**API Automation Document**

**RestAssured**

RestAssured is an API library through which we can automate RestAPI.

we will send different types of API, also we will able to validate the responses.

**Pre-Requisites :**

1 JAVA 9+ & Eclipse

2 TestNG

3 Maven (Automatically comes with Eclipse)

**steps:**

1 Install and open the Eclipse

2 Add TESTNG inside the Eclipse

1 click on help

2 go to eclipse marketplace

3 Search for TESTNG

4 After installation Restart your Eclipse

How to Add Maven Project Steps:

1 Go to File

2 click on new

3 Select project

4 Inside Maven select Maven Project

5 Select first check box & click on next

6 specify group\_id & artifact\_id and click on finish.

**Add Dependencies**

1 Add dependencies inside the pom.xml

2 goto MVNRepository add the dependencies what ever you want

3 Add following Dependencies

1 rest-assured

2 json-path

3 json

4 gson

5 testng

6 scriberjava-apis (generated ramdom-test data)

7 json-schema validation

8 xml schema validation

**HTTP Request**

Following are the HTTP request we used in API request

1 GET

2 POST

3 PUT

4 DELETE

**Use below link for Demo**

* reqres.in

Under the src/test java we have to create all the packages & classes

**Gerkin Keyword**

1 Given

2 When

3 Then

**1 Given**

Whenever we are sending for the API request there are some pre-requisites are required.

- Content-Type

- Set Cookies

- add Auth

- add Param

- Set Headers info

**2 When**

- We have to specifies request type

- GET, POST, PUT, DELETE all the request URL, we have to keep inside the WHEN section

**3 Then**

- All validation comes under the then section

- Validate Status code

- Extract response

- Extract headers & Cookies

- Extract Response Body

For Given, When, Then key word we need to import static packages (because its not normal package)

import org.testng.annotations.Test;

import static io.restassured.RestAssured.\*;

import static io.restassured.matcher.RestAssuredMatchers.\*;

When we import all the above packages need to add "static" keyword

**Format to write Gerkin Keywords**

given()

.when()

.then()

Which ever method start first no need to add "." dot before that method

**API Methods**

* Create User

-- Whenever we want to create any user then use "HASHMAP"

-- In hashmap value should be stored in "key" & "value" pair

HashMap data = new HashMap();

data.put("name", "Scott");

**Post request payload in Diff ways**

1 HashMap

2 Using org.json

3 Using POJO

4 USing External JSON File

**1 HashMap**

-- hash map is java collection we can store the data in the form of key & value

-- some time key having multiple value(one array contains single value)

-- If we have small set of data then we use hashmap

HashMap data = new HashMap();

data.put("name", "Scott");

data.put("location", "France");

data.put("phone", "123456");

String courseArr[] = {"C", "C++"};

data.put("courses", courseArr);

**2 Using org.json**

- we need to add "org.json" dependency

- Need to create JSON Object

JSONObject data=new JSONObject();

- Add data like HashMAp

.body(data.toString()) // we can not pass data directly inside the body we have to convert that dat into string format

* above entire data convert into string format
* then after converting it will be send data in json format

**3 Using POJO**

-- post request body creation using POJO class (plain old java object)

-- some time key having multiple value(one array contains single value)

-- here we use encapsulation concept

-- wrapping of variables and methods into one single class

-- here we use getter & setter methods (to generate/create data)

-- We have to create one class in same package Pojo-postRequest

-- we have to add required variable

-- this class contains variable getter and setter methods

-- here we require 4 variable name, location, phone ( this 3 are primitive variable ) & cources (array variable)

-- click on all variables & then click on Generate

-- to assign and retrive the data we have to write getter & setter

-- select above variables go to source click on "generate getter & setter"

-- setter methos will take one parameter & assign the value to the class variable

-- it will take phone value and assign to phone class variable

-- setter assign values to the variable

-- getter methos will get value from the variable

**4 USing External JSON File**

-- what ever data you want to send along with post request just past it inside Body.json file

-- suppose i have data in my external file so i can use that data for post request

-- to get the data from external file first we have to open the file

File f=new File(".\\body.json");

-- also we have to specify location of the file

-- here dot is representing current location of file

----- Also we need to add one json File inside the project

1 right click on project

2 select file

3 write the file name xyz.json

4 write / paste code inside that file

**Path & Query Parameter**

**Path Para**- Directions in the web link to find a specific place on the website

Query Para - Like telling a website what your lokking for , ass at the end of a websites

https://reqres.in/api/users?page=2&id=5

https://reqres.in called domain it will 1st reach to the server

wherever there is / or .in/.com/.org this is called domain

api is a path

users is also path

page=2 after ? what ever is there called query parameter

path para we have define our variable

page id exactly shown present on page

**Example :**

.get("https://reqres.in/api/{maypath}")

-- Correct the URL and use the path parameter

-- only query para we have to write inside the { } , query para goes along with path oara

