

Projektdokumentation Leiterspiel

Verfasserinnen:

Anna Staub

Neuwiesenstrasse 83

8400 Winterthur

E-Mail: anna.staub@stud.fhgr.ch

Morena Sager

Seonerstrasse 14

5600 Lenzburg

E-Mail: morena.sager@stud.fhgr.ch

Yara Wagner

Stampfibachstrasse 14

4663 Aarburg

E-Mail: yara.wagner@stud.fhgr.ch

Dozierende:

Elham Müller, Marc-Alexander Iten

Modul:

Frontend Development 1 und 2

Bearbeitungszeitraum:

von 13.09.2022 bis 31.05.2023

Zürich, 30.05.2023

Inhaltsverzeichnis

1. Projektidee	3
1.1. Ursprünglich geplant.....	3
1.2. Umsetzung im Projekt	3
2. Erläuterung des Source Codes	5
2.1. Dateistruktur	5
2.2. JavaScript.....	5
3. Aufbau des Repository und Arbeit mit Git	7
3.1. GitLab und branches	7
3.2. Arbeit mit der Versionsverwaltung.....	7
4. Reflexion	8
4.1. Persönliche Learnings	10
5. Ausblick.....	12
Eidesstattliche Erklärung	13

1. Projektidee

Wir haben das Ziel verfolgt, ein Leiterspiel zu programmieren. Das Spielfeld besteht aus hundert Feldern. Dabei ist das erste Feld gleichzeitig auch das Startfeld und das letzte Feld das Zielfeld. Im Spiel sind die nummerierten Spielfelder mit Leitern verbunden, die man je nachdem hochklettern kann, um den Weg abzukürzen, oder runterklettern muss, um somit einen Teil des Weges erneut zurückzulegen. Mit einem Würfel wird bestimmt, wie weit die Spielfigur fahren darf. Würfelt man die Zahl Sechs, darf man einen zusätzlichen Zug ausführen. Der Spieler¹, der als erstes das Zielfeld erreicht gewinnt.

1.1. Ursprünglich geplant

Für den Prototypen hatten wir uns vorgenommen, eine ganz klassische Version (wie oben beschrieben) zu programmieren. Diese sollte mit zwei Spielern, aber nur auf einem Gerät spielbar sein. Durch Klick auf das Würfelfeld wird eine Zahl zwischen 1 und 6 generiert. Die Figur des Spielers, der an der Reihe ist, wird anschliessend automatisch um die gewürfelte Anzahl Felder vorgerückt. Landet man auf einem Feld mit einer Leiter (bzw. anderem Element), wird die Figur auf das mit dem Element verbundene Feld verschoben (Verschiebung nur in eine Richtung, abhängig vom Element). Wird die Zahl Sechs gewürfelt, kann der Spieler erneut einen Zug machen (unbegrenzt oft nacheinander möglich). Nach Abschluss des Zuges ist die andere Person an der Reihe. Die Endversion des Spiels soll gleich funktionieren wie der Prototyp, neu soll jedoch auch von zwei verschiedenen Geräten aus miteinander gespielt werden können.

1.2. Umsetzung im Projekt

Bereits durch die administrative Umstrukturierung des Moduls und die daraus resultierenden Änderungen an den Anforderungen fürs Projekt, haben wir einige Änderungen an der finalen Version vorgenommen. Das Spiel ist nun nicht wie geplant an zwei Endgeräten spielbar. Stattdessen haben wir die freigewordenen Ressourcen darauf verwendet, das Spiel interaktiver zu gestalten. Anfängliche Unschönheiten wie beispielsweise, dass man das Zielfeld nicht genau treffen muss, sondern auch darüber hinausfahren kann, um zu gewinnen, haben wir überarbeitet. In der Endversion muss das Zielfeld genau getroffen werden, ansonsten wird für die restliche Anzahl Würfelaugen rückwärtsgefahren. Dadurch ist es nicht mehr gleich einfach zu gewinnen, da man beim Rückwärtsfahren auch noch nach unten fallen kann. Des Weiteren gibt es nun zwei verschiedene Würfel-Buttons: Einen gewöhnlichen Spielwürfel mit den Augenzahlen 1 bis 6 und einen Spezialwürfel, der Augenzahlen von -5 bis 10 (inklusive 0) ermöglicht. Man kann jederzeit auswählen, mit welchem Würfel man würfeln will. In der Mitte vom Spielfeld, auf Feld 55 befindet sich ein Tauschfeld. Landet ein Spieler darauf, erscheint ein Popup welches fragt, ob man den Platz mit der gegnerischen Spielfigur tauschen möchte. Bei Klick auf 'OK' tauschen die Figuren ihren Platz, bei Klick auf 'Abbrechen' bleiben die Figuren an ihrem bisherigen Platz. Neben diesen

