**SPRING REST USING SPRING BOOT 3**

**1.Create a Spring Web Project using Maven** 

Steps followed to create a project:

1. Go to <https://start.spring.io/>
2. Change Group as “com.cognizant”
3. Change Artifact Id as “spring-learn”
4. Select Spring Boot DevTools and Spring Web
5. Create and download the project as zip
6. Extract the zip in root folder to IntelliJIdea Workspace
7. Include logs to verify if main() method of SpringLearnApplication.
8. Run the SpringLearnApplication class.

**Code snippets:**

**SpringLearnApplication.java:**

package com.cognizant.spring\_learn;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class SpringLearnApplication {  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(SpringLearnApplication.class, args);  
 }  
  
}

**pom.xml:**

<?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 https://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>3.5.3</version>  
 <relativePath/> <!-- lookup parent from repository -->  
 </parent>  
 <groupId>com.cognizant</groupId>  
 <artifactId>spring-learn</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>spring-learn</name>  
 <description>Demo project for Spring Boot</description>  
 <url/>  
 <licenses>  
 <license/>  
 </licenses>  
 <developers>  
 <developer/>  
 </developers>  
 <scm>  
 <connection/>  
 <developerConnection/>  
 <tag/>  
 <url/>  
 </scm>  
 <properties>  
 <java.version>21</java.version>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
 <scope>runtime</scope>  
 <optional>true</optional>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>

Output:

The project has executed successfully.

**2. Spring Core – Load Country from Spring Configuration XML**   
  
An airlines website is going to support booking on four countries. There will be a drop down on the home page of this website to select the respective country. It is also important to store the two-character ISO code of each country.

|  |  |
| --- | --- |
| **Code** | **Name** |
| US | United States |
| DE | Germany |
| IN | India |
| JP | Japan |

Above data has to be stored in spring configuration file. Write a program to read this configuration file and display the details.  
  
Steps to implement

* Pick any one of your choice country to configure in Spring XML configuration named country.xml.
* Create a bean tag in spring configuration for country and set the property and values.
* Create Country class with following aspects:
  + Instance variables for code and name
  + Implement empty parameter constructor with inclusion of debug log within the constructor with log message as “Inside Country Constructor.”
  + Generate getters and setters with inclusion of debug with relevant message within each setter and getter method.
  + Generate toString() method
* Create a method displayCountry() in SpringLearnApplication.java, which will read the country bean from spring configuration file and display the country details. ClassPathXmlApplicationContext, ApplicationContext and context.getBean(“beanId”, Country.class).
* Invoke displayCountry() method in main() method of SpringLearnApplication.java.
* Execute main() method and check the logs to find out which constructors and methods were invoked.

**Code snippets:**

**SpringLearnApplication.java:**

package com.cognizant.spring\_learn;  
  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;

@SpringBootApplication  
public class SpringLearnApplication {  
 private static Country *country*;  
 static final Logger *LOGGER*= LoggerFactory.*getLogger*(SpringLearnApplication.class);  
 public static void main(String[] args) {  
 ApplicationContext context=new ClassPathXmlApplicationContext("country.xml");  
 *country* = (Country) context.getBean("country", Country.class);  
 *LOGGER*.debug("Country : {}", *country*.toString());  
 *displayCountry*();  
 }  
 public static void displayCountry(){  
 System.*out*.println(*country*);  
 }  
}

**Pom.xml:**

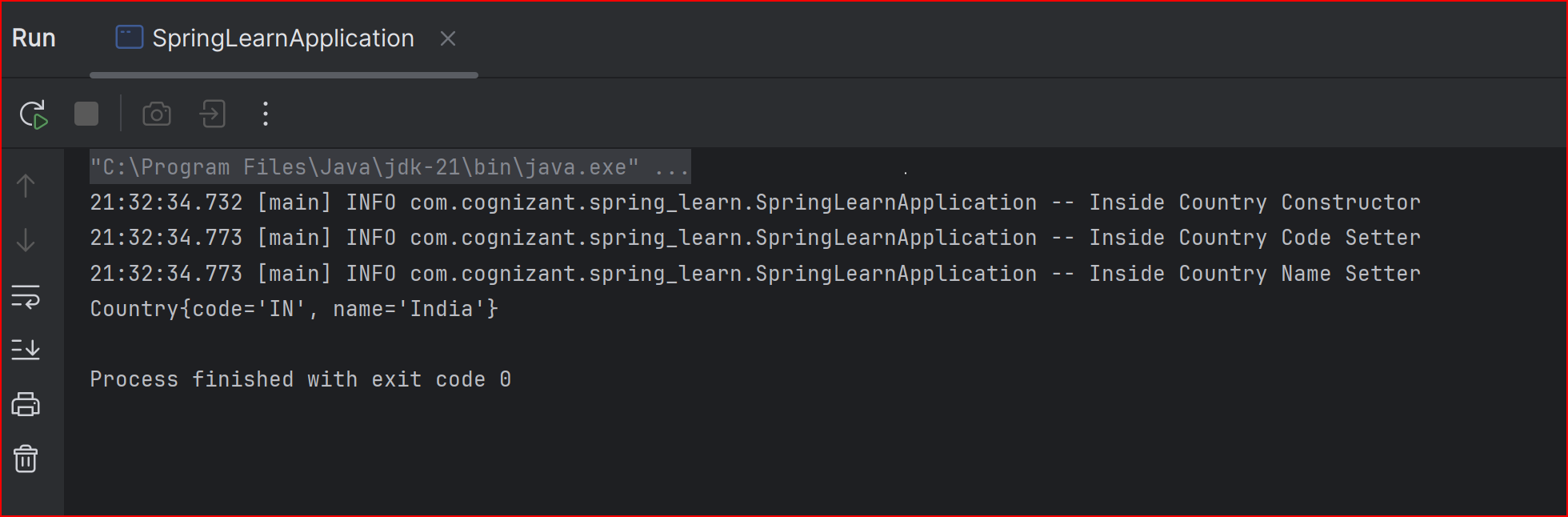
<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>3.5.3</version>  
 <relativePath/> <!-- lookup parent from repository -->  
 </parent>  
 <groupId>com.cognizant</groupId>  
 <artifactId>spring-learn</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>spring-learn</name>  
 <description>Demo project for Spring Boot</description>  
 <url/>  
 <licenses>  
 <license/>  
 </licenses>  
 <developers>  
 <developer/>  
 </developers>  
 <scm>  
 <connection/>  
 <developerConnection/>  
 <tag/>  
 <url/>  
 </scm>  
 <properties>  
 <java.version>21</java.version>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
 <scope>runtime</scope>  
 <optional>true</optional>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>

