THM
TECHNISCHE HOCHSCHULE MITTELHESSEN

THM | CAMPUS GIESSEN | MNI
Mathematik, Naturwissenschaften und Informatik
TECHNISCHE HOCHSCHULE MITTELHESSEN

Veranstaltung

Softwaretechnik - Projekt

Getränkemaschine: Frontend

Brian Runck, Amine Bouchrit, Beyza Duman

Dozent
Ebner, Andreas Sebastian

31. März 2023

Inhaltsverzeichnis

Erklärung

Hiermit bestätigen wir, dass wir unseren Projektbericht (CS1023) mit dem Titel

Entwicklung des Frontend für eine Getränkemaschine

eigenständig verfasst haben und die angegebenen Quellen und Hilfsmittel genutzt haben.

Gießen, 31.März.2023

X

_____-

Amine Bouchrit, Biran Runck u. Beyza Duman

Einleitung und Aufgabenstellung

Im Modul Softwaretechnik-Projekt wurde im Wintersemester22/23, die Getränkemaschine vom Blockkurs SS22 „Physical Computing“ weiter aufgebaut. Im Rahmen des Kurses wurde die Getränkemaschine in zwei Teams mit jeweils drei Teammitgliedern weiterentwickelt. Das Ziel des Projekts bestand darin, eine Maschine sowie eine Handy-App in dem Fall unsere GUI als Frontend-Team zu entwickeln, um eine benutzerfreundliche und intuitive Möglichkeit zur Bestellung von Getränken zu schaffen und das zusammenmischen seines eigenen Getränks zu ermöglichen.
Hierfür wurden im Lastenheft verschiedene Anforderungen definiert, welche in ein Backend und Frontend aufgeteilt worden sind. In der Bibliothek sollten sich dreißig Getränkevariationen befinden und das Frontend sollte auf allen gängigen Plattformen ausführbar sein. Die Umsetzung des Projekts erfolgte unter der Verwendung Open-Source-Bibliotheken und Frameworks.
Zu den verteilten Lastenheften, sollte daraufhin ein Pflichtenheft erstellt und die Rollen im Team festgelegt werden. Um die Aufgaben aus Sicht der Softwaretechnik zu überdenken, empfiehlt es sich, ein passendes Vorgehensmodell auszuwählen und mit der Technologierecherche zu beginnen, somit wurden die ersten Schritte in den Teams besprochen und festgelegt, man einigte sich auf bestimmte Bibliotheken und Tools, um im Nachhinein nicht auf Konflikte zu geraten. Es wurde einiges mit dem Backend abgesprochen, um die Schnittstelle zwischen der GUI und der Datenbank richtig zu definieren, die Verteilung der Kategorien zu bestimmen und die Datenabfrage und das Abstimmen von Datensätzen genau zu definieren.

Pflichten und Lastenheft

Die Aufgabenstellung lag darin eine grafische Endanwendung zu erstellen, die es dem Nutzer ermöglicht, die Getränkebestellung sowie die Verwaltung der Maschine vorzunehmen. Die Bestellung sollte entweder in einer Applikation mit getrennten Ebenen erfolgen oder in Form zweier unterschiedlichen Frontend-Anwendungen. Für die Getränkebestellung sollten verschiedene Funktionen zur Verfügung stehen, wie zum Beispiel das Sortieren der Getränke nach Kategorien, das Filtern nach Kriterien sowie die Ansicht der Informationen in jeder Einzelansicht. Weiterhin sollte es möglich sein, eigene Getränke zu kreieren und diese wie normale Listengetränke zu bestellen. Ein realer Bezahldienst war nicht vorgesehen, aber die Preise sollten ersichtlich sein und vom Backend abgefragt werden. Das Frontend sollte auf allen gängigen Plattformen ausführbar sein und als native Applikation gepackt werden.

Zusätzlich zu den bereits genannten Anforderungen an das Frontend sollten auch die folgenden Anforderungen erfüllt sein:

- User Applikation (Verwaltung): Für die Backend definierten Aspekte muss ebenfalls ein grafisches Interface zur Verfügung stehen, welches es dem Nutzer ermöglicht, die Maschine zu verwalten. Hierfür sollten CRUD-Dienste bereitgestellt werden, die es dem Nutzer erlauben, alle relevanten Assets zu verwalten.
- Login-Maske: Die Frontend-Applikationen sollten eine Login-Maske bereitstellen, um sicherzustellen, dass nur autorisierte Benutzer Zugriff auf die Funktionen haben.
- Optimierung für verschiedene Ausrichtungen: Die Anwendung zur Getränkebestellung muss sowohl für eine vertikale als auch horizontale Ausrichtung optimiert sein, um eine optimale Nutzung verschiedenen Endgeräten zu gewährleisten. Die Anwendung zur Verwaltung hingegen soll nur für eine horizontale Ausrichtung optimiert sein.
- Touch-Optimierung: Die Frontend Anwendung sollte für Touch-Eingaben optimiert sein, um eine intuitive Bedienung auf Touchscreen-Geräten zu ermöglichen.

