

Maturarbeit
Documentation

Game

2022



Version: 1.0

Date: 2022-08-22 17:20:51+02:00

Team: Marc Honegger
Elia Schürpf

Begleitperson: Martin Hunziker

Gymnasium
KZO - Kantonsschule Zürcher Oberland

Inhaltsverzeichnis

I	Ziel	1
1	Vision	2
2	Anforderungen	3
2.1	KZO	3
2.2	Unsere	3
3	Skizzen	9
4	Risiko	10
4.1	Hoch	10
4.2	Mässig	10
4.3	Tief	11
II	Produkt	12
5	Architektur	15
6	Feedback	16
7	Zukunft	17
III	Schwierigkeiten	18
8	Technologien	19
8.1	Unity	19
8.2	Github	19
8.3	LaTeX	19
8.4	Proposed Realization	20
9	Team Management	21

10 Time Management	22
10.1 Roadmap	22
 IV Reflexion	 23
11 Team	24
12 Elia	25
13 Marc	26
Glossar	27
Bibliography	29

Teil I

Ziel

Kapitel 1

Vision

Ein funktionsfähiges Videospiel, welches jede vergleichbare Maturaarbeit in den Schatten stellt. Die Ideen sind nicht neu, sondern waren schon vor Jahren in Marc's Kopf. Jedoch war für die Umsetzung ein Team notwendig. Uns war dennoch von Anfang an bewusst, dass wir auch zuzweit nicht alle Funktionen hinkriegen werden, dennoch haben wir uns ein hohes Ziel gesetzt und für die Betreuung uns für Martin Hunziker entschieden. Vieles konnten wir schlussendlich auch umsetzen. Das grösste Wagnis war der Multiplayer und das haben wir dann auch an eigenem Leib erfahren. Alleine für den Multiplayer wurden über 60h investiert. Alles in allem kam ein funktionsfähiges und lustiges Spiel heraus. Einige Kommentare von Testern:

'sick game' - Marc Honegger
'absolut krass' - Eli Schürpf
'Masterpiece' - Martin Hunziker

Wie das Spiel jetzt aussieht, aber auch wie es in Zukunft damit weiter geht, ist in dieser Arbeit beschrieben.

Kapitel 2

Anforderungen

2.1 KZO

Die Anforderungen der Schule halten sich in Grenzen. Zum einen gibt es nicht fast keine genauen Anforderungen und die restlichen sind sehr tief gehalten. Die drei Anforderungen der KZO sind:

1. **Zeit**

Geplant ist ungefähr eine Lektion pro Woche, denn diese wird im Stundenplan für das Semester abgezogen. jedoch ist diese Anforderung bei uns nicht relevant, denn unser Projekt ist eine riesen Maturaarbeit. Wir werden ohne Problem auf diese Zeitanforderung erreichen und auch bei weitem überschreiten.

2. **Begleitperson**

Jede Maturaarbeit braucht eine Begleitperson. Also eine Lehrperson, welche die Arbeit bewertet und einen Unterstützt. Letzteres, also Unterstützung, war bei uns sehr schwierig, denn keine Lehrperson kennt sich mit Unity und C# gut aus. Wir sind schon früh auf Herr Kernn, ein Informatik Lehrer, zugegangen. Er hat uns an Martin Hunziker, Leiter der IT, weitergeleitet. Er nahm uns sehr gern auf, mit der Anmerkung, dass unsere Arbeit sehr schwierig sein wird, umzusetzen.

3. **Plagiat**

Arbeiten dürfen nicht einfach kopiert werden. Wir haben zwar gewisse Code Stücke aus dem Internet kopiert, jedoch sind diese keine Plagiate. Der grösste Teil unseres Codes und alles unserer Schriftlichen Arbeit sind von uns geschrieben.

