# BOOK-SERVICE

*package* com.tutorial.bookservice.controller;

*import* com.tutorial.bookservice.entity.Book;

*import* com.tutorial.bookservice.service.BookService;

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

*import* org.springframework.http.ResponseEntity;

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

*import* java.util.List;

@RestController

@RequestMapping("/book")

public class BookController {

    @Autowired

    BookService bookService;

    @GetMapping

    public ResponseEntity<List<Book>> getAll() {

        List<Book> books *=* bookService.getAll();

*if*(books.isEmpty())

*return* ResponseEntity.noContent().build();

*return* ResponseEntity.ok(books);

    }

    @GetMapping("/{id}")

    public ResponseEntity<Book> getById(@PathVariable("id") int id) {

        Book book *=* bookService.getBookById(id);

*if*(book *==* null)

*return* ResponseEntity.notFound().build();

*return* ResponseEntity.ok(book);

    }

    @PostMapping()

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

        Book bookNew *=* bookService.save(book);

*return* ResponseEntity.ok(bookNew);

    }

    @GetMapping("/bystudent/{studentId}")

    public ResponseEntity<List<Book>> getByStudentId(@PathVariable("studentId") int studentId) {

        List<Book> books *=* bookService.byStudentId(studentId);

*return* ResponseEntity.ok(books);

    }

}

package com.tutorial.bookservice.entity;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

@Data

@NoArgsConstructor

@AllArgsConstructor

public class Book {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String title;

private String author;

private int studentId;

}

*package* com.tutorial.bookservice.repository;

*import* com.tutorial.bookservice.entity.Book;

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

*import* org.springframework.stereotype.Repository;

*import* java.util.List;

@Repository

public interface BookRepository extends JpaRepository<Book, Integer> {

    List<Book> findByStudentId(int studentId);

}

*package* com.tutorial.bookservice;

*import* org.springframework.boot.SpringApplication;

*import* org.springframework.boot.autoconfigure.SpringBootApplication;

*import* org.springframework.cloud.netflix.eureka.EnableEurekaClient;

@SpringBootApplication

@EnableEurekaClient

public class BookServiceApplication {

    public static void main(String[] args) {

        SpringApplication.run(BookServiceApplication.class, args);

    }

}

# PET-SERVICE

*package* com.tutorial.petservice.controller;

*import* com.tutorial.petservice.entity.Pet;

*import* com.tutorial.petservice.service.PetService;

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

*import* org.springframework.http.ResponseEntity;

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

*import* java.util.List;

@RestController

@RequestMapping("/pet")

public class PetController {

    @Autowired

    PetService petService;

    @GetMapping

    public ResponseEntity<List<Pet>> getAll() {

        List<Pet> pets *=* petService.getAll();

*if*(pets.isEmpty())

*return* ResponseEntity.noContent().build();

*return* ResponseEntity.ok(pets);

    }

    @GetMapping("/{id}")

    public ResponseEntity<Pet> getById(@PathVariable("id") int id) {

        Pet pet *=* petService.getPetById(id);

*if*(pet *==* null)

*return* ResponseEntity.notFound().build();

*return* ResponseEntity.ok(pet);

    }

    @PostMapping()

    public ResponseEntity<Pet> save(@RequestBody Pet pet) {

        Pet petNew *=* petService.save(pet);

*return* ResponseEntity.ok(petNew);

    }

    @GetMapping("/bystudent/{studentId}")

    public ResponseEntity<List<Pet>> getByStudentId(@PathVariable("studentId") int studentId) {

        List<Pet> pets *=* petService.byStudentId(studentId);

*return* ResponseEntity.ok(pets);

    }

}

*package* com.tutorial.petservice.entity;

*import* lombok.AllArgsConstructor;

*import* lombok.Data;

*import* lombok.NoArgsConstructor;

*import* javax.persistence.Entity;

*import* javax.persistence.GeneratedValue;

*import* javax.persistence.GenerationType;

*import* javax.persistence.Id;

@Entity

@Data

@NoArgsConstructor

@AllArgsConstructor

public class Pet {

    @Id

    @GeneratedValue(strategy *=* GenerationType.IDENTITY)

    private int id;

    private String type;

    private String name;

    private int studentId;

}

*package* com.tutorial.petservice.repository;

*import* com.tutorial.petservice.entity.Pet;

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

*import* org.springframework.stereotype.Repository;

*import* java.util.List;

@Repository

public interface PetRepository extends JpaRepository<Pet, Integer> {

    List<Pet> findByStudentId(int studentId);

}

*package* com.tutorial.petservice;

*import* org.springframework.boot.SpringApplication;

*import* org.springframework.boot.autoconfigure.SpringBootApplication;

*import* org.springframework.cloud.netflix.eureka.EnableEurekaClient;

