

Testing the CRUD Operations and performance of List of Users – WriteUp

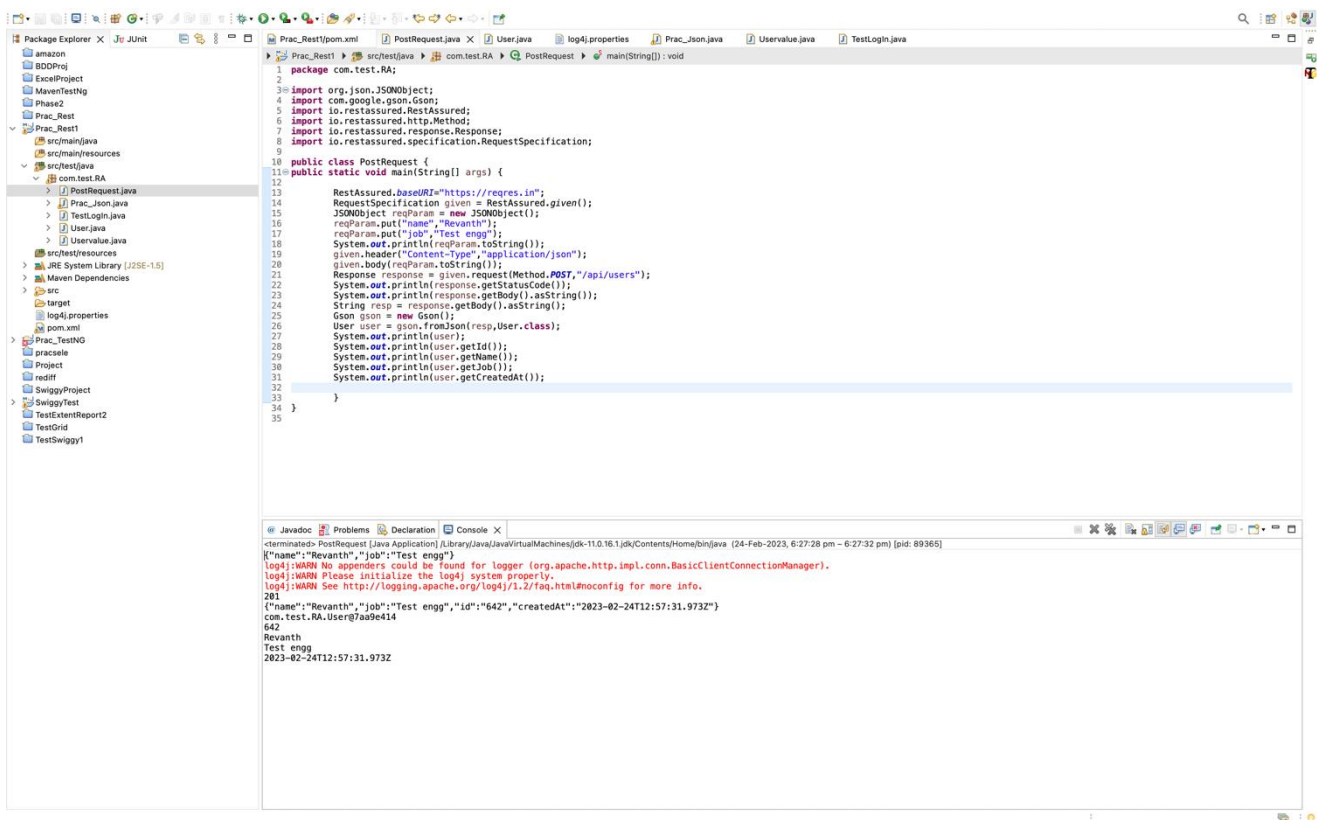
Introduction:-

First with the help of <https://reqres.in/api/users> this api in postman, We can able to access the list of users. The basic requirement:-

- Eclipse as the IDE
- Java 1.8
- Url(<https://reqres.in/api/users?page=2>)
- Postman
- Jmeter

We have created the :-

- PostRequest.java
- Prac_Json.java
- TestLogIn.java
- User.java
- Uservalue.java



