

# Aplikacja z przepisami

Filip Glinkowski, Emilia Kwolek

## Funkcjonalności aplikacji

Aplikacja będzie posiadać część backendową stworzoną w architekturze REST napisaną w Springu oraz część frontendową napisaną w języku JavaScript z wykorzystaniem technologii Ajax. Aplikacja będzie posiadać funkcjonalność rejestracji, co pozwoli użytkownikom na tworzenie kont, a następnie logowanie.

Użytkownik zalogowany będzie miał możliwość wyświetlania dostępnych przepisów, ale także dodawania, edytowania oraz usuwania swoich własnych. Dodatkowo, możliwe będzie wyświetlanie szczegółów poszczególnych przepisów.

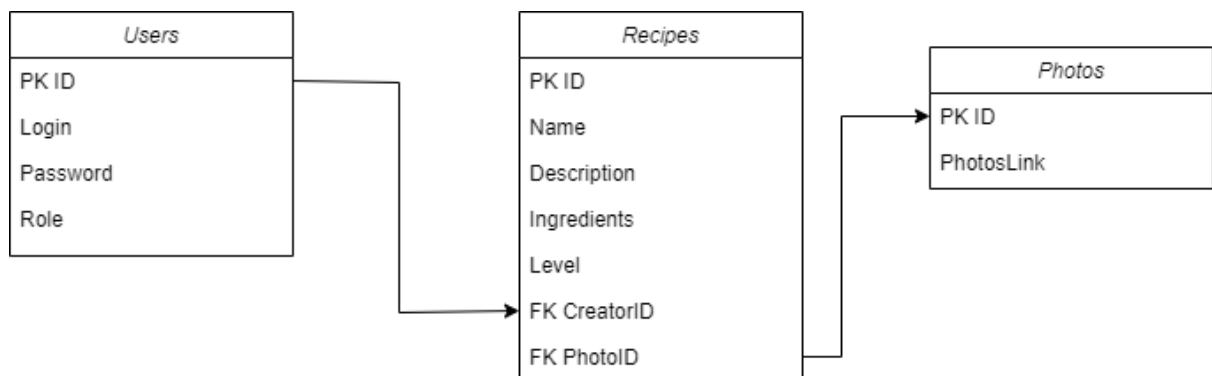
Użytkownik niezalogowany będzie miał możliwość wyświetlania przepisów.

Użytkownik admin będzie miał ponadto możliwość usuwania przepisów innych użytkowników oraz samych użytkowników. Usunięcie użytkownika, który posiada napisane przepisy spowoduje przypisanie dodanych przez niego przepisów do użytkownika admin. Możliwość wyświetlania przepisów oznacza widok dashboardu - spis dodanych przepisów, oraz widok szczegółowy - zawierający szczegóły danego przepisu.

Projekt tworzony jest z zamiarem osiągnięcia oceny 4.5 - dodana jest mechanizm wyjątków w aplikacji servera.

## Schemat bazy

Baza danych zawiera trzy tabele: tabelę użytkowników *users*, tabelę przepisów *recipes* oraz tabelę *photos*. Tabela użytkowników będzie zawierać wszystkich użytkowników aplikacji oraz ich informacje: id, login, hasło i rolę. Rola przechowywana jest jako enum. Tabela przepisów będzie zawierać wszystkie przepisy oraz ich informacje: id przepisu, nazwę przepisu, składniki, opis, poziom trudności oraz id użytkownika, który dodał przepis. Tabela *photos* będzie przechowywała adresy do zdjęć w internecie. Baza napisana jest w postgresql.



