

**UNIWERSYTET RZESZOWSKI**

**Kolegium Nauk Przyrodniczych**

Łukasz Domin

Nr albumu: 123631

Informatyka I stopnia

**Aplikacja internetowa do zarządzania ogrodem z elementami społecznościowymi**

Praca inżynierska

Praca wykonana pod kierunkiem

….....……………………………………..

Rzeszów, data

# Spis treści

1. [Spis treści 2](#_Toc184903973)
2. [Spis ilustracji 4](#_Toc184903974)
3. [Wstęp 4](#_Toc184903975)
4. [Część zasadnicza 5](#_Toc184903976)

[1 Narzędzia i technologie 5](#_Toc184903977)

[1.1 Wymagania 5](#_Toc184903978)

[1.2 Część frontendowa 5](#_Toc184903979)

[1.3 Część backendowa 6](#_Toc184903980)

[1.4 Scrapper 7](#_Toc184903981)

[1.4.1 Pobranie roślin 7](#_Toc184903982)

[1.4.2 Pobranie obrazów roślin 7](#_Toc184903983)

[2 Baza danych 7](#_Toc184903984)

[2.1 Źródła danych 8](#_Toc184903985)

[2.2 Diagram [TODO] 9](#_Toc184903986)

[2.3 Struktura bazy danych 9](#_Toc184903987)

[2.3.1 Węzły 9](#_Toc184903988)

[2.3.2 Relacje 14](#_Toc184903989)

[3 Projekt GUI 16](#_Toc184903990)

[4 Uruchomienie aplikacji 16](#_Toc184903991)

[4.1 Konfiguracja backendu 16](#_Toc184903992)

[4.2 Konfiguracja frontendu 19](#_Toc184903993)

[4.3 Aktualizacja danych pomiędzy backendem a frontendem 19](#_Toc184903994)

[5 Przedstawienie aplikacji 22](#_Toc184903995)

[5.1 Logowanie i rejestracja 22](#_Toc184903996)

[5.1.1 Logowanie 22](#_Toc184903997)

[5.1.2 Proces rejestracji 22](#_Toc184903998)

[5.2 CRUD pracownika aplikacji 22](#_Toc184903999)

[5.3 Przeglądanie ogólnodostępnych zasobów 23](#_Toc184904000)

[5.4 Zarządzanie swoimi zasobami 23](#_Toc184904001)

[5.4.1 Przykładowa walidacja dla tworzenia, edytowania oraz usuwania roślin 23](#_Toc184904002)

[5.4.2 Przykład działania walidacji 23](#_Toc184904003)

[5.4.3 Przykładowa walidacja dla dodawania, edytowania oraz usuwania rośliny na działce 23](#_Toc184904004)

[5.4.4 Przykład działania walidacji 23](#_Toc184904005)

1. [Zakończenie 24](#_Toc184904006)

# Spis ilustracji

[Obraz 2.3.1 Węzeł Roslina 10](#_Toc184904007)

[Obraz 2.3.2 Węzeł UzytkownikRoslina 10](#_Toc184904008)

[Obraz 2.3.3 Węzeł Wlasciwosc 12](#_Toc184904009)

[Obraz 2.3.4 Węzeł UzytkownikWlasciwosc 12](#_Toc184904010)

[Obraz 2.3.5 Węzeł ogrodu 13](#_Toc184904011)

[Obraz 2.3.6 Węzeł Dzialka 13](#_Toc184904012)

[Obraz 4.1.1 Parametry bazy danych 16](#_Toc184904013)

[Obraz 4.1.2 Parametry maila 16](#_Toc184904014)

[Obraz 4.1.3 Uruchomiona baza danych 17](#_Toc184904015)

[Obraz 4.1.4 Widok Swaggera 18](#_Toc184904016)

[Obraz 4.2.1 Widok strony głównej 19](#_Toc184904017)

[Obraz 4.3.1 Link do api-docs 20](#_Toc184904018)

[Obraz 4.3.2 Koniec pliku openApi.json 22](#_Toc184904019)

# Wstęp

Celem pracy jest stworzenie systemu umożliwiającego zarządzanie wirtualnym ogrodem oraz wymianę wiedzy i doświadczeń pomiędzy użytkownikami w przestrzeni społecznościowej.

