**Course End Project:**

**Non-Functional Testing Using Postman, REST Assured, and JMeter**

**Source Code:**

**Postman:**

****

**REST Assured:**

**GETRequest.java**

**package** in.reqres.restAssured;

**import** java.io.IOException;

**import** org.testng.annotations.Test;

**import** io.restassured.RestAssured;

**public** **class** GETRequest **extends** TestBase{

@Test

**public** **void** getRequest() **throws** IOException {

RestAssured

.given()

.contentType("application/json")

.when()

.get("https://reqres.in/api/users/2")

.then()

.assertThat()

.statusCode(200)

.log().all();

logger.info("User detials are displayed");

}

}

**POSTRequest.java**

**package** in.reqres.restAssured;

**import** java.util.HashMap;

**import** org.testng.Assert;

**import** org.testng.annotations.BeforeMethod;

**import** org.testng.annotations.Test;

**import** io.restassured.RestAssured;

**import** io.restassured.path.json.JsonPath;

**import** io.restassured.response.Response;

**public** **class** POSTRequest **extends** TestBase{

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

Response response;

JsonPath jsonPath;

@BeforeMethod

**public** **void** createPayLoad() {

map.put("name", "Morpheus");

map.put("job", "painter");

RestAssured.*baseURI* = "https://reqres.in/";

RestAssured.*basePath* = "/api/users";

logger.info("Payload Created Successfully");

}

@Test

**public** **void** postRequest() {

response = RestAssured

.*given*()

.contentType("application/json")

.body(map)

.when()

.post()

.then()

.statusCode(201)

.log().all()

.extract().response();

logger.info("User Created Successfully");

jsonPath = response.jsonPath();

Assert.*assertTrue*(jsonPath.get("name").equals("Morpheus"));

}

}

**TestBase.java**

**package** in.reqres.restAssured;

**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

**JMeter:**

****

**Blazemeter Result:**

[https://a.blazemeter.com/app/?public-token=O6VIln0pHsy1nhzW5L3Y2z7YQF033S5TMykVAu8VEKZznA2ncf#/accounts/1702235/workspaces/1763642/projects/2041967/masters/69299250/summary](https://a.blazemeter.com/app/?public-token=O6VIln0pHsy1nhzW5L3Y2z7YQF033S5TMykVAu8VEKZznA2ncf%23/accounts/1702235/workspaces/1763642/projects/2041967/masters/69299250/summary)

**Github Link:**