WEEK 04

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

**Project Structure**

spring-rest-demo/

├── src/

│ └── main/

│ ├── java/

│ │ └── com/example/springrest/

│ │ ├── controller/

│ │ │ └── StudentController.java

│ │ ├── model/

│ │ │ └── Student.java

│ │ ├── service/

│ │ │ └── StudentService.java

│ │ └── SpringRestDemoApplication.java

│ └── resources/

│ └── application.properties

├── pom.xml

**pom.xml**

<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.example</groupId>

<artifactId>spring-rest-demo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>Spring REST Demo</name>

<description>Spring REST API using Spring Boot 3</description>

<packaging>jar</packaging>

<properties>

<java.version>17</java.version>

<spring.boot.version>3.1.4</spring.boot.version>

</properties>

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-databind</artifactId>

</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>

<version>${spring.boot.version}</version>

</plugin>

</plugins>

</build>

</project>

**Main Application**

**SpringRestDemoApplication.java**

package com.example.springrest;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

public class SpringRestDemoApplication {

public static void main(String[] args) {

SpringApplication.run(SpringRestDemoApplication.class, args);

}

}

**Model**

**Student.java**

package com.example.springrest.model;

public class Student {

private int id;

private String name;

private String department;

public Student() {

}

public Student(int id, String name, String department) {

this.id = id;

this.name = name;

this.department = department;

}

public int getId() { return id; }

public void setId(int id) { this.id = id; }

public String getName() { return name; }

public void setName(String name) { this.name = name; }

public String getDepartment() { return department; }

public void setDepartment(String department) { this.department = department; }

}

**Service Layer**

**StudentService.java**

package com.example.springrest.service;

import com.example.springrest.model.Student;

import org.springframework.stereotype.Service;

import java.util.\*;

public class StudentService {

private final Map<Integer, Student> studentRepo = new HashMap<>();

public StudentService() {

studentRepo.put(1, new Student(1, "Alice", "CSE"));

studentRepo.put(2, new Student(2, "Bob", "ECE"));

}

public List<Student> getAllStudents() {

return new ArrayList<>(studentRepo.values());

}

public Student getStudentById(int id) {

return studentRepo.get(id);

}

public Student addStudent(Student student) {

studentRepo.put(student.getId(), student);

return student;

}

public Student updateStudent(int id, Student student) {

studentRepo.put(id, student);

return student;

}

public void deleteStudent(int id) {

studentRepo.remove(id);

}

}

**Controller Layer**

**StudentController.java**

package com.example.springrest.controller;

import com.example.springrest.model.Student;

import com.example.springrest.service.StudentService;

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

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

import java.util.List;

public class StudentController {

private StudentService service;

public List<Student> getAllStudents() {

return service.getAllStudents();

}

public Student getStudentById(@PathVariable int id) {

return service.getStudentById(id);

}

public Student addStudent(@RequestBody Student student) {

return service.addStudent(student);

}

public Student updateStudent(@PathVariable int id, @RequestBody Student student) {

return service.updateStudent(id, student);

}

public void deleteStudent(@PathVariable int id) {

service.deleteStudent(id);

}

}

**Application.properties**

Leave it empty or add a custom port:

server.port=8080

Top of Form

Bottom of Form

**2. Spring Core - Load Country from Spring Configuration XML**

**Spring Core: Load Country from XML**

**Country.java (POJO)**

public class Country {

private String countryName;

private String countryCode;

public String getCountryName() {

return countryName;

}

public void setCountryName(String countryName) {

this.countryName = countryName;

}

public String getCountryCode() {

return countryCode;

}

public void setCountryCode(String countryCode) {

this.countryCode = countryCode;

}

public String toString() {

return "Country [name=" + countryName + ", code=" + countryCode + "]";

}

}

**applicationContext.xml (Spring Configuration)**

<?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.example.demo.model.Country">

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

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

</bean>

</beans>

**MainApp.java (to load from XML)**

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.example.demo.model.Country;

public class MainApp {

public static void main(String[] args) {

ApplicationContextcontext=newClassPathXmlApplicationContext("applicationContext.xml");

Country country = (Country) context.getBean("country");

System.out.println(country);

}

}

**Spring REST Hands-on Example**

**CountryController.java**

public class CountryController {

private Country country; // This will be loaded from XML using a Bean configuration

public Country getCountry() {

return country;