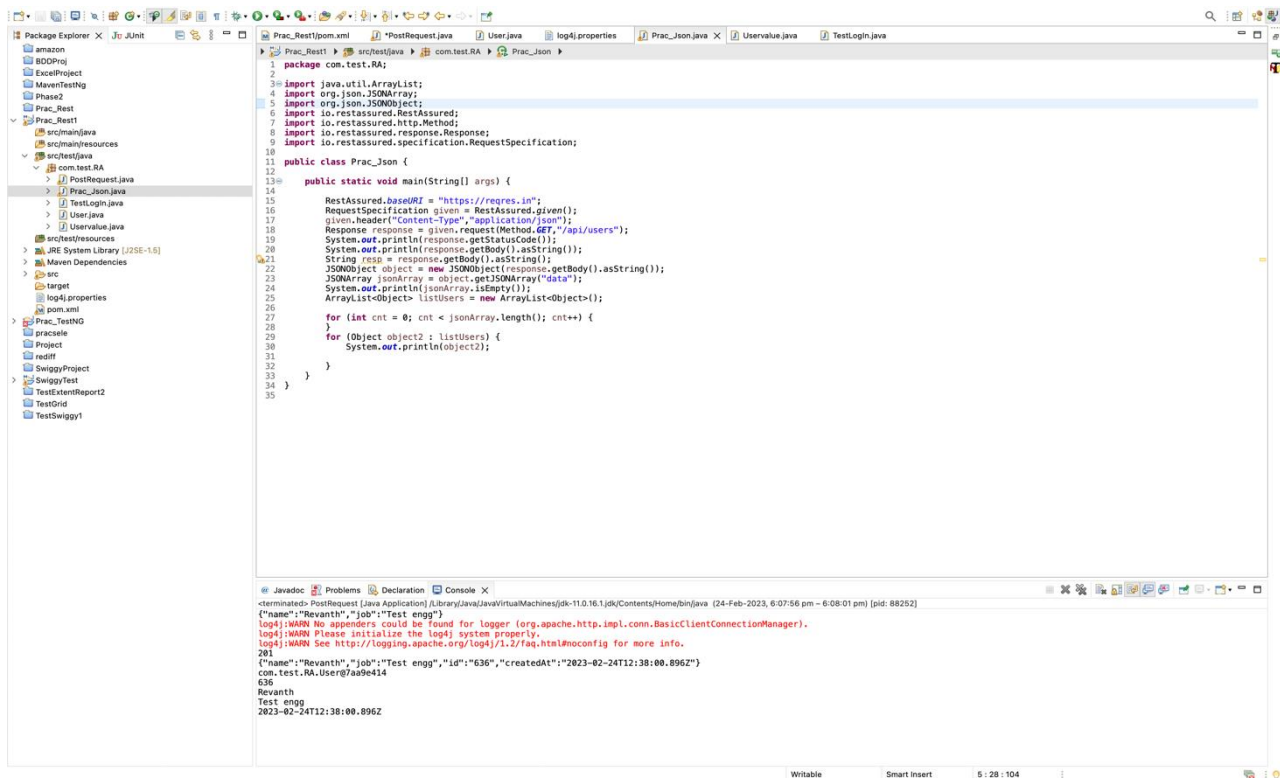
The screenshot shows the Eclipse IDE with the `PostRequest.java` file open. The code is as follows:

```
1 package com.test.RA;
2
3 import org.json.JSONObject;
4 import com.google.gson.Gson;
5 import io.restassured.RestAssured;
6 import io.restassured.http.Method;
7 import io.restassured.response.Response;
8 import io.restassured.specification.RequestSpecification;
9
10 public class PostRequest {
11     public static void main(String[] args) {
12         RestAssured.baseURI = "https://reqres.in";
13         RequestSpecification given = RestAssured.given();
14         JSONObject reqParam = new JSONObject();
15         reqParam.put("name", "Revanth");
16         reqParam.put("job", "Test engg");
17         System.out.println(reqParam.toString());
18         given.header("Content-Type", "application/json");
19         given.body(reqParam.toString());
20         Response response = given.request(Method.POST, "/api/users");
21         System.out.println(response.getStatusCode());
22         System.out.println(response.getBody().asString());
23         String resp = response.getBody().asString();
24         Gson gson = new Gson();
25         User user = gson.fromJson(resp, User.class);
26         System.out.println(user);
27         System.out.println(user.getId());
28         System.out.println(user.getName());
29         System.out.println(user.getJob());
30         System.out.println(user.getCreatedAt());
31     }
32 }
33
34
35
```

