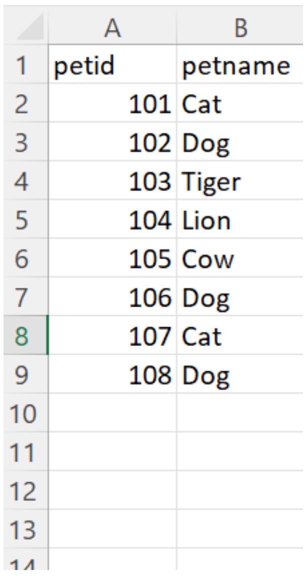
Postman Assignment_001:

Step 1: Create CSV File

- Create a CSV file with two columns: petid and petname



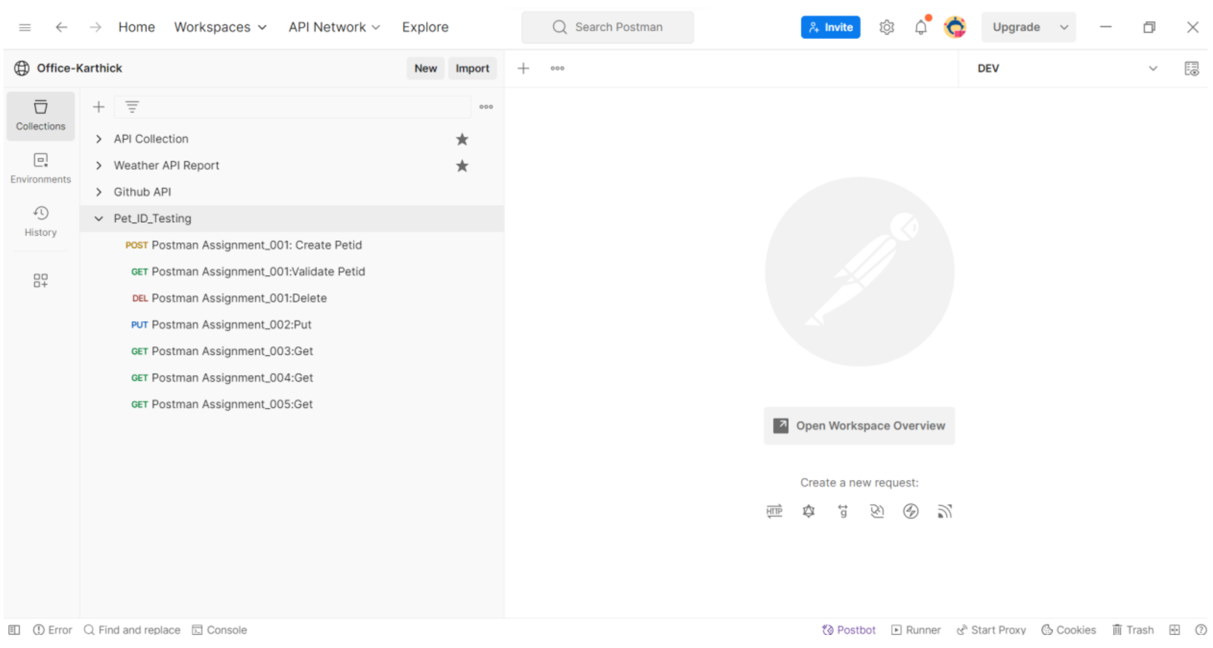
	A	B
1	petid	petname
2	101	Cat
3	102	Dog
4	103	Tiger
5	104	Lion
6	105	Cow
7	106	Dog
8	107	Cat
9	108	Dog
10		
11		
12		
13		

Step 2: Open Postman

- Open Postman.

Step 3: Create Collection

- Create a new collection named "Pet_ID_Testing."

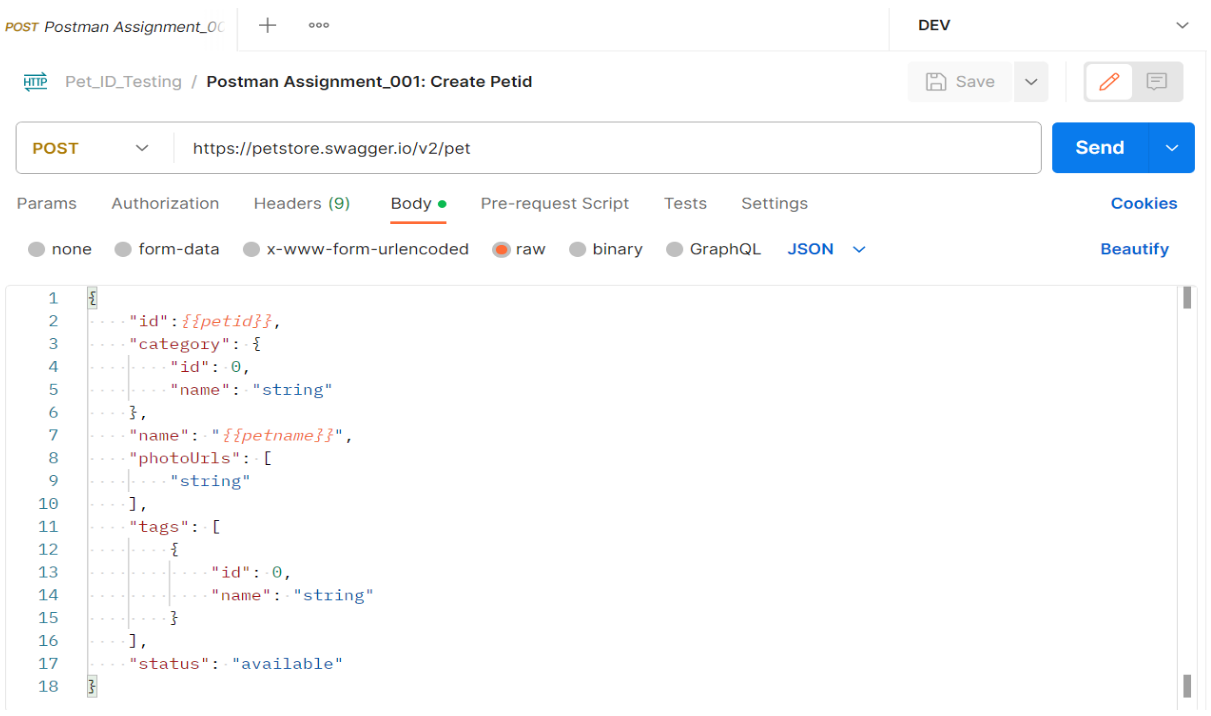


API Collection
Weather API Report
Github API
Pet_ID_Testing
POST Postman Assignment_001: Create Petid
GET Postman Assignment_001:Validate Petid
DEL Postman Assignment_001:Delete
PUT Postman Assignment_002:Put
GET Postman Assignment_003:Get
GET Postman Assignment_004:Get
GET Postman Assignment_005:Get

Step 4: Add POST Request

- Add a new POST service request to the collection.

- URL: https://petstore.swagger.io/v2/pet
- Service Type: POST
- JSON Body: Use the provided JSON body with parameterized petid and petname.