¹ Da wir auch im Code jeweils den Begriff "Spieler" verwendet haben, halten wir das auch in der Dokumentation so. Selbstverständlich sind bei dem Begriff immer alle Geschlechter gemeint.

Erweiterungen haben wir auch das Design unseres Spiels überarbeitet. Neu gibt es eine Startseite, auf welcher die Spieler ihre Namen eingeben und eine Farbe für ihre Spielfigur wählen können. Je nach Einstellungen des Geräts, an welchem man das Spiel spielt, erscheint das Spiel entweder in einem hellen oder in einem dunklen Modus. Im Spiel selbst gelangen die Spielfiguren ausserdem anstatt über Leitern neu durch Röhren zu anderen Feldern. Auch das Abfangen von möglichen Fehlern haben wir verbessert. Auf der Startseite ist es nicht möglich, zweimal denselben Namen oder dieselbe Farbe auszuwählen. Tut man dies dennoch oder lässt eines der Felder leer, wird eine Fehlermeldung angezeigt. Um zum Spiel gelangen zu können, müssen korrekte Eingaben gemacht werden. Die Länge des Spielernamens ist ausserdem auf 10 Zeichen beschränkt, um Unschönheiten in der Darstellung zu vermeiden. Im Spiel selbst werden während eines Spielzugs jeweils die beiden Würfelbuttons deaktiviert und erst wieder nach Ende des Zugs aktiviert. Dies verhindert, dass vor Ende des aktuellen Spielzugs bereits ein neuer Spielzug gestartet werden kann, was wiederum zu einem Fehlverhalten der Spielfiguren führen würde.

2. Erläuterung des Source Codes

2.1. Dateistruktur

Das root Verzeichnis enthält die html-Dateien index.html und spiel.html, die README-Datei sowie die Ordner 'assets', 'documents' und 'mockups'.

Der Ordner 'documents' enthält einen Ordner mit allen Dateien zum Thema JavaScript Libraries sowie einen Ordner mit allen Projektmanagement-Dateien. Des Weiteren befinden sich darin die Dateien für die Dokumentation und Präsentation des Projektes, die ursprünglich formulierte Projektidee und die Schnittstellendefinition, welche wir im HS22 erstellt haben.

Der Ordner 'assets' enthält seinerseits die Unterordner 'css' und 'js', in welchen die jeweiligen Dateien abgelegt sind, sowie den Unterordner 'images', welcher alle Bilder enthält, welche wir für die Hintergründe der einzelnen Felder und der gesamten Seite verwendet haben.

Im Ordner 'mockups' befinden sich – wie der Name bereits sagt – alle Mockups, welche wir für das Design und die Funktionalitäten erstellt haben.

2.2. JavaScript

Um die Funktionalitäten zu gruppieren und den Code übersichtlicher zu halten haben wir nach dem Prinzip 'pro Klasse eine Datei' mehrere JavaScript-Dateien erstellt. Anbei noch eine Bemerkung zu den JavaScript-Dateien: Da wir uns erst am Schluss für das Design mit den Röhren als Transportelemente entschieden haben und die Grundidee das Leiterspiel war, wird im Code immer von Leiterfeldern gesprochen, diese Bezeichnung belassen wir so, weil wir auch das Spiel selbst weiterhin als Leiterspiel bezeichnen.

Feld.js

In dieser Datei werden die Klassen Feld und Leiterfeld konstruiert. Darin enthalten sind Methoden, um die DOM-Elemente der Felder mit den richtigen Klassen und IDs zu attribuieren. Ebenso wird die Konfiguration der Leiterfelder in einer Konstante festgehalten (dabei hat uns eine Arbeitskollegin von Yara geholfen).

