Dokumentation unserer Erfahrungen während der Bearbeitung der sechsten Aufgabe („Server: Erster Node Server [sic!]“) in der Lehrveranstaltung „Entwicklung interaktiver Anwendungen II, Praktikum“ im zweiten Semester des Bachelor-Studiengangs „Medienkonzeption, Bachelor“ gemäß SPO-Version 14 mit Stand vom 29. Juli 2016 an der Hochschule Furtwangen University in Furtwangen im Schwarzwald, Baden-Württemberg, im Sommersemester 2019

**Kurztitel:** Aufgabe 6 – Dokumentation

**Autoren:** Pascal Michel | Michel Orlik

**Abgabedatum:** 2. Mai 2019

## Aufgabe 1

Kommentierte Datei `Server.ts` -> siehe Steckbrief.

## Aufgabe 2

2.1) Wir haben uns in der Eingabeaufforderung/PowerShell zum Pfad des Repository-Ordners navigiert und haben mithilfe des Befehls `npm i @types/node` selbiges installiert. Dies war uns ohne Probleme möglich.

2.2) Den lokalen Server haben wir in PowerShell im entsprechenden Pfad per Befehl `node Server.js` gestartet. Zuerst hatten wir mit dem Befehl `http-server` einen Server gestartet, was auch ohne Probleme möglich war, wobei wir später herausgefunden haben, dass der zuerst genannte Befehl der richtige war, da ansonsten zwar auch ein Server gestartet wird, dieser allerdings mit der Datei `Server.js` nichts zu tun hat.

2.3) Die URL `http://localhost:8100/` ließ sich in Firefox problemlos aufrufen. In der Systemkonsole wurden die Kommentare aus der Datei `Server.js` angezeigt, wie beispielsweise `Starting server` und `Listening`.

2.4) Nachdem wir etwas hinter den Schrägstrich geschrieben hatten, wurde dies im Browser-Fenster entsprechend angezeigt.

2.5) Hierbei hatten wir keine Probleme. Der eingegebene Text wurde in der Systemkonsole korrekt angezeigt.

2.6) Man beendet den lokalen Server, indem man im Terminal die Tastenkombination `Strg-C` drückt.

## Aufgabe 3

3.1) Wir haben die Datei `package.json` in unser Repository übernommen und den Pfad geändert. Es gab dabei keine Probleme.

3.2) Nachdem wir ein Leerzeichen aus unserem Pfad entfernten, hat es keine Probleme mehr gegeben. Auch sonst hat hier alles funktioniert, da im Browser-Fenster ein Schrägstrich dargestellt wurde. Die Registrierung bei Heroku hat ebenfalls ohne Probleme geklappt.

3.3) Den geforderten Link im Steckbrief haben wir erstellt. In der jeweiligen Heroku-App wird der eingegebene Text angezeigt.

## Aufgabe 4

4.1) Wir mussten den Code der Eisdealer-Seite hinsichtlich einzelner Variablen-Namen und in der Datenstruktur minimal anpassen. Außerdem haben wir eine eigene Funktion geschrieben, die die URL für den Server automatisch generiert. So lassen sich einzelne Namen anpassen, wobei die Daten dabei trotzdem unverändert mit der Datei `data.js` verknüpft bleiben.

4.2) Zuerst wussten wir nicht genau, wie man die Daten an die Heroku-App übermitteln kann. Wir haben herausgefunden, dass dies ziemlich einfach über den Befehl `submit` automatisch funktioniert. Wir haben uns allerdings dafür entschieden, die URL selbst zu generieren und diese mit `window.open(urlSchreiben);` zu öffnen, wobei `urlSchreiben` die Variable ist, in der wir die URL geschrieben haben. Dies hat gut funktioniert. Der neue Tab öffnet sich wie gewünscht und zeigt die Daten in der Heroku-App an.

**;-)**