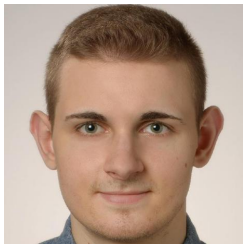
Dokumentation

über die Anwendung „ordered.online“
der Gruppe 4 im Praktikum „Service and Cloud Computing“
an der TU Dresden im Wintersemester 2019/2020

Felix Kästner, Philipp Matthes

Angaben zum Team, Vorgehensweise

Da unser Projekt mit dem Modul „Application Development for Mobile and Ubiquitous Computing“, gelesen von Dr. Springer, kombiniert wurde, bestand unser Team aus zwei Personen, statt drei Personen.



B. Sc. Felix Kästner
Master Informatik

Philipp Matthes
Diplom Informatik

Da von Beginn an fest stand, dass die im Rahmen des Praktikums zu erstellende Applikation mit beiden Modulen kombiniert werden würde, musste eine stringente und strukturierte Projektplanung, mit Hinblick auf die Erfüllung der Kriterien beider Modulpraktika, durchgeführt werden.

Zunächst zu finden war eine Idee, die sich in Serverapplikation und Clientapplikation separieren lässt, wobei die Serverapplikation vorrangig Konzepte aus der Vorlesung „Service and Cloud Computing“ und die Clientapplikation vorrangig Konzepte aus der Vorlesung „Application Development for Mobile and Ubiquitous Computing“ wiederspiegeln sollte.
Im Rahmen der Ideenfindung ergaben sich mehrere App-Ideen, von denen wir unsere finale App-Idee durch verschiedene Kriterien (Vermarktbarkeit, Innovativität, Applizierbarkeit) auswählten.

Unsere App soll es ermöglichen, an teilnehmenden Restaurants und Bars, allgemein bezeichnet als „Locations“, Produkte zu bestellen. Daher entschieden wir uns für den Namen „ordered.online“.

Wir erarbeiteten zunächst ein visuelles Grundkonzept, sowie ein minimalistisches Corporate Design mit Fonts, Farben und einem Logo. Anschließend an die Erarbeitung des visuellen Grundkonzeptes entwickelten wir im Rahmen der Kickoff-Präsentationen erste visuelle Konzepte und Mockups. Parallel hierzu entwickelten wir erste Konzepte zur Architektur der Applikation und entschieden uns für die Technologien, welche wir verwenden möchten.
Bezüglich der Server-Applikation entschieden wir uns für eine Serviceorientierte Architektur auf basis von miteinander vernetzten Microservices, jeweils basierend auf Python und dem Web-Framework Django. Für unsere Client-Applikation bestanden grundsätzlich mehrere Möglichkeiten der Auswahl der zugrundeliegenden Technologien. Trotz unserer Erfahrungen im professionellen Bereich in der nativen iOS- und Android-Entwicklung entschieden wir uns für das hybride React-Native Framework in Kombination mit dem Framework Expo. Grund der Entscheidungen, sowohl serverseitig, als auch clientseitig, war, dass wir unseren Lerneffekt nochmals amplifizieren wollten, indem wir uns jeweils mit neuen Technologien auseinander setzen.

Nach der theoretischen Konzeption der Applikation fingen wir mit der Implementation der Server-Applikation an, welche innerhalb weniger Wochen implementiert wurde. Anschließend containerisierten wir unsere Applikation mit Docker und Docker-Compose. Hierzu entwickelten und implementierten wir ein Containerisierungskonzept, welches die Orchestrierung und die Kommunikation der einzelnen Docker Container beschreibt.

Auf Grundlage der implementierten Server-Applikation konnten wir nun auch die Client-Applikation implementieren. Dies geschah kontinuierlich über fast den gesamten Zeitraum der Projektbearbeitung und brachte teilweise neue Anforderungen an die Server-Applikation mit sich, wodurch die Server-Applikation mehrere Male zu kleinen Teilen refaktorisiert wurde.

Parallel hierzu banden wir die in einzelne Microservices geteilte Server-Applikation in den Continuous-Integration-Service Travis ein und realisierten hierdurch auf unkonventionelle Weise eine kontinuierliche Aktualisierung unserer englischsprachigen Dokumentation, sowie die kontinuierliche Distribution der Dokumentation auf GitHub Pages (Continuous Delivery). Auch die hybride Client-Applikation banden wir zur Ausführung von Tests in den CI-Service Travis ein.