The console output shows the following logs:

```
@ Javadoc Problems Declaration Console X
<terminal> PostRequest [Java Application] [Library:java/javatoolmachinesjdk-11.0.16.1jdkContentsHome/bin/java (24-Feb-2023, 6:27:28 pm - 6:27:32 pm) [pid: 89365]
{"name":"Revanth","job":"Test engg"}
log4j:WARN No appenders could be found for logger (org.apache.http.impl.conn.BasicClientConnectionManager).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
201
{"name":"Revanth","job":"Test engg","id":"642","createdAt":"2023-02-24T12:57:31.973Z"}
com.test.RA.User@7a9be614
642
Revanth
Test engg
2023-02-24T12:57:31.973Z
```

In the above file the imports are implemented as per requirement. The website url is given and the name and job is also declared and Inside Postrequest class we are writing script to post request . For that we have created jsonobject and required data , also we are changing the setting of content-type to application/json and the print statement is used to print the getstatuscode, user id, name, job.



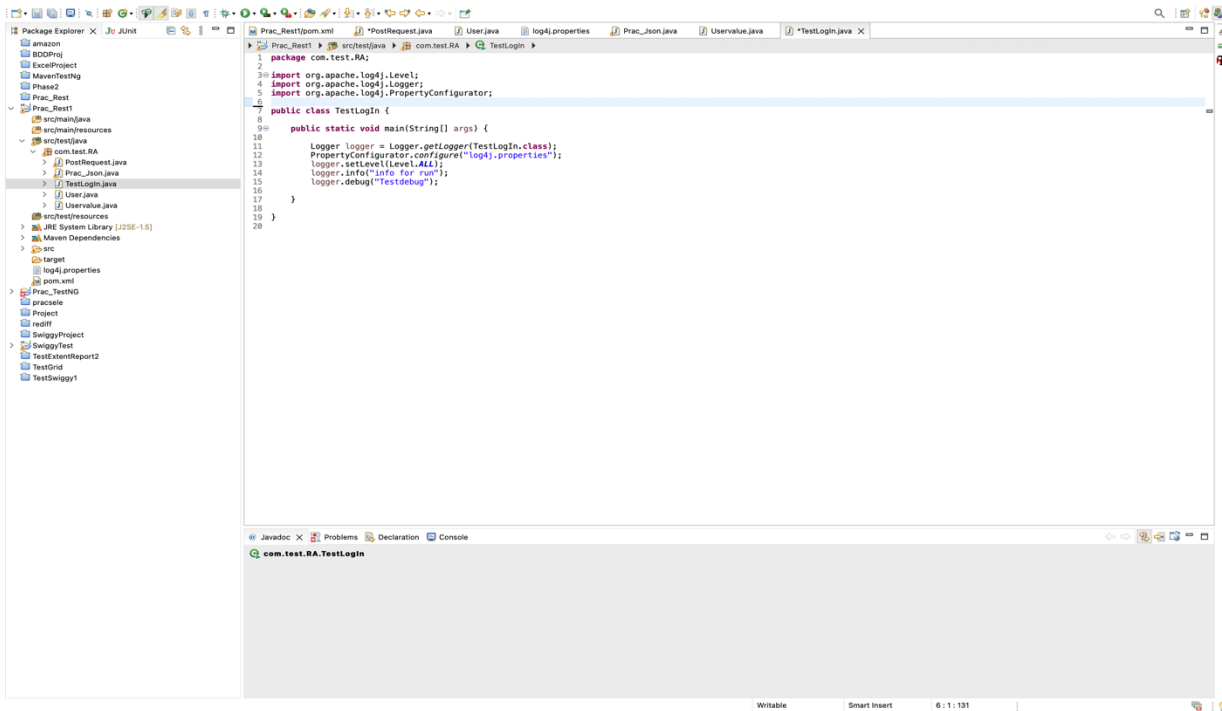
The screenshot displays an IDE with a project explorer on the left and a code editor on the right. The project explorer shows a project named 'Prac_Rest' with various files and folders. The code editor shows the 'Prac_Json.java' file, which contains the following code:

```
1 package com.test.RA;
2
3 import java.util.ArrayList;
4 import org.json.JSONArray;
5 import org.json.JSONObject;
6 import io.restassured.RestAssured;
7 import io.restassured.http.Method;
8 import io.restassured.response.Response;
9 import io.restassured.specification.RequestSpecification;
10
11 public class Prac_Json {
12
13     public static void main(String[] args) {
14
15         RestAssured.baseURI = "https://reqres.in";
16         RequestSpecification given = RestAssured.given();
17         given.header("Content-Type", "application/json");
18         Response response = given.request(Method.GET, "/api/users");
19         System.out.println(response.getStatusCode());
20         System.out.println(response.getBody().asString());
21         String resp = response.getBody().asString();
22         JSONObject object = new JSONObject(response.getBody().asString());
23         JSONArray jsonArray = object.getJSONArray("data");
24         System.out.println(jsonArray.isEmpty());
25         ArrayList<Object> listUsers = new ArrayList<Object>();
26
27         for (int cnt = 0; cnt < jsonArray.length(); cnt++) {
28             for (Object object2 : listUsers) {
29                 System.out.println(object2);
30             }
31         }
32     }
33 }
34
35 }
```