Tworzona aplikacja dostarcza użytkownikom intuicyjne rozwiązania, takie jak system powiadomień o pracach w ogrodzie, możliwość projektowania własnych działek z roślinami wybranymi z katalogu, które można następnie dostosować do swoich upodobań. Możliwe jest również stworzenie własnej rośliny i posadzenie jej na swojej działce. Dodatkowo, poprzez integrację części społecznościowej możliwa jest interakcja z innymi użytkownikami, udostępnianie widoku działki, dzielenie się poradami na temat roślin, a także możliwość prowadzenia rozmów prywatnych. Dzięki temu system wspiera zarówno początkujących, jak i zaawansowanych ogrodników, tworząc przyjazne środowisko do dzielenia się swoją pasją.

# Część zasadnicza

## Narzędzia i technologie

### 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)

### Część frontendowa

**Języki programowania:** TypeScript 5.5.2

TypeScript to rozszerzenie języka JavaScript dodające typowanie statyczne, klasy oraz interfejsy. Ma dużo mniejszą tolerancję na błędy, a sam jego kod jest kompilowany na końcu do JavaScript.

Link do pobrania: <https://www.typescriptlang.org/download/>

**Framework:** Angular 18.2.11

Jest to framework opartym na języku TypeScript do budowy aplikacji webowych, rozwijanego przez Google. Wspomniana wersja posiada mechanikę sygnałów do komunikacji pomiędzy komponentami, wsparcie dla SSR (rendererowania po stronie serwera), hydracji (użycie statycznej części wygenerowanej strony przez SSR), oraz lepsza obsługa ładowania asynchronicznego.

Link do pobrania: <https://v18.angular.dev/installation>

**Pozostałe narzędzia i technologie:**

**Bootstrap 5** - biblioteka CSS. Zawiera zestaw narzędzi ułatwiających tworzenie interfejsu graficznego stron oraz aplikacji internetowych. Bazuje głównie na gotowych rozwiązaniach HTML oraz CSS i może być stosowany m.in. do stylizacji poszczególnych elementów strony.

**Npm (Node Package Manager**) - Menedżer pakietów dla JavaScriptu, dostarczany wraz z Node.js. Służy do instalowania, zarządzania i udostępniania bibliotek oraz narzędzi JavaScript.

### Część backendowa

**Języki programowania:** Java 17

Java to język obiektowy działający na różnych systemach operacyjnych bez większych trudów. Jest szeroko stosowana do tworzenia przeróżnych aplikacji czy to internetowych, czy to mobilnych.

Link do pobrania: <https://www.oracle.com/java/technologies/javase/jdk17-archive-downloads.html>

**Framework:** Spring Boot 3.3.2

Spring Boot to framework języka Java, który upraszcza proces konfiguracji i uruchamiania aplikacji, automatycznie konfigurując większość komponentów, skracając kod przy użyciu adnotacji, a także upraszcza komunikację z bazą danych i frontendem.

Link do pobrania: <https://docs.spring.io/spring-boot/installing.html>

### Scrapper

Jest to scrapper napisany w języku Python używany do pobierania danych o roślinach ze strony <https://e-katalogroslin.pl> do pliku *katalog\_roslin.csv.* Wykorzystuje on sterownik Firefox, wielowątkowość oraz asynchroniczność do szybszego pobierania danych ze stron.

#### Pobranie roślin

Aby rozpocząć proces pobierania roślin, należy uruchomić plik scrap\_threads.py, który pobierze dane od początkowej strony aż do ostatniej. Należy mieć na uwadze, że czasami program może mieć trudności z dostępnością do strony, przejściem na inną stronę oraz inne czynniki zewnętrzne które mogą zakłócić bądź uniemożliwić scrappowanie danych. Pobranie wszystkich danych ze strony powinno zająć ok. 50-70 minut.

#### Pobranie obrazów roślin

Plik *katalog\_roslin.csv* zawiera informacje o roślinach takie jak nazwa rośliny, nazwa łacińska, opis, kwasowość gleby, a także link do obrazu rośliny (jeśli takowy posiada). Aby pobrać obrazy rośliny, należy uruchomić plik *download\_images.py*, który będzie iterował przez kolejne rzędy pliku *katalog\_roslin.csv* i zapisze obrazy do folderu *images.* Proces ten powinien zająć do 5 minut.

