**Exercise 5: REST- Get Country based on Country code**

1. **Country.java**

package com.example.demo.model;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

import jakarta.validation.constraints.NotBlank;

import jakarta.validation.constraints.Size;

@Entity

public class Country {

@Id

@Size(min = 2, max = 3, message = "Code must be 2-3 letters")

private String code;

@NotBlank(message = "Country name is required")

private String name;

@NotBlank(message = "Capital is required")

private String capital;

// Getters & Setters

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; }

public String getCapital() { return capital; }

public void setCapital(String capital) { this.capital = capital; }

}

1. **CountryRepository.java**

package com.example.demo.repository;

import com.example.demo.model.Country;

import org.springframework.data.jpa.repository.JpaRepository;

public interface CountryRepository extends JpaRepository<Country, String> {

}

1. **CountryService.java**

package com.example.demo.service;

import com.example.demo.model.Country;

import com.example.demo.repository.CountryRepository;

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

import org.springframework.stereotype.Service;

@Service

public class CountryService {

@Autowired

private CountryRepository repository;

public Country getCountryByCode(String code) {

return repository.findById(code.toUpperCase())

.orElseThrow(() -> new RuntimeException("Country not found: " + code));

}

public Country addCountry(Country country) {

return repository.save(country);

}

public Country updateCountry(String code, Country newData) {

Country existing = getCountryByCode(code);

existing.setName(newData.getName());

existing.setCapital(newData.getCapital());

return repository.save(existing);

}

public void deleteCountry(String code) {

repository.deleteById(code.toUpperCase());

}

}

1. **CountryController.java**

package com.example.demo.controller;

import com.example.demo.model.Country;

import com.example.demo.service.CountryService;

import jakarta.validation.Valid;

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

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

@RestController

@RequestMapping("/api/countries")

public class CountryController {

@Autowired

private CountryService service;

// GET /api/countries/IN

@GetMapping("/{code}")

public Country getCountry(@PathVariable String code) {

return service.getCountryByCode(code);

}

// POST /api/countries

@PostMapping

public Country addCountry(@Valid @RequestBody Country country) {

return service.addCountry(country);

}

// PUT /api/countries/IN

@PutMapping("/{code}")

public Country updateCountry(@PathVariable String code,

@Valid @RequestBody Country country) {

return service.updateCountry(code, country);

}

// DELETE /api/countries/IN

@DeleteMapping("/{code}")

public String deleteCountry(@PathVariable String code) {

service.deleteCountry(code);

return "Deleted country with code: " + code;

}

}

1. **GlobalExceptionHandler.java**

package com.example.demo.exception;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.MethodArgumentNotValidException;

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

@RestControllerAdvice

public class GlobalExceptionHandler {

@ExceptionHandler(RuntimeException.class)

public ResponseEntity<String> handleRuntimeException(RuntimeException ex) {

return new ResponseEntity<>(ex.getMessage(), HttpStatus.NOT\_FOUND);

}

@ExceptionHandler(MethodArgumentNotValidException.class)

public ResponseEntity<String> handleValidationErrors(MethodArgumentNotValidException ex) {

String error = ex.getBindingResult().getFieldError().getDefaultMessage();

return new ResponseEntity<>(error, HttpStatus.BAD\_REQUEST);

}

}

1. **Application.properties**

# H2 In-Memory DB settings

spring.h2.console.enabled=true

spring.datasource.url=jdbc:h2:mem:testdb

spring.datasource.driver-class-name=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

1. **DemoApplication.java**

package com.example.demo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class DemoApplication {

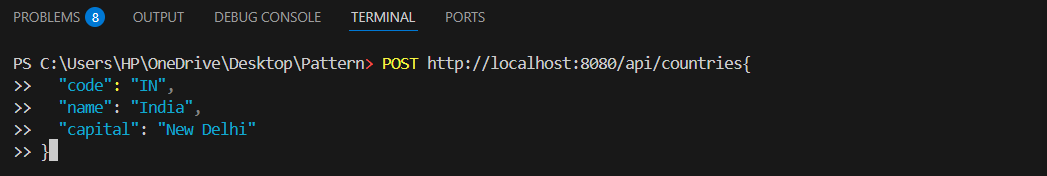
public static void main(String[] args) {

SpringApplication.run(DemoApplication.class, args);

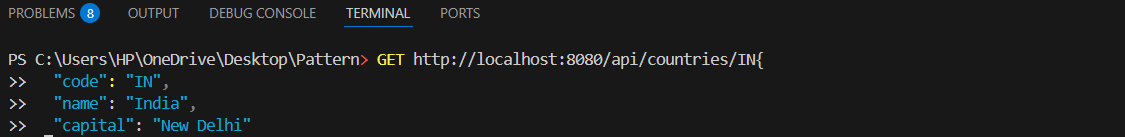
}

}

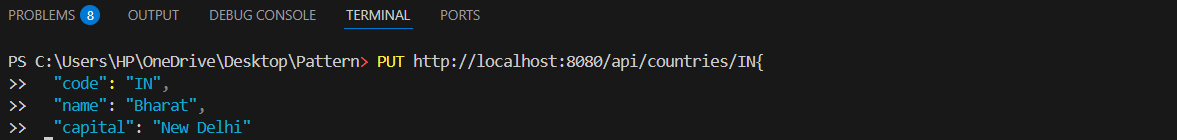
**Valid POST Country :**



**Valid GET Country :**



**Valid PUT Country :**



**Valid DELETE Country :**

