Uruchomienie aplikacji

Wymagania

Do testowania i działania aplikacji wymagane są następujące komponenty:

- Java 17
- npm (Node Package Manager)
- Baza danych Neo4j. W tym przypadku, używany jest program Neo4j Desktop
- Ustawiony adres email, który obsługuje żądania zmiany hasła, adresu email, czy aktywacji konta
- Zainstalowany python3 (Do uruchomienia Scrappera)

Przed uruchomieniem aplikacji należy wykonać parę kroków:

Konfiguracja backendu

Na początku, należy w pliku *application.properties* podać kolejno parametry:

- Adres bazy
- Nazwę bazy
- Nazwę użytkownika
- Hasło

```
4  spring.neo4j.uri=bolt://localhost:7687
5  spring.data.neo4j.database=test
6  spring.neo4j.authentication.username=neo4j
7  spring.neo4j.authentication.password=11111111
```

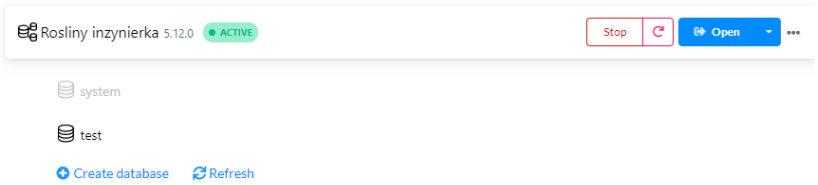
Rysunek 0.1 Parametry bazy danych

Potrzebna jest również konfiguracja maila. Poniżej wystarczy podać odpowiednie parametry do wybranego konta, które będzie wysyłało maile dotyczące zmiany hasła, adresu email, albo aktywacji konta.

```
13 # Ustawienia maila
14 spring.mail.host=smtp.gmail.com
15 spring.mail.port=587
16 spring.mail.username=[redacted]
17 spring.mail.password=[redacted]
18 spring.mail.properties.mail.smtp.auth=true
19 spring.mail.properties.mail.smtp.starttls.enable=true
```

Rysunek 0.2 Ustawienia maila

Po skonfigurowaniu bazy i maila, należy przejść do bazy danych. W programie Neo4j desktop albo w przeglądarce należy uruchomić wybraną bazę:



Rysunek 0.3 Uruchomiona baza danych

Backend Spring Boot musi mieć zainstalowane odpowiednie zależności. Można to zrobić poprzez przejście do folderu yukka i wpisanie komendy:

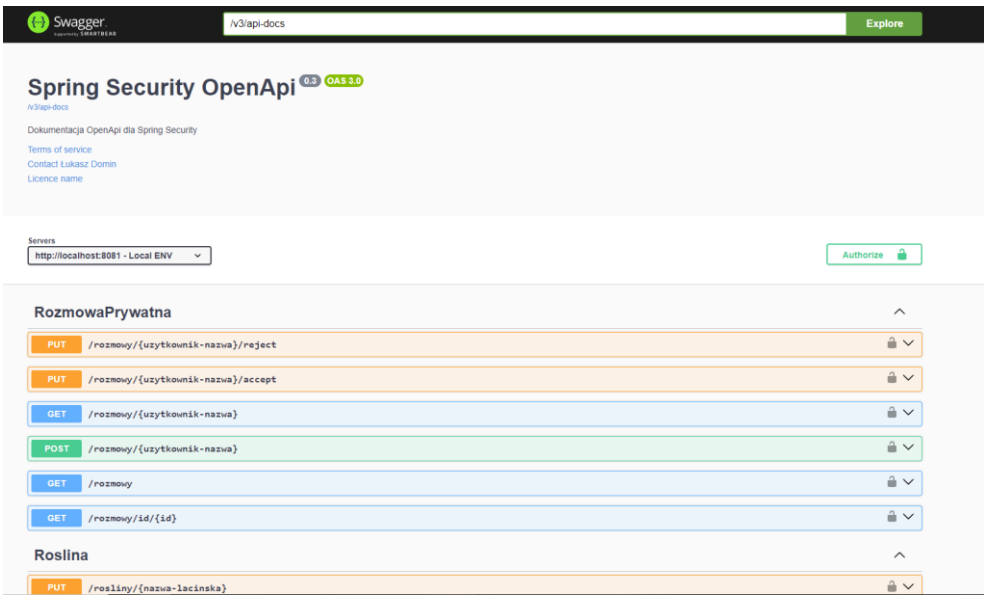
```
mvn clean install -D seed.database=true
```

Jeśli operacja wykona się poprawnie, zależności aplikacji zostaną zainstalowane, a sama baza danych zostanie zaseedowana. Jeśli ostatnia część nie jest potrzebna, można wykonać tą komendę bez argumentu lub zamieniając jego wartość na *false.*

Po instalacji, należy wykonać następującą komendę, aby uruchomić aplikację:

```
mvn spring-boot:run
```

Po uruchomieniu, można zobaczyć dokumentację OpenApi przechodząc do adresu http://localhost:8081/swagger-ui/index.html



Rysunek 0.4 Widok Swaggera

W tym adresie jest pokazana dokumentacja OpenApi dla Spring Security. Zawiera ona wszystkie endpointy, ich parametry, requesty i przykładowe odpowiedzi. Jest ona szczególnie ważna do komunikacji z frontendem.

