|  |
| --- |
| Klasse: IT6n |
| Sicherheitskonzept für ein Browsergame |
| Fabian Brinkmann, Kevin Graß, Timo Schmahl, Tom Stich und Fritz Sengelmann |

**Datum: 26.09.2018**

Inhalt

[1 Login und Registrierung 2](#_Toc525646989)

[1.1 Generelle Erläuterung 2](#_Toc525646990)

[1.1.1 Login – generell 2](#_Toc525646991)

[1.1.2 Registrierung via Benutzername und Passwort – generell 2](#_Toc525646992)

[1.2 Login und Registrierung im Frontend 3](#_Toc525646993)

[1.3 Login und Registrierung im Backend 3](#_Toc525646994)

[1.3.1 Login auf Backendseite 3](#_Toc525646995)

[1.3.2 Registrierung auf Backendseite 3](#_Toc525646996)

[1.3.3 Tokenauthentication 3](#_Toc525646997)

[1.2 Hashing 4](#_Toc525646998)

[1.3 SSL Verschlüsselung 4](#_Toc525646999)

[2 Reflexionen 4](#_Toc525647000)

[2.1 Reflexion David 4](#_Toc525647001)

[2.2 Reflexion Fabian 4](#_Toc525647002)

[2.3 Reflexion Kevin 5](#_Toc525647003)

[2.4 Reflexion Timo 5](#_Toc525647004)

[2.5 Reflexion Fritz 5](#_Toc525647005)

[2.6 Gruppenreflexion 5](#_Toc525647006)

# 1 Login und Registrierung

## 1.1 Generelle Erläuterung

### 1.1.1 Login – generell

Eine Login-Funktion ist ein Security-Feature, welches die Verifizierung eines jeden Benutzers vor der Benutzung der Anwendung erfordert. Die Verifizierung geschieht in zwei Schritten. Dies sind die Authentisierung und die Authentifizierung. Der Schritt der Authentisierung geschieht auf der Seite des Benutzers. In diesem Schritt muss der Benutzer seine Identität angeben und nachweisen. Anschließend muss anwendungsseitig im Schritt der Authentifizierung die Authentisierung des Benutzers überprüft werden.  
Für das Verifizieren eines Benutzers gibt es verschiedene Methoden. Die Methoden lassen sich gruppieren in Besitz, körperliche Merkmale (Biometrie) und Wissen.

* **Verifizierung via Besitz**  
  Ein Beispiel für eine Verifizierung via Besitz beim Login, ist das Verwenden eines SSH-Schlüssels. Hier besitzt der Benutzer einen digitalen Schlüssel mit dem er sich anmelden kann.  
  Der Nachteil der Verifizierung mittels Besitz ist, dass er physisch vorhanden ist und somit geklaut werden kann.
* **Verifizierung via Biometrie**  
  Für ein Login mit körperlichen Merkmalen ist die Verifizierung via Fingerabdruck eine Möglichkeit.  
  Diese Methode wird immer beliebter, da mittlerweile viele Smartphones einen Fingerabdruckscanner besitzen. Diese Methode wäre die sicherste, wenn die Geräte nicht immer noch kleine Ungenauigkeiten besäßen und körperliche Merkmale sich nicht z.B. durch Unfälle oder Wachstum verändern würden.
* **Verifizierung via Wissen**Aufgrund der Nachteile der anderen beiden Verifizierungsgruppen, ist die gängigste Methode für einen Login nach wie vor das Verwenden eines Benutzernamens und eines Passwortes. Diese Methode fällt unter die Gruppe Wissen und wird auch in unserem Projekt benutzt. Hierbei muss sich der Benutzer seine Anmeldedaten merken und diese in entsprechende Textfelder über die Tastatur eingeben.  
  Der einzige Nachteil hierbei ist, dass viele Benutzer ihr Passwort zu einfach wählen, es vergessen oder sie die Anmeldedaten unabsichtlich weiter geben (z.B. auf Phishing-Seiten).

