

HOCHSCHULE LUZERN

BACHELORARBEIT

JTI Pickup Station

Oliver Werlen

Betreut durch René Meier und Michael Handschuh
Experte Stefan Bernet

2. Juni 2021

Page

Bachelorarbeit an der Hochschule Luzern – Informatik**Titel:** JTI Pickup Station**Studentin/Student:** Oliver Werlen**Studiengang:** BSc Informatik**Jahr:** 2021**Betreuungsperson:** René Meier und Michael Handschuh**Expertin/Experte:** Stefan Bernet**Auftraggeberin/Auftraggeber:** JT International AG (Japan Tobacco International)**Codierung / Klassifizierung der Arbeit:**

- A: Einsicht (Normalfall)
- B: Rücksprache (Dauer: Jahr / Jahre)
- C: Sperre (Dauer: Jahr / Jahre)

Eidesstattliche Erklärung Ich erkläre hiermit, dass ich/wir die vorliegende Arbeit selbstständig und ohne unerlaubte fremde Hilfe angefertigt haben, alle verwendeten Quellen, Literatur und andere Hilfsmittel angegeben haben, wörtlich oder inhaltlich entnommene Stellen als solche kenntlich gemacht haben, das Vertraulichkeitsinteresse des Auftraggebers wahren und die Urheberrechtsbestimmungen der Fachhochschule Zentralschweiz (siehe Merkblatt «Studentische Arbeiten» auf MyCampus) respektieren werden.

Ort / Datum, Unterschrift _____

Ort / Datum, Unterschrift _____

Abgabe der Arbeit auf der Portfolio Datenbank:

Bestätigungsvisum Studentin/Student

Ich bestätige, dass ich die Bachelorarbeit korrekt gemäss Merkblatt auf der Portfolio Datenbank abgelegt habe. Die Verantwortlichkeit sowie die Berechtigungen habe ich abgegeben, so dass ich keine Änderungen mehr vornehmen kann oder weitere Dateien hochladen kann.

Ort / Datum, Unterschrift _____

Ort / Datum, Unterschrift _____

Verdankung

xxx

**Ausschliesslich bei Abgabe in gedruckter Form:
Eingangsvisum durch das Sekretariat auszufüllen**

Rotkreuz, den _____

Visum: _____

Hinweis: Die Bachelorarbeit wurde von keinem Dozierenden nachbearbeitet. Veröffentlichungen (auch auszugsweise) sind ohne das Einverständnis der Studiengangleitung der Hochschule Luzern – Informatik nicht erlaubt.

Copyright © 2021 Hochschule Luzern – Informatik

Alle Rechte vorbehalten. Kein Teil dieser Arbeit darf ohne die schriftliche Genehmigung der Studiengangleitung der Hochschule Luzern – Informatik in irgendeiner Form reproduziert oder in eine von Maschinen verwendete Sprache übertragen werden.

Zusammenfassung

Inhaltsverzeichnis

1 Problem und Vision	6
1.1 Problem	6
1.1.1 Versandkosten/Mindestbestellwert	6
1.1.2 Dauer bis Ware bei Endkonsumenten	6
1.1.3 Angebot nur in begrenztem Zeitraum möglich	6
1.2 Vision	6
2 Stand der Technik	7
2.1 MyPost 24	7
2.1.1 Allgemein	7
2.2 avec	8
2.2.1 avec now	8
2.2.2 avec Box	8
2.2.3 Ablauf Tabakkauf	9
2.3 Starbucks Progressive Web App	10
2.3.1 Standortsuche	10
2.3.2 Fazit	11
2.4 Fazit	11
3 Ideen und Konzepte	12
3.1 Vorwort	12
3.2 Systemarchitektur	12
3.2.1 Zusammenspiel der einzelnen Layer	14
3.2.2 Infrastruktur	14
3.3 Mobile First	15
3.4 Frameworks	16
3.4.1 Spring Boot	16
3.4.2 Angular	16
3.5 Weitere Technologien	16
3.5.1 Docker	16
3.5.2 MariaDB	16
3.5.3 Hibernate ORM	16
3.5.4 GitLab	17
3.5.5 Bezahlungsdienst	17
3.5.6 Alterverifikation	17
3.5.7 Karte	17
4 Methoden	18
4.1 SoDa	18
4.2 Domain Driven Design	19
4.2.1 Aufbau von Domain-Knowledge	19
4.2.2 Model Driven Design	19
4.3 CI/CD	20
4.3.1 build	20
4.3.2 package	20
4.3.3 deploy	20
4.3.4 Sequenzdiagramm	21
4.3.5 Testing	21

5 Realisierung	22
5.1 Initialisierungsphase	22
5.1.1 Kick-Off Meeting	22
5.1.2 Erstellen des Projektmanagementplans	22
5.1.3 Problem und Vision	22
5.1.4 Requirement Engineering	22
5.1.5 Stand der Technik	22
5.1.6 Meilenstein Abschluss Initialisierungsphase	22
5.2 Konzeptionsphase	23
5.2.1 Sprint 1	23
5.2.2 Sprint 2	25
5.2.3 Sprint 3	29
5.2.4 Sprint 4	31
5.2.5 Sprint 5	33
5.2.6 Sprint 6	37
5.2.7 Meilenstein Abschluss Bestellprozess und Zwischenpräsentation	45
5.3 Realisierungsphase	46
5.3.1 Sprint 7	46
5.3.2 Sprint 8	49
5.3.3 Sprint 9	57
5.3.4 Sprint 10	59
5.3.5 Meilenstein Abschluss Realisierungsphase	64
5.4 Sprint 11	64
5.5 Einführungsphase	65
6 Evaluation und Validation	67
6.1 Ziel der Arbeit	67
6.2 Validierung der Requirements	67
7 Ausblick	69
8 Verzeichnisse	70
9 Abkürzungsverzeichnis	74
A Testprotokolle	79
A.1 Testprotokolle Bestellung	79
A.2 Testprotokolle Abschluss Realisierungsphase	86
A.2.1 Altersverifikation	86
A.2.2 Station und Kartenfunktionalität	89
A.2.3 Inventur	92
A.2.4 Initialisierung neue Station	94
A.2.5 Abholung einer Bestellung	95
B Projektmanagementplan	96
B.1 Projektorganisation	96
B.1.1 Organisationsplan, Rollen, Zuständigkeiten	96
B.1.2 Projektstrukturplan	97
B.2 Projektführung	98
B.2.1 Rahmenplan	98
B.2.2 Meilensteine	99
B.2.3 Risikomanagement	100
B.2.4 Definition of done	102
B.3 Projektunterstützung	103
B.3.1 Tools für Entwicklung, Test und Abnahme	103

B.3.2 Konfigurationsmanagement	103
B.4 Teststrategie und Drehbuch	105
B.4.1 Teststrategie	105
B.4.2 Testdrehbuch	105
B.5 Bemerkungen	105
C System-Spezifikation	106
C.1 Systemübersicht	106
C.1.1 Systemarchitektur	106
C.1.2 Kontextdiagramm	107
C.2 Architektur und Designentscheide	108
C.2.1 Modelle und Sichten	108
C.2.2 Daten (Mengengerüst und Strukturen)	108
C.2.3 Entwurfsentscheide	109
C.3 Schnittstellen	113
C.3.1 Externe Schnittstellen	113
C.3.2 Benutzerschnittstellen	113
C.4 Environment-Anforderungen	113
C.4.1 Hardware	113
C.4.2 Software	114
C.4.3 Software	114
D Software Requirements Specification	115
D.1 Zweck	115
D.1.1 Zielgruppe	115
D.1.2 Produktumfang	115
D.1.3 Definitionen	115
D.1.4 Systemübersicht	115
D.1.5 Abhängigkeiten	115
D.2 Spezifische Anforderungen	116
D.2.1 Funktionale Anforderungen	116
D.2.2 Nicht funktionale Anforderungen	117
D.3 Änderungshistorie	118
D.4 Bemerkungen	118
E Sitzungsprotokolle	119
E.1 23.02.2021	119
E.1.1 Ordnungsaufruf	119
E.1.2 Teilnehmer	119
E.1.3 Genehmigung des Protokolls	119
E.1.4 Ankündigungen	119
E.1.5 Besprochene Punkte	119
E.1.6 Tagesordnung der nächsten Sitzung	120
E.2 04.03.2021	121
E.2.1 Ordnungsaufruf	121
E.2.2 Teilnehmer	121
E.2.3 Genehmigung des Protokolls	121
E.2.4 Ankündigungen	121
E.2.5 Besprochene Punkte	121
E.2.6 Tagesordnung der nächsten Sitzung	122
E.3 11.03.2021	123
E.3.1 Ordnungsaufruf	123
E.3.2 Teilnehmer	123
E.3.3 Genehmigung des Protokolls	123
E.3.4 Ankündigungen	123

E.3.5	Besprochene Punkte	123
E.3.6	Tagesordnung der nächsten Sitzung	123
E.4	18.03.2021	123
E.4.1	Ordnungsauftrag	123
E.4.2	Teilnehmer	124
E.4.3	Genehmigung des Protokolls	124
E.4.4	Ankündigungen	124
E.4.5	Besprochene Punkte	124
E.4.6	Tagesordnung der nächsten Sitzung	124
E.5	25.03.2021	125
E.5.1	Ordnungsauftrag	125
E.5.2	Teilnehmer	125
E.5.3	Genehmigung des Protokolls	125
E.5.4	Ankündigungen	125
E.5.5	Besprochene Punkte	125
E.5.6	Tagesordnung der nächsten Sitzung	126
E.5.7	Unterschriften	126
E.6	01.04.2021	127
E.6.1	Ordnungsauftrag	127
E.6.2	Teilnehmer	127
E.6.3	Genehmigung des Protokolls	127
E.6.4	Ankündigungen	127
E.6.5	Besprochene Punkte	127
E.6.6	Tagesordnung der nächsten Sitzung	127
E.7	15.04.2021	127
E.7.1	Ordnungsauftrag	127
E.7.2	Teilnehmer	128
E.7.3	Genehmigung des Protokolls	128
E.7.4	Ankündigungen	128
E.7.5	Besprochene Punkte	128
E.7.6	Tagesordnung der nächsten Sitzung	128
E.8	15.04.2021	129
E.8.1	Ordnungsauftrag	129
E.8.2	Teilnehmer	129
E.8.3	Genehmigung des Protokolls	129
E.8.4	Ankündigungen	129
E.8.5	Besprochene Punkte	129
E.8.6	Tagesordnung der nächsten Sitzung	129
E.9	21.04.2021	130
E.9.1	Ordnungsauftrag	130
E.9.2	Teilnehmer	130
E.9.3	Genehmigung des Protokolls	130
E.9.4	Ankündigungen	130
E.9.5	Besprochene Punkte	130
E.9.6	Tagesordnung der nächsten Sitzung	130
E.10	12.05.2021	131
E.10.1	Ordnungsauftrag	131
E.10.2	Teilnehmer	131
E.10.3	Genehmigung des Protokolls	131
E.10.4	Ankündigungen	131
E.10.5	Besprochene Punkte	131
E.10.6	Tagesordnung der nächsten Sitzung	131
E.11	17.05.2021	132
E.11.1	Ordnungsauftrag	132

E.11.2 Teilnehmer	132
E.11.3 Genehmigung des Protokolls	132
E.11.4 Ankündigungen	132
E.11.5 Besprochene Punkte	132
E.11.6 Tagesordnung der nächsten Sitzung	132
E.12 25.05.2021	133
E.12.1 Ordnungsaufruf	133
E.12.2 Teilnehmer	133
E.12.3 Genehmigung des Protokolls	133
E.12.4 Ankündigungen	133
E.12.5 Besprochene Punkte	133
E.12.6 Tagesordnung der nächsten Sitzung	133
E.13 01.06.2021	134
E.13.1 Ordnungsaufruf	134
E.13.2 Teilnehmer	134
E.13.3 Genehmigung des Protokolls	134
E.13.4 Ankündigungen	134
E.13.5 Besprochene Punkte	134
E.13.6 Tagesordnung der nächsten Sitzung	134
F Rahmenpläne	135
G Originale Aufgabenstellung	136
H Zwischenpräsentation	141

1 Problem und Vision

1.1 Problem

Der Onlinekauf ist beim Zigarettenkauf ein sehr selten genutzter Absatzweg. Vor allem die nachfolgenden Punkte sind verantwortlich für die seltene Nutzung dieses Angebots.

- Versandkosten/Mindestbestellwert
- Dauer bis Ware beim Endkonsumenten
- Angebot nur in begrenztem Zeitraum möglich

1.1.1 Versandkosten/Mindestbestellwert

Bei diversen Onlineshops kommen bei zu geringer Bestellmenge erhebliche Versandkosten hinzu. So kostet der Versand per Paket in der Regel 9 Franken. Bei Kioskolino ist der Versand ab einem Bestellwert von Fr. 139.00 portofrei [Genf, o.D.]. Bei Coop ist die Liefergebühr höher. Sie beträgt Fr. 17.90. Die Versandkosten nehmen mit zunehmendem Bestellwert ab. Ab Fr. 500.00 ist der Versand kostenlos [Coop, 2020].

1.1.2 Dauer bis Ware bei Endkonsumenten

Bei der Bestellung bei Kioskolino ist die Ware innerhalb von 1-3 Werktagen beim Konsumenten [Genf, o.D.]. Für die meisten Kunden dauert das zu lange. Coop verspricht die Lieferung am selben Tag. Dazu können bei der Bestellung verschiedene Zeitfenster ausgewählt werden, die Verfügbarkeit ist dabei von der Region abhängig. Die Ware muss aber frühzeitig bestellt werden, um die Lieferung am gleichen Tag garantieren zu können. Zudem bietet Coop auch die Möglichkeit, die Produkte direkt in der Filiale abzuholen [Coop, 2020].

Das Problem ist aber mit beiden Anbieter identisch. Es muss die Ware sehr früh bestellt werden. Zudem dauert die Lieferung immer zwischen 4 Stunden bis zu 3 Tagen. Eine Lieferung am Sonntag ist dabei nicht möglich, Samstags wird nur nachmittags geliefert.

1.1.3 Angebot nur in begrenztem Zeitraum möglich

Die bestellte Ware wird nur zu bestimmten Zeiten ausgeliefert. So ist eine Lieferung an Sonn- und Feiertagen nicht möglich.

1.2 Vision

Durch die JTI Pick-Up Station ist es dem Kunden möglich, seine Ware bequem im Onlineshop zu bestellen und ohne Wartezeit an der gewünschten Pick-Up Station abzuholen.

Die Artikel werden durch den Kunden an der gewählten Pick-Up Station bereitgestellt. Durch das Vorzeigen der Bestellbestätigung durch den Kunden wird der Artikel freigegeben und steht zur Abholung bereit.

Ein Mindestbestellwert muss nicht erreicht werden. Zudem werden keine zusätzlichen Gebühren verlangt.

Die Applikation soll durch eine einfache und intuitive Bedienung eine optimale Benutzerexperience bieten. Die Bestellung soll schnell und einfach ablaufen. Die Abholung soll in kurzer Zeit abgewickelt werden. Durch die Umsetzung als Progressive Web App ist die Applikation auch ohne aktive Internetverbindung nutzbar.

Durch das Verwenden eines bereits etablierten Altersverifikationsanbieters können die rechtlichen Bedingungen erfüllt werden. Der Bezahlvorgang wird durch einen etablierten Anbieter durchgeführt. Dies garantiert eine sichere und zuverlässige Bezahlabwicklung.

2 Stand der Technik

2.1 MyPost 24

2.1.1 Allgemein

Mit der Pick Post und MyPost 24 können Briefe und Pakete an die Pick-Up Station gesendet werden. Es besteht auch das Angebot, Pakete von einer Pick-Up Station zu versenden. Die Auswahl einer Pick-Up Station geschieht dabei mit der Angabe der entsprechenden Station. Der Dienst lässt sich somit bei jedem Onlineshop nutzen.

Die Abholung der Artikel muss innerhalb von 10 Tagen geschehen. Sobald die Artikel zur Abholung bereit sind, erhält der Kunde wahlweise eine Bestätigungsmail oder ein SMS. Mit dem darin enthaltenen Abholcode lässt sich die Ware abholen [Post-CH-AG, 2015].

Im Grossraum Luzern befinden sich acht MyPost 24-Abholstellen [AG, 2021a]. Das Design der Abholstelle sieht dabei konventionellen Briefkästen der Post sehr ähnlich.



Abbildung 1: MyPost 24-Abholstelle, Quelle: AG, 2021b

Die Lösung der Post behebt dabei aber nicht die in Kapitel 1 beschriebenen Probleme. Das Angebot richtet sich hauptsächlich an Personen, welche bei der Lieferung der Post nicht zuhause sind. Durch das Angebot kann ein Abholen an der Poststelle vermieden werden. Lieferzeit sowie Lieferkosten bleiben aber vorhanden. Der Bestell- und Bezahlvorgang wird beim jeweiligen Onlineshop durchgeführt.

2.2 avec

2.2.1 avec now

"— Dein Online Lieferservice von avec — Mit avec now haben wir rund um das beliebte Angebot unseres Convenience-Formats avec einen Online-Store lanciert. Zur Auswahl steht ein breites Convenience-Sortiment. Die bestellten Waren werden direkt in unseren Stores zusammengestellt und so schnell wie möglich ausgeliefert." [Valora, 2021e]

Auf der Website von avec now wird eine Lieferzeit von 60 Minuten angepriesen. Der Mindestbestellwert beträgt dabei Fr. 20.00 [Valora, 2021a]. Das Angebot gilt für einen grossen Teil des Sortiments, darunter auch Tabakwaren. Es befindet sich noch in der Pilotphase und wird nur im Raum Zürich angeboten [Valora, 2021f].

2.2.2 avec Box

Der Anbieter geht bei diesem Angebot einen neuen Weg. Das gesamte Einkaufen wird mittels App durchgeführt. Mittels dieser können Produkte dem Warenkorb hinzugefügt und so bezogen werden. In einer ersten Phase sind zu Stosszeiten Mitarbeiter präsent, um die Kunden beim Einkauf zu unterstützen.

Um Tabakprodukte zu beziehen, befindet sich in der App eine integrierte Altersverifikation. Die Tabakwaren werden im Store via Touchscreen ausgewählt. Durch die App wird eine Altersverifikation durchgeführt [Valora, 2021b].

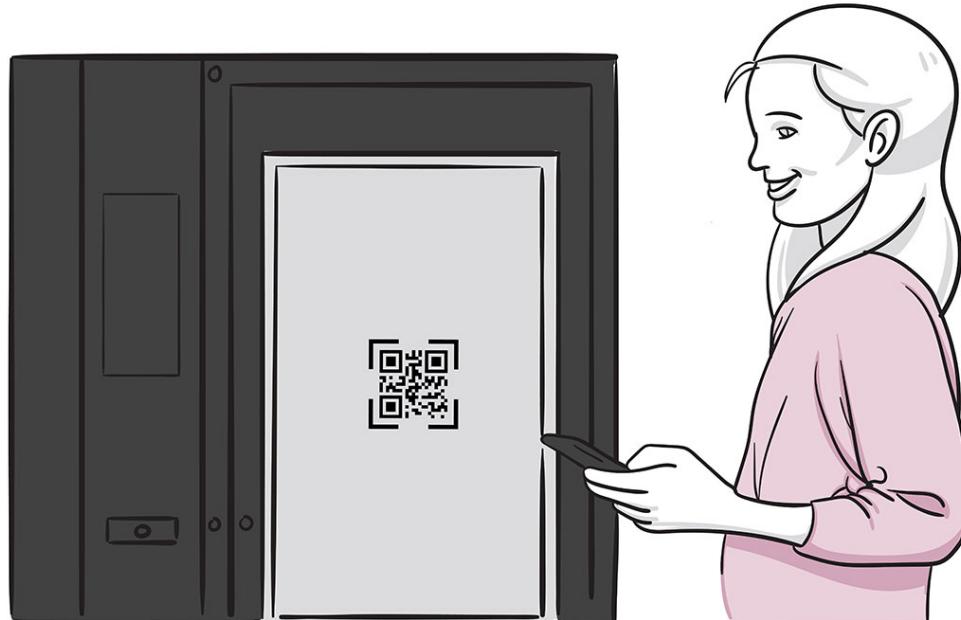


Abbildung 2: Tabakkauf avec box, Quelle: Valora, 2021b

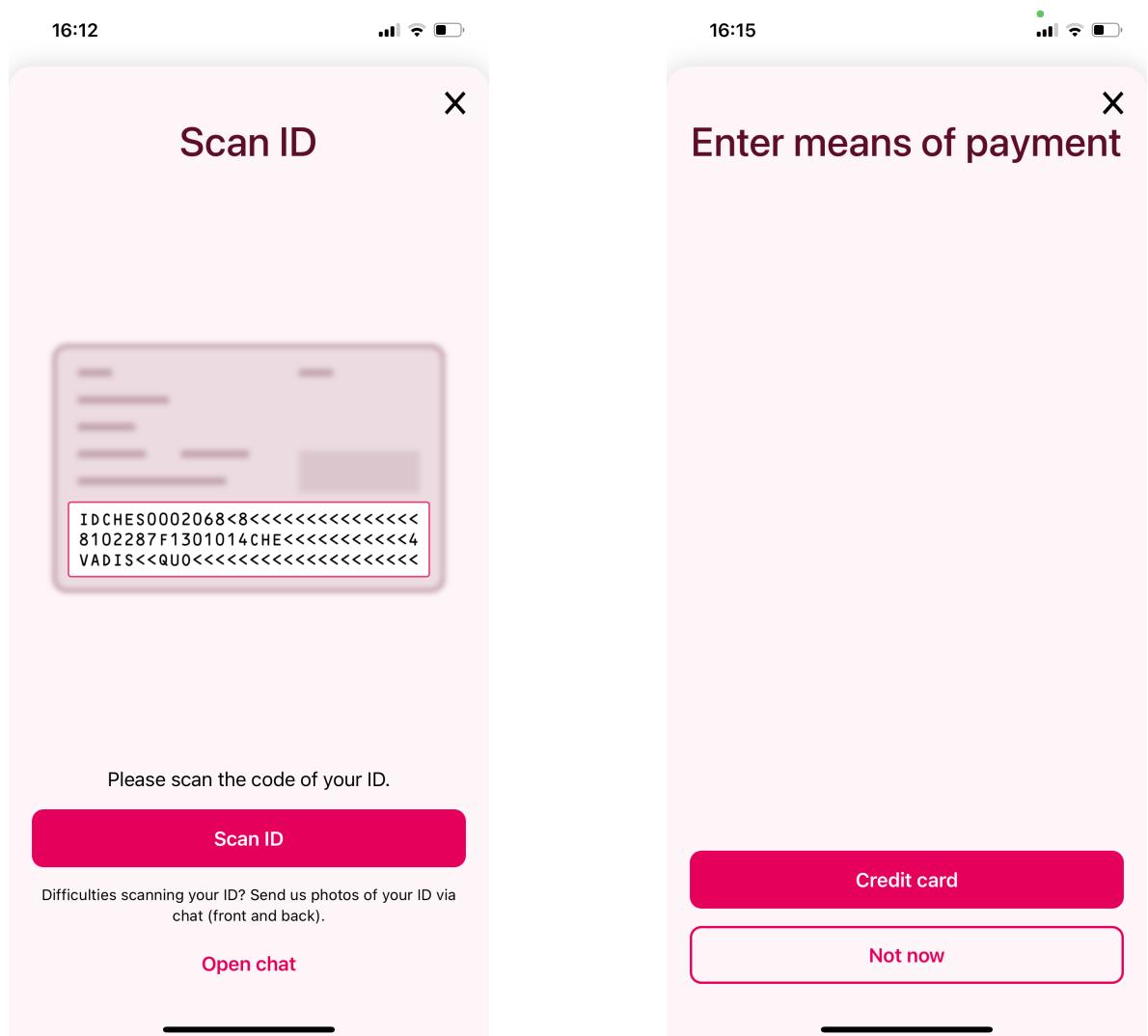
Nach momentanem Stand steht die avec box am Campus der ETH Zürich. Die Box ist von Montag bis Sonntag jeweils von 6:00 bis 22:00 in Betrieb. Es ist ein Rollout in weitere Regionen der Schweiz vorgesehen [Valora, 2021d].

Das Angebot von avec löst einige der genannten Probleme. So kann die Ware direkt bezogen werden. Es fallen keine Versandkosten an. Zur Zeit ist die avec box nur an einem Standort verfügbar, was die Verfügbarkeit erheblich einschränkt. Zudem ist sie nur von 6:00 bis 22:00 in Betrieb.

2.2.3 Ablauf Tabakkauf

Die avec box ist sehr ähnlich zur JTI Pick-Up Station. Aus diesem Grund wird der Registrierungsvorgang nachfolgend genauer betrachtet.

Registrierung Das Anlegen eines avec-Kontos verläuft analog zur Erstellung von anderen Accounts. Mittels Handynummer und Passwort wird der Account angelegt. Zur Verifikation wird ein Bestätigungscode an die Nummer gesendet. Dieser muss eingegeben werden. Anschliessend wird die Alterverifikation durchgeführt. Hierzu muss die Identitätskarte mit der Kamera eingelesen werden. In einem nächsten Schritt kann die Kreditkarte hinterlegt werden. Der Bezahlidienst wird dabei von Datatrans bereitgestellt. Anschliessend ist die Registrierung abgeschlossen und der Einkauf in der avec box könnte beginnen.



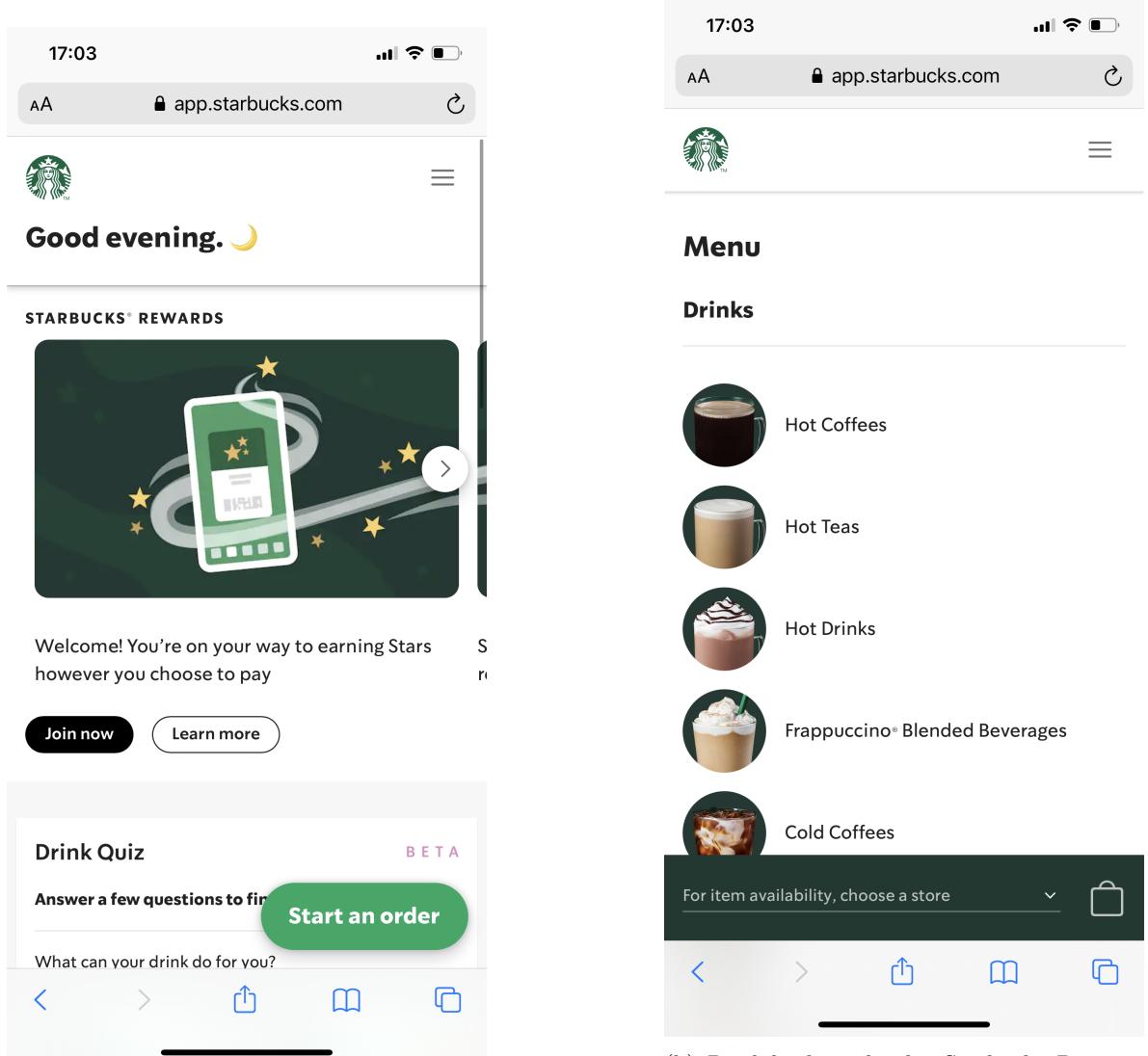
(a) Einlesen der Identitätskarte, Quelle: Valora, 2021c

(b) Einlesen der Kreditkarte in der avec App, Quelle: Valora, 2021c

Leider wird in der Applikation keine Auskunft über den Anbieter der Alterverifikation gegeben. Das Vorgehen ist sehr intuitiv und schnell.

2.3 Starbucks Progressive Web App

Gemäss SimiCart, 2021 befindet sich die Starbucks Progressive Web App in den Top-12 der besten Progressive Web App's. Die Applikation ermöglicht es dem Nutzer, die angebotenen Produkte zu bestellen und diese im Store abzuholen. Der Anwendungszweck ist somit ähnlich zur JTI-Pick-Up Station. Sie ist sehr nahe an einer nativen App, wodurch dem Benutzer die Bedienung sehr leicht fällt. Die Applikation reagiert sehr schnell, es sind keine Ladezeiten zu bemerken. Zudem ist die Progressive Web App auch offline nutzbar. Hierbei kommt es zwar zu Einschränkungen in der Nutzung, jedoch lässt sie sich weiterhin bedienen.



(a) Startseite der Starbucks Progressive Web App, Quelle: Starbucks, 2021a

(b) Produktübersicht der Starbucks Progressive Web App, Quelle: Starbucks, 2021b

2.3.1 Standortsuche

Die Applikation bietet auch ein Feature, um die nächstgelegene Starbucksfiliale anzuzeigen. Hierbei wird auf die aktuelle Position des Nutzers zugegriffen.

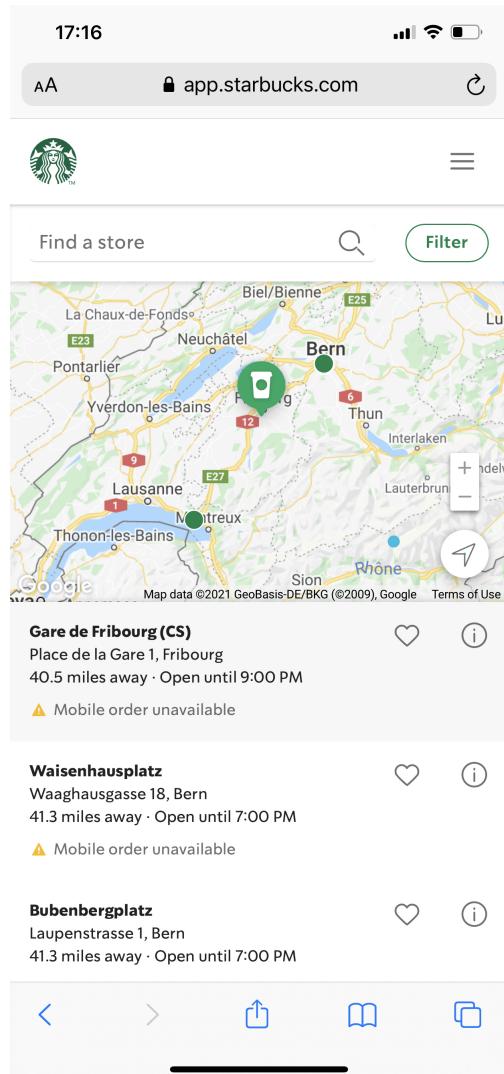


Abbildung 5: Standort in der Starbucks Progressive Web App, Quelle: Starbucks, 2021c

2.3.2 Fazit

Die Applikation von Starbucks zeigt auf, was eine Progressive Web App leisten kann. Sie dient als Vorbild für die Applikation der Japan Tobacco International (JTI)-Pick-Up Station.

2.4 Fazit

Es existieren diverse Produkte, welche einen ähnlichen Ansatz verfolgen wie dieses Projekt. Besonders hervorzuheben ist die avec box 2.2.2. Der Betreiber verfolgt hier einen ähnlichen Ansatz. Der Kaufvorgang bei Tabakwaren unterscheidet sich kaum von dem in diesem Projekt umzusetzenden. Durch die Analyse des dort verwendeten Vorgehens konnte ein guter Überblick gewonnen werden. Zudem konnte auch gesehen werden, wie die Integration der Altersverifikation umgesetzt wurde. Dieses Wissen ist für die spätere, eigene Umsetzung sehr wichtig.

Die Applikation von Starbucks liefert einen sehr guten Überblick über die Möglichkeiten von Progressive Web App's. Besonders designtechnisch ist diese Anwendung sehr wertvoll.

Die anderen analysierten Angebote lieferten keinen Mehrwert für das Projekt, da sie die gestellte Problematik nur bedingt oder gar nicht lösen.

3 Ideen und Konzepte

3.1 Vorwort

Durch die progressive Web App JTI Pick-Up Station ist es dem Kunden möglich, seine Ware bequem im Onlineshop zu bestellen und direkt und ohne Wartezeit an der gewünschten Pick-Up Station abzuholen. Durch den Prototyp sollen die Funktionalität und Zweckmässigkeit dieses für die Firma neuen Absatzkanals aufgezeigt werden. Im besten Fall findet die Applikation nicht nur in der Schweiz Verwendung, sondern wird von JTI auch in anderen Märkten weltweit eingesetzt. Das Hauptaugenmerk der Arbeit liegt auf der Implementierung eines Prototyps mit den folgenden Schwerpunkten: Bestellung, Kauf, Nutzererfassung, Suche nach Pick-Up Stations und der Abholung an der Station. Bei der Nutzererfassung muss das Alter des Nutzers verifiziert werden. Die Lösung soll so weit als möglich in die Projektpartner-Systeme integriert werden. Hinzu kommt die Recherche von artverwandten Technologien und das Requirements Engineering. Die BDA wird als interdisziplinäre Bachelorarbeit durchgeführt.

3.2 Systemarchitektur

Als Systemarchitektur stand die Erweiterbarkeit im Vordergrund. Allerdings sollte durch die Architektur die Applikation nicht unnötig komplex werden. Aus diesem Grund wurde bewusst gegen eine Microservicearchitektur entschieden. Die Umsetzung der Applikation mit Microservices würde zwar zu einer besseren Verteilbarkeit und Skalierung führen, der Aufwand der Umsetzung würde jedoch erheblich steigen.

Es wurde auf eine Schichtenarchitektur gesetzt. Die klassische logische drei Schichtenarchitektur wurde noch weiter verfeinert, final wurde eine sechs Schichtenarchitektur entworfen. Die Architektur wird zusätzlich in 3 physische Tier aufgeteilt.

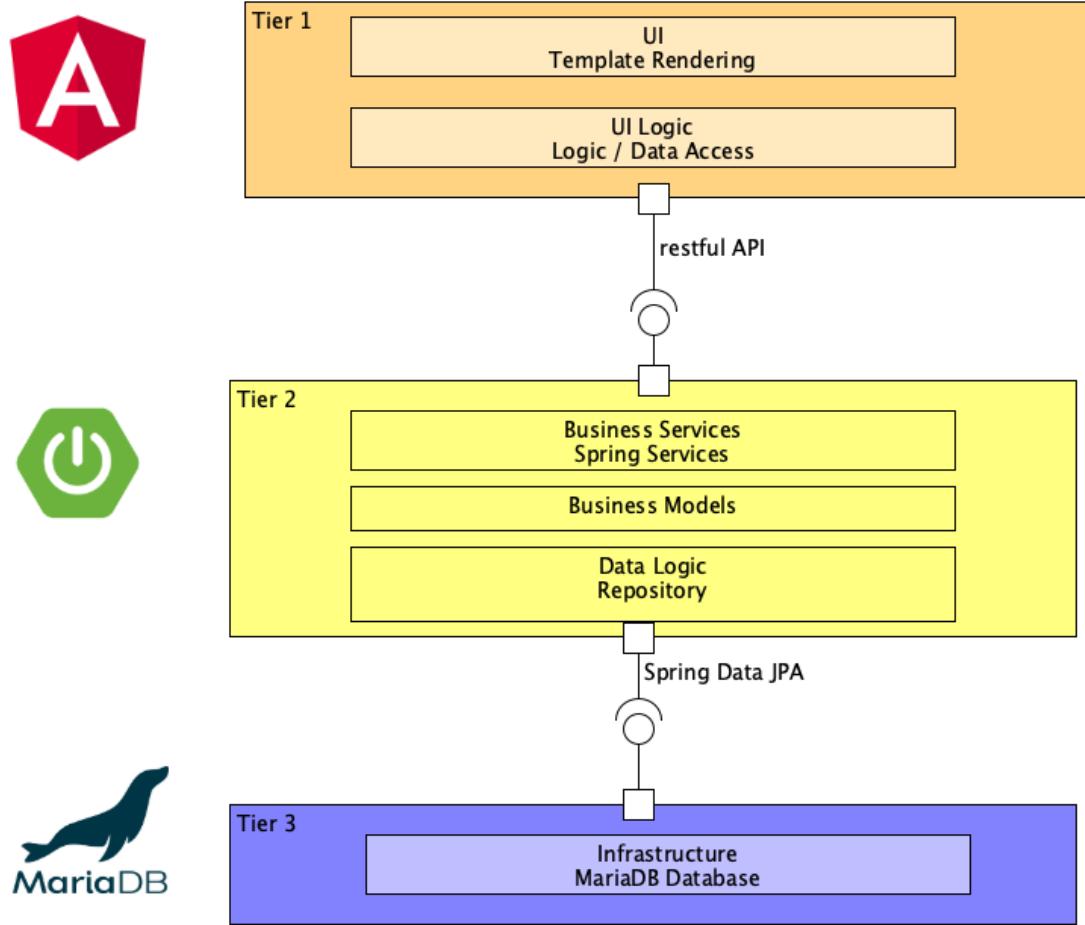


Abbildung 6: Architektur, Quelle: Autor

Die Architektur ermöglicht eine sehr gute Erweiterbarkeit. Zudem kann die Software verteilt werden und unabhängig voneinander skalieren. Durch die Aufteilung in sechs Tiers wird zudem vermieden, dass die einzelnen Tiers zu breit werden. Durch das Verwenden von Frameworks kann die Komplexität gering gehalten werden. Die Schnittstellen innerhalb der einzelnen Layer sind klar vorgegeben. Durch die Kommunikation via REST, bzw. Repositories sind die Komponenten untereinander austauschbar. Die Kopplung ist sehr gering.

UI Es wird eine Single-Page Applikation umgesetzt. Sie wird vorab im Browser geladen, später wird nur der entsprechende Inhalt neu aktualisiert. Das bringt eine verbesserte Nutzerexperience, da keine Abhängigkeiten zu den Serverladezeiten bestehen. Hingegen dauert das initiale Laden länger [Schulte, 2019].

UI Logic Aufgrund der gewählten Architektur werden die Daten der Applikation via Representational State Transfer (REST)-Schnittstelle geladen. Dies wird asynchron durchgeführt.

REST-Controller Die einzelnen REST-Controller definieren die Application Programming Interface (API). Es wurden die folgenden Punkte bei der Erstellung berücksichtigt:

- Konsistenz: Namensgebung und Regeln konsequent einhalten.
- Namensgebung: Einfache, eingängige, treffende Namen wählen.

- Verhalten: Klare Erwartungen erfüllen, ohne Nebeneffekte.
- Erweiterbarkeit: Offen für Weiterentwicklung (der API).
- Dokumentation: Einfache, hilfreiche, kompakte Dokumentation.
- Perspektive: Vom Anbieter für den Nutzer - es soll für den Nutzer einfach werden!
- KISS-Prinzip: Schnittstelle möglichst einfach halten.
- Sicherheit: Die Nutzung ist sicher zu gestalten.

[Gisler, 2020]

Es wird gemäss Richardson Maturity Model 72 eine REST-Schnittstelle von Level 3 angestrebt.

Business Services Die Business Services kommunizieren mit den Domänenmodellen. Weitere Details werden im Abschnitt 4.2.2 beschrieben.

Business Models Es handelt sich um die Entities des Projekts. Sie werden aus dem Domänenmodell erarbeitet und sind einzigartig. Weitere Details werden im Abschnitt 4.2.2 beschrieben.

