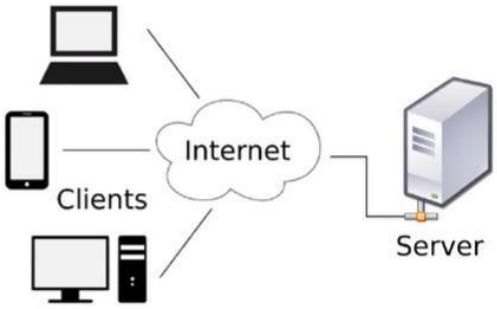
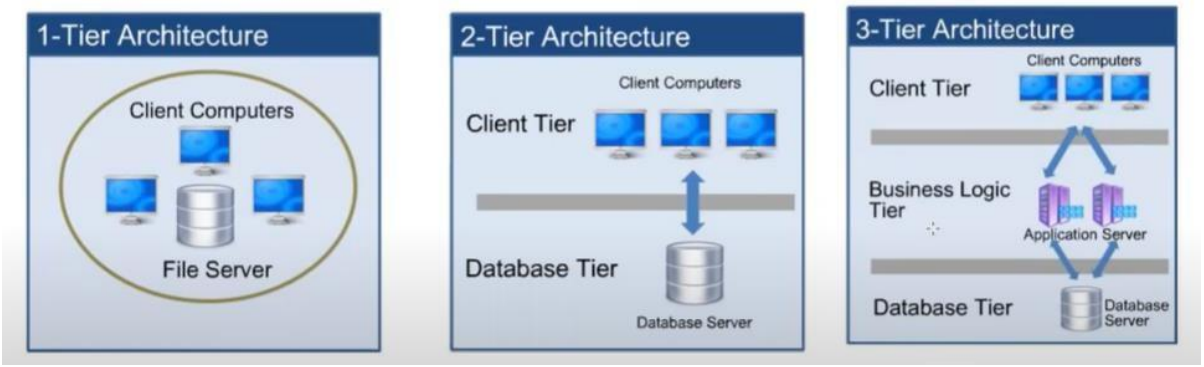
POSTMAN GUIDE

What is Client and Server?

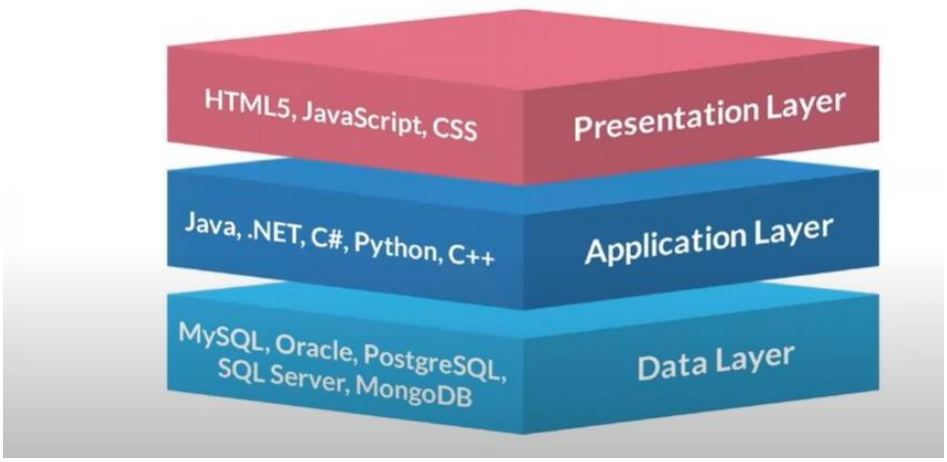
- What is a Client?
- A client is a computer hardware device or software that accesses a service made available by a server. The server is often (but not always) located on a separate physical computer.
- What is a Server?
- A server is a physical computer dedicated to run services to serve the needs of other computers. Depending on the service that is running, it could be a file server, database server, home media server, print server, or web server.

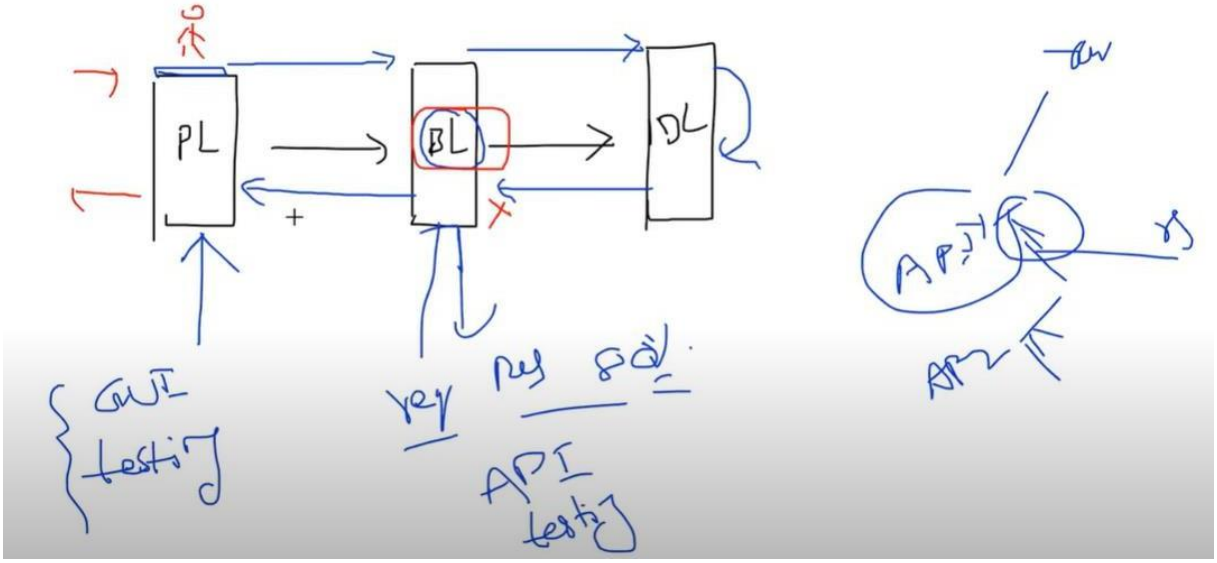


Client/Server Architecture



Client/Server Architecture





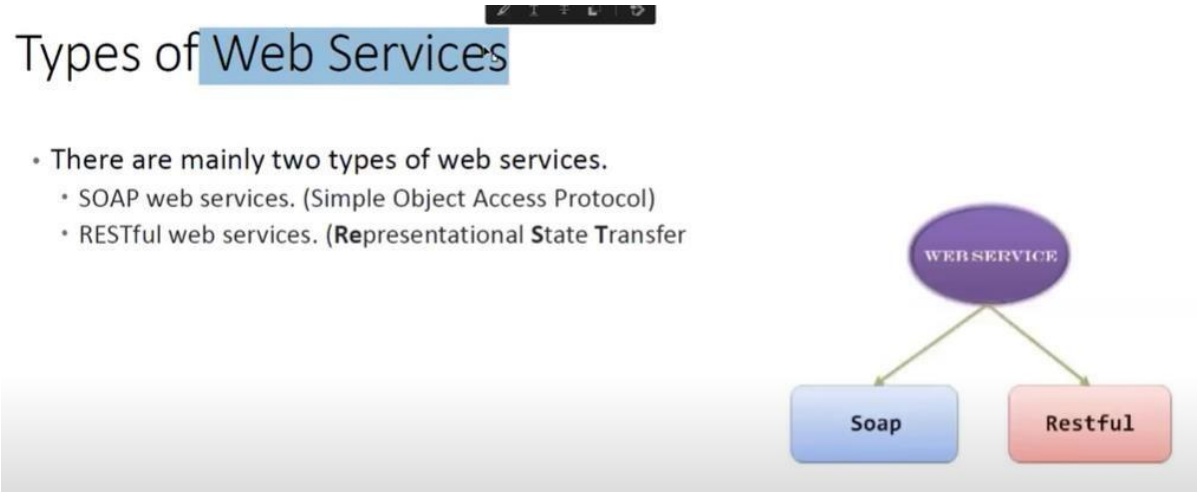
What is an API?

- API is means **A**pplication **P**rogramming **I**nterface.
- It enables communication and data exchange between two separate software systems.
- A software system implementing an API contains functions/sub-routines which can be executed by another software system.

API = When we have an API in our local environment than its called API
Web Service = When we move API from local to production environment then the API is called Web service. Web service is an API which is wrap in HTTP
Note: All services are API but all APIs are not web services

What is Web Service?

- What is a Web Service: Web Service - service available over the web
- Enables communication between applications over the web
- Provides a standard protocol/format for communication
- **Why we use it?**
- Platform independent communication - using web services two different applications (implementation) can talk to each other and exchange data/information



SOAP API

SOAP is a very old and complex technology and its only supports XML for communication. It supports only post requests only.

