

**Bachelorarbeit (Informatik)**

Foodsharing App mit Nährwert­berechnung

|  |  |  |
| --- | --- | --- |
| **Autoren** |  | Betim Kabashi  Julien Wenger |
| **Hauptbetreuung** |  | Beat Seeliger |
| **Datum** |  | 10.06.2022 |

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

**Erklärung betreffend das selbstständige Verfassen einer   
Bachelorarbeit an der School of Engineering**

Mit der Abgabe dieser Bachelorarbeit versichert der/die Studierende, dass er/sie die Arbeit selbständig und ohne fremde Hilfe verfasst hat. (Bei Gruppenarbeiten gelten die Leistungen der übrigen Gruppenmitglieder nicht als fremde Hilfe.)

Der/die unterzeichnende Studierende erklärt, dass alle zitierten Quellen (auch Internetseiten) im Text oder Anhang korrekt nachgewiesen sind, d.h. dass die Bachelorarbeit keine Plagiate enthält, also keine Teile, die teilweise oder vollständig aus einem fremden Text oder einer fremden Arbeit unter Vorgabe der eigenen Urheberschaft bzw. ohne Quellenangabe übernommen worden sind.

Bei Verfehlungen aller Art treten die Paragraphen 39 und 40 (Unredlichkeit und Verfahren bei Unredlichkeit) der ZHAW Prüfungsordnung sowie die Bestimmungen der Disziplinarmassnahmen der Hochschulordnung in Kraft.

**Ort, Datum:** **Name Studierende:**

Zürcher Fachhochschule

**Zusammenfassung**

Hier kommt die Zusammenfassung.

**Abstract**

Hier kommt das Abstract.

**Vorwort**

Hier kommt das Vorwort.

**Inhaltsverzeichnis**

[1. Einleitung 8](#_Toc102651824)

[1.1. Ausgangslage 8](#_Toc102651825)

[1.2. Zielsetzung 8](#_Toc102651826)

[1.3. Aufgabenstellung 8](#_Toc102651827)

[1.3.1. App Funktionen 8](#_Toc102651828)

[1.3.2. Nährwertberechnung 8](#_Toc102651829)

[1.4. Anforderungen 8](#_Toc102651830)

[2. Theoretische Grundlagen 9](#_Toc102651831)

[2.1. Aspekte der Nährwerte 9](#_Toc102651832)

[2.1.1. Kalorien (Energie/Brennwert) 9](#_Toc102651833)

[2.1.2. Fette 10](#_Toc102651834)

[2.1.3. Kohlenhydrate 10](#_Toc102651835)

[2.1.4. Proteine (Eiweiss) 11](#_Toc102651836)

[2.2. Mathematische Grundlage für Kalorienberechnung 11](#_Toc102651837)

[2.2.1. Durchschnittsberechnung anhand von n-Rezepten 12](#_Toc102651838)

[3. Vorgehen / Methoden 13](#_Toc102651839)

[3.1. Datenanalyse und Mahlzeitberechnung 13](#_Toc102651840)

[3.1.1. Rahmenbedingungen 13](#_Toc102651841)

[3.1.2. Datenimport und manuelles Merging 13](#_Toc102651842)

[3.1.3. Berechnung der Kilokalorien (kcal) anhand des Basisdatensatzes 14](#_Toc102651843)

[3.1.4. Genauigkeit der Mahlzeitschätzung 14](#_Toc102651844)

[3.2. Anforderungsanalyse des Prototyps 14](#_Toc102651845)

[3.2.1. Rahmenbedingungen 15](#_Toc102651846)

[3.2.2. Funktionale Anforderungen 15](#_Toc102651847)

[3.2.3. Nicht-funktionale Anforderungen 15](#_Toc102651848)

[3.2.4. User Stories 15](#_Toc102651849)

[3.3. Technische Evaluation Backend 18](#_Toc102651850)

[3.3.1. Einschlusskriterien 18](#_Toc102651851)

[3.3.2. Ansatz 1: Backend-as-a-Service (BaaS) 18](#_Toc102651852)

[3.3.3. Ansatz 2: Platform-as-a-Service (PaaS) 18](#_Toc102651853)

[3.3.4. Entscheidungsmatrix 19](#_Toc102651854)

[3.3.5. Ergebnis Entscheidungsmatrix 20](#_Toc102651855)

[3.4. Realisierung Backend mit Firebase 20](#_Toc102651856)

[3.4.1. Cloud Firestore vs. Realtime Database 21](#_Toc102651857)

[3.4.2. Erstellung der Datenstrukturen mittels Collections und Sub-Collections 21](#_Toc102651858)

[4. Resultate 23](#_Toc102651859)

[5. Diskussion und Ausblick 24](#_Toc102651860)

[6. Verzeichnisse 25](#_Toc102651861)

[6.1. Literaturverzeichnis 25](#_Toc102651862)

[6.2. Glossar 25](#_Toc102651863)

[6.3. Abbildungsverzeichnis 25](#_Toc102651864)

[6.4. Tabellenverzeichnis 25](#_Toc102651865)

[7. Anhang 26](#_Toc102651866)

[7.1. Projektmanagement 26](#_Toc102651867)

# Einleitung

## Ausgangslage

Ausgangslage

## Zielsetzung

Zielsetzung

## Aufgabenstellung

Der Prototyp einer App zum Foodsharing wird implementiert. Neben den zu erwartenden Funktionen wird ein spezieller Fokus auf die Berechnung der Nährwerte von Gerichten gelegt. Hierfür werden Daten gesammelt und mittels geeigneter Modelle ausgewertet. Die Berechnung geschieht nicht wie bei vergleichbaren Apps pro Zutat, sondern anhand der Gerichte.

### App Funktionen

* Rollen: Köche und Konsumenten
* Köche erfassen Gerichte und Zutaten
* Workflow zur Suche und Abholung
* Suche nach Gerichten
* Bewertungsmechanismus
* Evaluation einer geeigneten SW Architektur

### Nährwertberechnung

* Einarbeitung in die theoretischen Grundlagen der Nähwertberechnung
* Einbindung oder Erstellung einer Nährwertdatenbank
* Einbindung oder Erstellung einer Rezeptdatenbank
* Konzept zur Auswertung von Gerichten
* Evaluation einer geeigneten SW Architektur (spezielles Augenmerk auf Wiederverwendbarkeit auch ausserhalb der App)
* Implementierung Auswertungs-Algorithmus
* Geeigneter Mechanismus zur Validierung der Resultate

## Anforderungen

Anforderungen

# Theoretische Grundlagen

In diesem Kapitel werden die grundlegenden Informationen rund um das Thema Nährwertberechnung erläutert.