Data Logic Zur Persistierung der Entities werden Repositories eingesetzt. Sie bilden die Schnittstelle zwischen Applikation und Datenbank.

3.2.1 Zusammenspiel der einzelnen Layer



Abbildung 7: Zusammenspiel von zentralen Layern, Quelle: Autor

3.2.2 Infrastruktur

Die gesamten Infrastruktur läuft im Enterprise Lab der Hochschule Luzern. Das Front- und das Back-End laufen auf unterschiedlichen virtuellen Maschinen. Die einzelnen Teilapplikationen sind als Docker-Container umgesetzt.

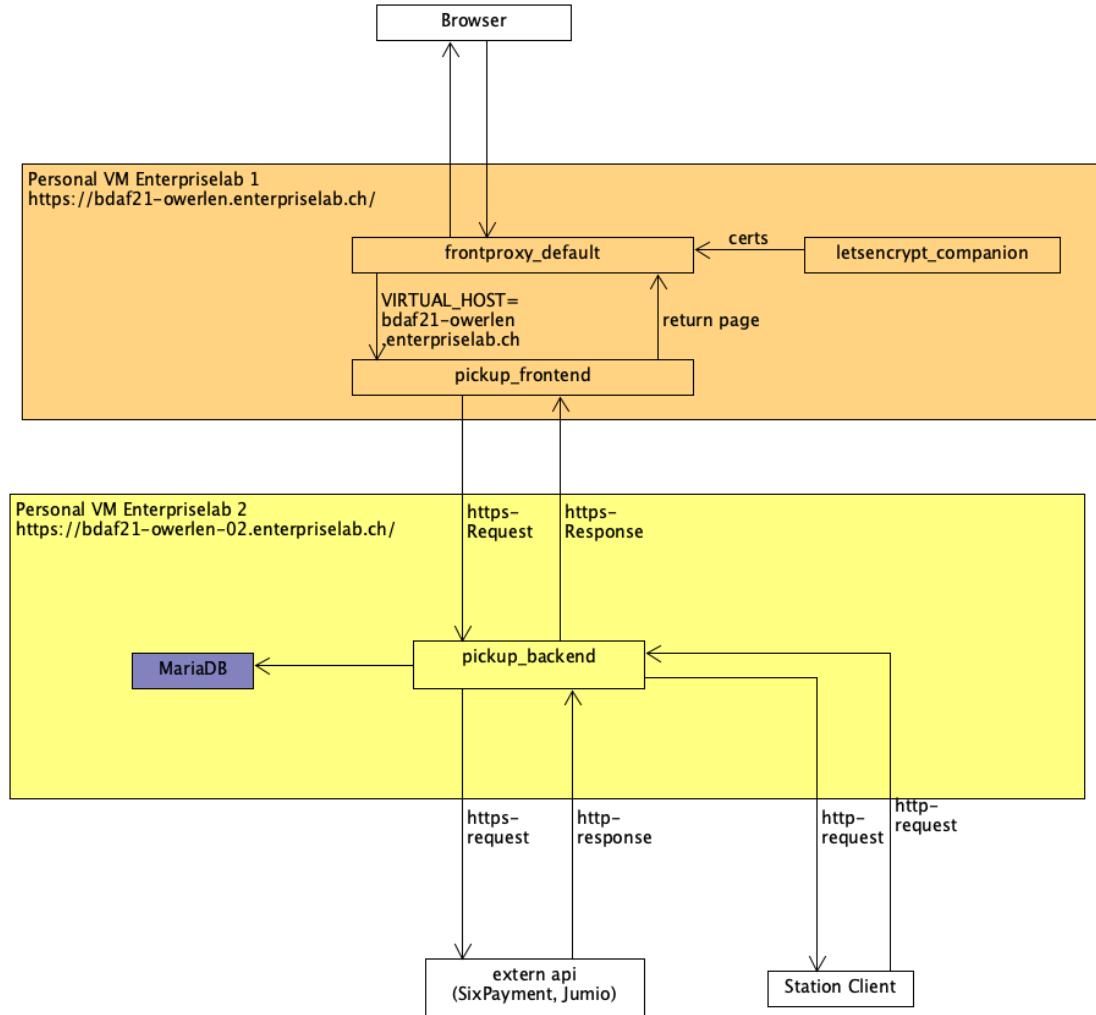


Abbildung 8: Container-Infrastruktur, Quelle: Autor

Die beiden Diagramme 6 und 8 sind farblich identisch gehalten. Gesamthaft umfasst das System sechs Container.

Der Station Client in diesem Diagramm befindet sich auf der physischen Station. Er ist das Bindeglied zwischen Informatik und Elektrotechnik und übernimmt Aufgaben wie die Produktausgabe oder das Inventar.

3.3 Mobile First

Es werden erheblich mehr Websiteaufrufe von mobilen, als von Desktop Geräten registriert. Der mobile First Ansatz nimmt sich dieser Thematik an und besagt, dass die Applikation in einem ersten Schritt für mobile Geräte entwickelt wird. Der Fokus liegt auf dem Design der Seite, wohingegen die Performance ebenfalls berücksichtigt wird.

Die Bedeutung von mobile First hat durch Google dazugewonnen. Der mobile Index ist der primäre Index, die mobile Friendliness der ausschlaggebende Punkt beim Ranking von Suchergebnissen [Ryte, 2021].

3.4 Frameworks

3.4.1 Spring Boot

Spring Boot baut auf dem Spring Framework auf. Dieses bietet sich an, um performante Enterprise-Applikationen mit Java zu erstellen. Spring Boot kann sehr einfach erweitert werden, sodass Spring Security, Spring MVC oder Spring Data eingesetzt werden können.

Spring Boot bringt einen integrierten Tomcat Server mit. Dies verringert den Konfigurationsaufwand. Es ist kein Einpacken in ein Web Applikation Archive nötig und kein zusätzliches deployen. Zudem wird das Debugging erleichtert.

Es übernimmt dabei einen grossen Teil der Beans-Konfiguration von Spring [Waldmann, 2020].

Spring Boot bietet dabei den grossen Vorteil, dass bereits Erfahrung in der Anwendung vorhanden ist. Es ist somit keine Einarbeitungszeit nötig, gängige Fehler können vermieden werden. Aus diesem Grund fiel die Entscheidung gegen Frameworks wie NodeJS.

3.4.2 Angular

Angular ist ein Application Design Framework, um effiziente Single Page Applikationen zu erstellen. Es basiert auf TypeScript [Böhm, 2017]. Zudem bietet Angular die Möglichkeit, als Progressive Web App genutzt zu werden [Google, 2021b]. Angular wurde dabei von Google entwickelt und bietet mit dem Material UI bereits viele Elemente, welche von nativen Android-Apps bekannt sind [Google, 2021a].

Wie auch bei Spring ist auch bei Angular bereits Projekterfahrung vorhanden. Ein Einarbeiten ist nicht mehr nötig, die gängigsten Funktionen sind bereits bekannt.

In vorhergehenden Projekten wurde das Frontend mit verschiedenen Technologien umgesetzt. Einerseits kam jQuery zum Einsatz, andererseits wurden auch Technologien wie React genutzt. Angular überzeugte von diesen Framework am meisten. Besonders durch die Unterstützung von TypeScript fiel die Wahl auf dieses Framework.

3.5 Weitere Technologien

3.5.1 Docker

Docker ist eine Containertechnologie. Sie erlaubt die Erstellung und den Betrieb von Linux-Containern. Die Container sind dabei sehr leichtgewichtig und modular.

Die einzelnen Teile der Applikation werden in verschiedenen Containern betrieben [RedHat, 2021]. Die Anwendung und Funktion von Docker wird in diversen Modulen an der Hochschule Luzern gelehrt. In vorhergegangenen Projekten wurde Docker bereits im selben Kontext genutzt.

3.5.2 MariaDB

MariaDB ist ein OpenSource Datenbankmanagementsystem. Es ist durch die Abspaltung von MySQL entstanden. Es handelt sich um ein Relationale Datenbanksystem. Aus Lizenzgründen wird in diesem Projekt MariaDB und nicht MySQL genutzt. Alternativ wäre auch ein Einsatz von PostgreSQL möglich. Der Unterschied zwischen PostgreSQL und MariaDB ist dabei nur marginal. Bei dieser Applikation wurde lediglich aus Erfahrung auf MariaDB gesetzt [DB-Engines, 2021].

Durch die Verwendung von Spring Data wäre es möglich, die Datenbank im Verlauf des Projektes auszutauschen.

3.5.3 Hibernate ORM

Hibernate ORM ist ein Object Relational Mapper. Er ermöglicht dem Entwickler, einfacher mit der Datenpersistierung umzugehen. Hibernate ist zudem ein Teil der Java Persistence API (JPA) und der Spring Data JPA. Ein wichtiger Punkt ist zudem die Performance, welche durch den Einsatz

des Lazy Loading Patterns erreicht wird. Die Initialisierung des Objektes wird dabei solange als möglich hinausgezögert. [Hat, 2021]

3.5.4 GitLab

Zur Versionsverwaltung kam GitLab zum Einsatz. Es wird dabei von der Hochschule Luzern zur Verfügung gestellt. Zudem sind zur Integration in die DevOps-Umgebung bereits GitLab Runner vorhanden, um Docker Images zu Builden.

3.5.5 Bezahlungsdienst

Als Bezahlungsdienst wurde Saferpay von Six Payment Services genutzt. Dabei wurde mit der Testversion gearbeitet, diese unterscheidet sich nicht von der Produktiven. Ein Wechsel wäre zudem innert kurzer Zeit durchführbar.

3.5.6 Alterverifikation

Bei der Altersüberprüfung wurde auf Jumio gesetzt. Das Produkt wurde vom Auftraggeber vorgegeben und eine Lizenz bereitgestellt.

3.5.7 Karte

Für die Implementierung der Kartenfunktionalität wurde Leaflet genutzt. Die Kartendaten stammen von OpenStreetMap. Zudem wurde Geoapify als MapsAPI eingesetzt.

4 Methoden

4.1 SoDa

Das Projektmanagement wird mit dem hybriden Projektvorgehen SoDa der Hochschule Luzern durchgeführt. SoDa wurde dabei bereits schon in vorhergehenden Softwareentwicklungsmodulen eingesetzt und hat sich hier bewährt. Die Userstories werden dabei auch im GitLab erfasst.

Abbildung 9: GitLab Board,
Quelle: Autor

Beim Sprint Review werden die Stories in reviewing mit den definierten Akzeptanzkriterien mit den Resultaten verglichen. Basierend darauf wird entschieden, ob an der User Story noch weiter gearbeitet werden, das heißt zurück zu doing oder die Story in done verschoben werden kann. Bei Beginn des nächsten Sprints wird das Vorgehen wiederholt.

Die einzelnen Items sind dabei priorisiert. Elemente mit einer hohen Einstufung werden dabei bei

der Bearbeitung vorgezogen.

4.2 Domain Driven Design

Domain Driven Design ist eine Herangehensweise an die Erstellung von komplexer Software. Sie befasst sich in den meisten Fällen mit Businessproblemen. Dieses Problem wird als Domäne bezeichnet. Die Software ist von Beginn an und durchgängig mit dieser Domäne verknüpft. Als Domäne wurde in diesem Projekt der Verkauf und die Abholung von Tabakprodukten identifiziert [Avram und Marinescu, 2006].

4.2.1 Aufbau von Domain-Knowledge

Um ein Domain-Wissen aufzubauen, wurde während dem gesamten Projekt enger Kontakt mit den Kontaktpersonen von JTI gepflegt. Besonders in der Initialisierungsphase wurde das Businessproblem besprochen und durch die Recherche von artverwandten Technologien erweitert. Als Resultat wurde die folgende Domäne erstellt:

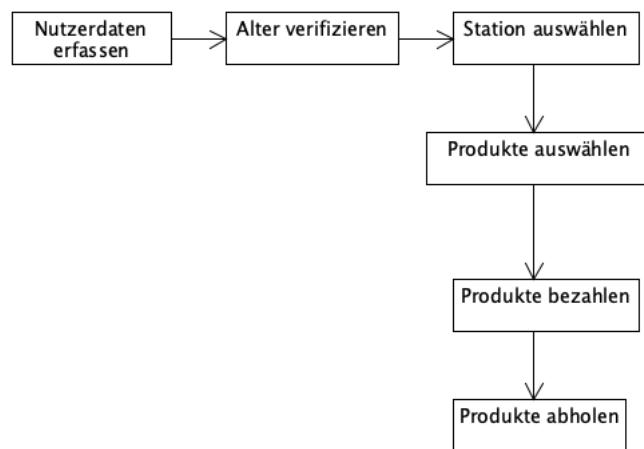


Abbildung 10: Domänenmodell, Quelle: Autor

4.2.2 Model Driven Design

Das erstellte Modell ist sehr stark mit der Domäne verknüpft. Basierend darauf wurde die Software designed und implementiert. Domain Driven Design setzt auf verschiedenste Pattern.

Entities Eine Entity hat immer eine Identität. Kann im System eindeutig identifiziert werden. Nicht alle Objekte in einem System sind Entities. Entities sind ressourcenintensiv, da immer sichergestellt werden muss, dass die Entity einzigartig ist.

Value Objects Es ist nicht in jedem Anwendungsfall nötig, das gesamte Objekt zu erhalten. Ein Value Object ist ein Objekt, welches nur einen Teilbereich aus einer Domäne beschreibt. Ohne eine Entity sind Value Objects sehr leichtgewichtig.

Es wird sehr stark empfohlen, Value Objects Immutable zu machen.

Value Objects können zudem Referenzen zu anderen Objekten enthalten. Date Transfer Object (DTO)'s entsprechen Value Objects.

Services Der Service bearbeitet Value Objects oder Entities. Dabei hat er drei Eigenschaften:

1. Der Service gehört zu einem Domänenkonzept, welches nicht direkt zum Domänenobjekt gehört.

2. Die Operation verweist auf andere Objekte in der Domäne.
3. Die Operation ist statuslos.

Repositories Das Repository entkapselt die Persistierung von Objekten und bildet somit die Schnittstelle zwischen Applikation und Datenbank.

4.3 CI/CD

Die gesamte Continous Integration and Continous Deployment (CI/CD) Pipeline wurde für dieses Projekt neu erstellt. Dabei besteht der Prozess aus drei Stages:

1. build
2. package
3. deploy

4.3.1 build

Abhängig vom Projekt wird das Projekt gebuilded. Bei der Java Applikation handelt es sich hier um ein maven-package, beim Angular Frontend um ein „ng build -prod“. Die resultierenden Artifakte werden für zwei Stunden im GitLab gespeichert.

4.3.2 package

In der Package Stage wird das Docker Image aus den vorherigen Artifakten erstellt. Dabei wird das Dockerfile des jeweiligen Produkts genutzt. Der Build wird von einem Shared Runner durchgeführt. Hier werden 4 vom Enterpriselab zur Verfügung gestellt. Anschliessend wird dieses in die Container Registry des GitLab Projekts gepushed.

4.3.3 deploy

Für das Deployment wurde auf den virtuellen Maschinen ein GitLab Runner installiert. Durch den deployment-Tag wird dieser adressiert. Auf diese Maschine wird ein Login ausgeführt und anschliessend das aktuellste Image heruntergeladen. Es wird der bestehende Container gelöscht und aus dem neuen Image ein neuer Container gestartet.

4.3.4 Sequenzdiagramm

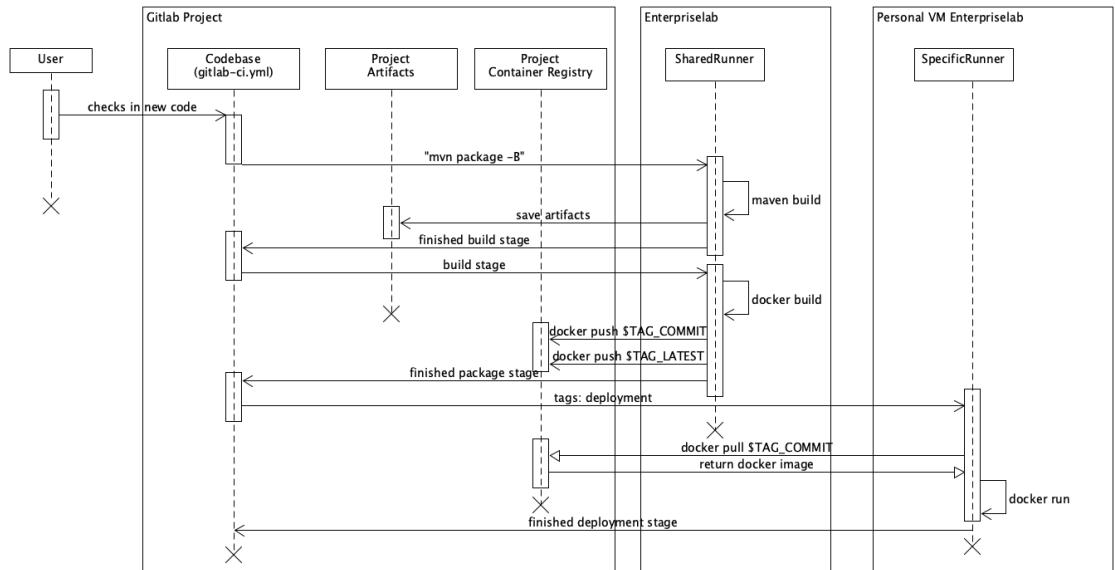


Abbildung 11: Ablauf der CI/CD Pipeline, Quelle: Autor

4.3.5 Testing

Das Projektziel ist ein Prototyp. Aus diesem Grund wurde der Fokus auf die Funktion gesetzt. Daher wurden die Unit- und Integrationstests nur konzeptuell umgesetzt. Bei Bedarf können diese erweitert werden.

5 Realisierung

5.1 Initialisierungsphase

5.1.1 Kick-Off Meeting

Das Kick-Off Meeting fand am 23.02.2021 als Zoom Meeting statt. Das Sitzungsprotokoll dazu ist im Kapitel E.1 zu finden.

Es waren bei diesem Meeting alle Projektbeteiligten anwesend. In erster Linie wurde von Herr Meier das genaue Vorgehen bei der Bachelorarbeit vorgestellt. Anschliessend stellte der Auftraggeber das Projekt genauer vor und zeigte dabei seine Erwartungen auf. Die Aufgabenstellung wurde finalisiert und von allen Projektbeteiligten akzeptiert. Zudem schlug der Betreuer vor, im Rythmus von zwei bis drei Wochen ein Meeting abzuhalten. Im Anschluss wurde ein Termin für das erste Meeting vereinbart.

5.1.2 Erstellen des Projektmanagementplans

Gemäss Software Development Agile (SoDa) wurde in einem ersten Projektschritt der Projektmanagementplan B erstellt. In diesem wurde der Rahmenplan erarbeitet. In diesem sind unter anderem die Meilensteine des Projekts dargestellt. Sie wurden in einem nächsten Schritt genauer spezifiziert und die Deliverables für den erfolgreichen Abschluss des Meilensteins definiert. Durch den Projektstrukturplan konnten die einzelnen Teilbereiche des Projekts aufgelistet werden. In der folgenden Risikoanalyse wurden die Risiken und entsprechenden Gegenmassnahmen erarbeitet. Als letztes wurde die Projektunterstützung genauer spezifiziert. Dabei wurden die zu verwendenden Tools sowie die Elemente der Konfigurationseinheit festgelegt.

Als letzter Teil des Projektmanagements wurde die Teststrategie und die Testdrehbücher formuliert. Die Testdrehbücher werden direkt in die Testprotokolle A integriert.

5.1.3 Problem und Vision

Das Kapitel 1 wurde zum Beginn der Initialisierungsphase bearbeitet. Die Hauptprobleme konnten dabei sehr schnell gefunden werden. Die Vision des Projektes konnte mithilfe der Aussagen des Auftraggebers im Kick-Off Meeting E.1 sehr gut beschrieben werden.

5.1.4 Requirement Engineering

Beim Requirements Engineering diente die IEEE Spezifikation 29148-2018 als Grundlage [Doran, 2018]. Um die Requirements zu finden, wurde in einem ersten Schritt eine Analyse der Aufgabenstellung durchgeführt. Ergänzt wurden diese durch Befragungen des Auftraggebers. Durch die so erlangten Informationen konnten die Anforderungen formuliert werden. In einem letzten Schritt wurden diese mit dem Auftraggeber besprochen. Hierbei wurden noch einige Anpassungen gemacht. Die Requirements Specification ist im Kapitel D zu finden.

5.1.5 Stand der Technik

In diesem Kapitel wurde eine Analyse der bestehenden, vergleichbaren Lösungen durchgeführt. Dabei wurde vor allem das Angebot der Post und von Valfloра genauer betrachtet. Die bekannte Progressive Web App von Starbucks wurde als Referenz-Progressive Web App (PWA) genutzt.

5.1.6 Meilenstein Abschluss Initialisierungsphase

Meilensteinbericht

Termin Meilenstein 2 Der Meilenstein 2 ist am 07.03.2021 abgeschlossen und somit pünktlich fertiggestellt worden.

Beschreibung Meilenstein 2 Die Beschreibung des Meilensteins ist im Abschnitt B.2.2 ersichtlich.

Meilensteinziele/Vorgaben Das übergeordnete Ziel dieses Meilensteins ist die Fertigstellung der Initialisierungsphase. Hierzu war die Auslieferung der nachfolgenden Artefakte notwendig:

- Projektmanagementplan
- Systemspezifikation
- Anforderungsliste

Zusätzlich wurden bereits die Kapitel 1 und 2 fertiggestellt.

Meilensteinzielerreichung Es konnten alle geforderten Artefakte geliefert werden. Die Artefakte wurden bereits mit der Betreuungsperson im Meeting E.3 besprochen und konnten abgenommen werden. Der Meilenstein wurde erfolgreich erreicht.

Fazit Es wurden alle Artefakte erarbeitet. Der Meilenstein wurde somit erreicht und es kann weiter nach Plan gearbeitet werden.

5.2 Konzeptionsphase

5.2.1 Sprint 1

User Story	Number
Das System ist auf eine physische Pick-Up Station abgestimmt.	F.2

Tabelle 1: User Stories Sprint 1, Quelle: Autor

Spezifikation der Schnittstelle zur Abholung Um eine Abholung der Produkte zu bekommen, braucht es eine Kommunikation zwischen Progressive Web App und Pick-Up Station. Um diese Schnittstelle genauer spezifizieren zu können, war es in einem ersten Schritt nötig, eine entsprechende Übertragungstechnologie festzulegen. Bei der Auswahl war dabei die Kompatibilität mit verschiedenen Geräten und Browsern ausschlaggebend.

NFC Als eine erste Idee wurde Near Field Communication (NFC) analysiert. NFC eignet sich dabei ideal für die Übertragung von geringen Datenmengen. Dabei funktioniert es bis zu einer Distanz von 10cm. Die Umsetzung wäre dabei sehr einfach mittels einem Raspberry Pi umsetzbar. Dabei könnte die gesamte Übertragung von der Informatik übernommen werden, eine weitere Schnittstelle zwischen Elektrotechnik und Informatik könnte vermieden werden. Die Technologie ist sehr robust. Ein Überkleben oder Zerkratzen des Lesers hat keinen Einfluss auf dessen Funktionalität [congstar, 2021]. Beliebte Bezahlmethoden wie Apple Pay oder vergleichbare nutzen NFC für die Übertragung zwischen Terminal und Smartphone. Der Abholvorgang wäre sehr einfach und schnell, da nur ein kurzer Kontakt mit dem Smartphone bereits ausreicht.

Apple schränkt den Zugriff von Webpages auf Standort- und Hardwaredienste unter iOS-Geräten sehr stark ein. Offiziell begründet wurde dies durch die Einschränkung von footprinting und des damit verbundenen Nutzertrackings. Für dieses Projekt ausschlaggebend ist vor allem das Entfernen des NFC- und Bluetooth-Supports. Somit ist es auch Progressive Web App's nicht mehr möglich, unter iOS auf die genannten Funktionen zuzugreifen [Apple, 2021].

Die Web-NFC-API wird zusätzlich bislang nur von Google Chrome unterstützt. Dies würde die Nutzbarkeit der Applikation sehr stark einschränken [Mozilla, 2021]. Aus diesem Grund eignet sich NFC nicht für die Verwendung in diesem Projekt.

QR-Code Die Restriktionen von Apple schränken die geeigneten Technologien sehr stark ein. Es bleibt nur noch der Zugriff auf die Kamera, um mit der Pick-Up Station zu kommunizieren. Ursprünglich war dies vom Auftraggeber nicht gewünscht. Das Verwenden einer anderen Technologie würde aber zu erheblichen Einschränkungen führen, sodass die Progressive Web App mit vielen Geräten unbrauchbar wäre.

Daher musste ein Konzept entwickelt werden, um die Warenausgabe mittels QR-Code auszulösen. Dabei sind zwei Ansätze möglich:

- Fixer QR-Code auf Pick-Up Station, wird von Progressive Web App eingelesen.
- Variabler QR-Code in Progressive Web App, wird von Pick-Up Station eingelesen.

Aus Hardware technischer Sicht ist die Umsetzung der ersten Lösung bedeutend einfacher umzusetzen, bietet aber auch erhöhtes Fehlerpotential. Ein Überkleben des QR-Codes auf der Station würde die gesamte Station unbrauchbar machen. Eine Abholung der Bestellungen wäre nicht mehr möglich.

Die zweite Variante ist Hardware technisch anspruchsvoller. Auf der Pick-Up Station muss ein optischer Leser verbaut werden. Mit diesem kann der QR-Code aus der Progressive Web App eingelesen und die Bestellung ausgegeben werden.



Abbildung 12: Beispieldarstellung QR-Code auf Gerät,
Quelle: tagmotion, 2018

Eine Grafik zum zweiten Lösungsansatz ist im Kapitel 2.2.2 zu finden. Die genaue Wahl wird im Meeting mit dem Auftraggeber besprochen und abgesegnet.

Sprintreview Sprint 1 Im Sprint 1 wurde eine genauere Analyse der Schnittstelle zur Produktabholung durchgeführt. Dabei müssen im Meeting von dieser Woche die gefundenen Lösungen mit dem Auftraggeber besprochen werden. Die User Story kann erst zu Beginn des nächsten Sprints abgeschlossen werden.

Meilensteinbericht

Termin Meilenstein 3 Der Meilenstein 3 ist am 18.03.2021 abgeschlossen und somit mit drei Tagen Verspätung fertiggestellt worden.

Beschreibung Meilenstein 2 Die Beschreibung des Meilensteins ist im Abschnitt B.2.2 ersichtlich.

Meilensteinziele/Vorgaben Das übergeordnete Ziel dieses Meilensteins ist die Fertigstellung der Initialisierungsphase. Hierzu war die Auslieferung der nachfolgenden Artefakte notwendig:

- Projektmanagementplan
- Systemspezifikation
- Anforderungsliste

Zusätzlich wurden bereits die Kapitel 1 und 2 fertiggestellt.

Meilensteinzielerreichung Es konnten alle geforderten Artefakte geliefert werden. Die Artefakte wurden bereits mit der Betreuungsperson im Meeting ?? besprochen und konnten abgenommen werden. Der Meilenstein wurde erfolgreich erreicht.

Fazit Es wurden alle Artefakte erarbeitet. Der Meilenstein wurde erreicht und es kann weiter nach Plan gearbeitet werden.

5.2.2 Sprint 2

User Story	Number
Das System ist auf eine physische Pick-Up Station abgestimmt.	F.2
Das System bietet dem Kunden die Möglichkeit, verschiedene Produkte zu bestellen.	F.6
Das System muss über eine CI/CD Pipeline verfügen.	L.6
Das System muss via HTTPS kommunizieren.	L.3

Tabelle 2: User Stories Sprint 2, Quelle: Autor

CI/CD Pipeline Nach Absprache mit dem Auftraggeber im Meeting wurde entschieden, dass der Prototyp im Enterprise Lab laufen soll. Daraufhin wurde eine Maschine beantragt. Es handelt sich hierbei um ein Ubuntu 16.07 LTS. Die Applikationen sollen als verschiedene Docker Container betrieben werden.

Zuerst wurde geplant, die Pipeline wie im offiziellen Tutorial des Enterprise Lab zu erstellen. Auf Anfrage wurde jedoch ein anderes Vorgehen empfohlen. Nachfolgend wird die Antwort zitiert.

"Wenn es dein Ziel ist eine Spring Boot Applikation zu bauen und dann auf der VM zu deployen dann würde den Container auf den Shared Runner unserer GitLab Instanz bauen lassen und in die Container Registry deines Projekts pushen. Für die Deploy Stage der CI/CD Pipeline kannst du deine VM als privaten GitLab Runner registrieren und so ohne SSH login den Container von der Registry pullen und laufen lassen. Die SSL Termination mit Lets Encrypt würde ich mit einem separaten nginx Container lösen der reverse proxy spielt. Dieser kann dann einfach laufen und muss für Änderungen an der Spring Boot Applikation auch nie modifiziert werden. Der Vorteil im Vergleich zur Docker Übung ist, dass hier alle Hosts von der GitLab CI/CD Pipeline kontrolliert werden. In der Übung ist der docker-cloud-exercise Host abgekapselt und pullt mit Watchtower einfach blind das neuste Image von einer Registry. Dieser Aufbau macht IMO mehr Sinn wenn man einfach Container Images von dritten konsumiert, aber ist weniger elegant wenn man selbst Kontrolle über die Source und CI/CD Pipelines hat." [von Uslar, 2021a]

In diesem Auszug aus der Email vom Enterprise Lab-Mitarbeiter Cyrill von Uslar sind sehr viele Informationen enthalten. Es war ein mehrmaliges Durchlesen nötig, um sich darunter etwas vorstellen zu können. Die Pipelineerstellung wurde in die folgenden Punkte aufgeteilt und umgesetzt.

- Builden auf dem Shared Runner
- Pushen in die Container Registry des Projekts
- VM als privaten Runner registrieren und deployen
- nginx-Server als reverse Proxy für SSL Termination

Builden auf dem Shared Runner In diesem Projekt wird die Pipeline für zwei Projekte aufgesetzt. In einem ersten Schritt wurde dies nur für die Spring Applikation durchgeführt. Das Vorgehen unterscheidet sich dabei nur im Buildprozess. Um die Spring Applikation zu erstellen, wurde das Dockerfile identisch zum Spring Boot Docker-Tutorial aufgebaut [VMWare, 2021].

```
FROM maven:3.6.3-jdk-11-slim
ARG JAR_FILE=target/*.jar
COPY ${JAR_FILE} app.jar
ENTRYPOINT ["java","-jar","/app.jar"]
```

Java 11 wurde sehr bewusst gewählt. Zu diesem Zeitpunkt war Java 11 die aktuellste Version, welche eine slim-Edition des Images anbot. Das wirkt sich erheblich auf die Imagegrösse aus, weshalb dieser Kompromiss eingegangen wurde. Zum Erstellen des gitlab-ci.yml-Files wurde eine Anleitung von GitLab genutzt. Diese war jedoch nicht mehr ganz aktuell. Das Deployment wird hier zudem auf ein Kubernetes Cluster durchgeführt und musste entsprechend angepasst werden. Es konnte nur der Build-Teil übernommen werden. [Lenzo, 2016]

GitLab stellt zum Erstellen von Docker-Images bereits mehrere Shared-Runner zur Verfügung.

Shared runners

These runners are shared across this GitLab instance.

The same shared runner executes code from multiple projects, unless you configure autoscaling with [MaxBuilds](#) set to 1 (which it is on GitLab.com).

[Disable shared runners](#) for this project

Available shared runners: 5

● e1f64da7

gitlabrunner-02

#37

[docker](#) [runner02](#) [test-runner](#) [ubuntu 18.04](#)

● piYF7yKo

gitlabrunner-06

#56

[docker](#) [runner06](#) [ubuntu 18.04](#)

● 8bU3YzuD

gitlabrunner-01

#62

[BigBuild](#) [docker](#) [runner01](#) [ubuntu 18.04](#)

● cf1ce3b4

gitlabrunner-03

#49

[docker](#) [runner03](#) [ubuntu 18.04](#)

● ydSz8fXG

gitlabrunner-05

#55

[docker](#) [runner05](#) [ubuntu 18.04](#)

Abbildung 13: Verfügbare Shared-Runner von GitLab, Quelle: Autor

Der jeweilige Runner wird dabei mittels eines fair usage algorithmus zugewiesen. Zudemachtet der Runner auf die verwendeten Tags. In diesem expliziten Beispiel sind mehrere Runner mit dem Docker-Tag versehen. Daher kann nicht sicher gesagt werden, welcher Runner den Docker Build durchführt.

Builden in die Registry des Projekts Um das erstellte Image in die Registry des Projekts zu pushen, werden von GitLab bereits die einzelnen Befehle vorgegeben. Diese können in die Package-Stage integriert werden.



Abbildung 14: CLI Commands GitLab Container Registry, Quelle: Autor

Das entsprechende Image wird in die interne Container Registry des GitLab Projekts gepushed. Dadurch kann auf das Verwenden eines Drittdienstes wie DockerHub verzichtet werden. Beim Erstellen werden immer zwei Images in die Registry geschrieben. Als Tag wurde bei einem Image latest genutzt, beim Anderen der Hash des Commits. Beim Deployment wird dann immer das Image mit dem aktuellen Commit-Hash verwendet. Dieser Mechanismus dient dazu, dass alle Versionen immer vorhanden sind. Latest zeigt dabei immer auf die aktuelle Version.

VM als privaten Runner registrieren und deployen Um das Deployment ohne Watchtower durchführen zu können, wurde auf die virtuelle Maschine als Docker-Runner hinzugefügt. Es handelt sich somit dabei um einen specific Runner. GitLab Runner kann dabei einfach mittels Paketmanager auf Ubuntu installiert werden. Anschliessend musste dieser noch konfiguriert werden. Dazu musste der Token sowie die URL von GitLab dem Runner hinzugefügt. Der Runner ist im gleichnamigen Bereich des Projekts zu finden. Zudem wurde die Option „Lock to current project“ entfernt, sodass dieser Runner auch direkt für das Angular Projekt genutzt werden kann. Um ein SSH-Login zu vermeiden, wurde der Executor als Shell-Executor definiert. Bei einem SSH-Executor wäre vorgängig die Verbindung via SSH nötig gewesen. Die entsprechenden Befehle konnten in das gitlab-ci.yml integriert werden. Es bestand zunächst das Problem, dass die Berechtigung fehlte. Erst nach dem Login mittels GitLab CI-Token konnte der Container erfolgreich heruntergeladen und gestartet werden.

Build und Deployment des Frontends Um mit dem nächsten Schritt weiterzufahren, musste zuerst das Deployment des Angular Projekts konfiguriert werden. Hierbei liegt der Hauptunterschied in dem Build-Prozess. Es ist wichtig, die beim Build entstehenden Artifakte in GitLab zu speichern. Nur so kann auf ein erneutes Builden beim Erstellen des Docker Containers verzichtet werden. Zudem werden die Node-Module in den Cache gespeichert. Die nachfolgenden Befehle sind identisch zum Vorgehen bei Spring.

nginx als Reverse Proxy Um den nginx-Server gegen Zugriffe von Extern zu schützen, wurde auf einen Reverse Proxy gesetzt. Der Reverse Proxy stellt dabei die einzige Verbindung ins öffentliche Netz dar. Zudem übernimmt er die Zertifikatsverwaltung des Frontends. Alternativ wäre auch ein Caching möglich, auf die Umsetzung wurde für diesen Anwendungsfall jedoch verzichtet. [KnowHow, 2020]

Die Umsetzung wurde analog zum Tutorial von Alexander Bohndorf durchgeführt [Bohndorf, o.D.]. Dabei wurde der nginx-Proxy von jwilder sowie die damit kompatible letsencrypt companion verwendet. Die Umsetzung wurde mittels docker-compose Files durchgeführt.

Abschliessende Bemerkungen Das Erstellen der CI/CD Pipelines des Projekts verlief problemlos. Durch die Grundlage des Enterprise Labs und der sehr guten Dokumentation von GitLab konnte dies sehr schnell umgesetzt werden. Durch das Hinzufügen des Reverse Proxies konnte die Sicherheit des Systems massiv erhöht werden. Zusätzlich wurde so auch gleich die Auslieferung via HTTPS hinzugefügt sowie für ein durchgehend gültiges Zertifikat gesorgt.

Es wurde auf zusätzliche Test oder Codeverification-Stages verzichtet. Die fertigen Konfigurationen sind im GitLab Projekt enthalten und können eingesehen werden.

Bestellen von Produkten Um eine Bestellung durchführen zu können, wurde im ersten Schritt das Article Object definiert. Basierend auf diesem wurde das passende DTO definiert. Um das Mapping zwischen DTOs und Objekt zu vereinfachen, wurde der Object Mapper modelmapper genutzt. Es direkt mit Hypermedia as the Engine of Application State (HATEOAS) gearbeitet. Der Controller stellt dabei die gewohnten CRUD-Operationen zur Verfügung.

Abstimmung auf physische Pick Up Station Im Meeting von dieser Woche wurden dem Auftraggeber die beiden in Kapitel 5.2.1 erarbeiteten Lösungen vorgestellt. Der Entscheid fiel zugunsten der ersten Alternative. Dabei befindet sich der QR-Code fix auf der Station. Diese ist deutlich robuster und einfacher umzusetzen.

Die Schnittstellen zwischen Elektrotechnik und Informatik sind essentiell für die Funktionalität des Endprodukts. Aus diesem Grund wurde in diesem Sprint ein Meeting zwischen den Projektbeteiligten einberufen. Besonders das Senden der Ausgabeanforderung führte dabei zu Problemen. Hierbei soll auf einen Busy-Waiting Ansatz verzichtet werden. Jedoch soll die Lösung auch sehr energieparend und effizient umsetzbar sein. Da beide Projektmitarbeiter keine Erfahrung im Internet of Things (IoT)-Umfeld besitzen, wurde hier der Betreuer Michael Handschuh um Hilfe gebeten.

Sprintreview Sprint 2 In Sprint 2 konnten nicht alle User Stories vollständig erfüllt werden. Es werden zwei von drei User Stories im nächsten Sprint weiter bearbeitet. Die Umsetzung der CI/CD Pipeline ist hingegen abgeschlossen. Dies war auch die Hauptarbeit in diesem Sprint. Für die Spezifizierung der Schnittstelle wird beim Meeting mit dem Betreuer eine geeignete Lösung erarbeitet. Bei dem Bestellprozess ist bereits die Abfrage von Produkten an der API möglich. In einem nächsten Schritt wird das passende Frontend entworfen und umgesetzt.

5.2.3 Sprint 3

User Stories In diesem Sprint wurden keine neuen User Story zum Sprint Backlog hinzugefügt. Es wird weiterhin an den beiden verbleibenden User Stories gearbeitet.

Abstimmung auf eine physische Pick Up Station Bei der Abstimmung auf die physische Pick Up Station musste das geplante Vorgehen mit dem Projektbetreuer besprochen werden. Im Meeting E.5 wurde eine geeignete Lösung gesucht. Dabei standen zwei Möglichkeiten zur Auswahl.

WebHooks WebHooks erfüllt alle Anforderungen, welche zur Kommunikation zwischen API und PickUp Station benötigt werden.

Allerdings ist noch keine Erfahrung mit WebHooks vorhanden. Jedoch kann zur Erstellung auch Node.js mit Express verwendet werden. Dieses ist bekannt und wurde auch schon in anderen Projekten genutzt.

Im späteren Projektverlauf wurde immer deutlicher, dass das Erstellen eines eigenen WebHooks-Endpoints sehr schlecht dokumentiert ist. In den meisten Beispielen wird die Anbindung an einen bereits bestehenden Endpoint umgesetzt. Verfügbare Libraries lieferten nicht den gewünschten Effekt. Es wurde eine eigene Implementation umgesetzt, welche in Kapitel 5.3.2 beschrieben wird.

Produktbestellung In diesem Sprint wurde an der Produktbestellung gearbeitet. Als erstes wurde das Darstellen von Produkten ins Auge gefasst. Auch wurde die Grundstruktur der Angular Applikation erstellt. Es wurde strikt nach dem MobileFirst gearbeitet. Durch Schematic konnte das responsive Menü sowie die Anzeige für Produkte sehr schnell erstellt werden. Im Anschluss wurde das Menü mit den gewünschten Inhalten gefüllt.