**Country.java:**

package com.cognizant.spring\_learn;  
  
import static com.cognizant.spring\_learn.SpringLearnApplication.*LOGGER*;  
  
public class Country {  
 private String code;  
 private String name;  
 Country(){  
 *LOGGER*.info("Inside Country Constructor");  
 }  
  
 public String getCode() {  
 *LOGGER*.info("Inside Country Name Getter");  
 return code;  
 }  
  
 public void setCode(String code) {  
 *LOGGER*.info("Inside Country Code Setter");  
 this.code = code;  
 }  
  
 public String getName() {  
 *LOGGER*.info("Inside Country Name Getter");  
 return name;  
 }  
  
 public void setName(String name) {  
 *LOGGER*.info("Inside Country Name Setter");  
 this.name = name;  
 }  
  
 @Override  
 public String toString() {  
 return "Country{" +  
 "code='" + code + '\'' +  
 ", name='" + name + '\'' +  
 '}';  
 }  
}

**Country.xml:**

<?xml version="1.0" encoding="UTF-8"?>  
<beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd">  
  
 <!-- Define a bean -->  
 <bean id="country" class="com.cognizant.spring\_learn.Country">  
 <property name="code" value="IN"/>  
 <property name="name" value="India"/>  
 </bean>  
  
</beans>

**Output:**

**3. Hello World RESTful Web Service**   
  
Write a REST service in the spring learn application created earlier, that returns the text "Hello World!!" using Spring Web Framework. Refer details below:  
**Method:** GET  
**URL:** /hello  
**Controller:** com.cognizant.spring-learn.controller.HelloController  
**Method Signature:** public String sayHello()  
**Method Implementation:** return hard coded string "Hello World!!"  
**Sample Request**: http://localhost:8083/hello  
**Sample Response:** Hello World!!

**Code snippets:**

**SpringLearnApplication.java:**

package com.cognizant.spring\_learn;  
  
import com.cognizant.spring\_learn.Controller.HelloController;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
@SpringBootApplication  
public class SpringLearnApplication {  
 public static void main(String[] args) {  
 SpringApplication.*run*(SpringLearnApplication.class,args);  
 }  
}

**pom.xml:**

<?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 https://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>3.5.3</version>  
 <relativePath/> <!-- lookup parent from repository -->  
 </parent>  
 <groupId>com.cognizant</groupId>  
 <artifactId>spring-learn</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>spring-learn</name>  
 <description>Demo project for Spring Boot</description>  
 <url/>  
 <licenses>  
 <license/>  
 </licenses>  
 <developers>  
 <developer/>  
 </developers>  
 <scm>  
 <connection/>  
 <developerConnection/>  
 <tag/>  
 <url/>  
 </scm>  
 <properties>  
 <java.version>21</java.version>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
 <scope>runtime</scope>  
 <optional>true</optional>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>

**HelloController.java:**

package com.cognizant.spring\_learn.Controller;  
  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.RestController;  
  
@RestController  
@Controller  
public class HelloController {  
 @GetMapping("/Hello")  
 public String sayHello(){  
 return "Hello World!!!";  
 }  
}

