

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: 2. Januar 2022

Inhaltsverzeichnis

1 Aufgabenbeschreibung	1
1.1 Was ist das Ziel der Projektarbeit?	1
1.2 Worin bestehen die (wahrscheinlichen) Herausforderungen? (allg. technisch und auch persönlich)	1
2 Anforderungen	2
2.1 Welche Techniken/ Technologien sollen eingesetzt werden, um die Aufgabe zu lösen/ realisieren?	2
2.2 Warum sollen gerade diese eingesetzt werden?	2
2.3 Gibt es besondere Anforderungen? (technisch, Benutzer, sonstige)	2
3 Herangehensweise	3
3.1 Wie soll das Ziel erreicht werden (Vorgehen, Architektur)	3
4 Vorstellung des Ergebnisses	4
5 Reflektion	5
Anhang	6

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

1 Aufgabenbeschreibung

1.1 Was ist das Ziel der Projektarbeit?

Beispielinhalt und Texte...

1.2 Worin bestehen die (wahrscheinlichen) Herausforderungen? (allg. technisch und auch persönlich)

Beispielinhalt und Texte...

2 Anforderungen

Beispielinhalt und Texte...

2.1 Welche Techniken/ Technologien sollen eingesetzt werden, um die Aufgabe zu lösen/ realisieren?

Beispielinhalt und Texte...

2.2 Warum sollen gerade diese eingesetzt werden?

Beispielinhalt und Texte...

2.3 Gibt es besondere Anforderungen? (technisch, Benutzer, sonstige)

Beispielinhalt und Texte...

3 Herangehensweise

Beispielinhalt und Texte:

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

Beispielinhalt und Texte:

4 Vorstellung des Ergebnisses

Beispielinhalt und Texte...

5 Reflektion

Beispielinhalt und Texte...

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 fortlaufender, 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 1: 2021-11-23-erster-entwurf-gameloop