# Backend

## Opis przebiegu pracy

Część backendowa została stworzona za pomocą frameworku Spring. Do projektu została podłączona baza danych w technologii PostgreSQL. Na początku utworzono klasy Photo, Recipe, User, które zostały zmapowane do bazy danych do tabel odpowiednio photos, recipes, users. Dla każdej z klas utworzono repozytorium, które będzie umożliwiało zarządzanie wpisami w tabeli. W repozytorium dla tabeli User niezbędne było dodanie metod findAllByLogin oraz existsByLogin. Pierwsza z metod umożliwia odnalezienie użytkownika na podstawie loginu, weryfikuje czy użytkownik o podanym loginie znajduje się obecnie w bazie danych. Repozytorium dla tabeli Recipe zostało rozszerzone o metodę findAllByUser\_Id, która umożliwia zwrócenie wszystkich przepisów użytkownika na podstawie podanego id użytkownika. W aplikacji utworzono dwa serwisy obsługujące zarządzanie użytkownikami oraz zarządzanie przepisami. UserService przeznaczony jest do obsługi użytkowników i posiada następujące metody:

- getAllUsers - zwraca wszystkich użytkowników znajdujących się w bazie danych;
- addUser - tworzenie nowego użytkownika na podstawie loginu oraz hasła, zwraca nowo utworzonego użytkownika.
- deleteUserById - usuwanie użytkownika na podstawie id;

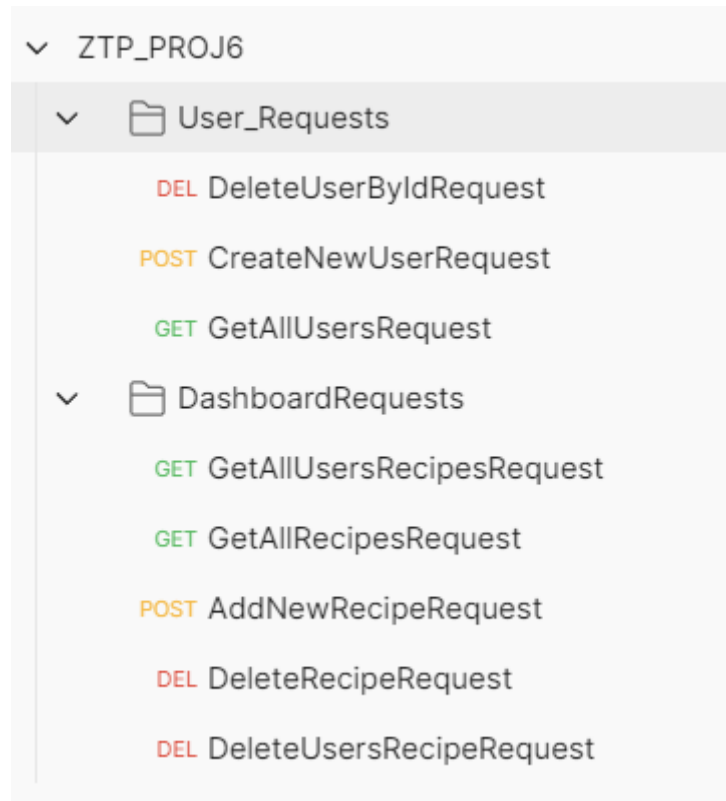
W serwisie DashboardService została zaimplementowana logika do obsługi przepisów, utworzono następujące metody:

- getAllRecipes - zwraca wszystkie przepisy znajdujące się w bazie danych;
- getAllUserRecipes - zwraca wszystkie przepisy użytkownika aktualnie zalogowanego;
- addRecipe - umożliwia dodanie przepisu;
- deleteRecipeById - usuwa przepis na podstawie podanego id;
- deleteUserRecipeById - umożliwia usunięcie przepisu aktualnie zalogowanego użytkownika;
- createPhoto - metoda do automatycznego dodawania wpisu do bazy, przy okazji dodawania przepisu.

W celu umożliwienia dodawania użytkowników/przepisów niezbędne było dodanie klas określających body, które będzie wysyłanie w requestcie. Odpowiednio utworzono klasy UserRequest oraz RecipeRequest. Dodana została obsługa wyjątków poprzez utworzoną klasę ApiExceptionHandler oraz dwa rodzaje customowych wyjątków ForbiddenException i NotFoundException. Wystawienie wszystkich potrzebnych endpointów odbywa się w kontrolerach. W DashboardController zostały wystawione endpoint umożliwiające wysyłanie requestów zarządzających zasobami związanymi z użytkownikami, natomiast UserdashboardController posiada endpoint umożliwiające wysyłanie requestów zarządzających zasobami związanymi z przepisami. Klasą odpowiadającą za logikę dostępu do zasobów jest klasa konfiguracyjna CustomSecurityConfiguration, rozszerza ona klasę WebSecurityConfigurerAdapter. W tej klasie określone są ścieżki dostępu dla poszczególnych ról, a także wskazany jest provider autoryzacyjny.

