Freie Universität Berlin

Bachelorarbeit am Institut für Informatik der Freien Universität Berlin

Arbeitsgruppe Software Engineering

Kooperationsspiele für den Schulunterricht - Visualisierung der Spieldaten

Zoe Wiget
Matrikelnummer: 5177548
zoe.wiget@fu-berlin.de

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

Berlin, 8. März 2022

Zusammenfassung

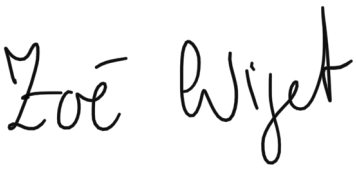
Im Wintersemester 2019/2020 wurde im Rahmen des Softwareprojektes „Kooperationsspiele für den Schulunterricht“ eine webbasierte Anwendung entwickelt, dessen Zweck es ist die Lehrkraft im Unterrichts spielerisch dabei zu unterstützen die Konzepte der Spieltheorie zu vermitteln. Bestandteil dieser Arbeit war es, Konzepte zu entwickeln um die erzeugten Spieldaten so zu visualisieren und implementieren, dass sie die Lehrkraft optimal in der Lehre unterstützten. Anschließend wurden die implementierten Diagramme und Tabellen anhand von Richtlinien bezüglich der Datenvisualisierung untersucht um festzustellen, ob sie den oben genannten Zweck erfüllen. Des Weiteren sollte eine Möglichkeit geschaffen werden, die Spieldaten in Form einer Datenbank zu speichern und für einen späteren Zeitpunkt wieder verfügbar zu machen. Die implementierten Diagramme und Tabellen wurde hierfür weitestgehend als geeignet befunden. Eine Datenbank wurde implementiert.

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.

8. März 2022

Zoe Wiget



Inhaltsverzeichnis

1 Einführung

Die Anwendung, welche als Grundlage dieser Arbeit dient, ist im Rahmen eines Softwareprojekts im Jahr 2019/2020 entstanden. Inhalt des Projekt war es für Dr. Stefan Nessler, aus dem Fachbereich Biologie, eine webbasierte Anwendung zu entwickeln, welche Lehrkräfte dabei unterstützen soll, die Konzepte der Spieltheorie im Unterricht zu verdeutlichen.
Die Anwendung wurde bis heute von Jannis Hamann in einer Bachelorarbeit um den Spielmodus „Öffentliche Güter“erweitert und von Jamie Schenker in einer Masterarbeit so umstrukturiert, dass sie auf einem Webserver gehostet werden kann.
Ziel dieser Arbeit ist es zum einen, die Spieldaten so zu visualisieren, dass sie für die Lehrenden besser im Unterricht eingesetzt werden können und für die Schüler*Innen/ Studenten leicht verständlich sind. Zum anderen sollen die Spielergebnisse in einer Datenbank gespeichert werden, so dass sie zu einem späteren Zeitpunkt, zum Beispiel am nächsten Tag, erneut aufgerufen und besprochen werden können, da sie zurzeit nicht mehr aufrufbar sind, nachdem der Administrator auf „Spiel verlassen“klickt.

2 Beschreibung des Projekts

In diesem Abschnitt beschreibe ich einmal die verschiedenen Spielmöglichkeiten der Spieltheortischen Grundlagen die in der Anwendung implementiert worden sind und gehe anschließend darauf ein, wie das Programm an sich funktioniert. Zusätzlich erläutere ich wie der aktuelle Stand im Bezug auf die Visualisierung der Daten aussieht und gehe kurz darauf ein, wieso diese für die Lehre nicht optimal dargestellt werden.

2.1 Spielerklärung

Es gibt mehrere Spielmodi, die man bei der Erstellung eines Spiels auswählen kann. Ich beschreibe hier nur, wie die einzelnen Modi ablaufen, diskutiere jedoch nicht die Spieltheoretischen Grundlagen dahinter, sondern gehe nur grob darauf ein. Bei dem einfachen Gefangenendilemma wird in Runden gespielt. Hierbei werden zwei Spieler jede Runde zufällig gepaart. Jeder Spieler kann pro Runde abstimmen, ob dieser mit seinem Gegenüber kooperieren oder diesen betrügen möchte. Je nachdem wie sich die beiden Spieler entscheiden, bekommen sie eine unterschiedliche Anzahl an Punkten. Diese werden am Anfang des Spiels in einer Punktematrix eingetragen und ändern sich während des Spiels nicht.
Sollten die Spieler miteinander kooperieren, so bekommen beide in der Regel 3 Punkte. Wenn jedoch einer der Spieler betrügt, so bekommt dieser 5 Punkte und der kooperierende Spieler 0 Punkte. Wenn beide Spieler betrügen, erhalten sie jeder einen Punkt. Dies hat Auswirkungen auf das Verhalten der Spieler in der darauf folgenden Runde. Wurde ein kooperierender Spieler betrogen und erhält keine Punkte, so ist es wahrscheinlicher, dass er in der nächsten Runde auch zum Betrügen neigt.

Beim iterativen Gefangenendilemma, werden in der ersten Runde zwei Spieler zufällig zugeteilt und spielen bis zum Ende des Spiels gegeneinander. Der Rest des Spiels verläuft genauso wie beim einfachen Gefangenendilemma.

Ein weiteres Spielformat ist Öffentliche Güter. Hierbei spielen die Spieler nicht gegen jeweils eine andere Person sondern als Gruppe. Am Anfang jeder Runde entscheidet jeder Spieler, ob er von seinen Punkten eine festgelegte Anzahl in einen großen Topf einzahlt oder nicht. Die eingezahlten Punkte werden mit einem zu Beginn des Spiels festgelegten Faktor multipliziert und dann unter allen Spielern wieder aufgeteilt. Da Spieler in diesem Spielformat erst einmal von ihren eigenen Punkten etwas in den gemeinsamen Topf einzahlen müssen und somit Punkte verlieren können, ist hier bereits schon zu Beginn die Motivation zu betrügen höher. Betrügen jedoch alle Spieler so bekommt keiner mehr Punkte.

2.2 Beschreibung der Anwendung

Die Lehrkraft kann die aktuelle Version der Anwendung über https://evo.imp.fu-berlin.de/ öffnen. Dort kann der Lehrende über „Spiel erstellen“ein neues Spiel starten. Auf der Spielerstellungseite kann der Spielmodus ausgewählt, die Rundenanzahl und die Spielmatrix eingetragen werden. Standartwerte für die Matrix sind bereits eingestellt. Nachdem erneut auf „Spiel erstellen“gedrückt wird, öffnet sich eine Seite auf der ein Link und der Spielcode der aktuellen Sitzung angezeigt wird. Über den Link können die Schüler, bei Eingabe des Spielcodes, der Sitzung beitreten. Sollte eine

ungerade Anzahl an Spielern an einem Spiel teilnehmen, so stimmt ein Bot automatisch mit ab, da eine gerade Anzahl an Spielern erforderlich ist.
Wenn mindestens ein Spieler beigetreten ist, kann das Spiel gestartet werden. Während eine Runde läuft, wird auf der Adminseite die Punkteübersicht der einzelnen Spieler, sowie eine Übersicht darüber wer alles abgestimmt hat angezeigt. Nachdem eine Runde abgeschlossen wird, aktualisiert sich die Punkteübersicht. Wenn das Spiel abgeschlossen ist, wird die Abstimmungsübersicht durch ein Diagramm ersetzt, welches den Kooperationsverlauf der Spieler in Prozent anzeigt. Die Diagramme werden im Kapitel „Analyse“ noch einmal genauer besprochen und analysiert.
Anschließend kann die Sitzung komplett beendet werden, indem man zum Hauptmenü zurückkehrt, oder es kann ein neues Spiel gestartet werden. Jeder Spieler der bereits Teil der Sitzung ist, kann automatisch am nächsten Spiel teilnehmen und muss sich nicht neu verbinden.
Wenn ein Spieler der Sitzung über https://evo.imp.fu-berlin.de/join beitritt und den Spielcode eingegeben hat, wird diesem eine kurze Erklärung zum aktuellen Spielmodus angezeigt. Wird das Spiel vom Admin gestartet, so wechselt die Ansicht zu einer Matrix, die erklärt welche Anzahl an Punkten der Spieler abhängig vom eigenen Verhalten und dem seines Spielpartners bekommt. Der Spieler kann nun anhand von zwei Schaltflächen entscheiden, ob er kooperieren oder betrügen möchte. Diese Entscheidung ist final und kann in der laufenden Runde nicht mehr geändert werden.
Zudem wird auf der Seite angezeigt, wie viele Punkte der Spieler bereits hat. Wenn das Spiel komplett abgeschlossen ist, wird auf der Spielerseite eine Gesamtpunkteübersicht aller Spieler angezeigt.

2.3 Analyse

2.3.1 Grundlagen der Diagrammkompetenz und Richtlinien der Visualisierung von Daten

Bevor ich untersuche weshalb die derzeit implementierten Diagramme und Tabellen die Lehrkraft bei der Lehre nicht optimal unterstützen können, gehe ich erst ein mal darauf ein, zu welchem Zweck man Diagramme und Tabellen erstellt und werde hervorheben welche der genannten Aspekte für die Anwendung wichtig sind. Zudem werde ich darauf eingehen, wie Diagramme von Schülern in der Lehre interpretiert und wahrgenommen werden, um anschließend Richtlinien und Design-Prinzipien vorzustellen, die für die Visualisierung von Daten wichtig sind um diese optimal und verständlich zu gestalten.

In dem Buch „Methoden Handbuch Biologie“[8], herausgegeben von Prof. Dr. Stripf, wird auf Seite 359 eine Übersicht darüber gegeben, wofür Tabellen und Diagramme überhaupt erstellt werden. Die Punkte die hier genannt werden sind folgende:[8][p.359]

- Festhalten/Protokollieren von Versuchsergebnissen oder Beobachtungen
- Veranschaulichung quantitativer Zusammenhänge
- Auswertung eines Textes auf wichtige Aussagen und deren Beziehungen hin, meist unter einem vorgegebenen Aspekt, und Veranschaulichung in grafischer

Form

- Hervorhebung, Strukturierung, Gegenüberstellung von Aussagen
- Konstruktion eigener Wissensnetze
- Erhöhung der Übersichtlichkeit durch schematische Darstellung
- leichte Einprägsamkeit von Zusammenhängen

Veranschaulichung quantitativer Zusammenhänge, sowie Erhöhung der Übersichtlichkeit durch schematische Darstellung sind zwei Punkte die hier genannt werden und aus meiner Sicht wichtig sind im Bezug auf die Diagramme die für diese Arbeit implementiert wurden. Die Veranschaulichung quantitativer Zusammenhänge ist im Falle unserer Anwendung wichtig, da anhand der Diagramme aufgezeigt werden soll, wie sich zum einen das Kooperationsverhalten des einzelnen, aber auch der Gruppe im Verlaufe des Spiels verändert. Dies kann durch eine schematische Darstellung des Kooperationsverhaltens als Graph noch verdeutlicht werden. Bei guter Umsetzung der Diagramme kann es durch die Veranschaulichung quantitativer Zusammenhänge auch eine leichtere Einprägsamkeit von besagten Zusammenhängen im Bezug auf die theoretischen Grundlagen der Spieltheorie zur Folge haben.