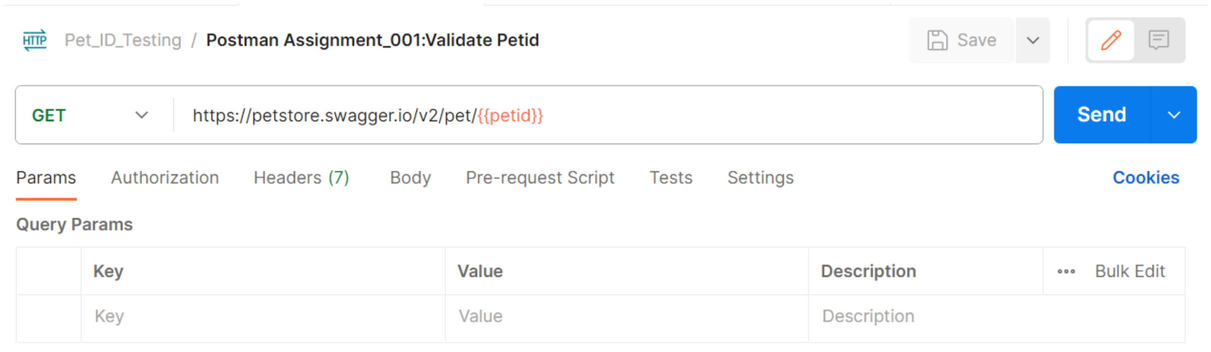
```
{
    "id":{{petid}},
    "category": {
        "id": 0,
        "name": "string"
    },
    "name": "{{petname}}",
    "photoUrls": [
        "string"
    ],
    "tags": [
        {
            "id": 0,
            "name": "string"
        }
    ],
    "status": "available"
}
```

Step 5: Add Tests

- In the Postman Tests section, validate the POST call:
  - Response status code should be 200.
  - Response body should contain text 'available.'

Step 6: Add GET Request

- Add a new GET service request to the collection.
  - URL: https://petstore.swagger.io/v2/pet/{{petid}}


Pet_ID_Testing / **Postman Assignment_001:Validate Petid**

GET https://petstore.swagger.io/v2/pet/{{petid}}

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

Query Params

Key	Value	Description
Key	Value	Description

Step 7: Add Tests for GET Request

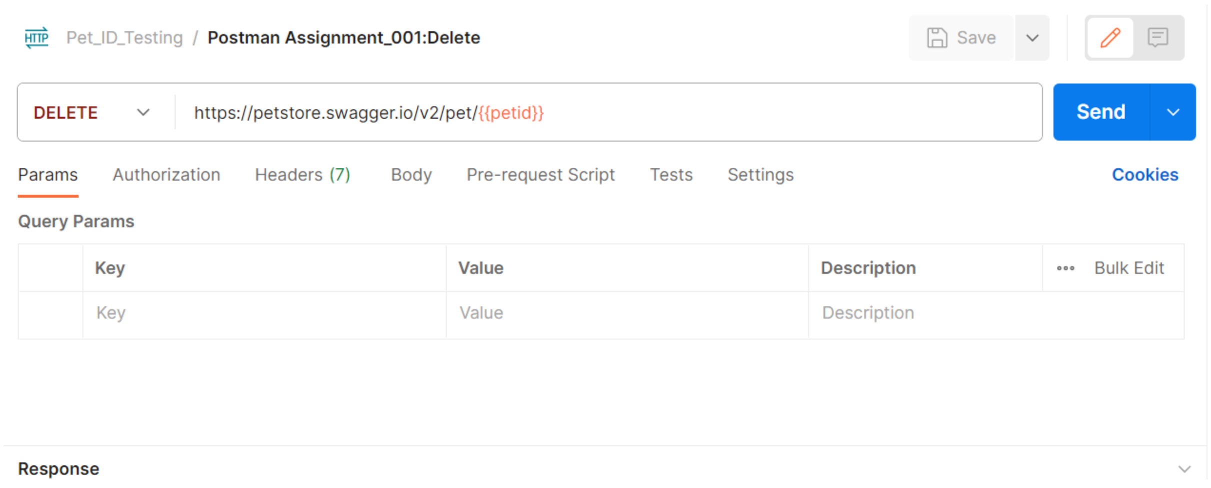
- In the Postman Tests section, validate the GET call:

  - Response status code should be 200.

Step 8: Add DELETE Request

- Add a new DELETE service request to the collection.

  - URL: https://petstore.swagger.io/v2/pet/{{petid}}


Pet_ID_Testing / **Postman Assignment_001:Delete**

DELETE https://petstore.swagger.io/v2/pet/{{petid}}

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

Query Params

Key	Value	Description
Key	Value	Description

Response

Step 9: Add Tests for DELETE Request

- In the Postman Tests section, validate the DELETE call:

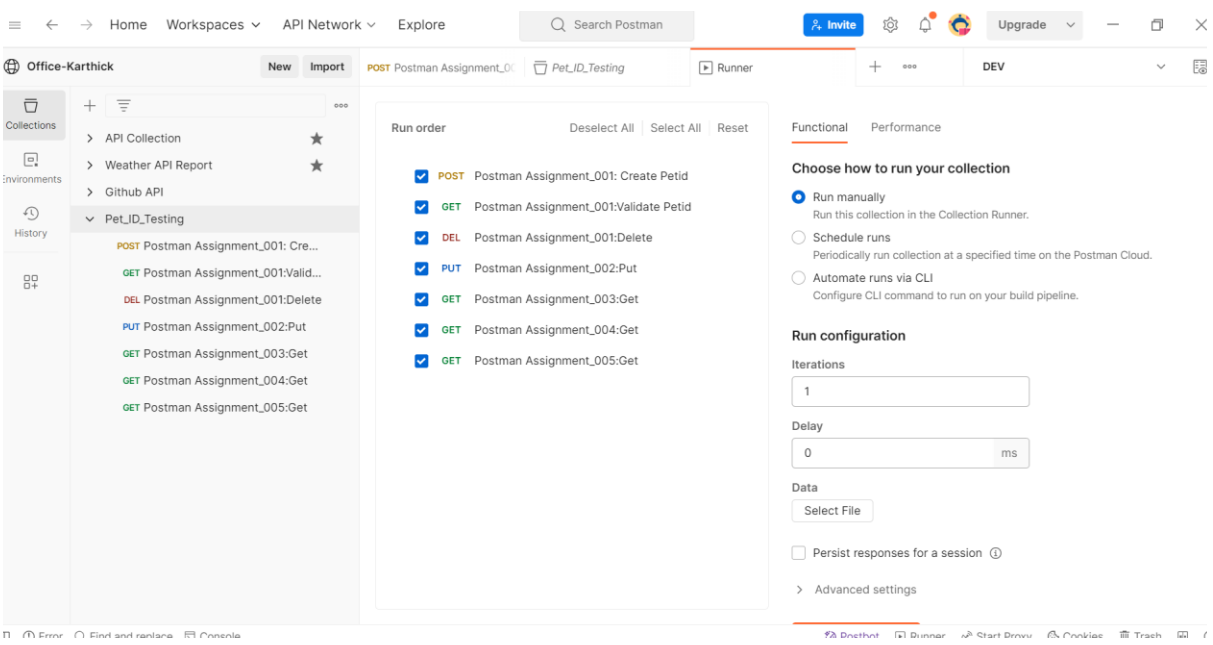
  - Response status code should be 200.

Step 10: Run Collection

- Run the collection.

- Select the CSV file to run the end-to-end scenarios 20 times.

- Check the checkbox for Save Response.



API Collection
Weather API Report
Github API
Pet_ID_Testing
POST Postman Assignment_001: Cre...
GET Postman Assignment_001:Valid...
DEL Postman Assignment_001:Delete
PUT Postman Assignment_002:Put
GET Postman Assignment_003:Get
GET Postman Assignment_004:Get
GET Postman Assignment_005:Get

Run order — Deselect All | Select All | Reset

- ✓ POST Postman Assignment_001: Create Petid
- ✓ GET Postman Assignment_001:Validate Petid
- ✓ DEL Postman Assignment_001:Delete
- ✓ PUT Postman Assignment_002:Put
- ✓ GET Postman Assignment_003:Get
- ✓ GET Postman Assignment_004:Get
- ✓ GET Postman Assignment_005:Get

Functional | Performance

Choose how to run your collection

- ● Run manually — Run this collection in the Collection Runner.
- ○ Schedule runs — Periodically run collection at a specified time on the Postman Cloud.
- ○ Automate runs via CLI — Configure CLI command to run on your build pipeline.

