

FOM Hochschule für Oekonomie & Management Essen
Hochschulzentrum Siegen



Berufsbegleitender Studiengang
Wirtschaftsinformatik, 5. Semester

Projektdokumentation als Seminararbeit
im Rahmen der Lehrveranstaltung
Web Technologie

über das Thema
Browser RPG-Adventure

Betreuer: Daniel Bitzer

Autoren: Rico (Matrikelnummer 12345)
Henning (Matrikelnummer 12345)
Julian (Matrikelnummer 12345)

Abgabe: 29. Januar 2022

Inhaltsverzeichnis

| | |
|--|-----------|
| Glossar | VI |
| 1 Einleitung | 1 |
| 2 Aufgabenbeschreibung | 2 |
| 2.1 Was ist das Ziel der Projektarbeit? Worin bestehen die (wahrscheinlichen) Herausforderungen? (allg. technisch und auch persönlich) | 2 |
| 3 Anforderungen | 4 |
| 4 Herangehensweise | 6 |
| 4.1 Wie soll das Ziel erreicht werden (Vorgehen, Architektur) | 6 |
| 5 Vorstellung des Ergebnisses | 7 |
| 5.1 Vorstellung des Ergebnisses Rico | 7 |
| 5.2 Vorstellung des Ergebnisses Julian | 11 |
| 5.3 Vorstellung des Ergebnisses Henning | 11 |
| 6 Reflektion | 12 |
| Anhang | 13 |
| Literaturverzeichnis | 39 |

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

Stack Software-, Technologie- oder Lösungsstack

Glossar

Matchmaking Das Matchmaking umfasst bei Multplayerspielen das Zusammenfinden oder auch automatische Zusammenstellen von verschiedenen Mitspielenden zu einem Spiel. 5

1 Einleitung

(TODO!): Rico schreibt das.

länge = 1 Seite max

Teamzusammenstellung

Themenfindung / Auswahl Projektthema Rahmenbedingungen (Studium, Webprojekt erstellen)

Vorstellung und Grundsätzliches (kurz/grob-abriss)

Das Projektteam, bestehend aus Julian Schäfer, Henning Beier und Rico Pursche, beschäftigt sich im Studiengang Wirtschaftsinformatik im Ramen einer Projektarbeit des Studienfachs Web-Technologie mit der Entwicklung eines Browserbasierten Onlinespiels. Die Rahmenbedingungen geben vor, dass das Spiel sowohl singelplayer- als auch multiplay-fähig sein und eine Charakterentwicklung in Form von z. B. Levelaufstiegen enthalten soll. Außerdem ist es wünschenswert, dass alle Teilnehmer am Spiel, die für sie relevanten Informationen vom Spiel und den anderen Spielteilnehmern in Echtzeit erhalten. Dabei sollen aktuelle und gegeignete Technologien zum Einsatz kommen, die den Kennnisstand und den individuellen Fähigkeiten der einzelnen Teammitglieder entsprechen und mit denen die oben genannten Kriterien, vor allem die Fähigkeit online, mit Hilfe eines Browsers, zu spielen, erfüllt werden können.

2 Aufgabenbeschreibung

(TODO!): Rico schreibt das.

Umfang 1-3 Seiten

2.1 Was ist das Ziel der Projektarbeit? Worin bestehen die (wahrscheinlichen) Herausforderungen? (allg. technisch und auch persönlich)

Primär: Fertiges, spielbares Programm Projektlauf erfolgreich gestalten Web-Technologien nutzen Fokus auf die technische Umsetzung, Gameplay sekundär

Sekundär: Lernkurve Projektmanagement leben Erfahrungen im Spieldesign

Herausforderungen (siehe ??)

- techn. Umsetzbark.
- Zeit
- Know How
- Fokusverlust
- geeignete Aufgabefelder für eine Aufteilung finden

Am Ende der Spieleanwendung soll als primäres Ziel der Projektarbeit eine Art RPG (RolePlayingGame) mit Fantasysetting stehen, bei dem es zu aller erst um die Fertigstellung des kompletten Spielumfangs und nicht nur einiger Teilbereiche geht. Das Spiel soll also von Anfang bis Ende durchgespielt werden können und alle an das Projekt geforderten gestellten Anforderungen erfüllen und die gewünschten Inhalte bieten. Im Einzelnen sind im o. g. Zusammenhang folgende Dinge zu nennen. Das Spiel muss mit einem gängigem Browser online spielbar sein, es soll sowohl Singleplayer- als auch Multiplayerelemente enthalten und es wird verschiedene Charakterklassen geben, die sich in ihren Fähigkeiten klar von einander unterscheiden. Im Laufe des Spiels werden sich, bei erreichen bestimmter Grenzen in einem für jeden Charakter eigenem Erfahrungspools, diese Fähigkeiten verbessern. Des weiteren wird es vom Projektkonzept unabhängig vom Endprodukt ebenfalls als primäres Ziel angesehen, den Projektlauf erfolgreich zu gestalten. Als Sekundäre Ziele, also den primären Zielen klar untergeordnet, sieht die Projektgruppe die Spielerfahrung der einzelnen Spieler. Es ist allerdings angedacht diese im Rahmen der Möglichkeiten

so intensiv wie möglich zu gestalten, jedoch steht wie oben bereits erwähnt die Umsetzung der Vorgaben im Vordergrund. Außerdem möchte jeder der Teilnehmer an diesem Projekt Erfahrung im Spieldesign sammeln. Die Herausforderungen in diesem Projekt sind recht vielfältig und sind grob in technisch und persönlich zu gliedern. Zu ersterem ist zu zählen, dass zu Beginn des Projekts nicht klar ist, ob das vom Team ersteinmal theoretisch entwickelte Konzept auch real, technisch umzusetzen ist. Zum einen weil nicht sicher ist ob die gewählte technische Plattform geeignet ist und zum anderen weil nicht klar ist ob das Know-how des Teams reicht um das gewollte in ein fertiges Endprodukt zu gießen. Außerdem könnten Rechte an Grafiken, Musik/Sounds und vielleicht sogar Programmcode zum Problem werden. Die Persönlichen herausforderungen liegen darin, das ganze Projekt zeitlich so zu gestalten, dass es neben der hauptberuflichen Tätigkeit des Projektteams, noch genug Zeit für die Umsetzung bleibt. Und Krankheit oder Karantäne in der aktuellen Pandemie von einzelnen Teammitgliedern könnte den zeitlichen Ramen ebenfalls in Gefahr bringen. Auch muss für jedes Mitglied des Teams ein geeignetes Aufgabenfeld gefunden werden, so dass die individuellen Fähigkeiten sich möglichst ideal ergänzen um z. B. eine Doppelbelastung eines anderen Teammitgliedes zu vermeiden.

3 Anforderungen

Die Anforderungen an das Projekt wurden aus verschiedenen Quellen definiert. Hier zum einen allein schon durch das Modulthema „**Web Technologie**“ sowie weiter im Allgemeinen sowie im Speziellen durch den **Dozenten** als auch abschließend auch durch uns als **Projektteam** selbst.

Das Modulthema gibt hierbei das grundsätzliche Umfeld vor und bestimmt damit auch wesentlich alle weiteren Details folgender Anforderungen. Der Dozent gab hier allgemeine Anforderungen an alle Teilnehmende des Moduls, und definierte weiter auch spezielle Anforderungen an die Durchführung unseres Projektes. Die dritte Quelle von Anforderungen ist das Projektteam selbst.

All diese Anforderungen wurden gebündelt in der Projektbesprechung vom 27.11.2021 erfasst, von uns bewertet und zusammen in eine Projektskizze (siehe Anhänge 5 und 6) transportiert. Mit den darin erfassten Inhalten unter den Punkten **Zielsetzung, Aufgaben und Ergebnisse** sowie **Randbedingungen** wurden alle gestellten Anforderungen systematisch erfasst und damit sichergestellt, dass diese im Projektverlauf auch entsprechend Beachtung finden und erledigt werden.

Die wentsentlichen Anforderungen, deren Erfüllbarkeit vor allem auch an die gewählten Technologien gebunden sind, wurden wie folgt besimmt: Das Spiel muss...

- ...im Web laufen.
- ...multiplayerfähig sein.
- ...eine gewisse Komplexität (Charkterentwicklung, Klassen) haben.
- ...performant und stabil laufen.

Die Anforderungen „im Web laufen“ sowie „performant und stabil“ sind bei korrekter Konfiguration, entsprechender Hardware und Wartung durch nahezu jeden Software-, Technologie- oder Lösungsstack (Stack) zu erfüllen.

Da das gesamte Projektteam unerfahren mit der Umsetzung von Webprojekten war und die ersten, in den Vorlesungen zum Modul gesammelten Erfahrungen, mit der dort Vorgestellten Lösung **Djnago** durchweg positiv waren, wurde dem eine deutliche Präferenz zugeschrieben. Die zuerst genannten Anforderungen, dürfen bei Django auch als erfüllt betrachtet werden. Django ist weit verbreitet und wird produktiv im gewerblichen Einsatz erfolgreich betrieben.

Die nächste Anforderung „multiplayerfähig“, stellte bereits höhere Anforderungen an die eingesetzte Technologie. Zuerst war hier zu klären bzw. zu definieren, wie genau der Ablauf des Spiels sein soll. Auch musste festgelegt werden welche Arten von Interaktion zwischen dem Spiel und den Spielern und auch unter den Spielern selbst gewünscht sind.

Wir haben hier in Projektbesprechungen schnell erkannt, dass das Spiel mit allen möglichen Interaktionen, jeweils einzeln genau definiert werden müssen. Nur so lassen sich Anforderungen prüfen und die spätere Programmierung erledigen.

Bei dieser Definition wurde schnell klar, dass es (bedingt durch die geforderte Multiplayerfähigkeit) nicht nur einen bedeutenden Aufwand zu Entwurf und Programmierung bezüglich der Spiellogik selbst, sondern auch bezüglich des **Matchmaking** geben wird.

Das Matchmaking sollte hier nicht automatisch erfolgen, sondern bewusst durch die Spieler. Ein gewisser Austausch ist dafür unabdingbar: Es wurde eine Chat-Funktion benötigt. Über eine Recherche dazu, wurde schnell das **WebSocket-Protokoll** gefunden. Gemäß der Beschreibung nach heise online, developer, Matthias Wessendorf (2011) sollten unsere Anforderungen umfassend erfüllt werden. Das WebSocket-Protokoll ist ähnlich einer TCP-Verbindung eine bestehende Verbindung, die es auch erlaubt Events von Serverseite aus, an den Client (ohne dessen gesonderte Anforderung), zu übertragen.

An vielen Stellen im Projekt, würde dies eine Anforderung sein, die mit den Websockets damit erfüllt werden konnten.

Alternative Frameworks (Backend, als auch Frontend) waren uns zwar teilweise vom Namen bekannt, wurden aber weiter nicht genauer in Betracht gezogen. Auch damit grundsätzliche Erfahrungen im Umgang mit HTML, CSS und JavaScript im Rahmen des Projekts durch das Team gesammelt werden können. Die Abkürzung über ein Framework wurde daher hier auch bewusst nicht gewählt.

4 Herangehensweise

(TODO!): Julian schreibt das.

4.1 Wie soll das Ziel erreicht werden (Vorgehen, Architektur)

Einstieg in die Arbeit war das etablieren eines Projekmanagemt nach Scheurer (lit. Verweis!!!!).

Im Kick-Off die Projektskizee auf ein weißes Blatt papier geschrieben.

Vorallem: Ziele definiert (aus denen sich viele der nächsten Schritte ergeben)

Hier inhalt analog Projektskizee durchegen. Dabei auch aufzeigen, dass der weitere Projektablauf definiert wurde (wöchentliche Treffen, Meilensteine, ...)

