



**WYŻSZA SZKOŁA  
INFORMATYKI i ZARZĄDZANIA**  
z siedzibą w Rzeszowie

**Programowanie obiektowe**

*Dziennik elektroniczny*

Prowadzący:

mgr inż. Ewa Żesławska

Autor:

Albert Szymański

66022

Kierunek: 3 II Z GP02

Rzeszów, r.a. 2022/2023

## Spis treści

<b>1. Cele projektu.</b>	<b>3</b>
1.1. Wymagania funkcjonalne	3
1.2. Wymagania нефunkcjonalne	3
<b>2. Opis techniczny projektu</b>	<b>4</b>
<b>3. Harmonogram realizacji projektu</b>	<b>4</b>
<b>4. Prezentacja warstwy użytkowej projektu.</b>	<b>5</b>
4.1. Auth controller	5
4.2. Classroom controller	7
4.3. Grade controller	10
4.4. Students controller	13
4.5. Student controller.	16
<b>5. Repozytorium, system kontroli wersji.</b>	<b>17</b>
<b>6. Podsumowanie.</b>	<b>17</b>

# **1. Cele projektu.**

## **1.1. Wymagania funkcjonalne**

- Możliwość zakładania konta przez nauczyciela
- Możliwość tworzenia konta dla ucznia
- Możliwość przeglądnięcia listy uczniów
- Możliwość przeglądnięcia szczegółów ucznia
- Możliwość utworzenia klasy przez nauczyciela
- Możliwość przeglądania listy utworzonych klas
- Możliwość przypisania ucznia do klasy której jest się właścicielem
- Możliwość dodania oceny uczniowi
- Możliwość edycji oceny przez nauczyciela
- Możliwość usunięcia oceny przez nauczyciela
- Możliwość wglądu do swoich ocen przez uczniów

## **1.2. Wymagania niefunkcjonalne**

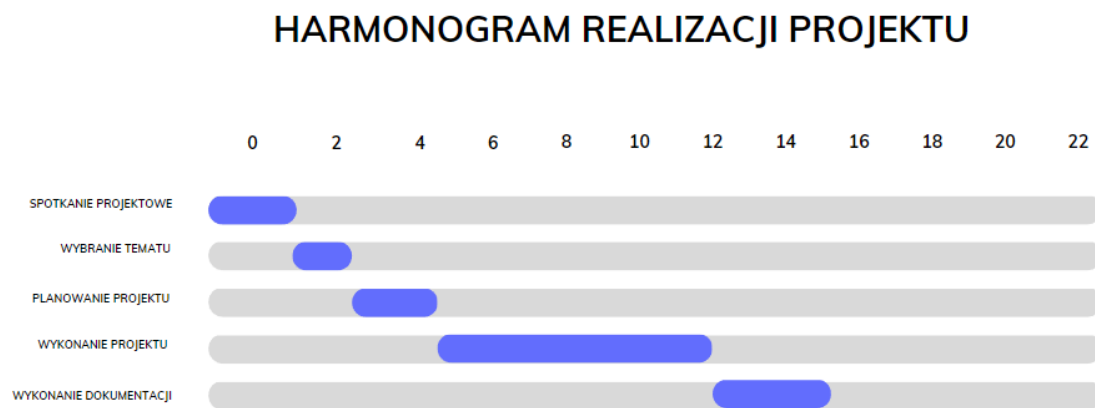
- Aplikacja ułatwia proces zarządzania ocenami i stanowi alternatywę dla dziennika tradycyjnego
- API udostępniane przez aplikację może być wykorzystywana przez aplikacje przeglądarkowe, mobilne i desktopowe.
- Aplikacja tworzona jest w języku Java z frameworkiem Spring Boot.
- Aplikacja nawiązuje połączenie z bazą danych i używa rekordów w niej zapisanych.

## 2. Opis techniczny projektu.

- Środowisko programistyczne Javy: Java JDK Kit 19.0.1.
- Środowisko programistyczne: IntelliJ IDEA Community Edition.
- Spring Boot
- Docker Compose 3.9
- PostgreSQL 14 alpine
- Postman
- Swagger UI

## 3. Harmonogram realizacji projektu.

Poniżej zamieszczono harmonogram realizacji projektu.