1.1.2 Registrierung via Benutzername und Passwort – generellDie Registrierung dient zur Autorisierung eines Benutzers. Das heißt es wird festgelegt wer überhaupt sich einloggen darf. Hierfür gibt es verschiedene Methoden

* **Registrierung via Admin**  
  Eine Methode ist, dass der Anwender einen Administrator der Anwendung bittet, ihn als Benutzer anzulegen.  
  Dies ist jedoch nur notwendig, wenn es sich um vertrauliche Daten handelt.
* **Die Selbstregistration**Da bei unserem Browsergame sich jedoch jeder registrieren dürfen soll, und wir den Administrationsaufwand gering halten möchten, haben wir die Methode der Selbstregistration gewählt. Hierbei können sich die Benutzer selber über ein entsprechendes Formular registrieren. Benutzerkriterien werden hierbei lediglich von der Anwendung bestätigt. Es empfiehlt sich zusätzlich die Validierung einer E-Mail Adresse zu implementieren, damit sich nicht jeder Anwender unendlich viele Benutzer erstellen kann.

## 1.2 Login und Registrierung im Frontend

Die Eingaben des Nutzers werden sowohl bei der Registrierung als auch beim Login in Variablen gespeichert („username“ und „password“). Diese werden dann ans Backend übergeben und entweder im Falle der Registrierung wird ein neuer Nutzer angelegt oder im Falle eines Logins erhält der Nutzer ein Token. Dieses Token wird im Session Storage gespeichert, damit der Nutzer eingeloggt bleibt, solange er sich auf der Seite befindet.  
Besitzt der Nutzer ein Token, so ist es ihm gestattet weiter als bis zum Login oder zur Registration zu gehen und ist in der Lage das Spiel zu starten.

## 1.3 Login und Registrierung im Backend

### 1.3.1 Login auf Backendseite

Beim Login werden Benutzername und Passwort im JSON-Format vom Client erwartet. Der Server holt sich das gehashte Passwort und den Salt aus der Datenbank. Dem übergebenen Passwort wird der Salt hinzugefügt und die Gesamtzeichenfolge gehasht. Stimmt der berechnete Hash mit dem Hash in der Datenbank überein, so war der Login erfolgreich und der Server antwortet mit dem Statuscode 200 („OK“) und einem Token, mit dem er sich in zukünftigen Anfragen authentisieren kann. Sollte der Login fehlerhaft sein, antwortet der Server mit dem Statuscode 401 („Unauthorized“). An dieser Stelle poppt im Frontend ein Fenster auf, dass der Benutzer den Login erneut mit anderen Anmeldedaten versuchen soll.

### 1.3.2 Registrierung auf Backendseite

Die Registrierung von neuen Benutzern läuft ähnlich wie der Login. Der Server erwartet einen Benutzernamen und ein Passwort im JSON-Format vom Client. Zuerst prüft der Server ob in der Datenbank bereits ein Benutzer mit dem gleichen Benutzernamen vorhanden ist. Sollte es bereits einen Benutzer geben, antwortet der Server mit dem Statuscode 409 („Conflict“) und einem „Custom Error Code“ sowie einer Erklärung im JSON-Format. Auch im Frontend wird der Benutzer via Popup gebeten einen anderen Benutzernamen zu wählen. Sollte der Benutzername noch nicht vergeben sein, generiert der Server eine zufällige Zeichenfolge (Salt) und fügt sie dem Passwort an. Die neue Zeichenfolge wird mittels SHA-256 gehasht und zusammen mit Benutzernamen und Salt in der Datenbank gespeichert. Der Server antwortet dann mit dem Statuscode 200 („OK“) und sendet dem Client ein Token, mit dem er sich in zukünftigen Anfragen authentisieren kann.

### 1.3.3 Token Authentication

Für die Token Authentication wurde JSON Web Tokens verwendet. Ein Token besteht aus drei Parts, die im JSON-Format geschrieben werden.

1. Der Header enthält allgemeine Daten wie den verwendeten Verschlüsselungsalgorithmus und den Typ des Tokens.
2. Die Payload enthält die wichtigen Daten, die für die Authentifizierung benötigt werden oder anders nützlich sein können.
3. Bei der Verifizierungssignatur wird der Base64 encodierte Header und die Base64 encodierte Payload mit einem Punkt getrennt, aneinandergeheftet und dann mit dem Secret Key verschlüsselt sowie anschließend gehasht.

Der Token hat dann das Format <base64 Header>.<base64 Payload>.<base64 Verifizierungssignatur>

Ein gültiger Token könnte dann z.B. so aussehen: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJodHRwOi8vc2NoZW1hcy54bWxzb2FwLm9yZy93cy8yMDA1LzA1L2lkZW50aXR5L2NsYWltcy9uYW1lIjoiRiIsImlzcyI6Imh0dHA6Ly9sb2NhbGhvc3Q6NDQzMTYvIiwiYXVkIjoiaHR0cDovL2xvY2FsaG9zdDo0NDMxNi8ifQ.xBHxCzGO\_MSsVmJlE6XSGNw1s-tImRKXf5LjnrJPchA

In der Payload unseres Tokens wird der Name des Benutzers, der Aussteller (in diesem Fall <http://localhost:44316/>) und die Audience (auch <http://localhost:44316/>) gespeichert.