Zu den genannten Anforderungen gab es außerdem erweiterte Anforderungen, die wie folgt geschildert worden sind:

- Barcode-Bestellung: Der Nutzer soll in der Lage sein, die Bestellseite der Maschine über einen Barcode an der Maschine auf seinem mobilen Endgerät aufzurufen.
- Multi-Plattform App oder Webanwendung: Die Bestellseite kann entweder als Multi-Plattform App oder als Webanwendung realisiert werden, um eine möglichst große Nutzerbasis zu erreichen.
- Kalkulation der Kosten: Wenn der Nutzer eine Eigenkreation bestellt, sollen die Kosten auf Basis der ausgewählten Zutaten und Mengen berechnet werden. Die notwendigen Informationen müssen in der Verwaltung ergänzt werden.
- Speichern und Teilen von Eigenkreationen: Der Nutzer soll in der Lage sein, eine Eigenkreation mit einem Namen zu versehen und lokal zu speichern. Außerdem soll es möglich sein, Eigenkreationen unter einem vom Nutzer festgelegten Namen für andere Nutzer zugänglich zu machen und als Barcode zu teilen. Darüber hinaus soll es eine Möglichkeit geben, Eigenkreationen per Knopfdruck über Social-Media oder auf einer Website online zu teilen.
- Bewertung von Rezepten: Die Rezepte in der App sollen bewertet werden können, um anderen Nutzern bei der Auswahl des Getränks zu helfen. Eine Bewertung ist jedoch nur möglich, nachdem der Nutzer das entsprechende Getränk bestellt hat.

Angewendete Softwareplanung

Während der Softwareentwicklung musste vorerst ein Vorgehensmodell ausgewählt werden nach diesem wird im Team entwickelt. Ein Vorgehensmodell definiert einen allgemeinen Rahmen für den organisatorischen Prozess der Softwareentwicklung. In diesem werden beispielsweise, die Tests, Implementierung, User Stories etc. in Phasen unterteilt und je nach Modell iteriert. Vorab werden alle durchzuführenden Aktivitäten, der Arbeitsablauf, Inhalt, Kriterien (Pflichten und Lastenheft) bestimmt und die Verantwortlichkeiten werden unter den Mitgliedern verteilt. Methoden, Werkzeuge sowie Lizenzen werden abgestimmt. Das Ziel eines Vorgehensmodell ist es die Zeit, Budget und die Qualität des Ergebnisses zu kontrollieren. Außerdem soll dies das Kommunikationsverhalten innerhalb des Projekts anspornen.

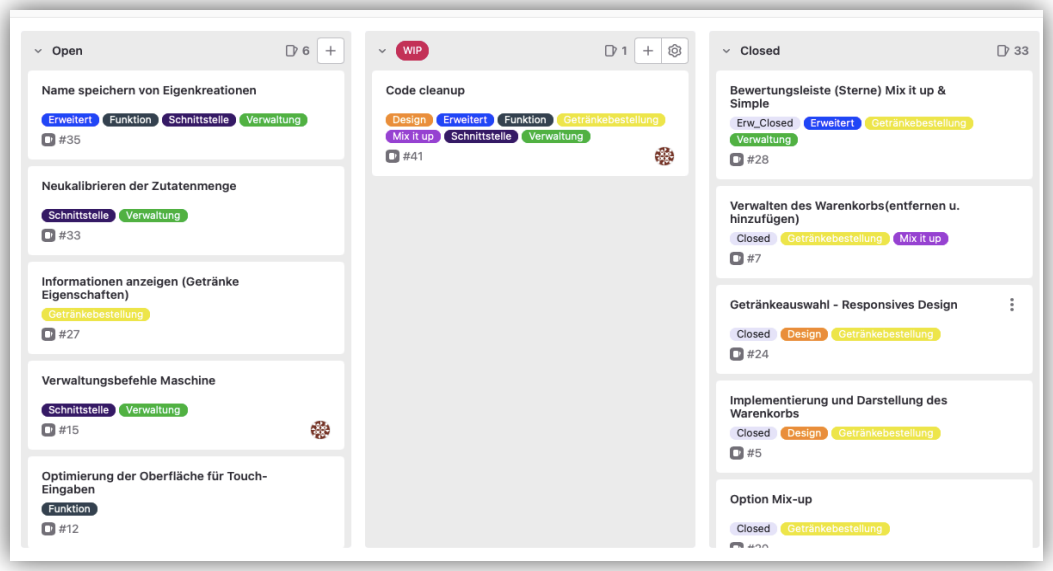
Vorgehensmodell – Kanban

Das Frontend-Team hat sich für das Kanban Modell entschieden. Das Kanban Modell wird für die agile Softwareentwicklung genutzt. Der Kern ist hierbei ein Just-in-Time-Produktionsverfahren (JIT), dass alle Projektziele in einem fließenden Fluss optimiert und für die Herstellung des Endprodukts zu erledigen ist in Abhängigkeit von der Kapazität des Entwicklungsteams. Das Kernkonzept besteht darin, den Produktionsprozess so ordentlich wie möglich zu halten, das bedeutet: verbesserte Organisation, fokussiertes Arbeiten, Steigerung der Produktivität, bessere Priorisierung von Aufgaben und lässt gleichzeitig Raum für Agilität und Fähigkeit, um schnell auf Veränderungen zu reagieren. Außerdem wird hierbei durch die täglichen Meetings am Kanban Board die Zusammenarbeit in den Teams gefördert.