## Weryfikacja działania za pomocą narzędzia Postman

Zostało wystawionych osiem endpointów, 3 służące do obsługi użytkowników, 5 do obsługi przepisów.



### Usługi służące do zarządzania użytkownikami

Metoda `DeleteUserByIdRequest` służy do usuwania użytkowników na podstawie ID użytkownika przesłanego w adresie URL. Metoda wysyła request typu DELETE. Do tej metody dostęp ma tylko i wyłącznie administrator. Jeżeli użytkownik o przesłanym ID nie znajduje się w bazie to zostanie zwrócony odpowiedni wyjątek. Po prawidłowym usunięciu użytkownika zostanie zwrócona zaktualizowana lista użytkowników.

- Adres endpointu: `http://localhost:8080/RecipeApp/user_dashboard/ID`

### Poprawne usunięcie

ZTP\_PROJ6 / User\_Requests / DeleteUserByIdRequest

Save

**DELETE** ▼ http://localhost:8080/RecipeApp/user\_dashboard/7c70e9e4-5007-463b-9cde-35d24bcd3f6b

Params Authorization ● Headers (8) Body Pre-request Script Tests Settings

Type Basic Auth ▼

The authorization header will be automatically generated when you send the request.  
[Learn more about authorization](#)

Username

Password

☒ Show Password

Body Cookies (1) Headers (17) Test Results ⌚ Status: 200 OK Time: 88 ms Size: 1.35 KB S

Pretty Raw Preview Visualize JSON ▼ ⌵

```
1 {
2   "id": "05509782-bc53-4ae6-8991-5fd8c778f97a",
3   "login": "maria111",
4   "pass": "ma33",
5   "role": "ADMIN"
6 }
7
```

### Komunikat o nieodnalezionym użytkowniku

**DELETE** ▼ http://localhost:8080/RecipeApp/user\_dashboard/7c70e9e4-5007-463b-9cde-35d24bcd3f6b

Params Authorization ● Headers (8) Body Pre-request Script Tests Settings

Type Basic Auth ▼

The authorization header will be automatically generated when you send the request.  
[Learn more about authorization](#)

Username

Password

☒ Show Password

Body Cookies (1) Headers (17) Test Results ⌚ Status: 404 Not Found Time: 8 ms Size: 749 B S

Pretty Raw Preview Visualize JSON ▼ ⌵

```
1 {
2   "message": "User not found!",
3   "httpStatus": "NOT_FOUND",
4   "timestamp": "2022-06-14T11:03:33.7807297+02:00"
5 }
```

Metoda CreateNewUserRequest służy do dodawania użytkowników na podstawie nazwy użytkownika i hasła przesyłanych w body. Metoda wysyła request typu POST. Do tej metody dostęp mają zarówno użytkownicy, jak i administrator. Jeżeli użytkownik o podanej nazwie użytkownika istnieje w bazie danych zostanie zwrócony odpowiedni komunikat. Po poprawnie zakończonej operacji dodawania użytkownika zostaną zwrócone jego dane.

- Adres endpointu:  
[http://localhost:8080/RecipeApp/user\\_dashboard/createNewUser](http://localhost:8080/RecipeApp/user_dashboard/createNewUser)
- Body:

```
{
  "username": "Piotr1234",
  "password": "123Piotr"
}
```

### Poprawne dodanie

**POST** ▼ http://localhost:8080/RecipeApp/user\_dashboard/createNewUser

Params Authorization Headers (9) **Body** ● Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▼

```
1 {
2   ... "username": "Piotr12334",
3   ... "password": "123Piotr"
4 }
```

Body Cookies (1) Headers (17) Test Results 🌐 Status: 200 OK Time: 34 ms Size: 738 B S

Pretty Raw Preview Visualize **JSON** ▼ ⌵

```
1 {
2   "id": "6236a6c2-a4ee-47e9-84a5-6ba1155fb222",
3   "login": "Piotr12334",
4   "pass": "123Piotr",
5   "role": "USER"
6 }
```

