Freie Universität Berlin

Bachelorarbeit am Institut für Informatik der Freien Universität Berlin

Arbeitsgruppe Software Engineering

Lehrsoftware für Spieltheorie um Öffentliche Güter erweitern

Jannis Hamann
Matrikelnummer: 4881439
jannis.hamann@fu-berlin.de

Betreuer: Victor Brekenfeld
Eingereicht bei: Prof. Dr. Lutz Prechelt
Zweitgutachter: Prof. Dr. Claudia Müller-Birn

Berlin, 14. Oktober 2020

Zusammenfassung

Hintergrund: In der Evolutionsbiologie werden spieltheoretische Konzepte genutzt, um Kooperationsverhalten zwischen Individuen zu erklären. Um diese Konzepte aus der Spieltheorie besser unterrichten zu können, wurde im Rahmen des Softwareprojekts Kooperationspiele 19/20 eine Anwendung erstellt, mit der Schüler und Studenten diese Konzepte miteinander spielen können.

Ziele: Die Anwendung wurde um das Konzept Öffentliche Güter ergänzt. Ein weiteres Ziel war es, die Verständlichkeit der Anwendung zu erhöhen.

Methoden: Dafür wurden Unklarheiten mit einer heuristischen Analyse und mit Benutzbarkeitstests aufgedeckt. Daraufhin wurde ein neues Design für die Anwendung erstellt, um Öffentliche Güter ergänzt und implementiert. Abschließend wurde die Anwendung durch Benutzbarkeitstests evaluiert.

Eidesstattliche Erklärung

Ich versichere hiermit an Eides Statt, dass diese Arbeit von niemand anderem als meiner Person verfasst worden ist. Alle verwendeten Hilfsmittel wie Berichte, Bücher, Internetseiten oder ähnliches sind im Literaturverzeichnis angegeben, Zitate aus fremden Arbeiten sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungskommission vorgelegt und auch nicht veröffentlicht.

14. Oktober 2020

Jannis Hamann

Inhaltsverzeichnis

1 Einführung

Im Semester 2019/2020 entstand an der Freien Universität Berlin im Softwareprojekt Kooperationsspiele eine Anwendung. Mit dieser Anwendung ist es möglich, Konzepte aus der Spieltheorie zu spielen und dadurch besser zu verstehen. Sie unterstützte bereits das spieltheoretische Konzept Gefangenendilemma. Ziel dieser Arbeit war es, diese Anwendung um das spieltheoretische Konzept Öffentliche Güter zu erweitern.

Bei Benutzbarkeitstests hat sich ergeben, dass es der Anwendung noch an Verständlichkeit mangelt. Das ist kritisch, da Verständlichkeit in der Lehre eine hohe Priorität spielt. Ich habe mir daher als Nebenziel gesetzt, die Verständlichkeit der Anwendung zu erhöhen.

2 Spieltheoretische Grundlagen

Bevor es mit der eigentlichen Durchführung los geht, kläre ich Grundlagen. Das sind zum einen die Konzepte aus der Spieltheorie, die für diese Arbeit relevant sind. Zum andern stelle ich den Stand des Softwareprojekts vor, bevor ich mit der Bachelorarbeit angefangen habe.

Konzepten aus der Spieltheorie gibt es zwei relevante: Das erste ist das Gefangenendilemma, was bereits im Softwareprojekt umgesetzt wurde. Das zweite ist Öffentlichen Güter, welches in meiner Arbeit ergänzt wird.

2.1 Gefangenendilemma

Das Gefangenendilemma wird normalerweise von zwei Spielern gespielt. Beide haben jeweils zwei Optionen. Sie können mit der anderen Person kooperieren oder sie können die andere Person betrügen. Das entscheiden sie ohne zu wissen, wie sich die andere Person entscheiden wird. Nachdem sich beide Personen für eine Option entschieden haben, erhalten beide Punkte.

Wie viele Punkte sie jeweils erhalten ist abhängig davon, für welche Optionen sich beide entschieden haben. Kooperieren beide, erhalten beide drei Punkte. Betrügen beide, erhalten sie stattdessen beiden einen Punkt. Kooperiert eine Person und die andere betrügt, erhalten beide unterschiedlich viele Punkte. Die Person, die kooperiert hat, erhält keine Punkte. Die Person, die betrogen hat, erhält 5 Punkte.

Diese Punkte können angepasst werden, solange sie in einem bestimmten Verhältnis zueinander stehen. Dieses Verhältnis und mehr Details zum Gefangenendilemma beschreiben Axelrod und Hamilton [1].

Im Softwareprojekt wurden zwei Varianten des Gefangendilemmas umgesetzt. Benannt haben wir diese Einfaches Gefangenendilemma und Iteratives Gefangenendilemma.

In beiden Varianten spielt man jetzt mehrere Runden hintereinander. Spieler sammeln in jeder Runde Punkte. Ihr Ziel ist es, nach der letzten Runde so viele Punkte zu haben, wie sie können.

Die beiden Varianten unterscheiden sich darin, wer mit wem spielt. Beim einfachen Gefangenendilemma werden jede Runde zufällige Paare von Spielern gebildet. Beim Iterativen Gefangenendilemmma werden diese Paare nur in der ersten Runde

gebildet und für alle weiteren Runden beibehalten. Spielen eine ungerade Anzahl von Spielern, spielt ein Bot mit. Da man beim Iterativen Gefangenendilemmma immer mit dem gleichen Spieler zusammen spielt, kann es sich hier lohnen Vertrauen aufzubauen und mit der anderen Person zu kooperieren.

2.2 Öffentliche Güter

Bei Öffentliche Güter spielen nicht nur zwei Personen zusammen, sondern die ganze Gruppe. Gespielt wird über mehrere Runden. Jede Runde haben die Spieler die Auswahl zwischen kooperieren oder betrügen.

Kooperieren Spieler, spenden sie etwas von ihren Punkten an die Gruppe. Betrügen sie, spenden sie nichts.

Am Ende einer Runde werden alle Spenden der Spieler zusammengezählt. Dann werden diese vervielfacht und dann alle gleichmäßig ausgezahlt.

Das bedeutet wenn man kooperiert und spendet, dann erhalten alle Spieler gleichmäßig mehr ausgezahlt. Altruistische gesehen lohnt sich also kooperieren, weil alle anderen Spieler dadurch Gewinn machen. Kooperieren alle, machen alle Gewinn.

Wenn man allerdings nicht spendet, kriegt man trotzdem genauso viel ausgezahlt wie alle anderen und verliert keine Punkte durch das Spenden. Spieler, die betrügen erhalten also mehr oder gleich viele Punkte wie die anderen Spieler. Egoistisch gesehen lohnt es sich also zu betrügen. Betrügen allerdings alle Spieler, erhält keiner mehr Punkte.

Diese Konflikt zwischen altruistisch oder egoistisch Handeln ist die Tragik der öffentlichen Güter.

Zu Öffentliche Güter gibt es ebenfalls Varianten. Zum einen kann man Öffentliche Güter mit Aussteigen spielen. In diesem Spielmodus haben die Spieler neben kooperieren und betrügen eine dritte Option. Sie können sich für die Runde aus dem öffentlichen Gut zurückziehen.

Das bedeutet, dass sie nicht in das öffentliche Gut spenden und auch nichts davon ausgezahlt bekommen. Stattdessen erhalten sie eine festgelegte Punktzahl. Diese Punktzahl ist weniger, als was man erhalten würde, wenn alle kooperieren.

Es lohnt sich also nicht auszusteigen, wenn viele Spieler kooperieren. Gibt es aber viele Spieler, die betrügen, kann es sich lohnen auszusteigen.

Die nächste Variante ist öffentliche Güter mit Bestrafung. Hier können Spieler vom öffentlichen Gut ausgeschlossen werden. Das heißt, dass sie keinen Anteil von den Spenden erhalten.

In dieser Variante wird normalerweise offengelegt, wer kooperiert und wer betrügt. Das wird in den anderen Varianten üblicherweise geheim gehalten.

Spieler können dann abstimmen, ob und wen sie vom öffentliche Gut ausschließen möchten. Idealerweise werden Spieler ausgeschlossen, die oft betrügen, weil diese selber wenig spenden und trotzdem einen Teil der Spenden erhalten. Diese Spieler bezeichnen Rankin et al. [4] als Free-Rider (deutsch: Trittbrettfahrer).

Dadurch, dass Trittbrettfahrer bestraft werden können, lohnt es sich mehr zu kooperieren. Was wiederum positiv für alle Spieler ist.

Das Gefangenendilemma und Öffentliche Güter haben also einiges gemeinsam. Beide Konzepte werden über mehrere Runden gespielt. Spieler entscheiden in jeder

Runde, ob sie kooperieren oder betrügen. Bei manchen Varianten können sie noch mehr entscheiden. Am Ende einer Runde erhalten Spieler Punkte. Und Ziel in beiden Konzepten ist es, möglichst viele Punkte zu erhalten.

Für meine Arbeit ist es praktisch, dass das Gefangenendilemma und die öffentliche Güter Gemeinsamkeiten haben. Das bedeutet, dass ich Teile vom Gefangenendilemma im Softwareprojekt für Öffentliche Güter wiederverwenden kann. Um dieses Softwareprojekt geht es im nächsten Abschnitt.

3 Softwareprojekt

Die Grundlage dieser Arbeit entstand im Softwareprojekt Kooperationsspiele im Semester 2019/2020 an der Freien Universität Berlin. Ziel des Softwareprojekts war es eine Software zu bauen, die spieltheoretische Konzepte vermitteln kann. Gewünscht wurde sich diese Software von Dr. Stefan Nessler. Er lehrt Evolutionstheorie und dabei kommt auch die Spieltheorie zum Einsatz. Diese eignet sich, um Kooperationsverhalten zwischen Individuen zu erklären. Victor Brekenfeld hat das Softwareprojekt betreut wurde und betreut auch diese Arbeit.

Stefan Nessler möchte diese Software für seine Lehre nutzen. Die Software soll zum Einsatz kommen, nachdem er die Spieltheoretischen Konzepte im Unterricht vorgestellt hat. Dann sollen diese mit der Software gespielt werden. Ziel davon ist es, die Konzepte praktisch erleben und dadurch besser verstehen und einprägen zu können.

3.1 Verwandte Arbeiten

Brauch man dafür eine Software und gibt es diese nicht schon? Ich beantworte diese Frage, indem ich auf verwandte Arbeiten eingehe.

Spieltheoretische Konzepte kann man auch analog spielen. So beschreiben Ortmann und Colander [3] beispielsweise ein solches Verfahren. Dabei schreiben Spieler ihre Entscheidungen auf Zettel und erhalten Münzen anstatt Punkte.

Wichtig für Stefan Nessler ist aber auch eine Auswertung der Runden. Dafür müsste man dann alle Zettel auswerten. Das ist aufwändig.

Eine Software kann diese Auswertung schnell übernehmen. Es gibt auch bereits Software, mit der man spieltheoretische Konzepte spielen kann. Milinski [2] schreibt, dass er solch eine Software im Klassenraum nutzt. Aus einem Bild geht allerdings hervor, dass dafür eigene Hardware genutzt wird. Stefan Nessler würde aber gerne Geräte nutzen, die schon zur Verfügung stehen. Das können Geräte vor Ort oder private Geräte der Schüler oder Studenten sein. Außerdem wird die Software nicht offengelegt.

Auf Geräten mit einem Browser kann man daher die Webseite `http://https://ncase.me/trust/` nutzen. Dort kann man das Gefangenendilemma spielen und auch verschiedene künstliche Intelligenzen gegeneinander antreten lassen. Das ganze ist auch anschaulich und unterhaltsam aufbereitet. Allerdings kann man dort nicht mit anderen Menschen zusammen spielen. Außerdem wird neben dem Gefangenendilemma kein weiteres Konzept unterstützt.

Das Softwareprojekt soll also etwas umsetzen was es bis dahin nicht gab: eine Möglichkeit spieltheoretische Konzepte mit anderen Menschen zeit-effizient zu spielen.

3.2 Zielgruppen

Das Softwareprojekt hat zwei Zielgruppen: auf der einen Seite eine Lehrkraft, welche im Verlauf der Arbeit als Admin bezeichnet wird. Auf der anderen Seite Schüler oder Studenten, die als Spieler bezeichnet werden.

Der wichtigste Admin ist Stefan Nessler, weil er der gleichzeitig der Auftraggeber der Software ist und die Software als erster nutzen wird. Ihm ist dabei wichtig, dass die Anwendung fachlich korrekt ist. Außerdem soll die Anwendung keine Vorurteile gegenüber bestimmten Entscheidungen verstärken. Z.B. sollte betrügen nicht als etwas besonders schlechtes dargestellt werden. Spieler sollen seiner Meinung nach ihre Entscheidung so treffen, dass sie maximal viele Punkte erreichen.

Außerdem ist ihm wichtig, dass kein Spieler ausgeschlossen wird. Z.B. haben wir daher in der Software Spieler gegendert, sprich Spieler*innen verwendet. Er selbst wird die Anwendung an einem Rechner nutzen und möchte die Ergebnisse auf einem Beamer teilen. Damit die Anwendung auf einem Beamer gut dargestellt wird, haben wir uns daher für ein kontrastreiches Farbschema entschieden.

Stefan Nessler möchte die Anwendung im Bereich der Evolutionsbiologie einsetzen. Als Admin kommen daher auch andere Lehrer oder Universitätsmitarbeiter in der Biologie in Frage. Denkbar ist auch, dass die Software später auch außerhalb der Biologie in Bereichen eingesetzt wird, die sich mit Spieltheorie beschäftigen, wie z.B. in Wirtschaftsveranstaltungen.

Spieler werden in der Regel Schüler oder Studenten sein. In der Schule können spieltheoretische Konzepte circa ab der zehnten Klasse im Biologieunterricht unterrichtet werden. An Universitäten kann man die Anwendung im Bachelor ab dem dritten Semester oder im Master nutzen. Wahrscheinlicher ist allerdings, dass sie im Master genutzt wird, weil Stefan Nessler die Anwendung eher in Seminaren nutzen würde und dafür im Master mehr Raum ist.

Spieler gehen also zur Schule oder studieren, haben biologisches Vorwissen und sind in der Regel mindestens 16 Jahre alt.

3.3 Seiten

Das Softwareprojekt läuft im Browser und besteht aus mehreren Seiten. Diese stelle ich vor, um Änderungen in meiner Arbeit an diesen Seiten besser erklären zu können. Dafür beschreibe ich, was Nutzer auf diesen Seiten machen können.

Wenn man die Anwendung startet, landet man erst einmal auf der Startseite. Dort kann man ein Spiel erstellen. Das ist für den Admin gedacht. Alternativ kann man als Spieler einem erstellten Spiel beitreten.

Von dort aus kann man entweder als Admin ein Spiel erstellen und dann starten oder man kann als Spieler einem erstellten Spiel beitreten.

Die restlichen Seiten werden in Admin- und Spieler-Seiten unterschieden. Die Admin-Seiten werden in der Regel vorne an einem Beamer dargestellt. Die Spieler

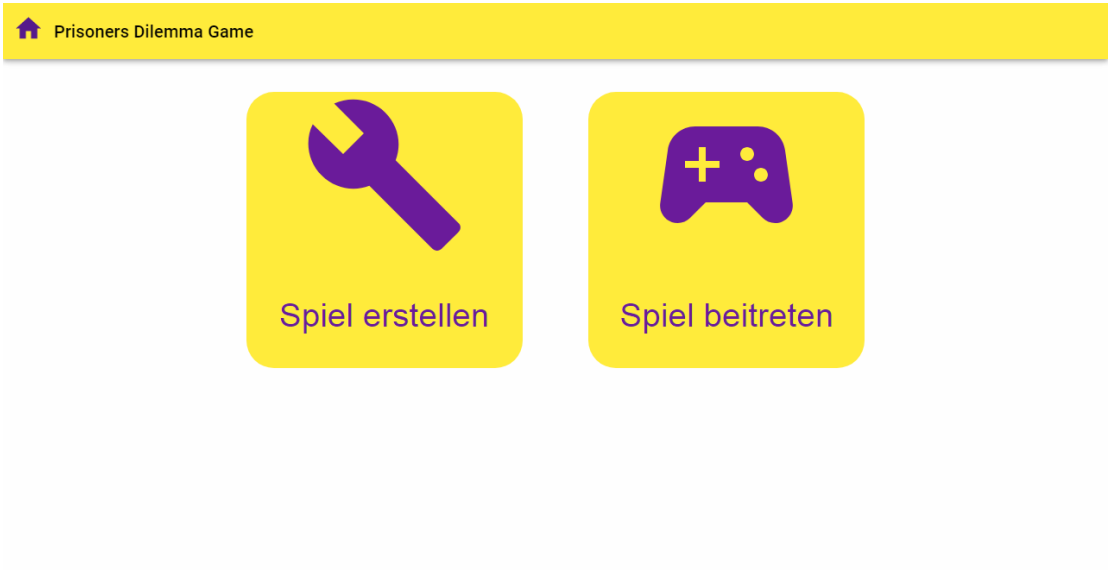


Abbildung 1: Startseite

können ihre Seiten auf einem Smartphone oder Laptop ansehen.

