Übungsblatt 11: Servertechnologien

Vorstellung in den Tutorien am 24. - 26. Januar 2022

11.1 Node.js: Einführung

Letzte Woche haben wir besprochen, wie man mittels HTTP mit einem lokal laufenden Webserver kommunizieren kann. Nun wollen wir diesen Ansatz erweitern, indem wir unseren eigenen Server mit Node.js implementieren. In der Vorgabe finden Sie in der Datei `server.js` einen teilweise implementieren Node.js Server. Machen Sie sich mit seiner Funktion vertraut und ergänzen Sie ihn anschließend um folgende Funktionalitäten:

1. Mittels der `GET`-Methode können Eintrage ausgelesen werden. Wird eine konkrete `id` als Query-Parameter übergeben, so wird nur der entsprechende Eintrag zurückgegeben. Ansonsten werden alle Einträge als Array zurückgegeben.
2. Mittels der `POST`-Methode werden neue Einträge angelegt. Hierfür werden im Request-Body der Name des Moduls (`modul`) und des Professors (`prof`) sowie die Leistungspunkte des Moduls (`ects`) im *x-www-urlencoded* Format angeben.
   - Bei Erfolg wird Statuscode 201 zurückgegeben.
   - Falls in der Anfrage kein `modul`, `prof` oder `ects` übergeben wurde, wird Statuscode 400 zurückgegeben.
3. Mittels der `PUT`-Methode können Einträge modifiziert werden. Dazu muss im Request-Body die id des zu modifizierenden Eintrags sowie die neuen Werte für `modul`, `prof` und `ects` im *x-www-urlencoded* Format angeben werden.
   - Bei Erfolg wird Statuscode 200 zurückgegeben.
   - Existiert kein Eintrag mit der angegebenen `id`, wird Statuscode 404 zurückgegeben.
   - Falls in der Anfrage kein `modul`, `prof` oder `ects` übergeben wurde, wird Statuscode 400 zurückgegeben.
4. Mittels der `DELETE`-Methode wird der Eintrag mit der `id` gelöscht, die als Parameter der Anfrage übergeben wird.
   - Wird keine `id` angeben, wird kein Eintrag gelöscht und Statuscode 400 zurückgegeben.
   - Bei erfolgreicher Löschung wird Statuscode 200 zurückgegeben.
5. Für jede andere HTTP-Methode wird Statuscode 405 zurückgegeben.

Testen Sie den Server, indem Sie mit *Postman* entsprechende Anfragen an ihn senden.

11.2 nginx

Richten Sie nun einen *nginx*-Webserver ein, der, genau wie in der Vorlesung vorgestellt, als *Reverse Proxy* für den zuvor implementierten Node.js-Anwendungsserver fungiert. Installieren Sie dazu zunächst nginx auf Ihrem Gerät:

Windows

1. Laden Sie sich nginx Version 1.23.3 für Windows herunter: `https://nginx.org/download/nginx-1.23.3.zip`
2. Entpacken Sie das heruntergeladene ZIP-Archiv in einen Ordner Ihrer Wahl.

Linux (Ubuntu)

1. Aktualisieren Sie ggf. das Verzeichnis Ihres Package Managers mit dem Befehl `apt-get update`
2. Nutzen Sie `apt-get`, um nginx zu installieren: `apt-get install nginx`

MacOS

1. Installieren Sie ggf. Homebrew, indem Sie folgenden Befehl im Terminal eingeben:

   ```
   /usr/bin/ruby -e "$(curl -fsSL
   https://raw.githubusercontent.com/Homebrew/install/master/install)"
   ```

2. Nutzen Sie Homebrew, um nginx zu installieren, indem Sie folgenden Befehl im Terminal eingeben: `brew install nginx`

In Unix-basierten System, also Linux und MacOS, ist der Installationsordner von nginx unter dem Pfad `/usr/local/etc/nginx/` zu finden. Sie können nginx jedoch von überall in Ihrem System über das Terminal starten. In Windows ist der Installationsorder der Ordner, in den Sie das ZIP-Archiv entpackt haben. Um nginx im Windows-Terminal ausführen zu können, müssen Sie sich im Installationsordner befinden oder die PATH-Umgebungsvariable um den Pfad zum Installationsordner ergänzen.

Sie können nginx mittels folgender Befehle steuern:

`nginx` Server starten

`nginx -s stop` Server schnell beenden

`nginx -s quit` Server kontrolliert herunterfahren

`nginx -s reload` Serverkonfiguration neu laden

Die Konfiguration von nginx erfolgt mittels der im Installationsordner befindlichen *nginx.conf* Datei. Um nginx wie gewünscht als Reverse Proxy für unseren Node.js-Anwendungsserver zu nutzen, muss diese Datei angepasst werden. Ersetzen Sie dazu in der Datei den vorgefertigten `server`-Block durch folgenden Code:

```
server {
  listen 8080;
  location / {
    proxy_pass http://localhost:3000;
  }
}
```

Starten Sie nun den nginx-Server und testen Sie ihre Anwendung, indem Sie mittels Postman Anfragen an die URL `http://localhost:8080` senden. Sie sollten nun die selben Antworten wie zuvor unter Port 3000 erhalten.