## Baza danych

**Baza danych:** Neo4j 5.12.0

Neo4j jest grafową bazą danych, która przechowuje dane w postaci węzłów, ich etykiet, relacji pomiędzy nimi oraz właściwościami węzłów bądź relacji. Została ona wybrana dlatego, że ten typ bazy danych jest polecany w budowaniu aplikacji społecznościowych ze względu na budowę relacji pomiędzy poszczególnymi elementami bazy.

### Źródła danych

Źródłem danych jest strona internetowa *e-katalogroslin.pl* należąca do Związku Szkółkarzy Polskich, który jest stowarzyszeniem producentów drzew, krzewów i bylin, a także roślin owocowych. Wymieniona strona jest katalogiem zawierający ponad 10 000 roślin o różnych właściwościach takich jak preferowany rodzaj gleby, okres owocowania, zimozieloność liści, zakres wysokości itp.

Link do strony: <https://e-katalogroslin.pl>

### Diagram [TODO]

### Struktura bazy danych

#### Węzły

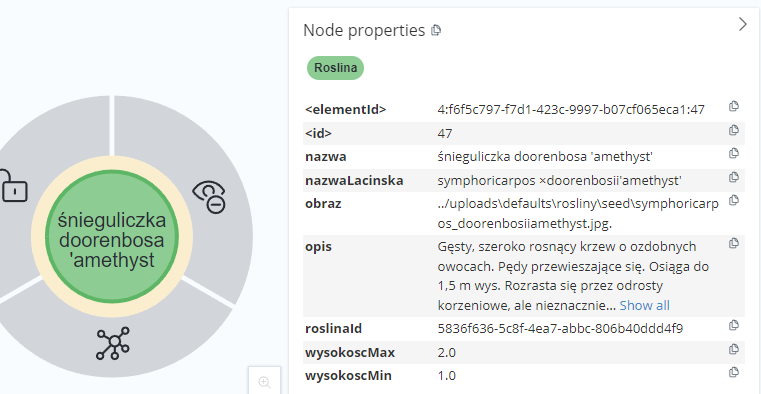
##### Roslina

Roślina jest jednym z głównym węzłów w bazie danych. Reprezentuje roślinę z katalogu oraz roślinę użytkownika i jest wykorzystywana w ogrodzie. Rośliny użytkownika posiadają etykietę Roslina oraz UzytkownikRoslina, a roślina z katalogu może mieć tylko etykietę Roslina.

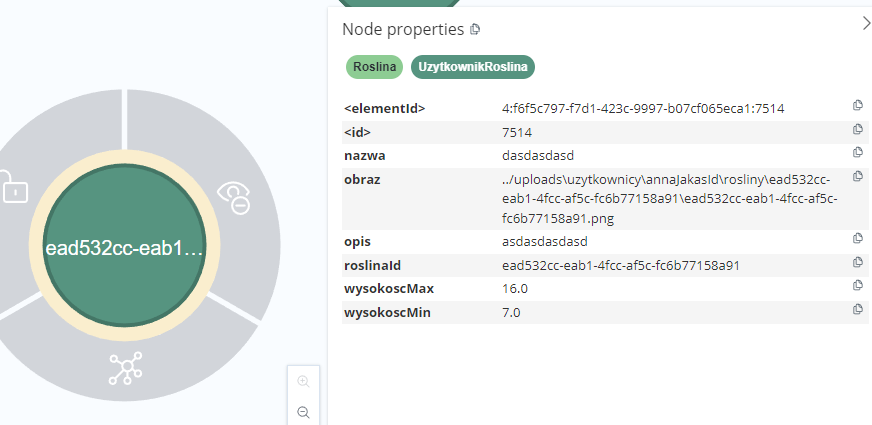
**Etykiety:** Roslina, UzytkownikRoslina

**Właściwości:**

* roslinaId - Id rośliny wygenerowany na backendzie. Jest on używany zamiast id, gdyż w Neo4J jest ono przestarzałe i zostanie w przyszłości zastąpione.
* nazwa – nazwa rośliny
* nazwaLacinska – nazwa łacińska rośliny
* opis – opis rośliny
* wysokoscMin, wysokoscMax – wysokość minimalna i maksymalna rośliny
* obraz – ścieżka do obrazu rośliny, jeśli takowy posiada. W przeciwnym wypadku obrazem rośliny jest obraz domyślny



