**Postman Tutorial: Api Testing Using Postman**

**Tutorial #1:** [Postman Introduction (This Tutorial)](https://www.softwaretestinghelp.com/api-testing-using-postman/)  
**Tutorial #2:** [How To Use Postman For Testing Diff API Formats](https://www.softwaretestinghelp.com/how-to-use-postman/)  
**Tutorial #3:** [Postman: Variable Scopes And Environment Files](https://www.softwaretestinghelp.com/postman-variables/)  
**Tutorial #4:** [Postman Collections: Import, Export And Generate Code Samples](https://www.softwaretestinghelp.com/postman-collections-import-export-generate-code/)  
**Tutorial #5:** [Automating Response Validations with Assertions](https://www.softwaretestinghelp.com/postman-assertions-automating-response-validation/)  
**Tutorial #6:** [Postman: Pre Request And Post Request Scripts](https://www.softwaretestinghelp.com/postman-request-scripts/)  
**Tutorial #7:** [Postman Advanced Scripting](https://www.softwaretestinghelp.com/postman-advanced-scripting/)  
**Tutorial #8:** [Postman – Command-Line integration with Newman](https://www.softwaretestinghelp.com/postman-newman/)  
**Tutorial #9:** [Postman – Reporting templates with Newman](https://www.softwaretestinghelp.com/newman-reporting-templates/)  
**Tutorial #10:** [Postman – Creating API Documentation](https://www.softwaretestinghelp.com/postman-api-documentation/)

**Postman Introduction**

POSTMAN is an API client used to develop, test, share and document APIs. It is used for backend testing where we enter the end-point URL, it sends the request to the server and receives the response back from the server. The same thing can be accomplished through API Templates like Swagger as well. In both Swagger and POSTMAN, we do not have to build a framework (unlike Parasoft) to fetch the response from the service.

This is the main reason for which POSTMAN is used frequently by the developers and automation engineers to ensure that the service is up and running along with the build version of an API that is being deployed into the region.

It essentially helps in hitting API endpoints by quickly creating requests as per the API specification and dissecting the various response parameters like the status code, headers, and the actual response body itself.vaScript if it is disabled in your browser.

**Postman offers a lot of advanced features like:**

* API development.
* Setting up Mock endpoints for APIs that are still under development.
* API documentation.
* Assertions for the responses received from API endpoint execution.
* Integration with CI-CD tools like Jenkins, TeamCity, etc.
* Automating API tests execution etc.

Now, we have gone through the formal introduction of the tool, let’s proceed to the installation part.

**Postman Installation**

**Postman is available in 2 options.**

1. As a Chrome app (this is already deprecated and has no support from the Postman developers)
2. Native App for different platforms like Windows, Mac OS, Linux, etc.

As Chrome apps are being deprecated and have a tight coupling with Chrome browser (in a few cases the actual browser version), we will focus mostly using the Native application that gives us more control and has lesser external dependencies.

**Postman Native App**

Postman Native app is a standalone app that is available for [download](https://www.getpostman.com/downloads/) in various OS platforms like Windows, Mac OS, Linux, etc. This can be downloaded just like any other application depending upon the user’s platform.

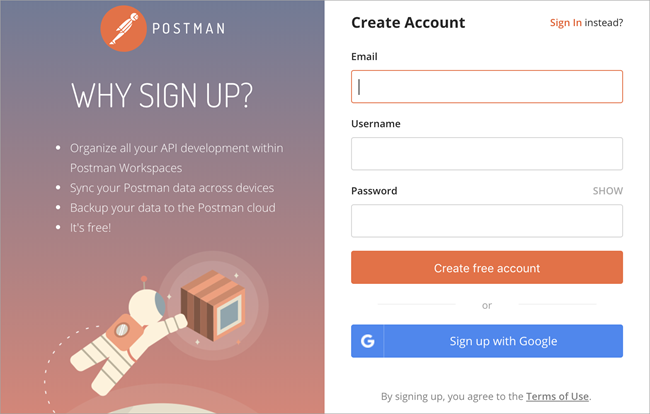
The installation process is pretty straightforward as well. You simply need to double click on the downloaded installer (for Windows and Mac) and follow the instructions.



Once the installation is successful, simply open the Postman Application to get started with.

We will see how to create a simple request for any available open-source API and see the various components of the request and response received when the request is executed using the Postman application.

It’s highly recommended, to sign-in/sign-up into the Postman application using an existing email account. A signed-in account preserves all the Postman collections and requests that are saved during the session and ensures that the requests are still available to work with when the same user logs-in next time.



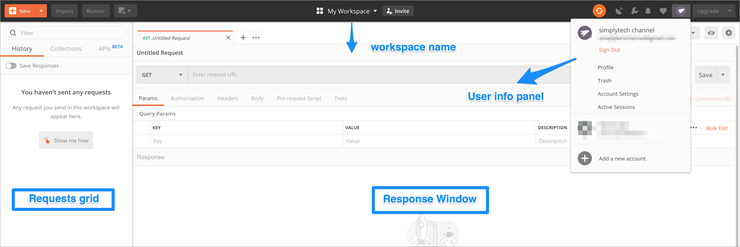
Please refer to the **Note**section to get details about the publicly available fake API endpoint.

We will illustrate a sample GET request to this [URL](https://jsonplaceholder.typicode.com/posts) which would return 100 posts in response as a JSON Payload.

