Kaufabwicklung für JTI Pick-Up Station

Themenbereiche:	Software-Erstellung, Mobile Systems, Human Computer Interaction Designs
Studierende:	Oliver Werlen
Betreuungsperson:	René Meier und Michael Handschuh
Experte:	Stefan Bernet
Auftraggebende:	JT International AG (Japan Tobacco International)
Keywords:	Progressive Web App, Spring Boot, Angular, Payment Integration, Age Verification Integration

1. Aufgabenstellung

Durch die Progressive Web App «JTI Pick-Up Station» ist es dem Kunden möglich, seine Ware bequem im Onlineshop zu bestellen und anschliessend 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.

2. Lösungskonzept

Das Lösungskonzept wurde in fünf Hauptpunkte aufgeteilt:

Identifikation Bei der Identifikation muss sich ein Nutzer für die Applikation Registrieren und Einloggen können. Um die Applikation benutzen zu können, muss der Nutzer eine Altersverifikation durchführen. Vom Auftraggeber wird der Dienst «Jumio» bereitgestellt. Die Authentifizierung wird mittels JSON Web Token und Spring Security umgesetzt.

Bestellung Die Produkte des Auftraggebers müssen im Shop angezeigt werden und dem Warenkorb hinzugefügt werden. Es dürfen nur verfügbare Produkte hinzugefügt werden.

Kauf Um eine Bestellung abschliessen zu können, ist die erfolgreiche Bezahlung nötig. Diese wird durch Six Payment abgewickelt.

Pick-Up Um eine Bestellung abholen zu können, muss der QR-Code auf der entsprechenden Station mit der PWA eingelesen werden. Anschliessend werden die Produktnummer und die Produktanzahl auf die UART-Schnittstelle geschrieben und weiterverarbeitet.

Suche von Stations Die verfügbaren Pick-Up Stations sollen auf der Karte dargestellt werden. Um eine Bestellung platzieren zu können, muss eine Station ausgewählt sein. Die Produktverfügbarkeit an einer Station wird aufgelistet.

3. Spezielle Herausforderungen

Die grösste Herausforderung in diesem Projekt war die Kommunikation mit den 3rd Party APIs von Jumio und Six Payment. Obschon die APIs sehr gut dokumentiert sind, kam es hier zu Beginn durchgehend zu CORS (Cross-Origin-Ressource-Sharing) Problemen, deren Lösung sehr viel Zeit beanspruchte. Die zuerst umgesetzten Lösungsansätze, wie der Einsatz vom Angular Proxy, die Implementierung eines eigenen Proxies oder die Kommunikation via localhost waren dabei nicht von Erfolg gekrönt. Schlussendlich wurde das Backend als Proxy genutzt. Die resultierende Server-to-Server Kommunikation löste das CORS-Problem und ermöglichte die direkte Verarbeitung der Server Antworten im Backend. Hierzu zählte das Auslesen des Bezahlstatus oder der Altersverifikation.

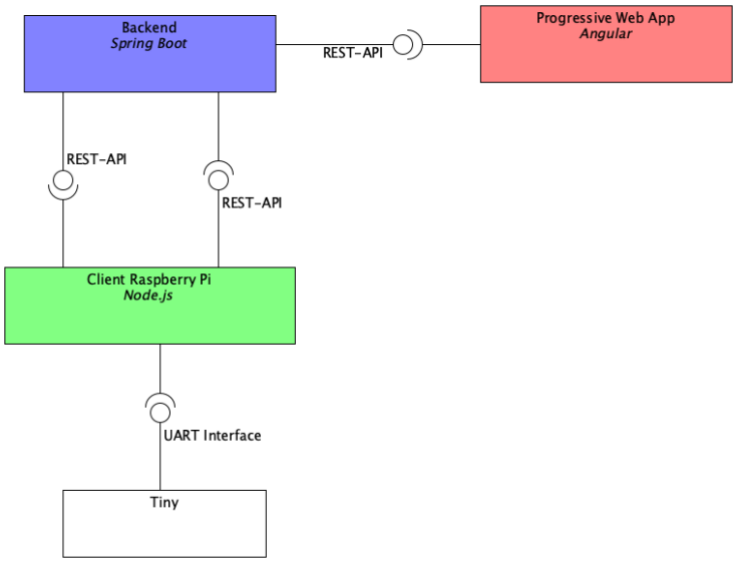
Eine weitere Herausforderung war die bidirektionale Kommunikation zwischen Station und Backend. Die Station ist via Mobilfunknetz mit dem Internet verbunden. Das Mobilfunknetz unterstützt Portforwarding nicht, was den Einsatz von DNS-Diensten wie NoIP verunmöglicht. Nach einigen Recherchen wurde der Dienst «zerotier» gefunden. Dieser bietet eine Kombination zwischen VPN und SD-WAN an und entspricht exakt den benötigten Anforderungen.

4. Ergebnisse

Im Rahmen der Bachelor-Diplomarbeit wurden die folgenden Erkenntnisse erarbeitet:

Progressive Web App «JTI Pick-Up Station»: Die Angular Applikation bietet die Schnittstelle zwischen Applikation und Anwender und entspricht Google's Anforderungen zu Progressive Web Apps.