Rysunek 1. Diagram Gantta

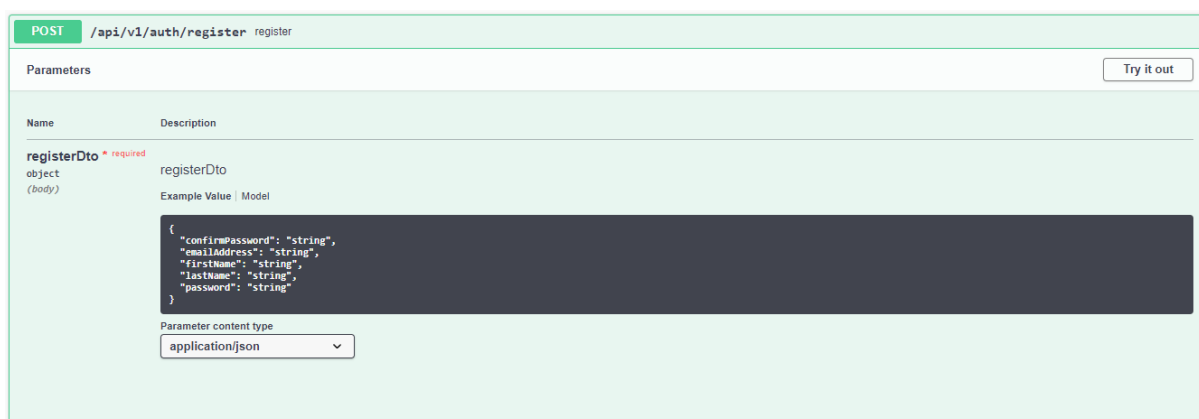
## 4. Prezentacja warstwy użytkowej projektu.

Endpoints w API podzielone są na 5 grup przedstawionych poniżej. Wszystkie endpointy oprócz tych które są używane do rejestracji, logowania i swaggera są chronione przez autoryzację z użyciem JWT.

### 4.1. Auth controller

Kontroler posiada 2 ścieżki */auth/login* i *auth/register*.

- Ścieżka **POST** *register* przyjmuje dane użytkownika wymagane do założenia konta w body.



Rysunek 2. api/v1/auth/register

- Ścieżka **POST** *login* przyjmuje dane użytkownika wymagane do zalogowania oraz zwraca rolę użytkownika, wiadomość powitalną oraz token JWT.

**POST** /api/v1/auth/login login

Parameters Try it out

Name	Description
<b>loginDto</b> * required object (body)	loginDto Example Value   Model <pre>{  "emailAddress": "string",  "password": "string"}</pre> <div>Parameter content type application/json</div>