Um zu ermitteln, ob sich die implementieren Diagramme für die Lehre eignen, muss man sich erst mal bewusst machen wie Schüler Diagramme wahrnehmen. Dies haben Lachmeyer, Nerdel und Prechtl in ihrem Artikel „Modellierung kognitiver Fähigkeiten beim Umgang mit Diagrammen im naturwissenschaftlichen Unterricht“[7] aus dem Jahr 2007 untersucht.
Ich werde im weiteren Verlauf einige Punkte aus dem Artikel hervorheben, die aus meiner Sicht von Bedeutung sind. Die Autoren erläutern zunächst, dass es zwei Arten gibt auf die das Lernen mit Diagrammen erfolgen kann. [7][p. 147] Schüler nutzen entweder ein bereits vorhandenes Diagramm oder visualisieren selbst eines anhand von Daten, welche vorgegeben oder eigenständig erhoben worden sind. Beides erfordert unterschiedliche kognitive Fähigkeiten. Für uns ist jedoch nur der erste Punkte wichtig, da die Diagramme in der Anwendung schon gegeben sind. Wir werden uns im weiteren Verlauf also darauf fokussieren und herausarbeiten, wie Schüler*Innen hieraus Informationen gewinnen.
In dem Artikel erörtern die Autoren mehrere theoretische Modelle zur Informationentnahme. Eins dieser Modelle wurde von Bertin (1974; zitiert nach [7]) aufgestellt. Dieser unterteilt die Informationsentnahme in zwei Bereiche: Identifizierung und Ablesen.
Die Identifizierung erfolgt in der Regel zuerst. Hierbei werden die Achsen eines Diagramms betrachtet und die betrachtende Person erörtert, welche Variable auf welcher Achse eingetragen ist und kann an diesem Punkt eventuell schon erkennen, bei welcher es sich um die abhängige und unabhängige Variable handelt. Somit kann die betrachtende Person bereits erste Aussagen über das Diagramm treffen, ohne sich mit dem genauen Inhalt auseinander auszusetzen. Zum Beispiel im Falle der Anwendung kann ein Schüler bei Betrachtung des Liniendiagramms, wenn dies optimal beschriftet ist, für sich die Aussage treffen: „Hier wird das Kooperationsverhalten der Spieler in Prozent pro Runde angezeigt.“

Die Autoren heben in ihrem Artikel die Bedeutsamkeit von Identifizierung noch einmal hervor. Sie geben hier die Studie von Maichle (1994; zitiert nach [7][p.148]) an. Lachmeyer, Nerdel und Prechtl geben an, dass in dieser Studie festgestellt wurde, dass Personen, die sich erst dem Inhalt des Diagramms widmeten und anschließend den Achsen, deutlich mehr Zeit für den Ableseprozess benötigten, als die Personen die sich vorher mit den Achsen auseinandergesetzt hatten.
Daraus kann man ableiten, dass eine aussagekräftige und korrekte Beschriftung der Diagrammachsen ein wichtiger Punkt bei der Visualisierung der Daten und deren Ableseprozessen ist.
Anschließend gehen Lachmeyer, Nerdel und Prechtl darauf ein, wie die Diagramme abgelesen werden. Sie erläutern, dass nach Bertin (1974), Wainer(1992 zitiert nach [7]) der Bertins Einstufung des Ablesens aufgriff und Schnotz(1993; zitiert nach [7]), welcher sich in seiner Kategorisierung an Bertins und Wainers anlehnt, das Ablesen in 3 Kategorien oder Stufen unterteilt wird. Die drei Stufen geben der Komplexität der Informationsentnahmen an. Die erste Stufe, auch Ablesen erster Ordnung (Schnotz(1993) ; zitiert nach [7] [p. 150]) bezieht sich auf einzelne Elemente. So könnte in unserem Fall eine Fragestellung lauten: „Wie viele Punkte hat Spieler x in Runde y erzielt?“ Hierfür muss lediglich ein Wert aus dem Diagramm abgelesen werden.
Die zweite Stufe, auch Ablesen zweiter Ordnung (Schnotz(1993); zitiert nach [7] [p. 150]), bezeichnet einen Vergleich von einzelnen Werten. Beispielsweise werden an dieser Stelle Punkte von zwei Spielern direkt verglichen. Bei der dritten Stufe, auch Ablesen dritter Ordnung (Schnotz(1993): zitiert nach [7]), wird eine Gruppe an Werten miteinander verglichen. Hier wird zum Beispiel der Punkteverlauf der Spieler mit dem Kooperationvsverhalten in der aktuellen Runde zur Vorrunde verglichen.
Die Autoren beschreiben in ihrem Artikel auch die Einteilung von Curcios (1987; zitiert nach [7] [p. 151]). Seine Einteilung ähnelt der von Schnotz, auch wenn sie etwas gröber ist. Curcios inkludiert in seiner Einstufung noch, das Lesen über die Daten hinaus als höchste Kategorie. Diese erfordert also vom Betrachter des Diagramms, die kognitive Fähigkeit, Schlussfolgerungen ziehen und Vorhersagen, basierend auf den Daten tätigen zu können. Die Fragestellungen zu den einzelnen Stufen werden also mit jeder Stufe komplexer. Aus diesem Grund ist es aus meiner Sicht wichtig, die Datenpunkte der Diagramme so übersichtlich und eindeutig wie möglich zu gestalten, damit die Schüler nicht mit unnötigen Informationen oder Darstellungen von der wesentlichen Fragestellung abgelenkt werden. Ob die implementierten Diagramme das Interpretieren und Ablesen unterstützen werde ich in der Diskussion im nächstem Kapitel erörtern.

Vorher werde ich noch einen Überblick über die Richtlinien und Design-Prinzipien für Datenvisualisation geben. Diese Richtlinien und Design-Prinzipien werde ich den Publikationen „Ten guidlines for effective data visulization in scientific publications“[9] von Christa Kelleher und Torsten Wagener, sowie „Design Principles for Data Visualization in Evaluation“[2] von Stepahnie Evergreen und Chris Metzner entnehmen und kurz die wichtigsten Punkte zusammenfassen. Anschließend werde ich in der Diskussion überprüfen, ob die Diagramme sowie Tabellen diesen Kriterien entsprechen und wie sie eventuell verbessert werden können.
In dem erst genannten Artikel stellen die Autoren 10 Richtlinien auf, die man für Graphen und Diagramme anwenden sollte, um deren Visualisierung so effektiv wie

möglich zu gestalten. Für die in der Anwendung verwendeten Diagramme ist eine wichtige Richtlinie *Guideline 1: create the simplest graph that conveys the information you want to convey (Tufte, 1983 [pp. 91-137])* [9][p.822]. Auch die Autoren Evergreen und Metzner heben in ihrem Artikel hervor, das es besser ist ein Design einfach zu halten und sogenanntes „Visual Noise“ (Tufte 2001; zitiert nach [2]) zu minimieren um den Betrachter nicht unnötig zu verwirren oder visuelle zu über stimulieren. Auch sollten die Daten, die in den Diagrammen präsentiert werden, eher knapp und präzise beschriftet werden, wenn eine Beschriftung überhaupt notwendig ist. Dies wird in Richtline 2 in Kelleher und Wagener als auch im Kapitel Data in dem Artikel Evergreen und Metzner erläutert. Evergreen und Metzner verweisen hier darauf, dass man Daten auch interaktiv gestalten kann. So kann man zum Beispiel mit dem Herüberfahren der Maus über einzelne Punkte in einem Graph, diese hervorheben, oder noch zusätzliche Beschriftungen einfügen, die dann sichtbar werden. So wird „Visual Noise“minimiert, aber die Informationen bleiben trotzdem erhalten.
In beiden Artikeln wird zudem hervorgehoben, eine sinnvolle Achsenbeschriftung zu verwenden. Die Achseneinteilung sollte bei einem zweckmäßigen Wert anfangen und die Anordnung eine sinnhafte Abstufung haben. Wenn sich zum Beispiel ein Graph auf der Y-Achse ausschließlich zwischen den Werten 600-800 bewegt, macht es Sinn den Achsenursprung bei 600 starten zu lassen. Die Abstufung könnte nun in fünfziger Schritten bis 800 oder 850 erfolgen. So bekommt man eine sinnvolle Darstellung des Graphen und kann besser einzelne Werte ablesen oder kleinere Veränderungen in der Steigung sehen. Wenn stattdessen der Achsenursprung bei 0 startet und bis 1000 geht wird der Graph zu stark gestaucht, um tiefgreifende Aussagen darüber zu treffen.
Ein weiterer wichtiger Aspekt der in beiden Artikeln aufgegriffen wird, ist die Verwendung von Farben. Zum einen wird in beiden Publikationen empfohlen, möglichst Farben in neutralen Tönen, wie Graustufen zu verwenden. Wenn die Farben eine wichtige Rolle spielen, macht es Sinn diese so zu wählen, dass man die wichtigen Punkte die vermittelt werden sollen, voneinander abgrenzen kann. Wenn zum Beispiel der Temperaturverlauf über ein Jahr hinweg farblich dargestellt werden soll, machen Graustufen wenig Sinn und es ist besser verschiedene Farben für verschiedene Temperaturabstufungen zu verwenden. Man sollte bei der Farbauswahl auch beachten, dass ähnliche Farben oder Farbstufen als zusammengehörig interpretiert werden können. Im Bezug auf das Beispiel mit dem Temperaturverlauf, sollte man beachten, dass unterschiedliche Farben, wie Blau für [0]°C und Rot für [100]°C genommen werden, um einen klaren Unterschied aufzuzeigen.
Kelleher und Wagner nennen als eine weitere Richtlinie sequentielle Daten in Zeitfolgen miteinander zu verbinden, da diese Daten eine lineare Veränderung zwischen den einzelnen Punkten implizieren. Nur wenn die Daten nichtsequentiell sind und sich in verschiedenen Gruppen befinden, wie in einem Streudiagramm oder kategorialen Daten, sollten die Linien nicht verbunden werden.

2.3.2 Analyse aktueller Diagramme

Meine ersten Schritte waren, mich mit dem aktuellen Stand der Anwendung vertraut zu machen. Hierfür habe ich zum einen die Abschlussarbeiten von Jannis Hamann und Jamie Schenker gelesen. Zum anderen habe ich mich mit der aktuell laufen-

den Version der Anwendung auf https://evo.imp.fu-berlin.de/ vertraut gemacht. Anschließend habe ich ein Gespräch mit Dr. Nessler geführt, um herauszufinden, welche wünsche er im Bezug auf die Visualisierung der Daten hat, da bereits in der Arbeit von Jannis Haman darauf eingegangen wurde, dass Dr. Nessler mit dem aktuellen Stand nicht 100% zufrieden ist, was die Visualisierung der Daten angeht.
Im folgenden werde ich eine kurze Übersicht darüber geben, wie die Spieldaten derzeit visualisiert werden und erläutern, was problematisch an der gewählten Darstellungsform ist und weshalb sie für die Lehre ungeeignet sind. Es gibt mehrere Zeitpunkte, zu denen in der Anwendung die Spielergebnisse angezeigt werden.
Nach jeder Runde gibt eine eine Übersicht darüber, wie sich die Punkte der einzelnen Spieler verändert haben. Diese Seite wird im Normalfall über einen Beamer von der lehrenden Person geteilt, so dass die Ergebnisse und die Punkteveränderung besprochen werden können. Zudem wird auf dieser Seite angezeigt, wer während der laufenden Runde schon abgestimmt hat.

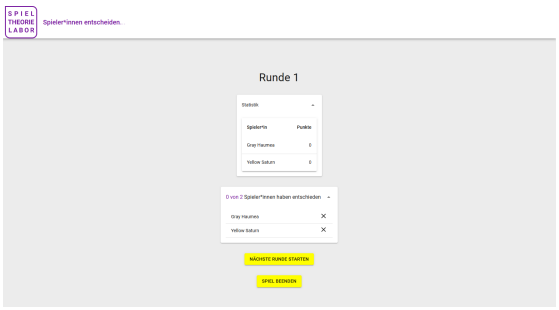


Abbildung 1: Punkteübersicht: Laufende Runde

Am Ende eines Spiels wird die Ansicht um einen Graphen erweitert, der den Kooperationsverlauf aller Spieler pro Runde anzeigt.

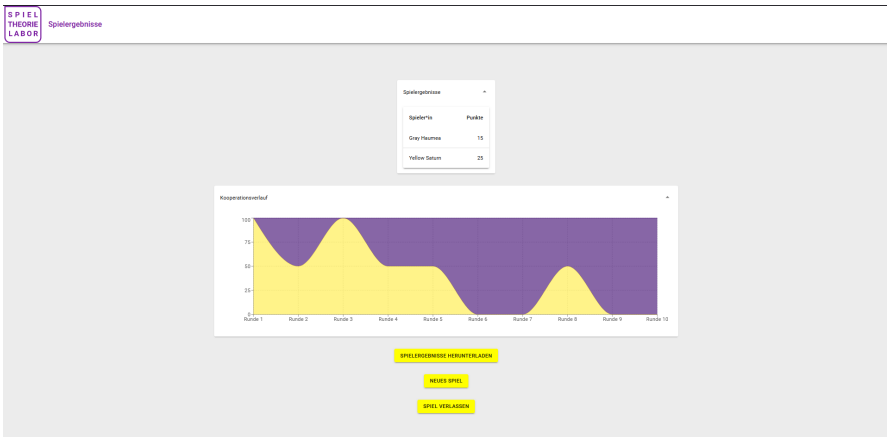


Abbildung 2: Punkteübersicht: Spiel abgeschlossen

Betrachten wir nun einmal die Diagramme, in denen die Spieldaten visualisiert werden. Als erstes gibt es die Tabelle mit der Punkteübersicht aus Abbildung 3.