Alle Aufgaben werden bspw. auf kleine Post-it‘s kurz definiert und in die Spalte „Stories“ gehängt. Hierbei haben wir das Frontend GitLab als digitales Kanban genutzt, es unterscheidet sich hierbei keinesfalls von einem Kanban Board an einer Wand. Die einzelnen Aufgaben kann man in Farben und Kategorien unterscheiden und priorisieren.
Nach Besprechung und der Verteilung der Aufgaben, zieht sich jedes Teammitglied seine für ihn verantwortliche Story und bearbeitet diese, somit rutschen die Aufgaben von „To-Do -> „in Progress“, im Anschluss werden diese getestet und nach dem letzten Schliff in die „Done“ -Spalte geschoben. Dieser Prozess iteriert so lange bis am Ende in der vorgegebenen Zeit keine Aufgaben noch offen sind. Natürlich wird jeden Morgen während der Entwicklung, sich im Team zusammengesetzt und über die Fortschritte, Entscheidungen und entstandenen Probleme diskutiert.

Bewertung des Vorgehensmodell

Wir haben persönlich das erste Mal mit einem Vorgehensmodell bewusst gearbeitet. Das Kanban Modell schien uns auf dem ersten Blick am besten anzusprechen. Die Idee dahinter sich täglich zusammenzusetzen und über die Ergebnisse zu sprechen um besser entwickeln und organisieren zu können klang für uns vielversprechend, um erfolgreich und schnell unser Projekt fertigzustellen. Die Umsetzung dahinter hat über das Git Tool gut funktioniert, aber durch die fehlende Erfahrung war die Unterteilung nicht ganz korrekt und wir mussten im Verlauf der Entwicklung immer wieder den Titel oder die Kategorisierung umändern. Es war ebenfalls schwer hierbei nur an einer Aufgabe dranzubleiben und sich fokussiert mit dieser einen gewissen auseinanderzusetzen. Durch fehlendes Feedback des Kunden wurden Aufgaben die schon als erledigt markiert worden waren wieder zurückgezogen und waren wieder im „Work-in-progress“. Hat man beispielsweise den Button für den Warenkorb unten rechts erstellt, wurde nach späterer Absprache dieser wieder neu oben rechts erstellt. Hat man eine Aufgabe erledigt, die Teil einer anderen war, wurde im späteren Verlauf bewusst, dass beide nicht gut miteinander fungieren, somit musste Aufgabe 1 angepasst an Aufgabe 2. Unsere Arbeit mit diesem Vorgehensmodell war nicht recht agil. Die Agilität hat in unserem Team und im Prozess gefehlt. Die tägliche Zusammenarbeit und Kommunikation untereinander und zwischen dem Kunden hat gefehlt. Dies ist sehr wichtig, um eine klare Vorstellung zu erhalten und um gezielter an dem Endprodukt zu arbeiten, denn es kann sein das sich der Kunde während der Entwicklung für andere Ideen entscheidet oder eine andere Vorstellung bekommt von seinem gewünschten Endprodukt. Natürlich haben wir morgens oftmals über den aktuellen Stand gesprochen und besprochen an was jeder nun weiterarbeitet, manchmal haben wir auch an einer Aufgabe zusammen dran gearbeitet oder es kam dazu, dass wir alle an derselben Aufgabe gleichzeitig gearbeitet haben, wir haben unsere Ergebnisse in diesem Fall fusioniert oder das Beste Ergebnis rausgepickt und implementiert. Ab und zu haben wir uns mit dem backend zusammengesetzt und über Problemstellungen oder die Entwicklungsart und Technologien diskutiert. Selbst zwischen den Teams hat das agile Arbeiten nicht wie in der Realität bzw. in der Arbeitswelt funktioniert. Der Hintergrund hierfür kann die geringe Zeit, der hohe Arbeitsaufwand gewesen sein und die nicht vorhandene Erfahrung sein.



