



Game Jam 2019
Dokumentation
Modul Game Development

Betreuende Professoren
Prof. Christoph Müller
Alexander Scheurer

Hochschule Furtwangen Fakultät DM
Sommersemester 2019
26.7.2019

Marko Fehrenbach
Nicolas Muster
Florian Neuweiler
Tobias Kuhn
Anna Liebermann

Inhaltsverzeichnis

1. Aufgabenstellung	3
2. Konzeptionsphase	3
2.1 Brainstorming	3
2.2 6-3-5 Methode	5
2.3 Themeneingrenzung & Auswahl	6
2.4 Konzeptentwicklung	7
2.5 Moodboards	8
2.6 Vorbereitung Modellierung, Programmierung, User Interface und Task-Board	9
3. Spielkonzept	10
3.1 Name, Genre, Plattform & Zielgruppe	10
3.2 Einführung des Spiels	11
3.3 Leitbild	13
3.4 Storyline & Charaktere	13
3.5 Spielablauf	14
3.6 Leveldesign & Spielumfang	15
3.7 Setting	16
3.8 Gameplay und Spielmechanik	17
3.9 Because Games Matter	18
4. Ästhetik des Spiels	19
4.2 Styleguide	21
4.2.1 Schrift	21
4.2.2 Elemente	21
4.2.3 Charakter	23
4.3 Logodesign	23
5. Interface-Design	25
5.1 Informationsarchitektur	25
5.2 Wireframes	25
5.3 Gestaltung	28
5.3.1 Moodboards	28
5.3.2 Screendesign	29
6. Sound-Design	32
7. Modellierung & Animation	33
7.1 Concept Art	33
7.2 Character	35
7.3 Animation	38
7.4 Items	38
7.5 Environment	40

8. Technische Umsetzung	41
8.1 Hardware	42
8.2 Unity	42
8.3 Vuforia	45
8.4 Photon	45
8.5 Programmierung	46
8.6 Umgebung	47
8.7 Beleuchtung	47
8.8 Post-Processing	48
8.9 Versionskontrolle und Kollaboration	48
9. Projektmanagement	49
10. Game-Jam Reflexion	50
11. Abbildungsverzeichnis	51
12. Anhang	53
12.1 GitHub-Link	53
12.2 Selbstdokumentation Anna Liebermann	53
12.3 Selbstdokumentation Florian Neuweiler	54
12.4 Selbstdokumentation Marko Fehrenbach	55
12.5 Selbstdokumentation Nicolas Muster	57
12.6 Selbstdokumentation Tobias Kuhn	58

1. Aufgabenstellung

Der Master-Game-Jam des Sommersemesters 2019 trug das Thema “Infektion” und brachte mehrere Teilaufgaben mit sich, welche innerhalb von vier Tagen in einem Game-Design Dokument mit Selbstdokumentationen aller Teilnehmer und einem lauffähigen, digitalen Prototypen umgesetzt werden sollen.

Die Teilbedingungen der Aufgabe setzten sich aus folgenden Punkten zusammen:

- “Location-Attractor-Game”
- Netzwerkfähigkeit
- Entwicklung eines Gamedesigns
- Implementierung eines lauffähigen Prototypen auf einem Mobilgerät

Der Zeitraum für die Umsetzung des Projektes war zwischen dem 22.07.2019 und dem 26.07.2019 angelegt, an welchem die Demonstrationen des Prototyps stattfanden.

2. Konzeptionsphase

Die Entwicklung eines passenden Konzepts wurde am ersten Tag mit verschiedenen Kreativ-Methoden durchgeführt und verfeinert. Wie es oft üblich ist, wurden zunächst in einer Brainstorming-Phase grobe Kategorien festgelegt, welche anschließend mit einer 6-3-5 Methode zu ersten Spielkonzepten führte. In mehreren kleinen internen Votings wurde ein passendes Konzept ausgewählt, welches abschließend mit der Erstellungs verschiedener Mood-Boards und weiteren Besprechungen ausgearbeitet wurde. Der letzte Schritt vor der Produktion des Prototypen war die Erstellung eines Task-Boards mit einer geschickten Rollenverteilung der Teilnehmer.

2.1 Brainstorming

Im ersten Teil der Brainstorming-Phase wurden Begriffe auf einem Pin-Board gesammelt, die allen Teilnehmern spontan innerhalb von 15 Minuten eingefallen sind,

um einen groben Überblick des Themenbereiches zu erstellen. Darauf folgend wurden diese Begriffe in Kategorien eingeteilt, um erste Schlüsse für die Konzeption eines möglichen Spiels zu ziehen.



Abb. 1: Brainstorming Schritt 1



Abb. 2: Brainstorming Schritt 2

2.2 6-3-5 Methode

Die 6-3-5 Methode, oder in diesem Fall eher 5-3-4 Methode wurde als Brainwriting-Kreativtechnik eingesetzt, um oberflächliche Ideen aller Teilnehmer gegenüberzustellen und diesen mögliche Erweiterungen vorzuschlagen. Jedes Gruppenmitglied notiert drei verschiedene Konzept-Ansätze, welche in vier Runden von allen Gruppenmitgliedern mit passenden Vorschlägen ergänzt werden. Innerhalb dieser Runden entstanden hierbei 15 verschiedene Ansätze möglicher Konzepte mit passenden Erweiterungen und Ideen der anderen Gruppenmitglieder. Ziel dieser Technik war es, einen großen Pool an Vorschlägen zu generieren, mit dem in weiteren Schritten ein passendes Konzept entstehen kann.

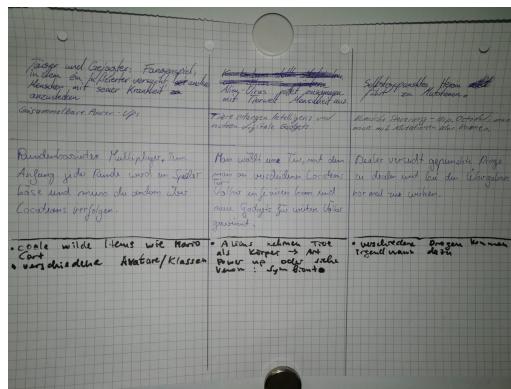


Abb. 3: Konzeptblatt Teil 1

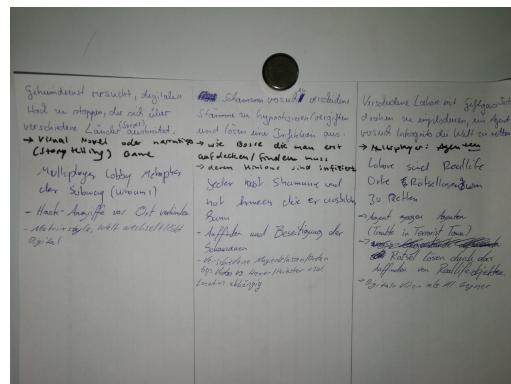


Abb. 4: Konzeptblatt Teil 2

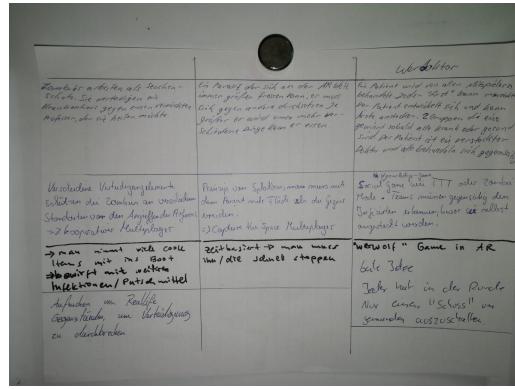


Abb. 5: Konzeptblatt Teil 3

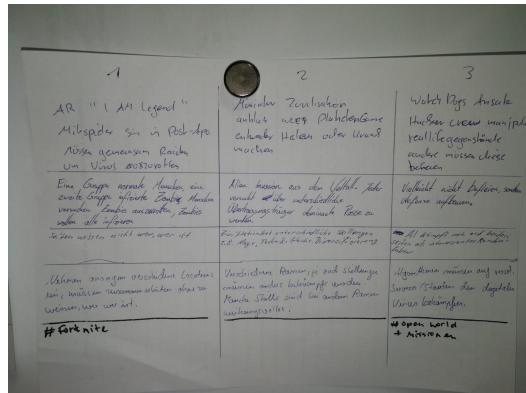


Abb. 6: Konzeptblatt Teil 4

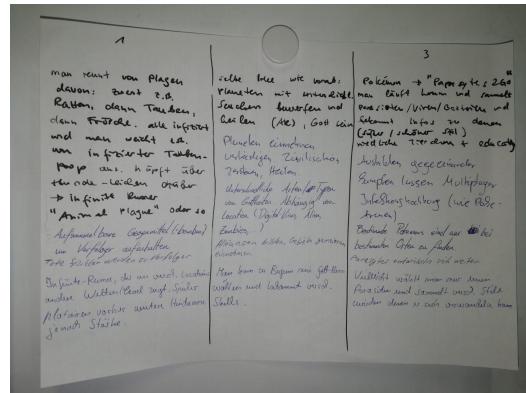


Abb. 7: Konzeptblatt Teil 5

2.3 Themeneingrenzung & Auswahl

Zu den zuvor entstandenen Konzept-Ideen wurden um 12:45 Uhr Vor- und Nachteile besprochen, aussortiert und für eine Wahl über zwei Runden vorbereitet. Hier konnte jedes Teammitglied abstimmen und neutral den jeweiligen Favoriten auswählen. Konzepte, welche zur Auswahl standen, waren folgende:

- Parasite-2-Go (Verschiedene Parasiten sammeln und trainieren)
- Parasit AR-Welt (Ein ständig wachsender Parasit, der sein Umfeld infiziert)
- Zombies als Seuchenschutz (Zombies verteidigen ihr eigenes Werk “Infektion” vor einem verrückten Professor)
- Schamanen und Zauberer (Schamanen und Zauberer “infizieren” verschiedene Stämme und steuern diese fern)
- Geheimdienst Hackerangriffe (Geheimdienst versucht Hackerangriffe auf der ganzen Welt zu stoppen)
- Wer-Doktor (Rundenbasiertes Spiel, bei dem zu Beginn jeder Runde eine geheime infizierte Person gegen die anderen Spieler antritt)

Schlussendlich wurde unter einer leichten Änderung des ursprünglichen Vorschlag die Parasiten AR-Welt ausgewählt und für die Entwicklung des Gesamtkonzepts vorbereitet.

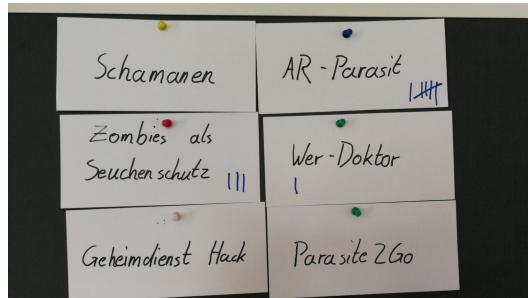


Abb. 8: Finale Auswahl eines Konzepts

2.4 Konzeptentwicklung

Um 13:30 Uhr wurde das ausgewählte Konzept im Detail besprochen, um einen fließenden Übergang zur Produktionsphase zu schaffen. Besonderer Wert wurde hierbei auf den Spielspaß, Möglichkeiten der technischen Realisierung, Gameplay und den Bezug zu den gestellten Anforderungen gelegt.

Kurz zusammengefasst sollen an verschiedenen Orten Viren eine zu ihrem Charakter passenden Welt unter Verwendung von Augmented Reality infizieren. Die Spieler

können sich an diesem Ort im Kampf gegen den Virus verbünden, um die Welten zu beschützen.

Näheres zum fertigen Konzept und den Begründungen dahinter werden im Spielkonzept besprochen.

2.5 Moodboards

Um allen Gruppenmitgliedern einen Einblick in mögliche Formen der visuellen Gestaltung zu geben, wurden um 14:20 Uhr verschiedene Moodboards erstellt und im Anschluss am Beamer besprochen und diskutiert. Wie im folgenden zu sehen ist, ähneln sich die Ergebnisse, was sich positiv auf das weitere Vorgehen auswirkte.

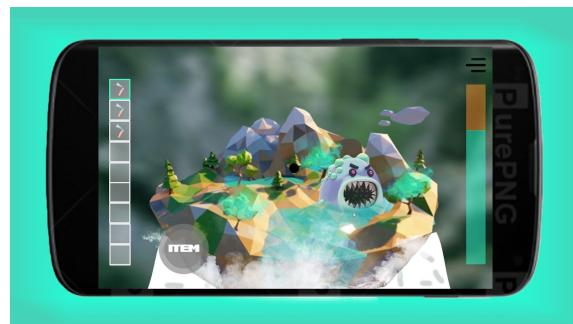


Abb. 9: Moodboard Nr. 1

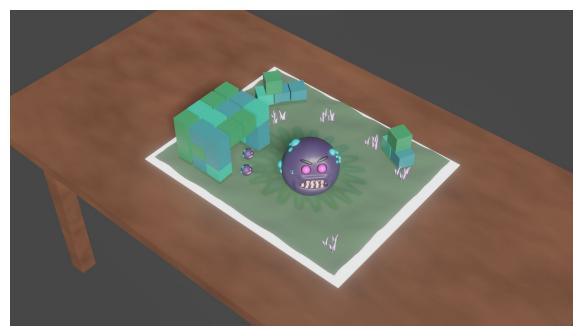


Abb. 10: Moodboard Nr. 2

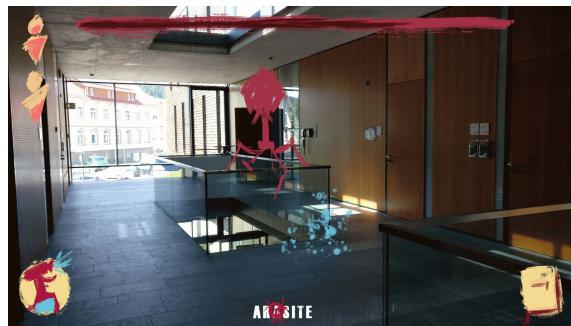


Abb. 11: Moodboard Nr. 3

2.6 Vorbereitung Modellierung, Programmierung, User Interface und Task-Board

Bevor es um 17:30 Uhr an die Aufgabenverteilung ging, wurden schon erste Einzelheiten für die Modellierung, die Programmierung und das User Interface besprochen, sodass die Rollenverteilung für den Zeitraum einfach festgelegt werden konnte. Ergebnisse dieser Ansätze sind in den einzelnen Kapiteln zur Umsetzung des Konzept auffindbar.

Die Aufgabenverteilung wurde für das gesamte Projekt mit Hilfe eines Task-Boards geregelt, worauf Aufgaben mit den Namen der bearbeitenden Person je nach Status auf dem Board platziert wurden, um das gesamte Team zu jeder Zeit auf dem aktuellsten Stand zu halten.