## Aspekte der Nährwerte

Um eine Nährwertmittelberechnung durchzuführen, muss zuerst eine grundlegende Basis dafür geschaffen und verstanden werden. Für in dieser Arbeit relevanten Informationen werden erst einmal die wichtigsten Nahrungsbestandteile erläutert. Es wird auf die sogenannten „Die Grossen 4“ eingegangen [1]. Unter den Grossen 4 zählen die Energie/ der Brennwert, Fett, Kohlenhydrate und Eiweiss. Dabei handelt es sich um sogenannte Makronährstoffe. Sie bilden den grössten Teil der Nahrungsaufnahme und bilden für die Nährmittelberechnung später die Basis.



Abbildung 1: Nährwertangaben [2]

Unter Abbildung 1 ist ein Beispiel ersichtlich, wie für ein Lebensmittel pro 100 g die Nährwerte abgebildet werden. Neben den gängigen Nährwerten wie Energie, Fett, Kohlenhydrate und Eiweiss werden weitere, wichtige Angaben aufgelistet, die seit der EU-Verordnung 1169/2011 [1] obligatorisch sind.

### Kalorien (Energie/Brennwert)

Kalorien werden als Mass für die Energiemenge verstanden. Die Lebensmittel werden meist in Kalorien und Joule angegeben. 1 Kalorie entspricht dabei etwa 4.2 Joule. Die meiste Energie liefern dabei die drei Makronährstoffe Fette, Kohlenhydrate und Eiweisse. Folglich werden die Energiewerte der drei Makronährstoffe für jeweils 1 Gramm angegeben:

* Fette enthalten pro g 37.8 kJ bzw. 9 kcal
* Kohlenhydrate enthalten pro g 16.8 kJ bzw. 4 kcal
* Eiweisse enthalten pro g 16.8 kJ bzw. 4 kcal

Sowohl die Energiezufuhr als auch der -verbrauch bestimmen das Körpergewicht eines Menschen. Wird dem Körper mehr Energie zugeführt als verbraucht, steigt das Körpergewicht. Wird jedoch weniger Energie zugeführt als verbraucht, so sinkt das Körpergewicht. Der Körper deckt dabei die fehlenden Kalorien aus den Energiespeichern wie beispielsweise aus dem Fettgewebe [3].

### Fette

Fette stellen die energiereichsten Nährstoffe und bedeutende Energieversorgung sowie -speicher dar. Fette werden in gesättigte, einfach ungesättigte und mehrfach ungesättigte Fettsäuren unterschieden. Während gesättigte Fettsäuren vorwiegend in tierische Produkte vorkommen, kommen die einfach und mehrfach ungesättigten Fettsäuren hauptsächlich in pflanzlichen Produkten vor. Des Weiteren sind Fette Träger von Geschmacks- und Aromastoffen sowie zuständig für die Aufnahme von fettlöslichen Vitaminen.

Der menschliche Körper kann überschüssige Energie in der Leber in Fett umwandeln. Dieses wird dann als Depotfett gespeichert. In geringer Menge ist es für den Körper unbedenklich und sogar notwendig für den Hormonhaushalt und die Vitaminaufnahme. In grösserer Menge allerdings kann es für das Herz und den Kreislauf belastend sein. Dies kann zu Übergewicht und zu Stoffwechselerkrankungen führen [4].

### Kohlenhydrate

Kohlenhydrate bezeichnet man auch als Saccharide, was aus dem Griechischen Zucker heisst. Kohlenhydrate setzt sich aus den chemischen Elementen Kohlenstoff, Wasserstoff und Sauerstoff zusammen und werden anhand der Anzahl der Baustoffe in verschiedene Gruppen eingeteilt. Diese Gruppen werden in Einfach-, Zweifach-, Mehrfach- und Vielfachzucker unterschieden. Von Mehrfachzucker spricht man bei einer Anzahl von drei bis neun Zucker-Baustoffe. Beim Vielfachzucker beträgt die Anzahl mindestens zehn Zucker-Baustoffen.

Kohlenhydrate gelten als zentrale Energielieferanten und werden nebst Fett auch als Energiespeicher genutzt. Wie bereits erwähnt, werden Kohlenhydrate in verschiedene Gruppen eingeteilt. Einfach- sowie auch Zweifachzucker begünstigen einen raschen Anstieg des Blutzuckerspiegels. Das bedeutet, dass dem Körper eine grosse Menge Energie für eine kurze Zeit zur Verfügung steht. Allerdings bewirken diese Zucker auch, dass die Energie schnell wieder verbraucht wird. Langkettige und komplexe Kohlenhydrate wie Mehrfach- und Vielfachzucker liefern hingegen kontinuierlich für einen längeren Zeitraum Energie. Das führt dazu, dass der Blutzuckerspiegel langsamer ansteigt und dem Körper längerfristig genügend Energie zur Verfügung steht [4].

### Proteine (Eiweiss)

Eiweisse versorgen den Körper mit wichtigen Aminosäuren und Stickstoff. Dabei wird zwischen 20 verschiedene Aminosäuren unterschieden, die für die körpereigene Proteinsynthese zuständig sind. Die Aminosäure ist der kleinste Baustoff, die für die Zusammensetzung der Proteine zuständig ist.

Bei diesen 20 Aminosäuren wird zwischen entbehrlichen und unentbehrlichen unterschieden. Bei den entbehrlichen Aminosäuren gibt es deren elf und bei den unentbehrlichen neun. Der menschliche Körper kann die unentbehrlichen Aminosäuren nicht selbständig produzieren. Aminosäuren können im menschlichen Körper nicht zwischen­gespeichert werden, um zu einem späteren Zeitpunkt Proteine aufzubauen. Daher ist eine tägliche Zufuhr dieser Aminosäuren nötig, um einem Eiweissmangel zu verhindern.

