1. What are the types of methods most used in RestAPI Testing?  
  
GET/POST/PUT/PATCH/DELETE/HEAD/OPTIONS  
  
P1 : GET POST DELETE  
P2 : PUT  
P3 : PATCH  
  
2. What are the types of Status codes?  
  
1xx informational response – the request was received, continuing process  
2xx successful – the request was successfully received, understood, and accepted  
3xx redirection – further action needs to be taken in order to complete the request  
4xx client error – the request contains bad syntax or cannot be fulfilled  
5xx server error – the server failed to fulfil an apparently valid request  
  
3. What are the status codes you have come across in your API testing project?  
<https://automationreinvented.blogspot.com/2019/03/what-are-most-used-api-status-codes.html>

4. What is GET Method?

GET    Retrieve information about the REST API resource

5. What is POST Method?

POST    Create a REST API resource

6.What is PUT Method?

PUT    Update a REST API resource

7. What is DELETE Method?

DELETE    Delete a REST API resource or related component  
  
8. What is HEAD method?  
  
The HEAD method asks for a response identical to that of a GET request, but without the response body.  
This is useful for retrieving meta-information written in response headers, without having to transport the entire content  
  
9. What is OPTIONS method?  
  
The OPTIONS method returns the HTTP methods that the server supports for the specified URL.  
This can be used to check the functionality of a web server by requesting '\*' instead of a specific resource.  
  
10. PUT VS POST

<https://automationreinvented.blogspot.com/2019/04/put-vs-post-example-lets-list-down-when.html>

**WHAT IS AN API?**

Imagine we are sitting at a table in a restaurant with a menu of choices to order from. The kitchen is the part of the “system” that will prepare your order. What is missing is a critical link to communicate your order to the kitchen and deliver your food back to your table. That’s where the waiter or API comes in. The waiter is the messenger – or API – that takes your request or order and tells the kitchen – the system – what to do. Then the waiter delivers the response back to you; in this case, it is the food.

Here is a real-life API example. We may be familiar with the process of searching for flights online. Just like the restaurant, you have a variety of options to choose from, including different cities, departure and return dates, and more. Let us imagine that you’re booking you are flight on an airline website.

We choose departure city and date, return city and date, cabin class, as well as other variables. In order to book your flight, you interact with the airline’s website to access their database and see if any seats are available on those dates and what the costs might be.

However, what if we are not using the airline’s website? What if we are using an online travel service, such as Kayak or Expedia, which aggregates information from a number of airline databases?

The travel service, in this case, interacts with the airline’s API. The API is the interface that, like your helpful waiter, can be asked by that online travel service to get information from the airline’s database to book seats, baggage options, etc. The API then takes the airline’s response to your request and delivers it right back to the online travel service, which then shows you the most updated, relevant information.

**11. Automate GET method and validate the status code?**

|  |
| --- |
| @Test(description="Verify status code for GET method-users/2 as 200") public static void verifyStatusCodeGET() {    Response resp = given()                                         .when()         .get("https://reqres.in/api/users/2");   assertEquals(resp.getStatusCode(),200);   } |

**12.  Automate GET method and fetch response body?**

 @Test(description="Verify status code for GET method-users/2 as 200")  
 public static void verifyStatusCodeGET() {  
       
     Response resp=given().when().get("https://reqres.in/api/users/2");  
     assertEquals(resp.getBody().asString(),200);  
       
 }

**Define API Authenticationmethod automation with Rest Assured for basic, Pre-emptive, Digest?**

[**Click here for Answers with Scripts**](https://automationreinvented.blogspot.com/2020/09/api-authentication-methods-automation.html)

**13. Automate GET method and verify value from response body?(validate that total number pages =12)**  
 

|  |
| --- |
| @Test(description="Verify status code for GET method-users/2 as 200") public static void verifyStatusCodeGET() {    Response resp=given().when().get("https://reqres.in/api/users");   System.out.println(resp.path("total").toString());   assertEquals(resp.getStatusCode(),200);  assertEquals(resp.path("total").toString(),"12"); } |

**14. How to pass query param with GET method in Rest Assured?**

API Query parameters can be defined as **the optional key-value pairs that appear after the question mark in the URL**. Basically, they are extensions of the URL that are utilized to help determine specific content or action based on the data being delivered. Query parameters are appended to the end of the URL, using a '?

|  |
| --- |
| @Test  public void validateQueryParamInGiven() {      Response resp = given().queryParam("page", "2").  when().get("https://reqres.in/api/users");   assertEquals(resp.getStatusCode(),200);   System.out.println(resp.getBody().asString());  } |