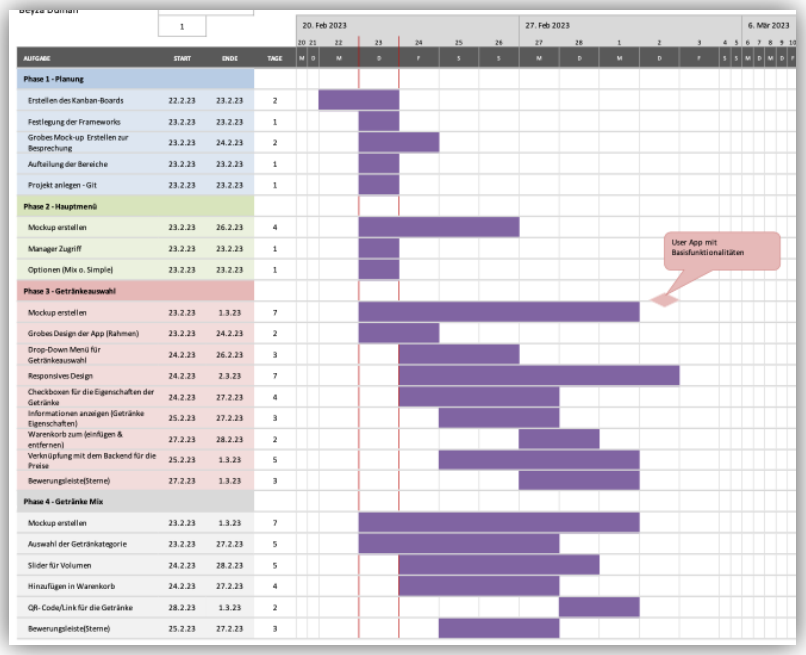
Am Anfang war es schwer sich in die agile Arbeitsweise zurecht zu finden, doch nach einigen Tagen funktionierte es immer mehr und mehr. Gegen Ende des Projekts haben wir uns unserer Meinung nach besser in die agile Methode und dem Kanban zurechtgefunden. Wir fragten öfters nach den genauen Kunden Spezifikationen nach, klärten Missverständnisse und präsentierten unseren aktuellen Stand. Das Kanban Board kam öfters zum Einsatz und wurde regelmäßiger aktualisiert.
Im Großen und Ganzen ist das Kanban Vorgehensmodell eine sehr gute agile Methode, um das bestmögliche Ergebnis zu erzielen. Es erfordert die Disziplin und Offenheit sich regelmäßig auszutauschen und zu kommunizieren. Wenn man den Dreh raushat, macht die Entwicklung noch viel mehr Spaß.

GANTT-Diagramm

Ein Gantt-Diagramm wird als Balkendiagramm erstellt, dieses visualisiert einen Projektplan und deren Phasen, welche Arbeit in welcher Reihenfolge durchgeführt wird, und ihre dazugehörige Zeitleiste werden in solch einem Diagramm veranschaulicht. Es eignet sich für eine strukturierte Vorgehensweise und um den Kunden zu illustrieren, wie und wann man vorgeht. Genauer gesagt, werden verschiedene Aufgaben dargestellt, definiert, wann sie beginnen und enden, welche Arbeitsdauer geplant ist, wo sich welche Aktivitäten überschneiden und für wie lange. Es werden Meilensteine festgelegt und eingetragen wann diese zu erreichen sind. Vor allem das Start- und Enddatum des gesamten Projekts wird repräsentiert. Das Gantt-Diagramm hilft dem Team auf einen Blick zu erkennen, welche Aufgaben wann erledigt werden und wieviel Zeit jede einzelne Aufgabe beanspruchen wird.

Bewertung des GANTT-Diagramm

Das GANTT-Diagramm war das erste, was wir in unserer Gruppe erstellt haben, nach Bearbeitung des Lasten und Pflichtenhefts. Es ist eine gute Übersicht, um sich klarzuwerden, wie viele Aufgaben und Phasen man eigentlich benötigt. Der Kunde kann hiervon auch profitieren und sieht nach welchen Phasen wir arbeiten werden und welche Aktivität wir wann Vorhaben zu erledigen. Uns ist bei der Erstellung schwer gefallen die Zeit richtig einzuschätzen. Wir hatten zu viele Aufgaben, die wir in einer geringen Zeit fertigstellen sollten, zumal waren wir auch nur drei Personen, die an allen Aufgaben gearbeitet haben. In Unserem Diagramm haben sich viele Balken überschnitten und es kam zu einem unrealistischen Ergebnis. Wie schon im Kanban beschrieben hatte uns allgemein die Kommunikation unter den Teams und dem Kunden am Anfang gefehlt, somit könnte man sagen wir haben uns nicht ganz an dem Gantt Diagramm orientiert und der Kunde oder Arbeitsgeber konnte dies ebenfalls nicht anhand des Diagramms. Wir sind der Meinung, es ist ein gutes Diagramm, um genau auf einem Blick darzustellen, was erledigen muss und in welchem Zeitraum, es regt die Arbeitsweise an und fördert das zielstrebige Erreichen aller Meilensteine. Somit kriegt der Arbeitsgeber ebenfalls ein sicheres Gefühl und weiß womit er bis wann rechnen kann und Ergebnisse vor sich liegen hat. Eine solche Darstellung wäre am Anfang jedes Projektes nicht falsch, aber sich 1:1 daran zu halten, würde in der Realität wahrscheinlich nicht hundertprozentig funktionieren, da es immer wieder im Verlauf der Entwicklung zu Problemen, Denkanstöße, Diskussionsbedarf etc. führt und eine Aufgabe sich eventuell somit nach hinten verschiebt und die darauffolgende nachziehen muss. Es war ebenfalls schwer diese ganzen Schritte auf drei Personen aufzuteilen in Relation der vorgegeben Zeit.