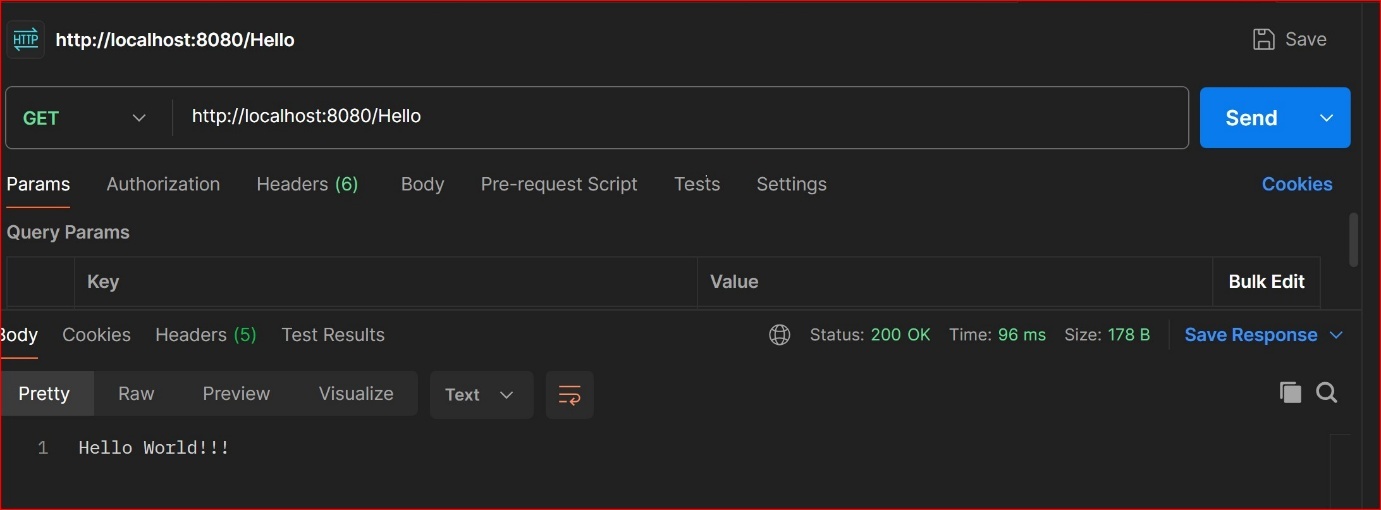
**Country.java:**

package com.cognizant.spring\_learn;  
  
public class Country {  
 private String code;  
 private String name;  
 Country(){  
  
 }  
  
 public String getCode() {  
 return code;  
 }  
  
 public void setCode(String code) {  
 this.code = code;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 @Override  
 public String toString() {  
 return "Country{" +  
 "code='" + code + '\'' +  
 ", name='" + name + '\'' +  
 '}';  
 }  
}

**country.xml:**

<?xml version="1.0" encoding="UTF-8"?>  
<beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd">  
  
 <!-- Define a bean -->  
 <bean id="controller" class="com.cognizant.spring\_learn.Controller.HelloController">  
  
 </bean>  
 <bean id="country" class="com.cognizant.spring\_learn.Country">  
 <property name="code" value="IN"/>  
 <property name="name" value="India"/>  
 </bean>  
  
</beans>

**Output:**

****

**4.REST - Country Web Service**   
  
Write a REST service that returns India country details in the earlier created spring learn application.  
  
**URL**: /country  
**Controller**: com.cognizant.spring-learn.controller.CountryController  
**Method Annotation**: @RequestMapping  
**Method Name**: getCountryIndia()  
**Method Implementation**: Load India bean from spring xml configuration and return  
**Sample Request**: http://localhost:8083/country  
**Sample Response**:

{

  "code": "IN",

  "name": "India"

}

**Code snippets:**

**SpringLearnApplication.java:**

package com.cognizant.spring\_learn;  
  
import com.cognizant.spring\_learn.Controller.HelloController;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.annotation.ImportResource;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
@SpringBootApplication  
@ImportResource("country.xml")  
public class SpringLearnApplication {  
 public static void main(String[] args) {  
 SpringApplication.*run*(SpringLearnApplication.class,args);  
 }  
}

**pom.xml:**

<?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 https://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>3.5.3</version>  
 <relativePath/> <!-- lookup parent from repository -->  
 </parent>  
 <groupId>com.cognizant</groupId>  
 <artifactId>spring-learn</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>spring-learn</name>  
 <description>Demo project for Spring Boot</description>  
 <url/>  
 <licenses>  
 <license/>  
 </licenses>  
 <developers>  
 <developer/>  
 </developers>  
 <scm>  
 <connection/>  
 <developerConnection/>  
 <tag/>  
 <url/>  
 </scm>  
 <properties>  
 <java.version>21</java.version>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
 <scope>runtime</scope>  
 <optional>true</optional>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>

**country.xml:**

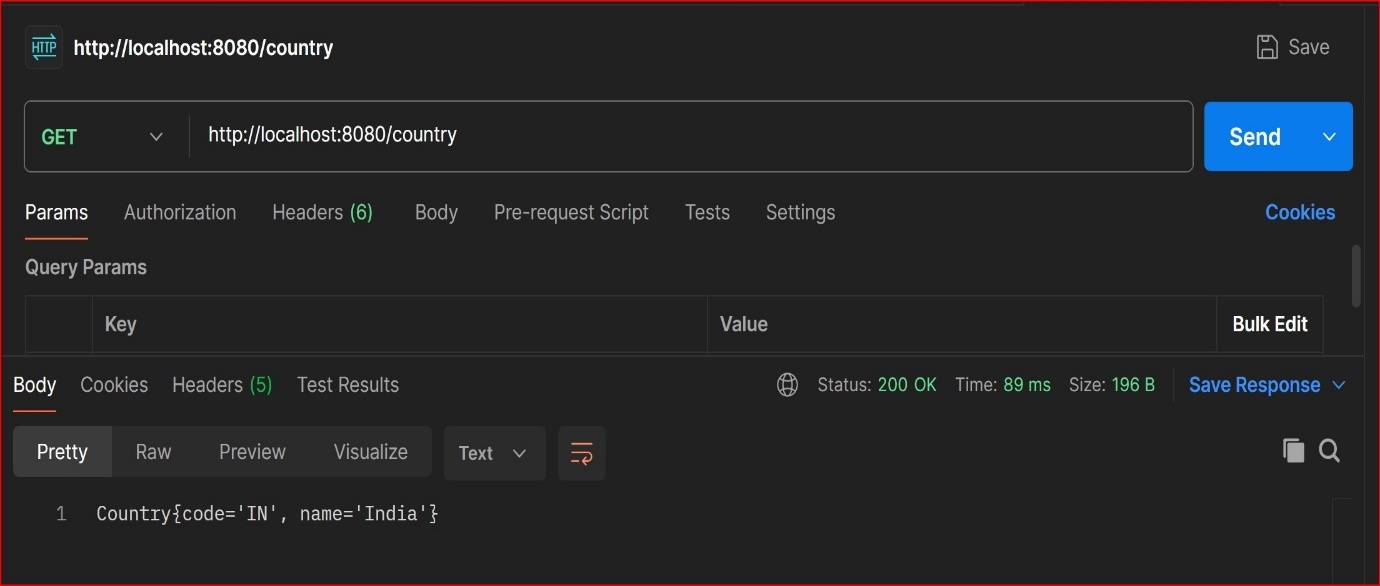
<?xml version="1.0" encoding="UTF-8"?>  
<beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd">  
  