Obraz 2.3.1 Węzeł Roslina



Obraz 2.3.2 Węzeł UzytkownikRoslina

###### Właściwości rośliny

Rośliny posiadają właściwości o 16 różnych etykietach takich jak *Owoc, Grupa, Gleba,* które klasyfikują typ danej właściwości. Zwykła właściwość ma etykietę *Wlasciwosc*, za to właściwość użytkownika ma dodatkowo etykietę *UzytkownikWlasciwosc.*

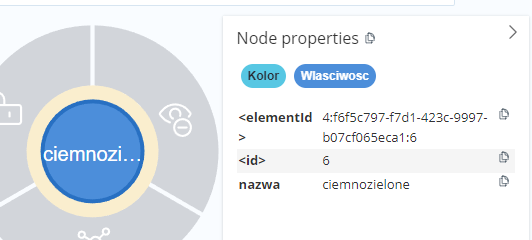
**Etykiety główne:** Wlasciwosc, UzytkownikWlasciwosc

**Etykiety dodatkowe:**

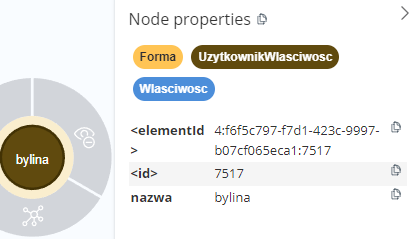
* Forma
* Gleba
* Grupa
* Kolor
* Kwiat
* Odczyn
* Okres
* Owoc
* Podgrupa
* Pokroj
* SilaWzrostu
* Stanowisko
* Wilgotnosc
* Zastosowanie
* Zimozielonosc

**Właściwości:**

nazwa – nazwa właściwości



Obraz 2.3.3 Węzeł Wlasciwosc



Obraz 2.3.4 Węzeł UzytkownikWlasciwosc

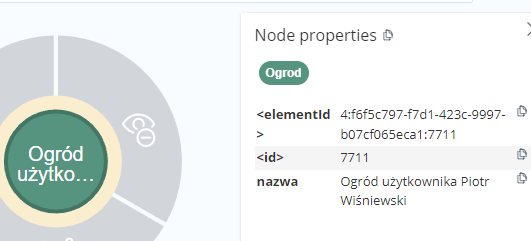
##### Ogrod

Ogród jest węzłem tworzonym dla każdego użytkownika. Reprezentuje ogród podzielony na 10 działek, które również są węzłami złączonymi z nim.

**Etykiety:** Ogrod

**Właściwości:**

* nazwa – nazwa ogrodu



Obraz 2.3.5 Węzeł ogrodu

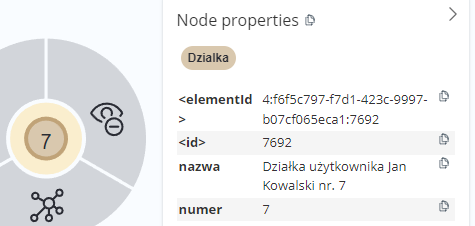
##### Dzialka

Działka jest węzłem związanym z ogrodem, tworzony wraz z nim. Łącznie jest tworzone po 10 działek na każdy ogród, gdzie każda działka ma wymiary 20x20 kafelków.

**Etykiety:** Dzialka

**Właściwości:**

* nazwa – nazwa działki
* numer – numer działki od 1 do 10



Obraz 2.3.6 Węzeł Dzialka

##### Uzytkownik

TODO: Dodać podział na role i resztę

###### Ustawienia

##### Post

##### Komentarz

##### RozmowaPrywatna

##### Powiadomienie

Aa

#### Relacje

##### Relacje roślin

Roślina posiada jakieś relacje.

STWORZONA\_PRZEZ

ZASADZONA\_NA

###### Relacje właściwości

Poniżej znajduje się kilka wybrany relacji z właściwościami rośliny

MA\_FORME

MA\_GLEBE

MA\_GRUPE

MA\_KOLOR\_KWIATOW

MA\_KOLOR\_LISCI

MA\_KWIAT

MA\_ODCZYNY

MA\_OWOC

MA\_PODGRUPE

MA\_POKROJ

##### Relacje społecznościowe

JEST\_W\_POSCIE

JEST\_W\_ROZMOWIE

MA\_KOMENTARZ

MA\_POST

OCENIL

ODPOWIEDZIAL

SKOMENTOWAL

ZGLASZA

##### Relacje użytkownika

MA\_USTAWIENIA

POWIADAMIA

MA\_DZIALKE

MA\_OGROD

## Projekt GUI

Poniżej będą widoki poszczególnych stron na frontendzie.