Spielfeld.js

Mithilfe der in feld.js erstellten Klassen wird in der Datei Spielfeld.js. das Array der einzelnen Feld-Instanzen erstellt, welches zur Anzeige des Spielfelds im HTML benötigt wird. Je nach Fall werden so die Felder entsprechend ihrer Aufgabe korrekt instanziiert und attribuiert.

Spielfigur.js

In dieser Datei wird die Klasse Spielfigur definiert, das DOM-Element Spielfigur erstellt und attribuiert. Durch entsprechende Methoden kann die Position der Figur ermittelt und aktualisiert werden.

Startseite.js

Hier werden die nötigen Variablen erstellt, um die Werte Spielernamen und Spielfigurfarbe im SessionStorage zu speichern. Dies wird benötigt, um die Farb- und

Namensselektion an das Spiel zu übergeben und bei einem Seitenrefresh zu wieder hervorzuholen.

Spielfiguranzeige.js

Hier wird das DOM-Element für die Anzeige des aktuellen Spielers im HTML erstellt. Angezeigt werden der Name und die Farbe des aktuellen Spielers, welche auf der Startseite eingegeben wurden.

Storage.js

Storage.js enthält die Klasse StorageService und alle notwendigen Funktionen und Methoden, um den SessionStorage des Browsers mit dem aktuellen Spielstand zu befüllen und bei einem Seitenrefresh wieder auszulesen.

Wuerfel.js

Die Datei enthält die Klasse Wuerfel und alle zugehörigen Funktionen und Methoden. Darin wird ebenso die Anzeige der zuletzt gewürfelten Zahl und das Sperren der Würfelbuttons, während dem der Spielzug durchgeführt wird und das Entsperren der Buttons, wenn der Spielzug beendet wurde, ermöglicht.

Spiel.js

Hierbei handelt es sich um die komplexeste Datei. Einerseits wird hier das gesamte Spiel konstruiert und mit allen zugehörigen Elementen wie Spielfigur und Würfel instanziiert. Andererseits enthält es die Methoden der Spielzüge. In den Spielzügen wird auf alle möglichen Eventualitäten getestet (muss ein Leiterfeld benutzt werden, wurde eine 6 gewürfelt, hat der Spieler gewonnen, ...) und der entsprechende Code ausgeführt. Weiterhin gehört zu einem Spielzug auch das Errechnen des Feldes, welches der Spieler mit seinem Wurf erreicht und das Platzieren des DOM-Elements auf diesem, so dass die Spielzüge auch im HTML entsprechend angezeigt werden. Weiterhin sind die Methoden erhalten, welche das Landen auf dem Tauschfeld prüfen und das Durchführen des Tausches ermöglichen, sowie das korrekte Abspeichern und Auslesen des Spielstands, sodass dieser bei einem Seitenrefresh nicht verloren geht. Es bietet auch die Möglichkeit zu jedem Zeitpunkt ein neues Spiel, entweder mit den gleichen Spielern (Button Nochmal Spielen) oder mit neu gewählten Spielernamen und Farben (Button Neues Spiel, Umleitung zurück auf die Startseite), zu starten. Auch der debug_modus sowie die Testmodi zug54_modus und zug98_modus sind in dieser Datei enthalten. Mit dem debug_modus kann man Console-Logs de- und aktivieren. Der zug54_modus dient dem Testen des Tauschfeldes. Der Modus ermöglicht, dass man die Spielfigur auf Feld 54 setzen kann und der Würfel nur noch die Zahl Eins würfelt. Dadurch kann die Spielfigur auf eine einfache Weise zum Tauschfeld gelangen. Analog dazu funktioniert der zug98_modus zum Testen des Sieges.