 <!-- Define a bean -->  
 <bean id="country" class="com.cognizant.spring\_learn.Country" autowire="byName">  
 <property name="code" value="IN"/>  
 <property name="name" value="India"/>  
 </bean>  
  
</beans>

**Country.java:**

package com.cognizant.spring\_learn;  
  
public class Country {  
 private String code;  
 private String name;  
 Country(){  
  
 }  
  
 public String getCode() {  
 return code;  
 }  
  
 public void setCode(String code) {  
 this.code = code;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 @Override  
 public String toString() {  
 return "Country{" +  
 "code='" + code + '\'' +  
 ", name='" + name + '\'' +  
 '}';  
 }  
}

**HelloController.java:**

package com.cognizant.spring\_learn.Controller;  
  
import com.cognizant.spring\_learn.Country;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.RestController;  
  
@RestController  
public class HelloController {  
  
 private final Country country;  
  
 @Autowired  
 public HelloController(Country country) {  
 this.country = country;  
 }  
  
 @GetMapping("/hello")  
 public String sayHello() {  
 return "Hello World!!!";  
 }  
  
 @GetMapping("/country")  
 public String getIndia() {  
 return country.toString();  
 }  
}

**Output:**

**5. REST - Get country based on country code**   
  
Write a REST service that returns a specific country based on country code. The country code should be case insensitive.  
  
**Controller**: com.cognizant.spring-learn.controller.CountryController  
**Method Annotation:** @GetMapping("/countries/{code}")  
**Method Name**: getCountry(String code)  
**Method Implemetation**: Invoke countryService.getCountry(code)   
**Service Method:**com.cognizant.spring-learn.service.CountryService.getCountry(String code)  
  
**Service Method Implementation**:

* Get the country code using @PathVariable
* Get country list from country.xml
* Iterate through the country list
* Make a case insensitive matching of country code and return the country.
* Lambda expression can also be used instead of iterating the country list

**Sample Request**: http://localhost:8083/country/in  
  
**Sample Response**:

{

  "code": "IN",

  "name": "India"

}

**Code Snippets:**

**SpringLearnApplication.java:**

package com.cognizant.spring\_learn;  
  
import com.cognizant.spring\_learn.Controller.HelloController;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.annotation.ImportResource;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
@SpringBootApplication  
@ImportResource("country.xml")  
public class SpringLearnApplication {  
 public static void main(String[] args) {  
 SpringApplication.*run*(SpringLearnApplication.class,args);  
 }  
}

**pom.xml:**

<?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 https://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>3.5.3</version>  
 <relativePath/> <!-- lookup parent from repository -->  
 </parent>  
 <groupId>com.cognizant</groupId>  
 <artifactId>spring-learn</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>spring-learn</name>  
 <description>Demo project for Spring Boot</description>  
 <url/>  
 <licenses>  
 <license/>  
 </licenses>  
 <developers>  
 <developer/>  
 </developers>  
 <scm>  
 <connection/>  
 <developerConnection/>  
 <tag/>  
 <url/>  
 </scm>  
 <properties>  
 <java.version>21</java.version>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
 <scope>runtime</scope>  
 <optional>true</optional>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>

**HelloController.java:**

package com.cognizant.spring\_learn.Controller;  
import com.cognizant.spring\_learn.Country;  
import com.cognizant.spring\_learn.Service.CountryService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.PathVariable;  
import org.springframework.web.bind.annotation.RestController;  
import java.util.List;  
@RestController  
public class HelloController {  
 private CountryService service;  
 @Autowired  
 public HelloController(CountryService service){  
 this.service=service;  
 }  
 @GetMapping("/country/{code}")  
 public Country getDetails(@PathVariable String code){  
 return service.getCountry(code);  
 }  
}

