

POSTMAN Notes - TheTestingAcademy

100+POSTMAN Questions and Answers

1) What is an application programming interface (API)?

- API stands for Application Programming Interface, and it is a set of routines, protocols, and tools for creating software applications. APIs define how one piece of software should communicate with another.
- API serves as a connection point between two software applications, allowing them to communicate. A programming interface (API) is a set of software capabilities that another application can use.
- It can be considered as the waiter which acts as the middleman between your requests and the chef. Similarly API refers as the middleman between a client and a server.

2) What are some tools used for API Testing?

A:- There are many API testing tools. The following six are the top most according to the users/downloads. These are not the rankings though.

- *Postman*
- *SoapUI*
- *Katalon Studio*
- *Tricentis Tosca*
- *Apigee*
- *Jmeter*

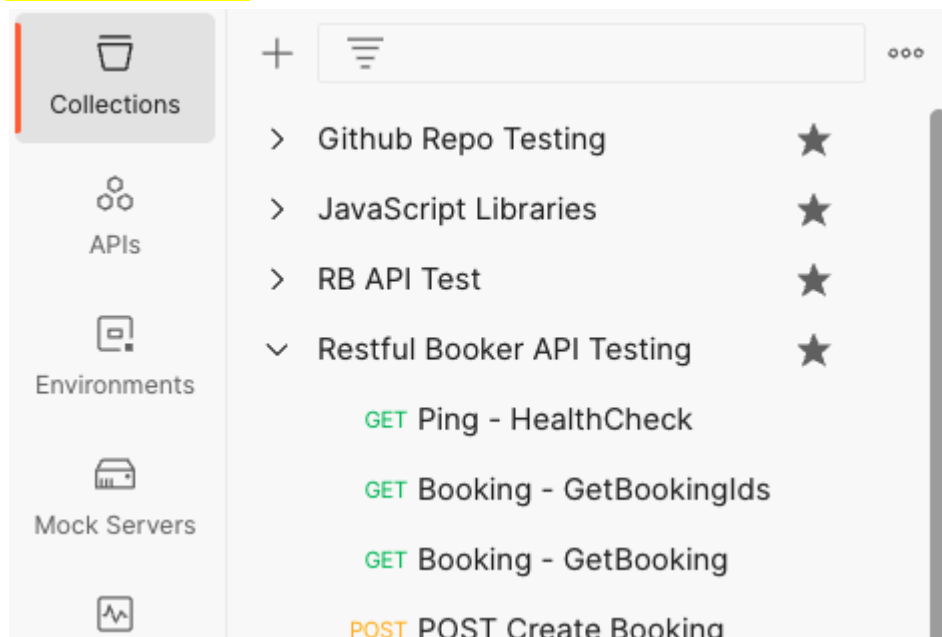
3) What is Postman?

- Postman is an API platform for developers to design, build, test and iterate their APIs.
- We can say Postman is an API platform for building and using APIs.
- Postman is an API (application programming interface) development tool which helps to build, test and modify APIs. Almost any functionality that could be needed by any developer is encapsulated in this tool.
- It is used by over 5 million developers every month to make their API development easy and simple.

4) What is a collection in Postman?

A :- A collection in Postman helps to group similar requests. It helps in systematically arranging the requests into folders.

Consider Below example - **Restful Booker is a collection where we have full CRUD requests present.**



5) Why do we use Postman?

A :- We use Postman for the below reasons:

- **It is free:** Postman is free software that we can use for API testing. It is free to download and use for teams of any size.
- **It is easy to use:** Postman is an easy-to-use software tool. We can send HTTP requests of various types (such as GET, POST, PUT, PATCH, etc.). We have to download it, and we can send our first request in minutes. It also gives us the ability to save environments for future use.
- **Community & Support:** It has a huge community forum for customer support and extensive documentation.
- **It is extensible:** Postman facilitates us customizing it according to our needs with the Postman API.

- **APIs Support:** It facilitates us to make any API call (REST, SOAP, or plain HTTP) and easily inspect even the largest responses. It also helps manage the end-to-end lifecycle of the API - starting from design to mocking to testing and finally maintaining the APIs.
- **Runtime Services:** Postman provides Runtime Services that help us manage API collections, environments, work-spaces, and different examples.
- **Integration:** Postman facilitates us to easily integrate test suites into our preferred CI/CD tools and services, such as Jenkins with Newman (command-line collection runner).

6) How will you log variable values in Postman?

A :- We can log the variable values in Postman in the console by using the command:

```
console.log(pm.variables.get("variable_name"));
```

7) How do you access postman variables?

A :- It can be accessed by using the variable name as: `{{variable_name}}`

8) What are the various authorization methods provided by Postman?

A:- Postman provides the below API request Authorization Options: **API Key Bearer Token, Basic auth, Digest auth, OAuth 1.0, OAuth 2.0, Hawk Authentication, AWS Signature, NTLM Authentication**

9) What are the different types of API requests supported in Postman?

A :- Postman supports the following type of requests:

- GET
- POST
- PUT
- PATCH
- DELETE
- COPY
- HEAD
- OPTIONS
- LINK
- UNLINK
- PURGE
- LOCK
- UNLOCK
- PROPFIND
- VIEW

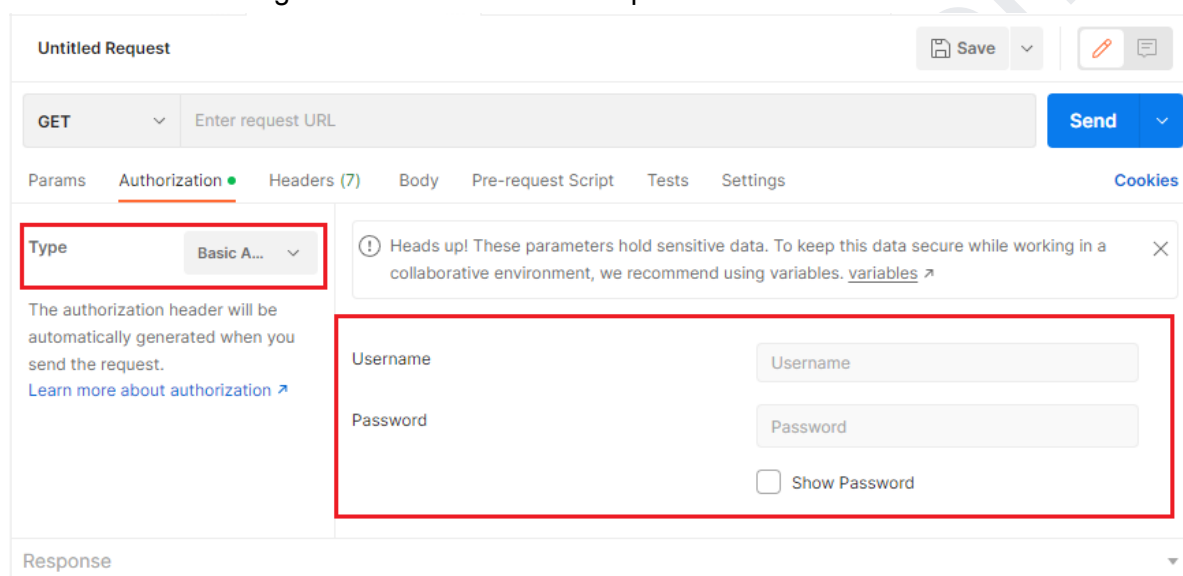
10) How are Query Params different from Path Variables?

A:- Path Variables are used for identifying specific resources and Query Parameters are used for sorting or filtering the resources.



11) What is Basic Auth in Postman?

A:- Basic Auth in Postman is a type of authorization technique provided in Postman for HTTP user agents like web browsers. It provides fields to enter username and password which when entered gets associated with the request.



12) What encoding is accepted by Postman in authorization credentials?

A:- Postman accepts authorization credentials in Base64 encoding format only. It is provided in Postman by default. If we do not want to use an inbuilt encoding system, we can refer to third-party websites for converting the credentials in base64 format.

13) Can we have the same names for global variables in postman?

A:- The scope of global variables is limited to the workspace and is global. Due to this, variables having global scope cannot have the same names. We can have the same names for local variables but they need to be part of different variables.

14) What do you know about postman monitor?

A:- **Monitoring is a method of staying in sync with the health and performance of the APIs.** Postman provides **inbuilt monitoring services that help us be in sync with the API** development and performance. The monitors provided by Postman are mainly based on the

working of collection runners. They run every request in the collection and analyze the values mentioned in the test scripts. Monitors use the test scripts for validating and monitoring the responses. The reports generated are shared with the developers over emails or alerts in slack, hipchat, etc based on our configuration settings.

15) What is a binary form in POST methods?

:- The binary form is designed to help send data in a format that is not possible to be entered manually. These options are used while sending large files like images, CSV files, etc in the POST request. Binary representation is the easiest representation for sending complex data with the request.

16) What are the limitations of Postman?

