FPT Education

FPT POLYTECHNIC

cdio™

Conceive Design Implement Operate

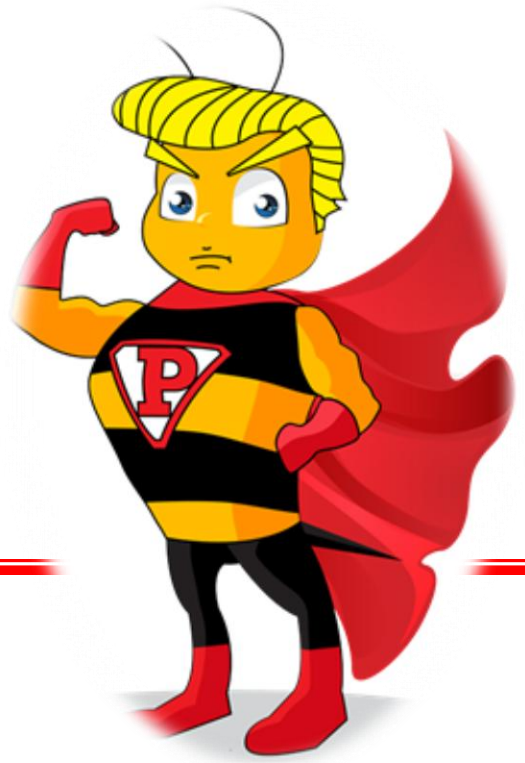
RESTful Web API 2

Giảng viên: Nguyễn Nghiệm

www.poly.edu.vn

NỘI DUNG

- Rest Consumer using HttpURLConnection
- RestConsumer using RestTemplate
- Spring Boot REST API
- Spring Boot REST API with JpaRepository



HttpURLConnection

HttpURLConnection là gì?

- HttpURLConnection cung cấp các phương thức tương tác với REST API
- Tạo HttpURLConnection
  - var url = "https://fpolyedu.firebaseio.com/students.json";
  - var connection = (HttpURLConnection) new URL(url).openConnection();
- Các phương thức cần quan tâm của HttpURLConnection
  - setRequestMethod(): REST Operation (GET, POST, PUT, DELETE)
  - setOutput(): Yêu cầu gửi dữ liệu
  - getResponseCode(): Lấy mã trạng thái phản hồi từ REST API
  - getInputStream(): Lấy InputStream để đọc dữ liệu gửi về từ REST API
  - getOutputStream(): Lấy OutputStream để gửi dữ liệu lên REST API
- Phương pháp cho phép làm việc với REST API trong các ứng dụng Java (Desktop, Mobile, Web...) mà ko cần sử dụng API của bên thứ ba.

ĐỌC CHUỖI TỪ INPUTSTREAM

```
String readData(InputStream is) throws IOException {
        ByteArrayOutputStream out = new ByteArrayOutputStream();
        byte[] block = new byte[4 * 1024];
        while (true) {
                int n = is.read(block);
                if (n <= 0) {
                        break;
                }
                out.write(block, 0, n);
        }
        return out.toString();
}
```

- Phương thức này sẽ được sử dụng để đọc dữ liệu trả về từ REST API

GET ALL

```
var url = "https://fpolyedu.firebaseio.com/students.json";
var connection = (HttpURLConnection) new URL(url).openConnection();
connection.setRequestProperty("Content-Type", "application/json; charset=utf-8");
connection.setRequestMethod("GET");
if (connection.getResponseCode() == 200) {
        String data = readData(connection.getInputStream());
        System.out.println(data);
}
connection.disconnect();
```

GET BY KEY

```
var url = "https://fpolyedu.firebaseio.com/students/-ORe_zNPXY0yhvSM9_uh.json";
var connection = (HttpURLConnection) new URL(url).openConnection();
connection.setRequestProperty("Content-Type", "application/json; charset=utf-8");
connection.setRequestMethod("GET");
if (connection.getResponseCode() == 200) {
        String data = readData(connection.getInputStream());
        System.out.println(data);
}
connection.disconnect();
```

POST

```
var url = "https://fpolyedu.firebaseio.com/students.json";
var connection = (HttpURLConnection) new URL(url).openConnection();
connection.setRequestProperty("Content-Type", "application/json; charset=utf-8");
connection.setRequestMethod("POST");
var json = """
           {"id": "SV09","name": "Sinh viên 09","mark": 5.5,"gender": true}
           """;
connection.setDoOutput(true);
connection.getOutputStream().write(json.getBytes());
if (connection.getResponseCode() == 200) {
    String data = readData(connection.getInputStream());
    System.out.println(data);
}
connection.disconnect();
```