**CountryService.java:**

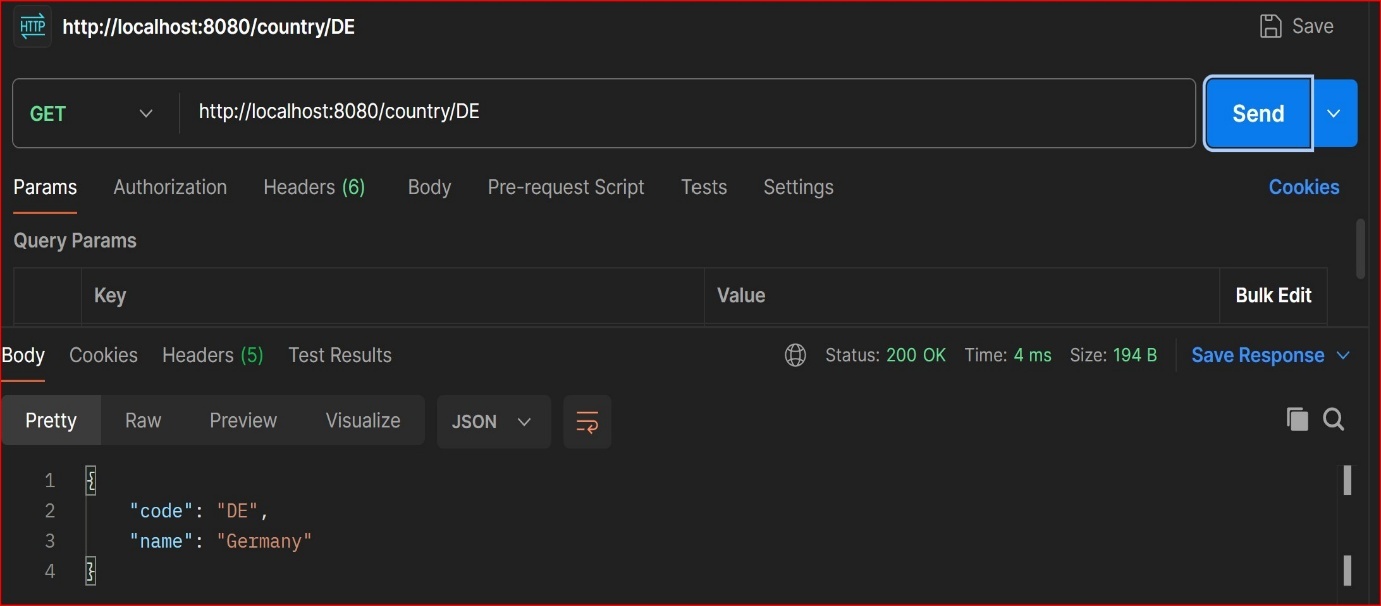
package com.cognizant.spring\_learn.Service;  
  
import com.cognizant.spring\_learn.Country;  
  
import java.util.List;  
import java.util.Optional;  
  
public class CountryService {  
 private List<Country> countryList;  
  
 public void setCountryList(List<Country> countryList) {  
 this.countryList = countryList;  
 }  
  
 public List<Country> getAllCountries() {  
 return countryList;  
 }  
 public Country getCountry(String code){  
 for(Country c:countryList){  
 if(c.getCode().equalsIgnoreCase(code)) return c;  
 }  
 return null;  
 }  
}

**Country.java:**

package com.cognizant.spring\_learn;  
  
public class Country {  
 private String code;  
 private String name;  
 Country(){}  
  
 public String getCode() {  
 return code;  
 }  
  
 public void setCode(String code) {  
 this.code = code;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 @Override  
 public String toString() {  
 return "Country{" +  
 "code='" + code + '\'' +  
 ", name='" + name + '\'' +  
 '}';  
 }  
}

**Country.xml:**

<beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="  
 http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd">  
  
 <!-- Define country beans -->  
 <bean id="country1" class="com.cognizant.spring\_learn.Country">  
 <property name="code" value="IN"/>  
 <property name="name" value="India"/>  
 </bean>  
 <bean id="country2" class="com.cognizant.spring\_learn.Country">  
 <property name="code" value="US"/>  
 <property name="name" value="United States"/>  
 </bean>  
 <bean id="country3" class="com.cognizant.spring\_learn.Country">  
 <property name="code" value="JP"/>  
 <property name="name" value="Japan"/>  
 </bean>  
 <bean id="country4" class="com.cognizant.spring\_learn.Country">  
 <property name="code" value="DE"/>  
 <property name="name" value="Germany"/>  
 </bean>  
 <!-- List bean -->  
 <bean id="countryList" class="java.util.ArrayList">  
 <constructor-arg>  
 <list>  
 <ref bean="country1"/>  
 <ref bean="country2"/>  
 <ref bean="country3"/>  
 <ref bean="country4"/>  
 </list>  
 </constructor-arg>  
 </bean>  
 <bean id="service" class="com.cognizant.spring\_learn.Service.CountryService" autowire="byName"></bean>  
</beans>

**Output:**

**6. Create authentication service that returns JWT**

**Request**

curl -s -u user:pwd http://localhost:8090/authenticate

**Response**

{"token":"eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ1c2VyIiwiaWF0IjoxNTcwMzc5NDc0LCJleHAiOjE1NzAzODA2NzR9.t3LRvlCV-hwKfoqZYlaVQqEUiBloWcWn0ft3tgv0dL0"}

**Request**

curl -s -u user:pwd http://localhost:8090/authenticate

**Response**

{"token":"eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ1c2VyIiwiaWF0IjoxNTcwMzc5NDc0LCJleHAiOjE1NzAzODA2NzR9.t3LRvlCV-hwKfoqZYlaVQqEUiBloWcWn0ft3tgv0dL0"}

This can be incorporated as three major steps:

* Create authentication controller and configure it in SecurityConfig
* Read Authorization header and decode the username and password
* Generate token based on the user retrieved in the previous step

**Code snippets:**

**jwtDemoApplication.java:**

package com.example.jwtDemo;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class JwtDemoApplication {  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(JwtDemoApplication.class, args);  
 }  
  
}

**AuthenticationController.java:**

package com.example.jwtDemo.Controller;  
  
import io.jsonwebtoken.JwtBuilder;  
import io.jsonwebtoken.Jwts;  
import io.jsonwebtoken.SignatureAlgorithm;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.RequestHeader;  
import org.springframework.web.bind.annotation.RestController;  
  
import java.util.Base64;  
import java.util.Date;  
import java.util.HashMap;  
import java.util.Map;  
  
