**Spring REST using Spring Boot 3**

**Exercise 1: Create a Spring Web Project using Maven**

**STEP 1: Go to Spring Initializr**

1. Open your browser and go to <https://start.spring.io/>.
2. Fill in the metadata.

**STEP 2: Add Dependencies**

* Spring Web
* Spring Boot DevTools

A screenshot of a computer

AI-generated content may be incorrect.

**STEP 3: Download and Extract the Project**

1. Click on Generate to download the zip file.
2. Unzip the file into a folder named spring-learn.

**STEP 4: Import into Eclipse**

1. Open Eclipse.
2. Go to **File > Import**
3. Choose **Maven > Existing Maven Projects**
4. Click **Browse**, select the spring-learn folder.
5. Click **Finish**

Now, Eclipse will load the project with all dependencies.

**STEP 5: Open & Modify SpringLearnApplication.java**

**SpringLearnApplication.java**

**CODE:**

**package** com.cognizant.springlearn;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

**public** **class** SpringLearnApplication {

**public** **static** **void** main(String[] args) {

System.***out***.println("SpringLearnApplication Started...");

SpringApplication.*run*(SpringLearnApplication.**class**, args);

}

}

**STEP 6:** **Run the Application**

A screenshot of a computer

AI-generated content may be incorrect.

**EXERCISE 2: Spring Core – Load Country from Spring Configuration XML**

**Step 1: Open Existing Project**

We already have the spring-learn project from exercise 1  
Just open it in Eclipse and continue working inside it.

**Step 2: Create the Country Class**

In Eclipse:

* In src/main/java, Right-click on com.cognizant.springlearn → New → **Class**
* **Class name**: Country
* Click **Finish**

**Country.java**

**CODE:**

**package** com.cognizant.springlearn;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**public** **class** Country {

**private** **static** **final** Logger ***LOGGER*** = LoggerFactory.*getLogger*(Country.**class**);

**private** String code;

**private** String name;

**public** Country() {

***LOGGER***.debug("Inside Country Constructor.");

}

**public** String getCode() {

***LOGGER***.debug("Getter for code called");

**return** code;

}

**public** **void** setCode(String code) {

***LOGGER***.debug("Setter for code called with value: {}", code);

**this**.code = code;

}

**public** String getName() {

***LOGGER***.debug("Getter for name called");

**return** name;

}

**public** **void** setName(String name) {

***LOGGER***.debug("Setter for name called with value: {}", name);

**this**.name = name;

}

@Override

**public** String toString() {

**return** "Country{code='" + code + "', name='" + name + "'}";

}

}

**Step 3: Create the Spring XML Configuration**

In Eclipse:

* Right-click src/main/resources → New → **File**
* Name: country.xml
* Click Finish

**Country.xml**

**CODE:**

<?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">

<bean id="country" class="com.cognizant.springlearn.Country">

<property name="code" value="IN" />

<property name="name" value="India" />

</bean>

</beans>

**Step 4: Modify SpringLearnApplication.java**

* Add the displayCountry() method and call it inside main():

**SpringLearnApplication.java**

**CODE:**

**package** com.cognizant.springlearn;

**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** **final** Logger ***LOGGER*** = LoggerFactory.*getLogger*(SpringLearnApplication.**class**);

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(SpringLearnApplication.**class**, args);

*displayCountry*();

}

**public** **static** **void** displayCountry() {

ApplicationContext context = **new** ClassPathXmlApplicationContext("country.xml");

Country country = context.getBean("country", Country.**class**);

***LOGGER***.debug("Country : {}", country.toString());

}

}

**Step 5: Add this in application.properties**

**application.properties**

**CODE:**

spring.application.name=spring-learn

logging.level.root=DEBUG

**Step 6: Run the Application**

A screenshot of a computer

AI-generated content may be incorrect.

**EXERCISE 3: Hello World RESTful Web Service**

**STEP 1: Open Your Existing Project**

You already have the spring-learn project from Exercise 1.

**STEP 2: Create the controller Package and HelloController Class**

In Eclipse:

1. Right-click on src/main/java > com.cognizant.springlearn
2. Click: **New > Package**
3. Name the package: com.cognizant.springlearn.controller
4. Right-click the new package → **New > Class**
5. Name the class: HelloController

**STEP 3: Add the Controller Code**

**HelloController.java**

**CODE:**

**package** com.cognizant.springlearn.controller;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**import** org.springframework.web.bind.annotation.GetMapping;

**import** org.springframework.web.bind.annotation.RestController;

@RestController

**public** **class** HelloController {

**private** **static** **final** Logger ***LOGGER*** = LoggerFactory.*getLogger*(HelloController.**class**);

@GetMapping("/hello")

**public** String sayHello() {

***LOGGER***.info("Start - sayHello()");

String message = "Hello World!!";

***LOGGER***.info("End - sayHello()");

**return** message;

}

}

**STEP 4: Configure Port in application.properties**

In Eclipse:

1. Go to src/main/resources
2. Open application.properties
3. Add this line:

server.port=8083

**Step 5: Modify SpringLearnApplication.java**

**SpringLearnApplication.java**

**CODE:**

**package** com.cognizant.springlearn;

**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);

}

}

**STEP 6: Run the Application**

A screenshot of a computer

AI-generated content may be incorrect.

