Prof. Dr. Sebastian von Klinski, klinski@beuth-hochschule.de Raum B230a

ÜBUNG 1 – MEILENSTEIN 2			
LV	**WEB-ENGINEERING II**		
THEMA	**REST-SERVER**	FÄLLIGKEIT	**SIEHE MOODLE**

ÜBUNG 1, MEILENSTEIN 2 - VORGABEN

Grundlage für die Umsetzung und Benotung dieses Meilensteins sind die „Allgemeinen Regelungen für Übungsaufgaben“ sowie die Aufgabenstellung für Übung 1, die Sie beide im Moodle finden.

In diesem Dokument werden sowohl die funktionalen als auch allgemeinen Anforderungen an den Meilenstein 2 von Übung 1 weiter detailliert. Bitte beachten Sie, dass alle nachfolgenden Vorgaben exakt eingehalten werden müssen, weil viele Aspekte über automatisierte Tests geprüft werden.

Die Vorgaben für Meilenstein 1 gelten auch für diesen Meilenstein! Insbesondere muss der Endpoint „/publicUsers“ immer noch uneingeschränkt funktionieren. Sollten Sie die Anforderungen von Meilenstein 1 bei der letzten Abgabe nicht alle erfüllt haben, müssen sie zu diesem Meilenstein umgesetzt sein.

FUNKTIONALE ANFORDERUNGEN

Setzen Sie die Endpoints für die Authentifizierung, für User und Studiengänge entsprechend der Aufgabenstellung um.

Authentifizierung

Der Endpoint unter der URL „./api/authenticate“ soll die Authentifizierung per Basic-Authentication umsetzen. Halten Sie dabei sowohl bei der Anfrage als auch bei der Antwort den Basic-Authentication-Standard ein, so wie er in den Vorlesungsfolien „Best-Practices bei Umsetzung von REST-Services“ und der Aufgabenstellung von Übung 1 beschrieben wird.

```
GET http://localhost/api/authenticate
Authorization: Basic YWRtaW46MTIz
```

Abbildung 1: GET-Request für Authentifizierung

Damit der Server genutzt werden kann, auch wenn am Anfang die Datenbank leer ist, sollte beim Hochfahren automatisch der Standardadministrator angelegt werden (User-ID „admin“, Passwort „123“), falls es den nicht schon gibt.

Bei einer erfolgreichen Authentifizierung sollte der Token entsprechend den Vorlesungsfolien im Header zurückgegeben werden.

Abbildung 2: Antwort bei erfolgreicher Authentifizierung

Im Fehlerfalle sollte eine entsprechende Fehlernachricht zurückkommen. Achten Sie darauf, dass auch der entsprechende http-Status-Code zurückgegeben wird.

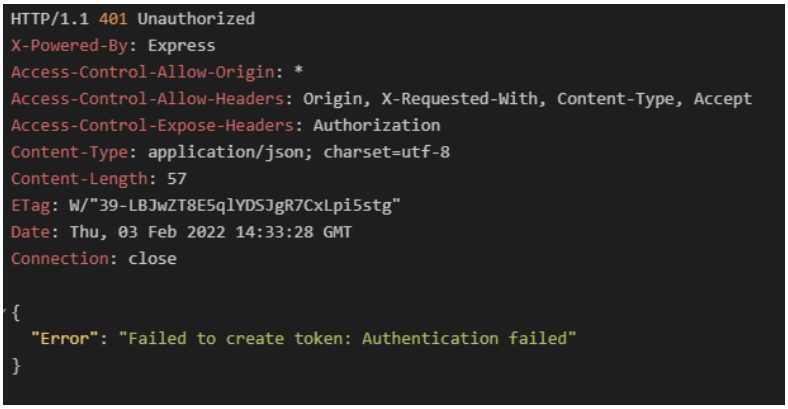

Abbildung 3: Antwort bei gescheiterter Authentifizierung

Im Payload vom Token sollte in jedem Fall die User-ID sein. Bei allen Anfragen mit dem Token sollte der User, der die betreffende Aktion ausführt, anhand des Payloads bestimmt werden.

Ergänzend dazu können auch weitere Informationen im Token abgelegt werden, wie beispielsweise die Information, ob der User ein Administrator ist oder wie der Vor- und Nachnamen sind. Alternativ können Sie diese Daten aber auch im Body von der Response zurückgeben (siehe nachfolgende Abbildung).