In Projektbesprechung abgestimmt, dass

- Wöchentliche Abstimmung Sonntags
- Zuerst Entwicklung Prototyp um Machbarkeit und Aufwand abschätzen zu können
- Gleichzeitig schon Einstieg in Setting und Spieldesign
- Gleichzeitig auch Einstieg in grafisches Design

5 Vorstellung des Ergebnisses

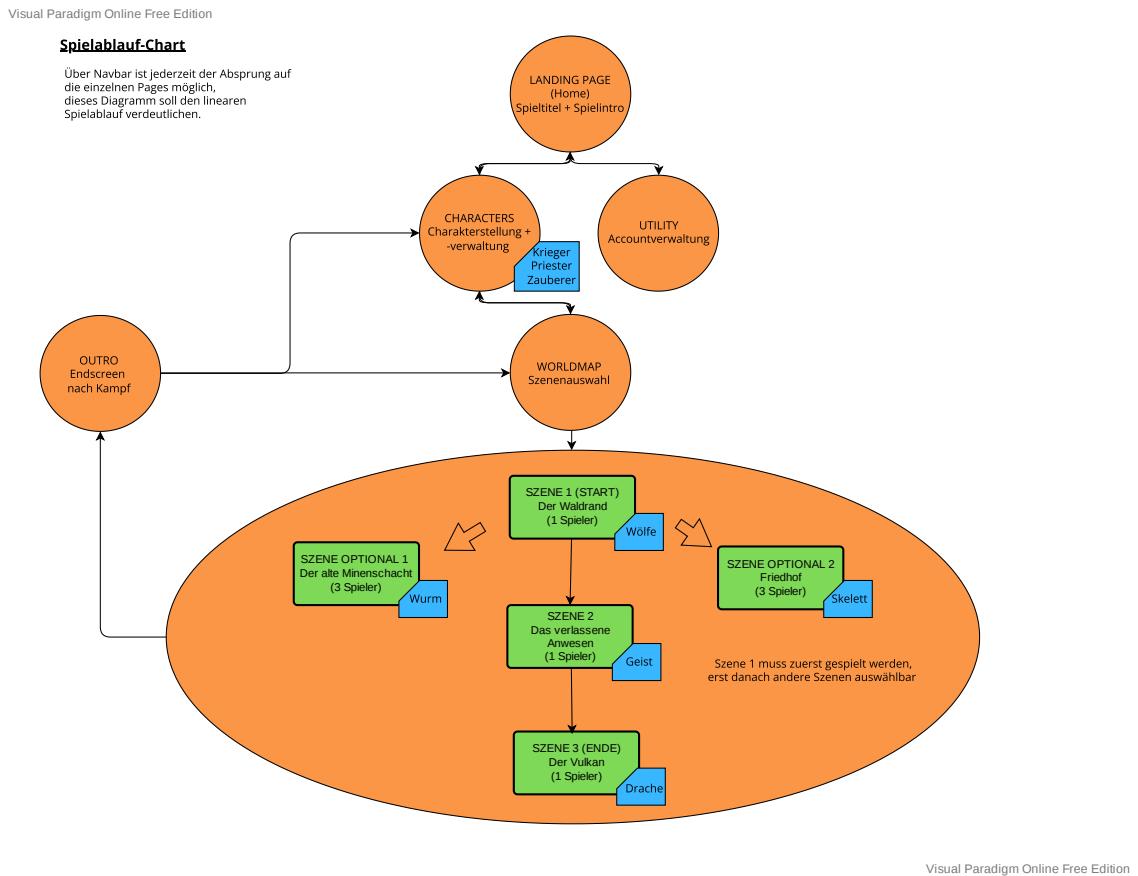
(TODO!): Julian schreibt die Übersicht hierzu.

Einleitung mit eher detaillierterem Überblick über das Spiel.

Gerne alle Funktionen benennen, nichts erklären. Überblick verschaffen.

Dazu Spielablaufskizze abarbeiten (Die Zeichnung(17) und Erläuterung erklären)

Abbildung 1: 08.01.2022: Diagram: Seitenaufbau und Spielablauf



(keine 5 Seiten)

5.1 Vorstellung des Ergebnisses Rico

Balancing Texte und Story: Spieldesign

Spieldesign: Am Anfang der Entwicklung musste ein geeignetes Setting für das geplante RPG* gefunden werden. Zur Auswahl standen die beiden klassischen, den meisten bekannten Arten von Spieleuniversen zur Auswahl. Das eine wäre futuristisches Scifi und

das andere eher mittelalterliches Fantasy. In der Frühen Phase der Entwicklung war beim Spieldesign ursprünglich geplant ein RPG zu entwickeln, dass viele Elemente vom klassischen Pen and Paper Rollenspielen*, wie z.B. Dungeons and Dragons enthält. Man wollte unterschiedliche Charakterklassen, die sich in ihrer Ausrüstung, wie individuelle Waffen und Rüstungen, sowie ihren spezifischen Fähigkeiten klar von einander unterscheiden. Ein Levelsystem, bei dem die einzelnen Charaktere von Abenteuer zu Abenteuer ihre Fähigkeiten verbessern und bessere Ausrüstung in Form von Beutegut finden können, sollte auch enthalten sein. Das Entwicklerteam hat sich dann für das Fantasysetting entschieden, weil man der Ansicht war, dass sich die vorher genannten Eigenschaften damit besser umsetzen lassen würden. Speziell die unterschiedlichen Eigenschaften der Charakterklassen waren bei dieser Entscheidung massgeblich, denn eine Eingrenzung auf die besonderen Fähigkeiten der unterschiedlichen Spielfiguren, wie z. B. einen Tank*, einen Heiler* und einen Damagedealer*, schien im SciFi-Hintergrund nicht so einfach und für den Spieler nicht schlüssig zugestalten. Die Überlegungen führten somit schon am Anfang recht schnell zu den bereits erwähnten unterschiedlichen Spielfiguren, die der Spieler verkörpern kann. Der Tank(, ein aus dem englischen entlehntes Wort für Panzer,) soll Schaden von anderen und sich selbst abhalten können. Der Heiler soll(, wie das Wort schon verrät,) seine Kameraden und sich heilen können und der Damagedealer soll (,wie das englische Wort andeutet) seinen Schaden und den der anderen Spielfiguren erhöhn. Wie sich dies im Einzelnen gestaltet, darauf wird später noch genauer eingegangen. Es war angedacht, dass die einzelnen Spielfiguren über Lebenspunkte, Magiepunkte und verschiedene Handlungsoptionen zu geben, die sich abhängig von der Charakterklasse, von einander unterscheiden. Auch schien klar zu sein, dass die Würfelmechanik, die Pen and Paper Rollenspielen zu Grunde liegt in das Spiel übernommen werden soll. Das Spiel sollte vom Ablauf her, aus aufeinander aufbauenden Abenteuern bestehen, die die Charaktere gemeinsam erleben und die mit Hilfe von unterschiedlichen Ansätzen gelöst werden können, bestehen. Es war geplant als Grundlage für die Regeln, denen das Spiel folgen, also wie ist zu bestimmen, wie und in welcher Reihenfolge der Kampf abläuft oder zu welchen Bedingungen genau wird bestimmt, wann z.B. ein Schwerthieb trifft oder nicht, dass sollte auf Grundlage der Opensource Licens von dem Rollenspiel Dungeons and Dragons 3.5 basieren. Nachdem dieser grobe Ramen festgelegt wurde, haben wir entschieden erst einmal einen Prototypen zu entwickeln, der exemplarisch anhand eines Abenteuers, im folgenden nur noch Spielszene oder Szene genannt, zu klären ob der gesetzte Ramen auch so umsetzbar ist. Dabei ist recht schnell aufgefallen, dass vor allem die geplanten Unterschiede der einzelnen Charakterklassen, in Form von sich unterscheidenden Handlungsmöglichkeiten in der Szene nicht klar abzugrenzen ist. Außerdem war nicht klar, wie es zu handhaben ist, wenn man z. B. vor einer verschlossenen Tür steht,

diese zu öffnen ist, und jede Charakterklasse eine andere Möglichkeit hat dieses Hindernis zu überwinden. Wenn auch bei diesem einfachen Beispiel recht einfach zu klären ist, wie die unterschiedlichen Handlungsmöglichkeiten aussehen, nämlich "Tür eintreten", SSchloss knackenünd die Tür mit einem Zauber öffnen, war nicht klar wie genau das ablaufen sollte. Wer darf den in diesem Moment handeln, der der zuerst auf den Button klickt und was geschied nach entsprechend der unterschiedlichen Auswahl der Möglichkeiten? Sollte alles zu unterschiedlichen Ergebnissen führen oder alles zur selben? Dies macht wiederum die Auswahl was zu tun ist völlig unnötig. Falls die Auswahl zum unterschiedlichen Ergebnissen führt, ensteht ein für uns nicht einzuschätzender Aufwand an Storytelling, grafischen Animationen und an Programmierarbeit. Außerdem ist die Komplexität der Kampfmechanik, wie sie in dem zu grunde liegenden Regelsystem von D. a. D. 3.5 recht anspruchsvoll und für einen gemütlichen Abend mit seinen Freunden und auf das Spielen von Angesicht zu Angesicht im Real Live ausgelegt. Ein Charakter in diesem Regelwerk besteht aus vielen Eigenschaften, die in Zahlenwerten festgehalten werden. Diese sind unter anderem Stärke, Geschicklichkeit, Konstitution, Intelligenz, Weisheit und Charisma und liegen zwischen 1 und 18, wobei 10-11 als Durchschnitt angesehen werden. All diese Werte beeinflussen wie er z.B. kämpfen, stehlen oder mit anderen Menschen interagieren kann. Dies ist nur eine grobe Zusammenfassung, denn es gibt noch viel mehr was die Regelmechanik beeinflusst und auch große Spieletitel wie z.B. Baldur's Gate, die ebenfalls auf d.u.d. 3.5 basieren, haben nicht das gesamte Regelwerk übernommen. Auf Grund all der oben genannten Unwägbarkeiten und Komplexität haben wir uns an diesem Punkt zu mehreren, im folgenden erklärten, Konzeptänderung entschieden. Der gravierendste ist sicher, dass aus dem RPG eine Art Beat em Up mit Fantasyhintergrund und atmosphärischen Texten und Grafiken geworden ist. Dabei gibt es zwar noch immer einzelne Level, die man durchspielt, diese sind aber recht linear und bestehen nur aus einem Kampf gegen einen Gegner, den man entweder gewinnt oder verliert. Ganz grob soll es so ablaufen. Am Anfang der Szene wird ein Hintergrundbild mit Texterklärung eingebendet und nachdem die oder der Spieler bestätigt haben, wechselt die Ansicht. Auf der linken Seite dieser Ansicht sieht man die Charaktere und auf der anderen Seite den Gegner. Je nach Ausgang des Kampfes wird ein anderer Outrottext angezeigt und man springt zurück in die Auswahl für neue Kämpfe. Es wird keine Ausrüstung geben, die die Charaktere finden können, da dies ein zu umfangreiches Datenbankmanagement, für das von uns anvisierte Ziel voraussetzt. Das Konzept der Möglichkeit, dass ein Charakter seine Eigenschaften verbessert in dem er bei den Kämpfen an Erfahrung gewinnt, besteht weiterhin, wenn auch etwas einfacher als Angedacht. Auch unterscheidet sich jede Charakterklasse von der anderen durch eine andere Spezialfähigkeit. Die Klassen sind ebenfalls geblieben nur hat sich die Bezeichnung etwas konkretisiert. Im Einzelnen