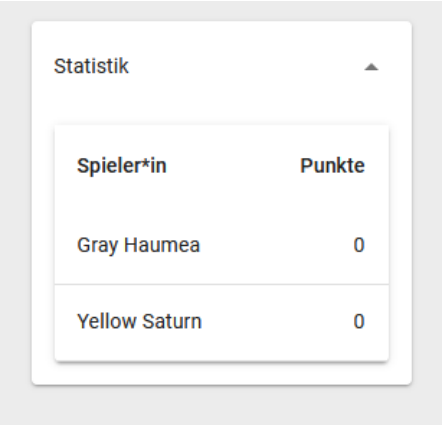


Abbildung 3: Tabelle: Punkteübersicht

In dieser Tabelle wird nur die Gesamtpunktzahl der einzelnen Spieler angezeigt. Man hat keinen direkten Vergleich zur Vorrunde. Gerade dieser Vergleich ist jedoch ein wichtiger Aspekt der Anwendung. Denn nach jeder Runde soll man die Möglichkeit haben, die Unterschiede besprechen zu können. Dies ist mit der aktuellen Darstellung so gut wie unmöglich.

Das andere Diagramm, welches derzeit implementiert ist, ist ein Liniendiagramm, welches den Kooperationsverlauf visualisiert.

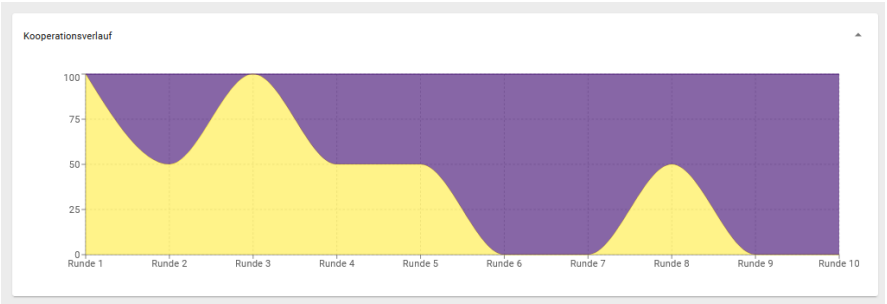


Abbildung 4: Tabelle: Punkteübersicht

An dieser Darstellung sind mehrere Punkte problematisch. Die auch schon in der Arbeit von Janis Hamann aufgegriffen wurden und von Dr. Nessler in einem Gespräch angemerkt wurden. Zum einen fehlt eine aussagekräftige Achsenbeschriftung an der Y-Achse. Zum anderen ist die farbliche Darstellung des Graphen irreführend und Dr. Nessler wäre ein simpleres Design lieber, bei dem die Hintergrundfarbe einheitlich ist.

3 Hauptteil

3.1 Konzeptentwicklung

3.1.1 Visuelle Daten

Zu Beginn der Arbeit war der Plan, dass ich zwei Konzepte für die Visualisierung der Daten entwickle und diese anschließend implementiere und vergleichen lasse. Der Sinn dahinter ist, dass ein Konzept für die Visualisierung entwickelt und implementiert wird, welches die Lehrenden dabei unterstützt um den Schüler die Konzepte der Spieltheorie leichter und verständlicher näher bringen zu können.

Um die Konzepte zu entwickeln habe ich erst mal ein Gespräch mit Dr. Nessler geführt und mich mit den anderen Arbeiten zu dieser Anwendung auseinander gesetzt, um Anhaltspunkte dafür zu bekommen, wie man die Daten am besten visualisieren kann. Zwei Punkte wurden hier am meisten hervorgehoben. Zum einen, dass der bestehen Graph nicht beschriftet und die Darstellung verwirrend ist. Zum anderen, dass es Dr. Nessler wichtig ist, dass man sehen kann, wie sich die Punkte der Spieler*innen im Vergleich zur Vorrunde verändert haben.

Wichtig für die Konzepte sind also zum einen die Gesamptpunkteübersicht für alle Spieler und das Diagramm, welches den Kooperationsverlauf visualisiert. Anschließend stelle ich die beiden Konzepte vor, die aus meiner Sicht zu Beginn der Arbeit sinnvoll waren:

- Konzept 1:
  Ich habe von Dr. Nessler eine Exceldatei bekommen, die bisher verwendet wurde, um die Spiele im Unterricht zum spielen. Diese habe ich auch als Grundlage für das erste Konzept verwendet. Es erfüllte die Punkte, die Dr. Nessler im Zusammengang mit der Visualisierung wichtig sind. Zudem hat es sich bereits in der Lehre als hilfreich erwiesen, weshalb ich es auch für eine gute Grundlage für die Visualisierung der Daten halte.

  - Gesamtpunkteübersicht:
    In der Exceldatei ist dies als zweifarbiges Säulendiagramm dargestellt. Hierbei werden jede Runde die Punkte, die neu dazugekommen sind, in einer anderen Farbe als der Rest angezeigt. Zusätzlich dazu hatte ich den Plan eine Funktion einzubauen, sodass Runden ausgewählt und nur diese angezeigt werden, um diese noch einmal deutlicher hervorheben zu können bei der Besprechung der Daten.

  - Kooperationsverlauf:
    Der Kooperationsverlauf wird als ein einfaches Liniendiagramm dargestellt.

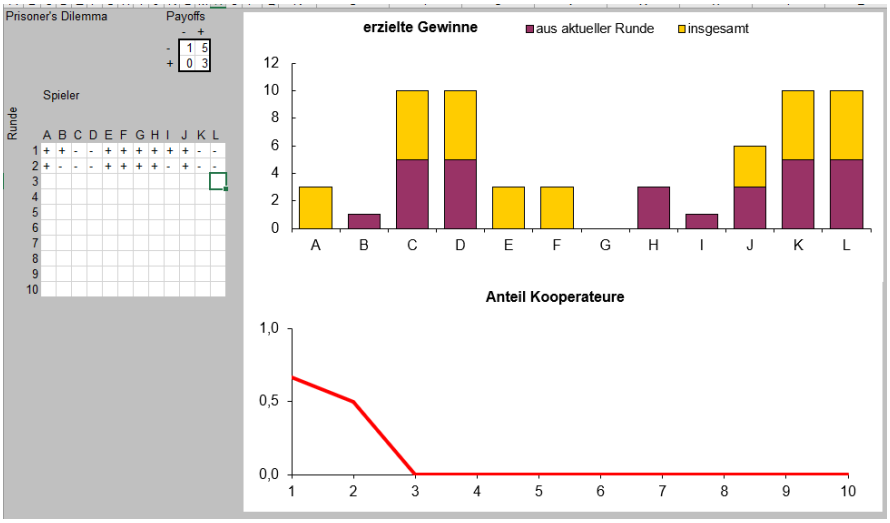


Abbildung 5: Konzept 1: Darstellung Exceldatei

- Konzept 2:
  Hierfür habe ich mir Gedanken gemacht, wie man die Daten auf eine andere Art und Weise darstellen kann, als in der Exceldatei. Die Bilder sollen hier nur als eine grobe Richtlinie dienen und entsprechen nicht der endgültigen Umsetzung.

  – Gesamtpunkteübersicht:
    Bei der Gesamtübersicht fand ich es tatsächlich schwierig, mir eine Alternative zu überlegen, die die Kriterien in einer übersichtlichen und leicht verständlichen Form erfüllt. Zum Vergleich hatte ich mir überlegt, dass man anstelle eines Säulendiagramms ein Balkendiagramm verwenden könnte, da dies einen Einfluss auf die Wahrnehmung der angezeigten Seite haben könnte. Man könnte dies so implementieren, dass die Spielernamen so angezeigt werden, dass sie anhand ihrer Punkte absteigend sortiert werden.

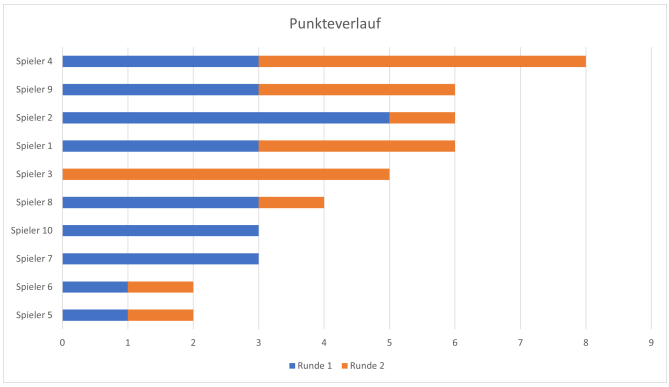


Abbildung 6: Konzept 2: Balkendiagramm

  – Kooperationsverlauf:
    Hierfür hatte ich mehrere Ideen. Dieses Diagramm wird bei der aktuellen Version des Spiels nur am Ende eines Spiels angezeigt. Ich hatte die Idee,

dass pro Runde ein Kuchendiagramm angezeigt wird, aus dem hervorgeht wie viele Spieler diese Runde kooperiert oder betrogen haben. Nach Abschluss des Spiels wäre trotzdem ein Liniendiagramm sichtbar um noch einmal eine Gesamtübersicht zu bekommen. Alternativ zu dem Liniendiagramm fand ich hier auch den Vorschlag von Jannis Hamann gut, den Kooperationsverlauf als Säulendiagramm darzustellen. Wobei ich hier eine einfarbige Variante wählen würde, die den Anteil an Kooperateuren pro Runde darstellt. Da aus meiner Sicht eine zweifarbige Darstellung an dieser Stelle eher für Verwirrung sorgt.

[a] 

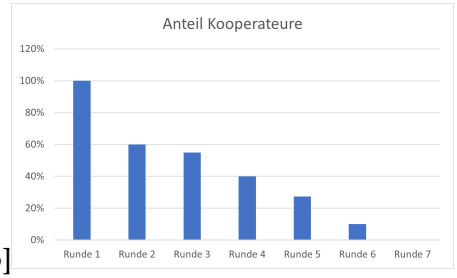
[b] 

Abbildung 7: zu Konzept 2: Kuchen und Säulendiagramm

Wichtig an dieser Stelle ist es zu erwähnen, dass für das endgültige Produkt eine Kombination aus beiden Konzepten sinnvoll sein könnte und es keinesfalls nur das einzig wahre Konzept gibt, welches richtig ist. Neben den Konzepten zur Visualisierung der Spieldaten soll auch noch eine Übersicht über die Gesamtstatistik aller gespielten Spiele der Sitzung mit der Punkteverteilung für jeden Spieler implementiert werden. Hierbei soll es möglich sein einen Wert einzutragen, der zum Beispiel angibt wie viele Smarties die Lehrkraft zur Verfügung hat. Anschließend wird für jeden Spieler berechnet, wie viele dieser aufgrund seiner erzielten Punkte erhält. Dies war bereits in der Exceldatei mit abgebildet und ich habe mich hierfür bei der Implementierung an der Darstellung orientiert. Wie man in Abbildung 8 erkennen kann werden in der Exceldatei eine Tabelle und ein Säulendiagramm verwendet. Ich habe mich jedoch dazu entschieden dies nur tabellarisch zu lösen, damit die Seite mit den Spielergebnissen nicht zu voll und unübersichtlich wird.

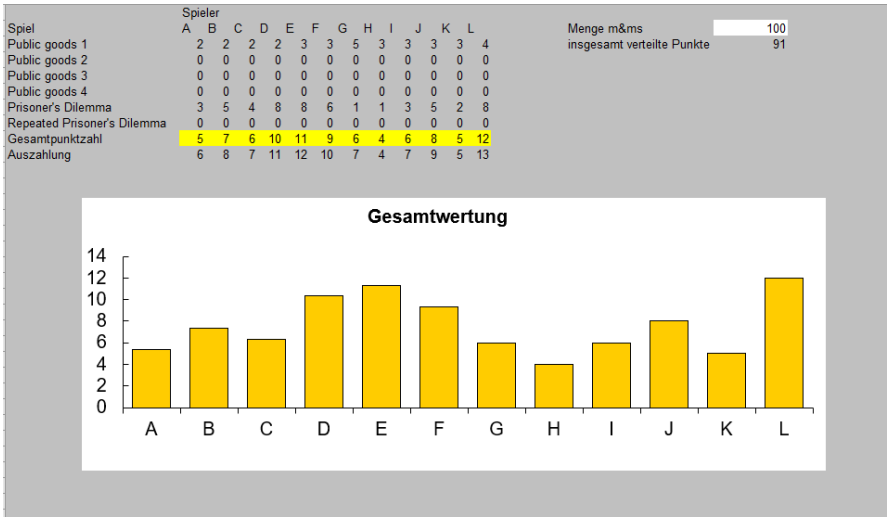


Abbildung 8: Darstellung Gestamtbewertung Exceldatei

3.1.2 Datenbank

Neben der Visualisierung der Spieldaten war es Dr. Nessler auch wichtig, dass die Daten nach Abschluss einer Sitzung zu einem beliebigen Zeitpunkt erneut aufrufbar sind. Bei dem derzeitigen Stand der Anwendung ist es nicht möglich nach Verlassen des Spiels erneut auf die Spielergebnisse zuzugreifen. Dies ist ungünstig, da es im Alltag vorkommen kann, dass es die Lehrkraft in einer Unterrichtseinheit zwar schafft ein Spiel zu Ende zu spielen, aber nicht mehr die Ergebnisse zu besprechen. Dies hat zur Folge, dass in der nächsten Unterrichtseinheit das Spiel erneut komplett gespielt werden muss, was viel Zeit kosten kann. Außerdem hat Dr. Nessler in einem Gespräch angemerkt, dass es für die Lehre auch sinnvoll sein könnte, die Spielergebnisse einer anderen Spielergruppe zu zeigen um Unterschiede im Kooperationsverhalten, zwischen den verschiedenen Gruppen, aufzuzeigen und zu analysieren.

