

HƯỚNG DẪN TẠO DỰ ÁN VỚI RESTful API

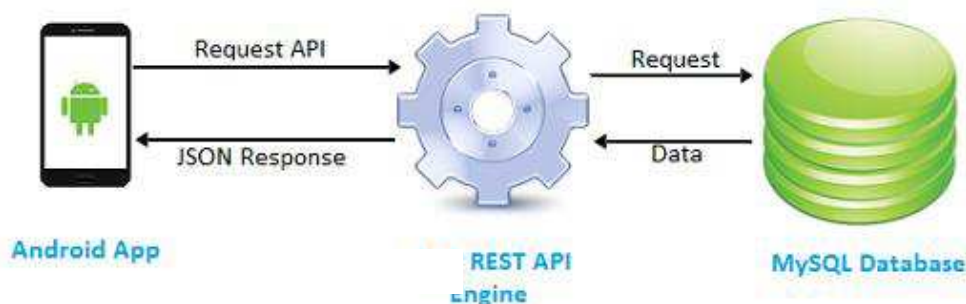
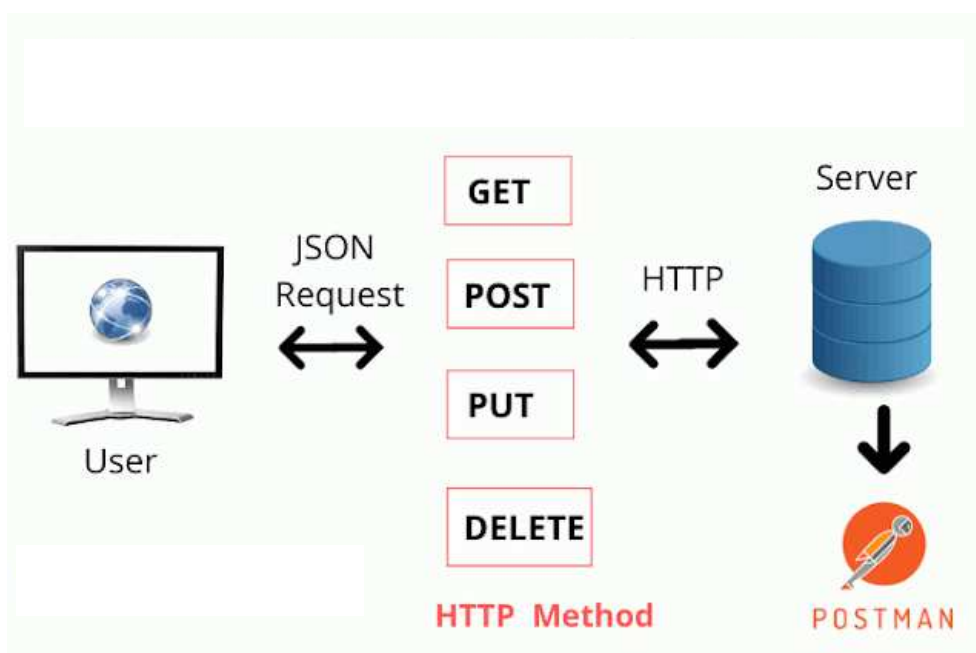


Mục lục

1. Giới thiệu	3
2. Các bước thực hiện tạo Restful API.....	4
3. Ví dụ mẫu	5
4. Tự làm.....	23

1. Giới thiệu

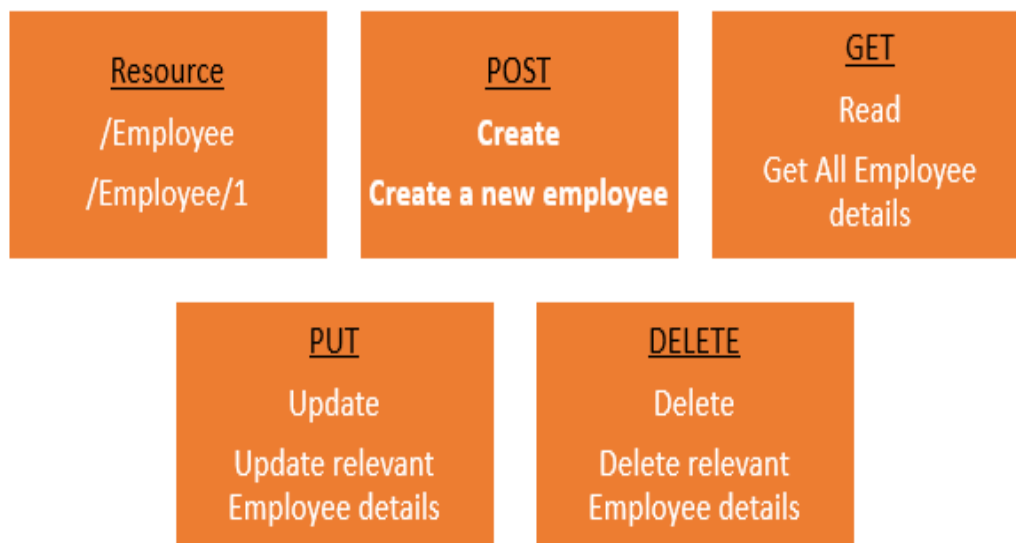
- a) API viết tắt **A**pplication **P**rogramming **I**nterface – Tạm dịch là giao diện lập trình ứng dụng. Nó cho phép các ứng dụng khác nhau có thể giao tiếp được với nhau. Ví dụ bạn thiết kế chức năng login thông Google, Facebook
- b) REST là một cách để truy cập các **tài nguyên** nằm trong một môi trường cụ thể (môi trường internet)
- c) RESTful là một trong những kiểu thiết kế API được sử dụng phổ biến ngày nay để cho các ứng dụng (web, mobile...) khác nhau giao tiếp với nhau
- d) Ví dụ về Restful



e) Các phương thức Restful

- POST - **Create**
- GET - **Read**
- PUT - **Update**
- DELETE - **Delete**

f) Ví dụ về các giao thức Restful



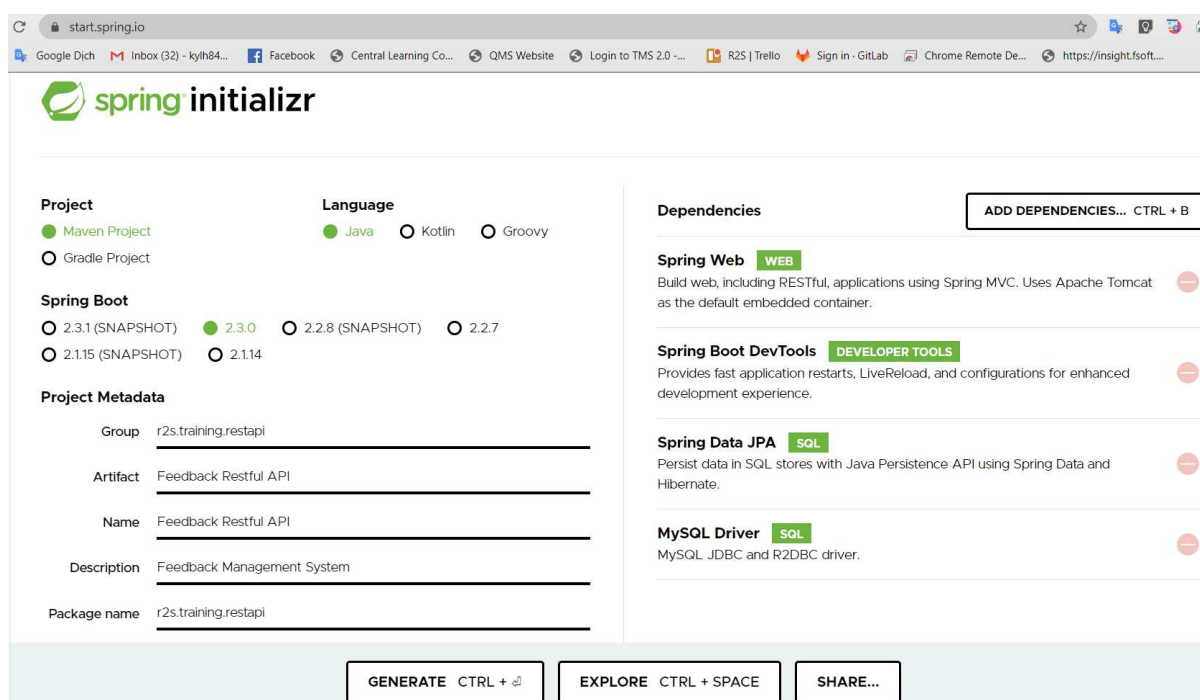
2. Các bước thực hiện tạo Restful API

- Trong bài lab này chúng ta sẽ học cách tạo REST API sử dụng Spring Boot, JPA, Hibernate và MySQL (Database)
- Bao gồm 6 các bước sau
 - Tạo một Spring Boot Project
 - Cấu hình Database
 - Tạo Entity Class
 - Tạo JPA Data Repository
 - Tạo Rest Controllers và điều phối API requests
 - Build và run project

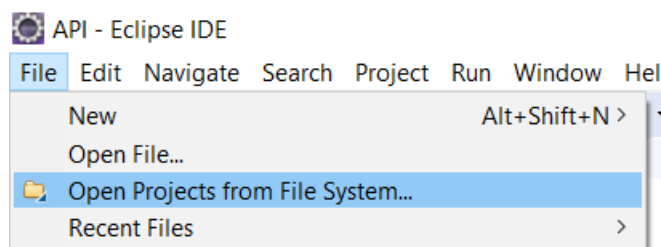
3. Ví dụ mẫu

1) Tạo một Spring Boot Project

- Truy cập <https://start.spring.io/> và tạo một project với những thiết lập sau
 - **Web:** Full-stack web development with Tomcat and Spring MVC
 - **DevTools:** Spring Boot Development Tools
 - **JPA:** Java Persistence API including spring-data-JPA, spring-orm, and Hibernate
 - **MySQL:** MySQL JDBC driver

