**LAB 8: THỰC HÀNH API AUTOMATION TESTING VỚI REST ASSURED**

**Tổng quan**

Trong bài thực hành này, chúng ta sẽ vận dụng những kiến thức đã học để luyện tập cài đặt, tạo và thực thi API automation testcase. Tất cả việc này sẽ được hoàn thành bằng việc kết hợp giữa TestNG và Rest Assured, Assertion.

Về bản chất TestNG là một test framework, nó tương tự với Junit test.  
Trong khóa học này, chúng ta sẽ cùng nhau là việc với TestNG.

**I/ Yêu cầu thực hành**

1. Học viên biết cách tạo một project Maven, sử dụng pom.xml file để thêm thư viện
2. Học viên đã có kiến thức cơ bản về API
3. Học viên nắm được kiến thức cơ bản về TestNG, Rest Assured và Assertion, Test suite
4. Thực hành đưa giải pháp vào viết code, thực thi và sửa lỗi code nếu có
5. Lưu lại code.

**II/ Thiết lập project**

<Tham khảo file hướng dẫn: REF\_3\_Setup\_RestAssured.docx>

Bổ sung các thư viện dependencies sau :

* Hamcrest library: thư viện dành cho các method assertion
* TestNG
* Rest-Assured
* jackson-databind: xử lý json

pom.xml

| <**dependency**>  <**groupId**>org.hamcrest</**groupId**>  <**artifactId**>hamcrest</**artifactId**>  <**version**>2.2</**version**> </**dependency**>  <**dependency**>  <**groupId**>org.testng</**groupId**>  <**artifactId**>testng</**artifactId**>  <**version**>7.4.0</**version**> </**dependency**>  <**dependency**>  <**groupId**>com.fasterxml.jackson.core</**groupId**>  <**artifactId**>jackson-databind</**artifactId**>  <**version**>2.9.8</**version**> </**dependency**>  <**dependency**>  <**groupId**>io.rest-assured</**groupId**>  <**artifactId**>rest-assured</**artifactId**>  <**version**>4.4.0</**version**>  <**scope**>test</**scope**> </**dependency**> |
| --- |

**III/ Ví dụ mẫu**

**1.** Tạo mới một User.

URL Endpoint: <https://reqres.in/api/users>  
Method: POST

Body:

* name: String (bắt buộc) Tên User
* gender: String (bắt buộc) Giới tính
* age: Int Tuổi
* job: String Công việc

**Kết quả mong muốn:** Trả về kết quả dưới dạng JSON, trong đó có đầy đủ thông tin các trường vừa tạo, ngoài ra còn có trường id của user thời gian user được tạo createdAt.

**Điều kiện để test case pass:**

Trong kết quả trả về phải có HTTP Status Code = 201 hoặc 400 nếu name hoặc gender bằng null hoặc rỗng

Kết quả trả về phải chứa trường **id** , trường **createdAt** của user

Kết quả trả về phải đủ các trường đã tạo và thông tin các trường phải khớp.

Trong trường hợp nhận 404, thông tin phản hồi phải có trường message và trường này phải chứa tên trường bị thiếu trong request.

*Giải pháp:*

* Tạo body mẫu cho 2 trường hợp 201 và 400.  
  (Các bạn có thể tạo nhiều bộ cho nhiều trường hợp kiểm thử khác)
* Với trường hợp mong muốn 201.  
   - Viết TC xác minh HTTP status code = 201 ?  
   - Viết TC xác minh kết quả trả về có trường id hay không.  
   - Viết TC xác minh kết quả trả về có trường createdAt hay không.  
   - Viết TC kiểm tra trường name có giống với lúc tạo hay không.  
   - Viết TC kiểm tra trường gender có giống với lúc tạo hay không.  
   - Viết TC kiểm tra trường age có giống với lúc tạo hay không.  
   - Viết TC kiểm tra trường job có giống với lúc tạo hay không.
* Với trường hợp mong muốn 400

- Viết TC xác minh HTTP status code = 400 ?  
 - Viết TC xác minh kết quả trả về có trường message hay không, trong message này có chứa tên trường thông tin bị thiếu hay không?

* Tạo test Suite : **testng.xml** để thực the test.