PUT

```
var url = "https://fpolyedu.firebaseio.com/students/-ORe_zNPXY0yhvSM9_uh.json";
var connection = (HttpURLConnection) new URL(url).openConnection();
connection.setRequestProperty("Content-Type", "application/json; charset=utf-8");
connection.setRequestMethod("PUT");
var json = """
           {"id": "SV09","name": "Sinh viên 09","mark": 5.5,"gender": true}
           """;
connection.setDoOutput(true);
connection.getOutputStream().write(json.getBytes());
if (connection.getResponseCode() == 200) {
    String data = readData(connection.getInputStream());
    System.out.println(data);
}
connection.disconnect();
```

DELETE

```
var url = "https://fpolyedu.firebaseio.com/students/-ORe_zNPXY0yhvSM9_uh.json";
var connection = (HttpURLConnection) new URL(url).openConnection();
connection.setRequestProperty("Content-Type", "application/json; charset=utf-8");
connection.setRequestMethod("DELETE");
if (connection.getResponseCode() == 200) {
        System.out.println("Success");
}
connection.disconnect();
```

RestTemplate

RestTemplate API

- RestTemplate được cung cấp bởi Spring Boot hỗ trợ tương tác với REST API
- RestTemplate: Thực hiện các REST Operations
  - **getForObject**(String url, Class<**DTO**>): **DTO**
  - **postForObject**(String url, Object data, Class<**DTO**>): **DTO**
  - **put**(String url, Object data)
  - **delete**(String url)

DATA TRANSFER OBJECT (DTO)

```java
@AllArgsConstructor
@NoArgsConstructor
@Builder

@Data
public class Student {
        String id;
        String name;
        boolean gender;
        double mark;
}
```

Mô tả cấu trúc dữ liệu JSON

```json
{
    "id": "SV006",
    "name": "Sinh viên 006",
    "mark": 4.5,
    "gender": true
}
```

DATA TRANSFER OBJECT (DTO)

```
{
      "SV01“: {"id": "SV01","name": "Sinh viên 01","mark": 6.5,"gender": false},
      "SV02“:{"id": "SV02","name": "Sinh viên 02","mark": 7.5,"gender": false},
      "SV03“:{"id": "SV03","name": "Sinh viên 03","mark": 8.5,"gender": false},
      "SV04“:{"id": "SV04","name": "Sinh viên 04","mark": 9.5,"gender": true},
      "SV05“:{"id": "SV05","name": "Sinh viên 05","mark": 5.5,"gender": true}
}
```

Mô tả cấu trúc dữ liệu JSON

```
public class StudentMap extends HashMap<String, Student>{}
```

GET ALL & GET BY KEY

GET ALL

```
RestTemplate restTemplate = new RestTemplate();
var url = "https://fpolyedu.firebaseio.com/students.json";
var students = restTemplate.getForObject(url, StudentMap.class);
```

GET BY KEY

```
RestTemplate restTemplate = new RestTemplate();
var url = "https://fpolyedu.firebaseio.com/students/-ORe_zNPXY0yhvSM9_uh.json";
var student = restTemplate.getForObject(url, Student.class);
```

POST, PUT & DELETE

POST

```
RestTemplate restTemplate = new RestTemplate();
var data = new Student("SV007", "Sinh viên 007", true, 8.5);
var url = "https://fpolyedu.firebaseio.com/students.json";
var key = restTemplate.postForObject(url, data, Map.class);
```

POST

```
RestTemplate restTemplate = new RestTemplate();
var entity = new Student("SV006", "Sinh viên 006", false, 5.5);
var url = "https://fpolyedu.firebaseio.com/students/-ORe_zNPXY0yhvSM9_uh.json";
restTemplate.put(url, entity);
```

DELETE

```
RestTemplate restTemplate = new RestTemplate();
var url = "https://fpolyedu.firebaseio.com/students/-ORe_zNPXY0yhvSM9_uh.json";
restTemplate.delete(url);
```

Spring Boot REST API

- Spring Boot cung cấp một số annotation hỗ trợ tạo REST API một cách nhanh chóng và thuận lợi
  - @RestController
  - @GetMapping <=> GET
  - @PostMapping <=> POST
  - @PutMapping <=> PUT
  - @DeleteMapping <=> DELETE