sind das der Kämfer, er hat die Möglichkeit den erhaltenen Schaden zu reduzieren. Dann gibt es noch den Priester, er hat die Fähigkeit zu Heilen und dann ist da noch der Magier, der den Schaden erhöhen kann. Jeder Kampf gibt eine individuelle Summe an Erfahrung, die dem Charakter gutgeschrieben wird und bei erreichen einer Festgelegten grenze, steigt dieser eine Stufe auf. Dabei werden seine Lebenspunkte erhöht und seine Spezialfähigkeit wird besser. Ein Charakter erhält immer Erfahrung, egal ob der Kampf gewonnen wird oder verloren geht. Bei einem Sieg ist diese jedoch deutlich höher als bei einer Niederlage. Das und die vordefinierten Levelgrenzen haben zur Folge das man einen Level eventuell mehrfach spielen muss um eine Erfahrungsstufe aufzusteigen. Es werden bestimmte Erfahrungsgrenzen nötig sein, um bestimmte Spielabschnitte spielen zu können. Und der endgültige Tod eines Charakters ist nicht vorgesehen. Die Kampfmechanik wird wie folgt aussehen. Der Kampf ist deutlich vereinfacht und wird Rundenbasiert ablaufen, wobei in jeder Runde der Gegner zuerst angreift. Danach wird jeder Charakter die Möglichkeit haben entweder anzugreifen oder seine Spezialfähigkeit einzusetzen. Falls der Charakter seine Spezialfähigkeit nutzt, kann er nicht angreifen und umgekehrt. Die Spezialfähigkeit wirkt mehrere Runden nach. Es ist nicht vorgesehen zu testen ob ein Angriff trifft oder nicht, ein Angriff trifft also immer und macht Schaden. Der Gegner verfügt über keine spezialfähigkeiten und macht allen Charakteren den selben für ihn spezifischen Schaden. D. h. der Wolf der 20 Punkte Schaden pro Angriff macht und mehreren Charakteren gegenübersteht, fügt jedem Spieler die 20 Punkte Schaden zu. Der Schaden der Spieler summiert sich, so dass drei Spieler die jeweils 20 Schaden zufügen, dem Wolf 60 Punkte Lebensenergie abziehen. Vor jedem neuen Kampf verfügen die Spieler wieder über ihre vollen Lebenspunkte. Storytelling: Auf Grund der Entwicklung war es nötig zu jeder einzelnen Szene eine Story zu schreiben um dem Spieler ein Gefühl zu geben, was gerade passiert und warum. Dabei war es wichtig sich auf die Beschreibung der dargestellten Szene zu konzentrieren und nicht abzuschweifen, da ein zu langer Text vom Spieler vielleicht nicht gelesen wird oder als störend empfunden wird. Jedoch muss er lang und intensiv genug sein, damit sich der Spieler eine Eindruck von dem Geschehen verschaffen und darin eintauchen kann. Schließlich soll das Spiel ja auch eine Geschichte erzählen und dem Spieler auch eine gewisse Spielerfahrung und im Idealfall einen Wiederspielwert geben. Beim Entwickeln der einzelnen Szenen ist aufgefallen, das man nicht immer genau sagen kann was zuerst da war, das Huhn oder das Ei, denn der Text und die Grafik stehen in engem Zusammenhang und haben sich gegenseitig beeinflusst. Die Beschreibung der Szene stand üblicherweise zuerst und danach wurde die Szenengrafik entwickelt. Aber manchmal war es auch umgekehrt oder es war nötig den Text anzupassen, weil die Animation z. B. von einem Rudel Wölfe schwieriger war als die von einem einzigen riesigen Wolf. So wurde aus dem Rudel Wölfe was die Ge-

gend terorisiert ein einziger großer Warg. Oder im Fall der Szene im Anwesen war bei ursprünglicher Planung ein Geist vorgesehen, aber beim Schreiben der Geschichte wurde daraus ein Vampier, der viel düsterer ist und sinniger passt. Daraus ist zu erkennen, dass bei einer Spieleentwicklung diese beiden Teilbereiche sehr eng zusammen arbeiten sollten. Dem Programmierer z. B. ist es egal, oder wie unser Leadprogrammer sagte "ich bin da total leidenschaftslos", was für eine grafik oder text an entsprechender Stelle im Code eingefügt werden. As ber Texte und Grafik müssen unbedingt Hand in hand gehen und sich ergänzen um eine überzeugende Spielerfahrung zu schaffen. Im Fall des kampfes gegen den Drachen bin ich bewusst von der reinen Beschreibung der Szene abgewichen und habe viel mehr die Motivation des charakters und etwas Hintergrundgeschichte in den Vordergrund gestellt um beim Spieler die Bewehgründe des Charakters in den Fordergrund zu stellen damit er sich nicht wie bei den anderen Szenen in Handlung sondern mehr in den Akteur hinein versetzen kann. Beim Endboss des Spiels sollte auch etwas besonderes im text stehen. Balancing: Bei der Ausarbeitung des Ballancings

5.2 Vorstellung des Ergebnisses Julian

Frontend Grafiken/Animationen

bissl was zu Spieldesign

5.3 Vorstellung des Ergebnisses Henning

Codebeispiele

Infrastruktur

6 Reflektion

(TODO!): Jeder einen Punkt ausschreiben.

3-4 gute Beispiele, max 1-2 Seiten

- Umänderung Spielkonzept von Szenenbasiertem Gameplay (Verschiedene Wege, Adventuremäßig, Dialoge, ...) hin zu reinem Kampf-RPG.
- Julian...
- **Rundenbasierter vs. chaotischer Spielablauf:** Der Ansatz die Entwicklung durch den Einsatz eines rundenbasierten Ablaufs bzw. Kampfsystems deutlich zu vereinfachen, muss wohl nach den Entwicklungen vom 27.12.2021 (Anhang 2.13) zumindest angezweifelt werden. Denn der Aufwand, der durch die notwendige Konzeptionierung und Detailplanung entsteht ist nicht zu unterschätzen. Demgegen stünde bei einem chaotischen Spielablauf lediglich das Handling der Events.
- **Kleinteilige Aufgabenpakete:** In der Entwicklung eine überschaubare Anzahl an kleinteiligen bzw. Teilufgaben vor sich zu haben, empfand Henning als sehr hilfreich. Man hat damit einen Überblick über die Arbeit der nächsten Tage. Bei der Entwicklung der Rundenlogik entstand durch die Kenntniss um die nächsten Rundenschritte hier ein stets guter Überblick. Der Abschluss jeder einzelnen Teil-Aufgabe sorgte weiter laufend für positive Motivation.
- **Lernkurve und Codequalität:** Eigentlich müsste am Ende eines Entwicklungs-Projektes stets noch einmal von vorne anfangen. Das allein um alles zu korrigieren und anzupassen bzw. auf einen gleichen Quaitätsstand zu bringen, was im Projektverlauf bei den Beteiligten an Fähigkeiten und Wissen gelernt wurde.

Anhang

In diesem Abschnitt sind aktuell einige Unterlagen eingefügt, die im Rahmen des Projektes eine Relevanz hatten. Vor Abgabe der Projektarbeit, soll dieser Abschnitt überarbeitet, ausgedünnt und ergänzt werden.

Anhang 1: Projektnotizen

Austausch und Zusammenarbeite erfolgte auf verschiedenen Platformen:

- Gezeichnet und Entwürfe wurden meist in Miro erstellt: <https://miro.com/app/board/uXjVOdN2haQ=/>.
- Besprechungen erfolgten meist in Teams: https://teams.microsoft.com/l/team/19%3aDoBvOwOIC6WNhsL9kOIYFKNtVftU1yBtcEn_gcyQtcg1%40thread.tacv2/conversations?groupId=850a22ff-34a2-4fe2-a506-f55ac4d595f8&tenantId=b9b6f99a-a243-422d-ab36-f726574c981a.
- Der gemeinsame Code und die Dokumentation wurden auf Github erstellt: <https://github.com/tstsrv-de/tstsrv-de>.

Anhang 1.1: Projektbesprechungen

Stets Sonntags erfolgten Projektbesprechungen. Notizen und Zusammenfassungen davon finden sich hier in fortschreitender, chronologischer Reihenfolge. Ebenso hier entsprechend eingesortiert, finden sich Konzeptzeichnungen und Entwürfe aller Art (UI, Code, Datenbankmodelle).

(TODO!) Bildbeschreibungen ergänzen, wichtige Bilder beschreiben.