Das Übertragen dieser Daten mit der Authentifizierung kann insbesondere bei der Umsetzung der Web-Anwendung in Übung 2 hilfreich sein, weil dann diese Daten nicht über einen zweiten Request abgerufen werden müssen.

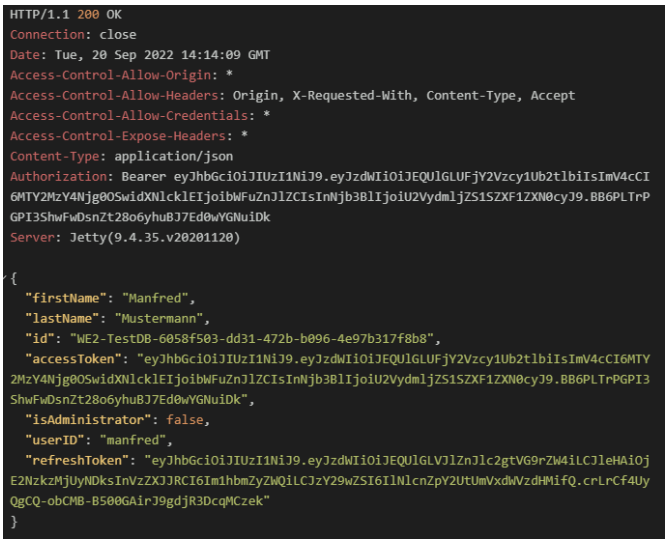

Abbildung 4: Authentication-Response mit ergänzenden Daten im Body

User-Management

Setzen Sie den User-Endpoint um, der unter der URL „/api/users“ erreichbar sein soll. Im Gegensatz zum Public-User-Endpoint soll nun bei allen Anfragen über den Token im Header geprüft werden, ob der User sich korrekt authentifiziert hat.

Beachten Sie die Vorgaben zu Autorisierung aus der Aufgabenstellung von Übung 1. Hierzu gehören unter anderem: Nur User, die Administratoren sind (isAdministrator ist true), dürfen den vollen Zugriff auf die User-Daten haben. User, die keine Administratoren sind, dürfen nur die eigenen User-Daten abrufen. Sie dürfen den eigenen Vor- und Nachnamen ändern, nicht jedoch die Eigenschaft „isAdministrator“. Die User-ID sollte generell nicht geändert werden können.

Alle Routen des Users-Endpoints dürfen nur noch jene Daten vom User zurückgeben, die problemlos über das Netz übertragen werden können. Insbesondere das Passwort muss aus der Antwort entfernt werden. Es können auch andere Systemdaten wie Änderungsdatum entfernt werden.

In den Express-Routen sollte die Prüfung, ob der User eingeloggt und ein Administrator ist, über eine Middleware-Funktion umgesetzt werden (siehe Vorlesungsfolien). Wenn beispielsweise ein User „manfred“ die User abrufen will und manfred kein Administrator ist, sollte eine entsprechende Fehlermeldung sowie der Status-Code 401 zurückgegeben werden.

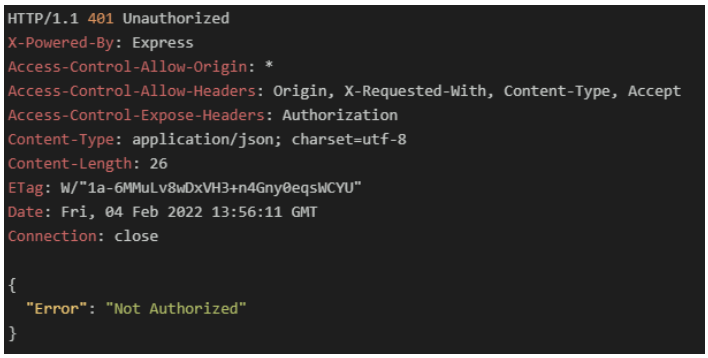

Abbildung 5: Fehlermeldung, wenn ein User, der nicht Administrator ist, alle User abrufen will.