## 1.2 Hashing

Hashing dient dazu das Passwörter nicht in Klartext gespeichert werden und sie auch nicht als Klarsicht einsehbar sind.

Dies passiert durch das Hashen des Passworts, die Hashfunktion wandelt das Passwort in eine Zeichenfolge mit einer festen Länge um. Dieser Vorgang kann nicht rückgängig gemacht werden und dadurch ist das Passwort für uns so wie für Fremde nicht als Klartext einsehbar. Um zu überprüfen ob das eingegebene Passwort mit dem Hinterlegten übereinstimmt, wird das eingegebene Passwort gehasht und mit dem hinterlegten Passwort verglichen.

## 1.3 SSL Verschlüsselung

Eine SSL-Zertifizierung ist essenziell erforderlich für Seiten, auf denen ein Login stattfindet, wie bspw. bei Banken, Online Shops oder eben unserem Spiel.  
Ist eine Seite SSL geschützt, so ist es Hackern deutlich erschwert die Eingaben des Nutzers abzufangen und für sich zu benutzen. Die Verschlüsselung dient dazu die Kommunikation zwischen Datenbank und Login sicher zu stellen und sicher zu stellen, dass kein Dritter an diese Daten kommt. Das schafft das Protokoll, indem es immer wieder in regelmäßigen Abständen Prüfziffern an beide Kommunikationspartner schickt und diese abgleicht. Im Grunde ist SSL ein reines Aufzeichnungsprotokoll. Aber auch schon bevor Daten ausgetauscht werden, sorgt SSL für die Übermittlung persönlicher Identifikationsdaten der Teilnehmer. Es sorgt dafür, dass beide Teilnehmer sich auf eine Art der Fragmentierung und Kodierung einigen und nur diese verwenden.  
Man sollte bei der Eingabe von persönlichen Daten immer darauf achten, dass die Website „https“ enthält, denn nur so kann man relativ sicher gehen, dass man an der Stelle in keine Phishing-Falle läuft.

# 2 Reflexionen

# 2.1 Reflexion David

Ich hatte es mir zur Aufgabe gemacht das Frontend für den Login- und Registrationsbereich zu programmieren und zu gestalten. Hier für musste ich wissen, wie das geschriebene Backend von Fabian genau funktioniert; sowohl was sein Backend von mir an Eingaben erwartet als auch was ich für Informationen erhalte. Da ich recht wenig von Backend-Programmierung verstehe, musste Fabian sehr viel Geduld mit mir haben. Er hat mir alles hervorragend erklärt, ging auf sämtliche meiner Fragen ein und bewies sehr viel Geduld.   
Schließlich war ich in der Lage die Benutzereingaben ans Backend zu übergeben, ein Token zu erhalten und mit dessen Erhalt das Spiel zu starten.  
Des Weiteren hat Fabian mich darin unterrichtet, wie man eine SSL-Zertifizierung einrichtet.

## 2.2 Reflexion Fabian

Meine erste Aufgabe war die Erstellung des Accountservers. Da ich mich mit C# und den asp.net Technologien bereits gut auskannte, nutzte ich die Möglichkeit mich mehr mit Dependency Injection(DI) und Best-Practises auseinanderzusetzten, um so eine skalierbare Grundlage für den Server zu schaffen. Das Salten und Hashen von Passwörtern war mir im Vorfeld schon bekannt und erforderte kaum neues Wissen. In der Tokenauthentifizierung kannte ich mich noch nicht aus und hatte so Zeit mich mit ein paar verschiedenen Verfahren auseinanderzusetzen.

Das Projektumfeld fand ich nicht sonderlich berauschend. In den umliegenden Räumlichkeiten wurde mehrmals die Woche lautstark gebohrt.  
Die Arbeit in der Gruppe war im Großen und Ganzen oberes Mittelmaß. Es ist des Öfteren vorgekommen, dass Gruppenmitglieder über eine längere Zeit nicht am Projekt gearbeitet haben.

Aus dem Projekt nehme ich mit, dass ich meine Motivation etwas zu tun weniger von der Arbeitsbereitschaft der Gruppe abhängig zu machen.

