Konzeptpapier: Battleship

Jakob Götz
j.goetz@oth-aw.de

Uwe Kölbel
u.koelbel@oth-aw.de

Maximilian Schlosser
m.schlosser@oth-aw.de

Oliver Schmidts
o.schmidts@oth-aw.de

Jan Schuster
j.schuster@oth-aw.de

Philipp Seufert
p.seufert@oth-aw.de

Fabian Wagner
f.wagner@oth-aw.de

***Zusammenfassung*—Wir beschäftigen uns mit der Speicherung und Suche vernetzter Informationen von Videospielen. Dazu stellen wir *SGDb* vor – eine webbasierte Anwendung mit einer Graphen-basierten Suche von Videospielen.**

I. Einleitung

Spiele, die im Browser ausführbar sind, erfreuten sich während der Corona-Pandemie größter Beliebtheit. Die Menschen durften sich aufgrund der hohen Ansteckungsrate des Virus nicht in der Realität treffen.

Dies war der Grund, weshalb diese regelmäßigen Treffen häufig in die digitale Form wechselten. Es wurden Spielabende mit beispielsweise Skribbl.io [1] veranstaltet, da für die Teilnahme keine leistungsfähigen Rechner, sondern meist lediglich ein Smartphone oder Ähnliches benötigt wurde.

Das Brettspiel „Schiffe versenken!“ ist ein sehr beliebtes und intuitives Spiel, das seit Jahrzehnten Generationen von Spielern begeistert. Dieses Spiel bietet eine perfekte Mischung aus Strategie, Glück und Unterhaltung, die es zu einem zeitlosen Klassiker gemacht hat.

Aus diesem Grund wollen wir in dieser Modularbeit das Spiel im Browser implementieren, damit unabhängig von äußeren Faktoren wie beispielsweise der Corona-Pandemie, das Gesellschaftsspiel Battleship gespielt werden kann.

II. Verwandte Arbeiten

Im Jahr 2015 veröffentlichte Matheus Valadares ein simples Browserspiel namens Agar.io [2]. Das Spielprinzip ist einfach: Man steuert eine Zelle in einer Petrischale und versucht größer zu werden, indem man Agar-Pellets und kleinere Zellen verschluckt. Das Spiel erfreute sich großer Beliebtheit und die Zahl der monatlichen Spieler ist schnell in die Millionen gestiegen [3]. Um einen solch rapiden Anstieg an Spielern ohne Ausfälle oder Überlastungen bewältigen zu können, ist eine Cloud-native Architektur ideal. Dazu wird typischerweise die Anzahl der aktiven Server an die Anzahl der aktiven Spieler angepasst. Ein solcher Skalierungsmechanismus ist auch für unser Spiel vorgesehen.

Agar.io bietet mittlerweile eine Vielzahl von Features, die für Online-Spiele typisch sind. Dazu gehören verschiedene Spielmodi, eine Login-Funktion, Mikrotransaktionen und Skins. Es ist denkbar, jedes dieser Features auch in unser Spiel zu integrieren.

III. Anforderungen

In der Anforderungsanalyse wurden drei primäre Komponenten identifiziert, die es umzusetzen gilt: (1) Die Menüführung (2) das Gameplay und (3) das Dashboard.

A. Menüführung

Als Benutzer möchte ich ein intuitiv zu bedienendes Menü, in dem ich:

- meinen Benutzernamen eingeben kann,
- ein Spiel starten kann,
- eine Anleitung aufrufen kann und
- meinen Benutzernamen eingeben kann,
- ein Dashboard mit Statistiken aufrufen kann.

Als Benutzer möchte ich zwischen verschiedenen Spielmodi wählen können:

- Gegen Computer
- Gegen zufälligen Gegner
- Gegen Freund

B. Gameplay

Im Spiel möchte ich nach folgenden Regeln spielen:

- Zu Beginn muss jeder Spieler eine festgelegte Anzahl an Schiffen unterschiedlicher Größe platzieren.
- Die Spieler dürfen abwechselnd einen Schuss abfeuern.
- Sind alle Schiffe eines Spielers versenkt, hat dieser Spieler verloren.

Im Spiel möchte ich:

- meine Schiffe zufällig oder händisch auf dem Spielfeld platzieren können
- mit den Waffen meiner Schiffe feuern können
- meine Treffer und die verfehlten Schüsse auf dem Spielfeld erkennen können
- ein übersichtliches Spielfeld mit allen relevanten Informationen
- (optional) mit meinem Gegner kommunizieren können
- (optional) Spiel pausieren, später fortsetzen

Als Spieler möchte ich:

- jederzeit ein reibungsloses Spielerlebnis genießen
- unabhängig von der aktuellen Spielerzahl spielen können
- nicht lange auf einen Server warten
- in Echtzeit gegen andere spielen (ohne merkliche Verzögerung zwischen den Zügen)
- Akzeptanzkriterien: Skalierbarkeit, Cloud-Deployment, Monitoring

C. Leaderboard / Dashboard

Als Benutzer möchte ich ein Dashboard, das Informationen über vergangene Spiele darstellt. Denkbar sind:

- Eine Bestenliste
- Anzahl der benötigten Schüsse bis zum Sieg
- Beliebteste Schiffspositionen
- Gewinnrate gegen Computer

IV. Methoden

Das Projekt setzt sich aus drei Komponenten zusammen: Frontend, Backend und Datenbank. Die einzelnen Komponenten werden jeweils in einem Docker-Container ausgeliefert. Sie werden mit einem in der Cloud gehosteten Kubernetes-Cluster orchestriert und erreichbar gemacht.

Für die Visualisierung des Spielgeschehens im Frontend wird das Game Framework Phaser mit JavaScript verwendet. Ein Webserver stellt das Frontend für den Nutzer zur Verfügung. Das Backend wird in Python implementiert und stellt eine RESTful-API sowie die Spiellogik zur Verfügung. Diese wird mit Hilfe des Frameworks FastAPI realisiert und dient zur Kommunikation mit dem Frontend. Als Datenbank wird MongoDB verwendet. Die Spielverläufe werden als JSON-Objekte serialisiert und für spätere Analysen gespeichert.

Literatur

[1] *Skribbl.io*, https://skribbl.io/. (besucht am 02. 05. 2023).

[2] Miniclip SA, *Agar.io*, https : / / www . agar . io/, 2015. (besucht am 02. 05. 2023).

[3] D. Takahashi, *The surprising momentum behind games like Agar.io*, Feb. 2017. Adresse: https://venturebeat.com/games/the-surprising-momentum-behind-io-games-like-agar-io/ (besucht am 02. 05. 2023).