

**BÁO CÁO**

**Môn : JAVA 6**

**Nội dung : Lab3 - Lab4, Quiz3 - Quiz4**

Giảng viên hướng dẫn : LÊ ANH TÚ

Nhóm : 3

PS20520 \_ NGUYỄN THANH HÀ

PS20125 \_ PHẠM THỊ CẨM VY

PS21421 \_ VŨ HUỲNH ĐĂNG KHÔI

PS21431 \_ THI NGỌC QUÍ

MENU

[1. Mục tiêu bài lab 2](#_Toc24625)

[1.1. Lab1 2](#_Toc28360)

[1.2. Lab2 3](#_Toc14511)

[2. Quiz1: 3](#_Toc21474)

[3. Silde1: 9](#_Toc18884)

[3.3. Tại sao sử dụng biểu thức Lambda 9](#_Toc15130)

[Json Concepts 10](#_Toc12753)

[4. Lab2 :  12](#_Toc25204)

[4.4. -Quiz 2: 12](#_Toc29543)

[4.5. Silde2 16](#_Toc29800)

# Mục tiêu bài lab

## Lab3

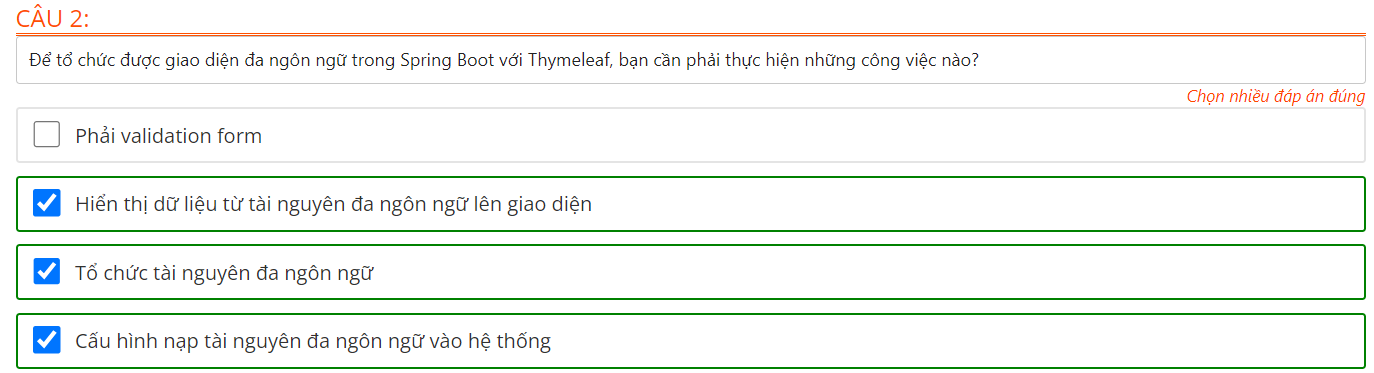
* Ràng buộc dữ liệu và kiểm lỗi form 
* Tổ chức layout
* Tổ chức đa ngôn ngữ

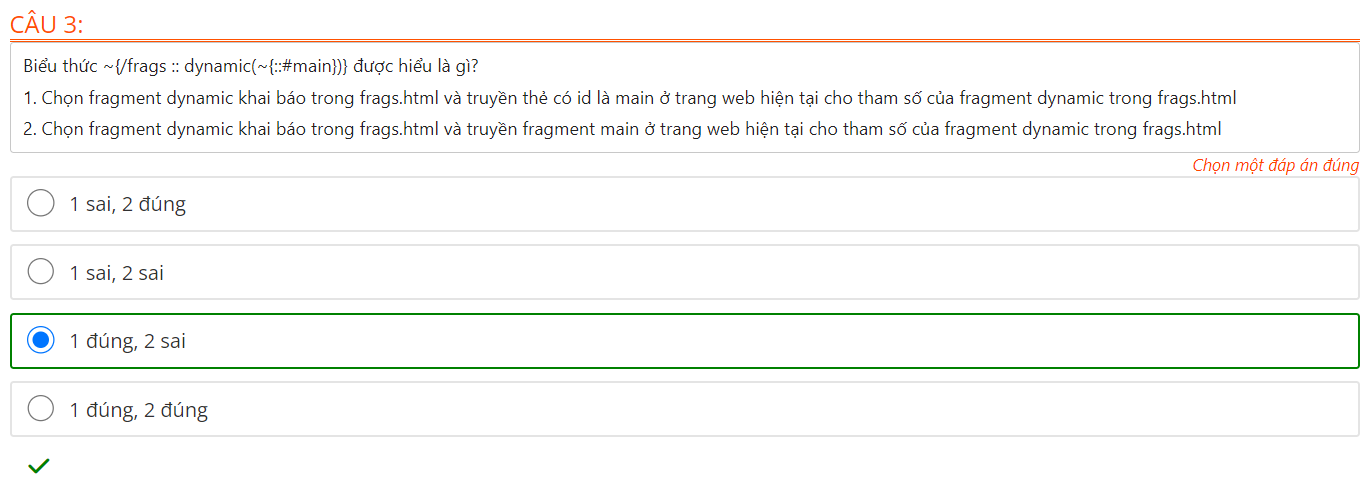
## Lab3

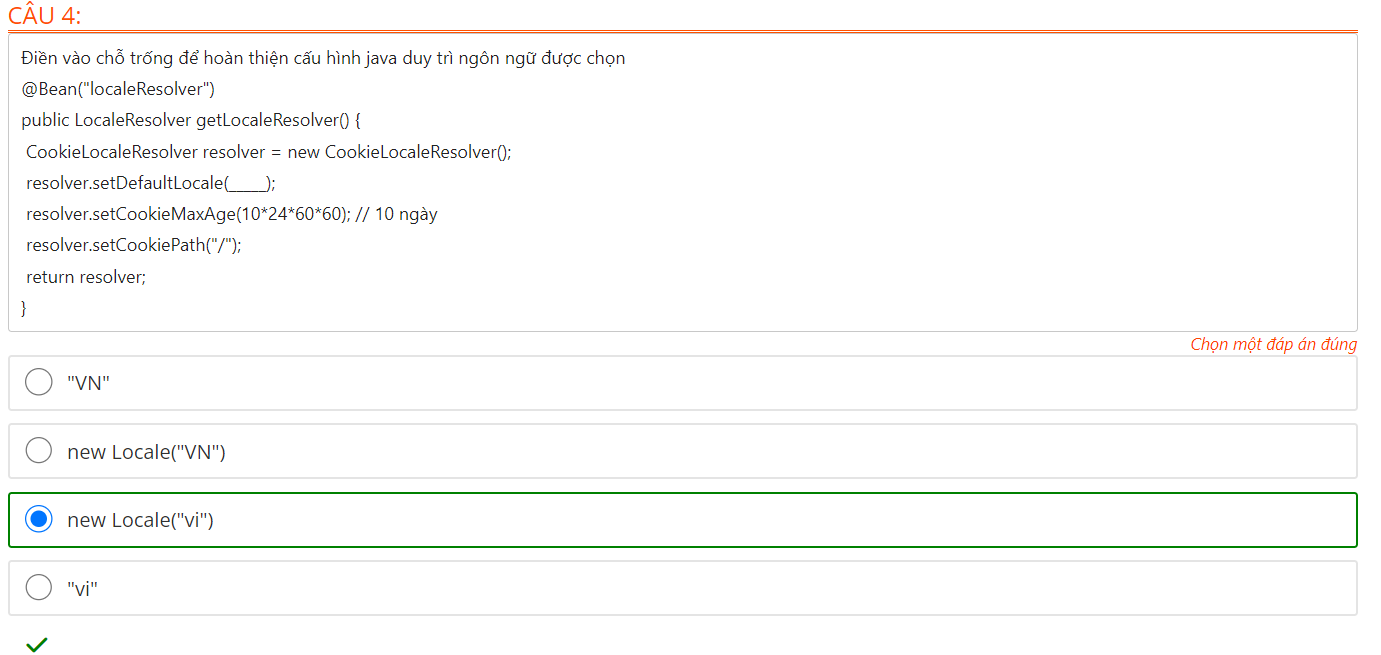
*  Khai thác RESTful API của Firebase
*  Sử dụng Firebase RESTful API bằng các thể loại client
* AngularJS
* Java Desktop
* Spring Boot RestTemplate

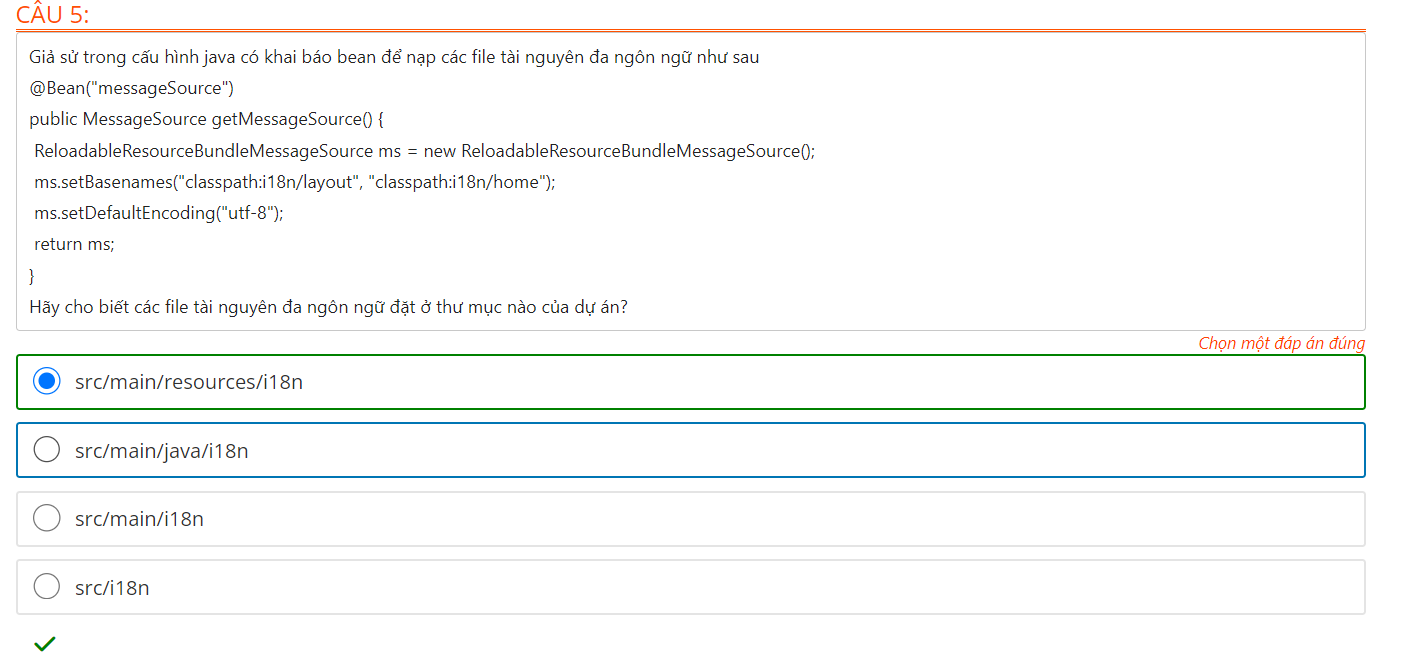
# Quiz3:

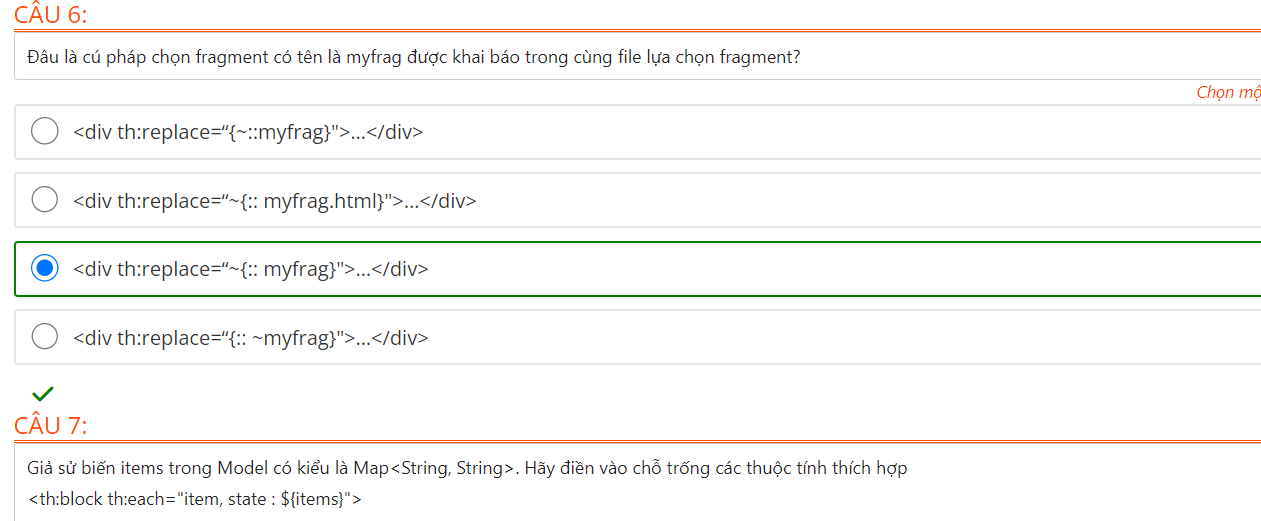




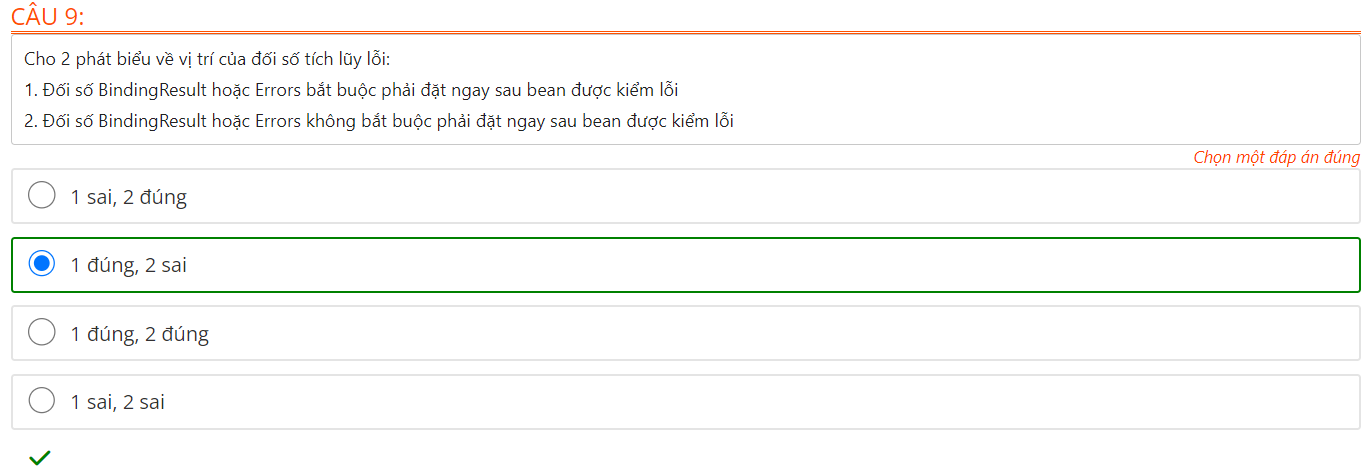


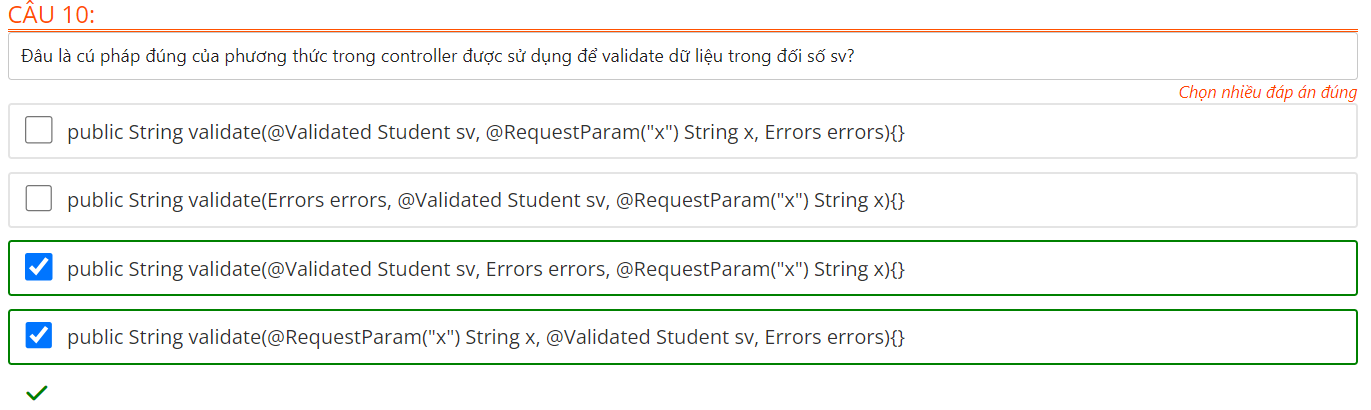


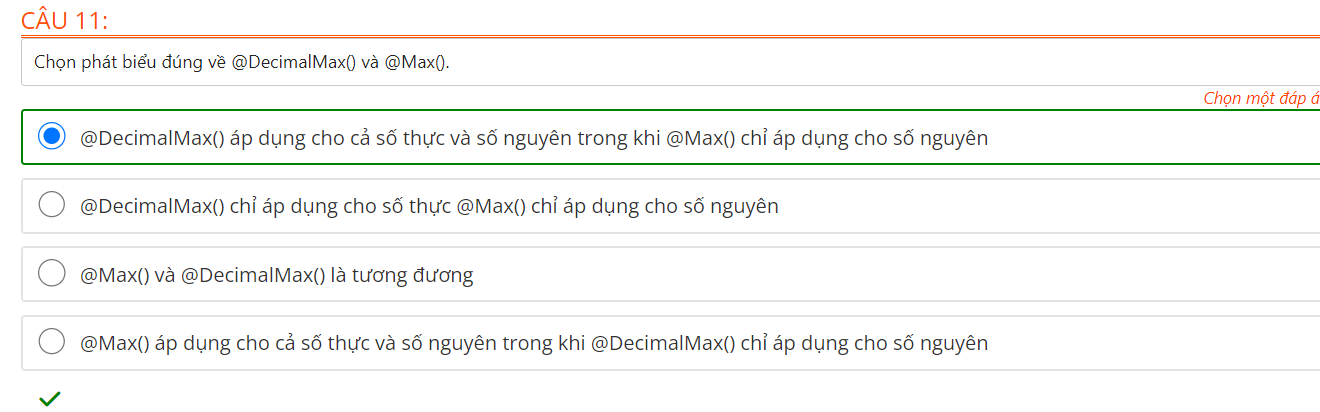


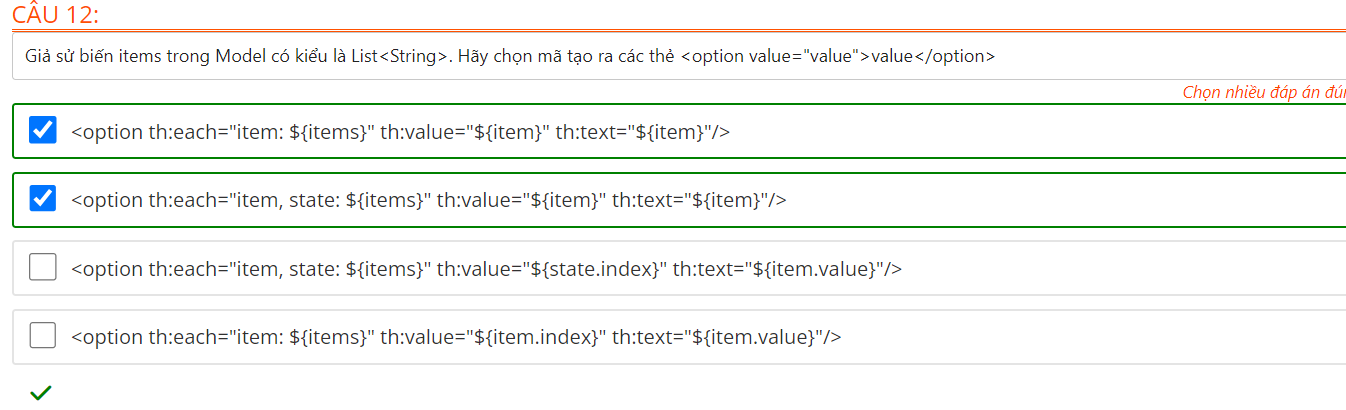


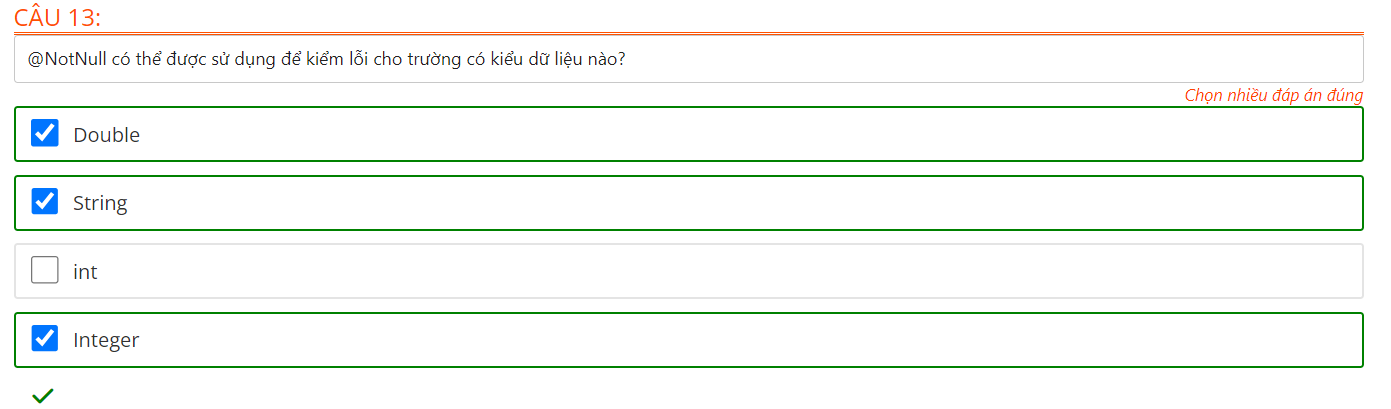


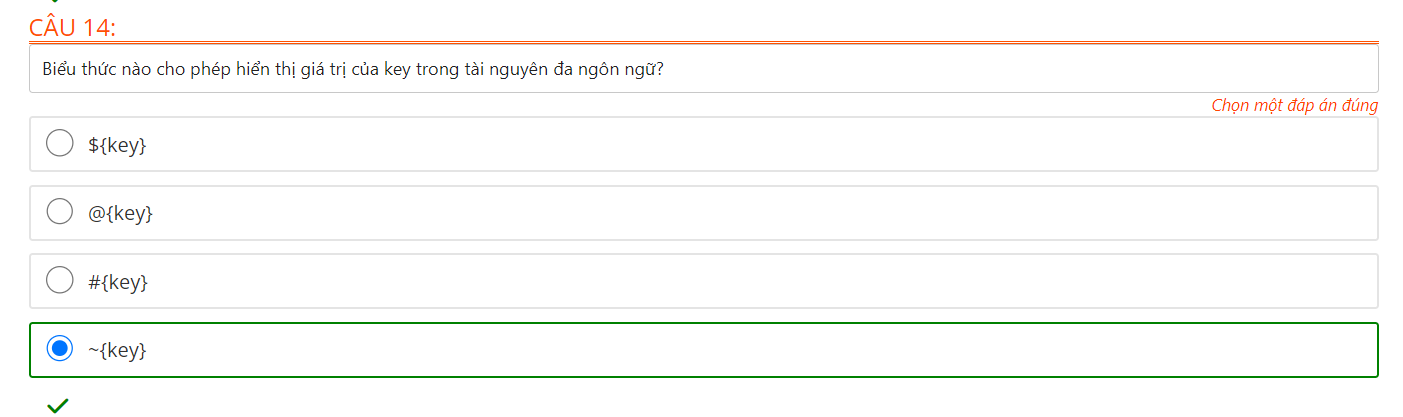


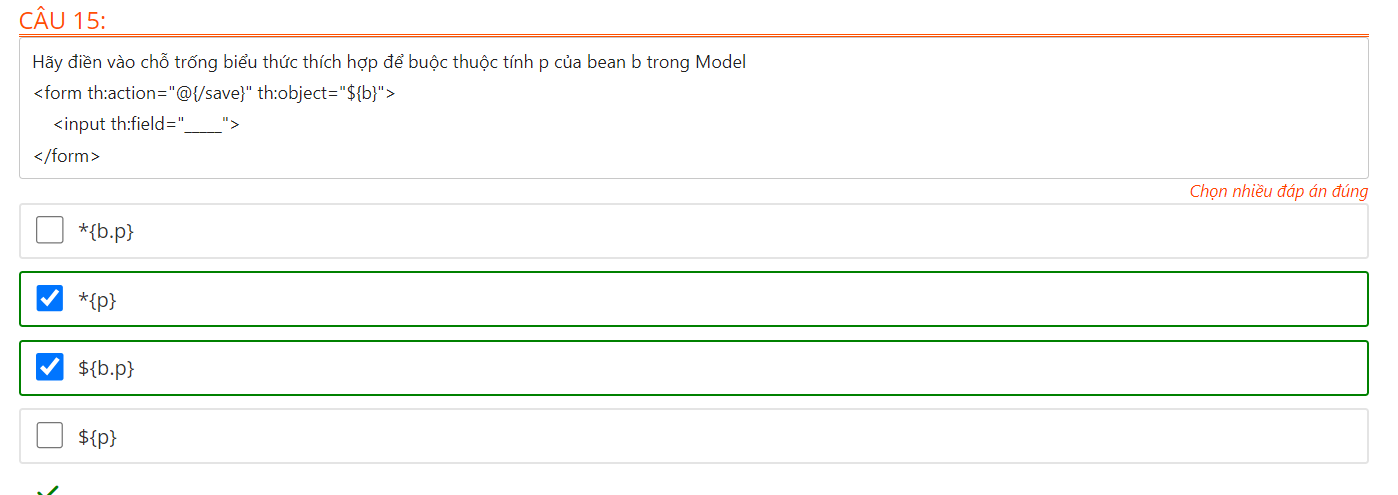












# Silde3:

## Data Binding and Validation trong thymeleaf có gì giống và khác so với trong jsp

Giống:

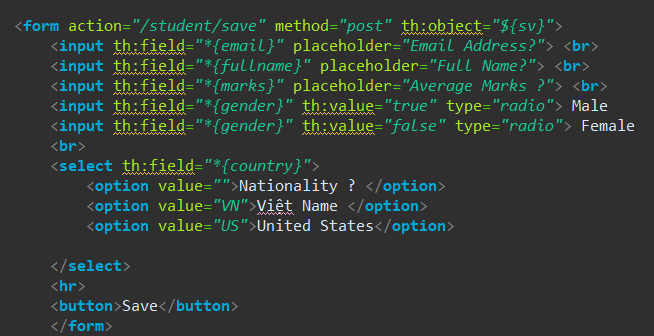
Cách tổ chức và quy trình thực hiện giống nhau

Các hàm được dụng giống và có chức năng tương tự

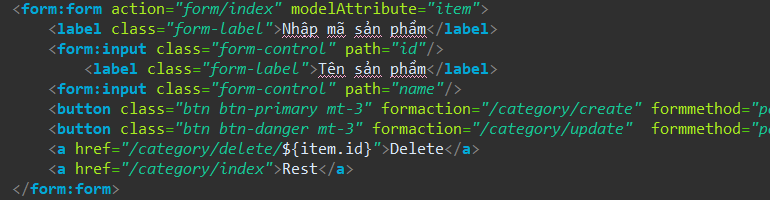