## 2.3 Reflexion Kevin

Da ich mich mehr um die Projektstruktur und die Dokumentation gekümmert habe, war ich nur an der Planung der Sicherheitsumsetzung intensiv beteiligt. Die grundlegende Diskussion galt der Frage nicht wie, sondern was wir absichern wollen. Dabei ging es um zwei Möglichkeiten. Die erste war es, Online einen Highscore zu führen und diesen mit einem Login zu schützen, die zweite einen Login zum beitreten des Spiels zu verwenden. Wir haben uns für die zweite Version entschieden, weil sich darauf besser aufbauen lässt und es für die Muss-Kriterien eine elegantere und einfachere Lösung darstellt. Nach dieser Diskussion, habe ich lediglich in den Meetings, neue Stände gezeigt bekommen und mich über die Funktionsweisen aufklären lassen.

## 2.4 Reflexion Timo

Ich habe mich im Rahmen des Projektes nur indirekt an den Sicherheitskonzept beteiligt, da bei drei Sicherheitsthemen und sechs Gruppenmitglieder nicht jeder direkt etwas implementieren konnte. Daher habe ich eine beratende und recherchierende Rolle übernommen. Die Gruppenatmosphäre war sehr gemischt. Beeinträchtigend für die Motivation war vor allem der Baulärm in der Schule. Auch war die Gruppe zu groß und es gab ein paar Probleme bei den Absprachen. Alles in allem hat das Projekt trotzdem Spaß gemacht und ich konnte viel neues Wissen mitnehmen. Insbesondere im Bereich des backendseitigen Logins.

## 2.5 Reflexion Fritz

Ich habe im Projekt nur geringfügig an der Sicherheit gearbeitet, da wir eine große Gruppe sind und nicht viel im Bereich Sicherheit zu tun hatten.

Wir haben uns am Anfang alle zusammengesetzt und uns überlegt welche Sicherheitskonzepte wir umsetzen wollen am Ende haben wir uns für Login und Registrierung, Hashing der Passwörter und die SSL Zertifizierung entschieden.

Die Projektarbeit lief im Ganzen sehr gut, die Aufteilung der Aufgaben viel aber nicht ganz so leicht da die Gruppe sehr groß ist. Das Projekt hat Spaß gemacht und ich habe vieles neues gelernt.

## 2.6 Reflexion Tom

Anfangs mussten wir uns erstmal überlegen, welche Sicherheitsaspekte wir in unser Projekt einbauen wollen. Nach ein paar Gesprächen haben wir uns für den Login und Registrierung, Password Hashing und das SSL Zertifikat entschieden. Bei der Umsetzung von den Sicherheitsaspekten konnten nicht alle aus der Gruppe gleichzeitig daran arbeiten und deshalb hat nicht jeder direkt dazu beigetragen, sondern andere Aufgaben, welche im Projekt anstanden, umgesetzt, jedoch haben wir den Aufbau für Login und Registrierung im Vorhinein gemeinsam durchgesprochen. Alles in allem hatte ich das Gefühl, dass die Umsetzung gut lief.

## 2.7 Gruppenreflexion

Wir haben unsere Muss-Kriterien erfüllt und alles umgesetzt. Jedoch müssen wir uns auch einige Sicherheitslücken eingestehen.   
Zum einem ist das SSL Zertifikat ein selbst erstelltes. Somit ist die Sicherheit nur intern von uns bestätigt worden und nicht von einem offiziellen SSL-Zertifizierer.   
Zum anderen ist das Spiel in JavaScript geschrieben und der Code im Browser der Benutzer ersichtlich. Der Login schützt so nur die Daten des Benutzers in der Datenbank aber nicht den Code des Spiels selbst. Außerdem sind wir nicht sicher davor, dass unsere Datenbank mit Benutzern vollgespamt wird, da wir keine E-Mail-Adresse vom Benutzer abfragen und validieren.  
Trotzdem sind wir mit unserem Ergebnis zufrieden, da wir unsere Lücken reflektieren können und sie somit einen Ausblick auf die zukünftigen Versionen unseres Spiels bilden.  
Dieser Ausblick umfasst das notwendige angeben von E-Mail-Adressen bei der Registration,   
das automatische Senden eines E-Mail-Bestätigungslinks an diese Adresse nach der Registration („Double-Opt -In“), das Minifizieren des JavaScript-Codes, um ihn vor Diebstahl und Manipulation etwas besser zu schützen, sowie das Erwerben eines richtigen SSL-Zertifikates, sobald das Spiel veröffentlicht wurde.