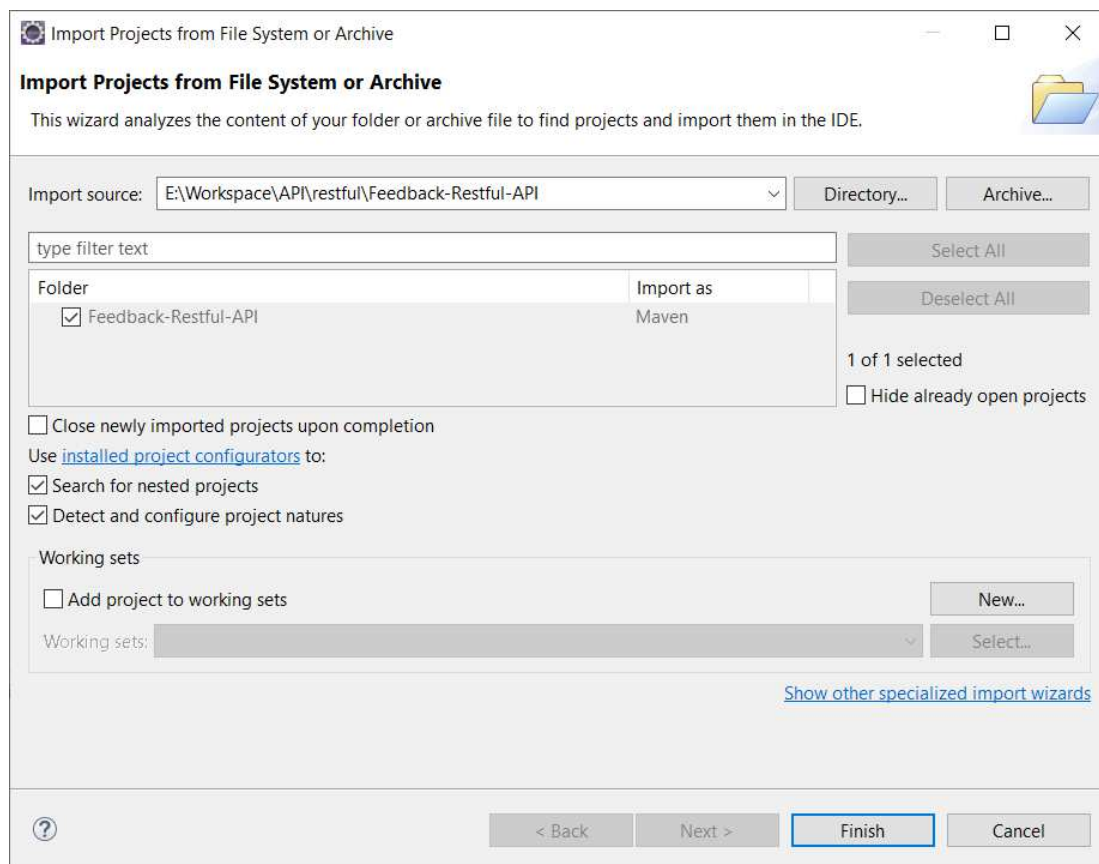


- Chọn Generate -> sau đó download và import vào eclipse
 - Mở eclipse -> chọn File -> chọn Open projects from File System



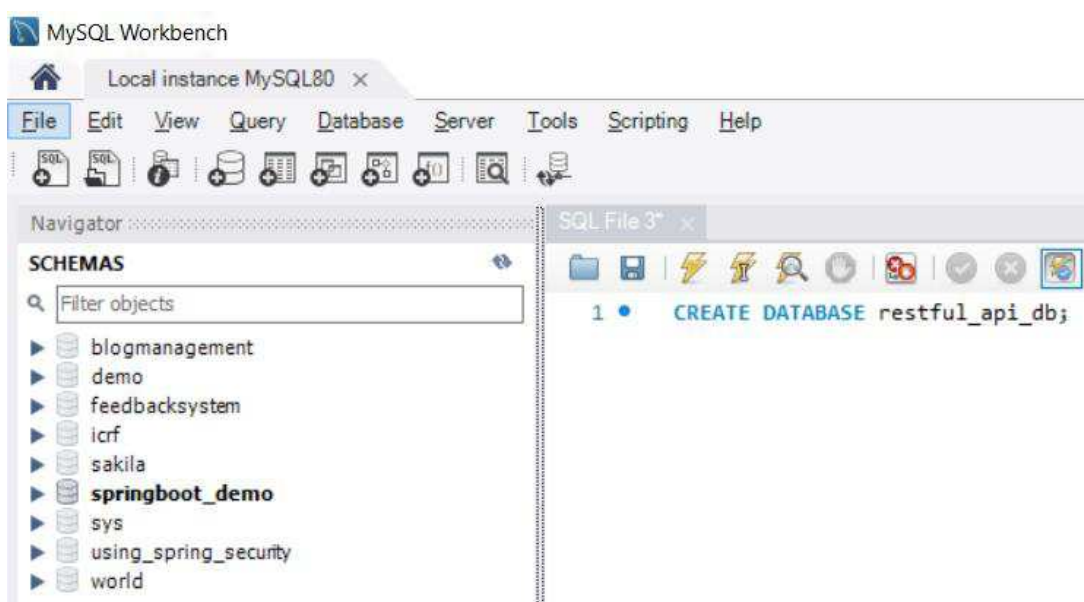
- Chọn Directory để chỉ định thư mục chứa project sau khi tải về và giải nén -> chọn Finish để import project vào eclipse, sau đó chờ

eclipse tải những library liên quan (Lúc này máy tính của bạn phải có kết nối internet)

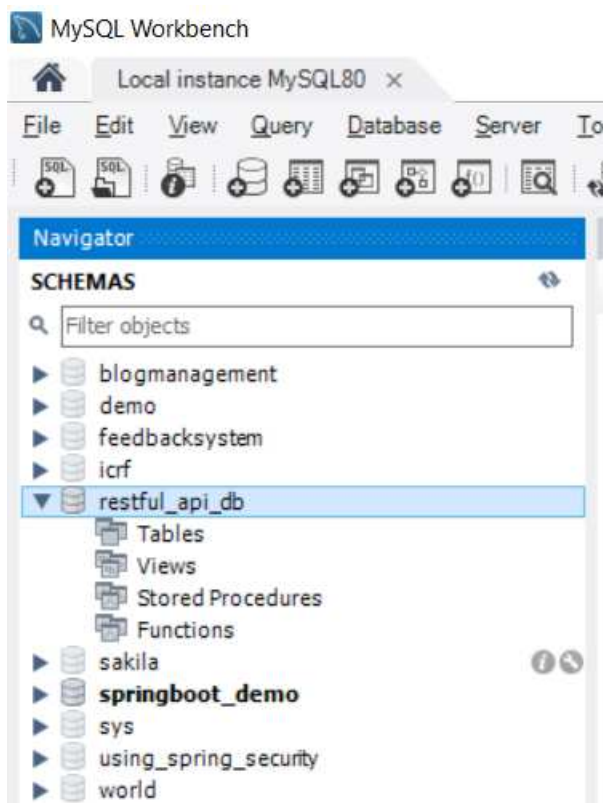


2) Cấu hình Database

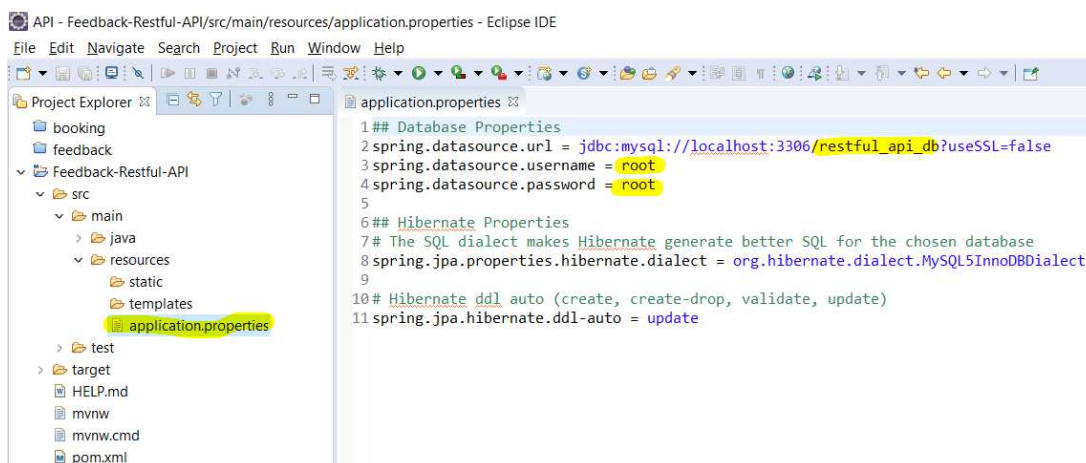
- Tại bước này chúng ta tạo một database tên **restful_api_db** trong MySQL