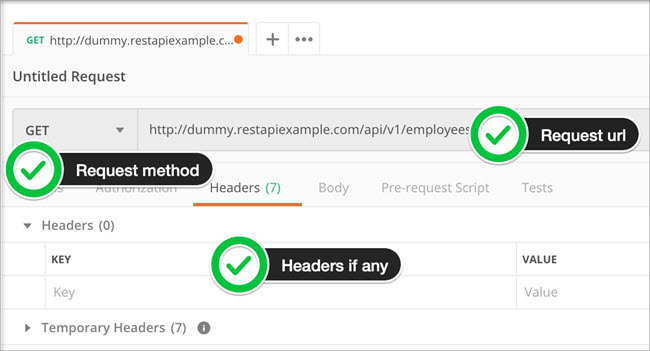
**Let’s get started and look at the steps that need to be followed:**

**#1)** Open the Postman Application (If not already logged in with the existing or new account, first log in with the appropriate credentials).

**Given below is the image of the Postman UI initial screen:**



**#2)** Create a new request and fill in the details as per the endpoint that we will be using for our test or illustration. Let’s test a get request for a REST API endpoint http://dummy.restapiexample.com/api/v1/employees



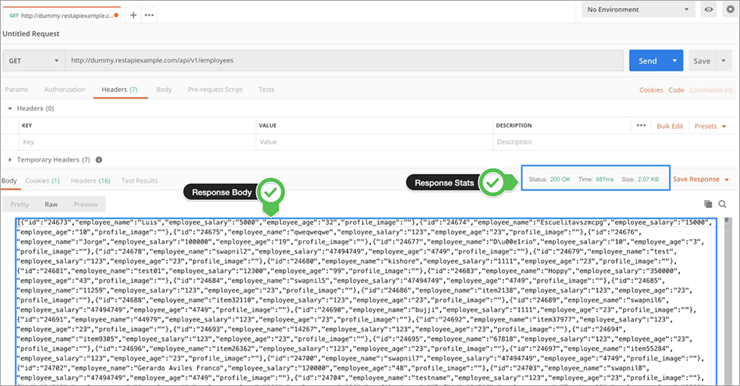
**#3)** Once the request properties are filled in, hit SEND to execute the request to the server hosting the endpoint.



**#4)** Once the server responds, we can check the various data around the response.

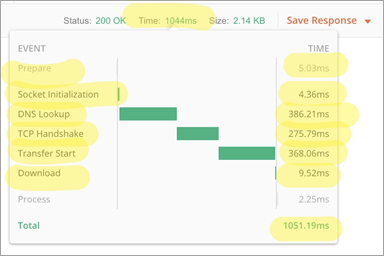
***Let’s see each of them in detail.***

By default, once the response completes, the response body tab is selected and displayed. Other parameters for a response like the response status code, the time taken for the request to complete, the size of the payload are shown just below the request headers (as in the below figure).

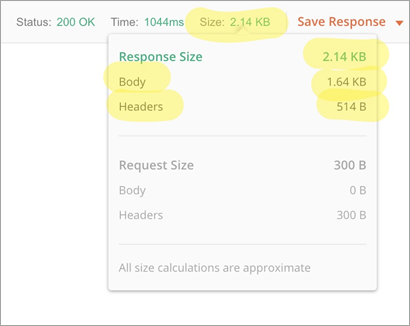


To get fine-grained details about the response parameters like Response size and Response time, you can simply hover over each of those values, and Postman will show you a detailed view with more fine-grained details for each of those properties.

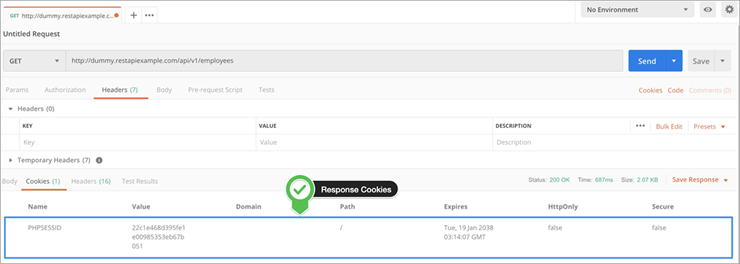
**For example,** for Request time – it will further dissect it into individual components like Connect time, Socket time, DNS lookup, Handshake, etc.



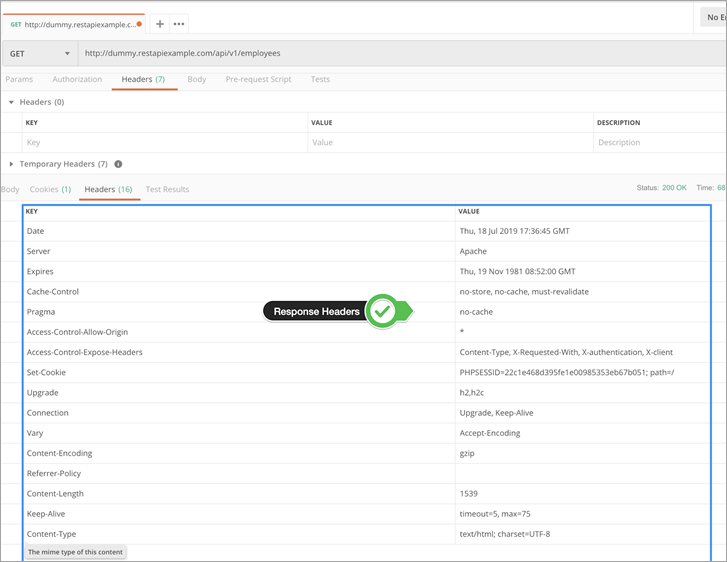
Similarly, for Response size, it will show you a breakup of how much size the headers are constituted, and what is the actual response size.



Now, let’s look at the other response tabs i.e. Cookies and Headers. In the web world, Cookies hold a lot of importance in terms of driving client-side experiences and lots of session related info to get all the information about the cookies that were returned from the server. You can switch to the cookies tab and see this.