RESTful API

RESTful is the latest technology and its supports multiple ways to communicate like XML, Text, HTML, and JSON, and it supports multiple operations like Post, put, get, set, etc. Most of companies are using RESTful API nowadays as it's the latest and has more features

All the tasks will be saved in the workspace which will be created in Gmail or logged in the account

COLLECTION

The collection is a combination of multiple requests

STRUCTURE OF URL

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

https://reqres.in = Domain
/api/users = **Path Parameters**
page=2 = **Query Parameters**

Note: Sometime query parameter will be there and sometime not

Create The First GET Request

1. Click on created collection
2. Click on + sign
3. Select the GET from the dropdown
4. Put the API URL
5. Hit Send

Once we hit it, we can see the response below in the body section. We can see the response in multiple ways XML, HTML, Text, JSON, etc. There are also multiple representations of response as well as Pretty, Raw, Preview & Visualize

We need to validate all of below mentioned

Once we are done with the then we can save the request into the desired collection

GET = Get all data
POST = Its creating new record and always need to create request body or request payload
PUT = Its updating the existing record and always need to create request body or request payload
DELETE = Delete data all or specific

Every request will get us response in JSON format

From the browser we can only send GET request. For all other request we need to use postman

After getting the response from the API if the status code is 200 , 201 then its mean our request was successful and all other codes mean there is some problem

PARAM = Parameters of the request

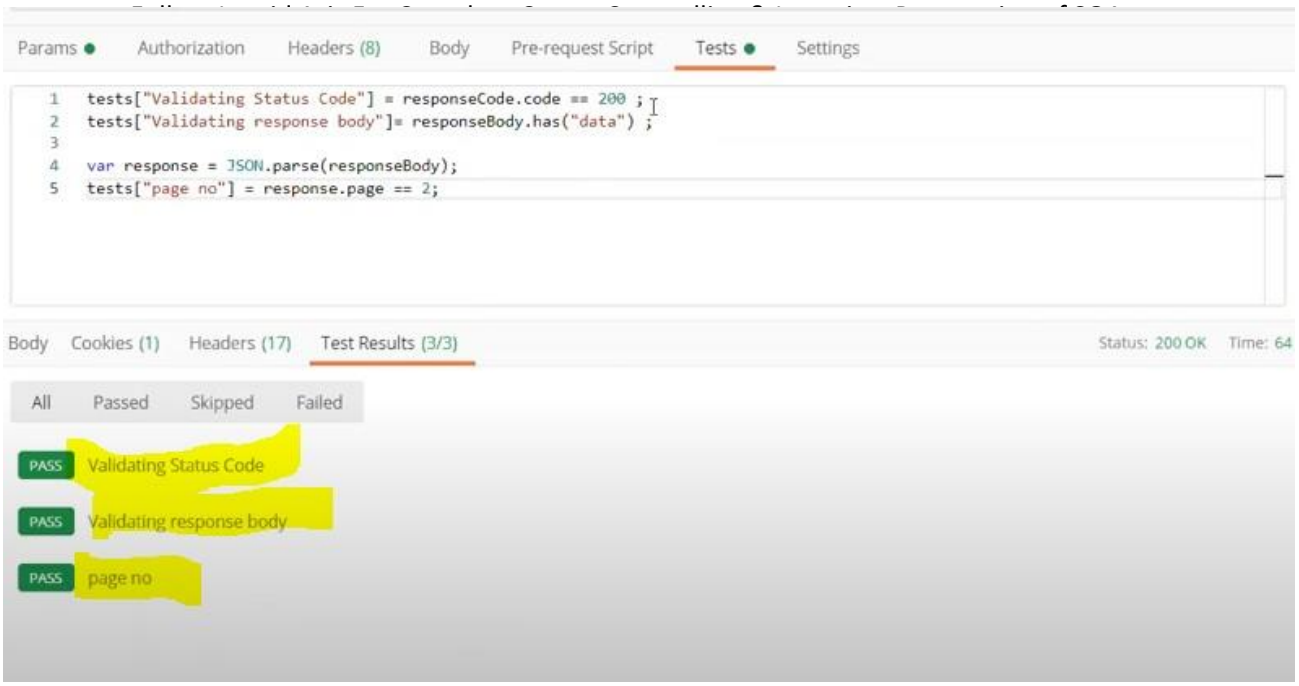
Authorization = Login Authorization , Token and some API are completely restricted so in that case we need to pass Authorization

Headers = Receive from server side

Body = Cntain all the response

Pre-Request Script = Any script that we want to execute before sending the request

Test = Most important , we add validation her. All other script will also part of this tab



Var response = Json.parse(responsebody)
//response variable is created and whole JSON response has passed to it

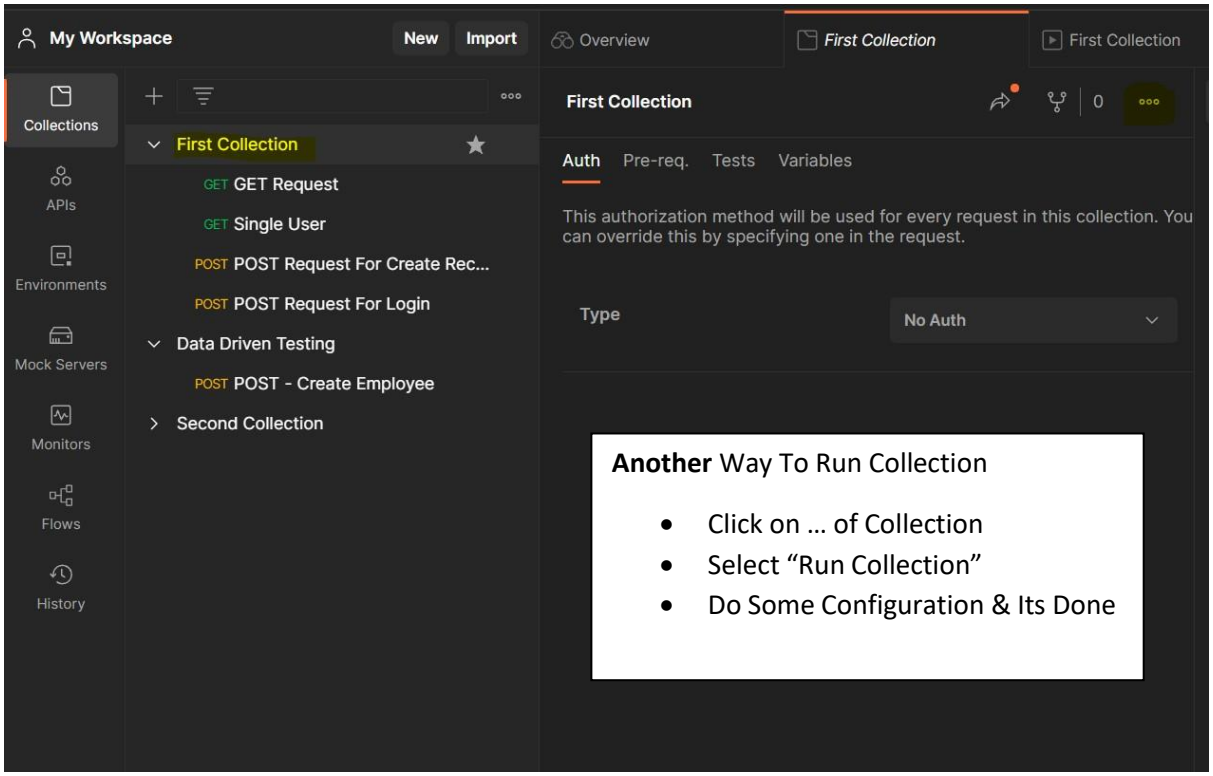
Sometimes we need to find the JSON path of the node in order to hit it.
We can find json path by one of online website https://jsonpathfinder.com/
Below code is demo of that

var response = JSON.parse(responseBody);
tests["verify_first_name"] = response.data.first_name == "Janet";

```
1  var response = JSON.parse(responseBody);
2  tests["verify_first_name"] = response.data.first_name == "Janet";
```

HOW TO RUN A COLLECTION

1. Launch Runner in new window
2. Select the desired collection
3. Select "…" and select run
4. Do some configuration



POST REQUEST

There 2 types of POST Request

1: When we add to create a new record in DB
2: We want to check the Authentication part like we login we send the credential but we don't create a new record. It's also POST Request

Post request means we want to add something to the DB
We must send data to the body to save it.