### Komunikat o zajętej nazwie użytkownika

**POST** ▼ http://localhost:8080/RecipeApp/user\_dashboard/createNewUser

Params Authorization Headers (9) **Body** ● Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▼

```
1 {
2   ... "username": "Piotr1234",
3   ... "password": "123Piotr"
4 }
```

Body Cookies (1) Headers (17) Test Results 🌐 Status: 403 Forbidden Time: 55 ms Size: 768 B

Pretty Raw Preview Visualize **JSON** ▼ ⌵

```
1 {
2   "message": "User with this login already exists!",
3   "httpStatus": "FORBIDDEN",
4   "timestamp": "2022-06-14T11:10:57.918206+02:00"
5 }
```

Metoda GetAllUsersRequest służy do wyświetlenia listy wszystkich użytkowników. Metoda wysyła request typu GET. Do tej metody dostęp ma tylko administrator.

- Adres endpointu: `http://localhost:8080/RecipeApp/user_dashboard`

#### Lista użytkowników

GET

http://localhost:8080/RecipeApp/user\_dashboard

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Type

Basic Auth

The authorization header will be automatically generated when you send the request.  
[Learn more about authorization](#)

Username

admin

Password

.....

☐ Show Password

Body

Cookies (1)

Headers (17)

Test Results

Status: 200 OK Time: 28 ms Size: 1.44 KB

Pretty

Raw

Preview

Visualize

JSON

```
4      "login": "maria111",
5      "pass": "ma33",
6      "role": "ADMIN"
7    },
8  ],
9    "id": "7c70e9e4-5007-463b-9cde-35d24bcd3f6b",
10   "login": "Test111",
11   "pass": "Testowy",
12   "role": "USER"
13 ],
14 ],
15   "id": "c9f94339-6a66-407e-a567-12836d777628",
16   "login": "admin",
17   "pass": "admin123",
18   "role": "ADMIN"
19 ]
```

## Usługi służące do zarządzania przepisami

Metoda DeleteRecipeRequest służy do usuwania przepisów na podstawie ID przepisu przesłanego w adresie URL. Metoda wysyła request typu DELETE. Do tej metody dostęp ma tylko i wyłącznie administrator. Jeżeli przepis o przesłanym ID nie znajduje się w bazie to zostanie zwrócony odpowiedni wyjątek. Po prawidłowym usunięciu przepisu zostanie zwrócona zaktualizowana lista przepisów.

- Adres endpointu: `http://localhost:8080/RecipeApp/dashboard/admin/`

### Poprawne usunięcie

ZTP\_PROJ6 / DashboardRequests / DeleteRecipeRequest Save

**DELETE** `http://localhost:8080/RecipeApp/dashboard/admin/729c2422-548f-44c4-9862-7dab6564bd1f`

Params Authorization Headers (8) Body Pre-request Script Tests Settings

Type: Basic Auth

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Username: admin

Password: .....

☐ Show Password

Body Cookies (1) Headers (17) Test Results Status: 200 OK Time: 23 ms Size: 6

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": "d065e124-f7e7-420a-a80d-77f523bc61d6",
3   "name": "SPA",
4   "description": "Salami",
5   "ingredients": "salami, cheese",
6   "level": 4,
7   "user": null,
8   "photo": null
9 }
```

### Komunikat o nieodnalezionym przepisie

**DELETE** `http://localhost:8080/RecipeApp/dashboard/admin/9cc6bab4-a0ae-4a5b-92ee-74b1c4a3a2de`

Params Authorization Headers (8) Body Pre-request Script Tests Settings

Type: Basic Auth

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Username: admin

Password: .....

☐ Show Password

Body Cookies (1) Headers (18) Test Results Status: 404 Not Found Time: 33 ms Size: 6

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Recipe not found!",
3   "httpStatus": "NOT_FOUND",
4   "timestamp": "2022-06-14T11:18:41.6616983+02:00"
5 }
```

Metoda AddNewRecipeRequest służy do dodawania przepisów na podstawie nazwy przepisu, opisu, składników, poziomu trudności i adresu url zdjęcia przesyłanych w body. Metoda wysyła request typu POST. Do tej metody dostęp mają zarówno użytkownicy, jak i administrator. Po poprawnie zakończonej operacji dodawania przepisu, zostanie zwrócona zaktualizowana lista przepisów..

