Scradles - Dokumentation der Softwarearchitektur

Aaron Kreuzer a.kreuzer@oth-aw.de
Amos Asmerom a.asmerom@oth-aw.de
Christoph Boni c.boni@oth-aw.de
Leonard Wöllmer l.woellmer@oth-aw.de
Adrian Rall a.rall@oth-aw.de
Tahata Djoumsi d.tahata@oth-aw.de

***Abstract*—Scradles ist eine online spielbare, crossplattformunabhängige Multiplayerversion von "Schere-Stein-Papier". Mehrere Spieler sollen dabei simultan, nicht rundenbasiert, mit der Plattform interagieren und gegeneinander antreten können. Eine Registrierung ist für den Nutzer optional; diese wird nur für den Eintrag in die globale Rangliste benötigt.**

I. Mission Statement

Onlinespiele wirken sich umso realer und realistischer auf die User Experience aus, wenn die Simultanität bei Interaktionen gegeben ist. Rundenbasierte Spiele können für taktische Spiele Übersicht und Kontinuität geben, für ein Gefühl realer Interaktivität ist diese Spielart im Onlinebereich aufgrund der entspehenden zeitlichen Verzögerungen jedoch unzureichend. Simultanität versprüht den Hauch von Nähe und realer Verständigung und Verbundenheit. Der Faktor Zeit spielt somit keine Rolle mehr, allein durch die räumliche Distanz unterscheiden sich simultan gespielte Onlinespiele von "real life"-Interaktionen. Ziel dieser Anwendung ist, Nutzern dieser Anwendung die Möglichkeit zu bieten, sich mit anderen Spielern online in Verbindung zu setzen und ihnen eine Plattform zu bieten, simultan und ohne erkennbare zeitlichen Interaktionsdifferenzen miteinander ein bereicherndes Erlebnis zu erschaffen.

II. Kontextabgrenzung

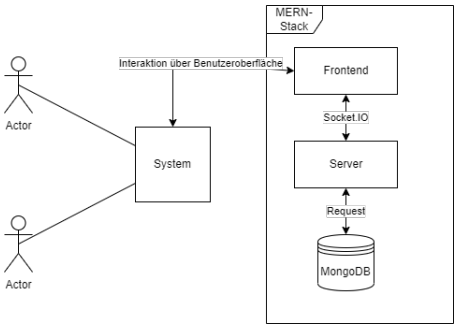

Fig. 1. Kontextabgrenzung

A. Rahmenbedingungen

Es wurde als Grundstruktur der MERN-Stack Framework verwendet. Dieses Framework erleichtert es, eine 3-Schichten-Architektur aufzubauen. Dabei ist "MERN" ein Akronym für die darin verbauten Technologien:
MongoDB **E**xpressJS **R**eact **N**ode.js

MongoDB wird als grundlegende Datenbank verwendet. Darin können registrierte Spieler gespeichert werden und darauf basierend eine Bestenliste mit Punktestand generiert werden. MongoDB gehört zu den sog. dokumentenbasierten NoSQL-Datenbanken. Ein großer Vorteil dieser Art von Abspeicherung und Strukturierung ist die einfache Verteilung der Arbeitslast auf mehrere Server.
Express (manchmal auch als Express.js bezeichnet) ist ein Back-End-Webanwendungsframework, das auf Node.js ausgeführt wird.
Node.js ist eine JavaScript-Laufzeitumgebung und ermöglicht die Implementierung von Javascript auf einem Server. Somit ist man in der Lage, einfach und unkompliziert Javascript außerhalb eines Browsers auszuführen und mit Datenbanken wie MongoDB zu verbinden.
React ist eine JavaScript-Bibliothek, geeignet für das Erstellen von Web-Benutzeroberflächen. Ein großer Vorteil liegt in der Generierung wiedervewendbarer Komponenten, die auf jeweiligen Webseiten eingebunden werden können. Für die Kommunikation zwischen den Schichten - Frontend und Server - also der Client-Server-Kommunikation, wird unter anderem Socket.IO verwendet. Diese Technologie erlaubt eine niedrige Latenzzeit und ermöglicht eine Skalierung auf mehrere Server und Senden von Ereignissen an alle verbundenen Clients.
Desweiteren wird "nginx" verwendet, eine Webserver-Software zur Lastverteilung (umfangreiche Anfragen an einen Serversystem), um die aufkommenden Anfragen auf mehrere Server zu verteilen. Dies hat eine effizientere Gesamtperformance zur Folge.
Es wird Socket.IO für die Kommunikation zwischen Server und Clients genutzt. Socket.IO ist ebenfalls eine JavaScript-Bibliothek mit großen Stärken in der Echtzeit-Webanwendungsnutzung aufgrund seiner bidirektionalen Signalübertragung.