Die täglich empfohlene Eiweissaufnahme eines gesunden Erwachsenen beträgt in der Regel 0.8 bis 1.0 g/kg Körpergewicht.

Nebst der Qualität spielt auch die Quantität des Eiweisses eine wichtige Rolle. Dabei spricht man von der biologischen Wertigkeit. Die Proteinsynthese wird begrenzt, wenn eine unentbehrliche Aminosäure im Verhältnis zum benötigten Bedarf in einer zu geringen Menge aufgenommen wird. Die Aminosäuren-Zusammensetzung des Menschen ähnelt sich mehr dem tierischen als dem pflanzlichen Eiweiss. Daher weist das tierische Eiweiss in der Qualität eine höhere biologische Wertigkeit auf als das pflanzliche Eiweiss. Da aber das tierische Eiweiss meist in Verbindung mit anderen eher ungesunden Nährstoffen auftaucht, ist es ratsam auch pflanzliche Eiweisse aufzunehmen. Am besten ist eine Kombination aus beidem, um eine hohe biologische Wertigkeit aufzuweisen [4].

## Mathematische Grundlage für Kalorienberechnung

Die Voraussetzung, um diese Berechnung genau durchzuführen, ist die Mengenangabe jeder einzelnen Zutat in einem Gericht zu kennen. Dies führt zum Problem, dass ein Koch während der Zubereitung jede Zutat abwiegen oder sich strikt an die Mengen im Rezept halten muss. Um diesem Problem entgegenzuwirken, wird im nächsten Kapitel ein Ansatz vorgestellt, bei dem die Berechnung der Kalorienanzahl auf Schätz- und Mittelwerte basieren und auf das fertig gekochte Gericht angewendet werden kann.

### Nährwertveränderungen durch Kochen

„Der Nährstoffgehalt eines gekochten Lebensmittels oder zusammengesetzten Gerichtes kann entweder analysiert oder berechnet werden. Analysieren hat den Vorteil, dass es genaue Werte liefert. Analysen sind jedoch anspruchsvoll, aufwändig und teuer. In der Praxis wird der Nährstoffgehalt eines zubereiteten Produktes deshalb meist mit Hilfe von durchschnittlichen Verlust- und Aufnahmefaktoren berechnet. Dies gilt auch für die Schweizer Nährwertdatenbank. Beim Berechnen des Nährstoffgehaltes eines zubereiteten Lebensmittels müssen zusätzlich zu den Nährstoffverlusten auch allfällige Gewichtsveränderungen berücksichtigt werden. Nimmt das Gewicht ab, entspricht dies einer Konzentration. Der Gehalt an Vitaminen und Mineralstoffen pro 100 g eines gekochten Lebensmittels kann deshalb trotz Nährstoffverlusten höher ausfallen als jener des entsprechenden rohen Produktes. Umgekehrt entspricht eine Gewichtszunahme (z.B. beim Kochen von Teigwaren) einer Verdünnung und der Nährstoffgehalt pro 100g des gekochten Lebensmittels ist deutlich geringer als jener des ungekochten trockenen Lebensmittels“ [5].

### Durchschnittsberechnung anhand von n Rezepten

Da sich die geschätzte Nährwertangabe auf ein gekochtes Gericht beziehen soll und sich das Gewicht von Zutaten nach dem Kochprozess erhöhen (Aufnahme von Flüssigkeit) oder verringern (Verlust von Flüssigkeit) kann, muss der Änderungsfaktor hinsichtlich Gewichts für jede Zutat bekannt sein. Ist der Änderungsfaktor einer Zutat grösser als Eins, bedeutet dies eine Gewichtsabnahme während dem Kochprozess, hingegen ein Änderungsfaktor kleiner als Eins eine Gewichtszunahme bedeutet. Mit diesem Änderungsfaktor werden die Kalorienangaben in den n Rezepten angepasst. Im zweiten Schritt werden die Rezepte vergleichbar gemacht, indem die Gesamtmenge auf eine Basis (auf 100g) umgerechnet wird. Als nächstes wird die Durchschnittsmenge für jede Zutat über die n Rezepte berechnet. Somit wird ein weiteres Rezept erzeugt, welches Durchschnittsmengenangaben enthält, von dem dann die Kalorien berechnet werden können. In Kapitel 3 wird die Berechnung, die Genauigkeit der Rechnung und die Realisierung dieser Idee im Rahmen vom Prototyp genauer beschrieben.

# Vorgehen / Methoden

Das Vorgehen wird in zwei Teilbereiche aufgeteilt. Der eine Teilbereich beinhaltet das Vorgehen, die Konstruktion und Realisierung des Prototyps einer Smartphone-Applikation. Der andere Teilbereich befasst sich mit der Nährwertberechnung im Detail.

## Datenaufbereitung und Mahlzeitberechnung

Wie in den theoretischen Grundlagen angedeutet, wird in diesem Abschnitt die konkrete Realisierung der Mahlzeitberechnung beschrieben.

### Rahmenbedingungen

Da es keine vorgefertigten Datensätze gibt und ein ETL Prozess (Extraktion, Transformation, Laden) nicht Teil der Arbeit ist, wird der initiale Datenimport der Gerichte manuell durchgeführt. Das Zusammenführen der Rezepte und die Durchschnittsberechnung wird mittels JavaScript Abfragen auf die Datenbank durchgeführt. Aus diesem Grund enthält der Prototyp initial nur 10 vordefinierte Rezepte.

