# Web-Entwicklung

Hausarbeit im Sommersemester 2020

## Aufgabenstellung

Entwickeln Sie eine einfache Microblogging-Web-Anwendung in der Art von z.B. tumblr.com. Die Anwendung besteht aus einem Server, der Blog-Beiträge speichert und auf Anfrage zur Verfügung stellt, sowie einer Browser-Anwendung, die neben der Liste der Beiträge (dem eigentlichen Blog) auch ein Formular zum Erstellen von Blog-Beiträgen anbietet.

Jeder Blog-Beitrag besteht aus

- einem Text (unformatierter Text; mindestens ein und maximal 1024 Zeichen) und
- dem Zeitstempel der Erstellung.

Optional kann ein Blog-Beitrag ergänzt werden mit

- entweder einem vom Nutzer hochgeladenen JPEG-Bild,
- oder einem GPS-Track in Form einer vom Nutzer hochgeladenen GeoJSON-Datei.[1]

Der Blog soll paginiert sein. Pro Seite sollen maximal 10 Beiträge angezeigt werden. Über entsprechende Schaltflächen soll zwischen den Seiten gewechselt werden können. Der aktuelle Seitenindex sowie die Gesamtanzahl der Seiten sollen angezeigt werden.

Die Blog-Beiträge sollen gemäß des Zeitstempels sortiert werden (neueste Beiträge zuerst).

Wurde ein Bild hochgeladen, wird es oberhalb des Textes angezeigt. Wurde eine GPS-Track hochgeladen, wird oberhalb des Textes eine Karte angezeigt, die den GPS-Track als Overlay darstellt. Achten Sie darauf, dass der gesamte Track im Kartenausschnitt sichtbar, zentriert und möglichst ausschnittfüllend ist. Sie dürfen den Kartendienst bzw. das hierfür benötigte npm-Modul selbst wählen.

Ein mithilfe von Node.js und Express realisierter Server soll einerseits die Browser-Anwendung als statische Dateien an den Browser ausliefern, andererseits die Persistierung und Auslieferung der Ressourcen über eine REST-konforme HTTP-Schnittstelle zur Verfügung stellen. Sie müssen keinerlei Benutzerverwaltung, Authentifizierung oder Autorisierung umsetzen – jeder darf Beiträge erstellen und einsehen. Zur Persistierung der Daten haben Sie die Wahl zwischen einer SQLite-Datenbank oder Dateien im Dateisystem. Falls Sie SQLite einsetzen, muss die Initialisierung der Datenbankstruktur automatisch durch die Server-Anwendung erfolgen.

Weitere Anforderungen sind:

- Ihre Anwendung muss zumindest in aktuellen Versionen von Google Chrome und Mozilla Firefox funktionsfähig sein.
- Sie dürfen keinen JavaScript-Präprozessor (z.B. TypeScript) einsetzen.
- Sie dürfen kein MV*-Framework (z.B. Angular, React oder Vue.js) oder jQuery einsetzen.
- Es ist Ihnen freigestellt, ob Sie ES6+-Sprachkonstrukte einsetzen oder nicht.
- Es ist Ihnen freigestellt, ob Sie ein CSS-Framework einsetzen oder nicht.

---

[1] Wie Sie den beigefügten Beispieldateien entnehmen können, sind das JSON-Dateien, die im Wesentlichen aus einer geordneten Liste von Geokoordinaten bestehen. Ein kleiner Tipp: Anders als in den meisten anderen Geodatenformaten wird eine Geokoordinate in GeoJSON in der Reihenfolge [Längengrad, Breitengrad, Höhe] und nicht [Breitengrad, Längengrad, Höhe] notiert. Die vollständige Spezifikation finden Sie unter http://geojson.org.

- Achten Sie auf eine sinnvolle Ordnerstruktur und Modularisierung Ihres Projekts mithilfe von CommonJS-Modulen und browserify.
- Sie dürfen neben ggf. explizit genannten auch weitere npm-Module einsetzen, insofern diese in den Build-Prozess (s.u.) eingebunden sind.
- Ihr Code darf auf Grundlage der semistandard-Regeln keine Fehler aufweisen. Anderenfalls führt dies bei der Prüfung automatisch zu einer Abwertung um einen Notenschritt.

## Bearbeitung in Teams

Die Bearbeitung der Hausarbeit (sowohl der Studienleistung als auch der Prüfung) erfolgt in Teams mit je zwei Studierenden. Die Zusammensetzung der Teams muss spätestens zu Beginn der Bearbeitungszeit via E-Mail an c.bettinger@hochschule-trier.de gemeldet werden. **Die Teilnahme an der Studienleistung und die Zulassung zur Prüfung setzt neben der Anmeldung im QIS zusätzlich diese Meldung voraus.**

Studierende, die keinen Team-Partner finden, müssen sich ebenfalls melden. Diese werden zufällig anderen Teilnehmern ohne Partner zugewiesen.