3. Aufbau des Repository und Arbeit mit Git

3.1. GitLab und branches

URL zum Repository: <https://gitlab.com/yxaw/front-projekt>

Wir haben während des gesamten Projektes GitLab verwendet, um die Versionskontrolle für unsere Dateien zu gewährleisten. Dazu haben wir folgende branches erstellt:

- **main:** Hauptbranch, enthält nur funktionierende Zustände
- **develop:** Entwicklungsbranch, darauf haben wir primär gearbeitet, wir haben uns dagegen entschieden, für jede Entwicklerin einen eigenen branch zu erstellen und deshalb alle auf diesem branch gearbeitet
- **develop_with_api:** Stand des Entwicklungsbranches vor der Umstrukturierung des Moduls, enthält erste Versuche zur Einbindung der Serverkommunikation, Stand festgehalten, falls wir zu späterem Zeitpunkt die Serverkommunikation doch einbinden möchten
- **develop_with_dragndrop:** Stand des Entwicklungsbranches, enthält den Versuch die Spielfigur per Drag & Drop bewegen zu können, falls wir dieses Feature zu einem späteren Zeitpunkt noch fertigstellen möchten
- **abgabe-prototyp:** branch zur Abgabe des Prototyps am Ende des HS22. Hält den damaligen Entwicklungsstand fest.
- **abgabe-projekt:** branch zur Abgabe des gesamten Projektes am Ende des Moduls im FS23

Diese Infos und weitere Guidelines sind auch in der Datei Readme.md einsehbar.

3.2. Arbeit mit der Versionsverwaltung

Die Arbeit mit git war für uns alle neu und hat zu Beginn eine gewisse Gewöhnungsphase gebraucht. Vor allem auch die Verwaltung von MS Word und Excel Dateien in git erschien uns zu Beginn fremd, da wir nicht über die IDE (in unserem Fall Visual Studio Code) darauf zugreifen konnten und wir uns ausserdem eher die Verwendung von Tools wie Google Docs gewöhnt waren. So mussten wir immer sicherstellen, dass niemand sonst gerade die Word- oder Excel-Dateien bearbeitet, um merge-Konflikte, die unseres Wissens nicht klug zu bearbeiten gewesen wären, zu vermeiden. Bei den Code-files hingegen konnten wir sehr von git profitieren. Durch die fehlende Erfahrung mit git haben wir jedoch sicherlich nicht das volle Potenzial ausgeschöpft. Wir haben generell möglichst vermieden, zur gleichen Zeit im Projekt zu arbeiten, um Konflikte zu vermeiden. Hier hätten wir sicher mehr gewagt, wenn wir mit der Benutzung von git vertrauter gewesen wären.

Gearbeitet haben wir grösstenteils auf dem develop-branch. Nur vor den Abgaben am Ende der Semester haben wir den Stand in den main-branch gemerged. Um Zwischenstände wie der der Serverkommunikation oder des manuellen Spielzugs nicht zu verlieren, haben wir diese in separaten branches abgelegt.

Für uns ist es aber auf jeden Fall denkbar, in zukünftigen Projekten, sei es im Studium oder ausserhalb, wieder mit git zu arbeiten und dadurch die Funktionalitäten und Möglichkeiten noch besser kennenzulernen und ausschöpfen zu können.

4. Reflexion

Wir sind der Meinung, ein solides Spiel auf die Beine gestellt zu haben. Abgesehen von der Serverkommunikation bzw. dem gemeinsamen Spielen von zwei verschiedenen Geräten aus, konnten wir alle Ziele erreichen, die wir uns zu Beginn gesetzt haben. Da es im zweiten Semester noch einige Änderungen am Modul und den Anforderungen an die Projekte gab, änderten sich auch unsere Ziele nochmal. Die Serverkommunikation fiel weg, stattdessen kamen folgende neue Anforderungen hinzu:

- Eine Startseite, wo Spielernamen und Spielfiguren gewählt werden können.
- Ein Spezialfeld, welches dem Spieler, der darauf landet, die Option gibt, die Position der eigenen Spielfigur mit derjenigen des Gegners zu tauschen.
- Ein zweites Würfelfeld mit einem Spezialwürfel, welcher sowohl über höhere Augenzahlen aber auch über negative Augenzahlen verfügt. Die Spieler haben bei jedem Zug die Wahl zwischen den beiden Würfeln.
- Die eigene Spielfigur wird selbst auf das gewürfelte Feld gesetzt (entweder durch Anklicken des entsprechenden Feldes oder per Drag&Drop). Wird das falsche Feld gewählt so erscheint eine Fehlermeldung.
- Der Spielstand und die Spielernamen werden im Storage gespeichert, sodass die Seite auch nach einem refresh wieder auf dem gleichen Stand ist.