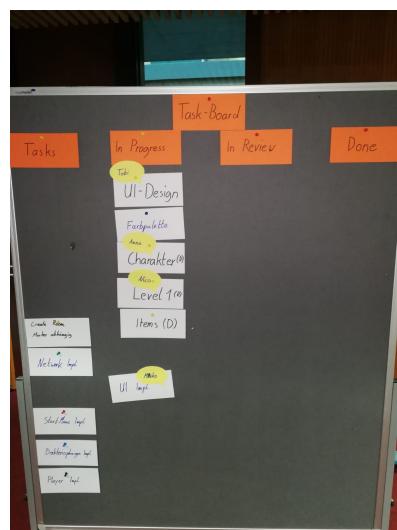


Abb. 12: Task-Board Nr. 1

3. Spielkonzept

Im Folgenden folgt die Beschreibung des Spielkonzept, welches eine Zusammenfassung, ein kurzes Leitbild, den Ablauf des gesamten Spiel und dazugehörige Begründungen beinhaltet.

3.1 Name, Genre, Plattform & Zielgruppe

Das Spiel nennt sich ARasite und setzt sich aus den Begriffen Augmented Reality und der Parasit (engl. parasite) zusammen. Parasit steht hier für den Virus im Spiel. Ein Virus kann auch ein Parasit sein, da er in Ausnahmefällen Wirtsorganismen infiziert und den Stoffwechsel des Wirts zur Vermehrung nutzt.

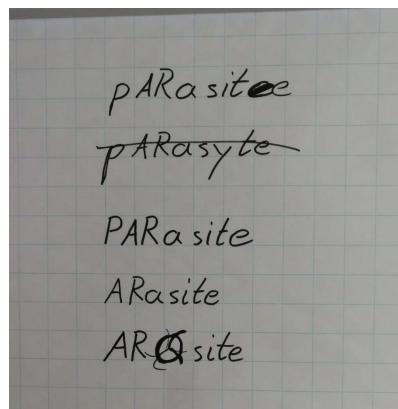


Abb. 13: Namensgebung

Das Spiel-Genre lässt sich im Bereich der Action-Rollenspiele einordnen, da eine Art Personalisierung mit dem Waffensystem und durch Rollenverteilungen während der Kämpfe stattfindet. Außerdem stehen im Spiel schnelle Aktionen und ein Click-To-Attack Prinzip mit weiteren Anhaltspunkten für den Anteil des Action-Subgenres.

ARasite ist hauptsächlich zur Nutzung auf mobilen Android-Geräten optimiert, kann theoretisch auch als iOS und UWP-Anwendung umgebaut werden. Genauere Informationen hierzu stehen im Kapitel der technischen Umsetzung.

Der Fokus liegt auf einer jungen, an Augmented Reality interessierten Zielgruppe, jedoch soll das Konzept auch weitere Altersklassen ansprechen. Das Spiel richtet sich praktisch an alle Spielertypen, aber hauptsächlich an Killer und Achiever, welche Spaß am Besiegen von Monstern und Sammeln von Belohnungen besitzen. Da Zusammenspiel gefordert wird und die Mehrspieler-Funktion ein zentrales Element darstellt, sowie verschiedene Welten, sind die Spielertypen Sozializer und Explorer ebenso angesprochen.

3.2 Einführung des Spiels

ARasite ist, wie zuvor schon erwähnt, ein mobile Augmented Reality Action-Rollenspiel mit einer Multiplayer Komponente und nutzt zusätzlich den Standort des Spielers, um die Spielinhalte anzupassen. Das Spiel findet zu einem Zeitpunkt im 21. Jahrhundert statt, in welchem Viren aller Art versuchen ihr Umfeld und alle Lebewesen großflächig zu infizieren. Ziel des Spiels ist es, an jedem Ort den dort spezifischen Virus zu besiegen, um die Welt vor dem Untergang zu beschützen. Jeder der Viren und auch die Welt, in der es sein Unwesen treibt, passt sich an den Ort an, an dem sich der Spieler befindet.

Auch die Art und Fähigkeiten der Viren variieren, sodass nicht jeder Virus auf die gleiche Art und Weise bekämpft werden kann. So kann ein Virus zum Beispiel ein digitalen oder organischen Virus darstellen. Der Prototyp zeigt ein Natur-Level mit einer Bakteriophage, in welcher er zunächst Sporen desinfizieren und im Anschluss den Virus besiegen muss.

Hierzu und für weitere Kämpfe erhält der Spieler ein Inventar mit speziellen Items und Fähigkeiten, die er gezielt im Kampf einsetzen muss. Jedes der Items hat eine bestimmte Anzahl von Nutzungen, welche er durch Item-Kisten wieder auffüllen kann, die im Spiel erscheinen. Wenn sich das Magazin oder der Füllstand des Items leert, muss dieses nachgeladen werden.

Spieler können sich am selben Ort verbünden, um gegen Viren zu kämpfen und Erfahrungspunkte und Belohnungen wie Items, Munition und Verbrauchsgegenstände zu sammeln. Da jeder Virus sein eigenes Set an Fähigkeiten, Angriffen und Eigenarten hat, müssen zur Bekämpfung auch unterschiedliche Items eingesetzt werden. Hierfür kann zu Beginn des Kampfes ein bestimmtes Set an Items in das Kampf-Inventar gelegt werden. Zusätzlich muss sich der Spieler auch mit der AR-Kamera um das Level bewegen, um sich versteckende oder schnellere Viren zu besiegen.

In jedem Level sollen spezifische Merkmale passend zum Ort eingebaut werden. Im Prototypen befinden sich im Level z.B. Kopien der HFU-Denkmalen vor dem I-Bau der Hochschule Furtwangen eingebaut, welches sich an genau diesem Ort abspielt. Im Hauptmenü des Spiels und zu Beginn des Kampfes bekommt der Spieler Informationen zu den Viren, wie sie auch im realen Leben definiert werden. Pro besiegttem „neuen“ Virus können Informationskarten über den Virus gesammelt werden, welche diese beschreiben und Stärken, Schwächen etc. auflisten. Nachdem der Virus besiegt wurde, verschwindet er für eine gewisse Zeit und taucht erneut wieder auf, um anderen Spielern die Möglichkeit zu bieten, auch an dem Kampf teilzunehmen.

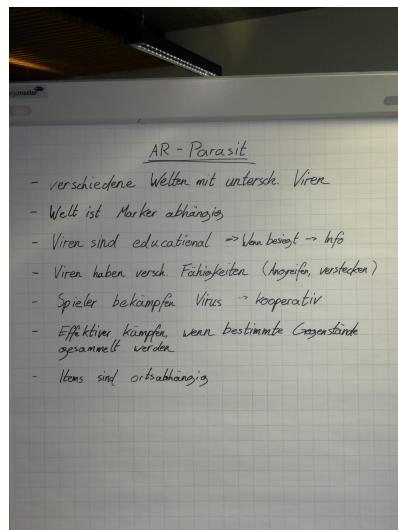


Abb. 14: Brainstorming

3.3 Leitbild

Ein mobiles Multiplayer AR-Game mit dem Namen ARasite, bei dem die Spieler unsere Welt gemeinsam an einer Location vor der Infektion durch bösartige Viren, mit einzigartigem Verhalten und Design, schützen können.

3.4 Storyline & Charaktere

Die Storyline des Spiels ist relativ simpel gehalten, da es sich um ein Action-Rollenspiel handelt, welches hauptsächlich von seinen Interaktionselementen und Kämpfen lebt. In der Story verkörpern die Spieler Virenjäger, werden jedoch visuell nicht angezeigt.

Die gegnerischen Charaktere sind die Viren, welche in verschiedenen Formen besiegt und gesammelt werden müssen. Für den Prototyp wurde ein kleines Set an verschiedenen Viren erstellt, darunter fallen zum Beispiel Backy (Bakteriophage), Xsekit (digitaler Power-Shell Virus), Prion (Hirn fressender Virus). Hierzu kann der Spieler Informationen der Charaktere in Form eines Steckbriefs aufrufen. Backy hat die Fähigkeit, Sporen auf der Welt zu pflanzen, um dadurch das Umfeld zu infizieren. Der Virus kann erst angegriffen werden, wenn diese Sporen zu großen Teilen entfernt und desinfiziert wurden. Alle Charaktere haben Gesichter und bewegen sich individuell, sodass der Charakter und das Gefühl des Spiels durchgängig ist. Der digitale Virus Xsekit sieht zum Beispiel wie eine Art Elektrokugel mit einem Glitch-Fehler aus und springt zwischen Servern hin und her, welche er mit Hackerangriffen infiziert. Prion setzt sich mit seinen Armen wie eine Baumwurzel an seinem Wirt fest und infiziert diesen bis zum bitteren Ende.

Die Virenjäger versuchen diese Charaktere zu besiegen und als Trophäen in ihrem Verzeichnis zu sammeln und stärkere Items zu finden. Jedes Level beinhaltet seine eigene Mini-Geschichte, in der die Spieler im individuellen Umfeld der Viren agieren müssen.

3.5 Spielablauf

Beim Start der Anwendung sehen die Spieler ein Startmenü. In diesem schwimmen bisher besiegte Viren als 2D Elemente, welche bei einem Klick Informationen über sich preisgeben. Zusätzlich hat der Spieler eine Schaltfläche für Einstellungen und sein Profil. In seinem Profil kann er Informationen über seinen Spielstand und Inventar finden.

Wenn ein Spieler einen Marker oder ein Objekt scannt, welches von einem Virus befallen ist, tritt er dem Kampf gegen diesen Virus bei. Im Prototyp geschieht dies über eine einfache Schaltfläche.

Ein Kampf besteht aus einer variablen Anzahl von Mitspielern, einem Virus, organisch (z.B. Bakteriophage) oder anorganisch (z.B. digital) und die an den Virus angepasste Umgebung, welche in das AR Kamerabild platziert wird. Ist die Welt zu sehr Infiziert ist der Virus zu mächtig um Schaden zu erleiden.

Die Spieler besitzen verschiedene Gegenstände, die sie innerhalb dieses Levels oder bei einem anderen Virus erhalten, um verschiedene Aktionen auszuführen. Dies erfordert Koordination. Diese Aktionen umfassen zum Beispiel Desinfizieren, Schaden hinzufügen, Fallen aufstellen uvm.

Der Virus reagiert mit entsprechenden Gegenaktionen, wie Infizieren, Schild, Eingraben, Verstecken uvm.

Zusätzlich erscheinen Item-Kisten, welche die Spieler unterstützen. Diese müssen aufgenommen werden, bevor sie auf den Boden aufkommen und selbst infiziert werden.

Der Spielstand ist für die Spieler ersichtlich, in dem man den Virus und sein Verhalten beobachtet. Er verändert sein Verhalten bei Interaktion, zum Beispiel Versteckt er sich, so dass die Spieler ihn in der Umgebung finden müssen, wenn er wenig Leben hat. Ebenso verändert sich sein Aussehen, wie zum Beispiel die Größe.

Die Spieler interagieren also mit dem KI gesteuerten Virus solange, bis dieser besiegt ist. Dies beendet eine einzelne Spielrunde. Für eine Interaktion bewegen sie sich mit ihrem mobilen Gerät durch den Raum, um ihre Ansicht zu bewegen. Aktionen werden

über Schaltflächen in einer Bedienoberfläche ausgelöst und Item-Kisten über eine Berührung auf dem Bildschirm aufgesammelt. Spieler wechseln ihren derzeitig verwendeten Gegenstand über ein Inventarsystem auf der Bedienoberfläche. Gezielt wird klassisch mit dem Fadenkreuz in der Bildschirmmitte.

Wird ein Level beendet, wird eine Zusammenfassung des Kampfergebnis, für jeden Spieler individuell, sowie Belohnungen auf dem Bildschirm angezeigt. Über eine Eingabe gelangt der Spieler wieder zurück ins Startmenü, hat dort einen Virus mehr über den er Informationen erlangen kann und rüstet sich für den nächsten Kampf. Um weiterzuspielen geht der Spieler entweder an einen anderen Marker, oder wartet an diesem auf das Wiedererscheinen des gerade bekämpften Virus.

So gerät der Spieler in eine Schleife in der er verschiedene Viren kennenlernen, die alle ein einzigartiges Erlebnis darstellen, mit einzigartiger Beute und Umgebungen, die erkundet werden können.

3.6 Leveldesign & Spielumfang

Das Spiel setzt sich aus vielen kleinen Levels mit erweiterbarem Inhalt zusammen, in denen jeweils ein Virus als Gegner besiegt werden muss. Jeder der Viren soll sich einzigartig anfühlen, sodass der Spieler mit jedem Kampf neue Eindrücke gewinnen kann.

Die Level sind auf kleine Welten reduziert, auf deren mit begrenzten Platz gekämpft wird. Jedes Level enthält explorative Elemente, welche für jede Location einzigartig sind.

In den Welten befinden sich sammelbare Gegenstände wie Item-Kisten oder Objekte, mit denen die Spieler im Kampf interagieren müssen. Im Prototypen müssen unter anderem Sporen desinfiziert werden, um das Virus verwundbar zu machen.

Die Schwierigkeit der Level passt sich an die Anzahl der Spieler an, welche an dem Kampf teilnehmen. Für die Level sollen insgesamt zwischen zwei und drei Minuten gebraucht werden und die Schwierigkeit soll nicht zu hoch sein, sodass Spieler ohne Vorerfahrung und Spieler mit Vorerfahrung nicht den Spaß verlieren.

Gefahren in jedem Level sind die vollständige Infizierung der Welt durch die Viren, oder das Versagen des Spielers durch die Mitnahme falscher Items und Verbrauchsgegenstände.

3.7 Setting

Aufgrund des Bezugs zu verschiedenen Orten, soll sich auch das Setting jeder einzelnen Welt an die Umgebung anpassen, wodurch sich der Virus, die gesamte Landschaft und notwendige Maßnahmen zur Bekämpfung der Gefahr verändern. Das Setting ändert sich demnach in jedem Level, spielt jedoch immer in unserem Zeitalter. Wenn sich der Spieler zum Beispiel an einem Ort mit viel Technik befindet, muss er gegen Viren (Beispiel: Xsekit) in einer digitalen Welt kämpfen. Hierzu und zu einem weiteren Virus Namens Prion werden im weiteren Verlauf der Arbeit auch Concept Arts, Bewegungsanimationen und Character Designs vorgestellt.

Da sich Viren und Bakterien an jedem Ort befinden und die konstante Gefahr besteht, dass sie Teile ihrer Umwelt infizieren befinden sich die Spieler in der Lage, in welcher die Viren kurz vor einer Übernahme und totalen Infizierung der gesamten Welt stehen und um jeden Preis gestoppt werden müssen.

Das Spiel soll aufgrund der Zielgruppe und der Story sowohl in einem geheimnisvollen als auch in einem komischen Setting spielen, in dem Größenverhältnisse, Farbe und Verhalten keine realistischen Werte repräsentieren, um das gewünschte Setting zu unterstützen.