Bevor die Datenbank implementiert werden konnte, musste erst mal die Entscheidung getroffen werden, welches Datenbankmanagementsystem verwendet werden soll. Ich habe mich hier für Postgresql[3] entschieden. Dies hat folgende Gründe: Postgresql ist eine opensource Software und es fallen somit keine Kosten an. Zudem ist es eine plattformunabhängige Software, was auch die Weiterentwicklung der Anwendung erleichtern kann. Außerdem sind in der Software schon Sicherheitsfunktionen mit enthalten, so dass sich zum Beispiel ein Benutzer einloggen muss, um auf die Datenbank zuzugreifen. Bei Abschluss der Arbeit ist dies noch nicht über die Webseite implementiert, sondern mit einem globalen Benutzer hardcodiert. Dies könnte jedoch in der Zukunft mit noch einer Arbeit mit umgesetzt werden.
Ein Nachteil an Postgresql ist jedoch, dass es eine Server-Client- Struktur aufweist und dies beim Aufsetzten der Datenbank komplexer ist. Auch kann es aufwändiger werden, wenn man im Nachhinein, die Struktur einer Tabelle anpassen möchte. Zudem ist Postgresql[11] im Vergleich zu anderen Systemen langsamer. Nichtsdestotrotz habe ich mich für Postgresql entschieden,da ich bereits im 4. Semester Erfahrungen im Umgang damit sammeln konnte, als wir im Modul Datenbanksysteme für ein Projekt eine Datenbank implementieren sollten. Auch die Sicherheitsfunktionen haben aus

meiner Sicht für dieses Datenbankmanagementsystem gesprochen.

Bei der Struktur für die Datenbank habe ich mich für eine Tabelle mit 3 Spalten entschieden. Diese 3 Spalten sind: id, playgroundID und playground. Hierbei ist die id der Platz, den der Eintrag in der Datenbank einnimmt. Die id dient auch dazu, um ermöglichen zwei „Playgrounds“ mit derselben PlaygroundID zu speichern ohne den jeweils anderen Eintrag zu überschreiben. Dies erläutere ich im Abschnitt Implementierung genauer. In die Spalte der PlaygroundID wird der generierte Spielcode der aktuellen Sitzung eingetragen. In die dritte Spalte "Playground"wird das gesamte Playgroundobjekt als Text eingefügt.

Neben dem Konzept für die Datenbank selbst, musste ich mir noch über die folgenden Aspekte Gedanken machen: Zu welchen Zeitpunkt sollen die Daten in der Datenbank gespeichert werden und an welcher Stelle im Frontend baue ich die Funktionalität ein, dass der Nutzer die Daten aus der Datenbank auszulesen kann. Für den ersten Punkt habe ich mich entschieden, dass immer wenn ein Spiel abgeschlossen wurde die gesamte Sitzung in der Datenbank eingetragen wird. Gibt es bereits einen Eintrag dieser Sitzung wird dieser aktualisiert. Um abzufangen, dass eine weitere Sitzung die anderen überschreibt, die die gleiche PlaygroundID haben, habe ich mir folgendes Konzept überlegt: Bevor ein Eintrag in die Datenbank gemacht wird, wird nach der PlaygroundID in der Datenbank gesucht. Ist diese noch nicht vorhanden, wird ein Eintrag gemacht. Ist jedoch bereits ein Eintrag vorhanden gibt es im Moment zwei Möglichkeiten. Entweder ist es die aktuelle Sitzung die aktualisiert werden soll oder es ist eine weitere Sitzung die dieselbe playgroundID aufweist.
Jede Sitzung hat einen Zeitstempel. Die Idee ist also nun das Datum von der aktuellen Sitzung wird mit den Datum der bereits eingetragenen Sitzungen verglichen. Stimmt das Datum überein, wird der Eintrag überschrieben. Gibt es keinen Eintrag der das gleiche Datum besitzt, so wird ein neuer Eintrag eingefügt. Jetzt kann man argumentieren, dass es möglich ist, dass genau an dem selben Tag zwei Sitzungen mit dem gleichen Spielcode erstellt werden. Dies ist jedoch aus folgendem Grund sehr unwahrscheinlich.

Wir haben einen Spielcode der aus 4 Ziffern besteht, wobei jede Zahl die Möglichkeit hat 0-9 anzunehmen. Wir können also sagen, dass wir mit Beachtung der Reihenfolge und mit Zurücklegen insgesamt n^k Möglichkeiten für den generierten Code haben. In unserem Fall bei 4 Ziffern und der Möglichkeit eine Zahl zischen 0-9 zu ziehen sind das :

$$p = 10^4 = 10000 \;\; \textit{Moeglichkeiten}$$

Wir können mithilfe der Formel von Laplace und dem Geburtstagsparadoxon die Kollisionswahrscheinlichkeit berechnen, d.h dass wir mindestes zweimal code x bekommen:

$$P = 1 - \frac{u}{m}$$

, wobei u = Anzahl der Ereignisse, bei denen das Ergebnis A eintritt und m = die Anzahl aller möglichen Ereignisse ist.

$$\frac{u}{m} = \frac{10000!}{10000^n \cdot (10000 - n)!}$$

, wobei alle n die Anzahl an Codes angibt die sich unterscheiden. Nachdem ich die Formel bei Wolframalpha eingegeben habe, wurde mir in einem Plot angezeigt, dass

die Wahrscheinlichkeit bei circa einem Wert von 120 bei 50% liegt. Ich habe dann für n=120,119 und 118 eingeben. Für n = 119 ergibt sich dann folgendes:

$$P = 1 - \frac{10000!}{10000^{1}19 \cdot (10000 - n)!} = 0,50583$$

Das heißt ab 119 Codes ist die Wahrscheinlichkeit, dass wir denselben Code bekommen können schon 50%. Das wir also am selben Tag den gleichen Spielcode herausbekommen ist sehr unwahrscheinlich und kann vernachlässigt werden. Ab einem Wert von $n \geq 1673$ ist die Wahrscheinlichkeit gleich 1, dass wir eine Codedoppelung auftritt. Wenn wir also davon ausgehen, dass an einer Schule in einem Jahrgang mit ca vier Klassen die Spieltheorie behandelt wird. Und von jeder dieser Klasse in einer Unterrichtseinheit eine Sitzung startet, so dauert es für diese eine Schule im Schnitt 29 Jahre bis wir mit einer Wahrscheinlichkeit von 50% einen doppelten Code generieren. Selbst wenn die Anwendung an mehreren Schulen eingesetzt wird, und sich dadurch der Zeitraum verkürzt, halte ich es noch für vertretbar, wenn es passieren kann, dass Spielstände überschrieben werden.

Den zweiten Punkt habe ich so gelöst, dass ich beschlossen habe eine weitere Seite einzubauen, bei der man einen Spielcode eingibt und dann die Daten automatisch in die Diagramme geladen werden. Gibt es mehr als ein Spiel, soll es die Funktion geben zwischen den verschiedenen Spielen auszuwählen. Dies soll über eine Dropdownmenü möglich sein, in dem alle Spiele der Sitzung aufgelistet werden.

3.2 Implementierung

3.2.1 Allgemeines Vorgehen

Ich möchte vorwegnehmen, dass ich zurzeit des Softwareprojekts im Backendteam tätig war und kein Vorwissen in der Frontendentwicklung und dem Umgang mit Docker aufweise. Zu Beginn des Bearbeitungszeitraums der Bachelorarbeit bestand meine Priorität darin, die Anwendung auf meinem Rechner zum laufen zu bringen, um mir einen Überblick über die Funktionsweise und den aktuellen Stand zu verschaffen. Hierbei stieß ich jedoch bereits auf erste Probleme. Aus einem mir unerfindlichen Grund, ließ sich für den Frontendcode alles installieren, wenn ich jedoch versucht habe „yarn“ auszuführen stieß ich auf immer neue Probleme mit den installierten Packages und Meldungen, dass diese nicht verfügbar seinen, obwohl ich sie installiert hatte. Ich habe nach ein paar Tagen debuggen versucht, das Programm mit Docker zu starten, was leider auch nicht funktionierte. Jedes mal wenn ich versuchte, den Container mit docker run zu starten, blieb der ohne eine Fehlermeldung in einer Endlosschleife stecken. Nachdem ich mehrfach mehrere Stunden gewartet hatte, habe ich es aufgegeben, dass Programm mit Docker auszuführen. Im Nachhinein betrachtet gibt es zwei Punkte an denen es gelegen haben könnte wieso docker run stecken blieb. Der erste Punkt ist, dass ich Docker nicht korrekt aufgesetzt hatte bzw Fehler in der Handhabung für das Problem verantwortlich waren. Der zweite Punkt ist, dass ich nginx, ein Open-Source-Webserver der unter anderem als Reverse-Proxy, HTTP-Cache und Load-Balancer verwendet wird, nicht korrekt aufgesetzt hatte und Docker sich somit nicht mit nginx verbinden konnte. Dies war auch später noch einmal ein

Problem. Nachdem nun diese Versuche alle fehlschlugen empfahl mir meine betreuende Person entweder eine Virtual Machine oder WSL [6], ein Windows-Subsystem für Linux eine mit Linuxdistribution aufzusetzten.

Ich entschied mich zweiteres zu nutzen, da ich bei der Docker installation bereits WLS mit Ubuntu installiert hatte. Hier funktionierte die Installation der Frontend- und Backendarchiteckturen schnell und unkompliziert. Nachdem ich die Anwendung gestartet hatte musste ich nur leider feststellen, dass ich kein Spiel starten konnte, da die Verbindung mit nginx nicht aufgebaut werden konnte. Dies war auf einen Fehler in der nginx-config Datei zurückzuführen. Nachdem mir meine betreuende Person hier helfen konnte die Konfiguration richtig umzusetzen konnte ich mich nach 4 Wochen endlich mit dem Einarbeiten und Implementieren der Konzepte widmen. Ein Grund dafür weshalb das Aufsetzen der Architektur so viel Zeit in Anspruch nahm, bestand darin, dass zwischendurch Weihnachten und somit Akademische Ferien an der FU waren, und es ein paar Tage dauerte bis es möglich war, dass meine betreuende Person und ich einen Termin hatten, um meine Probleme zu besprechen. Ich fing also zeitgleich an, mich mit dem Thema Diagrammkompetenz und Visualisierung von Daten auseinander setzten, um nicht untätig zu sein. Glücklicherweise konnte ich den aktuellen Stand der implementieren Diagramme unter `evo.imp.fu-berlin.de/` einsehen. Zusätzlich hatte ich bereits vor Anmeldung der Bachelorarbeit bereits ein Gespräch mit Dr. Nessel über seine Wünsche bezüglich der Visualisierung der Daten gesprochen. So konnte ich in der Zeit bis es mir möglich war ein Treffen mit meiner betreuenden Person zu organisieren, damit beginnen die Konzepte zu entwickeln. Nachdem nun meine Probleme gelöst waren und die Anwendung lief, konnte ich damit anfangen mich mit dem Frontendcode vertraut zu machen.

Im folgenden gebe ich einen kurzen Überblick darüber welche Technologien hier verwendet werden. Die erste Technolgie ist React[4]. Hierbei handelt es sich um eine Javascriptbibliothek zum Erstellen von Benutzeroberflächen. Hierbei werden gekapselte Komponenten erstellt, die ihren eigenen Zustand verwalten und anschließend zu einer Benutzeroberfläche zusammengefügt werden. Die nächste Technologie ist Redux[1] und wird auf der Webseite beschrieben als „A Predictable State Container for JS Apps“. Hiermit werden der Zustand der Anwendung zentralisiert verwaltet. Typerscript[5] erweitert Javascript mit zusätzlicher Syntax wie zum Beispiel static type checking. Durch einen eigenen Kompiler der in den Editor integriert wird, werden typisierte Syntaxfehler schon während des Programmierunsprozesses gefunden und es können im weiteren Verlauf Bugs und Fehler minimiert werden. Typescript wird während des Kompilierens in Javascript umgewandelt.

