

# Endbericht - Luck of Vegas

<b>Projektumfang</b>	<b>2</b>
<b>Vorgehensmodell</b>	<b>3</b>
<b>Planung</b>	<b>4</b>
<b>Arbeitsstunden</b>	<b>7</b>
Jonas:	7
Matias:	8
Silas:	8
<b>Akzeptanztest</b>	<b>9</b>
<b>ER-Modell und Datenbank</b>	<b>10</b>
Unser ER-Modell:	10
Relationen Modell:	10
Verwendete SQL-Befehle:	10
<b>Inhalt vom Projekt (Code)</b>	<b>11</b>
Blockchain:	11
Express Session:	11
<b>Schlussbericht</b>	<b>12</b>
Erweiterungsmöglichkeiten:	12
Fazit:	12

# Projektumfang

Das Ziel ist es eine Gamble-Seite zu erstellen auf welchem Nutzer ihre virtuelle Währung in verschiedenen Glücksspielautomaten verlieren oder vermehren können. Die Mindestanforderungen für unser Projekt sind folgende:

- Login und Registrierung von Benutzern welche in Datenbank gespeichert werden
- Startseite mit Slider wo man zwischen den vorhandenen Minispiele wählen kann
- Roulette, das über eine Blockchain Gewinne ermittelt

Zudem soll unsere Seite so aufgebaut sein, dass die Seite ein fertiges Endprodukt ist, es jedoch jederzeit möglich ist neue Minispiele hinzuzufügen.

Wir haben außerdem eine Theoretische Kostenschätzung erstellt, die natürlich nicht besonders realistisch ist. Um trotzdem einen einigermaßen realistischen Preis zu erhalten haben wir folgende Annahmen getroffen:

- Ein durchschnittlicher Programmierer kostet 77€ pro Stunde. Da wir keinen Abschluss haben und nicht besonders effizient arbeiten haben wir 35€ pro Stunde gerechnet.
- Wir bekommen je 21 Stunden Zeit für das Projekt. Also werden voraussichtlich insgesamt 63 Stunden am Projekt gearbeitet.
- Da wir insgesamt 63 Stunden um 35€ pro Stunde arbeiten werden kostet das Projekt 2205€.

## Vorgehensmodell

Wir haben das Kanban Board gewählt. Um das umzusetzen haben wir Trello verwendet. Wir haben uns mehrmals getroffen und immer besprochen wer was gemacht hat und was als nächstes zu erledigen ist. Bei diesen Treffen haben wir dann auch gemeinsam Fehler behoben. Während der Meetings haben wir immer unser Trello-Board aktualisiert. Im Folgenden sieht man anhand unseres Trello Boards wie sich das Projekt mit der Zeit entwickelt hat:

