**2. spring-data-jpa-handson**

**Demonstrate implementation of Query Methods feature of Spring Data JPA**

**CODE:**

controller 🡪 CountryController.java

package org.example.springdatajpahandson.controller;  
  
import org.example.springdatajpahandson.model.Country;  
import org.example.springdatajpahandson.service.CountryService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.PageRequest;  
import org.springframework.data.domain.Sort;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
  
@RestController  
@RequestMapping("/countries")  
public class CountryController {  
  
 @Autowired  
 private CountryService countryService;  
  
 @PostMapping  
 public Country addCountry(@RequestBody Country country) {  
 return countryService.saveCountry(country);  
 }  
  
 @GetMapping  
 public List<Country> getAllCountries() {  
 return countryService.getAllCountries();  
 }  
  
 @GetMapping("/name/{name}")  
 public ResponseEntity<Country> getCountryByName(@PathVariable String name) {  
 return countryService.getCountryByName(name)  
 .map(ResponseEntity::*ok*)  
 .orElse(ResponseEntity.*notFound*().build());  
 }  
  
 @GetMapping("/search/name/{keyword}")  
 public List<Country> getCountriesByNameKeyword(@PathVariable String keyword) {  
 return countryService.getCountriesByNameContaining(keyword);  
 }  
  
 @GetMapping("/search/code-prefix/{prefix}")  
 public List<Country> getCountriesByCodePrefix(@PathVariable String prefix) {  
 return countryService.getCountriesByCodePrefix(prefix);  
 }  
  
 @GetMapping("/code/{code}")  
 public ResponseEntity<Country> getCountryByCode(@PathVariable String code) {  
 return countryService.getCountryByCode(code)  
 .map(ResponseEntity::*ok*)  
 .orElse(ResponseEntity.*notFound*().build());  
 }  
  
 // @Query usage: find by name and code  
 @GetMapping("/name-code")  
 public ResponseEntity<Country> getByNameAndCode(@RequestParam String name,  
 @RequestParam String code) {  
 return countryService.getCountryByNameAndCode(name, code)  
 .map(ResponseEntity::*ok*)  
 .orElse(ResponseEntity.*notFound*().build());  
 }  
  
 // Pagination and Sorting  
 @GetMapping("/paged")  
 public Page<Country> getPagedCountries(@RequestParam(defaultValue = "0") int page,  
 @RequestParam(defaultValue = "3") int size,  
 @RequestParam(defaultValue = "name") String sortBy) {  
 return countryService.getCountriesPage(PageRequest.*of*(page, size, Sort.*by*(sortBy)));  
 }  
}

model🡪Country.java

package org.example.springdatajpahandson.model;  
  
import jakarta.persistence.\*;  
  
@Entity  
public class Country {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
 private String code;  
  
 public Country() {  
 }  
  
 public Country(String name, String code) {  
 this.name = name;  
 this.code = code;  
 }  
  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 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;  
 }  
}

repository🡪CountryRepository.java

package org.example.springdatajpahandson.repository;  
  
import org.example.springdatajpahandson.model.Country;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.data.jpa.repository.Query;  
import org.springframework.data.repository.query.Param;  
  
import java.util.List;  
import java.util.Optional;  
  
public interface CountryRepository extends JpaRepository<Country, Long> {  
 Optional<Country> findByName(String name);  
 List<Country> findByNameContaining(String keyword);  
 List<Country> findByCodeStartingWith(String prefix);  
 Optional<Country> findByCode(String code);  
 @Query("SELECT c FROM Country c WHERE c.name = :name AND c.code = :code")  
 Optional<Country> findByNameAndCode(@Param("name") String name, @Param("code") String code);  
}

service🡪CountryService

package org.example.springdatajpahandson.service;  
  
import org.example.springdatajpahandson.model.Country;  
import org.example.springdatajpahandson.repository.CountryRepository;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.Pageable;  
import org.springframework.stereotype.Service;  
  
import java.util.List;  
import java.util.Optional;  
  