- Sau khi tạo thành công database

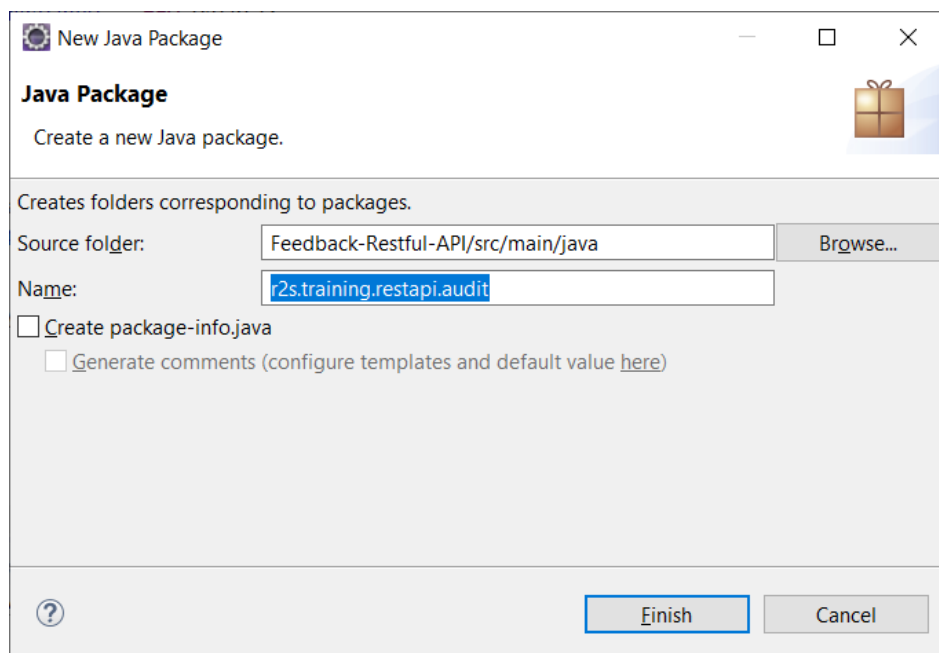


- Mở `src/main/resources/application.properties` để thiết lập kết nối đến MySQL; thiết lập Hibernate

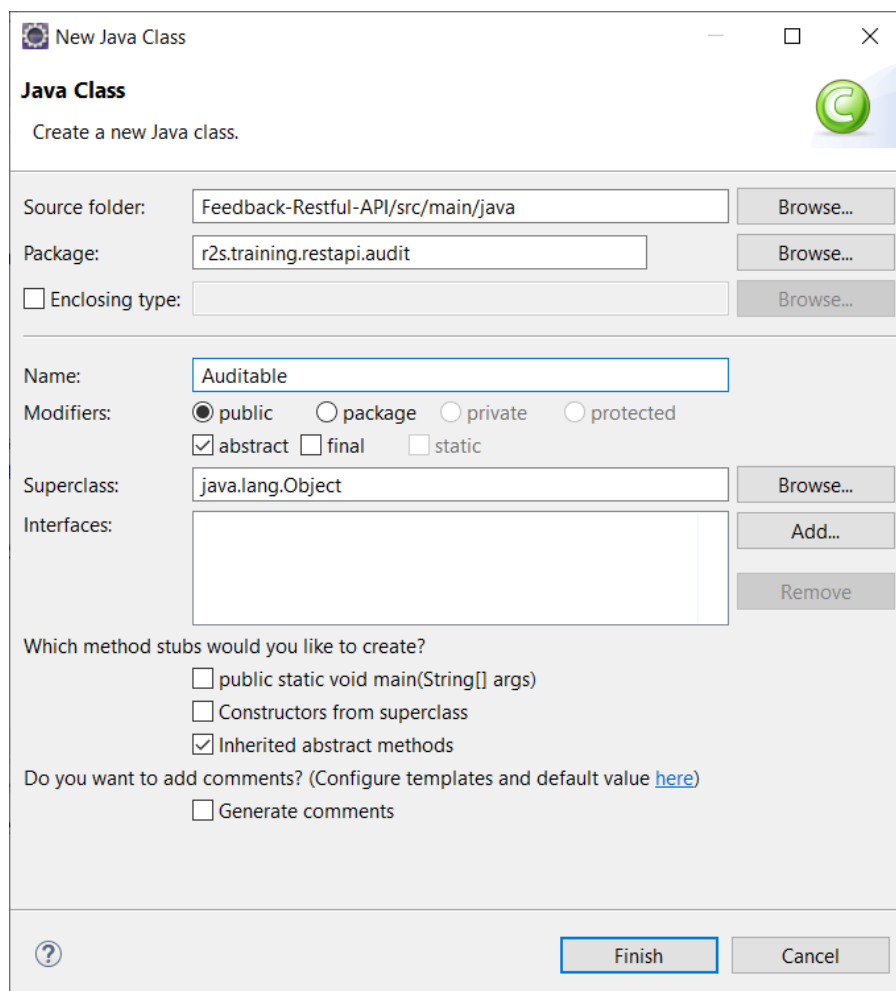


3) Tạo Entity Class

- Tạo mới package tên `r2s.training.restapi.audit`



- Chuột phải **r2s.training.restapi.audit** -> chọn New -> chọn Class -> chọn abstract và nhập **Auditable** tại Name -> chọn Finish



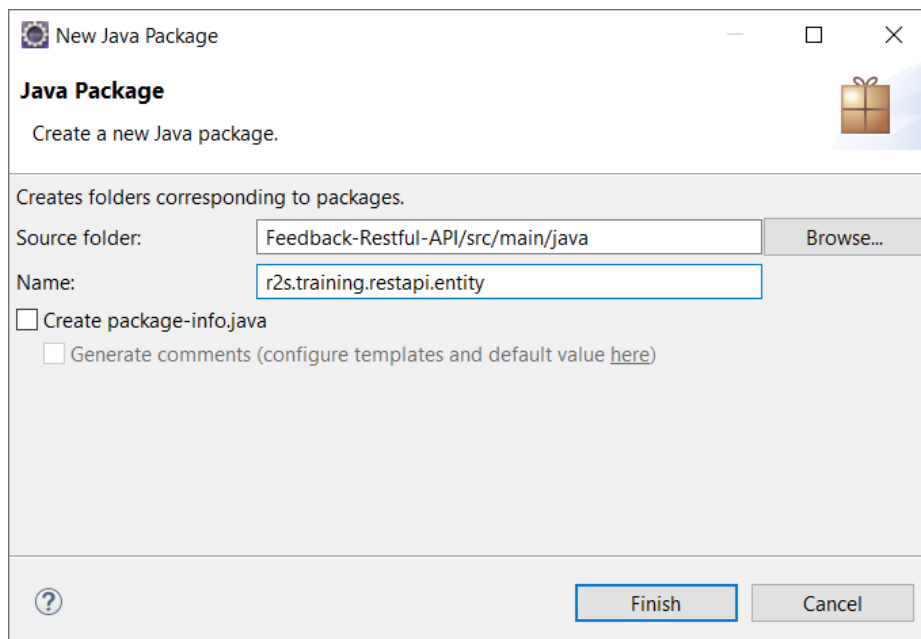
- Nhập code cho Auditable

```

3 import java.util.Date;
4
5 import javax.persistence.EntityListeners;
6 import javax.persistence.MappedSuperclass;
7 import javax.persistence.Temporal;
8 import javax.persistence.TemporalType;
9
10 import org.springframework.data.annotation.CreatedBy;
11 import org.springframework.data.annotation.CreatedDate;
12 import org.springframework.data.annotation.LastModifiedBy;
13 import org.springframework.data.annotation.LastModifiedDate;
14 import org.springframework.data.jpa.domain.support.AuditingEntityListener;
15
16 @MappedSuperclass
17 @EntityListeners(AuditingEntityListener.class)
18 public abstract class Auditable<U> {
19
20     @CreatedBy
21     protected U createdBy;
22
23     @CreatedDate
24     @Temporal(TemporalType.TIMESTAMP)
25     protected Date createdAt;
26
27     @LastModifiedBy
28     protected U lastModifiedBy;
29
30     @LastModifiedDate
31     @Temporal(TemporalType.TIMESTAMP)
32     protected Date lastModifiedDate;
33
34     public U getCreatedBy() {
35         return createdBy;
36     }
37
38     public void setCreatedBy(U createdBy) {
39         this.createdBy = createdBy;
40     }
41
42     public Date getCreatedAt() {
43         return createdAt;
44     }
45
46     public void setCreatedAt(Date createdAt) {
47         this.createdAt = createdAt;
48     }
49
50     public U getLastModifiedBy() {
51         return lastModifiedBy;
52     }
53
54     public void setLastModifiedBy(U lastModifiedBy) {
55         this.lastModifiedBy = lastModifiedBy;
56     }
57
58     public Date getLastModifiedDate() {
59         return lastModifiedDate;
60     }
61
62     public void setLastModifiedDate(Date lastModifiedDate) {
63         this.lastModifiedDate = lastModifiedDate;
64     }
65

```

- Tạo mới package tên r2s.training.restapi.entity



New Java Package

Create a new Java package.

Creates folders corresponding to packages.

Source folder: Feedback-Restful-API/src/main/java Browse...