2.2 Unsere

Gewisse Features sind wichtiger als andere für das funktionieren des Spiels. Zum Beispiel sind Einstellungen, Truppen und Multiplayer notwendiger als die Monetarisierung, zumindest nach uns. Deshalb ist dieser Abschnitt in 4 Gruppen unterteilt:

1. **Notwendig:** Ohne diese Features läuft das Spiel nicht oder ist sehr mangelhaft, Fehler erfüllt und unspielbar. Ohne ist Idee des Spieles nicht erkennbar.
2. **Anvisiert:** Diese Features sind unser Ziel. Sie wurden Herr Hunziker, also der Begleitperson, angekündigt und sind alle in einem Plan festgehalten. Diese Ziele sollen dazu führen, dass das Spiel gut spielbar ist und es Spass macht. Sie sind auch essentiell, das Spiel nicht zu monoton zu gestalten und sollten alle bis zum Abgabe Termin der schriftlichen Arbeit vollendet sein.
3. **Möglichkeiten:** Möglichkeiten das Spiel noch auf die Spitze zu bringen. Keine dieser Anforderungen ist nötig für das spielen, jedoch können sie das Spielerlebnis spannender und ausgereifter machen. Diese Ziele sind entweder in den Sternen geschrieben oder zu erreichen bis zur mündlichen Präsentation. Vereinzelt können diese, wie die anvisierten Anforderungen, bis zur schriftlichen Abgabe erreicht werden.
4. **Verworfen:** Weitere Ideen, welche wir hatten, jedoch als unmöglich oder nicht sinnvoll errachten.

Notwendig

- **System**

Unser Spiel sollte auf den meisten modernen Geräten funktionieren. Es soll auf macOS und Windows fließend gespielt werden können. Also mit mindestens 60FPS.

- **Benutzeroberfläche**

Eine Startseite mit einem "Spielen"-Knopf, einem "Einstellungen"-Knopf, einem "Credits"-Knopf und einem "Verlassen"-Knopf. Sie soll intuitiv sein und einigermaßen schön aussehen. Die Benutzeroberfläche soll mit Touch bedienbar sein, andere Eingabemöglichkeiten werden nicht unterstützt.

- **Lokaler Multiplayer**

Das Spiel soll im lokalen Netz gespielt werden können. So zum Beispiel mit Freunden oder Familie. Es sollte mithilfe der IP-Adresse innerhalb des gleichen Netztes zusammengespielt werden können.

- **Einstellungen**

Auflösung, Vollbild und Lautstärke müssen eingestellt werden können.

- **Kamera**

Die Kamera ist beweglich und das ganze Schlachtfeld ist ersichtbar.

- **Truppen**

Es braucht Truppen, die für einen kämpfen können. Diese sind notwendig für den Verlauf des Spieles und ohne sie hat das Spiel keinen Sinn.

- **Gewinnmöglichkeit**

Eine Chance das Spiel zu gewinnen oder verlieren und es somit zu beenden. Dies kann sehr simpel mit einer Linie vollendet werden, welche beim überschreiten das Spiel beendet.

- **Lanes**

Unser Spiel ist zwar 2D, aber hat dennoch eine Tiefe.

Anvisiert

- **Helden**

Beide Spieler haben eine Art Truppe, welche ihre Sigelinie beschützt. Sie haben einen Rache Effekt, welcher die Lane ausradiert, damit das Spiel nicht sofort vorbei ist.

- **Design**

Das Spiel sollte vom Aussehen her was hergeben. Es sollte nicht wie ein Prototyp, sondern wie ein vollendetes Spiel aussehen. Zu Beginn

- **Deck**

Spieler könne ihr eigenes Deck aus einer Auswahl von Karten zusammenstellen. Im Spiel werden davon per Zufall Karten gezogen.

- **Bot**

Ein sehr simpler Algorithmus um alleine gegen den Computer zu spielen auf den Schwierigkeitsstufen '*Einfach*', '*Mittel*', '*Schwierig*'. Es sollen rein zufällig Truppen geschickt werden, bei höherer Schwierigkeit mehr.

- **Truppen**

- **Nahkampf** Eine Truppe mit wenig Reichweite.
- **Fernkampf** Eine Truppe die auf Reichweite angreift. Sie schießt Projektile, zum Beispiel ein Bogenschütze, der mit Pfeilen schießt.
- **Suizid** Eine Truppe die bei Berührung mit einem Gegner stirbt und einen Effekt auslöst.

- **Effekte**

- **Gift:** Zeitlich limitierter und wiederholender Schadenseffekt. Eine visuelle Markierung soll vorhanden sein und das gleiche Gift, also der gleichen Truppe, soll nur einmal auf jemanden sein.
- **Dorn:** Truppen, welche Charaktere mit Dorn attackieren, erhalten Schaden. Soll nach Auswahl auf Nahkämpfer, Fernkämpfer oder beides wirken.
- **Rache:** Truppen mit Rache haben einen Effekt nach ihrem Tod. Zum Beispiel eine Truppe beschwören oder Schaden verursachen.