Request payload = We need to send data with our request and its optional
Response payload = We will always get it against any response

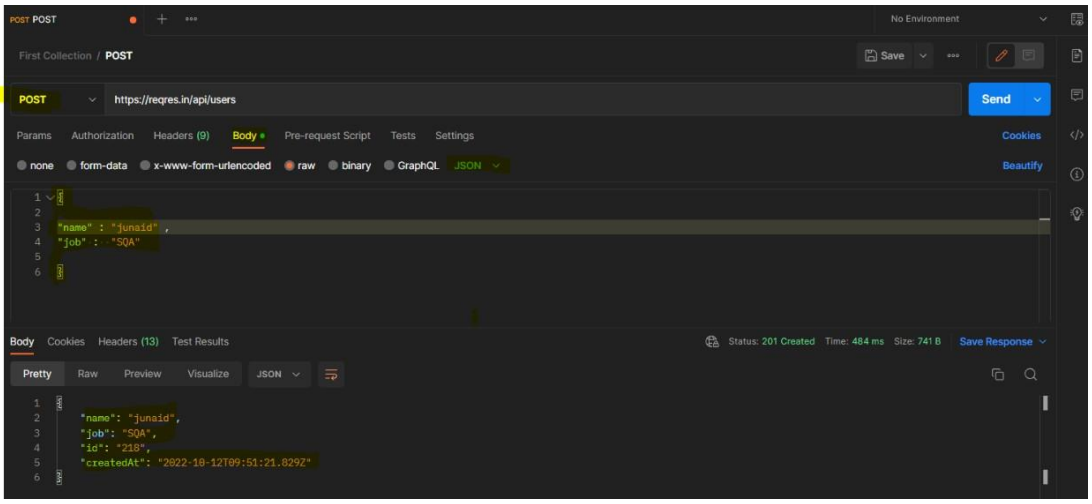
1. Create a new request by hitting + sign
2. Enter the URL
3. Click on Body and select the format in which we want to send data e.g JSON, Text, XML
4. Fill in the data and hit send
5. Response will be shown in the Body section
6. Response code should be 201 for created request

NOTE: we must send the data in the body in a specific format which we will select from drop-down

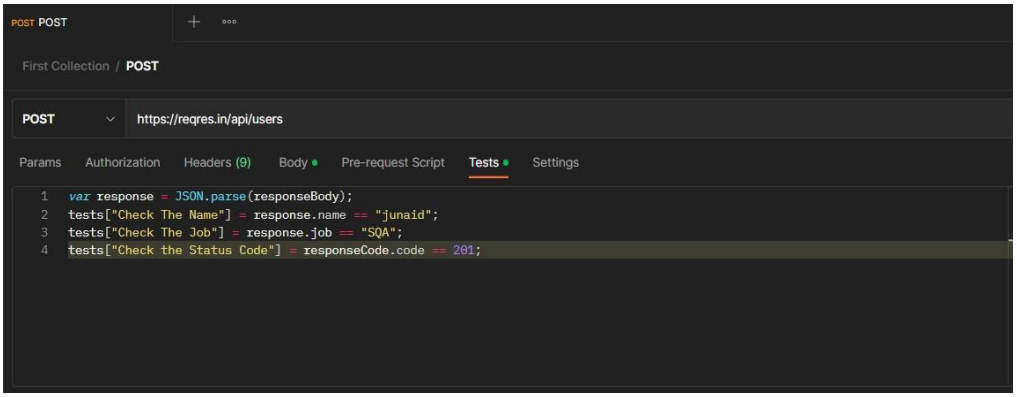
POST should be selected from the request type. If we select JSON for sending data then our data must follow the JSON format. This is very sensitive and we must follow this rule

We cannot validate token upon login as its always new and generated dynamically all the time but we can validate if the token is generated or not.

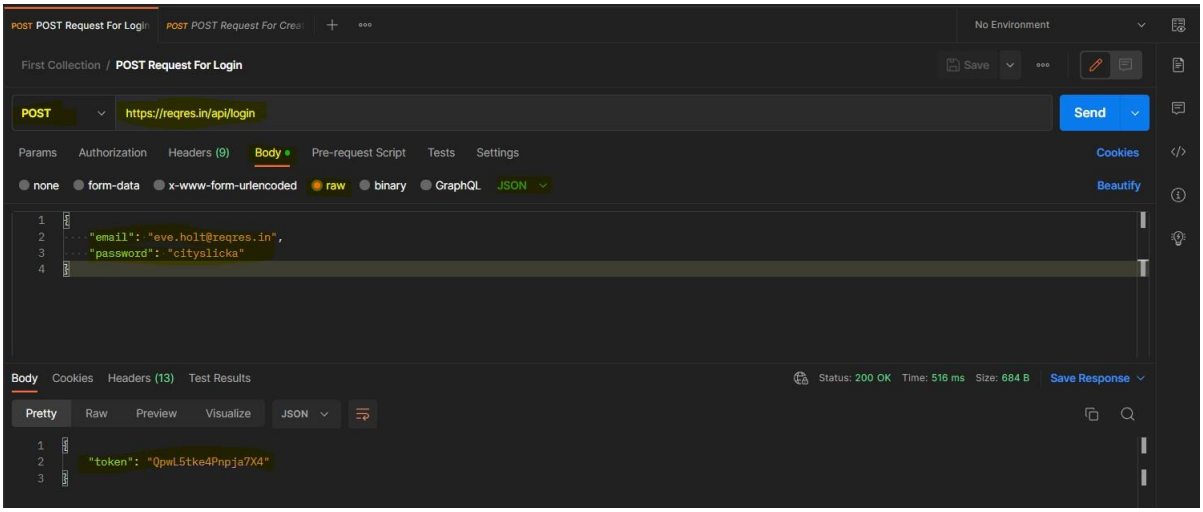
POST REQUEST FOR CREATING NEW RECORD



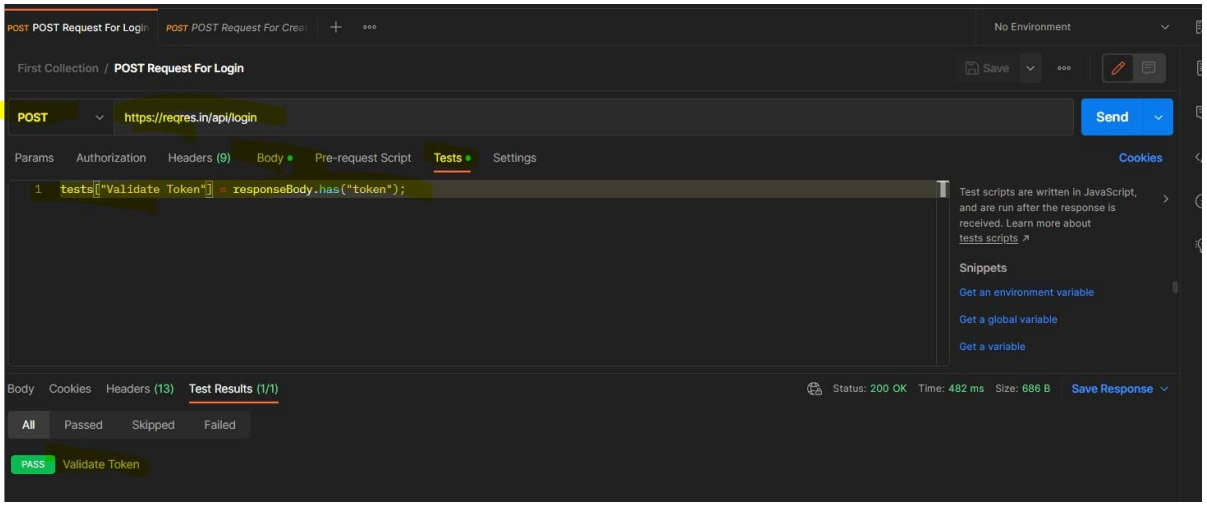
VALIDATION ON POST REQUEST



POST REQUEST FOR PASSWORD



LOGIN VALIDATION



PUT REQUEST

Old Data

{
“name” : “Junaid”
“age” : “17”
}

New Data Which is need send in the request body for updating

{
“name” : “Junaid”
“age” : “18”
}

http://dummy.restapiexample.com/api/v1/update/2

DELETE REQUEST

http://dummy.restapiexample.com/api/v1/delete/2

where 2 is the employee id

DATA-DRIVEN TESTING IN POSTMAN

Data-driven testing means we can run the same request with multiple sets of data
We can validate header by using below mentioned code

```
pm.test ( "Check the Response" , function() {
pm.response.to.be.header ("Response" , "429");
});
```

Here “Response” is element of header and we need to mentioned the desired element of header here. Keep in mind that we cannot validate the values of all the headers as some elements get value