Eine Eingrenzung bezüglich der Wahl aus technischer Sicht wurde in diesem Fall nicht getroffen. Hier könnten sowohl AR-Core und Wikitude genutzt werden, um die verschiedenen Level und Welten an Objekte aus verschiedenen Städten anzupassen und von der Community erweitern zu lassen. Eine Auswahl basierend auf den Standorten der Spieler mit Hilfe von GPS und der Einbindung eines Kartensystems kann ebenfalls zur ortsspezifischen Generierung der Welten genutzt werden.

Im Falle des Prototypen für den Game Jam wurde unter Verwendung von Vuforia ein Markersystem implementiert. Der Prototyp zeigt ein Level mit Bezug zur Natur, ist aber an die Farben des Farbguides angepasst, um den Stil über das gesamte Spiel beizubehalten.

3.8 Gameplay und Spielmechanik

Ziel eines jeden Spielers ist es, die Welt zu heilen und von dem Virus zu befreien. Als Motivation dient das Sammeln von Achievements wie zum Beispiel neue Waffen oder informelle Steckbriefe im Startbildschirm.

Da sich das Alter der Anwender an jüngeren Spielern orientiert, wird keine große Vorerfahrung vorausgesetzt. In unserer Anwendung gilt das Prinzip "Learning by doing". Des Weiteren wird ein verniedlichter Stil für das Gameplay angewendet, um die jüngere Generation der Spieler nicht allzu sehr vor dem Virus sowie Attacken auf diesen abzuschrecken.

Die Bedienung besteht aus Touch-Eingaben auf dem Bildschirm, dies beinhaltet das Drücken von Schaltflächen für das Wechseln/Nutzen von Items, sowie das gezielte Drücken auf ein Objekt in der Szene um es einzusammeln.

Die eigentliche Bewegung übernimmt der Spieler analog, indem er das Mobilgerät um den Marker herum bewegt, sich dem Marker nähert, oder sich entfernt.

Spielmechaniken umfassen Interaktionen zwischen den Spielern, mit der Umwelt und dem Virus, wer welche Aufgabe zu jedem Zeitpunkt übernimmt und wer sich die Item-Kisten nehmen darf.

Mit der Umgebung an welcher Stelle man den Gegenstand wie das Desinfektionsspray einsetzt, oder mit welchen Elementen der Umgebung man interagiert, werden weitere Aktionen wie z.B. eine Geröll-Lawine auslösen um den Virus zu beschädigen oder den Weg einzuschränken freigeschalten.

Man kann auch mit dem Virus selbst interagieren, in dem man ihn attackiert, festhält oder z.B. sichtbar macht. Der Virus kann jegliche denkbare Boss-Mechanik anderer Spiele innehaben, die keinen Avatar beeinflusst, da die Spieler keine Repräsentation in Form einer platzierten Spielfigur besitzen. Er könnte aber z.B. den Bildschirm verdunkeln, Hilfs-Viren spawnen uvm.

Eine wichtige Spielmechanik ist das Beobachten und Einschätzen der Situation, der Virus hat keinen Lebensbalken und das Interface allgemein gibt nur reduziert Zusatzinformationen. Die Spieler sollen anhand des Verhaltens und mittels Aufmerksamkeit einschätzen, was der Virus machen wird und wie sein Status ist. Im Prototyp ist daher der Status durch die Größe des Gegners und Schaden visualisiert und lässt den Virus kurz Rot aufleuchten. Denkbar sind allerdings auch Schadensmodelle in Form von fehlenden Gliedmaßen oder Kratzern, die man am Modell mit Blendshapes umsetzen kann, oder sich verändernde Animationen mittels Constraints, sodass der Virus anfängt sich schleppend zu bewegen.

Ein Abschnitt des Spiels hat keine Zeitbegrenzung, allerdings wird der Virus bis zu einem bestimmten Wert stärker, wenn man nicht interagiert. Der anschließende Kampf wird dann entsprechend härter. Die Schwierigkeit ist an die Anzahl der Spieler leicht angepasst, so darf nur ein kleinerer Anteil der Welt infiziert sein, bis der Virus unverwundbar wird.

3.9 Because Games Matter

Unser Spielkonzept erweitert sich über das Basisziel der Unterhaltung hinaus um einen informativen Bereich für eine zusätzliche Ebene, welche nicht zu invasiv in den Bereich der Unterhaltung des Spiels eindringt. Spieler können sich zusätzliche Informationen über die besiegten Viren im Startbildschirm aufrufen. Diese Informationen haben einen Lerneffekt, da wir hierfür Informationen über die realen Viren recherchiert haben. Einige der Informationen werden in Form von Skills grafisch

aufbereitet, wie schwer es ist sich mit dem Virus zu infizieren/ihn zu bekämpfen, wie schnell sich dieser verteilt und wie aggressiv dieser agiert.



Abb. 15: Informationen als Skills

Zusätzlich erhalten die Spieler eine kurze Beschreibung und Klassifizierung der Viren. Diese Funktionalität ist sehr Präsent und lädt dazu ein genutzt zu werden mittels kleinen Icons, und den sich bewegenden Viren auf dem Startbildschirm. Mit jedem getöteten Virus spawnt ein zusätzliches, so dass die Spieler zusätzlich auf diese Funktionalität hingewiesen werden. Für Spieler, die weniger nach Information suchen, aber sich an der Funktionalität erfreuen wollen, erhält man beim Öffnen und Schließen der Informationskarte eine Art personalisierten "Ruf". Ebenso ist in die Informationskarte eine interpretierte Darstellung des Virus integriert.

4. Ästhetik des Spiels

Die Gestaltung des Interface sollte mit Hilfe moderner Farben und Gestaltungsmethoden entstehen, sowie kinderfreundlich aber dennoch ansprechend für eine ältere Zielgruppe sein. Dazu wurden einige Inspirationen eingeholt. (Siehe 5.3.1 Moodboards)

4.1 Farben

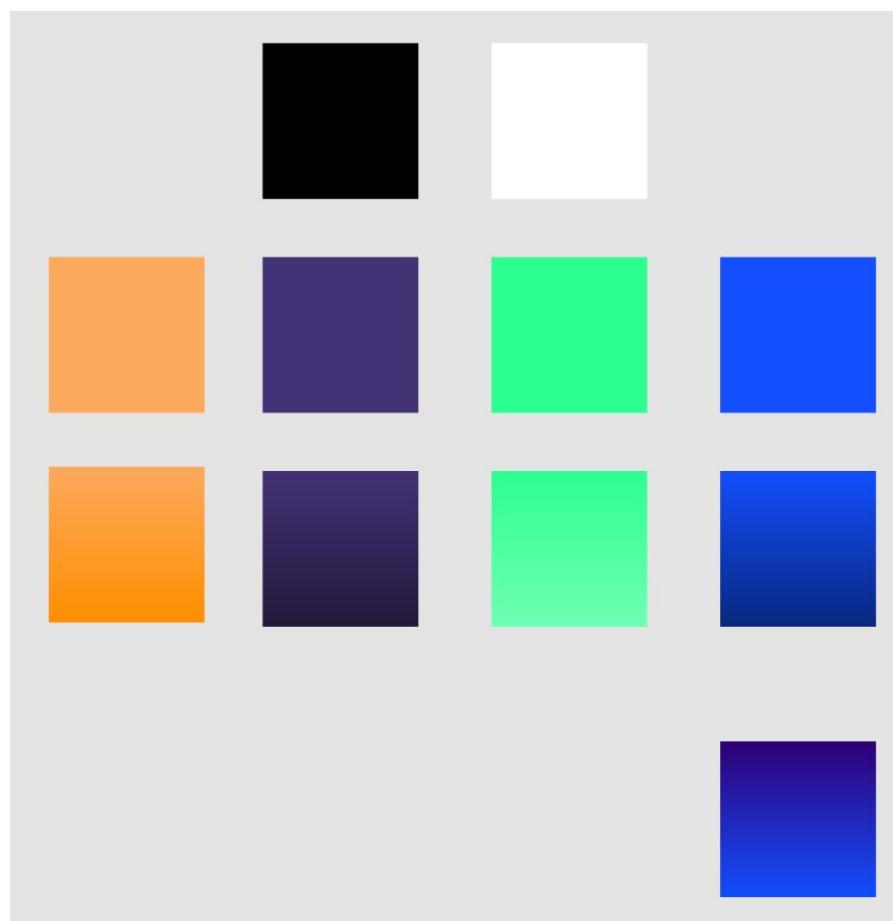


Abb. 16: Farbpalette

4.2 Styleguide

4.2.1 Schrift

H1 - 70pt - Rajdhani - Bold

The Fox Jumps over the grey Fence

H2 - 40pt - Rajdhani - Regular

The Fox Jumps over the grey Fence

Fliessstext - 24pt - DIN Pro - Regular

The Fox Jumps over the grey Fence

Abb. 17: Schrift

4.2.2 Elemente

Buttons Main Menu:



Abb. 18: Menübuttons

Buttons Battle:

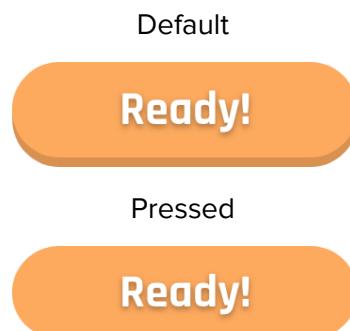


Abb. 19: InGame Buttons

Battle Interaction:

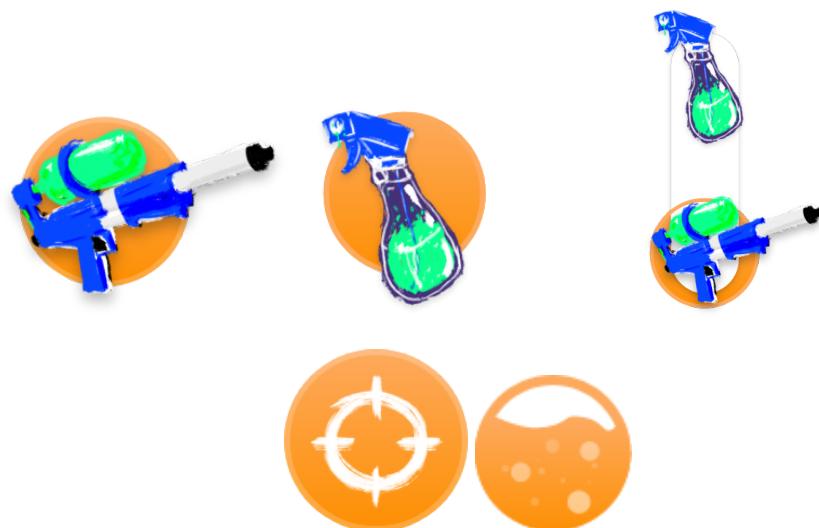


Abb. 20: inGame Buttons Nr. 2

Battle Heading:



Abb. 21: inGame Überschrift

Informationskarten:



Abb. 22: Informationskarte

4.2.3 Charakter

Für die Charakter-Karten wurden Scribbles der Viren verwendet. Wir haben uns auf 3 diverse Viren geeinigt, sowohl im Aussehen als auch in der Art. Aufgrund des interessanten Aussehens und des häufigen Vorkommens bzw. der Assoziation zu Viren durch die Bakteriophage haben wir uns auf die Umsetzung dieser geeinigt anstelle der anderen.

4.3 Logodesign

Wir haben uns im Team relativ schnell auf einen Namen geeinigt. Der erste Vorschlag “ARasite” (AR und “Parasite”, engl. “Parasit”) wurde auch bereits übernommen. Wir haben versucht auch weitere Ideen in Betracht zu ziehen, doch wir haben es bei diesem belassen.

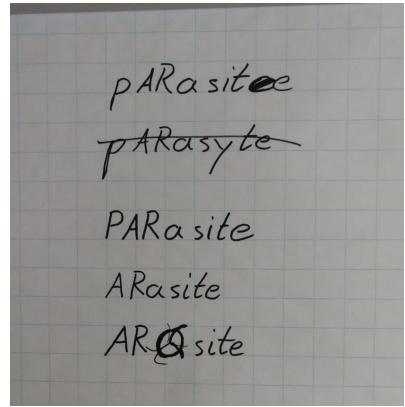


Abb. 23: Logodesign



Abb. 24: Logodesign

5. Interface-Design

5.1 Informationsarchitektur

Trotz der relativ überschaubaren Ebenenstruktur der App wurde eine Informationsarchitektur erstellt, um die Hierarchie der einzelnen Screens zu überblicken und die Abhängigkeiten zu dokumentieren.

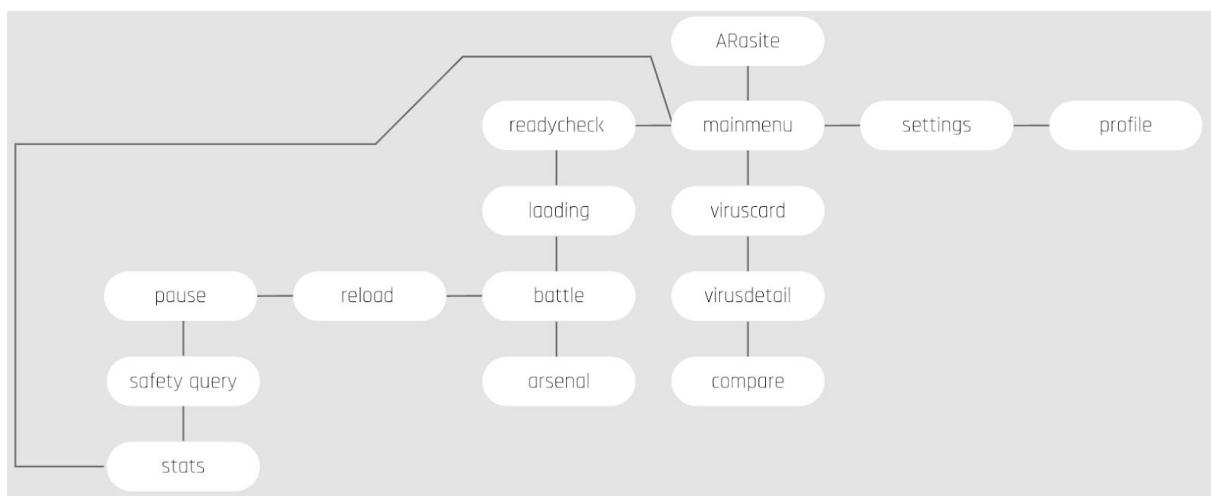


Abb. 25: Informationsarchitektur

5.2 Wireframes

Nachdem die allgemeine Struktur der App erstellt wurde, wurden Wireframes erstellt um erste Interaktionsprinzipien zu generieren, sowie die Nutzbarkeit durch Nutzertests festzustellen und erste Designansätze auszuarbeiten.

