**Week 4**

**Spring REST using Spring Boot 3**

**Create a Spring Web Project using Maven**

**demoApplication.java**

package com.example.demo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class DemoApplication {

public static void main(String[] args) {

SpringApplication.run(DemoApplication.class, args);

System.out.println("Application started on port 9090");

}

}

**HelloController.java**

package com.example.demo;

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

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

@RestController

public class HelloController {

@GetMapping("/")

public String home() {

System.out.println("Home endpoint called");

return "Welcome to Spring Boot!";

}

@GetMapping("/hello")

public String sayHello() {

System.out.println("Hello endpoint called");

return "Hello, World!";

}

}

A screenshot of a computer

AI-generated content may be incorrect.

**Spring Core – Load Country from Spring Configuration XML**

**Country.java**

package com.example;

public class Country {

private String name;

private String code;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

@Override

public String toString() {

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

}

}

**Main.java**

package com.example;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Main {

public static void main(String[] args) {

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

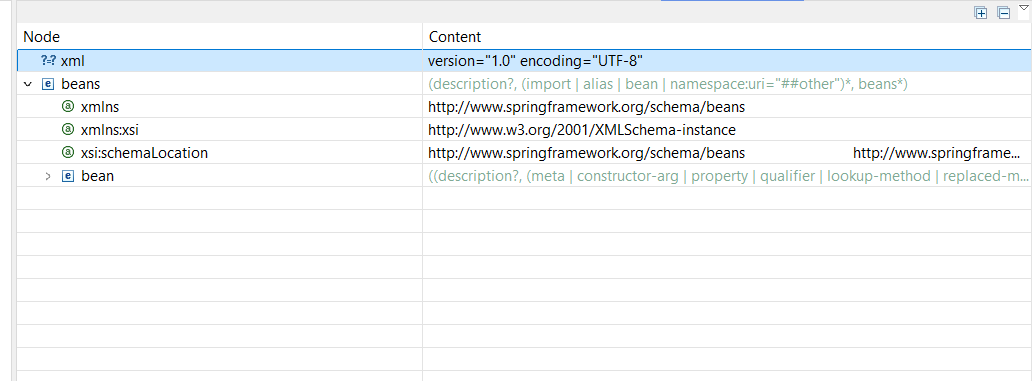
Country country = (Country) context.getBean("country");

System.out.println(country);

}

}

**Country-config-xml**

****

**Hello World RESTful Web Service**

**HelloWorldApplication.java**

package com.example.helloworld;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

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

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

@SpringBootApplication

public class HelloWorldApplication {

public static void main(String[] args) {

SpringApplication.run(HelloWorldApplication.class, args);

}

}

@RestController

class HelloController {

@GetMapping("/hello")

public String hello() {

return "Hello, World!";

}

}

**Application.propeties**

# Application configuration

spring.application.name=hello-world-rest

server.port=8089

**A screenshot of a computer

AI-generated content may be incorrect.**

**REST - Country Web Service**

**Countrycontroller.java**

package com.example.countryservice;

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

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

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

import java.util.HashMap;

import java.util.Map;

@RestController

public class CountryController {

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

static {

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

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

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

countries.put("JP", "Japan");

}

@GetMapping("/country/{code}")

public String getCountry(@PathVariable String code) {

return countries.getOrDefault(code.toUpperCase(), "Country Not Found");

}

@GetMapping("/countries")

public Map<String, String> getAllCountries() {

return countries;

}

}

**CountryServiceApplication.java**

package com.example.countryservice;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class CountryServiceApplication {

public static void main(String[] args) {

SpringApplication.run(CountryServiceApplication.class, args);

}

}

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

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

**CountryCodeApplication.java**

package com.example.countrycode;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class CountryCodeApplication {

public static void main(String[] args) {

SpringApplication.run(CountryCodeApplication.class, args);

}

}

**CountryCodeController.java**

package com.example.countrycode;

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

import java.util.HashMap;

import java.util.Map;

@RestController

@RequestMapping("/api")

public class CountryCodeController {

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

static {

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

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

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

countryMap.put("JP", new Country("Japan", "JP", "Tokyo"));

}

@GetMapping("/country/{code}")

public Country getCountryByCode(@PathVariable String code) {

return countryMap.getOrDefault(code.toUpperCase(), new Country("Not Found", code.toUpperCase(), "-"));

}

static class Country {

private String name;

private String code;

private String capital;

public Country(String name, String code, String capital) {

this.name = name;

this.code = code;

this.capital = capital;

}

public String getName() {

return name;

}

public String getCode() {

return code;

}

public String getCapital() {

return capital;

}

}

}

**Application.properties**

spring.application.name=country-code-lookup

server.port=8085

|  |  |
| --- | --- |
| **/api/country/IN** | **Returns details of India** |

|  |  |
| --- | --- |
| **/api/country/US** | **Returns United States** |

|  |  |
| --- | --- |
| **/api/country/XYZ** | **Returns "Not Found" fallback** |
|  |  |

**Create authentication service that returns JWT**

**Index.js**

const express = require('express');

const jwt = require('jsonwebtoken');

const bodyParser = require('body-parser');

const app = express();

const PORT = 3000;

const SECRET\_KEY = 'your\_secret\_key';

app.use(bodyParser.json());

app.post('/login', (req, res) => {

const { username, password } = req.body;

// Simple check, replace with DB logic in production

if (username === 'admin' && password === 'password') {

const token = jwt.sign({ username }, SECRET\_KEY, { expiresIn: '1h' });

return res.json({ token });

}

res.status(401).json({ message: 'Invalid credentials' });

});

app.listen(PORT, () => {

console.log(`Auth service running at http://localhost:${PORT}`);

});

**Package.json**

{

"name": "jwt-auth-service",

"version": "1.0.0",

"main": "index.js",

"scripts": {

"start": "node index.js"

},

"dependencies": {

"express": "^4.18.2",

"jsonwebtoken": "^9.0.0",

"body-parser": "^1.20.2"

}

}