Projektaufgabe TINF15B

Deadlne:
Letzter Push: 20.12.2016
Prüfung gibt 60 Punkte. (Diese sind auf die Features verteilt)
Verteilung:

Funktionalitäten: 50 Punkte
Code: 10 Punkte

Aufgabenstellung:
Um eine Webseite für Schüler einer bestimmten Schule zu implementieren wird ein Backend benötigt:
Zu implementieren ist also eine Web-Api welche Daten im JSON-Format an einen Client sendet. **(Eine Implementierung eines Frontend, d.h. HTML/ CSS ist nicht notwendig).** Die Daten werden aus einer MySQL-Datenbank ausgelesen. Diese ist selbst anzulegen, die Daten, sowie das Datenbankschema werden allerdings zur Verfügung gestellt.
Zusätzlich soll mit Hilfe einer Middleware und einem JSON-Webtoken die User Authentifizierung und Autorisierung stattfinden. Hierbei gibt eine Schnittstelle die einen Login entgegennimmt, einen Webtoken zurück an den Clien. Dieser Token wiederrum wird bei jedem Request mitgeliefert.

Anforderungen Technologien:
- NodeJS / Express
- JSON Web-Token
- .eslintrc
- Anbindung an eine MySQL Datenbank

Funktionale Anforderungen (50 Punkte):
- Daten sollen mithilfe einer Web-API zur Verfügung gestellt werden./ Implementierung der Routen **(30 Punkte)**
- Hierbei sollen die Daten in JSON an einen Client gesendet werden. **(5 Punkte)**
- Eine Authentifizierung und Autorisierung soll mithilfe eines JSON-WebToken stattfinden **(10 Punkte)**
- Es soll eine Login Funktionalität in dem Backend vorhanden sein **(5 Punkte)**

Die Web-Api Routen:
<u>Routen welche für den Schüler:</u>

- **Login: PUT** /api/V1/login

Input: Es wird ein Passwort und Username gesendet
Output: Ein JSON-WebToken

- **Profil löschen: DELETE** /api/V1/student

Input: Ein JSON-WebToken
Output: Success oder Fail Meldung

- **Abfrage des Schülers: GET** /api/V1/student

Input: Ein JSON-WebToken
Output:

```
studentSchema = {
+        forename: {type: 'string', required: true},
+        surname: {type: 'string', required: true},
+        studyGroups: {
+            type: 'array',
+            required: true,
+            class: {type: 'string', required: true},
+            imageUrl: {type: 'string', required: true},
+            imageUrlInactive: {type: 'string', required: true},
+            imageUrlBig: {type: 'string', required: true}
+        },
+        formteacher: {type: 'string', required: true},
+        school : {
+          type: 'object',
+          required: true,
+          properties: {
+            name: {type: 'string', required: true},
+            address: {type: 'string', required: true},
+            country: {type: 'string', required: true},
+            email: {type: 'string', required: true},
+            telefon: {type: 'string', required: true},
+            imageUrl: {type: 'string', required: true},
+            imageUrlInactive: {type: 'string', required: true},
+            imageUrlBig: {type: 'string', required: true}
+          }
+        },
+        avatarId: {type: 'string', required: true}
+    }
```

- **Schüler Passwort ändern: PUT** /api/V1/passwordRecovery/reset

Input: Ein JSON-WebToken, neues Passwort
Output: Success oder Fail Meldung und ein neuer JSON-WebToken

- **Schüler ändert sein Avatar: PUT** /api/V1/avatar/:avatarId

Input: Ein JSON-WebToken
Output: Success oder Fail Meldung

- **Abfrage aller Avatare GET** /api/V1/avatar/

Output (Beispiel):

```
[{
    "_id" : 0,
    "avatarUrl" : "/images/student/superhero-batman-active.png",
    "avatarInactiveUrl": "/images/student/superhero-batman-inactive.png",
    "avatarBigUrl":  "/images/student/superhero-batman-big-active.png",
},
{
    "_id" : 1,
    "avatarUrl" : "/images/student/superhero-catwoman-active.png",
    "avatarInactiveUrl": "/images/student/superhero-catwoman-inactive.png",
    "avatarBigUrl":  "/images/student/superhero-catwoman-big-active.png",
}
....
....
]
```

Kompetenzen die ein Schüler erreichen kann/ erreicht hat:

Eine Kompetenz ist einfach nur eine Fähigkeit welcher ein Student durch eine Prüfung erlangen kann. Kompetenzen sind noch in Kapiteln Kategorisiert (Mathematik/ Biologie/ Körper).

- **GET** /api/V1/studentcompetence/?checked='true'&chapterid='chapterId''&paging='count'

Input (via QueryString:)

- checked(**true/false**): Information ob nach dem Kriterium checked gefiltert wird.
- chapterId(**1-15**): Die Id des Kapitels nach was gefiltert werden soll
- Paging(**10 / 20**): Ein Paging welches angibt wieviele Kompetenzen auf einmal von dem Server geladen werden sollen

Output:
[{

```
        competenceId: {type: 'integer', required: true},
        number: {type: 'string', required: true},
        version: {type: 'int', required: true},
        studentText: {type: 'string', required: true},
        teacherText: {type: 'string', required: true},
        description: {type: 'string', required: true}
}
...]
```

Description: Ist der Name der Kompetenz
studentText und *teacherText* sind lediglich Beschreibungen der Kompetenz. Einmal so, dass es ein Grundschüler versteht und einmal eine Beschreibung, so, dass es ein Lehrer versteht ;)