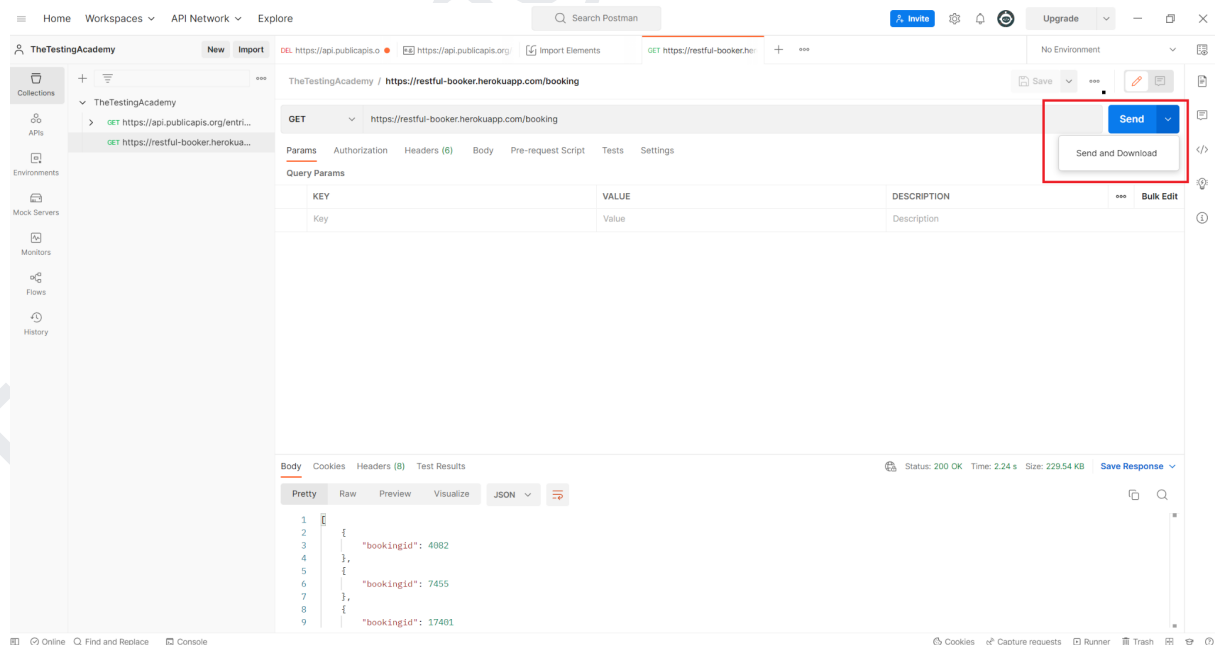
A:-

- Postman is not suitable for processing 1000+ API requests.
- If the project is very large, managing the collections and requests becomes cumbersome.
- It is not suitable if we want to manage the workspace in the form of code as there would be a lot of code duplication for dynamic API requests.

17) How can you save the responses of API to a file in Postman?

A:- We can do this in two ways:

- :- Click on the Download button in the response section.
- :- Click on the arrow beside the send button - There will be an option to send and download. Clicking on it will prompt Postman to ask the location of saving the response post successful execution of the request.



18) What are the two types of scripts in Postman

- **Tests script**
- **Pre-request script**

19) What is the significance of 301 status code?

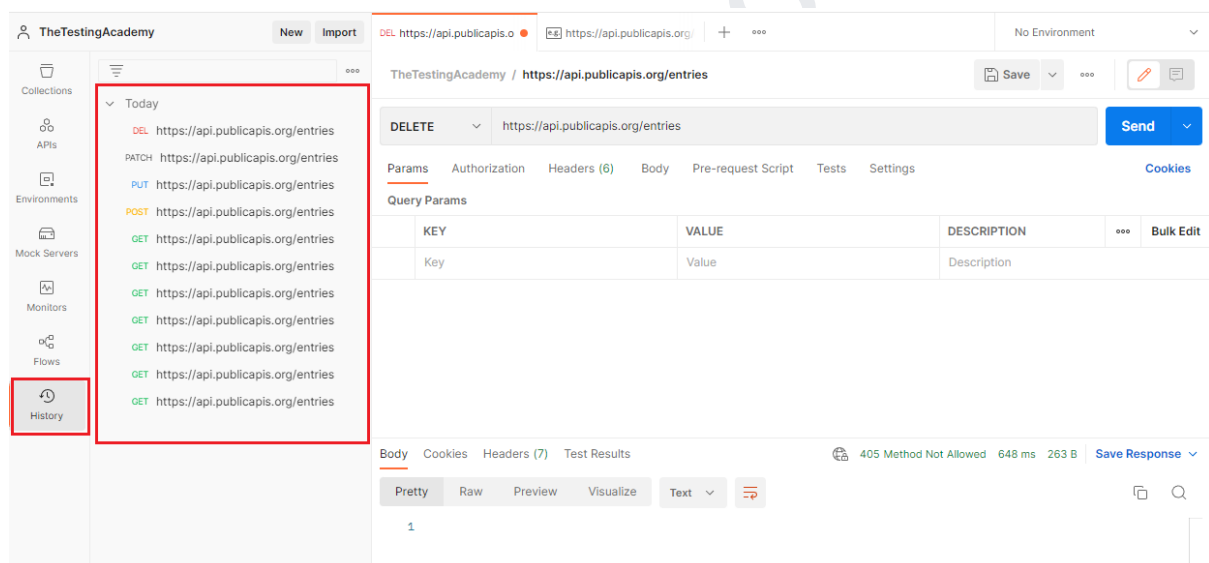
A:- 301 status code represents permanent redirect from one website page to another. It tells the search engine that the old page is outdated and the engine has to index the new page URL.

20) What is the History tab in Postman?

A:- All the requests you send in Postman appear under the History tab of the sidebar. It is very much similar to browser history, which you can clear whenever you want.

21) How do you access the history of requests in Postman?

A:- The request history can be accessed in the History tab provided on the Postman application. If we sign into the Postman account, then the history will be synced across the devices where you are logged in. When you click on any of the requests present in the History tab, the view opens the request that we have saved while we were working on it earlier. History also consists of the collection runs that were executed as summarized versions.



22) What is an HTTP request?

A:- An HTTP request is a program that the client makes to a name host located on a server. It works as a communication interface or a request-response protocol between a client and server. The primary use of the HTTP request is to access a resource on the server. To initiate the HTTP request, the client uses components of a URL (Uniform Resource Locator) that also includes the information needed to access the resource.

23) State The Core Components of an HTTP Response?

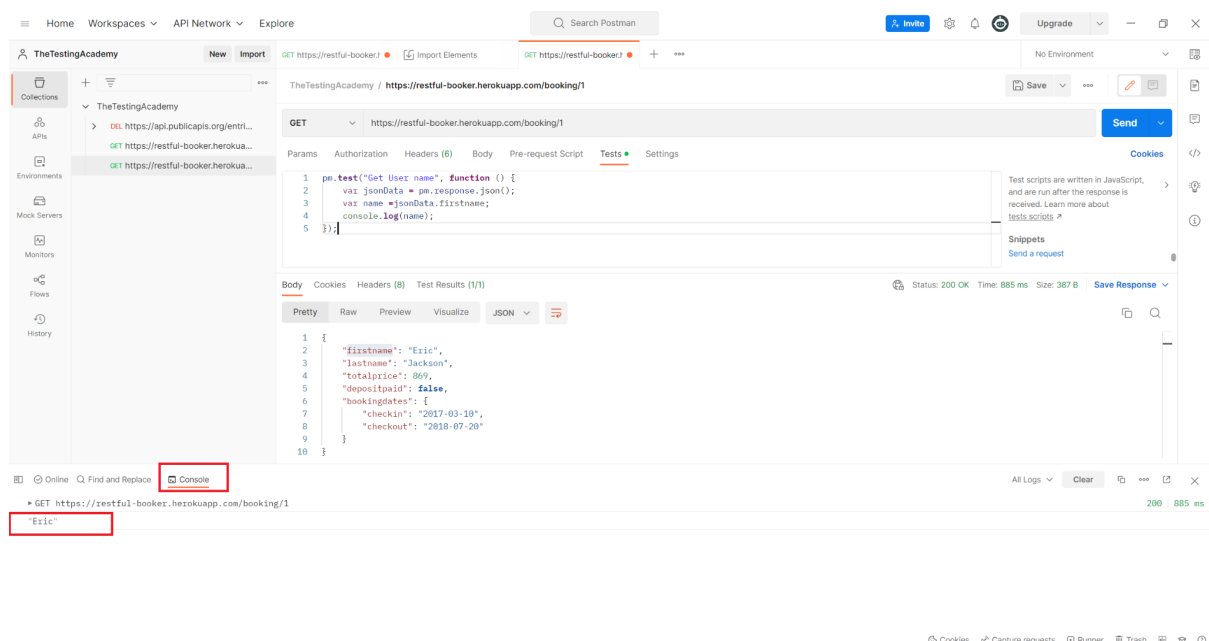
A:- In Postman, every HTTP response contains four key elements.

- **Response/Status Code-** There are response code issues by a server for client's request, as 404 means Page Not Found.

- **HTTP Version-** HTTP version name. For example, HTTP v2.2
- **Response Header-** It included information for the HTTP response message. For example, The content length, date, status, server type, etc.
- **Response Body** – It contains the data which a client requested from the server.

24) Does Postman provide a feature to log request and response?

A:- Postman does allow viewing of requests and response parameters in the software application itself. But it is important to see how the request was sent upon applying the pre-request scripts. In such cases, Postman has an additional tool called “Postman Console” which is used for viewing every request and response detail. We can also log the details in the console by using console.log statements in the scripts.



25) What is Basic Auth in Postman?

A:- In Postman, Basic Auth is an authorization method provided for HTTP user agents like web browsers to enter username and password. After entering the username and password that you can associate with the request.

26) What is a binary form in POST methods?

A:- Post binary form is designed to send the information in a format that is impossible to enter manually. These options are used while sending large files like CSV files, etc.

27) What is the main difference between Authorization and authentication?