NOTE: In Postman we can prepare test data in JSON or CSV format only

READING DATA FROM CSV/JSON FILE

Body Request

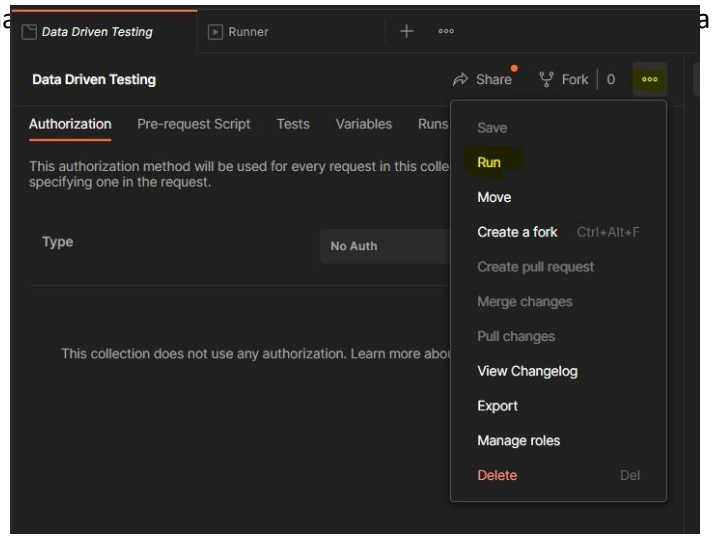
```
{
"name" : "{{name}}" ,
"salary" : "{{salary}}" ,
"age" :"{{age}}"
}
```

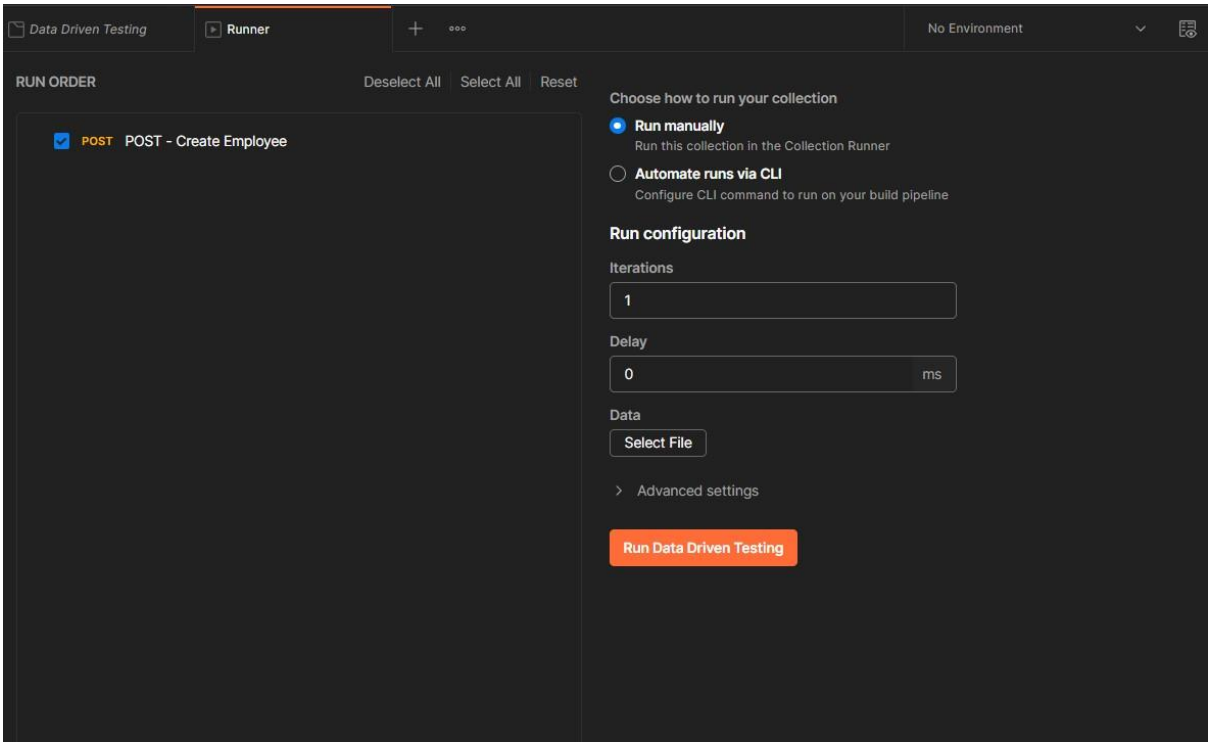
STEPS

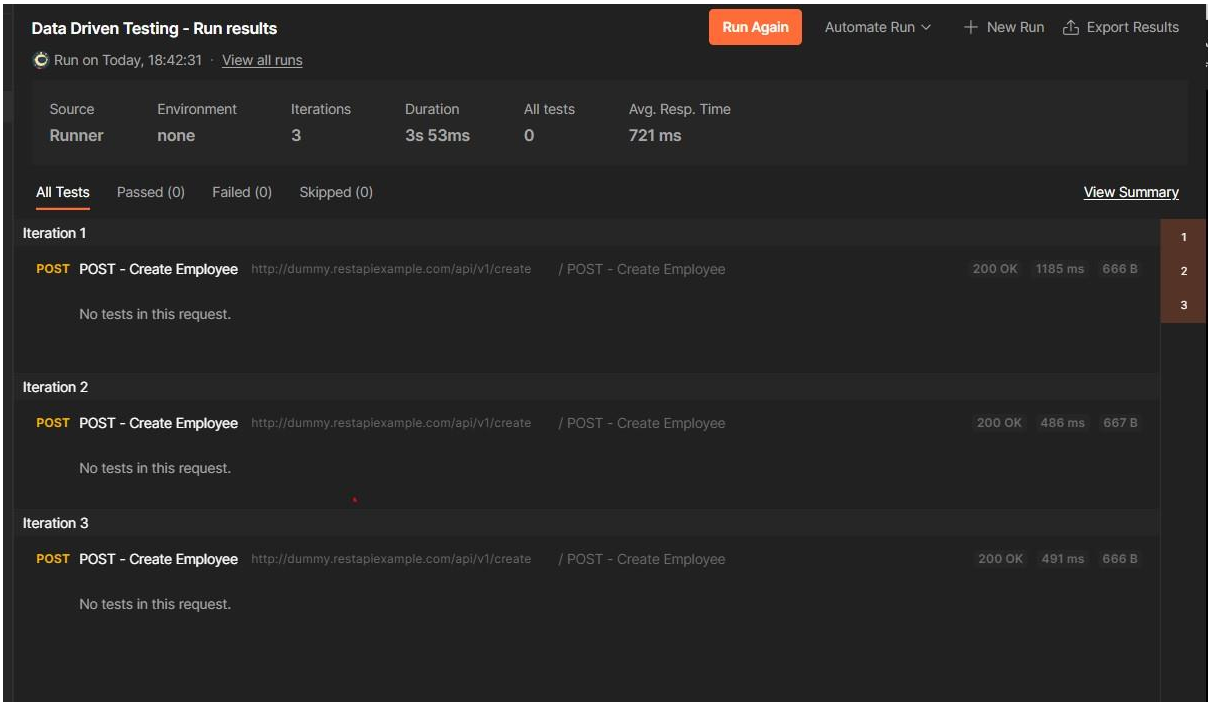
1. Prepare the CVS File
2. Add the above mentioned code in body
3. Run the Runner (File > New Runner Window)
4. Runner will get open in new window/sometime as a tab
5. Select the desired workspace if runner get open as new window
6. Select/drag the desired collection if runner get open as tab
7. Select the … and then click on run if runner get open as new window
8. Select the itereation (depend on data)
9. Select Delay = should be mini 10 ms because as a bulk API will take time to respond
10. Select CSV File From PC
11. Hit the Preview – It will only preview if our data is in correct format
12. Hit Run

NOTE: Both JSON & CSV file has exactly same method only file format is different.
In JSON we need to prepare and attach the file in JSON format.
In CSV we need to prepare and attach the file in the CSV

Follow Juna aration of SQA







IMPORT AND EXPORT COLLECTION

1. Click on 3 dots … of collection and export in JSON format
2. For Import Click on Import button which is available infront of workspace name

HOW TO RUN COLLECTION USING CMD WITHOUT POSTMAN

```
How to run postman collection/requests from commands Line
-------------------------

Pre-requisites:
----
1) Download & Install Node.js (npm)  - Refer document.
        open the command prompt
                node -v
                npm -v

2) Install newman
          npm install -g newman

3) To generate html report
        npm install newman-reporter-html

4) Export collection and then run from command prompt

How many ways we can run collection through command prompt
-------
Method1
----------
Syntax: newman run <<exported collection file.json>>
Example : C:\apitestdata>newman run EmployeeAPIS.postman_collection.json

Method2
----------
Example : C:\apitestdata>newman run EmployeeAPIS.postman_collection.json -r html
```

```
Method3 : Executing collection remotely
---------
You have to share your collection and get the URL.

https://www.getpostman.com/collections/d073b80c39b9330ebdf7

Example : C:\apitestdata>newman run https://www.getpostman.com/collections/d073b80c39b9330ebdf7 -r html
```

POSTMAN VARIABLES

Sometime we need to use the same data for multiple request in multiple API so we can create a variable to store that data so we don't need to write the same data again and again. We just need to send that variable in request

e.g base url of all API are almost same we can store that in variable so that if base url get change in future then we don't need to update it all APIs. We will just change in variable and it will get change automatically in all APIs.

