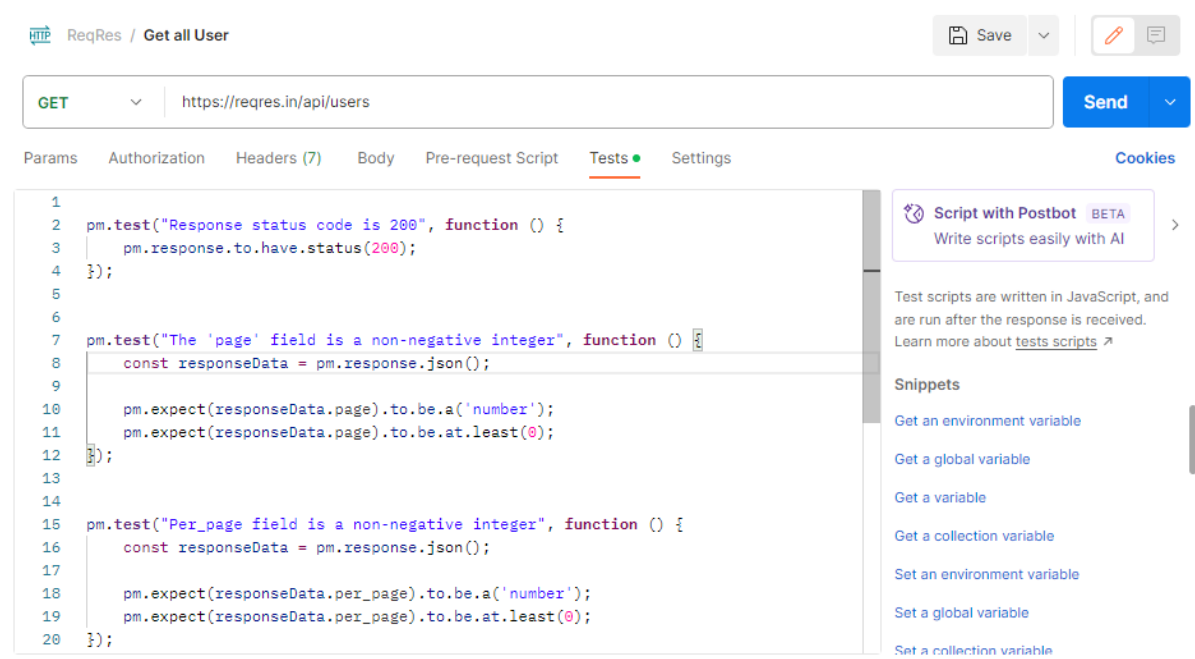


Non-Functional Testing Using Postman, REST Assured, and JMeter – Source code

GET Request



ReqRes / Get all User

GET https://reqres.in/api/users Send

Params Authorization Headers (7) Body Pre-request Script **Tests** Settings Cookies

```
1
2 pm.test("Response status code is 200", function () {
3   pm.response.to.have.status(200);
4 });
5
6
7 pm.test("The 'page' field is a non-negative integer", function () {
8   const responseData = pm.response.json();
9
10  pm.expect(responseData.page).to.be.a('number');
11  pm.expect(responseData.page).to.be.at.least(0);
12 });
13
14
15 pm.test("Per_page field is a non-negative integer", function () {
16   const responseData = pm.response.json();
17
18   pm.expect(responseData.per_page).to.be.a('number');
19   pm.expect(responseData.per_page).to.be.at.least(0);
20 });
```

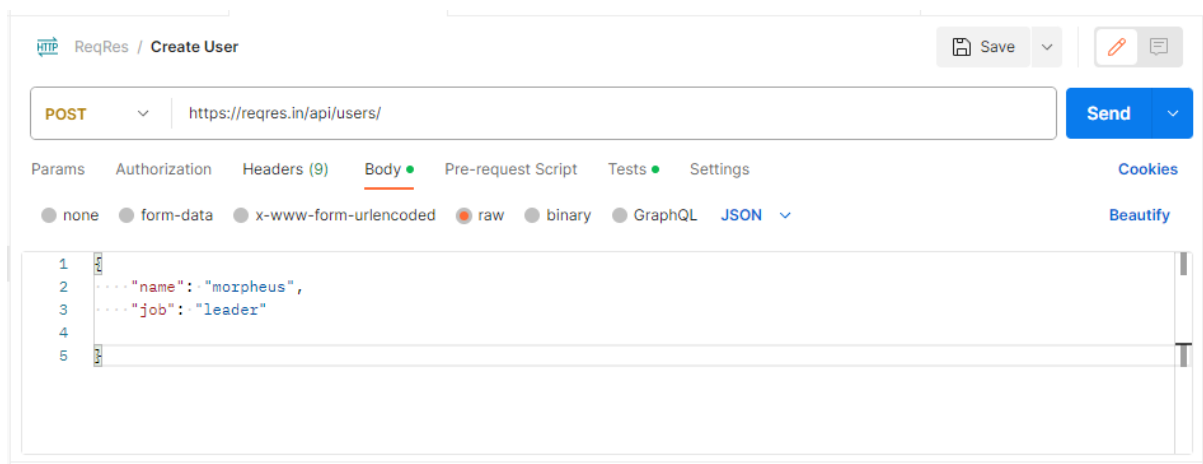
Script with Postbot BETA
Write scripts easily with AI