StudentRestController

```
@RestController
public class StudentRestController {
        Map<String, Student> map = new HashMap<>();

        @GetMapping("/api/students")
        public Collection<Student> get(){...}

        @GetMapping("/api/students/{id}")
        public Student get(@PathVariable("id") String id){...}

        @PostMapping("/api/students")
        public Student post(@RequestBody Student data){...}

        @PutMapping("/api/students/{id}")
        public Student put(@PathVariable("id") String id, @RequestBody Student data){...}

        @DeleteMapping("/api/students/{id}")
        public void delete(@PathVariable("id") String id){...}
}
```

GET(**url**): Collection<Student>

GET(**url**): Student

POST(**url**, **data**): Student

PUT(**url**, **data**): Student

DELETE(**url**)

StudentRestController Inplementation

```
public Collection<Student> get(){
        return map.values();
}
```

GET(url): Collection<Student>

```
public Student get(@PathVariable("id") String id){
        return map.get(id);
}
```

GET(url): Student

```
public Student post(@RequestBody Student data){
        map.put(data.getId(), data);
        return data;
}
```

POST(url, data): Student

```
public Student put(@PathVariable("id") String id, @RequestBody Student data){
        if(map.containsKey(data.getId())) {
                map.put(data.getId(), data);
                return data;
        }
        return null;
}
```

PUT(url, data): Student

```
public void delete(@PathVariable("id") String id){
        map.remove(id);
}
```

DELETE(url)

@CrossOrigin

- Mặc định chỉ có các Rest Consumer cùng domain được cho phép consume các REST API (khác domain sẽ không được phép).
- **@CrossOrigin**() được sử dụng để khai báo cho phép các nguồn địa chỉ Rest Consumer đáng tin cậy.

```
@CrossOrigin(origins = {"http://localhost:8080", "http://127.0.0.1:8080"})
@RestController
public class StudentRestController {...}
```

- Cấu hình này cho phép các trang web đặt tại các host: localhost:8080 và 127.0.0.1:8080 được phép truy cập.
- Sử dụng **origins="*"** để cho phép mọi host

RESPONSEENTITY API

```
@GetMapping("/api/students")
public Collection<Student> get(){
    return map.values();
}
```

```
@GetMapping("/api/students")
public ResponseEntity<Collection<Student>> get(){
    return ResponseEntity.ok(map.values());
}
```

- 2 cách viết mã (1) và (2) là tương đương. Tuy nhiên cách viết (2) cho chúng ta ***mở rộng mã để điều khiển các lỗi*** một cách chính xác (xem slide sau).

RESPONSEENTITY API

```
@GetMapping("/api/students")
public Collection<Student> get(){
        return map.values();
}
```

```
@GetMapping("/api/students")
public ResponseEntity<Collection<Student>> get(){
        if(map.values().isEmpty()) {
                return ResponseEntity.noContent().build();
        }
        return ResponseEntity.ok(map.values());
}
```

Rest Consumer sẽ nhận được trạng thái với mã 204. Từ đó có thể đưa ra các xử lý, thông báo phù hợp

RETURN VOID

- Void là class đại diện cho void, được khai báo cho các phương thức không trả về kết quả

```
@DeleteMapping("/api/students/{id}")
public ResponseEntity<Void> delete(@PathVariable("id") String id){
        map.remove(id);
        return ResponseEntity.ok().build();
}
```

