

# Future Retail – Ein Prototyp

NFC-SelfScanning-Einkaufs-App

Berufsmaturitätsschule Zürich  
Technik, Architektur, Life Sciences

Berufsmaturitätsarbeit  
«Im Zeichen der Zeit»

Sven Jäger und Timon Hüppi; BIN18a  
Willi Felchlin  
29. November 2021

## **Abstract**

Auf den folgenden Seiten wird die Implementierung einer Konzept-App beschrieben, die mithilfe von NFC-Chips den herkömmlichen Kassenprozess eines Einkaufs in einem Supermarkt komplett ersetzen könnte.

Die Arbeit umschreibt den Entwicklungsprozess dieser konzeptionellen App. Nach der Konzeptionierung, die Zweck und Funktionsumfang festlegt, folgt die Definierung der zu verwendenden Technologien. Anschliessend wird die Programmierung aller Software-Komponenten beschrieben. Hierzu werden die Arbeitsschritte erläutert und es werden einzelne Aspekte der Implementierung detailliert und verständlich erklärt. Darauf folgt der Abschluss, welcher neben einem Protokoll von App-Tests auch eine Beschreibung und Bewertung des Resultats umfasst.

Es stellt sich heraus, dass eine solche Konzept-App gut funktioniert und zumindest die grundlegende Funktionalität nicht sehr komplex sein muss. Die entwickelte Lösung ist zwar weit weg von einer verkaufsfertigen Software, sie zeigt jedoch, wie die Umsetzung einer solchen Idee aussehen könnte.

## Inhalt

|       |  |    |
|-------|--|----|
| 1     | Einleitung.....                          | 4  |
| 2     | Hauptteil .....                          | 5  |
| 2.1   | Idee .....                               | 5  |
| 2.2   | Planung und Diagramme .....              | 6  |
| 2.2.1 | Anforderungen an die App .....           | 6  |
| 2.2.2 | Use-Case-Diagramm .....                  | 7  |
| 2.2.3 | Datenbank-Modell.....                    | 8  |
| 2.2.4 | App-Wireframes .....                     | 9  |
| 2.3   | Software Tech-Stack .....                | 13 |
| 2.3.1 | Auflistung der Technologien.....         | 13 |
| 2.3.2 | App .....                                | 13 |
| 2.3.3 | Backend-Server .....                     | 14 |
| 2.3.4 | Datenbank.....                           | 14 |
| 2.3.5 | Versionsverwaltung.....                  | 14 |
| 2.3.6 | Webserver.....                           | 15 |
| 2.3.7 | Domain .....                             | 15 |
| 2.4   | Implementierung des Backend-Servers..... | 16 |
| 2.4.1 | User-Authentisierung .....               | 16 |
| 2.4.2 | Server Endpoints .....                   | 16 |
| 2.5   | Implementierung der App .....            | 19 |
| 2.5.1 | Activities.....                          | 19 |
| 2.5.2 | Funktionalität.....                      | 21 |
| 2.5.3 | HTTP-Requests .....                      | 21 |
| 2.5.4 | NFC-Tags .....                           | 22 |
| 2.6   | Testfälle.....                           | 23 |
| 2.6.1 | Erläuterungen der negativen Tests .....  | 26 |
| 2.7   | Resultat .....                           | 27 |
| 3     | Schluss.....                             | 31 |
| 4     | Dank .....                               | 32 |
| 5     | Quellenverzeichnis .....                 | 32 |
| 5.1   | Quellen des technischen Produkts .....   | 32 |
| 5.2   | Quellen der schriftlichen BMA .....      | 36 |
| 5.3   | Abbildverzeichnis.....                   | 37 |
| 5.4   | Tabellenverzeichnis.....                 | 37 |
| 6     | Glossar .....                            | 38 |

# 1 Einleitung

Der Bereich des Detailhandels ist in stetigem Wandel. Erst kürzlich sind in den verschiedensten Filialketten die Self-Scanning-Stationen erschienen, bei denen man selbst seine Artikel einscannt und bargeldlos bezahlt. Unsere BMA setzt sich mit der nächsten Iteration dieser Entwicklung auseinander, bei welcher auch die Self-Scanning-Stationen ersetzt werden. Dies geschieht durch eine Smartphone-App und NFC-Chips. Das Oberthema "Im Zeichen der Zeit" gibt den Rahmen für das Thema der BMA, die sich mit zeitgemässer Entwicklung des Detailhandels befasst.

Ein solcher Einkaufsprozess ist etwas, das uns täglich betrifft, da Besuche in Supermärkten mittlerweile ein elementarer Bestandteil des Alltags von uns allen sind. In den nächsten Jahren ist mit viel Weiterentwicklung in diesem Bereich zu rechnen. Wir beide absolvieren eine Informatiklehre, zum Teil in der Retail-Branche. Daher ist die Motivation entstanden, bei diesen Entwicklungen mitzumischen und mit modernen Technologien wie NFC zu forschen.

So resultiert folgende Fragestellung: «Kann durch eine Smartphone-App der Kassenprozess noch weiter vereinfacht und dezentralisiert werden?». Daraus ergibt sich die Hypothese, dass das Ersetzen eines herkömmlichen Kassenprozesses mittels einer neuen App möglich ist.

Hauptsächlicher Bestandteil der BMA ist die Entwicklung der besagten Konzept-App, die genau diesen Anwendungsfall praktisch demonstriert und die technische Machbarkeit beweisen soll. In dieser schriftlichen Arbeit werden der Implementierungsprozess, die technischen Hintergründe und das Konzept genauer beschrieben.

Um die Entwicklung zu planen, wird zu Beginn das Konzept definiert. Wir beschreiben, was die App alles können soll und wie dies erreicht wird. Es wird genau aufgezeigt, welche Technologien zu verwenden sind und wie diese zusammen funktionieren. Nach einer Dokumentation der Implementierung wird das Endresultat beschrieben und bewertet. Viel praktisches Wissen für die Umsetzung stammt aus unserem Erfahrungsschatz, den wir während der Ausbildung sammeln durften. Ein anderer, sehr grosser Teil des benötigten Wissens, stammt aus den Tiefen des Internets. Die entsprechenden Quellen sind am Ende in Listenform aufgeführt.

## 2 Hauptteil

### 2.1 Idee

Aktuell tauchen bei vielen Grossverteilern neue Self-Scanning-Angebote auf. Bei einer Variante nimmt man beim Ladeneingang dedizierte Scanner-Geräte mit, welche man am Schluss wieder zurückgibt. Mit diesen kann der Kunde während dem Einkauf alle Produkte direkt einscannen und anschliessend an einem Terminal mittels elektronischer Zahlungsmittel bezahlen. Ebenso gibt es Kassen, bei denen man am Schluss des Einkaufs all seine Produkte selbst einscannen kann und anschliessend auf gleiche Weise, wie oben beschrieben, bezahlt. Es gibt bereits erste Lösungen, die das Smartphone integrieren. Seit dem 1. November 2021 wird in den Migros-Filialen schrittweise das Programm "SubitoGO" ausgerollt. Dabei scannt der Kunde mit dem Handy die Barcodes seiner Artikel. Anschliessend kann er direkt in der App bezahlen. So entfällt der Gang zur Kasse komplett. (Migros 2021)

Hier soll unsere Konzept-App anknüpfen und den Gedanken einen Schritt weiterspinnen. In unserer Vision blüht die Idee, die Verwendung der Kamera des Smartphones zu eliminieren. Da Barcodes z.T. zerknittert sein können, oder aus anderen Gründen von der Kamera nicht lesbar sind. Schon das Suchen des Barcodes kann, je nach Verpackung, einige Sekunden und ein paar Nerven rauben. Unserer Meinung nach wäre die Integration von NFC-Chips am Gestell, z.B. als Bestandteil des Preisschildes, oder direkt am Produkt eine sinnvolle Alternative. So könnte man einfach das Handy mit der App an den Chip halten, die Menge eingeben und schon wird der Artikel in den digitalen Warenkorb aufgenommen.

Es wäre auch eine hybride Lösung aus Kamera/Barcode und NFC denkbar. Dabei steht dem Kunden beides zur Verfügung. Je nach Präferenz der Person, oder technischer Ausstattung des Smartphones, kann die gewünschte Option gewählt werden. Denn nicht jedes Smartphone hat einen NFC-Leser verbaut.

Unternehmen wie Amazon haben bereits die nächste Iteration dieser Reise des Detailhandels konzeptioniert. Amazon nennt ihre Implementierung der Idee "AmazonGO". Die Technologie, die dahintersteckt und auf Computer Vision, künstlicher Intelligenz und Sensorik basiert, nennen sie "Just Walk Out Technology". Mit dieser ist es möglich, das Smartphone nach dem Check-In in der Hosentasche oder wo auch immer zu lassen. Man kann die Produkte einfach nehmen und ohne erneuten Griff zum Smartphone das Geschäft auf direktem Weg verlassen. Hier

passiert alles im Hintergrund. Mittels Kameras und diversen Sensoren, kann exakt ermittelt werden, welche Produkte der Kunde genommen hat und wann er den Laden verlassen hat. Dafür werden komplexe Algorithmen benutzt. (YouTube/Amazon 2016)

## **2.2 Planung und Diagramme**

Für die Planung der Applikation wird mittels verschiedener Diagramme, Skizzen und auch in Textform konzeptionell beschrieben, wie die App auszusehen und zu funktionieren hat.

### **2.2.1 Anforderungen an die App**

Die App soll für den User das zentrale Element der Interaktion mit dem System sein. Über die App werden Einkäufe abgewickelt, NFC-Chips von Artikeln gescannt, frühere Einkäufe angezeigt und das Profil manipuliert bez. das Passwort geändert.

Die Artikel sollen über einen NFC-Chip, der am Artikel oder im Preisschild am Gestell angebracht ist, von der App erkannt werden. Auf dem Chip ist die Artikelnummer des Artikels gespeichert, mit der die App die Details zum Artikel aus der Datenbank holt und dem Käufer anzeigt. Durch Angabe der Menge und einer Bestätigung des Benutzers, wird der Artikel zum Warenkorb hinzugefügt. Wenn alle Artikel erfasst sind, kann mit dem Klick auf einen Button zum Zahlungsvorgang gewechselt werden. Nach Auswahl der Zahlungsmethode, wird die Zahlung bestätigt. Allerdings ist die Bezahlung nur fiktiv, da es sich hier nur um eine Proof-Of-Concept-App handelt und kein reelles Geschäft abgewickelt wird.

In der App kann dem User eine Liste von allen bisherigen Einkäufen angezeigt werden mitsamt den dazugehörigen Artikeln und deren Preise.

Für die Benutzung der App ist es erforderlich, ein Benutzer-Profil zu erstellen. Beim Login-Screen wird die Möglichkeit geboten, auf einen Registrierung-Screen abzuspringen und ein Profil zu erstellen.

## 2.2.2 Use-Case-Diagramm

In einem Use-Case-Diagramm wird dargestellt, welche Anwendungsfälle die Applikation abdeckt und was für Interaktionspunkte es in der Anwendung gibt, mit denen der Akteur interagieren kann.

Im Diagramm ist der Kunde, welcher in den Laden geht und etwas kaufen möchte, als Akteur bestimmt.