Cách viết code và xử lý back end giống nhau

Khác

Ở thymeleaf template



Ở JSP cách viết



=>> Đối với template ta sử dụng th:field=”\*{}” , còn đối với jsp ta ghi <form:input và đặt path có tên là trường đối tượng.

Đối với Validation thì giống hoàn toàn .

Các anomation của validation điển hình thường sử dụng

+ NotBlank : Sử dụng cho việc bắt chuỗi rỗng

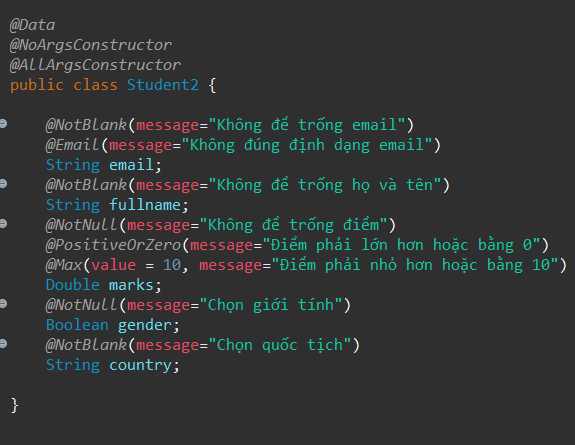
+ Email : Bắt định dạng email;

+NotNull : Bắt lỗi chuỗi hoặc 1 đối tượng rỗng

+PositionOrZero : Bắt lỗi vị trí hoặc vị trí 0

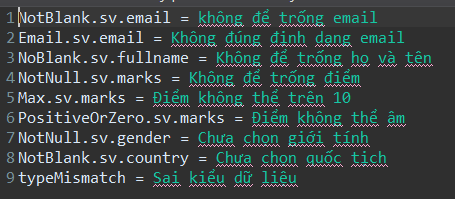
+Max : Bắt lỗi số có gá trị vượt mức max