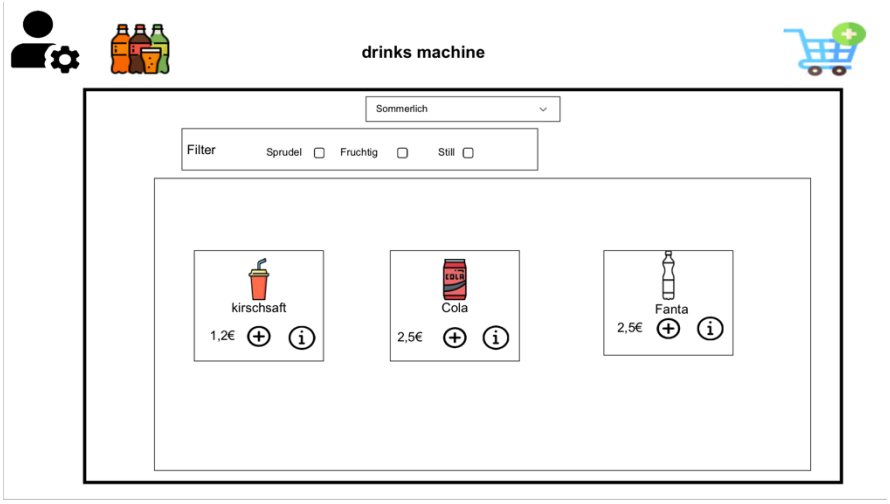


Software-Design und Entwurf

In den einzelnen Unterpunkten wird beschrieben, wie die Software entstanden ist, welche Mock-up‘s am Anfang des Prozesses erstellt worden sind, die Umsetzung der Benutzeroberfläche, um ein userfreundliches Ergebnis zu erzielen und die Schnittstelle zum Backend repräsentiert.

Mock-up’s

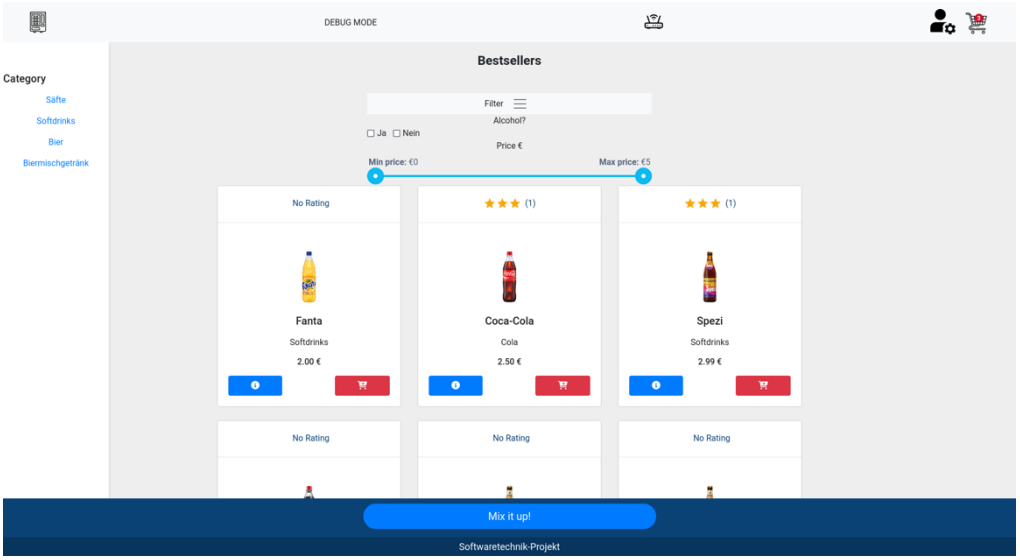
Zuallererst haben wir uns eine visuelle Darstellung von der angeforderten GUI überlegt. Das Lastenheft sowie das Pflichtenheft und die Kommunikation mit dem Dozenten hat es ermöglicht die naheliegendste Illustration zu erstellen, die dem Arbeitgeber sowie für uns entspricht. Die Mock-ups haben wir mit dem *Prototyping- und Wireframing-Tool* erstellt. Dieses Tool wird verwendet, um Highfidelity-Prototypen von Web- und Mobil-Apps zu erstellen. Es erzielt die realistischen Versionen eines fertigen Produkts und bietet Funktionen für Zusammenarbeit, Interaktion und Design.
Am Ende erzielt man eine App, die sich kaum mehr von dem fertigen Produkt unterscheidet. Der einzige Nachteil, der auftauchen kann, ist die hohe Detailtreue die zu Diskussionen über persönliche Vorliebe z.B, die Farbwahl oder Button-Designs führt.