@RestController  
public class AuthenticationController {  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(AuthenticationController.class);  
  
 @GetMapping("/authenticate")  
 public Map<String, String> authenticate(@RequestHeader("Authorization") String authHeader) {  
 *LOGGER*.info("Start /authenticate");  
 *LOGGER*.debug("Authorization Header: {}", authHeader);  
  
 Map<String, String> map = new HashMap<>();  
 String user = getUser(authHeader);  
 *LOGGER*.debug("Extracted user: {}", user);  
 String token = generateJwt(user);  
  
 map.put("token", token);  
 *LOGGER*.info("End /authenticate");  
 return map;  
 }  
  
 private String getUser(String authHeader) {  
 String encoded = authHeader.substring("Basic ".length());  
 byte[] decoded = Base64.*getDecoder*().decode(encoded);  
 String[] parts = new String(decoded).split(":");  
 return parts[0]; // return username  
 }  
  
 private String generateJwt(String user) {  
 JwtBuilder builder = Jwts.*builder*();  
 builder.setSubject(user);  
 builder.setIssuedAt(new Date());  
 builder.setExpiration(new Date(System.*currentTimeMillis*() + 20 \* 60 \* 1000));  
 builder.signWith(SignatureAlgorithm.*HS256*, "secretkey");  
 return builder.compact();  
 }  
}

**CountryController.java:**

package com.example.jwtDemo.Controller;  
  
import org.springframework.web.bind.annotation.\*;  
import java.util.List;  
  
@RestController  
public class CountryController {  
  
 @GetMapping("/countries")  
 public List<String> getCountries() {  
 return List.*of*("India", "USA", "Germany", "Japan");  
 }  
}

**SecurityConfig.java:**

package com.example.jwtDemo.Security;  
  
import com.example.jwtDemo.Security.JwtAuthorizationFilter;  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
import org.springframework.security.authentication.\*;  
import org.springframework.security.authentication.dao.DaoAuthenticationProvider;  
import org.springframework.security.config.Customizer;  
import org.springframework.security.config.annotation.web.builders.HttpSecurity;  
import org.springframework.security.core.userdetails.\*;  
import org.springframework.security.crypto.password.\*;  
import org.springframework.security.provisioning.InMemoryUserDetailsManager;  
import org.springframework.security.web.SecurityFilterChain;  
import org.springframework.security.web.authentication.UsernamePasswordAuthenticationFilter;  
  
import java.util.List;  
  
@Configuration  
public class SecurityConfig {  
  
 @Bean  
 public UserDetailsService userDetailsService() {  
 UserDetails user = User.*withUsername*("user")  
 .password(passwordEncoder().encode("pwd"))  
 .roles("USER")  
 .build();  
  
 UserDetails admin = User.*withUsername*("admin")  
 .password(passwordEncoder().encode("admin"))  
 .roles("ADMIN")  
 .build();  
  
 return new InMemoryUserDetailsManager(user, admin);  
 }  
  
 @Bean  
 public AuthenticationManager authenticationManager(UserDetailsService userDetailsService) {  
 DaoAuthenticationProvider provider = new DaoAuthenticationProvider();  
 provider.setUserDetailsService(userDetailsService);  
 provider.setPasswordEncoder(passwordEncoder());  
 return new ProviderManager(List.*of*(provider));  
 }  
  
 @Bean  
 public SecurityFilterChain filterChain(HttpSecurity http, AuthenticationManager authManager) throws Exception {  
 return http  
 .csrf(csrf -> csrf.disable())  
 .httpBasic(Customizer.*withDefaults*())  
 .authorizeHttpRequests(auth -> auth  
 .requestMatchers("/authenticate").hasAnyRole("USER", "ADMIN")  
 .anyRequest().authenticated()  
 )  
 .addFilterBefore(new JwtAuthorizationFilter(authManager), UsernamePasswordAuthenticationFilter.class)  
 .build();  
 }  
  
 @Bean  
 public PasswordEncoder passwordEncoder() {  
 return NoOpPasswordEncoder.*getInstance*(); // For development only  
 }  
}