Cách viết tại đối tượng class



Cách viết trên file.properties

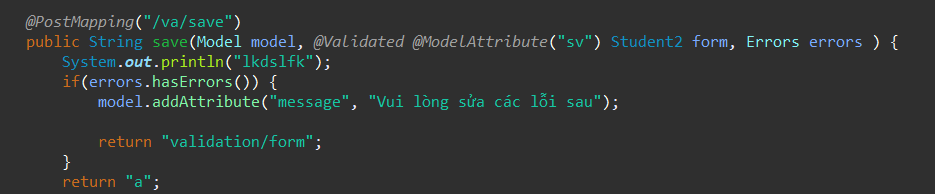
+ Cấu trúc Annotation.Bean.Property



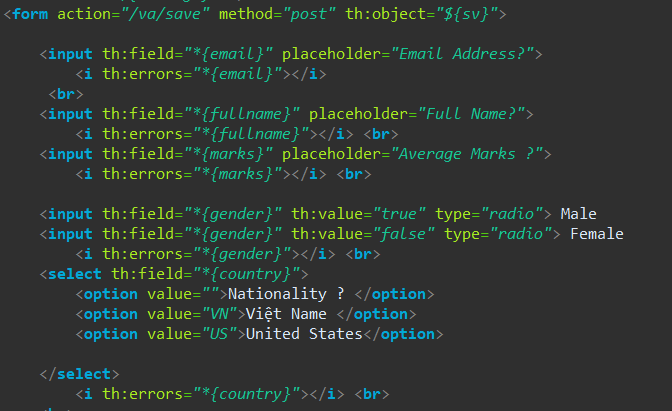
Cách thực hiện code trong controller

+ Tại hàm cần thực hiện kiểm tra ta đặt @Validated vào trước @ModelAttribute ta muốn kiểm tra

+ Muốn kiểm tra có lỗi hay không ta truyền tham số Errors errors

  
  
- Cách viết tại thymeleaf templates

+ Ta sử dụng th:errors = “\*{tên trường}” để xuất ra message của lỗi đó



## Layout Organization

Ý nghĩa sử dụng : Dùng để chèn, thay thế 1 template này vào 1 template khác

Các thymeleaf dùng :

+ th:fragment=”tên frag” : Dùng để định nghĩa template sẽ thay thế / chèn vào template khác

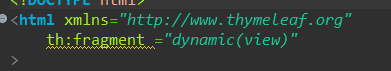
+th:replace=”~{file::tên frag của fragment}” : Cho biết khu vực sẽ bị thay thế bởi th:fragment

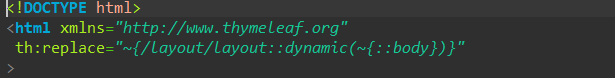
+th:insert=”~{file::tên frag của fragment}” : Cho biết vị trí mà template fragment sẽ được chèn vào

+ Nếu frag là 1 thẻ/ scope thì ta sẽ ghi : th:replace=”~{::tên thẻ}”

+Nếu các fragment có tham số th:fragment=”frag(x)”

=>> Tại th:replace=”~{file::frag(~{x})}” , x ở đây có thể là 1 dữ liệu , 1 thẻ , 1 scope.



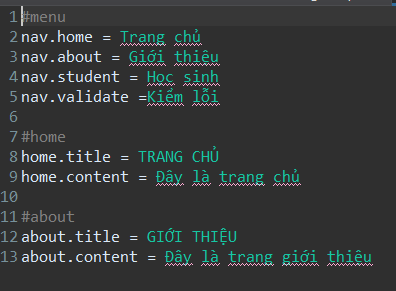
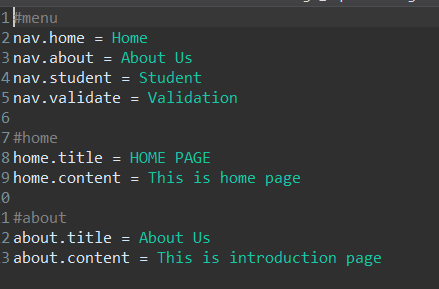
==>>

## Internationalzation

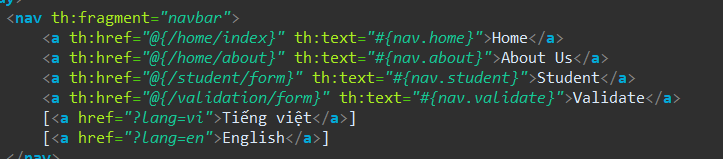
=> Nó dùng để triển khai cấu trức dữ liệu đa ngôn ngữ. Cho phép người dùng có thể tìu chỉnh ngôn ngữ.

=> Cách chuyển đổi

+ Chuẩn bị 2 file properties theo 2 ngôn ngữ muốn dùng :

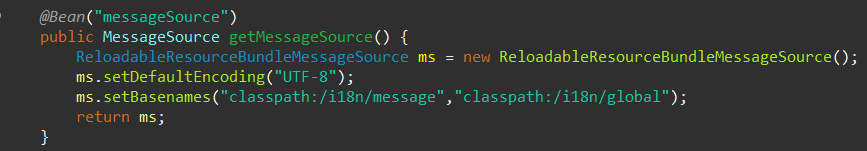


+ Ta thực hiện đưa vào templates



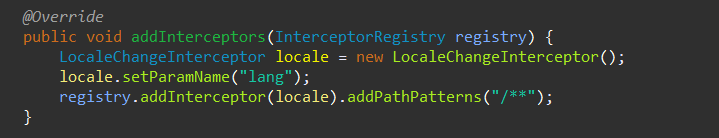
+ Xử lí cấu hình để nó tự động cập nhật

1.Khai báo cấu hình

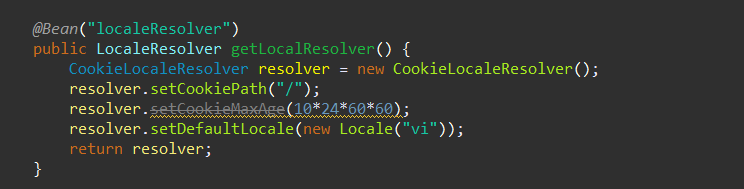


1. Áp dụng các đường dẫn

=> Ta chỉ ra tham số sẽ quyết định thay đổi ngôn ngữ. Và kiểm tra nó trên tất cả các đường dẫn

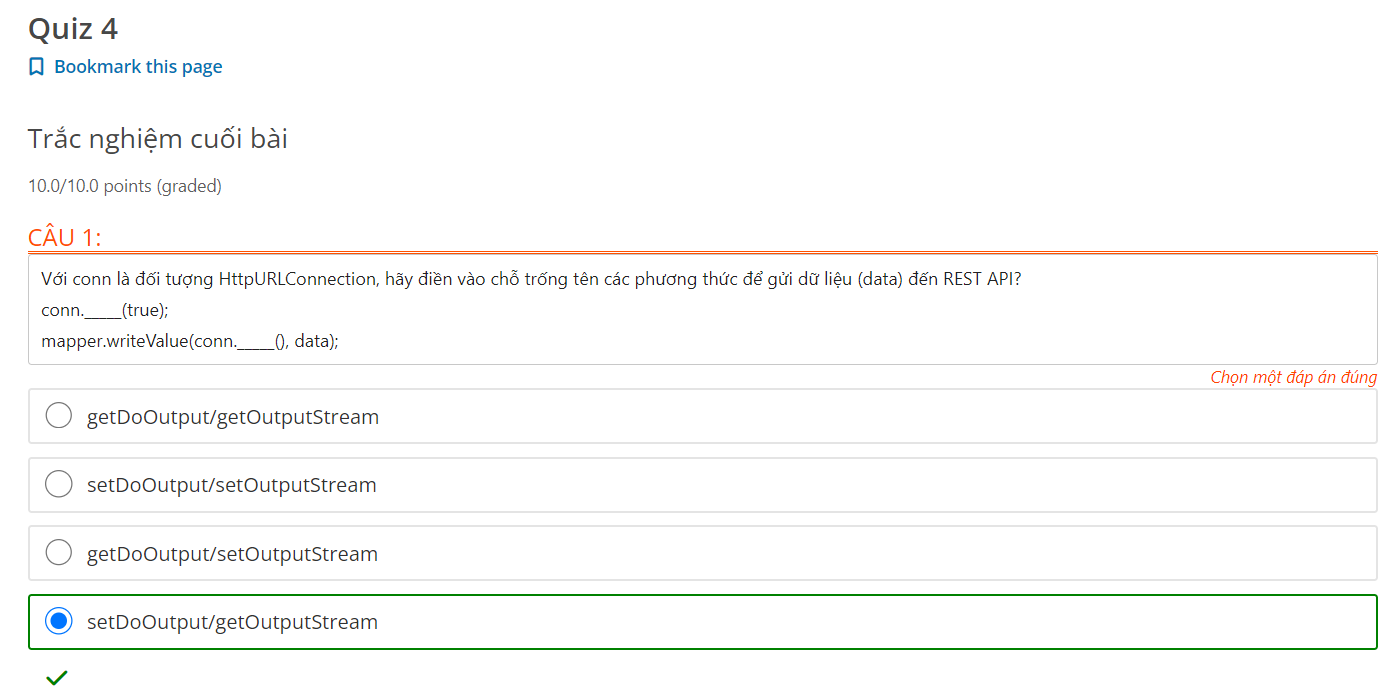


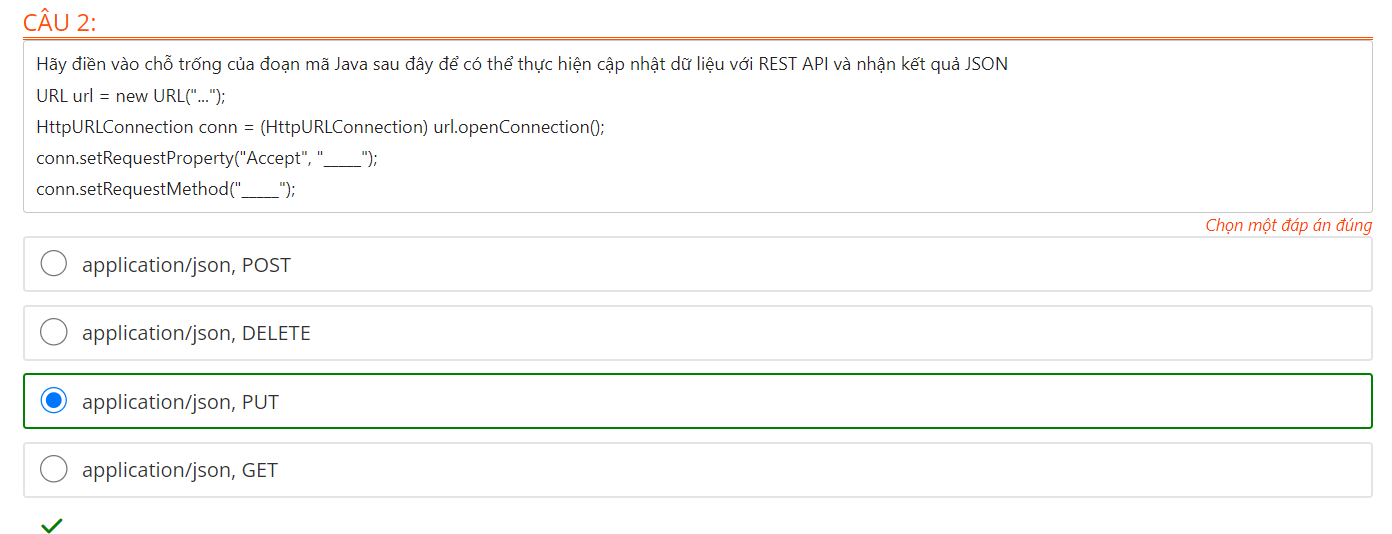
1. Thực hiện ghi nhớ vào Cookie.