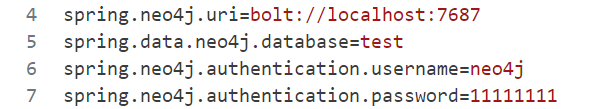
## Uruchomienie aplikacji

Przed uruchomieniem aplikacji należy wykonać parę kroków:

### Konfiguracja backendu

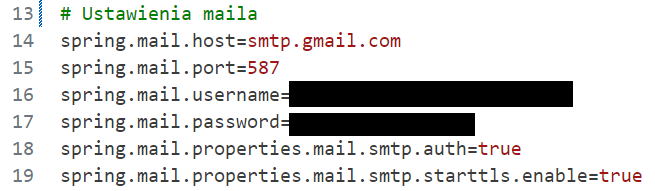
Na początku, należy w pliku *application.properties* podać kolejno parametry:

* Adres bazy
* Nazwę bazy
* Nazwę użytkownika
* Hasło



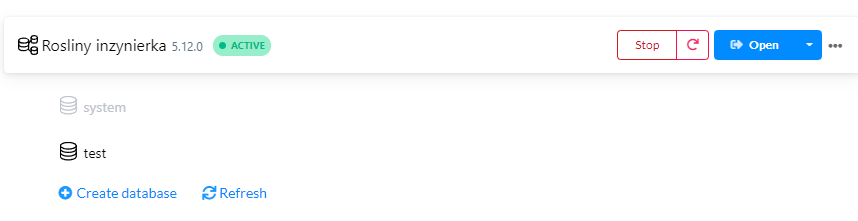
Obraz 4.1.1 Parametry bazy danych

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.