Wenn man als Admin ein Spiel erstellen will, landet man auf der Einstellungenseite. Hier kann man aussuchen, welches Spieltheoretisches Konzept man spielen will. Außerdem kann man Optionen anpassen. Z.B. kann man einstellen, wie viele Runden gespielt werden sollen. Oder man kann einstellen, wofür Spieler wie viele Punkte erhalten. Hat man sich für Optionen entschieden, kann man das Spiel erstellen.

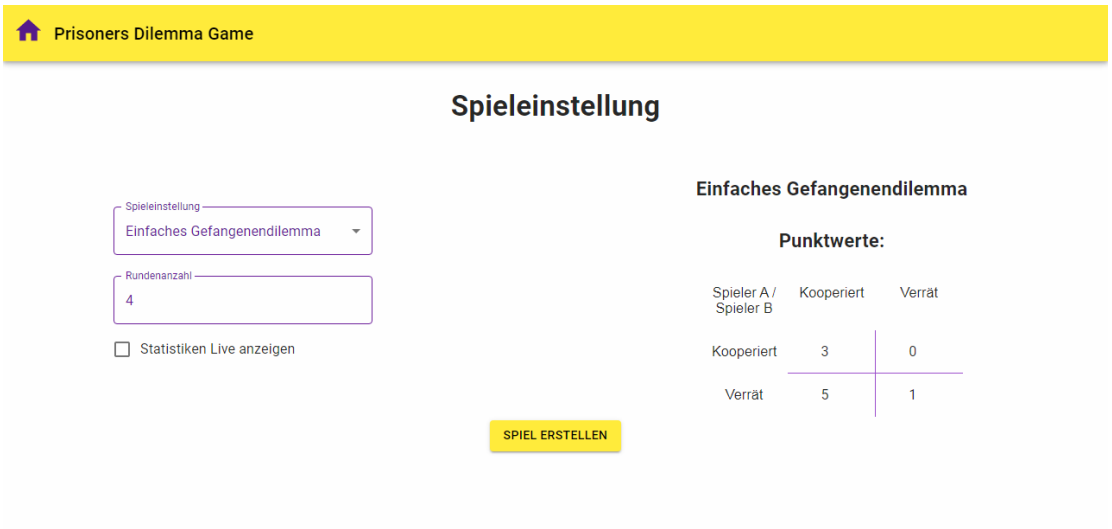


Abbildung 2: Einstellungenseite

Dann landet man auf der Admin-Warteraumseite. Hier wird Spielern angezeigt, wie man dem Spiel beitreten kann.

Das Beitreten funktioniert über einen Spielcode, den die Spieler auf der Beitretenseite eingeben können.

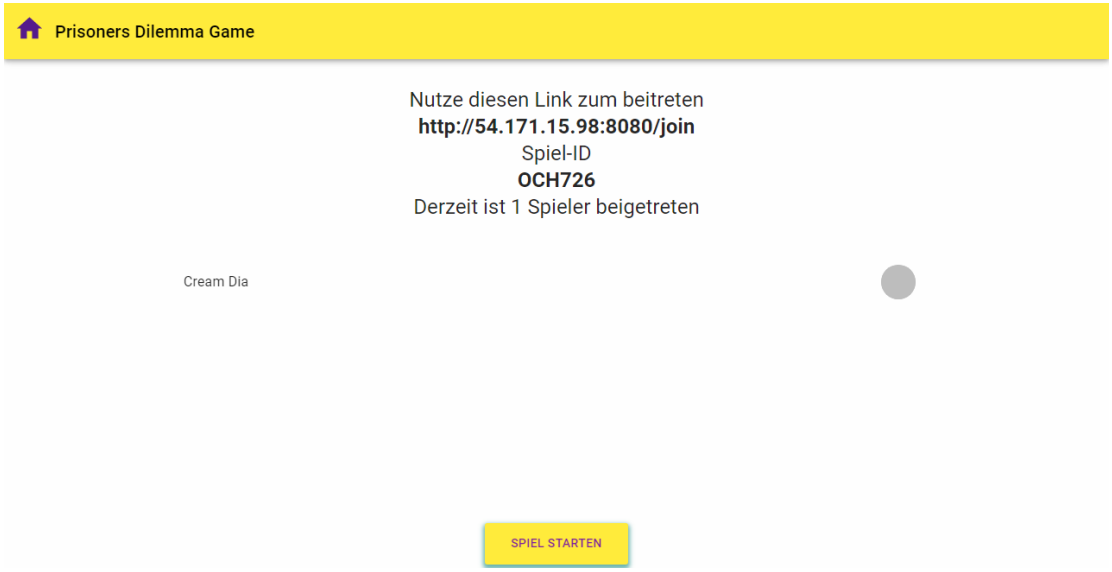


Abbildung 3: Admin-Warteraumseite

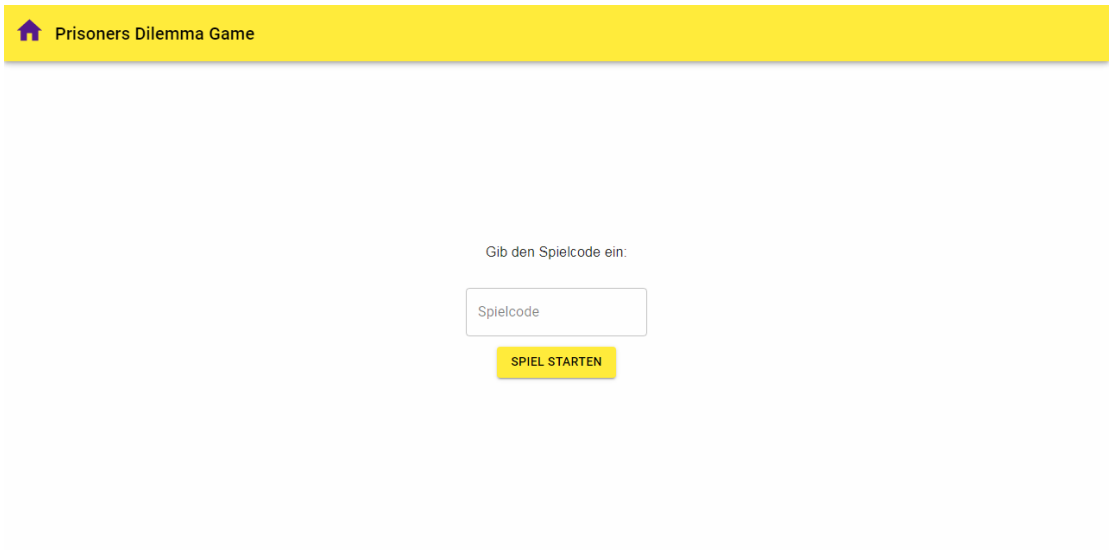


Abbildung 4: Beitretenseite

Nachdem sie das getan haben, landen sie auf der Spieler-Warteraumseite. Dort warten sie darauf, dass das Spiel vom Admin gestartet wird. Bis dahin können Sie die Zeit nutzen und sich durch Infotexte über das kommende Spiel informieren.

Sobald das Spiel vom Admin gestartet wurde, landen sowohl der Admin als auch die Spieler auf ihren jeweiligen Spielseiten.

Der Admin kann über die Admin-Spielseite das Spiel steuern. Er kann hier eine neue Runde starten oder das Spiel beenden. Außerdem sieht er hier verschiedene Spieldaten, wie z.B. welche Spieler bereits abgestimmt hat oder wie viele Punkte sie haben. Da der Admin kontrolliert, wann die nächste Runde gestartet wird, kann er das Spiel auch unterbrechen. Das kann genutzt werden, um etwas zu erklären oder die Resultate der Runden durchzugehen. Da er sieht, wer noch nicht abgestimmt hat,

Abbildung 5: Spieler-Warteraumseite

kann er Spieler ermahnen, die zu lange für ihre Entscheidung brauchen.

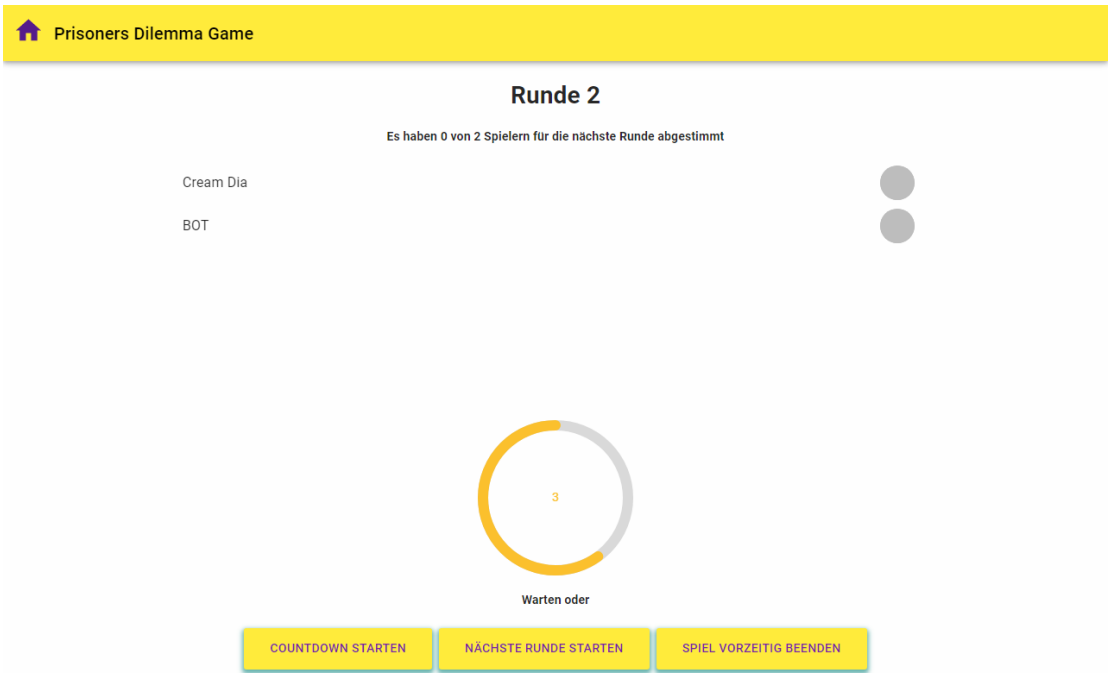


Abbildung 6: Admin-Spielseite

Spieler können auf der Spielseite zum einen Entscheidungen für die spieltheoreti-

schen Konzepte treffen wie kooperieren oder betrügen. Außerdem sehen sie Statistiken über das Spiel wie z.B. ihren aktuellen Punktestand.

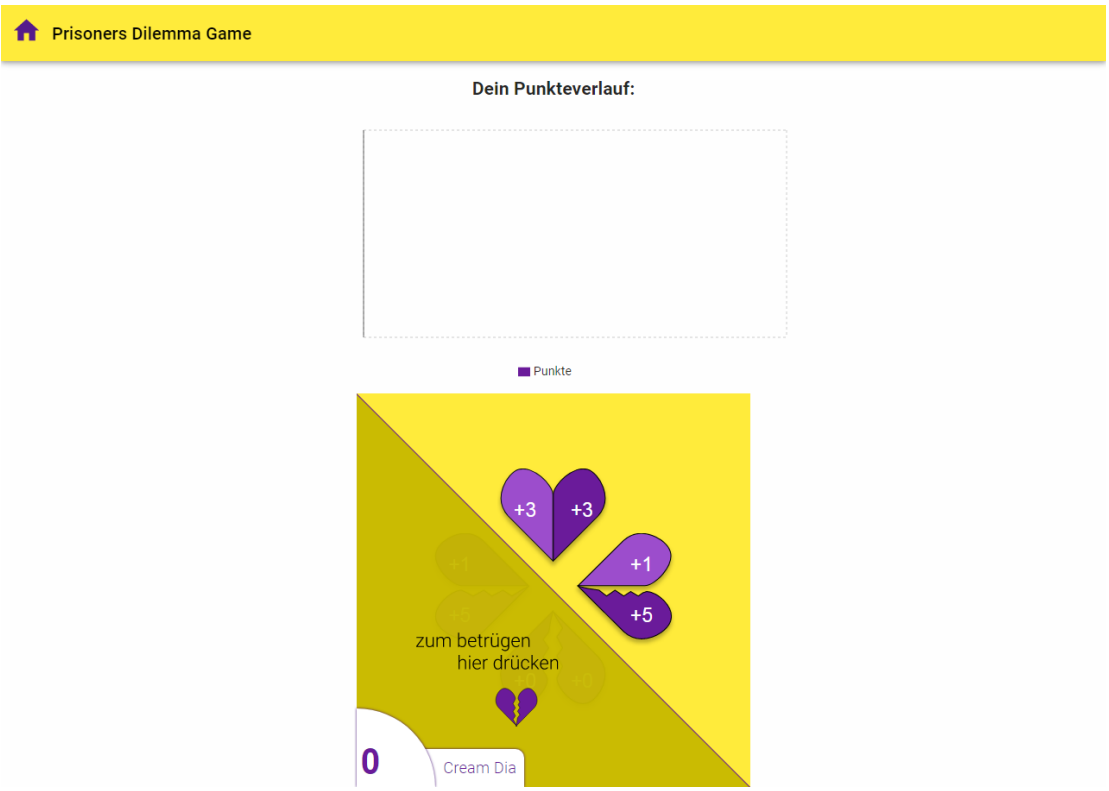


Abbildung 7: Spieler-Spielseite

Wenn das Spiel vorbei ist, landen Admin und Spieler auf ihren jeweiligen Ergebnisseiten. Dort sehen sie verschiedene Diagramme, die sie über die Ergebnisse des Spiels informieren.

3.4 Projektstruktur und Entwicklung

Die Seiten sind das was die Nutzer letztendlich zu Gesicht bekommen. Für die Arbeit ist aber auch wichtig zu verstehen, was im Hintergrund abläuft. Daher gehe ich jetzt auf den Programmcode und die Projektstruktur ein.

Der Programmcode ist aufgeteilt in zwei Bereiche: Frontend und Backend. Der Backendteil ist für die Spiellogik zuständig. Dieser ist in in der Programmiersprache Python geschrieben. Dabei wird Flask genutzt, um einen Server zu erstellen.

Der Frontendteil zeigt das Spiel den Nutzern an und ist in Programmiersprache Typescript geschrieben. Dabei werden React und Material UI genutzt, um die Benutzeroberfläche aufzubauen.

Jeder Bereich wurde von einem eigenständigen Team erstellt. Das Frontendteam bestand aus fünf und das Backendteam aus vier Studenten.

Ich war Teil des Frontendteams. Dabei war ich hauptsächlich für das Design und die Umsetzung der Spieler-Spielseite und der Hauptseite verantwortlich. Außerdem habe ich mich mit dem Farbschema der Anwendung auseinandergesetzt.