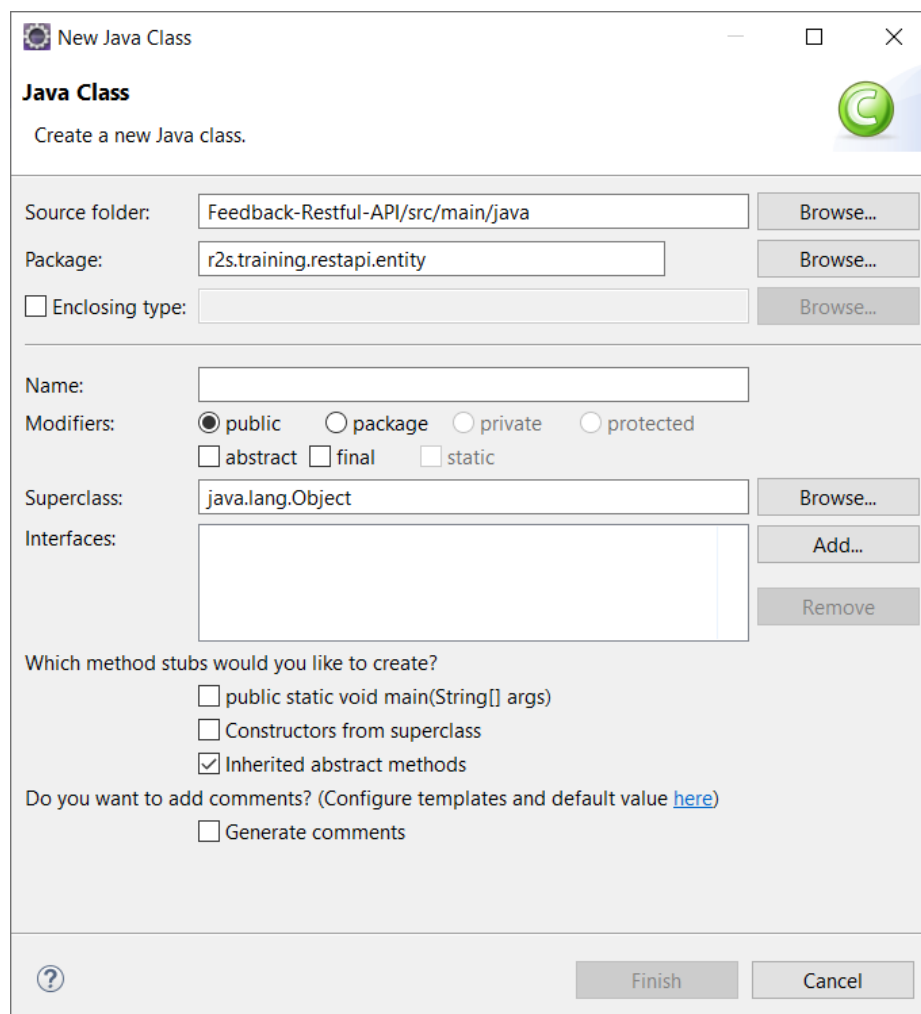
Name: r2s.training.restapi.entity

☐ Create package-info.java

☐ Generate comments (configure templates and default value [here](#))

? Finish Cancel

- Tạo một Java Class tên **User**
 - Chuột phải **r2s.training.restapi.entity** -> chọn New -> chọn Class



New Java Class

Create a new Java class.

Source folder: Feedback-Restful-API/src/main/java Browse...

Package: r2s.training.restapi.entity Browse...

☐ Enclosing type: Browse...

Name:

Modifiers: ☒ public ☐ package ☐ private ☐ protected

☐ abstract ☐ final ☐ static

Superclass: java.lang.Object Browse...

Interfaces: Add... Remove

Which method stubs would you like to create?

☐ public static void main(String[] args)

☐ Constructors from superclass

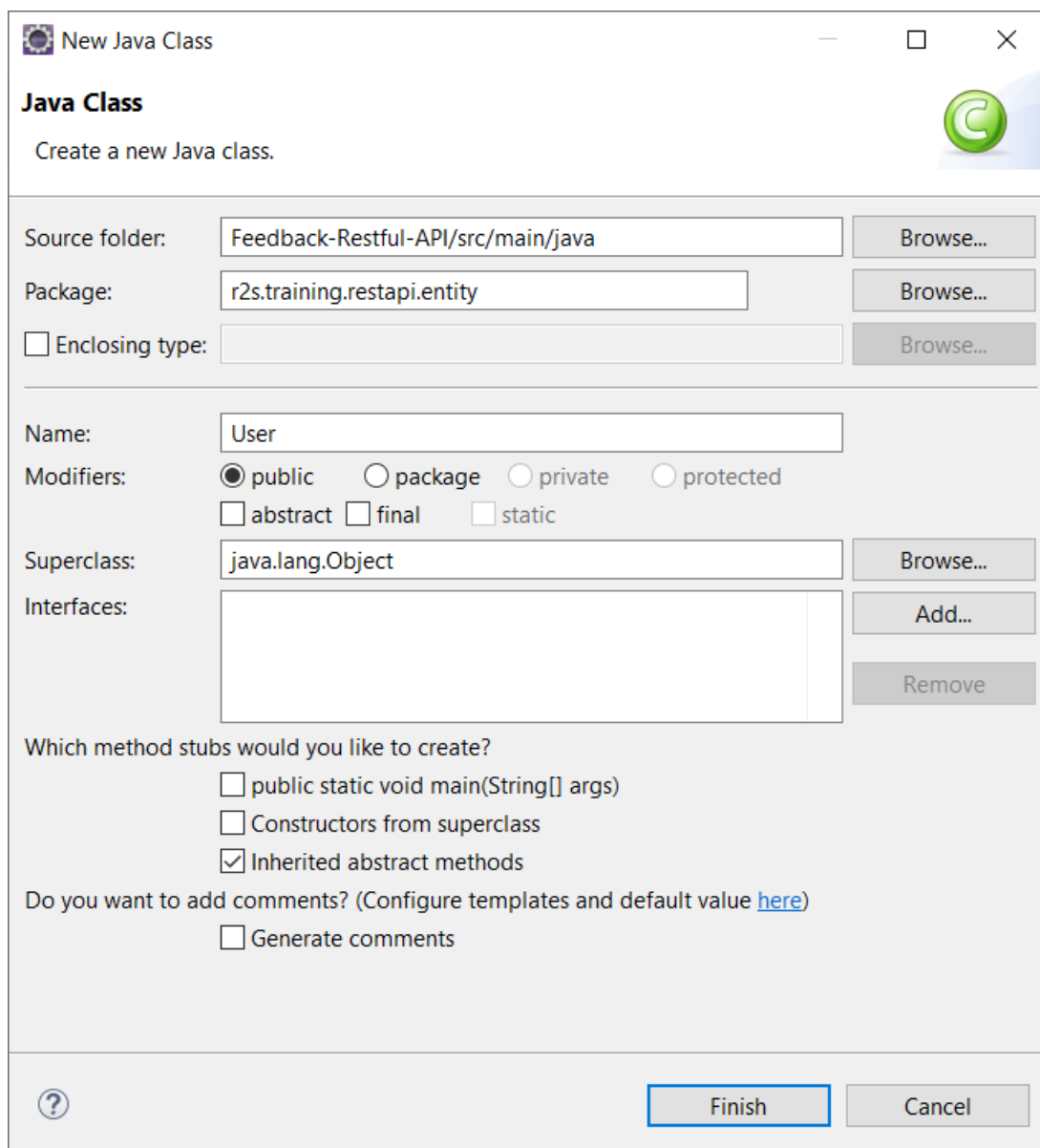
☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))

☐ Generate comments

? Finish Cancel

- Nhập **User** tại Name -> chọn Finish



New Java Class

Java Class
Create a new Java class.

Source folder: Feedback-Restful-API/src/main/java Browse...

Package: r2s.training.restapi.entity Browse...

☐ Enclosing type: Browse...

Name: User

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass: java.lang.Object Browse...

Interfaces: Add...
Remove

Which method stubs would you like to create?

☐ public static void main(String[] args)
☐ Constructors from superclass
☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))
☐ Generate comments

? Finish Cancel

- Nhập code cho User

```

11 import org.springframework.data.jpa.domain.support.AuditingEntityListener;
12
13 import r2s.training.restapi.audit.Auditable;
14
15 /**
16  * User entity
17  *
18  * @author kyle
19  *
20  */
21 @Entity
22 @Table(name = "users")
23 @EntityListeners(AuditingEntityListener.class)
24 public class User extends Auditable<String> {
25
26     @Id
27     @GeneratedValue(strategy = GenerationType.AUTO)
28     private long id;
29
30     @Column(name = "first_name", nullable = false)
31     private String firstName;
32
33     @Column(name = "last_name", nullable = false)
34     private String lastName;
35
36     @Column(name = "email_address", nullable = false)
37     private String email;
38
39
40     /**
41      * Gets id.
42      *
43      * @return the id
44      */
45     public long getId() {
46         return id;
47     }
48

```

```

49     /**
50      * Sets id.
51      *
52      * @param id the id
53      */
54     public void setId(long id) {
55         this.id = id;
56     }
57
58     /**
59      * Gets first name.
60      *
61      * @return the first name
62      */
63     public String getFirstName() {
64         return firstName;
65     }
66
67     /**
68      * Sets first name.
69      *
70      * @param firstName the first name
71      */
72     public void setFirstName(String firstName) {
73         this.firstName = firstName;
74     }
75
76     /**
77      * Gets last name.
78      *
79      * @return the last name
80      */
81     public String getLastName() {
82         return lastName;
83     }

```

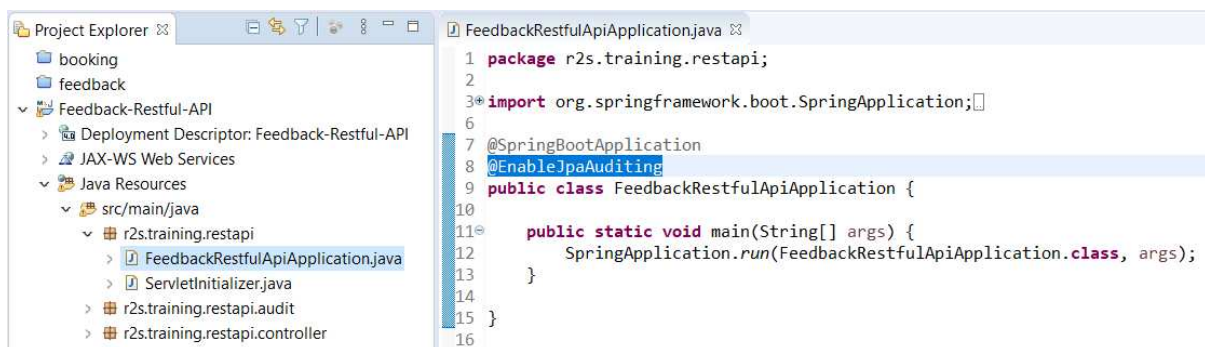
```

84
85  /**
86   * Sets last name.
87   *
88   * @param lastName the last name
89   */
90  public void setLastName(String lastName) {
91      this.lastName = lastName;
92  }
93
94  /**
95   * Gets email.
96   *
97   * @return the email
98   */
99  public String getEmail() {
100      return email;
101  }
102
103  /**
104   * Sets email.
105   *
106   * @param email the email
107   */
108  public void setEmail(String email) {
109      this.email = email;
110  }
111
112 }