Obraz 4.1.2 Parametry maila

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



Obraz 4.1.3 Uruchomiona baza danych

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

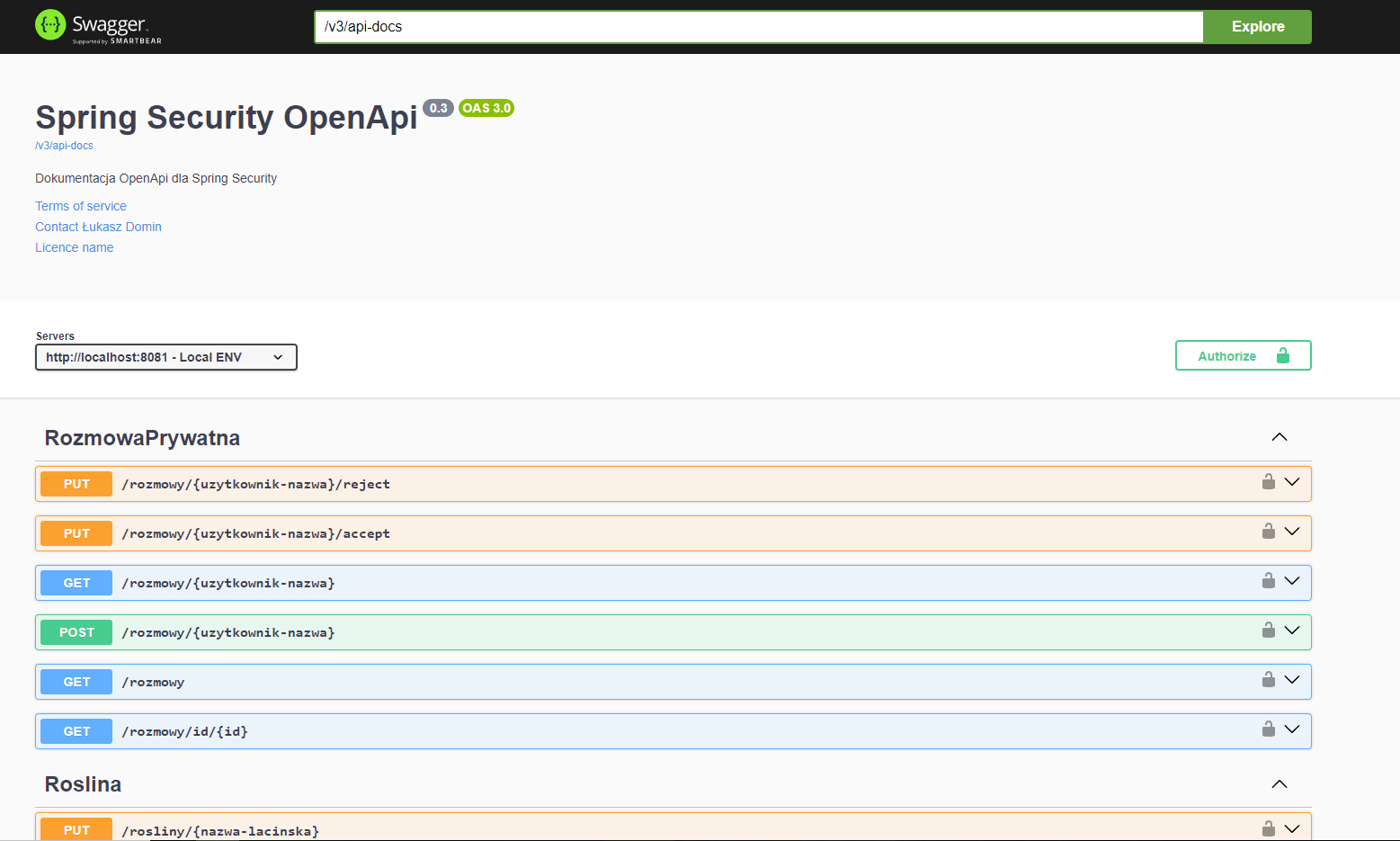
mvn clean install -D seed.database=true

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>



Obraz 4.1.4 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:

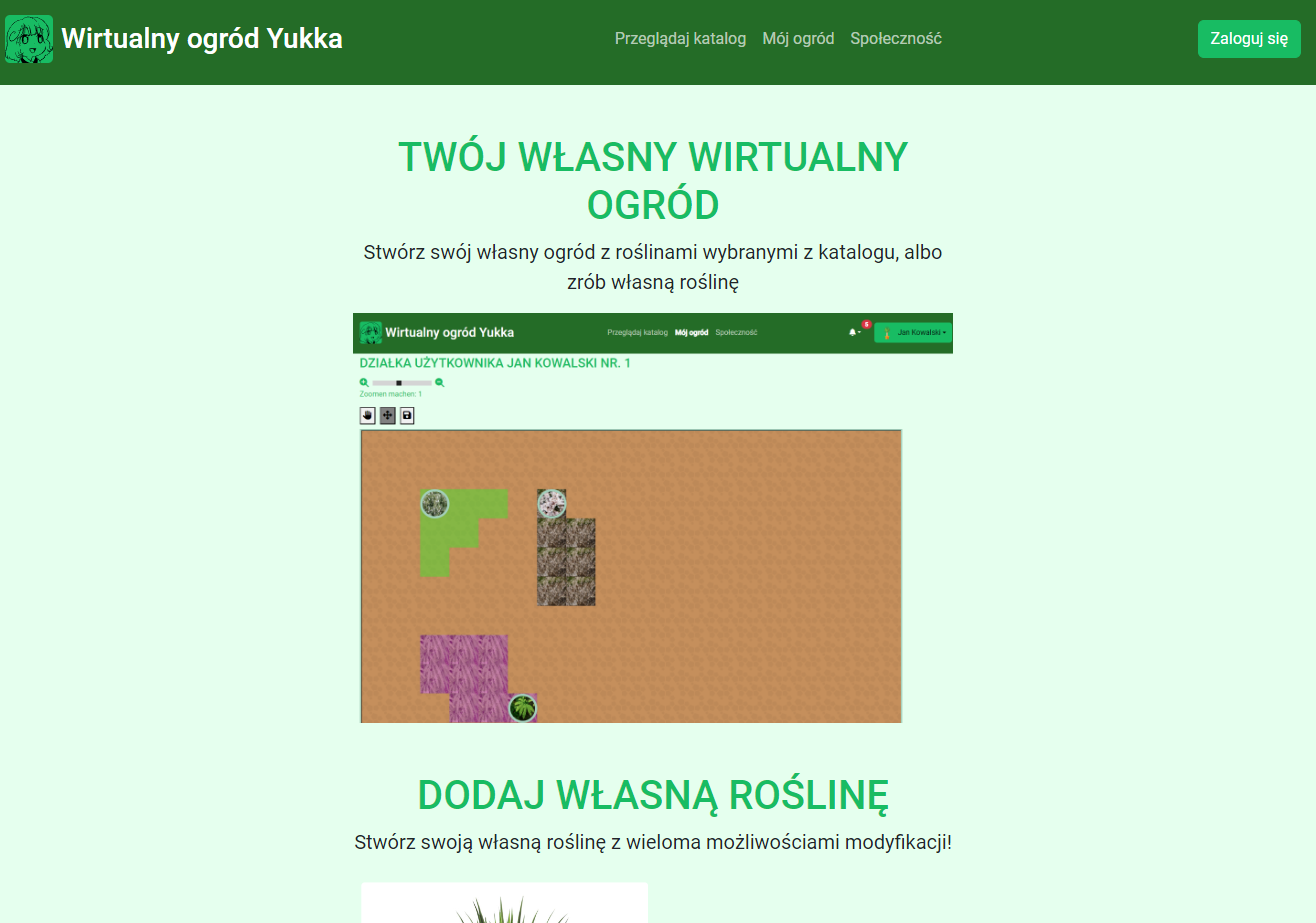
ng serve

Albo

npm run start

Po uruchomieniu serwera, należy przejść pod ten adres:

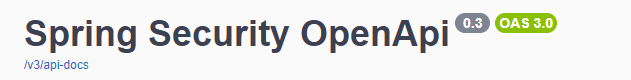
<http://localhost:4200>



Obraz 4.2.1 Widok strony głównej

### Aktualizacja danych pomiędzy backendem a frontendem

Jeśli na backendzie były wprowadzane jakieś zmiany (Modyfikacja obiektu, dodanie nowego frontentu itp.), należy go zresetować, a następnie przejść do dokumentacji OpenApi: <http://localhost:8081/swagger-ui/index.html>, a następnie w widoku przejść pod ten adres [/v3/api-docs](http://localhost:8081/v3/api-docs):

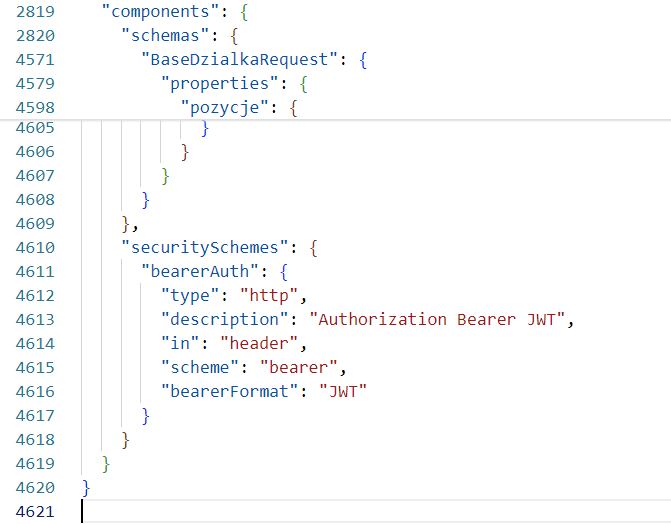


Obraz 4.3.1 Link do api-docs

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:



Obraz 4.3.2 Koniec 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.

## Przedstawienie aplikacji

### Logowanie i rejestracja

#### Logowanie

#### Proces rejestracji

### CRUD pracownika aplikacji

### Przeglądanie ogólnodostępnych zasobów

### Zarządzanie swoimi zasobami

#### Przykładowa walidacja dla tworzenia, edytowania oraz usuwania roślin

#### Przykład działania walidacji

#### Przykładowa walidacja dla dodawania, edytowania oraz usuwania rośliny na działce

TODO

#### Przykład działania walidacji

TODO

# Zakończenie

a

streszczenie pracy w języku polskim, tytuł pracy dyplomowej w języku polskim oraz w języku angielskim,

oświadczenie studenta o samodzielnym przygotowaniu pracy dyplomowej (załącznik do Zarządzenia w sprawie funkcjonowania procedury antyplagiatowej w Uniwersytecie Rzeszowskim).