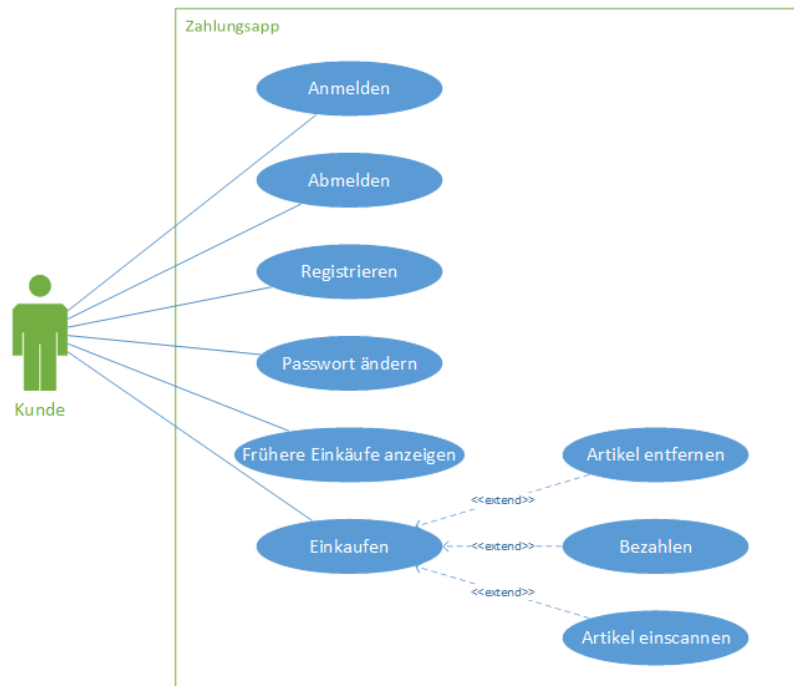
Dieser soll sich nun in der

Applikation bewegen können. Dafür

braucht es die Aktionen: "Anmelden", "Abmelden", falls der Kunde noch kein Profil hat "Registrieren" und "Passwort ändern".

Nach der Verifizierung des Kunden, kann der Einkauf starten. Er soll anschliessend alle Funktionen der Applikation verwenden können. Dazu soll er die Möglichkeit haben, Artikel via NFC-Chip einzuscannen und so dem Warenkorb hinzuzufügen. Falls ein Artikel falsch gescannt wurde, oder man sich umentschieden hat, soll es die Möglichkeit geben, diesen aus dem Warenkorb zu entfernen. Der Einkauf kann mittels Auswahl der Zahlungsmethode abgeschlossen werden.

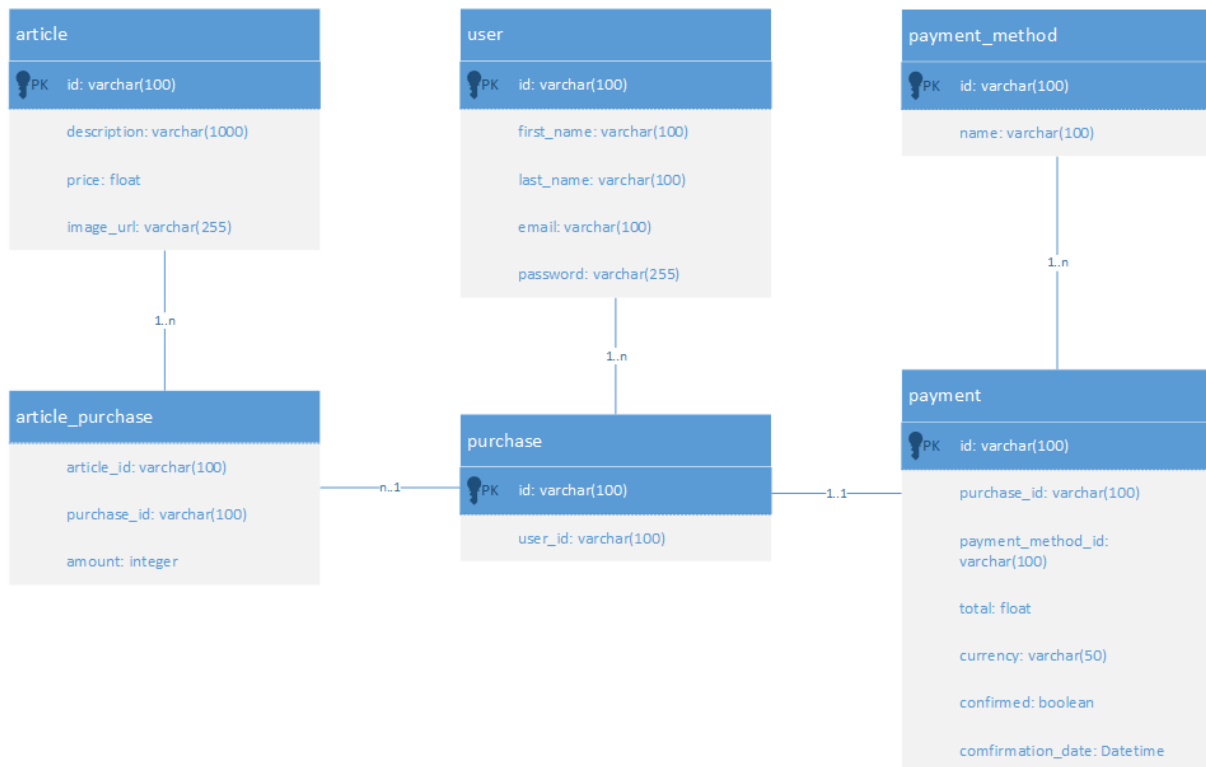
Damit man nach dem Einkauf eine gute Übersicht hat, kann man sich alle früheren Einkäufe und deren Artikel in einer Liste anzeigen lassen.



1 Use-Case-Diagramm der Funktionen

## 2.2.3 Datenbank-Modell

Im Datenbank-Modell ist aufgezeigt, wie die Datenbank aufgebaut ist. Es zeigt ebenso die Relationen zwischen den Tabellen und die jeweiligen Felder. (ÜK Modul 105 2019)



### 2 Datenbankmodell

Die Tabelle "user" speichert die Benutzer der Applikation. Sie beinhaltet die Felder "id" (eine UUID), "first\_name" (Vorname des Benutzers), "last\_name" (Nachname des Benutzers), "email" und "password".

Die Tabelle "payment\_method" speichert die verschiedenen Zahlungsmöglichkeiten. Sie beinhaltet eine "id" (eine UUID), als eindeutigen Schlüssel der unterschiedlichen Zahlungsmöglichkeiten und ebenfalls ein "name"-Feld, welches den Namen speichert. Diese Zahlungsmöglichkeiten sind statisch in der Datenbank gespeichert und können nicht von aussen verändert werden.

Die Tabelle "article" speichert alle Artikel, welche eingekauft werden können. Sie beinhaltet eine "id" (eine UUID), "description" (die Bezeichnung / die Beschreibung), "price" (den Preis) und "image\_url" (die URL zu einem Artikel-Bild). Das Feld für das Artikel-Bild ist zwar in der Datenbank vorhanden, wird aber sehr wahrscheinlich nicht verwendet, da in der App keine Darstellung eines Bildes geplant ist. Falls die App jedoch weiterentwickelt wird, kann dieses Feld verwendet werden. Die Artikel sind ebenfalls statisch gespeichert und können nicht von aussen verändert werden.



Die Tabelle "purchase" speichert Einkäufe, welche die Benutzer tätigen. Die Tabelle beinhaltet eine "id" (eine UUID) und eine "user\_id" (ebenfalls eine UUID, die der id eines Benutzers entspricht), da ein Einkauf immer einem Benutzer zugeordnet werden muss.

Die Tabelle "payment" enthält die Felder für die Zahlung, damit der Einkauf abgeschlossen werden kann. Sie beinhaltet die Felder "id" (eine UUID), "purchase\_id" (Referenz zum dazugehörigen Einkauf), "payment\_method\_id" (id der Zahlungsmethode, welche man ausgewählt hat), "total" (den errechneten Total-Preis, den man bezahlen muss), "currency" (Währung), "confirmed" (Zahlungsbestätigung) und "confirmation\_date" (Zeitpunkt der Zahlungsbestätigung).

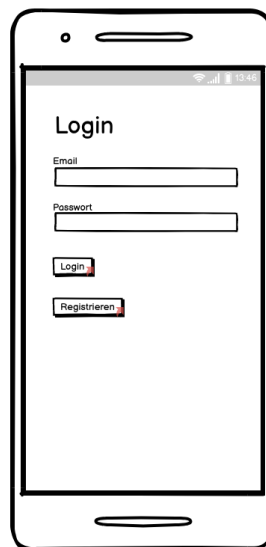
Da in relationalen Datenbanken keine Liste an Einträgen für ein Feld gespeichert werden kann, hier aber der Fall auftritt, dass ein oder mehrere Artikel einem Einkauf zugeordnet werden können, braucht es eine Zwischentabelle. Diese wird auch dann benötigt, wenn ein Artikel in einem oder mehreren verschiedenen Einkäufen vorkommt. Dafür gibt es die Tabelle "article\_purchase". Diese beinhaltet eine "article\_id" (Referenz zum Artikel), eine "purchase\_id" (Referenz zum Einkauf) und eine Mengenangabe des referenzierten Artikels im entsprechenden Einkauf.

#### **2.2.4 App-Wireframes**

Um eine App oder eine grafische Benutzerschnittstelle jeglicher Art zu entwerfen, gibt es das Wireframe. Dieses stellt die Anordnung der Elemente dar, ohne jedoch Details wie Farben, Schriftarten oder genauere Layouts vorzugeben. Für unsere App ist das sehr passend, da es bei diesem Projekt nicht um eine möglichst schöne, sondern um eine funktionale App geht.

Nachfolgend sind alle Screens der App mit einem Wireframe abgebildet. Während der Entwicklung der App haben sich noch ein paar wenige Dinge geändert. Mehr dazu steht im Kapitel der Implementation.

Login



A wireframe of a mobile app login screen. It features a status bar at the top with signal, Wi-Fi, and battery icons, and the time 12:46. The screen has a title 'Login' and two input fields labeled 'Email' and 'Passwort'. Below these are two buttons: 'Login' and 'Registrieren'.

3 Login-Wireframe

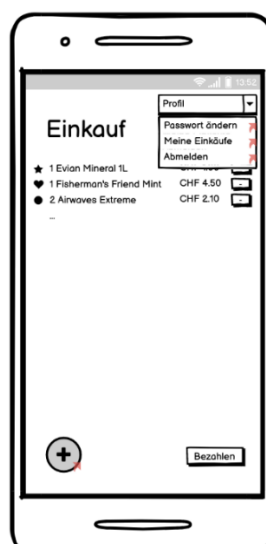
Registrieren



A wireframe of a mobile app registration screen. It features a status bar at the top with signal, Wi-Fi, and battery icons, and the time 13:32. The screen has a title 'Registrieren' and a 'Zurück' button at the top left. Below the title are five input fields labeled 'Email', 'Vorname', 'Nachname', 'Passwort', and 'Passwort bestätigen'. At the bottom is a 'Registrieren' button.

4 Registrieren-Wireframe

Einkauf



A wireframe of a mobile app shopping screen. It features a status bar at the top with signal, Wi-Fi, and battery icons, and the time 13:52. The screen has a title 'Einkauf' and a 'Profil' dropdown menu at the top right. Below the title is a list of items with their prices: '1 Evian Mineral 1L', '1 Fisherman's Friend Mint', and '2 Airwaves Extreme'. At the bottom are a '+' button and a 'Bezahlen' button.

5 Einkaufs-Wireframe

Passwort ändern

The wireframe shows a mobile app interface for changing a password. At the top, there is a 'Zurück' (Back) button. Below it is the title 'Passwort ändern'. The form contains three input fields: 'Aktuelles Passwort' (Current Password), 'Neues Passwort' (New Password), and 'Neues Passwort bestätigen' (Confirm New Password). At the bottom of the form is a button labeled 'Passwort ändern'.

6 Passwort-Änderungs-Wireframe

Meine Einkäufe

The wireframe shows a mobile app interface for viewing purchases. At the top, there is a 'Zurück' (Back) button. Below it is the title 'Meine Einkäufe'. The main content area is a list of purchases with the following data:

| Datum      | Produkt    | Preis |
|------------|------------|-------|
| 01.10.2021 | CHF 140.00 |       |
| 03.10.2021 | CHF 34.30  |       |
| 11.10.2021 | CHF 69.00  |       |
| ...        |            |       |

7 Meine-Einkäufe-Wireframe

Einkauf anzeigen

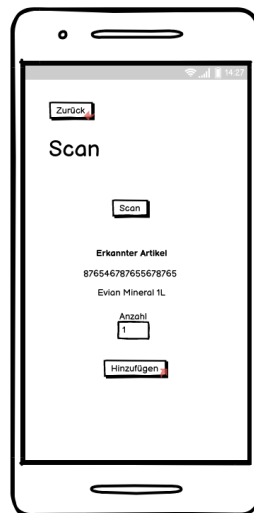
The wireframe shows a mobile app interface for viewing a detailed purchase receipt. At the top, there is a 'Zurück' (Back) button. Below it is the title 'Einkauf' and the date 'vom 01.10.2021'. The main content area is a list of items with the following data:

| Produkt                     | Preis    |
|-----------------------------|----------|
| ★ 1 Evian Mineral 1L        | CHF 1.00 |
| ♥ 1 Fisherman's Friend Mint | CHF 4.50 |
| ● 2 Airwaves Extreme        | CHF 4.20 |
| ...                         |          |

At the bottom, there is a 'Total' label and the amount 'CHF 140.00'.