Similarly, response headers contain a lot of beneficial information about the request that got processed. Just navigate to the headers tab in the response section to take a look at the response headers.



An important point to note here is that all the requests that you make to the server are stored in the Postman history for future reference (The History tab is available on the left side panel of the app).

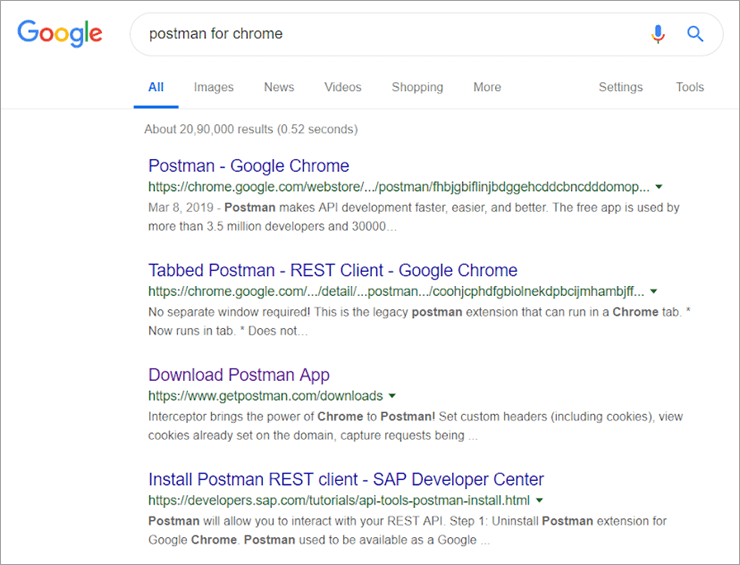
This helps in preventing the creation of requests every time when you need to get a response for the same request and also helps in avoiding mundane boilerplate tasks. If required, you can refer to the past requests (And responses as well) at a future point of time.

**Note:** To illustrate sample requests and responses, we will be using publicly available fake API servers that will allow all types of HTTP requests to be made and that return a valid HTTP response.

**Alternative Quick Postman Installation Guide**

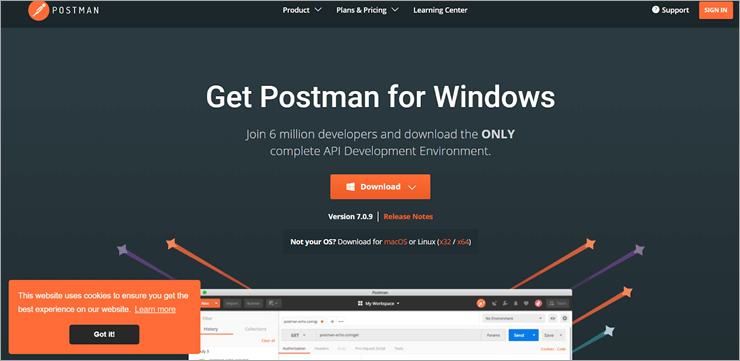
POSTMAN is an open tool and is available for anyone who surfs the internet. You can follow the below steps and get the POSTMAN tool installed in your local machine.

**Step 1:** Open [Google](https://www.google.com/) and search for the POSTMAN tool. You will get the below search-result. Then you can click on Download Postman App and you will be redirected to the getpostman website.



Else, you can directly navigate to this [URL](https://www.getpostman.com/downloads/) to get the POSTMAN tool.

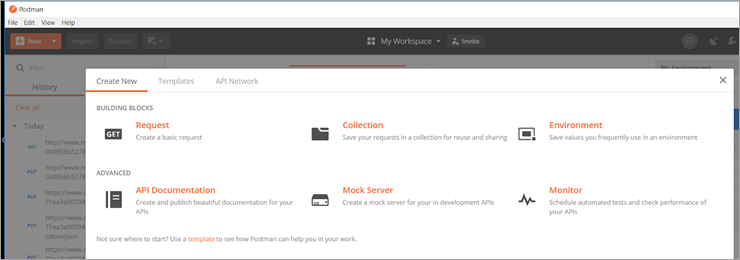
**Step 2:** Select the POSTMAN version based on your operating system. In our case, we are going to use POSTMAN for Windows OS. Moreover, we are using Window-64 bit, so we will download and install POSTMAN for 64 bit.



**Step 3:** Once you have clicked on the Download button, a postman.exe file will be download into your local. Click on that file. It’s a one-click installation just like any other application that will let you install the POSTMAN add-on for your browser.

**Step 4:** After you have installed the application, click on the application (which must be placed on your desktop). As you can see in the below image, we have six different entities for which you will basically need three building blocks i.e. Request, Collection, and Environment which will be discussed in the next section.

That’s it!! We have successfully installed and launched the POSTMAN application.



**Building Blocks Of POSTMAN**

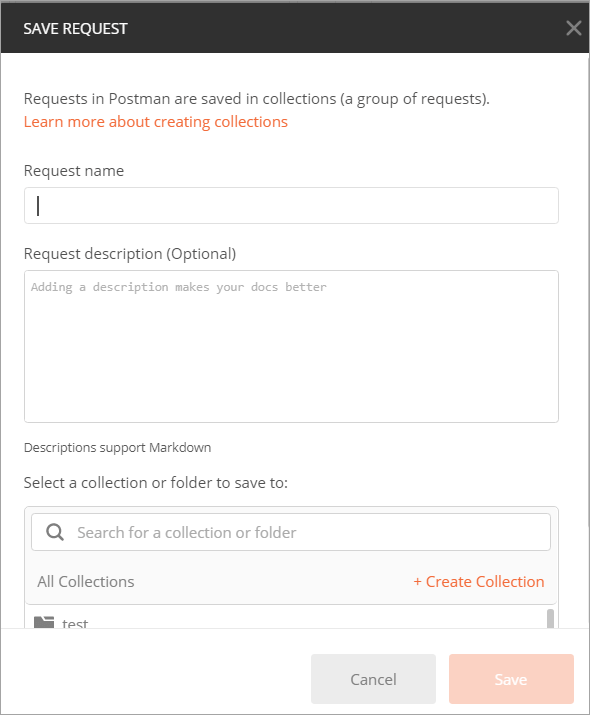
POSTMAN has a variety of building blocks but for our purpose, we are going to discuss the three major building blocks that are essential for every POSTMAN operation.