Responses Response content type \*/\*

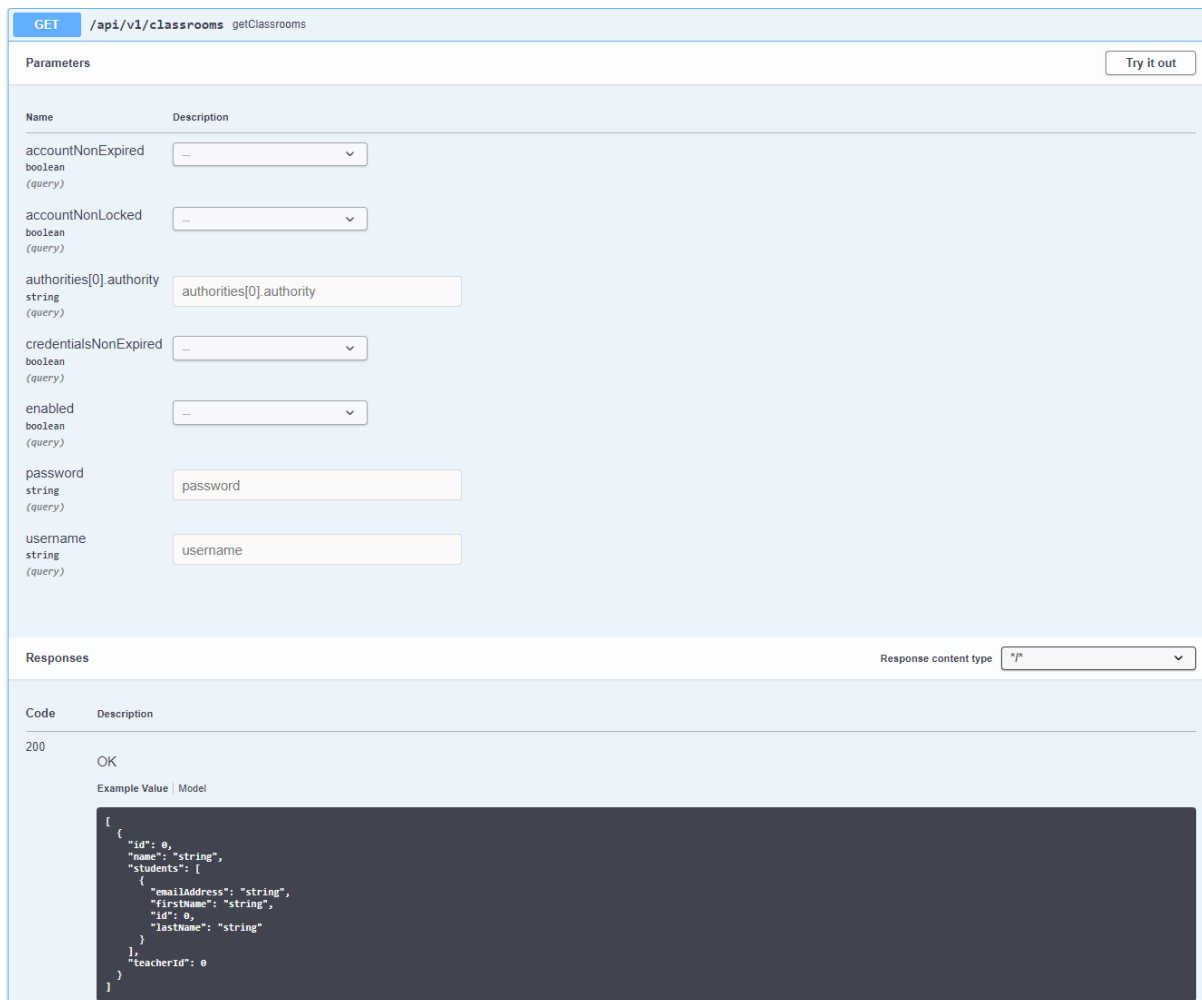
Code	Description
200	OK Example Value   Model <pre>{  "accessToken": "string",  "message": "string",  "role": "STUDENT"}</pre>

Rysunek 3. api/v1/auth/login

## 4.2. Classroom controller

Ścieżki w tej grupie endpointów są chronione przez autoryzację z użyciem JWT i są dostępne dla nauczyciela.

- ścieżka **GET *classrooms*** zwraca klasy należące do nauczyciela łącznie z listą uczniów



The image shows the Swagger UI for the GET /api/v1/classrooms endpoint. The interface is divided into two main sections: Parameters and Responses.

**Parameters:** This section lists query parameters for the endpoint. It includes a 'Try it out' button in the top right corner.

Name	Description
accountNonExpired boolean (query)	--
accountNonLocked boolean (query)	--
authorities[0].authority string (query)	authorities[0].authority
credentialsNonExpired boolean (query)	--
enabled boolean (query)	--
password string (query)	password
username string (query)	username

**Responses:** This section shows the response for the endpoint. It includes a 'Response content type' dropdown menu set to '\*/\*'. The response is a 200 OK status with an example value.

Code: 200, Description: OK

Example Value | Model

```
{
  "id": 0,
  "name": "string",
  "students": [
    {
      "emailAddress": "string",
      "firstName": "string",
      "id": 0,
      "lastName": "string"
    }
  ],
  "teacherId": 0
}
```

Rysunek 4. api/v1/classrooms

- ścieżka **POST** *classrooms* przyjmuje nazwę klasy w body oraz tworzy klasę z podaną nazwą

**POST**
/api/v1/classrooms
createClassroom

Parameters

Try it out

Name	Description
accountNonExpired boolean (query)	...
accountNonLocked boolean (query)	...
authorities[0].authority string (query)	authorities[0].authority
createClassroomDto * required object (body)	<div>createClassroomDto</div> <div>Example Value   Model</div> <pre>{   "className": "string" }</pre>