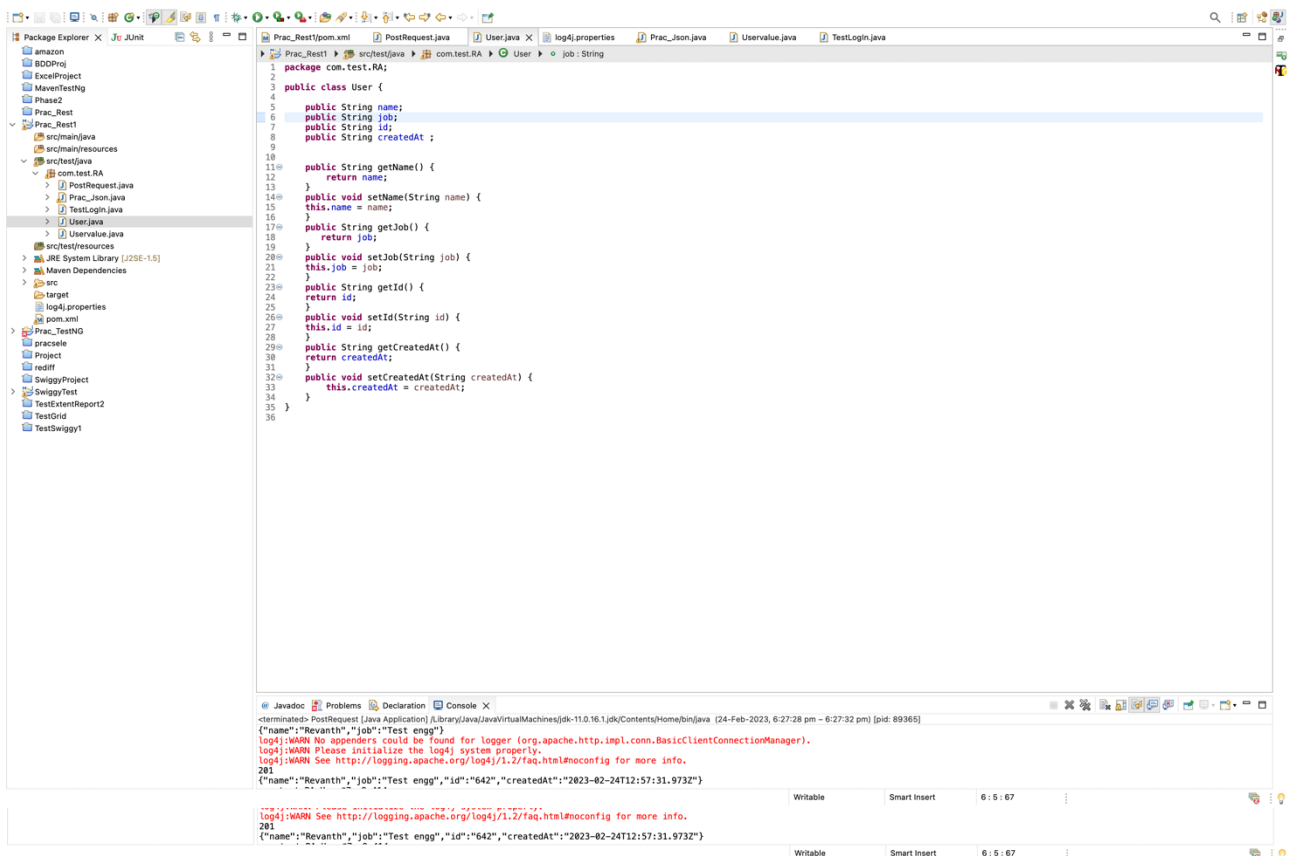
The console output at the bottom shows the execution results:

```
@ Javadoc Problems Declaration Console X
terminated: PostRequest [Java Application] /Library/Java/JavaVirtualMachines/jdk-11.0.16.jdk/Contents/Home/bin/java (24-Feb-2023, 6:07:56 pm - 6:08:01 pm) [pid: 88252]
{"name":"Revanth","job":"Test engg"}
log4j:WARN No appenders could be found for logger (org.apache.http.impl.conn.BasicClientConnectionManager).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
OK
{"name":"Revanth","job":"Test engg","id":"636","createdAt":"2023-02-24T12:38:00.896Z"}
com.test.RA.User@aa9e414
636
Revanth
Test engg
2023-02-24T12:38:00.896Z
```