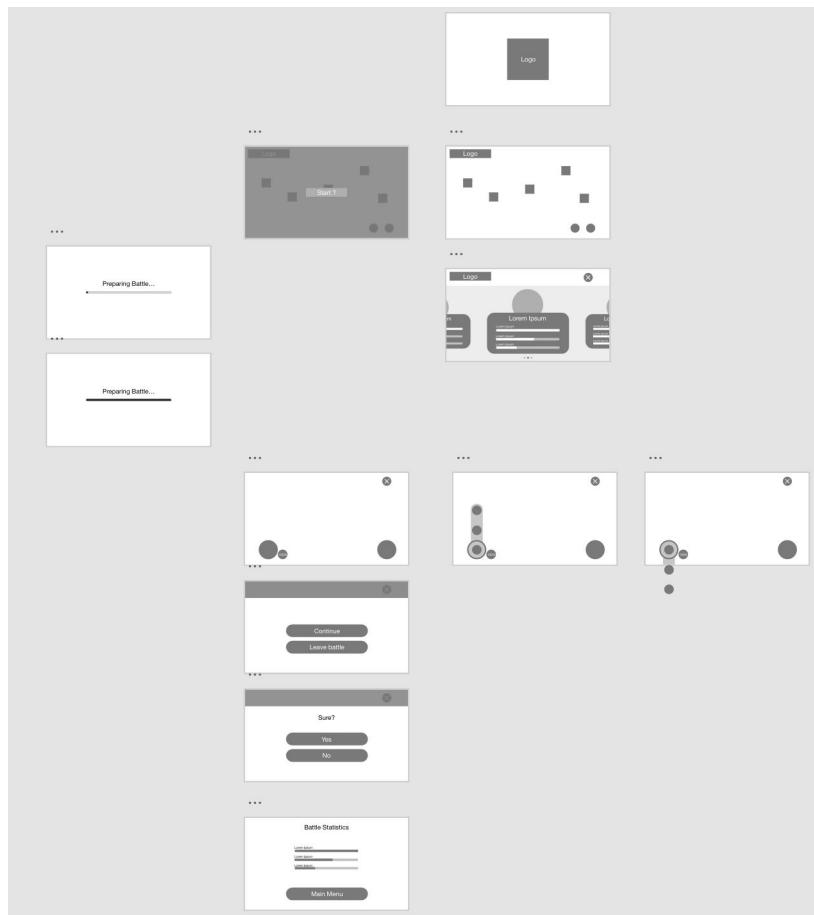


Abb. 26: Wireframes

Diese Wireframes entstanden unter berücksichtigung der "ThumbZone" bei Horizontal ausgerichteten Smartphones.

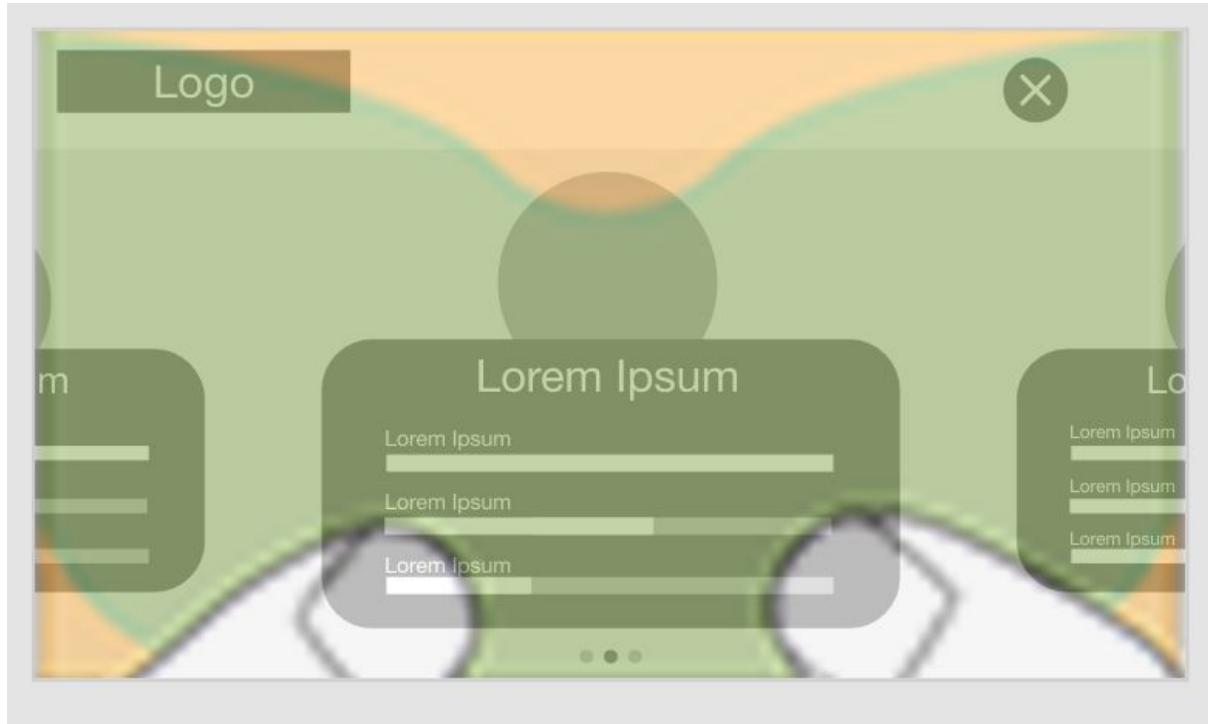


Abb. 27: Wireframe mit ThumbZone

Alle wichtigen und vor allem klickbaren Elemente wurden innerhalb der Zone platziert, welche durch beide Daumen bequem zu erreichen ist.

Nach mehreren kleinen Nutzertest wurden die endgültigen Designmetaphern und Anordnungen des Interfaces festgelegt und anhand des Styleguides gestaltet.



Abb. 28: Nutzertest

5.3 Gestaltung

5.3.1 Moodboards

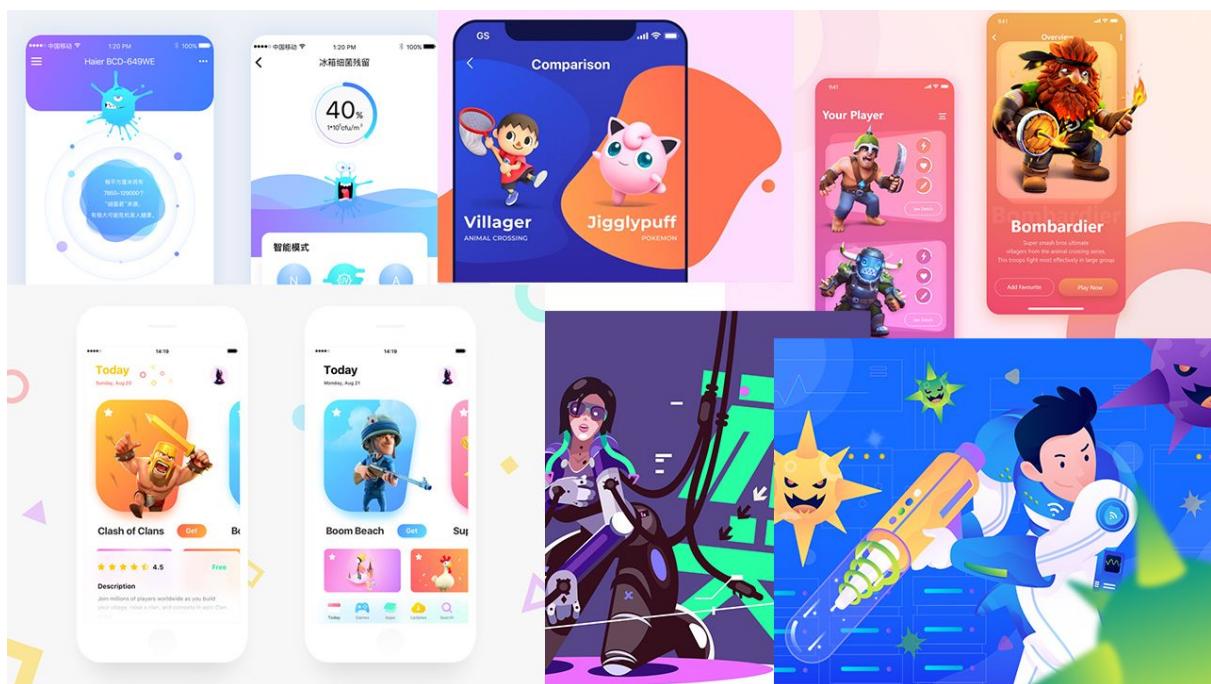


Abb. 29: Moodboard

5.3.2 Screendesign

Nachdem Farben und Schrift festgelegt wurden, konnten anhand der Moods und des visuellen Konzepts erste Screens erstellt werden. Uns war wichtig, den Educational-Anteil visuell vom Gameplay-Anteil abzugrenzen. Daher sind die Informationskarten auf weißem Hintergrund und die Hintergründe im Kampfbereich Lila und die Überschriften unterschiedlich. Um die Konsistenz und Verständlichkeit zu wahren, wurden alle Interaktionselemente in unserer Orangen Highligtfarbe gehalten.