8 Einkauf-Wireframe

Artikel scannen



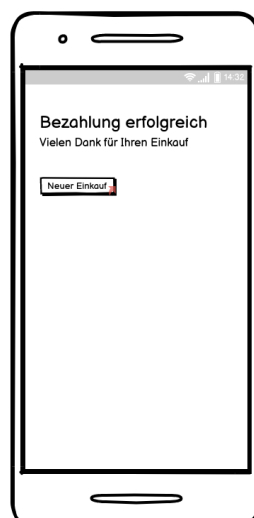
9 Scan-Wireframe

Bezahlen



10 Bezahlen-Wireframe

Bezahlung bestätigen



11 Bezahlen-Bestätigung-Wireframe

## 2.3 Software Tech-Stack

### 2.3.1 Auflistung der Technologien

Die technische Arbeit dieser BMA besteht aus mehreren Software-Komponenten, die hier aufgelistet sind:

- Android App
  - o Nativ Java
  - o Entwicklung mit der IDE "Android Studio"
- Backend-Server
  - o Node.js Express
  - o NPM Paketmanager
  - o Bietet die Schnittstelle zwischen App und Datenbank
- Datenbank
  - o MySQL
- Versionsverwaltung
  - o GitHub Open Source
  - o [github.com/tihuep/bma\\_nfc\\_retail](https://github.com/tihuep/bma_nfc_retail)
- Server-Deployment
  - o Ubuntu Server 21.04
  - o Linode "Nanode 1GB" in Frankfurt
  - o SSH, FTP
- Domain-Registrar
  - o Hostpoint
  - o Subdomain: bma.timonhueppi.ch

### 2.3.2 App

Die Entscheidung fiel auf eine native Android-App, da im überbetrieblichen Kurs mit der Nummer 335 genau dieses Thema behandelt wurde. Würde eine iOS-App entwickelt, wäre ein Mac-Computer erforderlich. Ein solcher stand uns jedoch nicht zur Verfügung. Zudem ist die NFC-Anbindung unter iOS weniger offen für Entwickler, als dies unter Android der Fall ist. Man hätte auch ein Framework wie "React Native" verwenden können. Allerdings ist auch da die Anbindung der NFC-Schnittstelle schwierig und man hätte eine externe Library verwenden müssen. Somit ist die Entscheidung dieser Konzept-App, der Einfachheit halber, auf eine native Android-App gefallen.

### **2.3.3 Backend-Server**

Für die Entwicklung der Backend-Server-Applikation hätten uns viele gute Möglichkeiten offen gestanden wie z.B. Java Spring oder Firebase. Gegen Java Spring hat vor allem die Komplexität gesprochen. Für eine solche Applikation ist die Java-Runtime-Environment erforderlich, die auf allen Entwicklungsstationen dieselbe Version, desselben Herausgebers haben sollte. Nur dies würde gewährleisten, dass auf allen Deployment-Plattformen dieselbe Funktionalität vorhanden ist. Zudem ist die Handhabung von Daten, durch die strenge Typisierung von Java, zwar sehr zuverlässig, jedoch auch ziemlich umständlich. Google's Firebase wäre eine sehr gute Alternative gewesen, da diese eigenständig in der Google-Cloud läuft. Es vereinfacht viele Dinge, weil sie Server-Applikation und Datenbank in Einem vereint, die User-Authentisierung bereits integriert hat und eine klassenbasierte Anbindung an Android-Apps bietet. Was hier doch dagegengesprochen hat, ist die Abhängigkeit von der Google-Cloud, die ab einem bestimmten Punkt nicht mehr kostenlos ist. So kam Node.js mit dem Express-Framework für Server-Endpoints zum Zuge. Diese Lösung bietet eine einfache Implementation der Endpoints und des Datenhandlings durch die lose Typisierung und den einfachen Aufbau des Express-Frameworks. Mit einem Befehl kann Node.js mit NPM auf dem Server installiert werden. Nach der Installation aller Abhängigkeiten ist der Server direkt startfähig.

### **2.3.4 Datenbank**

Die Auswahl einer Datenbank hat nicht lange gedauert, da MySQL in der Schule wie auch in überbetrieblichen Kursen schon zu Genüge benutzt wurde. Neben Oracle DB und Microsoft SQL-Server gehört sie zu den marktführenden relationalen Datenbank-Management-Systemen. (Statcounter 2021) MySQL ist sehr einfach aufzusetzen und zu warten, was sehr für unseren Entscheid spricht.

### **2.3.5 Versionsverwaltung**

Für die Versionsverwaltung geht heutzutage eigentlich kein Weg mehr an Git vorbei. Alternativen, wie SVN Subversion, sind ausserhalb von grossen Unternehmen nur sehr sporadisch verbreitet. Für Git gibt es verschiedene Online-Plattformen, die sogenannte Git-Repositories kostenlos zur Verfügung stellen, z.B. GitHub, GitLab, Sourceforge oder Bitbucket. Hier spielt es keine allzu grosse Rolle, welche Plattform man verwendet, denn schlussendlich tun alle dasselbe. Die Entscheidung ist auf GitHub gefallen, da diese am weitesten verbreitet ist.

### **2.3.6 Webserver**

Um die Datenbank und die Server-Applikation an einem zentralen Ort zu haben, ist ein Hosting von Nöten. Damit wird gewährleistet, dass jedes Smartphone mit der installierten App darauf zugreifen kann. Das Hosting wird entweder in virtueller Form von einem Hosting-Anbieter zur Verfügung gestellt oder man hat einen Server-Computer im eigenen Netzwerk und konfiguriert den Router so, dass vom Internet auf den Server zugegriffen werden kann. Wird letztere Option gewählt, muss das Aufsetzen und die Wartung von einem selbst gemacht werden. Das kann zeitaufwendig werden. Darum ist es in den meisten Fällen einfacher, einen virtuellen Server bei einem Hosting-Anbieter zu mieten, der diese Dinge erledigt. In unserem Fall reicht die günstigste Option des Anbieters "Linode" vollkommen aus. Diese kostet 5 Franken im Monat und bietet 1 GB Arbeitsspeicher, einen CPU-Core, 25 GB SSD-Speicher und 1 TB monatlicher Traffic-Quota. (Linode 2021)

Dieser Server kann über ein Konsolen-Fenster gesteuert werden, welches durch eine SSH-Session ermöglicht wird. An das Dateisystem kann zusätzlich ein FTP-Client, wie FileZilla, angeschlossen werden um den Zugriff darauf zu ermöglichen. Dadurch, dass unsere Arbeit auf GitHub gespeichert ist, kann der Server die Dateien selbst von dort holen und muss nicht manuell über FTP mit Dateien gefüttert werden. Auf diesem Server laufen schlussendlich die Backend-Server-Applikation auf Node.js und die MySQL-Datenbank.

### **2.3.7 Domain**

Um auf den eben erläuterten Server von den Apps her zugreifen zu können, kann entweder direkt die öffentliche IP-Adresse des Servers verwendet werden, was keine schöne Lösung wäre, oder man bindet eine Domain auf diese IP. Um keinen neuen Domain-Namen registrieren und kaufen zu müssen, haben wir uns für eine Subdomain auf timonhueppi.ch entschieden, die bereits in unserem Besitz ist. Somit haben wir den Namen bma.timonhueppi.ch, der auf den Server verweist, welcher unsere App verwendet.

## 2.4 Implementierung des Backend-Servers

### 2.4.1 User-Authentisierung

Damit die Applikation während des Gebrauchs weiss, welcher Kunde das Programm benutzt, braucht es eine Benutzer-Authentisierung. Diese verlangt eine E-Mail-Adresse zur Erkennung des Benutzers und ein Passwort, um den Zugang zu sichern. Mittels einem HTTP-Request (Methode 'POST') auf die '/register'-URL des Servers, kann ein Benutzer erstellt werden. Darauf wird dieser Benutzer direkt in der Applikation angemeldet. Die Anmeldung erfolgt via einem HTTP-Request (Methode 'POST') auf die '/login'-URL der Schnittstelle. Diese überprüft, ob E-Mail-Adresse und Passwort, welche im Request mitgeschickt wurden, mit den Daten eines Benutzers in der Datenbank übereinstimmt. Falls dies der Fall ist, wird ein JWT erstellt und als Antwort zurückgeschickt. Dieses wird im Anschluss für jeden Request gebraucht, damit der Server weiss, dass der aktuelle Benutzer authentisiert ist.

### 2.4.2 Server Endpoints

In der nachfolgenden Tabelle wird die Schnittstelle des Servers beschrieben, welche benutzt werden kann, um mit den Daten auf dem Server zu interagieren:

| URL                   | http-Request | Beschreibung   |
|-----------------------|--------------|--|
| /articles             | GET          | Gibt ein Array aller Artikel zurück  |
| /articles/{id}        | GET          | Gibt den Artikel mit der mitgegebenen id zurück  |
| /payment_methods      | GET          | Gibt ein Array aller Zahlungsmethoden zurück   |
| /payment_methods/{id} | GET          | Gibt die Zahlungsmethode mit der mitgegebenen id zurück                                      |
| /users                | GET          | Gibt ein Array aller Benutzer zurück   |
| /users/{id}           | GET          | Gibt den Benutzer mit der mitgegebenen id zurück   |
| /users/{id}           | PUT          | Aktualisiert den Benutzer mit der mitgegebenen id (Felder: id, first_name, last_name, email) |



| URL                                | http-Request | Beschreibung  |
|------------------------------------|--------------|---|
| /users/{id}/password               | PUT          | Aktualisiert das Passwort des Nutzers mit der mitgegebenen id. Die Schnittstelle wandelt das Passwort in einen Hash um, damit das Passwort nicht direkt aus der Datenbank gelesen werden kann |
| /users/{id}                        | DELETE       | Löscht den Benutzer mit der mitgegebenen id   |
| /purchases                         | GET          | Gibt ein Array aller Einkäufe für des angemeldeten Benutzer zurück  |
| /purchases/{id}                    | GET          | Gibt den Einkauf mit der mitgegebenen id zurück   |
| /purchases/user/{id}               | GET          | Gibt ein Array aller Einkäufe des Users mit der mitgegebenen id zurück  |
| /purchases                         | POST         | Erstellt einen Einkauf gemäss mitgegebenem JSON-Objekt (Felder: id (optional), user_id (Referenz zum Benutzer))   |
| /purchases/{id}                    | PUT          | Aktualisiert den Einkauf mit der mitgegebenen id (Felder: id, user_id)  |
| /purchases/{id}                    | DELETE       | Löscht den Einkauf mit der mitgegebenen id  |
| /purchases/{id}/items              | GET          | Gibt eine Liste aller Artikel-id's und der Anzahl, die sich im Einkauf befinden, für den Einkauf mit der mitgegebenen id zurück   |
| /purchases/{id}/items/{article_id} | GET          | Gibt ein Objekt zurück, das die Anzahl des angegebenen Artikels im angegebenen Einkauf enthält  |
| /purchases/{id}/items              | POST         | Fügt einen Artikel dem Einkauf mit der mitgegebenen id hinzu (Felder: purchase_id, article_id, amount)  |
| /purchases/{id}/items/{article_id} | PUT          | Aktualisiert die Menge des Artikels gemäss JSON-Objekt (Felder: purchase_id, article_id, amount)  |