Rysunek 5. api/v1/classrooms



- ścieżka **GET *classrooms/{classroomId}*** przyjmuje id klasy jako parametr oraz zwraca jej szczegóły: nazwa, uczniowie oraz ich oceny

GET
/api/v1/classrooms/{classroomId}
getClassroomById

Try it out

Name	Description
accountNonExpired boolean (query)	<input type="text"/>
accountNonLocked boolean (query)	<input type="text"/>
authorities[0].authority string (query)	<input type="text"/>
<b>classroomId</b> * required integer(\$int64) (path)	<input type="text"/>
credentialsNonExpired boolean (query)	<input type="text"/>
enabled boolean (query)	<input type="text"/>
password string (query)	<input type="text"/>
username string (query)	<input type="text"/>

Responses
Response content type
\*/

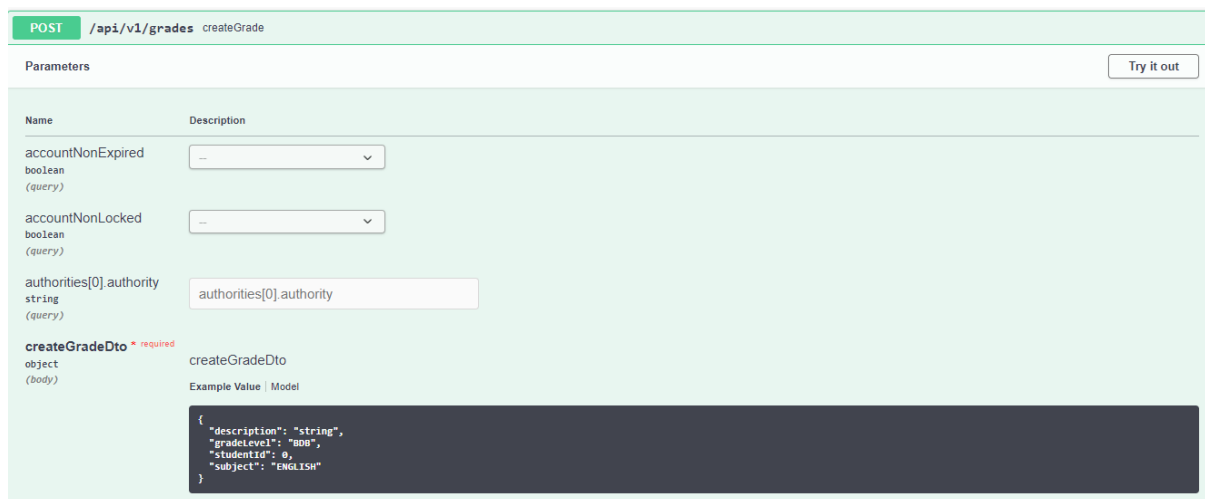
Code	Description
200	OK <div> Example Value Model </div> <pre> {   "id": 0,   "name": "string",   "students": [     {       "firstName": "string",       "id": 0,       "lastName": "string",       "studentGrades": [         {           "description": "string",           "id": 0,           "level": "800",           "studentId": 0,           "subject": "ENGLISH"         }       ]     }   ],   "teacherId": 0 } </pre>

Rysunek 6. api/v1/classrooms/{classroomId}

### 4.3. Grade controller

Ścieżki w tej grupie endpointów są chronione przez autoryzację z użyciem JWT i są dostępne dla nauczyciela.

- ścieżka **POST** *grades* przyjmuje BODY w którym podaje się id ucznia, nazwę przedmiotu, oraz wysokość oceny. Endpoint tworzy ocenę uczniowi.