There are two types of variable in Postman

Collection Variable – Use inside only collection

Environment Variable – Global variable and use for all collection

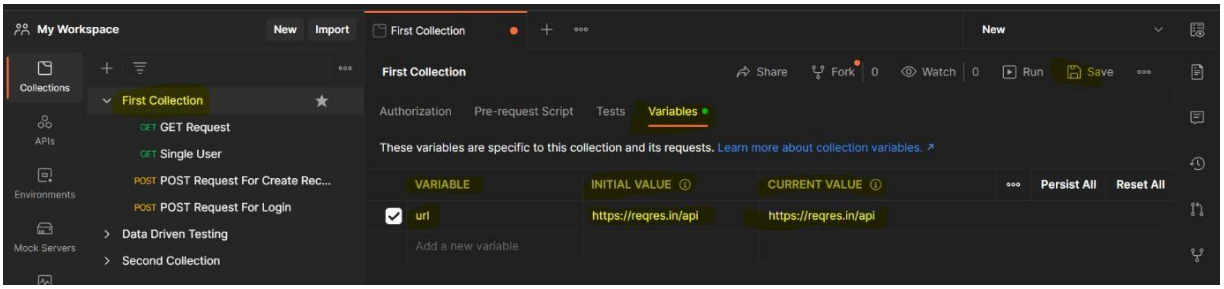
Collection Variable

Let suppose this is the API's

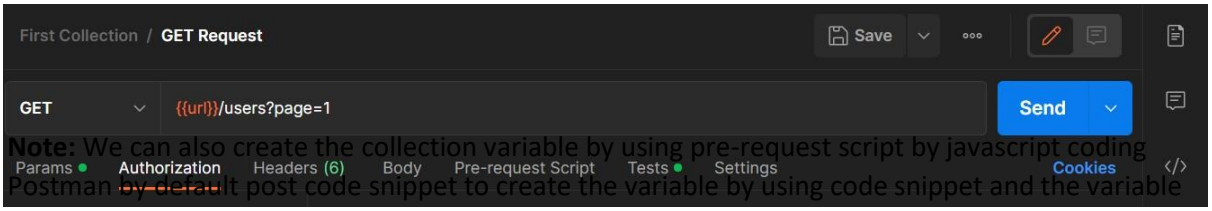
- https://reqres.in/api/users?page=1

Here https://reqres.in/api is going to use for request so we can store it in variable and use that variable.

1. Click on Collection
2. Select Variable
3. Give Varaible Name & Path
4. Hit Save

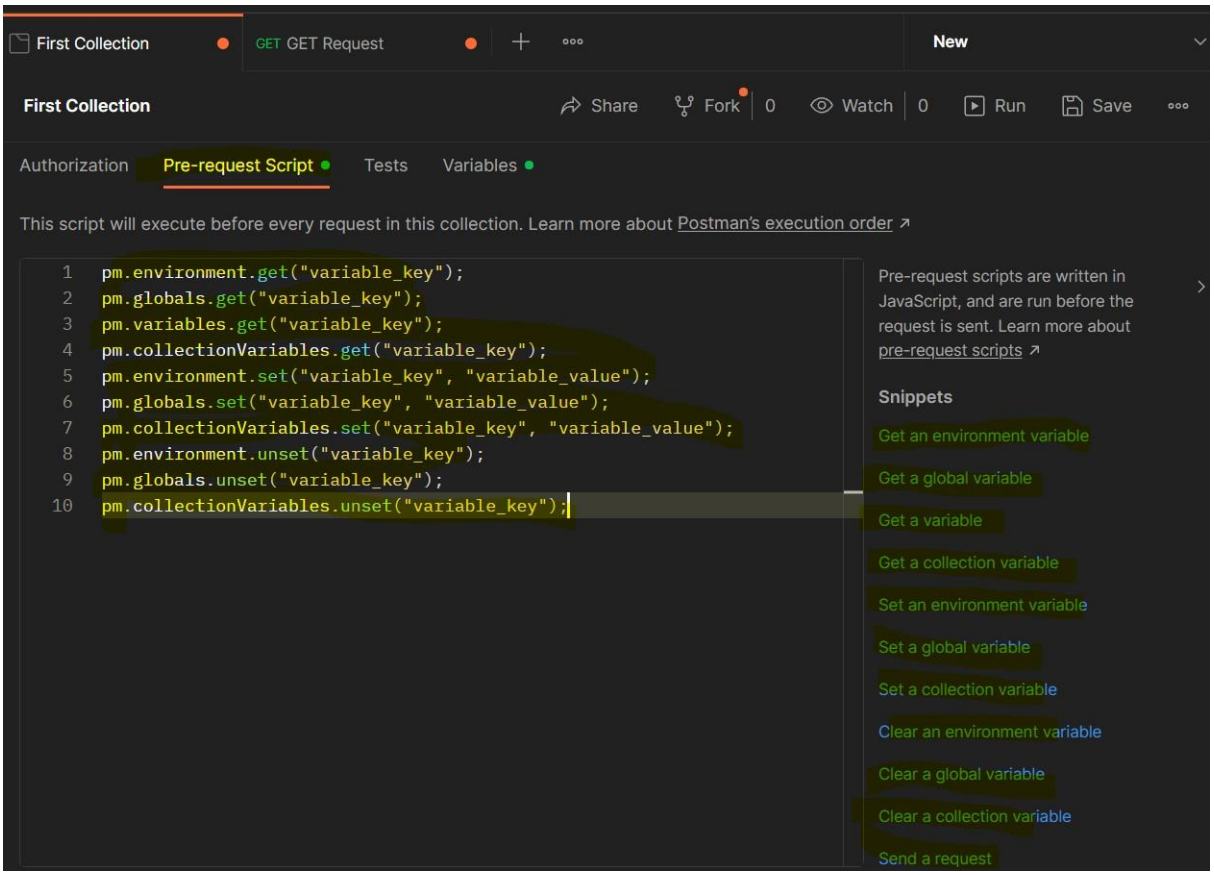


In order to use the variable we need to use {{variable name}}



Note: We can also create the collection variable by using pre-request script by javascript coding Postman by default post code snippet to create the variable by using code snippet and the variable

which will be create by using pre-requset script method, will be create dynamically on runtime