These three major building blocks are:

#1) Request

A request is nothing but a combination of the complete URL (which includes all parameters or keys), HTTP headers, body or payload. These attributes altogether form a request. POSTMAN lets you save your request and this is a good feature of the app that lets us use the same request as many times as we want.

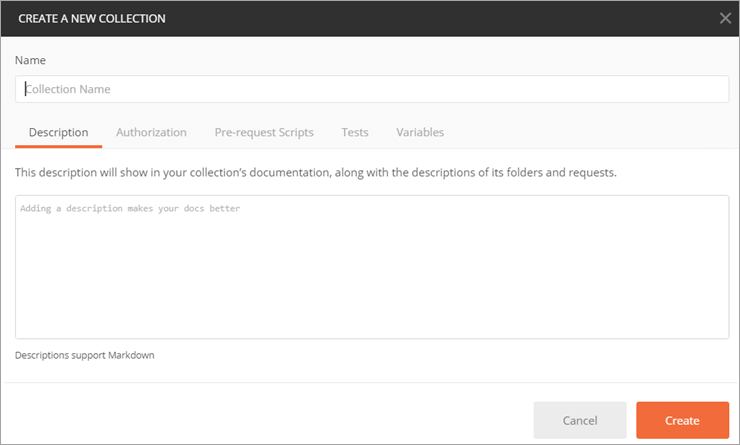
Click on **New -> Request**



#2) Collection

There must be something where you will save your bulk requests. This is the scenario where collection comes into the picture. We can say that a collection is a repository in which we can save all our requests. Generally, the requests that hit the same API are kept in the same collection.

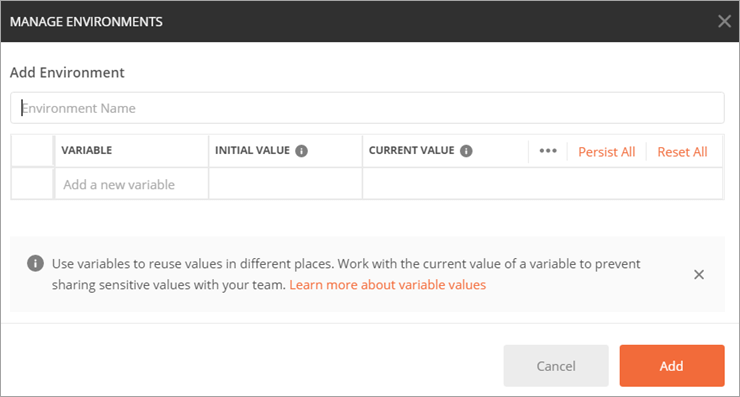
Click on **New -> Collection.**



#3) Environment

An Environment is a region where all your operations on an API will take place. It could be TUP, QA, Dev, UAT or PROD. Each project will already have the regions configured and you just have to declare your global variables such as URL, token’s id and password, context keys, API keys, document keys and so on in it.

Click on **New -> Environment.**



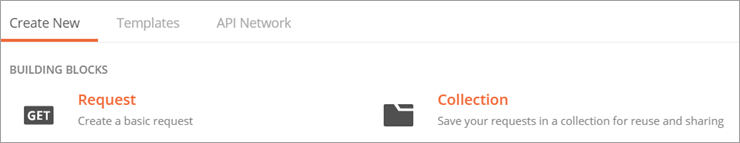
**Saving A Request Into Collection**

Now we will try saving a sample request into a collection and we will use the same request to hit an API.

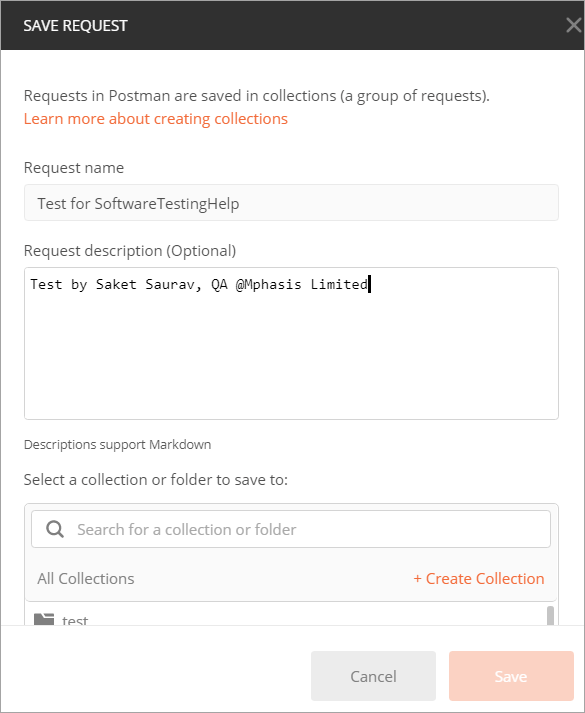
**Step 1:** In the top right corner, you will see the “+New” button. Click on this button and you will have the list of the building blocks that were shown when you launched the application for the first time.