A:- Here are a few differences between authorization and Authentication:

- Authorization is the act of allowing or permitting someone, whereas authentication is proving that something is genuine.

- Authorization always comes first, while authentication comes after authorization.
- Authorization is open to anyone with permission, whereas authentication requires you to have a password.

28) What is the Payload in Postman?

A:- The Payload of an API Module is the body of your request and response message. When making an API request, it contains the data you send to the server. You can send and receive Payload in various formats, for example, JSON or XML.

29) What is a Pre-Request script?

A:- Pre-request scripts help you to execute JavaScript before a request runs. It allows you to accomplish pre-processing tasks like setting variable values, parameters, headers, and body data.

30) What is the meaning of the term environment in Postman?

A:- The environment in Postman is a set of key-value pairs. Postman allows you to build multiple environments and switch among them with a click of a button.

31) Is it possible to import local variables in Postman Monitors?

A:- Postman monitors allow you to import local variables but not global variables.

32) Can you have two global scope variables with the same name in Postman?

A:- No, the global scope never has duplicate/same names, while variables having local scope can have the same name in various environments.

33) How do you remove local variables?

A:- Local variables are automatically removed once the tests have been executed.

34) What is 'Postman Collection runners'?

A:- Postman contains a collection runner that is useful for automating API testing. It helps visualize details of each iteration and test results. A postman collection runner is also used for Data-driven testing.

35) Why is saving your work in the Postman cloud is not advisable?

A:- You should not save your work in Postman as your business details do not remain confidential. Moreover, saving your on-Postman cloud may cause a security breach as it requires sign-in. Therefore, saving your work in the Postman cloud is not advisable.

36) What are the standard rules of an API test design?

A:- Here are the key principles of an API test design:

- **Setup:** Create objects, start services, and initialize data.
- **Execution:** Apply API or the scenario, including logging
- **Verification:** It is use for evaluating the result of the execution

- **Reporting:** Indicates Pass, failed, or blocked status
- **Clean up:** Pre-test state

37) Which programming language is used for Postman tests?

A:- JavaScript is used for Postman tests.

38) What are some of the JS libraries available in Postman?

A:- Some JS libraries available in Postman are

- 1) Lodash
- 2) Moment
- 3) GUID

39) What is GUID?

A:- GUID is short for Global Unique Identifier. It is hexadecimal digits that are separated by hyphens. This Postman identifier GUID solves the purpose of uniqueness.

40) What is the importance of setNextRequest in Postman?

A:- setNextRequest helps you to define the workflow. It is needed to change the order of the requests being executed.

41) What test code allows you to check whether the response status is 200 or not?

A:- Following is a test code to check whether the response status is 200 or not: `tests["Status Code is 200"] = responseCode.code === 200;`

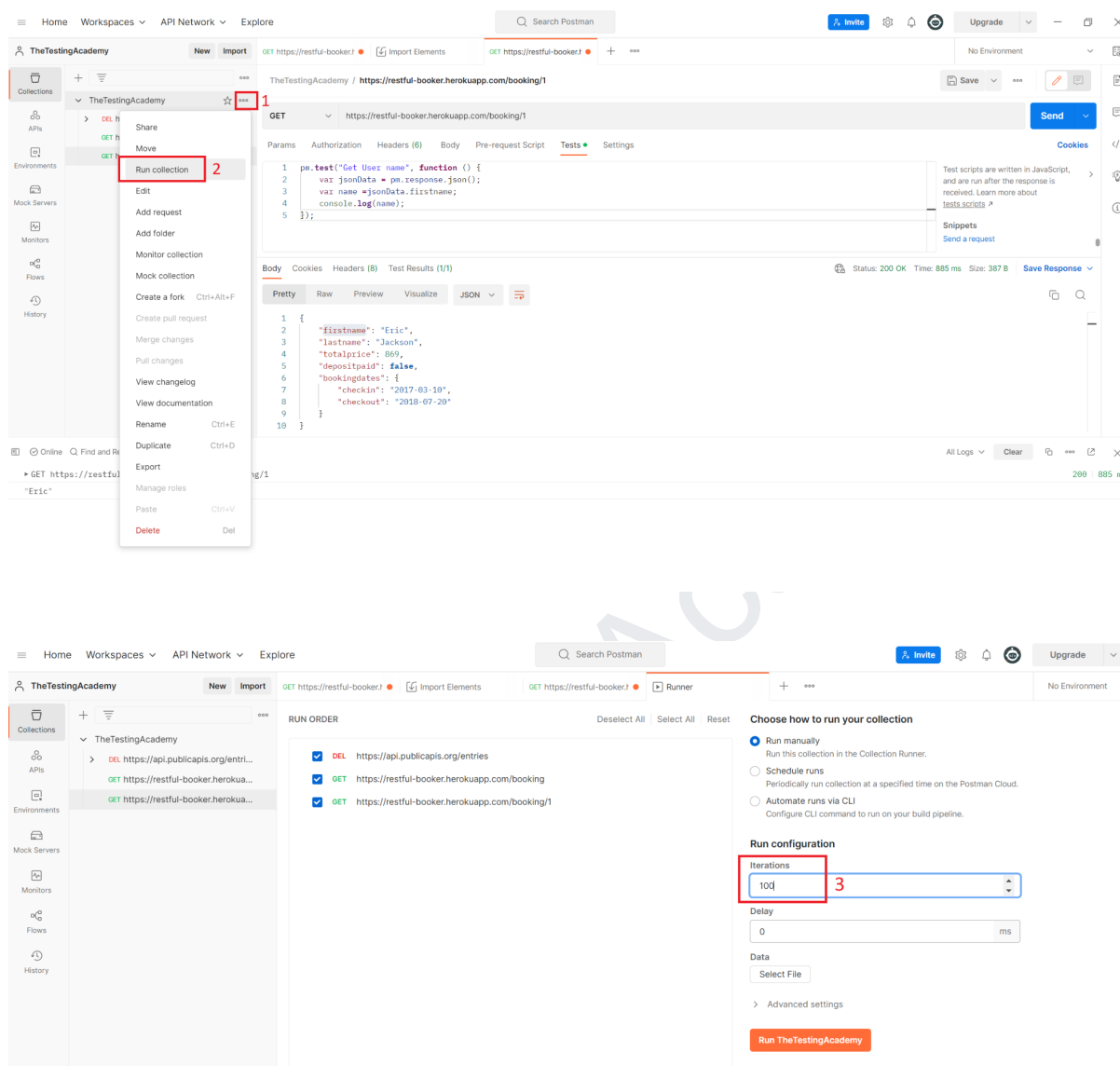
42) Describe any four response things you receive from a response (Correct or Incorrect)

A:- *Status Code*

- Response Status
- Response time
- Response Size
- Response Headers
- Response Cookies
- Response Date and Time
- Response Session limit
- Response Cookies
- Response Server
- Response type

43) How can you iterate a request 100 times in Postman?

A:- You can iterate a request 100 times in Postman by using Collection Runner.



44) Can you read the Postman Chrome application to read and write cookies?

A:- No, it is impossible to read and write cookies using the app.

45) Postman is available as a native desktop app for?

A :- [Postman API testing tool](#) is currently available for Mac, Windows (32-bit / 64-bit), and Linux (32-bit / 64-bit).

46) What is status code 201?

A:- Status code 201 is created only when a resource is successfully created using a PUT or POST request. It returns a link to a newly created one with the help of the location header.

47) What are the different types by which we can see the response body in Postman? Explain.

In Postman, a response body can be seen by three different types.

- Pretty
- Raw
- Preview

Although all the three have their own importance and value in Postman, the most commonly used is Pretty as it shows the response code in different format and colors which is easy to read and analyze the response. It is just like any good text editor used for coding

48) What is the "Bulk Edit" feature of Postman used for?

A:- Bulk Edit feature of Postman is used for the convenience of adding parameters to a new request from the previous request. Since a request can have many parameters and it is very difficult to copy and paste one by one, the bulk edit feature helps us copy all the keys and their respective values at once and paste them.

49) Why do we group requests under collections when collection is already a grouping of requests?

A:- A collection may have hundreds of requests under it. We need to sub categorize the requests according to a more specific category so that it is easier for us to find them, edit them or modify them. For this we use folders in collections. A collection may have many folders inside it and a folder may have many requests. This way we can generalise the types of requests to a deeper level than the collections which are already generalised. For easiness, a collection can be considered a folder "Movies" in your system which has all the movies. A folder can be considered as different folders inside "Movies" like Hollywood, Bollywood etc which have respective types of movies.

50) Which method should you prefer? Javascript or Functional to write the tests?

A:- It is advised and recommended to use the functional method while writing tests in Postman. Although there has been no notice of ending the support for JS method.

51) What is the need to Monitor the collections in Postman?

A:- It is very important that your API's responses and performance remain up to the mark throughout the day. Monitors can help you schedule a collection of test runs to monitor the performance and response of your APIs even if you are not available or not handling them.

52) Can we run monitors in Postman without Signing in?

A:- No, monitors cannot be run without signing in because monitors run your collection even if your system is shut down. So, you need a place to store the collection and let it run automatically. You also need a place to store the reports so that you can look at them when you are free. This all needs to be saved into your postman account and hence you need to sign in

53) What is the Chai Assertion Library?

Chai assertion library is an assertion library which is installed beforehand to use in Postman. This is used to write assertions in Postman which are very beneficial. Chai assertion helps us write many lines of test code in a few lines which is both understandable and readable. Chai uses the BDD approach which means that the chai library has codes that are more user friendly.