```

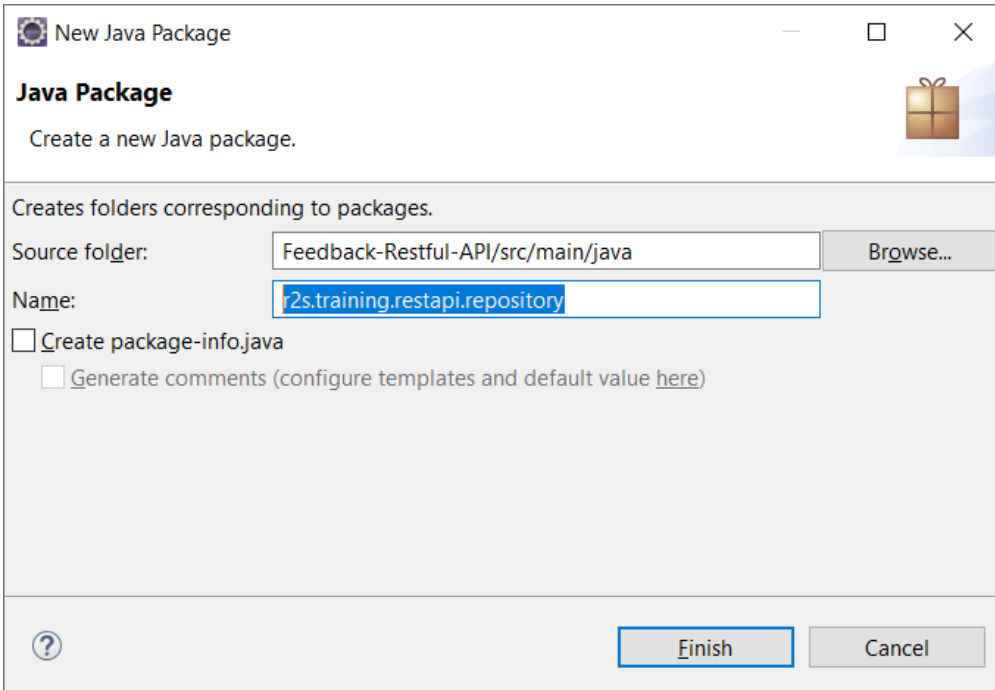
- Cấu hình **Auditing**

- Mở FeedbackRestfulApiApplication và thêm **@EnableJpaAuditing**



4) Tạo JPA Data Repository

- Tạo mới package tên **r2s.training.restapi.repository**



New Java Package

Create a new Java package.

Creates folders corresponding to packages.

Source folder: Feedback-Restful-API/src/main/java Browse...

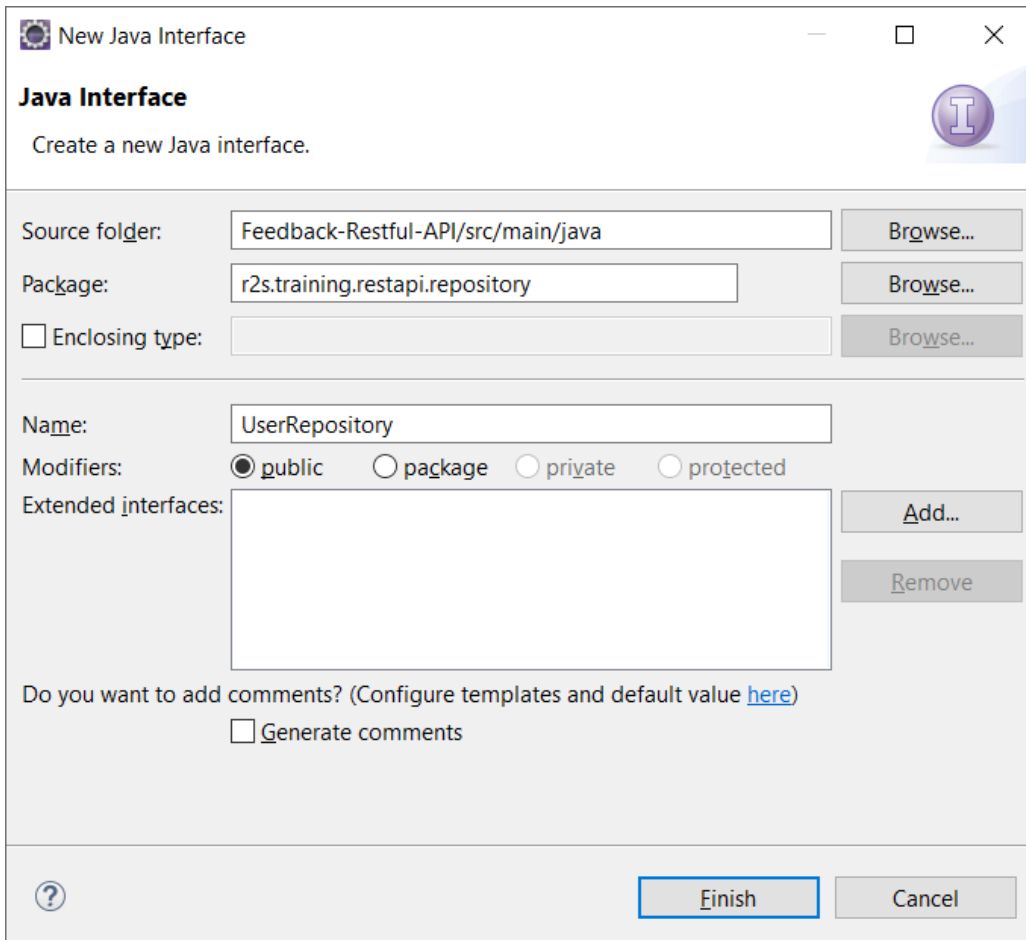
Name: r2s.training.restapi.repository

☐ Create package-info.java

☐ Generate comments (configure templates and default value [here](#))

? Finish Cancel

- Chuột phải **r2s.training.restapi.repository** -> chọn New -> chọn interface -> nhập **UserRepository** tại Name -> chọn Finish



New Java Interface

Create a new Java interface.

Source folder: Feedback-Restful-API/src/main/java Browse...

Package: r2s.training.restapi.repository Browse...

☐ Enclosing type: Browse...

Name: UserRepository

Modifiers: ☒ public ☐ package ☐ private ☐ protected

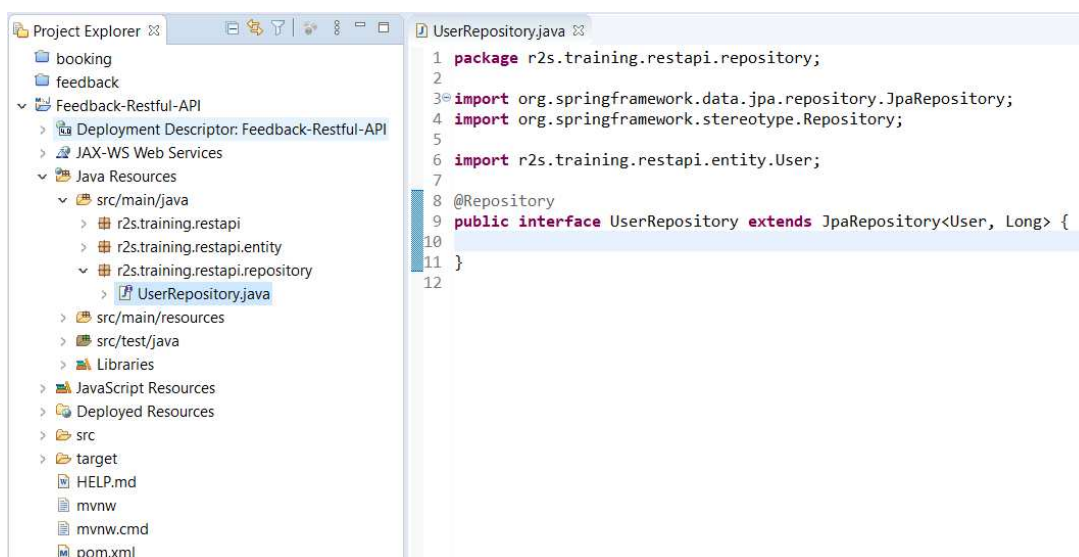
Extended interfaces: Add... Remove

Do you want to add comments? (Configure templates and default value [here](#))