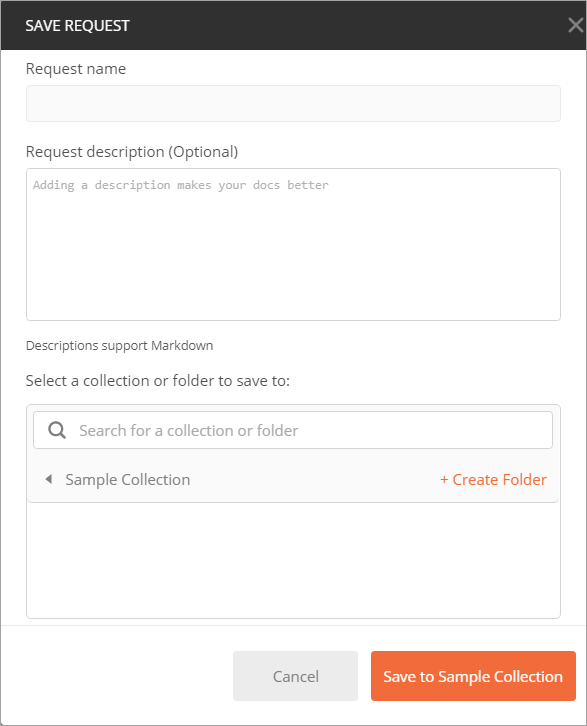
**Step 2:** Click on Request.



**Step 3:**Provide the request name which is a mandatory field. Then click on “+ Create Collection”.



**Step 4:**Once you have clicked on “+ Create Collection”, it will ask for a name (say Sample Collection). Input the name of the collection and hit enter.



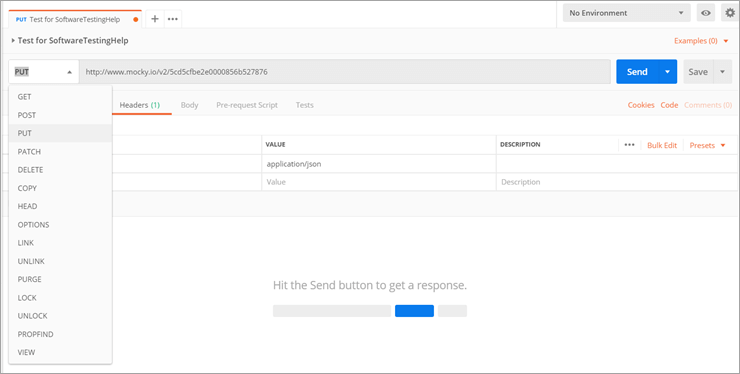
**Step 5:**Click on the **“Save to Sample Collection”**button.

**Sample Request And Response**

This particular section will give you a deep insight into how to test an API in POSTMAN.

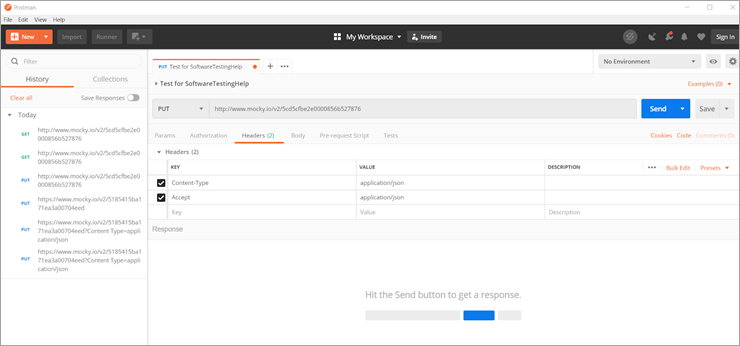
As you can see in the below image, we have our Request which we have already created (Test for SoftwareTestingHelp). Moreover, you can see a drop-down (just beside the URL) which has the verbs or methods that are supported by the POSTMAN.

These are called HTTP verbs. We will try to update something using the PUT method and then we will retrieve the same using the GET method. I assume readers are aware of the functionality of these HTTP verbs that are used in API testing.



Now, we have a URL and a request method. All we require is the headers and payload or body. In some cases, we need to generate tokens (based on the needs of an API).

We will declare our HTTP headers i.e. Content-Type and Accept. Accept is not always compulsory as it decides the format in which we will retrieve our response. By default, the response is always JSON.



There is no need to mug up the values of these headers as POSTMAN will provide you the suggestions when you type in the text areas of the key and value.

Then, we will move on to the next mandatory section which is Body. Here we will provide the payload in the form of JSON. We are aware of how to write our own JSON, hence we will try to create our own JSON.

**Sample Request**

**URL:** <http://www.mocky.io/v2/5cd6c3f23000004b006061c5>

**Headers**

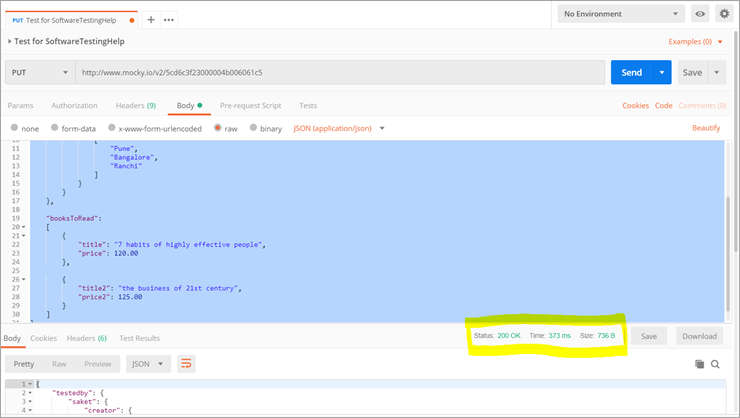
Content-Type : application/JSON  
Accept = application/JSON

**Body**

|  |
| --- |
| {      "testedby":      {          "saket":          {              "creator":              {                  "name": "Software Testing Help",                  "location":                  [                      "Pune",                      "Bangalore",                      "Ranchi"                  ]              }          }      },        "booksToRead":      [          {              "title": "7 habits of highly effective people",              "price": 120.00          },            {              "title2": "the business of 21st century",              "price2": 125.00          }      ]  } |

**Now Hit It**

Once you have a complete request, click on the “Send button” and see the response code. A 200 OK code stands for successful operation. In the image below you can see that we have successfully hit the URL.