Möglichkeiten

- **Zauber**

Als Alternative sollen nicht alle Karten einfach eine Truppe herbeirufen. Gewisse Karten sollten nur einen Effekt auslösen. Beispiele für Zauber: "Heile deine Helden um 5 Leben", "Gib deinen Helden 5 Rüstung", "Verursache allen gegnerischen Truppen 5 Schaden", "Ziehe 3 Karten"

- **Monetarisierung**

Hierfür sind uns drei Möglichkeiten eingefallen, hier jeweils die Vor- und Nachteile:

1. **Kaufbare Gegenstände**

- + Vielseitige und Nachhaltige Monetarisierung. Neue Features, bedeuten auch neue Geldmöglichkeit. Die Inhalte sollten auch erspielbar sein, somit kann bezahlen zwar zu Vorteilen führen, jedoch mit viel Spielen erreichbar sein.
- Pay-to-Win

2. **Skins**

- + Weit verbreitet und sehr beliebt bei Spielern. Gibt keinem Spieler Vorteile
- Wenig Nutzen und sehr aufwändig. Neue Designs zu erstellen, wäre bei uns nicht so sinnvoll. Wir haben noch sehr viel Features zu implementieren und sind nicht Designer. Für uns braucht es sehr viel Zeit eine brauchbare Darstellung zu erstellen und der Mehrwert, welcher dem Spiel beigetragen wird, ist marginal.

3. **Kostenpflichtiges Spiel**

- + Einmalige Bezahlung, was für viele Nutzer lukrativer ist. Jedoch gilt dies vor allem bei Singleplayer Spielen.
- Wir nehmen an, dass wenige Leute bereit wären, Geld für unser Spiel zu bezahlen. Dafür wird es nicht genug ausgereift sein. Auch ist diese Methodik nicht nachhaltig und führt nur zu einer einmaligen Geldspritze. Viele grössere Spiele führen deshalb später DLCs ein, um das Spiel zu erweitern. Jedoch ist dies bei einem Multiplayer Spiel Pay-to-Win. Abschliessen müssten wir höchstwahrscheinlich selbst Geld vorauswerfen, um unser Spiel anbieten zu können, z.B. auf Steam.

- **Online Multiplayer**

Besser als Multiplayer limitiert auf das selbe Netz, ist Multiplayer, der weltweit verfügbar ist. Jedoch ist von einem Computer auf einen anderen zu verbinden dank Firewall von Router und Computer, nahezu umöglich. Dies ist einer der Gründe, weshalb ein Server sehr praktisch ist. Mit diesem ist dieses Problem gelöst. Server sind aber teuer und aufwändig. Wir müssten das ganze Multiplayer System unseres Spieles anpassen.

- **Helden**

Eine Auswahl von Helden, mit unterschiedlichen Schaden, Leben und Effekten, würde das Spiel nochmals spannender machen. Auch sollten gewisse Truppen auf bestimmte Helden limitiert sein.

- **Design**

Wenn die Zeit vorhanden ist, kann das Design immer verbessert werden. Hier gehts es aber um den Feinschliff, z.B. mehr Hintergründe, und mehr Dinge selbst zu designen.

- **Tutorial**

Eine Anleitung für Anfänger wäre sehr schön. Sie soll neuen Spielern das Anfangen erleichtern. Es sollte aber auch überspringbar sein, damit alte Spieler, es nicht nochmals spielen müssen. Es soll nicht lange sein und nicht schwierig. Dennoch soll es alle Mechaniken des Spiels erklären, am Besten Anhand von Gameplay.

- **Truppen** Weitere Truppen können jederzeit erstellt werden. Hier ist kein Limit gesetzt. Leben, Schaden, Darstellung, Effekt und vieles mehr kann angepasst werden.

- **Effekte**

- **Wiederbelebung:** Die Truppe wird wiederbelebt, sofort oder mit einer Verzögerung am Startpunkt.
- **Rüstung:** Die Truppe hat zusätzlich zu den Leben Rüstung. Die Rüstung wird zuerst abgezogen und hat im Gegensatz zum Leben keine Limite.
- **Aura:** Die Truppe fügt gegenrassen Truppen in einem gewissen Radius permanent Schaden zu.