Bei den Produkten wurde das Card Element von Angular Material gesetzt. Dieses wurde in ein Grid Layout verpackt. Damit die Seite auf allen Geräten optimal aussieht, wird das Layout basierend auf der Displaygrösse angepasst. Um dies mit TypeScript umzusetzen, wurde ein Breakpoint Observer eingesetzt. Mithilfe von diesem wird die Anzahl Spalten und die Spaltenhöhe abhängig von der Geräteauflösung festgelegt. Sie wird beim Laden der Seite definiert. Es ist hier kein Listener Mechanismus nötig, da sich die Auflösung nicht während der Laufzeit ändert. Um die Anzahl Spalten bei einem Wechsel in den Landscape Modus anzupassen, wurde ein neuer Observer eingesetzt. Dieser legt die entsprechende Anzahl fest.

```
this.layoutChangesTabletLandscape.subscribe(result => {
  if(result.matches){
    this.cols = 4
    this.rowHeight = "400px"
  }
});
this.layoutChangesTabletPortrait.subscribe(result => {
  if(result.matches){
    this.cols = 3
    this.rowHeight = "350px"
  }
});
```

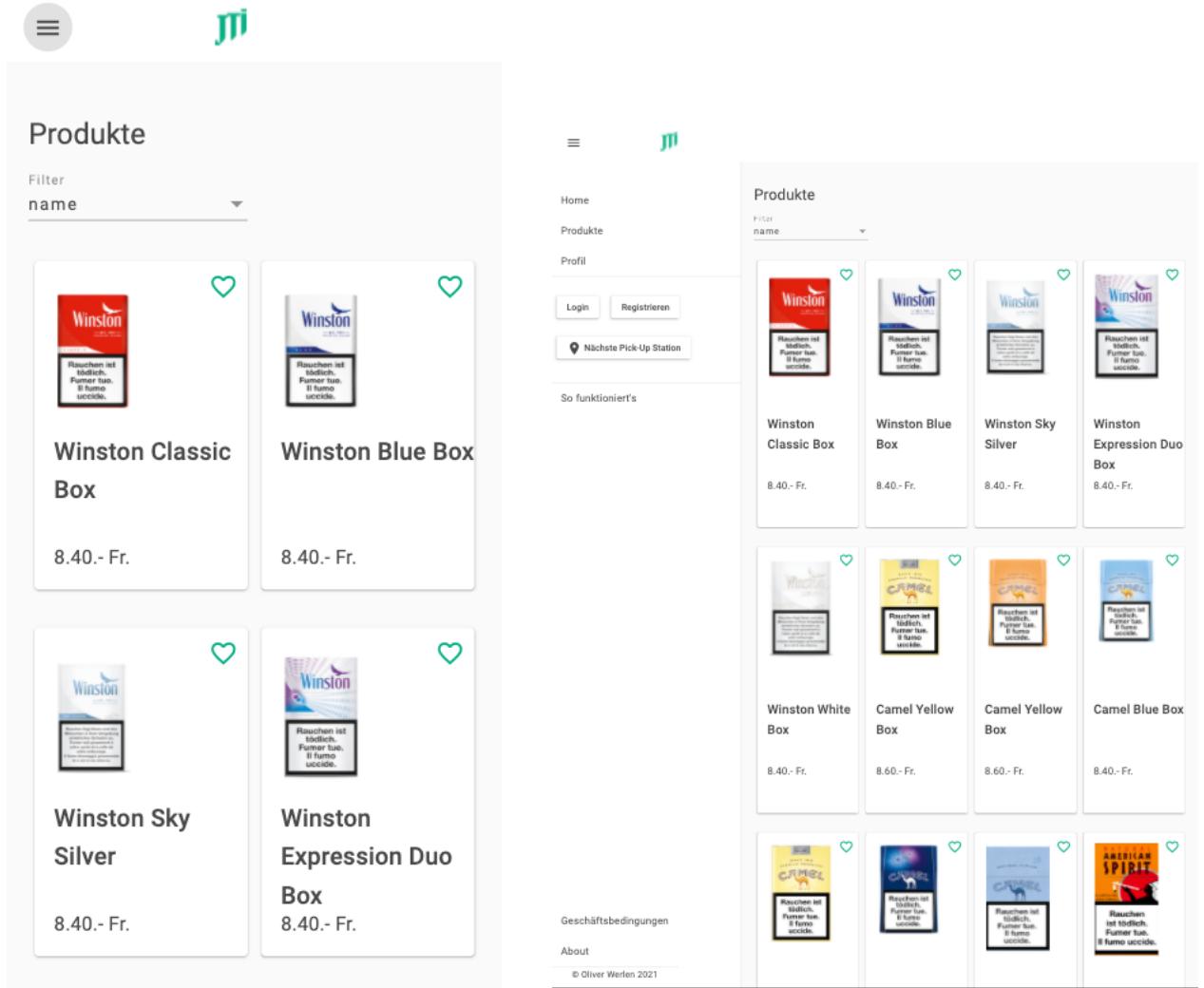
(a) Observer, Wechsel Landscape/Portrait
Tablet, Quelle: Autor

```
if(this.matchedSmartphone){
  this.cols = 2
  this.rowHeight= "300px"
}
if(this.matchedDesktop){
  this.cols = 4
  this.rowHeight = "450px"
}
```

(b) Spaltenanzahl- und Höhe abhängig von
Auflösung, Quelle: Autor

Anzeige der Produkte Die Abfrage der Produkte war bereits in einem vorderen Sprint 5.2.2 im Backend erstellt. Ergänzend wurde ein Image Controller definiert. Dieser ermöglicht es, via dem Bildnamen das passende Bild zu erhalten. Bei der Abfrage eines Produkts wird dieser als Link mitgesendet. Die Produkte wurden analog zur Java Klasse als TypeScript Klasse definiert. Die Links via HATEOAS wurden als eigenes Property definiert.

Per Angular-http Client wurde ein GET-Request an die entsprechende URL gesendet. Das resultierende Observable wurde abonniert und mittels Angular Direktive als Card Element dargestellt.



(a) Produktanzeige auf Pixel Phone, Quelle: Autor

(b) Produktanzeige auf iPad, Quelle: Autor

Sprintreview Sprint 3 Die User Story von dieser Woche konnten erneut nicht abgeschlossen werden. Aus diesem Grund wurden die Requirements aufgeteilt. Die neuen Requirements sind im Kapitel D zu finden.

5.2.4 Sprint 4

User Story	Number
Das System bietet die Möglichkeit, verschiedene Produkte anzuzeigen	F.5
Das System bietet die Möglichkeit, verschiedene Produkte dem Warenkorb hinzuzufügen	F.6

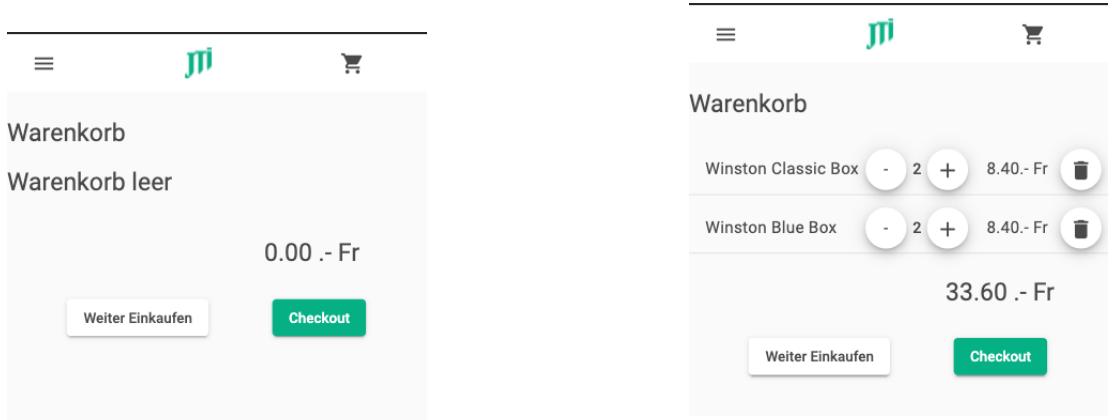
Tabelle 3: User Stories Sprint 4, Quelle: Autor

Produkte anzeigen Vom Auftraggeber wurden die Produkte in einer Excel Liste abgegeben. Hierbei war auch ein Packshot vorhanden. Es wurde relativ schnell klar, dass dieser nur für die mobile Ansicht geeignet war, da die Auflösung sehr gering war. Auf Nachfrage wurden hochauflösende Bilder zur Verfügung gestellt. Die Auflösung bei diesen war jedoch zu hoch. Das Laden der Seite verzögerte sich merklich. Aus diesem Grund mussten die Bilder komprimiert und in der Auflösung

optimiert werden. Zur Komprimierung wurde das Online Tool von Compress PNG genutzt. Die Auflösung wurde mit dem Preview Programm von MacOS angepasst. Dadurch konnte die Grösse eines Bildes von ca. 2MB auf 100KB heruntergebrochen werden. Die Bilder sehen für den Betrachter immer noch scharf aus. Zudem konnten die Bilder so alle auf dasselbe Format gebracht werden. Um die Produkte hinzuzufügen, wurde Postman genutzt. Die einzelnen Produkte-JSONs wurden einmal erstellt und können bequem via Runner hinzugefügt werden. Bei einem Klick auf ein Produkt soll die Detail View angezeigt werden. Dies war mittels Router Link sehr leicht umzusetzen. Mittels des Product-Services wird das ausgewählte Produkt gesetzt und angezeigt. Das Design dieser Komponente wird auf später verschoben, da die Warenkorbfunktionalität wichtiger ist.

Produkte dem Warenkorb hinzufügen Die Realisierung des Warenkorbs gestaltete sich als einfach. Angular liefert hierzu ein hauseigenes Tutorial. Dieses wurde erweitert, sodass Cart-Items gespeichert werden. Zusätzlich werden sie im Local Storage des Browser gespeichert, sodass der Warenkorb auch nach einem Reload noch vorhanden ist [Google, 2021d].

Der Warenkorb wird als Angular Material List dargestellt. Die Artikelanzahl kann mittels Buttons angepasst werden. Ein Löschen wird durch einen zusätzlichen Button ermöglicht. Das Gesamttotal wird nach jeder Änderung aktualisiert.



HTTPS bei Backend Moderne Browser erlauben das Laden von sogenanntem mixed Content per Default nicht. Um dieses Problem zu beheben, muss das Spring Boot Backend auch via https zugreifbar sein. Mit dem Tool Certbot kann ein gültiges Zertifikat erstellt werden. Dieses muss im Propertie-File des Spring Projekts angegeben werden. Das Tutorial verwendete dabei eine deprecated Version von Certbot. Dank einem Hinweis von Ubuntu wurde auf die neue Version gewechselt [Beni, 2021].

```
certbot certonly --webroot -d bdaf21-owerlen.enterpriselab.ch
--staple-ocsp -m oliverwerlen@bluewin.ch --agree-tos
```

Es wird empfohlen, in Spring mit PKCS12 Files zu arbeiten. Daher wurde das erstellte PEM File zu einem solchen konvertiert.

Certbot führt die Zertifikaterneuerung von selbst aus.

```
openssl pkcs12 -export -in fullchain.pem -inkey privkey.pem -out
keystore.p12 -name jtipickupbackend -CAfile chain.pem -caname root
```

Sprintreview Sprint 4 Im Sprint 4 konnte die Warenkorbfunktionalität abgeschlossen werden. Zudem läuft die Applikation nun auch im produktiven Umfeld. Um die Bestellung abschliessen zu können, muss im nächsten Schritt die Authentifizierung implementiert werden.

5.2.5 Sprint 5

User Story	Number
Das System bietet dem Anwender die Möglichkeit, sich zu registrieren und anschliessend einzuloggen.	F.3
Das System muss durch einen modernen und sicheren Authentifizierungsmechanismus geschützt sein.	L. 4

Tabelle 4: User Stories Sprint 5, Quelle: Autor

In vorherigen Sprints konnten die User Stories selten fertiggestellt werden. Aus diesem Grund wurde hier nur eine kleine Story hinzugenommen.

Authentifizierung

Backend In einem vorherigen Projekt wurde bereits eine Authentifizierung in Spring Boot umgesetzt. Das dort verwendete Tutorial funktionierte dabei tadellos. Aus diesem Grund wurde entschieden, bei diesem Projekt identisch vorzugehen. Nachfolgendes Sequenzdiagramm zeigt auf, wie die einzelnen Teile miteinander kommunizieren. Dabei wurde auf eine Token Based Authentifizierung realisiert. Umgesetzt wurde sie mittels Spring Security. Der Token muss bei jedem Request mitgesendet werden. [bezkoder, 2021]

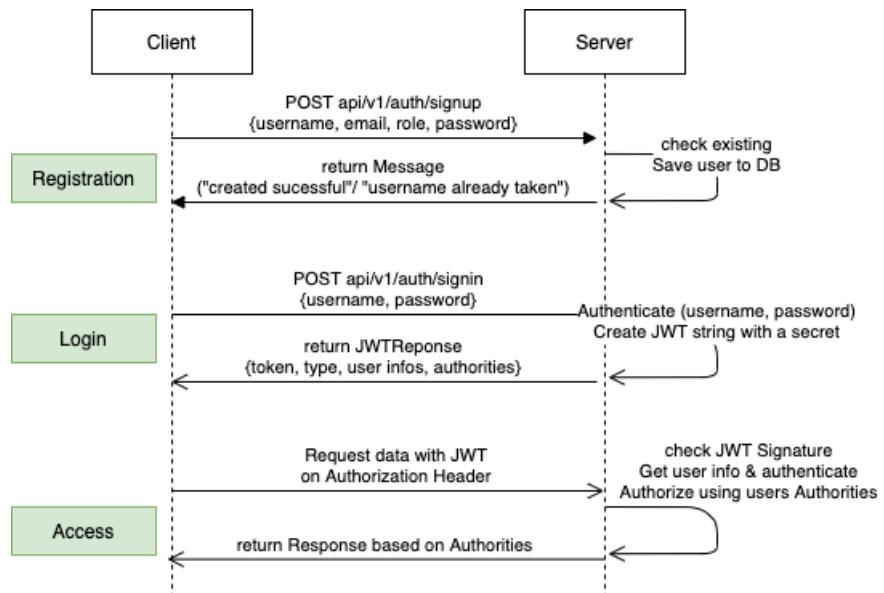


Abbildung 18: Authentication im Backend, Quelle: Autor

Algorithm HS512 ▾

Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJp  
YXd1cmxlbiIsImlhdCI6MTYxODQyMzQ3M  
ywiZXhwIjoxNjE4NTA5ODczfQ.IwSI_Mq  
7KsvS0scamA1UlQ72W8sVcXmAHzIlMIe9  
gsKTt18RXDpc4v4rIV-  
guEF5sSgL4YHYx7Fr0poiLvamSg|
```

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{  
  "alg": "HS512"  
}
```

PAYOUT: DATA

```
{  
  "sub": "iawerlen",  
  "iat": 1618423473,  
  "exp": 1618509873  
}
```

VERIFY SIGNATURE

```
HMACSHA512(  
  base64UrlEncode(header) + "." +  
  base64UrlEncode(payload),  
  your-256-bit-secret  
)  secret base64 encoded
```

Abbildung 19: Token Auflösung mit jwt.io, Quelle: Autor

Es wurden Rollen definiert, um die API feingranular sichern zu können. Die Rollen lauten dabei:

- User
- Admin
- Maintenance

Ein entsprechendes Rollenkonzept wird bei Bedarf erstellt und der Dokumentation beigefügt.

Frontend Um einen entsprechenden Nutzer zu erstellen bzw. die Eingabe von Logininformationen zu ermöglichen, mussten die entsprechenden Formulare erstellt werden. Dies wurde mittels Reactive Forms umgesetzt. Durch die damit verbundenen Validators wird clientseitig auf die Komplexität des Passworts, das Übereinstimmen der Beiden sowie die Korrektheit der Email-Adresse getestet.

(a) Login Formular, Quelle: Autor

(b) Registrierungsformular, Quelle: Autor

Die Email-Adresse sowie der Nutzernname wird serverseitig auf Einzigartigkeit überprüft.

Wie oben erwähnt, muss bei jedem Request der passende Header mitgesendet werden. Um dies ohne grossen Aufwand durchführen zu können, wird analog zum Backend ein passendes Tutorial zum Frontend bereitgestellt. Angular bietet die Möglichkeit, mit Auth-Interceptors zu arbeiten. Dieser fügt jedem API Call den entsprechenden Token im Header an. Der Interceptor dient als Proxy. [bezkoder, 2020]

```

const TOKEN_HEADER_KEY = 'Authorization';           // for Spring Boot back-end

@Injectable()
export class AuthInterceptor implements HttpInterceptor {
  constructor(private token: TokenStorageService) { }

  intercept(req: HttpRequest<any>, next: HttpHandler): Observable<HttpEvent<any>> {
    let authReq = req;
    const token = this.token.getToken();
    if (token != null) {
      authReq = req.clone({ headers: req.headers.set(TOKEN_HEADER_KEY, 'Bearer ' + token) });
    }
    return next.handle(authReq);
  }
}

export const authInterceptorProviders = [
  { provide: HTTP_INTERCEPTORS, useClass: AuthInterceptor, multi: true }
];

```

Abbildung 21: Authentication Interceptor, Quelle: Autor

Beim erfolgreichen Anmelden wird der Token im Session Storage gespeichert.

Key	Value
roles	ROLE_USER
auth-token	eyJhbGciOiJIUzUxMiJ9.eyJz...

Abbildung 22: Token im Session Storage, Quelle: Autor

Mittels Angular Direktiven wird das Menu für eingeloggte User angepasst. Es wird der Logout Button eingeblendet. Bei einem Klick auf diesen wird der Session Storage gelöscht.

Automatischer Logout Um die Sicherheit zu erhöhen, wird der Nutzer nach 30 Minuten Inaktivität automatisch ausgeloggt. Für die Umsetzung kam ein das angular-user-idle package zum Einsatz.

7	auth.service.ts:29
8	auth.service.ts:29
9	auth.service.ts:29
10	auth.service.ts:29
11	auth.service.ts:29
12	auth.service.ts:29
13	auth.service.ts:29
14	auth.service.ts:29
15	auth.service.ts:29
16	auth.service.ts:29
17	auth.service.ts:29
18	auth.service.ts:29
19	auth.service.ts:29
Time is up!	auth.service.ts:32
20	auth.service.ts:29

Abbildung 23: Ablauf von Timer und anschliessender Logout,
Quelle: Autor

Zur besseren Illustrierung wurden die Werte für das Erstellen des Bildes angepasst. Es wurden die Default-Werte beibehalten. Der Timer startet, wenn der Nutzer 10 Minuten inaktiv ist. Anschliessend läuft er 5 Minuten, ehe der Nutzer ausgeloggt wird. [rednez, 2021]

Sprintreview Sprint 5 Im Sprint 5 konnte die geplante User Story erfolgreich abgeschlossen werden.

5.2.6 Sprint 6

User Story	Number
Das System ermöglicht die Anbindung an einen bereits bekannten Bezahlungsdienst, um eine sichere Bezahlung zu garantieren.	F.7

Tabelle 5: User Stories Sprint 6, Quelle: Autor

Bezahlungsanbieter In der Sitzung E.2 wurde vom Auftraggeber bekannt gegeben, dass sie bei bereits bestehenden Lösungen den Bezahlvorgang mit dem Anbieter Six Payment Services durchführen.

Einführung Six Payment bietet einen Integration Guide an, um die Integration in den eigenen Online Store zu erleichtert [saferpay, 2021a]. Dabei wird eine JSON API zur Verfügung gestellt, um die Bezahlung durchzuführen. Die Implementierung wurde im Sprint 8 5.3.2 angepasst.

Anforderungen Das Vorgehen wurde identisch zum empfohlenen Vorgehen auf der Developer Seite durchgeführt. Dabei wurde zur Consultation die API-Dokumentation durchgelesen. Nachfolgend werden die wichtigsten Requirements aufgeführt.

- JSON API Basic Authentication
- TLS
- mindestens ein aktiver Terminal
- Terminal Nummer und Customer Nummer
- gültiges acceptance agreement für Kreditkarten

[saferpay, 2021b]

Data Security Um falsche Anwendungen und Missbrauch von Kreditkarten zu verhindern, wurde von den Kreditkartenorganisationen das Sicherheitsprogramm Payment Card Industry Data Security Standard (PCI DSS) ins Leben gerufen.

Dieser Standard kann dabei erfüllt werden, wenn die Bezahlung auf dem Saferpay Formular durchgeführt wird. Dieses Formular wird direkt beim Anbieter gehostet. Es werden keine Bezahlungsdaten auf dem eigenen Web Server verarbeitet, gesendet oder gespeichert. Somit ist es auch weniger aufwendig, die PCI DSS merchant certification zu erhalten [saferpay, 2021b].

Certification Levels Die PCI DSS Zertifikation ist in verschiedene compliance Level aufgeteilt. Die beiden Wichtigsten sind dabei Self Assigned Questionary (SAQ)-A und SAQ-A EP. Nachfolgend wird nur SAQ-A näher angeschaut.

Bei SAQ-A wird die Verantwortung vollständig an den Bezahlanbieter abgegeben. Saferpay bietet die Möglichkeit, die erforderlichen Bedingungen zu erfüllen. Die Bedingungen lauten dabei:

- Nutzen von eigenem HTML Formular verboten

- Jedes Feld auf der Payment Page muss von einem PCI zertifizierten Anbieter gehostet werden
- Änderungen an der Bezahlseite via CSS oder JavaScript sind verboten

Testsystem vs. Live System Saferpay stellt zum Testing einen Testaccount zur Verfügung. Die wesentlichen Unterschiede sind:

- Testsystem und Livesystem sind nicht miteinander verbunden
- Beim Testsystem werden richtige Kreditkarten nicht akzeptiert
- Kein Geld wird beim Testsystem transferiert
- Systemverhalten bleibt identisch
- URLs sehr ähnlich

Auf Livesystem wechseln Um von einem Testsystem auf ein Livesystem zu wechseln, sind einige kleine Änderungen vorzunehmen.

- Authorization Token anpassen
- Anpassen von IDs
- Anpassen der Request URL

Allgemeine Informationen In diesem Projekt wird nur mit dem Testsystem gearbeitet. Ein Wechsel auf ein Livesystem ist nicht geplant.

Integration in Web Applikation Um mit der JSON API kommunizieren zu können, sind im Header des Requests einige Parameter mitgegeben werden. Der korrekte Header sieht folgendermassen aus:

```
"Content-Type": "application/json",
"Authorization": "Basic QVBJXzI1Nzc1M183NTU3MTMyMjpKc29uQXBpUHdkMV9Sa1VEQzJzaw==",
```

Der Token wird im Testsystem generiert.

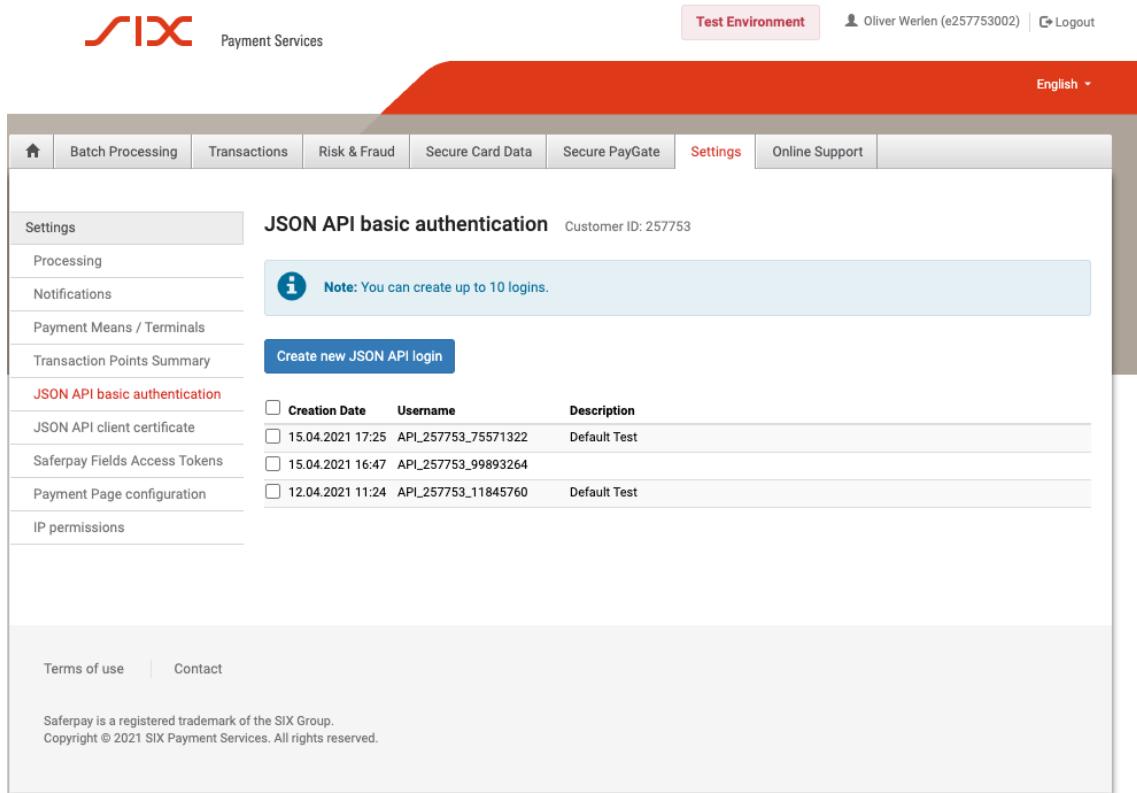


Abbildung 24: Erstellen des Basic Authentication Token im Back Office, Quelle: Autor

Payment Initialization Page An die API muss der entsprechende Request gesendet werden, um den Bezahlvorgang zu starten. Dazu muss der passende Request an die folgende URL gesendet werden:

`https://test.saferpay.com/api/Payment/v1/PaymentPage/Initialize`

In der API-Dokumentation ist ein Demo Request zu finden. Die Parameter lassen sich aus dem Backend herauslesen. Der Body der Anfrage ist in 38b zu finden. Dabei wurde bewusst auf das Erstellen eines entsprechenden TypeScript Interfaces verzichtet. Dies musste jedoch bei der Antwort vorgenommen werden. Es wurde dazu ein Interface, passend zur Antwort erstellt.

Die Anfrage an den Server geschieht asynchron. Somit ist nicht voraussehbar, wann der Request ausgeführt wird, bzw. wann die Antwort gesetzt wird. Jedoch muss in diesem Anwendungsfall auf die URL im Body weitergeleitet werden. Um auf die Antwort zu warten, wurde das selbe Vorgehen wie in 33 verwendet. Wie bereits beschrieben, wird der User auf die Zahlungsseite von Saferpay weitergeleitet. Nach dem Abschluss der Bezahlung wird entweder auf die Success-URL oder die Fail-URL von obigem Request umgeleitet.

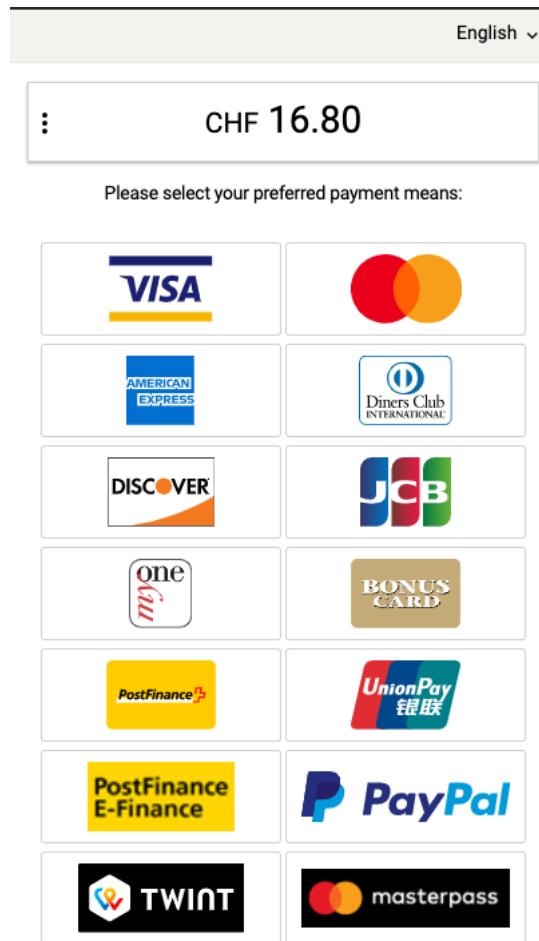


Abbildung 25: Saferpay Bezahlseite,
Quelle: Autor

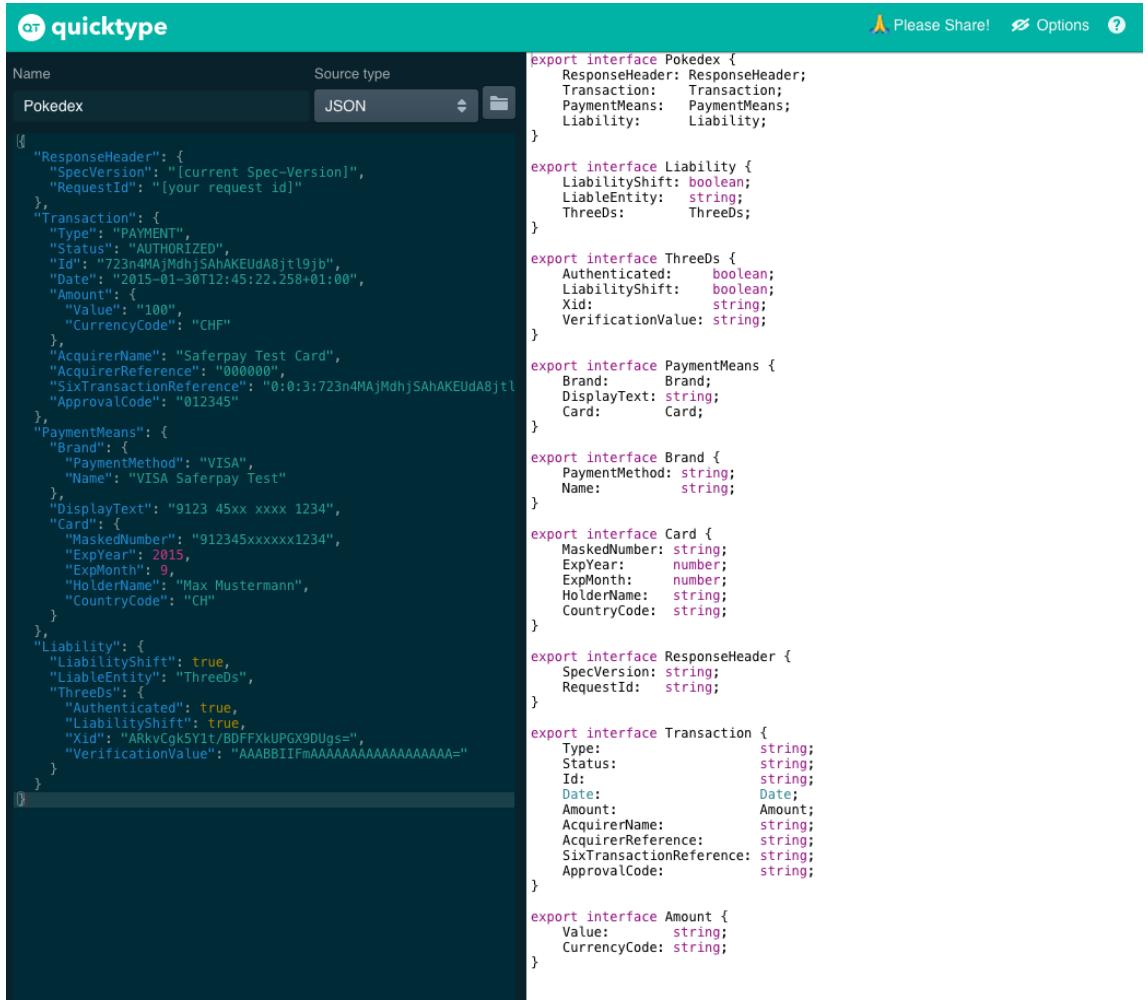
Payment Assert Das Vorgehen zur Bezahlverifikation ist dabei identisch zur Zahlungslösung.

<https://test.saferpay.com/api/Payment/v1/PaymentPage/Assert>

Um die Bezahlung identifizieren zu können, muss der Token aus vorheriger Antwort mitgegeben werden. Dieser wird im Session Storage gespeichert. Der genaue Request sieht folgendermassen aus:

```
let request = {
  "RequestHeader": {
    "SpecVersion": 1.21,
    "CustomerId": this.customerId,
    "RequestId": this.requestId,
    "RetryIndicator": 0
  },
  "Token": this.getToken()
}
```

Um das Interface für die Antwort zu definieren, wurde quicktype genutzt. Damit konnte aus dem Beispielrequest in der Dokumentation das passende Interface generiert werden.



The screenshot shows the quicktype interface with the following details:

- Name:** Pokedex
- Source type:** JSON
- Generated Code (TypeScript):**

```

export interface Pokedex {
  ResponseHeader: ResponseHeader;
  Transaction: Transaction;
  PaymentMeans: PaymentMeans;
  Liability: Liability;
}

export interface Liability {
  LiabilityShift: boolean;
  LiableEntity: string;
  ThreeDs: ThreeDs;
}

export interface ThreeDs {
  Authenticated: boolean;
  LiabilityShift: boolean;
  Xid: string;
  VerificationValue: string;
}

export interface PaymentMeans {
  Brand: Brand;
  DisplayText: string;
  Card: Card;
}

export interface Brand {
  PaymentMethod: string;
  Name: string;
}

export interface Card {
  MaskedNumber: string;
  ExpYear: number;
  ExpMonth: number;
  HolderName: string;
  CountryCode: string;
}

export interface ResponseHeader {
  SpecVersion: string;
  RequestId: string;
}

export interface Transaction {
  Type: string;
  Status: string;
  Id: string;
  Date: Date;
  Amount: Amount;
  AcquirerName: string;
  AcquirerReference: string;
  SixTransactionReference: string;
  ApprovalCode: string;
}

export interface Amount {
  Value: string;
  CurrencyCode: string;
}

```

Abbildung 26: PagePaymentAssertResponse,
Quelle: Autor

Es wird von diesem Request das Status-Property gecheckt. Bei „AUTHORIZED“ oder „CAPTURED“ wird die Bezahlung als Erfolgreich markiert. Zur Identifikation wird die Request-Id genutzt. Durch diese kann die Bezahlung im Backend eindeutig identifiziert werden.

The screenshot shows the SIX Payment Services Backoffice interface. At the top, there is a navigation bar with links for 'Batch Processing', 'Transactions' (which is highlighted in red), 'Risk & Fraud', 'Secure Card Data', 'Secure PayGate', 'Settings', and 'Online Support'. To the right of the navigation bar are buttons for 'Test Environment', user information ('Oliver Werlen (e257753002)'), and 'Logout'. Below the navigation bar is a red header bar with a dropdown menu for 'English'.

The main content area is titled 'Journal' and shows a list of transactions from the last 30 days. The table has columns for 'Authorization Date', 'State', 'Amount', 'Reference number', 'Payment means', 'Payment means details', 'Terminal', and 'Account'. The table lists seven transactions, all of which are 'Authorization (Debit)' type. The first transaction is dated 19.04.2021 at 16:51, with an amount of CHF 16.80 and a reference number of 1. The payment means is MasterCard with account number xxxx xxxx xxxx 0006 (US). The terminal is 17731797. The last transaction listed is dated 14.04.2021 at 18:59, with an amount of CHF 33.60 and a reference number of 1. The payment means is MasterCard with account number xxxx xxxx xxxx 0006 (US). The terminal is 17731797.

On the left side, there is a sidebar with various links: Failed transactions, Export, Analytics, Batch Close, Authorization, Payment, Authorized by phone, Credit, Webshop (which is highlighted in grey), All Offers, and Settings. Below the sidebar, there are links for 'Terms of use' and 'Contact'. A note at the bottom states: 'Saferpay is a registered trademark of the SIX Group. Copyright © 2021 SIX Payment Services. All rights reserved.'

Abbildung 27: Bezahlhistorie Backoffice, Quelle: Autor

Der Bezahlprozess ist abgeschlossen.

Ablauf Der Bezahlvorgang wird nachfolgend als Sequenzdiagramm dargestellt.

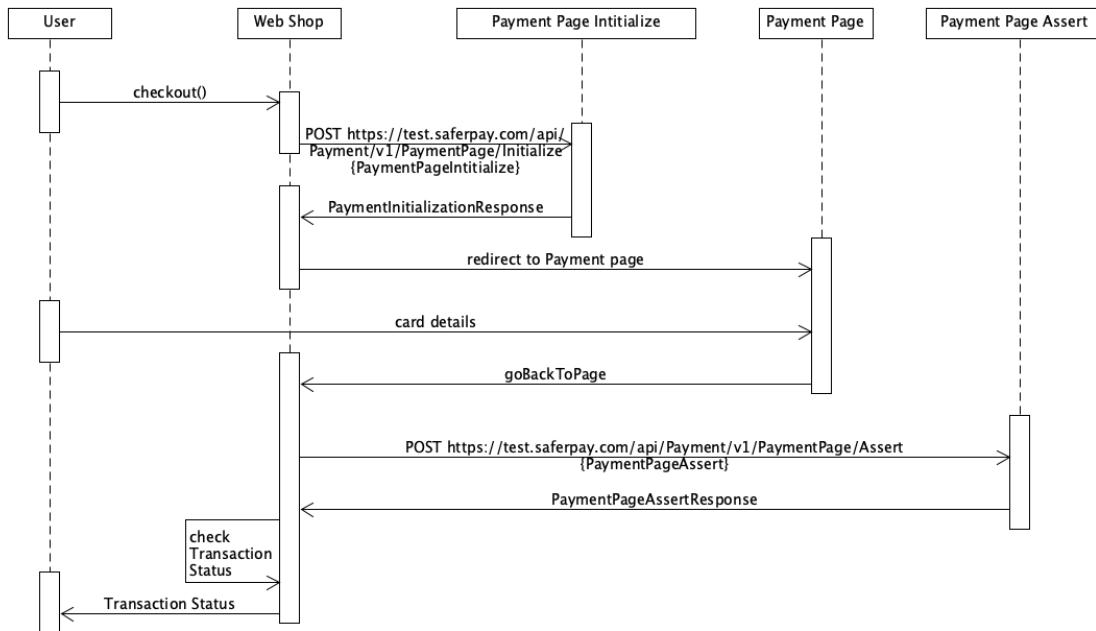


Abbildung 28: Sequenzdiagramm vom Bezahlvorgang, Quelle: Autor

CORS „Cross-Origin Resource Sharing (CORS) ist ein Mechanismus, der zusätzliche HTTP Header verwendet um einem Browser mitzuteilen, dass er einer Webanwendung, die auf einer anderen Domain(Origin) läuft, die Berechtigung erteilt auf ausgewählte Ressourcen von einem Server eines anderen Ursprungs(Origin) zuzugreifen. Eine Webanwendung stellt eine cross-origin HTTP-Anfrage, wenn sie eine Ressource anfordert, die einen anderen Ursprung(Domain, Protokoll und Port) hat, als ihren eigenen.“ [mozilla, 2021]

Dies wurde auch bei diesem Projekt bemerkt. Die Requests an die API von Six wurden von Cross-Origin Resource Sharing (CORS) geblockt.

Setzen von Access Header Wenn man Zugriff auf den betroffenen Server hat, kann mittels entsprechenden CORS Headern gearbeitet werden. Dies wurde beispielsweise beim eigenen Backend umgesetzt. Da es sich um eine externe API handelt und dementsprechend kein Zugriff auf die Maschine besteht, entfällt diese Möglichkeit.

Angular Proxy Angular bietet einen eigenen, integrierten Proxy um CORS zu umgehen. Dabei werden Request mit einer bestimmten Adresse abgefangen und an den passenden Server umgeleitet.

In diesem Anwendungsfall funktionierte der Angular Proxy nicht wunschgemäß. Zudem eignet sich dieser nur im Entwicklungseinsatz.

Zusätzlicher Proxy auf localhost Es wurde ein zusätzlicher Proxy umgesetzt. Dabei wurde ein NodeJS-Server mit express genutzt. Dieser besitzt keine Logik. Er nimmt lediglich Requests an und leitet diese an die Adresse im Header weiter [Patel, 2020]. Die Requests werden nun via Proxy an die API gesendet. Zudem wird nun auch mit der eigenen API über diesen Proxy kommuniziert. Dies löst einerseits die CORS-Problematik mit externen APIs und macht den Einsatz von HTTPS im Backend 5.2.4 überflüssig. Der Proxy ist nun via localhost erreichbar. Hierbei ist die Mixed Content Policy nicht aktiv. Bei der Entwicklung bleibt das Backend direkt erreichbar. Im produktiven Betrieb werden nur Anfragen vom Proxy akzeptiert.

Zudem ist die Verschlüsselung der Kommunikation zwischen Backend und Frontend überflüssig, da sie nur auf der eigenen Maschine abläuft. Es werden keine Daten über eine unsichere Leitung

gesendet.

Die Verbindung zur externen API läuft via HTTPS. Dies wird von Six umgesetzt.