☐ Generate comments

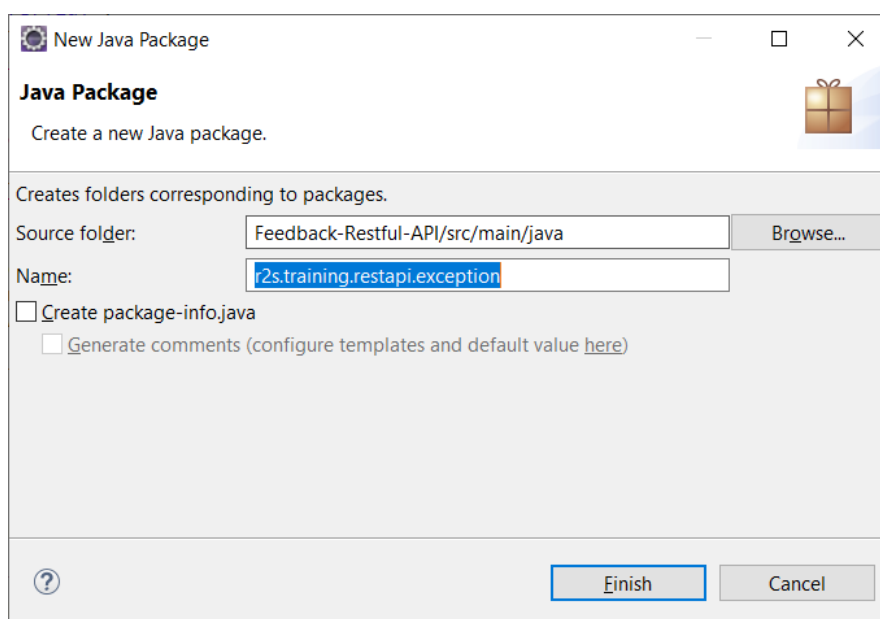
? Finish Cancel

- Nhập code cho UserRepository



5) Tạo Rest Controllers và điều phối API requests

- Tạo mới package tên **r2s.training.restapi.exception**



- Chuột phải **r2s.training.restapi.exception** -> chọn New -> chọn Class -> nhập **ResourceNotFoundException** tại Name -> chọn Finish

New Java Class

Create a new Java class.

Source folder: Feedback-Restful-API/src/main/java Browse...

Package: r2s.training.restapi.exception Browse...

☐ Enclosing type: Browse...

Name: ResourceNotFoundException

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass: java.lang.Object Browse...

Interfaces: Add... Remove

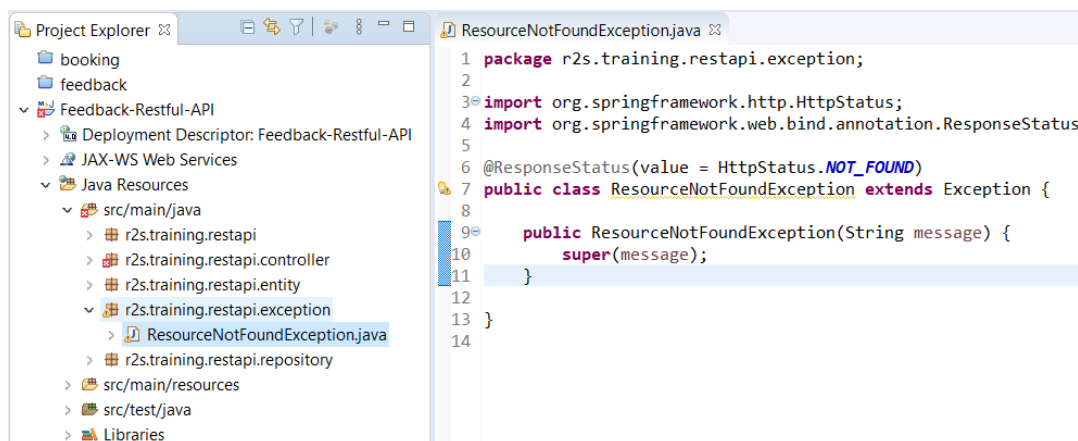
Which method stubs would you like to create?

☐ public static void main(String[] args)
☐ Constructors from superclass
☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))
☐ Generate comments

? Finish Cancel

- Nhập code cho ResourceNotFoundException



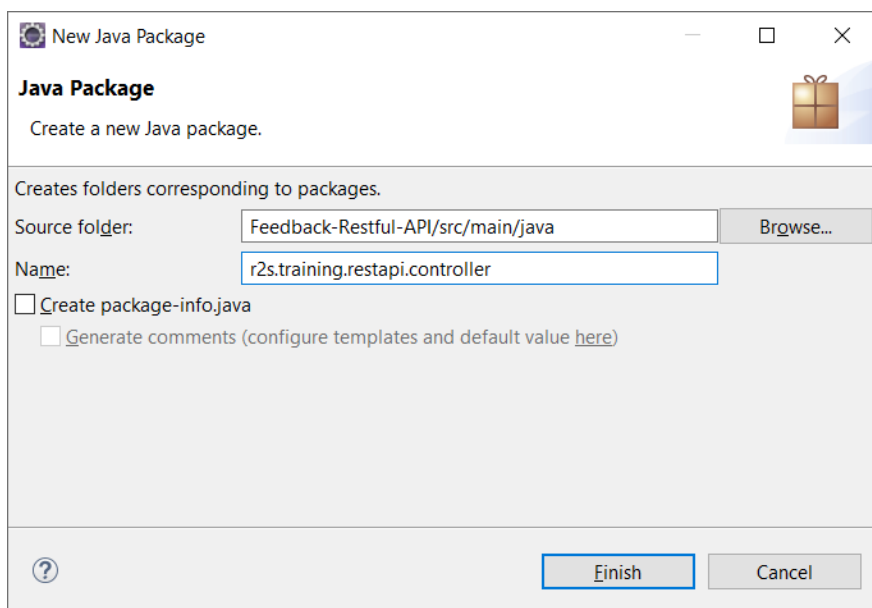
The screenshot shows the Project Explorer on the left with the package structure expanded to `r2s.training.restapi.exception`, where `ResourceNotFoundException.java` is selected. The main editor displays the following code:

```

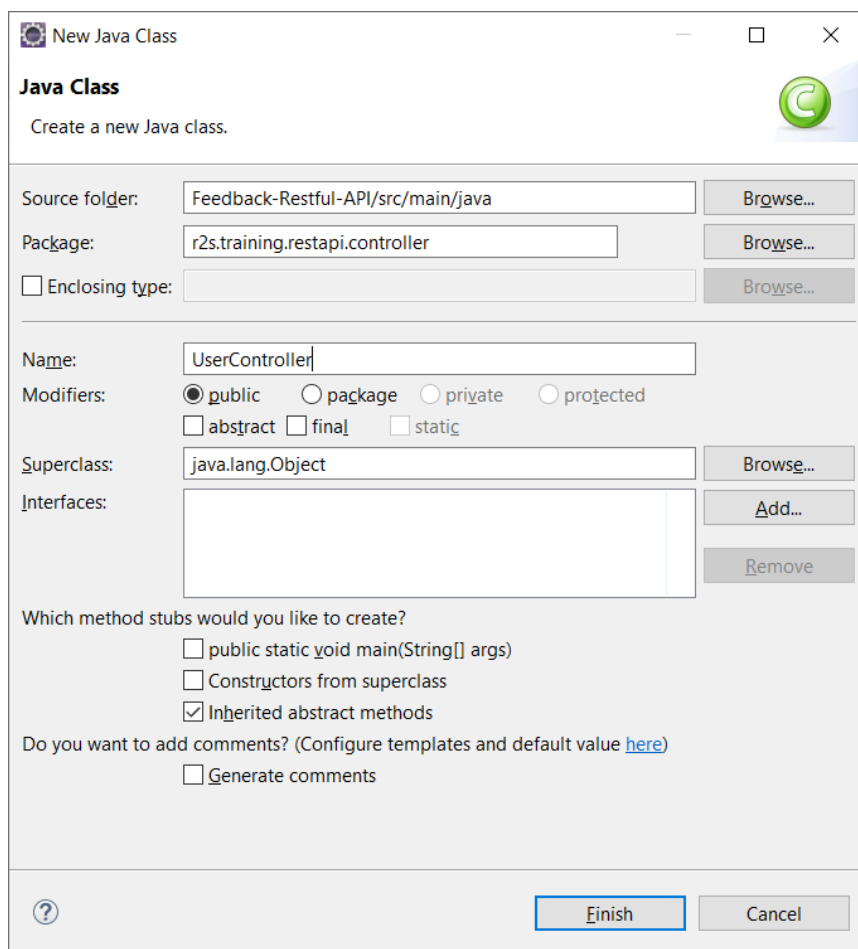
1 package r2s.training.restapi.exception;
2
3 import org.springframework.http.HttpStatus;
4 import org.springframework.web.bind.annotation.ResponseStatus;
5
6 @ResponseStatus(value = HttpStatus.NOT_FOUND)
7 public class ResourceNotFoundException extends Exception {
8
9     public ResourceNotFoundException(String message) {
10         super(message);
11     }
12 }
13
14

```

- Tạo mới package tên `r2s.training.restapi.controller`



- Chuột phải **r2s.training.restapi.controller** -> chọn New -> chọn Class -> nhập **UserController** tại Name -> chọn Finish



- Nhập code cho UserController

```

23 @RestController
24 @RequestMapping("/api/v1")
25 public class UserController {
26
27     @Autowired
28     private UserRepository userRepository;
29
30     /**
31      * Get all users list.
32      *
33      * @return the list
34      */
35     @GetMapping("/users")
36     public List<User> getAllUsers() {
37         return userRepository.findAll();
38     }
39
40     /**
41      * Gets users by id.
42      *
43      * @param userId the user id
44      * @return the users by id
45      * @throws ResourceNotFoundException the resource not found exception
46      */
47     @GetMapping("/users/{id}")
48     public ResponseEntity<User> getUsersById(@PathVariable(value = "id") Long userId)
49         throws ResourceNotFoundException {
50         User user =
51             userRepository
52                 .findById(userId)
53                 .orElseThrow(() -> new ResourceNotFoundException("User not found on :: " + userId));
54
55         return ResponseEntity.ok().body(user);
56     }