Von diesen neuen Zielen konnten wir auch wieder alle ausser eines wie geplant umsetzen können. Das Vorhaben, den Spielern zu ermöglichen ihre Figur selbst zu bewegen (entweder per Drag&Drop oder durch Anklicken des Landefelds), um das Spiel interaktiver zu machen, konnten wir leider nicht vollenden. Aus persönlichen Gründen kamen wir erst spät im Semester dazu, mit der Umsetzung dieses Features zu beginnen. Dabei stellte sich jedoch heraus, dass es nicht so einfach umsetzbar war. Die Implementierung hat zu Timing-Problemen geführt. Beim Versuch diese Probleme zu beheben, entstanden jedoch weitere Probleme. Da wir es ohne Hilfe nicht schafften, das Feature vollständig einzubauen und leider keine Zeit mehr für ein Coaching blieb, vereinbarten wir mit der Dozentin zu unserer grossen Erleichterung, das Feature wegzulassen. Den bisher umgesetzten Stand haben wir jedoch in einem separaten branch (`develop_with_dragndrop`) eingecheckt, damit zu einem späteren Zeitpunkt daran weitergearbeitet werden könnte. Ausserdem sind unsere Bemühungen so auch für die Dozentin sichtbar.

Beim Design, welches uns von anderen Online-Leiterspielen abheben sollte, hatten wir lange Mühe eine geeignete Lösung zu finden. Der Gedanke eines Space-Designs geisterte uns bereits von Beginn an durch den Kopf, aber wir schoben das Thema zugunsten der Funktionalitäten lange vor uns her. Es war uns ausserdem wichtig, dass das Spielfeld nicht zu vollgepackt wirkt, sondern die einzelnen Elemente stets gut sichtbar und verständlich sind. Dies konnten wir nun in einer Form umsetzen, welche uns gut gefällt. Das Design ist zwar immer noch eher schlicht, macht durch die verwendeten Hintergründe und das Farbkonzept aber einen ansprechenden und leicht Weltallmässigen Eindruck. Auch die unterschiedliche Darstellung bei dark bzw. light mode im Browser bereichert das Benutzererlebnis. Anstelle von Leitern haben wir uns gemeinsam dafür entschieden, ähnlich wie bei SuperMario Röhren zu verwenden, um dem Spiel nochmals eine etwas spannendere, weniger konservative Note zu geben.

In der Darstellung haben wir jedoch noch zwei Aspekte, welche uns stören, die zu beheben wir aber zeitlich nicht mehr in der Lage waren. Einerseits passiert es, dass bei kleineren, aber noch zugelassenen Bildschirmgrößen ein einzelnes Feld unschön vergrößert wird, wenn beide Spielfiguren gleichzeitig darauf stehen. Andererseits stehen die Spielfiguren noch unschön auf den Röhren. Doch auch das könnte man mit ein wenig mehr Zeit noch optimieren.

Gruppendynamisch gab es bei uns eher wenig Probleme, zumindest keine mit denen wir nicht umgehen konnten. Schon sehr früh wurde klar, dass Anna und Yara sich in der Programmierung sicherer fühlten als Morena. Aus diesem Grund hat Morena mehr administrative Arbeiten übernommen und mehr Zeit dazu aufgewendet, den von Yara und Anna geschriebenen Code zu verstehen, als selbst Codezeilen zu verfassen. Für Anna und Yara war das so in Ordnung. Wenn wir technische Probleme hatten oder nicht weiterwussten, haben wir uns externe Hilfe geholt, meistens direkt bei den Dozierenden. Die Coaching-Sessions haben uns sehr geholfen. Auch das gemeinsame Coden und Austauschen über den geschriebenen Code innerhalb der Gruppe hat uns weitergebracht.