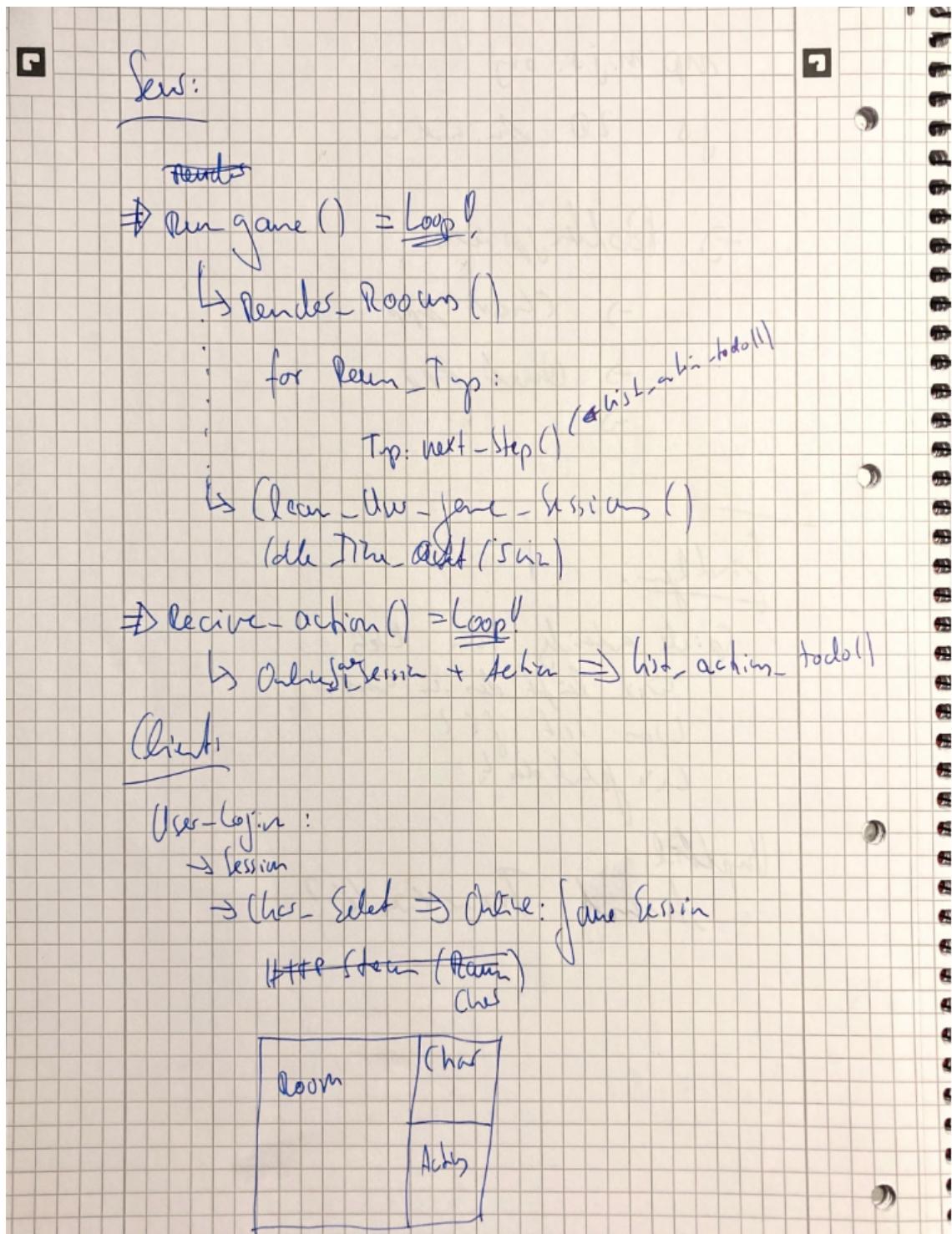


Abbildung 2: 2021-11-23-erstes-db-konzept

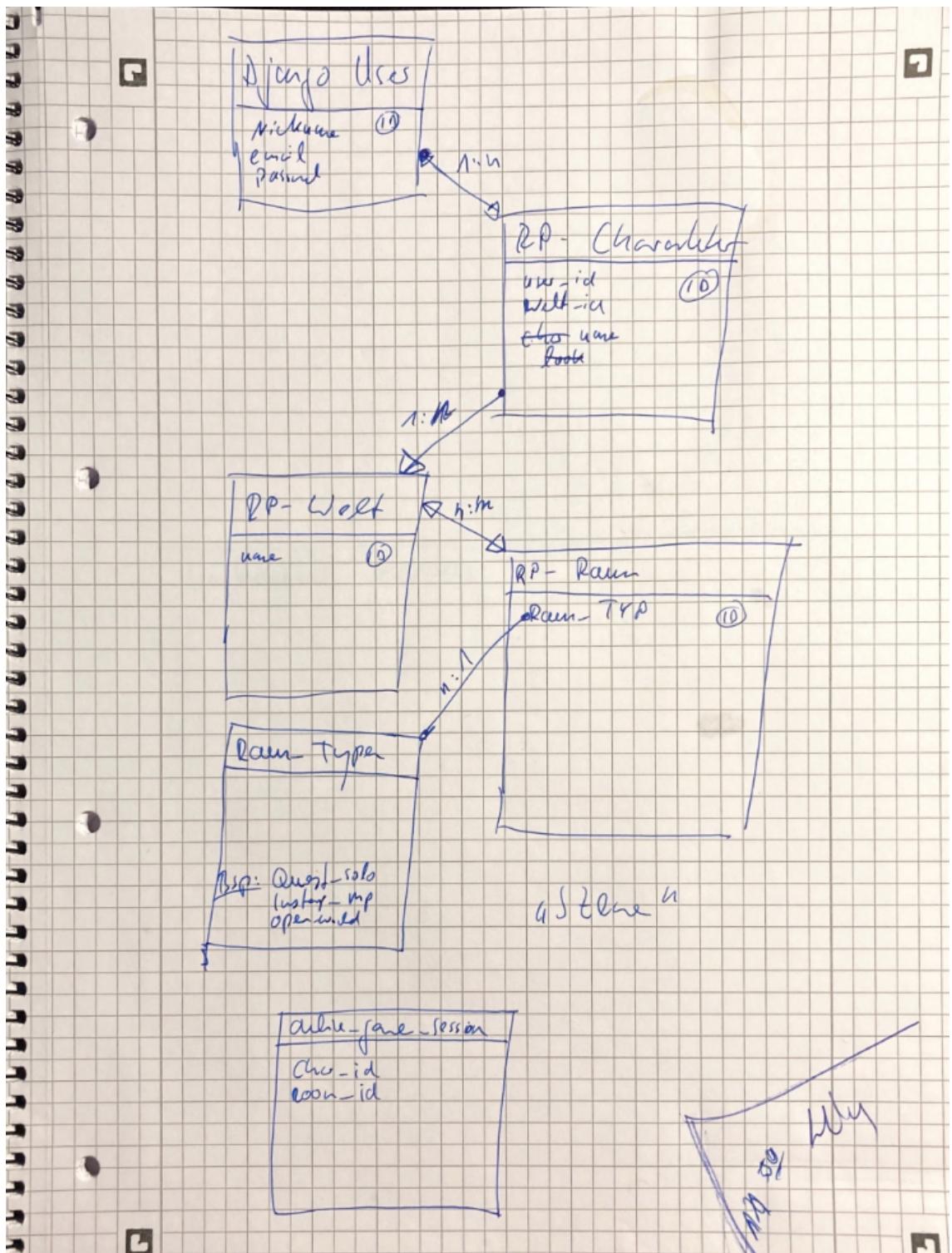
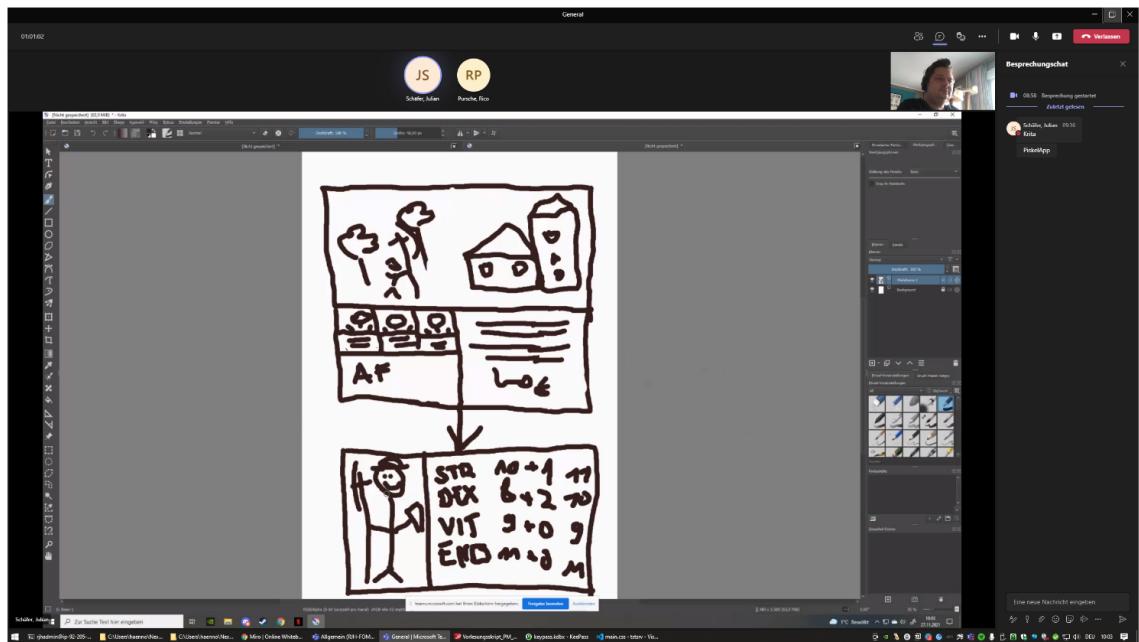