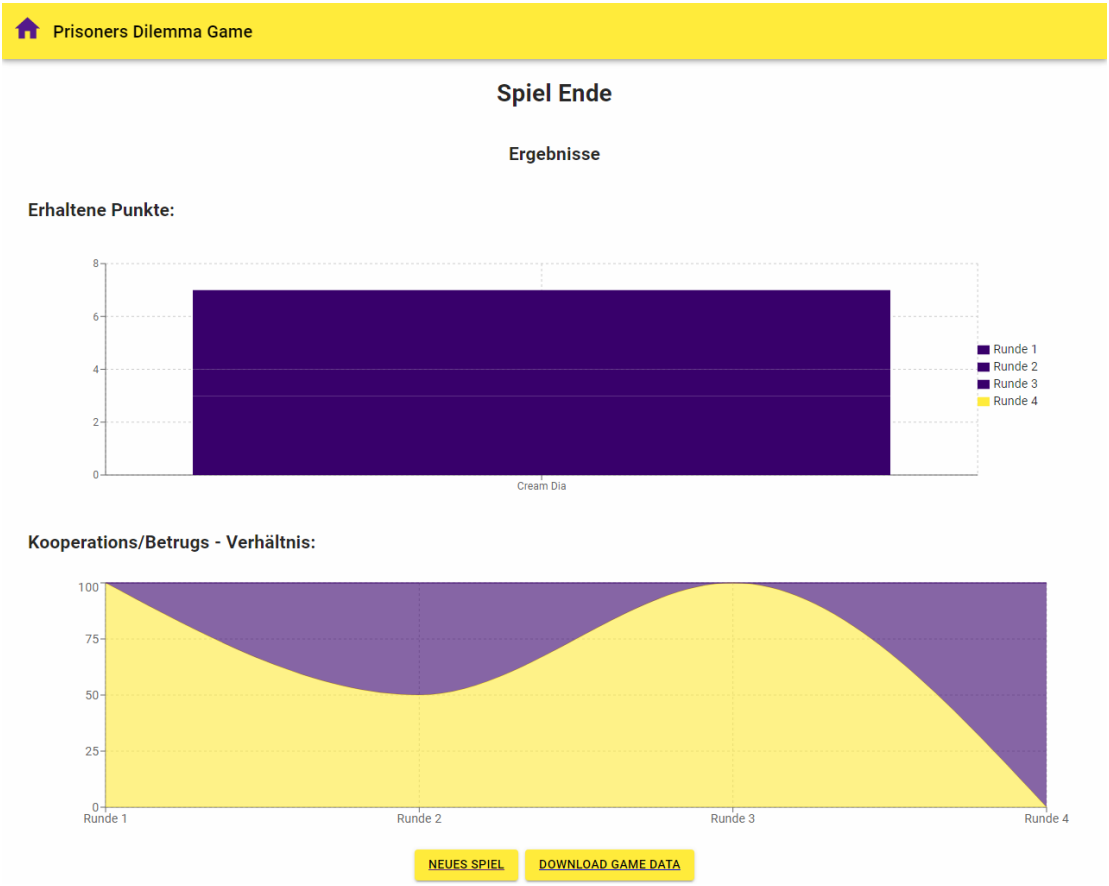


Abbildung 8: Admin-Ergebnisseite

Abbildung 9: Spieler-Ergebnisseite

Im Frontendteam war auch Jamie Schenker. Er möchte zum Stand der Abgabe dieser Arbeit auch seine Masterarbeit über das Softwareprojekt schreiben. Sein aktuelles Ziel dafür ist es, die Anwendung so umzubauen, dass sie als Webseite im Internet laufen kann. Aktuell kann sie nur in einem lokalen Netzwerk genutzt werden.

Im Backend ist eine API Schnittstelle definiert, wodurch das Frontend mit dem Backend kommuniziert. Diese Schnittstelle schätze ich als gut strukturiert und dokumentiert ein.

In dieser API werden auch Modelle der Anwendung und ihre Eigenschaften festgehalten. Diese Modelle definieren, was ein Spieler, ein Spiel und eine Runde ist.

Ein weiteres Modell ist der Spielplatz. Auf diesem kann man mehrere Spiele hintereinander mit den gleichen Spielern spielen und auswerten.

Die gesamten Daten werden in Backend in einem einzigen Dictionary festgehalten. Über verschiedene Schlüssel gelangt man dann an die entsprechenden Daten. So kommt man an die erreichte Gesamtpunktzahl eines Spielers über folgenden Befehl:

```
playground.PLAYGROUND[playgroundID]["player_array"][PlayerID]["
    pointsTotal"]
```

Ich sehe das als Problem. Da diese Zugriffe sehr lang werden können, wird der Quelltext unübersichtlich. Diese Vorgehensweise ist auch fehleranfällig. Ein Grund dafür ist, dass man sich nicht sicher sein kann, was tatsächlich in dem Dictionary steht, weil Python keine typisierte Sprache ist.

Ein weiteres Problem vom Backend ist, dass Teile des Quelltextes oft wiederholt werden. Das bedeutet, dass wenn man eine Änderung durchführen möchte, muss man diese mehreren Stellen wiederholen.

Zur Verdeutlichung gebe ich ein Beispiel. Während meiner Arbeit habe ich im Backend eine Änderung durchgeführt. Und zwar habe ich den Spielcode, welcher ursprünglich aus sechs Ziffern und Buchstaben bestand, auf vier Ziffern beschränkt. Dafür musste ich 66 Zeilen des Quelltextes anpassen. Drei Zeilen waren für die eigentliche Änderung der Logik verantwortlich. Die 63 anderen waren die gleiche Abfrage, welche im Quelltext verteilt ist: Und zwar wird an 63 Stelle abgefragt, ob der Spielcode sechs Stellen lang ist.

Meine Einschätzung des Backends ist also, dass es zwar gut strukturiert und dokumentiert ist, allerdings ist die Qualität des Quelltextes mangelhaft.

Das Frontend ist in verschiedene Bereiche aufgeteilt: Der Hauptteil des Quelltextes liegt im Bereich der Komponenten. Komponenten sind ein Teil der Benutzeroberfläche und können wiederum andere Komponenten enthalten. So ist z.B. ein Button eine Komponente. Ebenfalls ist jede Seite eine Komponente und stellt dann mehrere andere Komponenten dar.

In einem weiteren Bereich wird das Modell des Backends gespiegelt. Dort werden also ebenfalls Spieler, Spiel, Runde und Spielplatz definiert.

Dann gibt es den Bereich API, der alle Zugriffe auf das Backend definiert.

Und schließlich gibt es einen Hilfsbereich. Dieser hat den Zweck, bestimmte Daten als Cookies auf den Geräten der Nutzer zu speichern, damit die Spieldaten, nachdem der Browser geschlossen wurde, erhalten bleiben.

Während des gesamten Softwareprojektes wurde zwar Feedback von Stefan Nessler eingeholt, allerdings wurde kein Benutzbarkeitstest außerhalb des Projekts durch-

Abbildung 10: Änderung der Länge des Spielcodes

geführt. Daher wollte ich das in dieser Arbeit nachholen.

4 Planung

Bevor es mit der eigentlichen Arbeit los ging, habe ich einen Plan erstellt, anhand dessen ich mich durcharbeiten wollte. Dieser Plan ist in verschiedene Phasen aufgeteilt. Für jeder dieser Phasen habe eine bestimmte Wochenzahl an Arbeitszeit eingeplant. Außerdem habe ich auch Pufferzeit eingeplant. Mehr Details zu Planung befinden sich im Anhang (1).

Los ging es mit der Analysephase. Dabei hatte ich zwei Ziele. Zum einen wollte ich herausfinden, was Öffentliche Güter ist und was das für die Anwendung bedeutet. Und zum anderen wollte ich die bisherige Anwendung analysieren, um zu schauen, wie sinnvoll es ist, auf dieser Anwendung aufzubauen und diese zu erweitern.

Die Erkenntnisse, die ich dort erlangen wollte, wollte ich dann in der Designphase nutzen. Dabei wollte ich schauen, ob man Öffentliche Güter in das bisherige Design integrieren kann oder man zumindest für die Spielansicht ein ein neues Design erstellen muss. Teil der Designphase war auch Feedback zu bekommen von unter anderem Stefan Nessler.

Danach sollte ein Implementierungsphase folgen die letztendlich das Design umsetzt. Auch das sollte wieder getestet werden.

Die nächste Phase sollte wieder eine Implementierungsphase sein, nämlich die zweite Iteration. Hier habe ich eingeplant Feedback aus der Implementierungsphase umzusetzen, eventuell auch im Design. Abschießen sollte die Implementierungsphase mit einem finalen Test mit Stefan Nessler.

Die letzte Phase sollte das eigentliche Schreiben der Arbeit sein.

Rückblickend würde ich sagen, dass die Planung an sich geholfen hat Struktur zu schaffen und die Laufzeiten Blick zu haben. Allerdings habe ich gerade für die

letzten Phasen verhältnismäßig zu wenig Zeit eingeplant. Dadurch sind Phasen wie die zweite Iteration etwas kurz gekommen.

5 Analyse

Das erste Ziel war es herauszufinden, was Öffentliche Güter bedeutet. Dafür habe ich zuerst einmal Stefan Nessler interviewt, um herauszufinden was Öffentliches Güter für ihn bedeutet. Er hat mir auch Tipps für wissenschaftliche Quellen geben, in denen ich das weiter recherchieren kann. Auf dieser Grundlage ist der Teil Spieltheoretische Grundlagen der Öffentliche Güter (2.2) entstanden. Außerdem habe ich Anforderungen an Öffentliche Güter (A.2) definiert.

5.1 Heuristische Analyse

Erstmal habe ich selber die Anwendung auf Benutzerfreundlichkeit untersuchen. Problem dabei war, dass ich selber an der Anwendungen mitgearbeitet habe und damit auch vertraut mit der Anwendung war. Daher hätte ich mit einen Benutzbarkeitstest gar wenig aussagekräftigen Probleme identifizieren können. Deshalb habe ich mich stattdessen für eine heuristische Analyse entschieden, weil diese Kriterien mit sich bringt anhand dessen man eine Anwendung untersuchen kann.

Ich habe mich für die Nielsen Heuristik (https://www.nngroup.com/articles/ten-usability-heuristics/) entschieden, weil ich mit dieser schon Erfahrung hatte. Die Nielsen Heuristik definiert 10 Kriterien, anhand dessen man eine Anwendung untersuchen kann:

1. Visibility of system status: The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.
2. Match between system and the real world: The system should speak the users language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.
3. User control and freedom: Users often choose system functions by mistake and will need a clearly marked emergency exit to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.
4. Consistency and standards: Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.
5. Error prevention: Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.
6. Recognition rather than recall: Minimize the users memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

7. Flexibility and efficiency of use: Accelerators — unseen by the novice user — may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

8. Aesthetic and minimalist design: Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

9. Help users recognize, diagnose, and recover from errors: Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

10. Help and documentation: Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the users task, list concrete steps to be carried out, and not be too large.

Dafür bin ich jede Seite der Anwendung durchgegangen und habe jeweils geschaut, ob es meiner Meinung nach Verstöße gegen diese Kriterien gibt.

Besonders auffallend waren dabei drei Dinge: Es gab immer wieder Probleme mit der Darstellung von dem System Status (Kriterium 1). Z.B. wurde Spielern nach einer Runde des Gefangenendilemmas nicht mitgeteilt, wie sich der Mitspieler in der Runde entschieden hat.

Außerdem war die Anwendung inkonsistent (Kriterium 4). So wurde z.b. an manchen Stellen englisch statt deutsch benutzt, es gab verschiedene Terminologien für kooperieren und betrügen, und Buttons sehen verschieden aus.

Ein weiteres Problem war, dass die Anwendung zu größten Teilen schlecht auf Smartphones dargestellt wurde. Das ist ungünstig, weil anzunehmen ist, dass viele Spieler über Smartphone spielen wollen.

Alle Resultate der heuristischen Analyse sind im Anhang (A.3).

Zum Verfahren möchte ich noch anmerken, dass ich mir in einigen Stellen unsicher war, ob ich ein identifiziertes Problem dem richtigen Kriterium aus der Nielsen Heuristik zugewiesen habe. Aber auch wenn die Zuweisung vielleicht nicht immer korrekt war, glaube ich dass sie trotzdem geholfen hat Probleme aufzudecken. Denn durch das Verfahren konnte ich aus einem anderen Blickwinkel auf die Anwendung schauen und dadurch Probleme identifizieren.

5.2 Benutzbarkeitstests

Als nächstes habe ich Benutzbarkeitstests durchgeführt, um zu sehen wie die Anwendung bei anderen ankommt.

Dafür habe ich mir überlegt, mit wem ich diese Tests durchführen kann. Da die Anwendung lokal läuft, habe ich mich entschieden vor Ort zu testen. Mit wem ich vor Ort testen kann ist durch Corona eingeschränkt. Daher habe ich mit Bekannten getestet.

Bekannte haben den Nachteil, dass ich nicht für ihre Objektivität garantieren kann. Außerdem kenne ich keine Schüler, konnte also nicht den gesamten Bereich der Zielgruppe abdecken.

Vorteil von Bekannten ist, dass die Rekrutierung wenig Zeit in Anspruch genommen hat. Außerdem habe ich mich sicherer gefühlt, Rückfragen zu stellen.

Die Benutzbarkeitstest habe ich mit drei Bekannten durchgeführt. Diese sind zwischen 22 und 24 Jahre alt. Zwei davon studieren im Bachelor. Der dritte fängt gerade seinen Master an. Sie studieren alle keine Biologie. Allerdings kannte sich einer der drei sich mit dem Gefangenendilemma aus.

Bevor es mit dem eigentlichen Test los ging habe ich ein Skript (A.4) erstellt, um einen klaren Ablauf zu haben. Darin habe ich festgehalten, was ich vor dem eigentlichen Test erklären möchte und was die Aufgabe während des Tests ist. Außerdem habe ich mir Anschlussfragen überlegt, die ich im Anschluss besprechen wollte.

Zusätzlich habe ich eine Einverständniserklärung erstellt, in der unter anderem festgehalten wurde, dass ich in Bildschirm und Ton aufnehmen darf, um diese später auswerten zu können. Das hatte den Vorteil, dass ich mich während der Benutzbarkeitstest auf Fragen konzentrieren konnte, weil ich nebenbei keine Notizen machen musste.

Die Benutzbarkeitstest liefen folgendermaßen ab: Die Tester haben unabhängig voneinander getestet. Dabei haben sie gleichzeitig zwei Rollen übernommen: Einmal die des Admins und die eines Spielers. Dafür hatten sie in zwei Browsern jeweils einmal die Adminseite und einmal die Spielerseite offen. Sie haben dann das Gefangendilemma aus beiden Perspektiven gleichzeitig gespielt. Dabei sollten sie jeden Gedanken laut äußern.

Die Fragen, die ich mir für den Anschluss überlegt hatte, hatten sich bei allen dann schon erübrigt, weil sie während des Tests diese schon direkt oder indirekt beantwortet hatten.

Was bei dem Test herausgekommen ist, ist dass ein großer Teil der Anwendung unverständlich ist. Ein Teil davon mag darauf zurückzuführen sein, dass den Testern nötiges Vorwissen der Spieltheorien gefehlt hat. Das würde im Unterrichtskontext vorher ja nochmal erklärt werden. Z.B. waren nicht alle Einstellungen auf der Einstellungenseite klar. Normalerweise sollten Admins diese Einstellungen aber verstehen, weil sie die Parameter des Gefangenendilemmas kennen sollten.

Es gab aber genügen Probleme, die auch bei entsprechendem Vorwissen unverständlich gewesen wären. Z.B. hat man beim Gefangenendilemma gegen einen Bot gespielt. Bei der Auswertung war dann der Bot aber verschwunden. Das heißt man wusste zwar wie viele Punkte man selber bekommen, aber nicht wie viel der Bot bekommen hat. Daher war unklar, ob man gewonnen hatte oder nicht. Alle Ergebnisse der Benutzbarkeitstests sind im Anhang (A.6).

Als nächstes habe ich alle Probleme aus der heuristischen Analyse und den Benutzbarkeitstests zusammengefasst. Dafür habe ich eine Tabelle erstellt. Diese Tabelle ist im Anhang unter (B.1). In der Tabelle stehen die Beschreibungen des Probleme; außerdem die Seiten, auf der das Problem auftritt und Ideen, wie man die Probleme lösen kann. Außerdem habe ich für jedes Problem geschätzt, wie stark ich den Einfluss dieses Problem sehe, wie oft das Problem auftritt und wie aufwändig es sein könnte, das Problem zu beheben. Daraus habe ich eine Priorität des Problems her-

geleitet. Daraus habe ich hergeleitet, welche Probleme ich zuerst angehen sollte. Auf weitere Probleme gehe ich im Design ein, weil ich mit diesem rechtfertige, welche Änderungen ich durchgeführt habe.