Test scripts are written in JavaScript, and are run after the response is received. [Learn more about tests scripts](#)

Snippets

- Get an environment variable
- Get a global variable
- Get a variable
- Get a collection variable
- Set an environment variable
- Set a global variable
- Set a collection variable

POST Request



ReqRes / Create User

POST https://reqres.in/api/users/ Send

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** Beautify

```
1
2 {
3   "name": "morpheus",
4   "job": "leader"
5 }
```


POST https://reqres.in/api/users/

Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings

Cookies

```
1
2 pm.test("Response status code is 201", function () {
3   pm.response.to.have.status(201);
4 });
5
6
7 pm.test("Response has required fields", function () {
8   const responseData = pm.response.json();
9
10  pm.expect(responseData).to.be.an('object');
11  pm.expect(responseData).to.have.property('name');
12  pm.expect(responseData).to.have.property('job');
13  pm.expect(responseData).to.have.property('id');
14  pm.expect(responseData).to.have.property('createdAt');
15 });
16
17
18 pm.test("Name is a non-empty string", function () {
19   const responseData = pm.response.json();
```

 Script with Postbot BETA
Write scripts easily with AI

Test scripts are written in JavaScript, and are run after the response is received.
[Learn more about tests scripts](#)

Snippets

[Get an environment variable](#)[Get a global variable](#)[Get a variable](#)[Get a collection variable](#)[Set an environment variable](#)[Set a global variable](#)

Create Get Resource.java

```
package training.PhaseEnd;

import java.util.HashMap;

import org.testng.Assert;
import org.testng.annotations.Test;

import io.restassured.RestAssured;
import io.restassured.path.json.JsonPath;
import io.restassured.response.Response;

public class Create_Get_Resource extends TestBase {

    int id;

    HashMap<String, String> map = new HashMap<String, String>();

    @Test(priority = 0)

    public void createPayload() {
        map.put("name", "Yokesh");
        map.put("job", "Software Engineer");
        RestAssured.baseURI = "https://reqres.in/";
        RestAssured.basePath = "/api/users";
        logger.info("payload created");
    }

    @Test(priority = 1)

    public void getServerResponse() {

        RestAssured.when().get("https://reqres.in/api/users?page=2").then().
        assertThat().statusCode(200);
        Assert.assertTrue(jsonPath.get("data[0].first_name").equals("Michael
"));
        logger.info("Reterived Users");
    }

    @Test(priority = 2)

    public void createResource() {
        Response response =
        RestAssured.given().contentType("application/json").body(map).when().post()
        .then()

        .statusCode(201).extract().response();

        JsonPath jsonPath = response.jsonPath();

        Assert.assertTrue(jsonPath.get("name").toString().equals("Yokesh"));
        logger.info("Resource created");
    }
}
```

TestBase.java

```
package training.PhaseEnd;

import org.apache.log4j.Level;
import org.apache.log4j.Logger;
import org.apache.log4j.PropertyConfigurator;
import org.testng.annotations.BeforeClass;

import io.restassured.response.Response;
import io.restassured.specification.RequestSpecification;

public class TestBase {
    public static RequestSpecification httpRequest;
    public static Response response;

    public Logger logger;

    @BeforeClass
    public void setUp() {
        logger = Logger.getLogger("LogDemo");
        PropertyConfigurator.configure("log4j.properties");
        logger.setLevel(Level.DEBUG);
    }
}
```

log4j.properties

```
log4j.rootLogger=ERROR,stdout
log4j.logger.com.endeca=INFO
log4j.logger.LogDemo=DEBUG, dest1
# Logger for crawl metrics
log4j.logger.com.endeca.itl.web.metrics=INFO

log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%p\t%d{ISO8601}\t%r\t%c\t[%t
]\t%m%n
```