Abbildung 3: 2021-11-27-erstentwurf-ui



2021-11-27-projektskizze-1

Abbildung 4: 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 5: 2021-11-27-projektskizze-2**Aufgaben & Ergebnisse:**

- Technische Machbarkeit sicherstellen
- Technisches Grundgerüst bauen
- Inhaltliches Konzept erstellen
 - Setting
 - Grafikstil / Leveledesign
 - 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 !!!

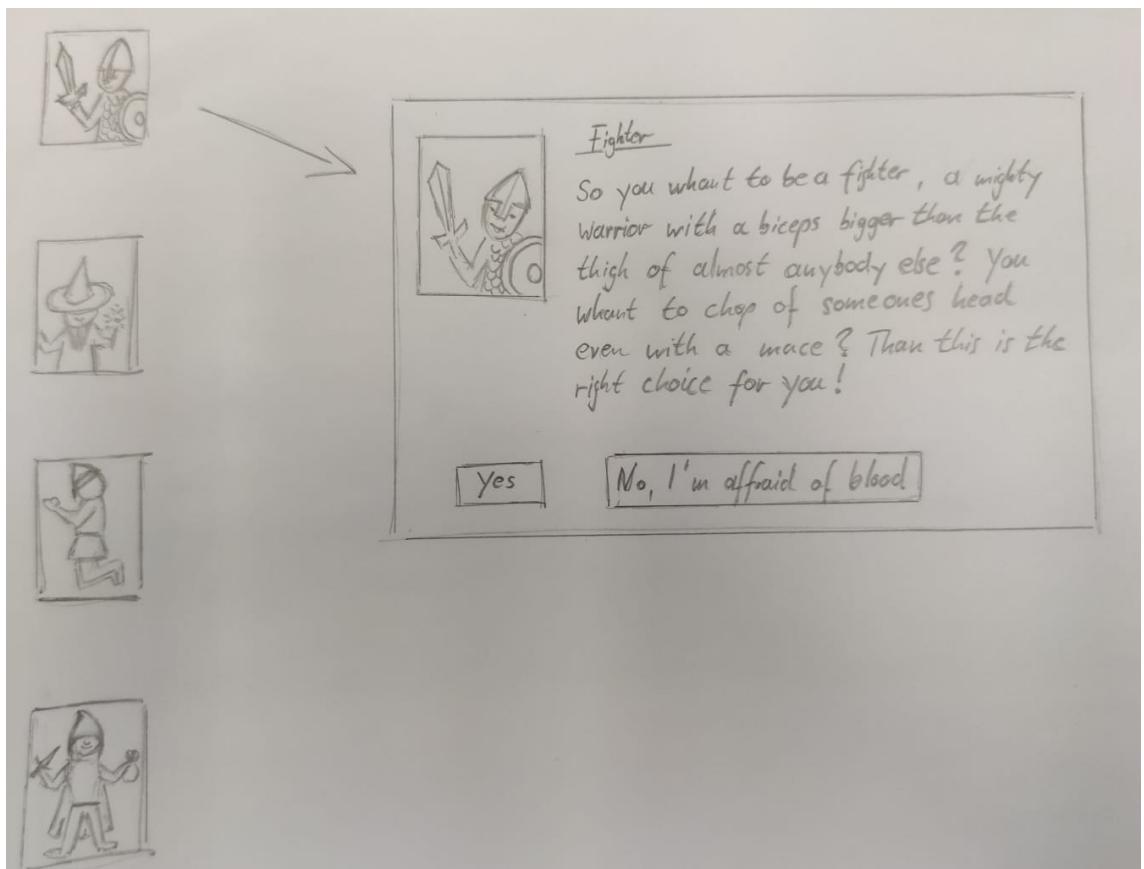
Auftraggeber:

- Daniel Bitzer / FOM

Aufwand in Personentagen:

<hier Meilensteine einsetzen>

2021-11-29-Entwurf-Klassen-Ui

Abbildung 6: 2021-11-29-Entwurf-Klassen-Ui

2021-11-30-Entwurf-Lobby-Logik

Abbildung 7: 2021-11-30-Entwurf-Lobby-Logik

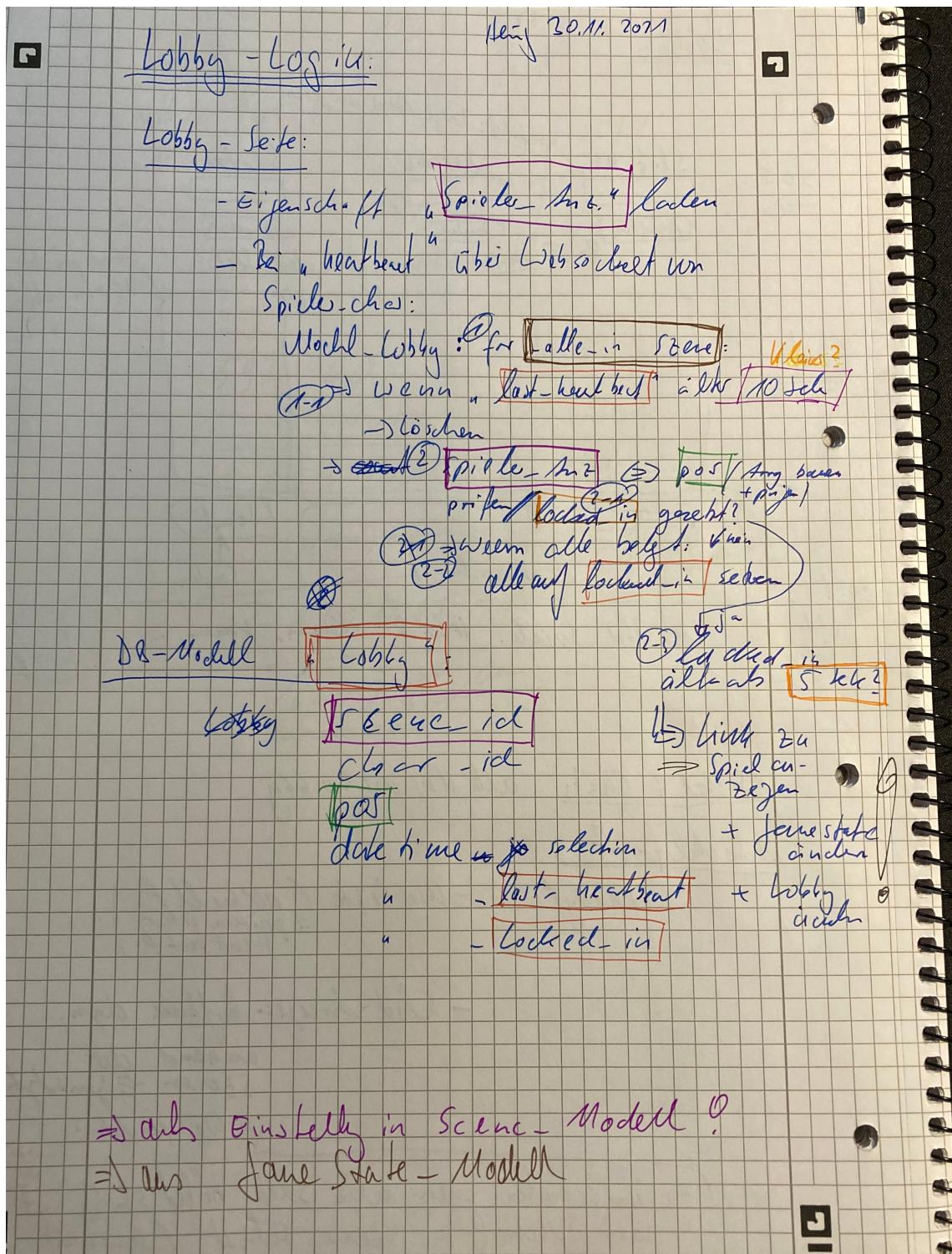


Abbildung 8: 2021-11-30-Entwurf-Lobby-UI

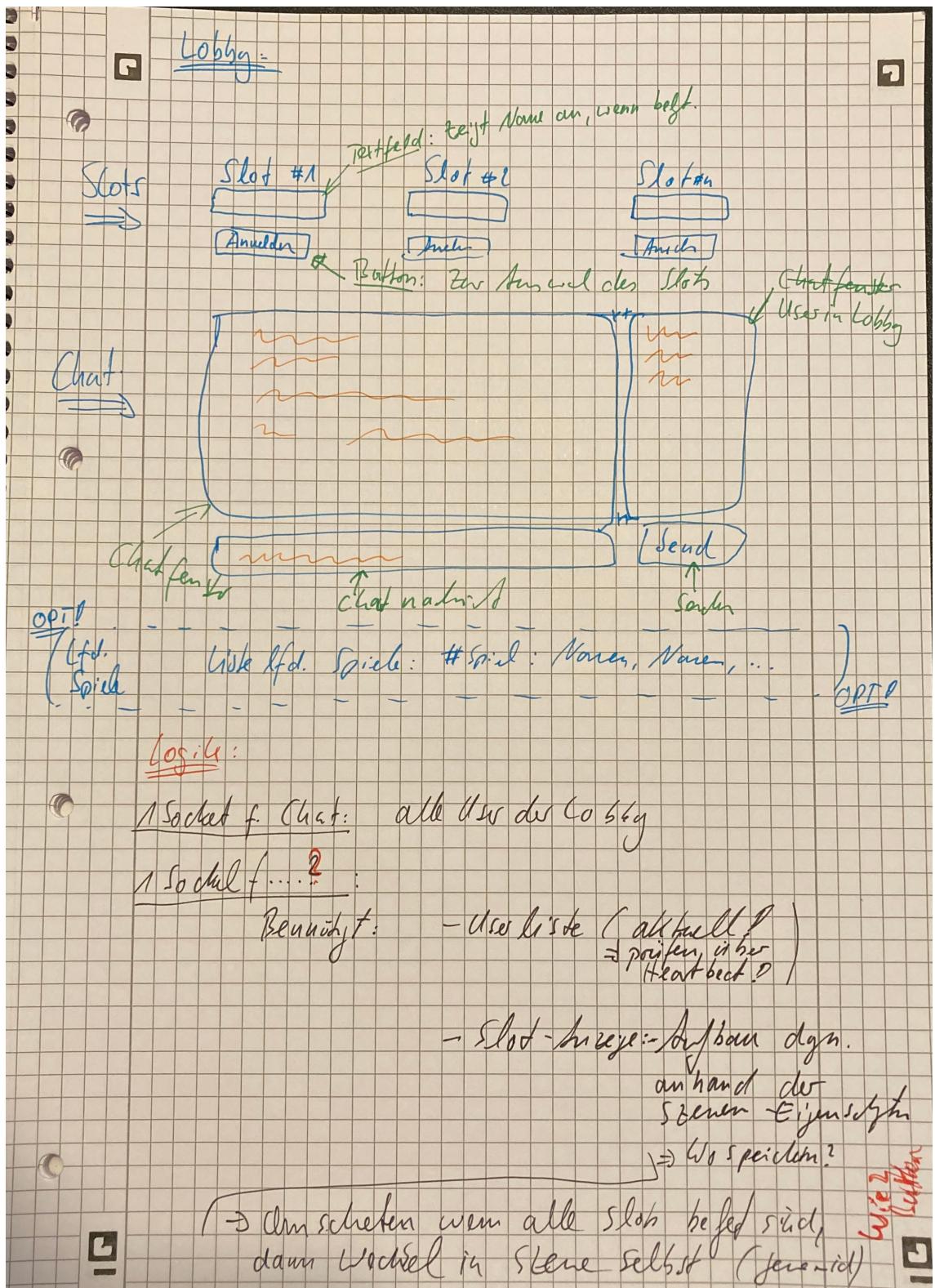


Abbildung 9: 2021-12-02-Countdown-Logik

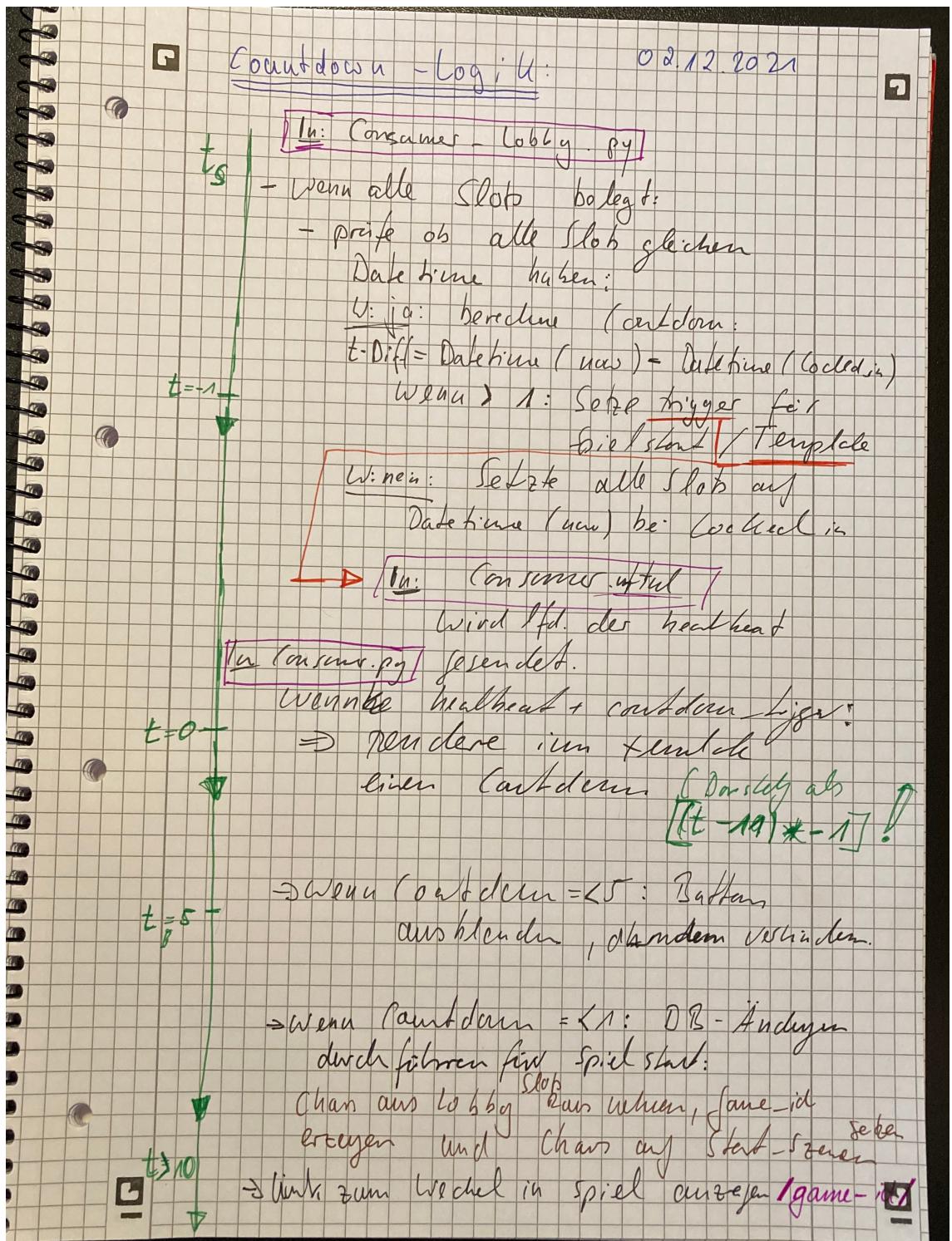


Abbildung 10: 2021-12-05-Projektbesprechung-Miro-b