-- along with request we have to pass parameter , no need to add further url the query para goes

along with request

**Cookies & Headers**

**Cookies -** Data Stored on a User device, to track state & authentication in API request

suppose you want to capture data from header and cookies

cookies are dynamically changes, we can't expect exact value of the cookies everytime it's keep changing

whenever value keep changing means our functionality working fine

whenever u get data from server then new cookies will be generated

**Headers -** Carry metadata about the request like Content-Type , authorization between the client &

server (message that guide how the API & use talk)

1 some info of header keep changing not every field

2 server info , content encoding, content type, cache control, (info most of the time constant) will not change

date when it will expire, P#P info will be change

3 whatever details are constant we have to validate that "content type & content encoding most of the time we have to validate

& some time server name

4 when we simple write log().all all header info will be displayed no need to write big code most of the time this is not used

5 also we can write log().headers() it will return only headers

**Logging**

-- we can use this log method along with then methods there are many sub methods inside the log

-- log().all() it prints all detailed information about the API request and response

Examples :

.log().all(); //if you want all response /info

.log().body(); //if you want specific body info

.log().cookies(); //if you want to print only cookie

.log().headers(); //print only headers

**Parsing Response Body**

* **JSON OBJECT**

-- Parsing response body is the technical process of extracting meaningful data from the raw response,

you get from API.

-- So that we can process analyze & use that data in our application

-- First we need to create one JSON file in your Computer i.e book.json

-- Extention of that file should be JSON

-- then on the same location go to search bar

-- write cmd & press enter

-- enter json.server filename.JSON

-- press enter

-- it will shows Resource name

http://localhost:3000/book

-- Do not close the cmd until you done your program/code on eclise

-- we didnt get output when we close cmd

Also folow the below steps :

1 Go to chrome

2 Search jsonpathfinder

3 paste our entire test/json code

4 what ever path we expect select that one

example : Moby Dick

5 Automatically it will generate one path for that

6 Copy that path and paste it in our program (dont copy x)

**ParsingXMLResponse**

- parsing means : what ever data we want we can extract it & compare it with our expected values

- Here we validate json response by using XmlPath method

- Also we do status code validation using Asser class

- XmlPath method is used for additional validation

- for that we have to create object of XmlPath method

T**OSTRING**- if you want to convert data in string format

**ASSTRING-** if you want to convert entire response as string

**File Upload & Download**

1 two things are important when you upload the file

2 multipart & form data

3 we have to use multipart & content Type to upload the file

4 First we have to create a file object with the path to the file

File myfile = new File("C:\\Users\\Admin\\Desktop\\Workspace\\API\\File.txt");

For File Download

1 If we want to download the file, send the request in POST method

2 If we check the response with the help of postman, we get what ever content inside the file that

will displayed

3 also in response we gey one URL file file Download copy that URL from there.

**JSON\_Schema\_Validation**

API JSON\_Schema\_Validation ensure that data is correctly formatted for sharing

Schema- Is like blueprint that shows how data should be organize/look & formatted when its send &

received (data shared between different progrmas & services)

we have to follows below steps :

1 we have to create one .Json file inside the src/main/recources

2 right click on src/main/recources select file

3 provide the name of that file with .json extenson

4 go to chrome

5 search json to json schema converter select one from there

link : https://transform.tools/json-to-json-schema

6 Paste our Json code there , automatically jsonschema will be generate

7 copy that jsonschema code from there and paste in our file

8 We have to use assertThat() to validate JSON\_Schema

**XML\_Schema\_Validation**

we have to follows below steps :

1 we have to create one .xsd file inside the src/main/recources

2 right click on src/main/recources select file

3 provide the name of that file with .xsd extenson

4 go to chrome

5 search xml to xml schema converter select one from there

link : https://easycodeforall.com/XMLTOXSDSUBMIT

https://www.convertsimple.com/convert-xml-to-xsd-xml-schema/

6 Paste our Json code there , automatically xmlschema will be generate

7 copy that xmlschema code from there and paste in our file

8 We have to use assertThat() to validate JSON\_Schema

**Serialization**

1 when we send request it will go along with body in json format, response also e get in json format

2 serialization means POJO to JSON

3 pojo is converting into json and json gothrought our request along with the

4 Here we use POHO so we follow encapsulation here so we need to create one class inside same package

5 need to add get & set methods

6 This class contains variable getter and setter methods

7 Here we require 4 variable name, location,phone(this 3 are primitive variable)& cources(array variable)

8 Select all variables & then click on Generate

9 To assign and retrive the data we have to write getter & setter

10 select above variables go to source click on "generate getter & setter"

11 setter methos will take one parameter & assign the value to the class variable