### Datenaufbereitung

**Zutaten und Nährwerte**

Für die Zutaten und deren Nährwerte wird die Schweizer Nährwertdatenbank vom Bundesamt für Lebensmittelsicherheit und Veterinärwesen verwendet. Diese Datensammlung enthält 129 Felder je Zutat, wobei im Rahmen dieser Arbeit nur die nötigen Nährwertangaben, nämlich Kilokalorienanzahl, Protein-, Fett und Kohlenhydratmenge, importiert werden. Ausserdem wird in der Datensammlung für ein Grossteil der Zutaten zwischen roh und zubereitet (gekocht, gebraten, geschmort) unterschieden. Durch diese Unterscheidung kann der Änderungsfaktor, wie in Kapitel 2.2 angeschnitten wurde, berechnet werden, indem die Kalorienanzahl der zubereiteten Zutat durch die Kalorienanzahl der rohen Zutat geteilt wird. In der finalen Tabelle wird diese Unterscheidung aufgehoben und nur der Datensatz mit den rohen Angaben plus dem Änderungsfaktor verwendet. Beispiel:

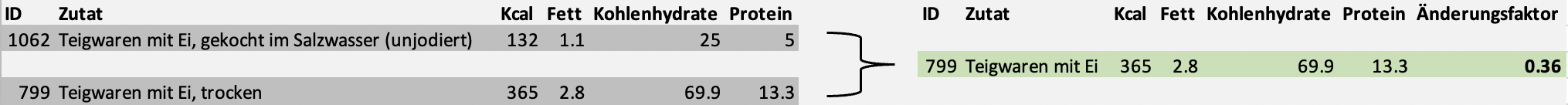


Abbildung 2: Zutat roh/gekocht, Änderungsfaktor

Wie in Abbildung 2 in grün zusätzlich zu erkennen ist, entfällt im Namen die Zusatzinformation der Zubereitung, da bei Rezeptangaben immer vom Rohzustand ausgegangen wird und die Unterscheidung zwischen roh und zubereitet in der finalen Tabelle nicht mehr existiert.

**Rezepte**

Mit Hilfe des Webs werden für jedes Gericht zwei bis drei Rezepte aus verschiedenen Quellen gesucht. Im zweiten Schritt werden Grössen, wie EL(Esslöffel), TL(Teelöffel) oder ähnliche Mengenangaben, mit Hilfe einer Masseinheit Tabelle von chefkoch.de in Gramm umgewandelt. Um die Rezepte vergleichbar zu machen, wird die Gesamtmenge auf die gleiche Basis (100g) umgerechnet. Dabei werden Zutaten, wie Gewürze, Kräuter und kalorienarme Flüssigkeiten in der Berechnung ignoriert, da deren Kaloriengehalt nur wenig bis gar keine Auswirkung auf das Resultat haben. Zusätzlich werden Zutaten für ein Gericht, im finalen Datensatz, als optional markiert, wenn diese nicht in der Mehrheit der Rezepte vorkommen. Optionale Zutaten sind standardmässig nicht Teil vom Rezept, können im Prototyp jedoch angewählt werden. Zum Schluss wird der Mittelwert für jede Zutat über alle verfügbaren Rezepte berechnet. Das Ergebnis ist ein neuer Datensatz bzw. Rezept, welches Durchschnittswerte der Mengen enthält und die Basis für die Kalorienberechnung für gekochte Gerichte darstellt. Im nächsten Kapitel wird auf die Berechnung und deren Genauigkeit gegenüber der effektiven Kalorienanzahl eingegangen.

In der folgenden Grafik ist der beschriebene Prozess an einem Beispiel für «Ghackets mit Hörnli» veranschaulicht.

Ein Bild, das Tisch enthält.

Automatisch generierte Beschreibung

Abbildung 3: Beispiel Rezept Merging

### Berechnung der Kilokalorien (kcal) anhand des Basisdatensatzes

### Genauigkeit der Mahlzeitschätzung

## Anforderungsanalyse des Prototyps

In diesem Kapitel werden die Rahmenbedingungen, die User Stories des Prototyps festgelegt und beschrieben.

### Rahmenbedingungen

|  |  |
| --- | --- |
| Entwickler | 2 |
| Zielgruppe | Privatpersonen, die Gerichte zur Verfügung stellen möchten oder aber auch nach verfügbaren Gerichten suchen und bestellen wollen. |
| Endgeräte | Alle gängigen Smartphones ab 2021.  Der Prototyp soll zuverlässig und stabil auf allen Android-Geräten laufen mit mindestens der Android-Version 9.0 [Stand: 6. August 2018].  Der Prototyp soll zuverlässig und stabil auf allen iOS-Geräten laufen mit mindestens der iOS-Version 15.1 [Stand: 25. Oktober 2021]. |
| Software | Für die Umsetzung des Frontends wird React Native eingesetzt. |

Tabelle 1: Rahmenbedingungen (Prototyp)

### User Stories

|  |  |
| --- | --- |
| Titel | Anzeige der Gerichte |
| User Story ID | US-1 |
| Beschreibung | Als Konsument möchte ich unter der Rubrik «Abholen» zufällige Gerichte sehen können, damit ich eines aussuchen kann. |
| Akzeptanzkriterien | * Folgende Informationen werden angezeigt:   + Das Bild des Gerichts   + Der Name des Gerichts   + Der Verfasser des Gerichts   + Die Entfernung   + Wann das Gericht abholbereit ist.   + Das Profilbild des Verfassers * Der Konsument kann auf ein Gericht klicken, um die Detailansicht eines Gerichts einsehen zu können. |

Tabelle 2: User Story 1 - Anzeige der Gerichte

|  |  |
| --- | --- |
| Titel | Detailansicht des Gerichts |
| User Story ID | US-2 |
| Beschreibung | Als Konsument möchte ich eine Detailansicht eines Gerichts einsehen können, damit ich detailliertere Angaben zum Gericht erhalte. |
| Akzeptanzkriterien | * Folgende Informationen werden angezeigt:   + Das Bild des Gerichts   + Der Name des Gerichts   + Der Verfasser des Gerichts   + Das Profilbild des Verfassers   + Wann das Gericht abholbereit ist.   + Zutaten   + Die berechneten Nährwerte * In der Detailansicht hat der Konsument die Möglichkeit, das Gericht zu bestellen. |

Tabelle 3: User Story 2 - Detailansicht des Gerichts

|  |  |
| --- | --- |
| Titel | Bestellvorgang |
| User Story ID | US-3 |
| Beschreibung | Als Konsument möchte ich ein Gericht bestellen können, damit ich es abholen kann. |
| Akzeptanzkriterien | * Folgende Informationen werden angezeigt:   + Das Bild des Gerichts   + Der Name des Gerichts   + Wann das Gericht abholbereit ist. * Nach dem Bestellvorgang kann der Konsument das Gericht schliesslich abholen. |