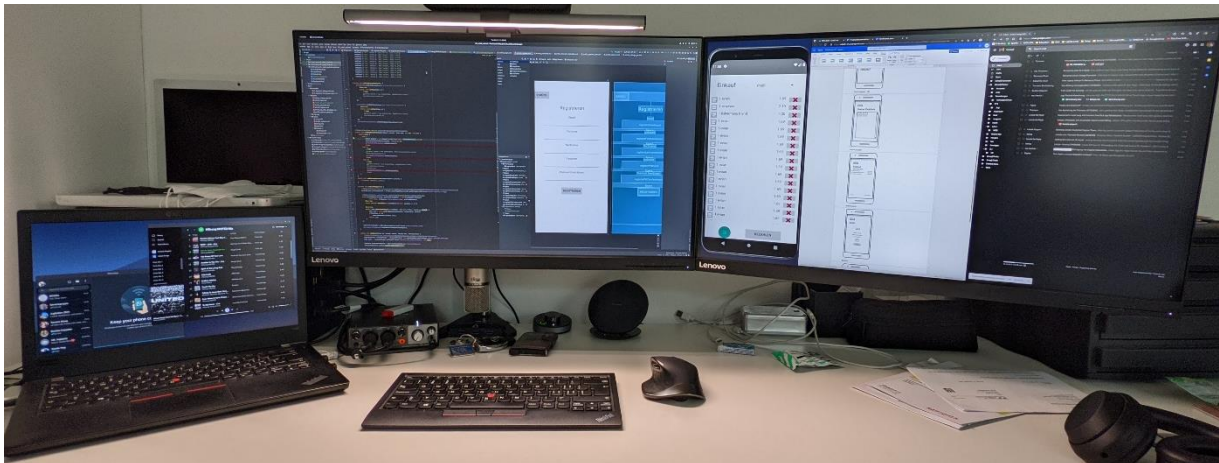
| URL                                | http-Request | Beschreibung  |
|------------------------------------|--------------|---|
| /purchases/{id}/items/{article_id} | DELETE       | Entfernt den Artikel mit der mitgegebenen id aus dem Einkauf mit der mitgegebenen id  |
| /payments                          | GET          | Gibt eine Liste aller Zahlungen zurück  |
| /payments/{id}                     | GET          | Gibt die Zahlung mit der mitgegebenen id zurück   |
| /payments/purchase/{purchase_id}   | GET          | Gibt die Zahlung für den Einkauf mit der mitgegebenen id zurück   |
| /payments                          | POST         | Erstellt eine Zahlung (Felder: id (optional), total, currency, confirmed (default ist false), confirmation_date, purchase_id (Referenz zum Einkauf), payment_method_id (Referenz zur Zahlungs-methode))                 |
| /payments/{id}                     | PUT          | Aktualisiert die Zahlung mit der mitgegebenen id (Felder: id, total, currency, confirmed (default ist false), confirmation_date, purchase_id (Referenz zum Einkauf), payment_method_id (Referenz zur Zahlungs-methode)) |
| /payments/{id}                     | DELETE       | Löscht die Zahlung mit der mitgegebenen id  |
| /payments/{id}/confirm             | POST         | Bestätigt die Zahlung mit der mitgegebenen id und setzt das confirmation_date auf das aktuelle Datum  |
| /login                             | POST         | Überprüft E-Mail-Adresse und Passwort, ob diese mit den Daten in der Datenbank übereinstimmen. Falls ja, wird ein JWT generiert und zurückgegeben   |
| /register                          | POST         | Erstellt einen neuen Benutzer. Anschliessend wird dieser direkt authentisiert und es wird ein JWT zurückgesendet  |

Tabelle 1 Server Endpoints

## 2.5 Implementierung der App

Die Entwicklung der Android-App wurde ausschliesslich mit der Entwicklungsumgebung "Android Studio" durchgeführt, da diese sehr umfangreiche Hilfsmittel bietet, die einem bei der Umsetzung helfen. z.B. macht es einem der grafische UI-Editor sehr einfach, die Benutzeroberfläche zu gestalten und die Elemente richtig zu platzieren. Diese IDE bietet die Möglichkeit, virtuelle Android-Geräte zu instanziiieren und die App darauf zu testen. Somit ist es für die meisten Fälle nicht nötig, ein physisches Gerät fürs Testen anzuschliessen. Ausnahme ist z.B. die NFC-Funktionalität, die im virtuellen Gerät nicht testbar ist.

Hier ein kleiner Eindruck, wie es aussehen könnte, wenn man die Benutzeroberfläche einer Android-App implementiert:



12 Android Entwicklungsumgebung

### 2.5.1 Activities

Eine Android-App besteht aus verschiedenen, sogenannten "Activities" oder "Aktivitäten". Eine Activity ist einem Screen gleichzusetzen. Jede Activity sieht anders aus und hat andere Funktionen, die einen oder mehrere Anwendungsfälle abdecken. Eine Activity beinhaltet jeweils eine View, die bestimmt, wie ein Screen auszusehen hat, und einen Controller, der die Logik, die hinter den Buttons und Eingabefeldern der View liegt, bestimmt.

Während in der View alles, was in der App zu sehen ist, bestimmt wird, reagiert der Controller im Hintergrund auf Interaktionen. Wird z.B. ein Knopf gedrückt, reagiert der "onClickListener" dieses Buttons, der im Controller definiert wird. Dieser sagt dann, was geschehen soll. So kann z.B. auf diese Aktion eine neue Activity geladen oder ein HTTP-Request an den Server gesendet werden. Die Möglichkeiten sind unbegrenzt. Auf diese Weise ist unsere ganze App aufgebaut. Auf jede Interaktion des Benutzers,

reagiert der Controller und löst eine Funktion aus, die etwas bewirkt. All das muss programmiert werden und ist darum sehr zeitaufwendig. Aufwendig macht es auch die Tatsache, dass Java, die Android-Programmiersprache unserer Wahl, zum Teil sehr umständlich ist und durch ihre starke Typisierung einige Tücken zu bieten hat. (ÜK Modul 335 2021)

Folgende Activities wurden implementiert:

- MainActivity
  - Wird gestartet, wenn die App geöffnet wird. Wenn ein Benutzer bereits angemeldet ist, wird an die PurchaseActivity weitergeleitet, ansonsten an die LoginActivity.
- LoginActivity
  - Zeigt die Login-Felder an und sendet die Daten an den Server. Bei einem erfolgreichen Login wird an die PurchaseActivity weitergeleitet.
- RegisterActivity
  - Ermöglicht das Registrieren eines neuen Benutzers. Leitet an die PurchaseActivity weiter oder bei einem Abbruch an die LoginActivity.
- PurchaseActivity
  - Zeigt den aktuellen Einkauf an. Bietet die Option, in die ChangePWActivity, MyPurchasesActivity, LoginActivity oder PaymentActivity abzuspringen.
- ScanActivity
  - Erscheint, wenn der NFC-Chip eines Artikels gescannt wird. Die Menge des Artikels kann angegeben werden. Bei Bestätigung wird an die PurchaseActivity weitergeleitet und der neue Artikel wird der Liste hinzugefügt.
- ChangePWActivity
  - Ermöglicht das Ändern des Passworts
- MyPurchasesActivity
  - Zeigt alle früheren Einkäufe an. Leitet bei der Auswahl eines Einkaufs an die ShowPurchaseActivity weiter.
- ShowPurchaseActivity
  - Zeigt die Artikel eines früheren Einkaufs.
- PaymentActivity

- Bietet die Option zur Auswahl der verschiedenen Zahlungsmethoden. Leitet an die PaymentConfirmActivity weiter.
- PaymentConfirmActivity
  - Bestätigt den Einkauf und die fiktive Zahlung. Leitet beim Abschluss an die PurchaseActivity weiter, um einen neuen Einkauf zu starten.

### **2.5.2 Funktionalität**

Jegliche Funktionalität, die hinter den Views der Activities und der direkt hinterlegten Controller-Funktionen, die auf Interaktion reagieren, liegt, kann separat zum Controller programmiert werden. Einfachheitshalber kann diese Funktionalität auch direkt in die Controller integriert werden, was unsere Herangehensweise war, denn so konnte noch etwas Zeit gespart werden.

Diese Funktionalitäten beinhalten zum Beispiel das Scanning der NFC-Chips. Dies war mit Abstand die grösste Hürde der funktionellen Implementierung. Denn die NFC-Schnittstelle und die Integration derselben in die App, ist unerwarteterweise relativ kompliziert umzusetzen. Um den Zugriff auf die NFC-Schnittstelle zu ermöglichen, muss die entsprechende Activity als NFC-Listener registriert werden. Geplant war, dass die Scan-Activity per Knopfdruck vom User gestartet wird, damit die App bereit ist, einen Artikel zu scannen. Es hat sich jedoch herausgestellt, dass dies so nicht möglich ist, weil das Smartphone die ganze Zeit aktiv nach NFC-Verbindungen sucht. Sobald ein Chip erkannt wird, wird die entsprechende Activity gestartet. Das hatte zur Folge, dass der Button zum Starten der Scan-Activity entfiel und die Scan-Activity nun automatisch erscheint, sobald ein Artikel erkannt wird. Die App funktioniert aber auch so, obwohl sie nicht mehr zu 100% der Planung entspricht.

Alle anderen Funktionen beschränken sich auf das Wechseln der Activities und Auswerten von Benutzereingaben. Beim Login z.B., oder beim Registrieren werden vom Controller die Werte, die in die Felder eingetragen werden, gelesen und auf deren Validität überprüft. Wenn die Werte valide sind, können sie weiterverarbeitet werden oder mit einem HTTP-Request an den Server gesendet werden.

### **2.5.3 HTTP-Requests**

Das Versenden von HTTP-Requests ist ziemlich simpel in Android. Wenn die Berechtigung für den Internet-Zugriff konfiguriert ist, werden verschiedene Parameter gesetzt, die relevant für das Versenden und Empfangen sind. Diese Parameter sind bei allen Requests gleich. Darum ist es sinnvoll, eine einzige Funktion für alle Requests

zu programmieren, mit der URL, den Headers, dem Body und der HTTP-Methode als Variablen, da sich diese jeweils unterscheiden können.

So kann auf die Endpoints des Servers zugegriffen und Daten abgeliefert und abgefragt werden. z.B. beim Registrierungsformular werden die Benutzerdaten an den Server gesendet. Und bei der Liste der früheren Einkäufe werden die Daten über den Server-Endpoint von der Datenbank abgefragt. So wird mit allen Daten, die in der App verwendet werden, umgegangen. Sie werden zentral verwaltet. Nur so ist ein MultiUser-System, wie es hier vorliegt, möglich. Denn wenn sich ein Benutzer auf einem anderen Smartphone anmeldet, will er dieselben Daten wie auf seinem eigenen Smartphone haben. Nur, wenn diese zentral auf dem Server gespeichert sind, können sie von überall abgefragt werden.

#### **2.5.4 NFC-Tags**

Die NFC-Tags, die wir verwenden, liegen in Form von kreisförmigen Stickern vor, etwa in der Grösse eines Fünflibers und fast so flach wie Papier. Ein NFC-Sticker kostet in unserem Fall Fr. -.80. Die Modellbezeichnung ist "NFC TAG Aufkleber Ntag213 13,56 MHz" und sie sind in Online-Shops wie der "[bastelgarage.ch](https://www.bastelgarage.ch)" bestellbar. Es gibt eine riesige Auswahl an NFC-Tags in allen Formen und Farben, die man in vielen Shops erwerben kann.

Um eine Artikelnummer auf die Sticker zu laden, ist entweder ein Schreibgerät oder eine NFC-App für das Smartphone von Nöten. Wir haben dafür die App "NFC Tools" von "wakdev" aus dem Android PlayStore verwendet. Die Artikelnummer kann damit aus der Datenbank auf die NFC-Tags geschrieben werden. Diese werden anschliessend mit einem Passwort gesichert, damit niemand die Artikelnummern überschreiben kann.