- **Mehrsprachig**

Das Spiel soll in Englisch, Deutsch und Französisch spielbar sein.

Verworfen

- **Online Multiplayer**

Besser als Multiplayer limitiert auf das selbe Netz, ist Multiplayer, der weltweit verfügbar ist. Jedoch ist von einem Computer auf einen anderen zu verbinden dank Firewall von Router und Computer, nahezu unmöglich. Dies ist einer der Gründe, weshalb ein Server sehr praktisch ist. Mit diesem ist dieses Problem gelöst. Server sind aber teuer und aufwändig. Wir müssten das ganze Multiplayer System unseres Spieles anpassen.

- **Shop**

Nicht alle Karten sind von Beginn an freigeben, sondern müssen freigespielt werden. So kann zum Beispiel eine Ingame Währung erspielt werden und damit im Shop Karten oder sonstige Dinge gekauft werden. Der Shop kann mit Zahlungsmethoden

ausgestattet werden und so zur Monetariesierung beitragen. Jedoch ist ein Shop ohne Anti-Cheat und/oder Server die perfekte Angriffsfläche für Cheater und hat somit nicht viel Sinn.

- **Kampagne**

Singleplayer gegen bestimmte vorprogrammierte Gegner. Sie sollen immer schwieriger werden und bestimmte Herausforderungen mit sich bringen. Bei Vollendung werden Belohnungen verteilt.

- **Anti-Cheat**

Dies ist für uns ohne Erfahrung in diesem Bereich und einem lokal berechneten Spiel ein Ding der Unmöglichkeit. Wir haben keine Erfahrung in diesem Bereich und es ist ein extrem komplexes Thema.

- **Errungenschaften**

Eine Übersicht und die Möglichkeit die Errungenschaften zu erfüllen. Gegebenenfalls Belohnungen verteilen, wie zum Beispiel bestimmte Karten freischalten. Diese Erweiterung ist keines Falls notwendig und ein absolutes nice-to-have feature.

Kapitel 3

Skizzen

Kapitel 4

Risiko

4.1 Hoch

1. Beide Teammitglieder haben keine Ahnung von Multiplayer (Wahrscheinlich und Kritisch)
 - (a) Reduktionsstrategie: Andere Spielideen erarbeiten, die ohne Multiplayer funktionieren.
 - (b) Reduziertes Risiko: Hoch

4.2 Mässig

1. Inaccurate estimations (likely and critical)
 - (a) Mitigation strategies: apply cone of uncertainty, apply definition of ready to ensure planning quality
 - (b) Mitigated risk: Low
2. Poor risk management (likely and critical)
 - (a) Mitigation strategies: likelihood calculation, risk mitigation plans and monitoring of risks every planning
 - (b) Mitigated risk: Low
3. Project reviewer's expectations are not aligned with project (possible and critical)
 - (a) Mitigation strategies: obtain frequent approval and acknowledgement (naturally happens for us with review meetings)
 - (b) Mitigated risk: None
4. Unexpected absence of team member (unlikely and catastrophic)

- (a) Mitigation strategies: Code changes need to be pushed on a daily basis, stories could at any point be taken over by another team member
- (b) Mitigated risk: Medium

4.3 Tief

1. Insufficient code quality (possible and marginal)
 - (a) Mitigation strategies: code reviews, clear coding standards, apply definition of done
 - (b) Mitigated risk: None
2. Lack of ownership (possible and marginal)
 - (a) Mitigation strategies: setting clear responsibilities for roles
 - (b) Mitigated risk: None
3. Losing sight of documentation tasks (possible and marginal)
 - (a) Mitigation strategies: documentation strategy, documentation part of definition of done
 - (b) Mitigated risk: Low
4. Failure of hardware like personal devices, OST GitLab, Jira, hosted environment (rare, catastrophic)
 - (a) Mitigation strategies: Code changes need to be pushed on a daily basis
 - (b) Mitigated risk: Low

Teil II

Produkt

Project name: JassTracker

Team Members

1. Pascal Honegger (pascal.honegger1@ost.ch)
2. Marcel Joss (marcel.joss@ost.ch)
3. David Kalchofner (david.kalchofner@ost.ch)
4. Jamie Maier (jamie.maier@ost.ch)