Ich fasse zusammen, dass es mit dem Ergebnis des Softwareprojektes zwar möglich war, das Gefangenendilemma komplett zu spielen. Allerdings war es an vielen Stellen unverständlich. Für die Lehre hat Verständlichkeit aber eine hohe Priorität. Daher habe ich mir als Nebenziel gesetzt, die Verständlichkeit der Anwendung zu erhöhen. Denn es nützt meiner Meinung nach wenig, Öffentliche Güter zu ergänzen, wenn die Anwendung an und sich noch gar nicht verstanden wird.

6 Design

Im folgenden gehe ich auf die Änderungen in der Designphase ein. Zuerst gehe ich auf allgemeine Änderung ein. Dann beschreibe ich spezielle Änderungen bestimmter Seiten. Als letztes beschreibe ich, wie ich Öffentliche Güter integriert habe. Das Design habe ich mit dem Programm Figma konzipiert.

Prinzipiell habe ich versucht, so viele bereits bestehende Komponenten aus mit Material UI zu nutzen wie geht. Material UI Komponenten wurden zwar schon an vielen Stellen eingesetzt, allerdings nicht konsequent und konsistent.

Außerdem habe ich das Farbschema angepasst. Die ursprünglichen Farben habe ich beibehalten. Allerdings habe ich diese so umverteilt, dass wichtige Elemente auf Seiten hervorgehoben werden.

Da unsere Anwendung bisher nicht gut auf mobil funktioniert hat, habe ich mich beim Design vor allem darauf konzentriert, dass die Anwendung mobil dann funktionieren soll. Für die Ansicht auf Desktopbildschirmen habe ich kein separates Design gebaut. Die Idee war, stattdessen dann auch das mobile Design zentriert darzustellen und evtl. hoch zu skalieren.

Auf der Hauptseite habe ich die Buttons des Menüs durch Buttons von Material UI ersetzt.

Abbildung 11: Hauptseite

Außerdem habe ich Anpassungen der Kopfleiste gemacht. Zum einen habe ich den Namen der Anwendung, Prisoner's Dilemma Game, geändert, weil ja auch Öffentliche Güter hinzugefügt werden sollte und die Anwendung auch deutsch sein sollte. Ich habe mich für den Namen Spieltheorie Labor entschieden. Dieser soll zum einen hervorheben, dass es bei dieser Anwendungen um Spieltheorie geht. Labor sollte für mich darstellen, dass man auf dieser Anwendung mit der Theorie experimentieren kann. Mit dem Namen Spieltheorie Labor habe ich dann ein Logo erstellt.

Den Platz neben dem Logo wollte ich nutzen, um den aktuellen Status der Anwendung darzustellen. Auf der Hauptseite habe ich dort den Text Spiel mit der Theorie vorgesehen. Das soll den Zweck der Anwendung hervorheben, weil das in Benutzbarkeitstests teilweise unklar war.

Bei den Einstellungen habe ich die Auswahl des Spielmodus beibehalten. Alle anderen Einstellungen sind optional und werden in den meisten Fällen nicht geändert. Daher habe ich sie hinter erweiterten Einstellungen versteckt. Damit wollte ich hervorheben, dass man diese Einstellungen nicht ändern muss.

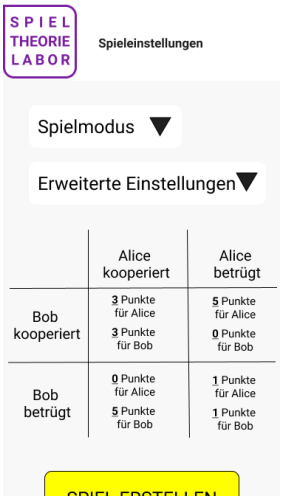


Abbildung 12: Einstellungenseite

Außerdem habe ich die Einstellung "Statistiken live anzeigen" entfernt. Diese hat gesteuert, ob auf der Admin-Spielseite Statistiken angezeigt werden oder nicht. Ich habe die Einstellung entfernt und auf der Admin-Spielseite die Statistiken so dargestellt, dass man sie ein- und ausblenden kann. Das hat den Vorteil, dass man sich umentscheiden kann.

In den Einstellungenseite befindet sich die Punktematrix, bei der man einstellen kann, wie viele Punkte Spieler für welche Entscheidungen bekommen. Punktmatrizen befinden sich auch noch auf zwei weiteren Seiten: auf der Spieler-Warteraumseite und der Spieler-Spielseite. Die Darstellung der Punktematrix hat sich stark unterschieden.

Mein Ziel war es, für alle diese drei Ansichten eine möglichst ähnliche Darstellung zu entwickeln. Dadurch wollte ich einen höheren Wiedererkennungswert und und auch eine leichtere Verständlichkeit erreichen. Am verständlichsten von diesem Matrizen war bisher die Darstellung in Tabellenform auf der Einstellungenseite, weil diese explizit benennt, wer unter welchen Voraussetzungen wie viele Punkte bekommt.

Allerdings konnte man in der Tabellenform bisher nur ablesen wie viele Punkte

man selber bekommt. Dadurch kann man auch indirekt schießen, wie viele Punkte der Mitspieler bekommt, in dem man sich gedanklich selbst mit dem Mitspieler tauscht.

Diesen gedanklichen Schritt wollte ich entfernen. Dafür habe ich in der Tabelle explizit geschrieben, wofür man selbst Punkte bekommt und wie viele Punkte dann der Mitspieler erhält.

Auf der Admin-Warteraumseite habe ich die Darstellung der einzelnen Spieler versteckt, weil für den Admin vor allem die Anzahl der beigetretenen Spieler wichtig ist um ein Spiel starten zu können. Ich habe es nicht komplett herausgenommen, weil es Fälle gibt, wo doch nachschauen möchte, wer schon beigetreten ist. Z.B. wenn man Spieler ermahnen möchte, die noch nicht beigetreten sind.

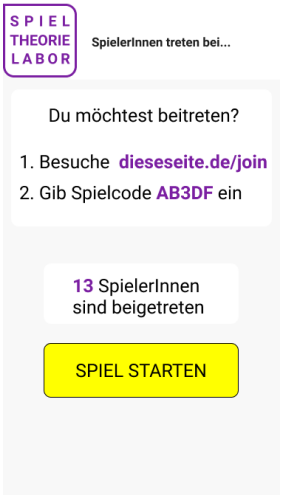


Abbildung 13: Admin-Warteraumseite

Außerdem habe ich das Texte, die erklären wie man beitreten soll überarbeitet, weil bei Tests teilweise die Spielcodes anstatt des Links in den Browser eingegeben wurden. Das kann auch daran gelegen haben, dass auf verschiedenen Seiten der Spielcode anders benannt wurde. Diese Inkonsistenz habe ich ebenfalls behoben.

Die Beitreten-Seite ist im Wesentlichen gleich geblieben.

Auf der Spieler-Warteraumseite habe ich den Erklärungstext überarbeitet. Ich habe den Hintergrund des Gefangenendilemma entfernt, weil das in der Regel schon im Unterricht erklärt wird und für das Spielen selbst unwichtig ist. Dieser Einführungstext hatte bei Tests auch eher verwirrt als geholfen.

Stattdessen habe ich ergänzt, wie das Spiel an sich funktioniert, damit Spieler besser auf das folgende Spiel vorbereitet sind. Bei der Struktur habe ich mich dabei an Brettspielerklärungen orientiert. Dabei habe ich wichtige Konzepte extra hervorgehoben.

Auf der Admin-Spielseite habe ähnlich wie au der Admin-Warteraum die Namen der Spielernamen versteckt. Ebenfalls wird stattdessen nur angezeigt, wie viele Spieler sich bereits entschieden haben. In Tests hatte sich ergeben, dass das die relevantere Information ist.

Bei den Änderungen der Einstellungenseite habe ich schon angesprochen, dass man hier jetzt die Statistiken anzeigen oder verstecken kann.

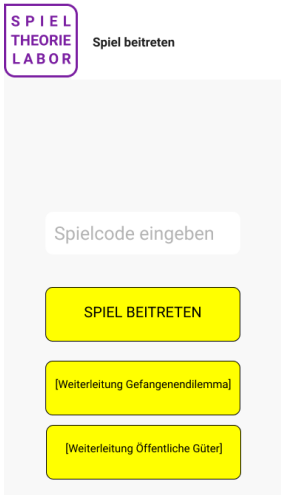


Abbildung 14: Beitreten-Seite

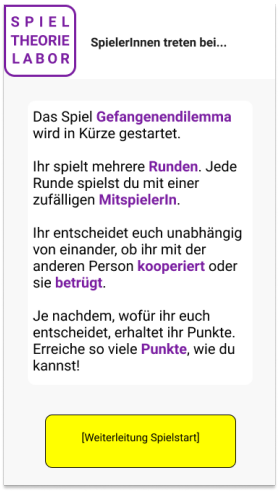


Abbildung 15: Spieler-Warteraumseite Gefangenendilemma

Ich habe mich ursprünglich dafür entschieden, den Spiel beenden-Button subtiler zu gestalten, weil dieser selten benutzt wird. Später hat sich herausgestellt, dass das der einzige subtile Button gewesen wäre. Aus Konsistenzgründen habe ich diesen später wie alle anderen Buttons gestaltet.

Ich habe den "Warten oder" Text entfernt, weil sich durch Warten nichts geändert hat.

Außerdem habe ich die Countdown starten- und Nächste Runde starten-Buttons zu einem zusammengefasst. Beide sind dafür da die nächste Runde zu starten. Der Unterschied zwischen beiden war, dass Countdown starten einen countdown auf der Adminseite dargestellt hat. Dieser hat 5 Sekunden herunter gezählt. In dieser Zeit konnten sich die Spieler noch mal umentscheiden und nach Ablauf der Zeit wurde eine neue Runde gestartet.

Für die Zusammenfassung habe ich mich entschieden, den Countdown zu entfernen. Der Countdown bekam zwar positives Feedback. Das war aber immer damit

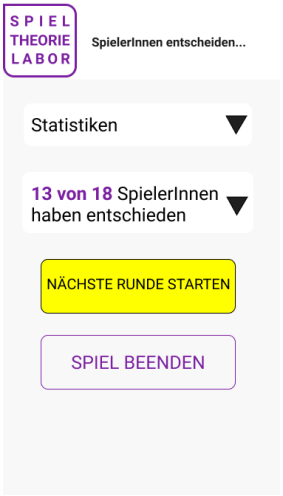


Abbildung 16: Admin-Spielseite

verbunden, dass der Countdown auch auf Spielerseiten angezeigt werden sollte. Das hätte meiner Einschätzung nach einen größeren Implementierungsaufwand bedeutet, als es der Nutzen rechtfertigt.

Auch auf der Spieler-Spielseite habe ich einige Änderungen durchgeführt: oben in der Kopfleiste sollte immer angezeigt werden, was gerade der Status der Anwendung ist. Z.b. sollte dort stehen, wenn man sich gerade entscheiden soll, oder wenn man darauf wartet, dass sich andere Spieler entscheiden oder wenn die Runde vorbei ist und das Ergebnis feststeht.

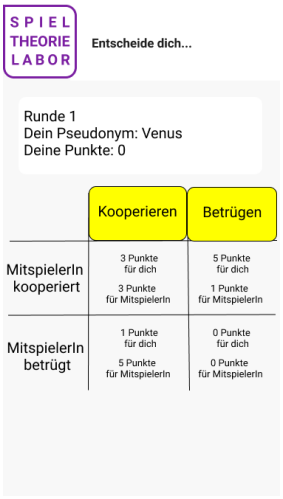


Abbildung 17: Spieler-Spielseite Gefangenendilemma

Außerdem habe ich den Punkteverlauf entfernt. Dieser war verwirrend, wenn das Spiel in der ersten Runde war, weil dann dort nichts angezeigt wurde. Außerdem waren die Informationen, die dort angezeigt wurden, nicht besonders interessant für die Spieler. Dort wurde angezeigt, wie viele Punkte man in welcher Runde bekommen hat. Allerdings hat die Spieler vor allem interessiert, wie viele Punkte sie in der letzten

Runde bekommen und wie viele Punkte sie insgesamt haben.

Da in den Benutzbarkeitstests teilweise unklar war, was die Zahlen Gesamtpunktzahl oder der Text Spielername bedeuten, habe ich das explizit dazu geschrieben.

Die Auswahl, ob man kooperiert oder betrügt, habe ich komplett überarbeitet. Grund dafür war, dass in den Tests überhaupt nicht verstanden wurde, wer die viele Punkte erhält. Ich habe mich für Darstellung Tabellenformen entschieden, ähnlich wie auf der Einstellungenseite. Die Tabellenform hat auch den Vorteil, dass sie mehr als 2 Optionen unterstützt, indem man eine neue Spalte hinzufügt.

Die Texte und Buttons in der Tabellenform habe ich so konzipiert, dass sie sich immer an die aktuelle Situation anpassen. Z.b. sollte man am Anfang einer Runde die Entscheidung zwischen Kooperieren- und Betrügen-Button haben. Sobald man sich dann für eine der beiden Option entschieden hatte, sollte der Button dann z.b. durch einen Text wie in Du kooperierst oder Du betrügst ersetzt werden.

Abbildung 18: Spieler-Spielseite Gefangenendilemma - Spieler kooperiert

Abbildung 19: Spieler-Spielseite Gefangenendilemma - Rundenende

Die Tabellen vom hatte auch den Vorteil, dass diese für Öffentliche Güter ebenfalls funktioniert. Dafür habe ich ebenfalls ein Design erstellt. Das wäre mit der bisherigen Ansicht nicht gegangen, weil diese sich immer auf einen Mitspieler bezieht, den es bei öffentlichen Gütern nicht gibt.

Die Admin-Ergebnisseite habe ich dahingehend geändert, dass die Spielergebnisse anders dargestellt werden. Ursprünglich wurde für jeden Spieler ein Balken dargestellt. Dieser Balken hat sich aus einzelnen Bereichen für jede Runde zusammengesetzt. Die Länge eines jeden Bereichs entsprach der Punkte, die man in dieser Runde bekommen hat. Das war unübersichtlich, weil man schwierig ablesen konnte, wie viele Punkte man in welcher Runde bekommen hat. Außerdem wurde Diagramm bei vielen Spielern unlesbar.

Die Hauptinformation des Diagramms war die Gesamtpunktezahlen der Spieler. Daher wollte ich diese Information beibehalten. Dafür habe ich mich wieder für eine Tabellenform entschieden, weil diese recht schmal sein kann und dadurch gut mobil dargestellt werden kann. In den Tabellenzeilen sollten jeweils ein Datensatz pro Spieler dargestellt werden. Spalten waren dann für verschiedene Daten der Spieler wie die Gesamtpuktahl gedacht.

Abbildung 20: Admin-Ergebnisseite

Außerdem habe ich die Spalte ergänzt, wie oft eine Spieler prozentual gesehen kooperiert. Damit wollte ich erreichen, dass man aus der Kombination von erreichter Punktzahl und wie oft jemand kooperiert hat ableiten kann oder diskutieren kann, ob es eher Sinn macht zu kooperieren oder zu betrügen. Vorteil einer Tabelle ist auch, dass man die einzelnen Spalten selbst sortieren kann. Standardmäßig sollten die Punktespalte in absteigender Reihenfolge sortiert werden, damit man ganz oben sieht, wer am meisten Punkte bekommen hat.

Die Spieler-Ergebnisseite habe ich ähnlich aufgebaut, wie die Admin-Ergebnisseite. Idee davon war, dass die Spieler sich dann im Detail mit den Daten auf Ihrem eigenen Gerät auseinandersetzen können. Der Spieler selbst sollten dann in den Daten hervorgehoben werden.

Im Design habe ich dann auch Öffentliche Güter ergänzt. Dafür musste ich zwei Seiten anpassen. Zum einen habe ich auf der Spieler-Warteraumseite ein Erklärungs-

Abbildung 21: Spieler-Ergebnisseite

text für öffentliche Güter eingebaut. Zum andern habe ich auf der Spieler-Spielseite die Tabelle auf öffentliche Güter zugeschnitten.