**#Đáp án  
 1.** Tạo đối tượng User chứa các trường như mô tả body

**UserInfo.java**

| **package** data;  **public** **class** UserInfor {  **private** String name;  **private** String gender;  **private** **int** age;  **private** String job;      **public** UserInfor() {}    **public** UserInfor(String name, String gender, **int** age, String job) {  **super**();  **this**.name = name;  **this**.gender = gender;  **this**.age = age;  **this**.job = job;  }  **public** String getName() {  **return** name;  }  **public** **void** setName(String name) {  **this**.name = name;  }  **public** String getGender() {  **return** gender;  }  **public** **void** setGender(String gender) {  **this**.gender = gender;  }  **public** **int** getAge() {  **return** age;  }  **public** **void** setAge(**int** age) {  **this**.age = age;  }  **public** String getJob() {  **return** job;  }  **public** **void** setJob(String job) {  **this**.job = job;  }    } |
| --- |

2. Tạo các test case cho trường hợp valid body

Example1Test.java

| **package** demo;  **import** io.restassured.RestAssured;  **import** io.restassured.http.ContentType;  **import** io.restassured.path.json.JsonPath;  **import** io.restassured.response.Response;  **import** io.restassured.response.ResponseBody;  **import** io.restassured.specification.RequestSpecification;  **import** org.testng.annotations.AfterClass;  **import** org.testng.annotations.BeforeClass;  **import** org.testng.annotations.Test;  **import** data.UserInfor;  **import** **static** io.restassured.RestAssured.*given*;  **import** **static** org.testng.Assert.*assertEquals*;  **import** **static** org.testng.Assert.*assertTrue*;  **public** **class** Example1Test {  **private** Response response; // lưu response của API  **private** UserInfor user; // thông tin request body  **private** ResponseBody resBody; //body của response  **private** JsonPath bodyJson; //body của response đã được convert sang JSON  @BeforeClass  **public** **void** init() {  //Tạo body mẫu  user = **new** UserInfor("Minh", "nam", 24, "Ky su");  // thiết lập endpoint  RestAssured.*baseURI* = "https://reqres.in"; //Setup Base URI  RestAssured.*basePath* = "/api/users"; //Setup Base Path  //tạo request  RequestSpecification req = RestAssured.*given*()  .contentType(ContentType.***JSON***)  .when()  .body(user);  //gọi POST và lấy về response  response = req.post();  resBody = response.getBody();  bodyJson = resBody.jsonPath();  }  @Test  **public** **void** T01\_StatusCodeTest() {  //Kiểm tra HTTP status có = 201 hay không  *assertEquals*(201, response.getStatusCode(), "Status Check Failed!");  }  @Test  **public** **void** T02\_IdChecked() {  //Kiểm tra phản hồi có chưa trường id hay không  *assertTrue*(resBody.asString().contains("id"), "id field check Failed!");  }  @Test  **public** **void** T03\_CreatedAtChecked() {  //Kiểm tra phản hồi có chứa trường createAt hay không  *assertTrue*(resBody.asString().contains("createdAt"), "createdAt field check Failed!");  }  @Test  **public** **void** T04\_verifyOnMatchingName() {  //Kiểm tra phản hồi tên có giống với request body không  String resName = bodyJson.get("name");  *assertEquals*(user.getName(), resName, "Name is not matched!");  }  @Test  **public** **void** T05\_verifyOnMatchingGender() {  //Kiểm tra phản hồi giới tính có giống với request body không  String resGender = bodyJson.get("gender");  *assertEquals*(user.getGender(), resGender, "Gender is not matched!");  }  @Test  **public** **void** T06\_verifyOnMatchingAge() {  //Kiểm tra phản hồi tuổi có giống với request body không  **int** resAge = bodyJson.getInt("age");  *assertEquals*(user.getAge(), resAge, "Age is not matched!");  }  @Test  **public** **void** T07\_verifyOnMatchingJob() {  //Kiểm tra phản hồi công việc có giống với request body không  String resJob = bodyJson.get("job");  *assertEquals*(user.getJob(), resJob,"Job is not matched!");  }  @AfterClass  **public** **void** afterTest() {  //kết thúc test hiện tại thì clear enpoint để chạy tiếp test liền sau trong test Suite  // ( có ý nghĩa nếu test sau không sử dụng chung enpoint với test hiện tại)  //vd test1.java : http://abc.com , test2.java: https://xyz.com  RestAssured.*baseURI* = **null**; //Setup Base URI  RestAssured.*basePath* = **null**;  }  } |
| --- |