2021-11-23-erster-entwurf-gameloop

Abbildung 2: 2021-11-23-erster-entwurf-gameloop

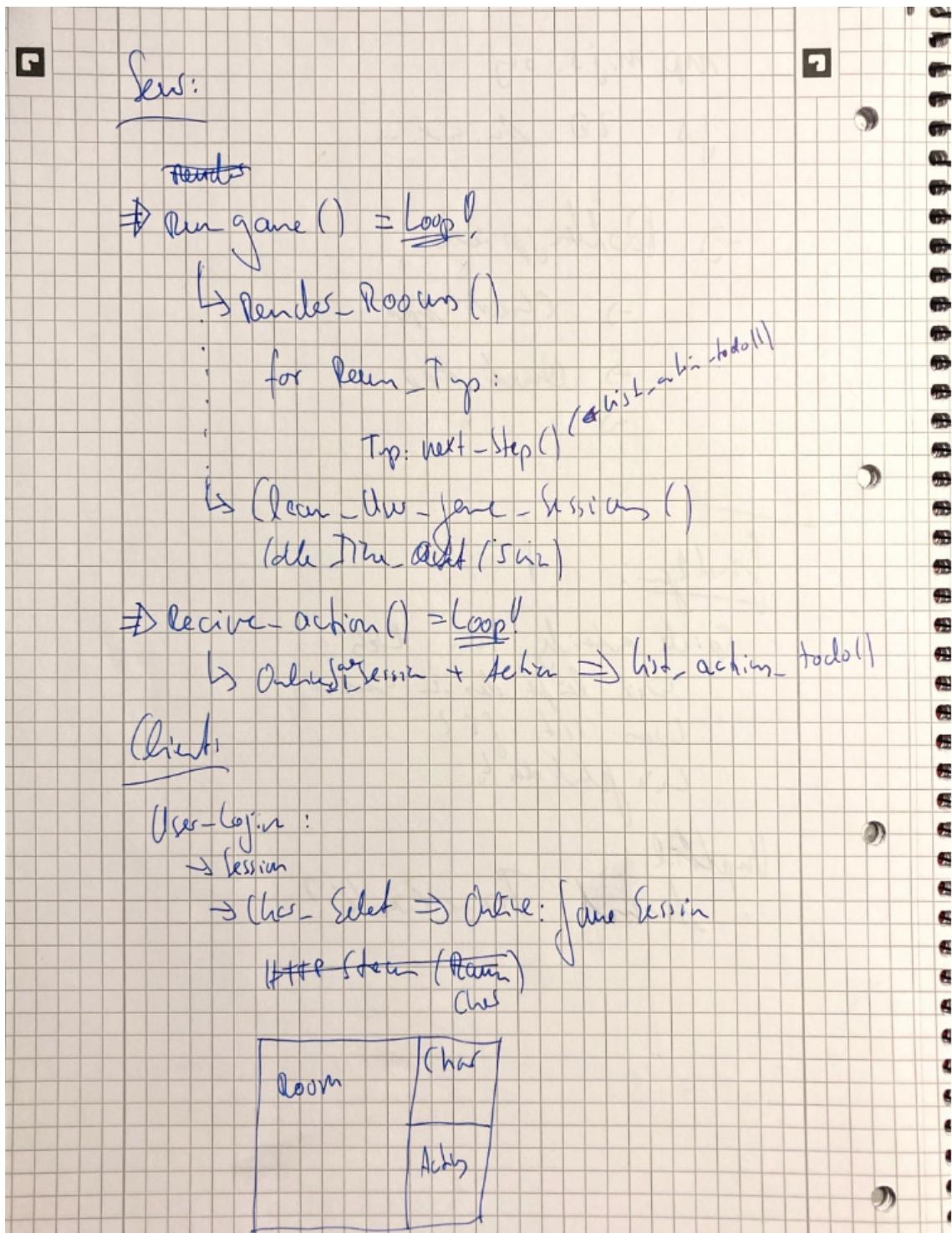


Abbildung 3: 2021-11-23-erstes-db-konzept

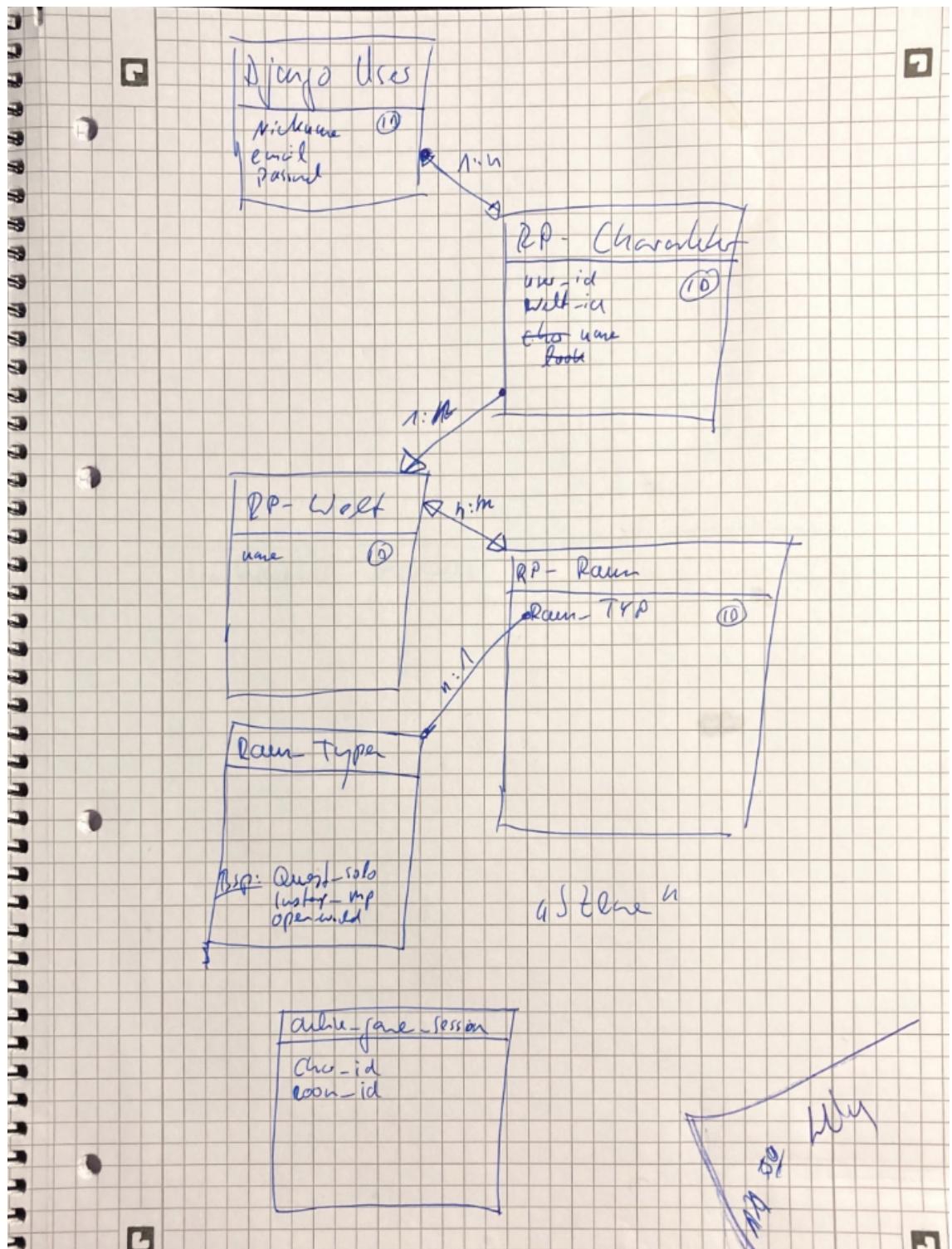
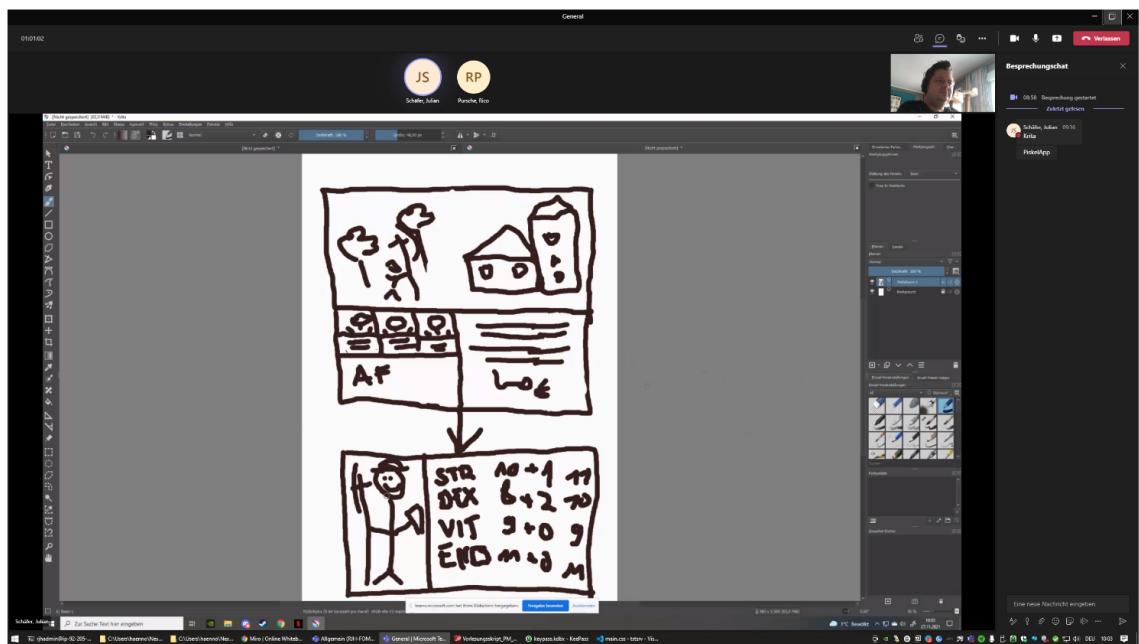


Abbildung 4: 2021-11-27-erstentwurf-ui

2021-11-27-projektskizze-1

Abbildung 5: 2021-11-27-projektskizze-1

Projektskizze

Samstag, 27. November 2021 09:28

Projektskizze

Name des Projektes:

<hier Spieldtitel einsetzen>

Zielsetzung:

- Multiplayer-Komponente
- Meilensteile mit Terminen setzen (und einhalten :))
- Benutzerverwaltung (Login etc.)
- Charaktererstellung (Creator)
- Balancing (Charakterfortschritt)
- Weltkarte als Hub -> Lobby -> Spielszene
- Weltkarte
 - Statisches Hintergrundbild (+ Sprites?)
 - Einzelne Sprites / Grafiken als Szeneneinstiegspunkte
 - Anzeige der aktiven Spieler
 - Weltchat
- Lobby
 - Jede Szene hat eine Lobby
 - Füllstandsanzeige auf Weltkarte
- Spielszene
 - Fortschrittsmechanik (mehrere aufeinander aufbauende "Level")
 - Kampfmechanik (Gegnerklassen etc.) / Ressourcenmechanik (Leben, Mana etc.)
 - Szenarten: Rätsel, Kampf, Charakterentwicklung
 - Events (Skriptereignisse)
 - Gestaltung im Stil eines klassischen RPG's (Szenenbild, Menüleiste, Kampfflog)
- Szeneneditor + Speicherung auf Datenbank
- Musik / Ton
- Datenbankanbindung
- Spielverwaltung (Überwachung eines laufenden Spiels, Schließen von Szenen etc.)
- Spiel muss durchspielbar sein
- Projektarbeit schreiben

2021-11-27-projektskizze-2

Abbildung 6: 2021-11-27-projektskizze-2**Aufgaben & Ergebnisse:**

- Technische Machbarkeit sicherstellen
- Technisches Grundgerüst bauen
- Inhaltliches Konzept erstellen
 - Setting
 - Grafikstil / Leveldesign
 - Musikstil
 - Spielfortschritt
 - Charaktere
 - Gameloop (Anfang->Mittelteil->Ende, Game Over Mechanik etc.)
 - Story
- Dokumentation (in Hinblick auf schriftlichen Teil der Arbeit)

--> Projektarbeit abgeben (Schrift + Code + Link)

Risiken:

- Technische Umsetzbarkeit
- Fehlende Zeit
- Fehlendes Know-How
- Rechte (Grafiken / Musik / Code)
- Personalmangel (Ausfall durch Krankheit / Jobs etc.)
- Verstrickung in Kleinigkeiten / Fokusverlust

Randbedingungen:

- Spiel muss im Web laufen
- Spiel muss Multiplayer-fähig sein
- Spiel sollte eine gewisse Komplexität haben (nicht zu simpel)
- Spiel muss performant und stabil laufen

Termine:

- 13.02.2022, 23:59 !!!

Auftraggeber:

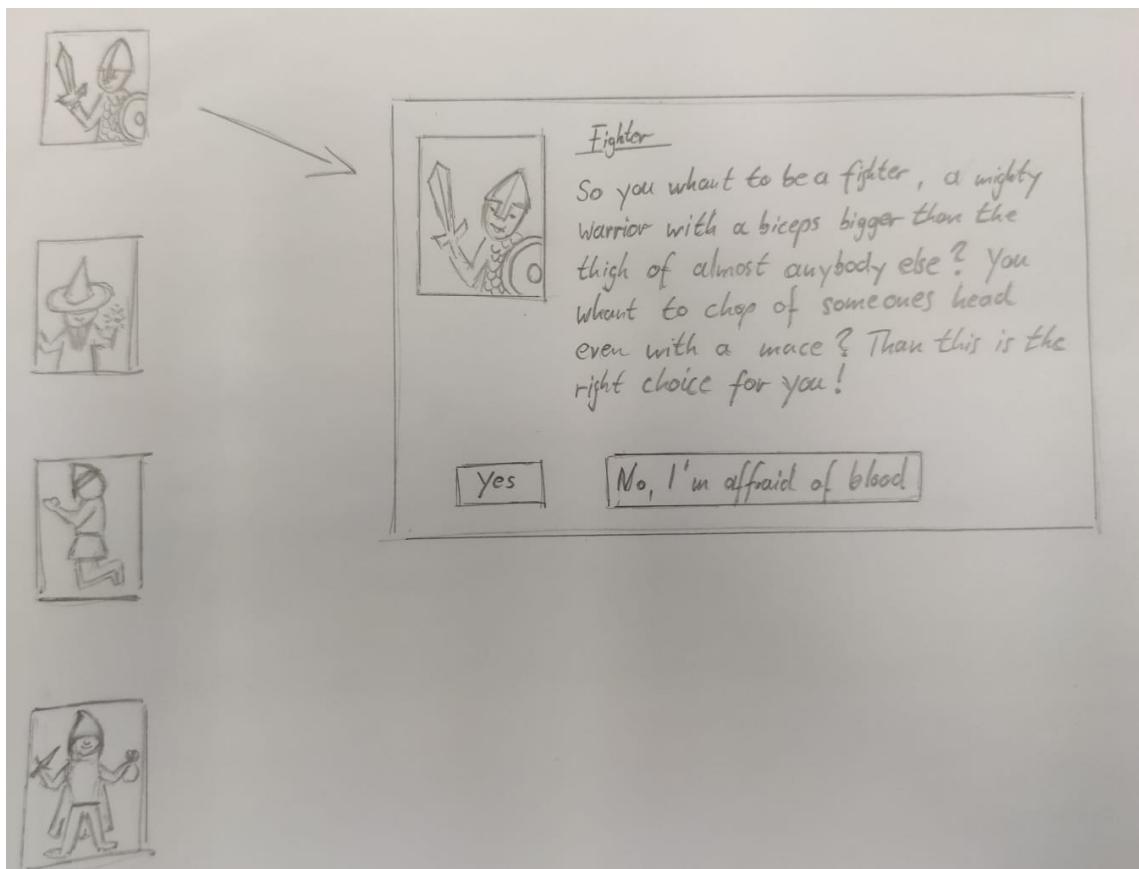
- Daniel Bitzer / FOM

Aufwand in Personentagen:

<hier Meilensteine einsetzen>

2021-11-29-Entwurf-Klassen-Ui

Abbildung 7: 2021-11-29-Entwurf-Klassen-Ui



2021-11-30-Entwurf-Lobby-Logik

Abbildung 8: 2021-11-30-Entwurf-Lobby-Logik

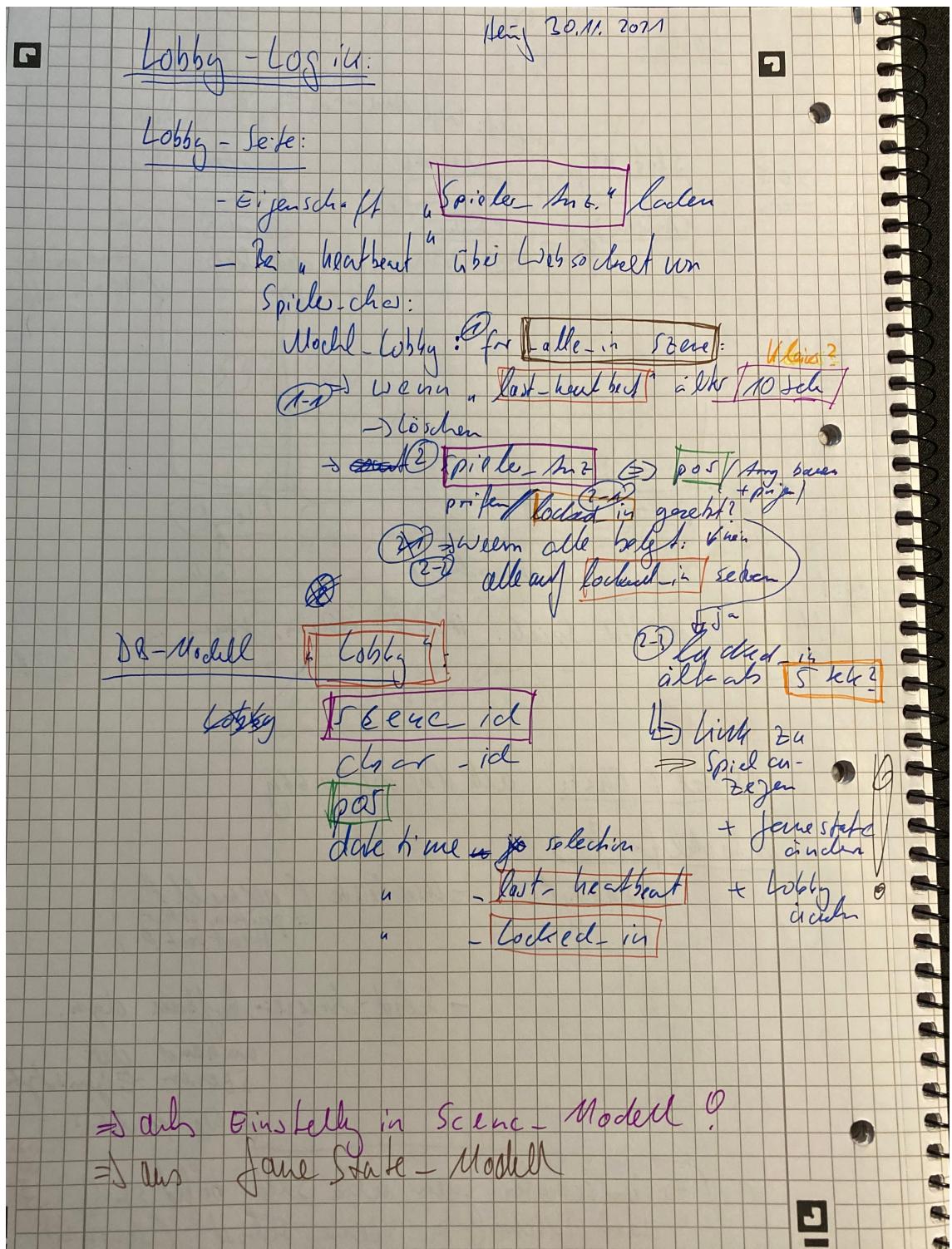


Abbildung 9: 2021-11-30-Entwurf-Lobby-UI

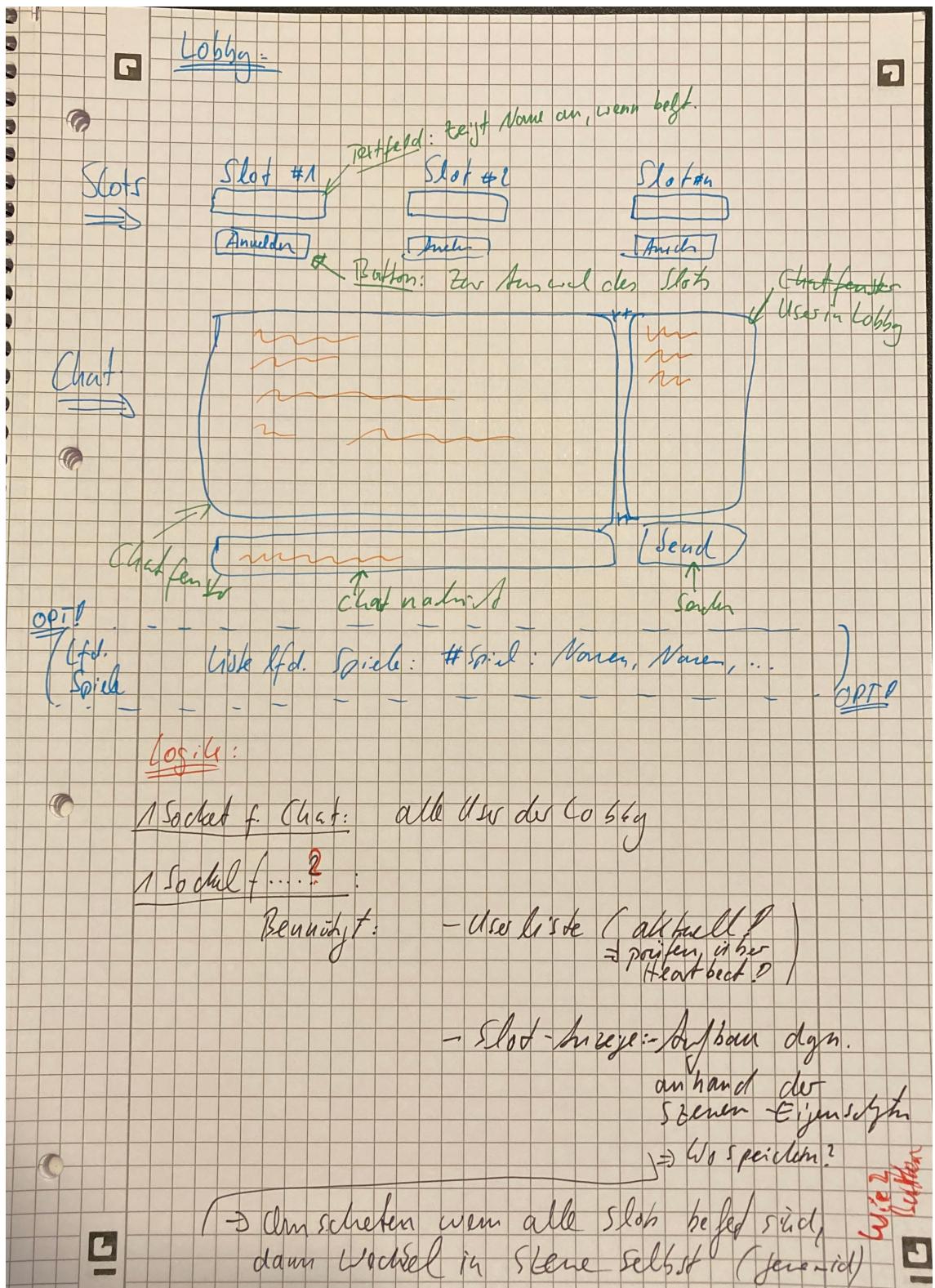


Abbildung 10: 2021-12-02-Countdown-Logik

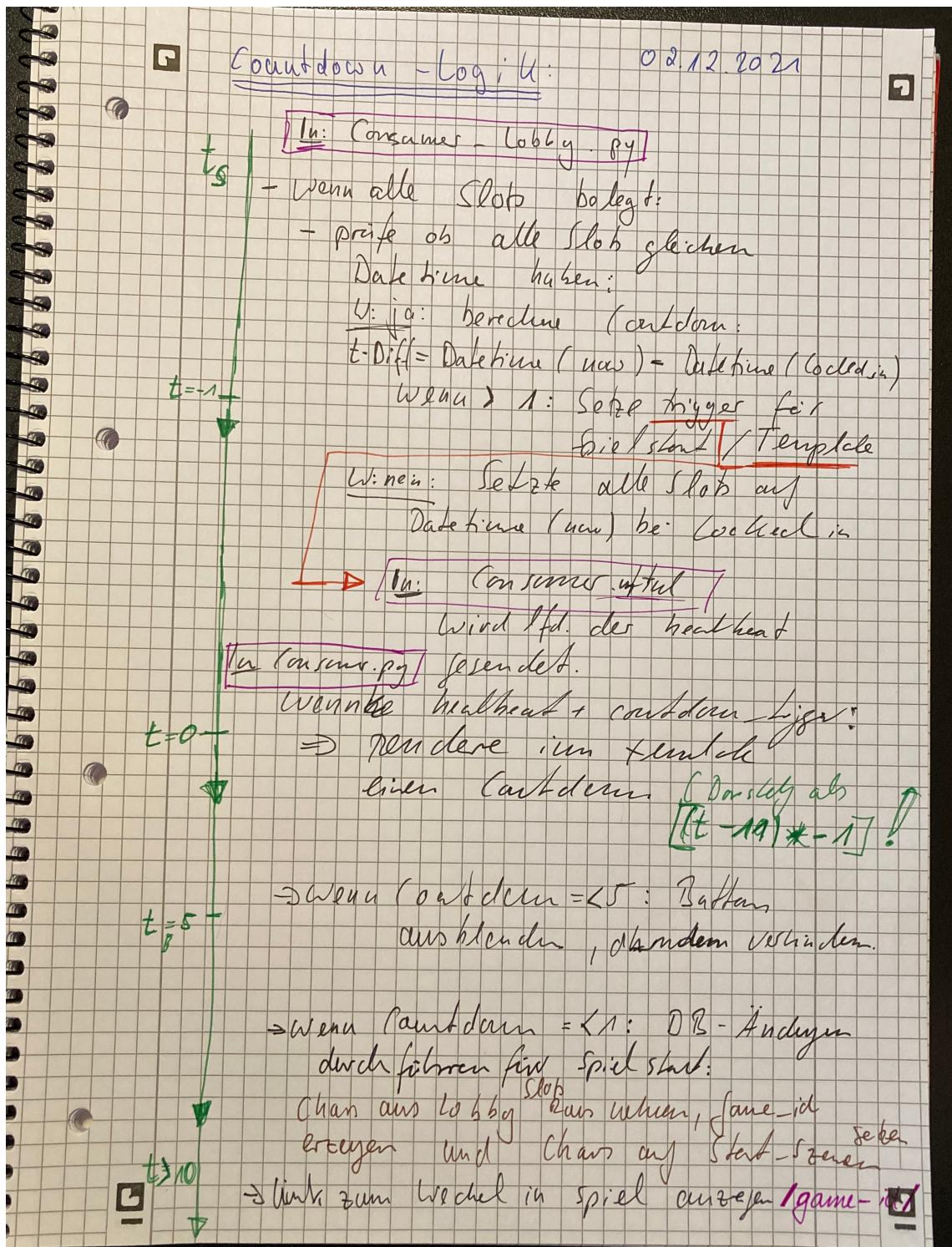


Abbildung 11: 2021-12-05-Projektbesprechung-Miro-b

Projekt-Besprechung 05.12.2021:

- Noch mal aufsetzen der lokalen Entwicklungsumgebungen
- Durchgehen der individuellen Anforderungen für weitere Arbeit:
 - Rico: Feedback zu Ansätzen der Story und dem Setting
 - Julian: Bittet um Konkretisierung der nötigen Grafik-Assets
 - Henning: Kurzes Brainstorming für Input zur Szenen-Logik
- Durchsprechen von Detailanforderungen zu Szenen und dem Ablauf von Spielen:
 - Würfel sollen Teil der Spieldynamik sein
 - Intro-Texte zu jeder Szene (!)
- Gemeinsam Mockup vom Spiel-Screen gezeichnet (s.unten)
- Spiel im Vollbild (ohne Nav usw.)
- Es soll immer nur ein einzelnes Spiel pro User laufen dürfen
- Nächste Programmierungen Henning:
 - Game Scenen erweitern
 - Anzeige der UserChars im Spiel
 - Aufbau der möglichen Aktionen anhand der 'Possible Actions'
 -

To-do's Rico:

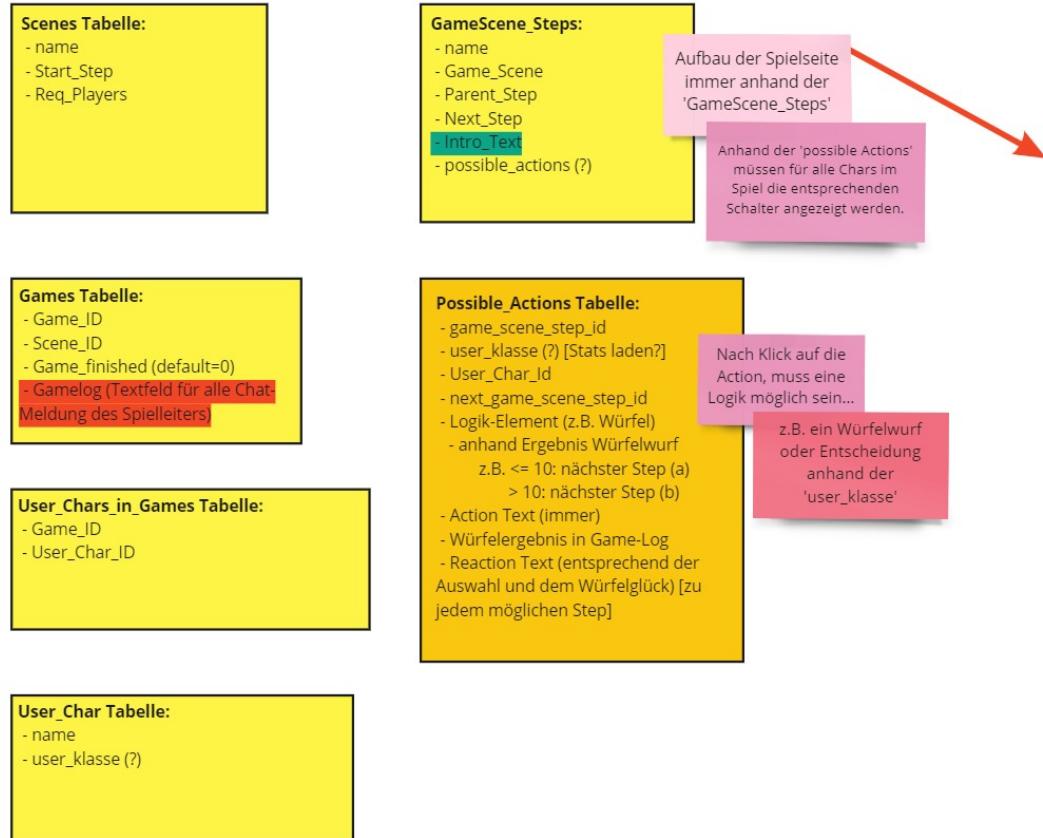
- Minimumwerte der Charaktere festlegen
- Klassen ausarbeiten
 - Infotexte etc.
- wie am Besten Kampfmechanik mit Würfelwurf abbilden

- To-Do's Julian:

- Szene schreiben
 - d.h. "Story"-Konzept ausarbeiten
 - Rätsel / Aktionen und generell den Ablauf festlegen
 - Entsprechende Grafiken erstellen (Szenenbilder)

Agenda für nächsten Sonntag (12.12.21):

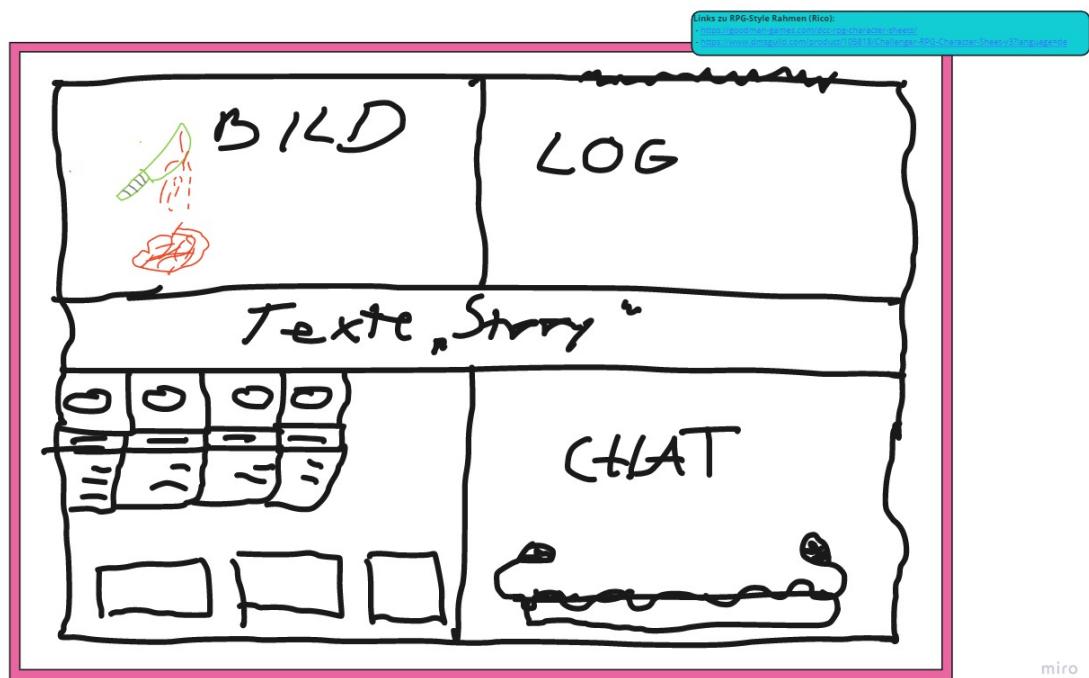
Abbildung 12: 2021-12-05-Projektbesprechung-Miro-c



miro

2021-12-05-Projektbesprechung-Miro-d

Abbildung 13: 2021-12-05-Projektbesprechung-Miro-d



miro

2021-12-11-Projekt-Besprechung-Klassenbeschreibung

Abbildung 14: 2021-12-11-Projekt-Besprechung-Klassenbeschreibung

Rahmenbedingungen Kampf:

- 3 Klassen (Krieger, Priester, Zauberer)
 - Krieger: Angriff / Blocken
 - Zauberer: Angriff / Schadensbuff
 - u.U. über mehrere Runden gültig
 - Priester: Angriff / Heilen
-
- Hitpoints
 - Angriffswert
 - Aggrowert

- Gegner (1 Gegner pro Szene)

- Angriff / Option: Enrage, Selbstheilung

- Hitpoints

- Angriffswert

- Rundenbasiert

- Gegner beginnt

- danach alle 3 Spieler

Abbildung 15: 11.12.2021: Projekt Besprechung: Gegenseitiges Update und Wechsel von Szenenlogik zu Kampfsystem für das RPG

Projekt-Besprechung 12.12.2021:

- Vorstellung erster Zeichnungen und Ideen zu Stats und Charakteren
- Gespräch über Dateiformate, Raster- und Vektorgrafiken.
- SVG Export und Inksacape getestet

- Ansatz verändert: Weg von Szenenlogik hin zu einem reinen Kampfsystem:

Kampfablauf:

- Begrüßungstext ausgeben ("Hoho, Ihr bekommt Prinzessin Peach nie!")
- Kampf läuft = true
- While(Kampf läuft):
 - Gegner fügt Schaden zu
 - für jeden (lebendigen) Spieler einzeln (in Reihenfolge der Slots):
 - Auswahl Aktion bestimmen (immer 3 Optionen: Schaden machen, Fähigkeit nutzen, Aussetzen [Aggro abbauen])
 - Aktion für Runde speichern
 - Aktionen durchgeführt (div. Rechnungen vorgenommen)
 - Gamelog wird fortgeschrieben
 - Aggrotabelle fortschreiben (Schaden 1:1 Aggro, Verspotten +100 Aggro)
 - stetige Prüfung ob ein HP unter 0 fällt:
 - Wenn Gegner unter 0: Abbruch: Win
 - Wenn kein Spieler mehr am leben ist: Game Over

| | HP | DMG | Fähigkeit |
|-----------|------|--------|---|
| Gegner: | 1500 | 80-120 | |
| Krieger: | 500 | 20-40 | Verspotten (Aggro aufbauen, klingt über Runden ab) |
| Zauberer: | 200 | 60-80 | Buffen (Schaden der Gruppe für x Runden erhöhen) |
| Priester: | 300 | 30-60 | Heilen (HP der Gruppe regenerieren [auch über Runden?]) |

Levelfortschritt nach Abschluss eines Levels:

- Gegner erledigt?
- ja: HP + 5-10 %, DMG min +8%, DMG max +10%
- nein: nix (kein Game-Progress)

Backlog:

- Mehr Klassen mit anderen Fähigkeiten (Totstellen, Verblassen, ...)

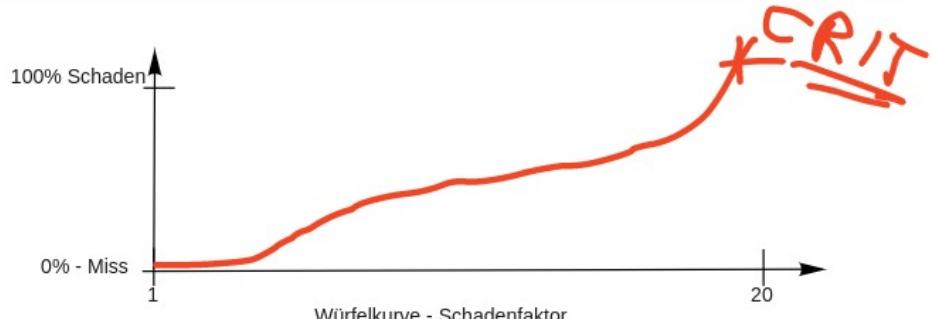


Abbildung 16: 06.01.2022: Projekt Besprechung: Ermitteln von restlichen To-Dos, Aufgabenverteilung, Meilenstein- und Terminplanung

To-Do's:

- Weitere Szenen: 5
 - 2 Multiplayer, 3 Solo
 - 1 Endszene (Endboss)
 - 1 Einstiegsszene
 - 3 Midgame-Szenen
- Inhalte:
 - Grafiken (Szenenbild + Boss)
 - Texte (Intro + Start- / Endtext für jede Szene)
 - Popup bei Szenenstart (Intro), Popup bei Szenenende (Outro)
- Name des Spiels (Setting)
- Synopsis
- Frontend (CSS-Verschönerung)
- Balancing (Werte berechnen (Angriffskraft, Leben, Boss-Angriffskraft, Fähigkeiten, EXP-Verteilung))
- Doku schreiben

Meilensteine:

Setting festlegen: 08.01.2022 (09:30 Uhr) 09.01.2022 (17:00 Uhr)