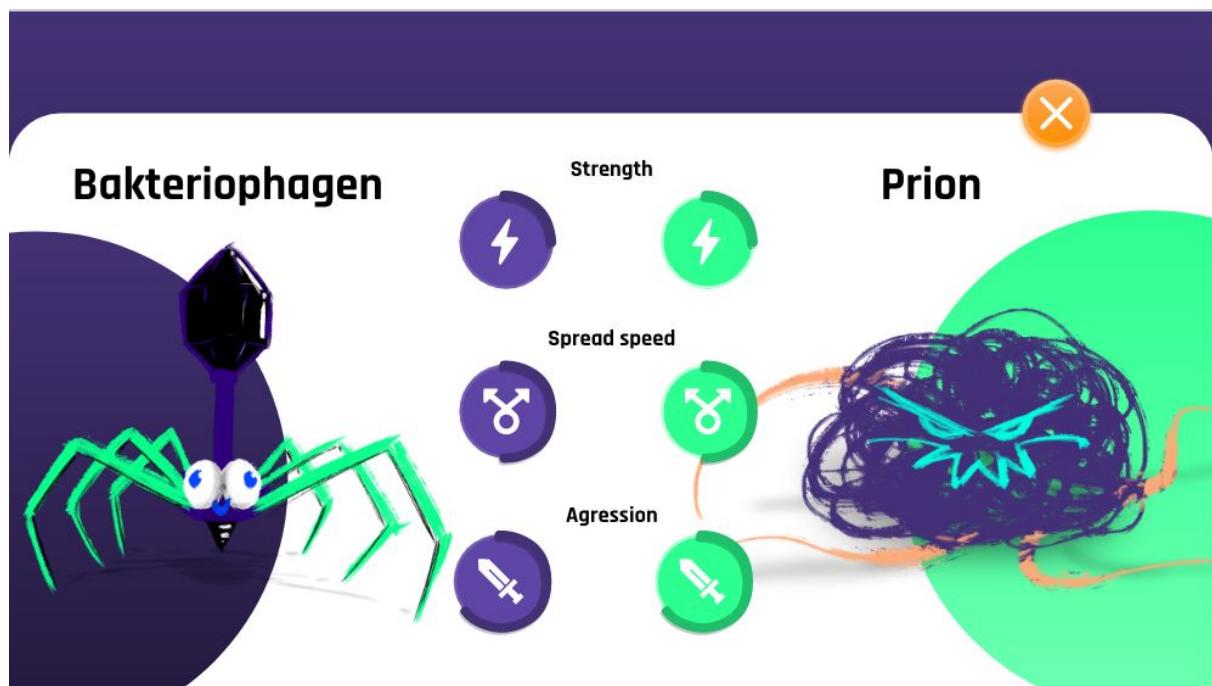


Abb. 30: Vergleich

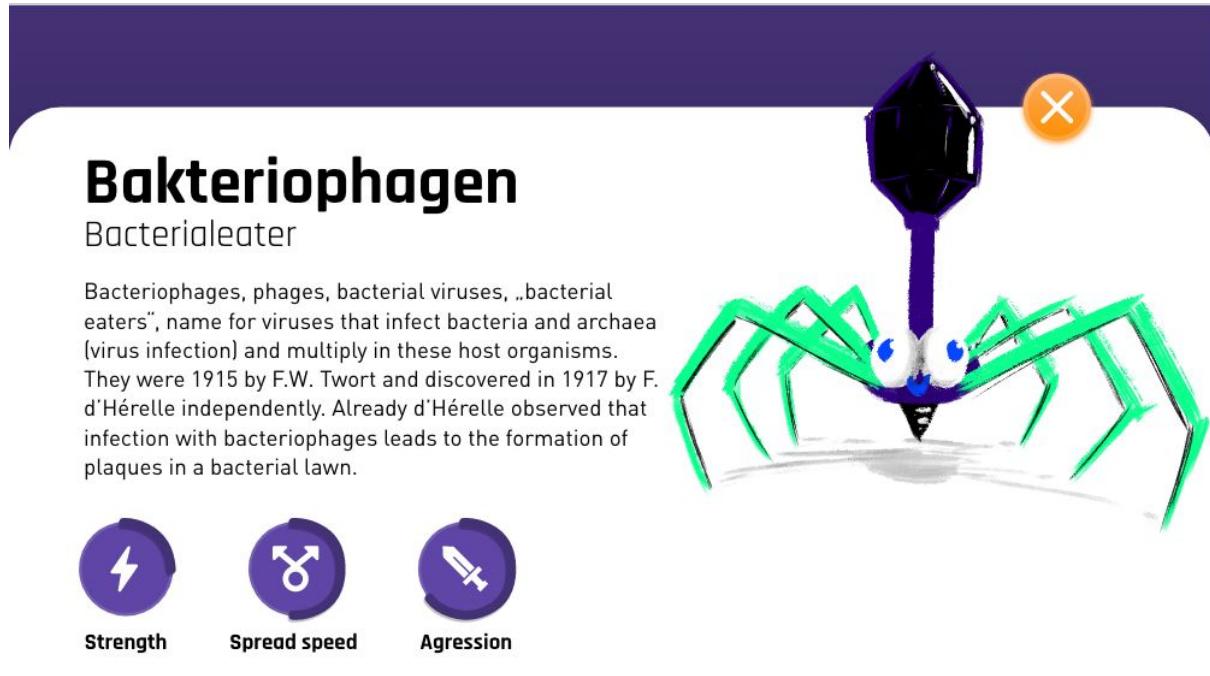


Abb. 31: Informationsdetails



Abb. 32: BattleUI

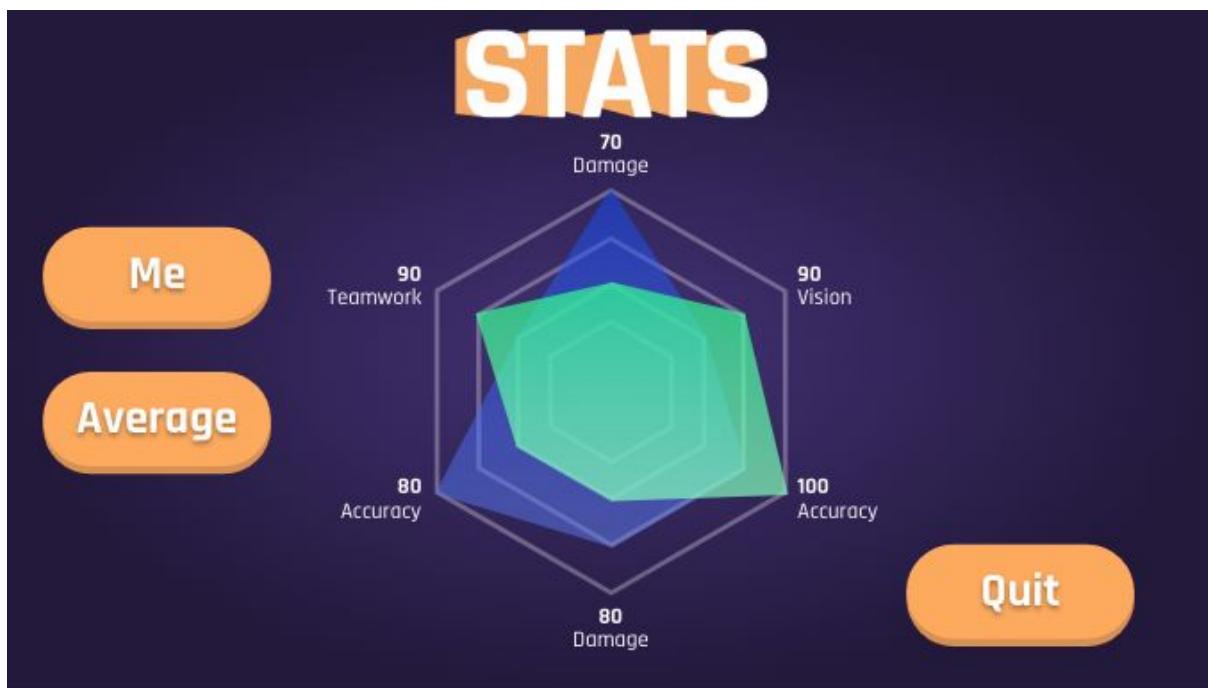


Abb. 33: Battlestats



Abb. 34: Nachladen

Für eine kurze Übersicht des Adobe XD Prototyps:

<https://www.youtube.com/watch?v=qLaHLIGaN98&feature=youtu.be>

6. Sound-Design

Alle Sounds, bis auf einen Multimedia Sound aus der Adobe Bibliothek und zwei Background Titeln (unten genannt), wurden selbst mit einem Mikrofon aufgezeichnet und editiert. Die Sounds für die Bacteriophage sind aus einer weiblichen Stimme entstanden, welche auf mindestens 150% Geschwindigkeit erhöht wurde. Viele der anderen Sounds sowie ein “Ploppen”-Geräusch des Buttons wurden ebenfalls mit dem Mund produziert. Geräusche wie das Herunterfallen der Kiste, das Sprühen oder Pumpen der Supersoaker wurden mit einigen Hilfsmitteln auf dem Tisch aufgezeichnet. Nachdem die Liste für die benötigten Sounds erstellt wurde, haben wir diese sehr kreativ und spontan in Premiere Pro aufgenommen.

Wir haben uns für diese Option entschieden, alle Sounds selbst aufzuzeichnen, da es sich für das Setting ziemlich gut ergeben hatte. Mit dem Mund aufgezeichnete Sounds passen nicht überall hinein, doch aufgrund des Themas und der organischen Charaktere im mikrobiologischen Bereich hatte das gut zusammen gepasst.

Genauso sind wir auch mit der Suche für einen Theme-Song bzw. nach einem Menü- sowie Battle-Sound umgegangen. Diese sollten im Menü etwas mysteriös und ein wenig unheimlich klingen, jedoch während des Kampfes, wenn man den niedlichen Viren konfrontiert wird etwas aufregend und Sympathie-erregend.

Ambient-Track im Hauptmenü: Infados by Kevin MacLeod is licensed under a Creative Commons Attribution licence

Link:

<https://soundcloud.com/user-458397066/cinematic-music-dramatic-mystery-investigation-royalty-free-content-no-copyright-infados>

Background-Track während dem Kampf: Hidden Agenda by Kevin MacLeod is licensed under a Creative Commons Attribution licence

Link:

<https://incompetech.com/music/royalty-free/index.html?isrc=USUAN1200102>

7. Modellierung & Animation

Modelliert wurden die Bacteriophage und der Spielraum, mitsamt einiger benötigter Gegenstände für die Spielwelt. Für das Gesamtkonzept wurden weitaus mehr Objekte geplant, welche nachträglich in der Zukunft umgesetzt werden können.

7.1 Concept Art

Neben den ersten Character Design Doodles wurden gleichzeitig auch Concept Arts zu den Welten erstellt. Diese wurden relativ ohne “High Quality” Digital Art simpel gehalten. Aus dem ersten Concept Art Doodle wurde danach der Vuforia Marker (Abb. 35). Es wurden sehr viele Details hinzugefügt, da der Marker von Vuforia viel Textur zur Erkennung benötigt.

Abbildung 36 diente uns als farbliche Orientierung dazu, dass es eine Art Lootbox geben sollte und eine Art Sporen bzw. Pilze, die neon grün eingefärbt werden sollten.



Abb. 35: simples Concept Art für das erste Level



Abb. 36: Vuforia Marker

Für den Xsekit Virus wurde ein weiteres Concept Art gezeichnet, damit man sich den Charakter und seine Welt gut vorstellen kann. Auch, weil wir ursprünglich überlegt haben dieses zweite Level umzusetzen.

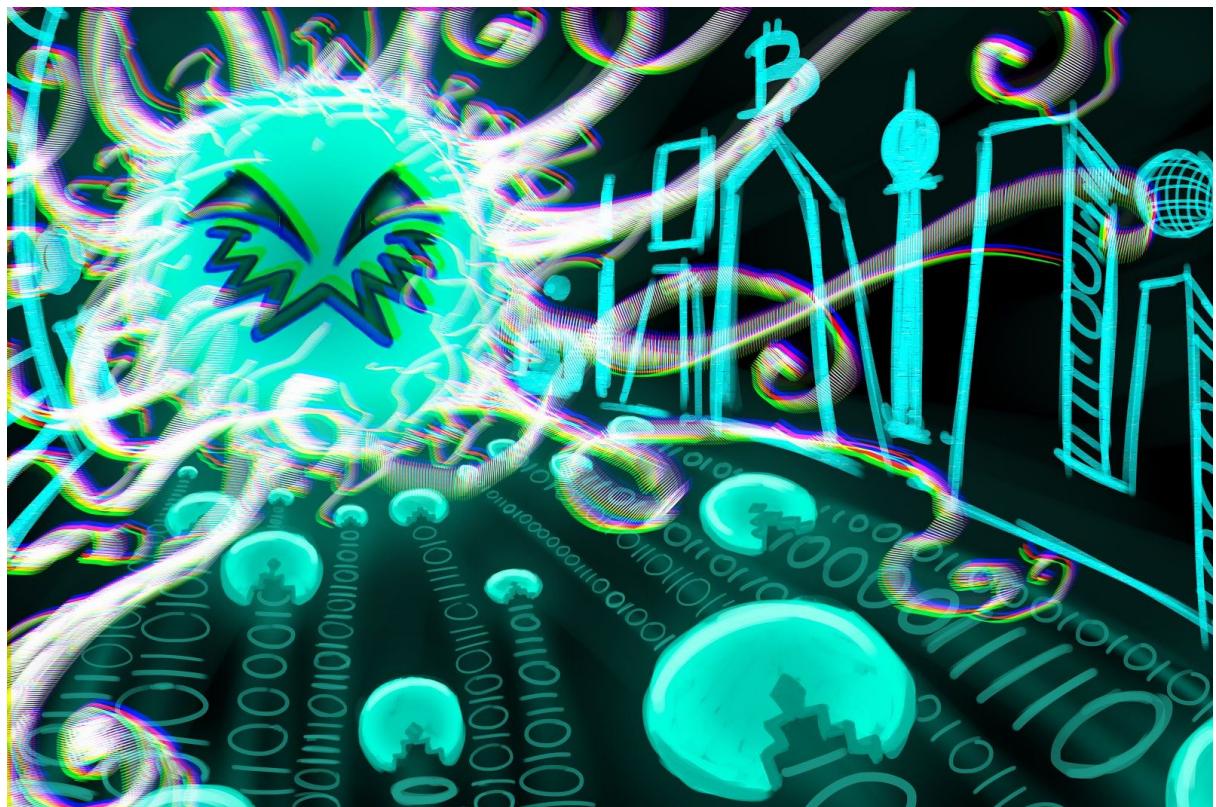


Abb. 37: Concept Art für das zweite Level

7.2 Character

Der zu bekämpfende Feind dieses Spiels ist im Prototypen die Bacteriophage, ein häufig vorkommender Virus. Dieser wurde zuerst mehrmals gescrabbelt, um ihn so modellierbar wie möglich zu machen. Dazu entstand auch etwas Concept Art sowie einige Character Design Entwürfe.

Die Bacteriophage besitzt einen, für sich typischen Körper und wurde mit großen Augen verniedlicht, damit beim Spieler bestenfalls großen Gefallen und Sympathie gegenüber dem Gegner entsteht.

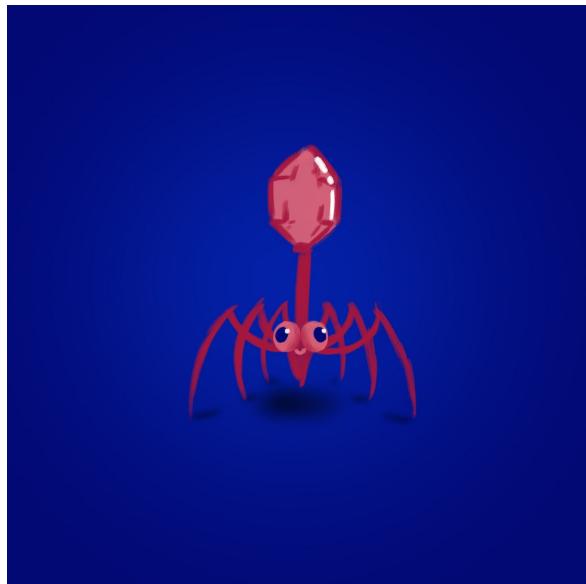


Abb. 38: Bacteriophage Character Design

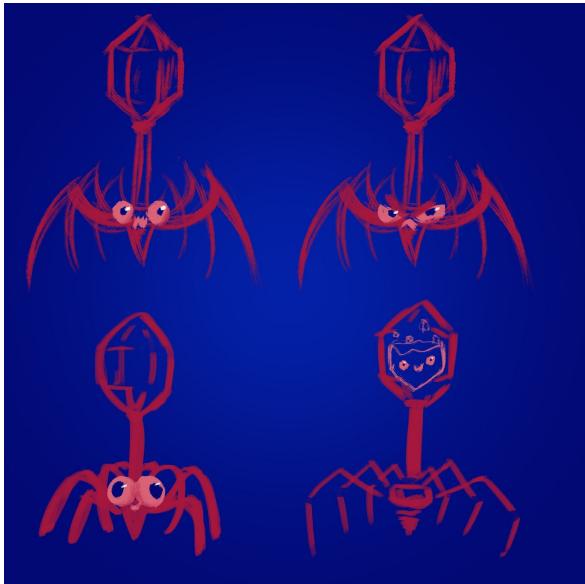


Abb. 39: Bacteriophage Entwürfe

Zusätzlich zur Bacteriophage wurden zwei weitere Character Designs für zwei weitere Viren-Arten angefertigt:

Zum einen ein typischer Internet-Virus und zum anderen eine weitere gewöhnliche Viren-Art, der Prion (Abb. 40) und der Xsekit (Abb. 41). Für weitere Sympathie-Faktoren wurden allen Viren ein Spitzname gegeben. Die Gewichtung davon, was wirklich relevant im Spiel ist, hat sich beim Character Design stark bemerkbar gemacht.



Abb. 40: Prion

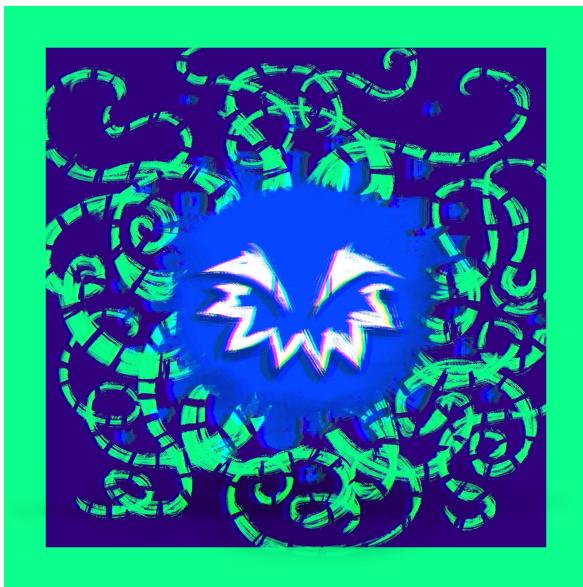


Abb. 41: Xsekit

Der Virus wurde mit Blender 2.8 umgesetzt, welches sich als einsteigerfreundliches Programm bot. Aufgrund der neuen Version musste man sich ein wenig einarbeiten, was jedoch einfach und zeitnah funktionierte. Der Charakter wurde im Low Poly Stil umgesetzt, in den Farben unserer Farbpalette eingefärbt und mit Augen durch Texture-Painting in Blender nachgezeichnet. Im Anschluss wurde der Charakter auf logische geriggt. Dies wurde etwas nachbearbeitet, damit man am nächsten Tag einwandfrei mit den Animationen beginnen konnte.

Aufgrund der einfachen Form des Virus, konnte er mit ca. 1500 Faces relativ unproblematisch umgesetzt werden.