Availabilities

Time slot	Mon	Tue	Wed	Thu	Fri	Sat
08h00-09h00	(XO)	-	-	-	(XO)	-
09h00-10h00	(XO)	-	-	-	(XO)	XO
10h00-11h00	(XO)	-	-	-	(XO)	XO
11h00-12h00	(XO)	-	-	-	(XO)	XO
12h00-13h00	(XO)	(XR)	(XR)	(XO)	(XO)	XO
13h00-14h00	(XO)	XR	-	(XO)	(XO)	XO
14h00-15h00	(XO)	XR	-	(XO)	(XO)	XO
15h00-16h00	(XO)	-	-	(XO)	(XO)	XO
16h00-17h00	(XO)	-	-	(XO)	(XO)	XO
17h00-18h00	-	-	-	(XO)	(XO)	-
18h00-19h00	-	-	-	(XO)	(XO)	-

Project Idea

JassTracker is a web app which allows tracking and analysis of the popular Swiss card game “Jass”. There are many forms of playing, but the one we’re focusing on is called “Coiffeur”. The two teams have two players each and need to keep track of what they’ve already played and which options are available to them. They also track whose turn it is, apply the correct multiplication to the score and sum it all up in the end. To work around this, a project team member is currently using a excel spreadsheet, but this solution provides limited functionality and has many drawbacks.

To make the scoring easier JassTracker allows players to easily track, analyze and sync games digitally. In a first step you will be able to create and arrange team members for a given game. Then you start playing the game physically and assign scored rounds to the correct member. During this phase you’ll also be able to see live stats such as average score by player so far. After the game some highlights (e.g. the best player) are highlighted. You can also gain exhaustive insight into your play style by looking at personal or group statistics such as average score by player or historic averages. To enable this, other physical members are able to associate their game to their personal account to track personal statistics across games.

Some potential ideas to expand on: configurable Jass game (e.g. Coiffeur with 8 instead of 10 options), prediction of scores based on past performance, current game

trajectory (win probability by team).

Proposed Realization

We plan on implementing a web app using Vue.js as a frontend library. For styling we plan on using Bootstrap for basic styles. The server will be implemented in Kotlin using the Ktor framework. Persistent data is stored in a PostgreSQL database and accessed using jOOQ. Development will be done locally in IntelliJ IDEA, production deployments will be using Docker containers. CI / CD will be implemented using the OST GitLab.

Kapitel 5

Architektur

Kapitel 6

Feedback

Time tracking is done exclusively in Jira. Detailed reports with hours spent per issue, epic and sprint can be found there. The following charts focus on the overall progress to notice early trends, in case certain members invest significantly more or less time than expected

Who	Logged Hours	Expected Hours
Pascal	135h	120h
Marcel	115h	120h
David	111h	120h
Jamie	106h	120h
Team	468h	480h

Kapitel 7

Zukunft

Teil III

Schwierigkeiten

Kapitel 8

Technologien

8.1 Unity

1. Pascal Honegger (pascal.honegger1@ost.ch)
2. Marcel Joss (marcel.joss@ost.ch)
3. David Kalchofner (david.kalchofner@ost.ch)
4. Jamie Maier (jamie.maier@ost.ch)

8.2 Github

Time slot	Mon	Tue	Wed	Thu	Fri	Sat
08h00-09h00	(XO)	-	-	-	(XO)	-
09h00-10h00	(XO)	-	-	-	(XO)	XO
10h00-11h00	(XO)	-	-	-	(XO)	XO
11h00-12h00	(XO)	-	-	-	(XO)	XO
12h00-13h00	(XO)	(XR)	(XR)	(XO)	(XO)	XO
13h00-14h00	(XO)	XR	-	(XO)	(XO)	XO
14h00-15h00	(XO)	XR	-	(XO)	(XO)	XO
15h00-16h00	(XO)	-	-	(XO)	(XO)	XO
16h00-17h00	(XO)	-	-	(XO)	(XO)	XO
17h00-18h00	-	-	-	(XO)	(XO)	-
18h00-19h00	-	-	-	(XO)	(XO)	-

8.3 LaTeX

JassTracker is a web app which allows tracking and analysis of the popular Swiss card game “Jass”. There are many forms of playing, but the one we’re focusing on is called “Coiffeur”. The two teams have two players each and need to keep track of what they’ve already played and which options are available to them. They also track whose turn it is, apply the correct multiplication to the score and sum it all up in the end. To work

around this, a project team member is currently using a excel spreadsheet, but this solution provides limited functionality and has many drawbacks.