3. Tạo các test case cho trường hợp inValid body.

Example2Test.java

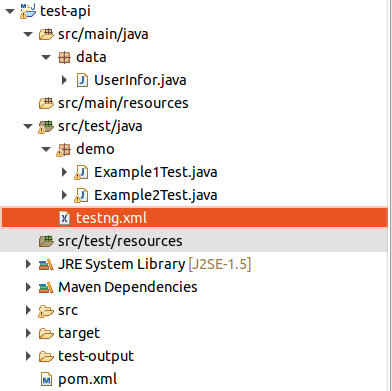
| **package** demo;  **import** io.restassured.RestAssured;  **import** io.restassured.http.ContentType;  **import** io.restassured.path.json.JsonPath;  **import** io.restassured.response.Response;  **import** io.restassured.response.ResponseBody;  **import** io.restassured.specification.RequestSpecification;  **import** org.codehaus.groovy.util.StringUtil;  **import** org.testng.annotations.AfterClass;  **import** org.testng.annotations.BeforeClass;  **import** org.testng.annotations.Test;  **import** data.UserInfor;  **import** **static** io.restassured.RestAssured.*given*;  **import** **static** org.testng.Assert.*assertEquals*;  **import** **static** org.testng.Assert.*assertTrue*;  **public** **class** Example2Test {  **private** Response response;  **private** UserInfor user;  **private** ResponseBody resBody;  **private** JsonPath bodyJson;  @BeforeClass  **public** **void** init() {  user = **new** UserInfor(**null**, "nam", 24, "Ky su");  RestAssured.*baseURI* = "https://reqres.in"; //Setup Base URI  RestAssured.*basePath* = "/api/users"; //Setup Base Path  //create request  RequestSpecification req = RestAssured.*given*()  .contentType(ContentType.***JSON***)  .when()  .body(user);  //take a post & get response  response = req.post();  resBody = response.getBody();  bodyJson = resBody.jsonPath();  }  @Test  **public** **void** T01\_StatusCodeTest() {  //Verify the http response status returned. Check Status Code is 404?  *assertEquals*( 404, response.getStatusCode(),"Status Check Failed!");  }  @Test  **public** **void** T02\_MessageChecked() {  *assertEquals*(**true**, resBody.asString().contains("message"), "Message field check Failed!");  }  @Test  **public** **void** T03\_verifyOnMessageContainName() {  String resName = bodyJson.get("message");  **if** (**null** == resName) resName = "";  *assertEquals*(**true**, resName.contains("name"),"Message not contain invalid field!");  }  @AfterClass  **public** **void** afterTest() {  //Reset Values  RestAssured.*baseURI* = **null**; //Setup Base URI  RestAssured.*basePath* = **null**;  }  } |
| --- |

4. Tạo Test Suite, đặt tên và liệt kê các java class test case vừa viết  
Thứ tự khai báo trong suite cũng là thứ tự thực thi.

**testng.xml**

| <?xml version = *"1.0"* encoding = *"UTF-8"*?>  <!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >  <suite name = *"Suite1"*>  <test name = *"validPost"*>  <classes>  <class name = *"demo.Example1Test"* />  </classes>  </test>  <test name = *"inValidPost"*>  <classes>  <class name = *"demo.Example2Test"* />  </classes>  </test>  </suite> |
| --- |

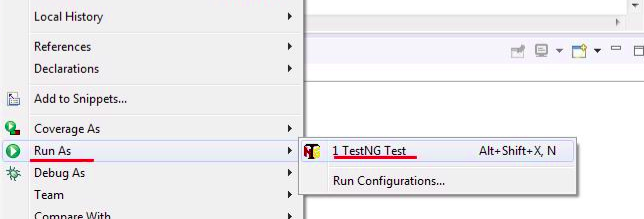
Trong ví dụ này, các file java được đặt trong package **demo** và **testng.xm**l được đặt ngang hàng với package này, nên ta các class sẽ được khai báo   
“demo.[ClassName]”



