieren und da der Server über eine direkte Verbindung zur Datenbank verfügt, ist es am sinnvollsten diesen Teil der Anwendungslogik auf dem Server durchzulaufen.
Die Berechnung der optimalen Anzahl einzelner Produkte erfolgt zwar über die Kommunikation mit dem Server, diese dient jedoch zur Beziehung der dokumentierten Daten des Benutzers. Die Berechnung an sich, die der Client durchführt, sind vom Ressourcenverbrauch eher gering einzustufen.