**STEP 7: Test the Endpoint in Chrome**

In Browser:

* Open: <http://localhost:8083/hello>

**OUTPUT:**

A screenshot of a computer

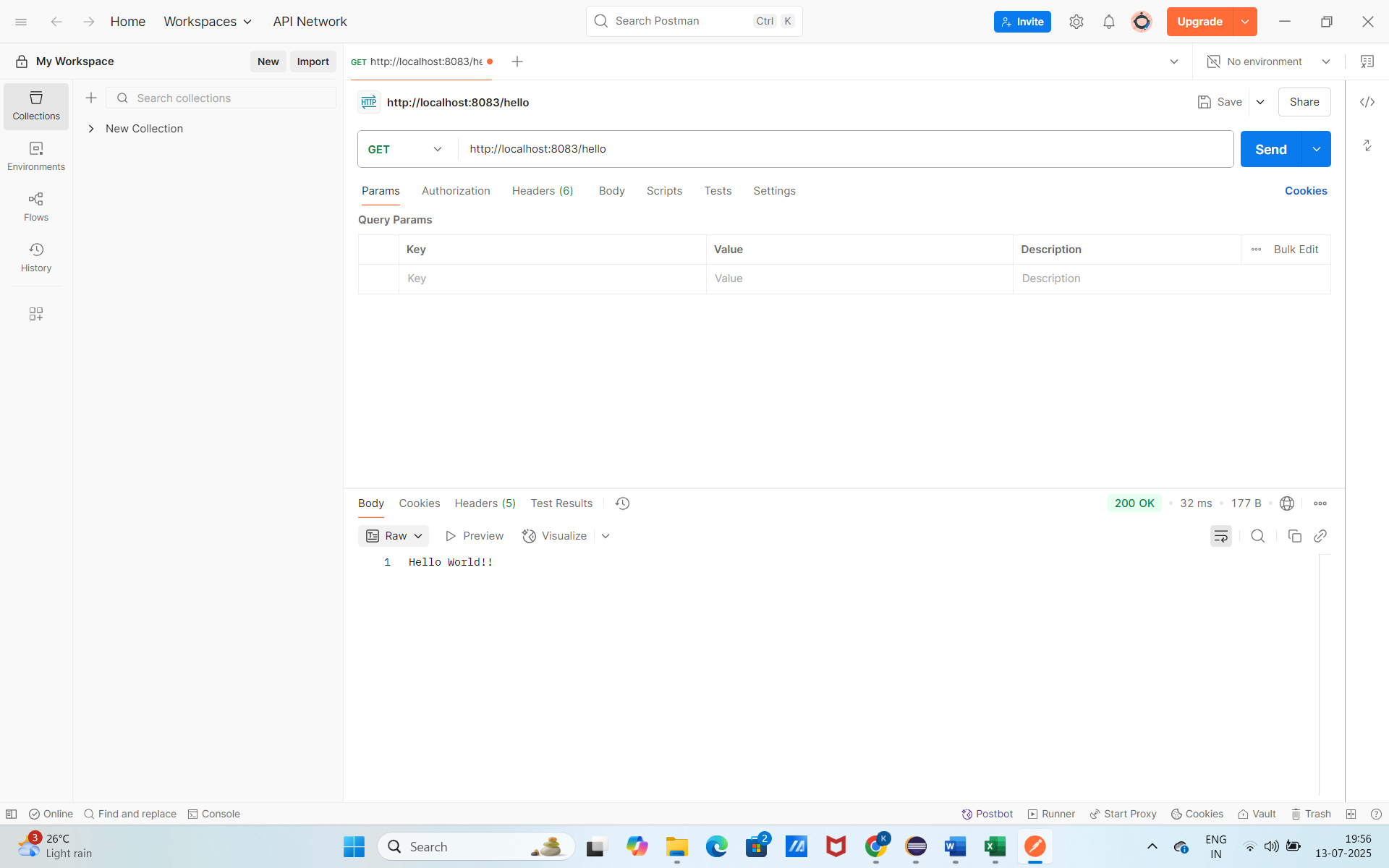
AI-generated content may be incorrect.

**STEP 8: Test in Postman**

In Postman:

* Method: GET
* URL: http://localhost:8083/hello
* Click **Send**

**OUTPUT:**



**STEP 9: View HTTP Headers**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Exercise 4: REST - Country Web Service**

**Step 1: Make sure country.xml is present**

**Step 2: Confirm Country.java is created**

**Step 3: Create a new Controller**

* Create CountryController.java in com.cognizant.springlearn.controller

**CountryController.java**

**CODE:**

**package** com.cognizant.springlearn.controller;

**import** com.cognizant.springlearn.Country;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**import** org.springframework.context.ApplicationContext;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.bind.annotation.RestController;

@RestController

**public** **class** CountryController {

**private** **static** **final** Logger ***LOGGER*** = LoggerFactory.*getLogger*(CountryController.**class**);

@RequestMapping("/country")

**public** Country getCountryIndia() {

***LOGGER***.info("Start - getCountryIndia()");

ApplicationContext context = **new** ClassPathXmlApplicationContext("country.xml");

Country country = context.getBean("country", Country.**class**);

***LOGGER***.info("End - getCountryIndia()");

**return** country;

}

}

**Step 4: Make sure your application runs on port 8083**

**Step 5: Run the app**

A screenshot of a computer

AI-generated content may be incorrect.