A simple code written in the chai library which tests if number 3 is already in the array or not.

```
pm.test("Number included", function(){ pm.expect([1,2,3]).to.include(3); });
```

Advance Postman QnA

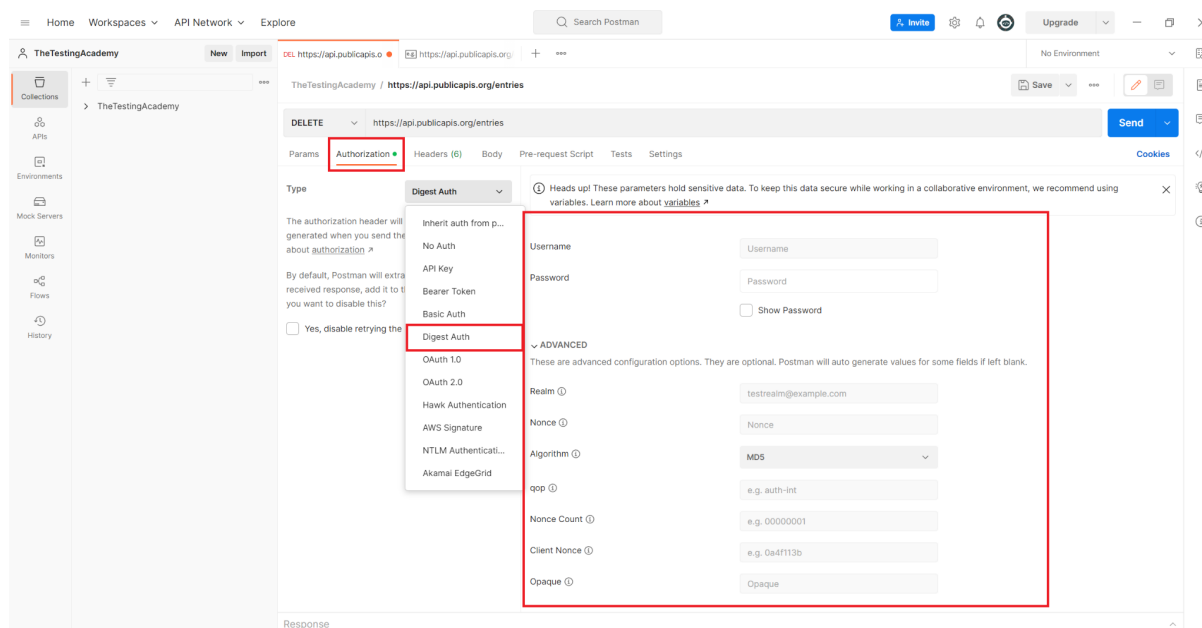
1) What is digest auth in Postman?

A:- Digest Authorization is one of the authorization techniques provided by Postman. In this technique, the client first sends the request to the API and get responses from the server including a number which is usable only once, a realm value and 401 unauthorized response. We will be then sent back an encrypted data array having both username and password along with the data received from the server earlier. The server uses this data to generate an encrypted data string and compares this with what was sent for authenticating the request. We can do this by selecting the Authorization tab, then selecting "Digest Auth" from the drop-down list. The Postman window presents the fields for both stages of the authentication request. The fields required for the second stage of the request are auto-filled based on the data received from the server.

More details as below

Setting the fields in the Advanced section is optional. Postman will populate them automatically when your first request runs.

1. **Realm** - A string specified by the server in the `WWW-Authenticate` response header.
2. **Nonce** - A unique string specified by the server in the `WWW-Authenticate` response header.
3. **Algorithm** - A string that indicates a pair of algorithms used to produce the digest and a checksum. Postman supports `MD5` and `SHA` algorithms.
4. **qop** - The quality of protection applied to the message. The value must be one of the options specified by the server in the `WWW-Authenticate` response header.
5. **Nonce Count** - The hexadecimal count of the number of requests (including the current request) that the client has sent with the nonce value in this request.
6. **Client Nonce** - An opaque quoted string value provided by the client, used by both client and server to avoid chosen plaintext attacks, to provide mutual authentication, and to provide some message integrity protection.
7. **Opaque** - A string of data specified by the server in the `WWW-Authenticate` response header, which will be used unchanged with URIs in the same protection space.



2) In a Collection Run, what will execute first?

A:- In a Collection run, pre-request scripts at the Collection level are executed first.

3) What is "x-www-urlencoded" in the Post method in Postman?

A:- Form data and x-www-form-urlencoded are very similar. They both are used for almost the same purposes. But the difference between the form data and x-www-form-urlencoded is that the url will be encoded when sent through x-www-form-urlencoded. Encoded means the data which is sent will be encoded to different characters so that it is unrecognizable even if it is under attack.

4) What command line interface is used with Postman normally to serve continuous integration.

A:- Newman is used with Postman normally as a command line interface to serve continuous integration.

5) Write the command for running a folder in Newman.

A:- In Newman it is not necessary to run the complete collection to check just a bunch of requests. This is obviously time consuming and not recommended. We can also run just a folder located inside a collection in the Newman. For running a folder in Newman, the following command is used

```
newman run <collection_name> --folder <folder_name>
```

6) How can Postman collections run through the command line?

Answer: Postman has a command-line integration tool called Newman with which you can run any existing Postman collection.

Newman is a node js based package, which requires just a node environment to execute the collection and has full parity with the Postman collection runner i.e. the Newman collection runner supports the Postman capabilities like Running assertions, Pre-request scripts or any other scripts that are associated with the requests that are a part of the collection.

To use Newman:

- You need to have a node installed.
- Now the Newman package needs to be installed through npm using the command.

```
npm install -g newman
```

- The collection needs to be executed and the associated environment configuration should be first exported to its JSON form through the Postman application
- Now run the below command to run the Postman collection through Newman.

```
newman run {{path to collection json}} -e {{path to environment json if any}}
```

7) How can you generate HTML based reports running tests through the Postman?

Answer: Newman uses the concept of reporters and templates to generate HTML reports for the executed collection.

Hence, to generate HTML reports, you first need to install a reporter. You can install any of the available HTML reporters like **Newman-reporter-html** as a node package through the below command.

```
npm install -g newman-reporter-html
```

Once the HTML reporter is installed, we can use the Newman command to run the collection with -r flag i.e. the reporter flag and specify the reporter name as HTML.

The below command is used:

```
newman run {{path to collection json}} -e {{path to environment json if any}} -r html
```

Please note that as we have not mentioned the name or folder where we want the reports to get generated, by default the reports will be generated in a folder named "Newman" that gets created in the same directory where the Newman command is executed from.

8) Why is Base64 encoding primarily used in Postman?

A:- Base64 encoding is primarily used because it does the task of data transmission in a textual format that is easier to be sent in the requests in HTML form statistics format. Another reason why we use this is that using identical 64 characters for encoding is heavily reliable in any language we use.

9) Why does Postman never accept any other encoding apart from Base64?

A:- You can use base64 as it helps us transmit the data into the textual form and send it as HTML form data. Moreover, we must rely upon the same 64 characters in any encoding language.

10) Is it preferable to save our work on Postman Cloud?

A:- When working on enterprise-level applications for organizations, it is not preferred to store our work on the Postman cloud because of the required privacy and security. In the Postman cloud, there are chances of security breaches by a skilled hacker.

11) What are the various variable scopes provided by Postman?

A:- Postman has the following variable scopes:

- **Global Variables**
- **Local Variables**
- **Environment Variables**
- **Collection Variables**
- **Data Variables**

12) Is it possible to reuse the authentication token for multiple requests?

A:- You can indeed use the authentication token more than once. To do this, create a collection, add all requests with the same authentication token, and then assign the Collection with the auth token to the Collection. By choosing "Inherit auth from parent" under the Authorization tab, we may apply it to each request separately.

13) What do you understand about ScratchPad?

A:- Scratch Pad is a space provided by Postman that helps us to work without being connected to Postman servers. It provides the flexibility of utilizing some of the features of postman offline. The features include- collection creation, creating requests and the ability to send requests. These are stored locally and once logged in, the work is saved into the workspace.

14) What is the Postman execution order for a collection?

A:- For all the requests in a collection, the scripts will execute in the following given order:

Step 1) A pre-request script associated with a collection will run before every request.

Step 2) A pre-request script associated with a folder will run before every request in a specific folder.

Step 3) A test script associated with a collection will run after every request.

Step 4) A test script associated with a folder will run after the request in the specific folder.

15) What is the purpose of the 304 status code?

A:- The HTTP 304 Not Modified client redirection response code indicates that there is no need to retransmit the requested resources. It is an implicit redirection to a cached resource. This happens when the request method is a [safe](#) method, such as [GET](#) or [HEAD](#), or when the request is conditional and uses an [If-None-Match](#) or an [If-Modified-Since](#) header.

The equivalent [200](#) OK response would have included the headers [Cache-Control](#), [Content-Location](#), [Date](#), [ETag](#), [Expires](#), and [Vary](#).

In Detail Answer to above Question.

A:- An [HTTP status code](#) is a response code sent between a browser and a web server every time the browser receives an HTTP request. For example, when you enter a URL to access a website.

When you make a request on your browser, it will send an **If-Modified-Since** request header to the web server. This request header is sent to know when the web page in question was last modified.

Then, the **Last-Modified** response header will specify when the web source was last modified. If there's no change, the server will send the HTTP 304 response code.

The **HTTP status code 304** means **Not Modified** – the web page you requested hasn't changed since the last time you accessed it.

After that, your browser will retrieve the cached version of the web page in your local storage. That way, the browser doesn't have to download the same information from the website's server repeatedly.