```

```

57
58     /**
59      * Create user user.
60      *
61      * @param user the user
62      * @return the user
63      */
64     @PostMapping("/users")
65     public User createUser(@Validated @RequestBody User user) {
66         return userRepository.save(user);
67     }
68
69     /**
70      * Update user response entity.
71      *
72      * @param userId the user id
73      * @param userDetails the user details
74      * @return the response entity
75      * @throws ResourceNotFoundException the resource not found exception
76      */
77     @PutMapping("/users/{id}")
78     public ResponseEntity<User> updateUser(@PathVariable(value = "id") Long userId
79         , @Validated @RequestBody User userDetails) throws ResourceNotFoundException {
80
81         User user =
82             userRepository
83                 .findById(userId)
84                 .orElseThrow(() -> new ResourceNotFoundException("User not found on :: " + userId));
85
86         user.setEmail(userDetails.getEmail());
87         user.setLastName(userDetails.getLastName());
88         user.setFirstName(userDetails.getFirstName());
89
90         final User updatedUser = userRepository.save(user);
91
92         return ResponseEntity.ok(updatedUser);
93     }

```

```

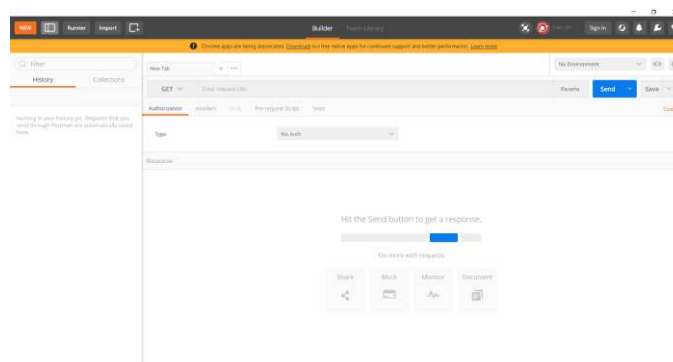
94
95  /**
96   * Delete user map.
97   *
98   * @param userId the user id
99   * @return the map
100  * @throws Exception the exception
101  */
102  @DeleteMapping("/user/{id}")
103  public Map<String, Boolean> deleteUser(@PathVariable(value = "id") Long userId) throws Exception {
104      User user =
105          userRepository
106              .findById(userId)
107              .orElseThrow(() -> new ResourceNotFoundException("User not found on :: " + userId));
108
109      userRepository.delete(user);
110      Map<String, Boolean> response = new HashMap<>();
111      response.put("deleted", Boolean.TRUE);
112
113      return response;
114  }
115 }

```

6) Build và run project

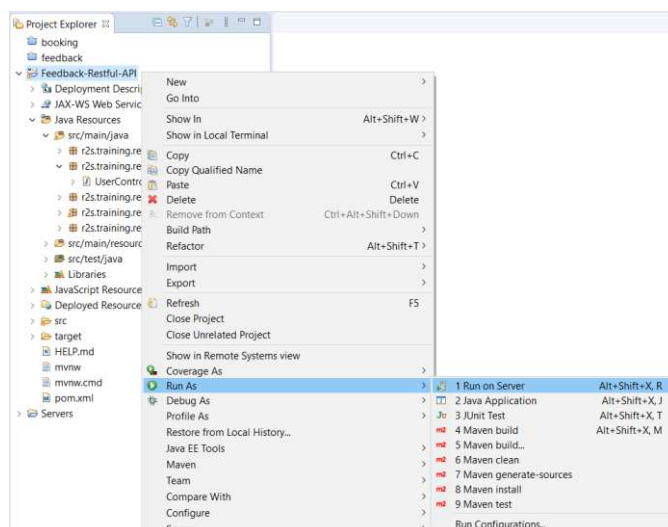
- Chuẩn bị phần mềm Postman

- Tải và cài đặt Postman: <https://www.postman.com/downloads/>
- Giao diện Postman sau khi chạy




- Build và run project

- Chuột phải project -> chọn Run As -> chọn Run on Server



- Chọn Finish


Run On Server

Select which server to use

How do you want to select the server?

☒ Choose an existing server
☐ Manually define a new server

Select the server that you want to use:

Server	State
<div> <div>localhost</div> <div> <div>Tomcat v9.0 Server at localhost</div> <div>Stopped</div> </div> </div>	

Apache Tomcat v9.0 supports J2EE 1.2, 1.3, 1.4, and Java EE 5, 6, 7, and 8 Web modules.