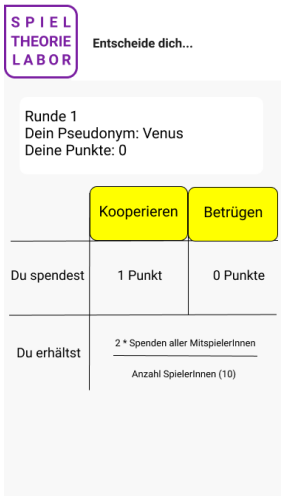


Abbildung 22: Spieler-Spielseite Öffentliche Güter

Die Einstellungenseite hätte auch noch angepasst werden müssen. Allerdings waren mir zu diesem Zeitpunkt die genauen Einstellungen für öffentliche Güter noch unklar.

Alle anderen Seiten konnten für Öffentliche Güter übernommen werden. Das lag unter anderem daran, weil ich mich entschieden habe, in beiden Varianten von kooperieren und betrügen zu sprechen.

Als letzten Schritt habe ich in Figma die Buttons so eingestellt, dass man zu anderen Seiten weitergeleitet wird. Dadurch war es möglich in der Prototyp Ansicht von Figma durch die gesamten Seiten zu navigieren, als wäre es eine richtige Software.

Dafür musste ich zusätzliche Buttons hinzuzufügen. Diese sollte später nicht übernommen werden und wurden dadurch markiert, dass die Texte auf den Buttons in

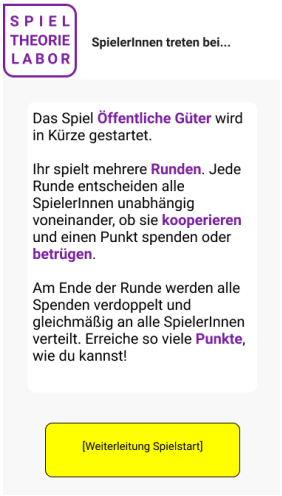


Abbildung 23: Spieler-Warteraumseite Öffentliche Güter

eckigen Klammern standen. Diese Buttons waren dafür da, dass man im Design von einer Ansicht zur nächsten navigieren kann, wenn das in der Anwendung automatisch würde.

Zu dem Design habe ich dann wieder Feedback eingeholt. Da mein Schwerpunkt im Design auf Verbesserung der Verständlichkeit lag, wollte ich diese mit der bisherigen Anwendungen vergleichen. Da die bisherige Anwendung aber kein aktuelles Design hatte, mit dem man es vergleichen konnte, habe ich mich entschieden, das Design mit Leuten zu testen, die die ursprünglich Anwendungen kannten und daher aus dem Gedächtnis heraus vergleichen konnten. Ich habe also das Design mit Stefan Nessler und Jamie Schenker getestet.

Bei dem Test habe ich sie jeweils alleine den Figma Prototyp durchklicken lassen und habe dabei alle meine Änderungen erklärt und Feedback von den beiden aufgenommen. Dabei kam noch einige Sachen raus die ich nicht beachtet hatte. Z.b. wurde man in der Spieler-Warteraumseite direkt an die Spieler-Spielseite weitergeleitet, sobald der Admin ein Spiel gestartet hat. Das konnte bedeuten, dass man den Einführungstext nicht zu Ende lesen konnte. Daher habe ich dann in der Implementierung dort einen Button hinzugefügt, mit dem man einem Spiel beitreten kann, sobald das vom Admin gestartet wurde. Das restliche Feedback zum Design befindet sich im Anhang (A.7 und A.8).

Ursprünglich hatte ich geplant, das Design der Arbeitsgruppe Human-Centered Computing der Freien Universität Berlin vorzustellen, weil diese sich mit unter anderem mit Design beschäftigen. Da mir der Designprototyp ohne Vorwissen der Anwendung allerdings zu abstrakt vorkam, habe ich mich entschieden, das Feedback auf nach der ersten Implementierungsphase zu verschieben.

Rückblickend gesehen hätte, wäre das wahrscheinlich sinnvoll gewesen. Denn in späteren Tests hat sich herausgestellt, dass die visuellen Änderungen, die ich gemacht habe, mehr Einfluss hatten, als ich gedacht hatte. Mir war zwar schon bewusst gewesen, dass die Tabellendarstellung der Spieler-Spielansicht nicht so attraktiv ankommen wird wie die vorherige Darstellung. Ich habe das aber nicht so relevant eingeschätzt wie die Verständlichkeit. Tests mit Designexperten hätten das vielleicht

schon an dieser Stelle abfangen können.

7 Implementierung

Für die Implementierung habe ich wie im Softwareprojekt auch die Versionskontrolle git genutzt. Dafür habe ich sowohl im Frontend als auch im Backend einen eigenen Branch angelegt, auf denen ich dann die gesamte Bachelorarbeit gemacht programmiert habe.

Dann bin ich die einzelnen Seiten durchgegangen und habe deren Komponenten auf das neue Design angepasst. Da ich auf jeder Seite dafür Änderungen durchführen musste, habe ich das als Möglichkeit gesehen parallel wie Refactorings zu machen, um die Codequalität zu steigern. Dafür habe ich die Komponenten refactored, um sie möglichst konsistent miteinander zu machen. Z.B. habe ich verbliebene HTML-Tags aus dem Softwareprojekt durch Material UI Komponenten ersetzt.

Dabei ist mir aufgefallen, das manche mit Material UI Komponenten öfters mit den gleichen Optionen, wie z.B. die Farbe der Komponente, vorkommen. Daher habe ich extra Komponenten gebaut, welche von den Material UI Komponenten sozusagen erben und die entsprechenden Optionen immer gesetzt haben. Diese können dann überall eingebunden werden und man muss nicht immer wieder diese Optionen setzen. Ein weiterer Vorteil von diesem Ansatz ist, dass sollte man dann doch mal diese Komponenten ändern wollen, kann man das an einer zentralen Stelle tun. Die Änderung wird dann auf allen Seiten übernommen, wo diese Komponente eingebunden wird.

Ein Beispiel für solch eine Komponente ist das Grid aus Material UI. Das ist dafür da, den Komponenten auf dem Bildschirm zu verteilen. Das habe ich immer so eingestellt, dass der Inhalt der Seite zentriert wird. Sollte man das später ändern wollen, muss man das als nur an einer zentralen Stelle ändern.

Diese Komponenten liegen im Ordner ThemedMaterialComponents. Rückblickend wäre es sinnvoll gewesen mit den Buttons ebenfalls so vorzugehen. Das habe ich aber erst gemerkt, nachdem ich die Buttons schon implementiert hatte.

Außerdem habe ich vereinheitlicht, wie die Komponenten gestylt werden. Im Softwareprojekt hatten wir dafür drei Varianten eingesetzt: CSS/SCSS, Styled components oder die Hook API (https://material-ui.com/styles/basics/#hook-api).

Die letzten beiden Varianten haben gegenüber CSS den Vorteil, dass man in diesen auch Javascript Code verwenden kann. Dadurch kann man Variablen importieren, Berechnungen durchführen und mehr. Zwischen den Varianten Styled components und Hook API konnte ich allerdings keine nennenswerten Unterschiede erkennen. Ich habe mich für die Hook Variante entschieden. In der Anwendung wird jetzt also eine einheitliche Variante verwendet, wie man Komponenten stylt.

In einigen Teilen, wo bisher gestylt wurde war das gar nicht nötig. Dort hat in vielen Fällen gereicht, die richtigen Attribute der Material UI Komponenten herauszusuchen und anzuwenden. Allgemein habe ich aufgeräumt und veraltete, überflüssige Dateien gelöscht.

Öffentliche Güter konnte man recht früh spielen. Dafür habe ich im Modell des Front- und Backends einen neuen Spieltyp hinzugefügt. Im Backend gab es bereits

eine leere Methode für Öffentliche Güter. Diese habe ich mit Logik gefüllt. Dabei habe ich mich am Gefangenendilemma orientiert. Tatsächlich ist Öffentliche Güter leichter zu implementieren als das Gefangenendilemma, weil man bei Öffentliche Güter keine Paaren von Spielern bilden muss. Dieser Schritt fällt also bei der Berechnung, wer wie viele Punkte erhält, weg. Gefehlt haben zu diesem Zeitpunkt die Einstellungen für Öffentliche Güter. Außerdem hatte ich die Spieler-Spielseite noch nicht an Öffentliche Güter angepasst. Man konnte also nur über die Gefangendilemmaansicht spielen.

Als nächstes habe ich das Modul redux (https://redux.js.org/) hinzugefügt. Mit redux kann an einer zentralen Stelle in der Anwendung den kompletten Status der Anwendung definieren. Komponenten können sich zu einem Teil des Status verknüpfen. Immer wenn dieser Status aktualisiert wird, passiert das dann auch automatisch in der Komponente. Dadurch können alle Komponenten auf einem Stand gehalten werden.

In Redux kann man auch Methoden definieren, die diesen Status beeinflussen können. Das sind sozusagen Setter Methoden. Der Status kann ausschließlich über Setter beeinflusst werden. Das heißt die Kontrolle über den Status wird über diese Methoden geregelt und es daher leichter zu überwachen und zu testen.

Ich habe redux eingeführt, weil bis dahin jede Komponente sich selbst um den Status gekümmert hat. Für Status, der nur diese Komponente betrifft ist das auch sinnvoll. Allerdings nutzen die meisten Komponenten Daten aus dem Backend. In diesen Komponenten muss dann über die API eine Verbindung zum Backend aufgebaut werden und nach den Daten gefragt werden. Da sich die Komponenten nicht untereinander absprechen, liegen die gleichen Daten bei verschiedenen Komponenten. Z.B. müssen verschiedene Komponenten auf die Punktzahl eines Spielers aus dem Backend zugreifen.

Mit redux kann eine zentrale mit dem Backend kommunizieren und den gesamten Status der Anwendung kommunizieren. Komponenten, die Zugriff auf diesen Status haben wollen, können sich dann mit der zentralen Stelle verbinden.

Langfristig gesehen hat das also den Vorteil, das Komponenten entschlackt werden. Denn sie brauchen keine Logik, um auf dem Status zurückzugreifen oder ihn zu ändern. Sie müssen lediglich den Status oder die Setter Methoden importieren.

Außerdem sind dann alle Methoden, die den Status beeinflussen können an einer zentralen Stelle. Dadurch muss man nicht jede Komponente testen, sondern kann die Setter selbst testen.

Die einzigen automatisierten Tests, die es bis daher in Frontend gab, waren Rendertests. Also, ob eine Komponente dargestellt werden kann oder nicht. Automatisierte Tests, welche die Logik der einzelnen Komponenten überprüfen, wären zu aufwändig gewesen.

Mit redux kann man gut die Systemlogik testen, indem man die Setter testet. Dadurch ist also auch das Testen unkomplizierter.

Zu diesem Zeitpunkt war ich mir trotzdem nicht sicher, ob sich die Einführung von redux auch kurzfristig lohnt. Denn für die Umstellung auf redux musste ich auch wieder Refactoringaufwand betreiben.

Der Refactoringaufwand ist allerdings geringer, als man annehmen könnte. Denn für den Status konnte ich das Modell nutzen. Denn im Status soll ja auch hinterlegt werden, wer der Spieler ist, welches Spiel gerade gespielt wird und in welcher Runde

man gerade ist.

Redux kann mit der Erweiterung redux persist (https://github.com/rt2zz/redux-persist) auch den Hilfsbereich ersetzen, der lokal Daten bei Spieler als Cookies hinterlegt. Denn redux-persist kann den ganzen Status der Anwendung automatisch beim Nutzer hinterlegen und beim nächsten Start des Browsers kann der Status dann geladen werden.

Das hat den Vorteil, dass der Code des Hilfsbereichs nicht gepflegt werden muss. Dieser war bisher auch nicht getestet.

Langfristig gesehen bietet redux also viele Vorteile. Diese Vorteile treffen aber erst ein, wenn die gesamte Anwendung auf redux umgestellt wurde. Das habe ich in der Bachelorarbeit nicht geschafft. Kurzfristig hat es also den Nachteil, dass aktuell 2 Verfahren parallel in der Anwendung bestehen, wie man den Status verwaltet.

Mit redux habe ich angefangen die Kopfleiste anzupassen. Dort wird immer der aktuelle Status der Anwendung angezeigt. Die einzelnen Seite können dann über einen Setter diese den Text der Kopfleiste ändern. Z.B. kann in der Spieler-Spielansicht ein Text angezeigt werden wie Bitte entscheide dich.

Bei manchen Texten wollte ich darstellen, dass gerade noch auf etwas gewartet wird. Dafür habe ich mich entschieden, drei Punkte (...) hinter dem Text anzuzeigen. Diese drei Punkte habe ich animiert und mich dafür bei https://martinwolf.org/before-2018/blog/2015/01/pure-css-savingloading-dots-animation/ inspirieren lassen.

Als nächstes habe ich die Spieleinstellungen um die Optionen aus Öffentliche Güter ergänzt. Diese Einstellungen sind: Wie viel spenden Spieler, wenn sie kooperieren und um welchen Faktor werden diese Spenden vervielfacht. Diese Einstellungen habe ich dann auch in redux umgesetzt. Als Status konnte ich dafür direkt die Einstellungen aus dem Modell übernehmen. Außerdem habe ich Setter definiert. Diese werden in der Einstellungenseite aufgerufen, nachdem dort die Einstellungen geändert wurden. Tritt man hingegen einem Spiel bei, werden aus dem Backend die Einstellungen gelesen und in redux übernommen.

Im Design gab es eine ähnliche Tabelle für die Punktematrix auf der Einstellungen-Seite und der Spieler-Spielseite. Diese beiden Tabellen gab es jeweils in zwei Varianten: einmal für das Gefangenendilemma und Öffentliche Güter.

Für diese vier Tabellenvarianten wollte ich eine übergreifende Komponente erstellen, von der dann die vier Varianten jeweils erben. In der react Dokumentation (https://reactjs.org/docs/composition-vs-inheritance.html#specialization) zu Vererbung react eher empfiehlt, statt Vererbung eine generelle Komponente zu bauen. Die Kind Komponenten sollen dann einen speziellen Fall davon implementieren.

Daran habe ich mich orientiert. Ich habe also eine generelle Tabellenkomponente gebaut und vier speziellere dazu. Diese Vorgehensweise hat den Vorteil, dass man alle prinzipiellen Sachen in der allgemeinen Komponente definieren kann und alle Elemente, welche die speziellen Komponenten von den anderen unterscheidet, kann man dann in den speziellen Komponenten definieren.

Dann habe ich bei bestimmten Buttons Bestätigungsdialoge hinzugefügt. Und zwar bei kritischen Aktionen wie Spiel beenden. In diesen Dialogen wird der Admin nochmal gefragt, ob man wirklich das Spiel beenden möchte. Dadurch wird verhindert, dass das aus Versehen passiert.

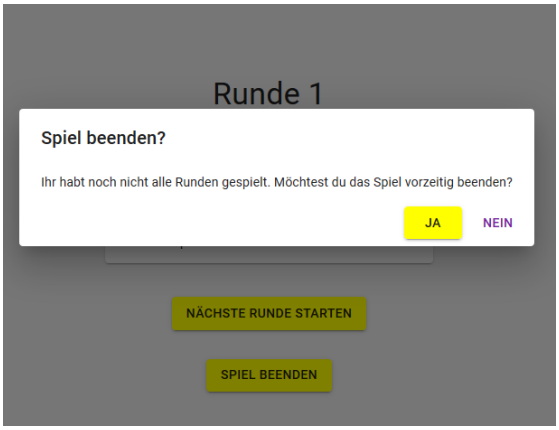


Abbildung 24: Bestätigungsdialog Spiel beenden

Außerdem habe ich Buttons so lange deaktiviert, wie sie noch keinen Sinn machen Punkt. Z.b. kann man erst ein Spiel starten, sobald mindestens ein Spieler beigetreten ist. Vorher ist der Button ausgegraut.

Abbildung 25: Deaktivierter Button

Dann habe ich die Anwendung noch für Benutzbarkeitstests vereinfacht: Z.b. wurde der Beitritt Link anklickbar. Außerdem werden Leerzeichen in Spielcode entfernt, die sonst beim kopieren der Spielcodes hinzugefügt werden.