Angedacht war dabei, dass der Proxy auch im produktiven Umfeld via Loopback-Adresse erreichbar ist. Dies bietet den Vorteil, dass die Browser dieser trauen und nicht via HTTPS zugreifbar gemacht werden muss. Der Traffic bleibt nur lokal auf der Maschine, daher bestehen keine Sicherheitsrisiken. Auf Google Chrome sowie auch Firefox funktionierte dies auch. Auf Safari ist die Loopback-Adresse nicht vertrauenswürdig, der Proxy müsste via HTTPS erreichbar sein. Das Problem ist, dass Letsencrypt keine Zertifikate für localhost ausstellt [Letsencrypt, 2017]. Es besteht keine Möglichkeit, dieses Problem unter Safari zu beheben.

Backend als Proxy Dieses Vorgehen wird in Sprint 8 5.3.2 genauer beschrieben.

Problem mit Enterpriselab Host Am Mittwoch, 14.04. nachmittags war der GitLab Host nicht mehr erreichbar. Die Nachfrage beim Enterpriselab Team ergab, dass die Maschine ihre IP-Adresse verloren hatte. Ein Neustart brachte keinen Erfolg, verschiedene wichtige Dienste starteten nicht mehr. Es wurde eine neue Maschine bereitgestellt. Der Grund konnte nicht herausgefunden werden.

Die Umgebung konnte sehr schnell wieder hergestellt werden. Es musste nur Docker und GitLab-Runner installiert und registriert werden. Zudem mussten die Daten neu migriert werden. Der Umfang der gesamten Neuumsetzung lag bei ca. 3h.

Backend Models Die Bestellungen sollen im Backend persistiert werden. Bei der Erstellung der Order-Entität fiel dabei auf, dass diese diverse Beziehungen zu anderen Klassen hat. Dadurch wurde entschieden, dass diese auch implementiert werden.

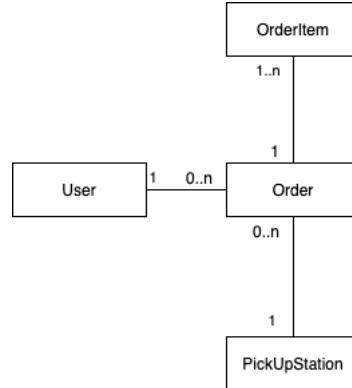


Abbildung 29: Entity Relationship Diagramm von Order,
Quelle: Autor

Bei der Implementierung wurde strikt darauf geachtet, dass die gesamte API dem REST-Level 3 nach Richardson Maturity entspricht. Entscheidend dafür ist der Einsatz von HATEOAS.

```

1
2   {
3     "pickUpStation_id": 1026,
4     "name": "Home station Oliver",
5     "latitude": "46.3983041",
6     "longitude": "7.7512328",
7     "links": [
8       {
9         "rel": "inventories",
10        "href": "http://localhost:8080/api/v1/inventories/pickUpStation/1026"
11      }
12    ]
13

```

Abbildung 30: Response bei Abfrage einer PickUp-Station,
Quelle: Autor

Durch den Einsatz von Model Mapper wurde das Mapping von DTO und Datenbankobjekt vereinfacht C.2.3.

5.2.7 Meilenstein Abschluss Bestellprozess und Zwischenpräsentation

Nachfolgend werden die Meilensteinberichte zu den Meilensteinen 3 und 4 zusammen aufgeführt.

Meilensteinbericht

Termin Meilenstein 3 Der Meilenstein 3 ist am 19.04.2021 abgeschlossen und somit pünktlich fertiggestellt worden. Der Meilenstein 4 ist am 21.04.2021 abgeschlossen und somit pünktlich fertiggestellt worden.

Beschreibung Meilenstein 3 Die Beschreibung des Meilensteins ist im Abschnitt B.2.2 ersichtlich.

Meilensteinziele/Vorgaben Das übergeordnete Ziel dieses Meilensteins ist die der Abschluss des Bestellprozesses.

- Testprotokolle
- Demo
- Release 1
- Zwischenpräsentation

Die Zwischenpräsentation ist im Anhang H, die Testprotokolle unter A.1 zu finden. Zudem ist die Konfigurationseinheit zum Release im Projektmanagementplan unter ?? aufgeführt.

Meilensteinzielerreichung Die Meilensteine konnten grösstenteils erfüllt werden. Einzig die Altersverifikation mittels Jumio wurde nach hinten verschoben. Jedoch ist das Backend bereits sehr weit fortgeschritten. Die Zwischenpräsentation wurde erfolgreich durchgeführt.

Fazit Es konnten viele der geplanten Artefakte abgeliefert werden. Die fehlende Integration von Jumio wurde durch den Fortschritt im Backend kompensiert. Zudem funktioniert die Bezahlung aufgrund von CORS-Problemen nicht auf allen modernen Browsern. Hier muss in den kommenden Sprints nachgebessert werden.

5.3 Realisierungsphase

5.3.1 Sprint 7

User Story	Number
Das System bietet dem Kunden die Möglichkeit, alle vorhandenen Pick-Up Stations anzuzeigen.	F.8
Das System bietet dem Kunden die Möglichkeit, die für ihn nächstgelegene Station auswählen zu können.	F.7

Tabelle 6: USer Stories Sprint 7, Quelle: Autor

Anzeigen von Pick-Up Stations

Google Maps Google Maps gilt als der Standard bei digitalen Karten. Seit einigen Jahren verfolgt Google jedoch ein sehr undurchsichtiges Bezahlmodell. Dem Kunden werden 200 Dollar Kredit pro Monat kostenlos zur Verfügung gestellt. Eine weitere Nutzung würde in diesem Anwendungsfall mit 2 Dollar je 1000 Anfragen belastet. Ausschlaggebend für den Entscheid gegen Google Maps war der Zwang, eine Kreditkarte zu hinterlegen.

Geoapify Als Alternative zu Google Maps wird Geoapify angeboten. Der Funktionsumfang ist sehr ähnlich, das Hinterlegen einer Kreditkarte wird nicht vorausgesetzt. Bei Geoapify sind 3000 Requests pro Tag kostenlos.

Leaflet Leaflet ist eine Open Source native Java Script Library für benutzerfreundliche, interaktive mobile Maps [Agafonkin, 2020].

OpenLayers OpenLayers dient als Alternative zu Leaflet. Dabei ist es für komplexere Applikationen ausgelegt [Sopenov, 2020]. Für dieses Projekt wird nur ein begrenzter Umfang gebraucht, daher wird mit Leaflet gearbeitet.

Open Street Map Als Karte wird Open Street Map genutzt. Open Street Map untersteht einer freien Lizenz. Die Daten werden dabei hauptsächlich von der Community gepflegt. Die Nutzung bleibt kostenlos. Die Open Street Map Contributors müssen auf der Applikation ersichtlich sein [OpenStreetMap, 2021].

Anzeigen einer Map mit Leaflet Geoapify stellt hierzu einige Tutorials zur Verfügung, um den Einstieg zu vereinfachen [Geoapify, 2020].

Marker und Popup Funktionalität Die einzelnen Marker werden aus den Daten vom Backend erstellt. Diese werden via Request vom Backend geladen und angezeigt. Zudem werden die entsprechenden Inventories zu jeder Station geladen. Da hier mit HATEOAS gearbeitet wurde, mussten die Produkte separat geladen werden.

Die Produktverfügbarkeit an einer Station sollte dabei als Popup dargestellt werden. Um die Verfügbarkeit darzustellen, wurden entsprechend gefärbte Kreise platziert. Beim Anzeigen der Popups ergab sich die Problematik, dass dies auf Geräten aus dem Hause Apple nicht funktionierte. Dabei wurde der Event von Leaflet nicht korrekt erkannt. Nach einigen Recherchen konnte keine Lösung gefunden werden. Ähnliche Fälle sind nicht bekannt.

In der Dokumentation von Leaflet ist jedoch eine Alternative zu den Popups zu finden. Diese Tooltips werden zum Anzeigen von kleinen Texten auf der Karte genutzt.

```
private addMarker(pickUpStation: PickUpStation){
    var marker = L.marker([parseFloat(pickUpStation.latitude), parseFloat(pickUpStation.longitude)], {
        icon: this.markerIcon
    }).bindTooltip(this.inventoryToString(pickUpStation.inventory!)).addTo(this.map!)
    marker.on('click', this.setCurrentItem, [])
}
```

Abbildung 31: Erstellung von Marker und Tooltip, Quelle: Autor

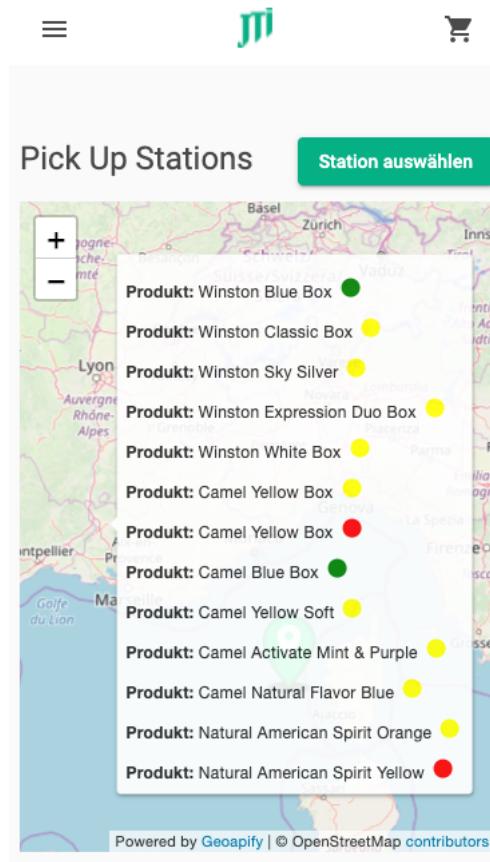


Abbildung 32: Darstellung von Marker und Tooltip, Quelle: Autor

Ein Problem war, dass die GET-Requests von RxJS asynchron ausgeführt werden. Dies bietet einerseits den Vorteil, dass die Applikation während dem Laden von Daten nutzbar bleibt, andererseits ist in diesem Fall essentiell, dass einige Prozesse aufeinander warten. Das Vorgehen wird dabei nachfolgend kurz chronologisch dargestellt:

1. Laden von aktueller Location des Benutzers
2. Erstellen der Map inkl. Kontrollelemente
3. Laden der PickUp-Stations vom Server
4. Laden des Inventars vom Server pro PickUp
5. Laden von einzelnen Produkten aus dem Inventar
6. Hinzufügen von Marker auf die Map

Wie hier zu sehen, sind die Daten voneinander abhängig. So kann das Inventar nicht geladen werden, ohne dass die PickUp Station vorhanden ist und so weiter. Es muss immer auf das Resultat gewartet werden. Um genau dies zu ermöglichen, triggered RxJS den State des Observables. Es gibt drei wesentliche Zustände: next, completed und error.



Abbildung 33: Observable State, Quelle: Autor

Next liefert einen Wert, Error einen JavaScript Error oder eine Exception und completed nichts zurück. Error wird bei den genutzten Abfragen immer direkt mittels pipe abgefangen. Mit diesem Vorgehen kann auf das vorhergehende Resultat gewartet werden.

Auswahl der PickUp Station In Abbildung 33 ist zu sehen, dass beim Öffnen des Tooltips ein click-Event abgefangen wird. Es wird ein Leaflet Event mitgegeben, welcher unter anderem auch die Koordinaten der geklickten Station enthält. Basierend auf diesen Daten wird die ausgewählte PickUp Station gesetzt. Diese wird im Local Storage des Browsers gespeichert.

```
select-pickup.component.ts:123
▼ {originalEvent: MouseEvent, containerPoint: Point, layerPoint:
  ▶ Point, latlng: LatLng, type: "click", ...} ⓘ
  ▶ containerPoint: Point {x: 188.01622464364036, y: 429.059036...}
  ▶ latlng: LatLng {lat: 42.29698, lon: 7.885466}
  ▶ layerPoint: Point {x: 195, LatLng}
  ▶ originalEvent: MouseEvent {isTrusted: false, _simulated: tr...
  ▶ sourceTarget: NewClass {options: {}, _latlng: LatLng, _ini...
  ▶ target: NewClass {options: {}, _latlng: LatLng, _initHooks...
  ▶ type: "click"
  ▶ __proto__: Object}
```

Abbildung 34: Mouse Event,
Quelle: Autor

Benutzerlocation Damit der Benutzer direkt beim Start der Karte die nächstgelegene Station sieht, wird der Standort von ihm abgefragt.

Der Standortzugriff wird nicht zwingend für die Benutzung der Applikation benötigt, weshalb beim Blocken des Standortzugriffs die Default-Location geladen wird.

Es bestand zunächst das Problem, dass von der Geolocation API nicht bemerkt wird, wenn die Systemeinstellung dem Browser den Standortzugriff verbieten. Es war hier ein Refactoring nötig, sodass die Applikation auch in diesem Fall funktionsfähig bleibt.

Sprintreview Sprint 7 Die User Story konnte erfolgreich durchgeführt werden.

5.3.2 Sprint 8

User Story	Number
Das System ermöglicht die Anbindung an einen bereits bekannten Bezahlidienst, um eine sichere Bezahlung zu garantieren.	F.9
Das System bietet die Möglichkeit, eine Bestellung dauerhaft zu speichern.	F.7
Das System bietet dem Kunden die Möglichkeit, eine Bestellung durch das Einlesen eines Codes an der Pick-Up Station abzuholen.	F.16

Tabelle 7: User Stories Sprint 8, Quelle: Autor

In diesem Sprint wird das CORS-Problem von Kapitel 5.2.6 neu angegangen.

Persistierung der Bestellung In einem vorhergehenden Sprint wurden die entsprechenden Methoden und Entitäten für die Erstellung von Orders bereits erstellt. Das Entity Relationship Diagramm 29 liefert einen Überblick.

Die Order hat dabei Beziehungen zu den folgenden Entities:

- User
- PickUpStation
- OrderItem

Es wird hier mit DTOs gearbeitet. Das Vorgehen ist in C.2.3 beschrieben.

Die Order wird erstellt, sobald der Benutzer auf den Checkout-Button im Warenkorb klickt. Es wird überprüft, ob eine PickUp-Station ausgewählt wurde und der Warenkorb nicht leer ist. Nach dem erfolgreichen Hinzufügen der Order 35 werden die Order-Items erstellt. Hierzu wird pro Item im Warenkorb ein POST-Request durchgeführt 36.

```
this.orderService.createOrder(this.currentPickUp!.pickUpStation_id).
subscribe(order =>
  {this.order = order, this.addOrderItems()})
```

Abbildung 35: Erstellen der Order, Quelle: Autor

```
addOrderItems(){
  this.cartItems.forEach(cartItem =>
    this.orderService.createOrderItems(cartItem.quantity, cartItem.product?.product_id, this.order!.order_id)
      .subscribe(orderItem =>
        {this.orderItems.push(orderItem), this.addOrderInitialization()})
  )
}
```

Abbildung 36: Erstellen der OrderItems, Quelle: Autor

Der Benutzer wird aus dem Authentifizierungstoken im Backend ausgelesen. Nach dem diese Abfragen completed sind, wird der Bezahlvorgang ausgelöst.

Inventar Jede Station hat bestimmte Inventuren. Diese werden beim Bestellen angezeigt, jedoch bei einer Bestellung nicht aktualisiert. Da das Inventar nur sehr selten von der Station an das Backend gesendet wird, muss dieses Softwareseitig geführt werden.

Die Implementierung dieser Funktionalität ist nicht banal. Es gibt diverse Punkte, welche berücksichtigt werden müssen:

- Sobald mit Bezahlung gestartet wurde, darf der Artikel nicht erneut verkauft werden.
- Bei Bezahlungsabbruch muss der Artikel wieder verfügbar sein.
- Was passiert mit Artikeln, welche bezahlt sind, aber nicht abgeholt werden?

Besonders der erste Punkt aus dieser Auflistung ist sehr wichtig. Stellen wir uns vor, dass ein Produkt an einer Station nur noch ein mal verfügbar ist. Es darf nicht möglich sein, dass zwei Personen dieses Produkt kaufen und auch bezahlen, es aber nur einer abholen kann. Das setzt voraus, dass das Inventar auf der Station zwingen aktualisiert werden muss, sobald der checkout-Button betätigt wurde.

Der Nutzer hat aber nun immer noch die Möglichkeit, den Bezahlprozess abzubrechen. In diesem Fall soll das Produkt wieder verfügbar werden.

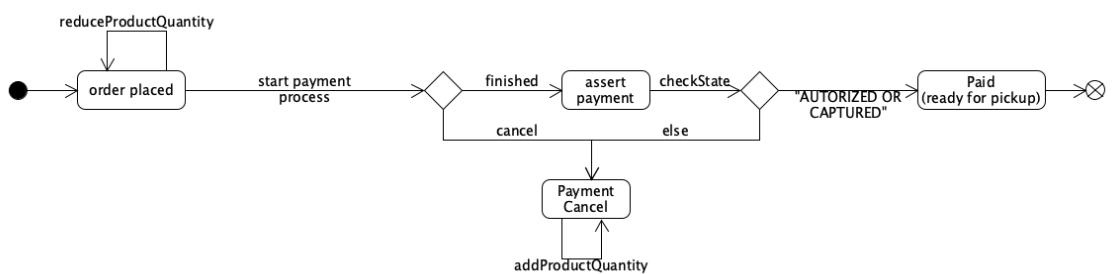


Abbildung 37: Endlicher Zustandsautomat Artikelreservierung, Quelle: Autor

Inventar im Frontend Bereits bevor der Nutzer die Bestellung platzieren kann, muss im Frontend abgefragt werden, ob die gewünschte Menge verfügbar ist und gegebenenfalls die Auswahl einschränken. Bislang wurde nur geprüft, ob ein Artikel an einer Station gar nicht mehr verfügbar ist. Dieses Konzept wird erweitert.

Jedem Produkt bei der Auswahl einer Station die Anzahl verfügbarer Produkte zugewiesen. Diese Eigenschaft kann abgefragt werden. Die Bestellung darf nur platziert werden, wenn genügend Produkte an der Station verfügbar sind.

Ein Ausnahmefall ist, wenn der Nutzer Produkte im Warenkorb hat, welche an Station A verfügbar sind, aber dann zur Station B wechselt, an der die Produkte nicht verfügbar sind. Dieser Fall wird abgefangen, die Bestellung kann nicht abgeschlossen werden.

Die Funktionalität wurde in A.2.3 getestet.

Behebung des CORS-Problems Um CORS zu umgehen, wurde ein eigener NodeJS-Proxy implementiert und auf die virtuelle Maschine im GitLab deployed. Wie bereits in 5.2.6 beschrieben, funktioniert das unter Safari nicht. Der Ansatz mit einem eigenen Proxy wurde verworfen. Dies hing damit zusammen, dass der Reverse-Proxy 5.2.2 die Anfragen an den Proxy nicht korrekt weiterleitete. Das Problem konnte nicht identifiziert werden.

Es wurde nach einem neuen Ansatz gesucht, wobei die Wahl auf die Nutzung des eigenen Backends als Proxy fiel. Das Backend übernimmt noch weiter Aufgaben bei der Bezahlung. So wird der finale Request erst im Backend erstellt. Vom Frontend kommt ein leeres Payment Initialize Objekt. In diesem sind nur die Order-Id sowie die redirect-URLs gesetzt. Es wird im Backend aus dem gesendeten leeren JSON ein entsprechendes Objekt erstellt. Das Erstellen der Klasse wurde analog zu Abbildung 26 mittels quicktype durchgeführt. Allerdings sind die einzelnen JSON-Properties bei der Saferpay API grossgeschrieben, was nicht mit der Java-Namenskonvention übereinstimmt. Es handelt sich hierbei jedoch nur um DTOs, daher wurde dies hier vernachlässigt.

```
let paymentInitialize1 = {
  "RequestHeader": {
    "SpecVersion": 1.21,
    "CustomerId": this.customerId,
    "RequestId": this.requestId,
    "RetryIndicator": 0
  },
  "TerminalId": this.terminalId,
  "Payment": [
    "Amount": {
      "Value": amount*100,
      "CurrencyCode": "CHF"
    },
    "OrderId": this.getOrderId(),
    "Description": "Description of payment"
  ],
  "ReturnUrls": {
    "Success": environment.paymentSucessfullRedirect,
    "Fail": environment.paymentNotSucessfullRedirect
  }
}
```

(a) Payment Initialization alt, Quelle: Autor

```
let paymentInitialize = {
  "RequestHeader": {
    "SpecVersion": "",
    "CustomerId": "",
    "RequestId": this.getOrderId(),
    "RetryIndicator": 0
  },
  "TerminalId": "",
  "Payment": {
    "Amount": {
      "Value": "",
      "CurrencyCode": "CHF"
    },
    "OrderId": this.getOrderId(),
    "Description": "Description of payment"
  },
  "ReturnUrls": {
    "Success": environment.paymentSucessfullRedirect,
    "Fail": environment.paymentNotSucessfullRedirect
  }
}
```

(b) Payment Initialization neu, Quelle: Autor

Ausschlaggebend bei diesem Request ist die Order-Id. Sie wird bei der Persistierung der Order gespeichert und im Local Storage gespeichert. Sie wird auch als Request-Id genutzt. Diese ist im gesamten System einzigartig. Die Bezahlung kann im Backoffice über diese Id eindeutig identifiziert werden.

Setzen der fehlenden Properties Wie in der Abbildung 38b zu sehen ist, beinhaltet der Request nur die Order-Id. Die restlichen Daten werden im Backend gesetzt. Es werden die Werte aus dem Properties-File gelesen und zur Laufzeit den Variablen zugewiesen.

```
@Value("${pickupbackend.app.saferpay}")
private String BASIC_AUTH_TOKEN_SAFERPAY;

@Value("${pickupbackend.app.saferpay.customerid}")
private String CUSTOMER_ID_SAFERPAY;

@Value("${pickupbackend.app.saferpay.terminalid}")
private String TERMINAL_ID_SAFERPAY;
@Value("${pickupbackend.app.saferpay.specVersion}")
private String SPEC_VERSION;
```

Abbildung 39: Auslesen der Werte aus .properties-File,
Quelle: Autor

Der Bestellwert wird ebenfalls im Backend gesetzt. Dies macht es unmöglich, dem Bestellwert im Frontend anzupassen und so eine Vergünstigung zu erhalten. Um den Wert zu bestimmen, werden alle OrderItems, die zur Order gehören, in der Datenbank gesucht und der Produktpreis mit der Anzahl Produkte summiert. Dank streams in Java kann dies sehr elegant durchgeführt werden. Java empfiehlt den Gebrauch von Atomic Datentypen, wenn diese mit Streams bearbeitet werden.

```
public String calculateOrderTotal(Integer order_id){
  AtomicReference<Double> total = new AtomicReference<Double>( initialValue: 0.0);
  this.orderItemService.getOrderItemByOrder(this.getOrder(order_id)).forEach(orderItem -> total.updateAndGet(v -> v + (orderItem.getProduct().getPrice()*100) * orderItem.getQuantity()));
  return new DecimalFormat(pattern: "#.####").format(total.get());
}
```

Abbildung 40: Berechnung des Order-Totals, Quelle: Autor

Zudem schreibt die API von Six vor, dass der Wert in Rappen, bzw. ohne Kommastellen gesendet wird. Durch das DecimalFormat werden die Nachkommastellen entfernt.

Senden eines POST-Requests von Spring Um einen POST-Request von Spring an eine andere API zu senden, wurde **RestTemplate!** (**RestTemplate!**) genutzt [baeldung, 2021]. Dazu wurde eine HttpEntity mit dem gewünschten Body und dem passenden Header erstellt. Das Erstellen dieser wurde in einen Service 42 ausgelagert.

```
public @ResponseBody PaymentInitializationResponse initializePayment(@RequestBody PaymentPageInitialize paymentPageInitialize) {
    final String uri = "https://test.safepay.com/api/Payment/v1/PaymentPage/initialize";
    RestTemplate restTemplate = new RestTemplate();
    PaymentInitializationResponse response = restTemplate.postForObject(uri, this.paymentService.setPaymentInitialization(paymentPageInitialize), PaymentInitializationResponse.class);
    return response;
}
```

Abbildung 41: Senden eines Request mit RestTemplate, Quelle: Autor

```
public HttpEntity<PaymentPageInitialize> setPaymentInitialization(PaymentPageInitialize paymentPageInitialize){
    paymentPageInitialize.getRequestHeader().setSpecVersion(SPEC_VERSION);
    paymentPageInitialize.getRequestHeader().setCustomerId(CUSTOMER_ID_SAFEPAY);
    paymentPageInitialize.setTerminalId(TERMINAL_ID_SAFEPAY);
    paymentPageInitialize.getPayment().setAmount().setValue(this.orderService.calculateOrderTotal(Integer.parseInt(paymentPageInitialize.getPayment().getOrderId())));
    return new HttpEntity<>(paymentPageInitialize, this.getHeader());
}
```

Abbildung 42: Erstellen der HttpEntity, Quelle: Autor

Die Antwort von diesem Request wird vom Backend direkt weitergeleitet. Im Frontend wird der Token im Local Storage gespeichert. Dieser wird beim Payment Assert gebraucht.

Im nächsten Schritt wird das PaymentAssert durchgeführt. Wie auch bereits im obigen Beispiel gesehen, werden auch hier die meisten Daten im Backend gesetzt. Hier wird der Token sowie die Request-Id aus dem Local Storage mitgegeben.

Als Antwort auf diesen Request wird der Statuscode zurückgegeben. Zudem wird die Order in der Datenbank als bezahlt markiert.

```
if(response.getTransaction().getStatus().equals("AUTHORIZED") || response.getTransaction().getStatus().equals("CAPTURED")){
    this.orderService.markAsPaid(Integer.parseInt(paymentPageAssertRequest.getRequestHeader().getRequestId()));
    return new ResponseEntity<>(
        HttpStatus.OK
    );
} else {
    return new ResponseEntity<>(
        HttpStatus.BAD_REQUEST
    );
}
```

Abbildung 43: Überprüfung des Zahlungsstatus, Quelle: Autor

Dieser wird im Frontend überprüft, eine entsprechende Meldung ausgegeben und ein Redirect auf das Nutzerprofil durchgeführt. Der Bezahlvorgang ist abgeschlossen.

Initiale Erstellung von Stations Wird eine Pick-Up Station initial eingeschaltet, wird ein POST-Request an das Backend ausgeführt, welcher eine neue Station erstellt. Die Internet Protocol (IP)-Adresse wird ausgelesen. Im Backend wird überprüft, ob bereits eine Station mit der selben IP vorhanden ist. In diesem Fall wird die bestehende zurückgegeben. Die Station ist noch nicht initialisiert, der Name und die Koordinaten sind zufällig gewählt. Im Frontend ist es dem Administrator möglich, diese Daten einzugeben und die Station zu initialisieren.

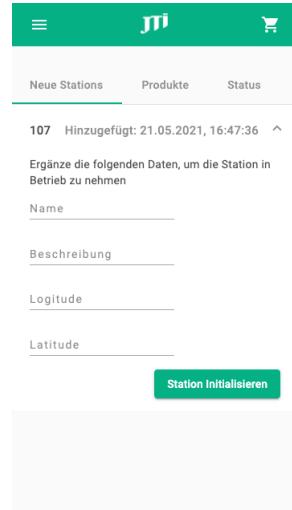


Abbildung 44: Initialisierung einer neuen Station,
Quelle: Autor

Nach dem Erstellen der Station werden die Inventar-Items erstellt. Dazu werden die Daten von der Universal Asynchronous Receiver Transmitter (UART)-Schnittstelle 5.3.2 gelesen und entsprechend zu einem Java Script Object Notation (JSON) umgewandelt. Im Anschluss werden sie an das Backend gesendet. Hier wird überprüft, ob es zu Änderungen im Vergleich zum vorherigen Stand kam und entsprechend die Daten erneuert.

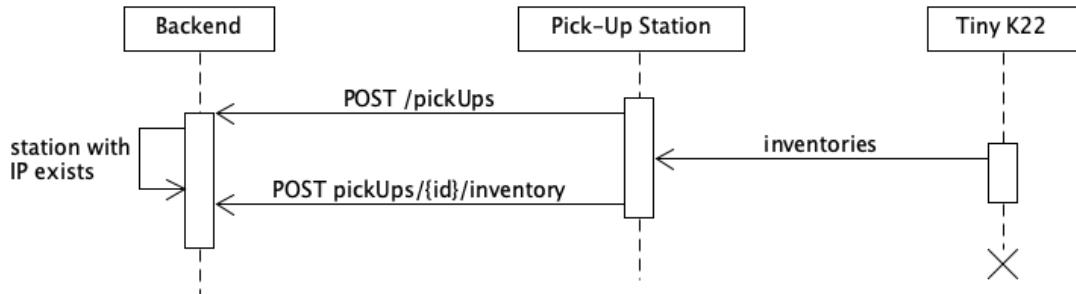


Abbildung 45: Sequenzdiagramm der Initialisierung einer neuen Station,
Quelle: Autor

Abholung einer Bestellung

Erstellen eines QR-Codes Bereits in Sprint 1 5.2.1 wurde spezifiziert, dass zur Abholung ein QR-Code eingelesen werden muss. Um dies umzusetzen, wurde in einem ersten Schritt ein PickUp-Identifier der PickUp-Station-Entity hinzugefügt. Hierbei handelt es sich um einen 40-stelligen String, welcher bei der Erstellung generiert wird.

Dieser String wird mit der Java Library ZXing in einen QR-Code verpackt. Bei ZXing handelt es sich um die Hauptlibrary, welche bei QR-Codes in Java zum Einsatz kommt. Der Code wird als Buffered Image zurückgegeben. [baeldung, 2020]

```
public BufferedImage generateQRCodeImage(Integer pickUpStation_id) throws Exception {
    QRCodeWriter qrCodeWriter = new QRCodeWriter();
    BitMatrix bitMatrix = qrCodeWriter.encode(this.getOne(pickUpStation_id).getPickUpStationIdentifier(),
        BarcodeFormat.QR_CODE, width: 300, height: 150);

    return MatrixToImageWriter.toBufferedImage(bitMatrix);
}
```

Abbildung 46: Erstellen eines QR-Code mit ZXing, Quelle: Autor



Abbildung 47: QR-Code Beispiel, Quelle: Autor

Dieser Code wird beim finalen Produkt auf der Station positioniert, sodass er gut einlesbar ist.

Lesen des QR-Codes mit Frontend Für das Lesen des QR-Codes wurde die reine JavaScript Library jsQR eingesetzt [cozmo, 2021]. Dabei wurde auf eine Implementierung auf Stackblitz zurückgegriffen und diese entsprechend angepasst [Yamaguchi, 2020].

Es bestand lange das Problem, dass anstatt der hinteren Kamera auf die Frontkamera zurückgegriffen wurde. Durch das Hinzufügen des entsprechenden Media-Constraints

```
video: {facingMode: "environment"}
```

wird per Default auf die hintere Kamera zurückgegriffen. [Mozilla, 2020] Damit dies auch Umgesetzt wird, muss das WebRTC adapter-package hinzugefügt werden.

Abholmeldung an Backend Wenn ein QR-Code gelesen wird, sendet RxJS einen Call mit dem Pick-Up Identifier und dem Order-Pick-Up Token an die API.

```
let pickUp = {
    "pickUpToken": JSON.parse(localStorage.getItem('order') || '{}').pickUpToken,
    "pickUpStationIdentifier": stationToken
}
```

Abbildung 48: PickUp Station Identifier und PickUp Token, Quelle: Autor

Im Backend kann durch diese beiden Token die PickUp Station und auch die Bestellung eindeutig identifizieren. Es wird analog zu 5.3.2 ein Request an die PickUp Station gesendet. Von dieser ist die entsprechende URL in der Datenbank vorhanden. Gesendet werden dabei die einzelnen Order Items.

Node Server auf Station Auf der Pick-Up Station läuft ein Node Server. Dieser hat dabei zwei Funktionen:

1. Senden von initialer Nachricht an Backend, wenn Station gestartet wurde
2. Empfangen von Bestellausgabeanforderungen vom Backend

Um das umzusetzen und den Aufwand im Rahmen zu halten, wurde ein Node Server aufgesetzt. Die Kommunikation mit der Elektrotechnik wurde noch nicht umgesetzt.

Bei jedem Neustart der Station wird dabei ein POST-Request durchgeführt und das momentane Inventar gesendet. Dabei handelt es sich um Updates, sofern die Daten bereits vorhanden waren oder sie werden erstellt. Dies kommt besonders beim Inventar sehr oft vor.

Um die Ausgabeanforderungen empfangen zu können, wurden zudem ein POST-Endpoint erstellt.

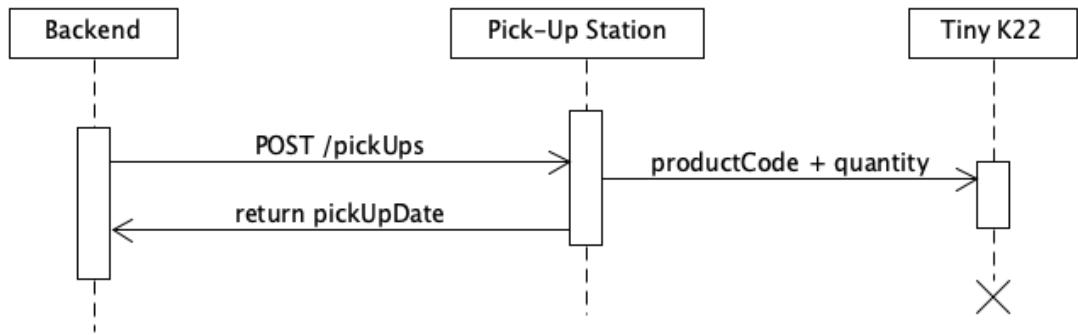


Abbildung 49: Sequenzdiagramm der Bestellabholung, Quelle: Autor

Abschliessende Bemerkungen Es wird von der Applikation kein Busy-Waiting genutzt, da der Event vom Backend ausgelöst wird. Die Node-Applikation hört nur auf den entsprechenden Port. Als Unterschied zu WebHooks wird auf die Rückgabe der Meldung gewartet. Gemäss Aussage von der Elektrotechnikseite hält sich diese Wartezeit mit <15 Sekunden sehr stark in Grenzen. Da die Applikation Requests asynchron ausführt, wird die Userexperience nicht beeinflusst. Zudem sind nur sehr wenige Informationen zur Implementation eines eigenen Webhook-Endpoints zu finden. Beinahe alle Dokumentationen befassen sich mit der Nutzung von bereits bestehenden Endpunkten.

Verbindung zwischen Station und Backend Damit die Station vom Backend aus bei einer Abholung angesprochen werden kann, benötigt diese eine fixe, erreichbare Adresse. Heutzutage werden keine öffentlichen IP-Adressen vergeben, beziehungsweise (bzw) der Preis für ein derartiges Angebot ist entsprechend hoch. Daher wurden verschiedene Alternativen getestet.

VPN Die virtuelle Maschine ist vom VPN des Enterpriselabs erreichbar. Der Ansatz bestand darin, das Raspberry Pi ebenfalls in dieses VPN zu integrieren, was auch auf Anhieb funktionierte. Ein Ping auf die virtuelle Maschine war auch erfolgreich, jedoch konnte das Pi vom Host nicht erreicht werden. Auf Anfrage gab das Enterprise Lab-Team bekannt, dass Verbindungen von der Maschine in das Virtual Private Network (VPN) aus Sicherheitsgründen von der Firewall geblockt werden [von Uslar, 2021b].

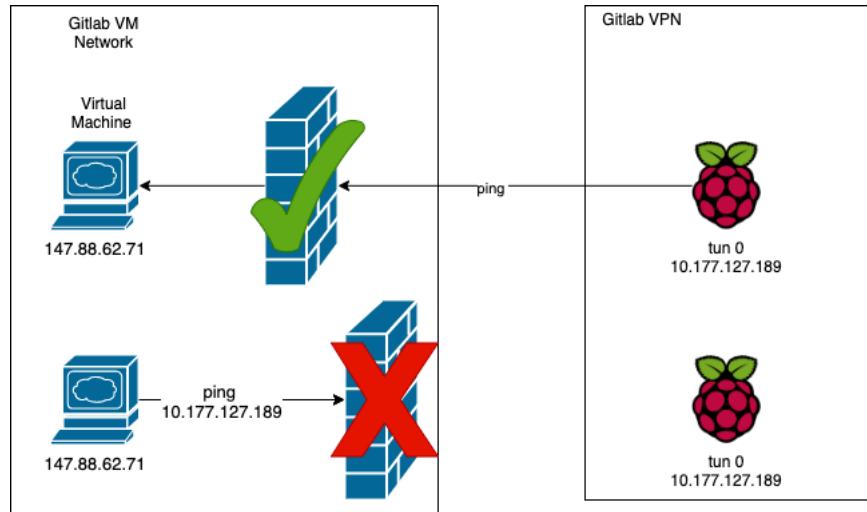


Abbildung 50: Ping zwischen Raspberry Pi und Gitlab VM, Quelle: Autor

Auf die Umsetzung eines eigenen VPN-Netzes wurde aus Zeitgründen verzichtet.

No-IP Vom Betreuer wurde im Meeting E.5 vorgeschlagen, mit No-IP zu arbeiten. Das Problem bei diesem Lösungsansatz ist, dass im Mobilfunknetz kein Portforwarding möglich ist. Dieses wird aber benötigt, um auf die Adresse zugreifen zu können. Daher musste auch diese Lösung verworfen werden.

zerotier Zerotier ist eine Kombination aus VPN und **SD-WAN!** (**SD-WAN!**). Es entspricht genau den Anforderungen an das System, eine sichere Verbindung zwischen verschiedenen Geräten zu schaffen. Die Einrichtung war sehr intuitiv und in wenigen Minuten abgeschlossen. Vom Enterprise Lab musste der User Datagram Protocol (UDP)-Port 9993 geöffnet werden. Als nächster Schritt wurde ein Konto und ein Netzwerk erstellt. Auf den Clients musste der entsprechende Client installiert werden. Es werden Implementierungen für nahezu jede Plattform angeboten. Die Geräte konnten nun verbunden werden.

```
sudo zerotier-cli join Netzwerk-ID
```

Im Admin Panel mussten die Geräte noch approved werden.

Auth?	Address	Name/Description	Managed IPs	Last Seen	Version	Physical IP
<input checked="" type="checkbox"/>	1fce291edb 66:09:ba:9c:5f:52	enlab back (description)	10.147.20.169 + 10.147.20.x	ONLINE	1.6.5	147.88.62.71
<input checked="" type="checkbox"/>	34d15fa90a 66:22:a5:ea:e8:83	pi (description)	10.147.20.185 + 10.147.20.x	ONLINE	1.6.5	194.230.155.158
<input checked="" type="checkbox"/>	43695681ef 66:55:1d:e3:c0:66	mac (description)	10.147.20.188 + 10.147.20.x	ONLINE	1.6.5	178.198.210.197

Abbildung 51: Verbundene Geräte in zerotier, Quelle: Autor

Es wurden die obigen drei Geräte hinzugefügt, wobei nur die beiden Ersten im Projekt miteinander kommunizieren. Die Entwicklermaschine wurden zum einfacheren Debugging hinzugefügt.

Zerotier ist bis zu 50 Geräte kostenlos, es ist ein Administrator verfügbar. Die IP-Adressen sind statisch, die Kommunikation End-to-End Verschlüsselt.

Kommunikation zwischen Pi und Tiny Auf der Elektrotechnikseite wird ein Tiny K22 zur Kommunikation mit der Hardware genutzt. Darauf wird hier nicht weiter eingegangen.

Die Kommunikation zwischen Pi und Tiny läuft via UART. Um mit Node mit seriellen Ports arbeiten zu können, wird die Library Serial Port genutzt. Die Verbindung zum UART-Port konnte dabei erfolgreich hergestellt werden. Das Schreiben auf den Port war auch möglich, die Daten konnten von der Elektrotechnik erfolgreich und vollständig gelesen werden.

Beim Lesen der Daten wurden immer nur die ersten 4 Byte angezeigt. Auf Anhieb konnte keine Lösung für dieses Problem gefunden werden, daher wurde ein neuer Github-Issue im Serial Port Projekt erstellt. UART with Raspberry Pi Zero only receiving 4 Bytes

Sprintreview Sprint 8 In diesem Sprint konnten die geplanten Userstories umgesetzt werden.

5.3.3 Sprint 9

User Story	Number
Das System bietet die Möglichkeit, durch die Anbindung an eine 3rd Party, eine Altersverifikation durchzuführen.	F.4
Das System muss die Punkte in der von Google aufgestellten Core Progressive Web App checklist erfüllen.	F.1

Tabelle 8: User Stories Sprint 9, Quelle: Autor

Einlesen der Identitätskarte Jumio bietet keine Testversion an. Um die Integration in die Applikation durchführen zu können, wurden vom Auftraggeber der API-Key zur Verfügung gestellt. Zudem wäre die Möglichkeit bestanden, initiale Konfigurationen im Backoffice durchführen zu können. Darauf wurde verzichtet, da die gewünschten Einstellungen direkt mit dem API-Call definiert werden können.