}

}

**WebConfig.java (If using Java config for REST)**

public class WebConfig {

}

**Spring Boot 3 Version**

**DemoApplication.java**

public class DemoApplication {

public static void main(String[] args) {

SpringApplication.run(DemoApplication.class, args);

}

}

**Autowiring Country in Spring Boot**

public class CountryController {

private Country country;

public Country getCountry() {

return country;

}

}

**Project Structure**

├── main

│ ├── java

│ │ └── com.example.demo

│ │ ├── DemoApplication.java

│ │ ├── model

│ │ │ └── Country.java

│ │ └── controller

│ │ └── CountryController.java

│ └── resources

│ └── applicationContext.xml

**3.Hello World Restful Web Service**

**Spring REST Hands-On (Spring MVC with XML config)**

**Project Structure**

spring-rest-handson/

├── src/

│ └── main/

│ ├── java/

│ │ └── com.example/

│ │ └── HelloController.java

│ └── webapp/

│ ├── WEB-INF/

│ │ ├── web.xml

│ │ └── spring-servlet.xml

**HelloController.java**

package com.example;

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

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

public class HelloController {

public String sayHello() {

return "Hello World!";

}

}

**web.xml (Deployment Descriptor)**

<web-app xmlns="http://java.sun.com/xml/ns/javaee" version="3.0">

<servlet>

<servlet-name>spring</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>spring</servlet-name>

<url-pattern>/</url-pattern>

</servlet-mapping>

</web-app>

**spring-servlet.xml**

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:context="http://www.springframework.org/schema/context"

xmlns:mvc="http://www.springframework.org/schema/mvc"

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

http://www.springframework.org/schema/context

http://www.springframework.org/schema/context/spring-context.xsd

http://www.springframework.org/schema/mvc

http://www.springframework.org/schema/mvc/spring-mvc.xsd">

<context:component-scan base-package="com.example" />

<mvc:annotation-driven />

</beans>

**Spring Boot 3 – Hello World REST API (Recommended)**

**Project Structure**

spring-boot-hello/

├── src/

│ ├── main/

│ │ ├── java/

│ │ │ └── com.example.hello/

│ │ │ ├── HelloApplication.java

│ │ │ └── HelloController.java

│ └── resources/

│ └── application.properties

**HelloApplication.java**

package com.example.hello;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

public class HelloApplication {

public static void main(String[] args) {

SpringApplication.run(HelloApplication.class, args);

}

}

**HelloController.java**

package com.example.hello;

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

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

public class HelloController {

public String sayHello() {

return "Hello World from Spring Boot 3!";

}

}

**4.REST - Country Web Service**

**Project Structure:**

country-rest-service/

├── src/

│ └── main/

│ └── java/

│ └── com/example/country/

│ ├── Country.java

│ ├── CountryService.java

│ ├── CountryController.java

│ └── CountryRestServiceApplication.java

└── pom.xml

**Dependencies (pom.xml)**

<dependencies>

<dependency>

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

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

</dependency>

</dependencies>

**Country.java – Model Class**

package com.example.country;

public class Country {

private String code;

private String name;

public Country() {}

public Country(String code, String name) {

this.code = code;

this.name = name;

}

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; }

}

**CountryService.java – Service Class**

package com.example.country;

import org.springframework.stereotype.Service;

import java.util.\*;

public class CountryService {

private static final Map<String, Country> countryMap = new HashMap<>();

static {

countryMap.put("IN", new Country("IN", "India"));

countryMap.put("US", new Country("US", "United States"));

countryMap.put("FR", new Country("FR", "France"));

}

public Country getCountryByCode(String code) {

return countryMap.get(code.toUpperCase());

}

public List<Country> getAllCountries() {

return new ArrayList<>(countryMap.values());

}

}

**CountryController.java – REST Controller**

package com.example.country;

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

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

import java.util.List;

public class CountryController {

private CountryService countryService;

public List<Country> getAllCountries() {

return countryService.getAllCountries();

}

public Country getCountryByCode(@PathVariable String code) {

return countryService.getCountryByCode(code);

}

}

**CountryRestServiceApplication.java – Main Class**

package com.example.country;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

public class CountryRestServiceApplication {

public static void main(String[] args) {

SpringApplication.run(CountryRestServiceApplication.class, args);

