# Ziele und Anforderungen

## Vorgaben

# Planung

## Rahmenbedingungen

### Sicherheit

### Performance

### Verfügbarkeit

### Transaktionsvolumen

## Datenbank Schema

## GUI Design

## Zeitplan

### Meilensteine und Arbeitsschritte

* Aufwände

### Zuteilung

# Technologie

## Server / Backend

### Node.js

JavaScript ist eine im Browser integrierte Clientseitige Programmiersprache fürs Web. Mit Node.js lässt sich jedoch JavaScript auch ausserhalb des Browsers einsetzen. Wir werden unseren Webserver mit Node realisieren, da der Datenaustausch über das JSON Format simpel und die für unser Spiel benötigte realtime mit Websockets verfügbar ist.

### Frameworks

#### Express

Um unsere API zu gestalten, werden wir auf das Web Framework Express setzen. Es stellt uns viele Funktionen bereit.

#### Socket.io

Socket.io ist eine Bibliothek basierend auf Websockets und erlaubt es uns, eine realtime, beidseitige Verbindung zwischen Client und Server herzustellen. Wir benötigen diese Funktionalität ist unserem Multiplayer Spiel, da ein Spielzug direkt dem Server und den anderen Spielern mitgeteilt werden muss. Nebst dem Server muss dies auch auf dem Client implementiert werden.

#### JWT Bearer Token

JSON Web Token ist ein Verfahren, um zwischen Client und Server eine Sichere Verbindung aufzubauen. Wir bieten zwar auch die Möglichkeit an, als Gast zu spielen, jedoch sollen Registrierte Benutzer über erweiterte Funktionalitäten Verfügen. Sie können so zum Beispiel ein Profil erstellen, eine Freundesliste Verwalten und Spielstatistiken einsehen. Das Bild zeigt den Ablauf bei einer Authentifikation Abfrage.

## Client / Frontend

### Vue.js

Vue.js ist ein clientseitiges Framework basierend auf JavaScript. Es wird zum Erstellen von Single Page Applications verwendet.

### Frameworks

#### Vuetify

Es ist das populärste Framework für Vue.js. Vuetify sehr viele Komponenten zur Verfügung und hält sich an die Material Design Spezifikationen. Es nimmt uns einen grossen Teil der Arbeit für die Anforderungen an das Design ab, wie etwa das Responsive Design für Desktop, Tablet und Mobile.

## Datenbank

### MS SQL

Die Datenbank Technologie wurde uns so vorgegeben. Um eine einfache Datenbankverbindung mit dem Backend herstellen zu können, klärten wir vorerst ab, ob unsere Backendtechnologie dies zulässt. Wir werden für unser Node.js backend das Package mssql verwenden, welches als Treiber alle gängigen moderne Versionen von SQL Servern unterstützt.

## Webserver

Ein Webserver ist uns als Technische Vorgabe vorausgesetzt. In der Entwicklungsumgebung wird dieser nicht benötigt, da Frontend und Backend selbst gebuildet werden können. In der Produktivumgebung muss das Frontend aber gebuildet und die resultierenden Dateien auf einem Webserver gehostet werden. Wir werden dafür vermutlich Apache verwenden, können diesen aber ohne Schwierigkeiten je nach System ersetzten. Als Alternative für einen Windowsserver wäre der IIS geeignet.

# Anhang

## Glossar

### Abkürzungen

MSSQL - Microsoft SQL Server  
IIS - Internet Information Services

### Allgemeine und Technische Bezeichnungen

### Projektbezogene Bezeichnungen

## Verzeichnis

### Grafiken

<https://nordicapis.com/why-cant-i-just-send-jwts-without-oauth/>

### Frameworks

<https://nodejs.org/en/>  
<http://expressjs.com/>  
<https://socket.io/>  
<https://jwt.io/>  
<https://vuejs.org/>  
<https://vuetifyjs.com/en/>  
<https://www.npmjs.com/package/mssql>