**1. Overview**

Nowadays, front-end and back-end components often separate a web application. Usually, we expose APIs as a back-end component for the front-end component or third-party app integrations.

In such a scenario, it is essential to have proper specifications for the back-end APIs. At the same time, the API documentation should be informative, readable, and easy to follow.

Moreover, reference documentation should simultaneously describe every change in the API. Accomplishing this manually is a tedious exercise, so automation of the process was inevitable.

In this tutorial, we'll look at Swagger 2 for a Spring REST web service, using the Springfox implementation of the Swagger 2 specification.

**2.Adding the Maven Dependency**

**- we need to add following 2 dependencies for swagger in pom.xml**

<dependency>

<groupId>io.springfox</groupId>

<artifactId>springfox-swagger2</artifactId>

<version>2.9.2</version>

</dependency>

<dependency>

<groupId>io.springfox</groupId>

<artifactId>springfox-swagger-ui</artifactId>

<version>2.9.2</version>

</dependency>

## **3.Integrating Swagger 2 Into the Project**

The configuration of Swagger mainly centers around the Docket bean:

Add the below code into main method-

@Bean

**public** Docket api() {

**return** **new** Docket(DocumentationType.***SWAGGER\_2***).select()

.apis((Predicate<RequestHandler>) RequestHandlerSelectors.*any*())

.paths((Predicate<String>) PathSelectors.*any*()).build();

}

}

Now, add the @EnableSwagger2 annotation in your main Spring Boot application. The @EnableSwagger2 annotation is used to enable the Swagger2 for your Spring Boot application.

The code for main Spring Boot application is shown below −



Now, the code to build two simple RESTful web services GET and POST in Rest Controller file is shown here −

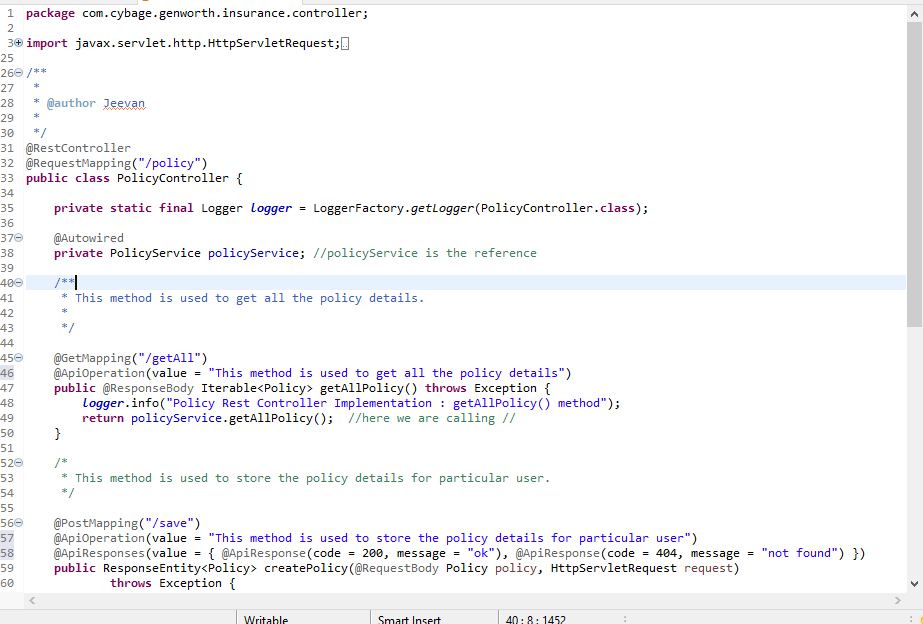
Note- adding below code is optional as it will just use to provide more information about our rest api to end user.

@ApiOperation(value="Use to create a policy for a user", httpMethod="Post")

@ApiResponses(value = {

@ApiResponse(code = 200, message = "ok"),

@ApiResponse(code = 404, message = "not found")})



1. **Finally pom.xml looks like this-**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<project xmlns=*"http://maven.apache.org/POM/4.0.0"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>com.cybage.genworth.insurance</groupId>

<artifactId>LifeInsurance</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>war</packaging>

<name>Insurance-Rest</name>

<description>Rest API for Life Insurance</description>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.3.4.RELEASE</version>

<relativePath /> <!-- lookup parent from repository -->

</parent>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-actuator</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>io.springfox</groupId>

<artifactId>springfox-swagger2</artifactId>

<version>2.9.2</version>

</dependency>

<dependency>

<groupId>io.springfox</groupId>

<artifactId>springfox-swagger-ui</artifactId>

<version>2.9.2</version>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.12</version><!--$NO-MVN-MAN-VER$ -->

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-tomcat</artifactId>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-mail</artifactId>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-devtools -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<version>2.3.9.RELEASE</version><!--$NO-MVN-MAN-VER$ -->

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

1. **Now run your springboot application**
2. **Enter this URL in any browser to get Swagger-UI-**

**<http://localhost:8080/swagger-ui.html>**

**Note- make sure your port no is 8080**

1. **Swagger-ui will look like this-**