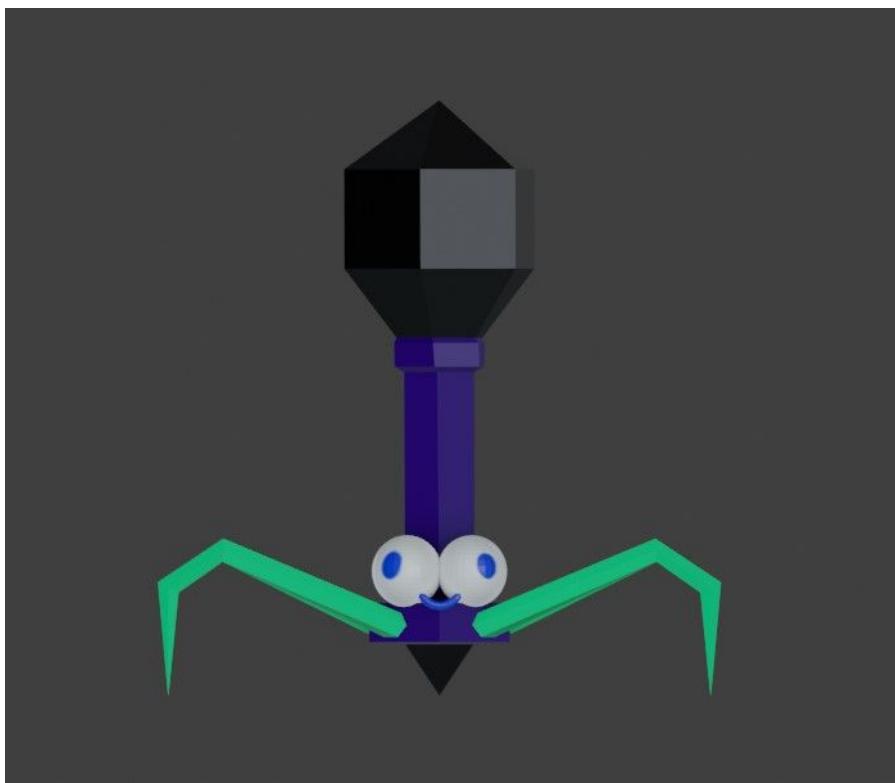


Abb. 42: Backy Rendering

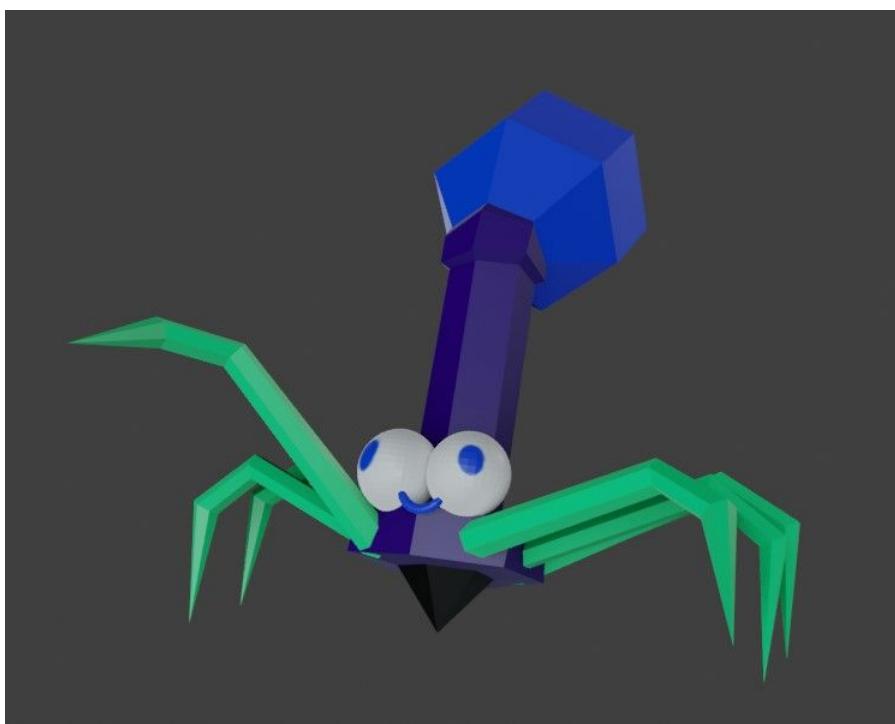


Abb. 43: Backy Rendering Nr.2

7.3 Animation

Zur Animation des Virus wurden zuerst mehrere Scribbles und Überlegungen gemacht, wie organisch sich so ein Wesen fortbewegen und wie es sehr niedlich und sympathisch rüberkommen würde. Wie auch andere Animatoren oft vorgehen, wurden die Bewegungen zuerst selber ausgeführt, danach gescribbelt, selektiert und zum Schluss in Blender animiert.

Die “Andock-” Animation zum Boden ist die aufwendigste und längste Animation, der Walk-Cycle die kürzeste. Auch eine Idle-Animation wurde umgesetzt.

Für den Startscreen wurden drei Arten von Viren Lauf- bzw. Schwimmanimationen hinzugefügt. Da wir den Startscreen mit einem Aquarium identifiziert haben, handelt es sich hierbei eher um Schwimm- anstelle von Lauf-Animationen. Die Bacteriophage schwimmt wie ein Tintenfisch mit 6 Frames, der digitale Virus Xsekit “glitcht” mit 4 Frames, wobei eines leer gelassen wurde. Prion bewegt sich mit 4 Frames fort.

Beispiel-Animation zum Glitchen von Xsekit:

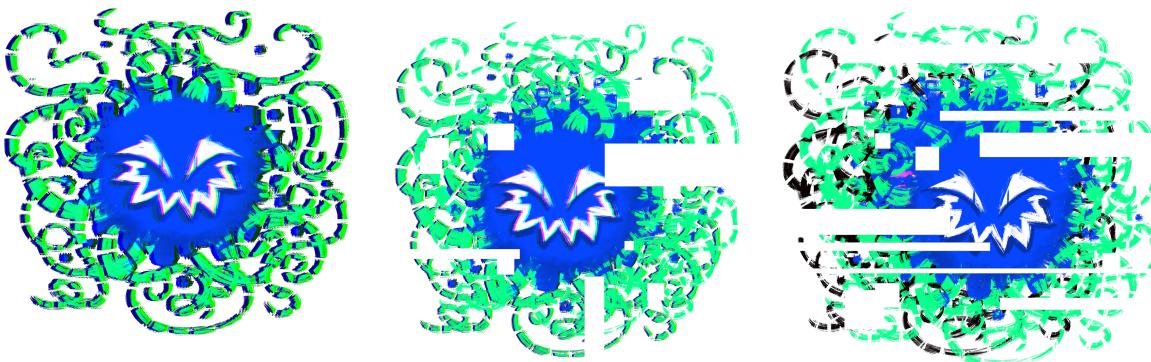


Abb. 44: Xsekit Frame 1-4

Abb. 45: Xsekit Frame 2-4

Abb. 46: Xsekit Frame 3-4

7.4 Items

Bei der Modellierung aller nachfolgenden Items wurde darauf geachtet, den orangenen Farbton zu verwenden. Dieser wurde auch im UI-Design für alle Elemente

genutzt, mit denen interagiert werden kann. Insgesamt wurden drei verschiedene Sporen und zwei Item-Kisten mit verschiedenen Texturen modelliert und eingefärbt.



Abb. 47: Poren

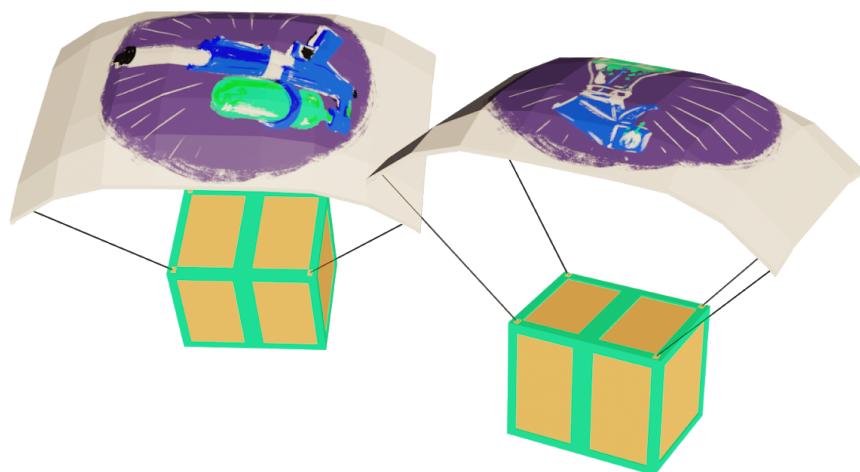


Abb. 48: Item-Kisten

7.5 Environment

Bei der Modellierung der Objekte wurde durchgehend Low-Poly gearbeitet, um einen gleichmäßigen Stil beizubehalten. Jedes der Objekte besitzt unter 2000 Faces, damit die Performance der Applikation nicht unter den 3D-Objekten leidet. Bei Materialien und Texturen wurde das Farb-Design berücksichtigt und nur mit wenigen Details texturiert.

Für die Welt wurde eine kleine Insel modelliert, welche von allen Seiten betrachtet werden kann. In die Form der Insel wurden Berge, bröckelndes Gestein und eine versteckte Höhle eingebaut, in der zwei Fackeln und ein Fuchs mit Animation als Easter für einen explorativen Zusatz sorgen.

Weitere Objekte wie Bäume, Wasser mit einem Wasserfall, verschiedene Kristalle und HFU-Denkmäler wurden auf der Insel platziert, um die Welt zugänglicher zu machen.



Abb. 49: Insel Seitenansicht

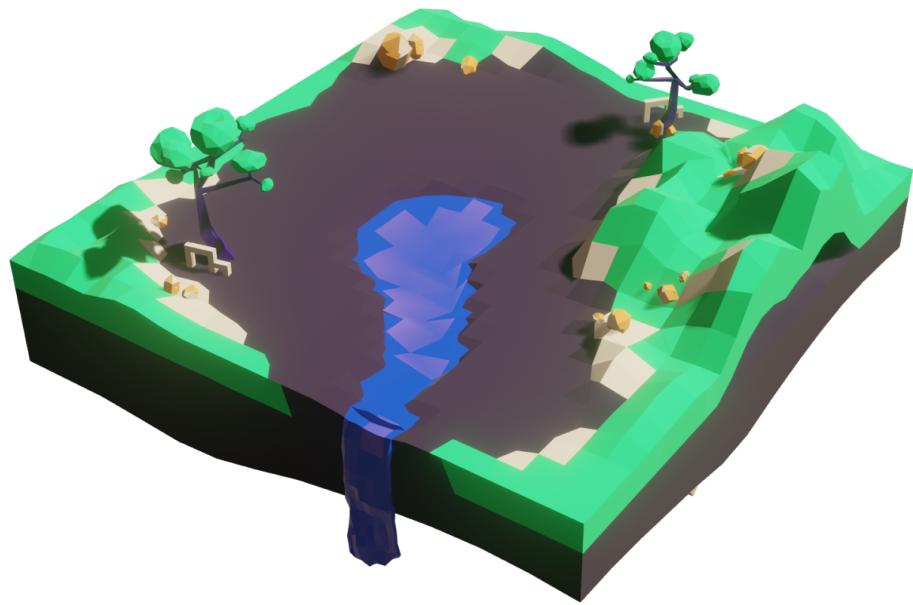


Abb. 50: Insel Vogelperspektive



Abb. 51: Fuchs als Easter-Egg

8. Technische Umsetzung

Im folgenden wird die Umsetzung aus technischer Sicht erläutert.

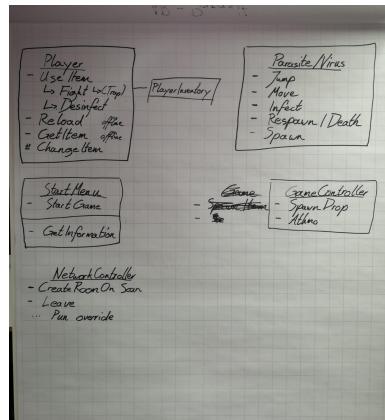


Abb. 52: Provisorisches Klassendiagramm

8.1 Hardware

Für die Anwendung verwendeten wir unsere eigenen Android Geräte, von verschiedenen Herstellern, z.B. Samsung und Huawei, stammen. Da jedoch für die Programmierung auf Vuforia zurückgegriffen wird (s. Kapitel 8.3), werden nur Smartphones mit einer Android-Version 4.4+ unterstützt. Das Alter und die Spezifikationen liegen recht weit auseinander. Für ältere Geräte muss in Unity die benutzte Grafik API auf eine ältere oder Auto gesetzt werden, anstatt Vulkan/OpenGLES3 und ein paar Effekte verringert.

Für Vuforia haben wir einen eigenen Marker designed/gezeichnet, dieser wird an Vuforia übergeben, welcher aus einem möglichst wenig symmetrischen Bild einen Marker generiert mit Erkennungs-Features.

8.2 Unity

Die Interaktion der Anwendung wurde komplett innerhalb Unity realisiert. Hierzu wurde die Version 2019.1.10f verwendet. Die Umsetzung beinhaltet die Programmierung von Interaktion in C#, Erstellen von Animationen mittels Animator, Shader Kreation im Shader-Graph und Einbindung einer Vielzahl von Assets.

Die Anwendung besteht aus einer Startszene und “im Prototyp”, aus einer Virus-Szene, die geladen wird. Angedacht ist hierbei, dass eine unterschiedliche Virus-Szene geladen wird anhand des Vuforia Markers der gescannt wird. Im Prototyp kann per Knopfdruck immer dieselbe Szene geladen werden. Der Name des Markers kann allerdings eingelesen werden.

Innerhalb des Startmenüs werden die bisher besiegten Aliens als Sprites dargestellt und über den Bildschirm bewegt, ein Klick auf diese aktiviert eine Animation um eine Infokarte zu öffnen.

Auf Netzwerkebene wird sich mit den Photon-Server verbunden. Ist dies geschehen darf das Spiel gestartet werden. Beim Spielstart wird einem zufälligen Photon-Raum beigetreten, existiert keiner wird einer erstellt. Beim Betreten des Raumes wird die Virus-Szene geladen.

Die Virus-Szene besteht aus einer Vuforia AR Kamera die dem Spieler das Kamerabild als Hintergrund platziert und die Spielszene auf dem erkannten Marker zentriert und daher die Steuerung übernimmt.

Über Photon-Callbacks wird beim Betreten ein Netzwerkspieler für jeden Spieler instanziert und für alle sichtbar gemacht. Der Master-Client, also der erste beigetretene Spieler übernimmt die Instanzierung und Kontrolle des Virus. Aktionen und Events werden über Remote Procedure Calls (RPC) an alle Clients über den Server gesendet. Auch der agierende lokale Spieler führt für Synchronisations-Zwecke seine Aktion über *target.AllViaServer* aus.

Zur Steuerung der Interaktion hat der Spieler ein UI, *Interface_Inventory.cs*, welches mit dem *ARPlayer.cs* die zentrale Einheit für den Spieler darstellt.

Spieler interagieren mit der Welt über *Raycasts* die entweder per *Touch* an der Screenposition, oder Bildschirmmitte durch *ScreenPointToRay* bestimmt werden und

auf entsprechende Collider treffen, die nach einem *CompareTag* die gewollte Funktion aufrufen.