ResponseEntity API

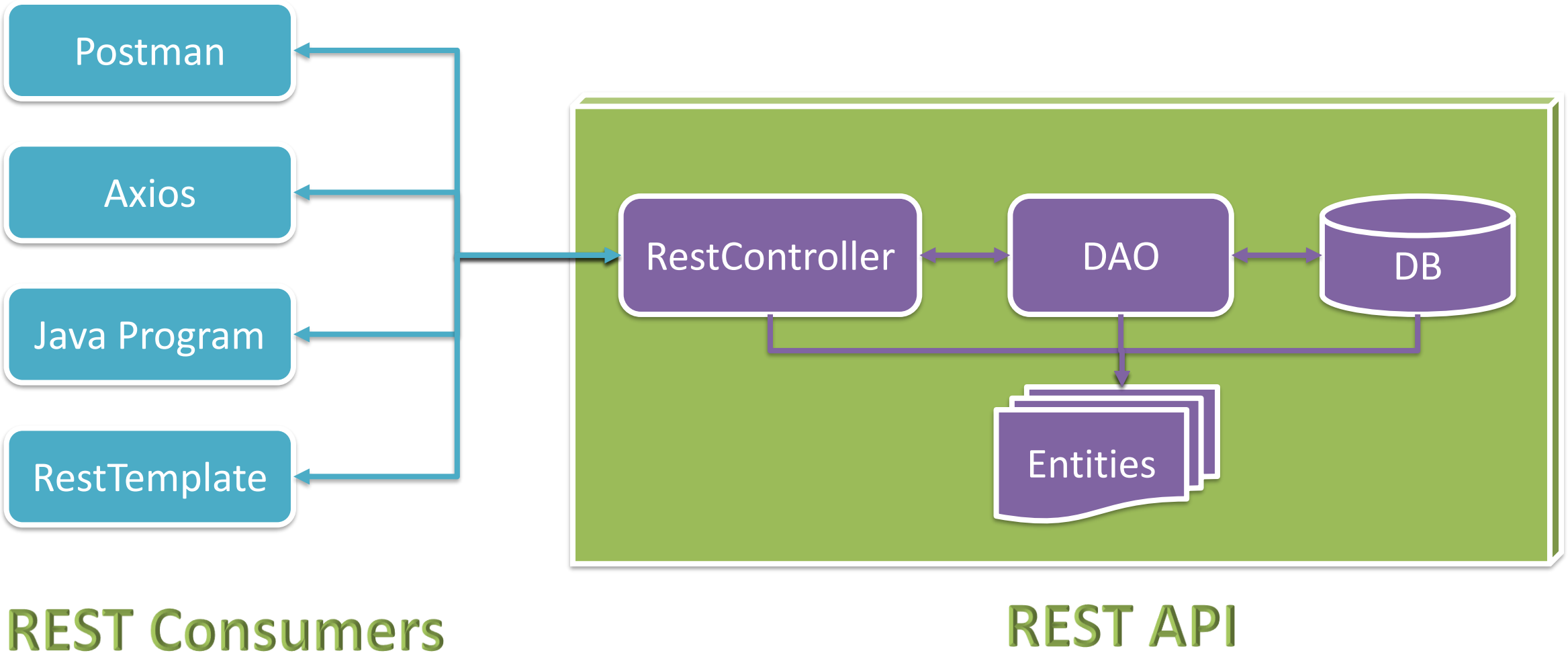
- ResponseEntity.**badRequest**().build()
  - **400** Bad Request: Địa chỉ tồi
- ResponseEntity.**noContent**().build()
  - **204** No Content: Không có nội dung
- ResponseEntity.**notFound**().build()
  - **404** Not Found: Không tìm thấy
- ResponseEntity.**ok**(body)
  - **200** OK: Thành công
- ResponseEntity.**status**(**HttpStatus**).build()
  - Status Code: Chứa trạng thái tùy chọn