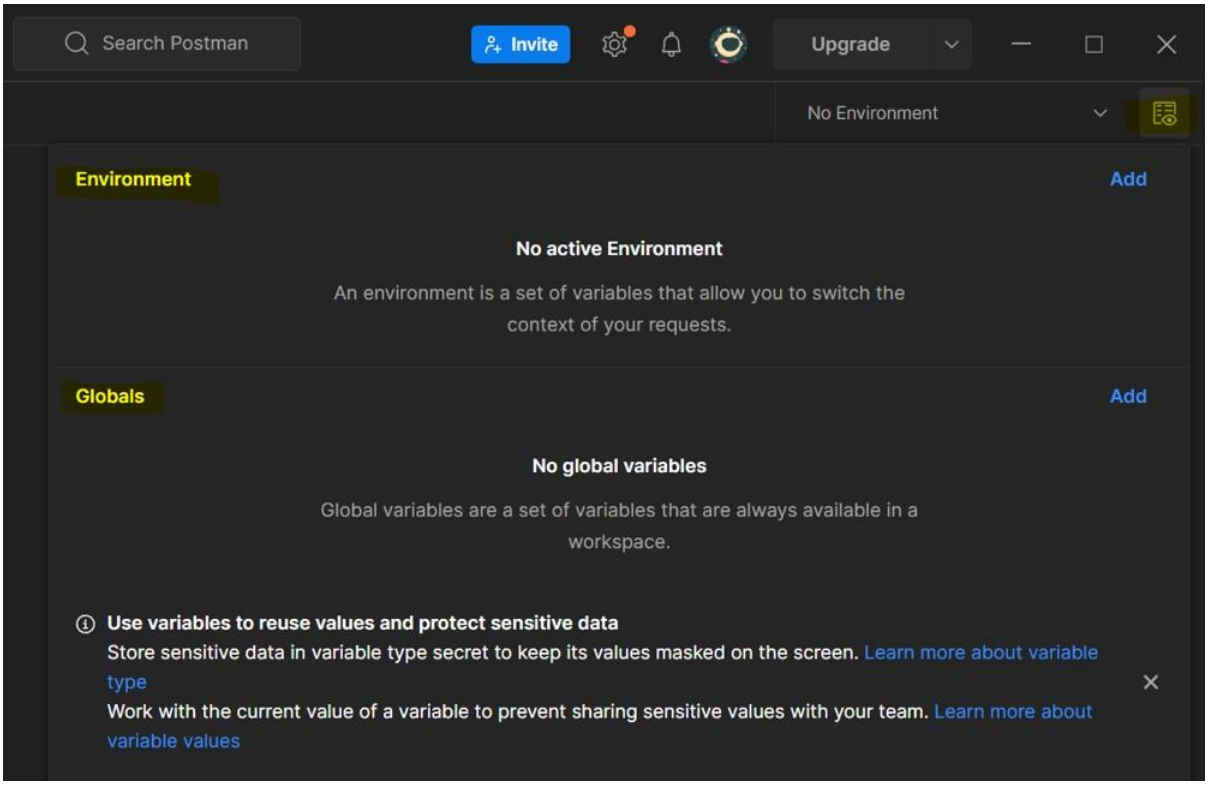
Environments/Global Variable

1. Click on eye icon on top right corner
2. Select Environment
3. Give Environment Name
4. Give Varaible Name & Path
5. Hit Save

Environments Variable – This variable varies from environment to environment. E.g there could be a chance the there will be different environment like QA , UAT , Staging etc and these environment will have different url etc. So we can create variable inside this environment and it will be effective with in that environment.

Global Variable – This variable will work all Environments

The procedure of using the variable is same {{variable name}}

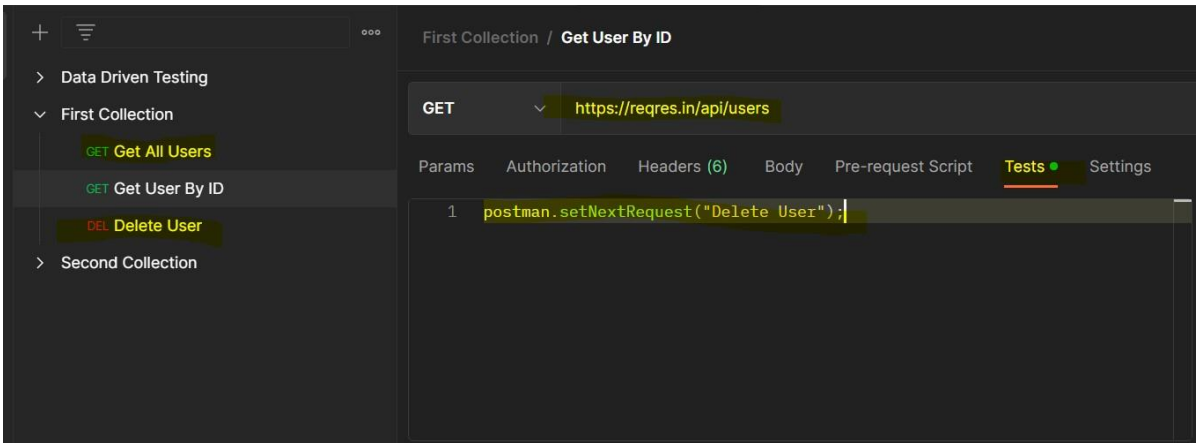


WORKFLOW

Let suppose that there are 5 request in one collection. By default they will run in sequence but if we want to control their flow and run them in our desired order then we need to small command in test tab.

Postman.setNextRequest("Next API URL HERE");

Whatever API url we will give in this statement will get executed and we can our own order in this way



HOW TO CHAIN API REQUEST

```
--------------------
How To Chain API Requests
Authorizations
--------------------

API Request1---> Response

API Request2--> Reponse

API Request3---> Reponse
```

API Chaining request mean, using the output of one request as an input of other request. For example ther are multiple users and we want to update the the user 2 with the name of user 3.

So instead of hardcoding we can do below mentioned

- Execute the GET all users
- Store the user 3 first name in collection variable
- By using below mentioned code

```
jsonData=JSON.parse(responseBody)
value=jsonData.data[0].first_name
pm.globals.set("username", "variable_value");
```

//Second line of came from JSON path finder webiset.

- Execute the PUT request and in first name send {{variable name}} as parameter

```
{
    "name": "{{username}}",
    "job": "zion resident"
}
```

HOW TO PASS AUTHORIZATION

```
Basic Auth:
--------------
https://postman-echo.com/basic-auth
        username: postman
        password: password

API Key:
------------
api.openweathermap.org/data/2.5/forecast/daily?q=Delhi&cnt=1
API Key/appid: fe9c5cddb7e01d747b4611c3fc9eaf2c

Bearer Token/ OAuth 2.0:
----------------------
https://developer.github.com/v3/repos/
```

Noticed

If we click on ... which are available in front of collection and click on documentation then it will give all basic code snippet that we can use while creating scripts