16) How do you write test cases for basic authentication in Postman?

A:- Basic Authentication is one of the authentication techniques provided in Postman that ensures that we can set the username and password along with the API requests. We can do this by first setting the credentials of the API by:

- Navigating to the Authorization tab.
- From the dropdown, select Basic Auth.
- Add the username and password to the API in the input fields given.

```
pm.test("Is the Request Authenticated?", function () {  
    var jsonData = pm.response.json();  
    //if authenticated then assert to true  
    pm.expect(jsonData.authenticated).to.eql(true);  
});  
  
pm.test("Is the Content-Type present?", function () {  
    pm.response.to.have.header("Content-Type");  
});  
  
pm.test("Is it a successful POST Request?", function () {  
    pm.response.to.have.status(200);  
});
```

17) How do you set the same headers for all requests in a Postman Collection?

A:- Postman collections allow using pre-request scripts at the individual request level and the collection level. We can add any script that applies to all requests in the collection in the pre-request scripts. We can do it by following the below steps:

- Right-click on the collection, navigate to the pre-request tab.
- Add the below lines of code in the script to add a request header for all the requests present in the collection.

```
pm.request.headers.add({  
    key: 'TestHeader',  
    value: 'testValue'  
});
```

- Click on Update for saving the script.
- Run the request in the Collection and check the Postman console to ensure that the headers have been added.

18) How will you stop the execution of upcoming requests or Stop execution of the collections?

A:- We can use the below code to stop the execution of the next request:

```
pm.setNextRequest(null);
```

19) What do you understand by the pre-request script?

A:- Pre-request scripts are those scripts that are used for executing Javascript code before a request is run. It is used for performing pre-processing tasks like setting variables, parameters, headers, body data, etc., are performed using it.

20) How can we use Custom Javascript libraries in our scripts with an example?

A:- Postman provides a lot of built-in tools and libraries that we can use to add in our pre-request or post-request scripts or test cases. Let us take the example of using the moment.js library. It provides a lot of useful functions to format data around time. Consider that we have a POST request that needs to specify the created date to the user which expects the format "DD/MM/YYYY". We can use the moment library to perform this using a single line of code. In our pre-request script, we need to add the below lines of code to get the correctly formatted data and then store that in an environment variable:

```
var moment = require('moment');  
pm.environment.set('createdDate', moment().format('DD/MM/YYYY'));
```

There are a lot of other useful libraries like crypto.js that are useful for converting text to encrypted values which can further be used anywhere in the request body.

21) If we have a global and a local variable of the same name, which one will be given the most preference in Postman?

A:- In such cases, the higher precedence is given to the local variable by overwriting the value of the global variable.

22) What are workspaces in Postman? What are their uses?

A:- Workspaces are the areas/space given by Postman for team collaboration to work on a specific or set of collections. It provides a way to logically separate requests or collections that are personal to the developer or the team so that the maintenance of requests is made easy.

There are two types of workspaces in Postman:

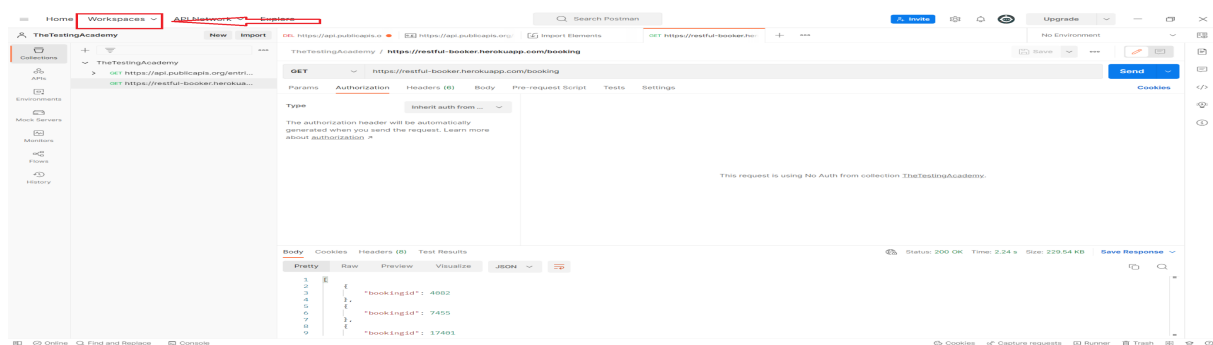
- Personal Workspace:
 - These workspaces are useful when we are working simultaneously on multiple projects and we require logical separation between the requests to handle the requests better.

- Team Workspace:
 - These are created for team collaboration so that more than one person can be part of testing requests.
 - We can invite new users to collaborate on our collections by sharing the email id of the users. Once the invite is accepted, the new users can start contributing to the workspace by adding or modifying the requests.

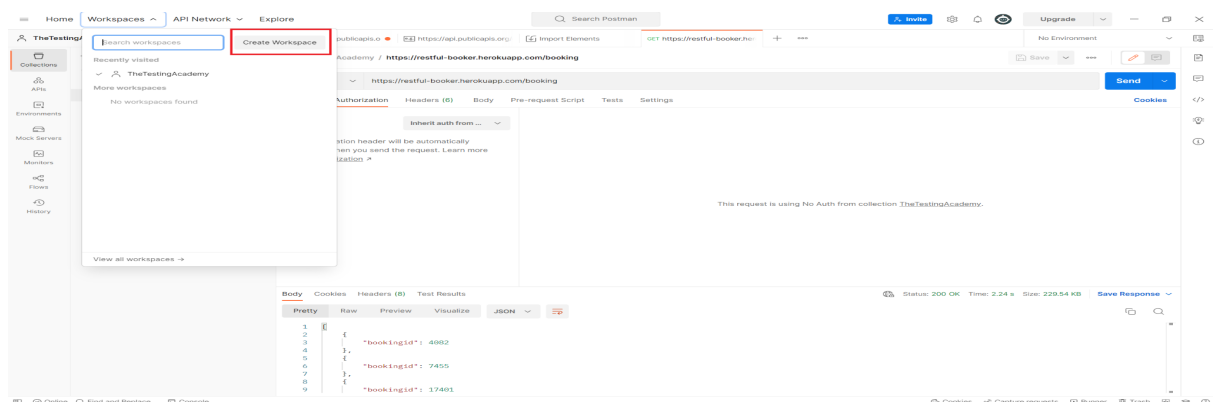
We can create a new workspace by clicking on the Workspace icon and then clicking on "Create New".

We can select our workspace to be personal or team workspace by configuring the properties in the create window.

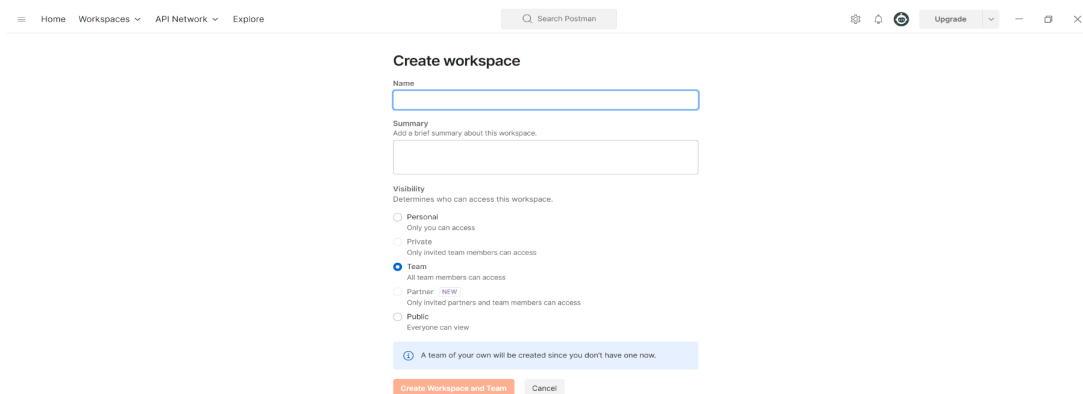
Step1



Step2



Step3 :- add the details and Create Workspace.



The screenshot shows the 'Create workspace' form in Postman. The form has three main sections: 'Name' with a text input field, 'Summary' with a text area, and 'Visibility' with radio button options. The 'Team' option is selected under 'Visibility'. A blue informational message states: 'A team of your own will be created since you don't have one now.' At the bottom are two buttons: 'Create Workspace and Team' (orange) and 'Cancel' (grey).

Home Workspaces API Network Explore Search Postman Upgrade

Create workspace

Name

Summary
Add a brief summary about this workspace.

Visibility
Determines who can access this workspace.

- ☐ Personal
Only you can access
- ☐ Private
Only invited team members can access
- ☒ Team
All team members can access
- ☐ Partner NEW
Only invited partners and team members can access
- ☐ Public
Everyone can view

A team of your own will be created since you don't have one now.

Create Workspace and Team Cancel

23) Does Postman allow flexibility to make use of the command-line?

A:- Postman provides a command-line tool called Newman using which we can run any Postman collection. It is a NodeJS based package that requires a node environment for executing collections using Newman Collection Runner. It has full parity with Postman's Collection Runner i.e it provides support for running assertions, pre-request scripts, or other request scripts linked with the requests that belong to the collection.

We can use Newman by following the below steps:

- Install Node
- Install Newman package using npm command as: `npm install -g newman`
- To run the collection, first export the environment to JSON format in Postman.

Then run the below command for running the collection in Newman:

```
newman run {{path to collection json}} -e {{path to environment json}}
```

24) How will you generate random numbers of a given range in Postman?