Run configuration

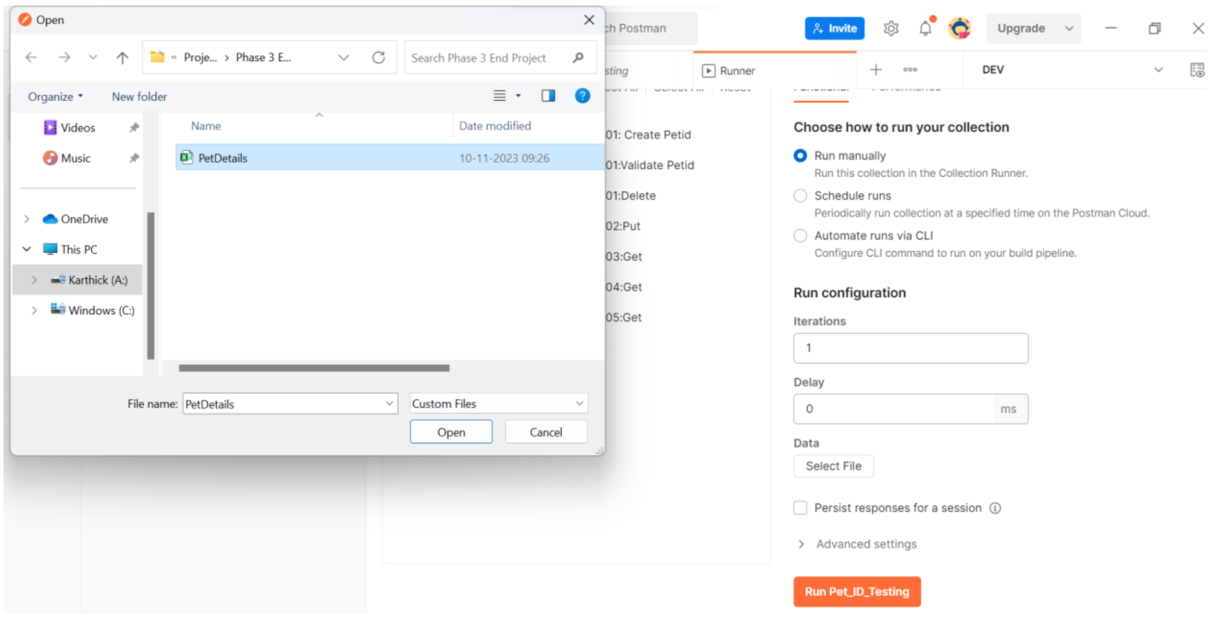
Iterations: 1

Delay: 0 ms

Data: Select File

☐ Persist responses for a session

Advanced settings



Name	Date modified
PetDetails	10-11-2023 09:26

File name: PetDetails — Custom Files

Choose how to run your collection

- ● Run manually — Run this collection in the Collection Runner.
- ○ Schedule runs — Periodically run collection at a specified time on the Postman Cloud.
- ○ Automate runs via CLI — Configure CLI command to run on your build pipeline.

Run configuration

Iterations: 1

Delay: 0 ms

Data: Select File

☐ Persist responses for a session

Advanced settings

Run Pet_ID_Testing

Step 11: Validate and Export

- Open Postman Console and run the collection.

- Validate that all parameterized data [PetID/PetName] is filled with CSV runtime data.

- All status codes should pass.

- Export the collection as a JSON file.

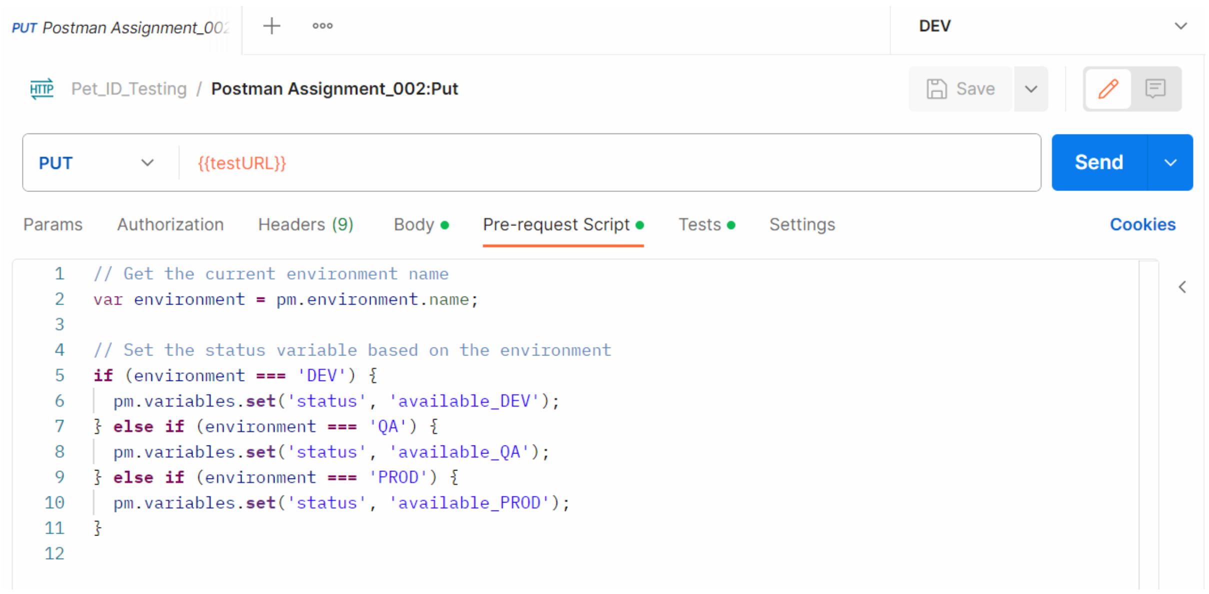
Step 12: Run JSON Collection Using Newman

- Run the JSON collection using the Postman Newman command from cmd/terminal.

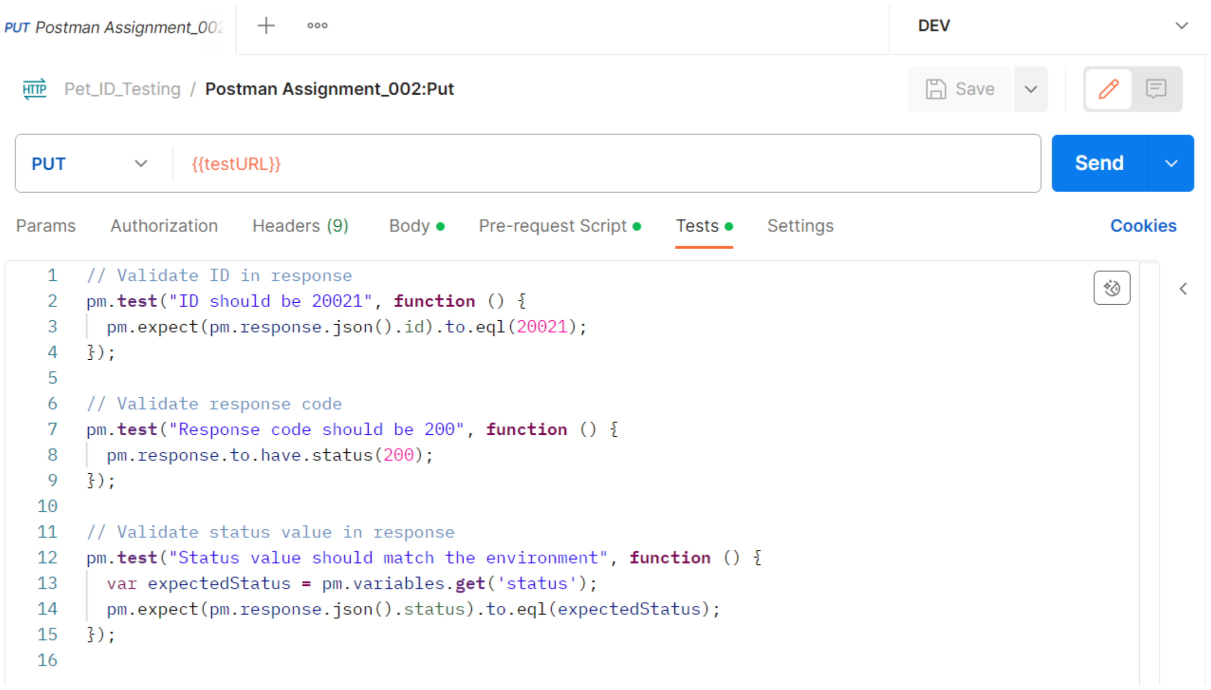
Postman Assignment_002:

Step 1: Create PUT Request

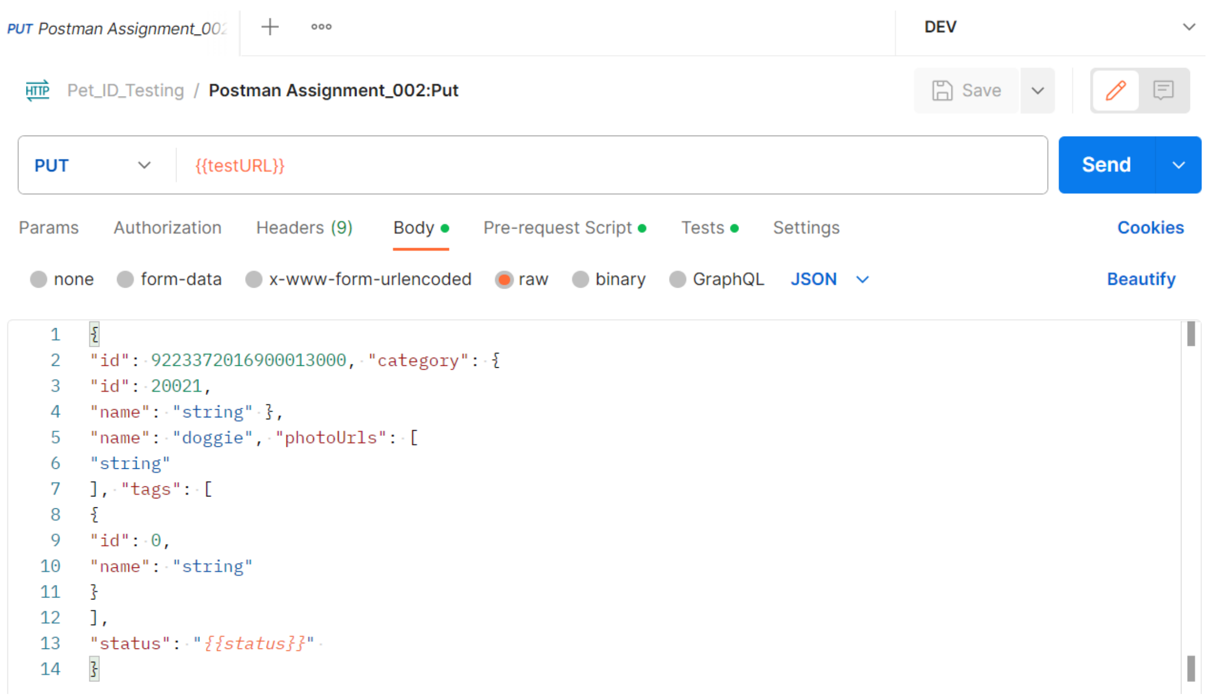
- Add a PUT call to the collection.
- URL: https://petstore.swagger.io/v2/pet
- JSON Body: Use the provided JSON body with parameterized status based on the environment.



```
// Get the current environment name
var environment = pm.environment.name;

// Set the status variable based on the environment
if (environment === 'DEV') {
  pm.variables.set('status', 'available_DEV');
} else if (environment === 'QA') {
  pm.variables.set('status', 'available_QA');
} else if (environment === 'PROD') {
  pm.variables.set('status', 'available_PROD');
}
```



```
// Validate ID in response
pm.test("ID should be 20021", function () {
  pm.expect(pm.response.json().id).to.eql(20021);
});

// Validate response code
pm.test("Response code should be 200", function () {
  pm.response.to.have.status(200);
});

// Validate status value in response
pm.test("Status value should match the environment", function () {
  var expectedStatus = pm.variables.get('status');
  pm.expect(pm.response.json().status).to.eql(expectedStatus);
});
```



```
{
"id": 9223372016900013000, "category": {
"id": 20021,
"name": "string" },
"name": "doggie", "photoUrls": [
"string"
], "tags": [
{
"id": 0,
"name": "string"
}
],
"status": "{{status}}"
}
```

Step 2: Create Global Variable

- Create a global variable for the URL, named testURL, with the value https://petstore.swagger.io/v2/pet.

Step 3: Create Environments

- Create three test environments: DEV, QA, PROD.

- Parameterize the status field in the PUT call JSON body based on the environment.

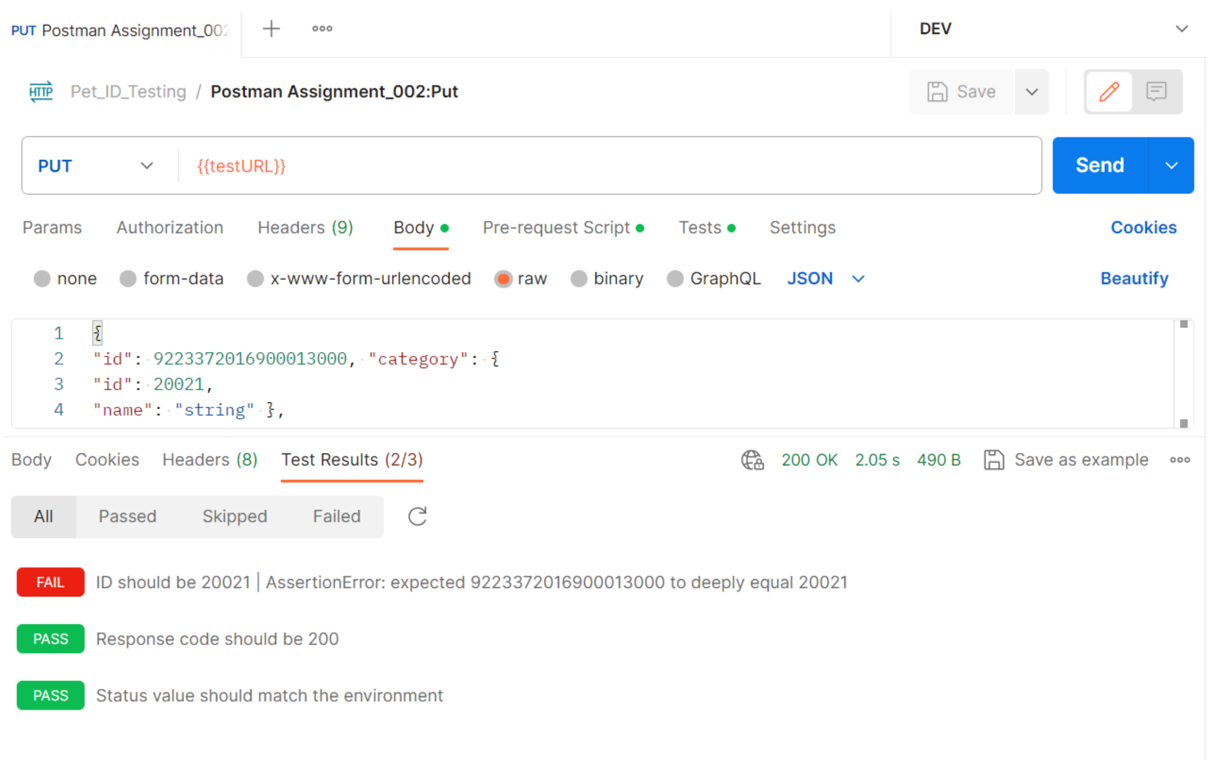
Step 4: Add Tests

- Validate the response:

  - Response status should be 200.

