Noviaal - NOVI Backend 2020

Instelling	NOVI HBO Software Development
Leerlijn	BackEnd
Document	Functioneel en Technisch Ontwerp voor de Eindopdracht
Docent	Nick Stuivenberg
Datum	20 november 2020
Auteur	Jurjen Vorhauer
Email	J.Vorhauer@novi-education.nl
Studentnummer	800009793
GitHub	https://github.com/jvorhauer/noviaal
Noviaal	Een Twitterig platform, de naam is een samentrekking van NOVI en Sociaal

Table of Contents

1. Inleiding

Noviaal is een applicatie die alleen REST endpoints presenteert aan de buitenwereld. Het is een applicatie die aan alle voorwaarden voor de eindopdracht van de module "Backend 2020/2021" van NOVI HBO Software Development voldoet.

Noviaal is een *platform* om:

- notities en media op te slaan,
- om die notities en media te kunnen ophalen, lezen en taggen.
- om andere Noviaal gebruikers te volgen
- om direct geïnformeerd te worden over nieuwe notities en media van die gevolgde gebruikers.

De doelgroep is **iedereen** op het internet.

2. Installatie

Het installeren van Noviaal kan met de volgende stappen gerealiseerd worden:

***NB**: Voor alle installatie stappen is een werkende breedband internet verbinding nodig* ***NB**: in de opdracht staat dat de applicatie binnen 30 minuten na een schone start van het installeren moet draaien, maar de doorlooptijd is van zo veel zaken afhankelijk, dat die voorwaarde nooit te garanderen is*

1. Installeer **Docker**
   a. Windows: Instructies
   b. Mac: Instructies
2. Start Docker
   a. Windows: Gebruiksaanwijzing
   b. Mac: Gebruiksaanwijzing
3. Lees de "Getting Started" van Docker: Docker Getting Started
4. Installeer de Java Development Kit (JDK), versie **11**
   a. Selecteer de juiste (Windows/Mac/Linux) versie van OpenJDK 11: AdoptOpenJDK
   b. Voer de stappen uit, zoals beschreven
   c. Test of de installatie geslaagd is door in een Terminal of Command Process/Powershell `java -version` uit te voeren.
5. Installeer `git`, de versie beheer client die gebruikt wordt:
   a. Windows: Downloading Git
   b. Mac: zie Getting Started en dan kopje "Installing on macOS"
6. Met de geinstalleerde `git` client is het nu mogelijk om het project uit te checken: `git clone https://github.com/jvorhauer/noviaal.git`, maar alleen als de ingelogde gebruiker ook toegang

heeft tot dit project. Om toegang te krijgen volstaat een email aan Jurjen Vorhauer, de auteur en eigenaar van dit project.

   a. *alternatief* : download het ZIP bestand dat ingeleverd werd via Teams. Pak dit ZIP bestand uit.

7. Na het succesvol clonen van het project of uitpakken van het ZIP bestand kan het gebouwd worden:

   a. ga naar de directory/folder met het project: `cd noviaal` vanuit de directory/folder van waaruit het `git clone` commando is uitgevoerd.

   b. voer `./mvnw verify` (mac/linux) of `mvnw.cmd verify` (windows) uit, zodat zeker is dat JDK en Maven installaties correct zijn uitgevoerd. **NB**: de eerste keer bouwen met Maven neemt veel tijd, aangezien alle gebruikte bibliotheken (afhankelijkheden benodigd voor de bouw) van Internet opgehaald worden.

   c. voer nu `./mvnw package` of `mvnw.cmd verify` uit, er zal nu een Java ARchive (JAR) bestand in de `target` directory/folder staan: `noviaal-1.0.0.jar`. Dit is de Noviaal applicatie.

8. Laatste stap: voer `docker-compose up --build` uit. Zodra het PostgreSQL Docker image gedownload is, de Docker image van de Noviaal applicatie gebouwd is de en twee containers op basis van die images draaien, is Noviaal klaar voor gebruik.

3. Gebruik

Voor het benaderen van de endpoints van de Noviaal applicatie is het uitgangspunt dat de bovenstaande installatie stappen compleet en succesvol zijn uitgevoerd.

De endpoints kunnen met een aantal tools of clients worden benaderd:

1. `Postman` is een veelgebruikte REST client
2. `IntelliJ IDEA, Ultimate Edition` heeft een meegeleverde en ingebouwde REST client.