Außerdem habe ich den sechsstelligen Spielcode geändert. Der bestand aus Buchstaben und Zahlen. Ich habe ihn dahingehend geändert, dass er nur aus vier Zahlen besteht. Das macht es leichter, sich den Spielcode zu merken. Außerdem ist er eindeutiger, weil Nullen und O's nicht mehr verwechselt werden können.

Zu diesem Zeitpunkt habe ich mit der Evaluation begonnen. Feedback auf den Evaluationen habe ich dann direkt umgesetzt. Zwischen verschiedenen Evaluationen bin ich also iterativ vorgegangen. Aus der geplanten zweiten Iteration wurde also mehrere kleinere Iterationen. Dabei habe ich mich auf Änderung beschränkt, die nicht aufwändig waren, z.b. Anpassung von Texten. Größere Änderungen habe ich nur gemacht, wenn ich absehen konnte, dass sie die Anwendung langfristig verbessern.

Ich habe zum Beispiel viel Feedback zur Optik der Anwendung bekommen. Ich kann mir aber vorstellen, dass das Design in Zukunft grundlegend überarbeitet wird. Daher habe ich keine Änderungen der Optik vorgenommen.

Stattdessen habe ich die Tabelle auf den Ergebnisseiten ergänzt.

Während eines Benutzbarkeitstests ist aufgefallen, dass das Iterative Gefangenendilemma nicht funktioniert. Das war mit und auch damals dem Front- und Backendteam nicht aufgefallen. Wahrscheinlich hatten wir immer nur das Einfache Gefangenendilemma getestet, weil die beiden Varianten sehr ähnlich sind.

Dann habe ich das Frontend und das Backend so geändert, dass die Resultate einer Runde den Spielern angezeigt werden. Das heißt, es wird jetzt dargestellt, wie sich der Mitspieler im Gefangenendilemma entschieden hat und wie viele Punkte man erhält.

Außerdem habe ich hinzugefügt, dass man statt um Punkte um Smarties spielen kann. Die Idee hatte ich von Stefan Nessler. Ich wollte wissen, ob es Einfluss auf die Motivation hat, wenn man um Smarties statt um abstrakte Punkte spielt.

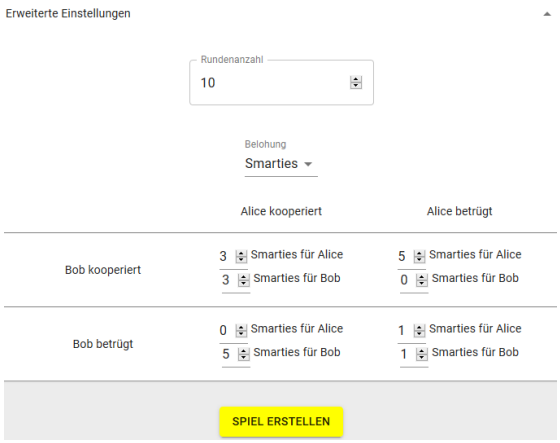


Abbildung 26: Bessere Belohnung als Punkte?

Neben den Benutzbarkeitstest habe ich die Anwendung hauptsächlich selbst per Hand getestet. Ich habe keine automatisierten Tests geschrieben, weil ich hauptsächlich Änderungen im Frontend gemacht habe. Diese Änderungen habe ich dann im Browser getestet.

Mit redux kann man zwar theoretisch den Status der Anwendung testen. Allerdings konnte ich den Status in redux noch nicht auf die Spiellogik an sich ausweiten. Bisher sind in redux vor allem die Einstellungen für das Spiel. Diese enthalten wenig Logik und brauchen meiner Meinung nach daher keine automatisierten Tests.

Alle bisherigen automatisierten Tests habe ich trotzdem aktualisiert. In git werden diese automatisch überprüft. Mein aktueller Stand der Anwendung besteht alle Tests. Der Branch, auf dem ich gearbeitet habe, kann daher gemergt werden. Mit Jamie Schenker habe ich vereinbart, dass er das übernimmt, weil er als nächstes damit weiterarbeiten wird.

Von der geplanten Implementierung fehlt neben der vollständigen Importierung vom Status des Frontends in redux noch, dass Bots am Anfang einer Runde abstimmen. Aktuell stimmen sie nämlich erst am Ende einer Runde ab. Bis dahin werden

sie angezeigt, als hätten sie sich noch nicht entschieden. Das hat Tester verwirrt.

Ich habe versucht meine Funktionen im Code und meine Commits aussagekräftig zu benennen. So habe ich bei Commits, die sich auf eine Komponente beziehen immer, diese Komponente immer am Anfang des Commitnames gestellt, damit man diese leichter wiederfinden kann. Außerdem habe ich Commits nie gleich gepusht. Dadurch konnte ich Änderungen, die mir später aufgefallen sind noch in den Commit integrieren, ohne einen nachträglichen Commit machen zu müssen. Dadurch wollte ich die Commithistorie verständlich und übersichtlich halten. Bevor ich dann ein Commit gepusht habe, bin ich die Änderungen durchgegangen und habe Durchsichten gemacht.

Die Implementierung des Frontend (B.2) und des Backends (B.3) habe ich mit eingereicht.

8 Evaluation

Für die Evaluation habe ich mit verschiedenen Personen und Gruppen zusammengearbeitet. Diese hatten verschiedene Vorkenntnisse und kamen aus unterschiedlichen Bereichen. Manche kannten das Projekt schon, andere nicht. Daher habe ich unterschiedliche Testformen verwendet, um diese besser auf die Tester zuschneiden zu können.

Die Tests sind also untereinander nicht vergleichbar. Das liegt auch daran, dass ich zwischen den einzelnen Testgruppen wieder Änderungen in der Implementierung durchgeführt habe.

8.1 Arbeitsgruppe Software Engineering

Das erste Feedback habe ich von der Arbeitsgruppe Software Engineering der Freien Universität Berlin bekommen. Dort habe ich einen Vortrag gehalten, indem ich das Softwareprojekt vorgestellt habe. Im Anschluss habe ich eine Live-Demo der Anwendung mit der Mitgliedern der Arbeitsgruppe gespielt. Ich war dafür der Admin, die Mitglieder haben die Rolle der Spieler übernommen. Die Live-Demo ist allerdings nach ersten Spielrunde im Gefangenendilemma abgestürzt. Ich vermute, das lag daran, dass mein Rechner mit gleichzeitig Spiel hosten und Bildschirm übertragen überfordert war.

Feedback, was ich bekommen habe (A.9), war vor allem zum schriftlichen Teil meiner Arbeit. Zur Software selbst gab es eher wenig Feedback. Ich glaube das lag daran, dass zum einen die Live-Demo abgestürzt ist. Außerdem hatte ich mir im Vornherein keine direkten Fragen überlegt, weil damit gerechnet hatte, dass trotzdem genug Feedback zur Anwendung kommt.

8.2 Arbeitsgruppe Human-Centered Computing

Daraus habe ich gelernt. Mein nächsten Vortrag war bei der Arbeitsgruppe Human-Centered Computing. Ursprünglich hatte ich geplant, diesen Vortrag genau so zu halten wie in der Arbeitsgruppe zuvor. Aber aufgrund des Feedbacks habe ich diesen umstrukturiert. Ich keine Live-Demo mehr gemacht, sondern Screenshots gezeigt,

damit man auf jeden Fall die ganze Anwendung besprechen kann. Außerdem habe ich immer zu jedem zu jeder Seite zwei nebeneinander Screenshots gezeigt. Der ein war immer der Stand der Anwendung vor und der andere nach meiner Bachelorarbeit. Dadurch konnte ich besser durchsprechen, was ich für Änderung gemacht habe. Diese konnten dann auch besser diskutiert werden.

Daher habe ich wesentlich mehr Feedback (A.10) bekommen. Es wurde einiges bemängelt, z.B. dass ich zu wenig Tests mit Schülern gemacht habe. Dem stimme ich zu. Das hätte ich auch gerne gemacht, allerdings wäre es durch Corona aufwändig gewesen, diese zu erreichen, da auch Stefan Nessler aktuell nicht mit Schülern arbeitet.

Dann wurden Designprobleme wie z.B. ungünstige Abstände bemängelt. Alexa Schlegel hat mir daraufhin angeboten nochmal detaillierteres Feedback zum Design zu geben.

Ein weiterer Kritikpunkt war, dass die neue Spieler-Spieleite in der Tabellenform nicht so attraktiv ankam wie ursprünglich Ansicht.

Außerdem wurde empfohlen über Gamification nachzudenken. Denn in Schulen werden bereits Anwendungen mit Gamification wie Kahoot eingesetzt.

8.3 Stefan Nessler

Als nächstes hat Stefan Nessler die Anwendung bewertet. Dafür habe ich ihm die Anwendung gegeben. Er hat als Admin und Spieler die Anwendung getestet. Ich habe dabei zugeschaut und Fragen gestellt.

Er hat mir noch sehr viele Ideen gegeben, die man in Zukunft noch machen kann (A.11). Außerdem findet er ebenfalls, dass die ursprüngliche Anwendung attraktiver aussah. Gleichzeitig hat er angemerkt, dass die Anwendung seiner Meinung nach intuitiver und verständlicher funktioniert; und dass ihm das wichtiger ist, weil der Hauptzweck der Anwendung ist zu vermitteln, wie Spielteoretische Konzepte funktionieren. Dafür hat Verständlichkeit bei ihm eine hohe Priorität.

8.4 Alexa Schlegel

Von Alexa Schlegel habe ich dann das versprochene Feedback (B.4) zum Design bekommen. Dazu hatte ich ihr Screenshots der alten und neuen Variante geschickt. Sie hat diese verglichen und einige Design Mängel bemerkt. Sie hat auch angemerkt, dass ihr aber für eine detailliertere Designanalyse Kontextwissen gefehlt hat.

8.5 Benutzbarkeitstests

Als nächstes habe Benutzbarkeitstests mit zwei Bekannten gemacht. Beide sind Mitte 30 Jahre alt, haben schon einen Master abgeschlossen und kennen das Gefangenendilemma. Da sie die Anwendung bisher nicht kannten, habe ich sie sowohl einmal die ursprüngliche Anwendung als auch meine Version testen lassen.

Den einen Tester habe ich erst die ursprüngliche und dann die neue Version der Anwendung testen lassen. Beim anderen Tester habe ich das anders herum gemacht. Dabei habe ich bewusst offen gelassen, welche Version die neuere Version ist. Damit wollte ich Vorurteile umgehen.

Es hat sich ein ähnliches Bild abgezeichnet wie beim Feedback bisher (A.12.1 und (A.12.2). Es wurden wieder einige Designmängel festgestellt und die alte Version kam attraktiver herüber. Beim Spielen wurde in der alten Version allerdings einiges nicht oder missverstanden. Allerdings lief die neue Version flüssiger.

Im Nachhinein habe ich beide gefragt, welche Version sie denken, dass es die neue ist. Beide vermuteten, dass jeweils die zweite Variante die neuere ist. Der Tester, der die neue Version richtig erkannt hatte, hat das durch das Logo geschlussfolgert. Der Tester, der gedacht hat, die alte Version sei die neue, hat das damit begründet, dass die Hauptseite durch die großen Buttons eindrucksvoller aussah.

Die letzte Feedbackrunde wollte ich mit den drei Testern aus dem Benutzbarkeitstest am Anfang der Arbeit machen. Einer der drei war allerdings nicht verfügbar.

Sie hatten also die ursprüngliche Variante bereits getestet. Die Version, die sie getestet haben enthielt ist die aktuellste. Sie enthielt gegenüber der anderen Tests, dass man um Smarties anstatt Punkte spielen kann. Sie haben dann entsprechend ihrer Performance von mir Smarties bekommen. Das kam auch recht gut an, wobei sich gewünscht wurde, dass man statt Smarties auch eine andere Belohnung einstellen kann. Ob die Smarties aber geholfen haben, kann ich nicht beurteilen.

In ihrer Version wurde auch das erste mal richtig das Ergebnis einer Runde dargestellt. Ich denke, dass das das Verständnis gefördert hat.

Beide hatten bei den ersten Tests die Konzepte überhaupt nicht verstanden. In der neuen Variante dagegen war für beide klar, was die Entscheidungen bedeuten. Außerdem konnten sie Strategien entwickelt wie man z.B. am meisten Smarties bekommen kann.

Das war leicht eingeschränkt bei Öffentliche Güter, weil da eine Formel steht, wie viele Punkte man bekommt. Diese Formel war umständlich zu verstehen. Das sollte in Zukunft noch vereinfacht werden.

9 Ausblick

Als nächste Schritte gibt es einige Optionen:

Auf jeden Fall sollte die Anwendung in Klassenräumen getestet werden. Meiner Einschätzung nach hat die Anwendung jetzt einen Grad an Verständlichkeit, sodass das möglich ist.

Dann kann genauer untersucht werden, ob und wie man Gamification einbauen möchte. Vom bisherigen Feedback ist das für Studenten auf jeden Fall interessant, für Schüler wahrscheinlich auch. Stefan Nessler steht Gamification kritisch gegenüber. Er würde das aber ausprobieren, solange dadurch nicht die fachlichen Aussagen der Anwendung einschränkt.

Entscheidet man sich für Gamification, muss wahrscheinlich ein neues Design gebaut werden. Falls man sich gegen Gamification entscheidet, sollten Designmängel behoben. Ein guter Anlaufpunkt dafür ist das Design Feedback von Alexa Schlegel.

Außerdem kann man den Rest der Spiellogik auf redux umstellen. Wenn man das erreicht hat, kann man beginnen das Frontend mit automatisierten Test zu versorgen.

10 Fazit

Meiner Meinung nach hat die Anwendung trotz des negativen Feedbacks zu Optik mehrere Schritte nach vorne gemacht: der Code ist konsistenter geworden. Die Anwendung ist verständlicher geworden. Und das wichtigste zum Schluss: Man kann Öffentliche Güter spielen.

Literaturverzeichnis

[1] Robert Axelrod and William Donald Hamilton. The evolution of cooperation. *science*, 211(4489):1390–1396, 1981.

[2] Manfred Milinski. Reputation zahlt sich aus und löst kooperationsprobleme. In *Jahrbuch 2004. Max-Planck-Gesellschaft.*, pages 383–387. Max-Planck-Gesellschaft, 2004.

[3] Andreas Ortmann and David Colander. A simple principal-agent experiment for the classroom. *Economic Inquiry*, 35(2):443–450, 1997.

[4] Daniel J Rankin, Katja Bargum, and Hanna Kokko. The tragedy of the commons in evolutionary biology. *Trends in ecology & evolution*, 22(12):643–651, 2007.

Wochen	Datum	Phase	Details
W1-W2	22.7. - 31.7.	Setup & Analyse aktuelle App	Planung Heuristische Analyse Usability Tests, Recherche Public Goods Git Setup
W3-W4	3.8. - 14.8.	Design	Anforderungen definieren Anpassung bisherige Views Ausarbeitung Public Goods View Feedback Stefan Feedback HCI Gruppe
W5-W6	17.8. - 28.8.	Implementierung	Frontend neue + alte Views Backend neue + alte Views Usability Tests Feedback Stefan
W7-W8	31.8. - 11.9.	2. Iteration	Änderungen umsetzen Finaler Test mit Stefan
W9-W11	14.9. - 2.10.	Text schreiben	Inhalt der Arbeit Dokmentation updaten Drucken
W12-W13	5.10. - 13.10.	Pufferzeit	
	14.10.	Abgabe	

Tabelle 1: Planung

A Anhang

A.1 Planung

A.2 Anforderungen Öffentliche Güter

A.2.1 Anforderungen Basisvariante

- Anforderung Öffentliches Gut: Es existiert eine Ressource, von der alle Spieler gleichgestellt profitieren.
- Anforderung Kooperation: Spieler können einen individuellen Beitrag zu leisten, der allen zu Gute kommt.
- Anforderung Betrug: Spieler können vom öffentlichen Gut zu profitieren, ohne darin zu investieren.

A.2.2 Anforderungen erweiterte Varianten

- Anforderung Reputation: Spieler kennen das Verhältnis von Kooperation zu Betrug der anderen Spieler.
- Anforderung Kontrolle: Spieler haben Einfluss darauf, wie stark andere Spieler vom öffentlichen Gut profitieren.