A:- Suppose you want to generate numbers between the range 1 to N, then it can be done in the pre-request script as follows:

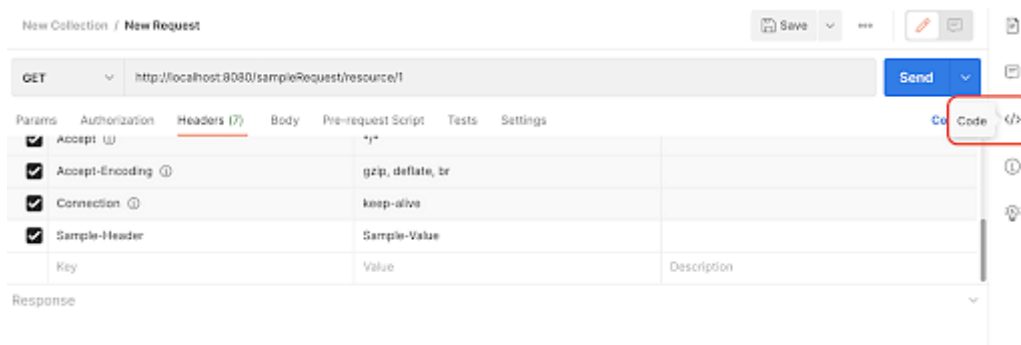
```
pm.globals.set('randomNumber', Math.floor(Math.random() * N));
```

We can then use this variable in the URL as: `{{randomNumber}}`

25) How do you get the cURL command based on the details of the REST API obtained from Postman?

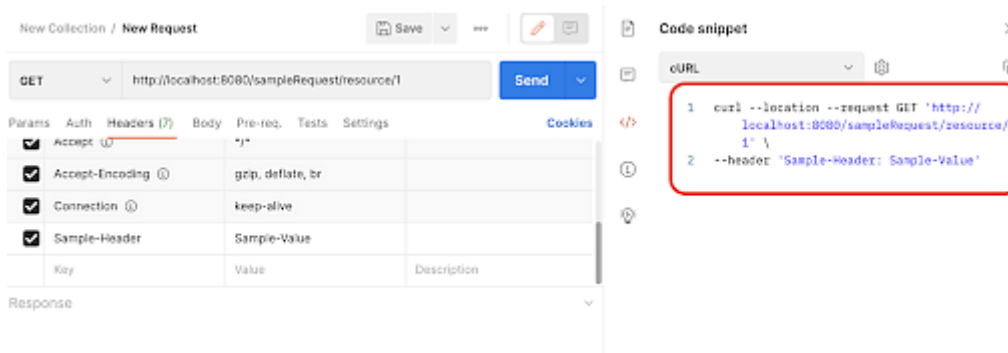
A:- You can use the steps listed below to get the cURL command equivalent:

As indicated below, click the Code icon.



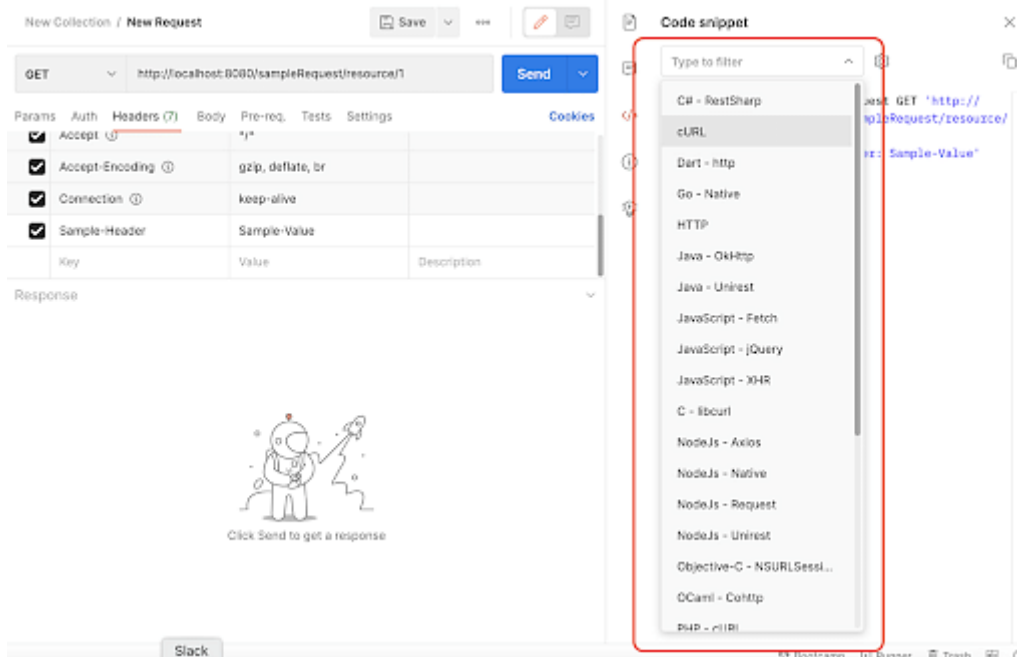
Step 1 - Click on Code

In response to the REST API request, you will receive the cURL command:



Step 2 - REST API request

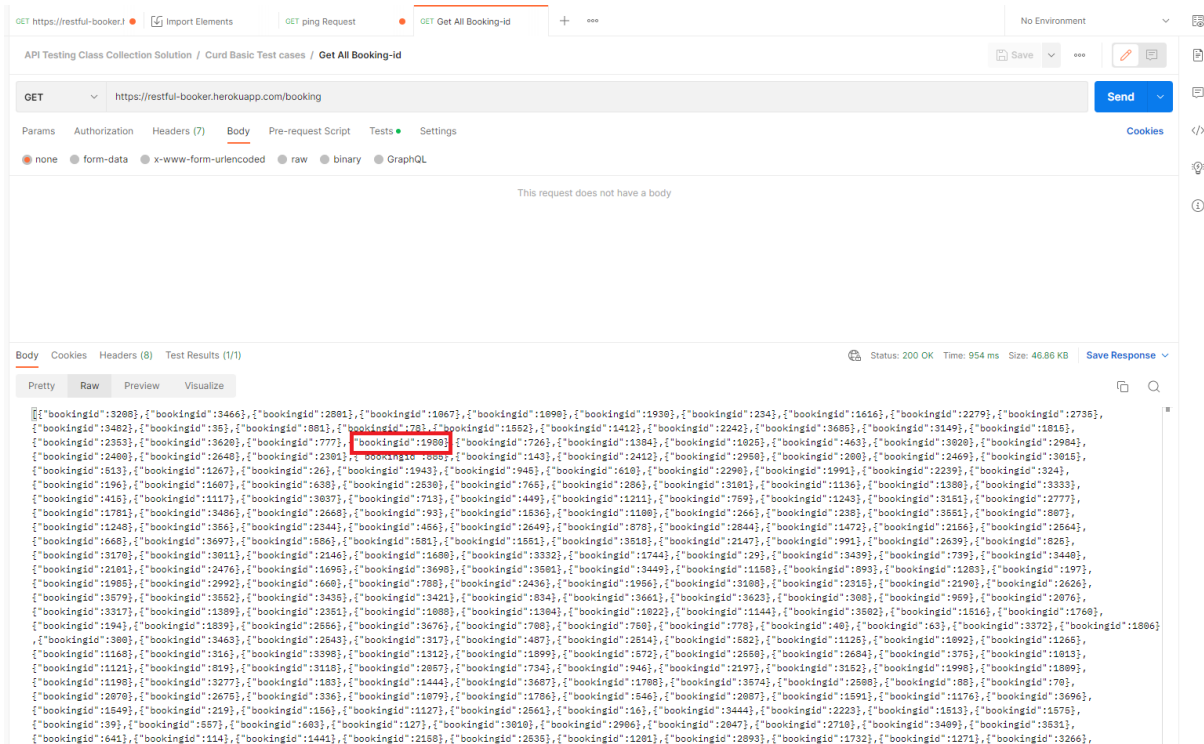
By choosing the necessary choice from the drop-down, as shown in the image below, we can also obtain the command for the request in several languages, such as C#, Javascript, NodeJs, PHP, etc.



Step 3 - Select Language

26) How to Find a specific Booking id From the array.

A :



The screenshot shows the Postman interface for a REST client request. The request is a GET to `https://restful-booker.herokuapp.com/booking`. The **Tests** tab is active, displaying a JavaScript script for finding a user ID. The script is as follows:

```
1 pm.test("Find The User ID", function () {
2
3
4   var jsonData = pm.response.json();
5
6   for (var i = 0; i < jsonData.length; i++) {
7     var counter = jsonData[i];
8     if(counter==1980){
9       console.log("We found it")
10    }
11  }
12 });
13
14
```

The **Test Results** tab at the bottom shows a single result: **PASS** Find The User ID.

Here is the code

```
pm.test("Find The User ID", function () {

var jsonData = pm.response.json();

for (var i = 0; i < jsonData.length; i++) {

    var counter = jsonData[i];

    if(counter==1980){

        console.log("We found it")

    }

}

});
```

27) How to Check that API response Contains Specific data?

GET filter by specific id + ... No Environment

API Testing Class Collection Solution / Curd Basic Test cases / filter by specific id

Save ...