USING RESPONSEENTITY

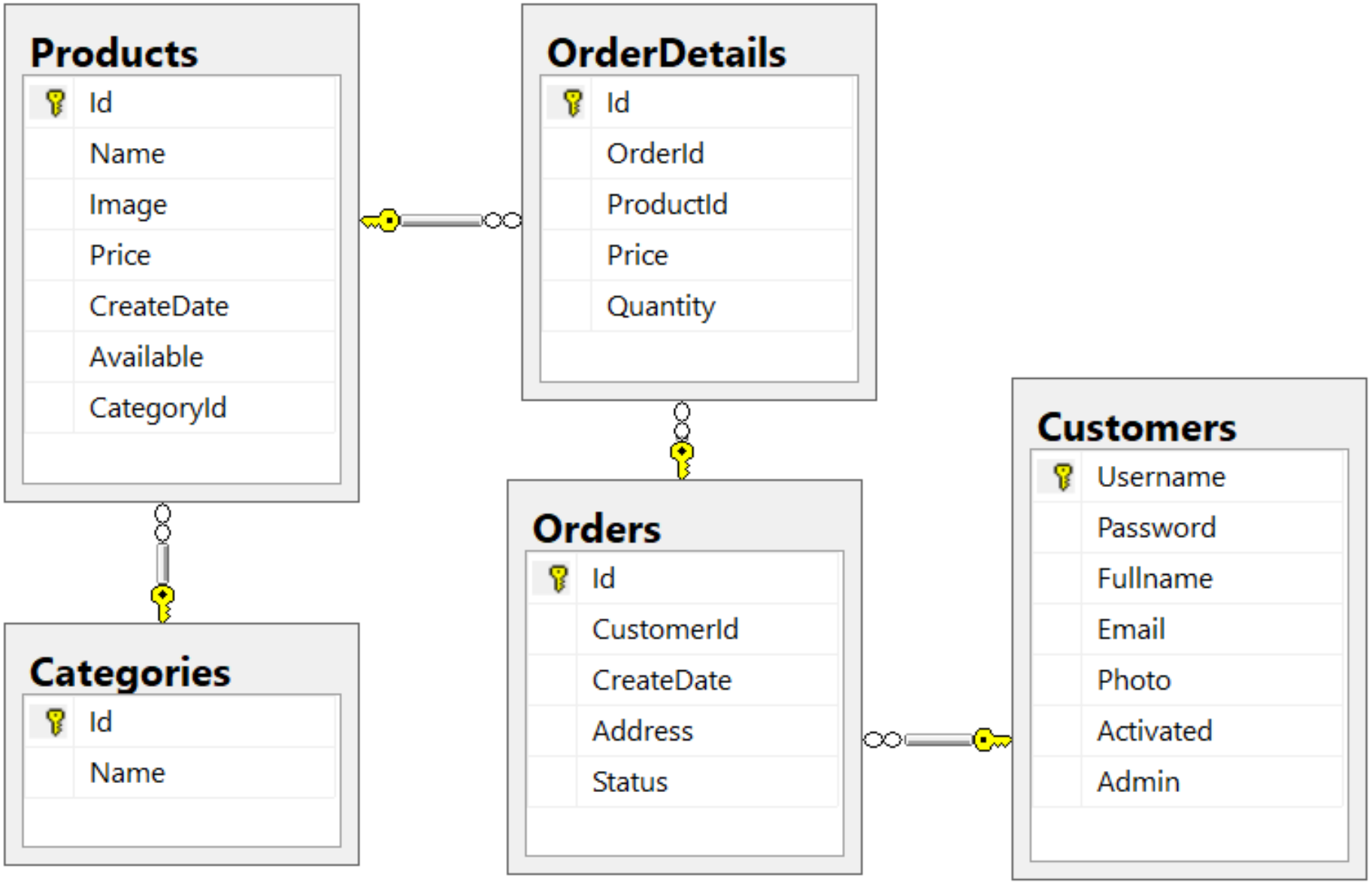
```
@CrossOrigin(origins = "*")
@RestController
public class StudentApiController {
    public ResponseEntity<Collection<Student>> get(){...}

    public ResponseEntity<Student> get(@PathVariable("id") String id){...}

    public ResponseEntity<Student> post(@RequestBody Student data){...}

    public ResponseEntity<Student> put(@PathVariable("id") String id,
            @RequestBody Student data){...}

    public ResponseEntity<Void> delete(@PathVariable("id") String id){...}
}
```