The image shows the Swagger UI for the **POST /api/v1/grades** endpoint, labeled **createGrade**. It features a "Parameters" section with a "Try it out" button. The parameters are:

Name	Description
accountNonExpired boolean (query)	...
accountNonLocked boolean (query)	...
authorities[0].authority string (query)	authorities[0].authority
<b>createGradeDto</b> * required object (body)	<b>createGradeDto</b> Example Value   Model <pre>{   "description": "string",   "gradeLevel": "000",   "studentId": 0,   "subject": "ENGLISH" }</pre>

Parameter content type

Rysunek 7. api/v1/grades

- ścieżka **PUT *grades/{:gradeId}*** przyjmuje w body dane do edycji oceny oraz wymaga podania parametru *gradeId* który jest id oceny w bazie

PUT

/api/v1/grades/{gradeId} editGrade

Parameters Try it out

Name	Description
accountNonExpired boolean (query)	--
accountNonLocked boolean (query)	--
authorities[0].authority string (query)	authorities[0].authority
credentialsNonExpired boolean (query)	--
<b>editGradeDto</b> * required object (body)	<div>editGradeDto</div> <div>Example Value   Model</div> <pre>{   "description": "string",   "gradeLevel": "B00",   "studentId": 0,   "subject": "ENGLISH" }</pre> <div>Parameter content type</div> <div>application/json</div>
enabled boolean (query)	--
<b>gradeId</b> * required integer(\$int64) (path)	<div>gradeId</div> <div>gradeId - gradeId</div>

Rysunek 8. *api/v1/grades/{gradeId}*

- ścieżka **DELETE** *grades/{gradeId}* służy do usuwania oceny przez nauczyciela i wymaga podania parametru gradeId który jest id oceny w bazie.

**DELETE** /api/v1/grades/{gradeId} deleteGrade

Parameters
 Try it out

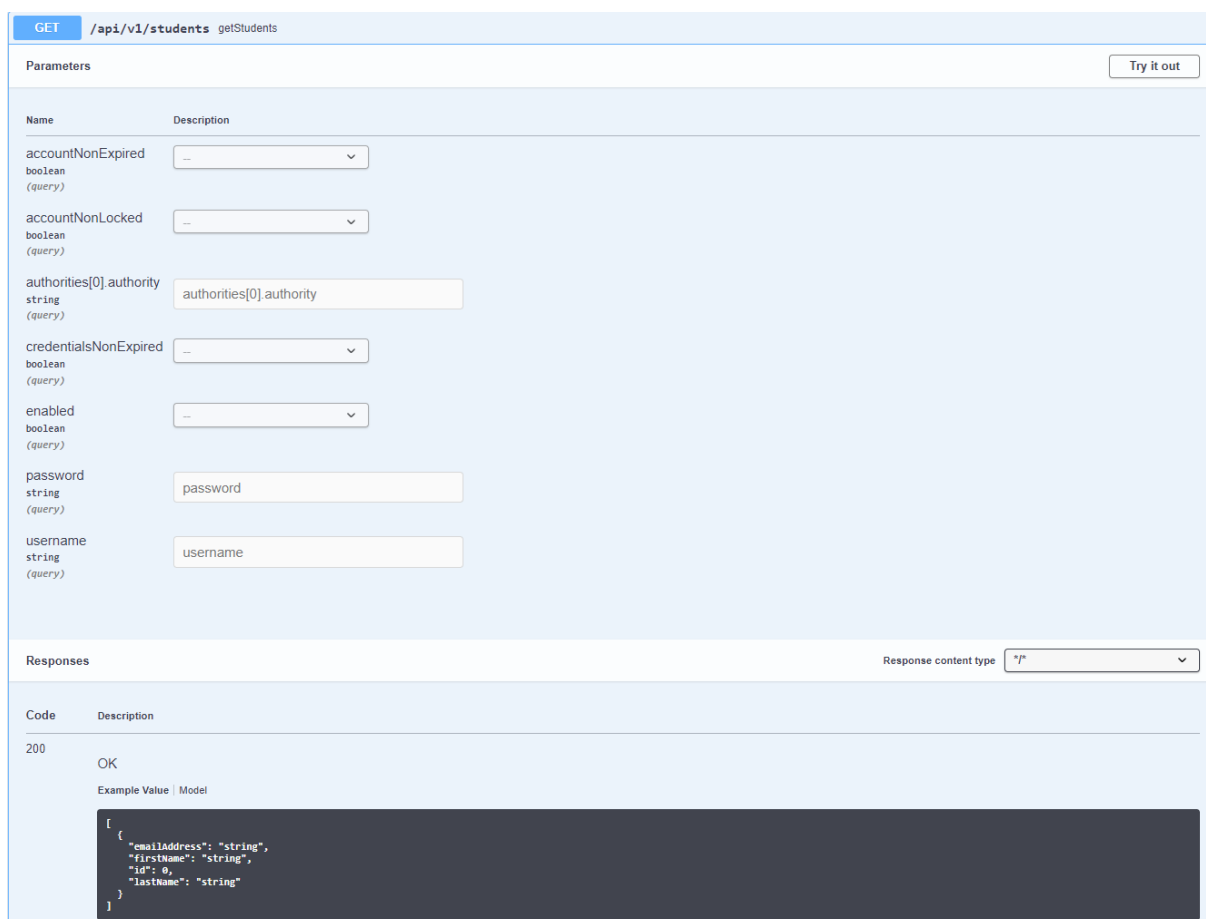
Name	Description
accountNonExpired boolean (query)	--
accountNonLocked boolean (query)	--
authorities[0].authority string (query)	authorities[0].authority
credentialsNonExpired boolean (query)	--
enabled boolean (query)	--
<b>gradeId</b> * required integer(\$int64) (path)	gradeId gradeId - gradeId
password string (query)	password
username string (query)	username