  - Validate id = 20021 in the response.

  - Validate status value changes as per the environment.


PUT Postman Assignment_002

DEV

Pet_ID_Testing / Postman Assignment_002:Put

PUT {{testURL}}

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

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

```
{
"id": 9223372016900013000, "category": {
"id": 20021,
"name": "string" },
```

Body Cookies Headers (8) Test Results (2/3) 200 OK 2.05 s 490 B Save as example

All Passed Skipped Failed

FAIL ID should be 20021 | AssertionError: expected 9223372016900013000 to deeply equal 20021

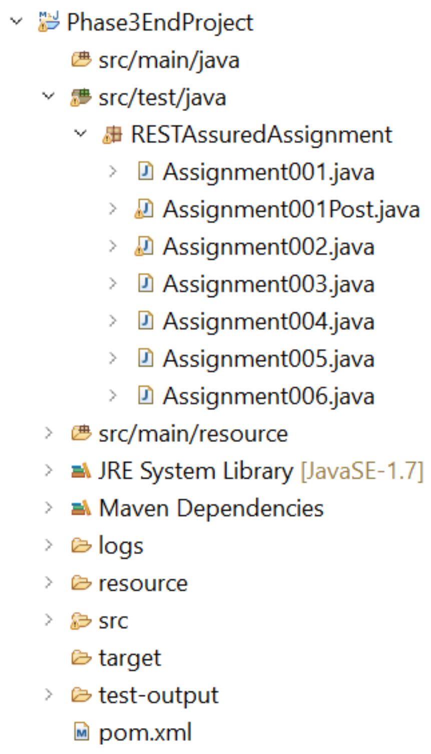
PASS Response code should be 200

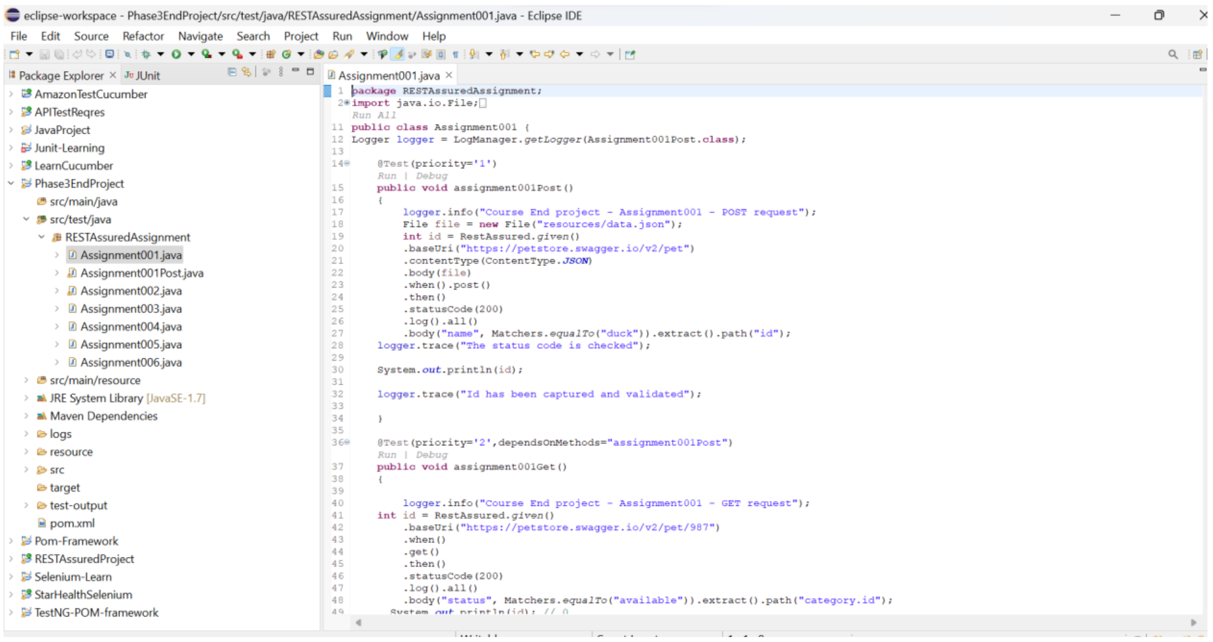
PASS Status value should match the environment

REST Assured Assignment:

Create a Maven project.

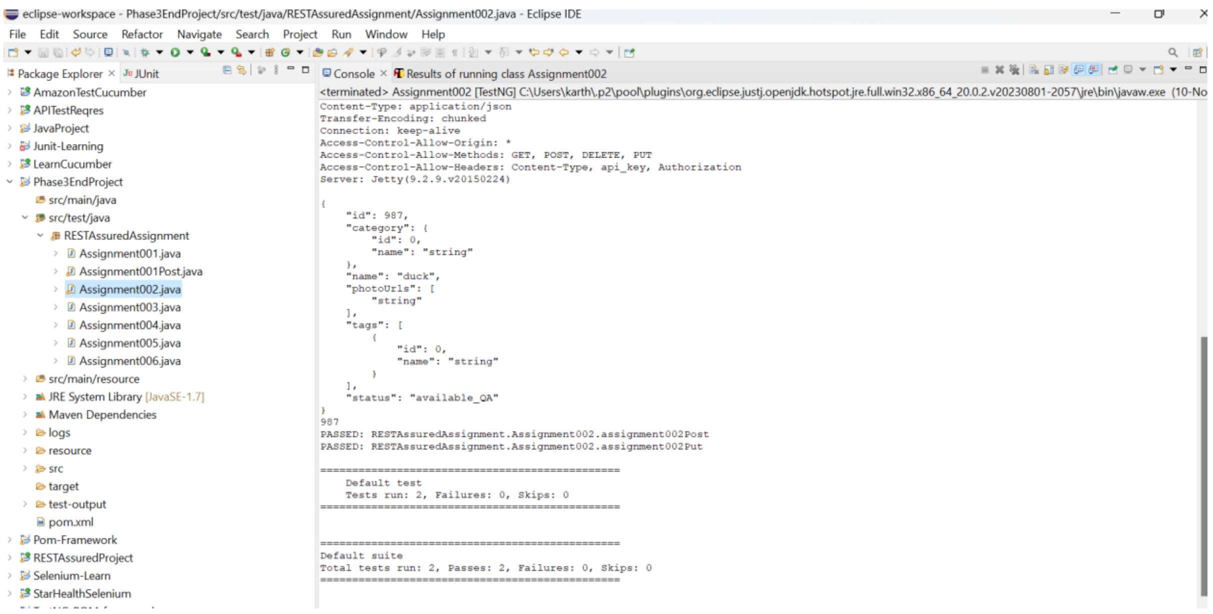
1. Add Dependencies
2. Add Maven dependencies for TestNG and REST Assured.


```
Phase3EndProject
  src/main/java
  src/test/java
    RESTAssuredAssignment
      Assignment001.java
      Assignment001Post.java
      Assignment002.java
      Assignment003.java
      Assignment004.java
      Assignment005.java
      Assignment006.java
  src/main/resource
  JRE System Library [JavaSE-1.7]
  Maven Dependencies
  logs
  resource
  src
  target
  test-output
  pom.xml
```