REST API WITH JPAREPOSITORY

REST API & JPAREPOSITORY

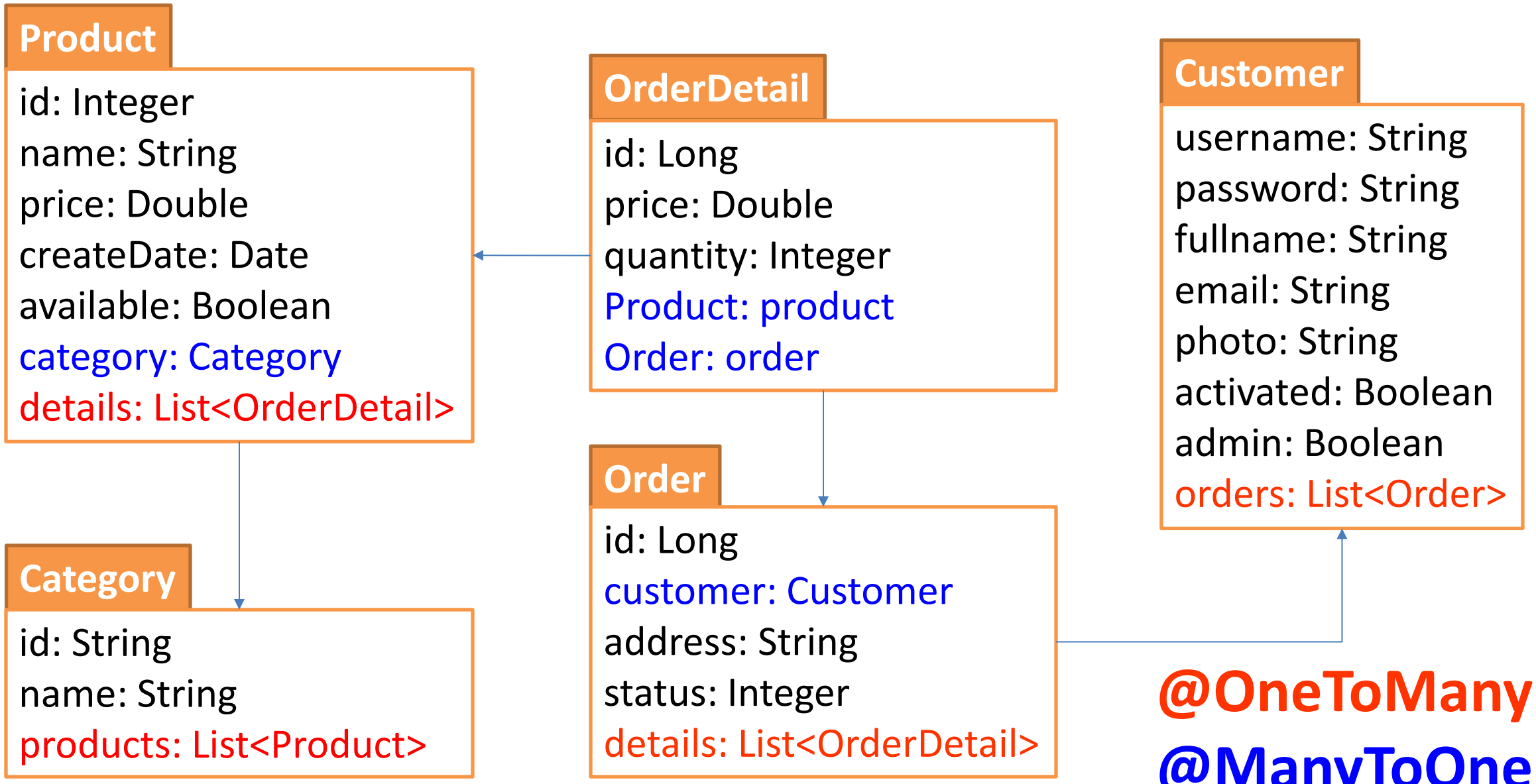


Database Relationship Diagram



FPT Education
FPT POLYTECHNIC

ERD – Entity Relational Diagram



Association Entity Class Mapping

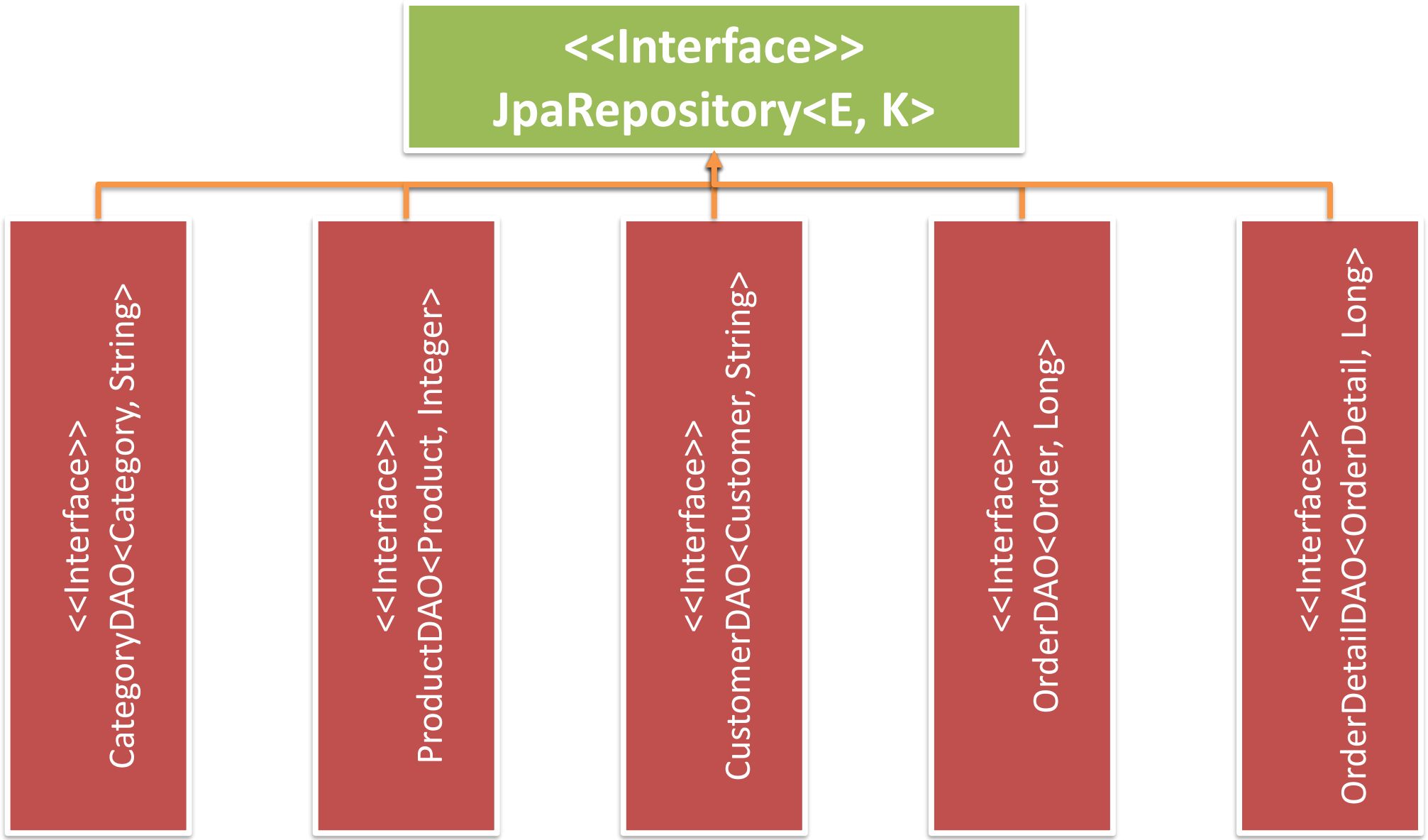
Categories

	Column Name	Condensed Type	Nullable	Identity