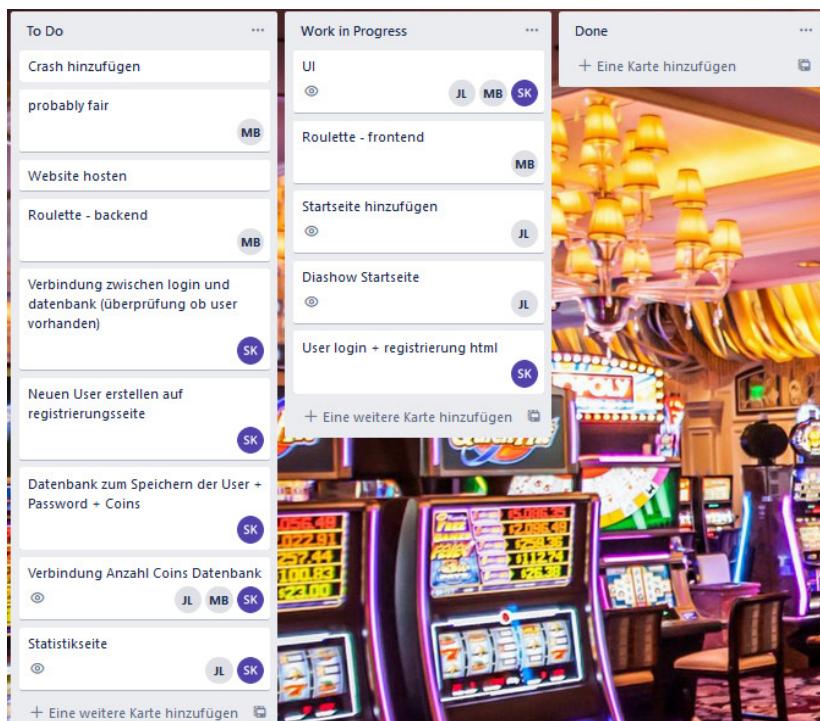


Abbildung 1: Erster Screenshot Trello-Board

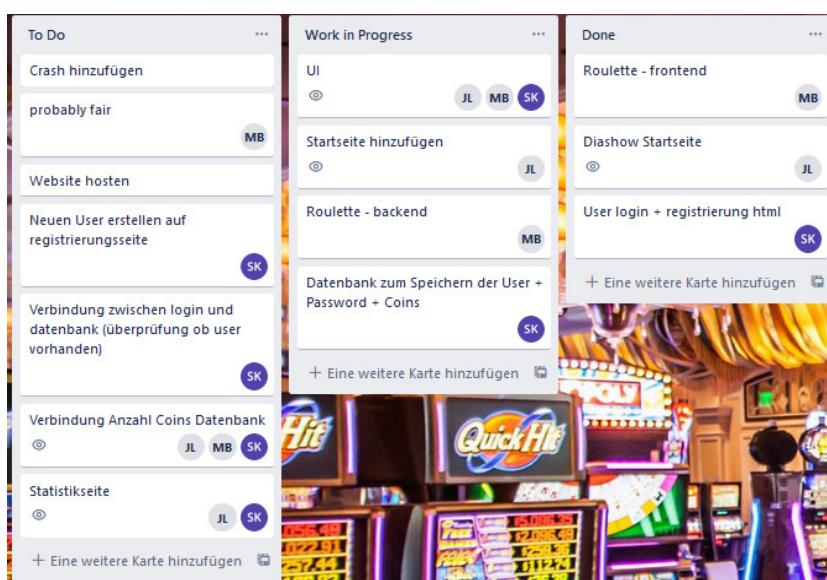


Abbildung 2: Zweiter Screenshot Trello-Board

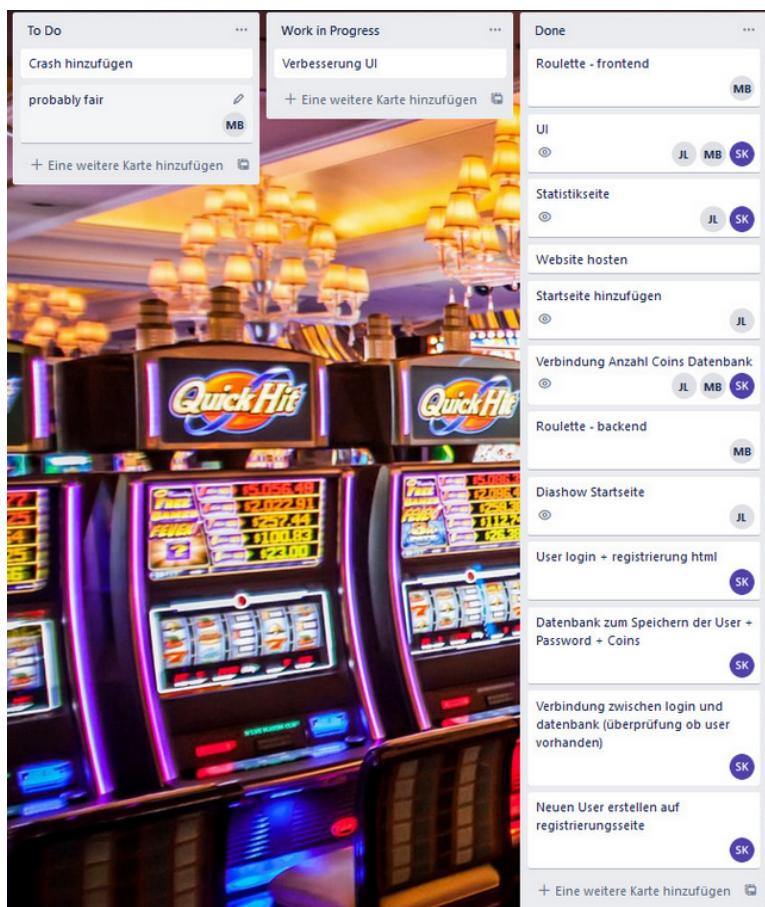


Abbildung 3: Dritter Screenshot Trello-Board

## Planung

Um unser Projekt grafisch zu planen haben wir Figma verwendet. So haben wir für die verschiedenen Seiten jeweils ein grobes Design erstellt, um eine ungefähre Vorstellung vom Endprodukt zu haben, und uns bei der Ausprogrammierung danach richten zu können.