III. Qualitätsziele

Es gibt eine Liste an Qualitätszielen, welche erreicht werden soll.

- Zuverlässigkeit
  - Das System soll immer in einem definierten Zustand verweilen.
  - Jeder Link soll dabei zu einer weiteren Webseite führen.
  - Alle Interaktionsartefakte (z.B. Buttons) sollen eine Funktionalität erfüllen. Interaktionsartefakte

oder Verlinkungen ohne Funktion sind nicht vorgesehen.
    * Damit wird verhindert, dass das System in einen undefinierten Zustand gerät und es zu einen Absturz der Anwendung kommt.
- Funktionalität
    * System soll zwei Clients verbinden können, sodass diese simultan an einen Spiel teilnehmen können.
    * Das System soll die Spiellogik von "Schere-Stein-Papier" anwenden können.
    * Das System soll den Nutzern Rückmeldung über den Spielstand geben.
- Benutzbarkeit
    * Durch einen minimalistischen Aufbau werden die Interaktionsmöglichkeiten mit der Oberfläche auf das Nötige beschränkt.
    * Dadurch wird das System schnell erlernbar und die Oberfläche erscheint intuitiv.
- Sicherheit
    * Nutzern wird nur eine geringe Interaktionsmöglichkeit geboten. Dies soll einen ausreichenden Zugriffs- und Veränderungsschutz von Daten ermöglichen.

.

IV. Übersichtsbild

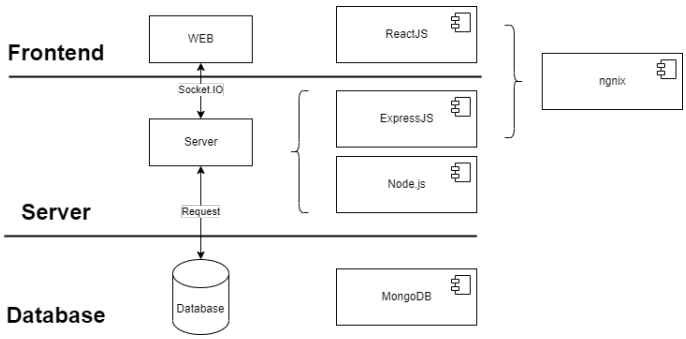

Fig. 2. MERN-Stack-Schichtenmodell

Das 3-Schichten-Archtitekturmodell sieht eine Untergliederung in Benutzeroberfläche (Web-Frontend), Applikation (Server) und Datenbanksystem (Backend) vor. Dabei werden wie bereits erwähnt, die Javascript-Bibliothek "React" für das Frontend, die Technologien "ExpressJS" und "Node.js" für die serverseitige Logikebene verwendet. Als Datenbank kommt MongoDB zum Einsatz.
Für die bidirektionale Kommunikation zwischen Client und Server wird Socket.IO verwendet.

Effiziente und gute Antwortzeiten	Mittels Lastverteilung durch nginx und Node.js sowie durch ihre Eigenschaften der guten Skalierbarkeit von Node.js und Socket.IO
Zuverlässigkeit und Funktionalität	Es wurden sich an verschiedene Programmierparadigmen gehalten und versucht, anzuwenden. Darunter: - "Separation of Concerns", - "Single Responsibility Principle" - "Don't repeat yourself" und - "Verständlichkeit"
Benutzbarkeit	Die 10 Heuristiken nach Jakob Nielsen [1] für ein nutzbares User Interface Design sind in die Anordnung und Strukturierung eingebracht worden. Ebenso ist die Norm ISO 9241-110 "Dialog [2] principles" mitbedacht worden.
Sicherheit	Es werden neben der geringen Eingabe- und Interaktionsmöglichkeit keine weiteren sicherheitsrelevanten (Verschlüsselungs-)Techniken eingesetzt

Desweiteren wird "nginx" für die Serverlastverteilung mit eingebunden. Nginx erlaubt durch seinen asynchronen Ansatz zur Verbindung mehrerer Request eine effizientere und geringe Speichernutzung und erhöht somit die Parallelisierung von Anfragen (Requests). Mithilfe der Docker-Software können alle benötigten Bibliotheken für alle Programmierer einheitlich und aktuell gehalten werden. Diese können dank der Software bereitgestellt und leicht verwaltet werden.