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:

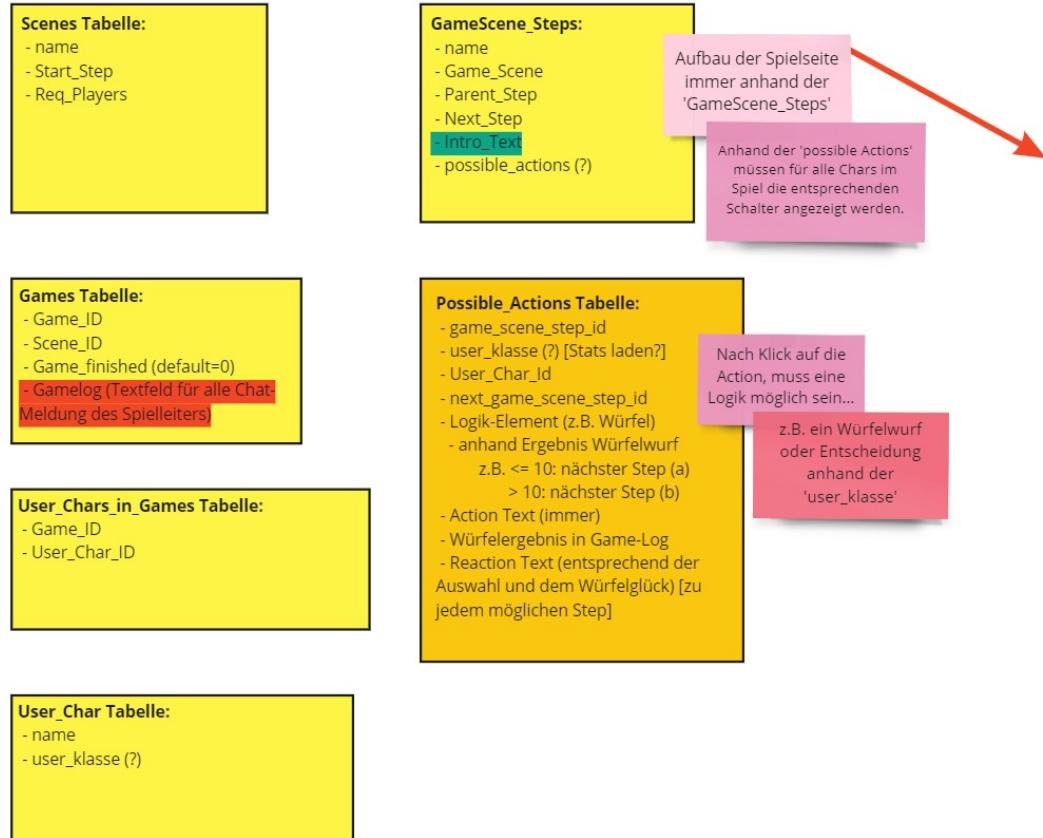
- Minimumwerte der Charaktere festlegen
- Klassen ausarbeiten
 - Infotexte etc.
- wie am Besten Kampfmechanik mit Würfelwurf abbilden