### Login:

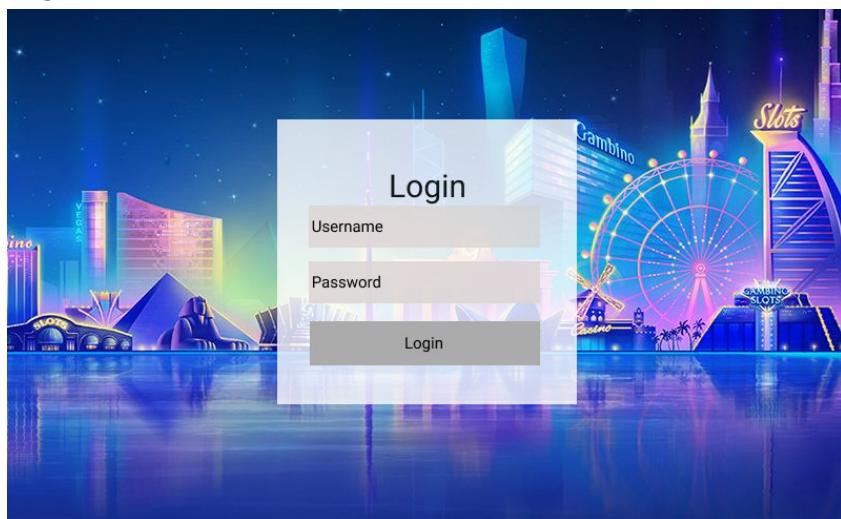


Abbildung 4: Login Design

**Startseite:**

Abbildung 5: Startseite Design

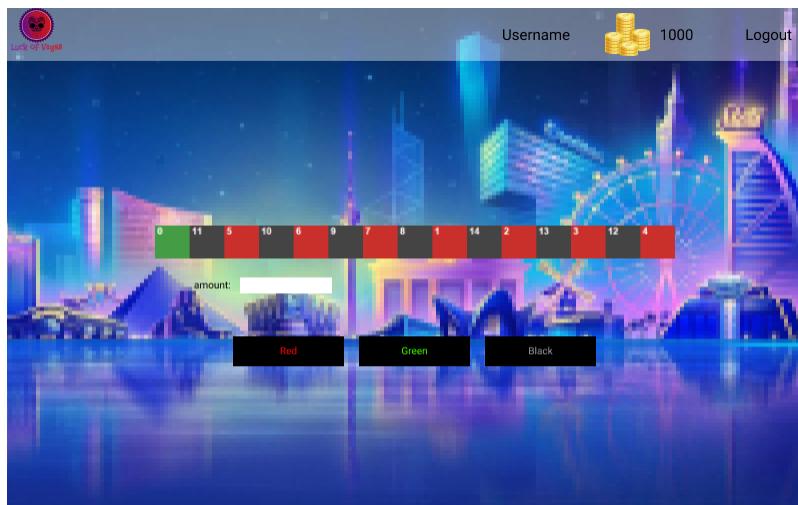
**Roulette:**

Abbildung 6: Roulette Design

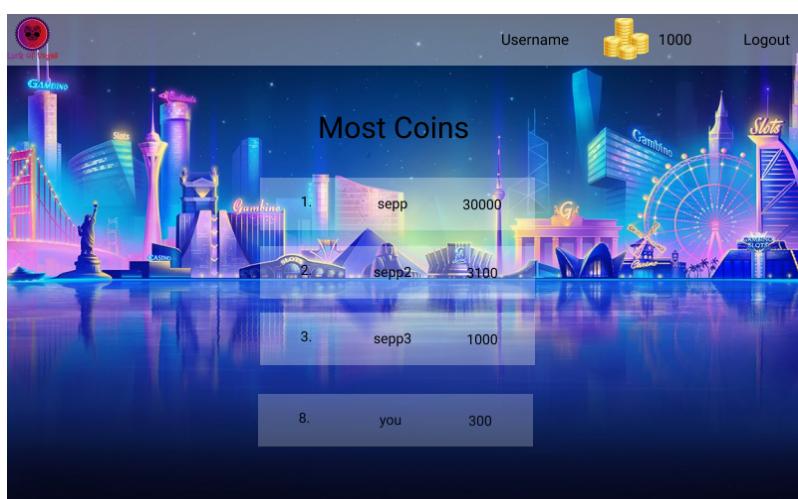
**Statistiken:**

Abbildung 7: Statistik Design

Wir haben unser Projekt mit der Multi-Tier-Architektur geplant. Unser Server läuft mit der dazugehörigen Datenbank auf dem PC im SN-Netz mit der IP 10.10.30.219. Somit können alle Rechner im SN-Netz unsere Website benutzen. Webservice haben wir mit Rest umgesetzt. Somit findet man auf unserer Website unter /home die Startseite, unter /roulette das Roulette und unter /Statistics findet man die Statistiken. Wir haben auch versucht die Website möglichst benutzerfreundlich zu gestalten.

Das haben wir zum Beispiel mithilfe von Metaphern umgesetzt: Wenn man auf den Pokal klickt kommt man zu den Statistiken und das Logo leitet auf die Home-Seite um.



Abbildung 8: Pokal, der zu Statistiken weiterleitet