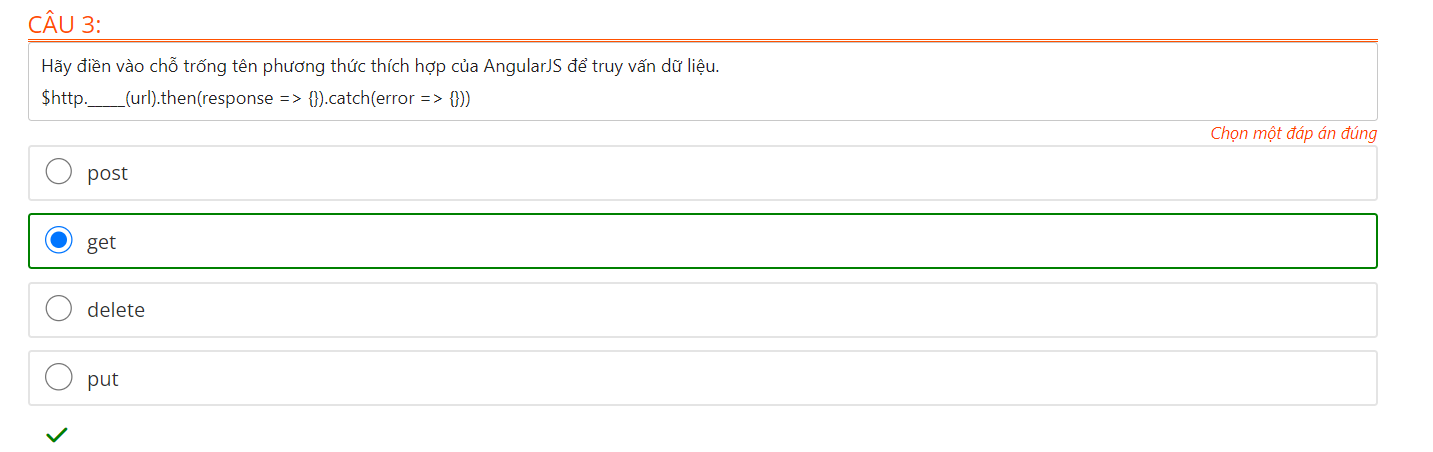


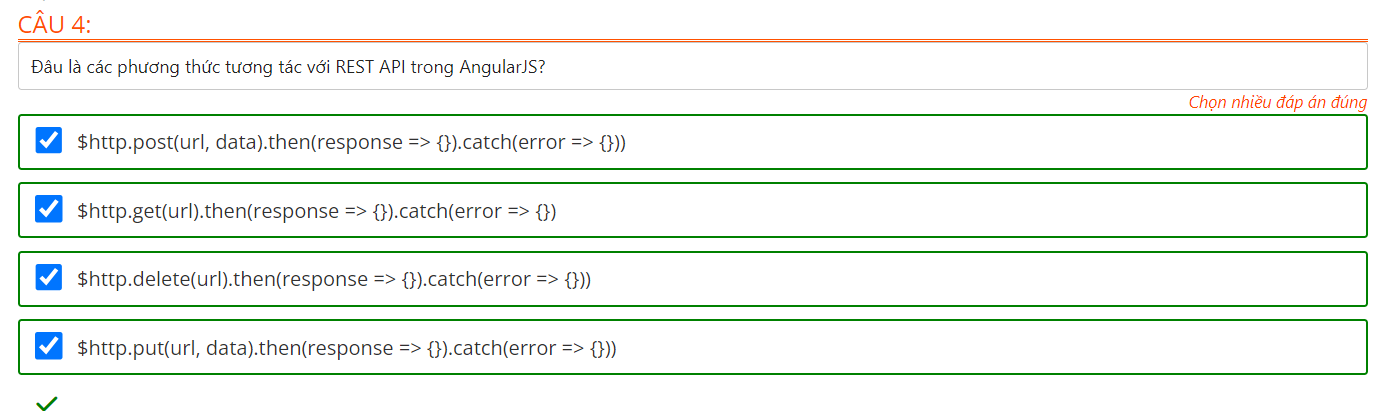
# Quiz4 :

