Otto-Friedrich-Universität Bamberg
Lehrstuhl für Medieninformatik
Prof. Dr. Andreas Henrich
Leon Martin, Tobias Hirmer, Martin Bullin

VAWi Web-Technologien, Sommersemester 2022

Studienleistung 3

Hinweise

- Die vollständige und korrekte Bearbeitung einer Aufgabe ergibt die volle Punktzahl dieser Aufgabe; in Ausnahmefällen werden halbe Punkte verteilt.
- Für diese Studienleistung erwarten wir einen **Bearbeitungsaufwand von insgesamt etwa 60 Stunden**, da wir von drei Personen pro Team ausgehen. Dies ist als Richtwert zu verstehen und keinesfalls als Unter- oder Obergrenze.
- In Verdachtsfällen behalten wir uns vor, Plagiate von der Benotung auszunehmen. Wir werden in einem solchen Fall Kontakt mit den Betroffenen aufnehmen.
- *Programmieraufgaben* sollen in nachvollziehbar kommentierter Form im Quelltext abgegeben werden. Verwenden Sie keine anderen als die angegebenen Bibliotheken, soweit sie nicht zum Standardsprachumfang gehören. Nicht lauffähige Programme können nicht bewertet werden. Fügen Sie gegebenenfalls eine README-Datei hinzu, die beschreibt, wie man Ihre Lösung kompilieren bzw. ausführen kann.

Abgabe der Lösungen

- *Abgabetermin* **26. Juni 2022, 23:55 Uhr**
- *Abgabemodus* Einreichung auf der Online-Lernplattform im entsprechenden Kurs; der Server ist erreichbar unter `http://lms.vawi.de/vawi/`.
- *Dateiformat der Abgabe* Die Abgabe erfolgt durch den Upload eines ZIP-Archivs, das alle notwendigen Dateien beinhaltet. Der Dateiname des ZIP-Archivs sollte das Namensschema **nachname1_nachname2_nachname3_03.zip** einhalten.
- *Update der Lösung* Bis zur oben angegebenen Deadline können Sie Ihre Lösung beliebig oft durch eine neue (korrigierte) Fassung ersetzen. Bitte beachten Sie, dass dabei die zuvor hochgeladene Lösung überschrieben wird. Wir haben keine Möglichkeit, alte Fassungen wiederherzustellen!

Wir wünschen Ihnen viel Erfolg
bei der Bearbeitung der Studienleistung!

Aufgabe: Erstellung eines SPA zur Visualisierung von Studiengangsdaten (30 Punkte)

Mit Ihrem frisch erlerntem Wissen wollen Sie sich einen Überblick über die Verteilung des Modulangebots des Bachelor-Studiengangs Angewandte Informatik an der Universität Bamberg verschaffen. Dafür haben Sie nach einer passenden Datenquelle gesucht und sind auf eine JSON-Datei mit Daten über Modulgruppen und Module des Studiengangs gestoßen. Diese Daten wollen Sie nun mit einem `Express.js`-Server selbst bereitstellen und mit einer Webseite abrufen und anzeigen.

Für die Bearbeitung der Aufgabe wird eine zu verwendende Vorlage als ZIP-Datei bereitgestellt. Gehen Sie zur Nutzung der Vorlage wie folgt vor:

1. Entpacken Sie die ZIP-Datei.
2. Öffnen Sie eine Kommandozeile Ihrer Wahl im entpackten Ordner.
3. Installieren Sie die benötigten Node.js-Module mithilfe des Befehls `npm install`.
4. Client und Server lassen sich gemeinsam mittels `npm run start` starten. Der Client ist anschließend unter `localhost:4200` und der Server unter `localhost:3000` aufrufbar.

Aufgabenteil 1: Implementierung des Servers (10 Punkte)

Der Server soll mittels `Express.js` implementiert werden. Für den Code steht in der Vorlage der Ordner `./server` und die `./server/index.js`-Datei bereit. Die Daten, die vom Server bereitgestellt werden sollen, befinden sich in der `./server/data.json`-Datei. Diese Datei soll eingelesen und die enthaltenen JSON-Daten ggf. mit Array-Funktionen so aufbereitet werden, dass anschließend die folgenden zwei Routen für den Server implementiert werden können:

1. `/modules`: Diese Route soll die Module aus dem JSON-Datensatz zurückliefern.
2. `/module_groups`: Diese Route soll die Modulgruppen aus dem JSON-Datensatz zurückliefern.

In der Vorlage ist Express.js bereits eingebunden. Der Server alleine lässt sich mittels `npm run startServer` starten.

Tipp: Testen Sie die beiden von ihnen implementierten Routen bereits vor der Anbindung an die Webseite, indem Sie die Routen unter `localhost:3000` manuell im Browser aufrufen.

Aufgabenteil 2: Implementierung der Webseite (20 Punkte)

Die Webseite soll mittels `Bootstrap` und `d3.js` implementiert werden, welche in der Vorlage bereits eingebunden sind. Die Abbildungen 1 und 2 zeigen Mockups der Webseite. Wie in Abbildung 2 zu sehen ist, soll beim Hovern über einen Ausschnitt des Piecharts der Name der jeweiligen Modulgruppe per Tooltip angezeigt werden.