Voor IntelliJ zijn een aantal request bestanden meegeleverd in de directory/folder `src/test/requests`. De requests in deze bestanden zijn eenvoudig uit te voeren door op het groene driehoekje voor een specifiek request te klikken en te kiezen voor `Run https://localhost:8080`. De `request` bestanden eindigen in (hebben extensie) `.http`.

Voor Postman is een collectie bewaard, in `src/test/requests/Noviaal.postman_collection.json`. Bij die collectie hoort ook een Postman environment bestand: `src/test/requests/Noviaal.postman_environment.json`.

Op basis van deze voorbeelden kunnen de endpoints aangestuurd worden en kunnen eenvoudig meer requests gedefinieerd en uitgevoerd worden.

Zowel de IntelliJ als de Postman requests slaan het token van de laatste inlog actie op in een variabele. Dit token verloopt echter na 24 uur, dus het is noodzakelijk om minstens één keer per dag opnieuw in te loggen.

NB: het opslaan van het token in een environment variabele `bearer_token` lukt om onduidelijke redenen niet altijd. Mocht Postman 401 responses van Noviaal terug krijgen, dan moet helaas het

token met de hand in de genoemde environment variabele gevuld worden, waarna een `Persist All` van het Noviaal environment nodig is.

NB 2: verkeer naar Noviaal wordt beschermd door encryptie, geimplementeerd met een self-signed SSL certificate. Dit certificate moet voor iedere client, IntelliJ en Postman, eenmalig (als het goed is) geaccepteerd worden. Alle verkeer over https heeft het voordeel dat account gegevens niet (makkelijk) onderschept kunnen worden.

4. Endpoints

4.1. Authenticatie

endpoint	http method	uitkomst
/api/auth/register	POST	• een CreateUser naar dit endpoint geeft een UserResponse van de nieuwe gebruiker. • Als de CreateUser JSON niet correct is, wordt een http 400 status teruggegeven. • Als het opgegeven email adres al in gebruik is door een andere gebruiker wordt een http 400 status teruggegeven.
/api/auth/login	POST	• een LoginUser met email adres en password geeft een JwtResponse met daarin het token om beschermde endpoints te kunnen gebruiken (authenticatie) • Als de LoginUser JSON niet correct is, wordt een http 400 status teruggegeven • Als het email adres en/of password niet bekend zijn, wordt een http 401 status teruggegeven

Deze endpoints zijn voor iedereen toegankelijk.

4.1.1. CreateUser

```
{
  "name": "Bilbo",
  "email": "bilbo.baggins@hobbiton.shire",
  "password": "password"
}
```

4.1.2. LoginUser

```
{
  "username": "pippin@tuckborough.shire",
  "password": "password"
}
```

4.1.3. JwtResponse

```
{
    "token":
"eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJwaXBwaW5AdHVja2Jvcm91Z2guc2hpcmUiLCJpYXQiOjE2MTQzMzc4
ODMsImV4cCI6MTYxNDQyNDI4M30.i2Kms1FHePDS7B2zprNzFIcEHDoTeDqtlCyeNEs1z-g-
emvKFb1adqvUnyHtH9KENU9mizj1lO-aAhIvr22WKQ",
    "id": "030e5dac-2311-4179-a6d4-aa7f60838205",
    "username": "Pippin",
    "email": "pippin@tuckborough.shire",
    "roles": [
        "USER"
    ],
    "type": "Bearer"
}
```

4.1.4. UserResponse

```
{
  "id": "030e5dac-2311-4179-a6d4-aa7f60838205",
  "name": "Pippin",
  "email": "pippin@tuckborough.shire",
  "joined": "2021-02-26 12:10:44"
}
```

4.2. Users

Deze endpoints zijn alleen toegankelijk als je ingelogd bent.

endpoint	http method	uitkomst
/api/users/	GET	Lijst van alle gebruikers in Noviaal in List<UserResponse> JSON, gepagineerd met als defaults de eerste pagina van 20 gebruikers.
/api/users/me	GET	Details van de ingelogde gebruiker in UserResponse JSON
/api/users/{id}	GET	Details van de geregistreerde gebruiker gekenmerkt door {id} levert een UserResponse van die gebruiker of een http status 404 als de gebruiker niet gevonden werd.