Jumio bietet eine GitHub Integration Guide an. In diesem wird sehr gut erklärt, wie der initiale Request definiert werden muss. Um einem erneuten CORS-Problem vorzusorgen, wurde direkt das Vorgehen von 5.3.2 übernommen.

```
let ageVerification = {
  "customerInternalReference" : "",
  "userReference" : "",
  "successUrl" : "https://bdaf21-owerlen.enterpriselab.ch/ageVerificationSucessfull",
  "errorUrl" : "https://bdaf21-owerlen.enterpriselab.ch/error",
  "callbackUrl" : "https://bdaf21-owerlen-02.enterpriselab.ch/api/v1",
  "reportingCriteria" : "",
  "workflowId" : 100,
  "presets" :
  [
    {
      "index" : 1,
      "country" : "CHE",
      "type" : "ID_CARD"
    }
  ],
  "locale" : "ch-CH"
}
```

Abbildung 52: Initialer Request an Jumio API, Quelle: Autor

Dabei wird nur die „customerInternalReference“ und „userReference“ im Backend gesetzt. CustomerInternalReference entspricht dabei dem HashCode vom User, als userReference dient die

userId. Zudem ist es vorgegeben, dass die Uniform Resource Locator (URL)s via Hypertext Transfer Protocol Secure (HTTPS) ausgeliefert werden sowie keine Sonderzeichen enthalten. Die Callback URL wird nicht aktiv genutzt.

Als Antwort auf den Request werden die transactionReference, der Timestamp und die redirectUrl zurückgegeben. Wie auch bei der Zahlung wird auf die redirectUrl weitergeleitet. Anschliessend beginnt der Prozess bei Jumio.

Nach dem Abschluss der Altersverifikation wird je nach Status auf die successUrl oder auf die errorUrl weitergeleitet. Im Success-Fall hängt Jumio der Url noch drei weitere Parameter an. Es handelt sich um den Status, die customerInternalReference und transactionReference.

Profil sperren Aus rechtlicher Sicht ist es essentiell, dass nur Personen, die nachweislich älter als 18 Jahre sind, den Dienst nutzen können. Um dies zu gewährleisten, dass dies erfüllt ist, wurde auf eine Eigenschaft von Spring Security zurückgegriffen. Bei der Nutzererstellung wird ein Objekt vom Typ User Details Implementation erstellt. Dieses implementiert das Interface User Details, in welchem unter anderem Methoden zum Sperren und Aktivieren von Profilen definiert sind. Bei der bisherigen Anwendung wurden diese Methoden überschrieben, ohne explizite Werte zu berücksichtigen. Allerdings bietet sich dieser Anwendungsfall an, um eine eigene Implementation der Methoden durchzuführen.

Es wurde die „isEnabled()“ Methode ausgewählt, da diese dem gewünschten Effekt entspricht. Anstatt das hier nur true zurückgegeben wird, greift die Methode auf ein Attribut der Klasse UserDetailsImpl zu.

```
@Override
public boolean isEnabled() {
    return this.ageVerified;
}
```

Abbildung 53: Überschreiben der isEnabled() Methode, Quelle: Autor

Altersverifikation Die eigentliche Altersverifikation wird nach dem erfolgreichen Einlesen des Ausweises durchgeführt. Die Daten werden von Jumio aus den Bildern ausgelesen und können abgefragt werden. Dazu wird die transaction-Reference benötigt.

Ein GET-Request auf die folgende URL liefert diese Daten zurück.

```
https://netverify.com/api/netverify/v2/scans/
xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/data
```

Der entscheidende Wert ist dob (Date of Birth). Im Backend wird die Zeitspanne zwischen jetzt und diesem Datum berechnet. Der Wert muss grösser oder gleich 18 sein, um die Applikation nutzen zu können.

```
public boolean verifyUser(ScanData scanData, String ageVerificationReference){
    User user = this.getByAgeVerificationReference(ageVerificationReference);
    if(Period.between(scanData.getDocument().getDob(), LocalDate.now()).getYears() >= 18){
        this.updateAgeVerified(user);return true;
    }else{return false;}
}
```

Abbildung 54: Altersverifikation, Quelle: Autor

Zusätzlich wird der Name auf dem Ausweis mit dem in der Datenbank verglichen. Somit kann sichergestellt werden, dass nicht mit dem Ausweis einer anderen Person die Altersverifikation umgangen wird.

Die Altersverifikation wurde ausführlich getestet. Die entsprechenden Testprotokolle sind unter A.2.1 zu finden.

Sprintreview Sprint 9 In diesem Sprint konnte nur eine der beiden User Stories abgeschlossen werden. Die User Story F.1 wird in den nächsten Sprint übernommen.

5.3.4 Sprint 10

User Story	Number
Das System muss die Punkte in der von Google aufgestellten Core Progressive Web App checklist erfüllen.	F.1

Tabelle 9: Userstories Sprint 10,
Quelle: Autor

Integration von Progressive Web App Funktionalität

Offlinefähigkeit Um die Applikation auch ohne aktive Internetverbindung nutzbar zu machen, wurden Service Worker genutzt. Das Hinzufügen zum Projekt übernahm die Angular CLI.

```
ng add @angular/pwa --project jtiPickUp
```

Service Worker funktionieren mit dem klassischen ng-serve nicht. Um das Projekt trotzdem lokal testen zu können, gibt es die Möglichkeit, einen lokalen Http-Server laufen zu lassen. Um diesen Prozess zu vereinfachen, wurde ein neuer Command im package.json hinzugefügt.

```
"start-pwa": "ng build --prod && http-server -p 8090 -c-1 dist/jtiPickUpStation"
```

Die Service Worker sind nun aktiv. Dies ist auch in der Developer Konsole von Google Chrome zu sehen.

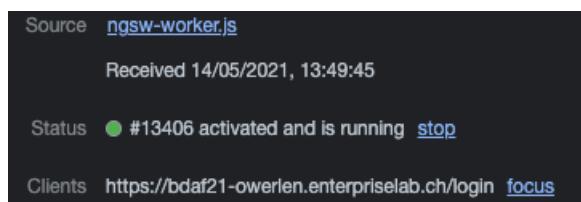


Abbildung 55: Aktiver Service Worker in Google Chrome,
Quelle: Autor

Per Default cached der Service Worker folgende Dateien:

- index.html
- favicon.ico
- build artifacts
- assets
- images und fonts

Google, 2021c

Der Service Worker speichert die Version im Hintergrund. Aus performance Gründen wird immer die gespeicherte Version ausgeliefert, auch wenn eine Neuere vorhanden ist.

Caching von Request Für die Applikation reichte das Speichern der obigen Dateien nur bedingt aus. Damit die App auch Offline benutzbar bleibt, war es nötig, die Request zu cachen. Angular bietet zwei Arten von Caching an: performance und freshness. Bei Performance werden die Daten immer aus dem cache geladen, sofern sie das maxAge noch nicht erreicht haben. Dies bringt, wie es der Name bereits sagt, einen enormen Performancegewinn. Bei Freshness werden die Daten immer von der API geladen, sofern das Timeout nicht erreicht wurde. Dies garantiert, dass die Daten aktuell sind.

In der folgenden Tabelle wird aufgezeigt, welche Request von welcher Strategie Gebrauch machen.

Performance	Freshness
Laden von Produkten	Laden von Inventar
	Orders by User
	PickUpStations

Tabelle 10: Übersicht über die Verwendung von Freshness und Performance,
Quelle: Autor

Mit der Performance-Strategie werden ausschliesslich Daten geladen, welche sich nur sehr selten ändern. Die Service Worker haben hauptsächlich zum Zweck, die Applikation Offlinefähig zu machen. Der Performancegewinn ist sekundär.

Nachfolgend werden noch die Features aufgeführt, welche im Offline-Modus nicht verfügbar sind:

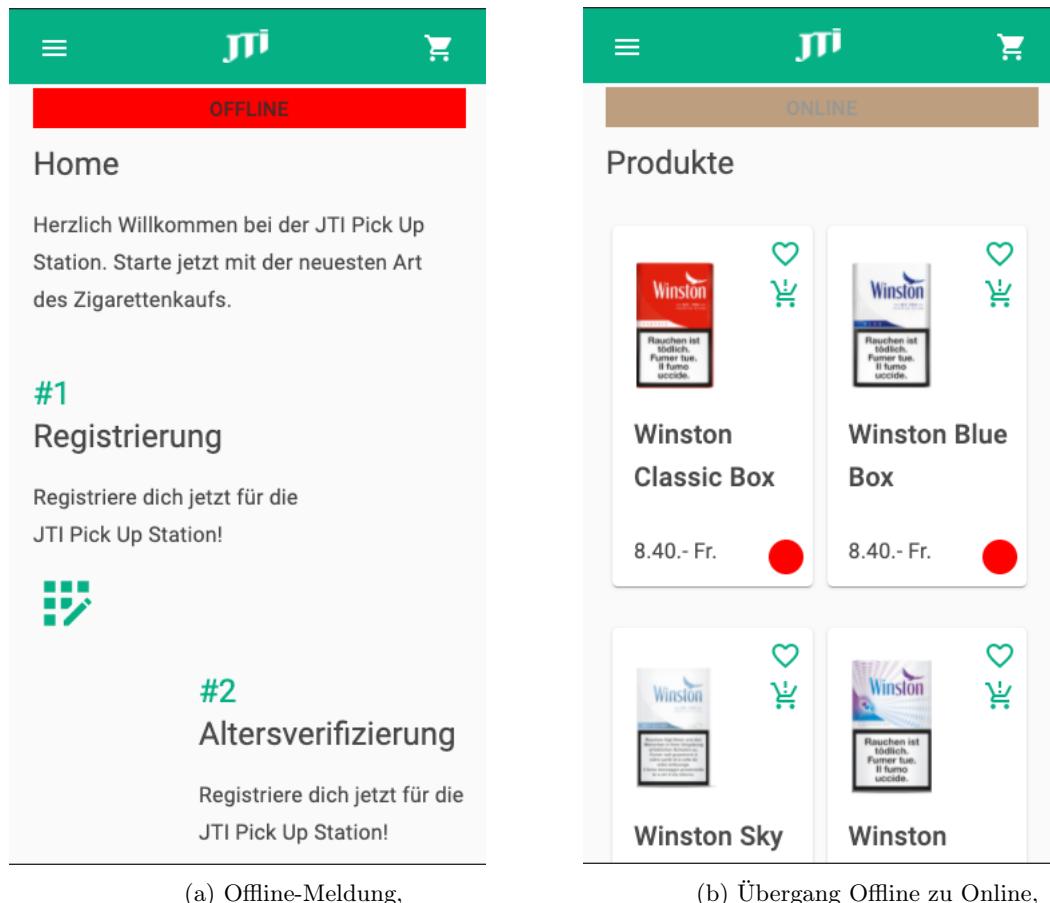
- Login und Registrierung
- Platzieren einer Bestellung
- Abholen einer Bestellung

Anzeigen von Online und Offlinestatus Der User sollte auch informiert werden, wenn die Internetverbindung unterbrochen ist. Um dies umzusetzen, wird ein ng-connection-service bereitgestellt. Dieser kann überwacht werden und somit eine Netzwerkveränderung bemerkt werden.

```
this.connectionService.monitor().subscribe(isConnected => {
  this.isConnected = isConnected;
  if (this.isConnected) {
    this.status = "ONLINE";
  } else {
    this.status = "OFFLINE";
  }
})
```

Abbildung 56: Montoring der Connection-Status,
Quelle: Autor

Entsprechend des Status wird eine Meldung in der Applikation angezeigt. Die Meldung in 57b verschwindet nach zwei Sekunden.



Splash Screen Um dem Nutzer beim Start der App Informationen zur Applikation abzugeben, wurde ein Custom-Splash-Screen hinzugefügt. Dieser ist beim Start der Applikation während drei Sekunden zu sehen, ehe er ausgeblendet wird. Die Implementierung wurde von einem Tutorial übernommen [Jaquez, 2020].

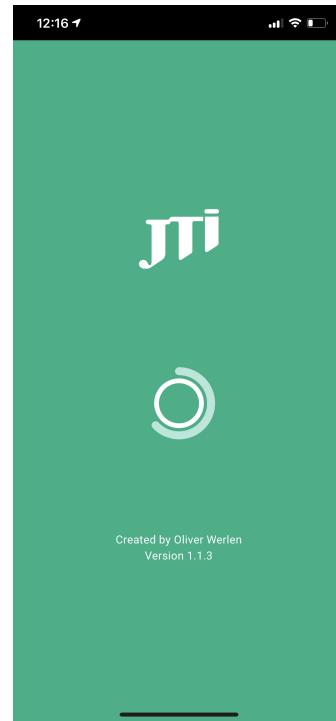


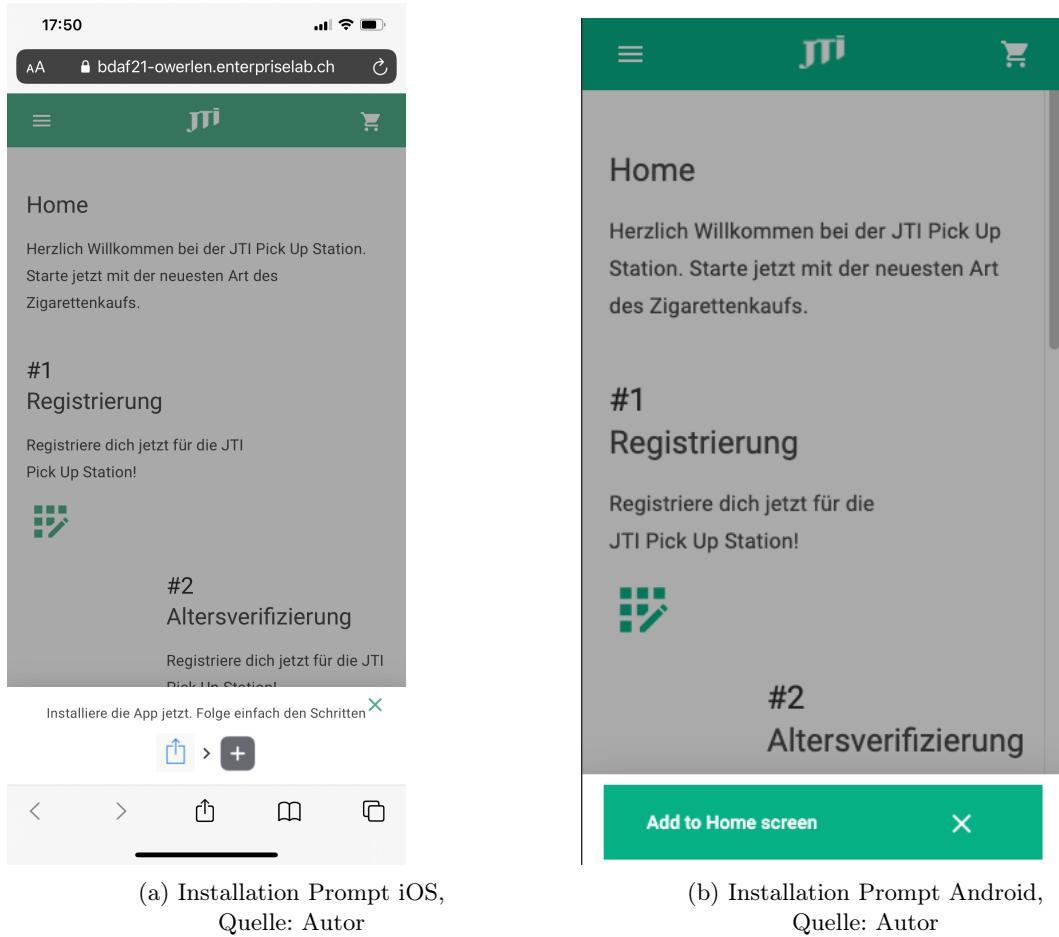
Abbildung 58: Splash Screen beim Starten der PWA,
Quelle: Autor

Bemerkung Der Splash Screen wurde in Version 1.2.1 wieder entfernt.

Favicons Das Favicon muss für verschiedene Anwendungsfälle verfügbar sein. Es dient zum Beispiel als App-Icon bei der Installation oder als Tab-Icon im Browsetab. Um die verschiedenen Icons zu erstellen, finden sich im Internet diverse Anbieter. Die Entscheidung fiel auf "realfavicon-generator". Dieser generiert alle benötigten Icons.

Install Prompt Um dem Nutzer die Möglichkeit zu bieten, die Applikation zu installieren, wurden entsprechende Prompts erstellt.

Android bietet die Möglichkeit, die App direkt von Browser zu installieren. Hier wird der Event abgefangen und eine eigene Meldung ausgegeben. Bei iOS existiert so etwas nicht. Daher wird die Anleitung angezeigt, wie die Applikation installiert werden kann [Korneiko, 2021].



Service Worker Update Der Service Worker bemerkt es, wenn neue Daten vorhanden sind. Es wird zwischen den Events available und activated unterschieden [Angular, 2020].

Lighthouse Um die Eigenschaften der Applikation zu messen, bietet Google Chrome in seinen Developer Tools die Möglichkeit, einen Lighthouse Report zu generieren. Es wird die Performance, Accessibility, Best Practices, Search Engine Optimization (SEO) und die Progressive Web App Funktionalität überprüft.

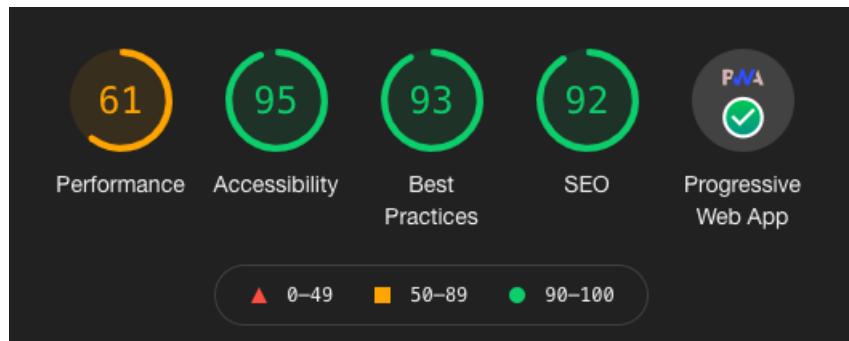


Abbildung 60: Lighthouse Report von der Home Seite,
Quelle: Autor

Der Report für die Seite gibt dabei an, dass die Performance nur mittelmässig ist. Bei einer

genauerer Betrachtung ist dies auf die lange Ladezeit beim Largest Contentful Paint zurückzuführen. Es handelt sich um die Zeit, welche benötigt wird, bis das grösste Element angezeigt wird. Als Hauptgrund für einen hohen Wert wird von Google eine langsame Serverantwortzeiten genannt [rednez, 202]. Die Maschine wird von Enterpriselab zur Verfügung gestellt, weshalb eine Optimierung nicht möglich ist. Die Daten werden jedoch von einem Service Worker gecached, sodass im Betrieb dieser Wert deutlich tiefer ausfallen wird.

Accessibility, Best Practices und SEO sind im grünen Bereich, es handelt sich auch um eine Progressive Web App. Die einzelnen Punkte der Checkliste werden im Kapitel 6 genauer betrachtet.

Sprintreview Sprint 10 Die User Story im Sprint 10 konnte abgeschlossen werden.

5.3.5 Meilenstein Abschluss Realisierungsphase

Meilensteinbericht

Termin Meilenstein 5 Der Meilenstein 5 ist am 24.05.2021 abgeschlossen und somit pünktlich fertiggestellt worden.

Beschreibung Meilenstein 5 Die Beschreibung des Meilensteins ist im Abschnitt B.2.2 ersichtlich.

Meilensteinziele/Vorgaben Das übergeordnete Ziel dieses Meilensteins ist die der Abschluss der Realisierungsphase.

- Testprotokolle
- Demo
- Release 3

Die Testprotokolle sind im Anhang A.1 zu finden.

Meilensteinzielerreichung Der Meilenstein konnte erfolgreich erreicht werden. Die Kommunikation auf der Station ist noch nicht voll funktionsfähig, an dieser wird im letzten Sprint weiter gearbeitet.

Fazit Die Realisierungsphase konnte abgeschlossen werden. In der letzten Projektphase werden letzte Tests durchgeführt sowie ein kompletter Systemdurchlauf durchgeführt.

5.4 Sprint 11

User Story	Number
Das System bietet dem Dienstleister die Möglichkeit, bei geringem Warenbestand eine Benachrichtigung zu senden.	F.14

Tabelle 11: User Stories Sprint 11,

Quelle: Autor

Anzeigen von kritischen Verfügbarkeiten Der Administrator soll zumindest Einsicht haben, wenn ein Produkt eine geringe Verfügbarkeit hat. Umgesetzt wird das durch eine neue Methode auf der REST-Schnittstelle. Diese liefert alle Inventuren zurück, bei denen weniger als eine bestimmte Anzahl Produkte vorhanden ist. Diese Information wird als Tabelle in dem Administratorenbereich angezeigt.

Um eine bessere Übersicht zu haben, wird neu beim Erstellen einer neuen Station per Default für jedes Produkt ein neues Inventar erstellt. Per Default sind ist die Produktanzahl auf 0 gesetzt.

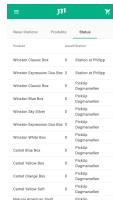


Abbildung 61: Kritische Verfügbarkeiten auf Stations,
Quelle: Autor

Benachrichtigung Die Implementierung einer Benachrichtigung bei einem kritischen Warenbestand könnte mittels Push-Benachrichtigung umgesetzt werden. Ein anderer Ansatz wäre das Senden von einer Email. In beiden Fällen müsste einer Station ein eigener Administrator zugewiesen werden. Im momentanen Modell ist das nicht implementiert, weshalb dieser Punkt nur konzeptuell umgesetzt wird.

Die Implementierung mittels Email wäre einfacher umzusetzen. So könnte nach jeder getätigten Bestellung das Inventar abgefragt werden und im entsprechenden Fall eine Email an die hinterlegte Adresse gesendet werden.

Der Ansatz mittels Push-Benachrichtigung ist technisch anspruchsvoller. So muss gewährleistet sein, dass der Administrator zum Zeitpunkt der Meldung angemeldet ist. Eine andere Lösung wäre das Hinterlegen der Geräte-Id. Der Administrator wäre aber an dieses Gerät gebunden, was ein grosser Nachteil ist.

Die Implementierung mittels Email wird dem anderen Ansatz vorgezogen. Dies könnte im Backend implementiert werden, es wäre das Bereitstellen eines Mail-Servers nötig.

Diverse Bugfixes und UI-Verbesserungen In diesem Sprint wurden zudem diverse Optimierungen vorgenommen. Diese Änderungen wird nachfolgend aufgelistet:

- Warenkorb
- Produktanzeige
- Login- und Registrierungsformular

Sprintreview Sprint 11 Um einen Handlungsbedarf beim Inventar zu bemerken, ist der momentane Ansatz ausreichend. Es ist hier zwar ein regelmässiges Abfragen der Daten nötig, für die prototypische Umsetzung reicht dies aber aus.

5.5 Einführungsphase

In der Einführungsphase wurde die Software mit dem Auftraggeber eingerichtet. Es wurden kleine Punkte zur Verbesserung vorgeschlagen und auch umgesetzt.

In einem ersten Schritt wurde die Software an Ralf Hohermut übergeben. Auf Ralfs Wunsch hin wurden einige Anpassungen vorgenommen. So werden die Preise neu mit CHF bezeichnet. Die SnackBar werden zum Teil von der Anforderung zur Installation überdeckt. Zudem werden die Packshots nicht immer angezeigt und die Grenze bei der Anzeige der Verfügbarkeiten soll neu gesetzt werden.

Die Meisten dieser Punkte waren sehr einfach umzusetzen. Die Anzeige der Packshots funktioniert in seltenen Fällen nicht, es aber kein Muster zu erkennen. Es wird davon ausgegangen, dass es sich hier um einer Fehler vom Angular Service Worker handelt. Die Installationsaufforderung wird

nur noch einmal angezeigt. Bei einem Schliessen von dieser Meldung wird das gespeichert, sie wird nicht erneut angezeigt. Dieser Status wird im Local Storage gespeichert. Die neue Grenze für die Verfügbarkeit liegt bei 3 Produkten. Bei null Produkten wird die Anzeige rot, zwischen 1-3 orange und darüber hinaus grün.

Es wurde der finale Release erstellt. Die restliche Zeit in dieser Projektphase wurde für die Bearbeitung der Dokumentation genutzt.

6 Evaluation und Validation

6.1 Ziel der Arbeit

Durch die progressive Web App «JTI Pick-Up Station» ist es dem Kunden möglich, seine Ware bequem im Onlineshop zu bestellen und direkt und ohne Wartezeit an der gewünschten Pick-Up Station abzuholen. Durch den Prototyp sollen die Funktionalität und Zweckmässigkeit dieses für die Firma neuen Absatzkanals aufgezeigt werden. Im besten Fall findet die Applikation nicht nur in der Schweiz Verwendung, sondern wird von JTI auch in anderen Märkten weltweit eingesetzt. Das Hauptaugenmerk der Arbeit liegt auf der Implementierung eines Prototyps mit den folgenden Schwerpunkten: Bestellung, Kauf, Nutzererfassung, Suche nach Pick-Up Stations und der Abholung an der Station. Bei der Nutzererfassung muss das Alter des Nutzers verifiziert werden. Die Lösung soll so weit als möglich in die Projektpartner-Systeme integriert werden. Hinzu kommt die Recherche von artverwandten Technologien und das Requirements Engineering.

6.2 Validierung der Requirements

Die Requirements werden in der Reihenfolge von D bearbeitet.

PWA-Checklist Es werden die Kriterien von Google genutzt. [Sam Richard, 2020]

Core Progressive Web App checklist

- ✗ Starts fast, stays fast
- ✓ Works in any browser
- ✓ Responsive to any screen size
- ✓ Provides Custom offline page
- ✓ Is Installable

Optimal Progressive Web App checklist Es wurde bei der Implementierung sogar noch weiter gegangen, als dass es dieses Requirement vorschreibt. So wurden die meisten Punkte von der Optimal checklist ebenfalls abgeschlossen.

- ✓ Provides an offline experience
- ✓ Is fully accessible
- ✗ Can be discovered through search
- ✓ Works with any input type
- ✓ Provides context permission requests
- ✓ Follows best practice for healthy code

Abstimmung auf physische Pick-Up Station Während dem gesamten Projekt wurde der Kontakt mit der Elektrotechnik gesucht und von Beginn an die Schnittstelle definiert. Die Kommunikation von der Informatik zur Elektrotechnik bei der Abholung einer Bestellung funktioniert. Auch kann in der entgegengesetzten Richtung eine Verbindung hergestellt werden, es werden aber nicht alle Daten erhalten. Eine Lösung für dieses Problem konnte nicht gefunden werden. Daher muss dieses Requirement als nicht erfüllt angesehen werden.

Registrierung und Login Es ist dem Nutzer möglich, sich zu registrieren und ein Login durchzuführen. Die Umsetzung dieses Mechanismus wurde durch JSON Web Token (JWT) umgesetzt. Damit der Nutzer sich nach der Registrierung anmelden kann, ist es zwingend nötig, die Altersverifikation erfolgreich abzuschliessen. Erst dann ist ein Login möglich.

Altersverifikation

7 Ausblick

8 Verzeichnisse

Abbildungsverzeichnis

1	MyPost 24-Abholstelle	7
2	Tabakkauf avec box	8
5	Standort in der Starbucks Progressive Web App	11
6	Architektur	13
7	Zusammenspiel von zentralen Layern	14
8	Container-Infrastruktur	15
9	GitLab Board	18
10	Domänenmodell	19
11	Ablauf der CI/CD Pipeline	21
12	Beispielanwendung QR-Code auf Gerät	24
13	Verfügbare Shared-Runner von GitLab	27
14	CLI Commands GitLab Container Registry	28
18	Authentication im Backend	33
19	Token Auflösung mit jwt.io	34
21	Authentication Interceptor	36
22	Token im Session Storage	36
23	Ablauf von Timer und anschliessender Logout	36
24	Erstellen des Basic Authentication Token im Back Office	39
25	Saferpay Bezahlseite	40
26	PagePaymentAssertResponse	41
27	Bezahlhistorie Backoffice	42
28	Sequenzdiagramm vom Bezahlvorgang	43
29	Entity Relationship Diagramm von Order	44
30	Response bei Abfrage einer PickUp-Station	45
31	Erstellung von Marker und Tooltip	47
32	Darstellung von Marker und Tooltip	47
33	Observable States	48
34	Mouse Event	48
35	Erstellen der Order	49
36	Erstellen der OrderItems	49
37	Endlicher Zustandsautomat Artikelreservierung	50
39	Auslesen der Werte aus .properties-File	51
40	Berechnung des Order-Totals	51
41	Senden eines Request mit RestTemplate	52
42	Erstellen der HttpEntity	52
43	Überprüfung des Zahlungsstatus	52
44	Initialisierung einer neuen Station	53
45	Sequenzdiagramm der Initialisierung einer neuen Station	53
46	Erstellen eines QR-Code mit ZXing	54
47	QR-Code Beispiel	54
48	PickUp Station Identifier und PickUp Token	54
49	Sequenzdiagramm der Bestellabholung	55
50	Ping zwischen Raspberry Pi und Gitlab VM	56
51	Verbundene Geräte in zerotier	56
52	Initialer Request an Jumio API	57
53	Überschreiben der isEnabled() Methode	58
54	Altersverifikation	58
55	Aktiver Service Worker in Google Chrome	59
56	Monitoring der Connection-Status	60
58	Splash Screen beim Starten der PWA	62

60	Lighthouse Report von der Home Seite	63
61	Kritische Verfügbarkeiten auf Stations	65
62	Organigramm	96
63	Projektstrukturplan	97
64	SoDa Rahmenplan	98
65	Risikomatrix	100
66	Risikomatrix nach Massnahmen	102
67	Systemarchitektur	106
68	Kontextdiagramm	107
69	Datenbankschema	109
70	Aufbau Produktives System	110
71	DTO Klassendiagramm von Martin Fowler	111
72	Richardson Maturity Model	111
73	Ausschnitt aus der Swagger Dokumentation	113
74	SoDa Rahmenplan Version 1	135

Glossar

Angular Framework für Cross Plattform Developing [Böhm, 2017]. 1, 16

DevOps "Der Begriff DevOps setzt sich aus „Dev“ (Development, Entwicklung) und „Ops“ (Operations, Vorgänge) zusammen und vereint Menschen, Prozesse und Technologien, damit Kunden kontinuierlich hochwertige Produkte erhalten.". 17

DockerHub Plattform, um Container Applikationen zu verteilen und zur Verfügung zu stellen.
. 28

Enterprise Lab Das Enterprise Lab stellt Studierenden modernste Computerressourcen zur Verfügung. Das Enterprise Lab ist als Rechenzentrum mittlerer größe aufgebaut und implementiert neueste Technologien [[enterpriselab](#)]. . 14, 25, 29, 55, 56

fair usage algorithmus Die Anzahl der momentan auszuführenden Jobs pro Projekt entscheidet die Auswahl des Runners. Das Projekt mit den am wenigsten laufenden Jobs kommt zuerst [GitLab, 2021].. 27

Github grösste Versionskontrollplattform [GitHub, 2021]. 57, 70

GitLab Versionskontrollsyste aufbauend auf git, alternative zu Github. 1, 17, 18, 26, 28, 29

Local Storage Datenspeicher im Browser, Domänen- und Protokollabhängig, über die Session-dauer hinaus verfügbar. . 32, 51, 52

mixed Content Abfragen einer http-Ressource aus einem https-Context [[mixedContent](#)]. . 32

nginx Open Source Web Server. 28

Progressive Web App Kombination von Webapplikation und nativer Applikation, untersteht den Kriterien von Google PWA-Checklist [Richard, 2020].. 1, 6, 10, 11, 16, 22–24, 59, 63, 64, 68

Propertie-File Datei, welche bei Java die Konfigurationen enthält. Hat die Dateiendung .property. 32

React Java Script Library zum Erstellen von User Interfaces [Facebook, 2021]. 16

RxJS Eine der meistgenutzten Libraries beim Webdevelopment. Ermöglicht den Einsatz von mächtigen funktionale Ansätzen wie Observables [[rxjs](#)]. . 47, 54

Session Storage Datenspeicher im Browser, Domänen- und Protokollabhängig, nur während der Session verfügbar. . 36

SoDa Hybrides Projektmanagementvorgehen der Hochschule Luzern. 1, 18

Spring Java Framework, um schnellere, einfachere und sicherere Java Applikationen zu erstellen. [Waldmann, 2020]. 16

Spring Boot Spring Boot nutzt das Spring-Framework und bietet zusätzlich einen eingebauten Tomcat Server. Mit Spring Boot bleibt Spring auch weiterhin interessant für Java Enterprise Applications. [Waldmann, 2020]. 1, 16

TypeScript Basiert auf JavaScript, fügt zusätzlich statische Typisierung hinzu [Google, 2021e]..
16

User Story Element in SoDa, werden aus Epics im Format Als Rolle möchte ich Ziel/Wunsch, um Nutzen erstellt [t2informatik, 2021]. 18, 24, 29, 31, 48, 59, 64

Web Applikation Archive Enthält alle Inhalte einer Web Applikation, reduziert die Übertragungszeit [javapoint, 2020]. 16

WebHooks WebHooks werden zur Kommunikation zwischen zwei Diensten genutzt. Das Vorgehen ist vergleichbar mit dem aus der Programmierung bekannten Observer Pattern. Es wird das Push-Verhalten genutzt. Der Event wird von der API durch einen POST-Request ausgelöst. Es handelt sich um eine asynchrone Kommunikation [[webhooksExplained](#)]..
29, 55

9 Abkürzungsverzeichnis

SoDa	Software Development Agile.....	22
URL	Uniform Resource Locator.....	58
bzw	beziehungsweise	55
REST	Representational State Transfer	13
API	Application Programming Interface.....	13
HTTPS	Hypertext Transfer Protocol Secure	58
JPA	Java Persistence API.....	16
CI/CD	Continous Integration and Continous Deployment.....	20
JSON	Java Script Object Notation.....	53
CORS	Cross Origin Ressource Sharing.....	43
HATEOAS	Hypermedia as the Engine of Application State	29
JSON	Java Script Object Notation.....	53
JWT	JSON Web Token	68
DTO	Date Transfer Object.....	19
JTI	Japan Tobacco International.....	11
PWA	Progressive Web App	22
NFC	Near Field Communication	23
IoT	Internet of Things	29
PCI DSS	Payment Card Industry Data Security Standard.....	37
SAQ	Self Assigned Questionary	37
CORS	Cross-Origin Ressource Sharing.....	43
JPA	Java Persistence API.....	16
PWA	Progressive Web App	22
SEO	Search Engine Optimization	63
IP	Internet Protocol	52
UART	Univeral Asynchronous Receiver Transmitter	53
UDP	User Datagram Protocol	56
VPN	Virtual Private Network	55

Tabellenverzeichnis

1	User Stories Sprint 1	23
2	User Stories Sprint 2	25
3	User Stories Sprint 4	31
4	User Stories Sprint 5	33
5	User Stories Sprint 6	37
6	User Stories Sprint 7	46
7	User Stories Sprint 8	49
8	User Stories Sprint 9	57
9	Userstories Sprint 10	59
10	Übersicht über die Verwendung von Freshness und Performance	60
11	User Stories Sprint 11	64
12	Testprotokoll Test 1, Quelle: Autor	79
13	Testprotokoll Test 2, Quelle: Autor	80

14	Testprotokoll Test 3, Quelle: Autor	81
15	Testprotokoll Test 4, Quelle: Autor	81
16	Testprotokoll Test 5, Quelle: Autor	82
17	Testprotokoll Test 6, Quelle: Autor	82
18	Testprotokoll Test 7, Quelle: Autor	83
19	Testprotokoll Test 8, Quelle: Autor	83
20	Testprotokoll Test 9, Quelle: Autor	84
21	Testprotokoll Test 10, Quelle: Autor	85
22	Testprotokoll Test 11, Quelle: Autor	86
23	Testprotokoll Test 12, Quelle: Autor	87
24	Testprotokoll Test 13, Quelle: Autor	88
25	Testprotokoll Test 14, Quelle: Autor	89
26	Testprotokoll Test 15, Quelle: Autor	90
27	Testprotokoll Test 15, Quelle: Autor	90
28	Testprotokoll Test 17, Quelle: Autor	91
29	Testprotokoll Test 18, Quelle: Autor	91
30	Testprotokoll Test 19, Quelle: Autor	92
31	Testprotokoll Test 20, Quelle: Autor	92
32	Testprotokoll Test 21, Quelle: Autor	93
33	Testprotokoll Test 22, Quelle: Autor	94
34	Testprotokoll Test 23, Quelle: Autor	95
35	Meilensteine, Quelle: Autor	99
36	Risikoanalyse, Quelle: Autor	100
37	Risikoanalyse nach Massnahmen, Quelle: Autor	101
38	Entwicklungstools, Quelle: Autor	103
39	Konfigurationseinheit Release 1, Quelle: Autor	103
40	Konfigurationseinheit Release 2, Quelle: Autor	103
41	Konfigurationseinheit Release 3, Quelle: Autor	104
42	Funktionale Anforderungen, Quelle: Autor	116
43	Nicht Funktionale Anforderungen, Quelle: Autor	117
44	Änderungshistorie der Anforderungen, Quelle: Autor	118
45	Sitzungsprotokoll, Quelle: Autor	119
46	Sitzungsprotokoll, Quelle: Autor	121
47	Sitzungsprotokoll, Quelle: Autor	123
48	Sitzungsprotokoll, Quelle: Autor	124
49	Sitzungsprotokoll, Quelle: Autor	125
50	Sitzungsprotokoll, Quelle: Autor	127
51	Sitzungsprotokoll, Quelle: Autor	128
52	Sitzungsprotokoll, Quelle: Autor	129
53	Sitzungsprotokoll, Quelle: Autor	130
54	Sitzungsprotokoll, Quelle: Autor	131
55	Sitzungsprotokoll, Quelle: Autor	132
56	Sitzungsprotokoll, Quelle: Autor	133
57	Sitzungsprotokoll, Quelle: Autor	134

Literatur

- AG, D. S. P. (2021a). Standorte und Öffnungszeiten. Zugriff unter https://places.post.ch/?PreselectText=&topic=1&from_directory=True&lang=de&service=places. (03.03.2021)
- AG, D. S. P. (2021b). Versenden und empfangen rund um die Uhr. Zugriff unter <https://www.post.ch/de/empfangen/empfangsorte/pickpost-my-post-24/my-post-24>. (03.03.2021)
- Agafonkin, V. (2020). an open-source JavaScript library for mobile-friendly interactive maps. Zugriff unter <https://leafletjs.com/index.html>. (23.04.2021)
- Angular. (2020). Service worker communication. Zugriff unter <https://angular.io/guide/service-worker-communications#service-worker-communication>. (14.05.2021)
- Apple. (2021). Tracking Prevention in WebKit. Zugriff unter <https://webkit.org/tracking-prevention/>. (12.03.2021)
- Avram, A. & Marinescu, F. (2006). Domain Driven Design Quickly. C4Media Inc. Zugriff unter <https://www.infoq.com/minibooks/domain-driven-design-quickly/baeldung>.
- baeldung. (2020). Generating Barcodes and QR Codes in Java. Zugriff unter <https://www.baeldung.com/java-generating-barcodes-qr-codes>. (02.05.2021)
- baeldung. (2021). The Guide to RestTemplate. Zugriff unter <https://www.baeldung.com/rest-template>. (27.04.2021)
- Beni, E. H. (2021). Spring Boot Secured By Let's Encrypt. Zugriff unter <https://dzone.com/articles/spring-boot-secured-by-lets-encrypt>. (06.04.2021)
- bezkoder. (2020). Angular 8 JWT Auth – Token based Authentication with Web Api example. Zugriff unter <https://bezkoder.com/spring-boot-jwt-authentication/>. (06.04.2021)
- bezkoder. (2021). Spring Boot Token based Authentication with Spring Security and JWT. Zugriff unter <https://bezkoder.com/spring-boot-jwt-authentication/>. (06.04.2021)
- Böhm, R. (2017). Was sind Angular und Angular JS. Zugriff unter <https://angular.de/artikel/was-ist-angular/>. (14.03.2021)
- Bohdorf, A. (o.D.). DOCKER-COMPOSE SETUP MIT NGINX REVERSE PROXY. Zugriff unter <https://sitegeist.de/blog/typo3-blog/docker-compose-setup-mit-nginx-reverse-proxy.html>. (22.03.2021)
- congstar. (2021). NFC-Near Field Communication. Zugriff unter <https://www.congstar.de/handys/technik-news-trends/nfc/>. (12.03.2021)
- Coop. (2020). Lieferbedingungen. Zugriff unter <https://www.coop.ch/de/wie-wir-liefern.html>. (03.03.2021)
- cozmo. (2021). jsQR. Zugriff unter <https://github.com/cozmo/jsQR>. (04.05.2021)
- Doran, T. (2018). IEEE/ISO/IEC 29148-2018 - ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes – Requirements engineering. Zugriff unter <https://standards.ieee.org/standard/29148-2018.html>. (27.02.2021)
- DB-Engines. (2021). Vergleich der Systemeigenschaften MariaDB vs. PostgreSQL vs. MySQL. Zugriff unter <https://db-engines.com/de/system/MariaDB%3BPostgreSQL>. (14.03.2021)
- Facebook. (2021). React. Zugriff unter <https://reactjs.org/>. (14.03.2021)
- Fowler, M. (2002). Data Transfer Object. Zugriff unter <https://martinfowler.com/eaaCatalog/dataTransferObject.html>. (7.05.2021)
- Fowler, M. (2010). Richardson Maturity Model. Zugriff unter <https://martinfowler.com/articles/richardsonMaturityModel.html>. (06.05.2021)
- Genf, K. (o.D.). Versand und Zahlungsbedingungen. Zugriff unter <https://www.kiosklino.ch/de/versand-und-zahlungsbedingungen>. (03.03.2021)
- Geoapify. (2020). Angular + Leaflet: step by step tutorial to add a map. Zugriff unter <https://www.geoapify.com/angular-leaflet-step-by-step-tutorial-to-add-a-map>. (23.04.2021)
- Gisler, R. (2020). *Applikationsentwicklung*. Hochschule Luzern.
- GitHub. (2021). Where the world builds software. Zugriff unter <https://github.com/about>. (03.03.2021)
- GitLab. (2021). Configuring runners in GitLab. Zugriff unter <https://docs.gitlab.com/ee/ci/runners/>. (18.03.2021)
- Google. (2021a). Angular Material. Zugriff unter <https://material.angular.io/>. (14.03.2021)