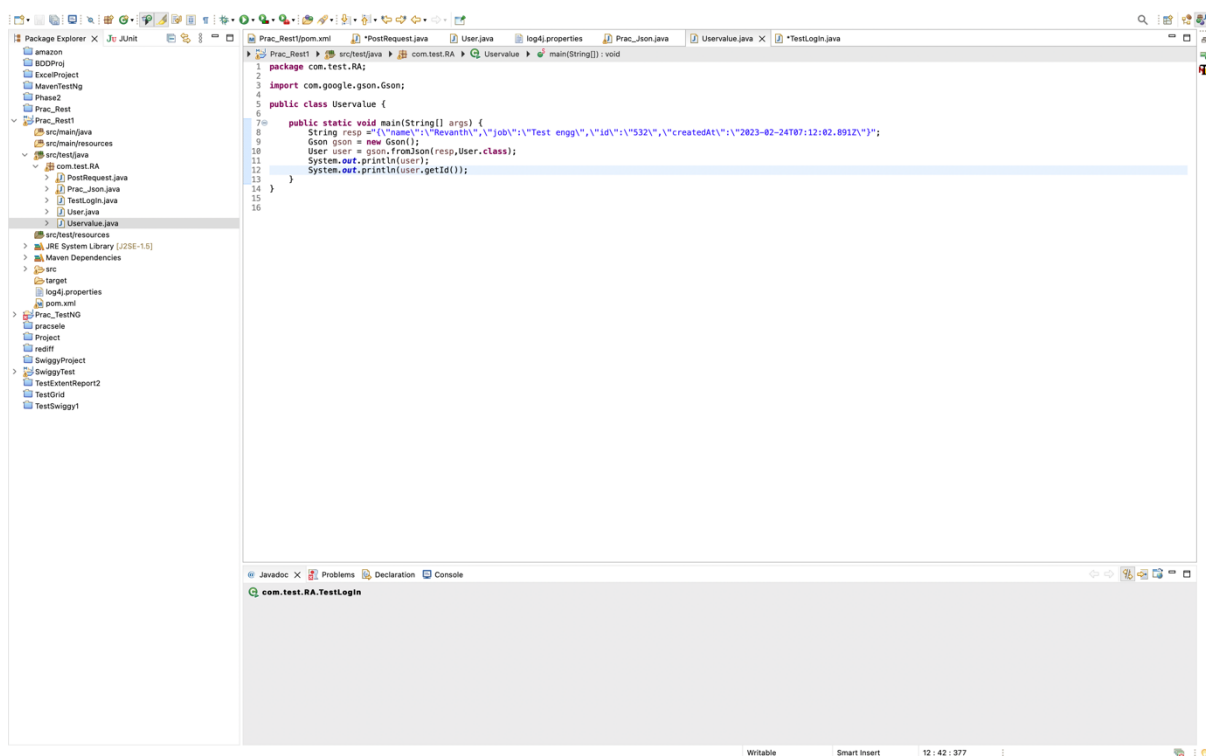
In the above file the imports are implemented as per requirement. In this file we have implemented the website url and getting the response from the website regarding the users list. For getting the users list the get method is used. The json object and json array is also used to print the users list the website.



In this above file the imports are implemented as per requirement. The TestLogIn class is given and the propertyconfiguration is also given in that we have given the log4jproperties file name. It will show the run info and testdebug. we are just creating log4j file so that we can store logs inside it , we are creating one folder name Usersreravichlogs inside this we are storing our logs and also we are just declaring some inbuild properties in log4j.properties file so that we can create log file.