Rysunek 9. api/v1/grades/{gradeId}

#### 4.4. Students controller

Ścieżki w tej grupie służą nauczycielom do zarządzania uczniami. Są chronione przez autoryzację z użyciem JWT i są dostępne dla nauczyciela.

- ścieżka **GET *students*** służy do otrzymania listy uczniów przez nauczyciela który ma do nich dostęp



The image shows a Swagger UI interface for the GET /api/v1/students endpoint. The interface is divided into two main sections: Parameters and Responses.

**Parameters:** This section lists query parameters for the endpoint. It includes a "Try it out" button. The parameters are:

Name	Description
accountNonExpired <small>boolean (query)</small>	--
accountNonLocked <small>boolean (query)</small>	--
authorities[0].authority <small>string (query)</small>	authorities[0].authority
credentialsNonExpired <small>boolean (query)</small>	--
enabled <small>boolean (query)</small>	--
password <small>string (query)</small>	password
username <small>string (query)</small>	username

**Responses:** This section shows the response for the endpoint. It includes a "Response content type" dropdown set to "\*/".

Code	Description
200	OK

Below the response table, there is a section for "Example Value" and "Model". The example value is a JSON object:

```
{  "emailAddress": "string",  "firstName": "string",  "id": 0,  "lastName": "string"}
```

Rysunek 10. api/v1/students

- ścieżka **POST** *students* przyjmuje dane ucznia w BODY oraz przeznaczona jest do tworzenia konta ucznia przez nauczyciela

POST

/api/v1/students createStudent

Parameters

Try it out

Name	Description
accountNonExpired boolean (query)	<input type="text"/>
accountNonLocked boolean (query)	<input type="text"/>
authorities[0].authority string (query)	<input type="text" value="authorities[0].authority"/>
credentialsNonExpired boolean (query)	<input type="text"/>
enabled boolean (query)	<input type="text"/>
password string (query)	<input type="text" value="password"/>
registerDto * required object (body)	<div>registerDto</div> <div>Example Value   Model</div> <pre>{   "classroomId": 0,   "confirmPassword": "string",   "emailAddress": "string",   "firstName": "string",   "lastName": "string",   "password": "string" }</pre> <div>Parameter content type</div> <div>application/json</div>
username string (query)	<input type="text" value="username"/>

Responses

Response content type

Code	Description
200	<div>OK</div> <div>Example Value   Model</div> <pre>string</pre>
201	Created

Rysunek 11. api/v1/students

- ścieżka **GET *students/{:studentId}*** służy do wglądu do szczegółów ucznia, łącznie z jego ocenami. Wymaga podania parametru studentId, który jest id ucznia w bazie.