-> Grafiken erstellen

-> Balancing

-> Texte erstellen 23.01.2022

--> Zusammenbauen: 26.02.2022

Gliederung der Dokumentation festlegen: 09.01.2022

Frontend verschönern (einheitliches Design): 06.02.2022

Dokumentation schreiben

-> Zusammenbauen + Einleitung schreiben: 30.01.2022

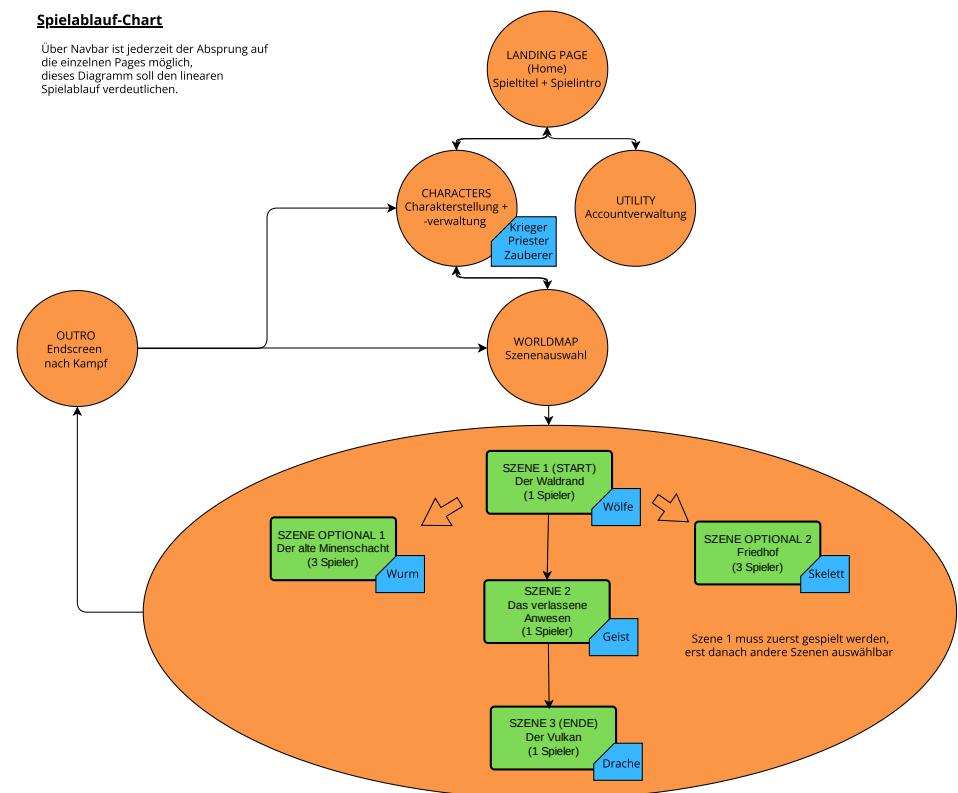
2022-01-08-Spielablauf-Chart

Abbildung 17: 08.01.2022: Diagram: Seitenaufbau und Spielablauf

Visual Paradigm Online Free Edition

Spielablauf-Chart

Über Navbar ist jederzeit der Absprung auf die einzelnen Pages möglich, dieses Diagramm soll den linearen Spielablauf verdeutlichen.



Visual Paradigm Online Free Edition

Anhang 2: Entwicklungsnotizen

Anhang 2.1: Entwicklung, Samstag 13.11.2021

Allererste Tests und Anlage der Django-App sowie hochladen des ersten Stands zu Github (Commits <https://git.io/JSq7n> und <https://git.io/JSYH4>). Eigene Notizen zum Ablauf dazu hier:

```

1 1. copy files (docker-compose, dockerfile, example.env, requirements.txt) from
   repo local
2 2. rename .env.example to .env and change settings
3 3. init django with: docker-compose run rpg django-admin startproject rpg .
4 4. edit rpg/settings.py:
5   add:
6     start:
7       import os
8       import environ
9       env = environ.Env()
10      environ.Env.read_env()
11      after 'BASE_DIR':

```

```

12     TEMPLATES_DIR = os.path.join(BASE_DIR, 'templates')
13     STATIC_DIR = os.path.join(BASE_DIR, 'static')
14     after STATIC_URL:
15         STATICFILES_DIRS = [
16             STATIC_DIR,
17         ]
18     change:
19         SECRET_KEY to:
20             SECRET_KEY = env('ENV_SECRET_KEY')
21         ALLOWED_HOSTS to:
22             ALLOWED_HOSTS = [env('ENV_ALLOWED_HOSTS')]
23         in Templates, Dirs to:
24             'DIRS': [TEMPLATES_DIR],
25         DATABASES to:
26             DATABASES = {
27                 'default': {
28                     'ENGINE': 'django.db.backends.postgresql',
29                     'NAME': env('ENV_POSTGRES_DB'),
30                     'USER': env('ENV_POSTGRES_USER'),
31                     'PASSWORD': env('ENV_POSTGRES_PASSWORD'),
32                 }
33             }
34
35 5. copy and rename .env.example also to rpg/.env (remeber to copy again @changes)
36 6. update django to new db: 'docker-compose run python manage.py makemigrations'
   and 'docker-compose run python manage.py migrate'
37 7. create superuser: docker-compose run rpg python manage.py createsuperuser
38 8. create django app: docker-compose run rpg python manage.py startapp rjh_rpg

```

Anhang 2.2: Entwicklung, Mittwoch 24.11.2021

Einbau von Grundlagen:

- Update der Django-Basis Instalition bzw. des Projektes (Commit <https://git.io/JSqbt>).
- Einbau der Benutzer-Anmeldung im Django-System insbesondere unter Nutzung von zwei Anleitungen ¹.
- Auto-Download und restart der Docker-Container bei neuen Commits im Repository auf Github per Cron-Jobjob-Script (siehe <https://git.io/JStjW>).

Anhang 2.3: Entwicklung, Donnerstag 25.11.2021

Einbau der Charaktere als Datenbank-Modell und View in Django (Commit <https://git.io/JSmOd>).

¹ Siehe <https://www.nintyzeros.com/2020/06/login-register-user%20page-in%20django.html> und <https://docs.djangoproject.com/en/3.2/topics/auth/default/#built-in-auth-forms>.

Anhang 2.4: Entwicklung, Samstag 27.11.2021

Einbau der Weltkarte bzw. Levelauswahl (Commit <https://git.io/JSmG8>).

Anhang 2.5: Entwicklung, Sonntag 28.11.2021

- Einbau einer Web-Sockets Chat-Funktion unter Nutzung einer Anleitung² (Commit <https://git.io/JSmIS> sowie folgender Commits von diesem Tag).
- (Teilweise) Installation der Entwicklungsumgebung, Docker sowie Git bei Julian und Rico sowie jeweils zwei erste Test-Commits incl. anschließendem Auto-Update des Servers (Rico: <https://git.io/JSmKV> und Julian: <https://git.io/JSmoF>).

Anhang 2.6: Entwicklung, Montag 29.11.2021

Erstellung eines Web-Sockets für eine Counter-Funktion incl. der notwendigen Anpassungen an Datenbankmodell, der Django-Reciver und -Consumer (Commit <https://git.io/JSmDa>).

Anhang 2.7: Entwicklung, Dienstag 30.11.2021

Einbau einer Lobby und einer Chatfunktion in dieser Lobby – Jeweils abhängig von der Lobby werden dynamisch Websockets geöffnet (Commits <https://git.io/JSm5n> und <https://git.io/JSm5N> sowie weitere Anpassungen und Korrekturen in den Commits vom 01.12.2021).

Anhang 2.8: Entwicklung, Samstag 04.12.2021

Flask8 als Code-Linter genutzt und einige Anpassungen entsprechend vorgenommen (Commit <https://git.io/JSmN2>).

² Siehe <https://github.com/veryacademy/YT-Django-Project-Chatroom-Getting-Started>.

Anhang 2.9: Entwicklung, Sonntag 05.12.2021

Einbau der Spielseite selbst als logischer Schritte nach der Weltkarte (als Levelauswahl) und der Lobby (Spielfindung /-erstellung) (Commits <https://git.io/JSYLS>, <https://git.io/JSYqd> und <https://git.io/JSYmV>).

Anhang 2.10: Entwicklung, Sonntag 12.12.2021:

Als Grundlage der Dokumentation eine an APA-Zietierrichtlinien angepasste Variante der LaTeX-FOM-Vorlage³ in der Git-Repository eingeführt und für das Projekt hier angepasst. Dort erfolgt nun laufen auch die Dokumentation (Commit <https://git.io/JSYOZ>).

Anhang 2.11: Entwicklung, Sonntag 19.12.2021:

An diesem Tag wurden weitere, notwenige Grundlagen für die Integration der Spiellogik eingebaut. Das insbesondere in Vorbereitung auf die kommenden Anpassungen und Entwicklungen die in der Projektbesprechung vom 11.12.2021 besprochen wurden. Konkret:

- Prüfung auf den Seiten Chars, Worldmap und Lobby ob dieser Benutzer ein aktives Spiel hat. Falls ja, wird der Benutzer auf diese Seite umgeleitet.
- Grundfunktion für das Beenden von einem Spiel eingebaut: Man kann nun per Klick im Spiel, das Spiel beenden.
- Daran anschließend eine Prüfung im laufendem Spiel, ob das Spiele beendet wurde und falls ja, Anzeige eines Endbildschirms.

Die Entwicklung der Grundlagen an diesem Tag wurde mit Fokus auf Modularisierung erledigt. Der Code der jeweiligen Funktionen wurde in einzelnen Dateien ausgelagert um Wiederverwendbarkeit und Lesbarkeit zu erhöhen.

Die zugehörigen Commits sind insbesondere: <https://git.io/JDjKI> und <https://git.io/JDjKB>

³ Siehe <https://github.com/andygrunwald/FOM-LaTeX-Template>.

Anhang 2.12: Entwicklung, Dienstag 21.12.2021:

Grundlagen des Kampfsystems entsprechend der Projekt-Besprechung vom 11.12.2021 (Abbildung 15) sollen implementiert werden.

Vorbereitungen:

- Tabelle "GamesScenesSteps" und Verknüpfungen entfernen (Commits <https://git.io/JDjKz> und <https://git.io/JDjKg>)
- Kampf-/Gamelog erzeugen: Darin werden alle Meldungen aus dem Spiel wie z.B. Kampftexte, Schaden, Aktionen, Systemmeldungen und alles andere denkbare angezeigt und gespeichert. Getrennt davon soll der Chat-Log dargestellt werden. Dazu werden in der Tabelle "Games" neue Textfelder erzeugt (Commit <https://git.io/JDj63>).
- Anzeige des Game-Logs auf der Spielseite. Schreiben von Nachrichten in das Gamelog als ersten Test des grundlegend umgestellten Seitenaufbaus: Es werden nur noch einzelne Elementinhalte per WebSocket transportiert, nicht mehr ganze HTML-Code-Blöcke (Commit <https://git.io/JyeJQ>).

Anhang 2.13: Entwicklung, Montag 27.12.2021:

Weitere Entwicklungen entsprechend der Projekt-Besprechung vom 11.12.2021 (Abbildung 15):

- Eintrag ins Gamelog zum Spielstart (Commit <https://git.io/JyBRf>).
- Rundensystem implementieren. Dazu mindestens notwendig: Lebens- und Angriffs-punkte der User-Chars sowie des Gegners.
 1. Erster Schritt: Definition des Ablaufes einer Runde als Pseudo-Code:
 - a) Gameloop-Schleife: [round-state]
 - b) Aktion von Gegner ausführen (Schaden) [100]
 - c) Prüfen ob User-Char tod ist (HP < 1 = Dead-Flag: True) [200]
 - d) Prüfen wie viele User-Chars noch leben (n < 1 = Gameover-Flag: True, break-Gameloop-Schleife) [300]

- e) Aktionen der User-Chars aufnehmen (Entscheidung für nächste Aktion von jedem Spieler annehmen + wegspeichern) [400]
- f) Alle Aktionen der User-Chars ausführen (Aktionen laden und ausführen: Schaden, Aktion, Passen) [500]
- g) Nach jedem Spieler, prüfen ob Gegner besiegt wurde ($HP < 1$ = Win-Flag: True, break-Gameloop-Schleife) [immernoch 500]
- h) Rundencounter +1 [600]
- i) Gameloop-Schleife nächster Durchlauf [700, zurück zu 100]...