@Service  
public class CountryService {  
  
 @Autowired  
 private CountryRepository countryRepository;  
  
 public Country saveCountry(Country country) {  
 return countryRepository.save(country);  
 }  
  
 public List<Country> getAllCountries() {  
 return countryRepository.findAll();  
 }  
  
 public Optional<Country> getCountryByName(String name) {  
 return countryRepository.findByName(name);  
 }  
  
 public List<Country> getCountriesByNameContaining(String keyword) {  
 return countryRepository.findByNameContaining(keyword);  
 }  
  
 public List<Country> getCountriesByCodePrefix(String prefix) {  
 return countryRepository.findByCodeStartingWith(prefix);  
 }  
  
 public Optional<Country> getCountryByCode(String code) {  
 return countryRepository.findByCode(code);  
 }  
  
 public Optional<Country> getCountryByNameAndCode(String name, String code) {  
 return countryRepository.findByNameAndCode(name, code);  
 }  
  
 public Page<Country> getCountriesPage(Pageable pageable) {  
 return countryRepository.findAll(pageable);  
 }  
}

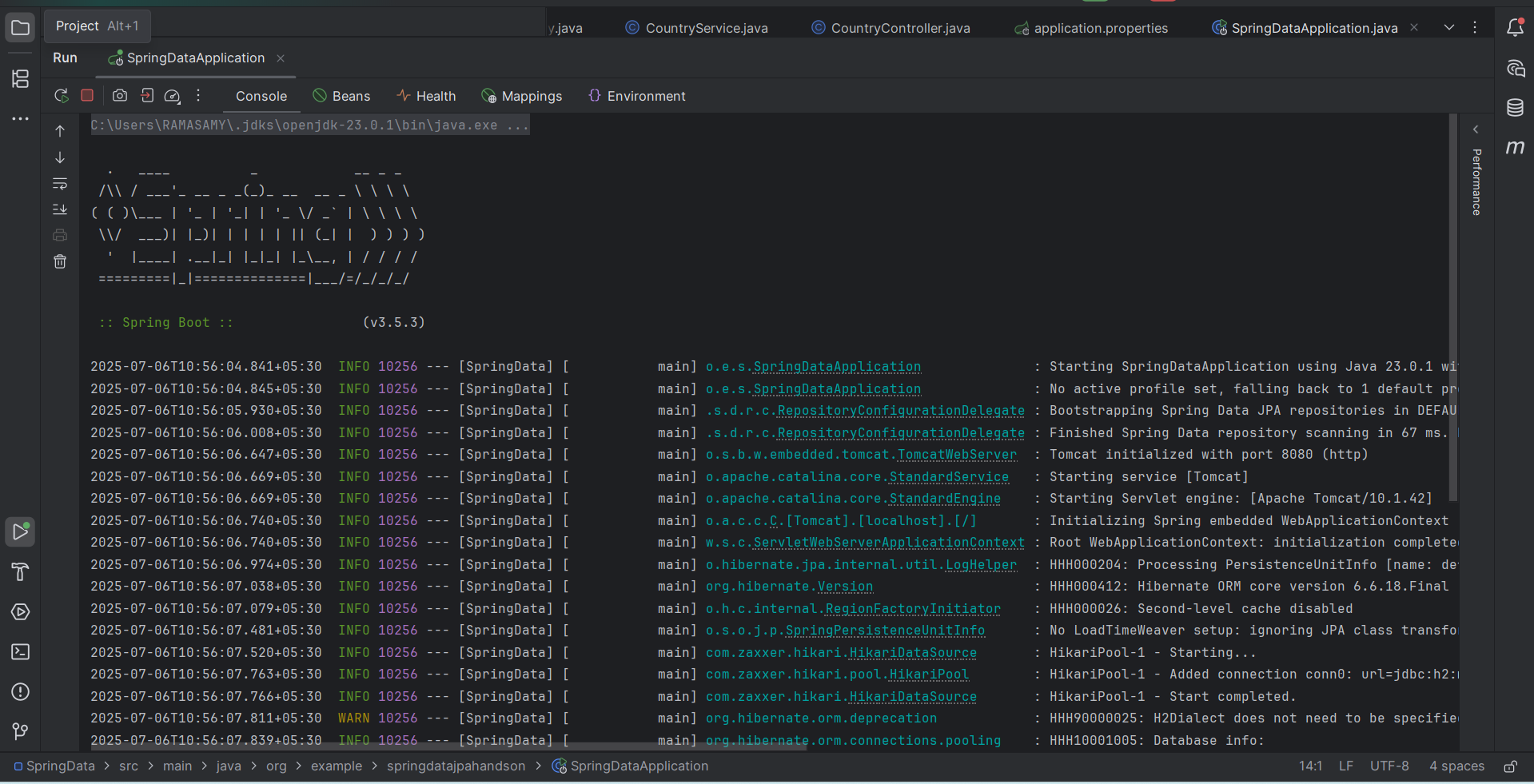
applicaton.properties

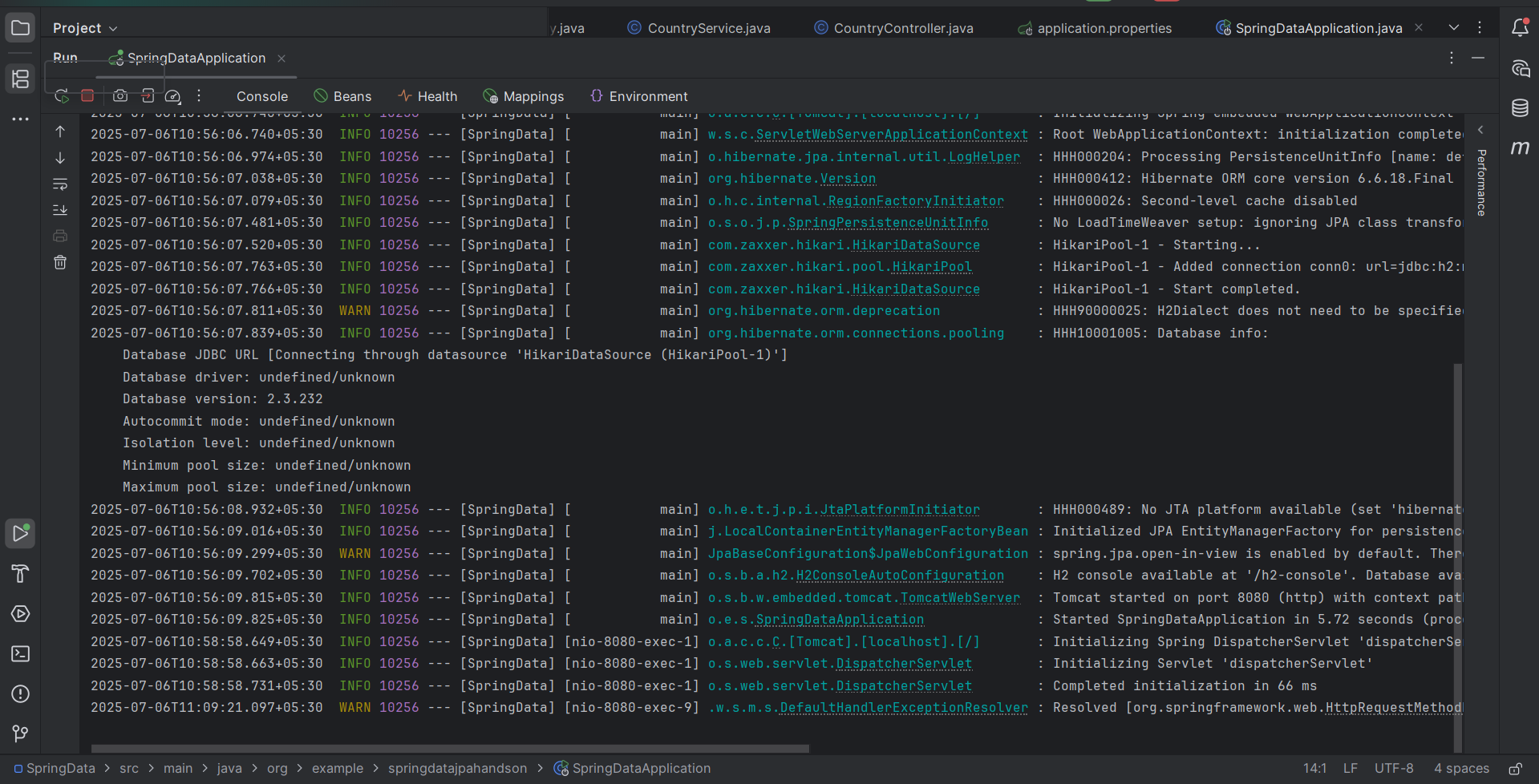
spring.application.name=SpringData  
spring.datasource.url=jdbc:h2:mem:testdb  
spring.datasource.driverClassName=org.h2.Driver  
spring.datasource.username=sa  
spring.datasource.password=  
spring.jpa.hibernate.ddl-auto=update  
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect  
spring.h2.console.enabled=true

