




Work on project. Stage 5/5: More chefs to the table

Project: [Recipes](#)

More chefs to the table

 Hard  6 hours  413 users solved this stage. Latest completion was 1 day ago.

Description

Imagine a service that supports the registration process, can handle a lot of users, and each of them can add their own recipes. Also, a user can update or delete only their recipes but can view recipes added by other users. In this stage, you will implement all this functionality with Spring Boot Security.

The stage is divided into 3 steps. In the first step, you need to add an endpoint responsible for the user registration. The endpoint receives 2 fields: `email` and `password`. The second step is to enable Spring Security and configure the access restrictions – only the registered users with the correct login and password should have the rights to use the service. After that, restrict the deletion and updating to the recipe author only.

Objectives

The service should contain all features from the previous stages. To complete the project, you need to add the following functionality:

- New endpoint `POST /api/register` receives a JSON object with two fields: `email` (string), and `password` (string). If a user with a specified email does not exist, the program saves (registers) the user in a database and responds with `200 (Ok)`. If a user is already in the database, respond with the `400 (Bad Request)` status code. Both fields are **required** and must be **valid**: `email` should contain `@` and `.` symbols, `password` should contain at least 8 characters and shouldn't be blank. If the fields do not meet these restrictions, the service should respond with `400 (Bad Request)`. Also, do not forget to use an encoder before storing a password in a database. `BCryptPasswordEncoder` is a good choice.
- Include the Spring Boot Security dependency and configure access to the endpoints – all implemented endpoints (except `/api/register`) should be available only to the registered and then authenticated and authorized via HTTP Basic auth users. Otherwise, the server should respond with the `401 (Unauthorized)` status code.
- Add additional restrictions – only an author of a recipe can delete or update a recipe. If a user is not the author of a recipe, but they try to carry out the actions mentioned above, the service should respond with the `403 (Forbidden)` status code.

For testing purposes, `POST /actuator/shutdown` should be available without authentication.

Hint

Examples






Example 1: `POST /api/recipe/new` request without authentication

```
{
  "name": "Fresh Mint Tea",
  "category": "beverage",
  "description": "Light, aromatic and refreshing beverage, ...",
  "ingredients": ["boiled water", "honey", "fresh mint leaves"],
  "directions": ["Boil water", "Pour boiling hot water into a mug", "Add fresh mint leaves", "Mix and let the mint 5 minutes", "Add honey and mix again"]
}
```

Status code: `401 (Unauthorized)`

Example 2: `POST /api/register` request without authentication

11 / 11 Prerequisites

- ✓  [Abstract class](#) ✓
- ✓  [Builder](#) ✓
- ✓  [Web security, OWASP](#) ✓
- ✓  [Authentication and Authorization](#) ✓
- ✓  [Getting started with Spring Security](#) ✓

Show all

[Join a study group for the project Recipes](#)

Discuss your current project with fellow learners and help each other.

```
{
  "email": "Cook_Programmer@somewhere.com",
  "password": "RecipeInBinary"
}
```

Status code: **200 (Ok)**

Further **POST /api/recipe/new** request with basic authentication; email (login): Cook_Programmer@somewhere.com, and password: RecipeInBinary

```
{
  "name": "Mint Tea",
  "category": "beverage",
  "description": "Light, aromatic and refreshing beverage, ...",
  "ingredients": ["boiled water", "honey", "fresh mint leaves"],
  "directions": ["Boil water", "Pour boiling hot water into a mug", "Add fresh mint leaves", "Mix and let the mint 5 minutes", "Add honey and mix again"]
}
```

Response:

```
{
  "id": 1
}
```

Further **PUT /api/recipe/1** request with basic authentication; email (login): Cook_Programmer@somewhere.com, password: RecipeInBinary

```
{
  "name": "Fresh Mint Tea",
  "category": "beverage",
  "description": "Light, aromatic and refreshing beverage, ...",
  "ingredients": ["boiled water", "honey", "fresh mint leaves"],
  "directions": ["Boil water", "Pour boiling hot water into a mug", "Add fresh mint leaves", "Mix and let the mint 5 minutes", "Add honey and mix again"]
}
```

Status code: **204 (No Content)**

Further **GET /api/recipe/1** request with basic authentication; email (login): Cook_Programmer@somewhere.com, password: RecipeInBinary

Response:

```
{
  "name": "Fresh Mint Tea",
  "category": "beverage",
  "date": "2020-01-02T12:11:25.034734",
  "description": "Light, aromatic and refreshing beverage, ...",
  "ingredients": ["boiled water", "honey", "fresh mint leaves"],
  "directions": ["Boil water", "Pour boiling hot water into a mug", "Add fresh mint leaves", "Mix and let the mint 5 minutes", "Add honey and mix again"]
}
```

Example 3: **POST /api/register** request without authentication

```
{
  "email": "CamelCaseRecipe@somewhere.com",
  "password": "C00k1es."
}
```

Status code: **200 (Ok)**

Further response for the **GET /api/recipe/1** request with basic authentication; email (login): CamelCaseRecipe@somewhere.com, password: C00k1es.

```
{
  "name": "Fresh Mint Tea",
  "category": "beverage",
  "date": "2020-01-02T12:11:25.034734",
  "description": "Light, aromatic and refreshing beverage, ...",
  "ingredients": ["boiled water", "honey", "fresh mint leaves"],
  "directions": ["Boil water", "Pour boiling hot water into a mug", "Add fresh mint leaves", "Mix and let the mint
5 minutes", "Add honey and mix again"]
}
```

Further `PUT /api/recipe/1` request with basic authentication; email (login): CamelCaseRecipe@somewhere.com, password: C00k1es.

```
{
  "name": "Warming Ginger Tea",
  "category": "beverage",
  "description": "Ginger tea is a warming drink for cool weather, ...",
  "ingredients": ["1 inch ginger root, minced", "1/2 lemon, juiced", "1/2 teaspoon manuka honey"],
  "directions": ["Place all ingredients in a mug and fill with warm water (not too hot so you keep the beneficial h
10 minutes", "Drink and enjoy"]
}
```

Status code: `403 (Forbidden)`

Further `DELETE /api/recipe/1` request with basic authentication; email (login): CamelCaseRecipe@somewhere.com, password: C00k1es.

Status code: `403 (Forbidden)`

 Report a typo

 See hint

↙ Write a program

[Code Editor](#)

[IDE](#)

CONNECTION STATUS

IDE / Checking the plugin's status

[Solve in IDE](#)

[Comments \(52\)](#)

[Hints \(14\)](#)

[Useful links \(3\)](#)

[Solutions \(81\)](#)

[Show discussion](#)