GET https://restful-booker.herokuapp.com/booking/1112 Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

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

This request does not have a body

Body Cookies Headers (8) Test Results (1/1) Status: 200 OK Time: 228 ms Size: 414 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "firstname": "John",
3   "lastname": "Smith",
4   "totalprice": 111,
5   "depositpaid": true,
6   "bookingdates": {
7     "checkin": "2018-01-01",
8     "checkout": "2019-01-01"
9   },
10  "additionalneeds": "Breakfast"
11 }
```

API Testing Class Collection Solution / Curd Basic Test cases / filter by specific id

Save ...

GET https://restful-booker.herokuapp.com/booking/1112 Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

```
1 pm.test("Breakfast text Found in API respoce", function () {
2   pm.expect(pm.response.text()).to.include("Breakfast");
3 });
```

Test scripts are written in JavaScript, and are run after the response is received. Learn more about [tests scripts](#)

Snippets

- Get an environment variable
- Get a global variable
- Get a variable
- Get a collection variable
- Set an environment variable

Body Cookies Headers (8) Test Results (1/1) Status: 200 OK Time: 228 ms Size: 414 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "firstname": "John",
3   "lastname": "Smith",
4   "totalprice": 111,
5   "depositpaid": true,
6   "bookingdates": {
7     "checkin": "2018-01-01",
8     "checkout": "2019-01-01"
9   },
10  "additionalneeds": "Breakfast"
11 }
```

Here is the code

```
pm.test("Breakfast text Found in API response", function () {  
  
pm.expect(pm.response.text()).to.include("Breakfast");  
  
});
```

28) How to pass User id, First name, Last Name in the API URL?

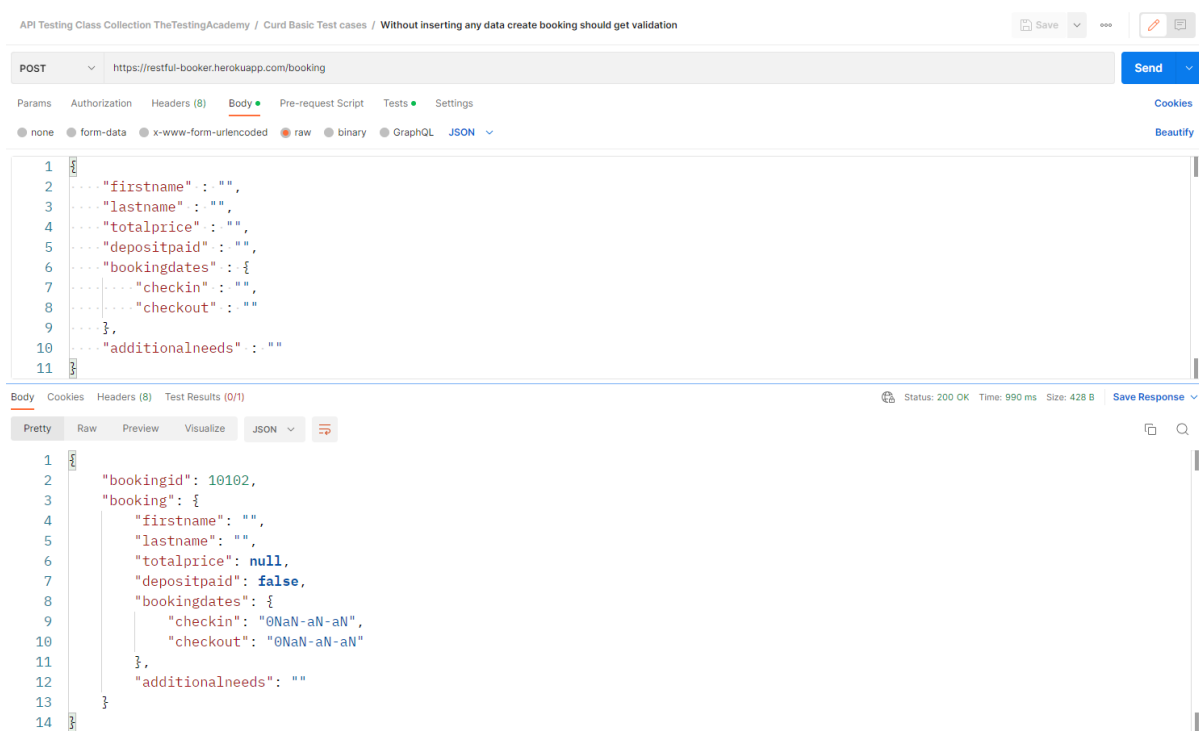
A:- <https://restful-booker.herokuapp.com/booking?firstname=Jim&lastname=Smith>

:- <https://restful-booker.herokuapp.com/booking/1112>

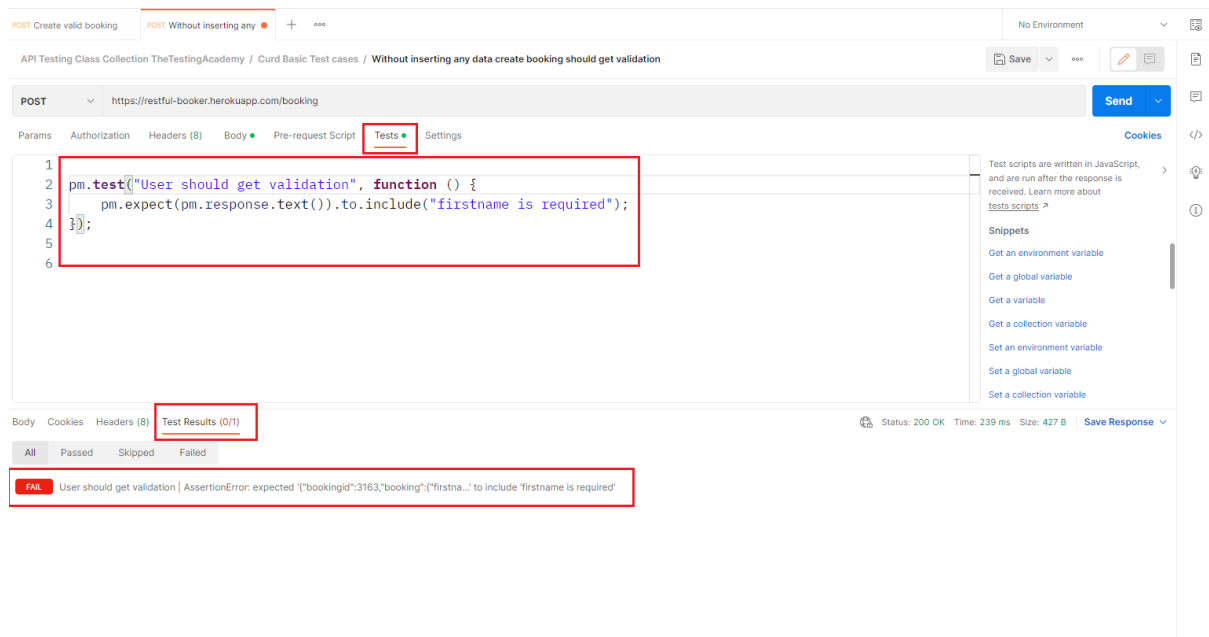
:- <https://restful-booker.herokuapp.com/booking?checkin=2018-01-01&checkout=2019-01-01>

29) How to Create Booking Without inserting data in the parameters in postman

In the image below we're Creating post with no data.



In the below image we're Checking that User getting the validation message or not after Creating the Booking without any data.