Tabelle 4: User Story 3 - Bestellvorgang

|  |  |
| --- | --- |
| Titel | Gerichte suchen |
| User Story ID | US-4 |
| Beschreibung | Als Persona möchte ich nach Gerichten suchen können, damit ich mein Wunschgericht finden kann. |
| Akzeptanzkriterien | * Es soll eine Eingabemaske geben, bei der der Konsument mittels Volltextsuche nach Gerichten suchen kann. * Es besteht die Möglichkeit, nach dem Namen eines Gerichts oder nach Gerichten zu suchen, die eine bestimmte Zutat beinhalten. * Bei der Suche sollen die Gerichte aufgelistet werden, nach der der Konsument gesucht hat. |

Tabelle 5: User Story 4 - Gerichte suchen

|  |  |
| --- | --- |
| Titel | Auflistung der bereits getätigten Bestellungen |
| User Story ID | US-5 |
| Beschreibung | Als Konsument möchte ich eine Auflistung meiner bereits getätigten Bestellungen einsehen können, damit ich eine Übersicht habe, was ich schon bestellt habe. |
| Akzeptanzkriterien | * Es soll eine Auflistung aller bereits getätigten Bestellungen angezeigt werden können. * Bei den bereits bestellten Gerichten werden folgende Informationen angezeigt:   + Das Bild des Gerichts   + Der Name des Gerichts   + Der Verfasser des Gerichts   + Das Profilbild des Verfassers |

Tabelle 6: User Story 5 - Auflistung der bereits getätigten Bestellungen

|  |  |
| --- | --- |
| Titel | Profilanzeige |
| User Story ID | US-6 |
| Beschreibung | Als Persona möchte ich mein Profil aufrufen können, damit ich auf meine persönlichen Daten zugreifen kann. |
| Akzeptanzkriterien | * Die Persona Koch und Konsument können auf ihr Profil zugreifen. * Es werden die wichtigsten persönlichen Daten angezeigt. * Der Koch kann zudem einsehen, wie gut seine Bewertung ist. |

Tabelle 7: User Story 6 - Profilanzeige

|  |  |
| --- | --- |
| Titel | Gerichte anbieten |
| User Story ID | US-7 |
| Beschreibung | Als Koch möchte ich ein Gericht anbieten können, damit Konsumenten die Möglichkeit haben, das Gericht bestellen zu können. |
| Akzeptanzkriterien | * Der Koch kann von den bereits verfügbaren Gerichten das passende auswählen und anbieten. |

Tabelle 8: User Story 7 - Gerichte anbieten

|  |  |
| --- | --- |
| Titel | Gerichte erfassen |
| User Story ID | US-8 |
| Beschreibung | Als Koch möchte ich ein Gericht erfassen können, das es in der Datenbank noch gibt, damit ich ein eigenes Gericht anbieten kann. |
| Akzeptanzkriterien | * Der Koch kann dem zu erfassenden Gericht einen Namen geben. * Der Koch kann anhand der Datenbank die jeweilige Zutat, deren Menge und die Masseinheit auswählen. |

Tabelle 9: User Story 8 - Gerichte erfassen

|  |  |
| --- | --- |
| Titel | Koch bewerten |
| User Story ID | US-9 |
| Beschreibung | Als Konsument möchte ich nach dem Bestellvorgang den jeweiligen Koch des Gerichts bewerten können, damit andere sehen können, wie gut der Koch ist. |
| Akzeptanzkriterien | * Der Konsument kann den Koch nach dem Bestellvorgang bewerten. |

Tabelle 10: User Story 9 - Koch bewerten

## Eingesetzte Software Frontend

Für die Realisierung des Prototyps seitens Frontend wird React Native eingesetzt.

Der grosse Vorteil besteht darin, dass ein Framework für mehrere Plattformen verwendet werden kann. So muss kein nativer Code für Android oder iOS einzeln programmiert werden. Eine Code-Basis reicht aus, um für diese Plattformen, sowie auch für Webanwendungen zu entwickeln. React Native bietet mittlerweile viele UI-Komponenten und Funktionen für die Entwicklung von nativen Anwendungen an, sodass gängige Prozesse in einer App abgedeckt werden können. Dadurch kann ein hoher Grad an Wiederverwendbarkeit erreicht werden [6].

Für die Entwicklung des Prototyps wird Node.js lokal installiert und react-native als Node-Modul installiert. Hinzu wird Expo verwendet, dass als Webdienst dient, um den Prototyp entweder als Webanwendung, in einem iOS- oder in einem Android-Emulator aufrufen zu können. Optional kann man Expo auch als iOS- oder Android-App installieren, um so den Prototyp auf ein physisches Endgerät anzeigen zu lassen.

### Deployment und Hosting

Für das Deployment und Hosting des Prototyps ist mittels der Hosting-Plattform Firebase gedacht, das im Kapitel 3.4.2. genauer erläutert wird.

Folgende Schritte müssen durchgeführt werden, damit der Prototyp deployed und somit gehostet werden kann:

1. Bevor ein Firebase-Hosting eingerichtet werden kann, muss ein Firebase-Projekt erstellt werden. Für die Umsetzung des Prototyps ist ein Firebase-Konto sowie ein Firebase-Projekt erstellt worden.
2. Als nächstes muss Firebase CLI installiert werden, das verschiedene Werkzeuge für die Verwaltung und das Deployment zur Verfügung stellt.
3. Sobald Firebase CLI installiert ist, muss man sich authentifizieren können, sodass man Zugriff auf die eigenen Firebase-Projekte erhält.
4. Um die lokalen Projektdateien mit dem Firebase-Projekt zu verbinden, muss folgender Befehl ausgeführt werden:

$ firebase init hosting

1. Während dem Initialisieren des Firebase-Projekts, müssen folgende Punkte durchgeführt werden:
   1. Das Firebase-Projekt auswählen, dass mit dem lokalen Projekt verbunden werden soll.
   2. Anschliessend soll ein Verzeichnis für das public root directory ausgewählt werden.
   3. Am Schluss der Initialisierung erstellt Firebase automatisch zwei Dateien im root directory:

firebase.json

.firebaserc

## Technische Evaluation Backend

In diesem Kapitel werden für die Realisierung des Backends zwei moderne Ansätze verglichen und evaluiert. Das Resultat der Evaluation sollte ein klares Bild verschaffen, welches Framework oder welche Technologie für das Projekt bzw. den Prototyp am geeignetsten ist. Aus diesem Grund werden für beide Ansätze zwei bis drei konkrete Technologien und/oder Frameworks in die Evaluation mit einbezogen. Das wichtigste ist hierbei, ob und wie gut die ausgewählten Lösungen unsere nicht-funktionalen Anforderungen erfüllen. Wir beschränken uns bei der folgenden Evaluation nur auf Cloud-basierte Lösungen, da es sich bei der App um einen Prototyp handelt und bei einer On-Premise Lösung der initiale Aufwand zu gross wäre.

