Aplikovaná informatika, 3.r Šk. rok 2021/2022 21.4.2022 Róbert Kica

Pokročilé programovacie techniky

**Semestrálna úloha: Tvorba backendu cez Nest.Js**

# Zadanie:

Vytvorte REST API vo frameworku NestJS pre aplikáciu, ktorá bude slúžiť na vytvorenie výučbových kurzov.

Obsah

[1 Zadanie: 1](#_Toc101425100)

[Obsah 1](#_Toc101425101)

[2 Popis Riešenia: 2](#_Toc101425102)

[3 Moduly: 4](#_Toc101425103)

[4 Endpointy: 5](#_Toc101425104)

[5 Príklady requestov: 8](#_Toc101425105)

[6 Záver: 15](#_Toc101425106)

# Popis Riešenia:

Projekt bol robený v prostredí Jetbrains Webstorm. Požiadavky na server boli taktiež v Webstorm, cez Http Client.

Databáza robená v postrgeSQL 14 resp. pgAdmin4 na sledovanie databázy. Používali sme Repositoty pattern, každý modul mal potrebné repository napr. typu <CoursesEntity>.

Použité moduly npm: nodemon, typeorm,, config, postgresql,

**Zabezpečenie autorizácie:**

Rozhodli sme sa nepoužiť passport, bscrypt a podobné knižnice. Pri prihlásení sa používateľa(cez požiadavku Post - email) je uložený jeho profil v premennej a cez ňu potom vieme kto je prihlásený. Znamená to však, že všetky akcie na serveri budú vykonané na tohto používateľa dokým sa neprihlási iný.

Autorizácia je vykonaná cez Guards dekorátorov.

V **global.service.ts** máme uloženého prihláseného používateľa

import { UserRole } from './users/entities/user.entity';

export class GlobalService {

static loggedUser: any;

static role: UserRole;

}

**Roles.decorator.ts**

import { UserRole } from './entities/user.entity';

export const ROLES\_KEY = 'roles';

export const Roles = (...roles: UserRole[]) => SetMetadata(ROLES\_KEY, roles);

**Overenie role:**

Export class RolesGuard implements CanActivate {

constructor(private reflector: Reflector) {}

canActivate(context: ExecutionContext): boolean {

const requiredRoles = this.reflector.getAllAndOverride<UserRole[]>(

ROLES\_KEY,

[context.getHandler(), context.getClass()],

);

if (!requiredRoles) {

return true;

}

//const { user } = context.switchToHttp().getRequest();

//return requiredRoles.some((role) => user.roles?.includes(role));

const roles = [GlobalService.role, null];

console.log(roles);

console.log(requiredRoles);

//return requiredRoles.some((role) => roles?.includes(role));

return requiredRoles.some((role) => GlobalService.role);

}

**Pripájanie na databázu:**

.development.env má connection parametre. Následne src/persistence/ **typeorm-config.service.ts** jeinjectnutý ako configservice v **Appmodule** cez **ConfigModule.forRoot**.

# Moduly:

Všeobecná štruktúra modulu:

Obrázok, na ktorom je text

Automaticky generovaný popis

**Courses** – správa kurzov, hľadanie kurzov, vytváranie kurzov

**Lectures**- tvorba lekcii a priradenie k kurzom

**Persistence**- napojenie na databázu

**Profile**- študentov profil

**Students**- vytváranie a prehľadávanie študentov

**Teachers**- vytváranie a prehľadávanie učiteľov

**Users**-dekorátor roly a guard

# Endpointy:

**Courses** : endpointy:

|  |  |  |  |
| --- | --- | --- | --- |
| Endpoint | metóda | Popis | Telo @Body |
| /courses | POST | Vytvorí kurz ak má oprávnenía Teacher | CreateCouseDto:  title: string;  slug: string; description: string; |
| /courses | GET | Vráti všetky existujúce kurzy | - |
| /courses/:id | GET | Nájde kurz podľa slugu | - |
| /courses/:slug | DELETE | Odstráni vybratý kurz |  |

**Lectures** : endpointy:

|  |  |  |  |
| --- | --- | --- | --- |
| Endpoint | metóda | Popis | Telo @Body |
| /lectures | POST | Vytvorí lekciu ak má oprávnenia Teacher | CreateLectureDto:  title: string;  slug: string; courseSlug: string; description: string;  url: string; |
| /lectures/ | GET | Vypíše všetky exitujúce lekcie | - |
| /lectures/:id | GET | Vráti lekciu podľa slug | - |
| /lectures/ c/:courseSlug | GET | Vypíše všetky lekcie patriace danému kurzu | courseSlug ako parameter id |
| lectures/ /id | Delete | Odstráni vybratú lekciu | - |

**Profile**

|  |  |  |  |
| --- | --- | --- | --- |
| Endpoint | metóda | Popis | Telo @Body |
| /profile | POST | Vytvorí profil | CreateProfileDto: |
| /profile | GET | Vráti všetky existujúce profily | - |
| /profile/:email | Patch | Vráti profil podľa email parametra | - |
| /profile/:email | Delete | Odstráni vybratý profil |  |