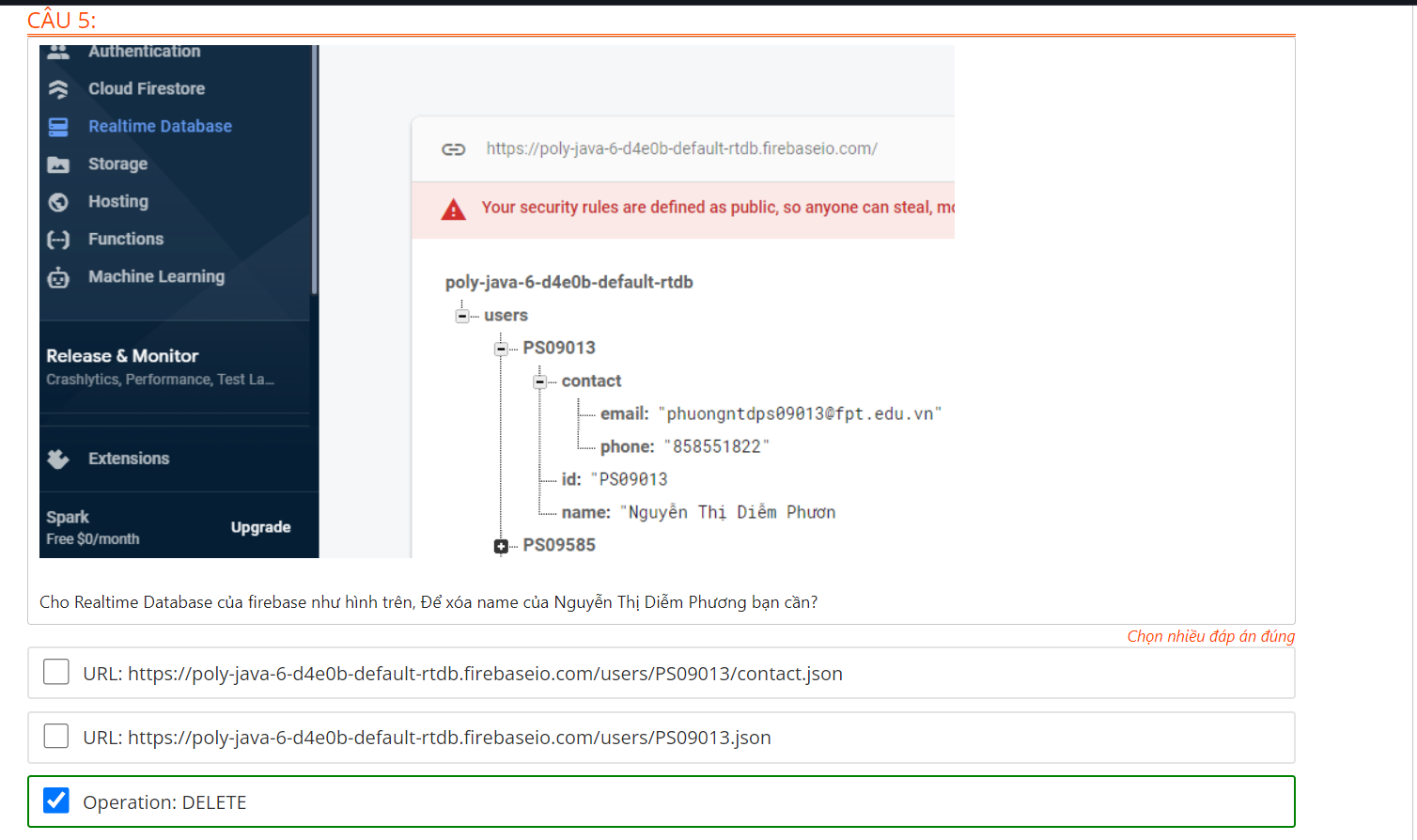
Quiz4

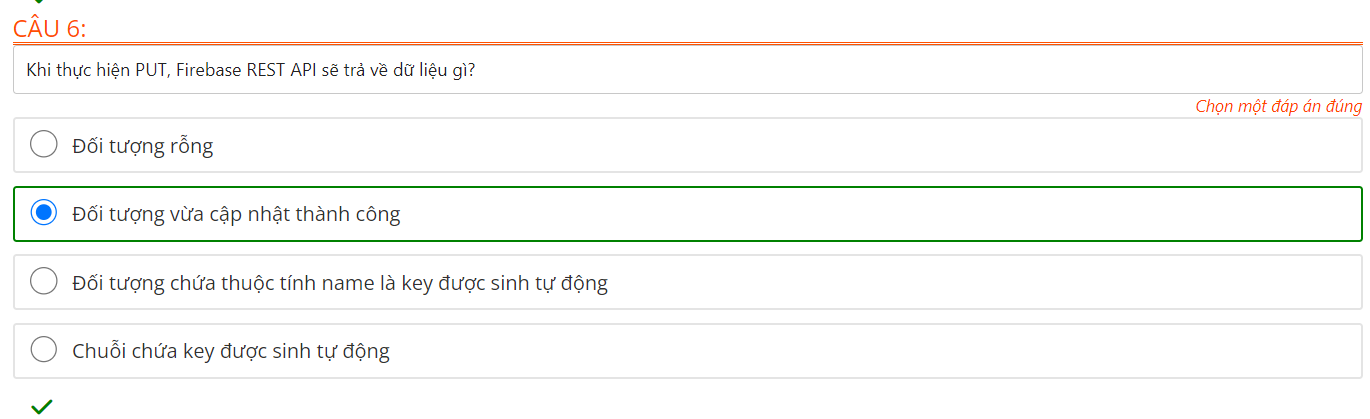


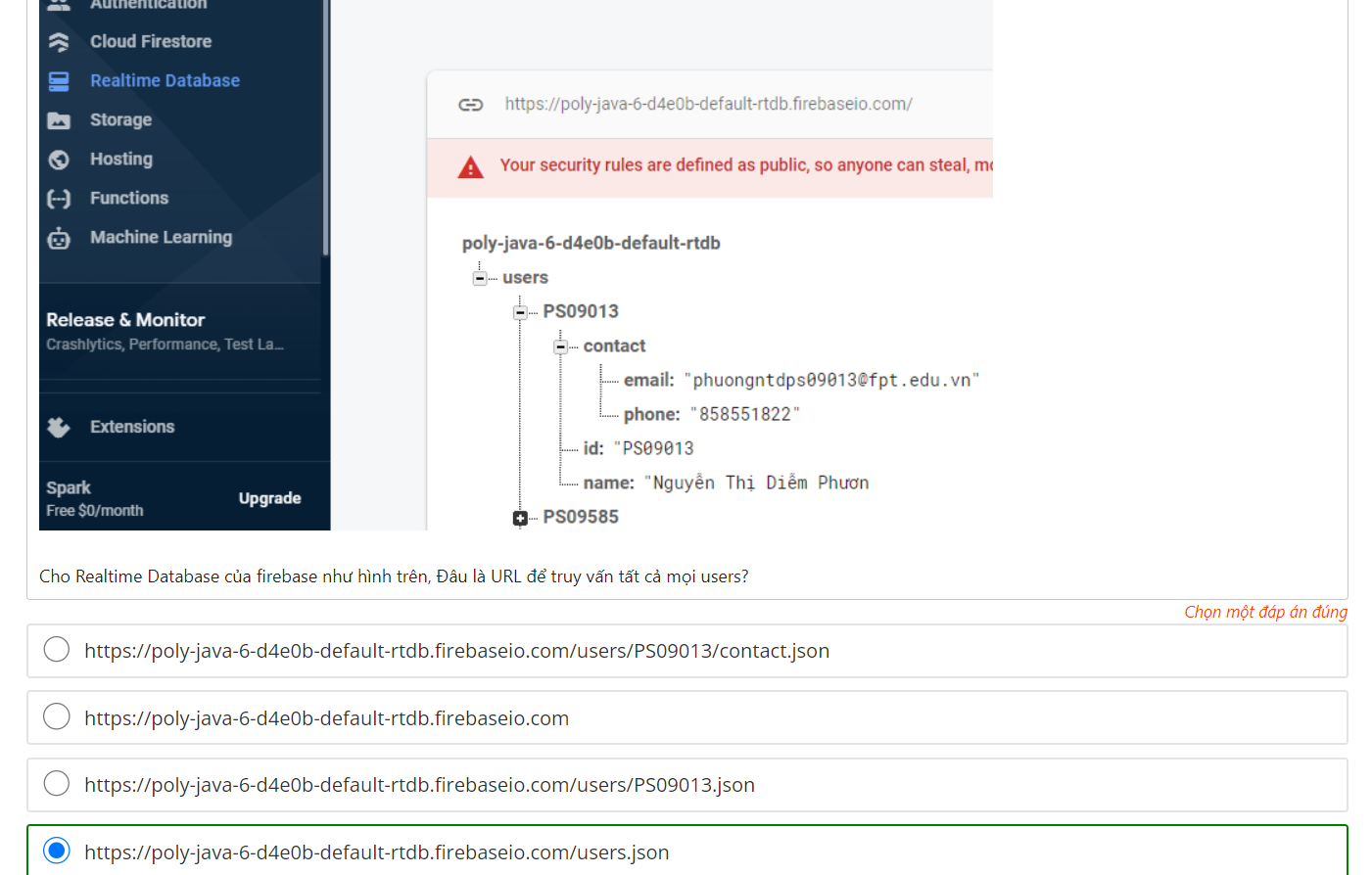


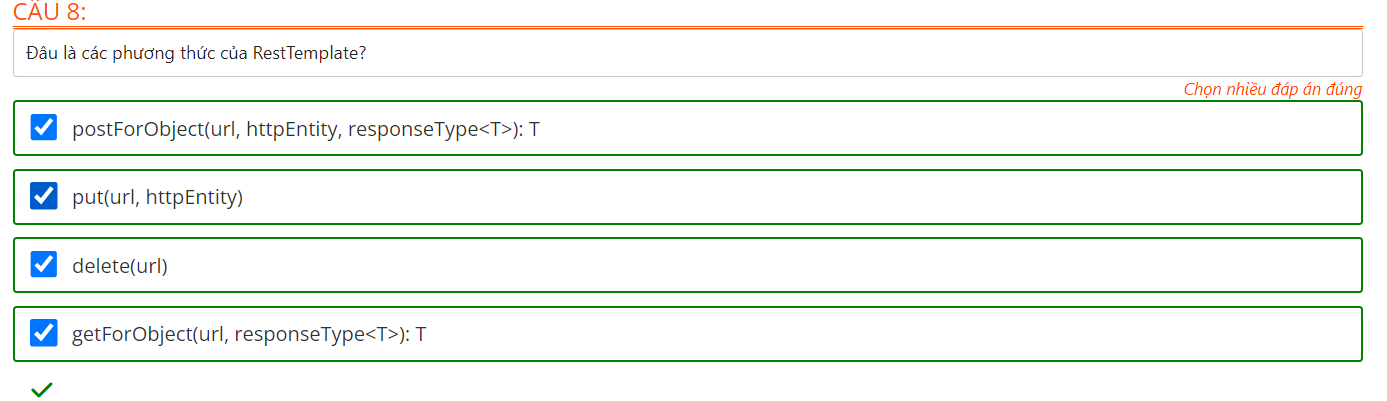


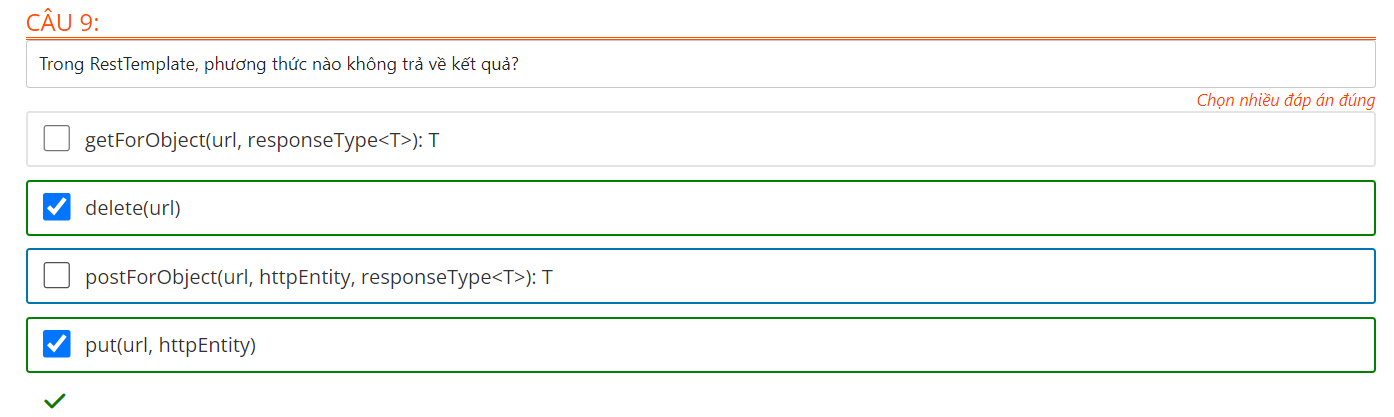




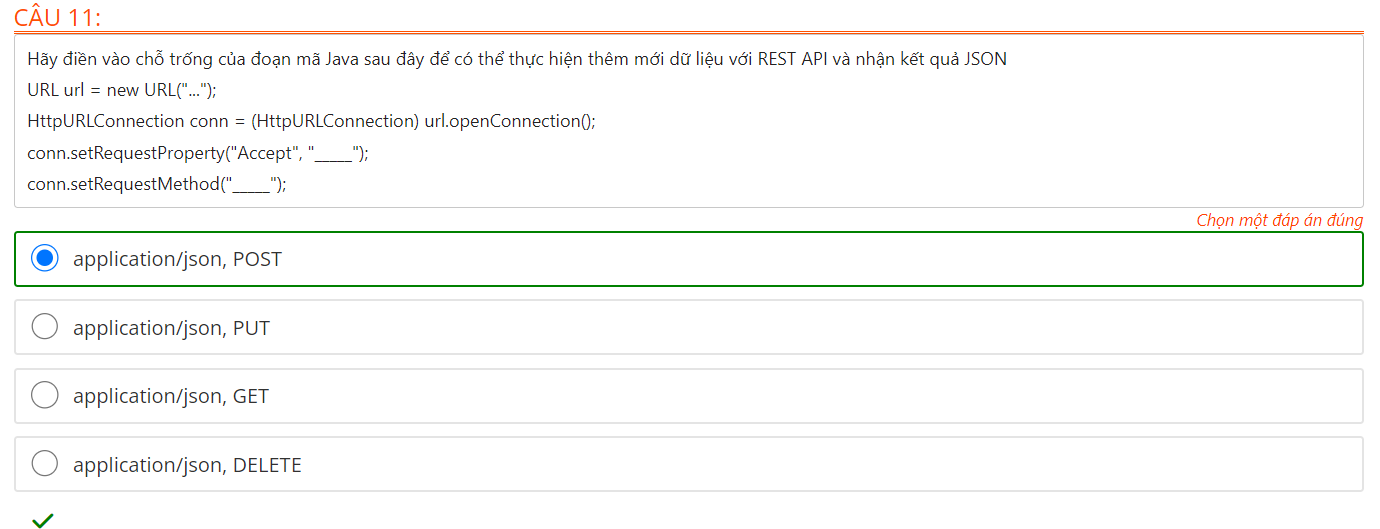


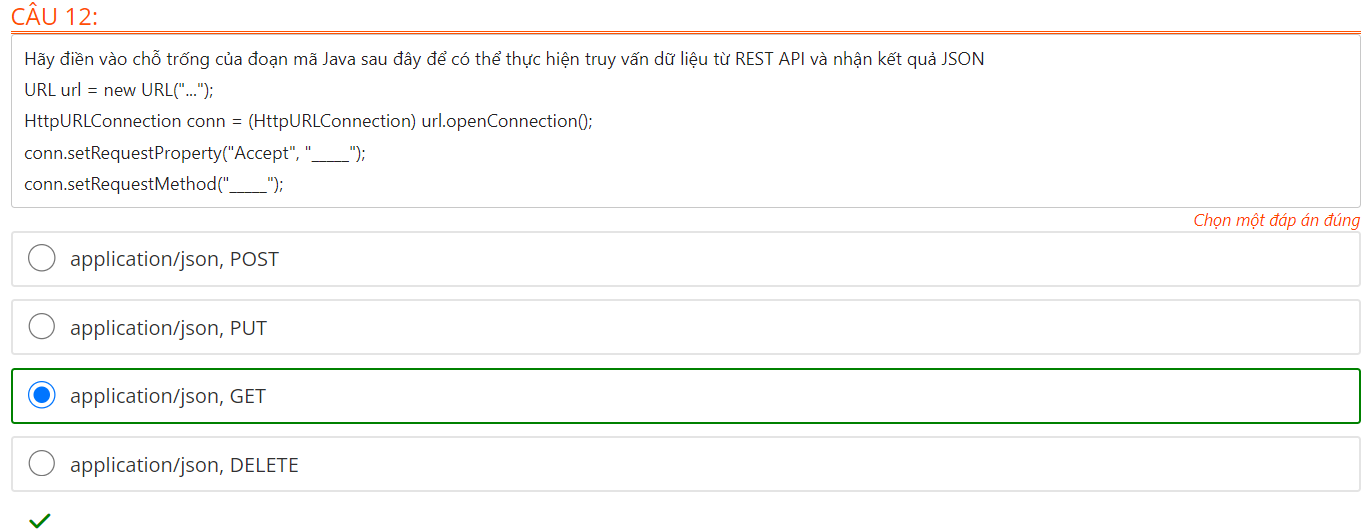


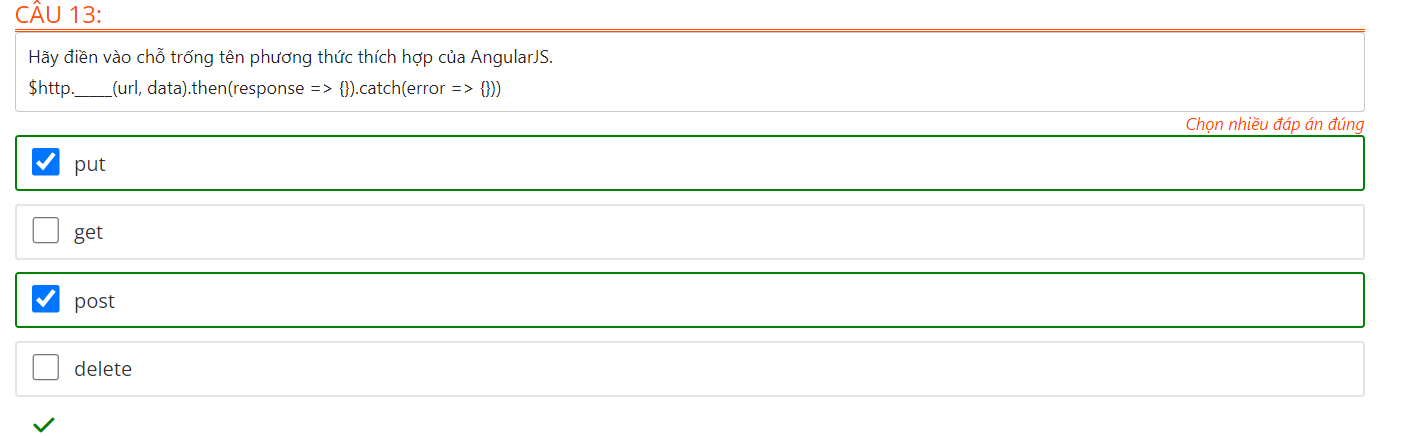




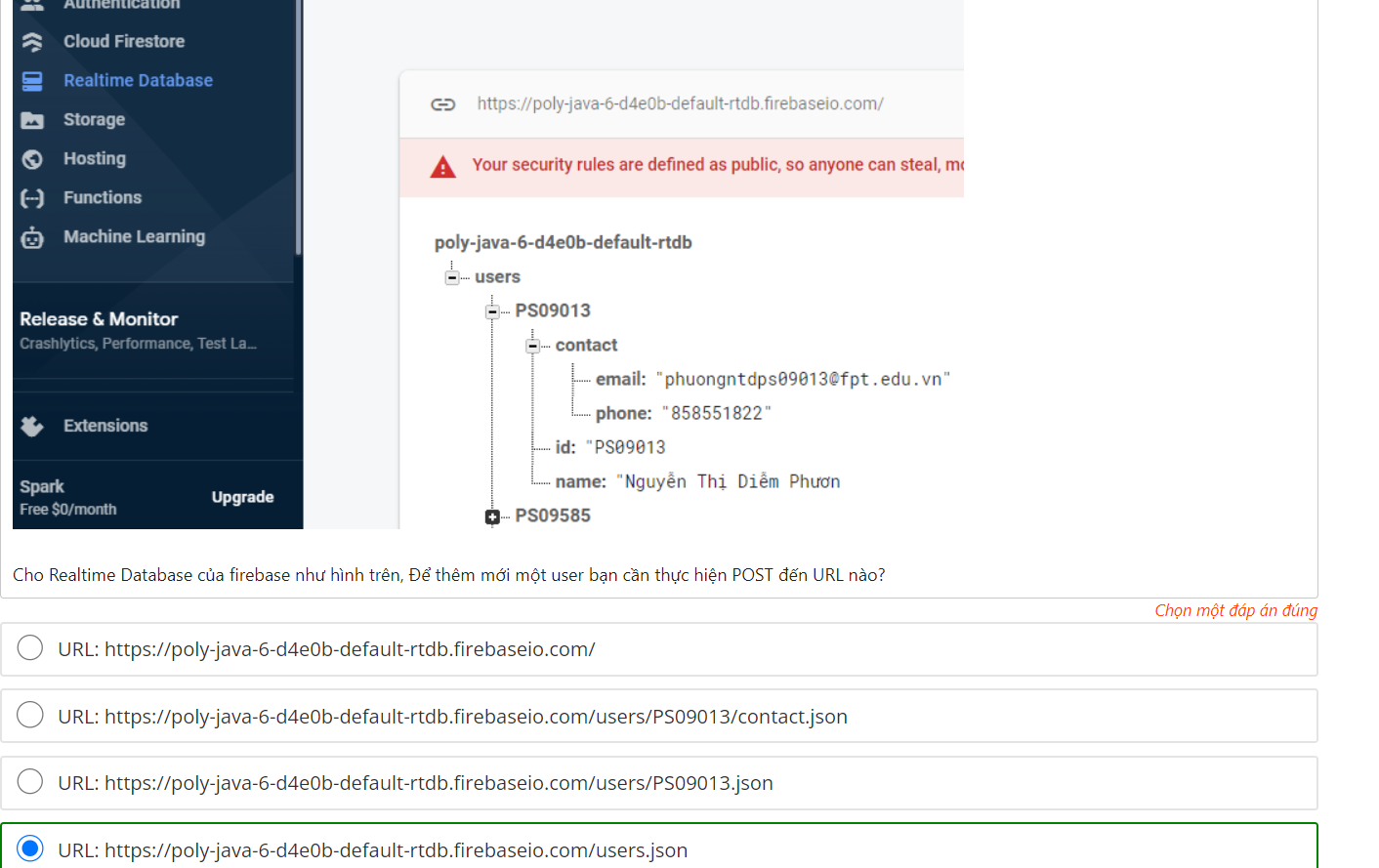




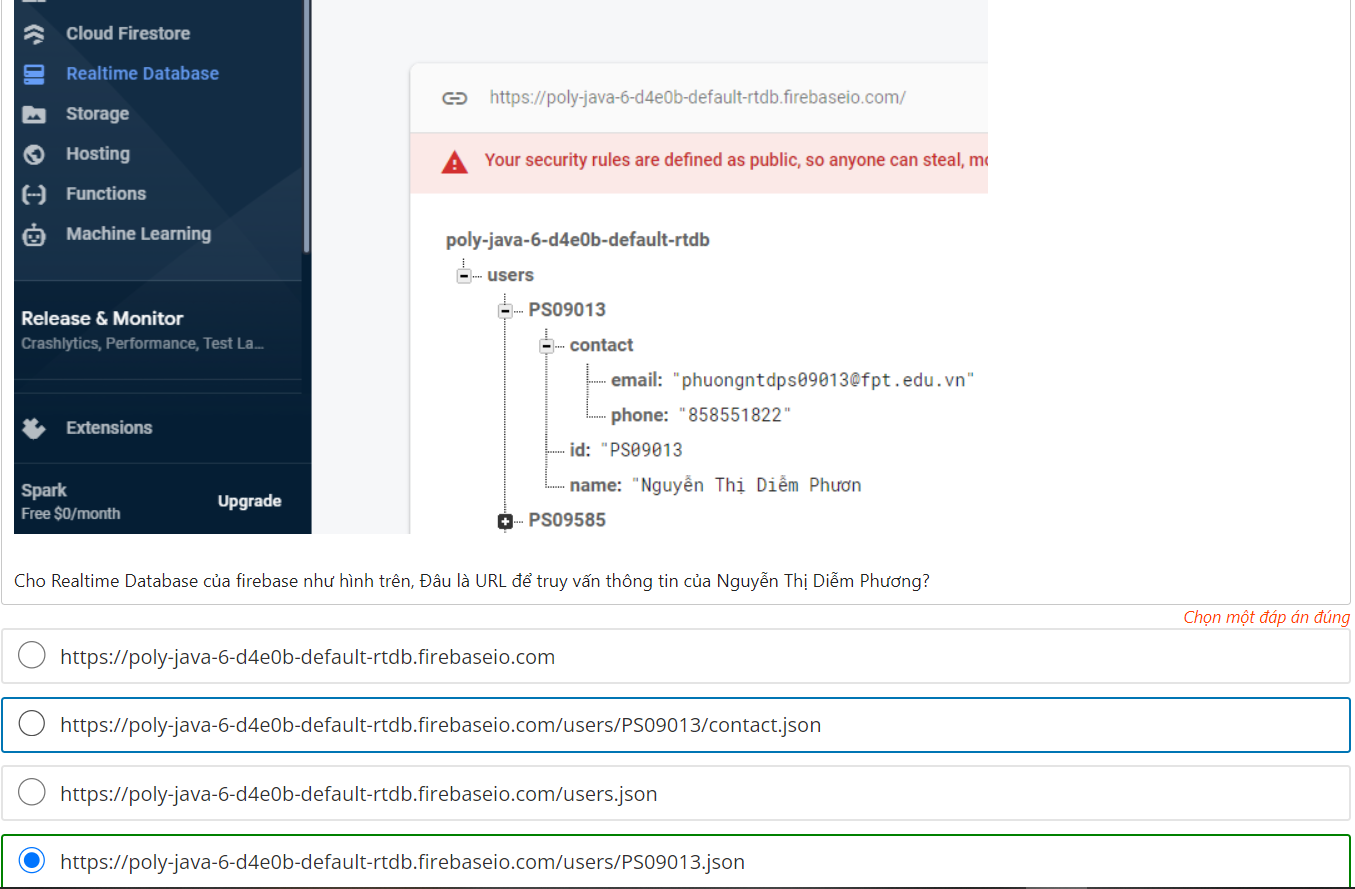




Câu14



Câu15



# Silde4

Rest là gì : Là các quy ước biểu diễn dữ liệu chuyển đổi giữa các ứng dụng

Rest API (RestFul) là web service hoạt động theo các tiêu chuẩn

Operation Tranfar Data

+Get: Lấy dữ liệu +Json : Lưu trữ dữ liệu

+Post: Gửi/ lưu dữ liệu + Xml

+Put: thay thế dữ liệu

+Patch: Cập nhật dữ liệu

+ Delete : Xoá dữ liệu

FireBase RealTime Data :  là một cơ sở dữ liệu NoSQL được lưu trữ đám mây cho phép bạn lưu trữ và đồng bộ dữ liệu. Dữ liệu được lưu trữ dưới dạng JSON và được đồng bộ hóa theo thời gian thực cho mọi máy kết nối.

Postman là gì : một phần mềm hỗ trợ phát triển, kiểm tra và chỉnh sửa [API](https://wiki.tino.org/api-la-gi/" \t "https://wiki.tino.org/postman-la-gi/_blank).

Các chức năng của Postman

