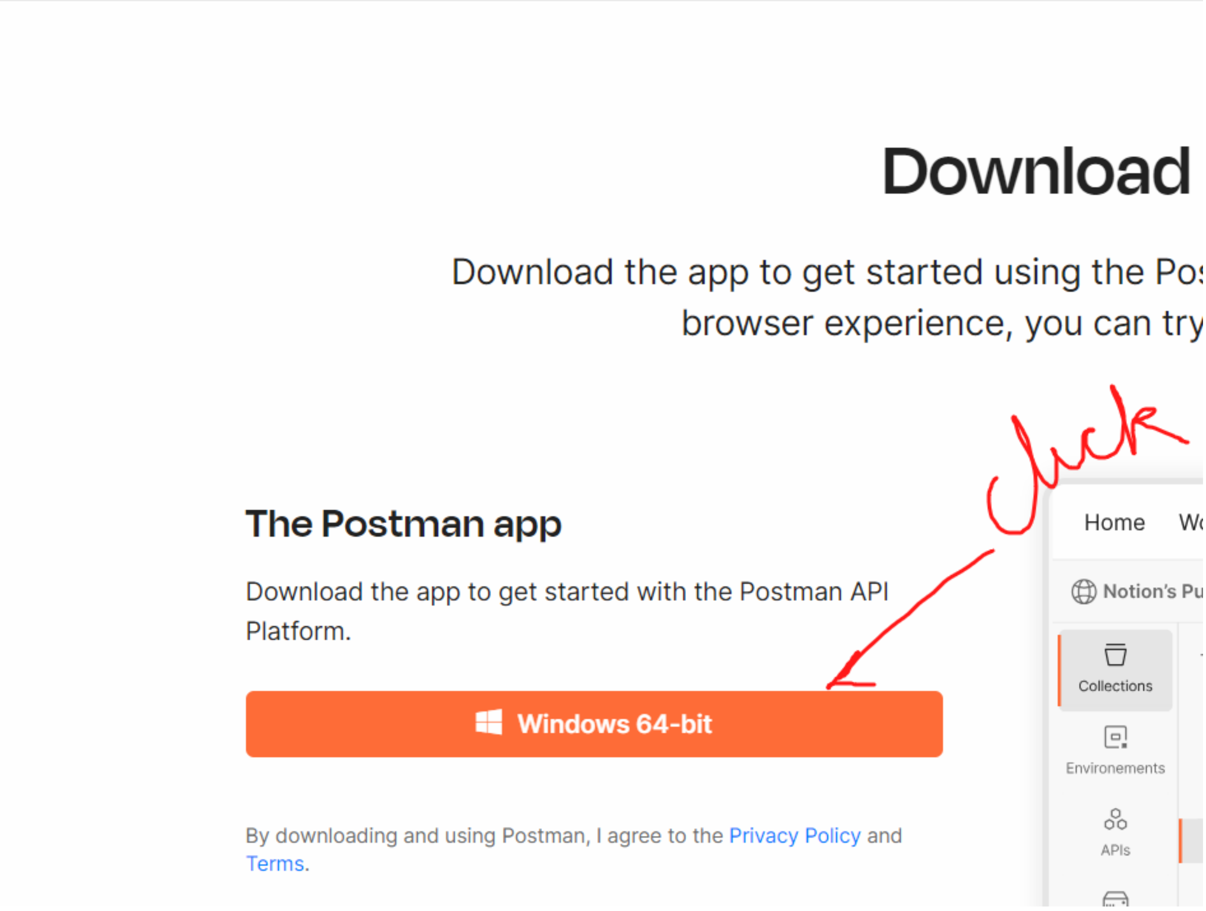
1) **Demonstrate how to set up Postman.**

2) https://www.postman.com/downloads/

3)



Download

Download the app to get started using the Pos
browser experience, you can try

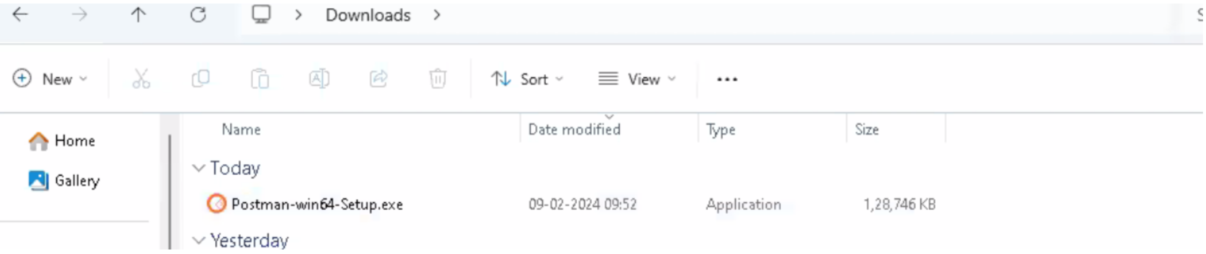
The Postman app

Download the app to get started with the Postman API Platform.