- 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):

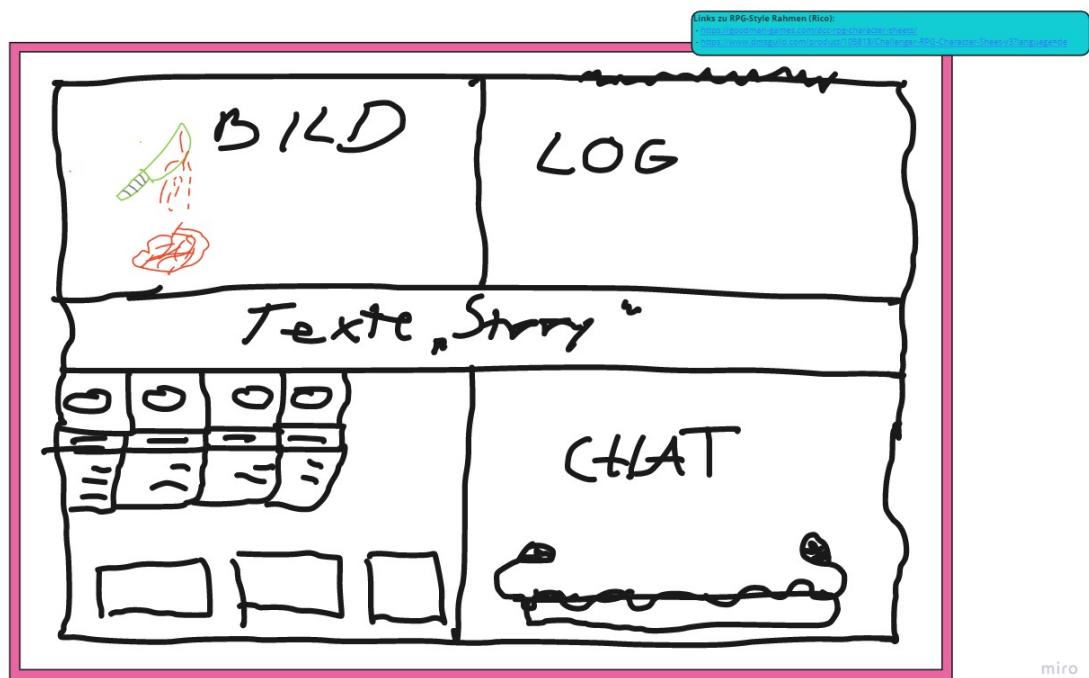
Abbildung 11: 2021-12-05-Projektbesprechung-Miro-c



miro

2021-12-05-Projektbesprechung-Miro-d

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



2021-12-11-Projekt-Besprechung-Klassenbeschreibung

Abbildung 13: 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 14: 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

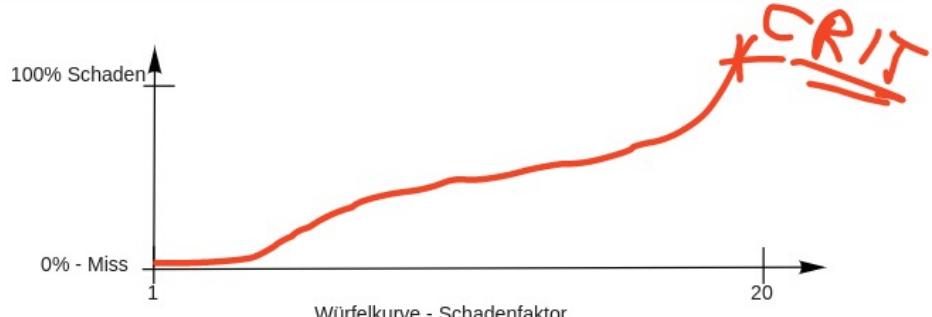
	HP	DMG	Fähigkeit
Gegner:	1500	80-120	
Krieger:	500	20-40	Verspotten (Aggro aufbauen, klingt über Runden ab)
Zauberer:	200	60-80	Buffen (Schaden der Gruppe für x Runden erhöhen)
Priester:	300	30-60	Heilen (HP der Gruppe regenerieren [auch über Runden?])

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, ...)