- Google. (2021b). Angular Service Worker Introduction. Zugriff unter <https://angular.io/guide/service-worker-intro>. (14.03.2021)
- Google. (2021c). Getting started with service workers. Zugriff unter <https://angular.io/guide/service-worker-getting-started>. (08.05.2021)
- Google. (2021d). Managing data. Zugriff unter <https://angular.io/start/start-data>. (03.03.2021)
- Google. (2021e). Typed JavaScript at Any Scale. Zugriff unter <https://www.typescriptlang.org/>. (27.05.2021)
- Hat, R. (2021). Hibernate ORM. Zugriff unter <https://hibernate.org/orm/>. (06.05.2021)
- HSLU. (o.D. a). Artefakte und Downloads Planungs- und Entwurfsdokumente. Zugriff unter <https://www.hslu.ch/de-ch/informatik/studium/soda/artefakte-und-downloads/>. (02.03.2021)
- HSLU. (o.D. b). Planung und Vorgehen «Lieber ungefähr richtig, als genau falsch». Zugriff unter <https://www.hslu.ch/de-ch/informatik/studium/soda/planung/>. (02.03.2021)
- Jaquez, R. (2020). Simple Splash Screen for your Angular Web Apps and PWAs. Zugriff unter <https://itnext.io/simple-splash-screen-for-your-angular-web-apps-and-pwas-f4fbf897540b>. (08.05.2021)
- javapoint. (2020). War-File. Zugriff unter <https://www.javatpoint.com/war-file>. (03.03.2021)
- KnowHow. (2020). Reverse-Proxy-Server-Kernkomponente in Sicherheitsarchitekturen. Zugriff unter <https://www.ionos.de/digitalguide/server/knowhow/was-ist-ein-reverse-proxy/>. (22.03.2021)
- Korneiko, O. (2021). A way to show a prompt to install your Angular PWA both on Android and iOS devices. Zugriff unter <https://medium.com/@oleksandr.k/a-way-to-show-a-prompt-to-install-your-angular-pwa-both-on-android-and-ios-devices-7a770ff55c54>. (14.05.2021)
- Krüger, N. (2018). How to Write a Software Requirements Specification (SRS Document). Zugriff unter <https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document>. (27.02.2021)
- Lenzo, M. (2016). Continuous delivery of a Spring Boot application with GitLab CI and Kubernetes. Zugriff unter <https://about.gitlab.com/blog/2016/12/14/continuous-delivery-of-a-spring-boot-application-with-gitlab-ci-and-kubernetes/>. (18.03.2021)
- Letsencrypt. (2017). Zertifikate für localhost. Zugriff unter <https://letsencrypt.org/de/docs/certificates-for-localhost/>. (19.03.2021)
- Mozilla. (2020). MediaStreamConstraints. Zugriff unter <https://developer.mozilla.org/en-US/docs/Web/API/MediaStreamConstraints>. (14.05.2021)
- Mozilla. (2021). Web NFC API. Zugriff unter https://developer.mozilla.org/en-US/docs/Web/API/Web_NFC_API#browser_compatibility. (12.03.2021)
- mozilla. (2021). Cross-Origin Resource Sharing (CORS). Zugriff unter <https://developer.mozilla.org/de/docs/Web/HTTP/CORS>. (19.03.2021)
- OpenStreetMap. (2021). OpenStreetMap provides map data for thousands of web sites, mobile apps, and hardware devices. Zugriff unter <https://www.openstreetmap.org/about>. (26.04.2021)
- Patel, S. (2020). A Simple Cors Proxy for Javascript Browser applications. Zugriff unter <https://medium.com/nodejsmadeeasy/a-simple-cors-proxy-for-javascript-applications-9b36a8d39c51>. (19.03.2021)
- Post-CH-AG. (2015). *Allgemeine Geschäftsbedingungen PickPost und My Post 24*. Zugriff unter <https://www.post.ch/de/empfangen/empfangsorte/pickpost-my-post-24/my-post-24/weitere-informationen>
- RedHat. (2021). Docker - Funktionsweise, Vorteile, Einschränkungen. Zugriff unter <https://www.redhat.com/de/topics/containers/what-is-docker>. (14.03.2021)
- rednez. (2021). Houssein Djirdeh. Zugriff unter <https://web.dev/optimize-lcp/>. (18.05.2021)
- rednez. (2021). angular-user-idle. Zugriff unter <https://www.npmjs.com/package/angular-user-idle>. (08.05.2021)
- Richard, S. (2020). What are Progressive Web Apps. Zugriff unter <https://web.dev/what-are-pwas/>. (29.05.2021)
- Ryte. (2021). Mobile First. Zugriff unter https://de.ryte.com/wiki/mobile-first#mobile-first-2c_google_und_seo. (27.04.2021)

- saferpay. (2021a). All the relevant information for developers. Zugriff unter <https://www.six-payment-services.com/en/site/e-commerce-developer/integration.html>. (12.03.2021)
- saferpay. (2021b). Saferpay Integration Guide. Zugriff unter <https://saferpay.github.io/sndbx/index.html>. (12.03.2021)
- Sam Richard, P. L. (2020). What makes a good Progressive Web App? Zugriff unter <https://web.dev/pwa-checklist/>. (24.02.2021)
- Schulte, J. (2019). Alles Wissenswerte über Single Page Applications. Zugriff unter https://www.magnolia-cms.com/de_DE/blog/alles-wissenswerte-ueber-single-page-applications.html. (25.05.2021)
- SimiCart. (2021). 12 Best Examples of Progressive Web Apps (PWAs) in 2021. Zugriff unter <https://www.simicart.com/blog/progressive-web-apps-examples/>. (03.04.2021)
- Spepanov, R. (2020). Openlayers vs Leaflet: What Makes One Better Than the Other. Zugriff unter <https://mapsvg.com/blog/openlayers-vs-leaflet>. (23.04.2021)
- Starbucks. (2021a). app.starbucks. Zugriff unter <https://app.starbucks.com>. (09.03.2021)
- Starbucks. (2021b). app.starbucks. Zugriff unter <https://app.starbucks.com/menu>. (09.03.2021)
- Starbucks. (2021c). app.starbucks. Zugriff unter <https://app.starbucks.com/store-locator?map=46.670396,7.455289,10z>. (09.03.2021)
- t2informatik. (2021). User Story. Zugriff unter <https://t2informatik.de/wissen-kompakt/user-story/>. (03.03.2021)
- tagmotion. (2018). NFC vs. QR-Code. Zugriff unter <https://www.tagmotion.de/nfc-vs-qr-code/>. (12.03.2021)
- Valora. (2021a). Avec Now. Zugriff unter <https://www.avecnow.ch>. (09.03.2021)
- Valora. (2021b). Bewährtes und Neues nur für dich. Zugriff unter <https://avec.ch/de/avecbox/>. (09.03.2021)
- Valora. (2021c). Scan ID. Zugriff unter Kontoerstellung. (09.03.2021)
- Valora. (2021d). Standorte und Öffnungszeiten. Zugriff unter <https://avec.ch/de/avecbox/>. (09.03.2021)
- Valora. (2021e). Über avec now. Zugriff unter <https://www.avecnow.ch/pages/uber-avec-now>. (09.03.2021)
- Valora. (2021f). Versand. Zugriff unter <https://www.avecnow.ch/policies/shipping-policy>. (09.03.2021)
- VMWare. (2021). Spring Boot with Docker. Zugriff unter <https://spring.io/guides/gs/spring-boot-docker/>. (18.03.2021)
- von Uslar, C. (2021a). private email communication.
- von Uslar, C. (2021b). private Teams communication.
- Waldmann, S. (2020). Spring oder Spring Boot, das ist hier die Frage. Zugriff unter <https://blog.doubleslash.de/spring-vs-spring-boot/>. (14.03.2021)
- Yamaguchi, N. (2020). Angular jsqr. Zugriff unter <https://stackblitz.com/edit/angular-jsqr?file=package.json>. (04.05.2021)

A Testprotokolle

A.1 Testprotokolle Bestellung

Test Nr.	1
Beschreibung	Durch diesen Test wird die Registrierung eines neuen Kunden getestet.
Randbedingungen	<ul style="list-style-type: none"> • Die Testperson hat sich mit ihrer Email-Adresse und Benutzernamen noch nie registriert. • Die Testperson nutzt den Nutzernamen „test“ , die Email-Adresse „test@gmail.com“ und das Passwort „ABC*1234“ ein.
erwartete Resultate	<ul style="list-style-type: none"> • Das Konto der Testperson wird korrekt angelegt. • Ein Popup zeigt dem Nutzer den Status an.
Testperson	Oliver Werlen
Datum	18.04.2021
Durchführung	<ol style="list-style-type: none"> 1. Die Testperson gibt die angegebenen Daten ein. Die restlichen Daten werden zufällig gewählt. 2. Die Testperson klickt auf den Button "Registrieren". 3. Es wird in einem Dialog die Erstellung des Nutzers dargestellt. 4. Es wird automatisch zur Login Page gewechselt.
erhaltenes Resultat	<ul style="list-style-type: none"> • Das Erstellen des Nutzers wurde vom System via Pop-Up bestätigt.
Test bestanden	Ja

Tabelle 12: Testprotokoll Test 1, Quelle: Autor

Test Nr.	2
Beschreibung	Durch diesen Test wird die Registrierung eines neuen Kunden getestet, wobei die Email Adresse und der Nutzernname bereits genutzt werden.
Randbedingungen	<ul style="list-style-type: none"> • Der Test 12 ist erfolgreich durchgeführt worden. • Die Testperson nutzt den Nutzernamen test , die Email-Adresse test@gmail.com und das Passwort ABC*1234 ein.
erwartete Resultate	<ul style="list-style-type: none"> • Das System gibt dem Nutzer die Antwort, dass ein Benutzer mit dieser Email oder Benutzernamen bereits existiert.
Testperson	Oliver Werlen
Datum	18.04.2021
Durchführung	<ol style="list-style-type: none"> 1. Die Testperson gibt die angegebenen Daten ein. Die restlichen Daten werden zufällig gewählt. 2. Die Testperson klickt auf den Button "Registrieren". 3. Es wird in einem Dialog mit der Meldung "Benutzer mit dieser Email oder Benutzernamen existiert bereits".
erhaltenes Resultat	<ul style="list-style-type: none"> • Das Popup wird wie geplant angezeigt.
Test bestanden	Ja

Tabelle 13: Testprotokoll Test 2, Quelle: Autor

Test Nr.	3
Beschreibung	Durch diesen Test wird die Login-Funktion getestet.
Randbedingungen	<ul style="list-style-type: none"> Der Test 12 ist erfolgreich durchgeführt worden. Die Testperson nutzt den Benutzernamen test und das Passwort ABC*1234.
erwartete Resultate	<ul style="list-style-type: none"> Eine Meldung wird angezeigt, mit welcher das erfolgreiche Erstellen des Nutzers bestätigt wird. Im Sidenav wird der Logout Button angezeigt.
Testperson	Oliver Werlen
Datum	18.04.2021
Durchführung	<ol style="list-style-type: none"> Die Testperson gibt die angegebenen Daten ein. Der Benutzer klickt auf den Login Button.
erhaltenes Resultat	<ul style="list-style-type: none"> Das Popup wird wie geplant angezeigt. Das Sidenav wird entsprechend angepasst.
Test bestanden	Ja

Tabelle 14: Testprotokoll Test 3, Quelle: Autor

Test Nr.	4
Beschreibung	Durch diesen Test wird die Login-Funktion getestet. Es wird eine falsche Kombination eingetragen.
Randbedingungen	<ul style="list-style-type: none"> Die Testperson nutzt den Benutzernamen „testNoAccess“ und das Passwort „ABC*1234*noAccess“.
erwartete Resultate	<ul style="list-style-type: none"> Eine Meldung Falscher Benutzername oder Passwort wird angezeigt
Testperson	Oliver Werlen
Datum	18.04.2021
Durchführung	<ol style="list-style-type: none"> Die Testperson gibt die angegebenen Daten ein. Der Benutzer klickt auf den Login Button
erhaltenes Resultat	<ul style="list-style-type: none"> Das Popup wird wie geplant angezeigt.
Test bestanden	Ja

Tabelle 15: Testprotokoll Test 4, Quelle: Autor

Test Nr.	5
Beschreibung	Durch diesen Test wird die Logout Funktion getestet.
Randbedingungen	<ul style="list-style-type: none"> Die Testperson ist erfolgreich eingeloggt.
erwartete Resultate	<ul style="list-style-type: none"> Popup mit der Meldung erfolgreich ausgeloggt.
Testperson	Oliver Werlen
Datum	18.04.2021
Durchführung	<ol style="list-style-type: none"> Die Testperson klickt auf den Logout Button.
erhaltenes Resultat	<ul style="list-style-type: none"> Das Popup wird wie geplant angezeigt.
Test bestanden	Ja

Tabelle 16: Testprotokoll Test 5, Quelle: Autor

Test Nr.	6
Beschreibung	Durch diesen Test wird das Hinzufügen von Produkten in den Warenkorb getestet.
Randbedingungen	<ul style="list-style-type: none"> Es sind Artikel im Shop vorhanden
erwartete Resultate	<ul style="list-style-type: none"> Der Artikel ist im Warenkorb vorhanden.
Testperson	Oliver Werlen
Datum	18.04.2021
Durchführung	<ol style="list-style-type: none"> Die Testperson befindet sich auf der Produktübersicht. Die Testperson klickt bei einem zufälligen Produkt auf das Warenkorbsymbol unten rechts. Die Person klickt ein anderes Produkt an. Die Testperson klickt auf den Button „in den Warenkorb“. Die Testperson wechselt zum Warenkorb (Item oben rechts).
erhaltenes Resultat	<ul style="list-style-type: none"> Die beiden Produkte sind im Warenkorb zu finden. Die beiden Produkte befinden sich in einfacher Ausführung im Warenkorb. Das Total ist korrekt summiert worden.
Test bestanden	Ja

Tabelle 17: Testprotokoll Test 6, Quelle: Autor

Test Nr.	7
Beschreibung	Durch diesen Test wird die Persistierung des Warenkorbs getestet.
Randbedingungen	<ul style="list-style-type: none"> Der Test 17 ist erfolgreich abgeschlossen worden.
erwartete Resultate	<ul style="list-style-type: none"> Die Artikel sind auch nach dem Verlassen der Webseite und einem erneuten Aufruf immer noch vorhanden.
Testperson	Oliver Werlen
Datum	18.04.2021
Durchführung	<ol style="list-style-type: none"> Die Testperson befindet sich im Warenkorb. Die Testperson schliesst das Browserfenster. Die Testperson öffnet das Browserfenster wieder und navigiert zum Warenkorb. Der Warenkorbinhalt bleibt bestehen.
erhaltenes Resultat	<ul style="list-style-type: none"> Die Produkte sind immer noch im Warenkorb.
Test bestanden	Ja

Tabelle 18: Testprotokoll Test 7, Quelle: Autor

Test Nr.	8
Beschreibung	Durch diesen Test werden die Funktionen des Warenkorbs getestet.
Randbedingungen	<ul style="list-style-type: none"> Der Test 17 ist erfolgreich abgeschlossen worden.
erwartete Resultate	<ul style="list-style-type: none"> Die Artikelanzahl wird erhöht. Das Gesamttotal wird erhöht. Die Artikelanzahl wird reduziert. Beim Erreichen von 0 wird der Artikel entfernt.
Testperson	Oliver Werlen
Datum	18.04.2021
Durchführung	<ol style="list-style-type: none"> Die Testperson befindet sich im Warenkorb. Die Testperson erhöht die Artikelanzahl auf 4. Die Testperson reduziert die Artikelanzahl auf 0.
erhaltenes Resultat	<ul style="list-style-type: none"> Es befinden sich keine Artikel mehr im Warenkorb. Es wurde ein Popup mit einer entsprechenden Meldung angezeigt.
Test bestanden	Ja

Tabelle 19: Testprotokoll Test 8, Quelle: Autor

Test Nr.	9
Beschreibung	Durch diesen Test wird die Bezahlfunktion getestet.
Randbedingungen	<ul style="list-style-type: none"> • Der Benutzer ist erfolgreich eingeloggt. • Der Benutzer hat Produkte im Warenkorb.
erwartete Resultate	<ul style="list-style-type: none"> • Der Benutzer wird auf die Bezahlseite weitergeleitet. • Das Gesamttotal wird bei der Bezahlung korrekt angezeigt. • Der Benutzer wird nach dem Bezahlabschluss auf eine entsprechende Seite weitergeleitet.
Testperson	Oliver Werlen
Datum	18.04.2021
Durchführung	<ol style="list-style-type: none"> 1. Die Testperson befindet sich im Warenkorb. 2. Die Testperson klickt auf checkout. 3. Die Testperson wird auf die Bezahlseite umgeleitet. 4. Die Testperson klickt durch den Bezahlprozess.
erhaltenes Resultat	<ul style="list-style-type: none"> • Der Testperson wird eine Meldung zur Bezahlbestätigung ausgegeben.
Test bestanden	Ja

Tabelle 20: Testprotokoll Test 9, Quelle: Autor

Test Nr.	10
Beschreibung	Durch diesen Test wird die Bezahlfunktion bei einem Abbruch getestet.
Randbedingungen	<ul style="list-style-type: none"> • Der Benutzer ist erfolgreich eingeloggt. • Der Benutzer hat Produkte im Warenkorb.
erwartete Resultate	<ul style="list-style-type: none"> • Der Benutzer wird auf die Bezahlseite weitergeleitet. • Das Gesamttotal wird bei der Bezahlung korrekt ausgegeben. • Der Benutzer bricht den Bezahlvorgang mittels Cancel Button ab. • Der Benutzer wird auf eine entsprechende Seite weitergeleitet.
Testperson	Oliver Werlen
Datum	18.04.2021
Durchführung	<ol style="list-style-type: none"> 1. Die Testperson befindet sich im Warenkorb. 2. Die Testperson klickt auf checkout. 3. Die Testperson wird auf die Bezahlseite umgeleitet. 4. Die Testperson klickt beim Bezahlvorgang auf den cancel Button.
erhaltenes Resultat	<ul style="list-style-type: none"> • Der Testperson wird auf eine Page not found Seite umgeleitet.
Test bestanden	Ja

Tabelle 21: Testprotokoll Test 10, Quelle: Autor

A.2 Testprotokolle Abschluss Realisierungsphase

Es werden die Features getestet, welche in A.1 noch nicht getestet wurden.

A.2.1 Altersverifikation

Test Nr.	11
Beschreibung	Durch diesen Test wird die Altersverifikation getestet.
Randbedingungen	<ul style="list-style-type: none"> • Der Benutzer hat erfolgreich die Registrierungsdaten eingegeben und befindet sich am Beginn der Altersverifikation. • Die Person ist über 18 Jahre alt.
erwartete Resultate	<ul style="list-style-type: none"> • Der Benutzer muss seinen Ausweis einlesen, • Der Benutzer wird nach dem Abschluss der Verifikation auf die Zielseite weitergeleitet. • Der Benutzer kann sich mit Email und Passwort einloggen.
Testperson	Oliver Werlen
Datum	25.05.2021
Durchführung	<ol style="list-style-type: none"> 1. Die Testperson hat die Registrierung abgeschlossen. 2. Die Testperson befindet sich auf dem Jumio-Startbildschirm. 3. Die Testperson lädt ein Foto von der Ausweis Vorder- und Rückseite hoch. 4. Die Testperson scannt ihr Gesicht ein. 5. Die Testperson wird auf die Zielseite weitergeleitet. 6. Die Testperson kann sich mit seinen Daten einloggen.
erhaltenes Resultat	<ul style="list-style-type: none"> • Das erwartete Resultat ist eingetreten.
Test bestanden	Ja

Tabelle 22: Testprotokoll Test 11, Quelle: Autor

Test Nr.	12
Beschreibung	Durch diesen Test wird die Altersverifikation getestet, wenn die Person nicht 18 Jahre alt ist.
Randbedingungen	<ul style="list-style-type: none"> • Der Benutzer hat erfolgreich die Registrierungsdaten eingegeben und befindet sich am Beginn der Altersverifikation. • Die Person ist unter 18 Jahre alt.
erwartete Resultate	<ul style="list-style-type: none"> • Der Benutzer muss seinen Ausweis einlesen. • Der Benutzer wird nach dem Abschluss der Verifikation auf die Zielseite weitergeleitet. Er wird auf die Fehlerseite der Altersverifikation weitergeleitet. • Der Benutzer kann sich nicht mit Email und Passwort einloggen.
Testperson	Elena Nujic
Datum	31.05.2021
Durchführung	<ol style="list-style-type: none"> 1. Die Testperson hat die Registrierung abgeschlossen. 2. Die Testperson befindet sich auf dem Jumio-Startbildschirm. 3. Die Testperson lädt ein Foto von der Ausweis Vorder- und Rückseite hoch. 4. Die Testperson scannt ihr Gesicht ein. 5. Die Testperson wird auf die Fehlerseite der Altersverifikation weitergeleitet. 6. Die Testperson kann sich nicht mit seinen Daten einloggen.
erhaltenes Resultat	<ul style="list-style-type: none"> • Das erwartete Resultat ist eingetreten.
Test bestanden	Ja

Tabelle 23: Testprotokoll Test 12, Quelle: Autor

Test Nr.	13
Beschreibung	Durch diesen Test wird die Altersverifikation getestet, wenn die Person einen Ausweis einer anderen Person benutzt.
Randbedingungen	<ul style="list-style-type: none"> • Der Benutzer hat erfolgreich die Registrierungsdaten eingegeben und befindet sich am Beginn der Altersverifikation. • Die Person besitzt einen Ausweis von einer mindestens 18 Jahre alten Person, die nicht sie selbst ist.
erwartete Resultate	<ul style="list-style-type: none"> • Der Benutzer muss den Ausweis einlesen. • Der Benutzer wird nach dem Abschluss der Verifikation auf die Zielseite weitergeleitet. Er wird auf die Fehlerseite der Altersverifikation weitergeleitet. • Der Benutzer kann sich nicht mit Email und Passwort einloggen.
Testperson	Elena Nujic
Datum	31.05.2021
Durchführung	<ol style="list-style-type: none"> 1. Die Testperson befindet sich auf dem Jumio-Startbildschirm. 2. Die Testperson lädt ein Foto von der Ausweis Vorder- und Rückseite hoch. 3. Die Testperson scannt ihr Gesicht ein. 4. Die Testperson wird auf die Fehlerseite der Altersverifikation weitergeleitet. 5. Die Testperson kann sich nicht mit seinen Daten einloggen.
erhaltenes Resultat	<ul style="list-style-type: none"> • Das erwartete Resultat ist eingetreten.
Test bestanden	Ja

Tabelle 24: Testprotokoll Test 13, Quelle: Autor

A.2.2 Station und Kartenfunktionalität

Test Nr.	14
Beschreibung	Durch diesen Test wird die Anzeige einer Station und die Produktverfügbarkeit an einer Station getestet.
Randbedingungen	<ul style="list-style-type: none"> • Es ist eine Station mit Artikeln verfügbar.
erwartete Resultate	<ul style="list-style-type: none"> • Dem Benutzer wird die Station auf der Karte angezeigt • Bei einem Klick auf die Station erscheint ein Popup mit den Produktverfügbarkeiten. • Bei einem Klick auf Station auswählen wird die Station ausgewählt und auf die Produktseite gewechselt. • Auf der Produktseite wird die Verfügbarkeit ebenfalls angezeigt.
Testperson	Oliver Werlen
Datum	31.05.2021
Durchführung	<ol style="list-style-type: none"> 1. Die Testperson wählt Nächste Station auswählen 2. Die Testperson wählt eine Station aus. Das Popup erscheint. 3. Die Testperson klickt auf Station auswählen. 4. Die Testperson wird auf die Produktseite weitergeleitet. Die Verfügbarkeiten werden angezeigt. 5. Die Testperson klickt auf ein Produkt. Auch hier wird die Verfügbarkeit identisch angezeigt.
erhaltenes Resultat	<ul style="list-style-type: none"> • Das erwartete Resultat ist eingetreten.
Test bestanden	Ja

Tabelle 25: Testprotokoll Test 14, Quelle: Autor

Test Nr.	15
Beschreibung	Durch diesen Test wird die Kartenanzeige bei ausgeschalteten System-Preferences getestet.
Randbedingungen	<ul style="list-style-type: none"> • In den Systempreferences ist die Standortberechtigung für den Browser deaktiviert.
erwartete Resultate	<ul style="list-style-type: none"> • Dem Benutzer wird die Karte mit einer Default-Initial-Position angezeigt.
Testperson	Oliver Werlen
Datum	31.05.2021
Durchführung	<ol style="list-style-type: none"> 1. Die Testperson deaktiviert den Zugriff auf den Standort für den gewählten Browser. 2. Der Testperson wird die Karte angezeigt. Der Fokus liegt im Wallis.
erhaltenes Resultat	<ul style="list-style-type: none"> • Das erwartete Resultat ist eingetreten.
Test bestanden	Ja

Tabelle 26: Testprotokoll Test 15, Quelle: Autor

Test Nr.	16
Beschreibung	Durch diesen Test wird die Kartenanzeige bei blockiertem Standortzugriff im Browser getestet.
Randbedingungen	<ul style="list-style-type: none"> • Es sind keine Randbedingungen bekannt.
erwartete Resultate	<ul style="list-style-type: none"> • Dem Benutzer wird die Karte mit einer Default-Initial-Position angezeigt.
Testperson	Oliver Werlen
Datum	31.05.2021
Durchführung	<ol style="list-style-type: none"> 1. Die Testperson wählt im Popup zur Berechtigung auf den Standort Block.
erhaltenes Resultat	<ul style="list-style-type: none"> • Das erwartete Resultat ist eingetreten.
Test bestanden	Ja

Tabelle 27: Testprotokoll Test 15, Quelle: Autor

Test Nr.	17
Beschreibung	Durch diesen Test wird die Produktverfügbarkeit ohne ausgewählte Station getestet.
Randbedingungen	<ul style="list-style-type: none"> Der Benutzer öffnet eine neue Instanz der App ohne gecachete Werte.
erwartete Resultate	<ul style="list-style-type: none"> Alle Produktverfügbarkeiten werden rot markiert.
Testperson	Oliver Werlen
Datum	31.05.2021
Durchführung	<ol style="list-style-type: none"> Die Testperson klickt direkt nach dem Starten der App auf die Produktansicht.
erhaltenes Resultat	<ul style="list-style-type: none"> Die Produktverfügbarkeit ist bei jedem Produkt rot. Die Produkte können nicht dem Warenkorb hinzugefügt werden.
Test bestanden	Ja

Tabelle 28: Testprotokoll Test 17, Quelle: Autor

Test Nr.	18
Beschreibung	Durch diesen Test wird der Bestellabschluss ohne aktive Station oder ohne Produkte getestet.
Randbedingungen	<ul style="list-style-type: none"> Der Benutzer öffnet eine neue Instanz der App ohne gecachete Werte. Vorgängig werden beliebige Produkte dem Warenkorb hinzugefügt.
erwartete Resultate	<ul style="list-style-type: none"> Ohne aktive Station wird eine entsprechende Meldung ausgegeben. Ohne Produkte wird eine entsprechende Meldung ausgegeben.
Testperson	Oliver Werlen
Datum	31.05.2021
Durchführung	<ol style="list-style-type: none"> Die Testperson klickt direkt auf den Warenkorb und den Checkout-Button. Eine entsprechende Meldung wird ausgegeben. Die Testperson löscht alle Artikel aus dem Warenkorb. Er wählt eine Station aus. Beim versuchten Checkout wird eine entsprechende Meldung ausgegeben.
erhaltenes Resultat	<ul style="list-style-type: none"> Die entsprechenden Meldungen wurden ausgegeben.
Test bestanden	Ja

Tabelle 29: Testprotokoll Test 18, Quelle: Autor

A.2.3 Inventur

Test Nr.	19
Beschreibung	Durch diesen Test wird die Überprüfung der entsprechenden Verfügbarkeiten im Warenkorb getestet.
Randbedingungen	<ul style="list-style-type: none"> • Der Benutzer besitzt einen aktivierten Account. • Die Station ist mit 10 verfügbaren Produkten gefüllt. • Der Benutzer hat eine Station ausgewählt.
erwartete Resultate	<ul style="list-style-type: none"> • Die Testperson kann maximal 10 Produkte dem Warenkorb hinzufügen.
Testperson	Oliver Werlen
Datum	31.05.2021
Durchführung	<ol style="list-style-type: none"> 1. Die Testperson erhöht im Warenkorb die Anzahl.
erhaltenes Resultat	<ul style="list-style-type: none"> • Bei 10 Produkten wird eine entsprechende Meldung ausgegeben. Ein weiteres Hinzufügen von diesem Produkt ist nicht möglich.
Test bestanden	Ja

Tabelle 30: Testprotokoll Test 19, Quelle: Autor

Test Nr.	20
Beschreibung	Durch diesen Test wird die Aktualisierung des Warenbestandes getestet.
Randbedingungen	<ul style="list-style-type: none"> • Der Testfall 18 ist abgeschlossen worden und 10 Produkte von einer Sorte bestellt worden.
erwartete Resultate	<ul style="list-style-type: none"> • Das Inventar auf der Karte, bzw. der Produktübersicht ist rot markiert. Der Artikel kann an dieser Station nicht mehr bestellt werden.
Testperson	Oliver Werlen
Datum	31.05.2021
Durchführung	<ol style="list-style-type: none"> 1. Die Testperson wählt die Station aus.
erhaltenes Resultat	<ul style="list-style-type: none"> • Das vorhin bestellte Produkt ist nicht mehr verfügbar und wird entsprechend markiert.
Test bestanden	Ja

Tabelle 31: Testprotokoll Test 20, Quelle: Autor

Test Nr.	21
Beschreibung	Durch diesen Test wird die Aktualisierung des Warenbestandes bei einem Abbruch des Bezahlvorgangs getestet.
Randbedingungen	<ul style="list-style-type: none"> • Es wird ein Produkt gewählt, welches verfügbar ist. • Der Testfall wird lokal ausgeführt. Das Backend läuft in der IDE • Vom gewählten Artikel sind 10 Stück verfügbar.
erwartete Resultate	<ul style="list-style-type: none"> • Nach dem Klick auf den Checkout-Button wird die Quantity in der Datenbank aktualisiert. • Nach dem Abbrechen des Bezahlvorgangs befindet sich dieser Wert wieder beim Startwert.
Testperson	Oliver Werlen
Datum	31.05.2021
Durchführung	<ol style="list-style-type: none"> 1. Die Testperson bestellt 5 Stück einen Artikel und klickt auf Checkout. 2. In der Datenbank wird der entsprechende Eintrag gesucht. Er wurde aktualisiert. 3. Die Bezahlung wird abgebrochen. 4. In der Datenbank wird der entsprechende Eintrag gesucht. Er befindet sich wieder beim Startwert.
erhaltenes Resultat	<ul style="list-style-type: none"> • Das erwartete Resultat ist eingetreten.
Test bestanden	Ja

Tabelle 32: Testprotokoll Test 21, Quelle: Autor

A.2.4 Initialisierung neue Station

Test Nr.	22
Beschreibung	Durch diesen Test wird die Initialisierung einer neuen Pick-Up Station von einem Administrator getestet.
Randbedingungen	<ul style="list-style-type: none"> • Es ist noch keine Station initialisiert. • Die Station ist mit dem Zerotier-Netzwerk verbunden.
erwartete Resultate	<ul style="list-style-type: none"> • Die neue Station wird angezeigt. • Die erforderlichen Parameter der Station können gesetzt werden. • Die Station wird als initialisiert markiert.
Testperson	Oliver Werlen
Datum	31.05.2021
Durchführung	<ol style="list-style-type: none"> 1. Der Administrator startet das entsprechende Script auf dem Raspberry Pi. 2. Der Administrator loggt sich in der WebApp ein. 3. Der Administrator wechselt zum Administrator-Tab. 4. Der Administrator wählt die Station unter neue Stations aus. 5. Der Administrator gibt die gewünschten Werte ein und klickt auf Initialisieren.
erhaltenes Resultat	<ul style="list-style-type: none"> • Das erwartete Resultat ist eingetreten.
Test bestanden	Ja

Tabelle 33: Testprotokoll Test 22, Quelle: Autor

A.2.5 Abholung einer Bestellung

Test Nr.	23
Beschreibung	Durch diesen Test wird die Abholung einer Bestellung überprüft.
Randbedingungen	<ul style="list-style-type: none"> • Es ist eine Station verfügbar. • Der Nutzer hat bereits eine Bestellung an dieser Station platziert. • Die Bestellung ist bezahlt.
erwartete Resultate	<ul style="list-style-type: none"> • Es wird der Bestellung bereit zur Entnahme Bildschirm angezeigt.
Testperson	Oliver Werlen
Datum	31.05.2021
Durchführung	<ol style="list-style-type: none"> 1. Der Benutzer wechselt zu seinem Profil. 2. Der Benutzer sieht die offene Bestellung. 3. Der Benutzer klickt auf PickUp 4. Der Benutzer scannt mit der Kamera den QR-Code der Station ein. 5. Der entsprechende Bildschirm wird angezeigt.
erhaltenes Resultat	<ul style="list-style-type: none"> • Das erwartete Resultat ist eingetreten.
Test bestanden	Ja

Tabelle 34: Testprotokoll Test 23, Quelle: Autor

B Projektmanagementplan

B.1 Projektorganisation

B.1.1 Organisationsplan, Rollen, Zuständigkeiten

In nachfolgendem Diagramm sind alle Projektbeteiligten aufgeführt. Die Projektmitglieder von der Hochschule Luzern Technik und Architektur unterstehen dabei in diesem Projekt keiner hier genannten Person. Sie haben aus ihrem Projekt entsprechend eigene Projektorganisationen.

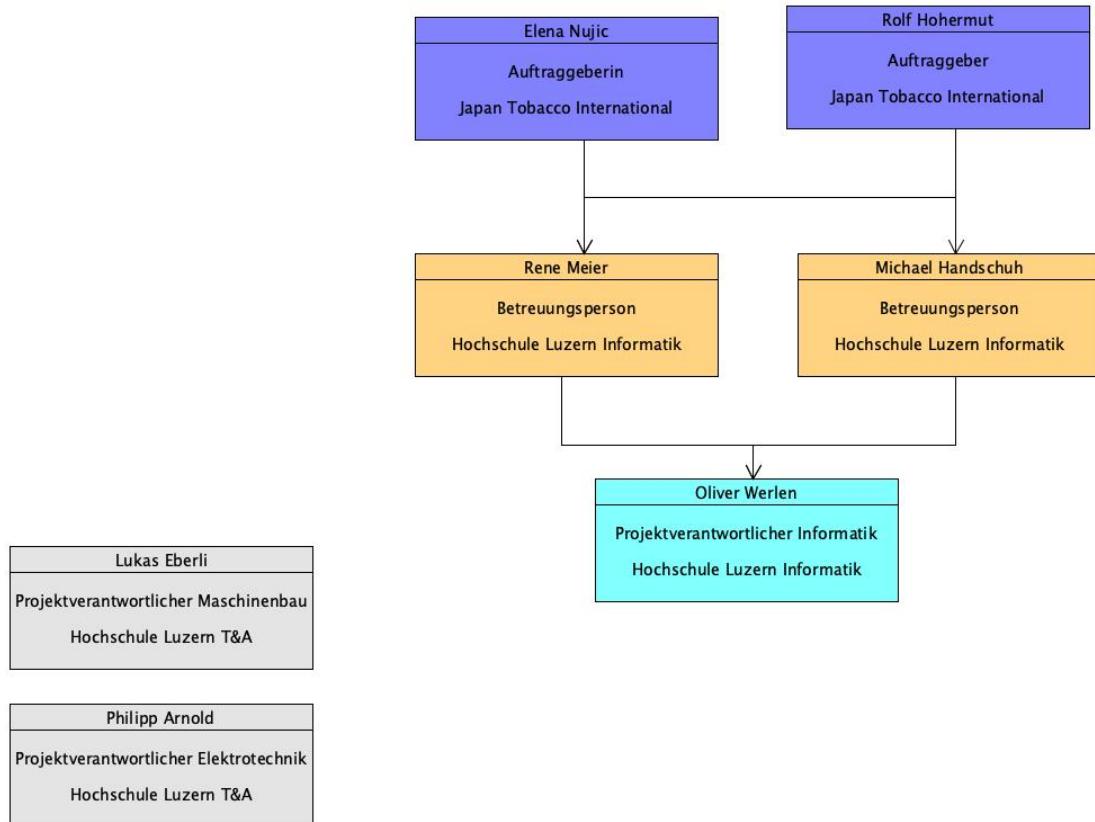


Abbildung 62: Organigramm, Quelle: Autor

Rollen Im Projekt wird nach dem hybriden Projektmanagementvorgehen SoDa gearbeitet. Es werden die hier genutzten Rollen beibehalten.

- Projektleiter/in
- Product Owner
- Scrum Master
- Scrum Team

[HSLU, o.D. b] Da es jedoch in diesem Projekt nur einen aktiven Projektmitarbeiter gibt, werden alle Rollen von Oliver Werlen übernommen.

B.1.2 Projektstrukturplan

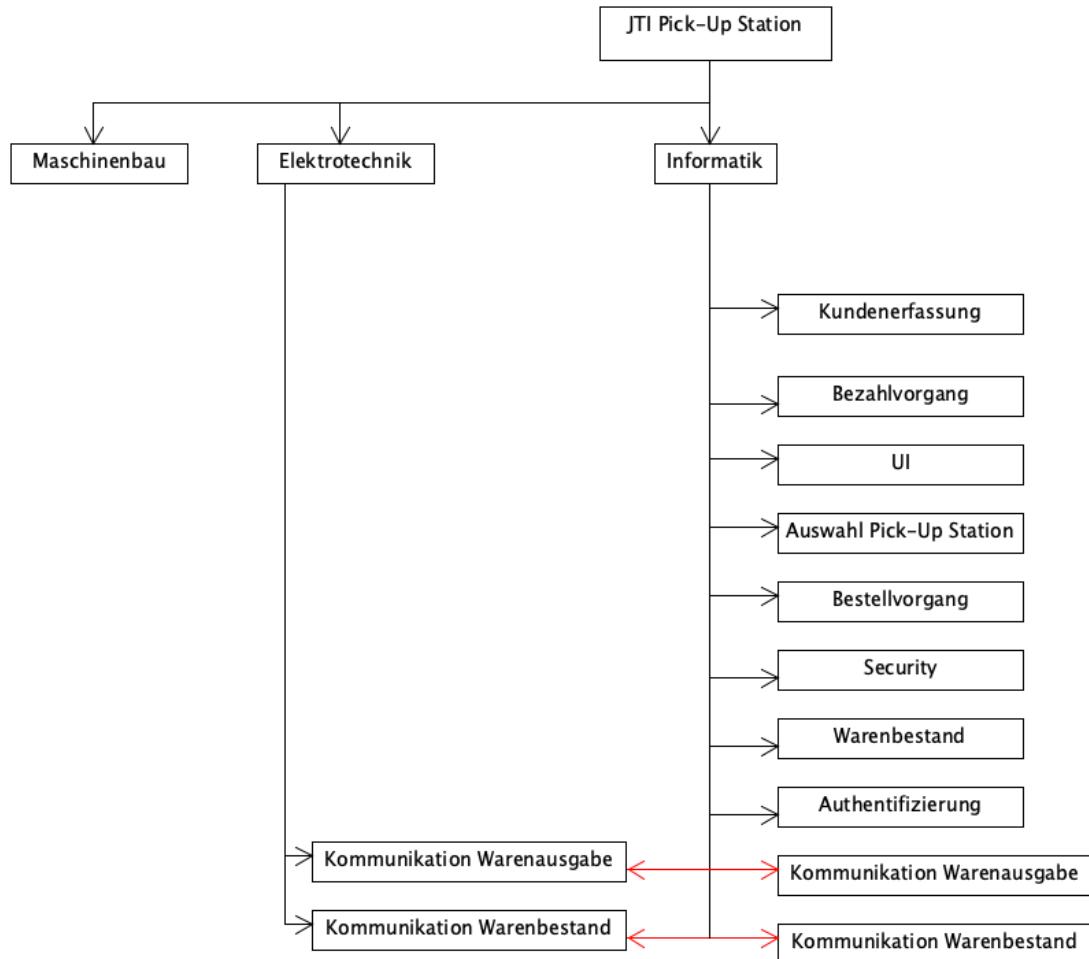


Abbildung 63: Projektstrukturplan,
Quelle: Autor

Beschreibung Im obigen Projektstrukturplan in Abbildung 63 werden die wichtigsten Teilbereiche der Applikation aufgelistet. Dabei wird der Fokus auf den Informatikteil gelegt. Es werden einzig die Schnittstellen zur Elektrotechnik berücksichtigt. Diese wurden rot eingezzeichnet. Die Teilbereiche beziehen sich dabei hauptsächlich auf die in D erarbeiteten Anforderungen.