To make the scoring easier JassTracker allows players to easily track, analyze and sync games digitally. In a first step you will be able to create and arrange team members for a given game. Then you start playing the game physically and assign scored rounds to the correct member. During this phase you'll also be able to see live stats such as average score by player so far. After the game some highlights (e.g. the best player) are highlighted. You can also gain exhaustive insight into your play style by looking at personal or group statistics such as average score by player or historic averages. To enable this, other physical members are able to associate their game to their personal account to track personal statistics across games.

Some potential ideas to expand on: configurable Jass game (e.g. Coiffeur with 8 instead of 10 options), prediction of scores based on past performance, current game trajectory (win probability by team).

8.4 Proposed Realization

We plan on implementing a web app using Vue.js as a frontend library. For styling we plan on using Bootstrap for basic styles. The server will be implemented in Kotlin using the Ktor framework. Persistent data is stored in a PostgreSQL database and accessed using jOOQ. Development will be done locally in IntelliJ IDEA, production deployments will be using Docker containers. CI / CD will be implemented using the OST GitLab.

Kapitel 9

Team Management

Kapitel 10

Time Management

10.1 Roadmap

Teil IV

Reflexion

Kapitel 11

Team

Rückblick

Nächstes Mal

Kapitel 12

Elia

Rückblick

Nächstes Mal

Kapitel 13

Marc

Rückblick

Nächstes Mal

Glossar

Anti-Cheat Software um Schummeln zu verhindern. Ist sehr komplex und bei allen grösseren online Videospielen vorhanden. Teilweise sehr tief in das System vernetzt.

cheater Jemand der unerlaubte Methoden oder Programme verwendet, um sich einen unfairen Vorteil zu verschaffen. Auf Deutsch *ein Schummler*.

Feature Eine bestimmte Funktion, welche die Software weiterbringt.

Firewall Schutzsystem eines technischen Systems um Viren fernzuhalten. Blockiert aus Sicherheit manchmal mehr als nötig und erschwert so zum Beispiel das verbinden von zwei Computern über das Internet.

Helden In Videospielen der wichtigste Charakter. In gewissen Spielen ist der Held der Charakter den man spielt. Der Tod des Helden bedeutet einen massiven Nachteil.

Ingame Im laufendem Spiel, nicht auf dem Computer oder beim Start oder Schliessen des Spiels.

Kamera Die Kamera nimmt in Unity auf, was der Spieler sieht. Mit Kamera ist also das gemeint, was der Spieler sieht..

Lane Bezeichnet generell eine Linie oder Bahn auf der gegangen werden kann. Sehr prägnant in MOBAs wie League of Legends oder Dota. In unserem Spiel eine Bahn der vier, auf welcher Truppen beschwört werden können..

Lokal Im selben Netz oder Computer.

Multiplayer Mehrspieler, also mehrere Spieler spielen zusammen das gleiche spiel..

Online nicht nur lokal auf dem eigenen Rechner sondern im Internet.

Pay-to-Win Übersetzt: Bezahlen-zum-Gewinnen, bedeutet, dass mit echter Währung Ingame Vorteile erkaufte werden können.

Shop ein Ort im Spiel, in dem Dinge gekauft werden können. Ein Ingame-Laden.

Singleplayer Einzelspieler, das Spiel wird alleine, meist nur lokal, gespielt.

Skins Erkaufbare Darstellungen, welche von dem Standard abweichen. In vielen Spielen nur im Tausch mit echter Währung zu erhalten und bringen nur ästhetische Unterschiede.

Steam Steam ist der grösste Videospiel-Anbieter der Welt. Gehört dem Unternehmen Valve..

Tutorial Eine Anleitung, in Videospielen meist eine kurze Spielsequenz, welche Tipps und eine fixe Abfolge besitzt.

Literaturverzeichnis

[Jen19] Jenschke. Medien in der kategorie “jass”. <https://commons.wikimedia.org/wiki/Category:Jass>, 2019. [Online; accessed 21-May-2022].