### Einschlusskriterien

In der Evaluation werden nur Technologien evaluiert, die folgende Kriterien erfüllen:

* Backend-App auf NodeJS (Gilt nur für Ansatz 2)
* Volltextsuche muss möglich sein
* DBMS Modell entweder Document Store oder Relationale DB
* Einfache Verbindung von DBMS und NodeJS (Gilt nur für Ansatz 2)

### Ansatz 1: Backend-as-a-Service (BaaS)

Als Backend-as-a-Service werden Produkte bezeichnet, bei dem der Anbieter vollständige Backend-Lösungen (inklusive Datenbank) zur Verfügung stellt. Der Konsument kümmert sich im Besten Fall nur noch um die Datenmodelle oder die Anbindung ans UI. Aufgrund der Tatsache, dass sich die Anbieter, auf die nicht-funktionalen Anforderungen bezogen, kaum unterscheiden, wird nur der bekannteste Anbieter in die Evaluation miteinbezogen.

**Kandidaten**

**Google Firebase**

Firebase ist eine Entwicklungs- und Hostingplatform spezialisiert auf Mobile- und Webanwendungen. Es verfügt über eine Realtime Dokumentdatenbank und hat zusätzlich noch weitere nützliche Backendfeatures, wie zum Beispiel User Authentifizierung sowie Monitoring- und Analyse Tools. Ein Nachteil der Firebase Datenbank ist die fehlende Volltextsuchfunktionalität, diese muss mit externen Suchanbieter realisiert werden. Firebase gehört zu den bekanntesten Backend-as-a-Service Anbieter.

### Ansatz 2: Platform-as-a-Service (PaaS)

Als Product-as-a-Service werden Produkte bezeichnet, die es Konsumenten ermöglichen, mit wenig Aufwand, ihre Applikation (inklusive Datenbank) auf konfigurierbare und wartbare Cloud Umgebungen zu verwalten. Dies hat den Vorteil, dass sich der Konsument nur auf die Entwicklung der Applikation fokussieren und infrastrukturbedingte Fragestellungen dem Anbieter überlassen kann. In diesem Segment gibt es sehr viele Anbieter. Im Wesentlichen unterscheiden sie sich in der Kompatibilität von Frameworks/Technologien und in der Anzahl von zusätzlichen Middleware Tools für die Wartung oder Analyse der Applikation oder der Daten. Aus diesem Grund ist es in diesem Ansatz primär wichtiger, den Fokus auf die Auswahl der Frameworks bzw. Technologien zu legen anstatt der Auswahl des PaaS Anbieters. Hierbei kommen nur Technologien in Frage, die mit dem Betreuer besprochen wurden und die Einschlusskriterien erfüllen.

**Kandidaten**

**ExpressJS + PostgreSQL auf Heroku**

ExpressJS ist ein NodeJS Framework für die Entwicklung von Backend Schnittstellen. Heroku gehört zur PaaS Kategorie, welches unter Anderem NodeJS unterstützt und es dem Entwickler mit geringem Aufwand ermöglicht, seine Applikation auf der Cloud zu bauen, hosten und zu verwalten. PostgreSQL ist im Gegensatz zu Firebase-DB oder MongoDB eine relationale Datenbank, die aber Volltextsuche, durch SQL-Funktionen, anbietet. CloudSQL ist ein PaaS von Google, die relationale Datenbanksysteme, wie MySQL, PostgreSQL und SQL-Server, vollständig verwalten kann.

**Apollo Server + PostgreSQL Heroku**

Einzigartig an dieser Lösung sind die Backendschnittstellen. Anders als bei ExpressJS oder anderen üblichen Frameworks werden hier mittels Apollo Server Framework GraphQL Schnittstellen zur Verfügung gestellt. GraphQLs grosse Stärke gegenüber REST ist, dass Schnittstellen anpassbar sind. Das heisst, während REST für einen Endpunkt immer die gleiche Struktur an Daten zurückliefert, kann man mittels GraphQL während der Abfrage definieren, welche Felder man wirklich braucht. Somit kann eine GraphQL Schnittstelle für mehrere Fälle benutzt werden.

**NodeJS + Elasticsearch auf Heroku + MongoDBAtlas**

Dieses Setup hat, verglichen mit den anderen Kandidaten, mit Elasticsearch Cloud die leistungsstärkste Volltextsuche. Zusätzlich bietet es Analyse und Visualisierungstools an. Es ist bekannt, dass sich Elasticsearch nicht als primäre Datenbank für die Speicherung von User- oder sonstigen statischen Daten eignet, deshalb braucht dieser Ansatz ein zusätzliches Speichermedium, nämlich PostgreSQL.

### Entscheidungsmatrix

In der folgenden Tabelle ist die Auswertung der Kandidaten zu sehen. Die Kriterien und Gewichtung in der Entscheidungsmatrix haben sich aus der Anforderungsanalyse herauskristallisiert und beschreiben folgendes:

* Setup: Wie einfach und aufwendig ist das initiale Setup des Backends? Wie schnell kann man mit der eigentlichen Entwicklung starten?
* Out-of-the-Box Volltextsuche: Enthält die Lösung eine Volltextsuchfunktionalität oder müssen zusätzliche Erweiterungen aufgesetzt und konfiguriert werden?
* Verwendet bekannte Protokolle und Sprache: Ist die Technologie hinter der Lösung bekannt? Werden Sprachen oder Protokolle verwendet, die neu sind und noch gelernt werden müssen?
* Einfache DB-Anbindung ans NodeJS-Backend: Gibt es standardisierte Guides für die Anbindung? Unterstützt das Framework die gewählte DB?
* Tools für Datenanalyse: Welche Analytik-Tools bietet die Lösung an? Können diese ohne grossen Aufwand genutzt werden oder braucht es zusätzliche Konfigurationen/ Installationen
* Wartbarkeit: Angebot an: Deploymentmöglichkeiten, Automatisierungen, Aufsetzen von Test-/Produktionsumgebungen.
* Vorhandenes Know-How: Vorhandene Erfahrungen mit den Technologien.