RestDemo.jmx

```
<?xml version="1.0" encoding="UTF-8"?>

<jmeterTestPlan version="1.2" properties="5.0" jmeter="5.6.2">

  <hashTree>

    <TestPlan guiclass="TestPlanGui" testclass="TestPlan" testname="Test Plan"
enabled="true">

      <boolProp name="TestPlan.functional_mode">false</boolProp>

      <boolProp name="TestPlan.tearDown_on_shutdown">false</boolProp>

      <boolProp name="TestPlan.serialize_threadgroups">false</boolProp>

      <elementProp name="TestPlan.user_defined_variables" elementType="Arguments"
guiclass="ArgumentsPanel" testclass="Arguments" testname="User Defined Variables"
enabled="true">

        <collectionProp name="Arguments.arguments"/>

      </elementProp>

    </TestPlan>

    <hashTree>

      <ThreadGroup guiclass="ThreadGroupGui" testclass="ThreadGroup" testname="Thread
Group" enabled="true">

        <stringProp name="ThreadGroup.on_sample_error">continue</stringProp>

        <elementProp name="ThreadGroup.main_controller" elementType="LoopController"
guiclass="LoopControlPanel" testclass="LoopController" testname="Loop Controller"
enabled="true">

          <stringProp name="LoopController.loops">1</stringProp>

          <boolProp name="LoopController.continue_forever">false</boolProp>

        </elementProp>

        <stringProp name="ThreadGroup.num_threads">1</stringProp>

        <stringProp name="ThreadGroup.ramp_time">1</stringProp>

        <boolProp name="ThreadGroup.delayedStart">false</boolProp>

        <boolProp name="ThreadGroup.scheduler">false</boolProp>

        <stringProp name="ThreadGroup.duration"></stringProp>

        <stringProp name="ThreadGroup.delay"></stringProp>

        <boolProp name="ThreadGroup.same_user_on_next_iteration">true</boolProp>

      </ThreadGroup>

    </hashTree>

  </jmeterTestPlan>
</xml>
```

```

</ThreadGroup>

<hashTree>

  <HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy"
testname="GET Request" enabled="true">

    <boolProp name="HTTPSampler.postBodyRaw">false</boolProp>

    <elementProp name="HTTPSampler.Arguments" elementType="Arguments"
guiclass="HTTPArgumentsPanel" testclass="Arguments" testname="User Defined Variables"
enabled="true">

      <collectionProp name="Arguments.arguments"/>

    </elementProp>

    <stringProp name="HTTPSampler.domain">reqres.in</stringProp>

    <stringProp name="HTTPSampler.protocol">https</stringProp>

    <stringProp name="HTTPSampler.path">/api/users?page=2</stringProp>

    <stringProp name="HTTPSampler.method">GET</stringProp>

    <boolProp name="HTTPSampler.follow_redirects">true</boolProp>

    <boolProp name="HTTPSampler.auto_redirects">false</boolProp>

    <boolProp name="HTTPSampler.use_keepalive">true</boolProp>

    <boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>

    <boolProp
name="HTTPSampler.BROWSER_COMPATIBLE_MULTIPART">false</boolProp>

    <boolProp name="HTTPSampler.image_parser">false</boolProp>

    <boolProp name="HTTPSampler.concurrentDwn">false</boolProp>

    <stringProp name="HTTPSampler.concurrentPool">6</stringProp>

    <boolProp name="HTTPSampler.md5">false</boolProp>

    <intProp name="HTTPSampler.ipSourceType">0</intProp>

  </HTTPSamplerProxy>

</hashTree/>

  <HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy"
testname="POST Request" enabled="true">

    <boolProp name="HTTPSampler.postBodyRaw">true</boolProp>

    <elementProp name="HTTPSampler.Arguments" elementType="Arguments">

      <collectionProp name="Arguments.arguments">

        <elementProp name="" elementType="HTTPArgument">