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 Installation bzw. des Projektes (Commit <https://git.io/JSqbt>).
- Einbau der Benutzer-Anmeldung im Django-System insbesondere unter Nutzung von zwei Anleitungen¹.
- 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>).

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-Receiver und -Consumer (Commit <https://git.io/JSmDa>).

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>.

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

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>).

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.

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

- Daran anschließend eine Prüfung im laufendem Spiel, ob das Spiel 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>

Anhang 2.12: Entwicklung, Dienstag 21.12.2021:

Grundlagen des Kampfsystems entsprechend der Projekt-Besprechung vom 11.12.2021 (Abbildung 14) 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" zwei neue Textfelder erzeugt (Commit <https://git.io/JDj63>).
- Anzeige des Game-Logs auf der Webseite. 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 14):

- Eintrag ins Gamelog zum Spielstart (Commit <https://git.io/JyBRf>).
- Rundensystem implementieren. Dazu mindestens notwendig: Lebens- und Angriffspunkte 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>).

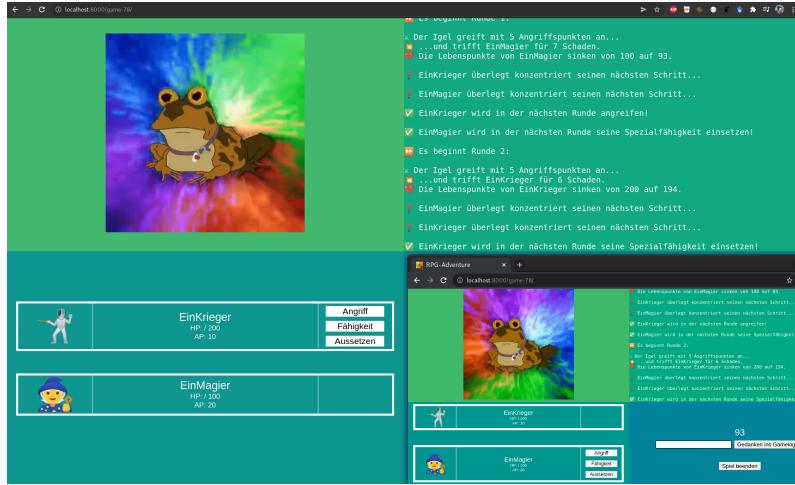
- Nächster Rundenschritt 200: Prüfen ob Spieler gestorben sind und Meldung im Game-Log ausgeben falls in aktueller Runde gestorben (Commit <https://git.io/Jya3H>).

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 Multplayer 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 Spielesseite, 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 15: 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:

- Fähigkeiten der Spieler in Rundenlogik implementieren.
- Abschlussbildschirm für Sieg und Gameover konzeptionieren und implementieren.
- Charakterentwicklung bei Sieg implementieren.
- 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 bekommen, Schaden austeilern, 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 relevanten 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. Interessante Alternative zur Nutzung der Buttons: Übertragen in Ausblick (Anhang 4).
- 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 3: Notizen für Reflektion

Anhang 3.1: 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 dadurch notwendige Konzeptionierung und Detailplanung entsteht ist nicht zu unterschätzen. Demgegenüber bei einem chaotischen Spielablauf lediglich das Handling der Events.

Anhang 3.2: Kleinteilige Aufgabenpakete

In der Entwicklung eine überschaubare Anzahl an kleinteiligen bzw. Teilufgaben vor sich zu haben, empfinde ich als sehr hilfreich. Man hat damit einen Überblick über die

Arbeit der nächsten Tage. Bei der Entwicklung der Rundenlogik hatte ich durch die Kenntniss um die nächsten Runden-Schritte hier einen guten Überblick. Der Abschluss jeder einzelnen Teil-Aufgabe sorgte da laufend für positive Motivation. Ich vermute, dass dieser Effekt sehr ähnlich bei den agilen Methoden zum tragen kommt und zur Motivation genutzt wird.

Anhang 3.3: Stichwortsammlung für Refektion

- Projektaufteilung: So abstimmen, dass verschiedene Arbeitsschritte gleichzeitig von verschiedenen Personen bearbeitet werden können.
- Lernkurve und Codequalität: Eigentlich müsste man am Ende eines Projektes stets noch einmal von Vorne anfangen. Allein um alles zu korrigieren bzw. anzupassen, was im Projektverlauf gelernt wurde.

Anhang 4: 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.

Ehrenwörtliche Erklärung

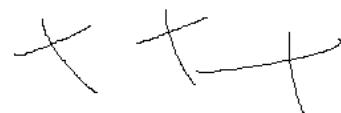
Hiermit versichere ich, dass die vorliegende Arbeit von mir selbstständig und ohne unerlaubte Hilfe angefertigt worden ist, insbesondere dass ich alle Stellen, die wörtlich oder annähernd wörtlich aus Veröffentlichungen entnommen sind, durch Zitate als solche gekennzeichnet habe. Weiterhin erkläre ich, 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, 2.1.2022

(Ort, Datum)



(Rico)



(Henning)



(Julian)