* Hỗ trợ gọi [HTTP](https://wiki.tino.org/request-va-response-trong-lap-trinh-web/" \t "https://wiki.tino.org/postman-la-gi/_blank)Request với các method GET, POST, PUT, PATCH và DELETE.
* Hỗ trợ uỷ quyền authorization Oauth1, Oauth2…
* Trả về các kết quả dạng hình ảnh, XML, text, JSON.
* Hỗ trợ POST dữ liệu dưới dạng key – value, JSON, text.
* Có thể sửa đổi Header của Request.

Vì sao nên sử dụng Postman?

Thông thường, công việc phát triển một API sẽ vô cùng phức tạp, tốn nhiều thời gian, tiền bạc và công sức. Nhưng với Postman, bạn sẽ được hỗ trợ tối đa cho việc phát triển, kiểm thử API như:

* Hỗ trợ tất cả các HTTP Method
* Khả năng chuyển đổi từ API sang mã
* Hỗ trợ thay đổi Environment phát triển API
* Bạn có thể tái sử dụng các dữ liệu và không phải bắt đầu lại từ đầu
* Hỗ trợ kiểm thử API mà không cần code
* Giao diện đồ họa trực quan và dễ dàng sử dụng
* Hỗ trợ lên lịch kiểm tra API một cách tự động

### AngularJs $Http Service API

-Sử dụng trên giao diện web , Http, AngularJS,…

Có 4 hoạt động cơ bản:

+ get().then().catch() : Cho phép đưa dữ liệu về

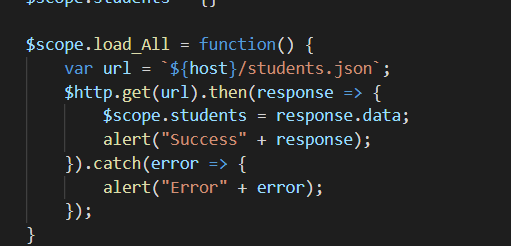
+ post().then().catch() : Cho phép tải dữ liệu lên json

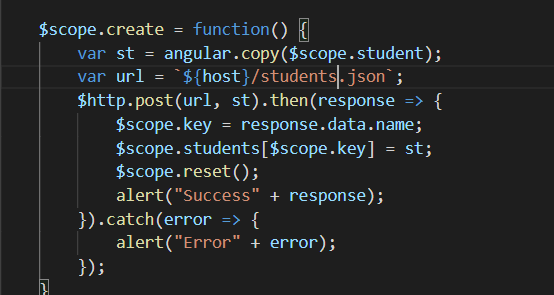
+ put().then().catch() : Cho phép thay thế dữ liệu

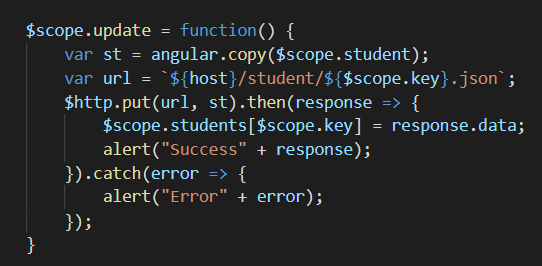