Erstes Mock-Up: Home Bildschirm

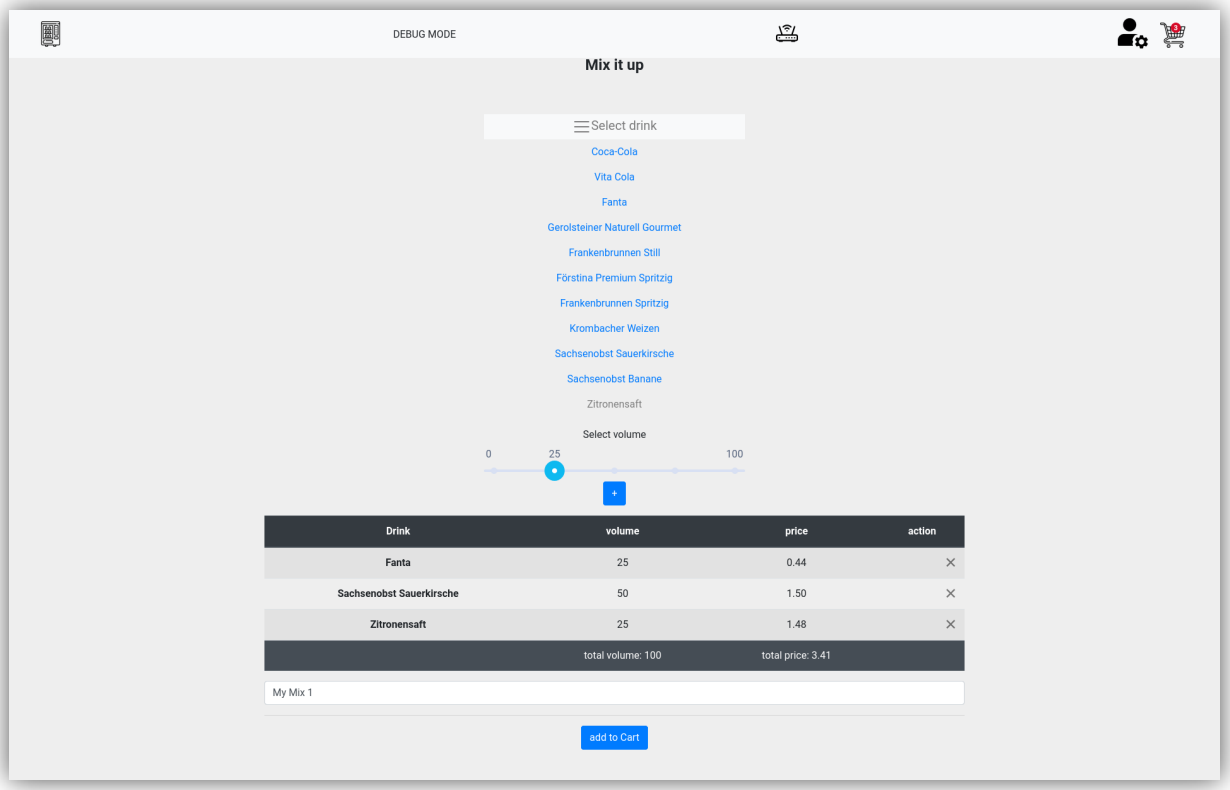
Realisierung des Home-Bildschirms:

Die Illustration unterscheidet sich zur Realisierung nicht gigantisch. Die Kernelemente und Kompositionen sind sehr ähnlich.



Umsetzung der Benutzeroberfläche

Die GUI, Abkürzung für Graphical User Interface ist unsere Schnittstelle, über die ein Benutzer mit Endgeräten wie bspw. einem Computer, Tablet oder Smartphone interagiert. Uns war wichtig diese Benutzeroberfläche mit so vielen klar definierten Buttons wie möglich zu gestalten, um das erstmalige Nutzen für den User zu vereinfachen und ein schnelles Navigieren bis hin zur Bestellung zu erzielen. Vorab gehörte die Identifizierung von Bedürfnissen des Nutzers, die wir nach Absprache unseres Dozenten erhielten.
Man kann die GUI mit einem Zeigegerät wie eine Maus, einem Stift oder dem Finger auf einem Touchpad steuern. Ziel ist es vorherzusagen, was Benutzer möglicherweise tun werden. Dazu enthält die Benutzeroberfläche bestimmte Elemente sowie Buttons, auf die leicht zugegriffen werden kann und die weitere Aktionen ermöglichen.
Die Navigationsleiste ist Bestandteil für eine Oberfläche. Sie ist am oberen Bildschirm fixiert und bewegt sich beim Scrollen mit. Sie enthält die wichtigsten Kernelemente, sowie das Verwalten der Maschine, das Login, den Warenkorb und das Logo, um von jeder Seite wieder auf den Home-Bildschirm zurückzukehren. Der Footer besteht im Home Bildschirm aus einem Titel und einem Button, der den User auf die Seite zum Mischen von Getränken bringt. Navigationsleiste und Footer sind beide fixiert, um eine freundliche Nutzung zu ermöglichen und das ständige Suchen von fundamentalen Elementen zu vermeiden.



Durch Drop-Down Menüs und einem Slider, der sowohl leicht mit der Maus, aber auch mit dem Finger bedienbar ist, wird das Kreieren von Getränken dem Nutzer einfach gestaltet (siehe obige Abbildung).

Kommunikation mit dem Backend

Die angezeigten Daten in der GUI werden über Abfragen im JSON-Format vom Backend angefragt und abgerufen. Nach Absprache und Diskurs mit dem Backend haben wir uns für das Datenformat JSON festgelegt.
Für die Verbindung zwischen Frontend und Backend wird ein MQTT-Broker verwendet. Dieses wurde damals bereits für die Ansteuerung der Maschine eingesetzt, somit haben wir beschlossen, dieses wiederzuverwenden.
Um eine Anfrage zu starten, schreiben wir unsere JSON formatierten Anfragen in eine TypeScript Datei und dem bestimmten MQTT-Topic, diese werden beim Starten an den Server (Backend) gesendet. Im Backend wird die Anfrage bearbeitet und wir erhalten eine Response zurück. Diese werden in unserer GUI angezeigt. Außerdem senden wir die Bestellbestätigung direkt an den Server und dieser verarbeitet die Anfrage und sendet eine Nachricht an die Maschine, die im Anschluss die Bestellung ausführt und die bestellten Getränke zusammen mischt.