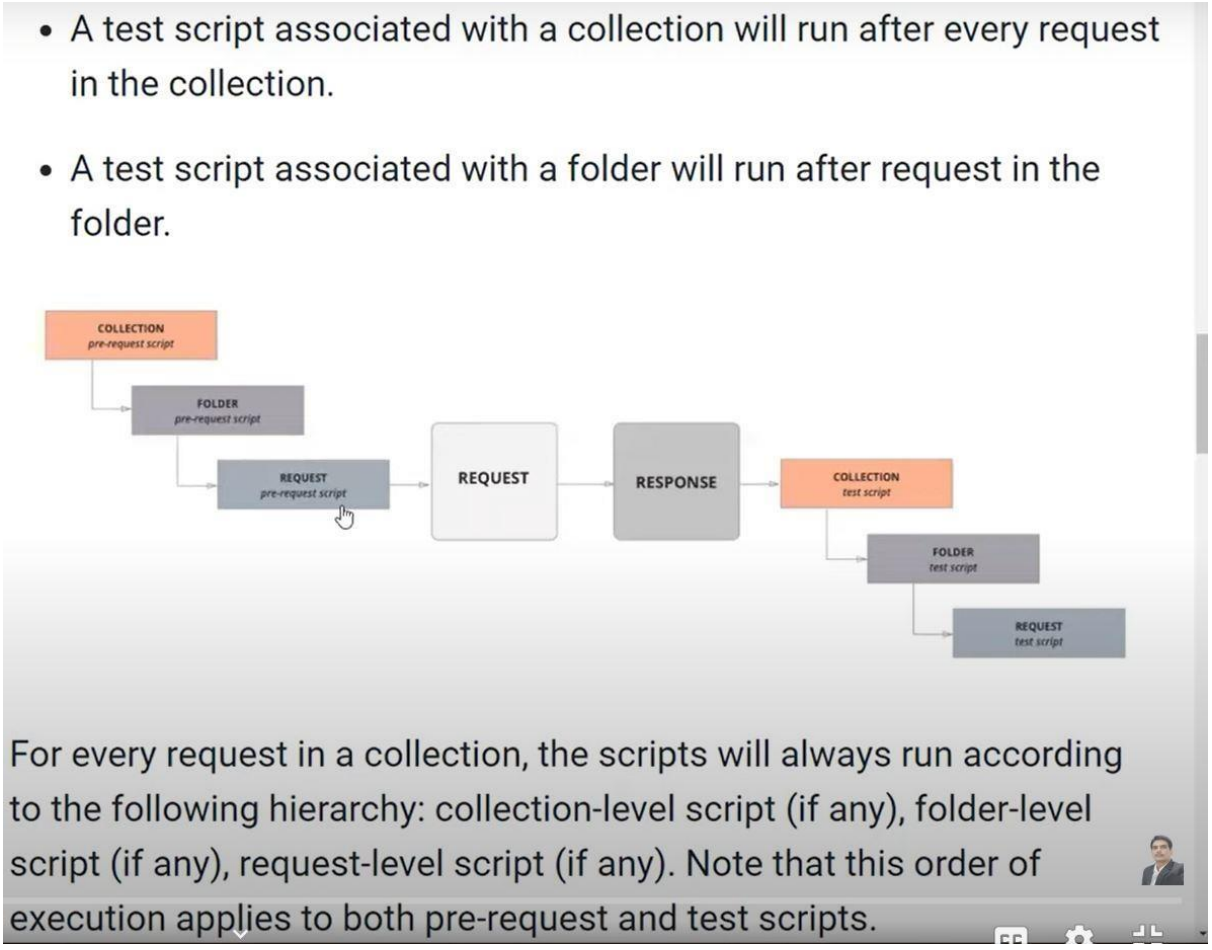
TYPE OF SCRIPTS

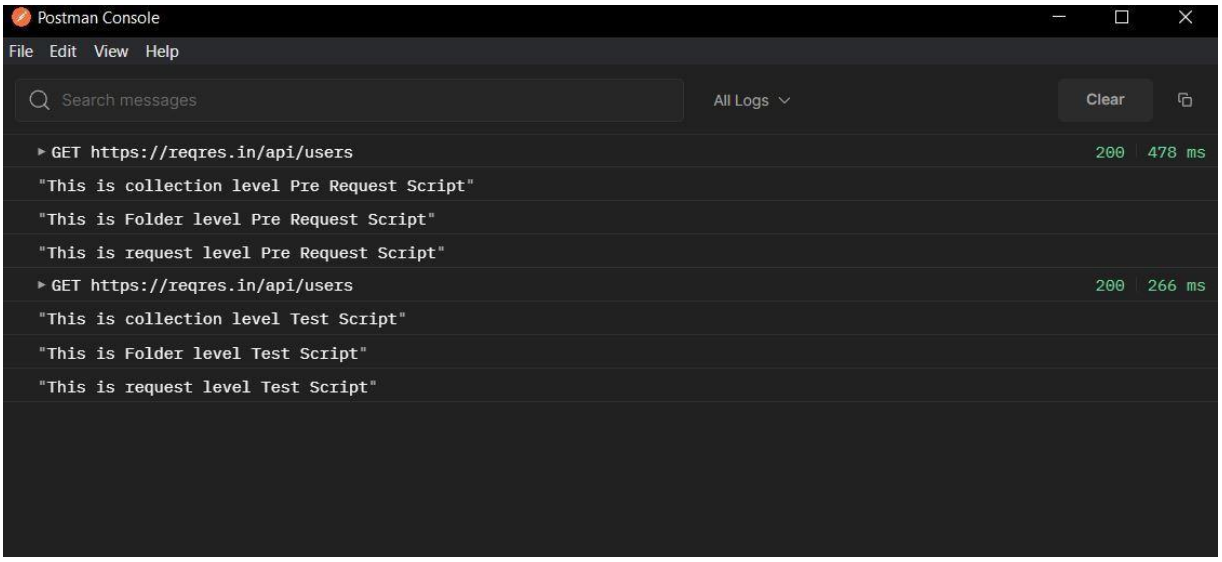
There are 2 types of scripts
Pre-request - Which run before the request
Test - Whic run after the request

We can have multiple folder inside collection and multiple request inside each folder
COLLECTION
  FOLDER
    REQUEST LIST

So, we can write scripts at collection level, folder level and request level. Its all depend on our need and requirement

Once all done then we can run the whole collection using runner and check the response in Console. We can open console by Alt + Ctrl + C





HOW TO ADD SCRIPTS AND MULTIPLE ASSERTION

We can verify all the data from response. There are some basic snippet which are pre available in postman and we can also create our own assertion as per the need. Below mention code will verify some element from JSON response and we can use same code to validate any JSON element

```
//Multiple Assertion
//This Assertion will verify the response ID is equal to expected or it
//Similar way we can verify every element like Name, age , email , course , etc

pm.test("The response has all properties", () => {
    //parse the response JSON and test three properties
    responseJson = pm.response.json();
    pm.expect(responseJson.data[1].id).to.eql(2);
   pm.expect(responseJson.data[1].first_name).to.eql('Janet');
   pm.expect(responseJson.data[1].last_name).to.eql('Weaver');
   pm.expect(responseJson.data[1].first_name).to.eql('Janet');
});
```

In above code we are verifying multiple element in one shot. If anyone of them get fail then it will show that whole assertion got failed

Note: we need to user JSON path finder in order to use that in code. In above code 'data.first_name[1]' is JSON path of first name which we have found by using below webisite:

https://jsonpathfinder.com/

HANDLING RESPONSE THAT DON'T PARSE

Sometimes JSON response don't get parse successfully. We can still validate data by handling the response is text format. Use the below code in such a situation

```
pm.test("Handle the UnParsed Response", () => {
    responseJson = pm.response.json();
    pm.expect(pm.response.text()).to.include("janet.weaver@reqres.in");
})
```

Expecting Status Code From List

Suppose there are different status code and we are expecting anyone of them then we can validate the list and test will get pass if anyone of expected code get received

```
//Expecting status code from the list
pm.test("Successfull Status Code list", () => {
    responseJson = pm.response.json();
    pm.expect(pm.response.code).to.be.oneOf([200,201])
})
```

How Validate if Specific Header is Present or Not

We can validate the specific header is present or not by using below code.

```
//Testing Headers
pm.test("Content-Type Header Present Or Not", () => {
    pm.response.to.have.header("Content-type");
    pm.response.to.have.header("Server");
})
```

How To Validate The Value of Header

We can also the validate the value of header if its as expected or not

```
//Validating the Header Value
pm.test("Validating The Content-Type Value", () => {

    pm.expect(pm.response.headers.get("Content-
type")).to.be.eql("application/json; charset=utf-8");

})
```

Note: We can validate the header and its value in once test as well. Above mentioned example are for making clear understanding. We can use below mentioned code to perform both validation.

```
//Testing Headers& Its Value
pm.test("Validating Content-Type Header & Its Value", () => {
    pm.response.to.have.header("Content-type");
    pm.expect(pm.response.headers.get("Content-
type")).to.be.eql("application/json; charset=utf-8");

})
```

By using above mentioned code we can easily validate all the header and its values

How to validate Cookies

```
//Testing Cookie is Present or Not
pm.test("Cookie is Present Or Not", () => {

    pm.expect(pm.cookies.has("__Cookie")).to.be.true;

})

//Validating the Header Value
pm.test("Cookie is Present Or Not", () => {

    pm.expect(pm.cookies.get("__Cookies NAME ").to.be.eql("expected value");

})
```

We can also validate cookie and its value in one test. Above mentioned code is written seperatly just for the clear understanding

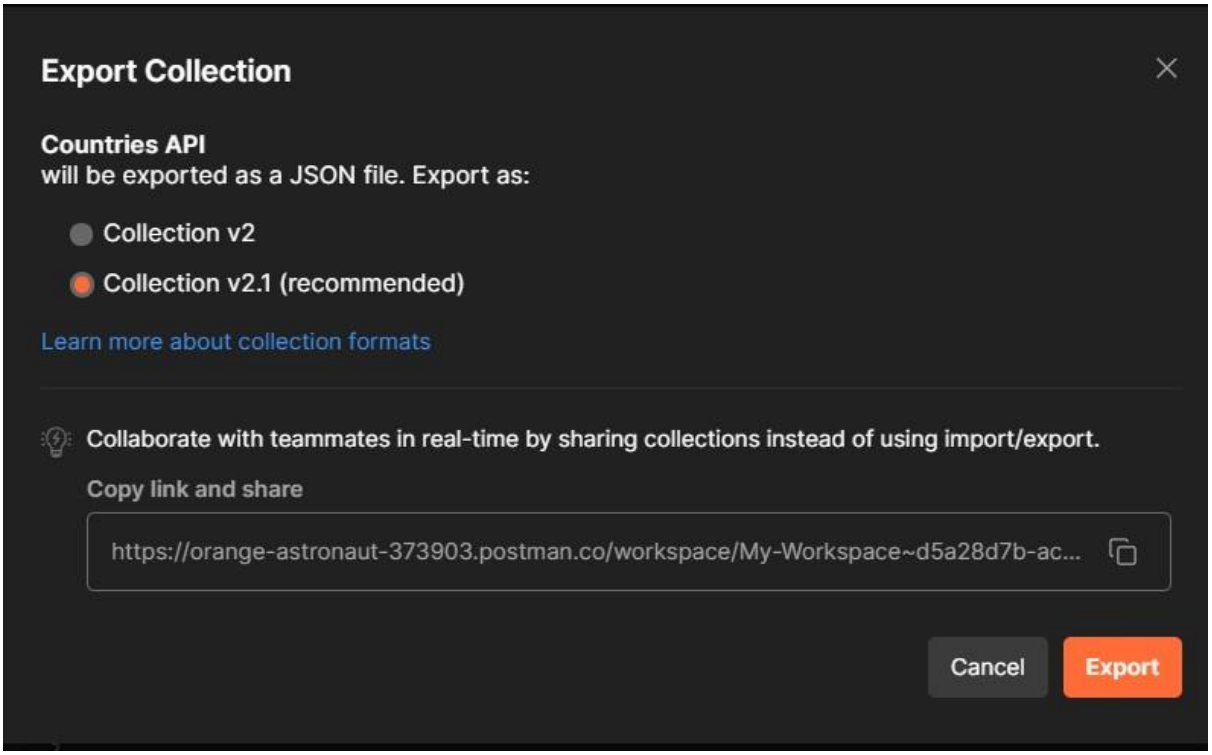
How to validate Response Time

```
//Validating Response Time
pm.test("Response Time Should Be Less Than 20MS", () => {

    pm.expect(pm.response.responseTime).to.be.below(120);

})
```