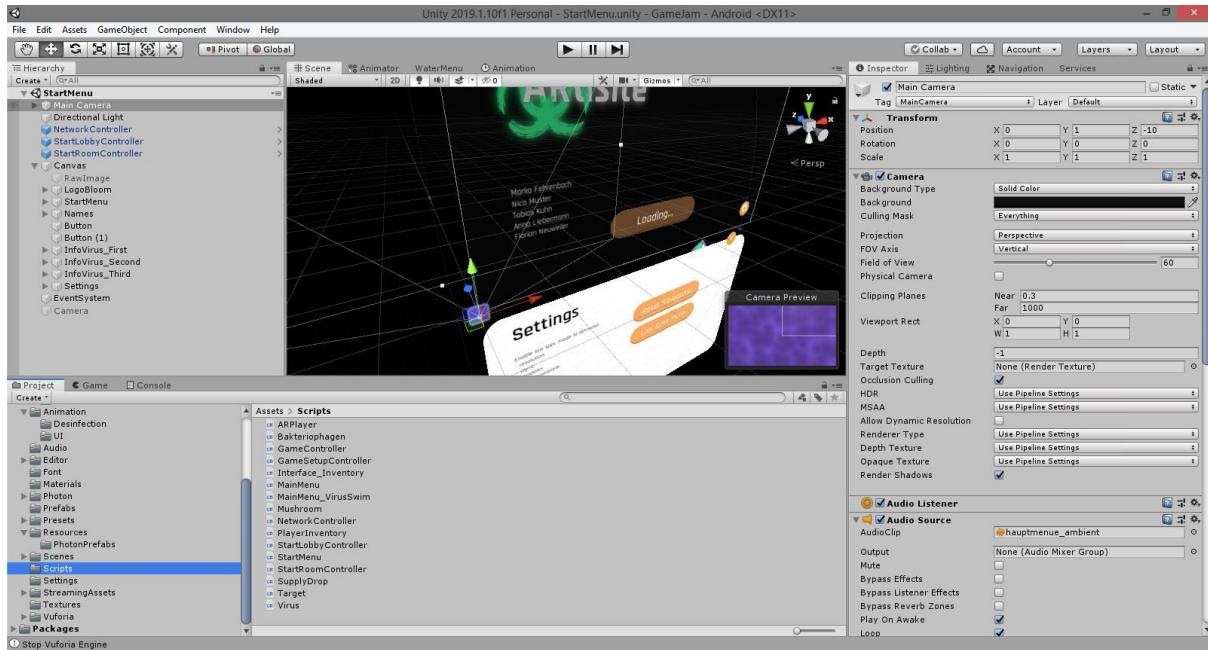


Abb. 53: Unity-Szenen-Ansicht

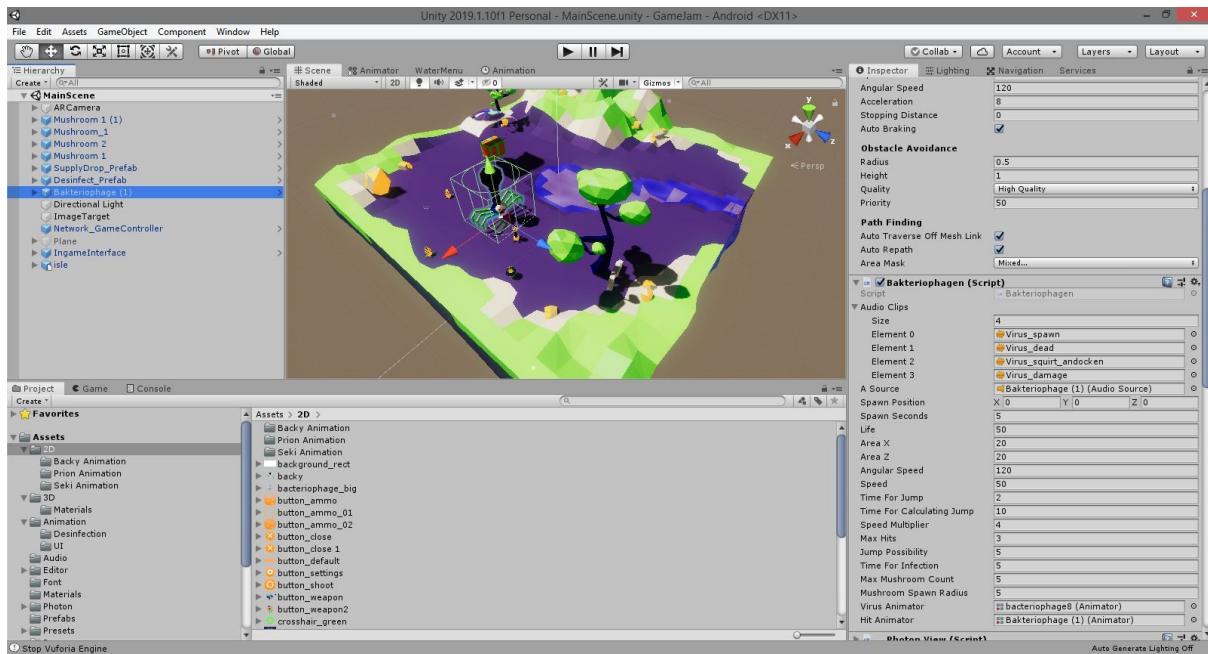


Abb. 54: Unity-Szenen-Ansicht

8.3 Vuforia

Vuforia ist ein Augmented Reality Software Development Kit (SDK) für mobile Endgeräte. Mithilfe von Vuforia lassen sich einfache AR-Anwendungen über Android Studio, XCode und Unity implementieren. Dadurch ist die Erstellung der Anwendung unter den Betriebssystemen Android, iOS und UWP möglich. Da die Bandbreite der möglichen Test- und Endgeräte größer ist als im Vergleich zu anderen AR-SDKs wie ARCore und ARKit, haben wir uns für die Erstellung des Prototypen für Vuforia Version 8.3 entschieden.

Vuforia ist hauptsächlich bekannt für markerbasiertes Tracking. Darunter fallen zum Beispiel das Verfolgen von selbst erstellten Bildern, sogenannte VuMarks und Objekten. Für unseren Prototypen haben wir eine Bilddatei konzipiert (s. Abb. 36), welche dem Concept Art unserer Anwendung entspricht. Diese wurde über die Vuforia-Website als Device-Datenbank für Unity heruntergeladen und in unser Projekt angebunden. In Unity wurde für die Szene eingestellt, dass an dem Punkt der Ursprung der Szene gesetzt wird, an dem das erste Bild der Datenbank erkannt wurde.

8.4 Photon

Als Netzwerk wurde auf das Photon SDK Version 2.13 für Unity, kurz PUN 2, zurückgegriffen. Photon bietet den Vorteil, nahezu in Echtzeit Remote Procedure Calls (RPCs) über die Netzwerkstruktur von Photon zu senden.

Für die Implementierung des Prototypen erstellt ein Spieler einen zufallsgenerierten Raum, den die anderen Spieler nachjoinen können. In Zukunft kann dieser Algorithmus abhängig vom Ort gemacht werden, an dem sich die Spieler momentan aufhalten. Direkt nach Start werden die Spieler über das Netzwerk instanziert. Zusätzlich erstellt der MasterClient für alle anderen Spieler die Instanz des Virus. Somit ist der MasterClient für die Steuerung und Synchronisation des Virus sowie des Gameplays zuständig. Aus Performance- und Latenzzeit-Gründen wurde entschieden, möglichst auf komplexe Datentypen zu verzichten und einfache Datentypen wie

Integer oder Float mit RPC-Aufrufen über das Photon-Netzwerk zu senden. Für die zukünftige Weiterarbeit am Projekt sollte das Netzwerk so angepasst werden, dass ein anderer Spieler die Rolle des MasterClients übernimmt, nachdem der ursprüngliche MasterClient den Raum verlassen hat und sich immer noch Spieler im Raum befinden.

8.5 Programmierung

Um Instanzen innerhalb des Unity-Projektes von jeder aus Klasse zugreifen zu können, wird auf das Singleton-Pattern von Unity zurückgegriffen. Komponenten wie der GameController, der Spieler selbst sowie das Inventar-Interface für den jeweiligen Spieler liegen innerhalb der Szene als Singletons vor. Somit wird ein umständliches und längeres Suchen von Komponenten umgangen.

Der GameController übernimmt die Hauptsteuerung unserer Anwendung. Er ist als Singleton implementiert und sorgt dafür, dass die Spieler bei bestimmten Aktionen synchronisiert werden.

Die jeweiligen Spieler wurden über die Komponente ARPlayer implementiert. Diese werden zu Beginn des Spiels über das Photon-Netzwerk vom GameController instanziert. Jeder Spieler ist für die Verwaltung seines eigenen Inventars verantwortlich und übermittelt dem GameController seine Aktionen.

Da in Zukunft weitere Virus-Typen vorgesehen sind, wurde speziell für die Viren ein Interface erstellt, über welches sich die KI des Virus von außen ansprechbar ist. Dadurch können verschiedene Viren-Typen unterschiedliche Verhalten besitzen, die die Viren hauptsächlich selbst festlegen können. Die Wegfindung basiert auf die NavMesh-Struktur von Unity, die schon in Kapitel 8.2 schon beschrieben wurde. Innerhalb der KI wird vom MasterClient eine zufällige Position auf dem NavMesh berechnet. Diese Position wird als x-y-Koordinaten über einen RPC-Aufruf an alle Viren-Instanzen auf den anderen Geräten gesendet. Somit wird auf jedem Endgerät der Weg zur Position separat berechnet. Eine Synchronisation der Position, Rotation und Skalierung der KI pro Frame wird nicht benötigt.

8.6 Umgebung

Wir haben uns entschieden nur ein Virus Level zu erstellen, einen biologischen, und an diesen wurde die Umgebung angepasst. Dazu wurde eine Außenszene einer kleinen schwebenden Insel erstellt, mit Elementen wie Bäumen, Bergen, einer Höhle und Wasser. Zudem wurde zum angedachten Marker-Ort, dem I-Bau, ein passendes Element eingebaut in Form der Beton-Bänke neben dem Gebäude.

Die Umgebung wurde anhand der erstellten Farbpalette eingefärbt. Beim Modellieren wurde auf realistische Maße und geringe Polygon-Anzahl geachtet. Für ein einzelnes Objekt erstreckt sich der Count selten über 2000, so dass die Gesamtszene mit den *spawnenden* Sporen, zur Laufzeit, selten über 100 Tausend Triangles besitzt um den Anforderungen eines Mobilen Gerätes zu entsprechen.

Die Umgebung ist in Unity als *Navigation-Static* markiert, damit ein NavMesh erstellt werden kann über das Navigations-Tab in Unity. Dies wird benötigt für die Navigationssteuerung des Virus. Zudem wurde an die Umgebung ein Mesh-Collider angeheftet damit die Item-Kisten *despawnen* wenn sie die infizierte Umgebung berühren. Ebenso wird der Collider benötigt, damit Partikel von Desinfektion und Angriff, am getroffenen Punkt *spawned* werden können.

8.7 Beleuchtung

Unsere Szene ist relativ klein, gut optimiert, aber mit viel dynamischen Inhalten und eine Außenszene, daher haben wir die Standard Unity Echtzeit-Beleuchtung verwendet. Hierfür beinhaltet die Szene ein einfaches direktionales Licht und in der Höhle noch zwei kleine Punktlichter.

Aus Zeitgründen wurde auf das Erstellen von Lightprobes für Mixed Lighting und das Nutzen von vorgerendertem, Licht zu ermöglichen, verzichtet, da auch so ein flüssiges Spielerlebnis ermöglichen konnten. Für eine Steigerung der Performanz in zukünftigen Iteration wäre dies ein sinnvoller Schritt.

Die Sporen, mit denen die Umgebung infiziert wird, besitzen ein Material mit Emission, nicht als Beleuchtung für die Umgebung zu beleuchten, sondern um den im Post-Processing verwendeten Bloom-Effekt zu verstärken.

8.8 Post-Processing

Um die Szene lebhafter zu gestalten und die Möglichkeiten der Scriptable Pipeline voll auszunutzen integrierten wir das Unity eigene Post-Processing Package. Hierbei werden Effekte auf die gerenderte Szene angewandt, zur Auswahl stehen die gängigen wie sie in modernen Spielen vorkommen, Ambient Occlusion, Bloom, Temporal Antialiasing uvm.

Wir entschieden uns für ein Color-Grading um eine etwas actiongeladenere Szenerie zu generieren, mit höheren Kontrasten, da es sich um einen Überlebenskampf gegen eine Virengefahr handelt. Zusätzlich eine wärmere Farbtemperatur für ein sommerliches Outdoor Design. Damit die Strahlende Sonne, eines Außenbereichs besser präsentiert wird, ergänzten wir das Ergebnis mit einem Bloom Effekt.

Um Post-Processing einzusetzen muss entweder Unity Lightweight Render Pipeline oder High Definition Render Pipeline verwendet werden. Auf der Kamera eine Post-Processing Layer Komponente hinzugefügt werden und auf einem Objekt ein Post-Processing Volumen. Wir entschieden uns dafür nur ein Post-Processing Volumen Global zu verwenden.

8.9 Versionskontrolle und Kollaboration

Wir nutzen für unsere Versionskontrolle und Zusammenarbeit GitHub, mit der grafischen Oberfläche GitKraken. Hierzu legten wir ein GitHub Master-Branch an, auf dem jeder agiert. Wenn ein Aufgabenpaket erledigt, oder eine wichtige Änderung geschehen ist, wird zuerst alle Änderungen *gestaged*, mit einem Namen und einer Beschreibung versehen, anschließend *committed* um den Master lokal zu *mergen* und

anschließend auf das Netzwerk *gepushed* für alle Sichtbar. Bevor man einen eigenen Push durchführen kann muss immer zuerst der letzte Stand *gepulled* werden.

Dabei musste beachtet werden, möglichst in unabhängigen Modulen zu arbeiten, in einer eigenen Szene, da Änderungen an der Main-Scene zu Merge-Konflikten führen, bei denen die Änderungen, an dieser Szene, von einer Person zurückgenommen werden müssen. Wegen Prefabs die zwar in der eigenen Szene bearbeitet wurden, aber in der Main-Szene vorkommen ist gab es teilweise trotzdem Merge-Konflikte die nicht immer nachvollziehbar waren.