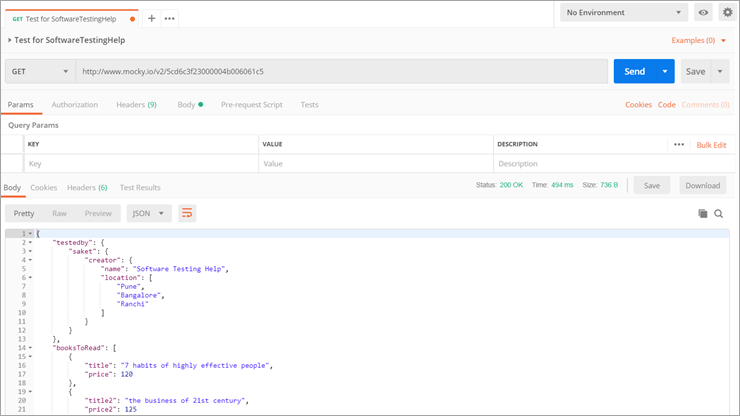


**The Next Step**

Now, we will perform another operation called GET. We will try to fetch the same record that we have just created.

We don’t require a body or payload for the GET operation. As we already have our sample request using the PUT method, all we require is to change the method to GET.

Once we have changed to GET we will hit the service again. As you can see in the below image, we have exactly got what we passed and this is how POSTMAN works.



**Update: Additional Information**

**What is an API?**

API (Application Programming Interface) is a jar file, which contains several methods and interfaces to perform a particular action.

**Refer to the below Example and screenshot:**

1. Create a sum method, which adds two variables and returns the sum of two variables.
2. Then create a calculator class that contains several other methods like addition, subtraction, multiplication, division and so on. There may be some helper classes as well. Now combine all the classes and interfaces and create a jar file called Calculator.jar and then publish it. Use Calculator API to access the methods present inside.
3. Some APIs are open source (Selenium) which can be edited and some are licensed ones (UFT) which cannot be edited.

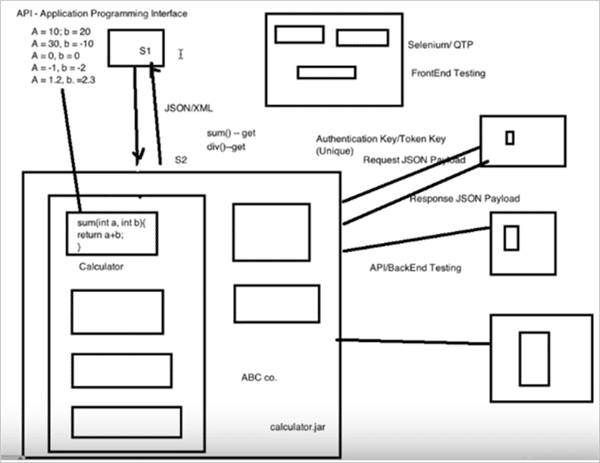
**How exactly are these methods getting called?**

Developers will expose an interface, a platform to call the calculator API and we create the object of calculator class and call the sum method or any method.

Suppose this calculator.jar file is created by some company and they use this utility through UI interface, then we test this calculator application using UI and automate it using QTP/Selenium and this is called Front End Testing.

Some applications don’t have UI, thus in order to access these methods, we create an object of the class and pass on the arguments to test and this is called [Back-End Testing](https://www.softwaretestinghelp.com/how-to-perform-backend-testing/). Sending the request and receiving back the response will happen through [JSON](https://www.softwaretestinghelp.com/json-tutorial/)/XML files.

**Refer to the below diagram:**

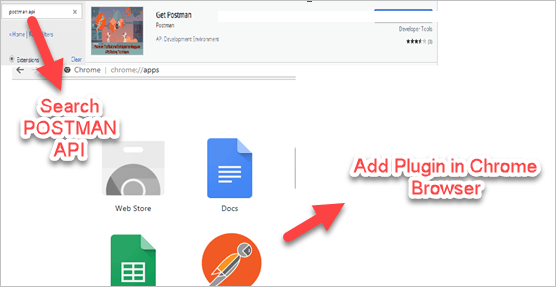


**POSTMAN Client**

* POSTMAN is a REST client used for performing backend API testing.
* In POSTMAN, we pass the API call and check the API response, status codes and payload.
* Swagger is another HTTP Client tool where we create API documentation and through swagger, we can also hit the API and get the response.
* Refer the link https://swagger.io/
* You Can use either Swagger or POSTMAN to test the APIs, and it depends on companies as which client to use.
* In POSTMAN mostly we use GET, POST, PUT and DELETE calls.

How to Download POSTMAN Client?

Open Google Chrome and download the POSTMAN app available in the Chrome app store.