}

}

**Sample API Calls (using Postman / curl / browser):**

* **List all countries:**

GET http://localhost:8080/countries

* **Get country by code:**

GET http://localhost:8080/countries/IN

**5. REST - Get country based on country code**

**Maven Project Setup (pom.xml)**

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

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>country-rest</artifactId>

<version>1.0.0</version>

<parent>

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

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

<version>3.1.0</version> <!-- or your version -->

</parent>

<dependencies>

<dependency>

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

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

</dependency>

</dependencies>

</project>

**Model – Country.java**

package com.example.countryrest.model;

public class Country {

private String code;

private String name;

public Country() {}

public Country(String code, String name) {

this.code = code;

this.name = name;

}

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; }

}

**Service – CountryService.java**

package com.example.countryrest.service;

import com.example.countryrest.model.Country;

import org.springframework.stereotype.Service;

import java.util.HashMap;

import java.util.Map;

public class CountryService {

private static final Map<String, Country> countries = new HashMap<>();

static {

countries.put("IN", new Country("IN", "India"));

countries.put("US", new Country("US", "United States"));

countries.put("UK", new Country("UK", "United Kingdom"));

}

public Country getCountryByCode(String code) {

return countries.get(code.toUpperCase());

}

}

**Controller – CountryController.java**

package com.example.countryrest.controller;

import com.example.countryrest.model.Country;

import com.example.countryrest.service.CountryService;

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

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

public class CountryController {

private CountryService countryService;

public Country getCountryByCode(@PathVariable String code) {

Country country = countryService.getCountryByCode(code);

if (country == null) {

throw new RuntimeException("Country code not found: " + code);

}

return country;

}

}

**Main Class – CountryRestServiceApplication.java**

package com.example.countryrest;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

public class CountryRestServiceApplication {

public static void main(String[] args) {

SpringApplication.run(CountryRestServiceApplication.class, args);

}

}

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

**Add Maven Dependencies (pom.xml)**

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-api</artifactId>

<version>0.11.5</version>

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-impl</artifactId>

<version>0.11.5</version>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-jackson</artifactId>

<version>0.11.5</version>

<scope>runtime</scope>

</dependency>

</dependencies>

**Create JwtUtil.java**

package com.example.jwt;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import io.jsonwebtoken.security.Keys;

import org.springframework.stereotype.Component;

import java.security.Key;

import java.util.Date;

public class JwtUtil {

private final Key key = Keys.secretKeyFor(SignatureAlgorithm.HS256); // generate secure random key

public String generateToken(String username) {

return Jwts.builder()

.setSubject(username)

.setIssuer("MyApp")

.setIssuedAt(new Date())

.setExpiration(new Date(System.currentTimeMillis() + 86400000)) // 1 day

.signWith(key)

.compact();

}

public Key getKey() {

return key;

}

}

**Create AuthRequest.java and AuthResponse.java**

package com.example.jwt;

public class AuthRequest {

private String username;

private String password;

}

package com.example.jwt;

public class AuthResponse {

private String jwt;

public AuthResponse(String jwt) {

this.jwt = jwt;

}

}

**Create AuthController.java**

package com.example.jwt;

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

public class AuthController {

private final JwtUtil jwtUtil;

public AuthController(JwtUtil jwtUtil) {

this.jwtUtil = jwtUtil;

}

public AuthResponse authenticate(@RequestBody AuthRequest authRequest) { if("user".equals(authRequest.getUsername())&&"password".equals(authRequest.getPassword())) {

String token = jwtUtil.generateToken(authRequest.getUsername());

return new AuthResponse(token);

} else {

throw new RuntimeException("Invalid credentials");

}

}

}

**Configure Spring Security to Allow Authentication Endpoint**

package com.example.jwt;

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;

public class SecurityConfig {

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http.csrf().disable()

.authorizeHttpRequests(authz -> authz

.requestMatchers("/auth/\*\*").permitAll()

.anyRequest().authenticated()

);

return http.build();

}

}

**Run and Test**

Use **Postman** or **cURL** to test the endpoint:

curl -X POST http://localhost:8080/auth/authenticate \

-H "Content-Type: application/json" \

-d '{"username":"user","password":"password"}'