@SpringBootApplication

@EnableEurekaClient

public class PetServiceApplication {

    public static void main(String[] args) {

        SpringApplication.run(PetServiceApplication.class, args);

    }

}

# STUDENT-SERVICE

*package* com.tutorial.studentservice.controller;

*import* com.tutorial.studentservice.entity.Student;

*import* com.tutorial.studentservice.model.Pet;

*import* com.tutorial.studentservice.model.Book;

*import* com.tutorial.studentservice.service.StudentService;

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

*import* org.springframework.http.ResponseEntity;

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

*import* java.util.List;

*import* java.util.Map;

@RestController

@RequestMapping("/student")

public class StudentController {

    @Autowired

    StudentService studentService;

    @GetMapping

    public ResponseEntity<List<Student>> getAll() {

        List<Student> students *=* studentService.getAll();

*if*(students.isEmpty())

*return* ResponseEntity.noContent().build();

*return* ResponseEntity.ok(students);

    }

    @GetMapping("/{id}")

    public ResponseEntity<Student> getById(@PathVariable("id") int id) {

        Student student *=* studentService.getStudentById(id);

*if*(student *==* null)

*return* ResponseEntity.notFound().build();

*return* ResponseEntity.ok(student);

    }

    @PostMapping()

    public ResponseEntity<Student> save(@RequestBody Student student) {

        Student studentNew *=* studentService.save(student);

*return* ResponseEntity.ok(studentNew);

    }

    @GetMapping("/books/{studentId}")

    public ResponseEntity<List<Book>> getBooks(@PathVariable("studentId") int studentId) {

        Student student *=* studentService.getStudentById(studentId);

*if*(student *==* null)

*return* ResponseEntity.notFound().build();

        List<Book> books *=* studentService.getBooks(studentId);

*return* ResponseEntity.ok(books);

    }

    @GetMapping("/pets/{studentId}")

    public ResponseEntity<List<Pet>> getPets(@PathVariable("studentId") int studentId) {

        Student student *=* studentService.getStudentById(studentId);

*if*(student *==* null)

*return* ResponseEntity.notFound().build();

        List<Pet> pets *=* studentService.getPets(studentId);

*return* ResponseEntity.ok(pets);

    }

    @PostMapping("/savebook/{studentId}")

    public ResponseEntity<Book> saveBook(@PathVariable("studentId") int studentId, @RequestBody Book book) {

*if*(studentService.getStudentById(studentId) *==* null)

*return* ResponseEntity.notFound().build();

        Book bookNew *=* studentService.saveBook(studentId, book);

*return* ResponseEntity.ok(book);

    }

    @PostMapping("/savepet/{studentId}")

    public ResponseEntity<Pet> savePet(@PathVariable("studentId") int studentId, @RequestBody Pet pet) {

*if*(studentService.getStudentById(studentId) *==* null)

*return* ResponseEntity.notFound().build();

        Pet petNew *=* studentService.savePet(studentId, pet);

*return* ResponseEntity.ok(pet);

    }

}

*package* com.tutorial.studentservice.entity;

*import* lombok.AllArgsConstructor;

*import* lombok.Data;

*import* lombok.NoArgsConstructor;

*import* javax.persistence.Entity;

*import* javax.persistence.GeneratedValue;

*import* javax.persistence.GenerationType;

*import* javax.persistence.Id;

@Entity

@Data

@NoArgsConstructor

@AllArgsConstructor

public class Student {

    @Id

    @GeneratedValue(strategy *=* GenerationType.IDENTITY)

    private int id;

    private String name;

    private String email;

}

*package* com.tutorial.studentservice.model;

*import* lombok.AllArgsConstructor;

*import* lombok.Data;

*import* lombok.NoArgsConstructor;

@Data

@NoArgsConstructor

@AllArgsConstructor

public class Book {

    private String title;

    private String author;

    private int studentId;

}

*package* com.tutorial.studentservice.model;

*import* lombok.AllArgsConstructor;

*import* lombok.Data;

*import* lombok.NoArgsConstructor;

@Data

@NoArgsConstructor

@AllArgsConstructor

public class Pet {

    private String type;

    private String name;

    private int studentId;

}

*package* com.tutorial.studentservice.repository;

*import* com.tutorial.studentservice.entity.Student;

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

*import* org.springframework.stereotype.Repository;

@Repository

public interface StudentRepository extends JpaRepository<Student, Integer> {

}

*package* com.tutorial.studentservice;

*import* org.springframework.boot.SpringApplication;

*import* org.springframework.boot.autoconfigure.SpringBootApplication;

*import* org.springframework.cloud.netflix.eureka.EnableEurekaClient;

@SpringBootApplication

@EnableEurekaClient

public class StudentServiceApplication {

    public static void main(String[] args) {

        SpringApplication.run(StudentServiceApplication.class, args);

    }

}