Die Ordner wollten wir ursprünglich ein wenig anders strukturieren. Wir wollten von den html-Dateien eigentlich nur index.html im root-Verzeichnis, die restlichen sollten unter assets im Ordner html abgelegt werden. Doch da machte uns der Browser Firefox einen Strich durch die Rechnung. Nur wenn index.html und spiel.html im selben Ordner lagen, blieben die auf der Startseite eingegebenen Werte im SessionStorage gespeichert. Wenn spiel.html wie geplant in einem Unterordner abgelegt wurde, wurde der SessionStorage jeweils beim Wechsel von der Startseite zum Spiel in Firefox gelöscht, wodurch die Spielfiguren auf dem Spielfeld fehlten. Um das Problem zu lösen, verschoben wir die spiel.html-Datei deshalb wieder in das root-Verzeichnis zurück.

Am Code selbst könnte man bemängeln, dass die spiel.js Datei sehr umfangreich geraten ist. Es sind einige Redundanzen vorhanden, beispielsweise bei den Spielzug-Arten, und mit etwas mehr Zeit hätten wir den Code noch um einiges kürzen und bereinigen können. Ansonsten sind wir mit der Aufteilung des objektorientierten Codes in die verschiedenen Dateien zufrieden.

Für die Versionsverwaltung der Word- und Excel-Dateien wäre es vermutlich sinnvoller gewesen, weiterhin mit gewohnten Tools wie Google Docs zu arbeiten und erst fertige Stände im Repository abzulegen. So hätte man zwar nicht alles in einem System gehabt, die Änderungen an diesen Dateien wären jedoch immerhin in einem Änderungsverlauf sichtbar gewesen und es wäre möglich gewesen, gemeinsam an derselben Datei zu arbeiten. Auch die Grösse der commits sowie die commit-messages hätten wir besser lösen können. Wir tendierten dazu, verschiedene Änderungen in einem commit zu pushen. Dadurch waren auch die commit-messages eher ungenau. Das würden wir bei einem nächsten Projekt versuchen besser zu machen. Des Weiteren haben wir den main-branch zu wenig aktiv weitergeführt. Wir hätten regelmässiger die funktionierenden Zustände mergen sollen. Stattdessen haben wir den main bis zu den Abgaben vernachlässigt. Auch das würden wir in Zukunft anders handhaben.

Code reviews, wie sie in IT-Projekten gängig sind, haben wir nicht durchgeführt. Wir haben uns jedoch regelmässig ausgetauscht und auf den neusten Stand gebracht, meistens wurden die gemachten Änderungen auch noch kurz gemeinsam angeschaut

und erklärt, damit sie von allen verstanden wurden. Das Testen des Codes wurde fortlaufend von allen gemacht.

4.1. Persönliche Learnings

Morena

Wie bereits oben erwähnt, habe ich selbst eher wenig Sicherheit in der Programmierung von JavaScript erlangt. Die Struktur des Unterrichts hat für mich nicht gepasst, in den Theorieblöcken gab es zu wenig praktische Übungen, und die Praxisblöcke haben zu fortgeschritten begonnen und gaben zu wenig Zeit zum begleiteten Üben, respektive man hatte einfach zu wenig Zeit zum selbst Code schreiben und musste zu sehr der Lektion hinterher hetzen und hatte keine Zeit zum Nachvollziehen, wo ein Fehler passiert ist, wenn etwas nicht funktioniert hat. Im Selbststudium habe ich versucht, mir diese Kenntnisse anzueignen, hatte dabei aber wenig Erfolg, was wiederum zu einer verringerten Motivation geführt hat. Ich möchte aber damit nicht sagen, dass das Modul ein «Reinfall» war, im Gegenteil. Ich fand es äusserst interessant. In der Theorie habe ich sehr viel mitgenommen und ich denke die Logik hinter dem Code sehr gut verstanden zu haben und somit ein gutes Verständnis für mögliche Problemlösungen zu haben. Ich sehe es hier ein bisschen wie das Erlernen einer gesprochenen Sprache, ich verstehe was gesagt wird, kann aber selbst nur gebrochen sprechen. Nichtsdestotrotz war ich in der Lage, einige Dinge selbstständig im Code zu implementieren, so beispielsweise, dass die Spielfiguren in der korrekten, auf der Startseite ausgewählten Farbe angezeigt werden.

