# Online Quiz App

Das Ziel ist die Umsetzung eines Web-Apps. Jeder Benutzer kann entweder online mit anderen Benutzer oder auch allein (Single-Player Mode) ein Quiz App spielen.

## Willkommen Seite

Das ist die erste Seite, welche jede Benutzer beim Start/Öffnen der Seite sieht. Es wird folgende Informationen auf dieser Seite dargestellt:

* Anzahl der aktiven/online Benutzer
* Willkommen Title und Beschreibung mit einem Hintergrund-Bild präsentiert, dieses Layout wird ganz oben dargestellt
* Spiel Regeln sowohl für Multiplayer als auch Singleplayer
* Top 5 Benutzer, die am meisten richtig beantwortet haben
* Top 5 Schwierige Fragen, die am meisten falsch beantwortet wurden
* „*Jetzt Spielen*“ Button um die Spiel-Beginn Seite zu öffnen

## Spiel-Beginn Seite

Auf dieser Seite soll sich der Benutzer dafür entscheiden ob er ein Spiel in Single-Model oder Multiplayer Mode starten, dafür werden folgende Aktionen nötig:

* Input Feld um den Benutzernamen einzugeben
* Ein Button für „Singleplayer Mode Starten“, darunter kurze Beschreibung über Singleplayer Mode

Wenn der Benutzer „Single Mode Starten“ auswählt, wird die Seite für das Single Player Spiel geöffnet

* Ein Button für „Multiplayer Mode Starten“, darunter kurze Beschreibung über Multiplayer Mode und Anzahl der aktiven Benutzer.  
  Wenn der Benutzer „Multiplayer Mode Starten“ auswählt, wird die Seite für das Multi Player Spiel geöffnet.

## Single Player Mode Seite

Auf dieser Seite kann der Benutzer sofort mit einem Spiel anfangen. Die Fragen werden über bereitgestellt Node.JS Server über REST API mit einem AJAX Aufruf geladen. Die UI ist nicht verantwortlich die Fragen zufällig darzustellen, weil der Server die Fragen bereits zufällig schickt.

Jede Runde dauert max. 5 Minuten und für jede Frage hat der Benutzer maximal 60 Sekunden Zeit. Wenn der Benutzer für eine nicht innerhalb von 60 Sekunden eine Antwort eingibt, muss sie automatisch als „falsch“ markiert werden.

Die Ergebnisse werden auf der Ergebnis Seite angezeigt, wenn der Benutzer alle Fragen beantwortet hat oder wenn die Zeit überschritten wurde.

## Ergebnisse für Single Player Seite

Auf dieser Seite wird der Benutzer folgende Informationen sehen:

* Richtig beantwortete Fragen
* Falsch beantwortete Fragen
* Wie lange Zeit hat er insgesamt verbraucht hat, wenn die Zeit nicht überschritten wurde

## Multiplayer Mode Seite

Um die Multiplayer Mode zu ermöglichen muss die App Websockets implementieren. Es gilt folgende Bedingungen bevor ein Spiel startet:

* Es wird eine „wird geladen“ oder ähnliches angezeigt, bis der Server über die Verfügbarkeit einen anderen Benutzer informiert
* Sobald der Benutzer bereit ist, wird der Server folgende Information geschickt:

{

username: „Kerim 2“

ready: true,

...

}

Konkrete Information wird noch definiert.

* Der Server schickt über Websocket die Fragen, nachdem Erhalt der Fragen startet die App sofort das Spiel
* Die Antwortzeit und gesamte Zeit ist gleich wie in Single Player Mode
* Der Server schickt im Laufe des Spiels folgende Informationen mehrmals, die dem Benutzer angezeigt werden:
  + Wie viele Fragen hat anderer Benutzer bereits beantwortet hat
  + Wie viele Fragen hat anderer Benutzer bereits richtig beantwortet hat
  + Wie viele Fragen hat anderer Benutzer bereits falsch beantwortet hat
* Der Benutzer wird auch seine Status, wie oben definiert wurde, sehen
* Nachdem Ende des Spiels muss die App die Fragen und die Antworten, die der Benutzer beantwortet hat, zum Server schicken
* Bis der Server die Ergebnisse schickt, muss die App „Bitte Warten...“ Dialog zeigen (Blocked-UI)
* Erst nachdem Erhalt der Rückmeldung vom Server werden die Ergebnisse auf der „Ergebnis Seite“ angezeigt.

## Ergebnis Seite für Multiplayer Mode

Hier wird der Benutzer das End-Ergebnis des Spiels, das vom Server erhalten wurde, sehen. Folgende Daten werden vom Server geschickt:

* Gewinner – Name, Verlierer Name
* Dauer des Spiels
* Wenn sie gleiche richtige Anzahl der richtigen Antwort haben und wegen schnelleren Antwort Zeit gewonnen hat, muss es auch textuell dargestellt werden
* Kommentar für das Spiel zusammenfassen und zum Server über REST API mit einem AJAX Aufruf schicken

# Technologie Stack

* HTML 5
* CSS 3
* Javascript
  + JQuery
  + Socket.io (Client Library)
  + Toastr
* Twitter Bootstrap
* Animate.css