Darlegung der Technologieentscheidung

MQTT

```
159     //Order for simple drinks
160     this.eventMqtt.setPublishTopic('/create/order')
161     for (let i=0;i<=shoppingCartEntry.quantity; i++) {
162       this.eventMqtt.setPublishMessage({"machineID": Number(localStorage.getItem('machineID')),
163                                         "recipeID": shoppingCartEntry.drink.recipeID});
164       this.eventMqtt.doPublish();
165     }
```

MQTT wird für die Kommunikation vom und zum Backend verwendet. Mit der Verwendung von MQTT wird die Schnittstellenverwaltung im Backend erleichtert, da das Backend für die Kommunikation zur Getränkemaschine ebenfalls MQTT verwendet. Für das Frontend stellte die Entscheidung keinen erheblichen Mehraufwand dar. Zudem wird mit MQTT die weltweite Erreichbarkeit von Frontend, Backend und Getränkemaschine vereinfacht.

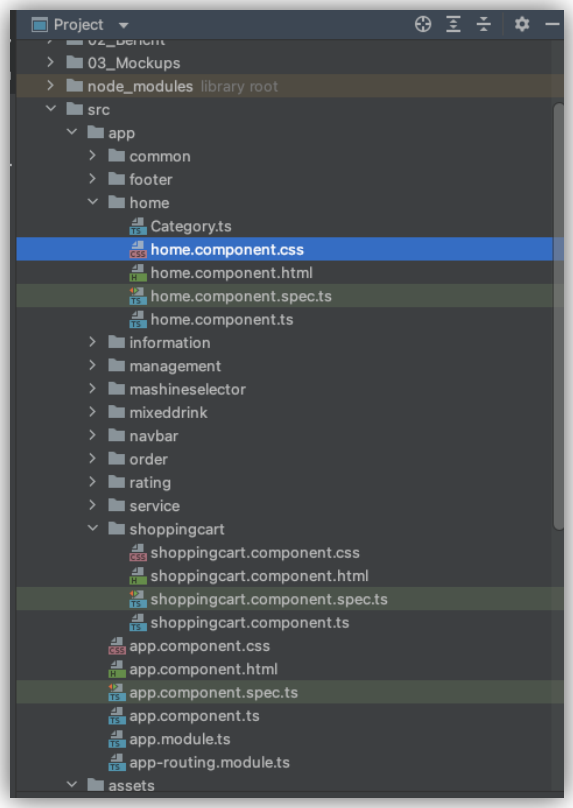
JSON

Für dein Transport der Nachrichtenpakete wurde auf das JSON Format gesetzt. Die JSON-Pakete sind menschenlesbar, und lassen sich leicht von und zu Javascript-Objekten umwandeln.

```
319     this.subscriptionCa = this.eventMqtt.subToTopic(
320       // Callback function for handling received messages
321       (data : IMqttMessage) => {
322         // Parses the received message as JSON
323
324         let response = JSON.parse(data.payload.toString());
325         // Checks if there was an error in the response
326         if(response.response == 'Error'){
327           console.error('error:', response.value);
328           return;
329         }
330         console.log('Observer got a next value: ' + response);
331
332         // Checks if the response is for the correct topic
333         if(response.response !== 'category') {
334           return;
335         }
336
337         // Adds each category to the list of all categories
338         for (let item of response.value) {
339           this.allCategoryList.push(item);
340          // console.log(item);
341         }
```

Angular

Angular ist ein TypeScript-basiertes Front-End-Webapplikationsframework und wurde als Open-Source-Software publiziert. AngularJS hat eine VC-Struktur, die View-Controller Struktur, mit dieser hat sich die GUI einfach und unkompliziert entwickeln lassen.
Das Prinzip besteht darin, je nach Aufgabe oder Seite eine neue Komponente zu erstellen, sprich alle Kernfunktionen sind in klar definierten Komponenten untergeordnet. Eine Komponente beispielsweise *shoppingcart* besteht aus vier weiteren Dateien, der css – für das Design, html – für die Struktur des Webdokuments, ts – die Typescript Datei in der wir unsere Funktionalitäten wie die Anfragen an das Backend definieren und die spec.ts der unsere Testfälle definiert und überprüft.



Bootstrap

Bootstrap ist ein freies Frontend-CSS -Framework, welches auf HTML und CSS basierende Gestaltungsvorlagen und Typografie, Formulare, Buttons und andere Oberflächengestaltungselemente enthält, sowie optionale JavaScript- Erweiterungen.
Wir haben größtenteils Bootstrap für das Design, die Kompositionen und der responsiven Seiten verwendet.
Bootstrap besitzt bestimmte Designflächen die mit Angular nicht kompatibel sind, deswegen ist es wichtig Angular/Bootstrap Widget über den Terminal runterzuladen. Alle Bootstrap-Widgets werden somit an das einheimische Angular angepasst und Widgets wie Karussell, Navs werden wieder nutzbar.

Test und Validierung

Zum Testen des User-Interfaces bekamen wir ein Tablet mit Touch Funktion zur Verfügung gestellt. Dieser wurde mit dem Rechner verbunden und wir testeten nach jeder Iteration, wie gut das Navigieren durch einzelne Etappen funktioniert.
Wir haben die Umsetzung mit der Touchpad Bedingung gut realisiert durch große und viele Buttons, Drop-Down Menüs und einem übersichtlichen und sauberen Benutzeroberfläche.