Abbildung 9: Logo, das auf Home-Seite weiterleitet

Des Weiteren haben wir mehrere Fehlermeldungen erstellt, welche nur die aller notwendigsten Informationen enthalten und die den Benutzer darauf hinweisen, was er falsch gemacht hat.

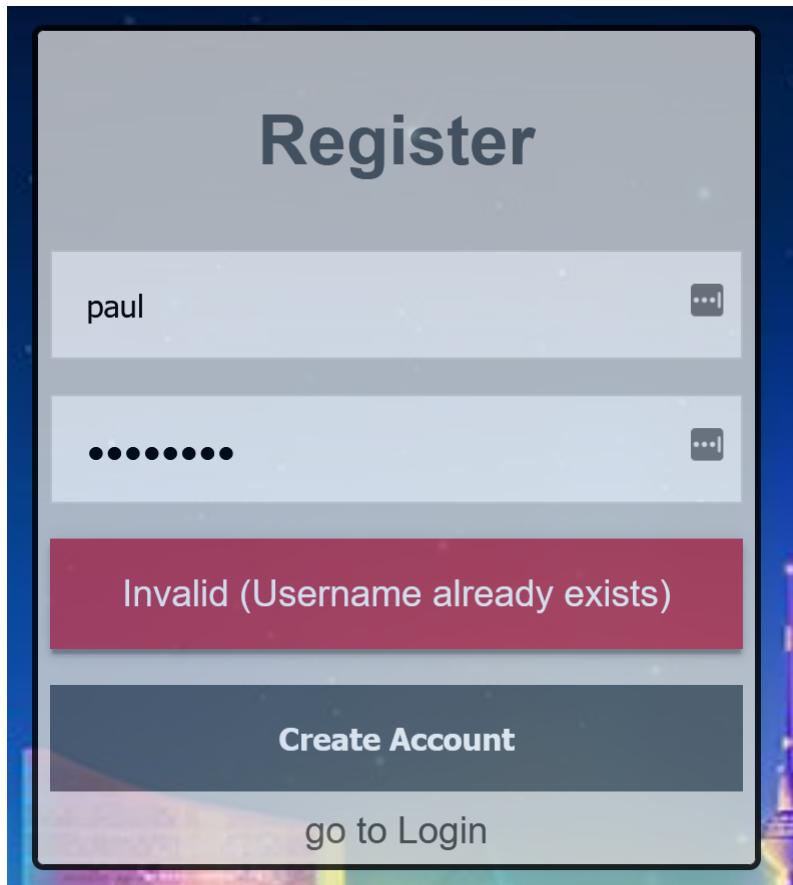


Abbildung 9: Fehlermeldung Login

Außerdem haben wir auch die Buttons auf der Startseite so geändert, dass sie besser erkennbar und somit benutzerfreundlicher sind.



Abbildung 10: Home-Seite Slideshow mit Buttons

## Arbeitsstunden

Wir haben leider vergessen die Arbeitsstunden von Anfang an immer zu notieren. Das heißt natürlich, dass wir jetzt nicht mehr genau wissen an welchem Tag was genau gemacht wurde. Trotzdem haben wir einen recht guten Überblick über die Stunden welche wir für das Projekt aufgewendet haben und können uns an den Meetings, welche wir gemacht haben orientieren. Jeder von uns hat seine Aufgaben zwischen den Meetings mit der dafür verwendeten Zeit nachgetragen.