Für den Code steht in der Vorlage der Ordner `./client` bereit. Die anzuzeigenden Daten sollen vom `Express.js`-Server mithilfe der Fetch-API geladen werden. Das ist in der Datei `./client/js/index.js` in der Funktion `loadDataFromServer` zu implementieren. Beachten Sie bitte, dass es sich beim Server um eine nicht-eigene Ressource handelt und deswegen in den Fetch-Optionen der Modus `cors` gesetzt werden muss.

Zusätzlich zur Ansicht des Piecharts soll ein Impressum über das Navigationsmenü aufgerufen werden können. Bei der Auswahl eines Elements in der Navigationsleiste soll jedoch keine andere HTML-Datei aufgerufen werden, sondern lediglich der Inhalt der Webseite ausgetauscht werden.

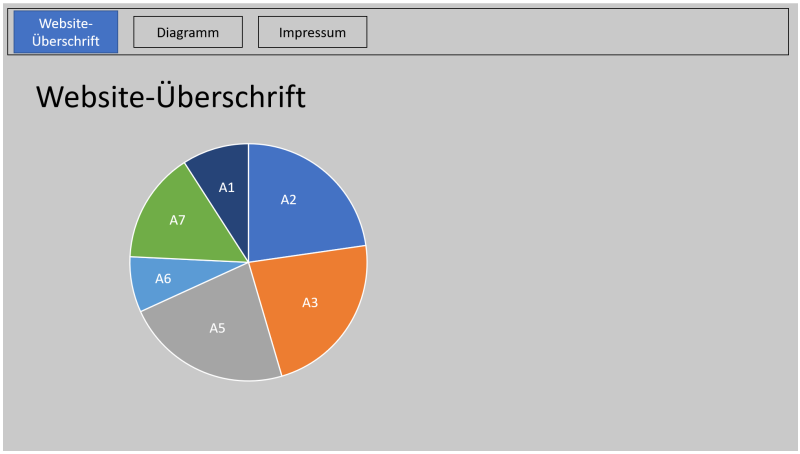

Abbildung 1: Die Webseite, wenn die Ansicht des Diagramms ausgewählt ist.

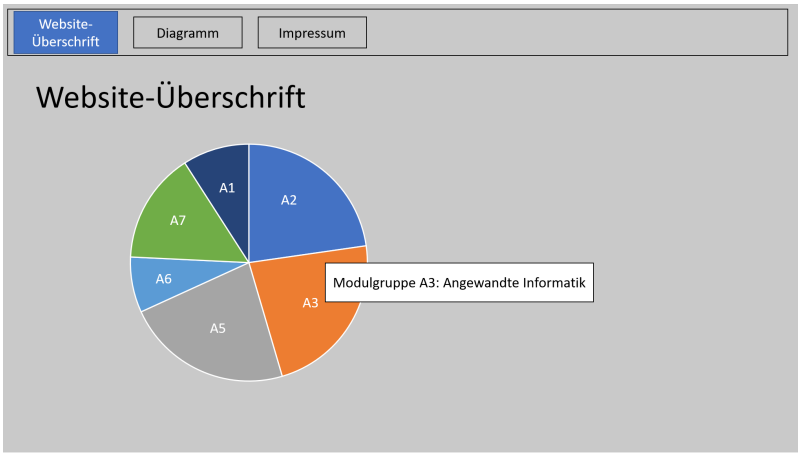

Abbildung 2: Beim Hovern über einen Ausschnitt.

Die beiden Mockups dienen zur Veranschaulichung der Struktur und Funktionalität der Webseite. Bei der Gestaltung (Farben, Fonts, etc.) haben Sie freie Hand.

Hinweis: Die beiden Aufgabenteile können auch in umgekehrter Reihenfolge oder parallel bearbeitet werden. Damit die Webseite unabhängig vom Server gestartet und verwendet werden kann, muss zuvor die Funktion `loadDataFromFile` in der Datei `./client/index.js` implementiert werden, um die Datei `./client/data-backup.json` einzulesen. Ziel ist es jedoch, dass die Daten am Ende vom Server abgefragt und nicht aus der Backup-Datei eingelesen werden.

Weitere Hinweise zur Teilleistung

- Nutzen Sie die Möglichkeiten der Konsolenansicht in den Entwicklungswerkzeugen von Firefox und Chrome zum Debuggen Ihres JavaScript- bzw. TypeScript-Codes.
- Kapseln Sie zusammengehörige Funktionalität in Klassen und Komponenten, um Ihren Code besser zu strukturieren und die Wiederverwendbarkeit von Code-Ausschnitten zu erhöhen.
- Geben Sie in einer README-Datei an, ob Sie Ihre Umsetzung mit Firefox oder Google Chrome getestet haben. Dokumentieren Sie in dieser README-Datei bitte auch weitere Besonderheiten, z.B. wenn Ihre Umsetzung unvollständig ist oder sich unter bestimmten Umständen fehlerhaft verhält.
- Ihr Abgabe-ZIP-Archiv muss alle zur Ausführung notwendigen Dateien beinhalten, nicht nur solche, die Sie neu erstellt haben.