eclipse-workspace - Phase3EndProject/src/test/java/RESTAssuredAssignment/Assignment001.java - Eclipse IDE

```java
package RESTAssuredAssignment;
import java.io.File;
public class Assignment001 {
Logger logger = LogManager.getLogger(Assignment001Post.class);

    @Test(priority='1')
    public void assignment001Post()
    {
        logger.info("Course End project - Assignment001 - POST request");
        File file = new File("resources/data.json");
        int id = RestAssured.given()
        .baseUri("https://petstore.swagger.io/v2/pet")
        .contentType(ContentType.JSON)
        .body(file)
        .when().post()
        .then()
        .statusCode(200)
        .log().all()
        .body("name", Matchers.equalTo("duck")).extract().path("id");
    logger.trace("The status code is checked");

    System.out.println(id);

    logger.trace("Id has been captured and validated");

    }

    @Test(priority='2',dependsOnMethods="assignment001Post")
    public void assignment001Get()
    {

        logger.info("Course End project - Assignment001 - GET request");
    int id = RestAssured.given()
        .baseUri("https://petstore.swagger.io/v2/pet/987")
        .when()
        .get()
        .then()
        .statusCode(200)
        .log().all()
        .body("status", Matchers.equalTo("available")).extract().path("category.id");
      System.out.println(id); // 0
```



eclipse-workspace - Phase3EndProject/src/test/java/RESTAssuredAssignment/Assignment002.java - Eclipse IDE

```
<terminated> Assignment002 [TestNG] C:\Users\karth\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_20.0.2.v20230801-2057\jre\bin\javaw.exe (10-No
Content-Type: application/json
Transfer-Encoding: chunked
Connection: keep-alive
Access-Control-Allow-Origin: *
Access-Control-Allow-Methods: GET, POST, DELETE, PUT
Access-Control-Allow-Headers: Content-Type, api_key, Authorization
Server: Jetty(9.2.9.v20150224)

{
    "id": 987,
    "category": {
        "id": 0,
        "name": "string"
    },
    "name": "duck",
    "photoUrls": [
        "string"
    ],
    "tags": [
        {
            "id": 0,
            "name": "string"
        }
    ],
    "status": "available_QA"
}
987
PASSED: RESTAssuredAssignment.Assignment002.assignment002Post
PASSED: RESTAssuredAssignment.Assignment002.assignment002Put

===============================================
    Default test
    Tests run: 2, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 2, Passes: 2, Failures: 0, Skips: 0
===============================================
```

JMeter Assignment:

Step 1: Open JMeter

- Open JMeter.

Step 2: Validate HTTP Authentication

- Validate https://httpbin.org/basic-auth/user/passwd using HTTP Authentication Manager.

Step 3: Validate Response with JSON Assertion

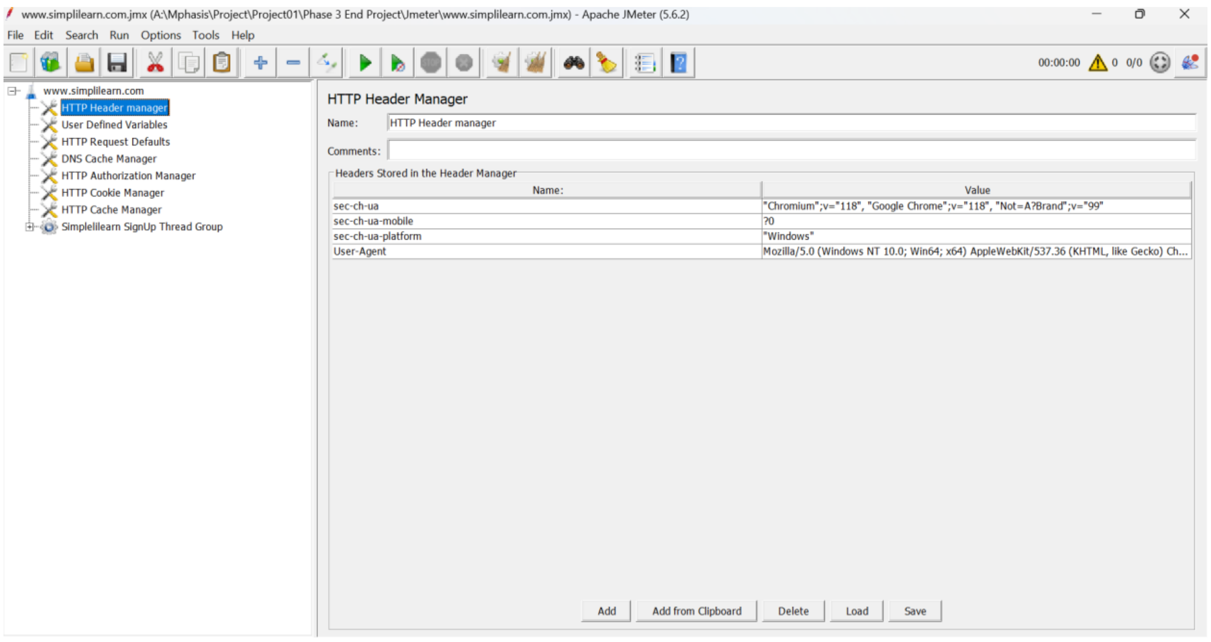
- Validate that the response is a JSON file using JSON assertion.

Step 4: Hit Simplilearn URL

- Using HTTP sampler, hit https://www.simplilearn.com and validate xpath.

Step 5: Increase Thread Count

- Increase the thread count and perform load testing for lead validation.



www.simplilearn.com.jmx (A:\Mphasis\Project\Project01\Phase 3 End Project\Jmeter\www.simplilearn.com.jmx) - Apache JMeter (5.6.2)

File Edit Search Run Options Tools Help

00:00:00 0 0/0

- www.simplilearn.com
  - HTTP Header manager
  - User Defined Variables
  - HTTP Request Defaults
  - DNS Cache Manager
  - HTTP Authorization Manager
  - HTTP Cookie Manager
  - HTTP Cache Manager
  - Simplelilearn SignUp Thread Group

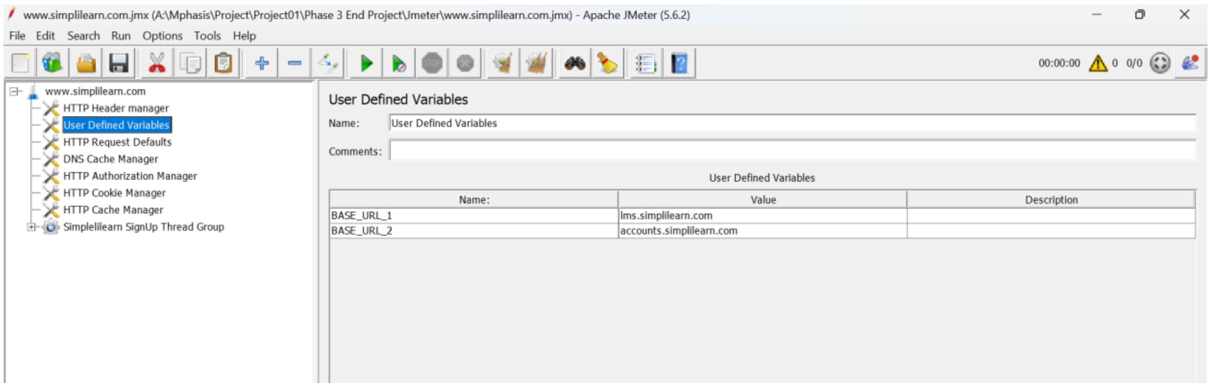
HTTP Header Manager

Name: HTTP Header manager

Comments:

Headers Stored in the Header Manager

Name:	Value
sec-ch-ua	"Chromium";v="118", "Google Chrome";v="118", "Not=A?Brand";v="99"
sec-ch-ua-mobile	?0
sec-ch-ua-platform	"Windows"
User-Agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Ch...



www.simplilearn.com.jmx (A:\Mphasis\Project\Project01\Phase 3 End Project\Jmeter\www.simplilearn.com.jmx) - Apache JMeter (5.6.2)

File Edit Search Run Options Tools Help

00:00:00 0 0/0

- www.simplilearn.com
  - HTTP Header manager
  - User Defined Variables
  - HTTP Request Defaults
  - DNS Cache Manager
  - HTTP Authorization Manager
  - HTTP Cookie Manager
  - HTTP Cache Manager
  - Simplelilearn SignUp Thread Group