SpringDataApplication.java

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

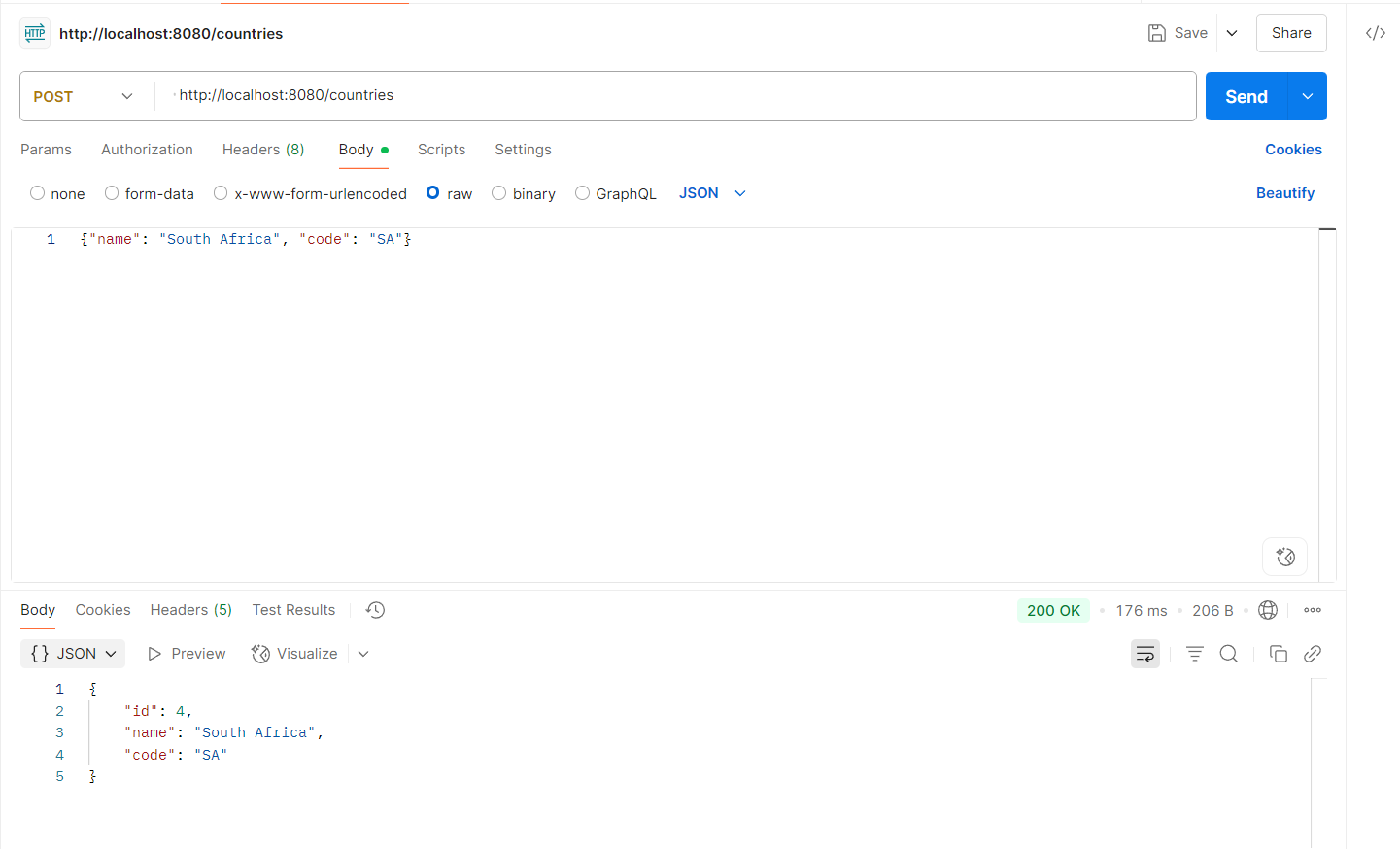
**OUTPUT**

****

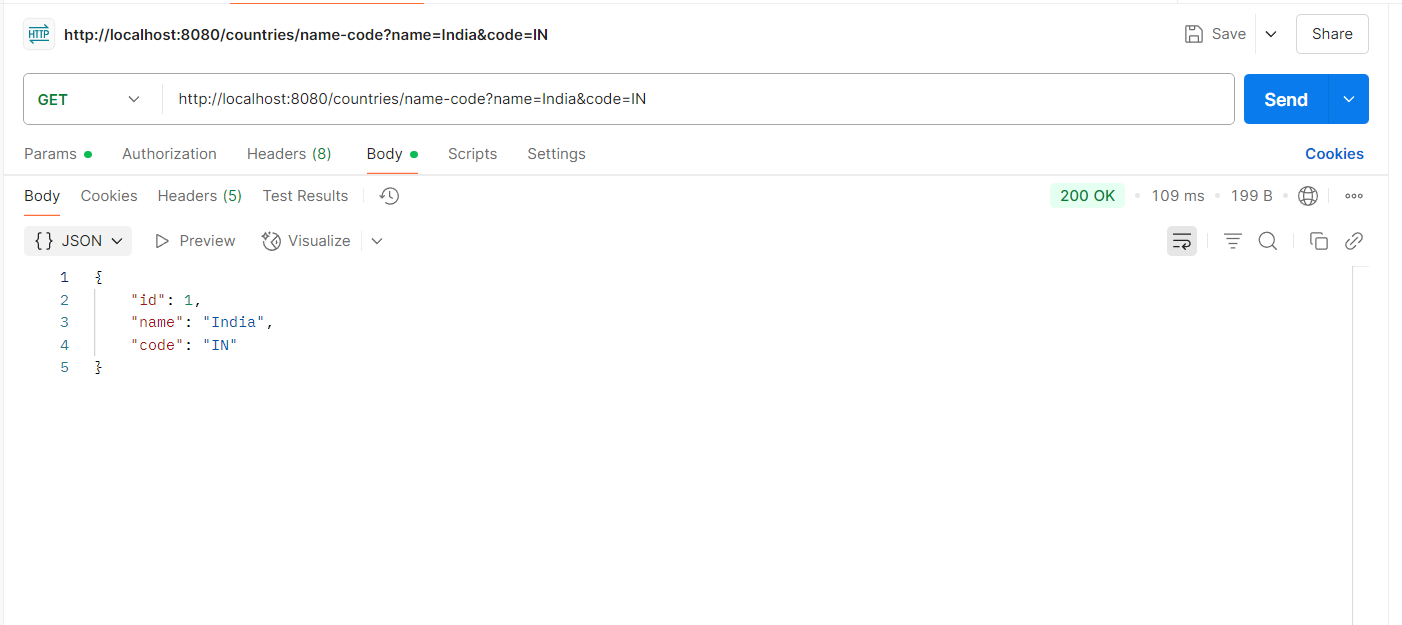