**Students**

|  |  |  |  |
| --- | --- | --- | --- |
| Endpoint | metóda | Popis | Telo @Body |
| /students | POST | Vytvorí študenta | createStudentDto  firstName: string;  lastName: string; email: string;  role: UserRole |
| /students/c/ | POST | Študent sa môže prihlásiť do kurzu-v tele | joinCourseDto: courseSlug: string; studentEmail: string; |
| /students/c/unroll | POST | Študent sa odhlási z kurzu | joinCourseDto: courseSlug: string; studentEmail: string; |
| /students | GET | Vráti všetky existujúce kurzy | - |
| /students/:email | DELETE | Odstráni vybratého študena | - |
| /students/auth/:id | Post | Zadá prihláseného používateľa | email |

**Teachers**-

|  |  |  |  |
| --- | --- | --- | --- |
| Endpoint | metóda | Popis | Telo @Body |
| /teachers | POST | Vytvorí učiteľa | CreateTeacherDto:  firstName: string;  lastName: string; email: string;  role: UserRole |
| /teachers | POST | Študent sa môže prihlásiť do kurzu-v tele | joinCourseDto: courseSlug: string; studentEmail: string; |
| /teachers | GET | Vráti všetky existujúce kurzy |  |
| /teachers | Patch | Upraví zadaný kurz | UpdateCouseDto:  title: string;  slug: string; description: string; |
| /teachers/:email | Delete | Odstráni vybraného učiteľa | - |
| /teachers/auth/:id | Get | Prihlási učiteľa, zadá potrebné oprávnenia | - |

# Príklady requestov:

V priešinku WebstormRequests máme uložené testovacie requesty.

V Html Client sme si natavili variable host = http://localhost:3000

### test aplikacie

GET http://localhost:3000

Accept: application/json

###

# @name GetAllCourses

GET {{host}}/courses

###

# @name LoginTeacher

GET {{host}}/teachers/auth/teacher1@email.com

###

# @name NewCourse

POST {{host}}/courses

content-type: application/json

{

"title": "REST API for beginners2",

"slug": "REST-API-for-beginners2",

"description": "2. Nestjs course..."

}

###

# @name NewCourse2

POST {{host}}/courses

content-type: application/json

{

"title": "New Course 2",

"slug": "another-course",

"description": "2. Nestjs course..."

}

###

# @name NewCourse3

POST {{host}}/courses

content-type: application/json

{

"title": "New Course 3",

"slug": "course3",

"description": "3. Nestjs course..."

}

###

# @name DelCourse

DELETE {{host}}/courses/course3

content-type: application/json

###

# @name GetOwnedCourses

GET {{host}}/teachers/courses

###

# @name GetCourseBySlug

GET {{host}}/courses/REST-API-for-beginners2

###

# @name AddLectureToCourse0

POST {{host}}/lectures

content-type: application/json

{

"courseSlug": "REST-API-for-beginners2",

"slug": "lecture-01",

"title": "FirstLecture",

"description": "email@email4.com",

"url": "www.youtu.be/watch=1235"

}

###

# @name AddLectureToCourse3

POST {{host}}/lectures

content-type: application/json

{

"courseSlug": "course3",

"slug": "lectureFor3-01",

"title": "LectureS03E01",

"description": "email@email4.com",

"url": "www.youtu.be/watch=1235"

}

###

# @name AddLectureToCourse3

POST {{host}}/lectures

content-type: application/json

{

"courseSlug": "another-course",

"slug": "another-course-lecture",

"title": "SecondLectureAgain",

"description": "lalalaa",

"url": "www.youtu.be/watch=404040"

}

###

# @name GetLecuresByCourseSlug

GET {{host}}/lectures/c/another-course

###

# @name GetLecuresByCourseSlug2

GET {{host}}/lectures/c/REST-API-for-beginners2

###

# @name GetLectures

GET {{host}}/lectures

###

# @name NewStudent

POST {{host}}/students

content-type: application/json

{

"firstName": "Json",

"lastName": "Api",

"email": "email@email4.com",

"role": 1

}

###

# @name StudentJoinCourse

POST {{host}}/students/c/

content-type: application/json

{

"courseSlug": "another-course",

"studentEmail": "email@email4.com"

}

###

# @name StudentUnrollCourse

POST {{host}}/students/c/unroll

content-type: application/json

{

"courseSlug": "another-course",

"studentEmail": "email@email4.com"

}

###

# @name NewTeacher

POST {{host}}/teachers

content-type: application/json

{

"firstName": "Mr",

"lastName": "Teacher",

"email": "teacher1@email.com",

"role": 2

}

<> 2022-04-20T102408.500.json

###

# @name NewTeacher2

POST {{host}}/teachers

content-type: application/json

{

"firstName": "Mr",

"lastName": "Teacher2",

"email": "teacher2@email.com",

"role": 2

}

###

# @name GetTeachers

GET {{host}}/teachers

###

# @name GetStudent

GET {{host}}/students/4

###

# @name LoginStudent

GET {{host}}/students/auth/email@email1.com

# Záver:

Zadanie je splnené ale kvalita nie je taká dobrá. Projekt má veľa častí ktoré by sa mohli vylepšiť. Samotná autentifikácia by mala byť spravená poriadne cez JWT+ nejaký role systém. Databáza je nastavená na development, čiže máme zapnutú synchronizáciu, čo ale nevadí pri vývoji. Niektoré endpointy sú zmiešané, funkcionalita sa trochu mieša. Pridať patch funkcionalitu.