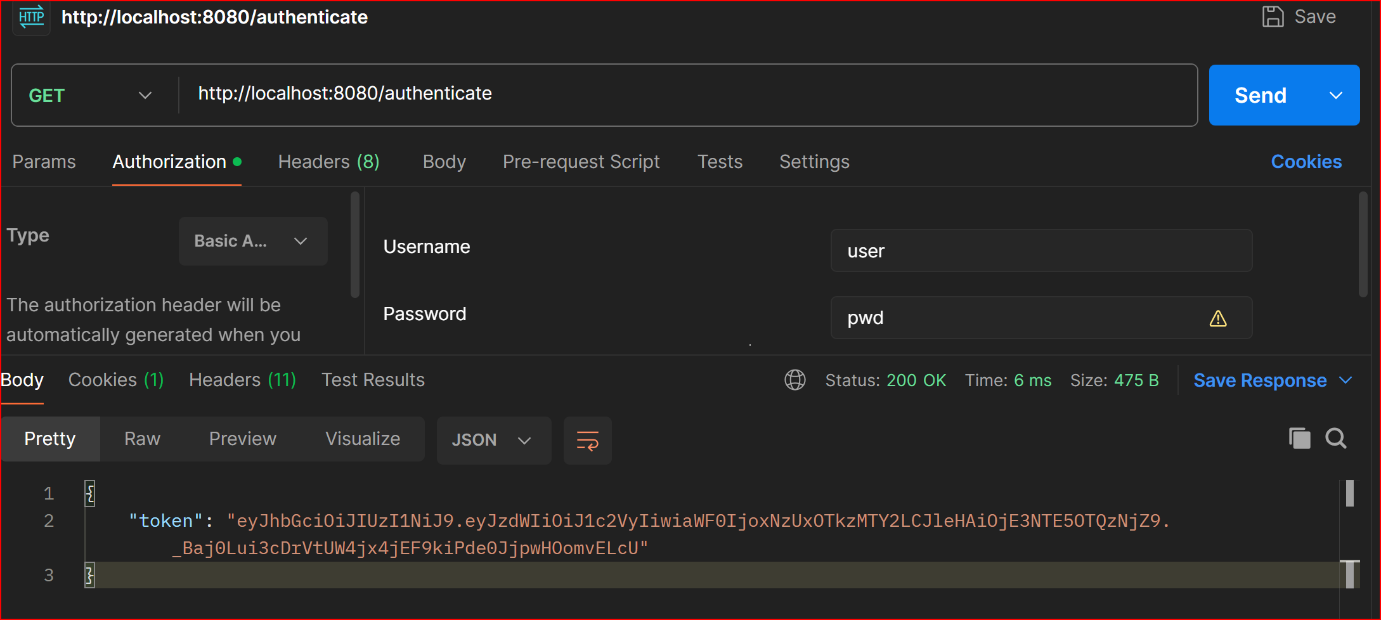
**JwtAuthorizationFilter.java:**

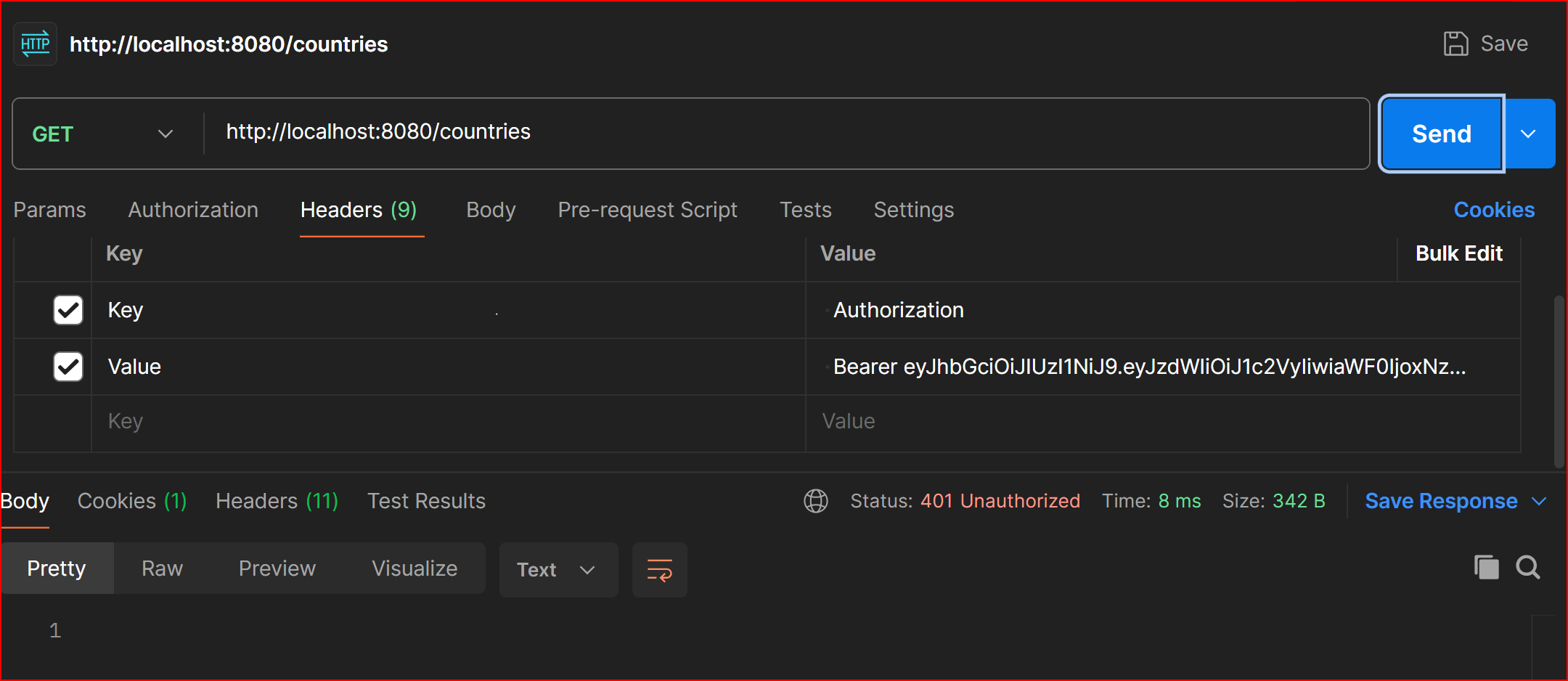
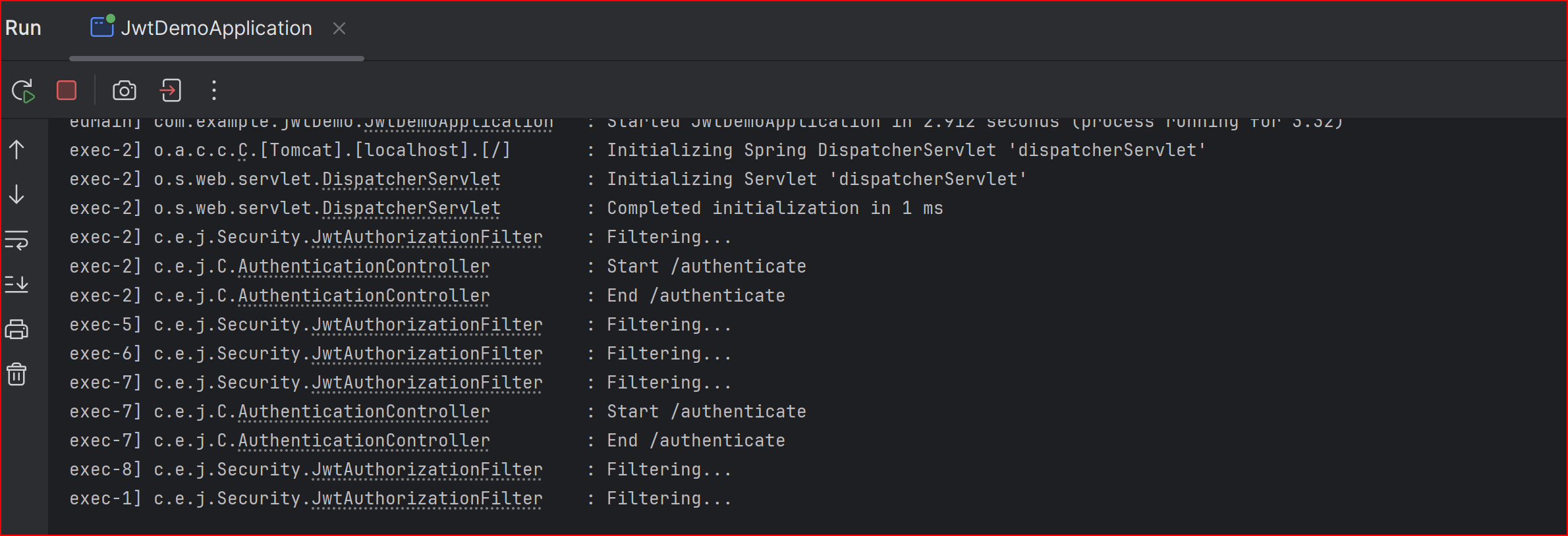
package com.example.jwtDemo.Security;  
  