```

```

        <boolProp name="HTTPArgument.always_encode">false</boolProp>
        <stringProp name="Argument.value">{&#xd;
&quot;name&quot;; &quot;Yokesh K&quot;;&#xd;
&quot;job&quot;; &quot;Software Engineer&quot;;&#xd;
}</stringProp>
        <stringProp name="Argument.metadata">=</stringProp>
    </elementProp>
</collectionProp>
</elementProp>
<stringProp name="HTTPSampler.domain">reqres.in</stringProp>
<stringProp name="HTTPSampler.protocol">https</stringProp>
<stringProp name="HTTPSampler.path">/api/users</stringProp>
<stringProp name="HTTPSampler.method">POST</stringProp>
<boolProp name="HTTPSampler.follow_redirects">true</boolProp>
<boolProp name="HTTPSampler.auto_redirects">false</boolProp>
<boolProp name="HTTPSampler.use_keepalive">true</boolProp>
<boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
<boolProp
name="HTTPSampler.BROWSER_COMPATIBLE_MULTIPART">false</boolProp>
<boolProp name="HTTPSampler.image_parser">false</boolProp>
<boolProp name="HTTPSampler.concurrentDwn">false</boolProp>
<stringProp name="HTTPSampler.concurrentPool">6</stringProp>
<boolProp name="HTTPSampler.md5">false</boolProp>
<intProp name="HTTPSampler.ipSourceType">0</intProp>
</HTTPSamplerProxy>
<hashTree/>
<HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy"
testname="PUT Request" enabled="true">
    <boolProp name="HTTPSampler.postBodyRaw">true</boolProp>
    <elementProp name="HTTPSampler.Arguments" elementType="Arguments">
        <collectionProp name="Arguments.arguments">
            <elementProp name="" elementType="HTTPArgument">

```

```

        <boolProp name="HTTPArgument.always_encode">false</boolProp>

        <stringProp name="Argument.value">{&#xd;
&quot;name&quot;; &quot;Yokesh K&quot;;&#xd;
&quot;job&quot;; &quot;Automation Tester&quot;;&#xd;
}</stringProp>

        <stringProp name="Argument.metadata">=</stringProp>
    </elementProp>
</collectionProp>
</elementProp>

<stringProp name="HTTPSampler.domain">reqres.in</stringProp>
<stringProp name="HTTPSampler.protocol">https</stringProp>
<stringProp name="HTTPSampler.path">/api/users/2</stringProp>
<stringProp name="HTTPSampler.method">PUT</stringProp>
<boolProp name="HTTPSampler.follow_redirects">true</boolProp>
<boolProp name="HTTPSampler.auto_redirects">false</boolProp>
<boolProp name="HTTPSampler.use_keepalive">true</boolProp>
<boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
<boolProp
name="HTTPSampler.BROWSER_COMPATIBLE_MULTIPART">false</boolProp>
<boolProp name="HTTPSampler.image_parser">false</boolProp>
<boolProp name="HTTPSampler.concurrentDwn">false</boolProp>
<stringProp name="HTTPSampler.concurrentPool">6</stringProp>
<boolProp name="HTTPSampler.md5">false</boolProp>
<intProp name="HTTPSampler.ipSourceType">0</intProp>
</HTTPSamplerProxy>
<hashTree/>

<HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy"
testname="DELETE Request" enabled="true">

    <boolProp name="HTTPSampler.postBodyRaw">false</boolProp>

    <elementProp name="HTTPsampler.Arguments" elementType="Arguments"
guiclass="HTTPArgumentsPanel" testclass="Arguments" testname="User Defined Variables"
enabled="true">

```



```
<collectionProp name="Arguments.arguments"/>
</elementProp>
<stringProp name="HTTPSampler.domain">reqres.in</stringProp>
<stringProp name="HTTPSampler.protocol">https</stringProp>
<stringProp name="HTTPSampler.path">/api/users/2</stringProp>
<stringProp name="HTTPSampler.method">DELETE</stringProp>
<boolProp name="HTTPSampler.follow_redirects">true</boolProp>
<boolProp name="HTTPSampler.auto_redirects">false</boolProp>
<boolProp name="HTTPSampler.use_keepalive">true</boolProp>
<boolProp name="HTTPSampler.DO_MULTIPART_POST">false</boolProp>
<boolProp
name="HTTPSampler.BROWSER_COMPATIBLE_MULTIPART">false</boolProp>
  <boolProp name="HTTPSampler.image_parser">false</boolProp>
  <boolProp name="HTTPSampler.concurrentDwn">false</boolProp>
  <stringProp name="HTTPSampler.concurrentPool">6</stringProp>
  <boolProp name="HTTPSampler.md5">false</boolProp>
  <intProp name="HTTPSampler.ipSourceType">0</intProp>
</HTTPSamplerProxy>
<hashTree/>
</hashTree>
</hashTree>
</hashTree>
</jmeterTestPlan>
```