Yara

Der Start in die Programmierung im ersten Semester war ziemlich holprig. Gleich wie bei Morena hatte ich das Gefühl, die Theorie verstanden zu haben, bei der selbständigen Umsetzung hatte ich jedoch Probleme. Durch die Repetition, das gemeinsame Coden in der Gruppe, die Coaching-Sessions mit den Dozierenden und das selbständige Ausprobieren habe ich jedoch sehr viel dazugelernt und immer besser verstanden, wie der Code geschrieben werden musste, um das gewünschte Ergebnis zu erhalten. Ab Mitte des zweiten Semesters fühlte ich mich schon viel sicherer und hatte Freude am Schreiben des Codes, was unter anderem an den immer öfter auftretenden Erfolgserlebnissen lag. Auch bei der Arbeit mit GitLab habe ich einiges dazugelernt und werde das Tool bestimmt auch in Zukunft nutzen.

Anna

Das gesamte Projekt hat mir sehr viel Freude bereitet – auch sehr viel Frust, aber die Freude überwiegt definitiv. Wären da nicht noch andere Module parallel gelaufen so hätte ich gerne mehr Zeit in das Projekt gesteckt. Die objektorientierte Programmierung leuchtete mir von Beginn weg ein, und da ich sehr gerne neue Sprachen lerne, machte mir auch das Lernen von JavaScript grossen Spass. Ausserdem glaube ich so langsam begriffen zu haben, wie viel man mit HTML, CSS und JavaScript eigentlich erschaffen kann, und das ist doch schon ziemlich cool. Gewisse Konzepte wie z. B. die Serverkommunikation bereiteten mir etwas Mühe, besonders in der Anwendung. Gerade hier haben mir, genau wie Yara, das gemeinsame Coden in der Gruppe sowie die Coaching Sessions jeweils noch einmal zu einem besseren Verständnis und einigen Aha-Momenten verholfen. In der Arbeit mit GitLab habe ich viel dazugelernt,

gewisse Git-Befehle sind mir zwar noch immer ein Rätsel, aber damit werde ich mich sicher noch öfters auseinandersetzen. Gegen Ende des zweiten Semesters staute sich bei mir persönlich noch einiges an, meine Teamkolleginnen nahmen mir da einen Grossteil der Arbeit ab, dafür bin ich sehr dankbar. Zum Schluss habe ich für das Design dann aber noch einen Sprint in Angriff genommen, und konnte das Projekt so doch noch aktiv und begeistert abschliessen.

5. Ausblick

Um das Spiel noch weiter auszubauen, gibt es diverse Features, welche in Zukunft noch eingebaut werden könnten. Dazu gehört folgendes:

- Die Umsetzung der Serverkommunikation, damit das Spiel von zwei Geräten aus gespielt werden könnte.
- Die Fertigstellung des begonnenen manuellen Spielzugs, damit die Spielfiguren per Drag&Drop oder durch Anklicken des Feldes bewegt werden können.
- Das Erweitern der möglichen Spielerzahl, sodass mehr als nur zwei Personen miteinander spielen können.
- Das Einbauen weiterer Spezialfelder und Zusatzoptionen sowie verschiedener Spielmodi, welche das Spiel spannender gestalten.

Die Ideen gehen uns besonders bei den Spezialfeldern, Zusatzoptionen und Spielmodi nicht aus und evtl. werden wir nach Semesterende sogar noch welche davon umsetzen.

Eidesstattliche Erklärung

Wir erklären hiermit, dass wir diese Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und erlaubten Hilfsmittel benutzt haben. Alle Stellen, die wörtlich oder sinngemäss aus Quellen entnommen worden sind, haben wir als solche gekennzeichnet. Uns ist bekannt, dass andernfalls die Hochschulleitung zum Entzug der aufgrund unserer Arbeit verliehenen Qualifikation oder der für unsere Arbeit verliehenen Titel berechtigt ist.

Lenzburg, 30.05.2023

Ort, Datum



Unterschrift

Morena Sager

Winterthur, 30.05.2023

Ort, Datum

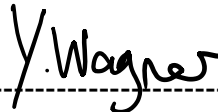


Unterschrift

Anna Staub

Aarburg, 30.05.2023

Ort, Datum



Unterschrift

Yara Wagner