🔑	Id	char(4)	No	☐
	Name	nvarchar(50)	No	☐
				☐

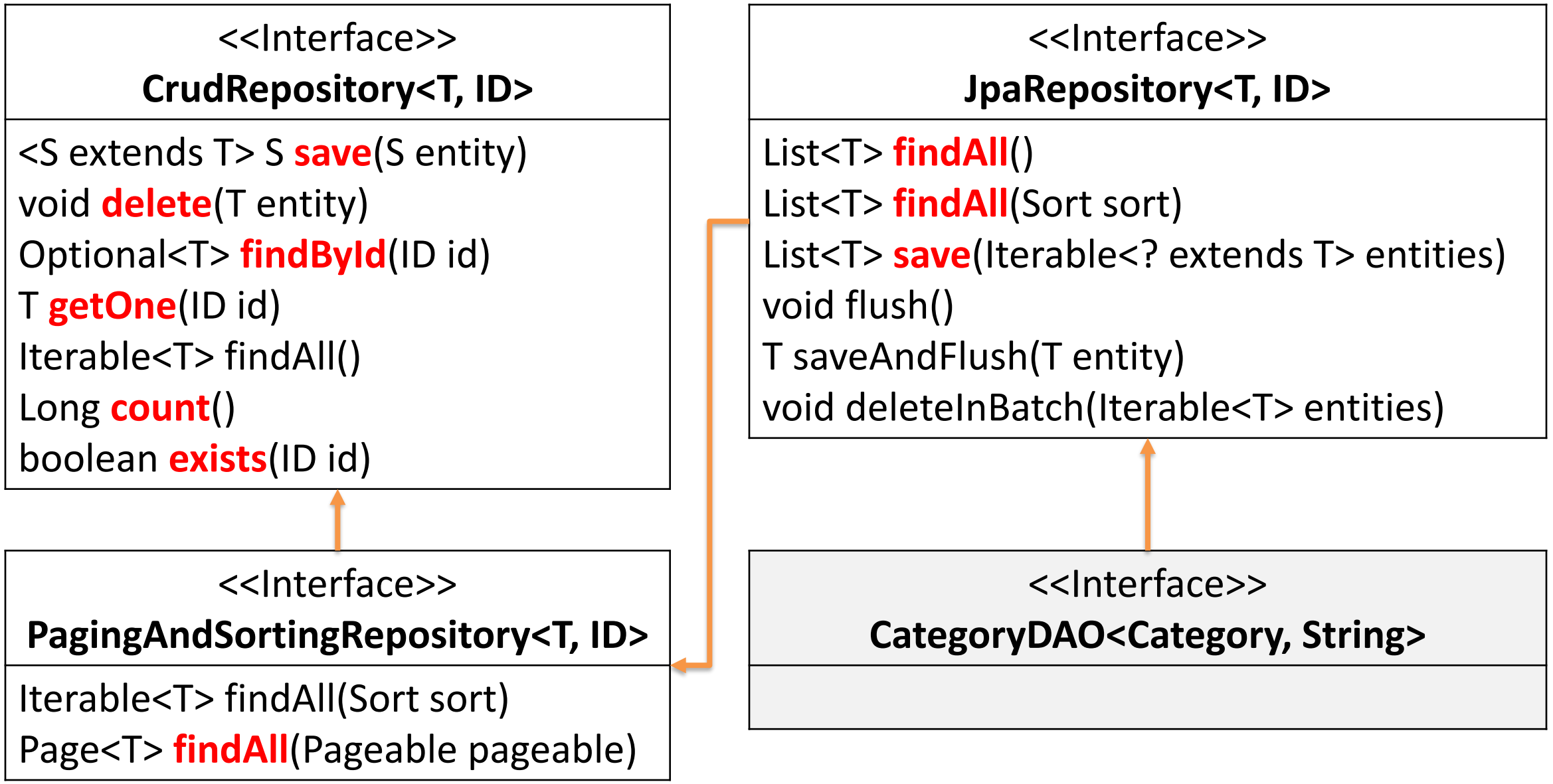
```
@Data
@Entity
@Table(name = "Categories")
public class Category {
    @Id
    String id;
    String name;
    @JsonIgnore
    @OneToMany(mappedBy = "category")
    List<Product> products;
}
```