3.2.2 Visuelle Daten

Ich fing erst einmal an mich Mithilfe von Youtubevideos und Onlinedocumentationen zu den genutzten Technologien, damit vertraut zu machen wie die Struktur einer Reactapp genau aufgebaut ist und funktioniert. Anschließend machte ich mit dem Aufbau der vorliegenden Anwendung vertraut. Insbesondere an welcher Stelle, welche Komponenten geladen werden und wo die Diagramme erzeugt werden.

Hierbei habe ich festgestellt, dass die Logik für das Säulendiagramm noch im Code vorhanden war, da dies Ursprünglich für die Visualisierung der Daten verwen-

det wurde. Glücklicherweise war auch schon die Funktion eingebaut, dass in jeder neuen Runde, die Punkte in einer anderen Farbe dargestellt werden. Ich musste hierfür also nur den Ort finden, an dem ich den Funktionsaufruf einbauen musste. Das hier verwendete Reactmodul ist „Recharts“ mit dem Diagrammtypen „StackedBarChart“. Das implementierte Diagramm ist ein zweifarbiges Säulendiagramm, welches in jeder neuen Runde die neu dazugekommenen Punkte farblich anders darstellt. Hierbei werden die Farben Gelb, für die neuste Runde, und Lila, für alle anderen Runden verwendet. Dieses Farbschema wurde ausgewählt, da es auch für Personen mit Farbblindheit einen Kontrast zwischen den Runden erzeugt. Dies wurde bereits während des Softwareprojekts so festgelegt und war auch das Farbschema der gesamten Anwendung, bevor diese in der Arbeit von Jannis Hamann überarbeitet wurde. Ich beschloss das Schema beizubehalten, da es aus meiner Sicht sinnvoll für die Lehre ist. An der Y-Achse des Diagramms wird die Punkteskala angezeigt. Diese passt sich dynamisch an der Maximalpunktzahl der Spieler an. Auf der X-Achse sind die Spielernamen abgebildet. An der rechten Seite des Diagramms ist eine Legende zu sehen, die zum einen die Anzahl der gespielten Runden anzeigt und zum anderen die Farben der verschiedenen Runden im Diagramm. Es ist zusätzlich implementiert worden, dass sich beim Rüberfahren mit der Maus ein Popup öffnet, welches noch einmal genau anzeigt wie viele Punkte der ausgewählte Spieler in welcher Runde erhalten hat. Das Diagramm als Ganzes hat die Überschrift „Punkteübersicht“. In den folgenden Abbildungen ist die endgültig implementierte Version des Diagramms abgebildet.

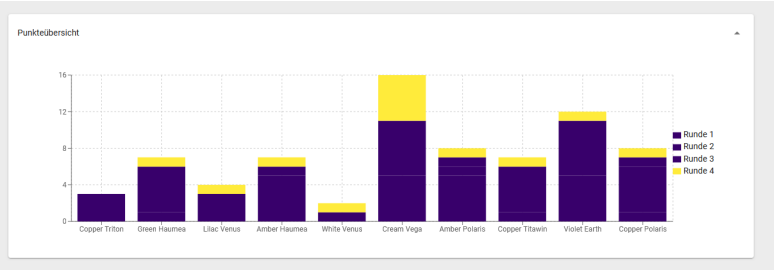


Abbildung 9: Diagramm: Punkteübersicht

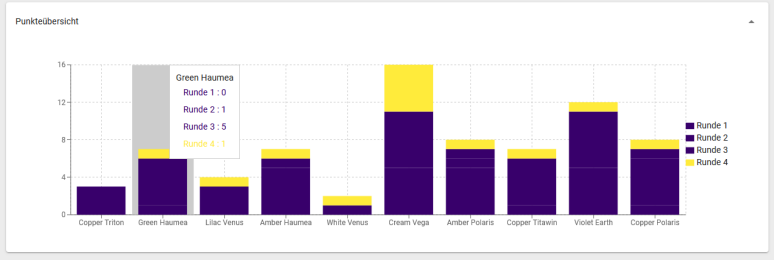


Abbildung 10: Diagramm: Punkteübersicht mit Popup

Anschließend habe ich mich der Funktion gewidmet mit der man einzelne Runden im Säulendiagramm hervorheben kann. Nach einiger Recherche, konnte ich nichts dazu finden, ob das mit dem verwendeten Reactmodul „Recharts“ überhaupt möglich

ist. Also entschied ich mich wie in der Exceldatei eine Tabelle zu bauen, in der für jeden Spieler pro Runde die Punkte eingetragen werden, um dann in der Tabelle die Zeilen auswählbar zu machen.
Hierfür sollten die Spielernamen generisch als Spaltenüberschrift eingefügt werden. Diese Tabelle zu bauen dauerte jedoch wesentlich länger als ich erwartet hatte, da ich einige Zeit Verständnisprobleme hatte, wann genau eine Komponente neu geladen wird und wie ich den State einer Variable anpassen konnte. An dieser Stelle habe ich einen Freund um Hilfe Gebeten mir einmal zu erklären wie ich bei meinem konkreten Problem weiterkommen könnte. Schlussendlich hatte ich eine Variable beim Erstellen der Tabellenüberschriften falsch benannt.

Nachdem ich die Tabelle erstellt hatte, fehlte nun noch die Funktion, einzelnen Zeilen auszuwählen, hierbei die Daten auszulesen und an das Diagramm zu übertragen. Hier machte sich jedoch mein Verständnisproblem bezüglich des State bemerkbar. Zusätzlich hatte ich hier ein strukturelles Problem. Ich hatte für die Tabelle eine neue Datei im Reactverzeichnis erstellt und versuchte eine lokalen Statevariable an eine Komponente in einem anderen Verzeichnis zu übergeben, ohne jene Varibale zu exportieren. Ich verschob also den Code in die selbe Datei wie die anderen Diagramme, in der Hoffnung nun mit der lokalen Statevarbable arbeiten zu können. Dies löste jedoch mein Problem nicht, da die Komponenten so aufgebaut sind, dass in einer sogenannten Parentkomponente, die einzelnen Diagramme aufgerufen werden, indem z.B. die globale State Variable Rundennummer übergeben wird und sich dann mit Änderungen dieser Komponente, das Diagramm neu lädt. In der Childkomponente, in der sich der Code für alle Diagramme befindet, wird dann mithilfe einer switch-case Anweisung nur das berechnet, was für das angefordert Diagramm nötig ist um Rechenzeit zu sparen.

Listing 1: Switch-case Anweisung aus Diagram.tsx

```
    // Only prep the needed Data ————————
    switch (props.diagData.type) {
            case ChartType.AdminCooperationPercentage:
                getAdminCooperationPercentageData()
                break
            case ChartType.AdminRoundResult:
                getRoundResultData()
                break
            case ChartType.PlayerRoundResult:
                getPlayerRoundResultData()
                break
            case ChartType.PointsTable:
                getColumns()
                getData()
                getRoundResultData()
                break
        }
```

Aus diesem Grund wurde das Säulendiagramm nicht aktualisiert, wenn ich in dem Aufruf der Tabelle versuchte mithilfe einer lokalen State Variable dieses zu verändern. Die Lösung des Problems war also die ausgelesen Daten mithilfe eines glo-

balen State zu übergeben. Hierfür mussten zwei Interfaces erstellt werden, die die Statevariable deklarieren und initliaisieren. Zusätzlich musste eine Funktion deklariert werden, die die Stateänderungen übergibt und die Statevaribale musste in die Reduxstruktur mit eingepflegt werden. Dies herauszufinden hat mich jedoch sehr viel Zeit gekostet. Die Tabelle selbst ist einfach aufgebaut. Auf der linken Seite ist ein Kasten der angeklickt werden kann und die dazugehörige Zeile auswählt. Als Spaltenüberschriften dienen die Spielernamen und die Bezeichnung Runde/Punkte. In den einzelnen Zeilen wird als erstes die Rundennummer und anschließend die Punkte der einzelnen Spieler angegeben. Die Tabelle als ganzes hat die Überschrift „Spielerergebnisse“. In den folgenden Abbildungen ist die endgültig implementierte Version der Tabelle abgebildet.

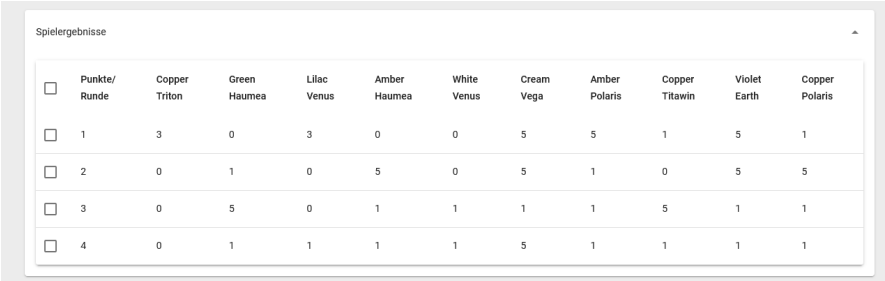


Abbildung 11: Tabelle: Spielerergebnisse

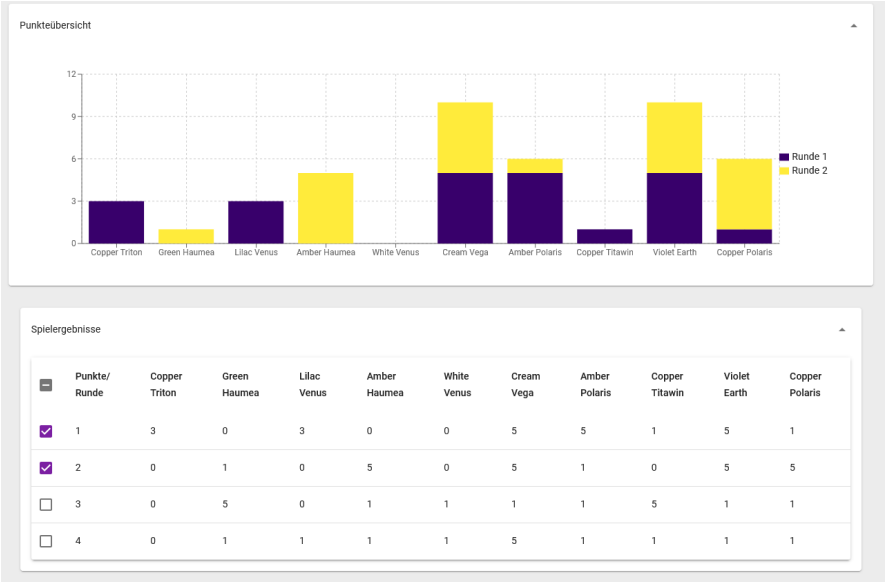


Abbildung 12: Tabelle: Spielerergebnisse mit Auswahl

Für das Liniendiagramm musste ich einfach nur eine Zeile Code auskommentieren, um die Hintergrundfarbe zu entfernen, so dass nur noch eine einfache Linie dargestellt wird. Ich habe jedoch zusätzlich auf Anraten von Professor Prechelt eingebaut, dass mehr als nur die gespielten Runden im Diagramm angezeigt werden, da das Liniendiagramm sonst sehr abgehackt wirkt, wenn das Kooperationsverhalten der zuletzt gespielten Runde nicht 100 % oder 0 % beträgt. Zusätzlich habe ich an

der Y-Achse des Diagramms eine Beschriftung eingefügt, da dies von Dr. Nessler so gewünscht wurde. Wenn man mit der Maus bei den einzelnen Runden über die Linie fährt, erscheint ein Punkt in dessen Beschriftung der genaue prozentuale Wert dargestellt wird. In der folgenden Abbildung ist die endgültig implementierte Version des Liniendiagramms abgebildet.