B.2 Projektführung

B.2.1 Rahmenplan

Im untenstehenden Rahmenplan wird mittels Zeitstrahl eine Grobplanung dargestellt.

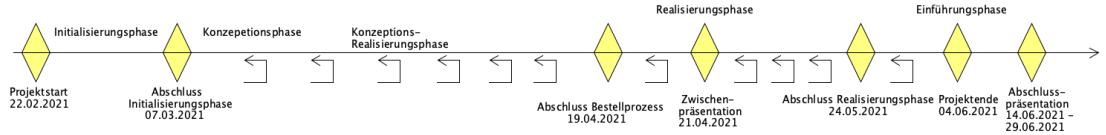


Abbildung 64: Rahmenplan,
Quelle: Autor

Der Rahmenplan wurde zu Beginn des Projekts grob dargestellt. Im Verlauf des Projekts kann dieser bei Bedarf angepasst werden. Die einzelnen Versionen des Rahmenplans sind im Anhang F zu finden.

B.2.2 Meilensteine

Wie in Abbildung 64 zu sehen gibt es insgesamt sieben Meilensteine. Diese werden in folgender Tabelle beschrieben sowie die nötigen Deliverables aufgezeigt.

Meilenstein	Beschreibung	Deliverables
Projektstart	Das Kick-Off Meeting mit allen Projektteilnehmern wurde durchgeführt und das Projekt freigegeben.	Finale Aufgabenstellung
Abschluss Initialisierungsphase	In der Initialisierungsphase wurden alle zum erfolgreichen Start benötigten Unterlagen erstellt. Die Anforderungen wurden von allen Projektmitgliedern akzeptiert.	Projektmanagementplan, Systemspezifikation, Anforderungsliste
Abschluss Bestellprozess	Der Bestellprozess „Artikelauswahl, Artikel in Warenkorb, Artikel Bezahlen“ sowie die Kundenregistrierung sind umgesetzt und getestet.	Testprotokolle zu Abschluss Bestellprozess, Demo Bestellprozess, Release Bestellprozess Release 1
Zwischenpräsentation	Die Zwischenpräsentation ist durchgeführt worden.	Zwischenpräsentation im Anhang, Sitzungsprotokoll
Abschluss Realisierungsphase	Die noch fehlenden Anforderungen aus dem vorherigen Meilenstein sind hier abzuliefern. Es handelt sich dabei um die Auswahl sowie die Abholung an einer Pick-Up Station. Zudem ist die Abfrage des Warenbestandes Teil dieses Meilensteins.	Testprotokolle zu Abholung, Testprotokolle Auswahl, Integration alte Daten, Demo verschiedene Features Release 2
Start Einführung	Der Auftraggeber erhält eine Einführung in die Software.	Sitzungsprotokoll zum Ende der Einführungsphase
Projektende	Der Auftraggeber erhält eine Einführung in die Software	Fertige Projektdokumentation, Abgeschlossene Testprotokolle Release 3
Abschlusspräsentation	Die Abschlusspräsentation ist durchgeführt worden.	-

Tabelle 35: Meilensteine, Quelle: Autor

B.2.3 Risikomanagement

Beim Risikomanagement werden die wichtigsten Risiken für das Projekt ermittelt und passende Gegenmassnahmen ausgearbeitet.

Risiko	Eintrittswahrsch.	Schaden
Falsche Zeiteinschätzung	70	80
Requirements nehmen zu / Requirements ändern sich	60	60
Entwicklerausfall	40	70
Unklare Schnittstellenspez.	40	70
Vernachlässigung Designprozess	20	60
Fehlende technische Kompetenz	20	90
Veränderung im Projektteam	10	70

Tabelle 36: Risikoanalyse, Quelle: Autor

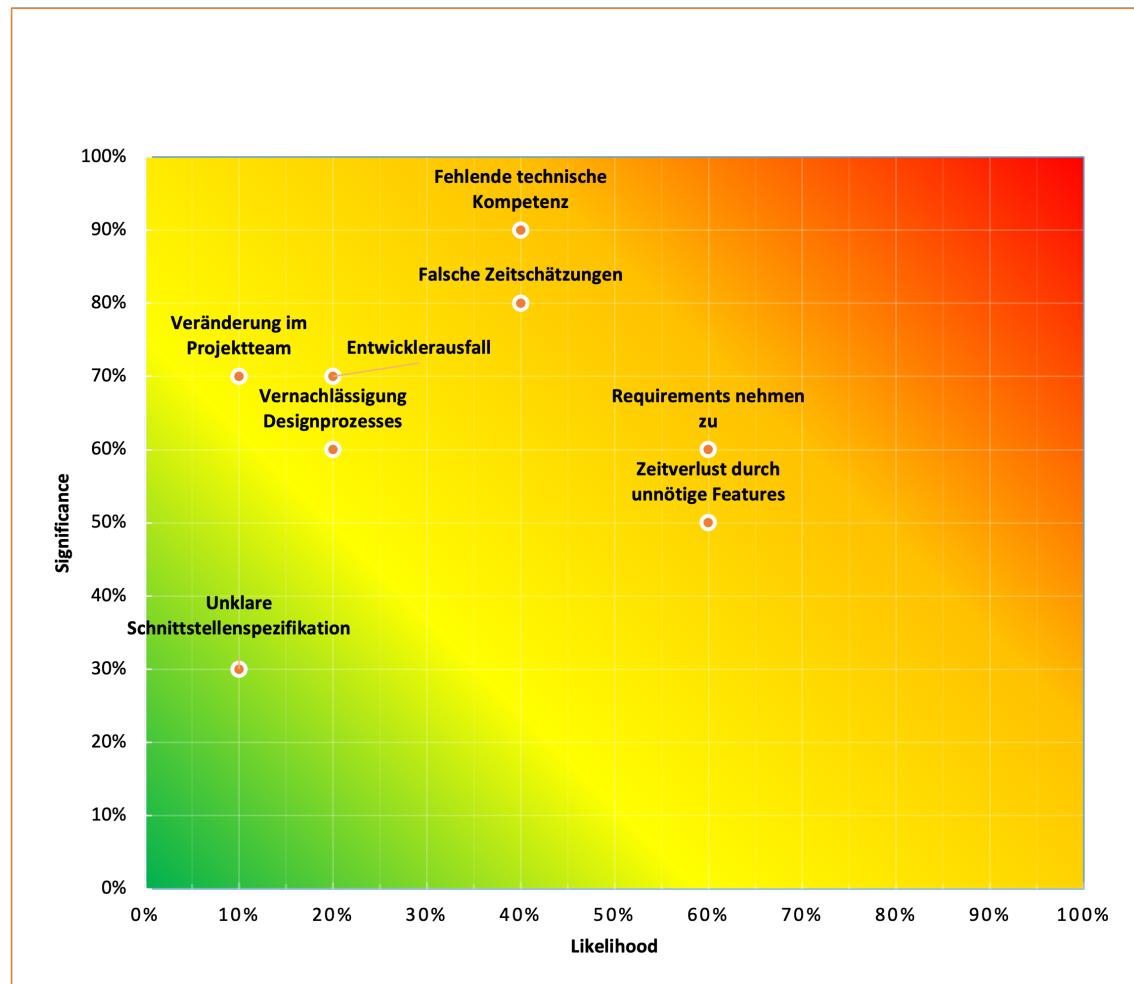


Abbildung 65: Risikomatrix,
Quelle: Autor

Beschreibung Basierend auf der Risikomatrix in Abbildung 65 müssen für die Risiken im rechten oberen Viertel Gegenmassnahmen erarbeitet werden. In diesem Viertel liegt allerdings nur das Risiko "Requirements nehmen zu". Aus diesem Grund werden hier noch weitere Risiken bearbeitet.

- Requirements nehmen zu
- Zeitverlust durch unnötige Features
- Falsche Zeiteinschätzung
- Fehlende technische Kompetenz

Gegenmassnahmen

Requirements nehmen zu Um eine Veränderung der Requirements während des Projekt zu vermeiden, werden die Requirements in ständigem Kontakt mit den Auftraggebern erarbeitet und von diesen abgenommen.

Zeitverlust durch unnötige Features Um dies zu verhindern werden die entsprechenden User Stories definiert. Es werden dabei nur die Requirements berücksichtigt, welche beim Requirements Engineering erarbeitet und vom Auftraggeber abgenommen wurden.

Falsche Zeiteinschätzung Um eine bessere Zeiteinschätzung zu erlangen, wird auf das Wissen aus vorherigen Projekten zurückgegriffen. Basierend darauf kann die Planung genauer durchgeführt werden.

Fehlende technische Kompetenz Es werden Technologien verwendet, welche bereits bekannt sind. Zudem finden diese in vielen Projekten Anwendung, sodass auf das Wissen von erfahrenen Entwicklern zurückgegriffen werden kann.

Risiko	Eintrittswahrsch.	Schaden
Requirements nehmen zu / Requirements ändern sich	20	10
Zeitverlust unnötige Features	30	40
Falsche Zeiteinschätzung	30	80
Fehlende technische Kompetenz	20	20

Tabelle 37: Risikoanalyse nach Massnahmen, Quelle: Autor

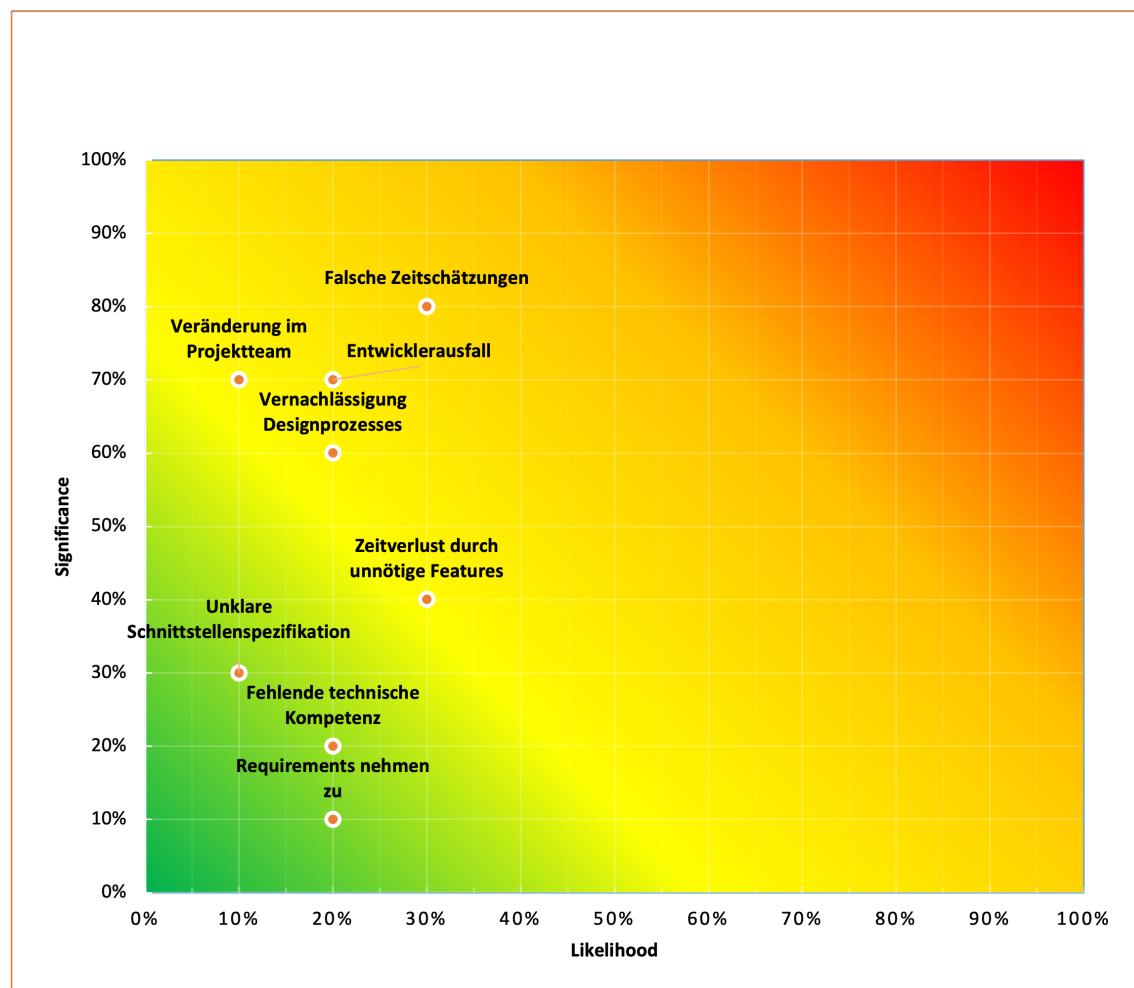


Abbildung 66: Risikomatrix nach Massnahmen,
Quelle: Autor

B.2.4 Definition of done

In jedem Sprint müssen die nachfolgenden Punkte zwingend erreicht werden, um ein potenziell auslieferbares Produkt zu erhalten.

- Review durchgeführt
- Akzeptanzkriterien erfüllt
- keine kritischen Bugs
- Clean Code Guidelines eingehalten
- Dokumentation aktuell

B.3 Projektunterstützung

B.3.1 Tools für Entwicklung, Test und Abnahme

Entwicklungstools Bei der Entwicklung des Projekts kommen folgende Programme zum Einsatz.

Typ	Tool	Version
IDE	InteliJ Ultimate	2020.1
IDE	Visual Studio Code	1.53.2
Versionsverwaltung	Git	2.27.0
Datenbank	Git	10.5.10
Frameworks	Node.js	14.13.0
	Spring Boot	12.4.3
	Angular	11
Sprachen	TypeScript	4.3
	Java	11

Tabelle 38: Entwicklungstools, Quelle: Autor

B.3.2 Konfigurationsmanagement

Konfigurationseinheit Bei diesem Projekt besteht eine Konfigurationseinheit aus mehreren Teilen. Dabei werden diese bei jedem Release aufgeführt. Zusätzlich dazu kommen noch die Reports der automatisierten Tests, falls vorhanden auch der Systemtests.

- Spring Applikation
- Webapplikation

Typ	Version
Spring Applikation	1.0.0 x
Webapplikation	1.0.0

Tabelle 39: Konfigurationseinheit Release 1, Quelle: Autor

Release 1

Testprotokolle Die gesamten Testprotokolle sind im Anhang A.1 zu finden.

Typ	Version
Spring Applikation	1.4.6
Webapplikation	1.1.3

Tabelle 40: Konfigurationseinheit Release 2, Quelle: Autor

Release 2

Testprotokolle Die gesamten Testprotokolle sind im Anhang A.2 zu finden.

Typ	Version
Spring Applikation	1.4.7
Webapplikation	1.2.3

Tabelle 41: Konfigurationseinheit Release 3, Quelle: Autor

Release 3

Testprotokolle Die gesamten Testprotokolle sind im Anhang A.2 zu finden.

B.4 Teststrategie und Drehbuch

B.4.1 Teststrategie

Es handelt sich hier um die Entwicklung eines Prototypen. Die Priorität wird entsprechend gesetzt.

- Funktionalität vor Design
- Sicherheit vor Tempo
- Funktionalität vor Testing

Aus diesem Grund wurden die Unit- und Integrationstests nur konzeptuell umgesetzt. Bei Bedarf können diese erweitert werden.

Automated Testing der REST-Schnittstelle Zum Testen der REST-Schnittstelle wird dabei in erster Linie Postman genutzt.

B.4.2 Testdrehbuch

Wie oben genannt wird auf manuelles Testing gesetzt. Die Tests gehen mit den gleichnamigen Meilensteinen einher. Nachfolgend werden diese inklusive den erhaltenen Resultate beschrieben.

Die Testdrehbücher sind dabei direkt in die Testprotokolle A integriert. An dieser Stelle wird auf ein erneutes Beschreiben verzichtet.

B.5 Bemerkungen

Zur Erstellung des Projektmanagementplans wurde die Vorlage der Hochschule Luzern verwendet.
[HSLU, o.D. a]

C System-Spezifikation

C.1 Systemübersicht

C.1.1 Systemarchitektur

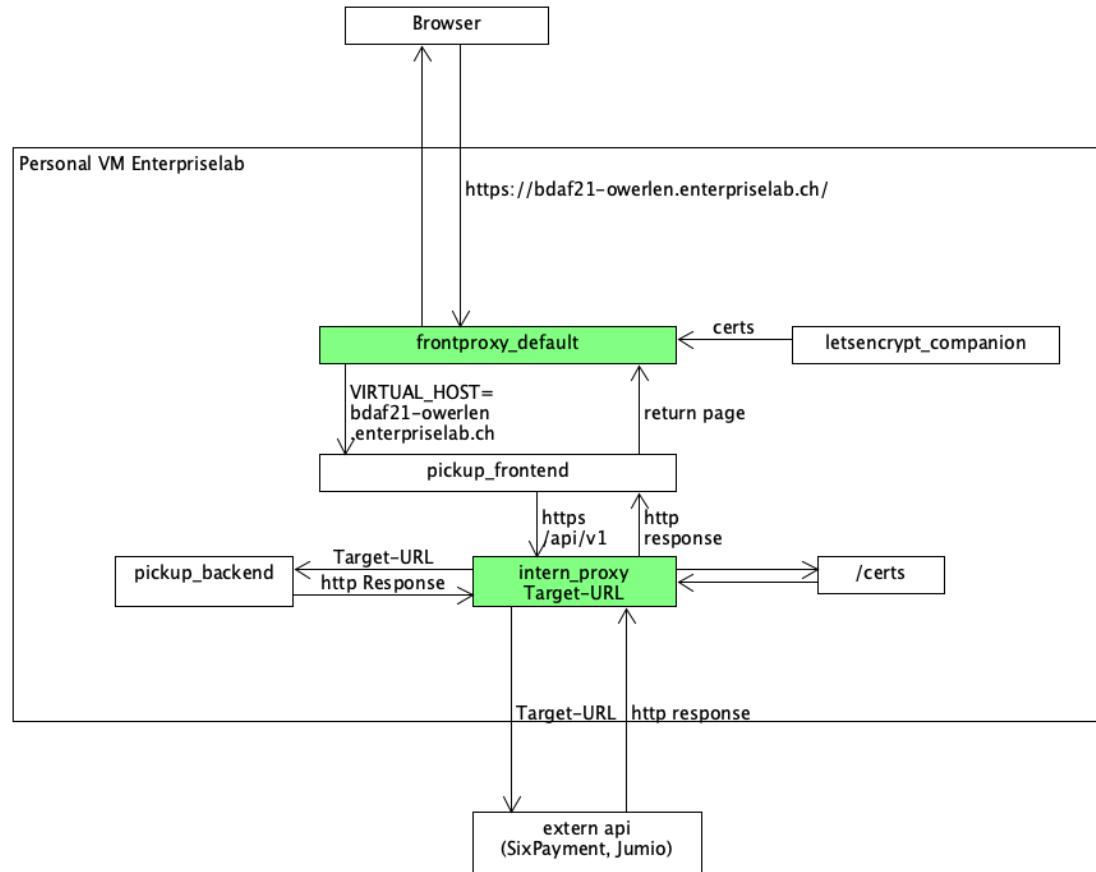


Abbildung 67: Systemarchitektur,
Quelle: Autor

C.1.2 Kontextdiagramm

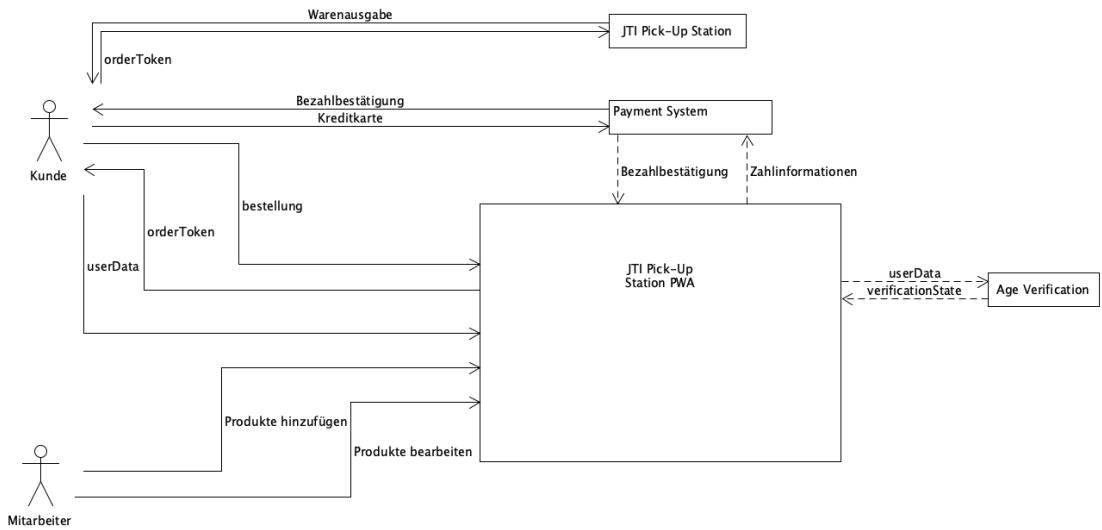


Abbildung 68: Kontextdiagramm,
Quelle: Autor

C.2 Architektur und Designentscheide

C.2.1 Modelle und Sichten

In diesem Projekt wird zwischen zwei verschiedenen Sichten unterschieden:

- **Kunde** Es handelt sich dabei um die Person, welche in der PWA Produkte bestellt und diese abholt.
- **Administrator** Dem Administrator ist es möglich, Produkt hinzuzufügen, zu verändern oder auch zu löschen.
- **Programmierer:** Dieser konzipiert und realisiert die Applikation gemäss den Anforderungen des Auftraggebers.

C.2.2 Daten (Mengengerüst und Strukturen)

Datenbankschema Das Datenbankschema wurde von IntelliJ generiert.

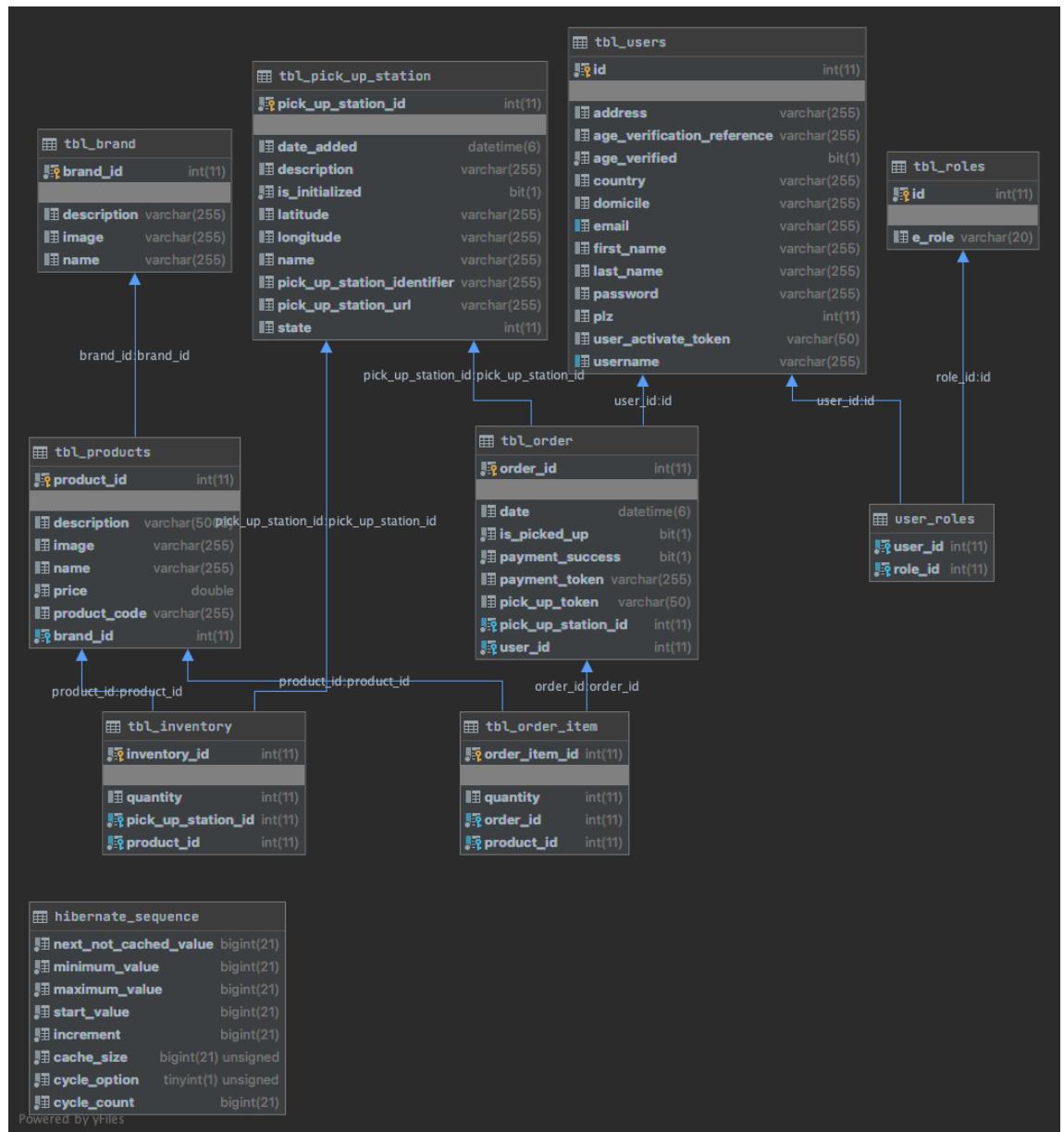


Abbildung 69: Datenbankschema, Quelle: Autor

C.2.3 Entwurfsentscheide

Frontend

Technologien

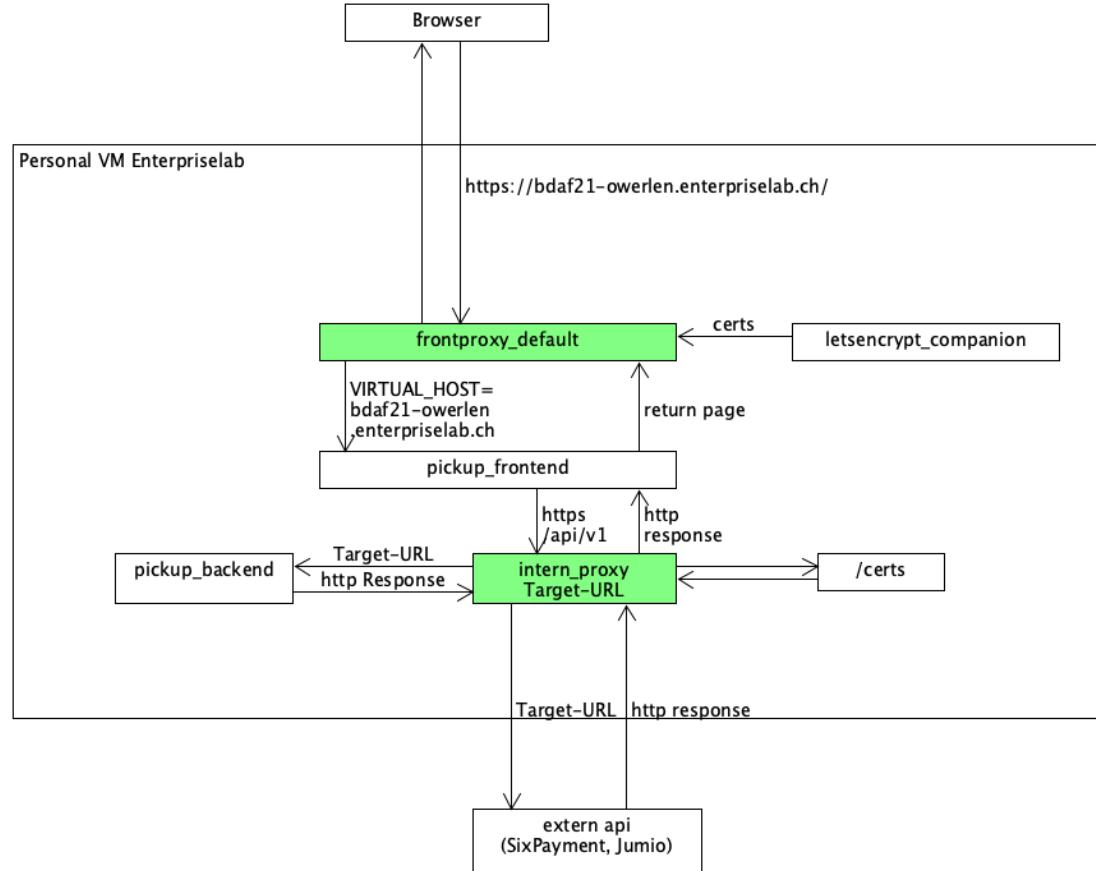


Abbildung 70: Aufbau Produktives System, Quelle: Autor

Projektstruktur

Backend

Spring Boot Für die Backendentwicklung wurde Spring Boot in der Version 2.3.4 genutzt.

Data Transfer Object Es handelt sich hier um ein Enterprise Application Architecture Pattern von Martin Fowler. Genauer handelt es sich um ein Distribution Pattern. Durch den Einsatz dieses Patterns können mehr Daten mit einem Aufruf übertragen werden. Ein weiterer Vorteil ist die klare Trennung zwischen serialisierendem Objekt und dem Domain Model. In der Regel wird ein Assembler genutzt, um das DTO auf das Domain Model zu mappen. [Fowler, 2002]

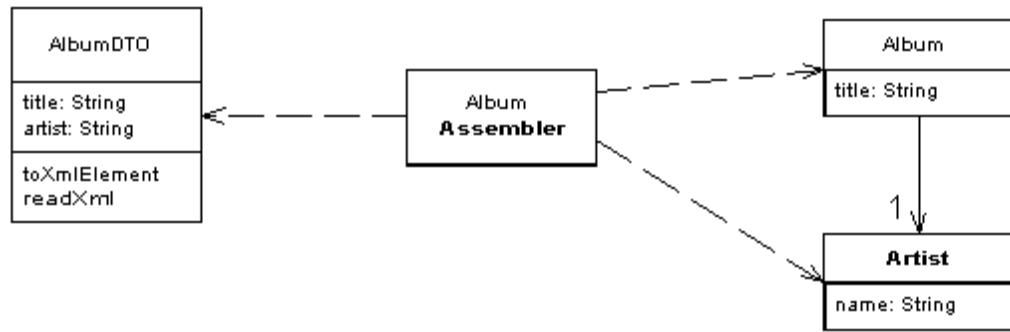


Abbildung 71: DTO Klassendiagramm von Martin Fowler,
Quelle: Fowler, 2002

HATEOAS In diesem Projekt stand der Einsatz des DTO-Pattern jedoch im Konflikt mit der REST-Abstufung von Leonard Richardson und dem von ihm entworfenen Richardson Maturity Model. Gemäss diesem sollen Daten von einem anderen Domain Model als Hyperlink zurückgegeben werden, um das höchste Level zu erreichen. Somit liefert jeder Aufruf nur das angeforderte Modell zurück. Das Data Transfer Object Pattern dient somit nur zur Abgrenzung zwischen dem zu sendenden Objekt und dem Domain Model.

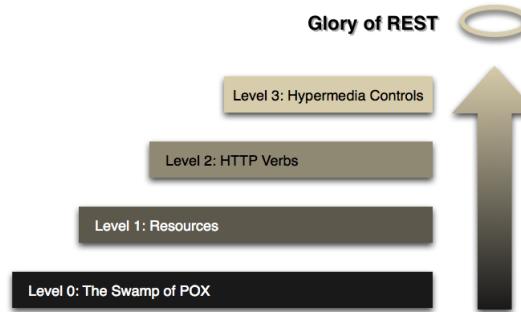


Abbildung 72: Richardson Maturity Model,
Quelle: Fowler, 2010

Dabei wurde auf REST-Level 3 hingearbeitet und entsprechend mit HATEOAS gearbeitet. Durch dies kann das Backend weiter vom Frontend getrennt werden, da bei einer Anpassung der URL auf dem Server keinen Einfluss auf die Funktionalität des Clients hat. Zudem kann die Verständlichkeit der API verbessert und ein Erweitern vereinfacht werden. [Fowler, 2010]

Datenbank Die Datenbank wird durch die Spring Data JPA aus den Entities in diesem Projekt erstellt.

Konfigurationen

Frontend

Backend

C.3 Schnittstellen

C.3.1 Externe Schnittstellen

REST API

API-Dokumentation Die Spring Doc ermöglicht eine sehr einfache und schnelle Dokumentation von REST-API basierend auf der OpenAPI Spezifikation. Um die Dokumentation anzuzeigen, wird das Open Source Tool Swagger UI genutzt. Hiermit lässt sich eine dynaOnline API-Dokumentation als HTML-Page erstellen.

The screenshot shows the Swagger UI interface for a REST API. At the top, it says "OpenAPI definition v0 OAS3". Below that is a "Servers" dropdown set to "http://localhost:8080 - Generated server url". The main area is titled "product-controller". It shows a single endpoint:

```
GET /api/v1/products/{id}
```

Under "Parameters", there is one parameter named "id" which is required and of type integer. There is a "Try it out" button next to the parameters section.

Under "Responses", there is a 200 OK response. The "Description" column shows "OK". In the "Links" column, it says "No links". Below the description, there is a "Media type" dropdown set to "application/json" and a note "Controls Accept header.". Underneath, there are "Example Value" and "Schema" buttons. The "Schema" button is expanded, showing the JSON schema for "ProductDTO":

```
ProductDTO {
    product_id*   integer($int32)
    name*         string
    description* string
    price*        number($double)
    image*        string
    links          > [...]
}
```

Abbildung 73: Ausschnitt aus der Swagger Dokumentation,
Quelle: Autor

C.3.2 Benutzerschnittstellen

Auf ein Beschreiben der Benutzerschnittstelle wird an dieser Stelle verzichtet.

C.4 Environment-Anforderungen

C.4.1 Hardware

Folgende Hardware wurde für diese Applikation verwendet und kann als ausreichend betrachtet werden:

- CPU: Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz
- RAM: 4GB

C.4.2 Software

Die ganze Applikation läuft auf virtuellen Maschine, auf der folgendes Betriebssystem installiert ist:

- Ubuntu v. 20.04.01
- Docker v. 19.03.8

C.4.3 Software

Browser Die Software wurde auf den folgenden Browsern getestet.

- Safari 12
- Chrome 90

D Software Requirements Specification

D.1 Zweck

Der Auftraggeber will durch die JTI Pickup Station einen neuen Absatzkanal zum Vertrieb seiner Produkte an Endkunden erstellen. Durch das Erstellen einer Softwarelösung soll es möglich sein, Kunden inklusive einer Altersverifikation zu erfassen. Auch ist die Umsetzung einer Kaufabwicklung sowie die Auswahl einer Pick-Up Station Teil dieses Projekts. Die Applikation wird dabei als PWA umgesetzt. Die Software soll dabei mit der physischen Pick-Up Station kompatibel sein. Die Umsetzung von dieser ist Teil von zwei weiteren Bachelorarbeiten an der Hochschule Luzern. "Verweis auf Aufgabenstellung".

D.1.1 Zielgruppe

Zur Zielgruppe dieser Software gehören Kunden und Kundinnen von JTI auf der ganzen Welt. Die Software in Kombination mit den physischen Pick-Up Stations soll international eingesetzt werden.

D.1.2 Produktumfang

Der Umfang der Software beginnt bei der Registrierung der Nutzer. Hierbei wird eine bereits vorhandene Alterverifikation eingesetzt, um dies gesetzeskonform umsetzen zu können. Im Onlineshop werden die verfügbaren Produkte von JTI gelistet. Der Nutzer kann diese auswählen, es werden ihm alle Pick-Up Stations, in denen das Produkt verfügbar ist, angezeigt. Der Käufer kann die von ihm gewünschte Station auswählen. Die Bezahlung wird mittels Kreditkarte durchgeführt. Bei JTI liegt ein Vertrag zur Nutzung eines entsprechenden Anbieters vor. Die Software wurde noch in keinen Projekten eingesetzt. In dieser Software wird darauf zurückgegriffen. Nach erfolgreicher Bezahlung wird ein Code auf dem Gerät des Nutzers gespeichert. Mit diesem kann an der gewünschten Pick-Up Station das bestellte Produkt abgeholt werden.

D.1.3 Definitionen

Risiken Das Risikomanagement wird im Projektmanagementplan in Kapitel B.2.3 detailliert aufgeführt.

D.1.4 Systemübersicht

Die Systemübersicht ist in der Systemspezifikation im Kapitel C zu finden.

D.1.5 Abhängigkeiten

Die Erfüllung der Requirements hängt von diversen Faktoren ab. Wesentlich dabei ist die Abhängigkeit von den Bachelorarbeiten der Studierenden an der Hochschule Luzern Technik und Architektur. Die hier vorhandenen Abhängigkeiten werden während der Realisierung möglichst minimiert.

D.2 Spezifische Anforderungen

D.2.1 Funktionale Anforderungen

F.1	Das System muss die Punkte in der von Google aufgestellten Core Progressive Web App checklist erfüllen. Sam Richard, 2020	Muss
F.2	Das System ist auf eine physische Pick-Up Station abgestimmt.	Muss
F.3	Das System bietet dem Anwender die Möglichkeit, sich zu registrieren und sich anschliessend einzuloggen.	Muss
F.4	Das System bietet die Möglichkeit, durch die Anbindung an eine 3rd Party, eine Altersverifikation durchzuführen.	Muss
F.5	Das System bietet die Möglichkeit, verschiedene Produkte anzuzeigen.	Muss
F.6	Das System bietet die Möglichkeit, verschiedene Produkte dem Warenkorb hinzuzufügen.	Muss
F.7	Das System bietet die Möglichkeit, eine Bestellung dauerhaft zu speichern.	Muss
F.8	Das System bietet dem Kunden die Möglichkeit, verschiedene Produkte zu bestellen.	Kann
F.9	Das System ermöglicht die Anbindung an einen bereits bekannten Bezahlungsdienst, um eine sichere Bezahlung zu garantieren.	Muss
F.10	Das System bietet dem Kunden die Möglichkeit, die für ihn nächstgelegene Station auswählen zu können.	Muss
F.11	Das System bietet dem Kunden die Möglichkeit, alle vorhandenen Pick-Up Stations anzuzeigen.	Muss
F.12	Das System bietet dem Kunden die Möglichkeit, seine beliebtesten Produkte direkt zu bestellen.	Kann
F.13	Das System bietet dem Dienstleister die Möglichkeit, einen aktuellen Warenbestand zu erhalten.	Kann
F.14	Das System bietet dem Dienstleister die Möglichkeit, bei zu geringem Warenbestand eine Benachrichtigung zu senden.	Muss
F.15	Das System bietet dem Betreiber die Möglichkeit, Artikel hinzuzufügen und Artikel zu bearbeiten.	Kann
F.16	Das System bietet dem Kunden die Möglichkeit, eine Bestellung durch das Einlesen eines Codes an der Pick-Up Station abzuholen.	Muss

Tabelle 42: Funktionale Anforderungen, Quelle: Autor

D.2.2 Nicht funktionale Anforderungen

ID	Anforderung	Muss/Kann
L.1	Das System soll dem Kunden die Möglichkeit bieten, eine Bestellung mit 5 Klicks zu platzieren.	Kann
L.2	Das System bietet die Möglichkeit international eingesetzt zu werden.	Kann
L.3	Das System muss via HTTPS kommunizieren.	Muss
L.4	Das System muss durch einen modernen und sicheren Authentifizierungsmechanismus geschützt sein.	Muss
L.5	Das System muss die Möglichkeit bieten, durch die Verwendung von bewährten Programmervorgehen von einem externen Fachmann verstanden zu werden.	Muss
L.6	Das System muss über eine CI/CD-Pipeline verfügen.	Muss

Tabelle 43: Nicht Funktionale Anforderungen, Quelle: Autor

D.3 Änderungshistorie

Datum	Änderung	Geändert von	Alte Version
21.03.2021	Aufteilung von Requirements F.6 in F.5, F.6, F.7	Oliver Werlen	Anhang ??
11.05.2021	Hinzufügen des Requirements F.16	Oliver Werlen	

Tabelle 44: Änderungshistorie der Anforderungen, Quelle: Autor

D.4 Bemerkungen

Als Grundstruktur für die SRS wurde eine Vorlage von Perforce genutzt. [Krüger, 2018] Als Basis dazu diente die IEEE Spezifikation 29148-2018. [Doran, 2018]

E Sitzungsprotokolle

Auf den nachfolgenden Seiten sind alle Protokolle von den durchgeführten Sitzungen ersichtlich.