12 it will take phone value and assign to phone class variable

13 setter assign values to the variable

14 getter methos will get value from the variable

**Deserialization**

1 Deserialization means json to POJO

2 json data should be in string format

3 Here we used jackson package along with obj mapper class

4 why this process implementes json is light weight format And that more secure

5 we cant send diff java obj through n/w thats not safe

6 so we always transfer the data in json format

7 In restassured by default serialization and deserialization process is there

**Faker Library**

1 The Faker library generates fake data for testing APIs without using real user information

2 we can generate randam /dummy data

3 need to add dependancy search java faker and click on 1st link copy dependancy from there

4 when we run program again and again it will generate different data

**Authentication**

**Authentication** - we need to check user is valid or not

**Authorization -** user is valid but we need to check permission/access for that user

Restassured supported different authentication

1 basic

2 digest

3 preempive

4 bearer token

5 auth 1.0, 2.0

6 API Key

**1 basic**

-- Basic Authentication in APIs requires sending a username and password

-- postman goes this url to support basic ,digest & preemptive auth

-- depends of which auth we choosen according to that algorithm for that auth will be triggered

Example

.auth().basic("postman", "password")

**2 digest**

-- digest Authentication in APIs requires sending a username and password

-- postman goes this url to support basic ,digest & preemptive auth

-- depends of which auth we choosen according to that algorithm for that auth will be triggered

Example

.auth().digest("postman", "password")

**3 preempive**

-- preempive Authentication in APIs requires sending a username and password

-- postman goes this url to support basic ,digest & preemptive auth

-- depends of which auth we choosen according to that algorithm for that auth will be triggered

Example

.auth().preemptive().basic("postman", "password") //combination of 2 type of authentication

1 above all 3 requires username & password but algorithms are different

2 depends on customer reqirement developer have do some kind of api they will implement what kind of

authentication is required for that api its depends on complexity of app

3 suppose u r app is more sensitive and lots of security is required that time devloper go with auth 2

authentication

4 auth 2 authentication is very very typical and no body can break that authentication

bcz 1st we haveto creake a key based on that we create the token multiple things are there

5 some appilcation prefers basic digest authentication

6 gmail yahoo we need to provide our gmail password

7 According to customer requirement developer build authentication same kind of authentication we have

to simulate to test the api

**4 bearer token**

1 Bearer tokens in APIs are like special keys that allow you to securely access and do specific things

with online services without revealing your full credentials.

How to create bearer token

1 Login to github.com

2 Goto Settings

3 Goto Developer settings

4 Select personal access token

5 Click on token(classic)

6 select generate new token

7 select generate new token (classic)

8 insert your password

9 add comment

10 Click on generate

**5 auth 1.0, 2.0**

-- for auth1 authentication we have to pass multiple parameter

-- developer will give us information how to generate Auth1 authentication

-- for auth one four parameters are required

1 consumerKey

2 consumerSecrate

3 accestoken

4 tokenSecrate

-- all these details required to generate auth1 authentication

Example :

.auth().oauth("consumerKey", "consumerSecrat", "accestoken", "tokenSecrate")

-- for auth2 neeed to pass access token

-- auth 2.0 authentication we have only one paramenter i.e token

-- auth 2.0 will reduce no of parameters

Example :

.auth().oauth2("ghp\_GuhGkUIASSJt36FVkVgTgW7lHISjNU45bXJZ")

-- Above details required to generate auth2 authentication

**6 API Key**

-- developer know about how to generate qpi key and authentication

-- developer will guid us accordingly

-- API Key Authentication is like a password that clients use when talking to a website or application's

behind-the-scenes code.

Example :

.queryParam("appid", "")

appid is api key

api key is pass as part of query parameter

**Chaining**

-- API chaining is using the results of one API as the input for another API to perform complex tasks.

-- API chaining is when you use multiple APIs one after another to get a specific job done, like getting

data from one API and using it as input for another API to achieve a more complex task.

**follow the below steps :**

1 First we have to create four different class

2 for update ,get, delete, create user

3 After that right click on the package

4 Select TestNg

5 Then select Convert to TestNg & provide the name with .xml & click on finish

6 One xml file will be generated with our all classes

7 Here we can do parallerl testing

8 If we want to or run multiple classes to gether the we have to use Chaining

9 Click on run then select TestNG Suite

Example :

<classes>

<class name="Chaining.CreateUser"/>

<class name="Chaining.GetUser"/>

<class name="Chaining.UpdateUSer"/>

<class name="Chaining.DeleteUser"/>

</classes>

Supoose we want to run all the classes seperately then follow below process

1 First we have to create four different class

2 After that right click on the package

3 Select TestNg

4 Then select Convert to TestNg & provide the name with .xml & click on finish

5 One xml file will be generated

6 We have to provide name for each test i.e <test name="Test3">

Example :

<test name="Test1">

<classes>

<class name="Chaining.CreateUser"/>

</classes>

</test>

<test name="Test2">

<classes>

<class name="Chaining.GetUser"/>

</classes>

</test>

7 Click on run then select TestNG Suite

8 All the classes runs seperately