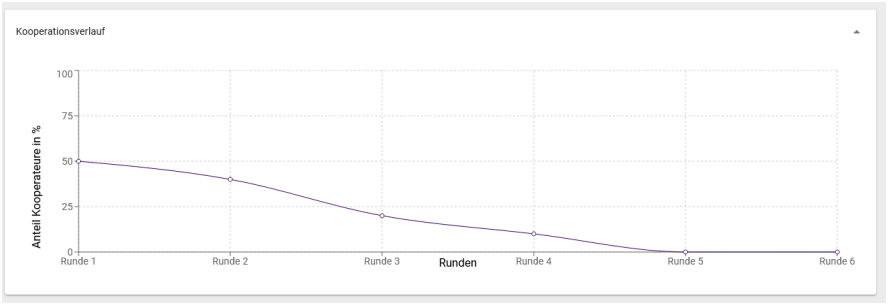


Abbildung 13: Diagramm: Kooperationsverhalten

Für die Gesamtstatistik habe ich wieder eine Tabelle gebaut. Ich habe mir Gedanken darüber gemacht, wie man die Eingabe der Belohnung am besten gestalten könnte und hielt es für die beste Lösung dies in die Tabelle selbst mit einzubauen. Zuerst habe ich die Grundstruktur aufgebaut, die ähnlich zu der Tabelle der Spielerergebnisse ist. Ich habe jedoch die erste Spalte in Spielmodi umbenannt. Zusätzlich habe ich eine Spalte am Ende der Tabelle hinzugefügt, in welcher der Wert der verfügbaren Belohnung steht. In die erste Spalte werden als Zeilentitel zum einen die verschiedenen Spielmodi angegeben, als auch einmal die Beschreibung, Gesamtpunkte und Auszahlung. Die Werte unter den Spielernamen stellen zum einen die Gesamtpunkte dar, die für alle Spiele des jeweiligen Spielmodus erzielt wurden, so wie die Gesamtpunktezahl über alle Spiele und die Menge an Belohnung die auszuzahlen ist. Um die jeweiligen Punkte zu berechnen, wird über das Playerarray iteriert. Hierbei hat jeder Spieler im Playerarray ein PlaygroundPointsArray in dem seine Punkte für jede Runde, jedes gespielten Spiels mit Angabe der GameID gespeichert werden. Anhand der GameID wird dann der Spielmodus herausgefunden. In einem temporären JSONObject werden nun für jeden Spielmodus die Punkte des aktuellen Spielers für jedes Spiel desselben Spielmodus aufaddiert. Zusätzlich wird aus dem Playerarray die Gesamtpunktzahl des Spieler ausgelesen, da diese bereits separat gespeichert wird.
Anschließend wird anhand der Gesamtpunktezahl eines jeden Spielers seine auszuzahlende Belohnung berechnet. Um zu berechnen, wie viele Smarties jedem Spieler ausgezahlt werden, wird zuerst die maximale Anzahl an Belohnungen durch die Gesamtzahl aller erzielten Punkte geteilt. Anschließend wird dieses Ergebnis mit der Gesamtpunktzahl jedes Spielers multipliziert. Ist die erste Nachkommastelle kleiner 4 so wird der Wert abgerundet, sonst wird aufgerundet. Dies ist notwendig, da nur ganze Smarties vergeben werden können. Diese Daten werden dann pro Spieler den Tabellendaten hinzugefügt. In der Tabelle selbst kann man in der letzten Zeile den letzten Wert bearbeiten. Hierfür bietet das verwendete Reactmodul „Material-Table“ bereits die gewünschte Funktion. Damit jedoch die Tabelle neu berechnet wird musste wieder eine gloable Statevariable erstellt werden. Ich hatte es zuerst mit einem lokalen State versucht, doch aus einem mir unerfindlichen Grund aktualisierte sich die

Komponente nicht. Nachdem ich die globale Statevariable implementiert hatte, funktioniert es nun ohne Probleme. Die Belohung hat wie in der Exceldatei selbst einen Defaultwert von 100. In der folgenden Abbildung ist die endgültig implementierte Version der Tabelle abgebildet.


Gesamtstatistik

Actions	Spielmodus	Copper Triton	Green Haumea	Lilac Venus	Amber Haumea	White Venus	Cream Vega	Amber Polaris	Copper Titawin	Violet Earth	Copper Polaris	Zu verteilende Belohnung
	Einfaches Gefangenendilemma	3	7	4	7	2	16	8	7	12	8	
	Iteratives Gefangenendilemma	0	0	0	0	0	0	0	0	0	0	
	Öffentliche Güter	0	0	0	0	0	0	0	0	0	0	
	Gesamtpunktzahl	3	7	4	7	2	16	8	7	12	8	
	Auszahlung	4	9	5	9	3	22	11	9	16	11	100

Abbildung 14: Tabelle: Gesamtstatistik

Alle Diagramme sind ein- und ausklappbar um die Seitenansicht übersichtlicher zu machen.

Nachdem ich beim implementieren der Konzepte starke Schwierigkeiten hatte, habe ich in Rücksprache mit Dr. Nessler, meiner betreuenden Person und Prof. Prechelt, den weiteren Plan für die Bachelorarbeit abgewandelt. Wir haben uns darauf geeinigt, dass das zweite Konzept nicht mehr zu implementierten. Zudem sollte auch die Bewertung durch Schülergruppen wegfallen. Stattdessen werde ich weiter unten die umgesetzten Visualisierungen auf verschiedene Kriterien untersuchen. Ich fokussierte mich bei den Diagrammen stattdessen noch auf das Implementieren der Gesamtstatistik. Außerdem war es Dr. Nessler sehr wichtig, dass es eine Möglichkeit zum Speichern der Spieldaten gibt, weshalb ich mich im weiteren Verlauf der Arbeit darauf fokussiert habe eine Datenbank zu implementieren. Das Vorgehen hierfür beschreibe ich im nächsten Abschnitt.

3.2.3 Datenbank

Für das implementieren der Datenbank wurde mithilfe von Postgesql ein lokaler Server aufgesetzt. Anschließend wurde eine Datenbank „evoGames “ mit hardcodierten Nutzerdaten erstellt. In dieser Datenbank selbst wurde die oben beschriebene Tabelle mit dem Namen playgrounds erstellt. Hierbei haben Id und playgroundID den Typ integer, da hier Nummern eingetragen werden. Die Spalte playground hat den Typ text, da der gesamte Playground gespeichert werden soll. Hierfür wird das JSON Object vor dem Einfügen zu einem String und beim Auslesen wieder zu einem Object konvertiert.
Um die Datenbank mit Python zu verbinden wird das Modul psycopg2 verwendet. Ich informierte mich jedoch vorher welches Modul sich am besten dafür eignet.[10] Kriterien die mir hier wichtig waren, sind, dass das Modul aktiv gepflegt wird und die verwendete Python und Postgresql Versionen unterstützt. Hierfür kamen mehrere Module in Frage, die alle sehr ähnlich waren. Schlussendlich entschied ich psycopg2 zu verwenden, da es von allen am aktuellsten war und damit als effizient und sicher beschrieben worden ist.

Wenn ein Spiel beendet wird, wird automatisch überprüft, ob die Sitzung bereits in der Datenbank existiert. Ist dies der Fall, wird der Eintrag mit der aktuellen Sitzung überschrieben. Sonst wird diese als neuer Eintrag eingefügt. An dieser Stelle wäre es sinnvoll mit einzubauen, dass unterschiedliche Sitzungen mit der selben PlaygroundID nicht überschrieben werden. Ich habe diese Funktion derzeit noch nicht mit implementiert. Dies hat folgenden Grund: Derzeit ist unser JSON Object welches wir zwischen dem Frontend und Backend hin- und her schicken so aufgebaut, dass wir alle wichtigen Strukturen der Sitzung als Key Value Paar speichern. Wobei der Key die PlaygroundID ist und das Value alle anderen Strukturen.
Zusammen mit der Datenbank soll umgesetzt werden, dass eine gespeicherte Sitzung auch wieder ausgelesen und dargestellt werden kann. Jetzt ist aber das Problem, dass man die Struktur des Playgrounds anpassen müsste um mehrere Sitzungen mit der selben ID in das JSON Object einzufügen und mir hierfür leider die Zeit fehlt. Man kann jetzt natürlich argumentieren, dass ich trotzdem schon implementieren könnte, das doppelte Sitzungen nicht überschrieben werden. Dies hätte jedoch mit der aktuellen Struktur zur Folge, dass es zufällig wäre, welche der Sitzungen ans Frontend zurückgegeben werden, wenn man die Daten ausliest. Das halte ich für die Lehre nicht für hilfreich. Zusätzlich habe ich in der Konzeptentwicklung bereits gezeigt, dass die Wahrscheinlichkeit, dass eine Dopplung des Spielcodes auftritt sehr gering ist. Aus diesem Grund kann man davon ausgehen, dass im Verlauf eines Schuljahres diese Dopplung nicht auftritt und somit die Sitzung zwischen Spielen und Besprechung nicht überschrieben wird. Zusätzlich könnte man hier auch noch die Funktion einbauen, die Diagramme herunterzuladen.
Um die Ergebnisse einer Sitzung wieder in die Anwendung reinzuladen, habe ich im Frontend erst mal eine neue Seite gebaut. Diese erreicht man direkt über das Hauptmenu der Anwendung unter „Spieldaten laden“. Für diese Seite habe ich bereits implementierte Funktionen wiederverwendet. Man kann über die Eingabe des Spielcodes die Daten wieder rein laden und die Diagramme, die bereits bei der Ergebnisansicht eines Spiels angezeigt werden, werden hier mit dem Ergebnis eines der Spiele gefüllt.
Man kann über ein Dropdownmenü zwischen den verschiedenen Spielen der Sitzung wechseln, wenn diese vorhanden sind. Im Menu selbst werden die Spiele mit dem Spielmodus und einem Zeitstempel angegeben. Bei Eingabe eines gültigen Spielcodes wird über SocketIo eine Anfrage an das Backend geschickt. Hier wird über die Funktion loadPlay eine Anfrage an die Datenbank gesendet und die Sitzung wird aus der Datenbank geladen. Die Sitzung wird dann zurück an das Frontend geschickt. Dort werden die Daten eines der Spiele in die globalen State Variablen wie z.B. gameID, gameType, playgroundID usw. geladen, damit Diagramme mit Daten gefüllt werden. Wird im Dropdownmenü ein anderes Spiel ausgewählt, so werden die notwentigen State Varibalen erneut aktualisiert.
Im Titel der Seite wird zusätzlich der Spielcode sowie der Spielmodus und Zeitstempel des aktuell ausgewählten Spiels angezeigt. Die nachfolgenden Abbildungen zeigen den endgültigen implementierten Stand der oben beschriebenen Komponten.