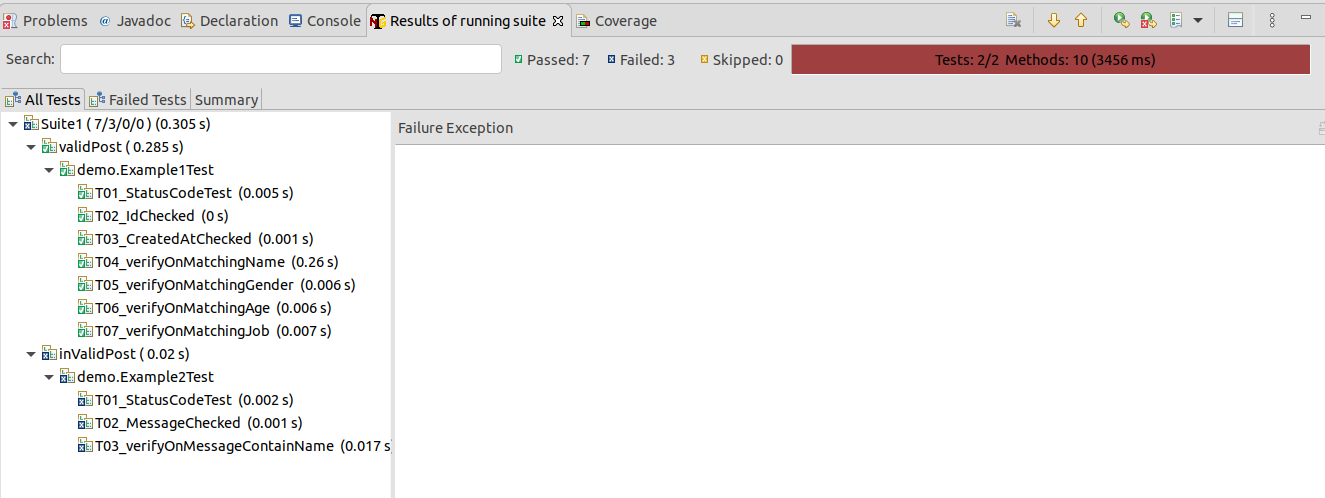
**Bố cục project:**

5. Chạy test suite

Chuột phải file **testng.xml → Run as → TestNG suite**



**#Kết quả**



**III. Thực hành**

**1.** Xem chi tiết thông tin user theo user id (cho trước)

URL Endpoint: [https://reqres.in/api/users/{userId}](https://reqres.in/api/users)  
Method: GET

Param:

* [userId](https://reqres.in/api/users): int (bắt buộc) id của user

**Kết quả mong muốn:** Trả về kết quả dưới dạng JSON, trong đó có đầy đủ thông tin các trường user như sau:

{

"data": {

"id",

"email":

"first\_name": ,

"last\_name": ,

"avatar":

},

"support": {

"url"

"text"

}

}

**Dữ liệu cho trước:**  
userId hợp lệ : 2  
userId không hợp lệ: 40

**Điều kiện để test case pass:**

Trong kết quả trả về phải có HTTP Status Code = 200 hoặc 404 nếu userId không tồn tại.

Kết quả trả về phải đầy đủ thông tin user như mô tả bên trên (id, email, first\_name, last\_name, avatar)

Trường id trả về phải trùng với user id tìm kiếm.

Trong trường hợp nhận 404, thông tin phản hồi phải có trường message = “User not found!”

*Gợi ý*

* Tạo 2 class Example1Test.java và Example2Test.java tương ứng với 2 kịch bản test userId.
* Trong @BeforeClass của TestNG: khởi tạo api request GET và lấy về kết quả:  
  Lưu lại kết quả cả dưới dạng chuỗi String và JSON để tiện cho việc xử lý ở các test case.  
  Tạo request:

| **public** **class** Example1Test {  **private** Response response; // lưu response của API  **private** ResponseBody resBody; //body của response  **private** JsonPath bodyJson; //body của response đã được convert sang JSON  **int** userId;  @BeforeClass  **public** **void** init() {  //  userId =... ;  // thiết lập endpoint  RestAssured.*baseURI* = ..; //Setup Base URI  RestAssured.*basePath* = ..; //Setup Base Path  //tạo request  RequestSpecification req = RestAssured.*given*()  .contentType(ContentType.***JSON***)  .when()  .pathParam("userId", userId);  //gọi GET và lấy về response  response = req.get("/{userId}");  resBody = response.getBody();  bodyJson = resBody.jsonPath(); // response JSON  } |
| --- |

* Với trường hợp mong muốn 200 OK.  
   - Viết TC xác minh HTTP status code = 200 ?  
   - Viết TC xác minh kết quả trả về có trường id hay không.  
   - Viết TC xác minh kết quả trả về có trường email hay không.  
   - Viết TC xác minh kết quả trả về có trường first\_name hay không.  
   - Viết TC xác minh kết quả trả về có trường last\_name hay không.  
   - Viết TC xác minh kết quả trả về có trường avatar hay không.  
   - Viết TC kiểm tra trường id có giống với user Id hay không.  
   Chú ý: để lấy tra 1 trường id, các bạn có thể tham khảo:  
   **int** id = bodyJson.getInt("data.id");
* Với trường hợp mong muốn 404

- Viết TC xác minh HTTP status code = 404 ?  
 - Viết TC xác minh kết quả trả về có trường message hay không.

- Viết TC kiểm tra nội dung message có phải là "User not found!" ?

Tạo test Suite : **testng.xml** để thực thi test.