### Jonas:

- 15. März 2h** Aufteilung Projekt, nähere Besprechung wie Vorgaben umsetzen und Mindestanforderung
- 2h** Designs für verschiedene Seiten mit Figma erstellen
- 1h** Startseite Grundstruktur erstellen
- 19. März ca. 1h** Besprechung der Fortschritte und Aktualisierung Trello-Board
- 4h** Startseite und Header Design anpassen
- 24. März 2h** Besprechung der Fortschritte und Aktualisierung Trello-Board, Start Restplanung
- 7h** gemeinsam mit Silas Recherche und erstellen des Node.js-Servers mit Hilfe von Rest und auf Startseite Slideshow für Minispiele erstellen
- 29. März 2h** Besprechung Bewertungskriterien und Teile des Projektes verbinden, dass man funktionierendes Endprodukt hat
- 30. März** Erste Vorstellung des Projekts

**1. April 3h** Nachbesprechung der Vorstellung, Diskussion was alles verbessert werden soll, neu Verteilung einiger Aufgaben, Planen der kommenden Arbeitsschritte

**5h** Erstellung der Statistik-Seite inklusive Design

**5. April 1h** Besprechung Fortschritte + Aktualisierung Trello-Board

**6h** Weitere Designänderungen auf Startseite und Statistikseite, Tipps von Herrn Trenkwalder(Slider anders/besser erkennbar machen und Logo soll auf Startseite weiterleiten) umgesetzt, Bugs beheben

**11. April 5h** Beheben einiger letzter Bugs, hosten des Projekts im SN-Netz, fertigstellen des Schlussberichts

## Matias:

**15. März 2h** Aufteilung Projekt, nähere Besprechung wie Vorgaben umsetzen und Mindestanforderung

**3h** recherchieren über Blockchain und Zufallsgenerator

**19. März ca. 1h** Besprechung der Fortschritte und Aktualisierung Trello-Board

**3h** Erste Tests mit Blockchain und ganze Back-end für Roulette programmiert

**24. März 2h** Besprechung der Fortschritte und Aktualisierung Trello-Board, Start Restplanung

**10h** recherchieren für Animation bei Roulette und programmiert.

**29. März 2h** Besprechung Bewertungskriterien und Teile des Projektes verbinden, dass man funktionierendes Endprodukt hat

**30. März** Erste Vorstellung des Projekts

**31. März 3h** Nachbesprechung der Vorstellung, Diskussion was alles verbessert werden soll, neu Verteilung einiger Aufgaben, Planen der kommenden Arbeitsschritte

**5h** Fertigstellung von Front und Back-end von Roulette (Verbindung zwischen Roulette und Datenbank)

**5. April 1h** Besprechung Fortschritte + Aktualisierung Trello-Board

**5h** überarbeitung von roulette mit mehreren Optionen

**11. April 5h** Beheben einiger letzter Bugs, hosten des Projekts im SN-Netz, fertigstellen des Schlussberichts

## Silas:

**15. März 2h** Aufteilung Projekt, nähere Besprechung wie Vorgaben umsetzen und Mindestanforderung

**3h** Beginn Entwicklung und Programmierung der html+css files für Login und Registrierung, Überlegen des Designs und der Struktur des Projekts

**19. März ca. 1h** Besprechung der Fortschritte und Aktualisierung Trello-Board

**4h** Fertigstellen der html+css files für Login und Registrierung, Erstellen der Datenbank und Aufbauen einer Datenbankverbindung zum Login

**24. März 2h** Besprechung der Fortschritte und Aktualisierung Trello-Board, Start Restplanung

**9h** gemeinsam mit Jonas Recherche und erstellen des Node.js-Servers mit Hilfe von Rest, Datenbankverbindungen erweitern und Login versuchen mit express Sessions zu

verwirklichen, Funktion User erstellen hinzufügen, Fehlersuche und kleinere Verbesserungen im ganzen Programm

**29. März 2h** Besprechung Bewertungskriterien und Teile des Projektes verbinden, dass man funktionierendes Endprodukt hat