Export Collections

1. Click on 3 dot of collection and select Export
2. Select v2 recommended option
3. Hit Export

It will export collection in JSON Form



Import Collections

1. Click on Import
2. Upload file and hit import
3. It will import collection
4. We can import collection multiple way

How to create the API document

We can create the small document of our collection which show which request we have sent and all other information. It will be public URL and anyone can see it as a report

Follow the below mentioned steps for documentation creation

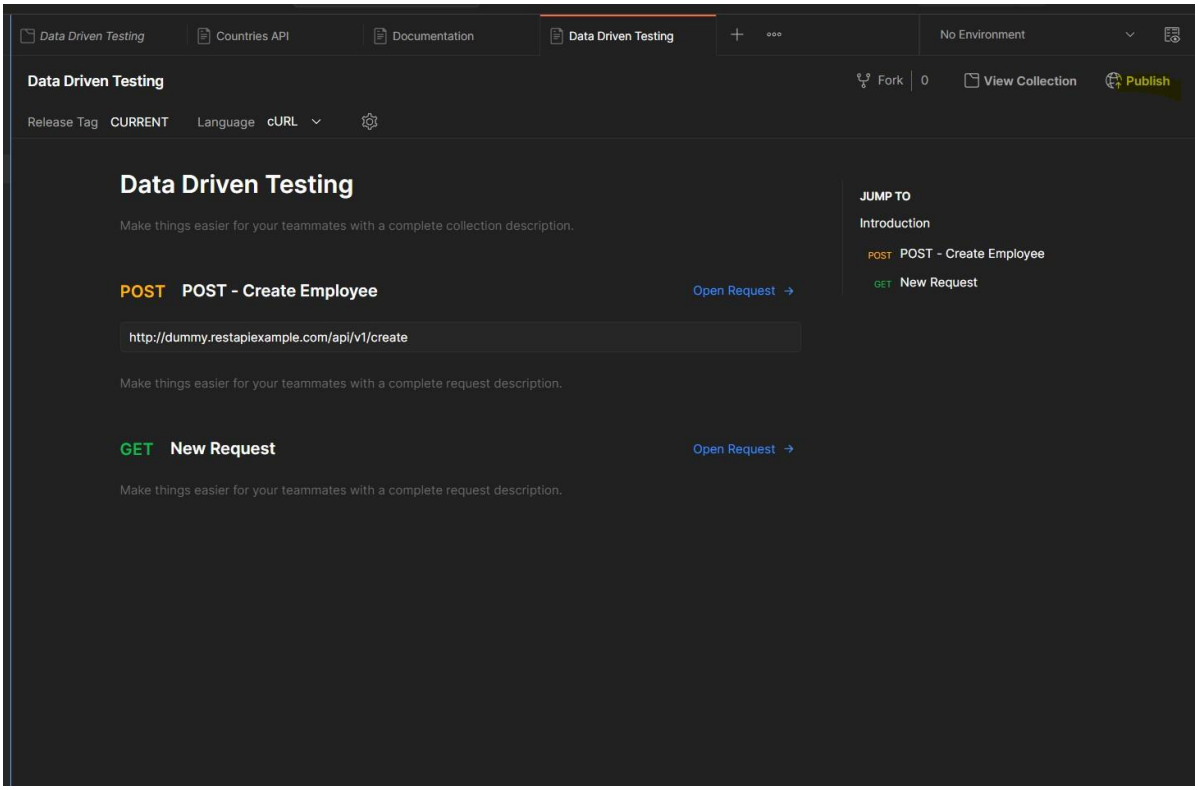
Click on 3 dot … of collection

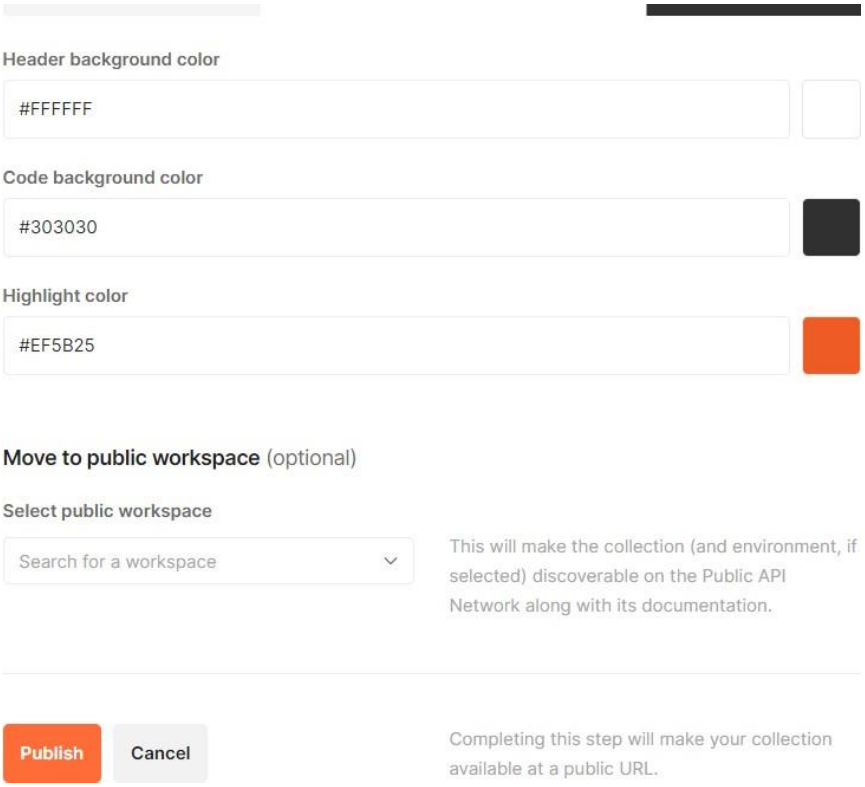
Select view documentation

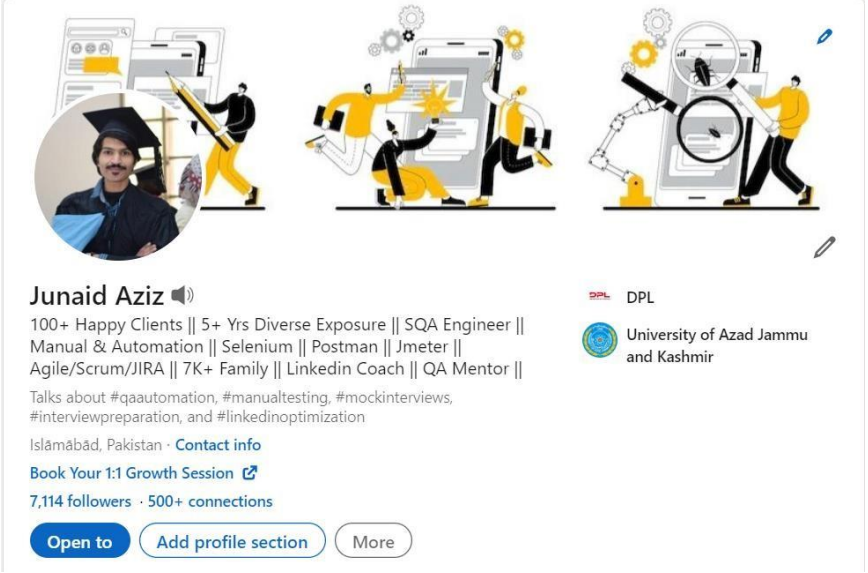
Click on Publish

Do setting as per the need

Hit publish




Header background color
#FFFFFF
Code background color
#303030
Highlight color
#EF5B25
Move to public workspace (optional)
Select public workspace
Search for a workspace
This will make the collection (and environment, if selected) discoverable on the Public API Network along with its documentation.
Publish
Cancel
Completing this step will make your collection available at a public URL.



Junaid Aziz

100+ Happy Clients || 5+ Yrs Diverse Exposure || SQA Engineer || Manual & Automation || Selenium || Postman || Jmeter || Agile/Scrum/JIRA || 7K+ Family || Linkedin Coach || QA Mentor ||

Talks about #qaautomation, #manualtesting, #mockinterviews, #interviewpreparation, and #linkedinoptimization

Islāmābād, Pakistan · Contact info

Book Your 1:1 Growth Session

7,114 followers · 500+ connections

Open to | Add profile section | More

DPL

University of Azad Jammu and Kashmir

Compiled By: Junaid Aziz