Zusammenfassung und Ausblick

Retrospektiv betrachtet wurden die Aufgabenstellungen in Bezug auf unser Ergebnis vollständig und sauber bearbeitet. Der Umfang war im Vergleich zum Backend breiter, aber es hat sich am Ende ein ausgezeichnetes Ergebnis herausgestellt mit der wir sehr zufrieden sind. Die Benutzeroberfläche ist komplett ausgebaut und kann weiterhin genutzt werden, sie ist funktionstüchtig und weist keine Fehler auf. Alle vorhandenen Fehler wurden mit großer Zuversicht behoben. Es war viel Detailliebe benötigt, um am Ende die perfekte App zu gestalten und alle Spezifikationen zu erfüllen (erweiterte ausgeschlossen).
Die Alkohol-Filter Option weist als einzige ein Problem auf, da hier die genauen Informationen vom Backend fehlen und man diese nicht aufrufen kann.
Der Benutzer kann nun über die App die Maschine sowie Getränke verwalten, Bestellungen aufgeben, Getränke kreieren und eine Bewertung abgeben.
Wir freuen uns unsere Arbeit weiterzureichen und hoffen man kann das Projekt weiterbearbeiten und erweitern.
Hierfür haben wir uns als Ausblick auf die bevorstehende weiter Entwicklung folgende Erweiterungen überlegt:

Home-Oberfläche

Die einzelnen Getränke Cards besitzen einen Button mit dem „i“ Symbol für Informationen. In dieser fehlen die Hersteller Informationen und Inhaltsstoffe, diese müssten mit Absprache des Backend festgelegt und implementiert werden.
Es herrschte im Laufe der Entwicklung immer wieder Diskussionsbedarf, wenn es um die Unterteilung der Kategorien ging, diese müssten in eine ordentlichere und einheitliche Struktur sortiert werden, es sollten Rezepte und Getränke klar voneinander unterschieden werden.
Des Weiteren sollte die Möglichkeit bestehen in den Cards die Getränkeanzahl zu bestimmen(+/-), dies ist momentan nur im Warenkorb möglich.
Dazu könnte man ebenfalls die Größenauswahl oder die Milliliter ergänzen. Der Nutzer kann die Menge an Flüssigkeit bestimmen oder sich sein eigenes Glas wählen bspw. entscheidet er sich ein Weißwein aus einem Weinglas zu trinken oder vielleicht doch in einem Shotglas.
Die Implementierung der Filterungsoption weist auch Fehler auf, der Hintergrund hierbei liegt am Backend. Es sollte sichergestellt sein, dass die genauen Anforderungen beschrieben sind und eine offene Kommunikation mit dem Backend-Team pflegt, um eine erfolgreiche Implementierung zu gewährleisten.
Schlussendlich könnte man den Filter umstrukturieren und aus einem Ja/Nein mit klickbaren Optionsboxen, den Promille Wert in % bestimmen.

Bewertung

Die Bewertung nach der Bestellung für einzelne Getränke wurde implementiert, aber ist bis dato noch nutzlos, da es noch keine Schnittstelle hierzu zum Backend existiert. Das Speichern von Daten aus dem Frontend an das Backend wurde nicht implementiert und ist ein fehlender Bestandteil der Bewertung.
Die Bewertungen, die vom Benutzer über die GUI getätigt werden, sollten im Backend gespeichert und neu kalkuliert werden, um diese wieder für die einzelnen Getränken anzuzeigen. Ein Bewertungsbogen wäre praktisch für den Prozess, wo der Benutzer beispielsweise persönliche Daten eingibt wie Name, Wohnort und Alter. Er kann einen Feedback Text darunterschreiben und diesen einreichen. Alle Bewertungen werden veröffentlicht und sind über die einzelnen Getränke für jeden Benutzer abrufbar.

Maschinenverwaltung

Die Verwaltung besitzt nur eine Maschine. Man könnte diese erweitern und warme Getränke anbieten durch Hinzufügen einer Kaffeemaschine, die nicht nur Kaffee, sondern auch eine heiße Schokolade anbietet.
Außerdem könnte man die Maschinen splitten und eine Softgetränkemaschine und eine Alkoholgetränkemaschine ergänzen.

Mix it up

Nach Kreierung der eigenen Getränke sollte ein Speichern der Getränke mit dem selbstkreierten Namen möglich sein. Am besten sollte darauf unter einer neuen Kategorie zugegriffen werden und ein erneuter Kaufprozess möglich sein.
Die selbsterstellten Getränke können über einen QR-Code an andere geteilt und eingelesen werden.
Wir möchten darauf hinweisen, dass die Barcode-Bestellung in unserem Projekt nicht implementiert wurde und somit als "missing feature" zu betrachten ist. Wir haben uns stattdessen auf die Umsetzung der anderen Anforderungen konzentriert und diese erfolgreich realisiert.

Social Media

Kreierte Getränke oder die Lieblingsgetränke können auf den sozialen Plattformen veröffentlicht werden und mit Freunden geteilt.

Quellenverzeichnis

Bildquelle: https://agilescrumgroup.de/kanban/