**30. März** Erste Vorstellung des Projekts

**31. März 3h** Nachbesprechung der Vorstellung, Diskussion was alles verbessert werden soll, neu Verteilung einiger Aufgaben, Planen der kommenden Arbeitsschritte

**8h** weitere Recherche und anschließende Überarbeiten der express Sessions, sodass diese erstmals richtig funktionieren, Verbindung zwischen Homescreen und Datenbank erstellt sodass angemeldeter User mit Coins angezeigt werden. Servererweiterung, Fehlerbehebungen

**5.April 1h** Besprechung Fortschritte + Aktualisierung Trello-Board

**10h** Erneutes Überarbeiten/Umbauen des Logins, sodass dieser nun komplett über fetch funktioniert und so Fehlermeldungen beim Login Vorgang benutzerfreundlich ausgegeben werden können. Programmieren der Datenbankabfragen und der Logik für die Statistikseite. Fehlerbehebungen bei Anmeldung Registrierung und Sessions.

**11.April 5h** Beheben einiger letzter Bugs, hosten des Projekts im SN-Netz, fertigstellen des Schlussberichts

## Akzeptanztest

Wir haben uns für die Library Selenium entschieden da ich schon einige Erfahrung mit dieser Library habe. Den Test kann man von GitHub herunterladen, das Dokument heißt test.py.

Um das Programm zu starten braucht man folgende Dinge:

- Python 3 oder höher
- Man muss sich die Selenium Library installieren (Befehl: pip3 install selenium)
- Den Chrome Webbrowser (würde auch mit anderen Webbrowsern funktionieren, aber ich habe Chrome spezifische Einstellungen gemacht)
- Einen Chromedriver den man [HIER](#) herunterladen kann

Um das Programm zu starten muss man es zusammen mit dem Chromedriver in 1 Ordner legen und dann kann man es ganz einfach, wie jedes Python Programm, starten.

Ablauf des Tests:

- Macht ein Login
- Geht auf die Roulette Seite
- Geht wieder zurück zum Home
- Logged sich out
- Wiederholt sich x-mal
- Speicher alle Fehlermeldung (inkl. JavaScript Fehlermeldungen) in ein .txt Dokument

Wir wollten auch den Test so machen, dass wir random Wetten machen können allerdings wussten wir nicht genau wie wir das umsetzen, sollten da wir nicht wissen können (außer über Bilderkennung) ob wir Gewonnen haben

# ER-Modell und Datenbank

Zum Speichern unserer Daten haben wir eine neue Datenbank "Vegasdata" auf mySQL angelegt. Die Datenbank haben wir mit XAMPP gehostet.

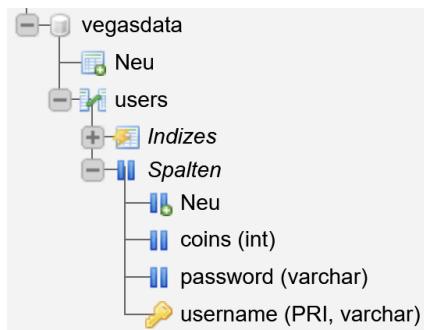


Abbildung 11: Datenbank Struktur

Aufgrund der Beschaffenheit unseres Projektes ist die von uns benötigte Datenbank und somit auch das ER-Modell sehr einfach. Wir haben uns überlegt wie wir unsere Tabelle vielleicht erweitern könnten, allerdings brauchen wir wirklich nicht mehr als die Daten die wir bereits haben. Wenn wir das Projekt noch weiter erweitert hätten, hätten wir noch weitere Daten hinzufügen können (wie z.B. Höchstgewinn, Anzahl der gespielten Spiele usw...) auf welche der Benutzer dann auf der Statistikseite Zugriff hätte. Am jetzigen Zeitpunkt kann der Benutzer dort Informationen über die reichsten Spieler finden.

