Vorgehen

Github

Als aller erstes haben wir uns ein Github Repository erstellt, damit wir gemeinsam am Projekt arbeiten können. Danach haben wir den Wöchentlichen Workshop erledigt und diese auf unser Repo gepusht. Kurz vor den Weihnachtsferien waren wir endlich fertig mit den Workshops und konnten damit beginnen zusätzliche Features zu implementieren. Damit dies geordnet und Strukturiert erfolgte haben wir einige Github Issues erstellt. Initial waren es nur 10 Stück aber während dem Arbeiten wuchs die Anzahl der Issues auf ganze 20 Stück an. Um den Status der einzelnen Tasks festzuhalten haben wir auf Github ein Projekt Board erstellt. Dabei handelt es sich um ein ganz Normales Caban Board auf dem man Issues vom initialen Status TODO in den Status Doing und schlussendlich auf Closed stellen konnte. Dank der Anwendung dieser Tasks war es für jedes Gruppenmitglied jederzeit möglich zu erkennen welche Aufgaben noch offen sind und welches das seine Nächsten Aufgaben sind.

Das Frontend

Nachdem das die Initialen Tasks erstellt wurden haben wir daran gearbeitet ein Frontend zu implementieren. Wir haben zuerst mit google nach verschiedenen Möglichkeiten gesucht und diese auch ausprobiert. Zwei möglichkeiten welche in die engere Auswahl kamen sind Thymeleaf und Angular als Frontend Frameworks. Wir haben uns schliesslich für Thymeleaf entschieden. Mithilfe von Thymeleaf haben wir dann eine Login Seite erstellt auf der man sein Camp auswählen kann. Damit dies klappt haben wir ein neues Modell Objekt Camp im Campservice erstellt.

Endpoints

Zudem haben wir verschiedene Endpoints erstellt, unter anderem einen um neue Camps zu erstellen, einen um neue Heroes zu erstellen, einen um Heroes umzubenennen und einen um auf die Profilbilder der Heroes zuzugreifen. Neben diesen änderungen haben wir auch noch neue Services erstellt. Wir haben auch alte Endpoints angepasst unter anderem haben wir den Promoterservice angepasst, so dass man ein Camp übergeben kann. Wir haben ausserdem ein Matchmaking eingeführt welches versucht möglichst gleich starke Gegner zu finden.

Dokumentation

Gegen Ende vom Projekt haben wir noch Diagramme für die Dokumentation erstellt und das ganze für Docker vorbereitet.

Schwierigkeiten

Veraltete Libraries

Beim Projekt kamen einige Schwierigkeiten auf. Als erstes beim arbeiten an den Workshops. Viele Klassen der Dependencies die im Aufgabenteil erwähnt wurden waren bereits veraltet und konnten nicht verwendet werden. Aus diesem Grund musste man zuerst herausfinden wie die aktuellen Klassen heissen, die für unseren Zweck geeignet waren.

Das richtige Frontend-Framework

Beim auswählen des Frontends hatten wir ein bisschen die Qual der Wahl. Es gibt nämlich viele verschiedene Möglichkeiten um das Frontend umzusetzen. Weil allerdings so ziemlich alle Frontend-Frameworks für uns neu waren mussten wir uns in diese ziemlich lange einlesen bis wir herausgefunden haben ob sie geeignet sind für unsere Bedürfnisse.

Der Datenfluss über die Endpoints

Bei den Endpoints kam es häufig vor, dass wir die Falschen Rückgabewert erwartet haben und deshalb immer das fallback verwendet wurde. Es kam auch vor, dass der Link allgemein einfach falsch geschrieben wurde. Wenn Daten von einem Service zu einem anderen übertragen werden, werden diese über viele Klassen hinweg weitergeleitet. Wollte man einen Endpoint bearbeiten, hatte dies die Folge, dass man an vielen Stellen im Projekt eine Änderung durchführen musste.

Belegte Ports

Bei dem History-Service wollten wir wie bei den anderen Servicen den Port einfach um 1111 erhöhen. Damit wäre der neue Port 6666 geworden. Als der Service konfiguriert war konnte man sich allerdings immer noch nicht auf diesen Service verbinden. Nach einer kurzen google suche haben wir herausgefunden, dass der Port 6666 für TCP reserviert war, deshalb haben wir nun die Port Nummer auf 7777 gesetzt.

Schlussstand

Registry	1111	Stellt Eureka zur verfügung
Camp	2222	Erstellt Heroes und Parties
Arena	3333	Führt die Kämpfe durch
Promoter	4444	Teilt die Resultate von Kämpfen mit
Shop	5555	Stellt Items zur Verfügung
History	7777	Speichert den Verlauf von verschiedenen Kämpfen
Frontend	8080	Frontend welches den Usern zugriff zum Spiel gewährt

Der Shop-Service

Der Shop bietet eine Reihe von Items an, die es einem Camp ermöglichen seine Heroes zu verbessern oder zu heilen.