☐ Always use this server when running this project

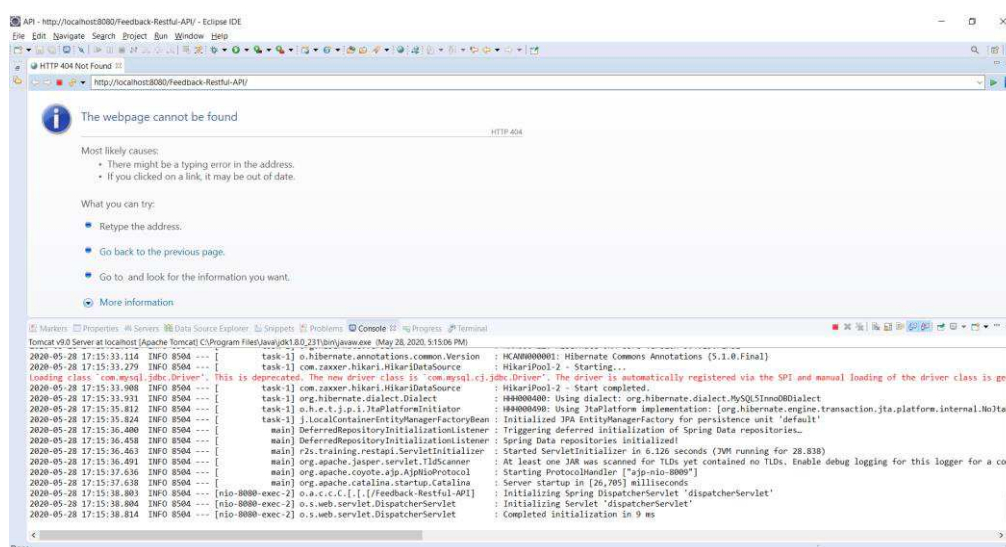
< Back

Next >

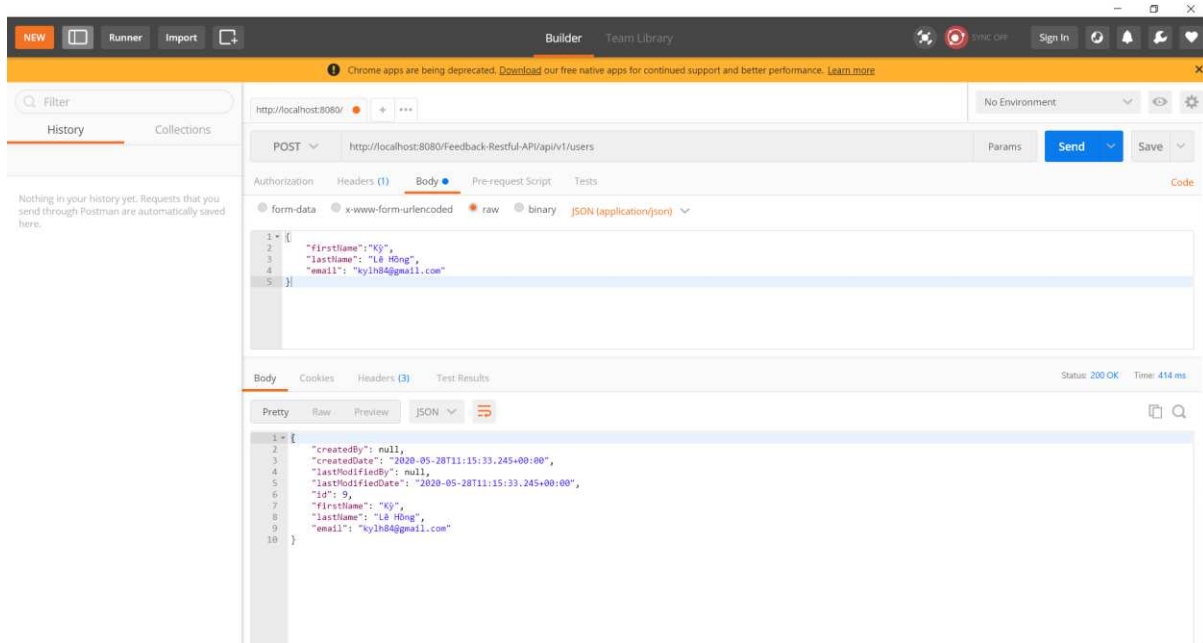
Finish

Cancel

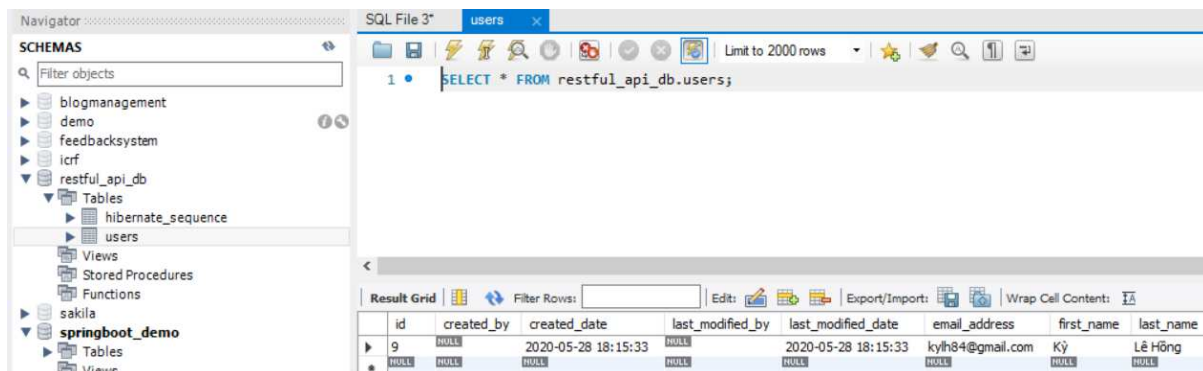
- Sau khi build và run thành công



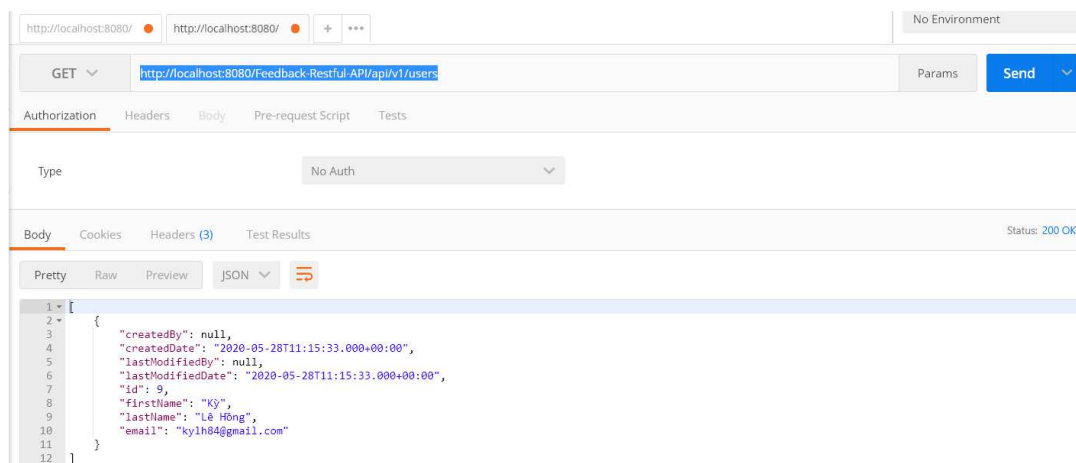
- Mở Postman
 - Thực hiện thêm mới User (Phương thức POST)
 - <http://localhost:8080/Feedback-Restful-API/api/users/users>
 - Nhập dữ liệu -> chọn Send



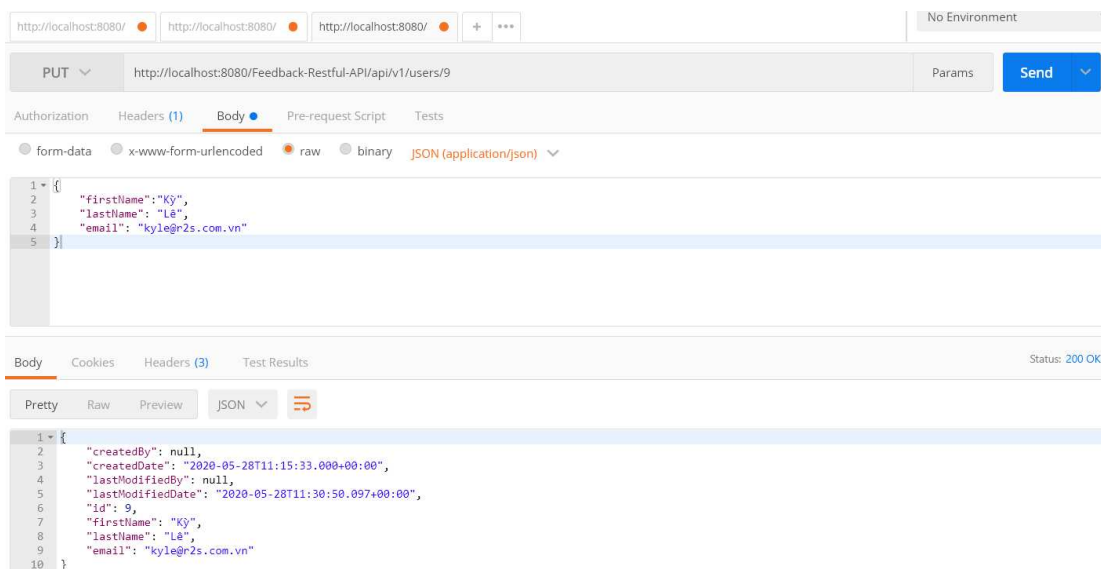
- Kiểm tra data trong MySQL



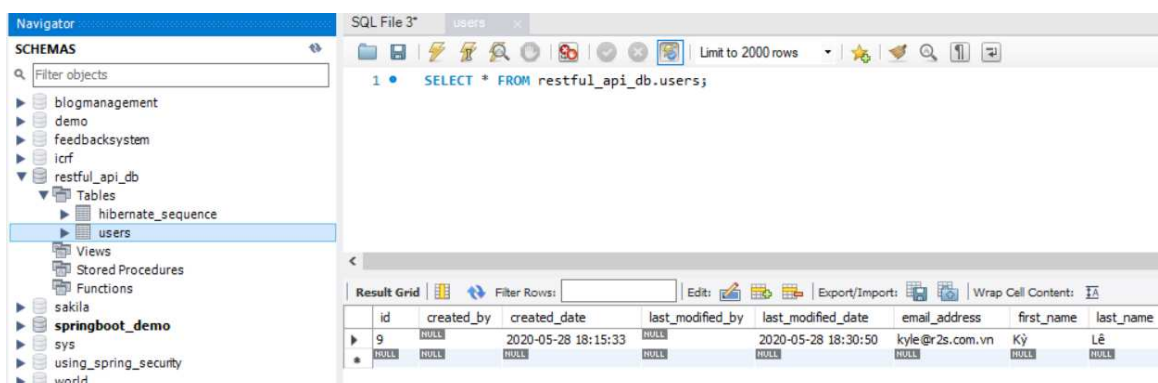
- Thực hiện lấy danh sách User (Phương thức GET)
 - <http://localhost:8080/Feedback-Restful-API/api/v1/users>
 - Chọn Send



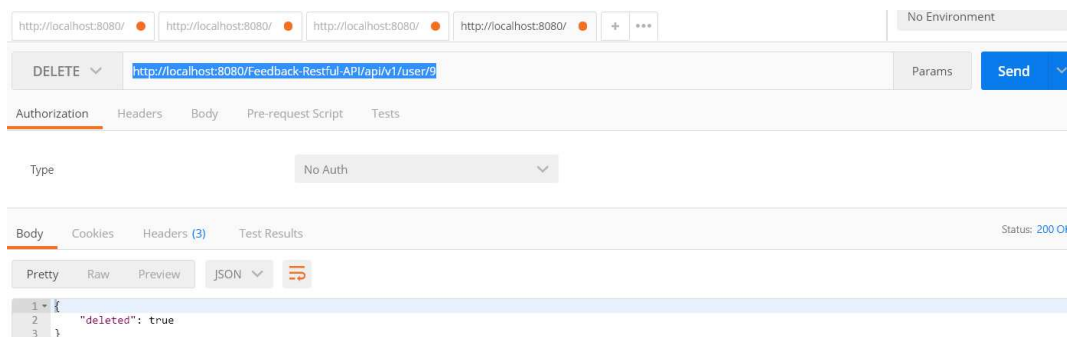
- Thực hiện cập nhật User (Phương thức PUT)
 - <http://localhost:8080/Feedback-Restful-API/api/v1/users/9>
 - Nhập thông tin muốn cập nhật -> chọn Send



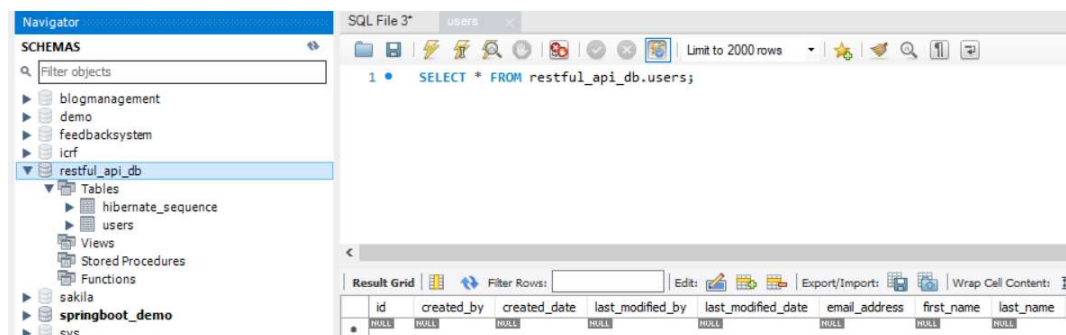
- Kiểm tra data trong MySQL



- Thực hiện xóa User (Phương thức DELETE)
 - <http://localhost:8080/Feedback-Restful-API/api/v1/user/9>
 - Chọn Send



- Kiểm tra data trong MySQL



- Tài liệu tham khảo:
 - <http://giasutinhoc.vn/website/spring-framework/huong-dan-sung-dung-spring-boot-de-cao-restful-api/>

4. Tự làm

Sử dụng project đã tạo ở mục 3 và bổ sung CRUD cho phần quản lý khóa học (Course)

Course ID	Course Name	Admin	Start Date	End Date
15	Android programming	binhnt44	13/02/2020	27/02/2020
16	Java programming	danpl44	13/02/2020	29/02/2020
27	Requirement and Design (RnD)	-	20/02/2020	21/02/2020