Ein Bild, das Tisch enthält.

Automatisch generierte Beschreibung

Abbildung 4: Entscheidungsmatrix Backend

### Ergebnis Entscheidungsmatrix

Wie in der Entscheidungsmatrix zu sehen, ist Firebase mit 29.9 Punkten die geeignetste Lösung für den Prototyp. Grundsätzlich kann man sagen, dass jede der Kandidaten genutzt werden kann, um den Prototyp zu realisieren, jedoch hat Firebase den grossen Vorteil, dass die Datenbank integriert ist und keine zusätzlichen Installationen bzw. Anbindungen notwendig sind. Die Evaluation hat aufgezeigt, dass es keine «perfekte» Backendlösung gibt. Vielmehr stellt sich die Frage, welche Strategie verfolgt werden soll. Bei Backend-as-a-Service Ansätzen kann die Entwicklung abhängig vom Lieferanten (in diesem Fall Google) sein. Dadurch können Engpässe entstehen, die die Realisation eines Produktes verzögern können. Auf der anderen Seite haben Eigenbaulösungen den grossen Nachteil, dass der initiale Aufwand grösser ist, jedoch ist die Abhängigkeit zu Lieferanten nicht gegeben und gewinnt so an Flexibilität.

## Realisierung Backend mit Firebase

In diesem Kapitel wird die Realisierung des Backends mittels Firebase beschrieben. Dazu zählen: Aufsetzen des Projekts, Erstellung der APIs und die Definition sowie Speicherung der Datenstrukturen.

### Cloud Firestore vs. Realtime Database

Google Firebase bietet zwei Arten von Datenbanken an, nämlich «Cloud Firestore» und «Realtime Database». «Cloud Firestore» ist die neuere Variante, die auf die "Realtime Database" aufbaut und zusätzliche Funktionen anbietet. Der essenzielle Unterschied ist die Speicherung. Während «Realtime Database» die Daten nur in reinem JSON Format abspeichert kann, bietet «Cloud Firestore» die Möglichkeit, die Daten hierarchisch zu strukturieren. Da man Gerichte und deren Zutaten sowie Nährwerte sinnvoll hierarchisch aufbauen kann, wird für dieses Projekt «Cloud Firestore» verwendet.

### Erstellung der Datenstrukturen mittels Collections und Sub-Collections

Der Aufbau und die Anordnung der einzelnen Dokumente bzw. der Daten wird hier als Datenstruktur bezeichnet. Dabei gilt die Regel, dass das Backend nur die Daten liefern soll, die für die Darstellung oder Funktion im GUI notwendig ist. Man spricht auch von einer schlanken API, bei der die Datenmenge, die über die REST API geschickt wird, so klein wie möglich gehalten werden soll. Um dieses Ziel zu erreichen, bietet Cloud Firestore Sub-Collections an. Hiermit können die Daten hierarchisch angeordnet werden, mit der Möglichkeit nur einzelne Elemente in der Hierarchie anzusprechen, ohne die darunterliegenden Elemente einzubeziehen. Am einfachsten lässt sich dies am folgenden Beispiel erklären:

Die Struktur eines Rezeptes bzw. Gericht kann folgendermassen aussehen:

Ein Bild, das Tisch enthält.

Automatisch generierte Beschreibung

Abbildung 5: Gliederung Rezept Daten

Anhand der Abbildung oben kann eine Hierarchie erkannt werden. Ein Rezept (recipe) hat mehrere Zutaten (ingredients). Eine Zutat hat die Nährwerte: Kohlenhydrate (carbs), Protein (protein) und Fett (fat). Bei einer üblichen Datenspeicherung ohne Sub-Collections gibt es keine Möglichkeit nur Teilobjekte über die API zu schicken, sondern es muss das ganze Objekt bzw. Dokument Rezept (recipe) zurückgeliefert werden. Wenn aber die Zutat (ingredients) als Sub-Collection definiert wird, ist es möglich nur die Zutaten (ingredients) oder nur das Rezept (recipe) ohne Zutaten (ingredients) über die API zu transferieren. Ein weiteres Beispiel wäre die Bewertungen eines Koches. Ein Koch ist in der Datenbank als User Dokument abgelegt. Ohne Sub-Collection wären die Bewertungen des Kochs Teil des User Dokumentes und werden somit bei jeder Anfrage mitgeliefert, auch wenn die Bewertungen im GUI für gewisse Use Cases gar nicht gebraucht werden. Mit Hilfe von Sub-Collections kann die mobile Datennutzung auf Endgeräten optimiert werden.

# Resultate

Resultate

# Diskussion und Ausblick

Diskussion und Ausblick

## Was noch für ein GoLive fehlt

Um den Prototyp zu hosten und somit für die gängige Zielgruppe zur Verfügung zu stellen, müssen allerdings noch rechtliche Aspekte berücksichtigt werden, die folgend beschrieben werden.

### Hygienemassnahmen und Allergiehinweise

Da es sich beim Prototyp um Hobbyköche und nicht um Gastronomiegewerbe, Hotellerie oder Restaurant handelt, stellt sich die Frage, wie Köche, die gelegentlich kochen und Gerichte verkaufen, gehandhabt werden. Dies müsste abgeklärt werden.

Es heisst, Hobbyköche dürfen Gerichte kochen und diese verkaufen, aber sie müssen die Hygienemassnahmen ebenso beachten. Solange der Verkauf von Gerichten nicht zur Regelmässigkeit wird, untersteht ein Hobbykoch nicht der Kontrolle des Lebensmittelinspektorats [7].

