**Rest Assured API Automation Assignment**

# Objective

The purpose of this assignment is to understand and practice API automation using Rest Assured by covering various HTTP methods, payload handling techniques, response validations, and advanced concepts such as API chaining and authentication mechanisms.

# Assignment Topics

## 1. HTTP Methods

\*\*Question 1:\*\* Write a Rest Assured test script to perform the following operations:

Send a GET request to the endpoint `https://jsonplaceholder.typicode.com/posts` and validate the status code is 200.

Extract the first post’s `title` and log it.

\*\*Question 2:\*\* Perform a POST request to the endpoint https://jsonplaceholder.typicode.com/posts` with the following payload:

```json

{

"title": "foo",

"body": "bar",

"userId": 1

}

```

Validate that the status code is 201 and extract the `id` from the response.

## 2. Different Ways of Sending Payload to the Request

\*\*Question 1:\*\* Demonstrate how to send a payload using:

JSON as a string in the body.

External JSON file.

Java Map object converted to JSON.

Validate the status code and response body for each method when sending a POST request to `https://jsonplaceholder.typicode.com/posts`.

## 3. Response Validation - Response Body and Other Parameters

\*\*Question 1:\*\* Send a GET request to `https://jsonplaceholder.typicode.com/posts/1` and validate:

The status code is 200.

The response contains the key `userId` with value 1.

Log the response time and ensure it’s less than 2 seconds.

## 4. Response Validation - JSON Schema Validator

\*\*Question 1:\*\* Validate the response of a GET request to https://jsonplaceholder.typicode.com/posts/1` against the following schema:

```json

{

"$schema": "http://json-schema.org/draft-07/schema#",

"type": "object",

"properties": {

"userId": { "type": "integer" },

"id": { "type": "integer" },

"title": { "type": "string" },

"body": { "type": "string" }

},

"required": ["userId", "id", "title", "body"]

}

```

## 5. Response Validation Against Expected List

\*\*Question 1:\*\* Perform a GET request to `https://jsonplaceholder.typicode.com/posts` and validate:

The response contains exactly 100 posts.

The list of `userId`s contains only values between 1 and 10.

## 6. Response Mapped to POJO Class

\*\*Question 1:\*\* Create a POJO class for the following JSON structure:

```json

{

"userId": 1,

"id": 1,

"title": "Sample Title",

"body": "Sample Body"

}

```

Perform a GET request to `https://jsonplaceholder.typicode.com/posts/1`, map the response to the POJO class, and validate the `userId` is 1.

## 7. Hamcrest Matchers

\*\*Question 1:\*\* Perform a GET request to `https://jsonplaceholder.typicode.com/posts/1` and validate:

- The `title` starts with "s" (case-insensitive).

- The `body` contains the word "quia".

## 8. Headers

\*\*Question 1:\*\* Perform a GET request to `https://jsonplaceholder.typicode.com/posts` and validate:

The `Content-Type` header is `application/json; charset=utf-8`.

Log all the headers in the response.

## 9. Cookies

\*\*Question 1:\*\* Perform a GET request to an endpoint of your choice that returns cookies. Validate the presence of at least one cookie and log its details.

## 10. Logs

\*\*Question 1:\*\* Demonstrate how to log:

- Request details.

- Response details.

Perform a GET request to `https://jsonplaceholder.typicode.com/posts` with logging enabled for both request and response.

## 11. Different Types of Authentications

\*\*Question 1:\*\* Demonstrate Basic Authentication by sending a GET request to `https://postman-echo.com/basic-auth` with the username `postman` and password `password`. Validate the response status code is 200 and the response body contains `{ "authenticated": true }`.

\*\*Question 2:\*\* Demonstrate Bearer Token Authentication by sending a GET request to an endpoint of your choice. Use a sample bearer token and validate the response.

## 12. API Chaining

\*\*Question 1:\*\* Perform the following:

Send a POST request to `https://jsonplaceholder.typicode.com/posts` with a payload and extract the `id`.

Use the extracted `id` to perform a GET request to fetch the post details.

Validate that the details in the GET response match the payload sent in the POST request.

## Submission Guidelines

1. Provide all scripts with meaningful comments.

2. Include screenshots of execution results.

3. Provide a summary report highlighting validations performed.

\*\*Due Date:\*\* [**10/01/2025**]

Good luck with the assignment!