GET
/api/v1/students/{studentId}
getStudentById

Parameters
Try it out

Name	Description
accountNonExpired boolean (query)	...
accountNonLocked boolean (query)	...
authorities[0].authority string (query)	authorities[0].authority
credentialsNonExpired boolean (query)	...
enabled boolean (query)	...
password string (query)	password
<b>studentId</b> * required integer(int64) (path)	studentId studentId - studentId
username string (query)	username

Responses
Response content type
\*/

Code	Description
200	OK

Example Value | Model

```

{
  "emailAddress": "string",
  "firstName": "string",
  "id": 0,
  "lastName": "string",
  "studentGrades": [
    {
      "description": "string",
      "id": 0,
      "level": "B00",
      "studentId": 0,
      "subject": "ENGLISH"
    }
  ]
}

```

Rysunek 11. api/v1/students/{studentId}

## 4.5. Student controller.

Endpointy z tej grupy służą uczniom do zobaczenia ich ocen.

- ścieżka **GET grades** zwraca listę ocen przypisanych do ucznia

GET

/api/v1/student/grades

getMyGrades

Try it out

Name	Description
accountNonExpired boolean (query)	--
accountNonLocked boolean (query)	--
authorities[0].authority string (query)	authorities[0].authority
credentialsNonExpired boolean (query)	--
enabled boolean (query)	--
password string (query)	password
username string (query)	username

Responses

Response content type

\*/

Code	Description
200	OK Example Value   Model <pre>[   {     "description": "string",     "id": 0,     "level": "none",     "studentId": 0,     "subject": "ENGLISH"   } ]</pre>

Rysunek 10. api/v1/student/grades



## 5. Repozytorium, system kontroli wersji.

Projekt został zrealizowany z wykorzystaniem systemu kontroli wersji Girt. Na rysunku poniżej przedstawiono zrzut ekranu pokazujący historię komitów. Dokumentacja oraz projekt został umieszczony w repozytorium dostępnym pod adresem: <https://github.com/KopfSzmercen/College-java-project> .

```
Tomek@Tomek-NTT MINGW64 /f/Tomek/C# Projekty/GitLog (master)
$ git lg
0f674b4 - (HEAD -> master) Work 11 (1 second ago) <Tomasz>
b6e2b59 - Work 10 (25 seconds ago) <Tomasz>
8a49655 - Work 9 (28 seconds ago) <Tomasz>
793bd00 - (tag: 1.0.2) Work 8 (6 minutes ago) <Tomasz>
22f3f6a - Work 7 (6 minutes ago) <Tomasz>
a305a4f - (tag: 1.0.1) Work 6 (6 minutes ago) <Tomasz>
4139fa5 - Work 5 (6 minutes ago) <Tomasz>
5e7f3db - Work 4 (6 minutes ago) <Tomasz>
792c770 - Work 3 (6 minutes ago) <Tomasz>
8777de7 - (tag: 1.0.0) Work 2 (7 minutes ago) <Tomasz>
7c91927 - Work 1 (7 minutes ago) <Tomasz>
e00e9af - Some work (7 minutes ago) <Tomasz>
```

Rysunek 1. Historia komitów

## **6. Podsumowanie.**

Projekt realizuje założenia określone na etapie jego planowania. Pozwala na podstawowe zarządzanie klasami, uczniami i ocenami przez nauczycieli. Baza danych PostgreSQL zapewnia stabilność i bezpieczeństwo przechowywania danych. Mechanizm autoryzacji JWT chroni przed dostępem do prywatnych ścieżek przez niezalogowanych użytkowników. Ponadto informacje zawarte w dekodowanym tokenie pozwalają na zabronienie dostępu do zasobów użytkownikom którzy nie powinni mieć do nich wglądu. Zastosowanie autoryzacji JWT zamiast np. sesji pozwala na wykorzystanie API przez aplikacje na różnych platformach a nie ogranicza jego wykorzystania do aplikacji przeglądarkowych.

## 7. Literatura

Materiały źródłowe – wskazanie literatury i materiałów źródłowych wykorzystanych przy realizacji projektu.

1. <https://www.youtube.com/watch?v=KxqlJblhzfI>
2. <https://www.baeldung.com/swagger-2-documentation-for-spring-rest-api>
3. <https://www.youtube.com/watch?v=A8qZUF-GcKo>
4. <https://www.javatpoint.com/spring-boot-tutorial>
5. <https://geshan.com.np/blog/2021/12/docker-postgres>