- Adres endpointu: `http://localhost:8080/RecipeApp/dashboard`
- Body:

```
{
  "name": "Zapieksha",
  "description": "Zapieksha z ketchupem",
  "ingredients": "",
  "level": "1",
  "photo_url":
    "https://blizejprzedszkola.pl/upload/konkursy/79edc8a56b5ee62644c3e3c5ea2fdfb5.jpg"
}
```

*Poprawne dodanie*

The screenshot displays a REST client interface with a POST request to `http://localhost:8080/RecipeApp/dashboard`. The request body is a JSON object containing recipe details. The response status is 200 OK, and the response body is a JSON object containing the created recipe's ID and the user's details.

**Request:**

```
POST http://localhost:8080/RecipeApp/dashboard
Body
{
  "name": "Zapieksha",
  "description": "Zapieksha z ketchupem",
  "ingredients": "",
  "level": "1",
  "photo_url": "https://blizejprzedszkola.pl/upload/konkursy/79edc8a56b5ee62644c3e3c5ea2fdfb5.jpg"
}
```

**Response:**

```
{
  "id": "2337b8f5-5ac5-4a47-b792-2057f1c742ff",
  "name": "Zapieksha",
  "description": "Zapieksha z ketchupem",
  "ingredients": "",
  "level": 1,
  "user": {
    "id": "b41d41b5-d6dd-4e64-8b4b-26431bc29379",
    "login": "Stefan",
    "pass": "Testowy",
    "role": "USER"
  }
}
```



Metoda GetAllRecipesRequest służy do wyświetlenia listy wszystkich przepisów. Metoda wysyła request typu GET. Do tej metody nie ma żadnych ograniczeń autoryzacyjnych, ponieważ dostęp do listy przepisów mają również niezalogowani użytkownicy.

- Adres endpointu: <http://localhost:8080/RecipeApp/dashboard/all>

*Lista użytkowników*

GET

http://localhost:8080/RecipeApp/user\_dashboard

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Type

Basic Auth

Username

admin

Password

.....

☐ Show Password

The authorization header will be automatically generated when you send the request.

[Learn more about authorization](#)

Body

Cookies (1)

Headers (17)

Test Results

Status: 200 OK

Time: 28 ms

Size: 1.44 KB

Pretty

Raw

Preview

Visualize

JSON

```
4      "login": "maria111",
5      "pass": "ma33",
6      "role": "ADMIN"
7    },
8
9      "id": "7c70e9e4-5007-463b-9cde-35d24bcd3f6b",
10     "login": "Test111",
11     "pass": "Testowy",
12     "role": "USER"
13   },
14
15     "id": "c9f94339-6a66-407e-a567-12836d777628",
16     "login": "admin",
17     "pass": "admin123",
18     "role": "ADMIN"
19   ],
20 }
```

Metoda GetAllUsersRecipesRequest służy do wyświetlenia listy wszystkich przepisów użytkownika, który aktualnie jest zalogowany. Metoda wysyła request typu GET. Do tej metody mają dostęp zarówno użytkownicy zalogowani, jak i administrator..

- Adres endpointu: `http://localhost:8080/RecipeApp/dashboard/user`

#### Lista użytkowników

ZTP\_PROJ6 / DashboardRequests / GetAllUsersRecipesRequest

GET `http://localhost:8080/RecipeApp/dashboard/user`

Params Authorization Headers (8) Body Pre-request Script Tests Settings

Type Basic Auth

The authorization header will be automatically generated when you send the request.  
[Learn more about authorization](#)