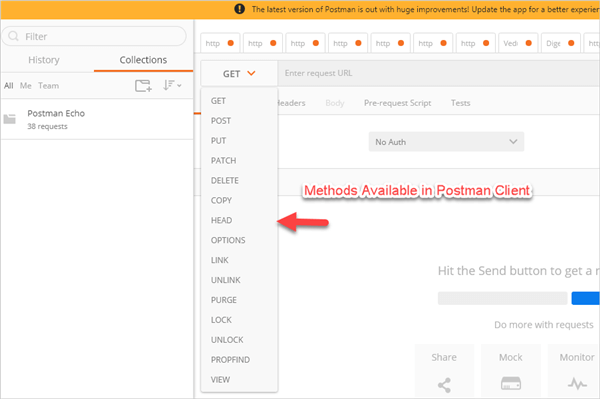
**Call REST APIs Using POSTMAN Client**

In POSTMAN we got many methods but we use only GET, PUT, POST and DELETE

* **POST** – This Call creates a new entity.
* **GET** – This call sends the request and receives the response.
* **PUT** – This call creates a new entity and updates the existing entity.
* **DELETE**– This call deletes the existing entity.

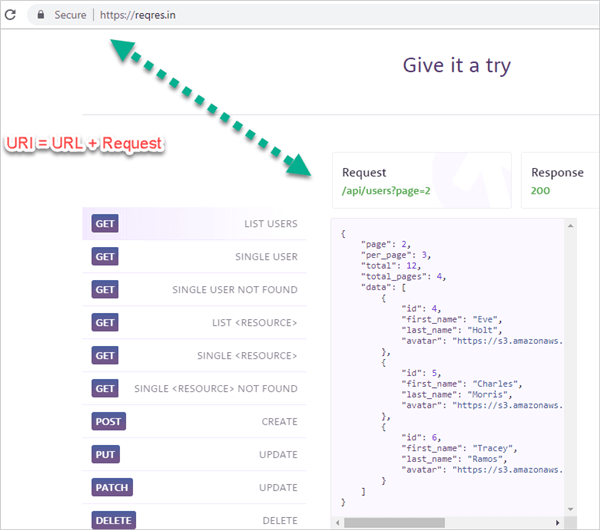
API’s can be accessed either by using UI like banking platform or where UI is not available like the backend systems where we use REST API client like POSTMAN.

Other clients are also available like SOAP UI which is a REST and SOAP client, advanced REST clients like [JMeter](https://www.softwaretestinghelp.com/jmeter-tutorials/)can call the APIs directly from the browser. POSTMAN is the best tool for performing POST and GET operations.



**Send a request and get a response in POSTMAN Client:**

For testing purpose, we use the API’s provided [here](https://reqres.in/).



Check each CRUD call in the POSTMAN client using the API’s provided by the dummy site.

**In API testing we mainly validate the below points:**

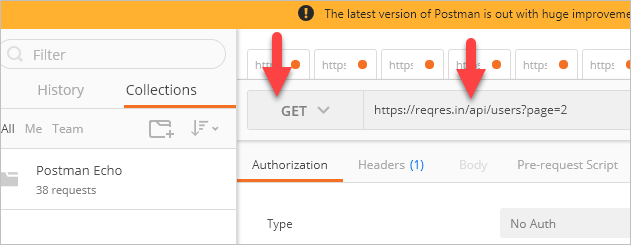
* The response status codes, refer [wiki page](https://en.wikipedia.org/wiki/List_of_HTTP_status_codes) for details.
* The response message and the JSON response body.
* In API testing, first, ask the developer to give the endpoint URL. See this endpoint URL https://reqres.in/.

#1) GET Call

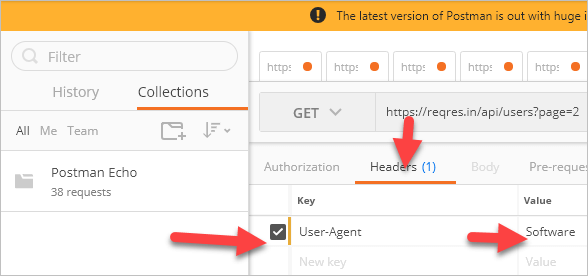
**Sends the request and receives the response.**

**Steps for testing REST API:**

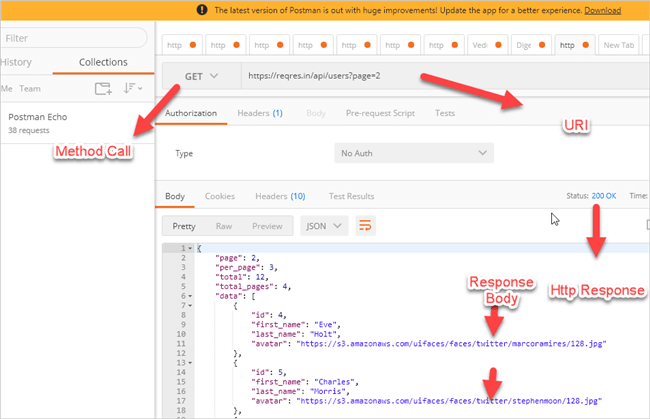
* Pass **https://reqres.in//api/users?page=2** [? is a query parameter which filters out the result like print all the information of the user in page 2, the query parameter depends upon the developer as how they will define] as URI in the POSTMAN client.
* The query parameter is defined by (?) and the path parameter is defined by (/).
* Select the GET method.



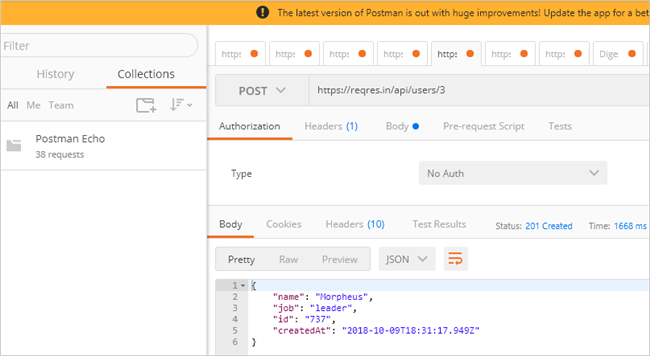
* Provide Headers (if required) like User-Agent: “Software”.