## 2.6 Testfälle

In der folgenden Tabelle sind verschiedene Testfälle, mit welchen die App getestet wurde, definiert. Diese sind zwar umfassend, aber es wird nicht jeder mögliche Fall abgedeckt, da es sich hier nicht um eine verkaufsfertige Lösung handelt, sondern um ein Konzept oder vielmehr um ein "Proof-Of-Concept". Anschliessend werden die negativ ausgefallenen Fälle erläutert.

| Testfall   | Erwartetes Resultat                                | Tatsächliches Resultat                                      | i.O. |
|--|--|---|------|
| Registration:<br>Email: "asdf"<br>Vorname: "Max"<br>Nachname:<br>"Mustermann"<br>Passwort: "asdf"                              | Fehler: Ungültige Daten (Email und Passwort)       | Fehler: "Bitte eine gültige Email-Adresse angeben"          | Ja   |
| Registration:<br>Email:<br>"max.musterm@bms-zuerich.ch"<br>Vorname: "Max"<br>Nachname:<br>"Mustermann"<br>Passwort: "asdf"     | Fehler: Ungültige Daten (Passwort)                 | Fehler: "Bitte gültiges Passwort angeben (mind. 6 Zeichen)" | Ja   |
| Registration:<br>Email:<br>"max.musterm@bms-zuerich.ch"<br>Vorname: "Max"<br>Nachname:<br>"Mustermann"<br>Passwort: "asdfasdf" | Registration erfolgreich, Einkauf-Screen erscheint | Wie erwartet  | Ja   |

| Testfall  | Erwartetes Resultat  | Tatsächliches Resultat                                      | i.O. |
|---|--|---|------|
| Login:<br>Email: "asdf"<br>Passwort: "asdf"                               | Fehler: Ungültige Daten (Email, Passwort)                      | Fehler: "Bitte eine gültige Email-Adresse angeben"          | Ja   |
| Login:<br>Email: "asdf"<br>Passwort: asfasdf"                             | Fehler: Ungültige Daten (Email)                                | Fehler: "Bitte eine gültige Email-Adresse angeben"          | Ja   |
| Login:<br>Email:<br>"max.musterm@bma-zuerich.ch"<br>Passwort: "asdfasdf"  | Login erfolgreich, Einkauf-Screen erscheint                    | Wie erwartet  | Ja   |
| Einkauf-Screen:<br>Klick auf Profil und Abmelden                          | User ist abgemeldet und Login-Screen erscheint                 | Wie erwartet  | Ja   |
| Einkauf-Screen:<br>Klick auf Profil und Passwort ändern                   | Passwort-Änderungs-Screen erscheint                            | Wie erwartet  | Ja   |
| Passwort-Änderungs-Screen:<br>Akt. PW: "jlöklök"<br>Neues PW: "qwer"      | Fehler: Ungültiges neues Passwort, Falsches aktuelles Passwort | Fehler: "Bitte gültiges Passwort angeben (mind. 6 Zeichen)" | Ja   |
| Passwort-Änderungs-Screen:<br>Akt. PW: "jlöklök"<br>Neues PW: "qwerqwer"  | Fehler: Falsches Aktuelles Passwort                            | Kein Fehler, PW wird geändert                               | Nein |
| Passwort-Änderungs-Screen:<br>Akt. PW: "qwerqwer"<br>Neues PW: "uiopuiop" | Kein Fehler, PW wird geändert                                  | Wie erwartet  | Ja   |



| Testfall   | Erwartetes Resultat   | Tatsächliches Resultat                    | i.O. |
|--|---|---|------|
| Einkauf-Screen:<br>Klick auf Profil und<br>Meine Einkäufe    | Meine-Einkäufe-Screen erscheint,<br>Einkäufe werden mit<br>Datum und Total<br>angezeigt, falls welche<br>existieren | Wie erwartet                              | Ja   |
| Meine-Einkäufe-Screen:<br>Aufgelisteter Einkauf<br>anklicken | Mein-Einkaufs-Screen<br>wird angezeigt mit<br>allen Artikeln in dem<br>Einkauf                                      | Wie erwartet                              | Ja   |
| Mein-Einkaufs-Screen:<br>Button Zurück                       | Meine-Einkäufe-Screen wird angezeigt  | Wie erwartet                              | Ja   |
| Meine Einkäufe-Screen:<br>Button Zurück                      | Einkaufs-Screen wird<br>angezeigt   | Wie erwartet                              | Ja   |
| NFC-Tag mit keiner<br>Artikelnummer wird<br>gescannt         | Nichts passiert   | App stürzt ab                             | Nein |
| NFC-Tag mit einer<br>Artikelnummer wird<br>gescannt          | Scan-Screen erscheint<br>und Menge kann<br>eingegeben werden  | Wie erwartet                              | Ja   |
| Scan-Screen:<br>Anzahl "0" wird<br>eingegeben                | Fehler: Ungültige<br>Anzahl   | Fehler: "Bitte zuerst<br>Artikel scannen" | Jein |
| Scan-Screen:<br>Anzahl "69" wird<br>eingegeben               | Gescannter Artikel mit<br>der Menge 69 wird<br>eingetragen  | Wie erwartet                              | Ja   |
| Einkauf-Screen:<br>Klick auf Bezahlen                        | Zahl-Screen erscheint<br>mit den Zahlungsmethoden   | Wie erwartet                              | Ja   |

| Testfall   | Erwartetes Resultat   | Tatsächliches Resultat                          | i.O. |
|--|---|---|------|
| Zahl-Screen:<br>Klick auf Bezahlen<br>ohne Selektion       | Fehler: Zahlungsmethode<br>auswählen                                      | Fehler: "Bitte<br>Zahlungsmethode<br>auswählen" | Ja   |
| Zahl-Screen:<br>Klick auf Bezahlen mit<br>Selektion        | Bestätigungs-Screen<br>erscheint  | Wie erwartet                                    | Ja   |
| Bestätigungs-Screen:<br>Klick auf "Neuer<br>Einkauf"       | Leerer Einkaufs-<br>Screen erscheint                                      | Wie erwartet                                    | Ja   |
| Einkaufs-Screen: Klick<br>auf Profil und Meine<br>Einkäufe | Meine-Einkäufe-<br>Screen erscheint mit<br>dem eben erstellten<br>Einkauf | Wie erwartet                                    | Ja   |

Tabelle 2 Testfälle

### 2.6.1 Erläuterungen der negativen Tests

21 von 24 Tests sind positiv ausgefallen, was einer Erfolgsquote von 87.5% entspricht. Das ist sehr gut, jedoch nicht perfekt. Da es sich hier nur um eine Konzept-App handelt, ist dies nicht sehr schlimm. Folgend sind die 3 nicht-positiven Testfälle kurz erläutert: Der erste negative Testfall, der die Passwortänderung betrifft, resultiert daraus, dass beim Ändern nicht überprüft wird, ob das aktuelle Passwort stimmt. Ergo kann man ohne das aktuelle Passwort zu wissen ein neues Passwort setzen, was in einer produktiven Anwendung ein ziemlich grosses Sicherheitsrisiko wäre.

Der zweite negative Testfall betrifft das NFC-Scanning. Hier wurde keine Validierung des Inhalts vom NFC-Tag implementiert. So versucht die App, was auch immer auf dem NFC-Tag steht, als Artikelnummer zu behandeln. Dies läuft schief, wenn es keine Artikelnummer ist. Die App stürzt darum ab.

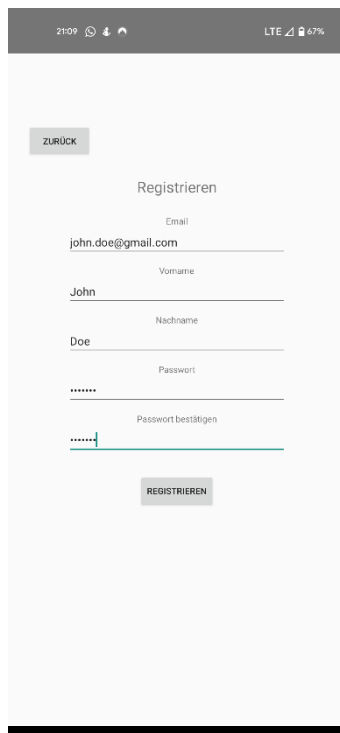
Der dritte nicht-positive Testfall ist lediglich eine etwas unpassende Fehlermeldung, die zwar im Ansatz Sinn macht, aber nicht klar das Problem widerspiegelt.

## 2.7 Resultat

Das Resultat dieser BMA ist weitgehend überzeugend. Die App funktioniert und sieht in etwa so aus, wie wir sie geplant haben. Somit kann unsere Leitfrage mit einem Ja beantwortet werden. Der Kassenprozess eines Einkaufes kann mit einer App und NFC-Chips abgebildet und ersetzt werden.

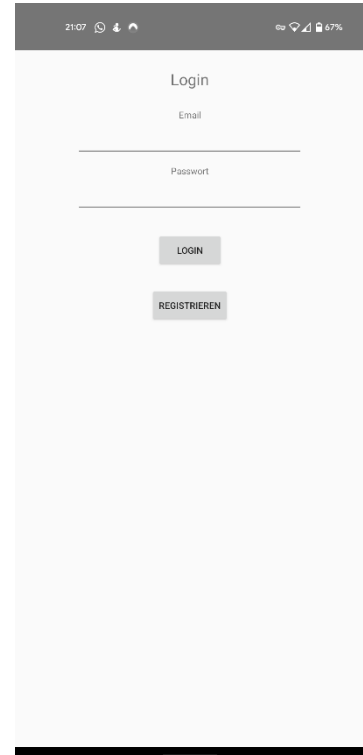
Im Folgenden präsentieren wir die App aus Benutzersicht:  
Wird die App gestartet, wird man vom Login-Screen begrüßt:

Hier hat man die Option, sich anzumelden oder sich zu registrieren, in dem man auf den Registrieren-Button klickt.  
Dies sieht dann folgendermassen aus:

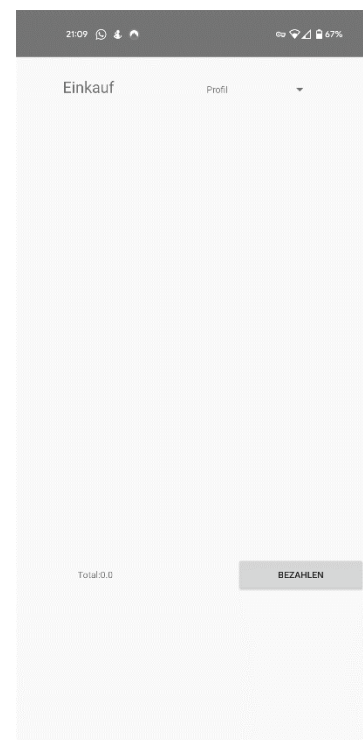


14 Registrieren-Seite

Gibt man hier seine persönlichen Daten ein und bestätigt das Formular, so wird man direkt mit dem neu erstellten Account angemeldet und man bekommt die Einkaufs-Seite zu sehen:



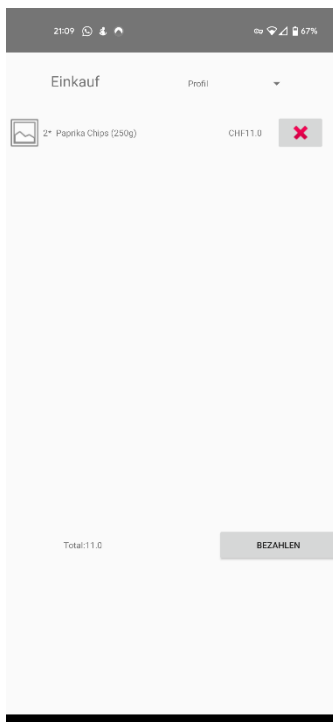
13 Login-Seite



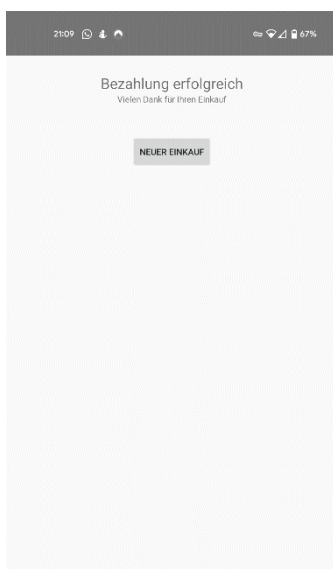
15 Leere Einkaufs-Seite

Diese ist zu Beginn noch leer, da noch keine Artikel gescannt wurden.