E.1 23.02.2021

Kick Off Meeting

E.1.1 Ordnungsaufruf

Eine Besprechung aller Projektbeteiligten fand Online als Zoom Meeting am 23.02.2021 um 15:00 Uhr statt.

E.1.2 Teilnehmer

Anwesende Mitglieder	Nicht anwesende Mitglieder
René Meier, Betreuungsperson Michael Handschuh, Betreuungsperson Oliver Werlen, Projektleiter Philipp Arnold, Elektrotechnik Lucas Eberli, Maschinenbau Elena Nujic, Auftraggeber Rolf Hohermut, Auftraggeber	

Tabelle 45: Sitzungsprotokoll, Quelle: Autor

E.1.3 Genehmigung des Protokolls

Es handelt sich hierbei um die erste Sitzung in diesem Projekt. Es ist noch kein Protokoll vorhanden.

E.1.4 Ankündigungen

Es handelt sich hierbei um das Kickoff Meeting. Als Ziel wird die Finalisierung der Aufgabenstellung genannt.

E.1.5 Besprochene Punkte

Aufgabenstellung

- Register, Altersüberprüfung -> Bereits vorhanden, Shop (Brands, Produkt), Warenkorb, Bezahlung mit Kreditkarte, beim Kauf definieren, wo abgeholt werden soll, reserviert in PickUp Station, An PickUp -> Mittels QR Code, Bestellung ausgeben
- Zahlung mit Twint, Kreditkarte, keine Nachnahme -> data trans, payrex
- Lösungen bereits in Onlineshop
- Postautomat als Beispiel
- Automat muss wissen, welche Artikel er noch hat
- Preis muss variabel sein, (Gratis Paket)
- Website als Informationsquelle

Sitzungen

- Abhängig von Projektphase, alle 2-3 Wochen
- Zwischenpräsentation von 20 Min, Resultate vorstellen, Fragen
- Schlusspräsentation 20-25 Minuten

Fragen zur Dokumentation

- Benutzen von vorhandenen Texten aus WiPro?
- Auftrag Start WiPro auch in BAA?

E.1.6 Tagesordnung der nächsten Sitzung

- Besprechung Anforderungen
- Rahmenplan, erste Doku-Teile

E.2 04.03.2021

E.2.1 Ordnungsaufruf

Eine Besprechung mit den Auftraggebern fand Online als Microsoft-Teams-Meeting am 04.03.2021 um 15:00 Uhr statt.

E.2.2 Teilnehmer

Anwesende Mitglieder	Nicht anwesende Mitglieder
Oliver Werlen, Projektleiter Philipp Arnold, Elektrotechnik Lucas Eberli, Maschinenbau Elena Nujic, Auftraggeber Rolf Hohermut, Auftraggeber	René Meier, Betreuungsperson Michael Handschuh, Betreuungsperson

Tabelle 46: Sitzungsprotokoll, Quelle: Autor

E.2.3 Genehmigung des Protokolls

Es handelt sich hierbei um die erste offiziell zu protokollierende Sitzung in diesem Projekt. Auf eine Genehmigung des Protokolls zum Kick-Off Meeting E.1 wird daher verzichtet.

E.2.4 Ankündigungen

In diesem Meeting werden offene Punkte besprochen, welche in der Initialisierungsphase aufgetaucht sind.

E.2.5 Besprochene Punkte

Deployment

- Wo soll die Applikation laufen? Enterpriselab für Entwicklung ausreichend, **evtl! (evtl!).** Deployment auf Umgebung von JTI
- Vorgabe vom Auftraggeber?

Vorhandene Anbieter

- Anbieter für Altersverifikation? Winston-Camel Registrieren -> Swisscom, Sunrise
- Neue Lösung mittels -> Jumio
- Anbieter für Bezahlvorgang? Datatrans, Payment von Six Payment
- Kontakt zu Experten von JTI

Requirements

- Durchgehen, finalisieren, erweitern
- F.13 Im Kiosk im Aussenbereich -> Nachfüllanfrage an Kioskbetreiber -> Muss Features
- Ändern und hinzufügen von Produkten

Probleme

- Probleme bei aktuellem Absatzkanal -> Wichtigsten Punkte gefunden, AVEC mit Ihrem Produkt reinnehmen.

Kommunikation der Pick-Up Station

- Internet und Strom vorhanden
- Einlesen des Abholcodes mit RFID

E.2.6 Tagesordnung der nächsten Sitzung

Die nächste Sitzung wird in zwei Wochen angesetzt.

E.3 11.03.2021

E.3.1 Ordnungsaufruf

Eine Besprechung mit der Betreuungsperson fand Online als Zoom-Meeting am 04.03.2021 um 13:00 Uhr statt.

E.3.2 Teilnehmer

Anwesende Mitglieder	Nicht anwesende Mitglieder
Michael Handschuh, Betreuungsperson Oliver Werlen, Projektleiter	René Meier, Betreuungsperson Lucas Eberli, Maschinenbau Philipp Arnold, Elektrotechnik Elena Nujic, Auftraggeber Rolf Hohermut, Auftraggeber

Tabelle 47: Sitzungsprotokoll, Quelle: Autor

E.3.3 Genehmigung des Protokolls

Es handelt sich hierbei um die erste offiziell zu protokollierende Sitzung in diesem Projekt. Auf eine Genehmigung des Protokolls zum Kick-Off Meeting E.1 wird daher verzichtet.

E.3.4 Ankündigungen

In diesem Meeting werden offene Punkte besprochen, welche in der Initialisierungsphase aufgetaucht sind.

E.3.5 Besprochene Punkte

Allgemeine Punkte

- Schnittstelle spezifizieren mit ET und Maschinen evtl. als Meilenstein
- Meilenstein Zwischenpräsentation
- Meilenstein Abschlusspräsentation
- Wieso NFC und nicht QR-Code
- Datenschutz bei jumio ->
- Schnittstelle 3rd Party Systeme in Risikoanalyse
- Datenschutz bei 3rd Party
- Doku auf GitHub Michael Handschuh freigeben

E.3.6 Tagesordnung der nächsten Sitzung

Die nächste Sitzung mit dem Betreuer wird in 2 Wochen stattfinden.

E.4 18.03.2021

E.4.1 Ordnungsaufruf

Eine Besprechung mit den Auftraggebern fand Online als Microsoft-Teams-Meeting am 18.03.2021 um 13:00 Uhr statt.

E.4.2 Teilnehmer

Anwesende Mitglieder	Nicht anwesende Mitglieder
Oliver Werlen, Projektleiter Philipp Arnold, Elektrotechnik Lucas Eberli, Maschinenbau Elena Nujic, Auftraggeber Rolf Hohermut, Auftraggeber	René Meier, Betreuungsperson Michael Handschuh, Betreuungsperson

Tabelle 48: Sitzungsprotokoll, Quelle: Autor

E.4.3 Genehmigung des Protokolls

Das Protokoll der letzten Sitzung wurde von allen Anwesenden bestätigt.

E.4.4 Ankündigungen

In diesem Meeting wird der Abschluss der Initialisierungsphase besprochen. Zusätzlich werden die offenen Punkte zu 5.2.1 besprochen und finalisiert.

E.4.5 Besprochene Punkte

Spezifikation Schnittstelle Abholung

- Problem mit NFC/Bluetooth
- Vorstellen der Alternativen
- Besprechung Alternativen -> Alternative 1 wird bevorzugt, QR-Code auf Maschine
- Wenn keine Bestellung, dann auf Webpage
- Produkte, Liste von Produkten wird gesendet, jtiproducts.ch sind alle Produkte vorhanden
- Keine Policy von JTI vorhanden, jtiproducts als Referenz
- Statusmeldungen an Nachfüller

Schnittstelle Produktbestand

- Finalisierung mit Philipp Arnold individuell
- Übertragung mittels JSON Format

E.4.6 Tagesordnung der nächsten Sitzung

Die nächste Sitzung mit den Auftraggebern findet in zwei Wochen statt.

E.5 25.03.2021

E.5.1 Ordnungsaufruf

Eine Besprechung mit der Betreuungsperson fand Online als Zoom-Meeting am 25.03.2021 um 13:00 Uhr statt.

E.5.2 Teilnehmer

Anwesende Mitglieder	Nicht anwesende Mitglieder
Michael Handschuh, Betreuungsperson Oliver Werlen, Projektleiter	René Meier, Betreuungsperson Lucas Eberli, Maschinenbau Philipp Arnold, Elektrotechnik Elena Nujic, Auftraggeber Rolf Hohermut, Auftraggeber

Tabelle 49: Sitzungsprotokoll, Quelle: Autor

E.5.3 Genehmigung des Protokolls

Es handelt sich hierbei um die erste offiziell zu protokollierende Sitzung in diesem Projekt. Auf eine Genehmigung des Protokolls zum Kick-Off Meeting E.1 wird daher verzichtet.

E.5.4 Ankündigungen

In diesem Meeting werden offene Punkte besprochen, welche in der Initialisierungsphase aufgetaucht sind.

E.5.5 Besprochene Punkte

CI/CD

- Vorzeigen der Pipeline

Besprochene Punkte mit Auftraggebern

- QR-Code anstatt NFC
- Liste mit allen verfügbaren Produkten bereitgestellt

IoT Allgemein

- Webserver auf Pick-Up Station, um Busy Waiting zu vermeiden
- Pick Up meldet auf Webserver -> No-IP
- Genug Power, um sichere Kommunikation zu gewährleisten
- Web-Hock Prinzip
- Experte Stefan Bernet
- sbernet@icloud.com

E.5.6 Tagesordnung der nächsten Sitzung

Die nächste Sitzung mit dem Betreuer wird in 2 Wochen stattfinden.

E.5.7 Unterschriften

E.6 01.04.2021

E.6.1 Ordnungsaufruf

Eine Besprechung mit den Auftraggebern fand Online als Microsoft-Teams-Meeting am 01.01.2021 um 13:30 Uhr statt.

E.6.2 Teilnehmer

Anwesende Mitglieder	Nicht anwesende Mitglieder
Oliver Werlen, Projektleiter Philipp Arnold, Elektrotechnik Lucas Eberli, Maschinenbau Elena Nujic, Auftraggeber Rolf Hohermut, Auftraggeber	René Meier, Betreuungsperson Michael Handschuh, Betreuungsperson

Tabelle 50: Sitzungsprotokoll, Quelle: Autor

E.6.3 Genehmigung des Protokolls

Das Protokoll der letzten Sitzung wurde von allen Anwesenden bestätigt.

E.6.4 Ankündigungen

Es wurden keine Ankündigungen geplant

E.6.5 Besprochene Punkte

Content der Page

- Qualität der Bilder für Desktop kritisch
- Beschreibung der Produkte
- AGBs
- Content von der Seite allgemein -> Vorgegeben vom Auftraggeber

E.6.6 Tagesordnung der nächsten Sitzung

Die nächste Sitzung mit den Auftraggebern findet in zwei Wochen statt.

E.7 15.04.2021

E.7.1 Ordnungsaufruf

Eine Besprechung mit dem Projektbetreuer fand Online als Zoom-Meeting am 15.01.2021 um 13:00 Uhr statt.

E.7.2 Teilnehmer

Anwesende Mitglieder	Nicht anwesende Mitglieder
René Meier, Betreuungsperson Michael Handschuh, Betreuungsperson Oliver Werlen, Projektleiter	Philipp Arnold, Elektrotechnik Lucas Eberli, Maschinenbau Elena Nujic, Auftraggeber Rolf Hohermut, Auftraggeber

Tabelle 51: Sitzungsprotokoll, Quelle: Autor

E.7.3 Genehmigung des Protokolls

Das Protokoll der letzten Sitzung wurde von allen Anwesenden bestätigt.

E.7.4 Ankündigungen

Es wurden keine Ankündigungen geplant.

E.7.5 Besprochene Punkte

Zwischenpräsentation

- Content -> Allgemein vorhanden
- Vorgehen

Fokus auf Funktionalität

- Tests
- Dokumentation

E.7.6 Tagesordnung der nächsten Sitzung

Die nächste Sitzung mit den Auftraggebern findet in zwei Wochen statt.

E.8 15.04.2021

E.8.1 Ordnungsaufruf

Eine Besprechung mit dem Projektbetreuer fand Online als Zoom-Meeting am 15.01.2021 um 13:00 Uhr statt.

E.8.2 Teilnehmer

Anwesende Mitglieder	Nicht anwesende Mitglieder
Oliver Werlen, Projektleiter Philipp Arnold, Elektrotechnik Lucas Eberli, Maschinenbau Elena Nujic, Auftraggeber Rolf Hohermut, Auftraggeber	René Meier, Betreuungsperson Michael Handschuh, Betreuungsperson

Tabelle 52: Sitzungsprotokoll, Quelle: Autor

E.8.3 Genehmigung des Protokolls

Das Protokoll der letzten Sitzung wurde von allen Anwesenden bestätigt.

E.8.4 Ankündigungen

Es wurden keine Ankündigungen geplant.

E.8.5 Besprochene Punkte

Allgemein

- Zuerst Auswahl von Pick Up Station, danach die Produkte
- Bestellung wiederholen
- Produktnummer auf Liste vorhanden

E.8.6 Tagesordnung der nächsten Sitzung

Die nächste Sitzung mit den Auftraggebern findet in zwei Wochen statt.

E.9 21.04.2021

E.9.1 Ordnungsaufruf

Die Zwischenpräsentation fand als Online Meeting am 21.04.2021 um 14:00 Uhr statt.

E.9.2 Teilnehmer

Anwesende Mitglieder	Nicht anwesende Mitglieder
Michael Handschuh, Betreuungsperson Oliver Werlen, Projektleiter Stefan Bernet, Experte	René Meier, Betreuungsperson Philipp Arnold, Elektrotechnik Lucas Eberli, Maschinenbau Elena Nujic, Auftraggeber Rolf Hohermut, Auftraggeber

Tabelle 53: Sitzungsprotokoll, Quelle: Autor

E.9.3 Genehmigung des Protokolls

Das Protokoll der letzten Sitzung wurde von allen Anwesenden bestätigt.

E.9.4 Ankündigungen

Es wurden keine Ankündigungen geplant

E.9.5 Besprochene Punkte

Abhaltung Zwischenpräsentation

- Präsentation inkl. Live Demo
- Proxy musste während Demo noch gestartet werden.

Feedback

- Überblick über Projekt
- Live Demo bei Abschlusspräsentation sehr gut vorbereiten
- sehr interessantes Projekt

E.9.6 Tagesordnung der nächsten Sitzung

Die nächste Sitzung mit dem Experten wird die Abschlusspräsentation sein.

E.10 12.05.2021**E.10.1 Ordnungsaufruf**

Eine Besprechung mit dem Projektbetreuer fand Online als Zoom-Meeting am 12.05.2021 um 13:00 Uhr statt.

E.10.2 Teilnehmer

Anwesende Mitglieder	Nicht anwesende Mitglieder
Michael Handschuh, Betreuungsperson Oliver Werlen, Projektleiter	René Meier, Betreuungsperson Philipp Arnold, Elektrotechnik Elena Nujic, Auftraggeber Rolf Hohermut, Auftraggeber

Tabelle 54: Sitzungsprotokoll, Quelle: Autor

E.10.3 Genehmigung des Protokolls

Das Protokoll der letzten Sitzung wurde von allen Anwesenden bestätigt.

E.10.4 Ankündigungen

Es wurden keine Ankündigungen geplant.

E.10.5 Besprochene Punkte**Planung Abschlusspräsentation**

- Fixierung auf Woche von 14. Juni 2021

Projektstand

- Keine Probleme

Lösung des Kommunikationsproblems

- VPN
- Dokumentation

E.10.6 Tagesordnung der nächsten Sitzung

Die nächste Sitzung mit der Betreuungsperson findet in zwei Wochen statt.

E.11 17.05.2021

E.11.1 Ordnungsaufruf

Eine Besprechung mit dem Auftraggeber fand als Teams-Meeting am 17.05.2021 um 17:30 Uhr statt.

E.11.2 Teilnehmer

Anwesende Mitglieder	Nicht anwesende Mitglieder
Oliver Werlen, Projektleiter Elena Nujic, Auftraggeber Rolf Hohermut, Auftraggeber	René Meier, Betreuungsperson Michael Michael, Betreuungsperson Philipp Arnold, Elektrotechnik

Tabelle 55: Sitzungsprotokoll, Quelle: Autor

E.11.3 Genehmigung des Protokolls

Das Protokoll der letzten Sitzung wurde von allen Anwesenden bestätigt.

E.11.4 Ankündigungen

Es wurden keine Ankündigungen geplant.

E.11.5 Besprochene Punkte

Verbesserungsvorschläge

- Pop up bei ausgeschalteten Location Services auf iPhone
- Felderanordnung bei Registrierung
- Rückkamera bei Scan von QR-Code
- Bestellung erscheint nicht direkt bei offene Bestellungen
- JTI Logo im Header verzerrt

E.11.6 Tagesordnung der nächsten Sitzung

Die nächste Sitzung mit den Auftraggebern findet in zwei Wochen statt.

E.12 25.05.2021**E.12.1 Ordnungsaufruf**

Eine Besprechung mit dem Projektbetreuer fand Online als Zoom-Meeting am 25.05.2021 um 16:00 Uhr statt.

E.12.2 Teilnehmer

Anwesende Mitglieder	Nicht anwesende Mitglieder
Michael Handschuh, Betreuungsperson Oliver Werlen, Projektleiter	René Meier, Betreuungsperson Philipp Arnold, Elektrotechnik Elena Nujic, Auftraggeber Rolf Hohermut, Auftraggeber

Tabelle 56: Sitzungsprotokoll, Quelle: Autor

E.12.3 Genehmigung des Protokolls

Das Protokoll der letzten Sitzung wurde von allen Anwesenden bestätigt

E.12.4 Ankündigungen

Es wurden keine Ankündigungen geplant

E.12.5 Besprochene Punkte**Abgabe**

- Per Mail senden, Feedback bis morgen
- Pitching Video: 90s, wird zu Beginn der Präsentation gezeigt.
- Abgabe pdf-A

E.12.6 Tagesordnung der nächsten Sitzung

Es findet keine nächste Sitzung statt.

E.13 01.06.2021

Es handelt sich hier um das Meeting zum Abschluss des Meilensteins Start Einführung.

E.13.1 Ordnungsauftrag

Eine Besprechung mit dem Projektbetreuer fand Online als Zoom-Meeting am 01.06.2021 um 16:30 Uhr statt.

E.13.2 Teilnehmer

Anwesende Mitglieder	Nicht anwesende Mitglieder
Oliver Werlen, Projektleiter Philipp Arnold, Elektrotechnik Elena Nujic, Auftraggeber Rolf Hohermut, Auftraggeber	René Meier, Betreuungsperson Michael Handschuh, Betreuungsperson

Tabelle 57: Sitzungsprotokoll, Quelle: Autor

E.13.3 Genehmigung des Protokolls

Das Protokoll der letzten Sitzung wurde von allen Anwesenden bestätigt

E.13.4 Ankündigungen

Es wurden keine Ankündigungen geplant

E.13.5 Besprochene Punkte

Abnahme

- Test von diversen Features
- Altersverifikation funktioniert mit Kind, kein Zugriff erlaubt.

E.13.6 Tagesordnung der nächsten Sitzung

Es findet keine weitere Sitzung statt.

F Rahmenpläne

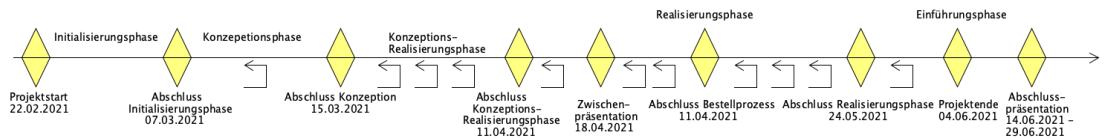


Abbildung 74: Rahmenplan Version 1,
Quelle: Autor

G Originale Aufgabenstellung

Definitive Aufgabenstellung: Wirtschaftsprojekt / Bachelorarbeit

1. Starttermin:

spätmöglichster Starttermin: HS KW 38; FS KW 8

22.02.2021

2. Abgabetermin:

Dauer einer BAA: max. 15/16 Kalenderwochen (Um zur ordentlichen Diplomierung im Sommer zugelassen zu werden, muss die Abgabe bis spätestens Freitag, eine Woche nach Semesterende, erfolgen)

Dauer eines WIPRO: max. 14/15 Kalenderwochen

04.06.2021

3. Studierende:

Student/in 1:

Student/in 2:

Name, Vorname:

Werlen, Oliver

Studiengang:

BSCI

Mobile:

E-Mail:

oliver.werlen@stud.hslu.ch

Projekt mit Arbeitgeber (bb-Studierende)

Ja Nein

Ja Nein

4. Auftraggeber/in (Rechnungsadresse):

Firma:

JT International AG (Japan Tobacco International)

Ansprechperson:

Elena Nujic

Funktion:

Consumer Activation Managerin

Strasse:

Baselstrasse 65

PLZ / Ort:

6252 Dagmersellen

Telefon:

T +41 62 748 03 29

Email:

M +41 79 888 68 61

Website:

Elena.Nujic@jti.com

5. Betreuungsperson:

René Meier und Michael Handschuh

6. Aufgabenstellung

Titel:	Kaufabwicklung für JTI Pick-Up Station
Ausgangslage und Problemstellung:	<p>Der Auftraggeber will einen neuen Absatzkanal entwickeln. Seine Produkte sollen über eine App* bestellt und gekauft und danach an Pick-Up Stationen abgeholt werden. Über die App müssen folgende Funktionen erfüllt werden: Kundenerfassung (Mechanik für Altersüberprüfung vorhanden), Kaufabwicklung und Suche von der näheren Pick-Up Station. Sowohl die App als auch die Pick-Up Stationen sollen als Prototypen an der HSLU entwickelt werden.</p> <p>Diese Arbeit wird als interdisziplinäre (Bachelor) Arbeit durchgeführt: Maschinentechnik-Elektronik-Informatik.</p> <p><i>* Bitte beachten: aufgrund der App-Store Bestimmungen von Apple bezüglich Tabak kann keine herkömmliche App – verfügbar im Store – angeboten werden. Es muss zwingend eine «progressive web app» (PWA) sein welche über den Browser funktioniert.</i></p>
Ziel der Arbeit und erwartete Resultate:	<p>Entwicklung einer progressiven Web-App zur Abwicklung eines Kauf-Abholvorganges an einem Verteilautomat. Die Funktionalität und Zweckmässigkeit dieses für die Firma neuen Absatzkanals soll mit Hilfe eines Prototyps - physische Pick-Up-Station und damit abgestimmte App – demonstriert werden.</p> <p>Im besten Fall findet die Pick-up Station nicht nur in der Schweiz Verwendung, sondern wird von JTI auch in anderen Märkten weltweit eingesetzt.</p> <p>Konkret sollen die folgenden Aspekte bearbeitet werden:</p> <ul style="list-style-type: none"> - Recherche. Es soll eine Recherche zu artverwandten Technologien und Ansätzen durchgeführt werden. - Anforderungen. Die Anforderungen an die Kaufabwicklung sollen erarbeitet und sowohl mit den Projektpartnern (T&A M, T&A E) wie auch mit dem Auftraggeber abgeglichen werden. - Konzept: Es soll ein Lösungskonzept erarbeiten werden, welches die Kundenerfassung, die Kaufabwicklung (Identifikation, Bestellung, Kauf, Pick-Up), und die Pick-Up Suche ermöglicht. - Umsetzung: Das Lösungskonzept soll prototypisch umgesetzt und getestet werden. - Integration: Soweit möglich und sinnvoll soll die Lösung mit den Projektpartnern-Systemen integriert werden.
Gewünschte Methoden, Vorgehen:	<ul style="list-style-type: none"> - Entwicklung und Planung einer Anwendung ausgehend von einem Pflichtenheft. - Programmieren einer Progressiven Web App (PWA) mit dafür geeigneter Sprache (z.B. JavaScript). - Im Kontakt mit dem Auftraggeber müssen die benötigten Funktionseinheiten im Detail spezifiziert werden. - Die Priorisierung der Entwicklungstätigkeiten soll die Funktionalität die Prototypen – wenn auch mit eingeschränkter Funktionalität – bei Abschluss der Arbeit garantieren.
Kreativität, Varianten, Innovation*	Abgesehen von Fixanforderungen – die auf unterschiedliche Art realisiert werden können – bietet die neue Schnittstelle zwischen Anbieter und Kunden viele Möglichkeiten, um Prozessverbesserungen vorzuschlagen und zu implementieren. Dies sowohl im Austausch mit dem Auftraggeber und den Entwicklern der Pick-Up Station sowie bei der Programmierung der App selbst (Kundenerfassung, Produktlogistik, Zahlvorgang).

Schlagwörter:	Progressive Web-App, Mobile Systems, User Experience, Kaufabwicklung, interdisziplinäres Arbeiten
Wirtschaftsprojekt oder Bachelorarbeit:	<input type="checkbox"/> Wirtschaftsprojekt: 180 Stunden pro Studierender <input checked="" type="checkbox"/> Bachelorarbeit: 360 Stunden

* Bitte heben Sie in diesem Punkt hervor, inwiefern Ihre Projektidee **über kreativen Spielraum** verfügt. Dabei sind folgende Kriterien relevant: Die Idee erlaubt den Studierenden eigene Ideen zu entwickeln und Varianten zu erarbeiten, ist ausserhalb vom Tagesgeschäft angesiedelt, beinhaltet Neuland/Innovation und ist nicht durch Produkte & Tools getrieben.

Bitte kreuzen Sie eine Projektart und die zutreffenden Schwerpunkte an.

Projektarten:

- Einsatz von Standardsoftware und Services
- Software- und Produkt-Entwicklung
- Innovationsprojekte (Projekte mit Erkenntnisgewinn, Forschungsprojekte)
- IT-Infrastrukturentwicklung
- Strukturierte Analyse und Konzeption von Systemen und Abläufen

Schwerpunkte:

- Artificial Intelligence & Machine Learning
- Business Process Modelling
- Data Science
- Hardwarenahe Software-Erstellung
- Human Computer Interaction Design
- ICT Business Solutions
- ICT Infrastrukturen
- Internet of Things
- Mobile Systems
- Security/Privacy
- Software-Erstellung
- Visual Computing (Grafik, Bildverarbeitung, Vision, VR, AR)
- _____
- _____

7. Zeiteinteilung

Vorschlag für die Zeiteinteilung pro Person

WIPRO:

pro Woche:	ca. 12h
für Modulendprüfung:	<u>ca. 10h</u>
Total:	180 h

BAA:

pro Woche:	ca. 20h
Schlusswoche:	ca. 50h
Für Modulendprüfung:	<u>ca. 10h</u>
Total:	360 h

8. Rechtliche Grundlagen und Reglemente

Folgende Rechtsgrundlagen und Reglemente sind für die Wirtschaftsprojekte und Bachelorarbeiten an der Hochschule Luzern – Informatik massgebend:

- Studienordnung für die Ausbildung an der Hochschule Luzern, FH Zentralschweiz ([Link](#))
- Studienreglement für die Bachelor-Ausbildung an der Hochschule Luzern - Informatik ([Link](#))

3

9. Bestätigung

Mit der Kenntnisnahme der Aufgabenstellung bestätigen Student/in und Auftraggeber/in, dass

- Sie mit der Aufgabenstellung einverstanden sind.
- die Auftraggeberin/der Auftraggeber damit einverstanden ist, dass die Hochschule Luzern – Informatik für die Organisation einer Bachelorarbeit von ihr/ihm einen Kostenbeitrag von CHF 1'000.00 (inkl. MwSt.) pro Student/in erhebt. Dies gilt nicht für Arbeiten, welche berufsbegleitend

Studierende in Verbindung mit ihrem Arbeitgeber/ihrer Arbeitgeberin machen und für HSLU interne Auftraggeber/innen. Für die Wirtschaftsprojekte wird kein Kostenbeitrag verrechnet.

- Betreuungspersonen und Experten uneingeschränkten Einblick in die Arbeit erhalten. Auch anlässlich von Präsentationen und Marketingaktivitäten kann die Arbeit der Öffentlichkeit gezeigt werden. Eine Zusammenfassung der Arbeit wird in jedem Fall veröffentlicht. Falls das Thema vertraulich behandelt werden soll, muss der Aufgabenstellung eine entsprechende Vertraulichkeitserklärung beiliegen.

Datum: 23.02.2021

Die definitive Aufgabenstellung (pdf-Format) bitte per E-Mail an die Transferstelle senden, zwingend in Kopie an alle involvierten Parteien.

Anlaufstelle für alle Informationen im Zusammenhang mit studentischen Arbeiten sowie für Entgegennahme von Projektideen & Aufgabenstellungen:

Hochschule Luzern - Informatik
Transfer Services
Suurstoffi 1
6343 Rotkreuz
T: 041 228 24 66
E: transfer.informatik@hslu.ch

H Zwischenpräsentation

Zwischenpräsentation JTI PickUp Station

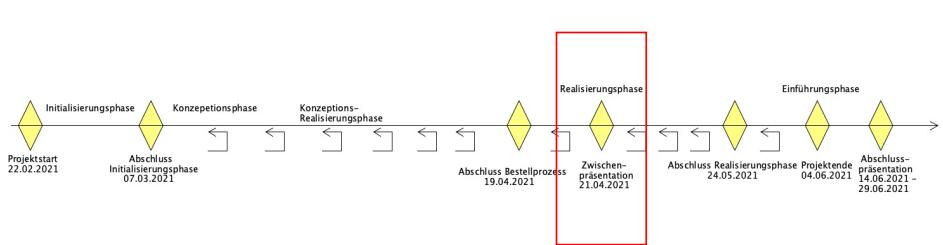
Oliver Werlen

21. April 2021

Inhalt

- Vorgehen
 - Projektstand
 - Technologien
 - Probleme
- Demo
- Technische Details
 - Infrastruktur
 - Transport Layer Security
 - Autorisierung
- Nächste Schritte
 - Kommunikation mit PickUp Stations
 - Progressive Web App
 - Div. Wunsch-Features vom Auftraggeber

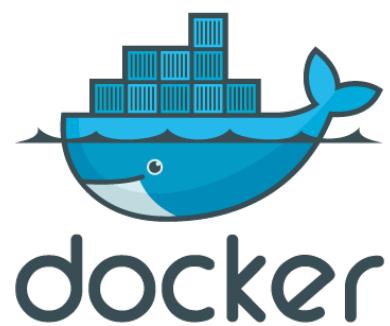
Projektstand



- Software Development agile (SoDa) Zeitstrahl
- Montag: Meilenstein 3 erreicht
 - Grösstenteils erfüllt
 - Anbindung an Jumio zur Alterverifikation
 - Dafür aber bereits Pick Up Funktionalität in Backend

Technologien

- Spring Boot
- Angular
- Docker
- MariaDB
- LaTEX
- Material UI



Probleme

Infrastruktur

- Letzte Woche, EnterpriseLab Maschine kaputt gegangen
- Nicht wirklich nachvollziehbar
- Deployment musste neu aufgebaut werden

CORS

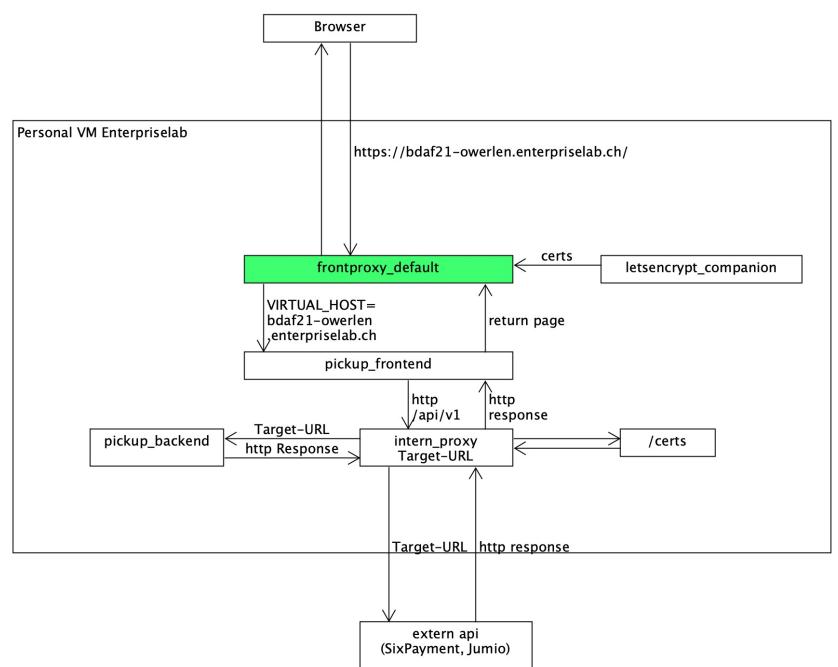
- Proxy benötig, um Anbindung an 3rd Party API zu ermöglichen
 - Six Payment
 - Jumio
- Via Node-Proxy gelöst
- 127.0.0.1 bei Safari nicht Vertrauenswürdig



Demo

Technische Details

- Reverse Proxy für TLS
- Interner Proxy zur Kommunikation mit API's
- Alles lokal, daher keine zusätzliche TLS nötig



Autorisierung

- JSON Web Token
- Rollenbasiert
 - Admin
 - User
 - Maintenance
- Spring Security

Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJp  
YXd1cmxlbiIsImlhdCI6MTYxODQyMzQ3M  
ywiZXhwIjoxNjE4NTA50DczfQ.IwSI_Mq  
7KsvS0scamA1UlQ72W8sVcXmAHzIlMie9  
gsKTt18RXDpc4v4rIV-  
guEF5sSgL4YHYx7Fr0poiLvamSg|
```

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{  
  "alg": "HS512"  
}
```

PAYOUT: DATA

```
{  
  "sub": "iawerlen",  
  "iat": 1618423473,  
  "exp": 1618509873  
}
```

VERIFY SIGNATURE

```
HMACSHA512(  
  base64UrlEncode(header) + "." +  
  base64UrlEncode(payload),  
  your-256-bit-secret  
) □ secret base64 encoded
```

Nächste Schritte

- 
- Kommunikation mit PickUp Station
 - Progressive Web App mit Angular
 - Div. Kleinere Wunsch-Features des Auftraggebers nach Zeit
 - Merkliste
 - Schnelle Bestellung

Quellen

- <https://bgasparotto.com/change-spring-boot-server-port>
- <https://www.trendreport.de/docker-im-hinblick-auf-devops/>
- <https://icon-icons.com/icon/angular-logo/169598>
- <https://www.informatik-aktuell.de/betrieb/datenbanken/mariadb-interview-mit-ceo-patrik-sallner.html>

D Software Requirements Specification

D.1 Zweck

Der Auftraggeber will durch die JTI Pickup Station einen neuen Absatzkanal zum Vertrieb seiner Produkte an Endkunden erstellen. Durch das Erstellen einer Softwarelösung soll es möglich sein, Kunden inklusive einer Altersverifikation zu erfassen. Auch ist die Umsetzung einer Kaufabwicklung sowie die Auswahl einer Pick-Up Station Teil dieses Projekts. Die Applikation wird dabei als PWA umgesetzt. Die Software soll dabei mit der physischen Pick-Up Station kompatibel sein. Die Umsetzung von diesen ist Teil von zwei weiteren Bachelorarbeiten an der Hochschule Luzern. "Verweis auf Aufgabenstellung"

D.1.1 Zielgruppe

Zur Zielgruppe dieser Software gehören Kunden und Kundinnen von Japan Tobacco International (JTI) auf der ganzen Welt. Die Software in Kombination mit den physischen Pick-Up Stations soll international eingesetzt werden.

D.1.2 Produktumfang

Der Umfang der Software beginnt bei der Registrierung der Nutzer. Hierbei wird eine bereits vorhandene Alterverifikation eingesetzt, um dies gesetzeskonform Umsetzen zu können. Im Onlineshop werden die verfügbaren Produkte von JTI gelistet. Der Nutzer kann diese Auswählen, anschliessend werden ihm alle Pick-Up Stations, in denen das Produkt verfügbar ist, angezeigt. Der Käufer kann die von ihm gewünschte Station auswählen. Anschliessend wird die Bezahlung per Kreditkarte durchgeführt. In anderen Projekten wurde dabei von JTI bereits ein bekannter Anbieter genutzt. In dieser Software wird darauf zurückgegriffen. Nach erfolgreicher Bezahlung wird ein Code auf dem Gerät des Nutzers gespeichert. Mit diesem kann an der gewünschten Pick-Up Station das bestellte Produkt abgeholt werden.

D.1.3 Definitionen

Risiken Das Risikomanagement wird im Projektmanagementplan in Kapitel B.2.3 detailliert aufgeführt.

D.1.4 Systemübersicht

Die Systemübersicht ist in der Systemspezifikation im Kapitel C zu finden.

D.1.5 Abhängigkeiten

Die Erfüllung der Requirements hängt von diversen Faktoren ab. Wesentlich dabei ist die Abhängigkeit von den Bachelorarbeiten der Studierenden an der Hochschule Luzern Technik und Architektur. Die hier vorhandenen Abhängigkeiten werden während der Realisierung möglichst minimiert.

D.2 Spezifische Anforderungen

D.2.1 Funktionale Anforderungen

F.1	Das System muss die Punkte in der von Google aufgestellten Core Progressive Web App checklist erfüllen. Sam Richard, 2020	Muss
F.2	Das System ist auf eine physische Pick-Up Station abgestimmt.	Muss
F.4	Das System bietet dem Anwender die Möglichkeit, sich zu registrieren.	Muss
F.5	Das System bietet die Möglichkeit, durch die Anbindung an eine 3rd Party, eine Altersverifikation durchzuführen.	Muss
F.6	Das System bietet dem Kunden die Möglichkeit, verschiedene Produkte zu bestellen.	Kann
F.7	Das System ermöglicht die Anbindung an einen bereits bekannten Bezahlungsdienst, um eine sichere Bezahlung zu garantieren.	Muss
F.8	Das System bietet dem Kunden die Möglichkeit, verschiedene Produkte zu bestellen.	Kann
F.9	Das System bietet dem Kunden die Möglichkeit, die für ihn nächstgelegene Station auswählen zu können.	Muss
F.10	Das System bietet dem Kunden die Möglichkeit, alle vorhandenen Pick-Up Stations anzuzeigen.	Muss
F.11	Das System bietet dem Kunden die Möglichkeit, seine beliebtesten Produkte direkt zu bestellen.	Kann
F.12	Das System bietet dem Dienstleister die Möglichkeit, einen aktuellen Warenbestand zu erhalten.	Kann
F.13	Das System bietet dem Dienstleister die Möglichkeit, bei zu geringem Warenbestand eine Benachrichtigung zu senden	Muss
F.14	Das System bietet dem Betreiber die Möglichkeit, Artikel hinzuzufügen und Artikel zu bearbeiten	Kann

Tabelle 9: Funktionale Anforderungen, Quelle: Autoren

D.2.2 Nicht funktionale Anforderungen

ID	Anforderung	Muss/Kann
L.1	Das System soll dem Kunden die Möglichkeit bieten, eine Bestellung mit 5 Klicks zu platzieren	Kann
L.2	Das System bietet die Möglichkeit, International eingesetzt zu werden.	Kann
L.3	Das System muss via HTTPS kommunizieren	Muss
L.4	Das System muss durch einen modernen und sicheren Authentifizierungsmechanismus geschützt sein	Muss
L.5	Das System bietet die Möglichkeit, durch die Verwendung von bewährten Programmervorgehen von einem externen Fachmann verstanden zu werden	Muss

Tabelle 10: Nicht funktionale Anforderungen, Quelle: Autoren

D.3 Bemerkungen

Als Grundstruktur für die SRS wurde eine Vorlage von Perforce genutzt. [Krüger, 2018] Als Basis dazu diente die IEEE Spezifikation 29148-2018. [Doran, 2018]

D.4 Unterschriften

Mit der Unterschrift gilt die Software Requirement Specification als bestätigt.

Ort, Datum: _____

Visum: _____