endpoint	http method	uitkomst
/api/users/items	GET	Lijst van Items, Notes en Media, van de ingelogde gebruiker in de vorm van een lijst van ItemResponse
/api/users/{id}	DELETE	Verwijder de gebruiker met {id}, kan alleen als de ingelogde gebruiker de ADMIN rol heeft.
/api/users/follow/{id}	POST	De huidige gebruiker gaat de gebruiker met {id} volgen en geeft een UserFollowedResponse terug.
/api/users/followers	GET	Lijst met UserResponse van alle gebruikers die de huidige ingelogde gebruiker volgen.
/api/users/{id}/promote	PUT	De user met {id} heeft na deze call ook de rol ADMIN; dit endpoint geeft http status 202 terug.
/api/users/timeline	GET	lijst met ItemResponse van alle Items van gebruikers die de huidige ingelogde gebruiker volgt.

4.2.1. ItemResponse

Er zijn twee varianten van een Item: Note en Media. De inhoud van die twee soorten verschilt enigszins en beide uitingen worden hier getoond:

```
{
  "id": "68fe87bd-6446-4504-862e-0d4cda040e60",
  "created": "2021-02-26 15:19:17",
  "type": "Note",
  "title": "Note number One",
  "body": "Nevermind that, my lad..."
}
```

```
{
  "id": "ff881164-7253-4974-afc6-6aa049f4e70a",
  "created": "2021-02-26 15:48:22",
  "type": "Media",
  "name": "JDrivenLogoMailSmall-4.png",
  "contentType": "image/png"
}
```

4.2.2. UserFollowedResponse

```
{
  "userId": "307f446f-4f84-4c72-8bb5-fb2eed36c38f",
  "followedId": "49899834-6467-4355-b3c2-e08f34d8b6a3"
}
```

4.3. Notes

Deze endpoints zijn alleen toegankelijk als je ingelogd bent.

endpoint	http method	uitkomst
/api/notes	POST	nieuwe note voor huidige ingelogde gebruiker, geeft een ItemResponse met die note terug
/api/notes	GET	lijst met ItemResponse van de notes van huidige gebruiker
/api/notes/{id}	GET	geeft ItemResponse van note met {id} terug
/api/notes/user/{id}	GET	geeft lijst van ItemResponse van notes van user met {id} terug
/api/notes/{id}/comments	POST	voeg een Comment toe aan note met {id}, geeft een CommentResponse terug
/api/notes/{id}/comments	GET	lijst met CommentResponse voor Note met {id}

4.3.1. CommentResponse

```
{
  "comment": "test comment",
  "created": "2021-02-26 16:15:02",
  "author": "Pippin"
}
```

NB: Comments kunnen op zichzelf niet bestaan en worden daarom afgehandeld in de NoteController.

4.4. Media

Deze endpoints zijn alleen toegankelijk als je ingelogd bent.

endpoint	http method	uitkomst
/api/media	POST	upload bestand naar Noviaal en krijg een MediaUploadResponse terug. Maximale bestandsgrootte voor upload is terug te vinden in `application.properties` in de `src/main/resources` folder van dit project en is nu 200 MegaByte.
/api/media/{id}	GET	download een eerder ge-upload Media bestand met {id}, geeft een Resource terug met het juiste Content-Type terug.
/api/media/{userId}/list	GET	lijst met ItemResponse van Media van de user met {userId}.

4.4.1. MediaUploadResponse

```
{
  "id": "0e16d227-295b-4494-943f-8c3a67648a44",
  "name": "JDrivenLogoMailSmall-4.png",
  "contentType": "image/png",
  "size": 3823
}
```

4.5. Tags

endpoint	http method	uitkomst
/api/tags	POST	maak een nieuwe Tag op basis van CreateTag input en antwoord met een TagResponse of een http status 400 als de Tag met de opgegeven naam al bestaat.
/api/tags	GET	Lijst met TagResponse van alle Tags in Noviaal.
/api/tags/{name}/items	GET	Lijst met ItemResponse van alle Notes en Media die de Tag met naam {name} hebben.

4.5.1. CreateTag

```
{
  "name" : "test tag"
}
```

4.5.2. TagResponse

```
{
  "id": "db8baaa0-2e15-4665-aa39-2be1500f7993",
  "name": "personal"
}
```

Deze endpoints zijn alleen toegankelijk als je ingelogd bent.