Wird nun ein NFC-Tag, auf dem eine Artikelnummer gespeichert ist, an das Smartphone gehalten oder umgekehrt, erscheint die Scan-Ansicht, auf welchem der gescannte Artikel angezeigt wird. In einem Eingabefeld kann die Menge des Artikels angegeben werden:



17 Gefüllte Einkaufs-Seite



19 Bezahlungsbestätigung-Seite

Mit einem Klick auf «Hinzufügen» kommt man zurück auf die Einkaufs-Seite, welche nun den eben gescannten Artikel in der Liste enthält:

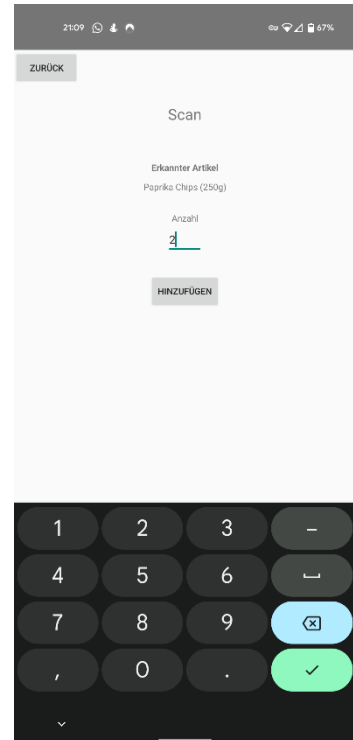
So kann das mit beliebig vielen Artikeln wiederholt werden.

Wenn man auf Bezahlen klickt, wird der Einkauf abgeschlossen. Anschliessend wählt man eine Zahlungsmethode aus:

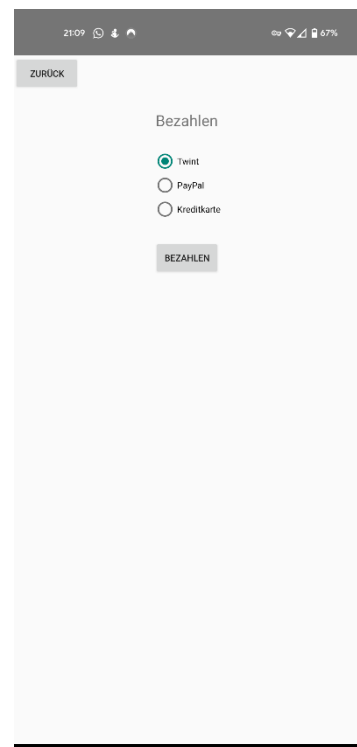
Nachdem eine Zahlungsmethode gewählt wurde, wird der Einkauf abgeschlossen.

Eine Bestätigung folgt:

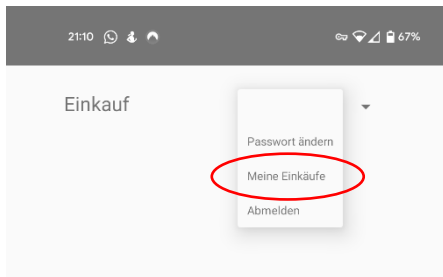
Anschliessend könnte ein neuer Einkauf gestartet werden.



16 Scan-Seite

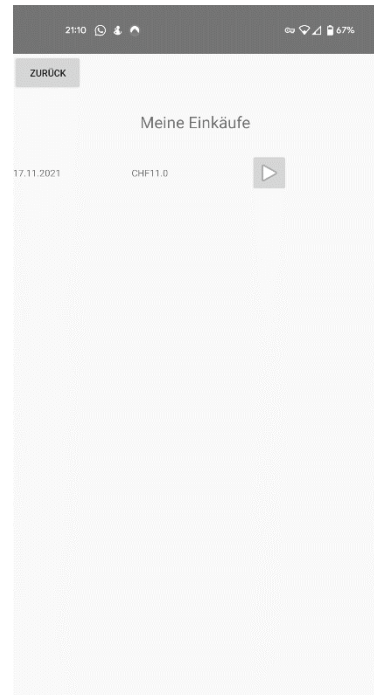


18 Bezahlungs-Seite

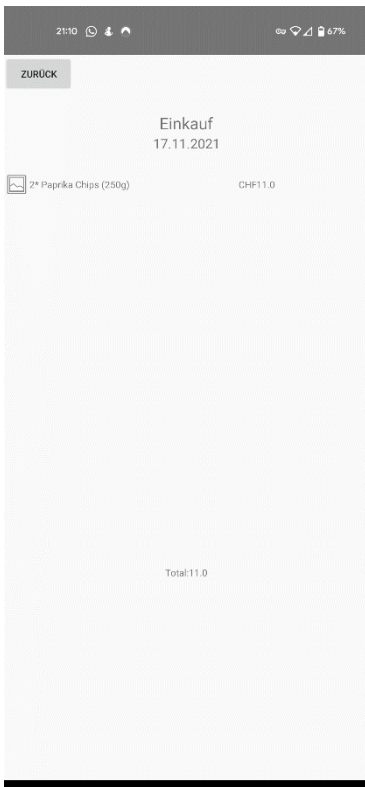


20 Benutzer-Menü "Meine Einkäufe"

Zurück auf der Einkaufs-Seite kann man sich unter «Profil» und «Meine Einkäufe» alle früheren Einkäufe anzeigen lassen:



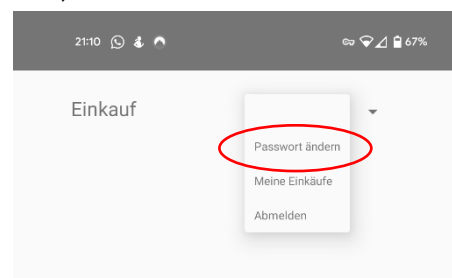
21 Meine Einkäufe-Seite



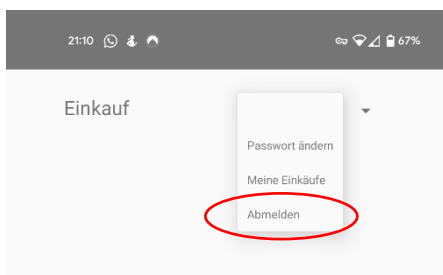
25 Einkaufs-Anzeige-Seite

Jeder der angezeigten Einkäufe kann ausgewählt werden. Es werden einem alle einzelnen Artikel des entsprechenden Einkaufs aufgelistet:

Zurück auf der Einkaufs-Seite, kann unter «Profil» und «Passwort ändern» das Passwort geändert werden:

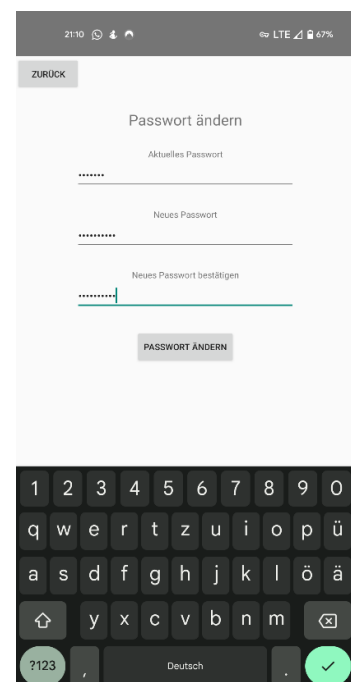


22 Benutzer-Menü "Passwort ändern"



24 Benutzer-Menü "Abmelden"

Ebenfalls auf der Einkaufs-Seite unter «Profil», besteht die Möglichkeit sich von seinem Account abzumelden:



23 Passwort-Änderungs-Seite

In der Datenbank sind nun der neue Benutzer und der erstellte Einkauf angelegt:

```
mysql> select * from user;
```

| id                                   | first_name | last_name | email              | password   |
|--------------------------------------|------------|-----------|--------------------|--|
| ab05e3bc-eee5-4307-b997-013f0c7ce02f | John       | Doe       | john.doe@gmail.com | \$2b\$10\$Bmn/aD\$7bbSKmDAbd9rmQuhaMIH9W8yPBMhA1NM4sOWQ7zM192Fhk |

6 rows in set (0.00 sec)

#### 26 Datenbankauszug der Benutzer

```
mysql> select * from purchase;
```

| id                                   | user_id                              |
|--------------------------------------|--------------------------------------|
| b5acf673-a70e-4ea6-8935-b113fbedfc90 | ab05e3bc-eee5-4307-b997-013f0c7ce02f |

5 rows in set (0.00 sec)

#### 27 Datenbankauszug der Einkäufe

```
mysql> select * from payment;
```

| id                                   | total | currency | confirmed | confirmation_date   | purchase_id                          | payment_method_id                    |
|--------------------------------------|-------|----------|-----------|---------------------|--------------------------------------|--------------------------------------|
| a68bebce-7989-4285-82cb-b1a096bd8fd4 | 11    | CHF      | 1         | 2021-11-17 21:09:50 | b5acf673-a70e-4ea6-8935-b113fbedfc90 | 5d4b02c5-8ede-44f5-9d18-a6d26b7ea7cc |

5 rows in set (0.00 sec)

#### 28 Datenbankauszug der Bezahlungen

```
mysql> select * from article_purchase;
```

| amount | purchase_id                          | article_id                           |
|--------|--------------------------------------|--------------------------------------|
| 2      | b5acf673-a70e-4ea6-8935-b113fbedfc90 | 6586d884-5945-4a2a-a56b-2fd9aa1536b4 |

9 rows in set (0.00 sec)

#### 29 Datenbankauszug der Zwischentabelle Artikel/Einkauf

Die ID des Benutzers (User) ist somit «ab05e3bc-eee5-4307-b997-013f0c7ce02f», die des Einkaufs (Purchase) «b5acf673-a70e-4ea6-8935-b113fbedfc90», die der Bezahlung (Payment) «a68bebce-7989-4285-82cb-b1a096bd8fd4» und die des Artikels (Article) namens «Paprika Chips (250g)», der vorher gescannt wurde, «6586d884-5945-4a2a-a56b-2fd9aa1536b4».

### 3 Schluss

Mit unserer Arbeit wollten wir herausfinden, ob es möglich ist, mithilfe einer App und NFC-Chips den Kassenprozess eines Einkaufes in einem Supermarkt zu vereinfachen oder gar zu ersetzen. Diese Leitfrage kann schliesslich mit Ja beantwortet werden, denn die Konzept-App funktioniert und der Einkaufsprozess konnte abgebildet werden. Das Oberthema "Im Zeichen der Zeit" konnte eingehalten werden, da eine aktuelle, moderne Entwicklung am Markt aufgegriffen und weitergedacht wurde.

Unsere Arbeit hat ergeben, dass es ohne grosse Komplexität möglich ist, die grundlegenden Funktionalitäten einer solchen Konzept-App zu implementieren. Doch das Resultat ist noch weit davon entfernt auch so im täglichen Gebrauch verwendet zu werden. Um die App weiter zu verbessern und bereit für eine produktive Anwendung zu machen, müssten noch einige Änderungen vorgenommen werden. Die Sicherheit der Daten, welche verwendet werden, ist noch nicht sehr weit entwickelt. Dies ist auch in den Testfällen zu sehen. Man kann zum Beispiel das Passwort eines Benutzers ändern, ohne das aktuelle Passwort zu wissen. Dies zu sichern, wäre ein erster Schritt, um die Datensicherheit in der App gewährleisten zu können. Eine strengere Sicherung der Daten auf dem Server ist auch nötig, unter anderem auch eine Beschränkung des Datenzugriffs der einzelnen Benutzer.

Auch die Zahlung der Einkäufe ist bisher sehr einfach implementiert und nur fiktiv abgebildet. Man müsste die Bezahlungssysteme richtig einbinden. Ebenfalls müsste überprüft werden, ob die Beträge über die Zahlungssysteme korrekt verbucht werden und ob das Geld am richtigen Ort ankommt, damit das Geschäft keine Artikel umsonst verkauft.

Zusätzlich wäre ebenfalls noch einiges am Optischen der App zu ändern. Sie sieht zwar nicht schlecht aus, doch sie ist nicht sehr attraktiv gestaltet. Manche Buttons sind noch an eher unpraktischen Stellen platziert und es fehlt etwas an Farbe. Doch da sich das Produkt der BMA nur auf eine funktionelle App und nicht auf eine möglichst schöne App konzentriert, ist es für diesen Zweck absolut ausreichend. Die Ergänzung durch eine iOS-App für Apple-Geräte wäre bei einer produktiven Anwendung ebenfalls unverzichtbar.