Username Stefan

Password Testowy

☒ Show Password

Body Cookies (1) Headers (17) Test Results


Pretty Raw Preview Visualize JSON


```
1 {
2   "id": "9d6f29ae-a430-49ef-8d88-61e4cd7f88d9",
3   "name": "Kanapki",
4   "description": "Kanapki z serem",
5   "ingredients": "4 kromki, ser",
6   "level": 1,
7   "user": {
8     "id": "b41d41b5-d6dd-4e64-8b4b-26431bc29379",
9     "login": "Stefan",
10    "pass": "Testowy",
11    "role": "USER"
12  },
13  "photo": {
14    "id": "aad1a0ee-b623-43c2-916b-9fc6112acd59",
```

Metoda DeleteUsersRecipeRequest służy do usuwania przepisów aktualnie zalogowanego użytkownika na podstawie ID przepisu przesłanego w adresie URL. Metoda wysyła request typu DELETE. Do tej metody dostęp mają użytkownicy zalogowani, jak i administrator. Jeżeli przepis o przesłanym ID nie znajduje się w bazie, lub aktualnie zalogowany użytkownik nie jest jego właścicielem to zostanie zwrócony odpowiedni wyjątek. Po prawidłowym usunięciu przepisu zostanie zwrócona zaktualizowana lista przepisów..


- Adres endpointu: `http://localhost:8080/RecipeApp/dashboard/admin/`


### Poprawne usunięcie

**DELETE**  http://localhost:8080/RecipeApp/dashboard/user/7402bc15-0de0-4923-83d1-cfa1cafc2117

Params **Authorization**  Headers (8) Body Pre-request Script Tests Settings

Type 

Basic Auth 

The authorization header will be automatically generated when you send the request.  
[Learn more about authorization](#) 


Username 



Stefan

Password 

Testowy

☒ Show Password




Body Cookies (1) Headers (17) Test Results  Status: 200 OK Time: 23 ms Size: 7.78 KB


Pretty Raw Preview Visualize JSON  


```
388     "name": "Zapieksha",
389     "description": "Zapieksha z ketchupem",
390     "ingredients": "",
391     "level": 1,
392     "user": {
393       "id": "b41d41b5-d6dd-4e64-8b4b-26431bc29379",
394       "login": "Stefan",
395       "pass": "Testowy",
396       "role": "USER"
397     },
398     "photo": {
399       "id": "2e09e805-0fa1-4364-95de-4559c4491d8f",
400       "url": "https://blizejprzedszkola.pl/upload/konkursy/79edc8a56b5ee62644c3e3c5ea2fdfb5.jpg"
401     }
402   }
```

### Komunikat o nieodnalezionym przepisie


ZTP\_PROJ6 / DashboardRequests / DeleteUsersRecipeRequest


 Save  

**DELETE**  http://localhost:8080/RecipeApp/dashboard/user/7402bc15-0de0-4923-83d1-cfa1cafc2117

Params **Authorization**  Headers (8) Body Pre-request Script Tests Settings

Type 

Basic Auth 

The authorization header will be automatically generated when you send the request.  
[Learn more about authorization](#) 


Username 



Stefan

Password 

Testowy

☒ Show Password

Body Cookies (1) Headers (17) Test Results  Status: 404 Not Found Time: 11 ms Size: 751 B

Pretty Raw Preview Visualize JSON  

```
1  {
2    "message": "Recipe not found!",
3    "httpStatus": "NOT_FOUND",
4    "timestamp": "2022-06-14T11:35:24.9883869+02:00"
5  }
```

*Komunikat podczas próby usunięcia przepisu, którego użytkownik nie jest właścicielem*

DELETE

⌵

http://localhost:8080/RecipeApp/dashboard/user/39ea8563-0ec7-4bfa-9b16-36ba8102c113

Params

Authorization ●

Headers (8)

Body

Pre-request Script

Tests

Settings

Type

Basic Auth ⌵

Username

Stefan

Password

Testowy

☒ Show Password

The authorization header will be automatically generated when you send the request.  
[Learn more about authorization](#) ➤

Body

Cookies (1)

Headers (17)

Test Results

⌕ Status: 403 Forbidden Time: 16 ms Size: 774 B

Pretty

Raw

Preview

Visualize

JSON ⌵

⌵

1

2

3

4

5

```
"message": "You do not have access to this resource!",
"httpStatus": "FORBIDDEN",
"timestamp": "2022-06-14T11:36:44.4431021+02:00"
```