*Thêm **@JsonInorge** vào các kết hợp **@OneToMany** để loại bỏ thuộc tính này khi chuyển đổi từ Java Object sang JSON.*

DAO Class Diagram



REVIEW JPAREPOSITORY API



CATEGORYRESTCONTROLLER

- **GET**: **/api/categories**
  - *() => List<Category>*
- **GET**: /api/categories**/{id}**
  - *(id) => Category*
- **POST**: /api/categories & **Catetory**
  - *(Category) => Category*
- **PUT**: /api/categories**/{id}** & **Category**
  - *(id, Category) => Category*
- **DELETE**: /api/categories**/{id}**
  - *(id) => Void*

CategoryRestController Structure

```
@RestController
@CrossOrigin(origins = "*")
@RequestMapping("/api/categories")
public class CategoryController {
        @GetMapping
        public ResponseEntity<List<Category>> findAll() {...}
        @GetMapping("{id}")
        public ResponseEntity<Category> findById(@PathVariable("id") String id) {...}
        @PostMapping()
        public ResponseEntity<Category> post(@RequestBody Category category) {...}
        @PutMapping("{id}")
        public ResponseEntity<Category> put(@PathVariable("id") String id,
                                            @RequestBody Category category) {...}
        @DeleteMapping("{id}")
        public ResponseEntity<Void> delete(@PathVariable("id") String id) {...}
}
```

CategoryRestController Implementation

```
@Autowired
CategoryDAO cdao;
```

findAll()

```
return ResponseEntity.ok(cdao.findAll());
```

findById(String id)

```
Optional<Category> optional = cdao.findById(id);
if(!optional.isPresent()) {
    return ResponseEntity.notFound().build();
}
return ResponseEntity.ok(optional.get());
```

CategoryRestController Implementation

```
@Autowired
CategoryDAO cdao;
```

post(Category category)

```
if(cdao.existsById(category.getId())) {
        return ResponseEntity.badRequest().build();
}
cdao.save(category);
return ResponseEntity.ok(category);
```

**put(String id,
Category category)**

```
if(!cdao.existsById(id)) {
        return ResponseEntity.notFound().build();
}
cdao.save(category);
return ResponseEntity.ok(category);
```

delete(String id)

```
if(!cdao.existsById(id)) {
        return ResponseEntity.notFound().build();
}
cdao.deleteById(id);
return ResponseEntity.ok().build();
```

TỔNG KẾT

- ☑ Rest Consumer using HttpURLConnection
- ☑ RestConsumer using RestTemplate
- ☑ Spring Boot REST API
- ☑ Spring Boot REST API with JpaRepository


TIN HỌC VĂN PHÒNG QUỐC TẾ THEO CHUẨN ICDL
Syllabus 5 - Office 2007
LUẬT KINH TẾ
LUẬT KINH TẾ
KỸ THUẬT SOẠN THẢO VÀ TRÌNH BÀY VĂN BẢN
TIN HỌC VĂN PHÒNG QUỐC TẾ THEO CHUẨN ICDL
Syllabus 5 - Office 2007
Cảm ơn