- Anforderung Ausstieg: Spieler können vom Öffentlichen Gut aussteigen. Anstatt darin einzuzahlen und davon zu profitieren, erhalten sie eine feste Anzahl an Punkten.

A.3 Ergebnisse heuristische Analyse

Die Zahlen in Klammern stehen für die Nummer des Kriteriums der Nielsen Heuristik.

A.3.1 Startseite

- (1) Unklar, wofür die Anwendung eigentlich da ist.
- (4) Mischung aus deutsch und englisch.

A.3.2 Einstellungenseite

- (1) Unklar, was Statistiken live anzeigen bedeutet.
- (1) Unklar, was ein Iteratives Gefangenendilemma bzw. ein Einfaches Gefangenendilemma ist.
- (4) Mischung aus deutsch und englisch.
- (4) Punktwerte anders dargestellt als in der Spieleransicht.
- (4) Dropdown für Spielmodus Auswahl heißt Spieleinstellung, obwohl das auch der Titel der gesamten Seite ist.

A.3.3 Admin-Warteraumseite

- (1) Unklar, wer mit dem angegebenen Link beitreten soll.
- (1) Unklar, was die Spiel ID bedeutet.
- (1) Unklar, das Cream Dia einE SpielerIn ist und was der Kreis dahinter darstellen soll.
- (1) Unklar, das auf Spielerbeitritt gewartet wird.
- (4) Mischung aus deutsch und englisch (auch bei Spielernamen).

A.3.4 Admin-Spielseite

- (1) Unklar, was nach Warten passiert.
- (1) Unklar, was nach Countdown starten passiert.
- (1) Unklar, was Kreise neben Spielernamen bedeuten.
- (1) Maximal Rundenzahl wird nicht angezeigt, auch wenn bekannt.
- (4) Mischung aus deutsch und englisch.

- (5) Aus Versehen Spiel vorzeitig beenden wird nicht abgefangen.
- (5) Aus Versehen Nächste Runde starten wird nicht abgefangen.

A.3.5 Admin-Ergebnisseite

- (4) Mischung aus deutsch und englisch.
- (8) Erhaltene Punkte Diagram wird bei mehr Spielern unübersichtlich.
- (10) Im Kooperations/Betrugs Verhältnis Diagram ist unklar, welcher Bereich was bedeutet.

A.3.6 Beitretenseite

- (4) Unklar, das Spiel starten nicht tatsächlich das Spiel startet.
- (10) Unklar, woher der Spielcode kommt.

A.3.7 Spieler-Warteraumseite

- (4) 1 Unklar, was passiert, wenn man auf Spieler wartet oder was man stattdessen machen kann.
- (1) Unklar, das Cream Dia der eigene Spielername ist.
- (2) Punktevergabematrix und Erklärung des Spiels über Verbrecher nutzen domänen spezifische Sprache.
- (4) Matrix wird anders dargestellt als in den Settings und ingame.

A.3.8 Spieler-Spielseite

- (1) Unklar, wann eine Runde vorbei ist und wann eine startet.
- (1) Unklar, was das Resultat einer Runde ist.
- (1) Punkteverteilung ist unklar.
- (1) Leeres Punkteverlaufsdiagramm ist unverständlich.
- (1) Unklar, welches die aktuelle Runde ist.
- (1) Unklar, wie viele Runden gespielt werden.
- (4) Herzen von Kooperieren und Betrügen werden sonst in der App nicht genutzt.
- (4) Sowohl Punkteverlauf als auch untere Ansicht zeigen Punktestand.
- (8) Punkteverlauf zeigt nicht besonders relevante Informationen.
- (10) Unklar, dass Zahl unten links Punkte bedeuten soll.

A.3.9 Spieler-Spielseite

- (1) Unklar, warum der Name im Das Spiel ist vorbei Text steht.
- (1) Unklar, wie man im Verhältnis zu anderen abgeschlossen hat.
- (8) Erhaltene Punkte Diagramm wird bei mehr Runden unübersichtlich.
- (10) Es gibt keine nächste mögliche Aktion.

A.3.10 Mobil

- (8) Seite ist unbrauchbar in vertikaler Ausrichtung.

A.4 Skript Benutzbarkeitstest

A.5 Einverständniserklärung Benutzbarkeitstest

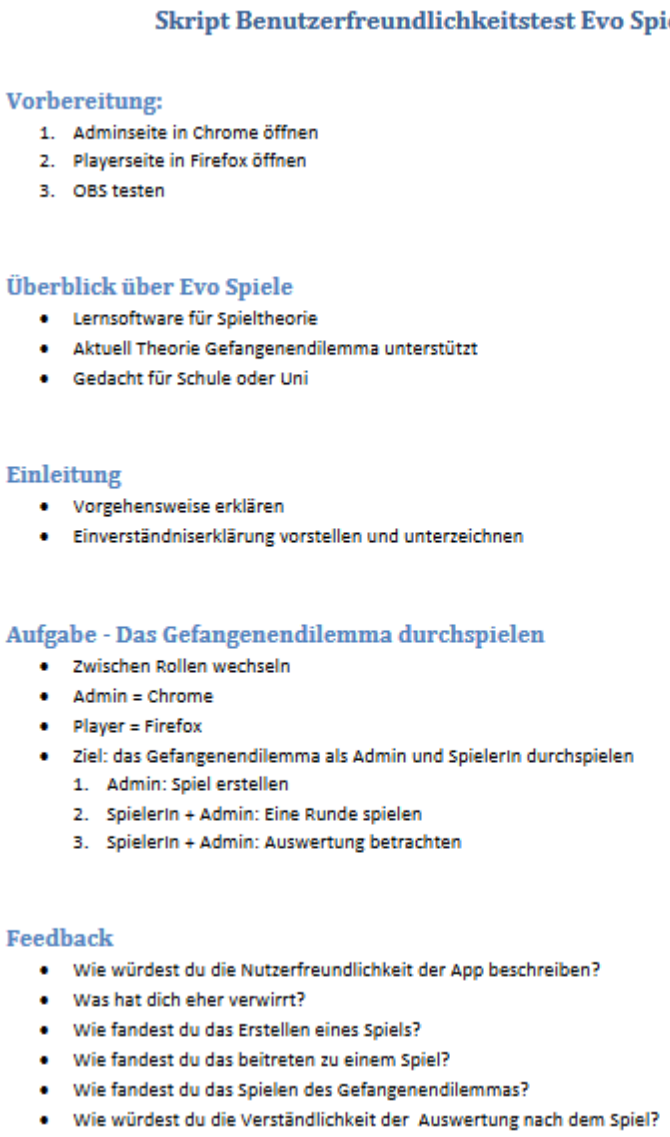
A.6 Ergebnisse Benutzbarkeitstests

A.6.1 Test 1

- Unterschied einfaches/iteratives Gefangenendilemma unklar.
- Link wurde statt Spielcode genutzt, um Spiel beizutreten. Ein falscher Code wird dabei auch nicht abgefangen.
- Unklar, was man kooperieren / betrügen bezweckt. Unklar, wer der Gegner ist.
- Unklar, was die Werte in den Herzen bedeuten.
- Unklar Unterschied zwischen Countdown starten und nächste Runde starten.
- Unklar, was das Ergebnis der ersten RUnde ist (vor allem, wenn man keine Puntke erhält).
- Punkteverlauf ersichtlich
- Punkte in der Abstimmung erstmal unklar
- Unklar, wer gewonnen hat, weil Bot nicht angezeigt wird.
- Unklar, was Kooperations/Betragsverhältnis bedeutet. Vor allem, dass es sich auf mehrere Spieler bezieht.
- Nicht verstanden, worums ging. Sinn des Spiels

A.6.2 Test 2

- Erstmal unklar, dass man die Punktewerte ändern kann.
- Versteht die Bedeutung der Einstellungen nicht und ändert daher nichts.
- Vermutet, das Spielcode und Spiel-ID das Gleiche ist.



Skript Benutzerfreundlichkeitstest Evo Spiele

Vorbereitung:

1. Adminseite in Chrome öffnen
2. Playerseite in Firefox öffnen
3. OBS testen

Überblick über Evo Spiele

- Lernsoftware für Spieltheorie
- Aktuell Theorie Gefangenendilemma unterstützt
- Gedacht für Schule oder Uni

Einleitung

- Vorgehensweise erklären
- Einverständniserklärung vorstellen und unterzeichnen

Aufgabe - Das Gefangenendilemma durchspielen

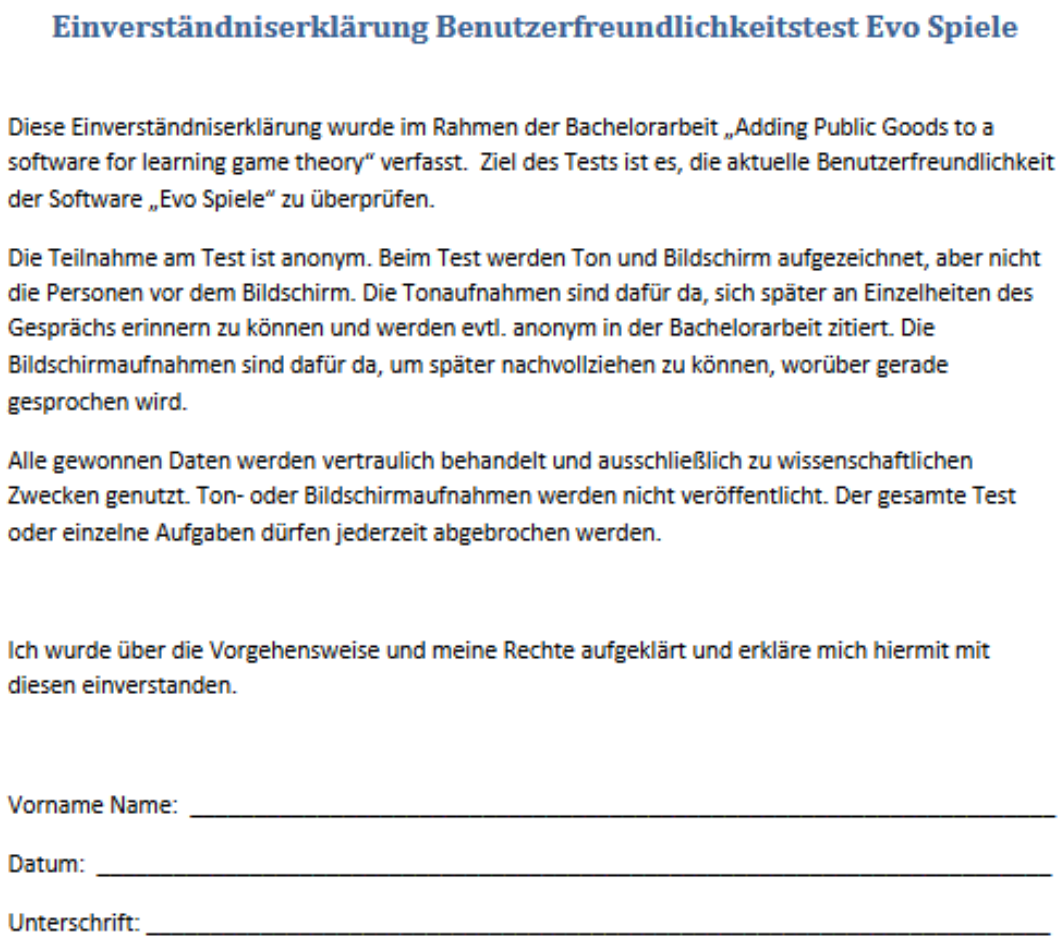
- Zwischen Rollen wechseln
- Admin = Chrome
- Player = Firefox
- Ziel: das Gefangenendilemma als Admin und SpielerIn durchspielen
  1. Admin: Spiel erstellen
  2. SpielerIn + Admin: Eine Runde spielen
  3. SpielerIn + Admin: Auswertung betrachten

Feedback

- Wie würdest du die Nutzerfreundlichkeit der App beschreiben?
- Was hat dich eher verwirrt?
- Wie fandest du das Erstellen eines Spiels?
- Wie fandest du das beitreten zu einem Spiel?
- Wie fandest du das Spielen des Gefangenendilemmas?
- Wie würdest du die Verständlichkeit der Auswertung nach dem Spiel?

Abbildung 27: Skript Benutzbarkeitstest

- Unklar, warum das Joinen lädt.
- Ich hab keine Ahnung, was das ist -> Deine Punkteverlauf
- Unklar, wo man drücken muss, weil sich die Maus nicht verändert.
- Versucht auf Punkteverlauf zu klicken.
- Unklar, mit wem man kooperiert. Ob mit der Polizei, oder mit dem Mitspieler.
- Warte mal, was? -> Unklar was Herzen bedeuten.
- Erwartet, dass man Countdown auf bei Spielern sieht.
- Ich würde gerne wissen, wie der Bot funktioniert.



Einverständniserklärung Benutzerfreundlichkeitstest Evo Spiele

Diese Einverständniserklärung wurde im Rahmen der Bachelorarbeit „Adding Public Goods to a software for learning game theory“ verfasst. Ziel des Tests ist es, die aktuelle Benutzerfreundlichkeit der Software „Evo Spiele“ zu überprüfen.

Die Teilnahme am Test ist anonym. Beim Test werden Ton und Bildschirm aufgezeichnet, aber nicht die Personen vor dem Bildschirm. Die Tonaufnahmen sind dafür da, sich später an Einzelheiten des Gesprächs erinnern zu können und werden evtl. anonym in der Bachelorarbeit zitiert. Die Bildschirmaufnahmen sind dafür da, um später nachvollziehen zu können, worüber gerade gesprochen wird.

Alle gewonnen Daten werden vertraulich behandelt und ausschließlich zu wissenschaftlichen Zwecken genutzt. Ton- oder Bildschirmaufnahmen werden nicht veröffentlicht. Der gesamte Test oder einzelne Aufgaben dürfen jederzeit abgebrochen werden.

Ich wurde über die Vorgehensweise und meine Rechte aufgeklärt und erkläre mich hiermit mit diesen einverstanden.

Vorname Name: _____

Datum: _____

Unterschrift: _____

Abbildung 28: Einverständniserklärung Benutzbarkeitstest

- Versteht, wie viele Punkte erhalten wurden und was das bedeutet.
- Ich will sehen, was meine letzte Runde ist. (wieveiele Punkte in der letzten Runde erhalten)
- Gesamtpunktzahl unklar.
- Kein Fan der Farben.
- Anmerkungen zu Styling auf Hauptseite.
- Anmerkungen zu Styling auf Einstellungsseite.
- Immernoch unklar, was Punktewerte ändern bewirkt.
- Statisiken live anzeigen rückwirkend klar.
- Stylingkommentare Playerjoin.

- Erneutes Spiel starten funktioniert nicht mehr.
- Leerzeichen wird nicht gelöscht beim Spielcode.

A.6.3 Test 3

- Erklärungstext unklar. Müsst ich mehr jetzt auf jeden Fall mehrmals durchlesen, ums richtig zu verstehen.
- Erklärungstext gendern.
- Punktematrix abgeschnitten.
- Bedeutung der Werte in Punktematrix sind unklar.
- Unklar, das Green Venus der Spielername ist. Wurde vermutet, dass das der Name des Spiels ist.
- Haken, wer abgestimmt hat, wird als richtig liegen interpretiert.
- Werte in der Matrix sind falsch zugewiesen.
- Erhaltene Punkte in der Auswertung werden komplett missverstanden (Geld = kooperieren, Blau = nicht kooperieren).
- Unklar, was die Spieler- und was die Lehrerauswertung ist.
- Farben eindeutiger einsetzen.