Neben der Implementation und Integration des Projektes hielten wir stets unsere Dokumentation des Projektes aktuell und verfassten eine Open API 3 Spezifikation unserer Services. Als wesentlicher Teil der Entwicklung unserer Client-Applikation recherchierten wir Design-Ideen über die Plattform Pinterest, welche wir zum Teil in der Finalisierungsphase der Client-Applikation integrieren konnten.

Installationshinweise und Bedienungsanleitung

Zur Installation des Clients klonen Sie sich bitte zunächst das Repository rekursiv:

```
$ git clone https://bitbucket.org/tudresden/ws2019-gruppe4 --recursive
```

Vergewissern Sie sich, dass sich im geklonten Repository ein Submodul „infrastructure" befindet. Dies ist der Ordner, in dem sich alle Applikationen befinden. Wechseln Sie in diesen Ordner.

Die Applikation benötigt Docker und Docker-Compose. Wir empfehlen, die aktuellste Version beider Dienste vorab zu installieren.

Zum Starten der Applikation befolgen Sie bitte die Anweisungen auf der Dokumentationsseite zur Applikation:

http://documentation.ordered.online/infrastructure

Beispiel für das Starten:

```
$ cd ws2019-gruppe4/infrastructure
$ docker-compose --project-name ordered-online -f docker-compose.yml up
```

Der Build-Prozess dauert i.d.R. 10 bis 30 Minuten.
Wenn alle Services gestartet sind, öffnen Sie den Browser unter der URL:

http://localhost/app für die Client-Applikation
http://localhost/manager für die Manager-Applikation

Schnittstellenbeschreibung

Wir stellen eine Open API 3 Schnittstellenbeschreibung zur Verfügung:

- Als yaml:
  https://github.com/ordered-online/documentation/blob/master/specification.yaml
- Als auf SwaggerHub integrierte Beschreibung:
  http://documentation.ordered.online/specification.html

Feedback und Kritik zum Praktikum

Wir haben uns bewusst für das Praktikum und die Vorlesung „Service and Cloud Computing“ entschieden, da wir den praxisnahen Charakter des Praktikums und der Vorlesung schätzen. Wir konnten im Rahmen des Praktikums unsere Kompetenzen in der Entwicklung und Konzeption von serviceorientierten Anwendungen in großem Maße fortbilden. Besonders gut fanden wir den freien Handlungsspielraum und die Flexibilität in der Herangehensweise an die Entwicklung der Applikation.

Allerdings resultiert aus ebendieser Freiheit auch ein großer Spielraum in der notwendigen Komplexität und Größe der Applikation. Wir würden uns wünschen, wenn zukünftig im Praktikum ein ähnlich hoher Anspruch an die Umsetzung der Konzepte aus der Vorlesung gestellt wird, wie wir ihn an uns selbst hatten. Didaktisch sinnvoll wäre zum Beispiel, dass eine serviceorientierte Architektur klar als notwendiges Kriterium an die Applikationen gestellt wird. Die Applikationen anderer Gruppen machte, nach der Anfangspräsentation, im Unterschied zu den in der Vorlesung vermittelten Konzepten, teilweise einen eher monolithischen Eindruck, wie man ihn im Rahmen des Softwarepraktikums häufig findet.

Um dies zu vermeiden, schlagen wir ein ähnliches didaktisches Konzept vor, wie es bereits in „Application Development for Mobile and Ubiquitous Computing“ umgesetzt wird. Das Konzept besteht darin, dass es mindestens eine Zwischenpräsentation gibt, in der vorgestellt wird, wie sich Konzepte aus der Vorlesung in der Applikation anwenden lassen. In „Application Development for Mobile and Ubiquitous Computing“ findet dies in Form einer Zwischenpräsentation statt, in der Adaptionskonzepte und der Kontext der Applikation ausgearbeitet werden sollen. Somit orientiert man die Umsetzung der Applikation erneut auf die Konzepte der Vorlesung.

Besonders interessant fanden wir, zusätzlich zum regulären Vorlesungsteil, die Vorlesungen, in denen externe Spezialisten eingeholt wurden.

Uns hat das Praktikum zu „Service and Cloud Computing“ sehr gut gefallen, besonders dessen Kombinierbarkeit mit „Application Development for Mobile and Ubiquitous Computing“. Wir würden die Lehrveranstaltung daher grundsätzlich jedem Studierenden der Fakultät Informatik empfehlen.