V. Lösungsstrategie

VI. Architekturprinzipien

- Modularität durch Components
("Separation of Concerns" +
- Verständlichkeit des Codes
("DRY" + "KISS")

Bei "DRY" (engl. für "Don't repeat yourself") ist ein Prinzip für eine gut strukturierte und verständliche Codekonvention. Ziel dieses Prinzips ist die Verminderung von Coderedundanz; also Vermeidung von Codereplikationen. Module sollen dabei einmal definiert und an geeigneter Stelle eingesetzt werden. Dies hat den Vorteil, dass Änderungen am Code nur einmal vorgenommen werden müssen und hat somit eine bessere Modularität zur Folge.
Dieses Prinzip geht Hand-in-Hand mit dem Prinzip "KISS" (engl. "Keep it short, simple", deutsch: "Halte es kurz und simpel"). Dieses Apronym beschreibt die Idee, Code möglichst auf seine wesentlichen Aufgaben zu reduzieren. In Kombination mit dem DRY-Prinzip bedeutet dies, eine Funktion oder ein Modul mit lediglich einer Aufgabe zu definieren und mehrere Module, wenn

nötig, ineinander zu verbauen.
Das Prinzip "Seperation of Concerns" findet im Bereich der Programmiersprachenaufteilung statt. Es wird JavaScript für Funktions- und Moduldefinitionen genutzt, die äußere Erscheinung wird jedoch mittels CSS bearbeitet. (vgl. Wikipedia, 2022 [5])

VII. MVP - MINIMUM VIABLE PRODUCT

Das "Minimum viable product", kurz MVP, sieht vor, dass zwei Spieler gegeneinander in Echtzeit gegeneinander antreten können. Dabei soll jeder Spieler sich für eine von drei nutzbaren Antrittszustände wählen können (Schere, Stein oder Papier). Je nach Wahl des Artefaktes soll der Spielstand für die Nutzer ausgegeben werden. Es können mehrere Runden gegeneinander gespielt werden. Nach einer gewissen Inaktivität, soll automatisch eine Beendigung des Spieles inkraft treten. Ist ein Spieler registriert und angemeldet, wird die Anzahl seiner gewonnen Spiele in einer globalen Rangliste vermerkt. Es ist ein Spiel ohne Anmeldung oder Registrierung möglich.

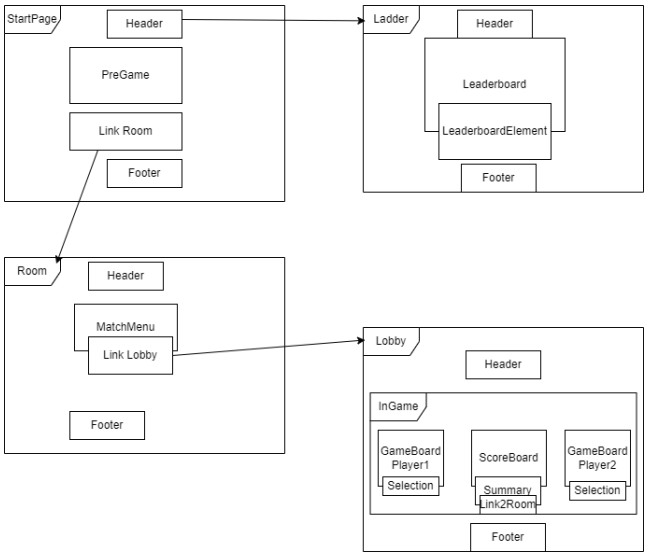

Fig. 3. Abbildung FrontEnd

REFERENCES

[1] Nielsen, Jakob. "usability heuristics. Nielsen Norman Group." (10).

[2] DIN, ENISO. Ergonomie der Mensch-System-Interaktion–Teil 110: Grundsätze der Dialoggestaltung (ISO 9241-110: 2006)[Ergonomics of human-system interaction–Part 110: Principles of dialogue design]. Deutsche Fassung EN ISO, S. 9241-110, 2006.

[3] S. Zörner.(2019, Dezember 09). Architektur ohne Firlefanz – Ihre Lösung auf einem Bierdeckel [online] Available: https://www.informatik-aktuell.de/entwicklung/methoden/architektur-ohne-firlefanz-ihre-loesung-auf-einem-bierdeckel.html

[4] MEHRA, Monika, et al. MERN stack Web Development. Annals of the Romanian Society for Cell Biology, 25. Jg., Nr. 6, S. 11756-11761, 2021.

[5] Wikipedia.(2022, Juni 17). Separation of concerns [online] Available: https://en.wikipedia.org/wiki/Separation of concerns

[6] Express Dokumentation (aufgerufen am 26.06.2022) Availabe: https://expressjs.com/de/4x/api.html

[7] MongooseJS Dokumentation. (aufgerufen am 26.06.2022) Available: https://mongoosejs.com/docs/guide.html

[8] ReactJS Dokumentation Available: https://reactjs.org/docs/getting-started.html

[9] MongoDB Dokumentation (aufgerufen am 26.06.2022) Available: https://www.mongodb.com/docs/

[10] Docker Dokumentation Available: https://docs.docker.com

[11] nginx Dokumentation Available: https://nginx.org/en/docs/