Zum Abrufen eines konkreten Users sollten Sie die einfache Suchfunktion entsprechend der Aufgabenstellung Übung 1 umsetzen.

```
GET http://localhost/api/users/manfred
Authorization: {{adminToken}}
```

Abbildung 6: Abrufen des Users „manfred“

Die Antwort auf diese Anfrage ist entsprechend:

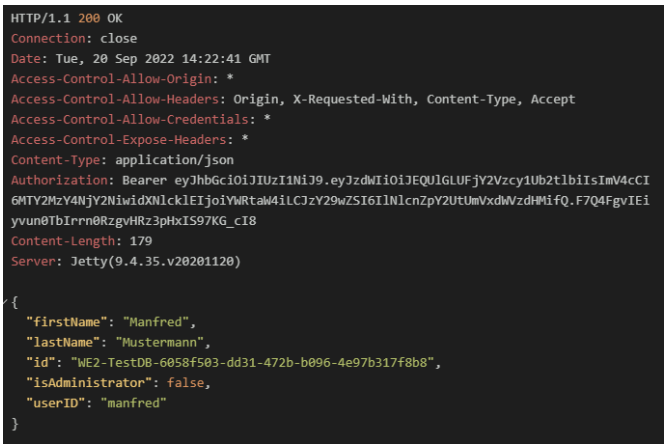

Abbildung 7: Antwort auf die Anfrage zum Abrufen des Users „manfred“

Studiengang-Endpoint (DegreeCourse)

Setzen Sie auch den Studiengang-Endpoint mit den CRUD-Methoden entsprechend den Best-Practices für REST-Services um. Die Resource-ID für diesen Endpoint ist „degreeCourses“. Der Request zum Abrufen aller Studiengänge ist dementsprechend:

```
### Auflisten aller Studiengänge
GET http://localhost/api/degreeCourses
Authorization: {{adminToken}}
```

Abbildung 8: Abrufen des Users „manfred“

Bei den Antworten sollten dann stets alle Daten des Studiengangs zurückgegeben werden.

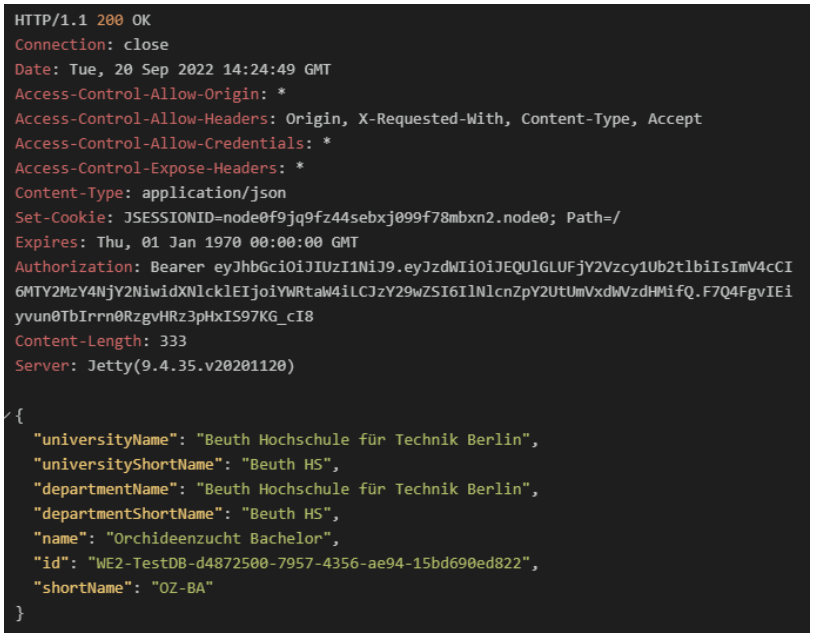

Abbildung 9: Rückgabe der Studiengangdaten nach dem Anlegen

Zusätzlich zu den CRUD-Methoden soll der Studiengang-REST-Service die Such-Route umsetzen, um alle Studiengänge einer Hochschule abzurufen.

```
### Auflisten der Studiengänge einer bestimmten Hochschule
GET http://localhost/api/degreeCourses?universityShortName=Beuth HS
Authorization: {{adminToken}}
```

Schreibende Zugriffe auf Studiengangdaten dürfen nur Administratoren nutzen, lesende Zugriffe dürfen alle User.

ALLGEMEINE ANFORDERUNGEN

Es gelten die gleichen allgemeinen Anforderungen, die bereits für den Meilenstein 1 aufgeführt wurden. Es soll insbesondere bei diesem Meilenstein noch http und Port 80 verwendet werden. Bitte achten Sie darauf, dass noch alle Funktionen von Meilenstein 1 korrekt funktionieren und beide Endpoints, /users und /publicUsers, auf die gleichen Daten zugreifen.

TESTS ZUR PRÜFUNG DER FUNKTIONEN/ ENDPOINTS

Im Moodle finden Sie wieder eine Test-Datei, in der alle wesentlichen Anfragen aufgelistet sind, die Ihr REST-Server bei diesem Meilenstein erfüllen muss.

Viel Erfolg!