****

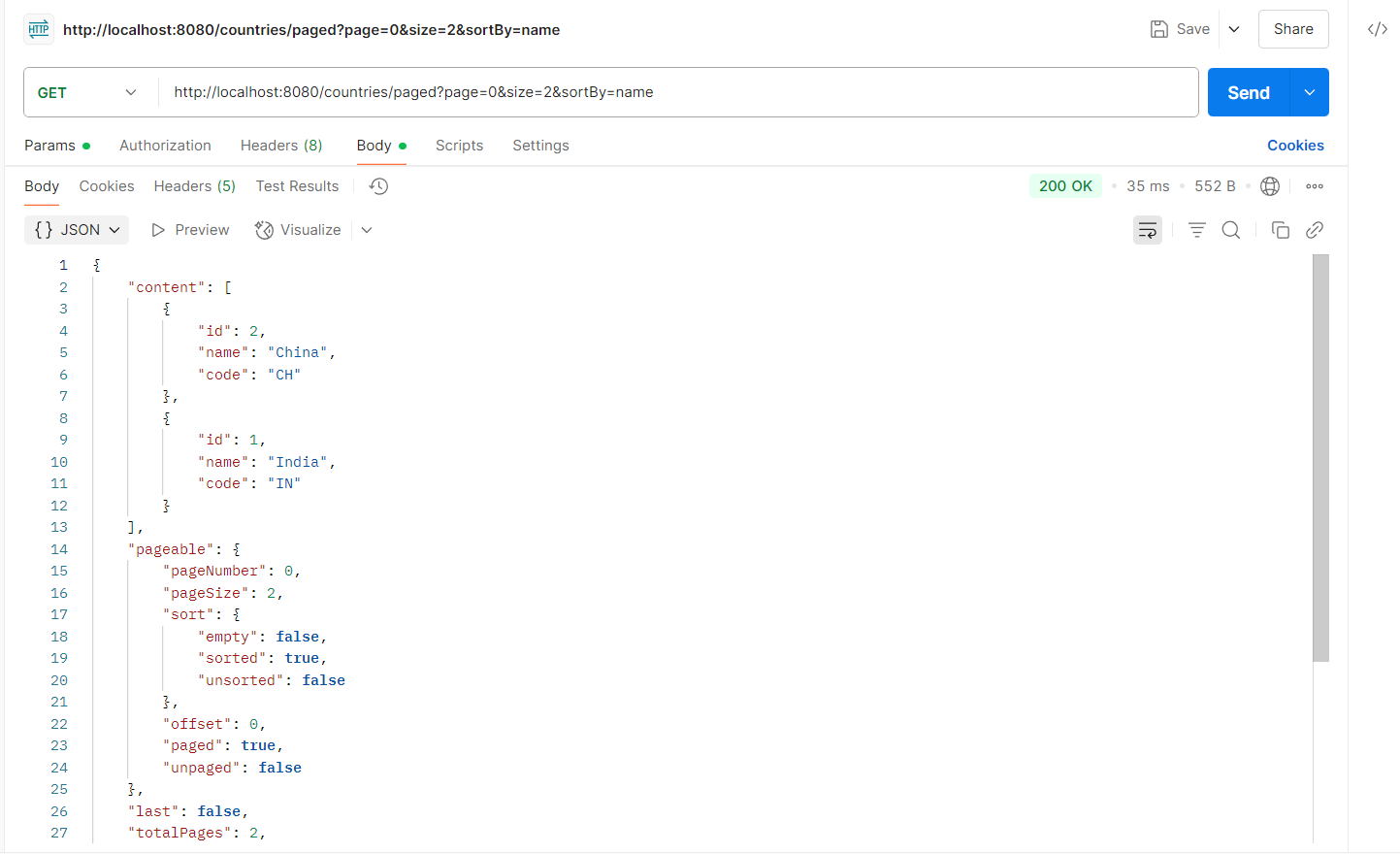
**POSTMAN:**

**POST**

****

**GET**

****

****