## Studienleistung

Voraussetzung zur Zulassung zur Prüfung ist der Nachweis der Studienleistung. Um diesen zu erbringen, muss ein Grundgerüst der Hausarbeit erstellt und fristgerecht abgegeben werden.

Dieses Grundgerüst muss über folgende Komponenten verfügen und die folgenden Anforderungen erfüllen:

- Die clientseitige Browser-Anwendung
  - referenziert **eine einzige CSS-Datei**, die aus einer oder mehreren Less-Datei(en) erzeugt wird,
  - referenziert **eine einzige JS-Datei**, die das Ergebnis des Bundlings aller Abhängigkeiten mithilfe von browserify ist.
- Die serverseitige Node.js-Anwendung
  - startet einen HTTP-Server an einem Port, der als Kommandozeilenargument übergeben werden kann,
  - liefert die clientseitige Anwendung als statische Dateien an den Browser aus.
- Einen npm-Build-Prozess,
  - der sowohl auf unixoiden Betriebssystemen als auch unter Microsoft Windows in einer **bash-Shell** ausgeführt werden kann,
  - der durch Aufruf von `npm run clean` das Projekt bereinigt, d.h. **alle** Dateien löscht, welche durch den Build-Prozess heruntergeladen oder generiert wurden,
  - der durch Aufruf von `npm run lint` **alle** JS-Dateien im Projekt (sowohl Client als auch Server) mithilfe von semistandard überprüft,
  - der durch Aufruf von `npm run debug` das gesamte Projekt erzeugt,
  - der durch Aufruf von `npm run build` ebenfalls das gesamte Projekt erzeugt und dabei die CSS- und JS-Datei minifiziert,
  - der das Erzeugen des Projekts mit `npm run debug` oder `npm run build` abbricht, falls semistandard Fehler aufdeckt,
  - der durch Aufruf von `npm run start` oder `npm start` den HTTP-Server an Port 8080 startet.

Zum erstmaligen Starten der Anwendung an Port 8080 muss also lediglich `npm install && npm run build && npm start` ausgeführt werden. Bricht dieser Prozess ab oder ist das Ausführen der Anwendung im Browser anschließend nicht möglich, gilt die Studienleistung als *Nicht bestanden*.

Der Bearbeitungszeitraum für die Studienleistung beginnt mit Veröffentlichung dieser Aufgabenstellung und endet am 22.06.2020. Die Abgabe muss erfolgen bis zum **22.06.2020 00:00 Uhr MESZ**.

## Prüfung

Der minimale Funktionsumfang der finalen Abgabe ergibt sich aus der zu Beginn dargestellten Aufgabenstellung sowie den Anforderungen in den Abschnitten *Aufgabenstellung* und *Studienleistung*.

Hinsichtlich der Gestaltung sowie der Nutzerinteraktionsmechanismen gibt es keine weiteren Vorgaben oder Einschränkungen.

Die Bewertung der Prüfung erfolgt auf Grundlage der Abgabe sowie eines Abnahmegesprächs.

Der Bearbeitungszeitraum für die Prüfung beginnt ebenfalls mit Veröffentlichung dieser Aufgabenstellung und endet am 27.07.2020. Die Abgabe muss erfolgen bis zum **27.07.2020 00:00 Uhr MESZ**.

In der Woche ab dem 03.08.2020 finden die Abnahmegespräche statt. Genaue Termine werden individuell vereinbart. Das Abnahmegespräch findet entweder in den Räumlichkeiten der Hochschule oder in Form einer Videokonferenz statt. Halten Sie in beiden Fällen Ihren Studierendenausweis oder den Personalausweis bereit. Das Abnahmegespräch muss unabhängig von der Bewertung der Abgabe bestanden werden. Abwesenheit führt zu der Gesamtbewertung *Nicht bestanden*.

## Abgabe

Die Abgabe (sowohl der Studienleistung als auch der Prüfung) erfolgt via Stud.IP in den entsprechend benannten Ordner der Lehrveranstaltung. Sie umfasst das **bereinigte Projekt als ZIP-Datei (nicht als RAR-Datei o.ä.)**. Versehen Sie den Dateinamen mit Ihren **rzht-Benutzerkennungen (nicht mit Ihren Matrikelnummern)**, um eine eindeutige Zuordnung zu ermöglichen.

Eine fehlende, unvollständige oder verspätete Abgabe führt zur Gesamtbewertung der Studienleistung bzw. der Prüfung mit *Nicht bestanden*. Maßgeblich ist jeweils die Zeitangabe in Stud.IP.