Steuerung über "round-state" Hilfsvariable, gespeichert in Games-Tabelle (Default=0, Gameover=990, Win=995). Da der Aufruf der Spiele-Logik über den WebSocket-Heartbeat der Spieler erfolgt, müssen die Arbeitsschritte sehr kleinteilig sein und diese laufend in kleinen (kleinsten?) Schritten weggepeichert werden. Möglicherweise ergibt sich ein Sync-Problem (Commit <https://git.io/JyRml>)

- 2. Zweiter Schritt: HP und AP bei User-Chars implementieren, damit AP aus "round-state 100: Gegner führt Schaden aus" durchgeführt werden kann (Commit <https://git.io/JyRWD>).

Anhang 2.14: Entwicklung, Dienstag 28.12.2021:

Fortführung der Entwicklungen vom Vortag. Hier insbesondere nun die Implementation der aller Funktionen der Runden- bzw. Spiellogik:

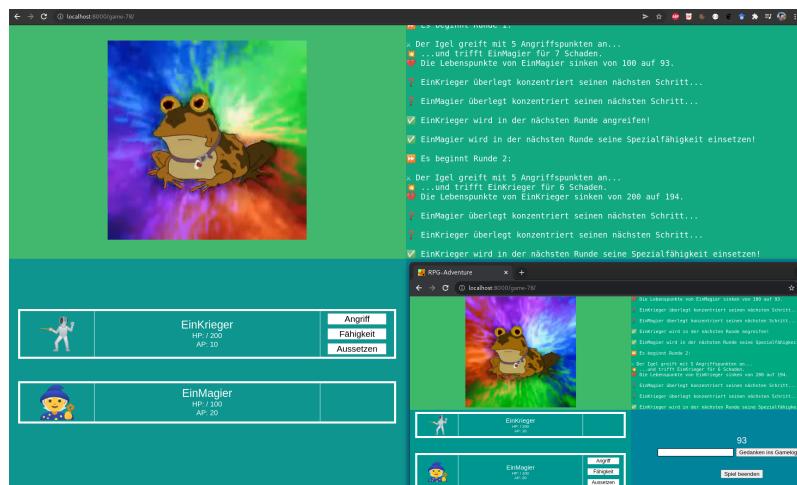
- Auswahl eines zufälligen, lebenden Spielercharakters und zufügen von Schaden durch den Gegener. Außerdem Erweiterung Runden-/Gameloop und Fortschreiben des Game-Logs (Commit <https://git.io/JygDz>).
- Nächster Rundenschritt 200: Prüfen ob Spieler gestorben sind und Meldung im Game-Log ausgeben falls in aktueller Runde gestorben (Commit <https://git.io/Jya3H>).

Anhang 2.15: Entwicklung, Mittwoch 29.12.2021:

Weitere Implementation von Rundenlogik:

- Rundenschritt 300: Feststellen ob alle Spieler verstorben sind, falls ja in Spielende springen (Commit <https://git.io/Jy1Lu>).
- Tests ergaben Probleme beim Spiel mit mehreren Spielern. Die Rundenlogik wird dann gleichzeitig vorangetrieben. Dadurch werden manche Aktionen und Rundenschritte mehrfach ausgeführt. Ein Versuch das Problem mit einem Token (ähnlich einem Semaphor) zu lösen, brachte leider noch keinen abschließenden Erfolg. Der Singleplayer aber, geht fehlerfrei. Das Problem wird daher zurückgestellt und nun zuerst die Entwicklung weiterer Punkte fortgeführt (Commit <https://git.io/Jy1qZ>).
- Als Workaround für oben genanntes Problem im Multitplayer wurde nun eingestellt, dass immer nur der erste Spieler eines Spiels die Rundenlogik vorantreibt. Das ist etwas mehr fehleranfällig als eine korrekte Tokenlösung, wird für das Projekt hier aber vorerst ausreichend sein (Commit <https://git.io/Jy1su>).
- Umfangreiche Erweiterungen und Anpassungen für das Anzeigen der User-Chars auf der Spieldeseite, darstellen und aussteuern der Aktions-Buttons, das einsammeln der Aktionen in der Datenbank und ebenso bereits das zurücksetzen beim Rundenwechsel (Commit <https://git.io/JyDlp>).

Abbildung 18: 29.12.2021: Bildschirmfoto zum Entwicklungsstand mit Rundenstatus 400.



- Die Chatfunktion wurde implementiert. Hier jedoch abweichend vom Plan direkt als Meldungen im Game-Log und nicht in einem gesonderten Chat-Log und Chat-Fenster. Julian war damit im kurzen Teams-Gespräch gestern einverstanden. Den Aufbau der Game-Seite entsprechend angepasst und dem Game-Log deutlich mehr Raum eingeräumt. Außerdem weitere Anpassungen am Design und Style (Commit <https://git.io/JyDRK>).

Anhang 2.16: Entwicklung, Donnerstag 30.12.2021:

Abschließender Schritt in der Implementation der Rundenlogik:

- Die Schadensfunktion für Spieler wurde eingebaut: Damit kann dem Gegner nun Schaden zugefügt werden. Außerdem Prüfung auf Tod des Gegners incl. entsprechender Medlung (Commit <https://git.io/Jy9E7>).

Offen sind nun als nächste Schritte noch:

1. Fähigkeiten der Spieler in Rundenlogik implementieren.
2. Abschlussbildschirm für Sieg und Gameover konzeptionieren und implementieren.
3. Charakterentwicklung bei Sieg implementieren.
4. Erweiterung der Infrastruktur aus Scripten, Anleitungen und des Git-Repos hin zum laden eines Default-Datensatzes (JSON-Datei) mit nutzbaren Spieldaten (insbesondere für Testzwecke und Nutzungen durch Dritte).

Anhang 2.17: Entwicklung, Freitag 31.12.2021:

Abarbeiten der zuletzt genannten, nächsten Schritte. Hier:

- Die Charakterentwicklung wird mit Erfahrungspunkten gelöst: Durch Aktionen im Spiel bzw. Kampf (Schaden bekennen, Schaden austeilten, Fähigkeiten nutzen) bekommen die entsprechenden Charaktere sofort entsprechende Erfahrungspunkte gutgeschrieben. Wird das Spiel gewonnen, werden alle von allen teilnehmenden Spielern in dieser Runde erspielten Erfahrungspunkte noch einmal verdoppelt und jedem der Spieler gutgeschrieben. Verwendet werden können diese Erfahrungspunkte dann in der Charakteransicht um damit z.B. Lebenspunkte (HP) oder Angriffspunkte (AP) zu erhöhen (Commit <https://git.io/Jy7gl>).

Anhang 2.18: Entwicklung, Samstag 01.01.2022:

Nach kurzer Abstimmung und Vorstellung meiner Ergebnisse bei Julian, letzte Anpassungen bzw. Entwicklungen:

- Ausgabe der erhaltenen XP beschleunigt: Es werden nun immer 10% (aufgrundet auf die nächste Ganzzahl) der XP ausgegeben (Commit <https://git.io/JSTIq>).
- Einbau der Charakter-Fähigkeiten (Commit <https://git.io/JSkKJ>):
 - Priester: Gruppe heilen=Alle HP erhöhen
 - Zauberer: Schaden der Gruppe erhöhen=Alle AP erhöhen
 - Krieger: Gegner blocken=AP Gegner verringern

Damit die Fähigkeiten über mehrere Runden wirken können, musste eine Hilfstabelle angelegt werden: "AbilitysToApply". Hieraus werden zum Rundenwechsel die jeweils anzuwendenden Fähigkeiten gelesen und auf die relevaten Zahlen gewirkt.

- Beispieldaten hinzugefügt und automatisches laden in die Datenbank über die local/dev-Skripte eingefügt (Commit <https://github.com/tstsrv-de/rpg/commit/1bfa99e89ddf2dbe817bdee2c870d1ddbe4f23c1>).

Anhang 2.19: Entwicklung, Sonntag 02.01.2022

Ein kurzer Anwendungstest mit einem versierten RPG-Spieler und IT-affinen Nutzer. Notizen dazu:

- Hinweis beim betreten der Lobby, dass das Spiel erst startet, wenn man einen "Platz-belegt" hat: Sofort erledigt!.
- Man versuchte die Interaktion mit dem Spiel per Textkommandos über die Chatzeile. Ineteresante Alternative zur Nutzng der Buttons: Übertragen in Ausblick(Anhang 3).
- Der erste Level ist mit über 10 Runden deutlich zu lang und muss verkürzt werden: Sofort erledigt!

Weiter wurden alle im Code enthaltenen Variablen und Konstanten in die Datenbank ausgelagert. Dazu wurde eine neue Tabelle "MyRpgConfig" angelegt, alle Werte dorthin transportiert und im Code entsprechende Abfragen sowie weitergehende Anpassungen (z.B. Schleifen x-Fach durchlaufen) vorgenommen. Auch wurden die Beispiel- bzw. nun Basisdaten für die Datenbank entsprechend erweitert (Commits <https://git.io/JSGJD> und <https://git.io/JSGJ7>).

Anhang 2.20: Entwicklung, Sonntag 09.01.2022

Nach letzter Projektbesprechung¹⁶ in der vor allem Details zum weiteren Ablauf, Meilensteinen und zeitlichen Fristen besprochen wurden, kommen nun erste konkrete Texte zum Spiel-Inhalt. Dazu wurde das Spiel erweitert: Der bisherige Introtext ergänzt um eine Begrüßung durch den Gegner, sowie 2 Varianten (Gameover und Win) zum Spielende (Commit).

Anhang 3: Ideen für Erweiterungen über diese Projektarbeit hinaus (Ausblick)

Da der Umfang dieser Projekt beschränkt ist, können nicht alle erdachten Funktionen umgesetzt werden. Einige Ideen, sollen aber nicht ungenannt bleiben:

1. Würfel bei Schaden bzw. Angriff implementieren.
2. Aggrotabelle und verbundene Funktionen implementieren.
3. Inaktive und getrennte Spieler bzw. Nutzer aus den Chatfunktionen der Weltkarte und Lobby entfernen. Ggf. auch einen Timer für das Spiel anlegen der anderen Spielern anzeigt, wenn ein Spieler inaktiv bzw. getrennt vom Server ist.
4. E-Mail Funktion von Django konfigurieren, damit das Zurücksetzen von Passwörtern u.A. möglich wird.

Literaturverzeichnis

heise online, developer, Matthias Wessendorf (2011). *WebSocket: Annäherung an Echtzeit im Web*. Verfügbar unter <https://heise.de/-1260189> [15. 06. 2011].

Ehrenwörtliche Erklärung

Hiermit versichere ich, dass die vorliegende Arbeit von mir selbstständig und ohne unerlaubte Hilfe angefertigt worden ist, insbesondere dass ich alle Stellen, die wörtlich oder annähernd wörtlich aus Veröffentlichungen entnommen sind, durch Zitate als solche gekennzeichnet habe. Weiterhin erkläre ich, dass die Arbeit in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde/Prüfungsstelle vorgelegen hat. Ich erkläre mich damit **nicht einverstanden**, dass die Arbeit der Öffentlichkeit zugänglich gemacht wird. Ich erkläre mich damit einverstanden, dass die Digitalversion dieser Arbeit zwecks Plagiatsprüfung auf die Server externer Anbieter hochgeladen werden darf. Die Plagiatsprüfung stellt keine Zurverfügungstellung für die Öffentlichkeit dar.

Siegen, 29.1.2022

(Ort, Datum)



(Rico)



(Henning)



(Julian)