**Step 6: Test the API**

* In Browser:

Open: <http://localhost:8083/country>

**OUTPUT:**

A screenshot of a computer

AI-generated content may be incorrect.

* In Postman:

Method: GET

URL: <http://localhost:8083/country>

**OUTPUT:**

A screenshot of a computer

AI-generated content may be incorrect.

**Exercise 5: JWT Authentication Service**

**Objective:**

Create an authentication service that:

* Takes username & password
* Verifies credentials
* Returns a **JWT token**

**Step 1: Add Dependencies in pom.xml**

* Open your pom.xml and add these:

<!-- JWT Library -->

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt</artifactId>

<version>0.9.1</version>

</dependency>

<!-- Spring Security -->

<dependency>

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

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

</dependency>

**Step 2: Set Port Number**

* Open src/main/resources/application.properties and set: server.port=8090

**Step 3: Create JwtUtil Class**

* Create package com.cognizant.springlearn.util and class JwtUtil in it.

**JwtUtil.java**

**CODE:**

**package** com.cognizant.springlearn.util;

**import** io.jsonwebtoken.Jwts;

**import** io.jsonwebtoken.SignatureAlgorithm;

**import** org.springframework.stereotype.Component;

**import** java.util.Date;

@Component

**public** **class** JwtUtil {

**private** String secretKey = "secret123";

**public** String generateToken(String username) {

**return** Jwts.*builder*()

.setSubject(username)

.setIssuedAt(**new** Date())

.setExpiration(**new** Date(System.*currentTimeMillis*() + 1000 \* 60 \* 60)) // 1 hour validity

.signWith(SignatureAlgorithm.***HS256***, secretKey)

.compact();

}

}

**Step 4: Create Authentication Controller**

* Create AuthController.java in package com.cognizant.springlearn.controller.

**AuthController.java**

**CODE:**

**package** com.cognizant.springlearn.controller;

**import** com.cognizant.springlearn.util.JwtUtil;

**import** jakarta.servlet.http.HttpServletRequest;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.http.ResponseEntity;

**import** org.springframework.web.bind.annotation.\*;

**import** java.nio.charset.StandardCharsets;

**import** java.util.Base64;

@RestController

**public** **class** AuthController {

**private** **static** **final** Logger ***LOGGER*** = LoggerFactory.*getLogger*(AuthController.**class**);

@Autowired

**private** JwtUtil jwtUtil;

@GetMapping("/authenticate")

**public** ResponseEntity<?> authenticate(HttpServletRequest request) {

***LOGGER***.info("Start authenticate()");

String authHeader = request.getHeader("Authorization");

**if** (authHeader != **null** && authHeader.startsWith("Basic ")) {

// Decode Base64 encoded username:password

String base64Credentials = authHeader.substring("Basic ".length());

**byte**[] credDecoded = Base64.*getDecoder*().decode(base64Credentials);

String credentials = **new** String(credDecoded, StandardCharsets.***UTF\_8***);

// Split username and password

**final** String[] values = credentials.split(":", 2);

String username = values[0];

String password = values[1];

**if** ("user".equals(username) && "pwd".equals(password)) {

String token = jwtUtil.generateToken(username);

***LOGGER***.info("Authentication successful");

**return** ResponseEntity.*ok*().body("{\"token\": \"" + token + "\"}");

}

}

***LOGGER***.warn("Authentication failed");

**return** ResponseEntity.*status*(401).body("Unauthorized");

}

}

**Step 5: Configure Spring Security**

* Create a new package com.cognizant.springlearn.config and class SecurityConfig.java in it.

**SecurityConfig.java**

**CODE:**

**package** com.cognizant.springlearn.config;

**import** org.springframework.context.annotation.Bean;

**import** org.springframework.context.annotation.Configuration;

**import** org.springframework.security.config.annotation.web.builders.HttpSecurity;

**import** org.springframework.security.web.SecurityFilterChain;

@Configuration

**public** **class** SecurityConfig {

@Bean

**public** SecurityFilterChain filterChain(HttpSecurity http) **throws** Exception {

http

.csrf(csrf -> csrf.disable())

.authorizeHttpRequests(auth -> auth

.requestMatchers("/authenticate").permitAll()

.anyRequest().authenticated()

);

**return** http.build();

}

}

**Step 6: Run and test the application**

* Running the application

A screenshot of a computer

AI-generated content may be incorrect.

* Test in postman

1. Open Postman.
2. Select method **GET**.
3. Enter: http://localhost:8090/authenticate
4. Go to **Authorization tab**:

* Type: **Basic Auth**
* Username: user
* Password: pwd

1. Click **Send**

A screenshot of a computer

AI-generated content may be incorrect.