

WinSCP

Spring Boot with Spring

Other connections

Register to Plus

Dark mode

Updates & Help

Logout

1. Add Swagger documentation to your pom.xml file

```
<dependencies>
    <dependency>
        <groupId>org.springdoc
        <artifactId>springdoc-openapi-swagger</artifactId>
        <version>1.6.9</version>
    </dependency>
</dependencies>

<dependency>
    <groupId>org.springdoc
    <artifactId>springdoc-openapi-ui</artifactId>
    <version>1.6.9</version>
</dependency>
```

2. Create a new Java class and annotate it with `@SwaggerGenerator` and `@SwaggerModel` annotations

```
@SwaggerGenerator
@SwaggerModel
public class SwaggerConfig {
    // Swagger configuration class
}
```

3. Configure Swagger using `Swagger` class

```
@Configuration
public class SwaggerConfig {

    @Bean
    public Swagger api() {
        return new SwaggerBuilder().apiInfo(apiInfo())
            .build();
    }

    private ApiInfo apiInfo() {
        return new ApiInfoBuilder()
            .title("Swagger API")
            .description("Swagger API")
            .version("1.0.0")
            .build();
    }
}
```

4. Run the Spring Boot application and open the Swagger UI in a web browser

```
http://localhost:8080/swagger-ui.html
```

Swagger UI page will open and it will show the list of all the RESTful APIs in your application. You can test these APIs from the Swagger UI itself.

By following these simple steps, you can easily integrate Swagger with your Spring Boot project and generate API documentation and test your APIs from a web interface.

1. Add OpenAPI dependency to your pom.xml file

```
<dependencies>
    <dependency>
        <groupId>org.springdoc
        <artifactId>springdoc-openapi-swagger</artifactId>
        <version>1.6.9</version>
    </dependency>
</dependencies>
```

2. Create a new Java class and annotate it with `@SwaggerGenerator` annotation

```
@SwaggerGenerator
public class SwaggerConfig {
    // Swagger configuration class
}
```

3. Configure OpenAPI using `@SwaggerGenerator` and `@SwaggerModel` annotations

```
@SwaggerGenerator
@SwaggerModel
public class SwaggerConfig {

    @Bean
    public Swagger api() {
        return new SwaggerBuilder().apiInfo(apiInfo())
            .build();
    }

    private ApiInfo apiInfo() {
        return new ApiInfoBuilder()
            .title("Swagger API")
            .description("Swagger API")
            .version("1.0.0")
            .build();
    }
}
```

4. Define API endpoints using `@SwaggerModel` and `@SwaggerResponse` annotations

```
@SwaggerModel
@SwaggerResponse
public class SwaggerConfig {

    @Bean
    public Swagger api() {
        return new SwaggerBuilder().apiInfo(apiInfo())
            .build();
    }

    private ApiInfo apiInfo() {
        return new ApiInfoBuilder()
            .title("Swagger API")
            .description("Swagger API")
            .version("1.0.0")
            .build();
    }
}
```

5. Run the Spring Boot application and open the OpenAPI UI in a web browser

```
http://localhost:8080/swagger-ui.html
```

OpenAPI UI page will open and it will show the list of all the RESTful APIs in your application. You can test these APIs from the OpenAPI UI itself.

By following these simple steps, you can easily integrate OpenAPI with your Spring Boot project and generate API documentation and test your APIs from a web interface.

1. Create a new Java class and annotate it with `@SwaggerGenerator` annotation

```
@SwaggerGenerator
public class SwaggerConfig {
    // Swagger configuration class
}
```

2. Add the `springdoc-openapi-ui` and `springdoc-openapi-swagger` dependencies to your pom.xml file

```
<dependencies>
    <dependency>
        <groupId>org.springdoc
        <artifactId>springdoc-openapi-swagger</artifactId>
        <version>1.6.9</version>
    </dependency>
</dependencies>

<dependency>
    <groupId>org.springdoc
    <artifactId>springdoc-openapi-ui</artifactId>
    <version>1.6.9</version>
</dependency>
```

3. Configure Swagger security using `@SwaggerModel` and `@SwaggerRequirements` annotations

```
@SwaggerModel
@SwaggerRequirements
public class SwaggerConfig {

    @Bean
    public Swagger api() {
        return new SwaggerBuilder().apiInfo(apiInfo())
            .build();
    }

    private ApiInfo apiInfo() {
        return new ApiInfoBuilder()
            .title("Swagger API")
            .description("Swagger API")
            .version("1.0.0")
            .build();
    }
}
```

4. Use JWT Token for authentication in your API endpoints

```
@SwaggerModel
@SwaggerResponse
public class SwaggerConfig {

    @Bean
    public Swagger api() {
        return new SwaggerBuilder().apiInfo(apiInfo())
            .build();
    }

    private ApiInfo apiInfo() {
        return new ApiInfoBuilder()
            .title("Swagger API")
            .description("Swagger API")
            .version("1.0.0")
            .build();
    }
}
```

By following these steps, you can easily implement Swagger with JWT Token authentication in your Spring Boot project.

Register to Plus

Dark mode

Updates & Help

Logout

WinSCP

Spring Boot with Spring

Other connections

Register to Plus

Dark mode

Updates & Help

Logout

https://chat.openai.com/chat

1/1