A.7 Designfeedback Stefan Nessler

- Es soll klar sein, dass es bei der Anwendung um Evolutionsbiologie geht.
- Der Beitrittslink / der Link der Webseite sollte relativ kurz und verbal sein.
- Leichterer Spielcode. Z.B. nur Ziffern (wie bei Mentimeter).
- Es wäre vllt. sinnvoll bei den Einstellungen eine Spieleranzahl anzugeben, damit man beim Beitritt leicht schauen kann, wie viele noch fehlen. Halte ich nicht für besonders sinnvoll, da das eine Einstellung mehr ist, und man auch manchmal garnicht die genaue Teilnehmerahl bestimmen kann oder will.
- Maximale Rundenzahl kann optional angegeben werden -> wird dann auch den SpielerInnen angezeigt.
- Klärung: Beim iterierten Gefangenendilemma spielt man immer mit der gleichen Person, beim einfachem mit verschiedenen Personen.
- Es wäre nicht schlecht, wenn die Resultate der Spiele auch als Grafik exportiert werden können.
- SpielerInnen sollen eine Möglichkeit haben, in Player Wait den Text zu Ende zu lesen und eigenständig einem Spiel beizutreten, nachdem dieses gestartet wurde.

- Das gleiche gilt für das Resultat einer Runde.
- Die bevorzugte Genderingvariante ist das *.
- Es soll die Möglichkeit / einen Zeitpunkt geben, wo die Lehrkraft die volle Aufmerksamkeit hat und Dinge erklären kann. SchülerInnen sollten daher dann keine Interaktionsmöglichkeit in der App haben. Das passiert idealerweise am Rundenende, damit man die Ergebnisse auswerten kann. Eine Pausierungsfunktion wäre dafür auch möglich.
- Rechnung bei Ergebnis in Public Goods ist zu kompliziert. Spenden daher da evtl. weglassen.
- Spenden sollten mit - dargestellt werden, um zu verdeutlichen, dass man Punkte abgibt.
- Beim Öffentlichen Gut wäre es nicht schlecht, wenn man tatsächlich von TriffbrettfaherInnen spricht in der Auswertungsgrafik zum Kooperationsverhalten.
- Einstellungen Öffentliches Gut: Spendenwerte für kooperieren und betrügen, Belohungsfaktor
- Varianten öffentliches Gut: a) Bestrafung / Reputation: Man kann gezielt Leute vom öffentlichem gut ausschließen auf Grund ihrer Reputation. Dass kann auch automatisch passieren. b) Aussteiger: SpielerInnen steigen aus dem öffentlichen Gut aus, kriegen keine Ausschüttung davon, sondern einen festen Betrag.
- Klärung: Indirekte Reziproität ist keine Variante des Öffentlichen Gutes.

A.8 Designfeedback Jamie Schenker

- Namen beim Joinen könnten Prozess spannender machen.
- Fetter Rahmen bei Markiertem in Tabelle -> stärker hervorheben.
- Bei Public Goods Bereiche Du spendest und Du erhälst voneinander trennen.
- Rechnung ersezten mit Gesamtspende = ... und Dein Anteil = ...

A.9 Evaluation Arbeitsgruppe Software Engineering

- Sollte man bei Public Goods schon mit Punkten starten?
- Sollte man bei Public Goods zwei Gruppen bilden, die dann gegeneinander antreten?

A.10 Evaluation Arbeitsgruppe Human-Centered Computing

- Header ist sehr groß -> überlegen, ob dieser kleiner geht.
- Sollte die Zielgruppe näher definieren und Tests mit dieser Gruppe machen.

- Alexa hat angeboten Feedback zum Design (insbesondere Konsitenz des Designs) per PDF zu geben.
- Inkonsistent sind z.B. Ränder an der Seite, weiße Balken sind unterschiedlich breit, Buttons genauso. Mal wird Spieler*innen mal Mitspieler*innen gesagt.
- Das alte Design im Playerrunning kam optisch verspielter und ansprechender an.
- Es könnte Sinn machen sich näher mit Gameification a la kahoot auseinanderzusetzen. Z.B. animierte Endplazierung, wackelnde Icons...

A.11 Evaluation Stefan Nessler

- Der Bereich für die Zahlen in den Einstellungen ist zu klein, sodass 2-stellige Zahlen abgeschnitten werden.
- Wer sind Alice und Bob in den Einstellungen?
- AdminWait Screen könnte während des Wartens auf Spieler*innen noch Hintergrund zum Spielmodus anzeigen.
- Bot soll immer abgestimmt haben.
- Stefan findet die direkte Ansprache z.B. Du kooperierst hilfreich und klar.
- Stefan findet den Spielansicht der Spieler zwar Zitat Nicht so sexy wie vorher, findet aber, dass die Anwendung verständlicher und klarer ist, was er auch als wichtiger gewichtet.
- Die Abstände zwischen den Spalten der Spieleransicht sind in der Desktopansicht zu groß, mobil sind sie aber okay.
- Nächste Runde Runde starten
- Punkte aus letzter Runde im Spielerscreen hervorheben sollte auf jeden Fall mit rein und ist sehr wichtig.
- Es wäre evtl. auch nicht schlecht, wenn man die Spielergebnisse als Grafiken exportieren könnte.
- Die Auswertungstabelle ist übersichtlicher als die vorherige Statistik. Allerdigs wäre sie noch nützlicher, wenn sie die Spalten kooperiert/kooperiert, betrügt/kooperiert, kooperiert/betrügt und betrügt/betrügt enthalten würde. Für Public Goods würden die Spalten kooperiert und betrügt reichen. Also im Prinzip sollte noch die Punktematrix in der Tabelle ergänzt werden, mit wie oft die entsprechenden Moves jeweils stattgefunden haben.
- Beim Erklärungstext, wenn man eine Runde vorzeitig weiterführen möchte, sollte das Wort Bot raus, weil es mit dem Bot für ungerade Spielerzahlen verwechselt werden kann.

- Beim Iterativem Gefangenendilemma könnte sich Stefan beim Kooperationsverlauf vorstellen, dass man dort 2 Spieler*innen (Partner im Dilemma) auswählen kann, für die man das Diagramm dann anschaut. Bei den Spielerresults kann dann immer der Kooperationsverlauf von einem selbst und dem/der Parter*in angezeigt werden.
- In der Spieleransicht sollten evtl. keine Daten über die gesamte Gruppe angezeigt werden. Bzw die Lehrkraft sollte darüber die Kontrolle haben. Grund dafür ist, dass der Fokus am Ende des Spiels dann wieder nach vorne zur Lehrkraft gelenkt werden soll. Eine Idee dafür ist, dass die Lehrkraft im Auswertungsscreen an den Grafiken toggeln kann, ob diese auch den Spieler*innen angezeigt werden.
- Der Datenexport wird beschrieben mit schöne, einfach Daten.
- Es sollte Feedback kommen, wenn man einen falschen Spielcode eingibt.
- Die Einstellung, dass man verschiedene Werte beim Betrug in Public Goods machen kann, ist unnötig.
- Bei den Erklärungstexten der Spielmodi könnte noch jeweils die englischen Namen in Klammern dahinter, weil diese in der Wissenschaft eher genutzt werden.
- Bei Spenden sollte angezeigt werden, wie das vom Gesamtbetrag abgezogen wird.
- Die gleiche Optik der Spielmodi findet Stefan gut, weil dadurch es leichter ist, verschiedene Modi zu spielen.
- Der Kooperationsverlauf sollte eine Legende haben.
- Der Kooperationsverlauf sollte vllt. doch eher ein Balkendiagramm sein, weil die Fläche verwirren könnte.
- Der Dateiname beim export von Public Goods sollte auf Public Goods geändert werden.
- Stefan steht Gameification skeptisch gegenüber, da er die Anwendung eher fachlich korrekt und sachlich haben möchte. Se sollte nicht zu verspielt und abgelenkt sein. Das war die erste Reaktion, er wäre aber auch Versuchen in diese Richtung offen.
- Die Idee, dass man statt Punkten auch Smarties bekommen könnte, kam erstmal gut an.
- Bei Public Goods wäre es spannend, wenn man die Klasse in 2 Gruppe aufteilen könnte, und 2 seperate Matches mit gemeinsamer Auswertung spielt.
- Zielgruppe: 10+ Klasse (16+ Jahre). Studium (eher Master, weil in Seminaren, sonst aber auch ab dem 2. oder 3. Semester). Biologie (sowohl Lehramt als auch Fachwissenschaftlich)

- Das Programm wird in den Unterricht eingebettet. Vorher werden erst einmal die Spieltheoretischen Konzepte und Hintergründe erklärt.

A.12 Evaluation Benutzbarkeitstests

A.12.1 Test 1

Neue Version Gefangenendilemma

- Findet irritierend, das Logo mit dem Rand abschließt.
- Findet Startseite clean.
- Kennt Unterschiede zwischen iterativen und normalen Gefangenendilemma nicht.
- Verwundert, warum es bei beiden Rundenanzahlen gibt.
- Wundert sich, warum Erweiterte Einstellungen so breit sind.
- Findet gut, dass wichtige Begriffe in der Spielerklärung hervorgehoben sind.
- Freut sich über Genderneutrale schreibweise.
- Ausgrauen von nicht benutzbaren Buttons funktioniert intuitiv.
- Ist neugierig, wie der Bot spielt.
- X bei Spieler*innen, die sich nicht entschieden haben ist verwirrend, solang noch niemand abgestimmt hat.
- Verwirrt, warum Bot sich noch nicht entschieden hat.
- Abfrage, ob man nächte Runde wirklich starten will, wenn noch nicht alle abgestimmt haben, funktioniert.
- Würde gerne in der Admin Spielansicht sehen, wie sich Spieler*innen entschieden haben.
- Punktematrix verständlich.
- Intuitiv wird erwartet, dass nach einer Runde ausgegraut wird, welche Entscheidung die Mitspieler*in nicht getroffen hat.
- Würde gerne in der Spieler*innen Ansicht sehen, wie man sich selbst und wie sich die Mitspieler*innen entschieden haben und wie viele Punkte man dafür bekommen hat.
- Irritert bei Spielabbruch, dass Spielabbruch hervorgehoben ist.
- Beim Kooperationsverlauf muss man hovern für Legende, Prozentwerte sind verwirrend. Würde absolute werte anstatt Prozentwerte bevorzugen.
- Hätte gerne eine ausführliche Auswertung auch über die einzelnen Runden. Interessiert ich für Kooperations-/Betrugskurven für einzelne Personen.

- Würde als Spieler*in gerne in der Auswertung sehen, was auch auf der Adminansicht zu sehen ist. Der Tester*in ist an der Stelle nicht bewusst, dass die Adminansicht für Spieler*innen ja auch auf dem Beamer zu sehen ist.
- Unklar, was man als Spieler*in nach dem Ende eines Spiel machen kann.
- Unklar, dass man auf das Logo klicken kann.

Neue Version Öffentliche Güter

- Rechnung in den Öffentlichen Güter Einstellungen ist schwierig zu lesen. Wünscht sich Darstellung als Bruch. Rechnung ist zu abstrakt, evtl. mit konkreten Werten.
- Verwirrt über Verzögerungen.
- Hervorheben, dass Spenden Minuspunkte bedeutet.

Alte Version

- Findet Startscreen ansprechender.
- Klarer, das Homebutton klickbar ist.
- Bemängelt kleinen Infotext bei Statistike live zeigen.
- Keine Genderneutrale Formulierung.
- Link zum Beitreten ist nicht klickbar.
- ID ist schwieriger zu merken.
- Button zum Spiel erstellen ist außerhalb des Bildschirms.
- Verbringt viel Zeit in dem Erklärungstext, bemängelt eininge Inkonsistenzen.
- Fand Pseudonyme vorher besser (obwohl diese intern gleich geblieben sind). Der vorher war wohl leichter zu merken.
- Unklar, was der Kreis beim Beitreten bedeutet, der anzeigen soll, dass jemand beigetreten ist.
- Findet Spieler*innen Seite sehr ansprechend.
- Fragt sich, ob Herzchen Entscheidung beeinflussen.
- Verwirrt, dass es keinen Hovereffekt gibt. Daher ist es unklar, dass man auf eine Seite klicken kann.
- Versteht falsch, wer wie viele Punkte bekommt. Grund dafür ist vor allem die Farbgebung der Lila Herzen, weil zum einen erwartet wird, dass das dunkle lila man selbst ist, und dass dunkle lila auch sonst verwendet wird.
- Verwundert, dass in bisherigen Tabelle nicht diese Farbgebung verdendet wird.

- Entdeckt, dass man sich umentscheiden kann.
- Rechnet nicht damit, dass man das in der Neuen Version konnte, weil das ausgegraut war.
- Überrascht, dass man Countdown starten kann.
- Erwartet, dass dieser auch bei Spieler*innen angezeigt wird.
- Würde gerne sehen, in welcher Runde man gerade ist.
- Findet Auswertungsgrafik verwirrend.
- Rechnet mit Warnung für Spielabruch.
- Verwirrt, warum Punkte des Bots nicht in der Auswertung zu finden ist.
- Möchte allgemein mehr Daten in der Auswertung.

A.12.2 Test 2

Alte Version

- Bemängelt kleinere Designmängel, die in der neuen Version schon behoben wurden.
- Iteratives Gefangenendilemma stürzt ab.
- Bemerkt, dass es mal Spielcode und mal Spiel ID heißt.
- Spielbeitritt stürtzt ab, weil ein Leerzeichen hinter dem Code eingegeben wurde.
- Wünscht sich schon Erklärung des Spiel während des Erstellens.
- Punkteverlauf irritierend.
- Bemerkt, dass nicht die gleichen Begriffe für z.B: kooperien verwendet werden.
- Ist von der Punktematrix verwirrt, weil angenommen wird, dass man das dunkle lila ist.
- Verwirrt, dass Bot noch nicht abgestimmt hat.
- Würde gerne wissen, was Mitspieler*in entschieden hat.
- Bemerkt, dass Punktematrix falsch eingestellt ist.
- Nicht ganz sicher, dass unten die Gesamtpunktzahl angezeigt wird.
- Würde gerne eigene Punktezahl besser einordnen können. Interessiert sich für Punktezahlen, Platzierungen und auch die Entscheidungen der anderen Spieler*innen.
- Versteht Kooperationsverhältnis schnell.

- Wünscht sich, Einstellungen zurücksetzen zu können.
- Versteht Punktetabelle in den Einstellungen nur durch Kontextwissen. Sonst wäre es unklar, wer wie viele Punkte bekommt.
- Vermutet durch Ausschluss andere Strategien, dass der Bot random entscheidet. Findet es nicht wichtig, dass das in der Anwendung erklärt wird, es sei denn es kommen noch weitere Bots hinzu.

Neue Version Gefangenendilemma

- Irrittiert vom Logo, wegen den Abständen zwischen den Buchstaben.
- Fand größere Buttons von der alten Version im Hauptmenu besser.
- Punktewerte werden besser erklärt.
- Wünscht sich Erklärung der Spielmodi auch bei den Einstellungen.
- Verwirrt von lilanen Hervorhebungen, weil diese wie links wirken. Hervorhebungen werden als sinnvoll eingeschätzt, ma sollte diese aber evtl. anders hervorheben als Links, indem man z.B. links unterstreicht.
- Findet Erklärung der Punkteverteilung verständlicher.
- Verwirrt, dass man nach dem Beitreten nochmal auf Spiel beitreten drücken muss. Das sollte wahrscheinlich in Spiel starten umbenannt werden.
- Spielansicht bleib in einer Runde stecken.
- Denkt, dass man über das X, das anzeigt um jemand abgestimmt hat, Spieler*innen aus dem Spiel werfen kann.
- Starten vom Iterativen Gefangenendilemma funktioniert nicht.
- Entdeckt, dass man Entscheidung noch wechseln kann.

Neue Version Öffentliche Güter

- Matrix in den Einstellungen wurde übersprungen, weil sie erstmal nicht so verständlich wirkt.
- Nach einem Spiel weiterspielen funktioniert nicht immer. Anscheinend sind da zu viele / noch alte Spieler*innen im Spiel.
- Versteht die Matrix und deren Bedeutung im Spiel.
- Wünscht sich Punktevorrat am Anfang und Klärung, dass Punkte abgezogen werden.
- Identifiziert Unterschied zwischen den beiden Gefangenendilemmas im Warteraum der Spieler*innen.

B Eingereichte Dateien

B.1 Zusammenfassung Analyse

Dateinname: Ergebnisse Heuristische Analyse und Benutzerfreundlichkeitstests.ods

B.2 Quelltext Frontend

Dateinname: Frontend.zip

B.3 Quelltext Backend

Dateinname: Backend.zip

B.4 Feedback Alexa Schlegel

Dateinname: Feedback von Alexa Schlegel.pdf