Im Prototyp müsste auf die Regelung der Hygienemassnahmen hingewiesen werden. Bei den Allergiehinweise wird es etwas schwieriger, wenn für jedes Gericht einzeln auf die Allergene hingewiesen werden soll. Ebenso müsste die Haftung geregelt werden, falls ein Konsument anhand eines gekauften Gerichts gesundheitliche Beschwerden erfolgen könnten.

## Ausbaumöglichkeiten

Dieser Prototyp kann beliebig erweitert und optimiert werden. Folgend werden einige mögliche Erweiterungen erläutert.

### Chatsystem

Damit der Konsument und der Koch nach einer Bestellung direkt kommunizieren können, kann im Prototyp ein eigenes Chatsystem implementiert werden, ohne Drittanwendungen nutzen zu müssen. Dabei können sich die Benutzer darauf einigen, wie und wann das Gericht abgeholt werden soll. Ohne private Daten preisgeben zu müssen, können die Benutzer das integrierte Chatsystem kommunizieren.

### Zugeschnittene Angebote

Künftig wäre denkbar, dass mittels eines Algorithmus das Verhalten des Konsumenten ermittelt wird. Dadurch wird festgestellt, welche Gerichte der Konsument häufiger und welche seltener oder gar nicht bestellt hat. So können dem Konsumenten Vorschläge angeboten werden, die ähnlich sind, wie diejenigen Gerichte, die der Konsument schon von früher bestellt hat.

So macht es Sinn, dass beispielsweise ein Vegetarier dann Gerichte vorgeschlagen bekommt, die kein Fleisch enthalten.

### Kartenfunktion

Eine weitere, denkbare Implementation wäre eine integrierte Kartenansicht mit Routenberechnungen zwischen Konsumenten und Koch. So kann direkt abgelesen werden, wie lang und wohin der Konsument gehen muss, um das Gericht abholen zu können.

### Zahlungssystem

Für die Bezahlung des Gerichts kann auch ein integriertes Zahlungssystem implementiert werden. Dadurch kann die Zahlungsabwicklung direkt im Prototyp erfolgen.

# Verzeichnisse

## Literaturverzeichnis

|  |  |
| --- | --- |
| [1] | S. Täuber, «Neues zur Kennzeichnung von Lebensmitteln,» Oktober 2013. [Online]. Available: https://www.ufag-laboratorien.ch/fileadmin/Content/05\_Lebensmittel/Lebensmittel\_Produktanalysen%20und%20Naehrwerte/UFAG\_Kennzeichnung\_von\_Lebensmitteln\_Lebensmittel-Technologie\_10-2013.PDF. [Zugriff am 2 April 2022]. |
| [2] | M. Blättler, «Nährwerttabelle: Darauf sollte man beim Kauf von Lebensmitteln achten,» [Online]. Available: https://gymperformance.ch/naehrwerttabelle-darauf-sollte-man-beim-kauf-von-lebensmitteln-achten/. [Zugriff am 2 April 2022]. |
| [3] | K. Schmidt-Prange, «So viele Kalorien brauchst du täglich,» 9 November 2021. [Online]. Available: https://www.menshealth.de/gesunde-ernaehrung/so-viele-kalorien-verbrauchst-du-taeglich/. [Zugriff am 3 April 2022]. |
| [4] | DEBInet, «Ernährungsinformationen - Nahrungsbestandteile,» [Online]. Available: https://www.ernaehrung.de/tipps/allgemeine\_infos/ernaehr11.php. [Zugriff am 3 April 2022]. |
| [5] | BLV, «Nährwertveränderungen durch Kochen,» 2021. [Online]. Available: https://naehrwertdaten.ch/de/nahrwertveranderungen-durch-kochen/. [Zugriff am 11 Mai 2022]. |
| [6] | P. D. E. Behrends, «React Native: Einstieg in die Entwicklung mobiler Apps,» 20 März 2018. [Online]. Available: https://www.informatik-aktuell.de/entwicklung/programmiersprachen/react-native-einstieg-in-die-entwicklung-mobiler-apps.html. [Zugriff am 7 Mai 2022]. |
| [7] | F. Meyer, «Du kochst zu viel? Dann verkauf den Rest!,» 9 Juli 2016. [Online]. Available: https://www.zentralplus.ch/gesellschaft/du-kochst-zu-viel-dann-verkauf-den-rest-745341/#:~:text=%C2%ABHobbyk%C3%B6che%20d%C3%BCrfen%20Mahlzeiten%20von%20zu,%E2%80%93%20wie%20alle%20anderen%20auch.%C2%BB. [Zugriff am 14 Mai 2022]. |

## Abbildungsverzeichnis

[Abbildung 1: Nährwertangaben [2] 9](#_Toc105196479)

[Abbildung 2: Zutat roh/gekocht, Änderungsfaktor 13](#_Toc105196480)

[Abbildung 3: Beispiel Rezept Merging 15](#_Toc105196481)

[Abbildung 4: Entscheidungsmatrix Backend 23](#_Toc105196482)

[Abbildung 5: Gliederung Rezept Daten 24](#_Toc105196483)

## Tabellenverzeichnis

[Tabelle 1: Rahmenbedingungen (Prototyp) 16](#_Toc105196484)

[Tabelle 2: User Story 1 - Anzeige der Gerichte 16](#_Toc105196485)

[Tabelle 3: User Story 2 - Detailansicht des Gerichts 17](#_Toc105196486)

[Tabelle 4: User Story 3 - Bestellvorgang 17](#_Toc105196487)

[Tabelle 5: User Story 4 - Gerichte suchen 17](#_Toc105196488)

[Tabelle 6: User Story 5 - Auflistung der bereits getätigten Bestellungen 18](#_Toc105196489)

[Tabelle 7: User Story 6 - Profilanzeige 18](#_Toc105196490)

[Tabelle 8: User Story 7 - Gerichte anbieten 18](#_Toc105196491)

[Tabelle 9: User Story 8 - Gerichte erfassen 19](#_Toc105196492)

[Tabelle 10: User Story 9 - Koch bewerten 19](#_Toc105196493)

# Anhang

Anhang

## Projektmanagement

Projektmanagement