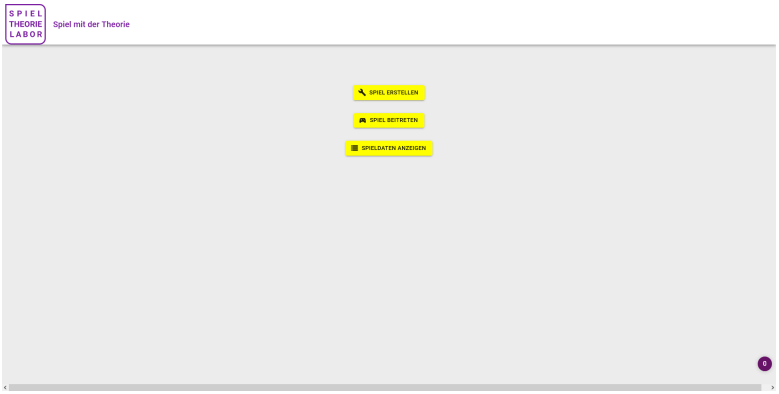


Abbildung 15: Neues Hauptmenu

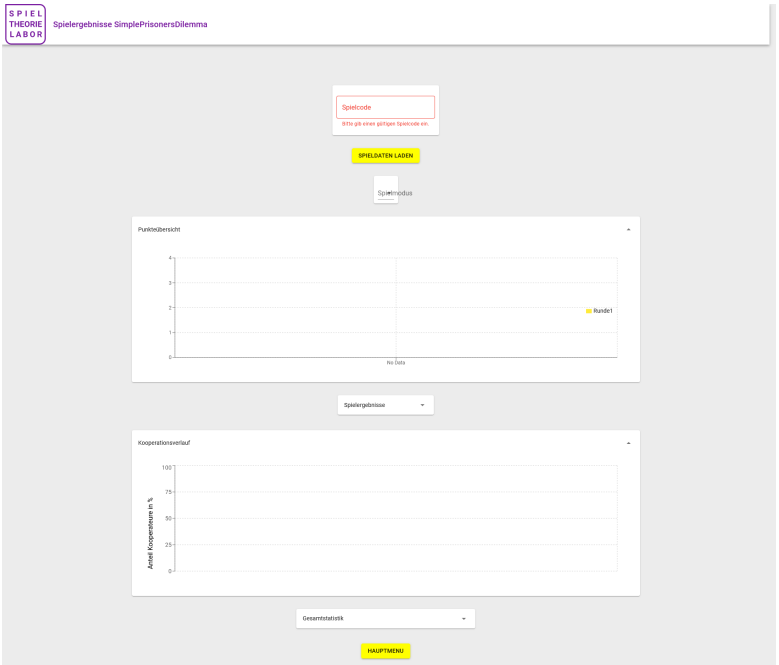


Abbildung 16: Übersicht: Spieldaten laden im leeren Zustand

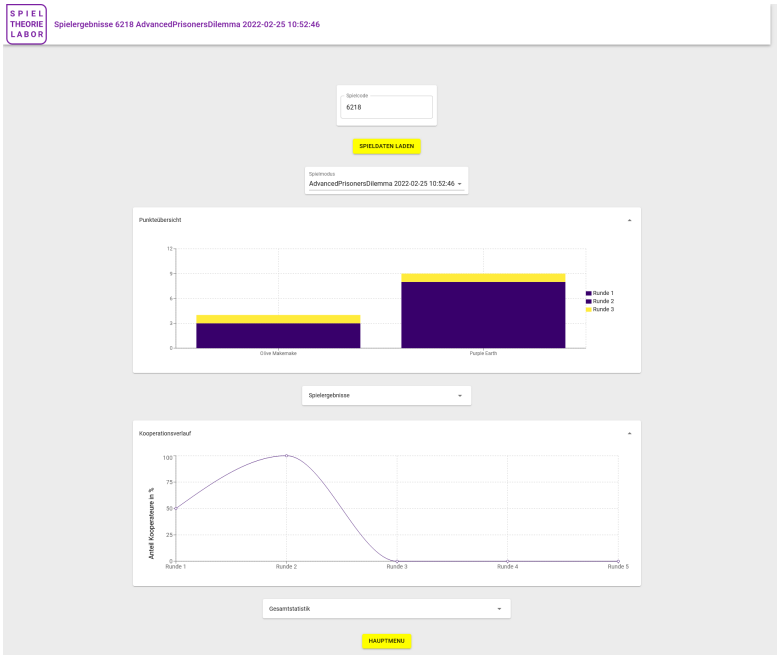


Abbildung 17: Übersicht: Spieldaten laden im geladenen Zustand

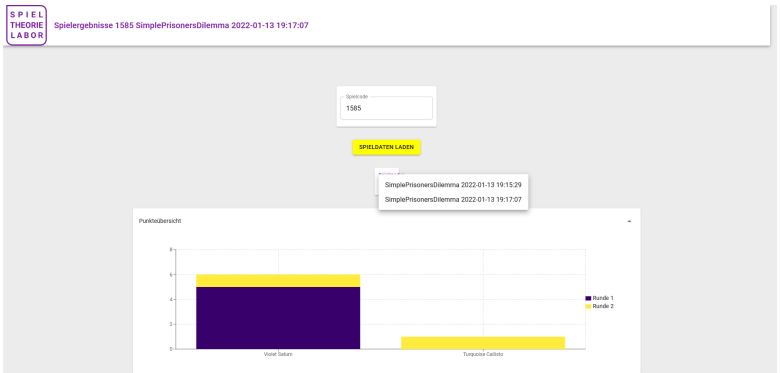


Abbildung 18: Dropdownmenü

4 Diskussion

Im folgenden Abschnitt diskutiere ich, ob die in dieser Arbeit umgesetzte Visualisierung der Daten komplementär zur Lehre eingesetzt werden kann. Zudem werde ich darauf eingehen, ob die dargestellten Diagramme und Tabellen so implementiert wurden, dass sie effektiv und nach Design-Prinzipen für Datenvisualisierung umgesetzt sind. Diese habe ich im Kapitel 2.3 Analyse ausführlich erläutert. Ich werde sie hier noch einmal Kurz zusammenfassen.
Die erste Richtlinie dich ich beschrieben hatte besagte, dass ein Diagramm einfach gehalten werden soll und zu viel „Visual Noise“ vermieden werden soll[9]. In der zweiten Richtlinie wurde genannt, dass eine kurze und präzise Beschriftung der Daten von Vorteil ist.[9][2] Die dritte Richtlinie die ich beschrieben habe machte darauf aufmerksam, dass eine Achsenbeschriftung von großer Wichtigkeit ist. Was auch deshalb wichtig ist, da ich in der Analyse beschrieben habe, dass ein Betrachter häufig zuerst auf die Beschriftung von Diagrammen guckt, um sich einen Überblick zu verschaffen bevor, die Daten selbst abgelesen werden.[7] Die vierte Richtlinie die ich in der Analyse benannt habe, besagt, dass die Farbauswahl eine wichtige Rolle spielt und so einfach wie möglich gehalten werden soll. Zusätzlich sollten Farben so gewählt werden, dass sie sinnvoll sind und eventuell auch im Zusammenhang mit den Daten selbst stehen.

Betrachten wir das erste Diagramm. Es wird zum einen auf der Seite der Punkteübersicht für jede Runde, als auch am Ende des Spiels angezeigt. Es ist ein Säulendiagramm, welches im Kapitel „Implementierung“ unter dem Titel „Abbildung 9: Diagramm: Punkteübersicht“9 und „Abbildung 10: Diagramm: Punkteübersicht mit Popup“10 abgebildet ist. Das Diagramm erfüllt die Kriterien, die Dr. Nessler aufgestellt hat. Die Punkte die in einer neuen Runde hinzukommen, werden in einer anderen Farbe dargestellt, als die der vorherigen Runden. Überprüfen wir nun, welche der in der Analyse genannten Richtlinien, das Säulendiagramm erfüllt.

Das Diagramm ist simpel gehalten. Die x-Achse ist eindeutig mit den Spielernamen benannt und auf der rechten Seite ist eine einfache Legende abgebildet, welche die Farben den dazugehörigen Punkten zuordnet. Auf der Y-Achse, sind die Punkte in einer sinnvollen Abstufung angegeben. Hier muss jedoch gesagt werden, dass dies von React selbst eingeteilt wird. Hierfür wird im Code angegeben, dass die Y-Achse dynamisch an die eingetragenen Werte angepasst wird. Dies ist in diesem Fall auch sinnvoll, da man bei der Erstellung des Diagramms nicht weiß, was die maximale Punktzahl ist. Versetzen wir uns nun in den Betrachter, der sich im Prozess der Identifizierung befindet. Die Beschriftung ist nicht optimal, da die Y-Achse keine Benennung aufweist. Was das Diagramm darstellt kann für den Betrachter also erst aus dem Konzept erschlossen werden, wenn die Daten abgelesen werden und man hier nur das Diagramm selbst betrachtet. Es ist jedoch auf der Abbildung zu sehen, dass das Diagramm selbst einen Titel hat, welcher Informationen über den Inhalt liefern kann und aus meiner Sicht in diesem Fall auch aussagekräftig genug ist.
Wenn der Nutzer sich jetzt aber weiter mit dem Diagramm auseinander setzt und mit der Maus über eine der Säulen fährt, sieht dieser einen kleinen Kasten, in dem die Daten der Säule, Spielername und erzielte Punkte pro Runde noch einmal hervorgehoben werden. Dies ist nach den Richtlinien empfohlen, damit das Diagramm nicht

zu unübersichtlich wird. In diesem Fall ist es zusätzlich hilfreich, um noch einmal die genauen Punkte eines Spielers zeigen zu können.
Kommen wir nun zu den verwendeten Farben. Hier wird jeweils nur für die neuste Runde Gelb verwendet, sonst für alle anderen Runden einheitlich Lila. Man hätte hier auch für jede Runde eine andere Farbe verwenden können, zum Beispiel in Abstufungen aus Lilatönen, um direkt den unterschied zwischen den einzelnen Runden zu zeigen. Dies würde aus meiner Sicht jedoch dem eigentlichen Zweck der verschiedenen Farben widersprechen, das veränderte Kooperationsverhalten in der neusten Runde, welches man aus den Punkten ablesen kann, und ist aus diesem Grund hier kontraproduktiv. Bei zu vielen Farben könnte es schwieriger und unübersichtlicher werden direkt die Unterschiede zur Vorrunde hervorzuheben, da zum einem nicht jeder Spieler in jeder Runde Punkte bekommt und man, je mehr Runden man spielt, umso mehr Farben benötigt. Deshalb wird aus meiner Sicht hier die Richtlinie für die Visualisierung der Daten in Farbe erfüllt. Das einzige was hier noch verbessert werden könnte, ist das die Schrift im Kästen in einem dunkleren Gelbton oder einer anderen Farbe dargestellt wird, da der zurzeit gewählte Farbton, für die neueste Runde in schriftlicher Form, schwer leserlich ist. Die Farbwahl an sich ist eine Designentscheidung aus dem Softwareprojekt selbst gewesen, die ich an dieser Stelle übernommen habe. Lila und Gelb wurden damals ausgewählt, damit Schüler*Innen die an einer Farbblindheit leiden trotzdem die Unterschiede erkennen können. Zusammenfassend kann man für dieses Diagramm also sagen, dass es die in der Analyse besprochenen Richtlinien und die Anforderungen die Dr. Nessler aufgestellt hat, erfüllt. Somit erfüllt aus meiner Sicht seinen Zweck und kann die Lehrkraft in der Lehre unterstützten.

Als nächsten betrachten wir die Tabelle, die im Kapitel „Implementierung“ unter dem Titel „Abbildung 11: Tabelle: Spielergebnisse“11 abgebildet ist. Hier sind als Spaltentitel zum einen Punkte/Runde, sowie die Spielernamen angegeben. In den einzelnen Zeilen ist in der vordersten Spalte die Runde angegeben, und unter den Spielernamen die jeweils dazugehörige Punktzahl. Auf der linken Seite ist ein Kästchen, welches angeklickt werden kann. Wenn dies angewählt wird, aktualisiert sich das Säulendiagramm von oben und zeigt nur noch die Punkte aus der ausgewählten Runde an. Dies kann man in „Abbildung 12: Tabelle: Spielergebnisse mit Auswahl “12 sehen.
Die Tabelle ist einfach gehalten und es ist nichts farblich hervorgehoben. Dies erfüllt somit die Richtlinie, die Tabelle simpel zu gestalten. Wenn man hier wieder als betrachtende Person herangeht die im Prozess der Identifizierung ist stellt man folgendes fest: In der ersten Spalte kann man aus der Beschriftung ablesen, dass es sich um eine Runden-/Punkteüberischt für die einzelnen Spieler handelt. Zusätzlich ist hier auch wieder ein Titel für das Diagramm angegeben, welches zusätzliche Hinweise dazu liefert, was in der Tabelle abgebildet wird. Das Kriterium der eindeutigen Beschriftung ist hier also schon einmal gegeben. Was jedoch nachteilig an der Darstellung der Tabelle ist, wird sichtbar, wenn man zum Ablesen der Daten übergeht. Wenn zum Beispiel nur 2 oder 4 Spieler an dem Spiel teilgenommen haben, sind die Einträge in der Tabelle noch einigermaßen übersichtlich. Aber wie auf dem Bild zu erkennen ist, wird es ab einer größeren Spieleranzahl, wie hier 10 Spieler, schon schwierig den Überblick zu bewahren. Man könnte es übersichtlicher gestalten indem man zum Beispiel Linien zwischen den Spalten einfügt oder die einzelnen Zeilen mit

verschiedenen Grautönen hinterlegt. Alternativ reicht das Säulendiagramm auch für die Punkteübersicht an sich aus, und man könnte das Filtern der Runden im Säulendiagramm auf eine andere Art und Weise lösen. Zum Beispiel, über ein Dropdownmenu, bei dem man die gewünschte Runde auswählt. Zusammenfassend kann man sagen, dass die Tabelle einige Kriterien, wie Einfachheit erfüllt, aber Übersichtlichkeit bei größeren Datenmengen nicht. Die Funktion einzelne oder mehrere Runden auszuwählen, kann die Lehre Unterstützen, indem noch einmal jene Runden bei der Besprechung hervorgehoben werden können. Dr. Nessler fand, dass das Auswählen der Runden ein guter Zusatz für die Lehre ist. Mein Fazit zu der Tabelle lautet somit, sie kann die Lehre unterstützen, aber es noch einiges verbessert werden kann.