## Unser ER-Modell:

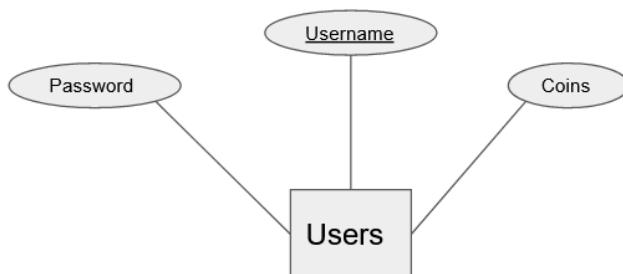


Abbildung 12: ER-Modell unserer Datenbank

## Relationenmodell:

Users: {Username: Varchar[20], Password: Varchar[20], Coins: int}

## Verwendete SQL-Befehle:

```

"SELECT * FROM `users` ORDER BY coins DESC"
"INSERT INTO users (username, password, coins) VALUES ?"
"SELECT * FROM users WHERE username = ?"
"SELECT coins FROM users WHERE username = ?"
"UPDATE users SET coins = '" + userMoney + "' WHERE users.username = ?"
    
```

# Inhalt vom Projekt (Code)

## Blockchain:

Attribute:

- Date (erstell Datum der Chain)
- Seed (ein sha256 Wert der zufällig generiert wird der aber für die gesamte Chain gleich bleibt)
- gameName
- previousHash (SHA256 Hash vom vorherigen Block)

Eine Chain ist 500 Blöcke lang und sobald sie fertig ist wird eine neue erstellt.

Wir haben uns dazu entschieden den Zufallsgenerator mit Hilfe einer Blockchain zu erstellen. Das Ergebnis bei Roulette wird aus einem Hashwert eines Blockes über folgenden Algorithmus generiert.

```
var erg = parseInt(blockchain[index].getHash(), 16) % 15;
```

Dadurch dass unser Zufallsgenerator über eine Blockchain funktioniert könnten wir eine Provably Fair System implementieren (nicht genug Zeit).

Das Provably Fair System würde wie folgt funktionieren. Der Algorithmus und alle Werte der Blockchain sind public. Dadurch ist es nicht mehr zufällig generiert und jeder Benutzer könnte den Algorithmus nachprogrammieren, die Daten der Runde eingeben und kann somit das gleiche Ergebnis bekommen. Damit dieses System nicht exploited werden kann wird der hash jeder runde und der Seed am Anfang versteckt. Nach jeder runde, wird der dem entsprechende Hash veröffentlicht und sobald die Blockchain fertig ist wird der Seed veröffentlicht und eine neue Blockchain wird generiert mit einem neuen Seed.

## Express Session:

Um ein Anmelden auf unserer Seite zu ermöglichen haben wir uns entschieden dafür Sessions zu verwenden. Bei einer Express Session wird nur die Session ID in einem Cookie gespeichert und nicht die ganzen Daten. Diese werden auf dem Server gespeichert.

```
app.use(session({
  secret: 'asuidh6cJSZDBsklu87sbj',
  name: 'uniqueSessionID',
  saveUninitialized:false
}))
```

secret: Das secret wird verwendet um die session zu sichern. Man sollte hierfür im besten Fall eine Zeichenfolge verwenden, die nicht erraten werden kann.

name: Der Name des Session ID Cookie

saveUninitialized:false: verhindert, dass die neue, noch nicht initialisierte Session gespeichert wird

```
req.session.loggedIn = true
req.session.username = res.locals.username
```

mit req.session.loggedIn = true ist man angemeldet und die neue Session wird gespeichert

```
Session {
  cookie: { path: '/', _expires: null, originalMaxAge: null, httpOnly: true },
  loggedIn: true,
  username: 'Michael'
}
```

# Schlussbericht

## Erweiterungsmöglichkeiten:

Unser Projekt ist relativ leicht erweiterbar. Zum Beispiel kann die Statistik Seite mit mehreren Daten über den User gefüllt werden. Man kann zum Beispiel den höchsten Gewinn des Users, die Anzahl an gespielten Spielen, oder die insgesamt verspielten Coins hinzufügen und dem User so mehrere interessante Informationen über seine Karriere liefern. Außerdem können natürlich nach Belieben weitere Minispiele hinzugefügt werden, welche für mehr Abwechslung sorgen und dem User ein besseres Erlebnis auf der Seite bieten würden.

## Fazit:

Wir haben es geschafft die uns am Anfang gesetzten Ziele zu erreichen und haben außerdem eine Statistik-Seite hinzugefügt. Anfangs haben wir auch mit dem Gedanken gespielt weiter Minispiele hinzuzufügen jedoch haben wir bald gemerkt, dass wir mit den am Anfang gesetzten Zielen, genug zu tun haben. Insgesamt sind wir mit unserem Projekt zufrieden, da wir das geschafft haben was wir uns vorgenommen haben.