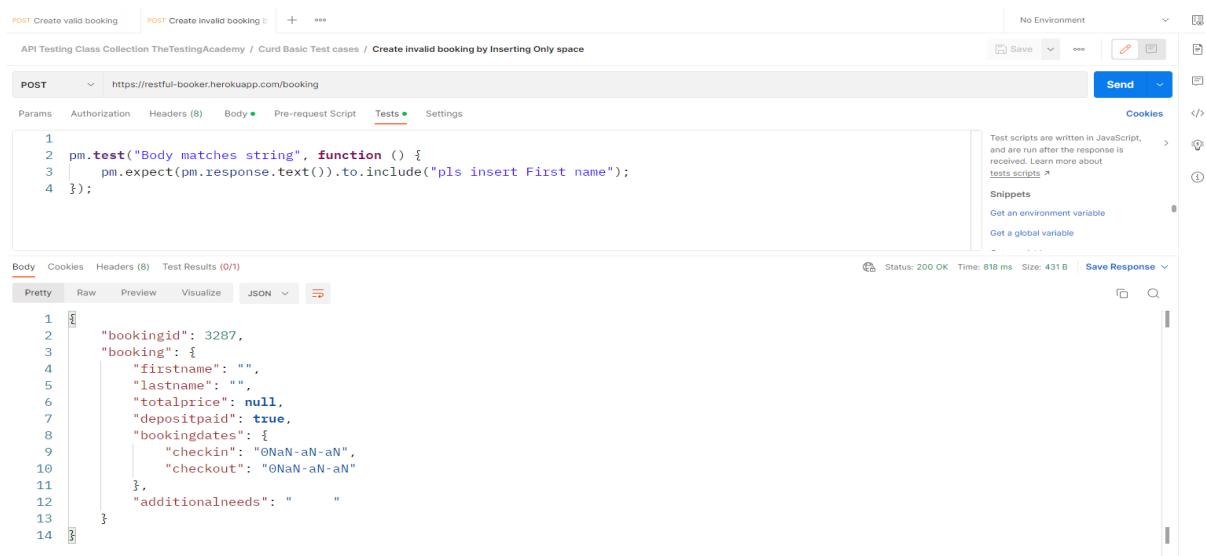
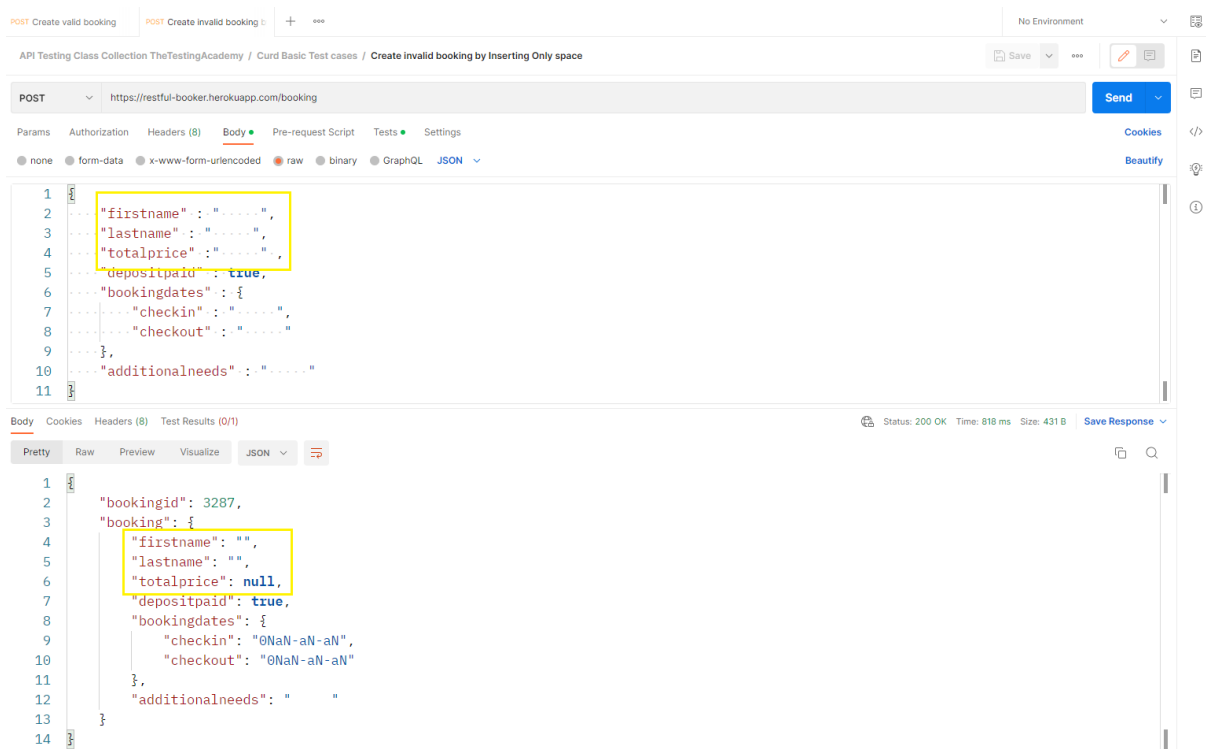


Here is the code

```
pm.test("User should get validation", function () {  
  
    pm.expect(pm.response.text()).to.include("first name is  
required");  
  
});
```

30) How to Create Booking By inserting space in the parameters in postman and validate the user getting any validations or not?

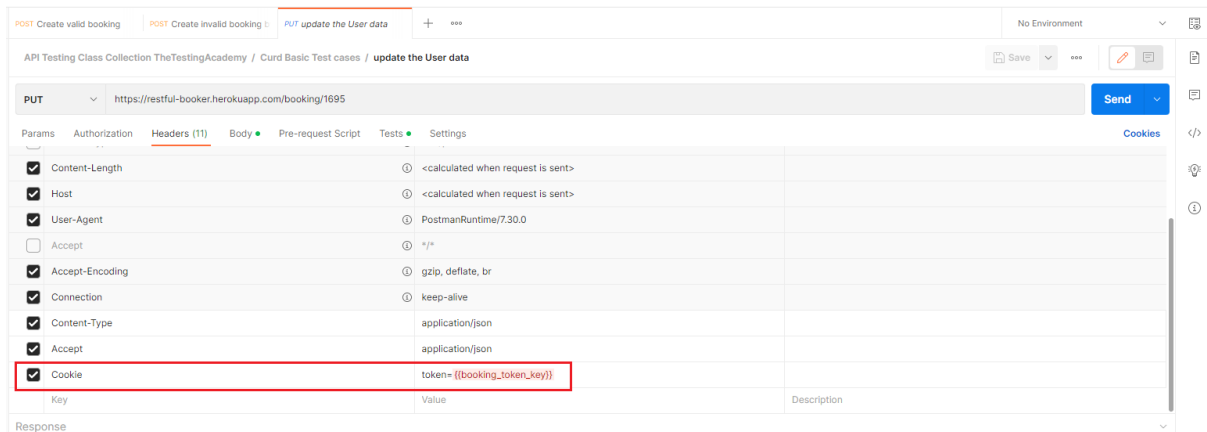
A:- in the below image you can see that we passed the space in the First name, last name and other parameters. And in 2nd image we're checking the validation in the Test section.



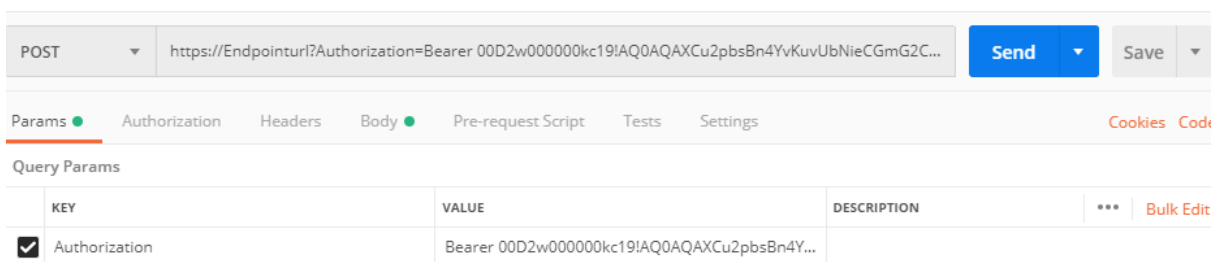
31) How can we pass a token into the postman?

A:- We can pass the token in the Header, in the URL Using the params and From the Authorization section using the Oath/Outh 2.0

As per below image, Passing The Token in Header.



As per below image, passing a token in The URL.



As per below image, passing a token in The Oauth 2.0

The screenshot shows the Postman interface with the 'Authorization' tab selected. On the left, under 'TYPE', 'OAuth 2.0' is chosen. Below it, a note states: 'The authorization data will be automatically generated when you send the request. [Learn more about authorization](#)'. Under 'Add authorization data to', 'Request Headers' is selected. A red bracket on the left side of the 'Configure New Token' section is labeled 'Token Generation Information inside the request'. The 'Configure New Token' section contains the following fields:

- Token Name: New access token
- Grant Type: Authorization Code
- Callback URL: http://localhost:3000/redirect
- ☐ Authorize using browser
- Auth URL: http://localhost:3000/auth
- Access Token URL: http://localhost:3000/token
- Client ID: {{clientId}}
- Client Secret: {{clientSecret}}
- Scope: read
- State: State
- Client Authentication: Send as Basic Auth header

A 'Get New Access Token' button is located at the bottom right of the configuration area.

32) How to verify the expected and actual Name is the same in the postman test?

A:- as per the 1st image we created the Booking that Contains First name as "TheTestingAcademy".

API Testing Class Collection TheTestingAcademy / 1) Integration - Create a Booking, Update the Booking Name, Get The Booking By ID and Verify / Update only First Name

PATCH `https://restful-booker.herokuapp.com/booking/{TC1_bookingid}` Send

Params Authorization Headers (11) **Body** Pre-request Script Tests Settings

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

```

1 {
2   "firstname": "TheTestingAcademy"
3 }

```

Body Cookies Headers (8) Test Results (1/1) Status: 200 OK Time: 247 ms Size: 426 B Save Response

Pretty Raw Preview Visualize **JSON**

```

1 {
2   "firstname": "TheTestingAcademy",
3   "lastname": "par",
4   "totalprice": 1018,
5   "depositpaid": true,
6   "bookingdates": {
7     "checkin": "2018-01-01",
8     "checkout": "2019-01-01"
9   },
10  "additionalneeds": "Breakfast"
11 }

```

As per the below image we're verifying that after creating the Booking the First Name is **"TheTestingAcademy"** or not.

API Testing Class Collection TheTestingAcademy / 1) Integration - Create a Booking, Update the Booking Name, Get The Booking By ID and Verify / Update only First Name

PATCH `https://restful-booker.herokuapp.com/booking/{TC1_bookingid}` Send

Params Authorization Headers (11) **Body** Pre-request Script **Tests** Settings

```

1
2 pm.test("Verify Updated UserName", function () {
3   var jsonData = pm.response.json();
4   pm.expect(jsonData.firstname).toEqual("TheTestingAcademy");
5 });

```

Test scripts are written in JavaScript, and are run after the response is received. Learn more about [tests scripts](#)

Snippets
Get an environment variable

Body Cookies Headers (8) Test Results (1/1) Status: 200 OK Time: 247 ms Size: 426 B Save Response

Pretty Raw Preview Visualize **JSON**

```

1 {
2   "firstname": "TheTestingAcademy",
3   "lastname": "par",
4   "totalprice": 1018,
5   "depositpaid": true,
6   "bookingdates": {
7     "checkin": "2018-01-01",
8     "checkout": "2019-01-01"
9   },
10  "additionalneeds": "Breakfast"
11 }

```


TheTestingAcademy