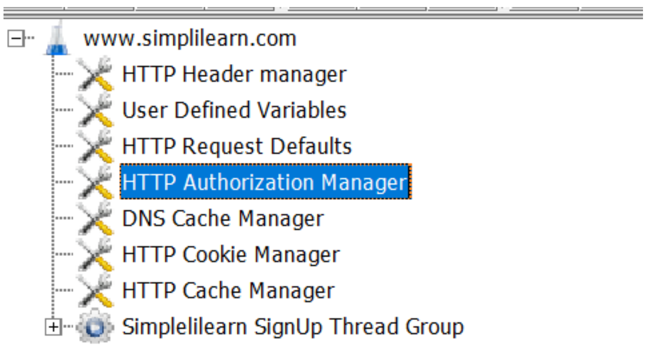
User Defined Variables

Name: User Defined Variables

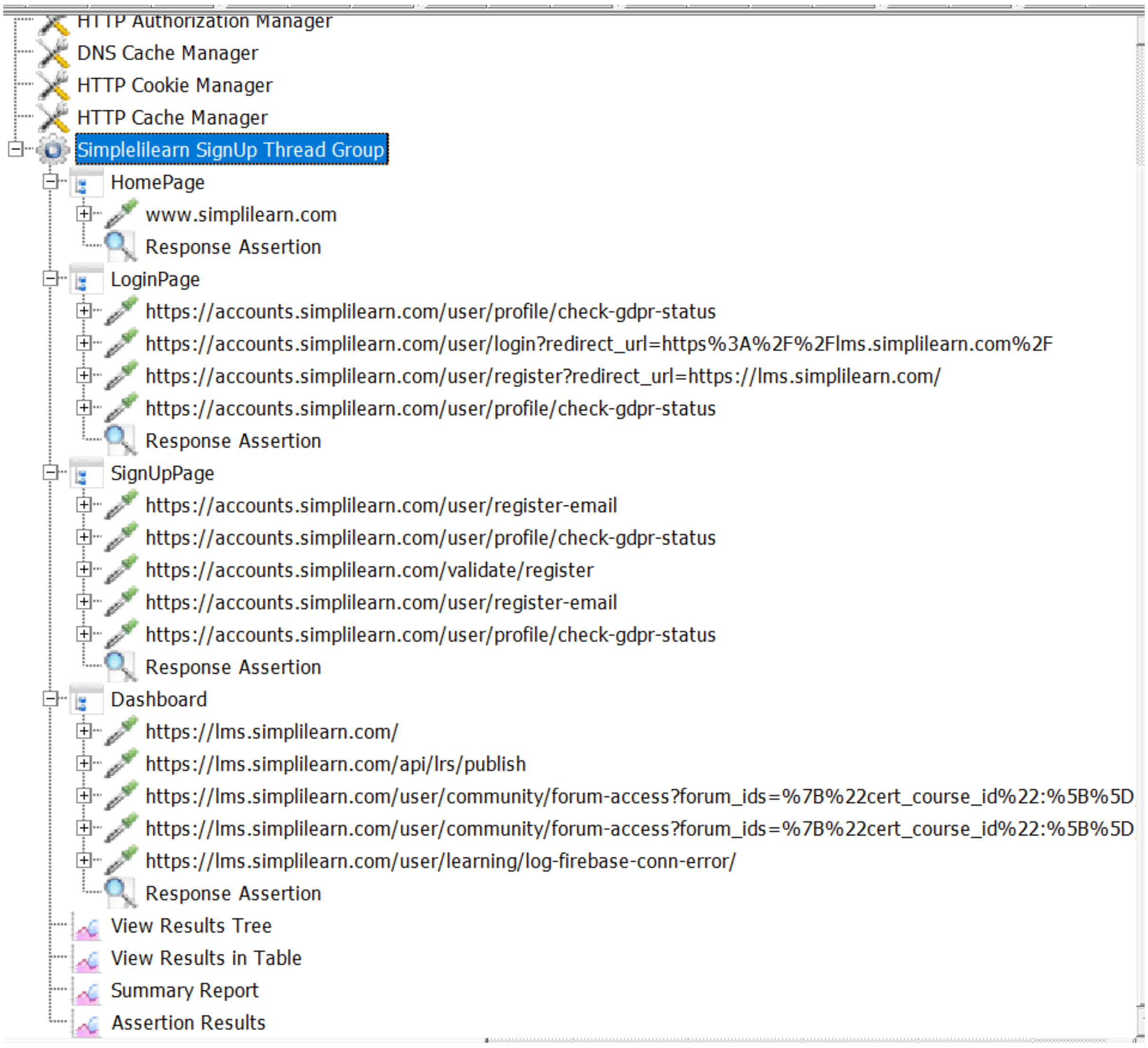
Comments:

User Defined Variables

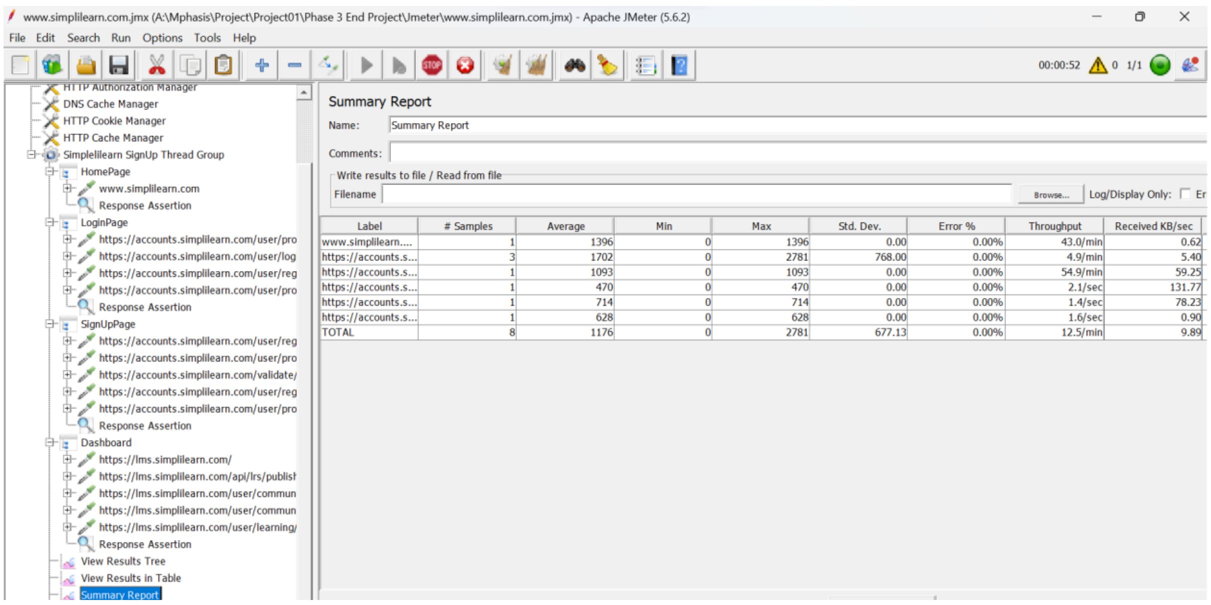
Name:	Value	Description
BASE_URL_1	lms.simplilearn.com	
BASE_URL_2	accounts.simplilearn.com	