**How to Setup API automation framework from scratch using Rest Assured, Gradle with code?**

[**Click here to Answer with Code**](https://automationreinvented.blogspot.com/2020/10/setup-api-automation-framework-from.html)

     
**15. How to pass header for GET method in Rest Assured?**

|  |
| --- |
| @Test  public void validateGivenHeader() {      Response resp = given().header("Content-Type", "application/json").  when().get("https://gorest.co.in/public-api/users");   assertEquals(resp.getStatusCode(),200);  System.out.println(resp.getBody().asString());  } |

16.**How to automate PATCH method in rest Assured?**

The HTTP PATCH method can be used when a resource needs to be updated. This method is especially useful if a resource is large and the changes being made are small.

|  |
| --- |
| @Test(description="validate with jsonpath and json object and pass post body as json file")  public void MethodValidationPUT() throws IOException, ParseException {      FileInputStream file = new FileInputStream(new File (System.getProperty("user.dir")+"\\TestData\\put.json"));   Response resp =  given().header("Content-Type" , "application/json").body(IOUtils.toString(file,"UTF-8")).          when().patch("https://reqres.in/api/users/2");   assertEquals(resp.getStatusCode(),200);  assertEquals(resp.path("job"),"tester");      } |

**17. How to automate PUT method in Rest Assured?**

**A PUT method puts or places a file or resource precisely at a specific URI**. In case a file or a resource already exists at that URI, the PUT method replaces that file or resource. If there is no file or resource, PUT creates a new one. 

|  |
| --- |
| @Test(description="validate with jsonpath and json object and pass post body as json file")  public void MethodValidationPUT() throws IOException, ParseException {      FileInputStream file = new FileInputStream(new File (System.getProperty("user.dir")+"\\TestData\\put.json"));   Response resp =  given().header("Content-Type" , "application/json").body(IOUtils.toString(file,"UTF-8")).          when().put("https://reqres.in/api/users/2");   assertEquals(resp.getStatusCode(),200);  assertEquals(resp.path("job"),"tester");      } |

[***How to read String, List and Map in JSON response from API ?***](https://automationreinvented.blogspot.com/2021/04/how-to-read-string-list-and-map-in-json.html)

[**Click Here For Answer with Examples**](https://automationreinvented.blogspot.com/2021/04/how-to-read-string-list-and-map-in-json.html)

**18. How to automate POST method in Rest Assured?**

 POST requests are **used to send data to the API server to create or update a resource**. The data sent to the server is stored in the request body of the HTTP request

   **21. How to use Basic authentication in automation?**

        Response resp = given()  
                .auth()  
                .basic("sid", "sid").when().get("https://reqres.in/api/users/2");  
                  
    **22. How to use Pre-emptive authentication in automation?**

        Response resp1 = given()  
                .auth()  
                .preemptive().basic("sid", "sid").when().get("https://reqres.in/api/users/2");  
                  
    **23. How to use digest authentication in automation?**

        Response resp2 = given()  
                .auth()  
                .digest("sid", "sid").when().get("https://reqres.in/api/users/2");  
      
**24. How to use Oauth2 authentication in automation?**

        Response resp3 = given()  
                .auth()  
                .oauth2("").when().get("https://reqres.in/api/users/2");  
      
   **25. How to use Oauth authentication in automation?**    
      
        Response resp4 = given()  
                .auth()  
                .oauth("consumerKey", "consumerSecret", "accessToken", "secretToken").when().get("https://reqres.in/api/users/2");  
                  
   **26. How to use header for authorization(oauth2) in automation?**  
        Response resp5 = given().header("Authorization","accessToken")  
                    .when().get("https://reqres.in/api/users/2");  
                  
    }

**27. How to automate GET method using Rest Assured?**

[**Click here for answer**](https://automationreinvented.blogspot.com/2020/05/automate-get-method-with-rest-assured.html)

**28. Difference between status code 401 and 403?**

[**Click here for answer 401 vs 403**](https://automationreinvented.blogspot.com/2020/09/difference-between-api-status-code-401.html)

**29.** **What are main differences between API and Web Service?**

    - All Web services are APIs but not all APIs are Web services.

    - Web service uses three styles of use: SOAP, REST and XML-RPC for communication whereas   API  may be exposed in multiple ways.

   -  Web service needs a network to operate but APIs don’t need a network to operate.

**30. What is REST?**  
- REST (Representational State Transfer) is an architectural style for developing web services which exploit the ubiquity of HTTP protocol and uses HTTP method to define actions. It revolves around resource where every component being a resource that can be accessed through a shared interface using standard HTTP methods.

- In REST architecture, REST Server provides access to resources and client accesses and makes these resources available.

- Each resource is identified by URIs or global IDs, and REST uses multiple ways to represent a resource, such as text, JSON, and XML.