By downloading and using Postman, I agree to the Privacy Policy and Terms.

4)

5)



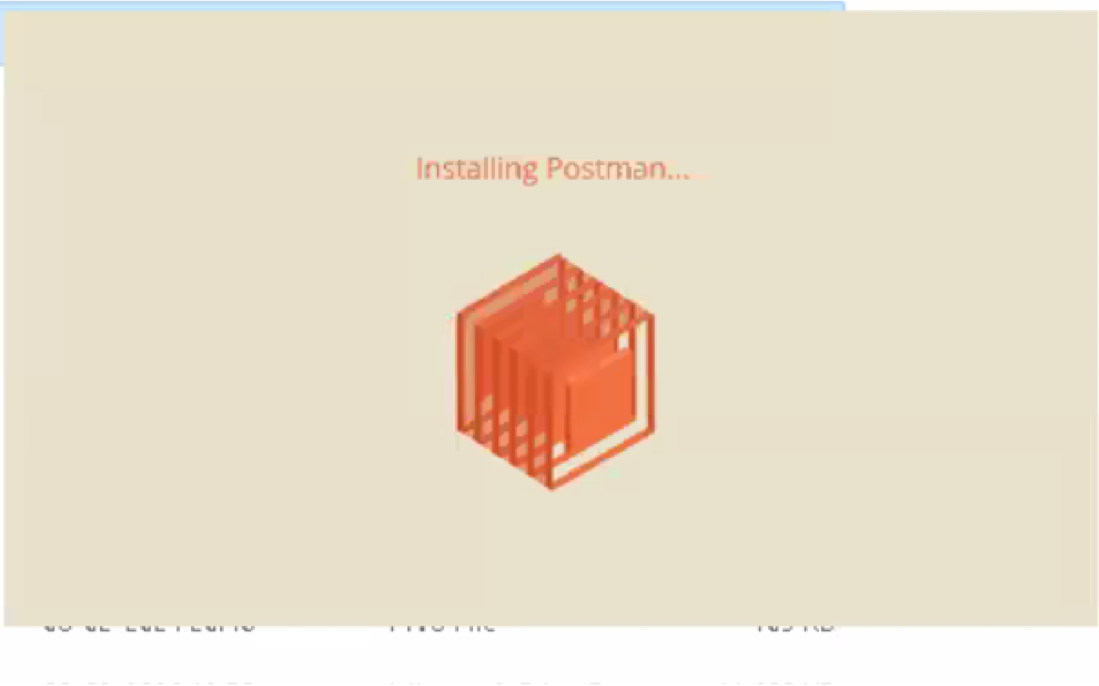
Name	Date modified	Type	Size
Today			
Postman-win64-Setup.exe	09-02-2024 09:52	Application	1,28,746 KB
Yesterday			

6)

7)