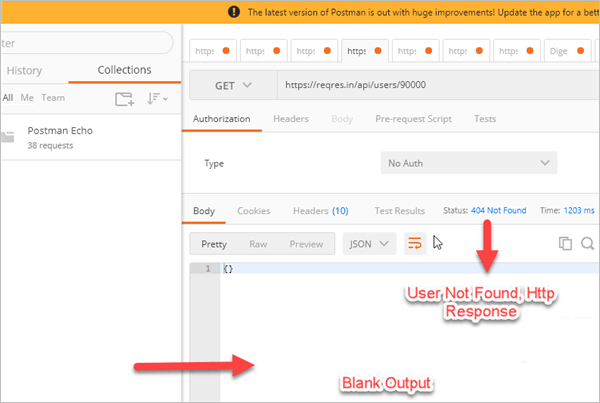
* Click on the SEND button.
* If API is working fine, in response we get:
  + Status 200 – OK, this means that the response is received successfully.
  + Response JSON Payload.
  + String Message



* Another **Example** of the GET METHOD, where we searched for information about a specific user i.e. user id =3. Enter URI = <https://reqres.in/api/users/3>



* If data is not available against our search, we get blank JSON and 404 status message.

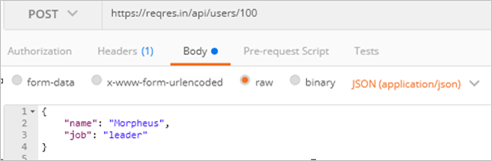


#2) POST Call

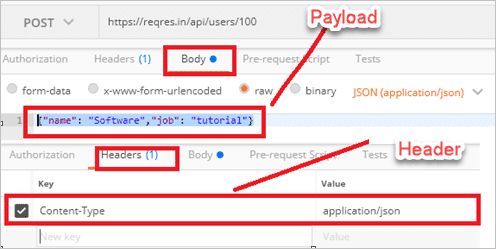
**Create a new user or an entity.**

**Steps to Execute:**

* Select a POST from the dropdown and use this service URL “https://reqres.in/api/users/100”  
  
* Go to Body – > select RAW -> As we are passing JSON.
* Select JSON from the dropdown and paste the payload script.
* Pass this payload {“name”: ”Morpheus”, ”job”: ”leader”}



* JSON starts with curly braces and stores data in the key, value format.
* Pass the header content type = application/json.
* Press the SEND button.



* On successful request, we get the below response:
  + Status 201 – Created, the response is received successfully.
  + Response Payload
  + Header

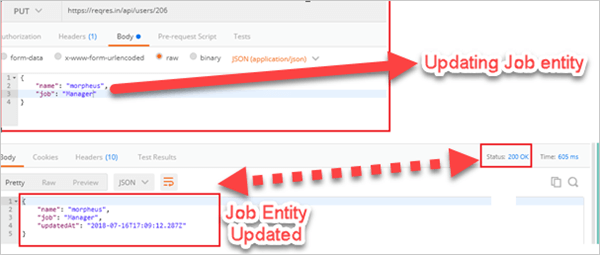


#3) PUT Call

Updates or creates a new entity.

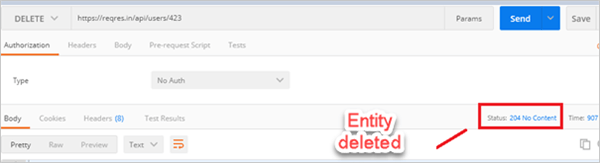
**Steps to create PUT call:**

* Use this service URL “https://reqres.in/api/users/206” and payload {“name”: “Morpheus”,”job”: “Manager”
* Go to POSTMAN client and select PUT method -> Go to Body – > Select RAW > pass JSON and select JSON from the dropdown and paste the payload script.
* JSON starts with curly braces and stores data in the key-value format.
* Press the SEND button, for a successful request, you will get the below response.
  + Status 200 – Ok, the response is received successfully.
  + Response Payload
  + Header
  + Job updated to “manager”

****

#4) Delete Call

* Delete the user, use this service URL “/api/users/423” and this payload {“name”: “Naveen”,”job”: “QA”}.
* Go to POSTMAN and select DELETE method, payload not required.
* Deletes user id =423 if available in the system.
  + Status 204 – No Content, the response received successfully.
  + No Payload received, user id deleted.
  + Header



**Challenges in API Testing**

* Test cases should be designed in such a way that it should cover test coverage.
* Designing test cases are simple when the API carries fewer parameters but the complexity increases when the number of parameters is large.
* Regularly update your test coverage with a change in the business requirement. If a new parameter is added increase the test harness suite
* Sequencing API calls properly.
* Explore boundary conditions and performance.

**Conclusion**

In this tutorial, we discussed the important pointers to get started with the Postman API testing tool. We learned to install the Postman tool as a standalone application and discussed how we can go about creating a simple request and look at the response generated.

We saw how to navigate to the different parts of the response information as well as how to review and retrieve the requests from the history tab.

We believe that by now, you can carry out a successful operation on an API. Successful operation on an API does not mean copy and paste of the entire body, headers, and other necessary blocks and making the test execution successful.

It’s about how comfortable you are in writing your own JSON, navigating to any particular field in a JSON with the help of the document key or params, understanding arrays in JSON, etc.

POSTMAN client tool is used for performing back-end testing and mainly to perform GET, PUT, POST, DELETE calls.

From this tutorial, we learned how to hit the calls from the POSTMAN client and how to validate the response which we get back from the server and also covered the challenges in API testing.

API testing is very important to find loopholes in APIs as hackers will exploit them and cause financial losses. API testing seems simple but we should be very cautious and try testing with every possible com.