+ delete().then().catch() : Cho phép xoá dữ liệu

-Rest API Operations trong AngularJs $Http

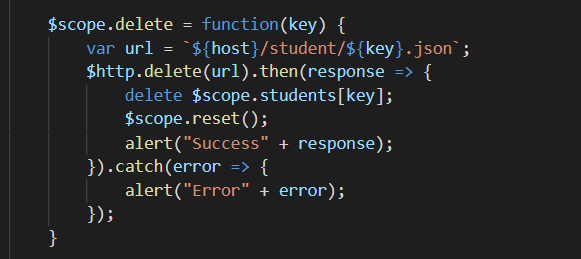
* $http.get(url).then(response => {}).catch(error => {})



* $http.post(url, data).then(response => {}).catch(error => {}))
* 
* $http.put(url, data).then(response => {}).catch(error => {}))



* $http.delete(url).then(response => {}).catch(error => {}))



Lưu ý : url là đường dẫn dẫn đến firebase chứa dữ liệu json

Response yêu cầu thực hiện

catch(error=>{}) Cho phép bắt lỗi ngoại lệ

### Java.net.URL

- Sử dụng trên giao diện java swing

Sử dụng ObjectMapper và JsonNode

ObjectMapper JsonNode

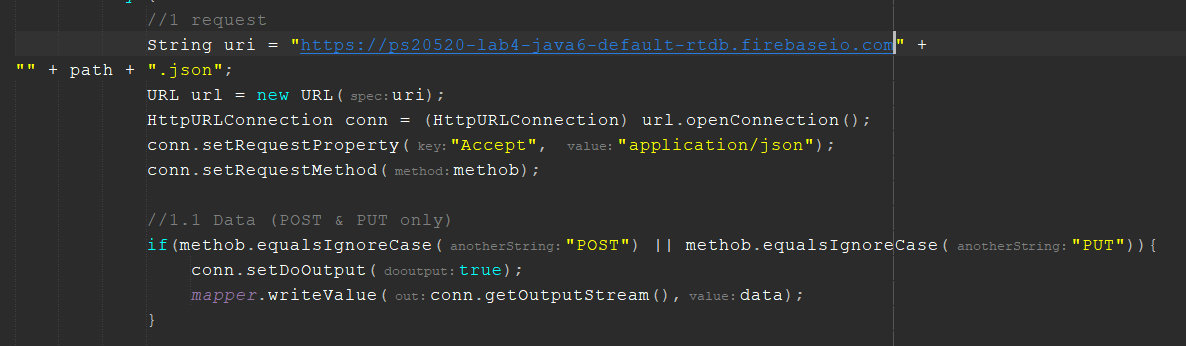
+readValue() +get()

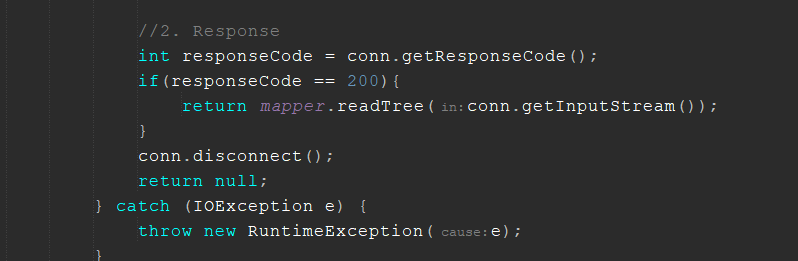
+readTree() +findvalue()

+writeValue() +asType()

+writeValueAsString +iterator

+createObjectMapper

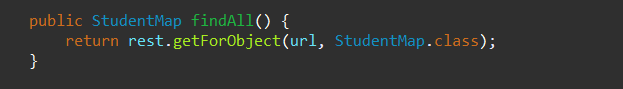




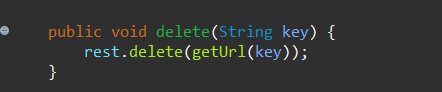
### String Resttemplates

Sử dụng trên giao diện Spring Boot

Rest Template sử dụng các Rest Operations

+ getForObject(url,reponseType<T>); T 

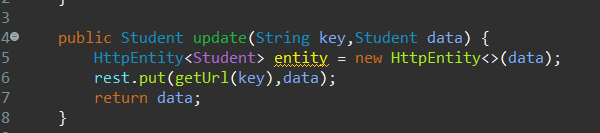
+delete(url)



+postForObject(url,httpEntity,responseType<T>); T



+put(url,httpEntity);



trong đó httpEntity<T> là đóng gói json gửi đến Rest API