import io.jsonwebtoken.\*;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import org.springframework.security.authentication.\*;  
import org.springframework.security.core.context.SecurityContextHolder;  
import org.springframework.security.web.authentication.www.BasicAuthenticationFilter;  
  
import jakarta.servlet.\*;  
import jakarta.servlet.http.\*;  
import java.io.IOException;  
import java.util.\*;  
  
public class JwtAuthorizationFilter extends BasicAuthenticationFilter {  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(JwtAuthorizationFilter.class);  
  
 public JwtAuthorizationFilter(AuthenticationManager authManager) {  
 super(authManager);  
 *LOGGER*.info("JwtAuthorizationFilter initialized");  
 }  
  
 @Override  
 protected void doFilterInternal(HttpServletRequest req, HttpServletResponse res,  
 FilterChain chain) throws IOException, ServletException {  
 *LOGGER*.info("Filtering...");  
 String header = req.getHeader("Authorization");  
  
 if (header == null || !header.startsWith("Bearer ")) {  
 chain.doFilter(req, res);  
 return;  
 }  
  
 UsernamePasswordAuthenticationToken auth = getAuthentication(req);  
 SecurityContextHolder.*getContext*().setAuthentication(auth);  
 chain.doFilter(req, res);  
 }  
  
 private UsernamePasswordAuthenticationToken getAuthentication(HttpServletRequest req) {  
 String token = req.getHeader("Authorization");  
 if (token != null) {  
 try {  
 Jws<Claims> parsedToken = Jwts.*parser*()  
 .setSigningKey("secretkey")  
 .parseClaimsJws(token.replace("Bearer ", ""));  
  
 String user = parsedToken.getBody().getSubject();  
 if (user != null) {  
 return new UsernamePasswordAuthenticationToken(user, null, new ArrayList<>());  
 }  
 } catch (JwtException e) {  
 *LOGGER*.error("Invalid JWT", e);  
 }  
 }  
 return null;  
 }  
}

**Pom.xml:**

<?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 https://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>3.5.3</version>  
 <relativePath/> <!-- lookup parent from repository -->  
 </parent>  
 <groupId>com.example</groupId>  
 <artifactId>jwtDemo</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>jwtDemo</name>  
 <description>Demo project for Spring Boot</description>  
 <url/>  
 <licenses>  
 <license/>  
 </licenses>  
 <developers>  
 <developer/>  
 </developers>  
 <scm>  
 <connection/>  
 <developerConnection/>  
 <tag/>  
 <url/>  
 </scm>  
 <properties>  
 <java.version>21</java.version>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-security</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
 <scope>runtime</scope>  
 <optional>true</optional>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.security</groupId>  
 <artifactId>spring-security-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 <dependency>  
 <groupId>javax.xml.bind</groupId>  
 <artifactId>jaxb-api</artifactId>  
 <version>2.3.1</version>  
 </dependency>  
 <dependency>  
 <groupId>io.jsonwebtoken</groupId>  
 <artifactId>jjwt</artifactId>  
 <version>0.9.0</version>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>

**Output:**

****

****