Mit dem Resultat sind wir zufrieden und es freut uns, dass die Idee funktioniert und die Funktionalität des Konzepts somit bewiesen ist.

## 4 Dank

Zum Schluss dieser Arbeit wollen wir uns bei allen Beteiligten bedanken. Vor allem bei den Personen, die die Arbeit gegengelesen haben, namentlich Mojca Jäger, Marlene Hüppi, Patrick Hüppi und Rahel Frei. Insbesondere möchten wir uns auch bei Willi Felchlin bedanken, da er unsere Klasse während dem gesamten BMA-Prozess betreut hat.

## 5 Quellenverzeichnis

### 5.1 Quellen des technischen Produkts

#### Android-Entwicklung

- <https://developer.android.com/guide/topics/connectivity/nfc/nfc>  
Android NFC Basics [Abrufdatum: 10.2021]
- <https://developer.android.com/training/basics/firstapp/starting-activity#java>  
Andere Activity starten [Abrufdatum: 10.2021]
- <https://stackoverflow.com/questions/33711380/how-to-remove-the-blue-action-bar-in-android>  
Blaue Action-Bar entfernen in Android [Abrufdatum: 10.2021]
- <https://stackoverflow.com/questions/4830864/android-find-item-by-id>  
Android XML-Element ansteuern [Abrufdatum: 10.2021]
- <https://developer.android.com/guide/topics/ui/controls/button>  
Button Click Handlers [Abrufdatum: 10.2021]
- <https://stackoverflow.com/questions/4038479/android-go-back-to-previous-activity>  
Zur letzten Activity wechseln [Abrufdatum: 10.2021]
- [https://developer.android.com/reference/android/app/Activity#finish\(\)](https://developer.android.com/reference/android/app/Activity#finish())  
Android Activity beenden [Abrufdatum: 10.2021]
- <https://stackoverflow.com/questions/18463848/how-to-tell-if-a-random-string-is-an-email-address-or-something-else/54122601>  
E-Mail überprüfen [Abrufdatum: 10.2021]
- <https://stackoverflow.com/questions/41194680/close-another-activity-from-current-one/41195489>  
Andere Activity beenden [Abrufdatum: 10.2021]



- <https://stackoverflow.com/questions/9807650/dynamically-cloning-a-linearlayout-in-android>  
Android View Inflate (duplizieren) [Abrufdatum: 10.2021]
- <https://www.tabnine.com/code/java/methods/android.view.View/findViewsByText>  
Views finden nach Text [Abrufdatum: 10.2021]
- <https://stackoverflow.com/questions/5178588/how-we-can-get-the-arraylistboth-string-and-integer-from-the-resources-xml>  
List generieren aus Ressourcen-Array [Abrufdatum: 10.2021]
- <https://android--code.blogspot.com/2015/08/android-spinner-disable-item.html>  
Spinner-Item disable (ausschalten) [Abrufdatum: 10.2021]
- <https://developer.android.com/guide/topics/ui/controls/spinner>  
Android Dropdown-Menu [Abrufdatum: 10.2021]
- <https://stackoverflow.com/questions/2091465/how-do-i-pass-data-between-activities-in-android-application>  
Daten zur nächsten Activity übertragen [Abrufdatum: 10.2021]
- <https://stackoverflow.com/questions/1124548/how-to-pass-the-values-from-one-activity-to-previous-activity>  
Daten zur letzten Activity übertragen [Abrufdatum: 10.2021]
- <https://stackoverflow.com/questions/7075349/android-clear-activity-stack>  
Activity-Stack leeren [Abrufdatum: 10.2021]
- <https://stackoverflow.com/a/2624385>  
Passwort hashen Java (Hash-Wert des Passwortes generieren) [Abrufdatum: 10.2021]
- <https://www.youtube.com/watch?v=n-8Aq3tp5IE>  
NFC ansteuern in Android [Abrufdatum: 10.2021]
- <https://developer.android.com/training/data-storage/shared-preferences>  
SharedPreferences in Android (Lokaler App-Speicher auf dem Handy) [Abrufdatum: 10.2021]
- <https://developer.android.com/training/volley/simple#java>  
Absetzen von Requests auf Android [Abrufdatum: 10.2021]

- <https://stackoverflow.com/a/12384156>  
GSON parse Array (Serialisierten Array in Java-Liste umwandeln)  
[Abrufdatum: 10.2021]
- <https://stackoverflow.com/questions/45940861/android-8-clear-text-http-traffic-not-permitted>  
Android Cleartext Web Traffic erlauben [Abrufdatum: 10.2021]
- <https://stackoverflow.com/a/48424181>  
Post Request in Android mit JSON Body [Abrufdatum: 10.2021]
- <https://stackoverflow.com/a/2400981>  
Java Datumsformatierung [Abrufdatum: 10.2021]
- <https://stackoverflow.com/a/34920059>  
GSON Deserialisierung boolean-hook (Serialisierten Boolean-Wert in einen Java-Boolean-Wert umwandeln) [Abrufdatum: 10.2021]
- <https://stackoverflow.com/a/46229554>  
Android HTTP Authentifizierungs-Headers für einen Request setzen  
[Abrufdatum: 10.2021]

### Server-Setup

- <https://www.digitalocean.com/community/tutorials/how-to-install-mysql-on-ubuntu-20-04>  
MySQL Installation auf dem Server [Abrufdatum: 10.2021]
- <https://medium.com/@benmorel/creating-a-linux-service-with-systemd-611b5c8b91d6>  
Einrichten eines systemd Services auf dem Server [Abrufdatum: 10.2021]

### Backend-Server Entwicklung Node.js

- <https://expressjs.com/de/starter/hello-world.html>  
Node.js Express Applikation Beispiel [Abrufdatum: 10.2021]
- <https://expressjs.com/de/guide/database-integration.html#mysql>  
JavaScript Datenbank-Anbindung [Abrufdatum: 10.2021]
- [https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express\\_Nodejs/routes](https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/routes)  
Express Node.js Routing mit Router [Abrufdatum: 10.2021]

- <https://stackoverflow.com/questions/35256272/using-variables-in-a-node-js-mysql-node-query>  
Variablen für SQL-Statements verwenden [Abrufdatum: 10.2021]
- <https://www.uuidgenerator.net/version4>  
UUID-Generator [Abrufdatum: 10.2021]
- <https://stackoverflow.com/a/35865605>  
Error-Message mit Status senden [Abrufdatum: 10.2021]
- <https://developer.mozilla.org/de/docs/Web/HTTP/Status>  
HTTP-Status-Beschreibungen [Abrufdatum: 10.2021]
- <https://youtu.be/Ud5xKCYQTjM>  
Node.js User Authentication [Abrufdatum: 10.2021]
- <https://youtu.be/mbsmsi7l3r4>  
Node.js JWT Authentication [Abrufdatum: 10.2021]
- <https://www.npmjs.com/package/bcrypt>  
Bcrypt-Package für Node.js um einen Hashwert zu generieren [Abrufdatum: 10.2021]
- <https://www.npmjs.com/package/express>  
Express-Package, um eine Schnittstelle zu bauen [Abrufdatum: 10.2021]
- <https://www.npmjs.com/package/jsonwebtoken>  
JsonWebToken-Package, um JWT zu generieren oder zu überprüfen, ob das Token gültig ist [Abrufdatum: 10.2021]
- <https://www.npmjs.com/package/mysql>  
MySQL-Package für die Datenbankanbindung [Abrufdatum: 10.2021]
- <https://www.npmjs.com/package/uuid>  
UUID-Package für die UUID-Generierung [Abrufdatum: 10.2021]

## 5.2 Quellen der schriftlichen BMA

- DB-Engines: Ranking / Complete Ranking <https://db-engines.com/en/ranking> [11.2021] [Abrufdatum: 08.11.2021]
- Statcounter: Desktop Operating System Market Share Worldwide <https://gs.statcounter.com/os-market-share/desktop/worldwide> [10.2021] [Abrufdatum: 10.11.2021]
- Amazon (05.12.2016): Introducing Amazon Go and the world's most advanced shopping technology. <https://www.youtube.com/watch?v=NrmMk1Myrxc> [Abrufdatum: 22.11.2021]
- Migros: subito / subitoGo <https://www.migros.ch/de/services/zahlungsmoeglichkeiten/subito.html#heading-subitogo> [11.2021] [Abrufdatum: 22.11.2021]
- Bastelgarage.ch: NFC TAG Aufkleber Ntag213 13,56 MHz <https://www.bastelgarage.ch/nfc-tag-aufkleber-ntag213-13-56-mhz?search=nfc%20tag%20aufkleber%20ntag213%2013%2C56%20mhz> [Abrufdatum: 22.06.2021]
- ICT Berufsbildung: 105 Datenbanken mit SQL bearbeiten <https://www.modulbaukasten.ch/module/f074c9da-716c-eb11-b0b1-000d3a830b2b/de-DE?title=Datenbanken-mit-SQL-bearbeiten> [28.07.2021] [Abrufdatum: 22.11.2021]
- ICT Berufsbildung: 335 Mobile-Applikation realisieren <https://www.modulbaukasten.ch/module/b475c9da-716c-eb11-b0b1-000d3a830b2b/de-DE?title=Mobile-Applikation-realisieren> [20.08.2021] [Abrufdatum: 22.11.2021]
- Linode: Compute – Shared CPU <https://www.linode.com/docs/products/compute/shared-cpu/> [16.11.2021] [Abrufdatum: 22.11.2021]

### 5.3 Abbildungsverzeichnis

Alle in der Arbeit verwendeten Abbildungen sind eigene Aufnahmen.

|  |    |
|--|----|
| 1 Use-Case-Diagramm der Funktionen .....                     | 7  |
| 2 Datenbankmodell .....                                      | 8  |
| 3 Login-Wireframe .....                                      | 10 |
| 4 Registrieren-Wireframe .....                               | 10 |
| 5 Einkaufs-Wireframe .....                                   | 10 |
| 6 Passwort-Änderungs-Wireframe .....                         | 11 |
| 7 Meine-Einkäufe-Wireframe .....                             | 11 |
| 8 Einkauf-Wireframe .....                                    | 11 |
| 9 Scan-Wireframe .....                                       | 12 |
| 10 Bezahlen-Wireframe .....                                  | 12 |
| 11 Bezahlen-Bestätigung-Wireframe .....                      | 12 |
| 12 Android Entwicklungsumgebung .....                        | 19 |
| 13 Login-Seite .....   | 27 |
| 14 Registrieren-Seite .....                                  | 27 |
| 15 Leere Einkaufs-Seite .....                                | 27 |
| 16 Scan-Seite .....  | 28 |
| 17 Gefüllte Einkaufs-Seite .....                             | 28 |
| 18 Bezahlungs-Seite .....                                    | 28 |
| 19 Bezahlungsbestätigung-Seite .....                         | 28 |
| 20 Benutzer-Menü "Meine Einkäufe" .....                      | 29 |
| 21 Meine Einkäufe-Seite .....                                | 29 |
| 22 Benutzer-Menü "Passwort ändern" .....                     | 29 |
| 23 Passwort-Änderungs-Seite .....                            | 29 |
| 24 Benutzer-Menü "Abmelden" .....                            | 29 |
| 25 Einkaufs-Anzeige-Seite .....                              | 29 |
| 26 Datenbankauszug der Benutzer .....                        | 30 |
| 27 Datenbankauszug der Einkäufe .....                        | 30 |
| 28 Datenbankauszug der Bezahlungen .....                     | 30 |
| 29 Datenbankauszug der Zwischentabelle Artikel/Einkauf ..... | 30 |

### 5.4 Tabellenverzeichnis

Alle in der Arbeit verwendeten Tabellen sind aus eigenen Quellen.

|                                  |    |
|----------------------------------|----|
| Tabelle 1 Server Endpoints ..... | 18 |
| Tabelle 2 Testfälle .....        | 26 |