Backend-Applikation: Die Spring Applikation ist der Mittelpunkt des Projekts. Die REST-Schnittstelle ermöglicht das Abfragen, die Erstellung und die Bearbeitung von Daten. Sie entspricht dem REST-Level 3 nach Richardson Maturity. Zugleich dient das Backend als Proxy zu 3rd Party APIs. Die Daten werden in einer relationalen Datenbank (MariaDB) persistiert.



Endpoint zur Abholung auf Raspberry Pi mit Schnittstelle zu Elektrotechnik: Die leichtgewichtige Node.js Applikation dient als Schnittstelle zwischen Informatik und Elektrotechnik via UART und kommuniziert mit dem Backend via REST.

Infrastruktur

Die PWA sowie die Spring Applikation laufen auf virtuellen Maschinen im Enterpriselab. Sie befinden sich auf unterschiedlichen Hosts, die Kommunikation läuft via https. Der Zugriff auf die PWA wird durch einen reverse Proxy geschützt. Beide Applikationen laufen als Docker Container und lassen sich via CI/CD Pipeline automatisch deployen.

Die Applikation ist vom Internet aus erreichbar.

Identifikation

Um Bestellungen platzieren zu können, wird zwingend ein Account benötigt. Dieser wird erst durch eine erfolgreiche Altersverifikation aktiviert. Um die Altersverifikation durchzuführen, wird ein Request an das Backend gesendet. Dieses leitet die Anfrage an die Jumio-API weiter. Anschliessend wird der Redirect-Link zurückgegeben und im Frontend weitergeleitet. Nach dem Abschluss der Verifikation wird der Status an das Backend gesendet und entsprechend gehandelt.

Suche von Station, Bestellung und Kauf

Bei erfolgreicher Verifikation kann sich der Nutzer nun anmelden und seine Bestellung platzieren. Dabei muss zuerst die gewünschte Station ausgewählt werden, ehe Produkte dem Warenkorb hinzugefügt werden können. Es wird bei den Produkten direkt die Verfügbarkeit an der Station angezeigt. Bei einem gültigen Warenkorb kann die Bestellung abgeschickt werden. Im Backend wird der Warenwert berechnet. Die folgende Bezahlung läuft identisch zur Altersverifikation ab. Nach der Bezahlung wird das Inventar der Station im Backend entsprechend angepasst.

Pick-Up und physische Station

Die bezahlten Bestellungen werden im Nutzerprofil angezeigt. Hier kann eine bestimmte Bestellung ausgewählt und anschliessend abgeholt werden. Zum Abholen ist das Einlesen des QR-Codes auf der Station nötig. Dieser liefert den Pick-Up Station Identifier. Basierend auf dem Pick-Up Token und dem Station Identifier können die bestellten Produkte an der jeweiligen Station ausgegeben werden.

Das Hinzufügen der Pick-Up Station wird von ihr selbst initiiert. Beim Starten der Applikation sendet die Station eine Anfrage an das Backend. Entscheidend dabei ist die URL, bzw. die IP-Adresse. Existiert diese bereits in der Datenbank, wird sie zurückgegeben. Alternativ wird eine neue Station mit einem zufälligen Namen erstellt. Diese Station ist noch nicht initialisiert und kann im Frontend vom Administrator konfiguriert werden. Dazu gehört das Setzen vom Namen, Beschreibung aber auch der Koordinaten. Bei jedem Start oder dem Öffnen der Nachfüllklappe der Station wird eine Inventur durchgeführt. Auch dieses wird an die Station gesendet. Nur wenn sich etwas geändert hat, wird ein Update durchgeführt. Grundsätzlich wird das Inventar vom Backend verwaltet, da das Erstellen einer vollständigen Inventur nur im Ausnahmefall durchgeführt wird.

5. Ausblick

Die Bachelordiplomarbeit zeigt auf, dass der Absatzkanal für die Japan Tobacco AG auf diese Art und Weise entwickelbar ist. Die gewünschten Dienste lassen sich integrieren, die rechtlichen Anforderungen an den Verkauf von Tabakprodukten einhalten.

In diesem Projekt wurde das Vorgehen als Prototyp umgesetzt. Die Entwicklungstätigkeiten wurden entsprechend priorisiert, wobei das automatisierte Testing dieser Strategie zum Opfer fiel. Dennoch ist die Applikation manuell getestet worden. Um ein späteres Refactoring zu vereinfachen, müssen automatisierte Unit- und Integrationstests implementiert werden. Es sollten den CI/CD Pipelines entsprechende Testing Pipelines hinzugefügt werden. Zudem sollte ein Test- und Entwicklungsumgebung bereitgestellt werden.

Die Usability wurde nur mit sehr wenigen Nutzern getestet. Ein Testing mit einer breiten Nutzergruppe wäre sehr aufschlussreich und würde aufzeigen, wo nacharbeitet geleistet werden muss. Die Applikation ist momentan nur in Deutsch verfügbar. Es müssten entsprechende Versionen für andere Sprachen erstellt werden, damit die Applikation international genutzt werden kann.

Damit die Applikation besser Skalieren kann, wäre ein Deployment auf einen Kubernetes Cluster sehr zu empfehlen. Die Applikationen liegen bereits als Docker-Container vor, der Aufwand wäre entsprechend gering.