- www.simplilearn.com
  - HTTP Header manager
  - User Defined Variables
  - HTTP Request Defaults
  - HTTP Authorization Manager
  - DNS Cache Manager
  - HTTP Cookie Manager
  - HTTP Cache Manager
  - Simplelilearn SignUp Thread Group



```
HTTP Authorization Manager
DNS Cache Manager
HTTP Cookie Manager
HTTP Cache Manager
Simplelilearn SignUp Thread Group
    HomePage
        www.simplilearn.com
        Response Assertion
    LoginPage
        https://accounts.simplilearn.com/user/profile/check-gdpr-status
        https://accounts.simplilearn.com/user/login?redirect_url=https%3A%2F%2Flms.simplilearn.com%2F
        https://accounts.simplilearn.com/user/register?redirect_url=https://lms.simplilearn.com/
        https://accounts.simplilearn.com/user/profile/check-gdpr-status
        Response Assertion
    SignUpPage
        https://accounts.simplilearn.com/user/register-email
        https://accounts.simplilearn.com/user/profile/check-gdpr-status
        https://accounts.simplilearn.com/validate/register
        https://accounts.simplilearn.com/user/register-email
        https://accounts.simplilearn.com/user/profile/check-gdpr-status
        Response Assertion
    Dashboard
        https://lms.simplilearn.com/
        https://lms.simplilearn.com/api/lrs/publish
        https://lms.simplilearn.com/user/community/forum-access?forum_ids=%7B%22cert_course_id%22:%5B%5D
        https://lms.simplilearn.com/user/community/forum-access?forum_ids=%7B%22cert_course_id%22:%5B%5D
        https://lms.simplilearn.com/user/learning/log-firebase-conn-error/
        Response Assertion
    View Results Tree
    View Results in Table
    Summary Report
    Assertion Results
```



www.simplilearn.com.jmx (A:\Mphasis\Project\Project01\Phase 3 End Project\Jmeter\www.simplilearn.com.jmx) - Apache JMeter (5.6.2)

```
HTTP Authorization Manager
DNS Cache Manager
HTTP Cookie Manager
HTTP Cache Manager
Simplelilearn SignUp Thread Group
    HomePage
        www.simplilearn.com
        Response Assertion
    LoginPage
        https://accounts.simplilearn.com/user/pro
        https://accounts.simplilearn.com/user/log
        https://accounts.simplilearn.com/user/reg
        https://accounts.simplilearn.com/user/pro
        Response Assertion
    SignUpPage
        https://accounts.simplilearn.com/user/reg
        https://accounts.simplilearn.com/user/pro
        https://accounts.simplilearn.com/validate/
        https://accounts.simplilearn.com/user/reg
        https://accounts.simplilearn.com/user/pro
        Response Assertion
    Dashboard
        https://lms.simplilearn.com/
        https://lms.simplilearn.com/api/lrs/publist
        https://lms.simplilearn.com/user/commun
        https://lms.simplilearn.com/user/commun
        https://lms.simplilearn.com/user/learning/
        Response Assertion
    View Results Tree
    View Results in Table
    Summary Report
```

Summary Report

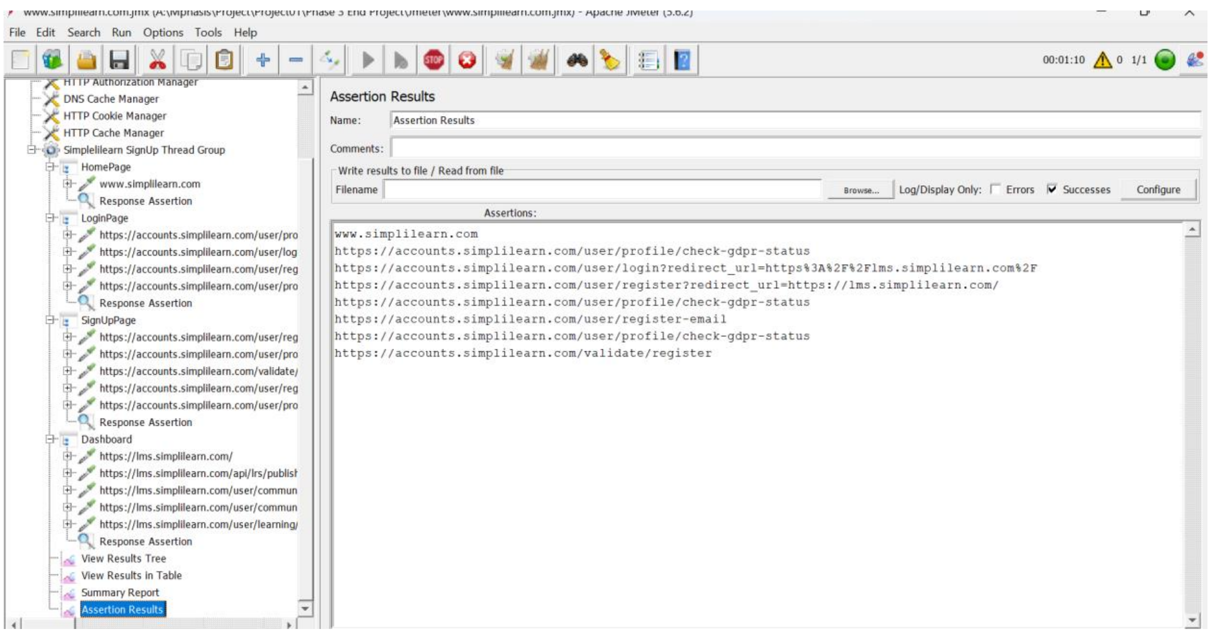
Name: Summary Report

Comments:

Write results to file / Read from file

Filename

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec
www.simplilearn....	1	1396	0	1396	0.00	0.00%	43.0/min	0.62
https://accounts.s...	3	1702	0	2781	768.00	0.00%	4.9/min	5.40
https://accounts.s...	1	1093	0	1093	0.00	0.00%	54.9/min	59.25
https://accounts.s...	1	470	0	470	0.00	0.00%	2.1/sec	131.77
https://accounts.s...	1	714	0	714	0.00	0.00%	1.4/sec	78.23
https://accounts.s...	1	628	0	628	0.00	0.00%	1.6/sec	0.90
TOTAL	8	1176	0	2781	677.13	0.00%	12.5/min	9.89



```
www.simplilearn.com.jmx (A:\Mphasis\Project\Project01\Phase 3 End Project\Jmeter\www.simplilearn.com.jmx) - Apache JMeter (5.6.2)
```

File
Edit
Search
Run
Options
Tools
Help
00:01:10
0
1/1
HTTP Authorization Manager
DNS Cache Manager
HTTP Cookie Manager
HTTP Cache Manager
Simplelilearn SignUp Thread Group
HomePage
www.simplilearn.com
Response Assertion
LoginPage
https://accounts.simplilearn.com/user/pro
https://accounts.simplilearn.com/user/log
https://accounts.simplilearn.com/user/reg
https://accounts.simplilearn.com/user/pro
Response Assertion
SignUpPage
https://accounts.simplilearn.com/user/reg
https://accounts.simplilearn.com/user/pro
https://accounts.simplilearn.com/validate/
https://accounts.simplilearn.com/user/reg
https://accounts.simplilearn.com/user/pro
Response Assertion
Dashboard
https://lms.simplilearn.com/
https://lms.simplilearn.com/api/lrs/publist
https://lms.simplilearn.com/user/commun
https://lms.simplilearn.com/user/commun
https://lms.simplilearn.com/user/learning/
Response Assertion
View Results Tree
View Results in Table
Summary Report
Assertion Results

Assertion Results

Name: Assertion Results

Comments:

Write results to file / Read from file

Filename: Browse... Log/Display Only: [ ] Errors [✓] Successes Configure

Assertions:

```
www.simplilearn.com
https://accounts.simplilearn.com/user/profile/check-gdpr-status
https://accounts.simplilearn.com/user/login?redirect_url=https%3A%2F%2Flms.simplilearn.com%2F
https://accounts.simplilearn.com/user/register?redirect_url=https://lms.simplilearn.com/
https://accounts.simplilearn.com/user/profile/check-gdpr-status
https://accounts.simplilearn.com/user/register-email
https://accounts.simplilearn.com/user/profile/check-gdpr-status
https://accounts.simplilearn.com/validate/register
```

GitHub repository : https://github.com/Karthick-Office/Project01.git

Non-Functional Testing Using Postman, REST Assured, and JMeter Project is present under the "Phase 3 End Project" folder in the My GitHub repositor