## 6 Glossar

Da die Softwareentwicklung einen grossen Fachjargon bietet und viele Anglizismen verwendet, sind folgend einige in der BMA erwähnten Begriffe erklärt:

| Begriff                     | Erläuterung  |
|-----------------------------|--|
| Android                     | Mobiles Betriebssystem primär auf Smartphones, entwickelt von Google.  |
| API                         | Engl.: « <b>A</b> pplication <b>P</b> rogramming <b>I</b> nterface»: Programmier-Schnittstelle, die Kommunikation zweier Programme oder Geräte erlaubt.  |
| Array                       | Eine Liste an Daten.   |
| Authentisierung             | Eine Überprüfung, dass der zugreifende User angemeldet ist und nicht gefälscht wird.   |
| Autocompleter               | Ein Hilfsmittel einer IDE, die dem Programmierer vorausdenkt und Vorschläge zur Vervollständigung des Codes gibt.  |
| Backend                     | Der Teil einer Software, die die Daten für das Frontend zur Verfügung stellt und Daten vom Frontend entgegennimmt, verarbeitet und ggf. abspeichert.   |
| Computer Vision             | Eine auf künstlicher Intelligenz basierende Technologie, die Computern erlaubt, mittels Kameras Bildinformationen auszuwerten.   |
| CPU                         | Die zentrale Recheneinheit eines Computers. CPUs sind in mehrere Cores aufgeteilt, die unabhängig voneinander und parallel, Rechenoperationen ausführen können.  |
| Datenbank                   | Übergreifender Begriff eines Datenspeichers. In diesem Kontext ein tabellarischer Speicher auf einem Server.   |
| Datenbank, relationale-     | Siehe «Relationale Datenbank»  |
| Datenbank-Management-System | Eine Software, die Daten strukturiert oder unstrukturiert abspeichert und für spätere Zugriffe bereithält. Je nach dem, bieten diese Systeme Möglichkeiten, die Daten zu indexieren, um so die Zugriffssperformance zu steigern. |
| DBMS                        | siehe «Datenbank-Management-System»  |
| Deployment                  | Die Installierung einer Software auf einem Server.   |

|                  |  |
|------------------|--|
| Domain           | Ein Name unter einer Top-Level-Domain, wie dem Ländercode .ch, der auf die IP eines Internet-Servers verweist.<br>z.B. timonhueppi.ch  |
| domain, Sub-     | siehe «Subdomain»  |
| Domain-Registrar | Eine Organisation oder ein Unternehmen, das Domains registriert und dem Administrator die Möglichkeit bietet, die Domain zu konfigurieren.   |
| Endpoint         | Ein Schnittpunkt, über welchen man von aussen, über ein Netzwerk, auf ein Programm zugreifen kann, meist eine Server-Applikation.  |
| Firebase         | Eine cloudbasierte, fertige Datenbanklösung von Google.  |
| Framework        | Programmiergerüst, das in der Softwaretechnik verwendet wird.  |
| Frontend         | Nimmt Daten vom Backend entgegen und präsentiert diese dem User. Ggf. schickt es auch erfasste Daten wieder zurück an das Backend.   |
| FTP              | Engl.: « <b>F</b> ile <b>T</b> ransfer <b>P</b> rotocol»: Ein Protokoll, das für die Übertragung von Dateien über ein Netzwerk zuständig ist.  |
| Git              | Eine Software zur Versionsverwaltung von Source-Code. Ermöglicht die gleichzeitige Entwicklung von mehreren Programmierern am selben Projekt.  |
| GitHub           | Eine kostenlose, cloudbasierte Git-Plattform. Gehört Microsoft.  |
| Hash             | Eine Zeichenkette, welche mit einem bestimmten Algorithmus so verschlüsselt wurde, damit diese nicht mehr in den ursprünglichen Text zurückgewandelt werden kann. Ein Hash wird oftmals in der Softwareentwicklung bei Passwörtern verwendet, um diese zu sichern. |
| Hosting          | Wenn Dateien oder Server-Applikationen auf einem Server liegen und dem Internet zur Verfügung gestellt werden, spricht man von Hosting.  |
| HTTP-Methode     | Zu jedem HTTP-Request wird eine Methode mitgeschickt, die dem Server mitteilt, was mit den gesendeten Daten geschehen soll.  |

|               |   |
|---------------|---|
| HTTP-Request  | Über ein Netzwerk versendete Botschaften zwischen Systemen (Applikation und Server), die so kommunizieren und Daten austauschen können.   |
| IDE           | Engl.: « <b>I</b> ntegrated <b>D</b> evelopment <b>E</b> nvironment»: Eine Entwicklungsumgebung für Software, die viele kleine Helferlein beinhaltet, wie z.B. einen Autocompleter oder einen Live-Syntax-Checker.                                |
| iOS           | Mobiles Betriebssystem für Smartphones der Marke Apple, entwickelt von Apple selber.  |
| IP (-Adresse) | Eine in einem Netzwerk einmalige Adresse, die ein Gerät identifiziert und für die Adressierung von Datenübertragungen verwendet wird  |
| Java          | Programmiersprache, deren Source-Code auf einer Plattformspezifischen Runtime läuft, und somit Plattformunabhängig lauffähig ist. Auf Android läuft z.B. auch Java, sowie unsere App.   |
| Java-Runtime  | Bietet die Schnittstelle zwischen Java-Code und dem System. Auf der Seite des Java-Codes ist die Runtime auf jeder Plattform identisch, wodurch derselbe Java-Code auf jeder Plattform verwendet werden kann und nicht umgeschrieben werden muss. |
| JSON          | Engl.: « <b>J</b> ava <b>S</b> cript <b>O</b> bject <b>N</b> otation»: Ein Datenformat welches vielfach in HTTP-Requests verwendet wird, um Daten zu serialisieren und auszutauschen.   |
| JWT           | Engl.: « <b>J</b> SON <b>W</b> eb <b>T</b> oken»: Ein auf JSON basiertes Zugangs-Token, welche in der Softwareentwicklung verwendet wird, um einem Benutzer den Zugang zu einer Applikation zu ermöglichen.                                       |



|                  |   |
|------------------|---|
| Linux            | GNU/Linux. Ein Betriebssystem für PCs oder Server, dessen Source Code öffentlich einsehbar ist und laufend von einer Community aus Unternehmen und Privatpersonen weiterentwickelt wird. Linux ist im Serverbereich sehr weit verbreitet, allerdings im Desktop-Bereich einen Marktanteil von nur etwas über 2.15% hat. (Statcounter.com, 2021) |
| Library          | Ein Softwarepaket, das von anderer Software verwendet werden kann und so dessen Funktionalität weiterverwendet wird. Wird meistens mit Paketmanagern vertrieben.  |
| Mac-Computer     | PC der Marke Apple mit dem MacOS-Betriebssystem.  |
| MySQL            | Eine relationale SQL-Datenbank, die kostenlos installiert und verwendet werden kann.  |
| Nativ            | Auf Software bezogen: Wenn eine Software oder Laufzeitumgebung ohne Zwischenschicht auf dem Betriebssystem läuft und mit diesem kommunizieren kann.   |
| Netzwerk         | Eine Gruppe von verbundenen Geräten, die über ein Adress-System (IP) miteinander kommunizieren können.  |
| NFC              | Engl.: « <b>N</b> ear <b>F</b> ield <b>C</b> ommunication»: Eine Technologie zur Übertragung von Informationen zweier nahe gelegenen Geräte.  |
| Node (Node.js)   | Plattformübergreifende Open-Source JavaScript-Laufzeitumgebung, die JavaScript-Code ausserhalb eines Webbrowsers ausführen kann.  |
| NPM              | Engl.: « <b>N</b> ode <b>P</b> ackage <b>M</b> anager»: Ein Paketmanager, der unzählige Software-Pakete zur Entwicklung auf Node.js bereitstellt.   |
| Open Source      | Eine Software, deren Quellcode öffentlich im Internet zugänglich ist und/oder von freiwilligen «Contributors» weiterentwickelt wird.  |
| Paketmanager     | Hält Softwarepakete bereit für andere Software, damit diese darauf zugreifen kann und diese für ihre Zwecke verwenden kann.   |
| Proof-Of-Concept | Ein englischer Begriff für etwas, dessen alleiniger Zweck es ist, zu beweisen, dass ein Konzept funktionieren könnte.   |

|                       |  |
|-----------------------|--|
| Registrar             | Eine Organisation (Firma/etc.), die Domains registrieren kann und dem Administrator Konfigurationsmöglichkeiten zur Verfügung stellt.  |
| Relationale Datenbank | Eine Datenbank, die Daten strukturiert in meist tabellarischer Form speichert und deren Felder aufeinander referenzieren können. So nehmen verschiedene Daten aufeinander Bezug und bilden «Relationen». |
| Router                | Ein Gerät, das einem lokalen Netzwerk, z.B. eines Haushaltes, den Zugang zum Internet ermöglicht.  |
| Server                | Entweder ein Rechner, der ans Internet angeschlossen ist und diesem Daten oder Datenverarbeitungen zur Verfügung stellt oder ein Programm, das ebendieses tut.   |
| Server (Virtueller -) | Siehe Virtueller Server  |
| Server (Web-)         | Siehe Webserver  |
| Server-Applikation    | Ein Programm, das auf einem Server läuft und einem Netzwerk Daten zur Verfügung stellt.  |
| Source-Code           | Die ausgeschriebenen Befehle eines Computer-Programms  |
| Spring                | Ein Framework für Java, das die Implementierung von Endpoints z.B. als eine Server-Applikation ermöglicht.   |
| SQL                   | Engl.: « <b>S</b> tructured <b>Q</b> uery <b>L</b> anguage»: Eine Computer-Sprache, die für Datenbanken verwendet wird und mit welcher man Daten selektieren, filtern, sortieren, etc. kann.             |
| SSH                   | Engl.: « <b>S</b> ecure <b>S</b> hell»: Ein Protokoll, mit dem man gesicherten Internetverkehr ermöglichen und somit z.B. auf eine Computer-Konsole über ein Netzwerk zugreifen kann                     |
| Subdomain             | Ein der übergeordneten Domain hierarchisch untergeordneter Name, der wie die Domain selber auf eine IP zeigen kann. z.B. bei bma.timonhueppi.ch ist "bma" die Subdomain von "timonhueppi.ch".            |
| Syntax                | Ein Muster, wie ein Computer-Befehl auszusehen hat, damit der Computer diesen versteht.  |

|                   |  |
|-------------------|--|
| Traffic-Quota     | Ein begrenztes Datenlimit, das während einem Zeitraum aufgebraucht werden darf.  |
| Typisierung       | Die Festlegung eines Datentyps für ein Datenfeld, z.B. Zeichenketten, Ganzzahlen, Gleitkommazahlen, etc.   |
| Ubuntu            | Eine Distribution des Linux-Betriebssystems, basierend auf Debian.   |
| URL               | Engl.: « <b>U</b> niform <b>R</b> esource <b>L</b> ocator»: Eine Zeichenkette, die den Ort einer digitalen Ressource beschreibt. Meistens in Form einer Internetadresse.   |
| Use-Case-Diagramm | Es wird dargestellt, welche Anwendungsfälle eine Applikation abdeckt und welche Interaktionspunkte es in einer Anwendung gibt, mit welchen der Akteur interagieren kann.   |
| UUID              | Engl.: « <b>U</b> niversally <b>U</b> nique <b>I</b> dentifier»: Eine einzigartige, zufällig generierte Zeichenkette, welche zur Identifikation von Informationen in einem System verwendet werden kann.   |
| Virtueller Server | Ein Server, der physisch nicht existiert, jedoch in einem Rechenzentrum virtuell errechnet wird und so die gleichen Funktionen bietet wie ein physischer Server. Virtuelle Server sind weit verbreitet, da sie sehr skalierbar sind und bei Nicht-Bedarf einfach gelöscht werden können. |
| Webserver         | Ein Server oder ein Programm, der/das eine Ressource über das Internet zur Verfügung stellt.   |
| Wireframe         | Eine Skizze für eine Benutzerschnittstelle, die lediglich die Umrisse von Elementen zeigt und jegliche Details wie Farben und Schriften weglässt.  |