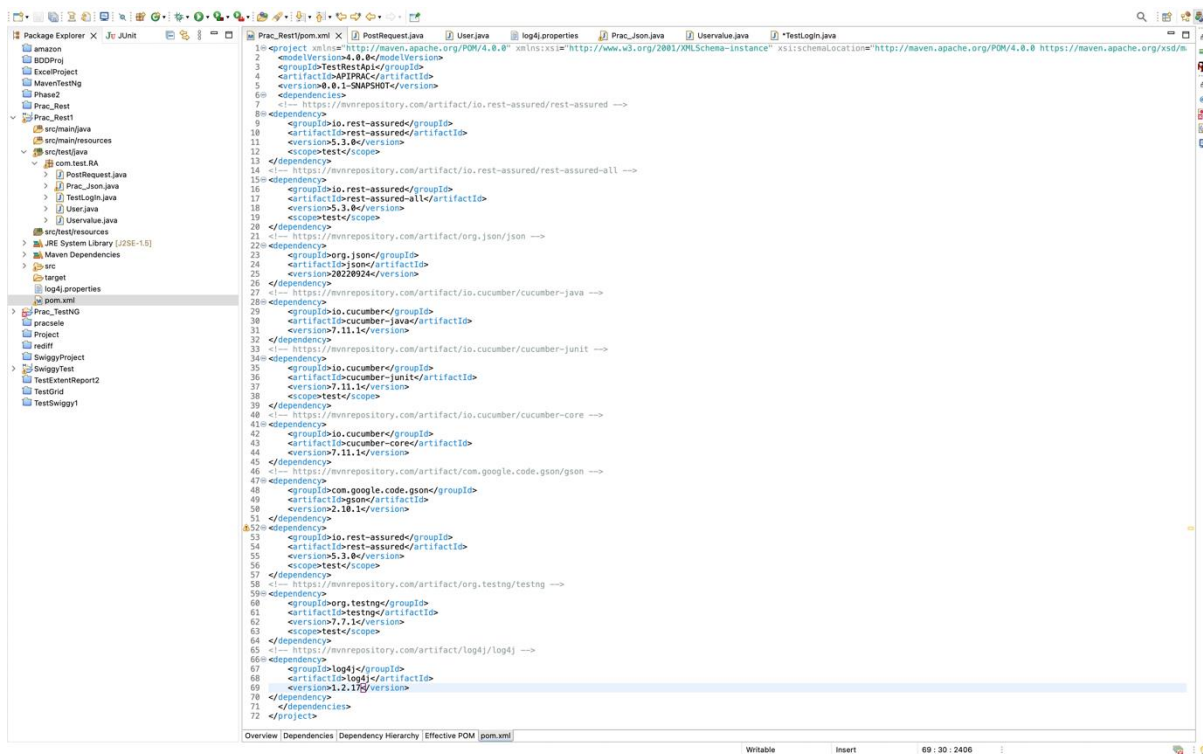


In this above file the imports are implemented as per requirement. The getter and setter method is used here. In this file it will first get the name and set the name and return the value, get the job and set the job and return the value. Then get the id and set the id and return it. And get the createdAt and set the createdAt and return it. Then , we are just calling required functions to post request into the server and with the help of set method we can check that our data has been posted on the server.



In this above file the imports are implemented as per requirement. The data are given the string format. And the user file is linked with it. The name, job, id, createdAt will be printed using the print statement in console.

The dependencies are given in the pom.xml as per requirement. The rest-assured, json, cucumber, google.code.gson, testng, log4j are given.



After this coding part. We have to test using the Jmeter. In that we have to create the thread group and inside that we have to add the HTTP request with the name of getallusers inside that we can get the list of all users. And the regular expression is used with that we can find the id of each and every user and we have to store that data.

In the ForEach Controller, we have to given Id in the field of input variable prefix. In the start index for loop field we have to given 1 and in the End index for loop field we have to given 6. Then it will get the list of six users. After that we have to adds the http Request in the server name we have to give the reqres.in. And in the path we have to given /api/users/2 it will show the details of user 2.

After that we have to add the view result tree, In that we can able to view the request which is sent. And then we have added the graph result in that we can able to view the data, average, median, deviation, throughput in the different color. To get the understanding of the graph.