Konfiguracja frontendu

Przejdź do folderu *yukka-ui* i zainstaluj wymagane pakiety do frontendu, wykonując komendę:

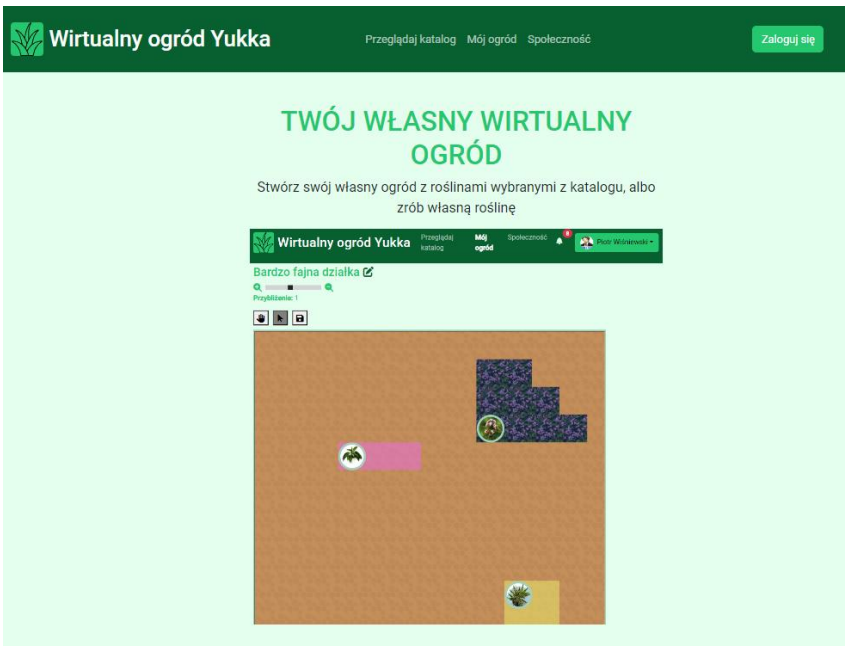
```
npm i
```

Po zainstalowaniu pakietów, możesz uruchomić serwer na frontendzie używając jednej z tych komend:

`ng serve` Albo `npm run start`

Po uruchomieniu serwera, należy przejść pod ten adres:

http://localhost:4200

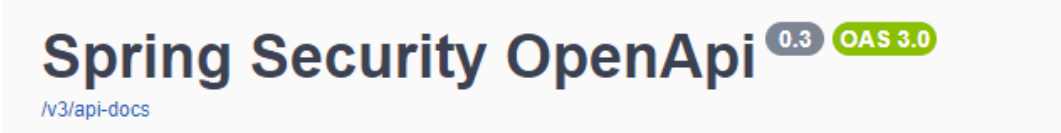


Rysunek 0.5 Widok strony głównej

Aktualizacja danych pomiędzy backendem a frontendem(dać do Przedstawienie aplikacji)

Jeśli na backendzie były wprowadzane jakieś zmiany (Modyfikacja obiektu, dodanie nowego frontentu itp.), należy go zresetować i przejść do dokumentacji OpenApi: http://localhost:8081/swagger-ui/index.html,

Następnie przejść pod ten adres /v3/api-docs:



Rysunek 0.6 Link do api-docs

Ten adres zwraca JSON zawierający informacje o dokumentacji OpenApi. Należy zastosować formatowanie stylistyczne i skopiować go CTRL + A:

Następnie należy przejść do pliku na frontendzie pod ścieżką *yukka-ui/src/app/openApi/openApi.json* i wkleić zawartość:

Rysunek 0.7 Fragment pliku openApi.json

Następnie będąc w folderze yukka-ui wykonać komendę:

```
npm run api-gen
```

Ta komenda używając pakietu *ng-openapi-gen* wygeneruje modele odwzorowane na podstawie wklejonej dokumentacji i przerabia je tak, aby można było używać ich na frontendzie. Przykładowo, zamiast ręcznie pisać obiekt rośliny, request i funkcję obsługującą żądanie wyszukiwania rośliny, generator robi to za programistę, oszczędzając czas.