Betrachten wir nun das einfache Liniendiagramm, welches den Kooperationsverlauf visualisiert. Hier hatte sich Dr. Nessler eine simplere Variante als, die derzeit auf der öffentlichen Version, umgesetzte gewünscht. Diese Variante ist im Kapitel Beschreibung des Projekts im Unterkapitel 2.3 in der Abbildung 2 dargestellt. Man kann hier auf Anhieb erkennen, dass die Achsenbeschriftungen fehlen und die Hintergrundfarben eher für Verwirrung sorgen.
Dies wurde in der neuen Darstellung korrigiert. Die neue Darstellung ist im Kapitel „Implementierung“ unter dem Titel „Abbildung 13: Diagramm: Kooperationsverhalten “13 abgebildet. Es wurde eine Achsenbeschriftung an der Y-Achse hinzugefügt und die Hintergrundfarben wurden entfernt. Wenn man nun wieder als betrachtende Person die sich im Prozess der Identifizierung befindet herangeht, kann man aus der Achsenbeschriftung direkt herauslesen, dass in diesem Diagramm der Anteil der Kooperateure in Prozent pro Runde angezeigt wird. Da es sich hier um einen sequentiellen Datensatz handelt, können die einzelnen Punkte miteinander verbunden werden, so dass sich ein Liniendiagramm ergibt. Dies wurde in einer Richtlinie von Kelleher und Wagner so angegeben, da die Daten eine lineare Veränderung zwischen den einzelnen Punkten implizieren. Beim Ablesen kann man dann entnehmen, wie sich der prozentuale Anteil an Kooperateuren in jeder Runde verändert hat.
Die Darstellung ist einfach gehalten und erfüllt somit auch dieses Kriterium der Datenvisalisierung. Es werden kein verwirrenden Farben verwendet und bei jeder Runde ist ein kleiner Kreis abgebildet, der wenn man mit dem Mauszeiger hinüber fährt, wird der genaue prozentuale Anteil an Kooperateuren anzeigt.
Derzeit ist es so, dass mehr Runden in dem Diagramm angezeigt werden, als tatsächlich gespielt wurden, damit das Ergebnis nicht so „abgehackt“ aussieht. Dadurch das die Werte der ungespielten Runden mit 0 intialisiert wurden, können sie jedoch das abgelesene Ergebnis verfälschen. Leider ist das Diagramm derzeit so implementiert, dass diese Daten mit irgendeinem Wert initialisiert werden müssen. Dies könnte man noch ausbessern, so das hier einfach keine Datenpunkte angezeigt werden. Was auch ein wenig ungünstig ist, dass man nach der ersten Runde nur sehr schwer erkennen kann, ob ein Datenpunkt eingetragen wurde, da dieser direkt auf der Y-Achse eingezeichnet wird. Dies könnte noch verbessert werden. Zusammenfassend kann man jedoch sagen, dass das Diagramm die Richtlinien und die Anforderungen von Dr. Nessler erfüllt. Aus diesem Grund komme ich für dieses Diagramm zum Ergebnis, dass es die Lehrkraft bei der Vermittlung der Spieltheoretischen Grundlagen, durch die schematische Darstellung des Kooperationsverhaltens unterstützt.

Das letzte Diagramm, das auf der Runden/Spielergebnisseseite angezeigt wird,

ist eine Tabelle, welche die Gesamtstatistik über alle gespielten Spiele der Sitzung anzeigt. Die Tabelle ist im Kapitel „Implementierung“ unter dem Titel „Abbildung 14: Tabelle: Gesamtstatistik“14 abgebildet. In dieser Tabelle werden für jeden Spieler die Gesamtpunkte, zum einen jeder Spielvariante, so wie aller Spiele angezeigt. In der letzten Spalte „Belohnung“ kann ein beliebiger Positiver Zahlenwert eingestellt werden. Analysieren wir nun diese Tabelle wieder aus der Sicht einer betrachtenden Person, die sich im Interpretationsprozess befindet. Der Titel der Tabelle gibt hier schon einmal eine eindeutige Beschreibung darüber, was in der Tabelle dargestellt wird. Der erste Titel der Tabelle „Actions“ kann für den Betrachter verwirrend sein, da dieser eine Funktion beschreibt und nicht den Inhalt der Tabelle. Die nächsten Spaltentitel, bis auf den letzten, sind eindeutig. Jedoch könnte man Spielmodus in der zweiten Spalte zu „Spielmodus Punkte“ umbenennen, um zu verdeutlichen, dass es sich hier nicht nur um den Spielmodus sondern auch um die Spielerpunkte handelt. Der Letzte Spaltentitel kann für einen Betrachter auf den ersten Blick, ohne Kontext, eher verwirrend wirken, da es in keinem Zusammenhang zu den anderen Titeln steht. Hier wäre es von Vorteil wenn man noch ein Popup oder eine Beschreibung einfügt, um zu erklären, dass man durch das klicken auf den Stift diesen Wert ändern kann und sich die anderen in dieser Zeile dann automatisch anpassen. Was hier auch nachteilig ist, ist dass der Eindruck erweckt wird, dass man jeden Wert in der Zeile „Auszahlung“ anpassen kann. Dem ist jedoch nicht so und dies müsste noch angepasst werden. Leider ist dies erst beim Erstellen der Diskussion aufgefallen und es fehlt zu diesem Zeitpunkt die Zeit dies zu korrigieren. Wenn man jedoch die Funktion der Belohnungsverteilung und Anpassung außen vorlässt und erst einmal nur die Werte in den Reihen darüber betrachtet, ist die Tabelle einfach und verständlich gehalten. Sie kann wie auch schon die andere Tabelle jedoch bei vielen Spielern sehr schnell unübersichtlich und werden. Ein Abtrennen der einzelnen Spalten voneinander und eine dickere Abtrennung zwischen den Spielmodi, Gesamtpunktzahl und Auszahlung könnte hier für mehr Überblick sorgen. Man könnte auch hier wieder über eine farbliche Abgrenzung nachdenken oder indem man die Punkte unter Gesamptpunktzahl dicker darstellt. Die Nutzerfreundlichkeit ist ohne weitere Erklärungen nicht gegeben und bedarf einiger Verbesserung. Man könnte hier die Einstellung der „Anzahl an Belohnung“ aus der Tabelle ausgliedern und neben der Tabelle selbst implementieren. Das könnte es übersichtlicher und verständlicher machen. Für die Lehre erfüllt die Tabelle dennoch ihren Zweck, da sie eine Übersicht über die Gesamtpunkte gibt und kann trotzdem komplementär zur Lehre eingesetzt werden.

Abschließend müssen wir noch einmal die Seite als Ganzes betrachten. Vorteilhaft ist hier, dass man alle Diagramme ein- und ausklappen kann. So kann der Fokus durch die Lehrkraft auf das Diagramm gerichtet werden, welches gerade wichtig ist oder besprochen wird. Wenn jedoch alle Diagramme aufgeklappt sind, ist die Seite sehr voll gerade durch die erste Tabelle. Aus meiner Sicht hat könnte man hier eine bessere Lösung für die Darstellung finden. Zudem ist hier der Inhalt der Tabelle redundant und nur eine andere Form der Darstellung, als Diagramm eins. Allgemein kann man sich auch überlegen, ob es Sinn macht den Kooperationsverlauf pro Runde anzuzeigen, anstelle des Liniendiagramms. Hier könnte man auch die Idee aus dem zweiten Konzept aufgreifen und es für die aktuelle Runde als Kuchendiagramm darzustellen, um die Seite nicht mit so vielen Informationen voll zupacken.

Zurzeit wird während der Runde zusätzlich zu den Diagrammen „Punteübersicht“, „Spielergebnisse“ und „Kooperationsverhalten“ noch angezeigt, welche Spieler schon abgestimmt haben.

5 Fazit und Ausblick

5.1 Fazit

Der Sinn dieser Arbeit war es, die Daten welche während eines Spiels erzeugt werden, in einer Form zu visualisieren, so dass sie einfach verständlich sind und die Lehrkraft in der Lehre unterstützen. Zusammenfassend kann man sagen, dass bis auf einige kleine Punkte, dies mit dieser Arbeit erreicht wurde. Die Diagramme und Tabellen erfüllen die vorgestellten Richtlinien bis auf wenige Punkte, die aus meiner Sicht jedoch vernachlässigt werden können, wenn es um die Funktionalität der Anwendung geht.
Zusätzlich sollte noch eine Datenbank implementiert werden, um die visualisierten Daten zu speichern. Dies ist erfolgt, muss jedoch noch verbessert werden, da ein kritischer Punkte, das Überschreiben von Daten in der Datenbank aus meiner Sicht nicht zufriedenstellend umgesetzt wurde. Dies ist allerdings ein sehr wichtiger Punkt und muss dringend ausgebessert werden.

5.2 Ausblick

Die Anwendung kann noch durch mehrere Funktionen erweitert werden. Zum einen können natürlich die in der Diskussion und im Fazit genannten Punkte ausgebessert werden. Was jedoch auch noch eine wichtige Funktion ist die nun mit der Anbindung der Datenbank in Angriff genommen werden kann, ist die Funktion, dass ein Spiel unterbrochen und in der nächsten Unterrichtseinheit beziehungsweise zu einem späteren Zeitpunkt weitergespielt werden kann. Hierbei ist die größte Schwierigkeit den Spielern ihre alten Spielernamen wieder zuzuordnen. Auch könnte man jetzt Nutzerkonten für Lehrkräfte anlegen, so dass es zum Beispiel nur noch dem Admin vorbehalten ist, ein Spiel zu starten, oder die Spiele in der Datenbank nur noch diesem Nutzer zuzuweisen. Dies könnte auch dabei helfen, das Problem zu lösen, dass Sitzungen mit dem gleichen Spielcode in der Datenbank überschrieben werden.
Zusätzlich sollte auch der Bot weiterentwickelt werden. Dieser wird immer dann aktiv, wenn eine ungerade Anzahl an Spielern in einer Sitzung spielt. Jedoch werden zum einen seine Punkte und sein Kooperationsverhalten, in den Spieldaten mit einbezogen und das Verhalten des Bot ist nicht steuerbar. Es wurden während des Softwareprojekts allerdings schon verschiedene Abstimmungsverhalten für den Bot implementiert. Die Funktion dies auszuwählen muss im Frontend noch implementiert werden. Auch wäre es sinnvoll, den Bot aus den Berechnungen der Spieldaten auszugliedern oder sein Verhalten hervorzuheben, um erörtern zu können, welchen Einfluss dies auf den Spielverlauf haben könnte. Gerade bei kleineren Gruppen ist dies wichtig. Jannis Hamann und Jamie Schenker haben in ihren Arbeiten noch weitere Aspekte genannt, die ich hier jedoch nicht erneut ausführen werde.

Literatur

[1] Dan Abramov und the Redux documentation authors. *Typescript Website*. Accesed last: 2022-03-08. URL: https://www.typescriptlang.org/.

[2] Chris Evergreen Stephanie; Metzner. "Design Principles for Data Visualization in Evaluation". In: *New Directions for Evaluation vol. 2013 iss. 140* 2013 (140 Dez. 2013). DOI: 10.1002/ev.20071. URL: https://onlinelibrary.wiley.com/doi/epdf/10.1002/ev.20071.

[3] The PostgreSQL Global Development Group. *PostgreSQL: About*. Accesed: 2022-02-19. URL: https://www.postgresql.org/.

[4] Inc. Meta Platforms. *React Website*. Accesed last: 2022-03-08. URL: https://reactjs.org/.

[5] Microsoft. *Typescript Website*. Accesed last: 2022-03-08. URL: https://www.typescriptlang.org/.

[6] Microsoft. *WLS Dokumentation*. Accesed last: 2022-03-08. URL: https://docs.microsoft.com/de-de/windows/wsl/.

[7] Helmut Prechtl Simone Lachmeyer; Claudia Nerdel. "Modellierung kognitiver Fähigkeiten beim Umgang mit Diagrammen im naturwissenschaftlichen Unterricht". In: *Zeitschrift für Didaktik der Naturwissenschaften* 13 (2007).

[8] Prof. Dr. Rainer Stripf. *Methoden Handbuch Biologie*. band 2. Aulis Verlag Deubner, 2006.

[9] Christa Kelleher; Thorsten Wagener. "Ten guidelines for effective data visualization in scientific publications". In: *Environmental Modelling Software vol. 26 iss. 6* 26 (6 Juni 2011). DOI: 10.1016/j.envsoft.2010.12.006. URL: https://www.sciencedirect.com/science/article/abs/pii/S1364815210003270.

[10] wiki.postgresql.org. *postgreql wiki: python modules*. Accesed last: 2022-03-08. URL: https://wiki.postgresql.org/wiki/Python.

[11] Asaf Yigal. *Sqlite vs. MySQL vs. PostgreSQL: A Comparison of Relational Databases*. Accesed: 2022-02-19. URL: https://logz.io/blog/relational-database-comparison/.

A Anhang

A.1 Backend-Programmcode Der Backendprogrammcode ist als Archiv verpackt in den eingereichten Daten unter *team-backend-feature-add-database* zu finden.
A.2 Frontend-Programmcode Der Frontendprogrammcode ist als Archiv verpackt in den eingereichten Daten unter *team-frontend-feature-datavisualisation* zu finden.