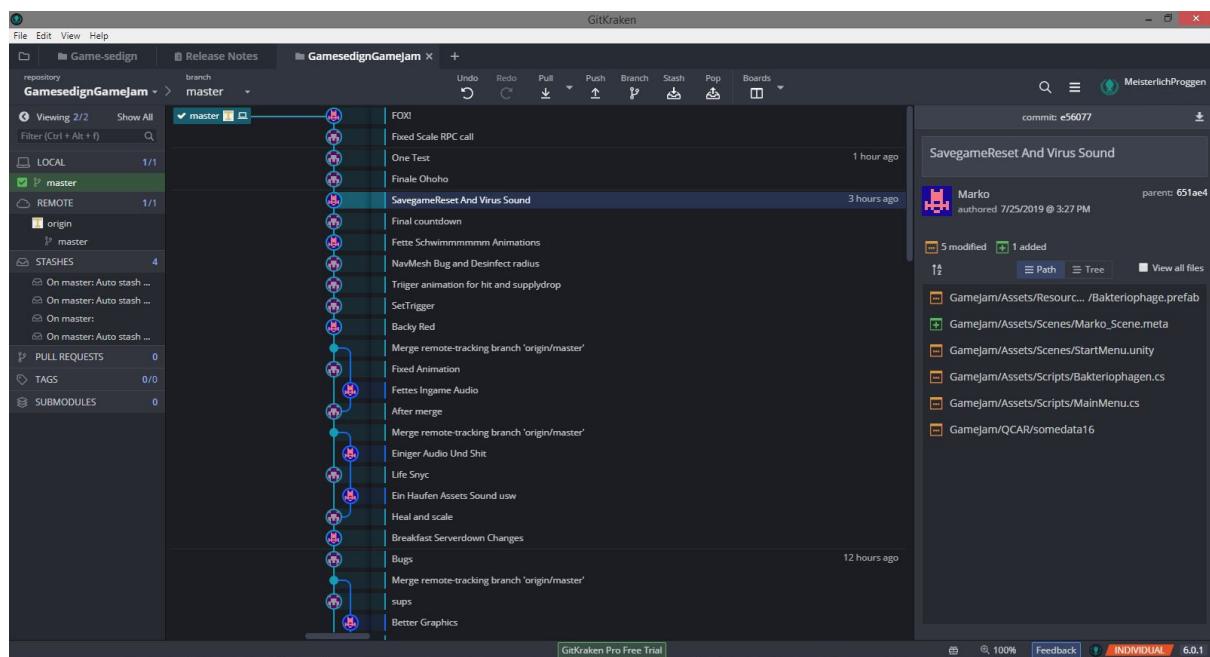


Abb. 55: GitKraken - Grafisches Git Interface

9. Projektmanagement

Noch vor dem Game-Jam wurden alle notwendigen Applikationen und Tools auf allen Geräten installiert und auf gleiche Versionen gebracht.

Auch die Rollenverteilung wurde schon im Vorhinein besprochen. Marko sollte als Tech-Artist mit einem Fokus auf Asset-Implementierung und der Programmierung der Interaktionselemente arbeiten. Florian beschäftigte sich als Lead-Programmer um die Netzwerk-Programmierung und weitere Arbeiten aus technischer Sicht. Anna war als

Concept Artist und Sound-Designer für das Character Design und Animationen zuständig. Tobias kümmerte sich um die Gestaltung der 2D-Elemente als UX- und UI-Designer. Nicolas arbeitete als 3D-Artist und Projektmanager an der Modellierung der 3D-Objekte und an der Kommunikation zwischen allen Team-Mitgliedern.

Unter morgendlichen und abendlichen Meetings wurden Fortschritte und Updates besprochen und neue kurzfristige Ziele geplant und verteilt. Alle Aufgaben und deren Status wurden im Verlauf des gesamten Projekts an einem Task-Board festgehalten. Zu überprüfende und fertige Aufgabenbereiche konnten mit dem Task-Board einfach koordiniert werden.

Für das Projekt wurden zur Verwaltung aller Dateien hauptsächlich GitKraken und Google Drive genutzt. Mit GitKraken konnten technische Änderungen verwaltet werden, während in Google Drive wichtige Assets, Animationen und UI-Design ausgetauscht wurden.

10. Game-Jam Reflexion

Insgesamt haben wir im Team sehr unkompliziert und produktiv miteinander gearbeitet und sind zu einem für alle zufriedenem Ergebnis gekommen. Das gut durchmischte Team mitsamt all seinen Fähigkeiten konnte in jeder Hinsicht etwas zum entstandenen Resultat beitragen.

Aufgrund dessen, dass wir den ersten Tag bzw. Dreiviertel davon für das Brainstorming, das Konzept und die gesamte Idee investiert haben, sind kaum Probleme aufgetreten. Während andere Teams mit der Umsetzung begonnen haben, ist unsere Planung, auch Wochenplanung zu Ende geführt und eingehalten worden. Dazu haben zusätzlich tägliche Besprechungen stattgefunden.

Unserer Meinung nach passen das Konzept und der Prototyp gut zusammen, sodass wir uns überlegen dieses Spiel auch weiterzuführen. Nach unserem Empfinden ist in der kurzen Zeit ein toller Prototyp entstanden.

Da wir uns ein realistisches Ziel gesetzt und nicht allzu viel vorgenommen haben, haben wir wenige bis kaum Probleme, vor allem keine gravierenden Komplikationen gehabt. Zeitlich wurde ebenso jeder notierte Punkt bzw. jede Tagesaufgabe

eingehalten. Unsere Probleme waren unter anderem die Merge-Konflikte durch GitKraken, ansonsten nur wenige weniger erwähnenswerte Kleinigkeiten.

Bei der 3D-Charakter-Modellierung ist ein Bug aufgetreten, der sich ständig wiederholt hat, was jedoch am Rechner gelegen haben könnte und schnell gefixed wurde.

Aufgrund unserer Erfahrung in der Arbeit mit größeren Teams oder der Erfahrung mit Game-Jams selber, war uns von Anfang an bewusst, wie wir mit allem umgehen sollten. Die verbrachte Zeit und Arbeit am Spiel haben wir sehr genossen und werden den Prototyp vermutlich weiter entwickeln.

11. Abbildungsverzeichnis

- Abb. 1: Brainstorming Schritt 1
- Abb. 2: Brainstorming Schritt 2
- Abb. 3: Konzeptblatt Teil 1
- Abb. 4: Konzeptblatt Teil 2
- Abb. 5: Konzeptblatt Teil 3
- Abb. 6: Konzeptblatt Teil 4
- Abb. 7: Konzeptblatt Teil 5
- Abb. 8: Finale Auswahl eines Konzepts
- Abb. 9: Moodboard Nr. 1
- Abb. 10: Moodboard Nr. 2
- Abb. 11: Moodboard Nr. 3
- Abb. 12: Task-Board Nr. 1
- Abb. 13: Namensgebung
- Abb. 14: Brainstorming
- Abb. 15: Informationen als Skills
- Abb. 16: Farbpalette
- Abb. 17: Schrift
- Abb. 18: Menübuttons
- Abb. 19: InGame Buttons

- Abb. 20: inGame Buttons Nr. 2
- Abb. 21: inGame Überschrift
- Abb. 22: Informationskarte
- Abb. 23: Logodesign
- Abb. 24: Logodesign 2
- Abb. 25: Informationsarchitektur
- Abb. 26: Wireframes
- Abb. 27: Wireframe mit ThumbZone
- Abb. 28: Nutzertest
- Abb. 29: Moodboard
- Abb. 30: Vergleich
- Abb. 31: Informationsdetails
- Abb. 32: BattleUI
- Abb. 33: Battlestats
- Abb. 34: Nachladen
- Abb. 35: simples Concept Art für das erste Level
- Abb. 36: Vuforia Marker
- Abb. 37: Concept Art für das zweite Level
- Abb. 38: Bacteriophage Character Design
- Abb. 39: Bacteriophage Entwürfe
- Abb. 40: Prion
- Abb. 41: Xsekit
- Abb. 42: Backy Rendering
- Abb. 43: Backy Rendering Nr.2
- Abb. 44: Xsekit Frame 1-4
- Abb. 45: Xsekit Frame 2-4
- Abb. 46: Xsekit Frame 3-4
- Abb. 47: Poren
- Abb. 48: Item-Kisten
- Abb. 49: Insel Seitenansicht
- Abb. 50: Insel Vogelperspektive
- Abb. 51: Fuchs als Easter-Egg

Abb. 52: Provisorisches Klassendiagramm

Abb. 53: Unity-Szenen-Ansicht

Abb. 54: Unity-Szenen-Ansicht

Abb. 55: GitKraken - Grafisches Git Interface

12. Anhang

12.1 GitHub-Link

<https://github.com/nefl1011/GamesedignGameJam>

12.2 Selbstdokumentation Anna Liebermann

Montag (22.07.2019):

- Ideenfindung, Brainstorming, Konzeption (6-3-5)
- Research
- Dribbble bzw. Design Research
- Moodboard
- Mini-Animation, Scribbles
- Concept Art
- Character Design

Dienstag (23.07.2019):

- Blender Wiederholung/Einführung
- 3D Modelling für den Character
- Rigging
- Assets gezeichnet für UI/2D

Mittwoch (24.7.2019):

- 2D Assets verbessern aufgrund neuer Farbwahl, Verbesserung Concept Art

- Ingame Icons (Supersoaker, Desinfikationsspray)
- Marker Bild gezeichnet
- Scribbles für mögliche Animationen
- Character Animation für wichtige Zustände (Dead, Idle, Squirt, Walk)
- Sounds mit Mikro aufgenommen und editiert laut Sound Liste

Donnerstag (25.7.2019)

- Character Design für weitere Viren
- Letzten Sounds editiert
- Detailierteres Concept Art für ein weiteres Level/Welt
- Gif-Animationen für alle 3 Viren
- Playtesting
- Dokumentation
- 2D Assets

12.3 Selbstdokumentation Florian Neuweiler

Montag (22.07.19):

- Brainstorming
- 6-3-5-Kreativtechnik
- Erstellung Moodboard
- Konzeption eines Klassendiagramms und C#-Skripte
- Anbindung von Vuforia und Netzwerk

Dienstag (23.07.19):

- Implementierung KI
 - Randomisierte Bewegung
 - Wechseln von Zuständen
- Anbindung NavMesh und -Agent
- Synchronisierung KI über PunRPC
- Verknüpfung Drops und KI

- Verknüpfung Fight-Funktion mit Treffer an Bakteriophage

Mittwoch (24.07.19):

- Verbesserte Wegfindung des NavMeshAgents
- Pilze im Kreis um KI instanziieren, anschließend Funktionalität auf das Netzwerk erweitert
- Zerstören der Drops, sobald sie die Welt berühren
- Anpassung der Fight-Funktion
- Anpassung des NetworkControllers an neues Konzept der Raumerstellung im Netzwerk
- Einbindung der neuen Weltoberfläche und Anpassung entsprechender KI-Funktionalitäten
- Anbindung von Animationen an die KI

Donnerstag (25.07.19):

- Generierung Vuforia-Marker
- KI heilt sich, wenn Pilze instanziert werden und skaliert mit Leben
- Synchronisation der KI mit allen Spielern
- Animation über Photon getriggert
- Merge mit fehlenden UI- und Sound-Komponenten
- Diverse Bug-Fixes
- Dokumentation

12.4 Selbstdokumentation Marko Fehrenbach

Montag (22.07.19)

- Konzeption / Brainstorming (6-3-5 Methode)
- Moodboard / Konzeptentscheidung, Verfeinerung
- Planung der Klassenstruktur und technischer Spielaufbau
- Planung Netzwerkstruktur
- Programmierung Maingame UI

- Erste Tests des bisherigen Stands

Dienstag (23.07.19)

- Ausbau der Funktionalität der UI Interaktion
- Programmierung PlayerScript Interaktion
- Programmierung Netzwerk ARPlayer Script
- Programmierung von Item-Kiste und Item-Kiste Netzwerk
- Hilfestellung bei Erstellung von 3D-Character
- Nacharbeiten von 3D-Character
- Absprache weiteres Vorgehen, Abstimmung aller Teilmodule

Mittwoch (24.07.19)

- Implementierung eines Start Screens
- Implementierung von *Because Games Matter* Informationen über besiegte Viren im Startbildschirm
- Verbesserungen von UI/Gameplay Utility (Animationen, Partikel)
- Import und Einsatz von 2D/3D Assets
- Hilfestellungen/Tipps beim Animieren / Modellieren
- Testen des Builds

Donnerstag (25.07.19)

- Implementierungen Sound
- Implementierungen UI -> Savegame Reset, weitere Viren-Informationen, Endscreen bei Erfolg
- Bug-Fixing und Polishing, Implementierung Low Spec Mode
- Endgültiges Testing der App als Build / Playtesting
- Dokumentation Technische Umsetzung
- Rigging / Animation - Fuchs Easter Egg

12.5 Selbstdokumentation Nicolas Muster

Montag (22.07.19)

- Konzeptionsphase
- Brainstorming und 6-3-5-Methode
- Moodboard
- Konzeptauswahl und Verfeinerung
- Asset-Liste und Planung Modellierung
- Aufsetzen, Gliedern vom Game-Dokument

Dienstag (23.07.19)

- Recherche für Modellierung und Referenzbilder downloaden
- 3D-Modellierung von 3 Sporen für die Sporenspur
- 3D-Modellierung der Item-Kiste
- Beginn 3D-Modellierung Schwebende Insel
- Absprache Sound Design, KI-Intelligenz und To-Do für Folgetag

Mittwoch (24.07.19)

- Sound-Asset-Liste
- Coloring Sporen und Item-Kiste und Insel mit allen 3D-Modellen
- Anpassung Texturen und UV-Mapping
- 3D-Modellierung Insel, Wasserfall, Höhle, Fackeln, Bäume, Kristalle, HFU-Objekt
- Beginn mit dem Verfassen des Konzepts im Game-Dokument
- Recherche und Besorgung eines Royalty Free Ambient Sound

Donnerstag (25.07.19)

- Druck der Vuforia-Marker
- Modellierung und Kolorierung eines Fuchs - Easter Egg
- Erstellung eines Titelblatt für die Dokumentation
- Playtesting
- Dokumentation

12.6 Selbstdokumentation Tobias Kuhn

Montag (22.07.19)

- Ausarbeitung des Konzepts
- Aufteilung der Arbeitsbereiche
- Informationsarchitektur
- Erste Wireframes mit Berücksichtigung der ThumbZone bei Smartphones
- Zielgruppenrecherche
- Nutzertest 1

Dienstag (23.07.19)

- Zweite Version Wireframes mit kleinem Nutzertest
- Planung der User Experience
- Ausarbeitung des Interaktionskonzepts
- Schriftauswahl/Farbauswahl
- Stilfindung
- Beginn der Ausgestaltung
- Nutzertest 2

Mittwoch (24.07.19)

- Ausgestaltung unter Berücksichtigung des Interaktionskonzepts und UX/UI Guidelines
- Fertigstellung des UI
- Exportieren und Anpassen der Assets für Unity Anwendung
- Dokumentation
- Fertigstellung des Interaktionsprototyps (Adobe xD)

Donnerstag (25.07.19)

- Erstellung der Viruskarten (Prion, Xsekit, Acinetobacter)
- Erstellung Video Interaktionsprototyp (AfterEffects)
- Dokumentation
- App Icon erstellen