**PROJECT QUESTIONS**

**1) LIBRARY MANAGEMENT MICROSERVICE:**

**UserRepository.java**

package com.wecp.library.repository;

import com.wecp.library.domain.User;

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

public interface UserRepository extends JpaRepository<User,Long>{

}

**BookRepository.java**

package com.wecp.library.repository;

import com.wecp.library.domain.Book;

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

public interface UserRepository extends JpaRepository<Book,Long>{

}

**LibraryController.java**

package com.wecp.library.controller;

import com.wecp.library.domain.Book;

import com.wecp.library.domain.User;

import com.wecp.library.domain.repository.BookRpository;

import com.wecp.library.domain.repository.UserRpository;

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

import org.springframework.beans.http.ResponseEntity;

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

import java.util.optional;

@RestController

@RequestMapping("/api/v1")

public class LibraryController{

@Autowired

private UserRepository userRepository;

@Autowired

private BookRepository bookRepository;

@GetMapping("/user/{id}")

public ResponseEntity<User> getUser (@PathVariable Long id){

Optional<User> user = userRepository.findById(id);

return user.map(ResponseEntty::ok).orElseGet(()->ResponseEntity.nocontent().build());

}

@PostMapping("/user")

public ResponseEntity<User> createUser(@RequestBody User user){

User savedUser = userRpository.save(user);

return ResponseEntity.ok(savedUser);

}

@GetMapping("/book/{id}")

public ResponseEntity<User> getBook (@PathVariable Long id){

Optional<Book> book = bookRepository.findById(id);

return book.map(ResponseEntty::ok).orElseGet(()->ResponseEntity.nocontent().build());

}

@PostMapping("/user")

public ResponseEntity<Book> createBook(@RequestBody Book book){

Book savedBook = bookRpository.save(book);

return ResponseEntity.ok(savedBook);

}

}

**2) IMPLEMENT CRUD USING COLLECTIONS IN SPRING BOOT FOR REAL ESTATE PROPERTY MANAGEMENT SYSTEM**

**PropertyController.java**

package com.property.controller;

import com.property.entity.Property;

import com.property.sevice.PropertyService;

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

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

import java.util.List;

@RestController

@RequestMapping("/properties")

public class PropertyController{

private final PropertyService propertySrevice;

@Autowired

public PropertyController(PropertyService propertyService ){

this.propertyService = propertyService;

}

@PostMapping

public Property addBooking(@RequestBody Property property){

return propertyService.addProperty(property);

}

@GetMapping

public List<Property> getAllBooings(){

return propertyService.getAllProperties();

}

}

**PropertyService.java**

package com.property.service;

import com.property.entity.Property;

import org.springframework.stereotype.Service;

import java.util.ArrayList;

import java.util.List;

@Service

public class PropertyService{

private static int counter=1;

private final List<Property> propertyList = new ArrayList<>();

public Property addProperty(Property property){

property.setId(counter++);

propertyList.add(property);

return property;

}

public List<Property> getAllProperties(){

return propertyList;

}

}

**Property.java**

package com.property.entity;

public class Property{

private int id;

private String address;

private String description;

}

public classs Property(){

}

public class Property(int id, String address, String description){

this.id=id;

this.address=address;

this.description;

}

public int getId(){

return id;

}

public void setId(int id){

this=this.id;

}

public String getAddress(){

return Address;

}

public void setAddress(String address){

this=this.address;

}

public int getDescription(){

return description;

}

public void setDescription(String description){

this=this.description;

}

**3) STUDENT MICROSERVICE**

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

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

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

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

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

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

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

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

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

import com.library.MysqlLibrary.model.Book;

import com.library.MysqlLibrary.model.User;

import com.library.MysqlLibrary.repository.BookRepo;

import com.library.MysqlLibrary.repository.UserRepo;

import com.library.MysqlLibrary.service.BookService;

import com.library.MysqlLibrary.service.UserService;

import java.util.List;

@RestController

@RequestMapping("/api/v1")

public class UserBookController {

@Autowired

UserService userService;

@Autowired

BookService bookService;

@Autowired

BookRepo bookRepo;

@Autowired

UserRepo userRepo;

@PostMapping("/user")

public ResponseEntity<User> addUser(@RequestBody User user) {

return new ResponseEntity<User>(userRepo.save(user), HttpStatus.CREATED);

}

@GetMapping("/user")

public ResponseEntity<List<User>> getAllUser() {

// return new ResponseEntity<List<User>>(userService.getAll(), HttpStatus.OK);

return ResponseEntity.ok().body(userRepo.findAll());

}

@GetMapping("/book/{id}")

public ResponseEntity<Book> getBookById(@PathVariable Long id) {

Book obj = bookService.getById(id);

if (obj == null)

return ResponseEntity.noContent().build();

return new ResponseEntity<Book>(obj, HttpStatus.OK);

}

@PutMapping("/book/{id}")

public ResponseEntity<Book> addBook(@PathVariable Long id,@RequestBody Book book) {

return new ResponseEntity<Book>(bookRepo.save(book), HttpStatus.CREATED);

}

@PutMapping("/book/{id}")

public ResponseEntity<Book> addBookById(@PathVariable Long id,@RequestBody Book book) {

Book obj = bookRepo.findById(id).get();

if (obj == null)

return ResponseEntity.noContent().build();

obj.setName(book.getName());

bookRepo.save(obj);

return new ResponseEntity<Book>(obj, HttpStatus.CREATED);

}

@DeleteMapping("/book")

public ResponseEntity<Book> Book(@RequestBody Book book) {

bookRepo.delete(book);

return new ResponseEntity<Book>(null);

}

}

**Another method:**

@Autowired

private UserRepository userRepository;

@Autowired

private BookRepository bookRepository;

@GetMapping("/user")

public ResponseEntity<List<User>> getAllUser() {

return ResponseEntity.ok().body(userRepository.findAll());

}

@GetMapping("/book/{id}")

public ResponseEntity<Book> getBook(@PathVariable Long id) {

Optional<Book> book = bookRepository.findById(id);

return book.map(ResponseEntity::ok).orElseGet(()->ResponseEntity.noContent().build());

}

@PostMapping("/user")

public ResponseEntity<User> addUser(@RequestBody User user) {

User savedUser = userRepository.save(user);

return ResponseEntity.ok(savedUser);

@PutMapping("/user")

public ResponseEntity<User> addUser(@RequestBody User user) {

User savedUser = userRepository.save(user);

return ResponseEntity.ok(savedUser);

@PutMapping("/book/{id}")

public ResponseEntity<Book> addBook(@PathVariable Long id) {

Optional<Book> book = bookRepository.findById(id);

return book.map(ResponseEntity::ok).orElseGet(()->ResponseEntity.noContent().build());

}

@DeleteMapping("/book")

public ResponseEntity<Book> Book(@RequestBody Book book) {

Book deleteBook = bookRepository.delete(book);

return ResponseEntity.ok(deleteBook)

}