Das Frontend

Alle Use Cases werden durch unseren Frontend-Service und die darin enthaltenen HTML-Files und stylesheets grafisch dargestellt.

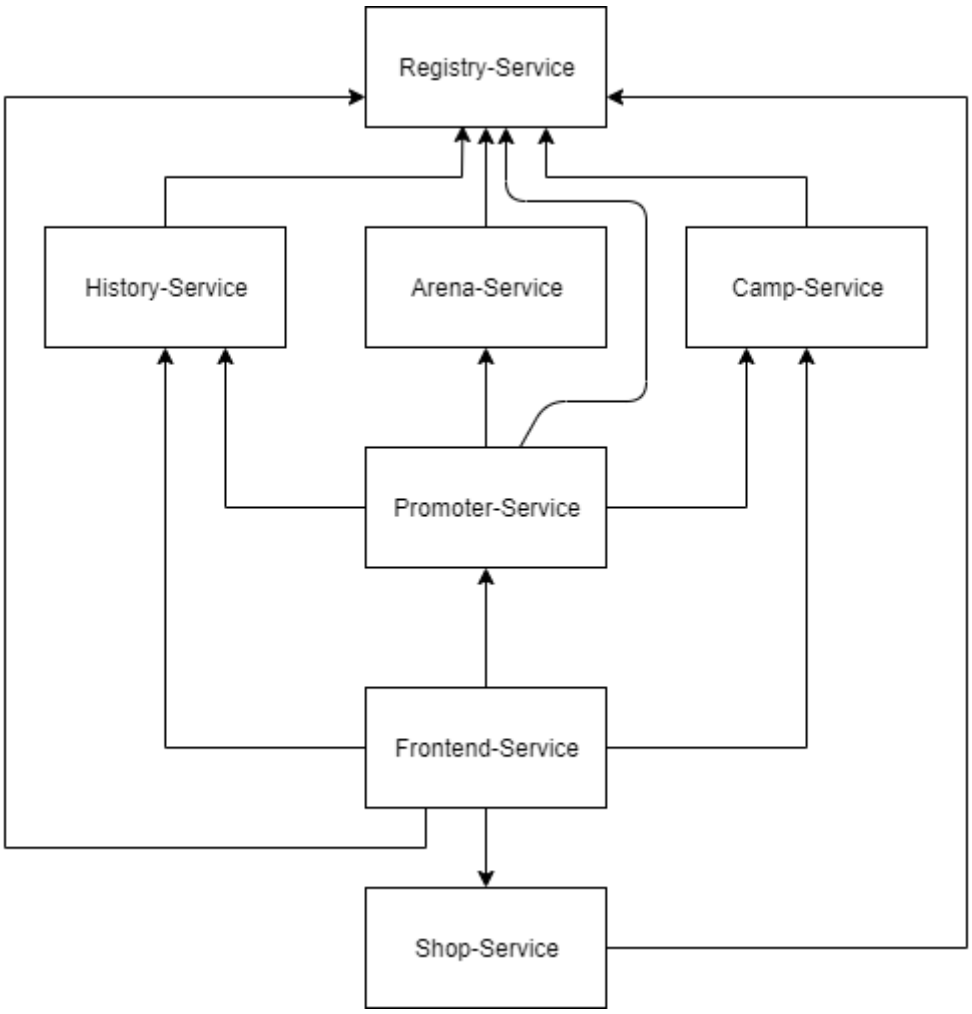
Das Währungssystem

Im Spiel wird mit Gold gehandelt. Items und neue Heros zu kaufen fordert einen bestimmten Betrag an Gold. Gewinnt eine Party einen Kampf in der Arena, so wird diese mit Gold belohnt.

Verwalten der Party

Es ist möglich die einzelne Party zu bearbeiten. Man kann neue Helden kaufen und dann die Party zusammenstellen wie man will. Man kann mithilfe von Ajax den Namen von Helden verändern. Und man kann die Werte der Helden dank dem Shop verbessern.

Diagramm Service Architektur



Installations Betriebsanleitung

Manuell ohne Docker

1. Das Projekt von unserem Github-Repository pullen (https://github.com/timmmmmb/heroes) und als Maven-Projekt importieren
2. Alle Dependencies laden: mvn clean install oder direkt über die IDE
3. Die Startklasse des Registry-Service starten
4. Die Startklassen der restlichen Services starten
5. Im Browser auf http://localhost:8080 navigieren