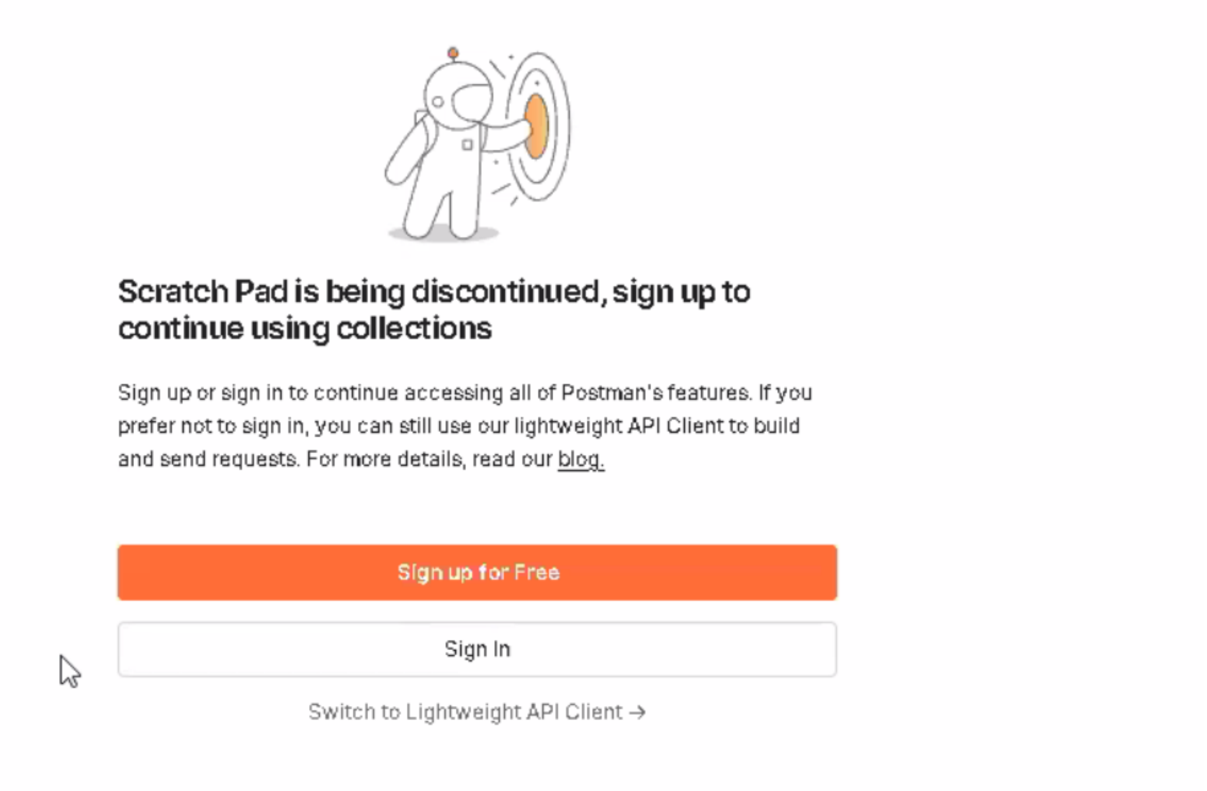
8) Double click on the downloaded file

9)



Installing Postman...

10)
11)
12)Click on signup for free
13)

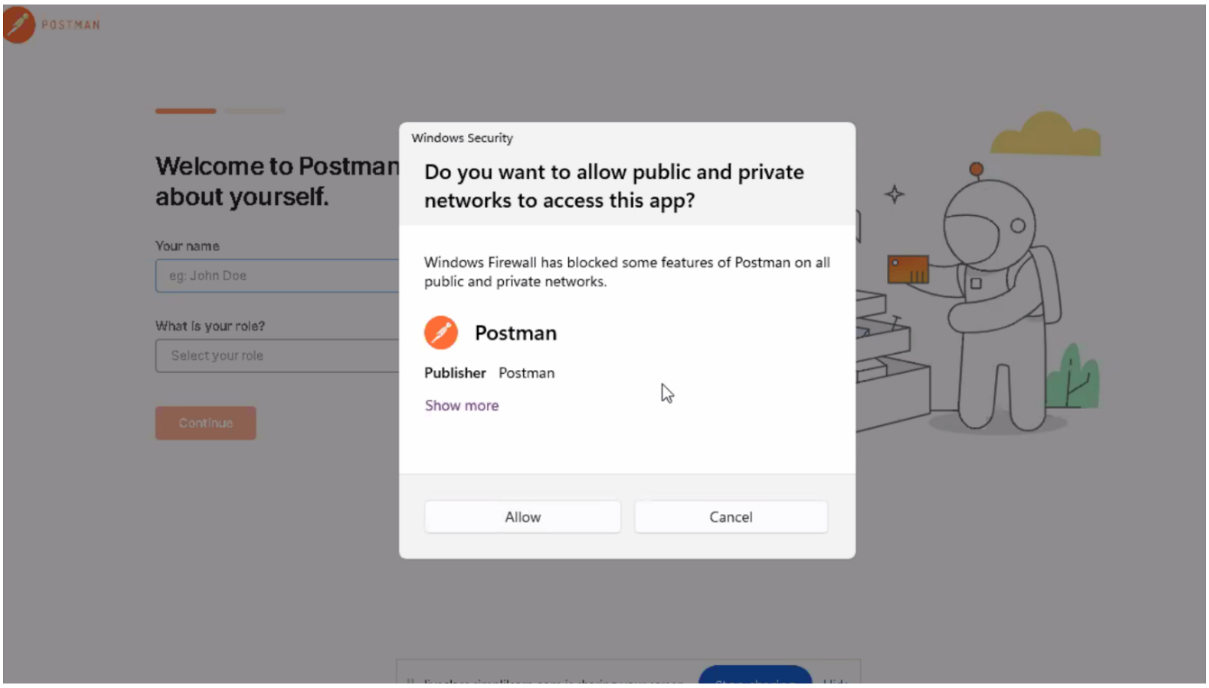


Scratch Pad is being discontinued, sign up to continue using collections

Sign up or sign in to continue accessing all of Postman's features. If you prefer not to sign in, you can still use our lightweight API Client to build and send requests. For more details, read our blog.

14)
15)
16)After you sign up for free

17)



Welcome to Postman about yourself.

Your name: eg: John Doe

What is your role? Select your role

Continue

Windows Security

Do you want to allow public and private networks to access this app?

Windows Firewall has blocked some features of Postman on all public and private networks.

Postman

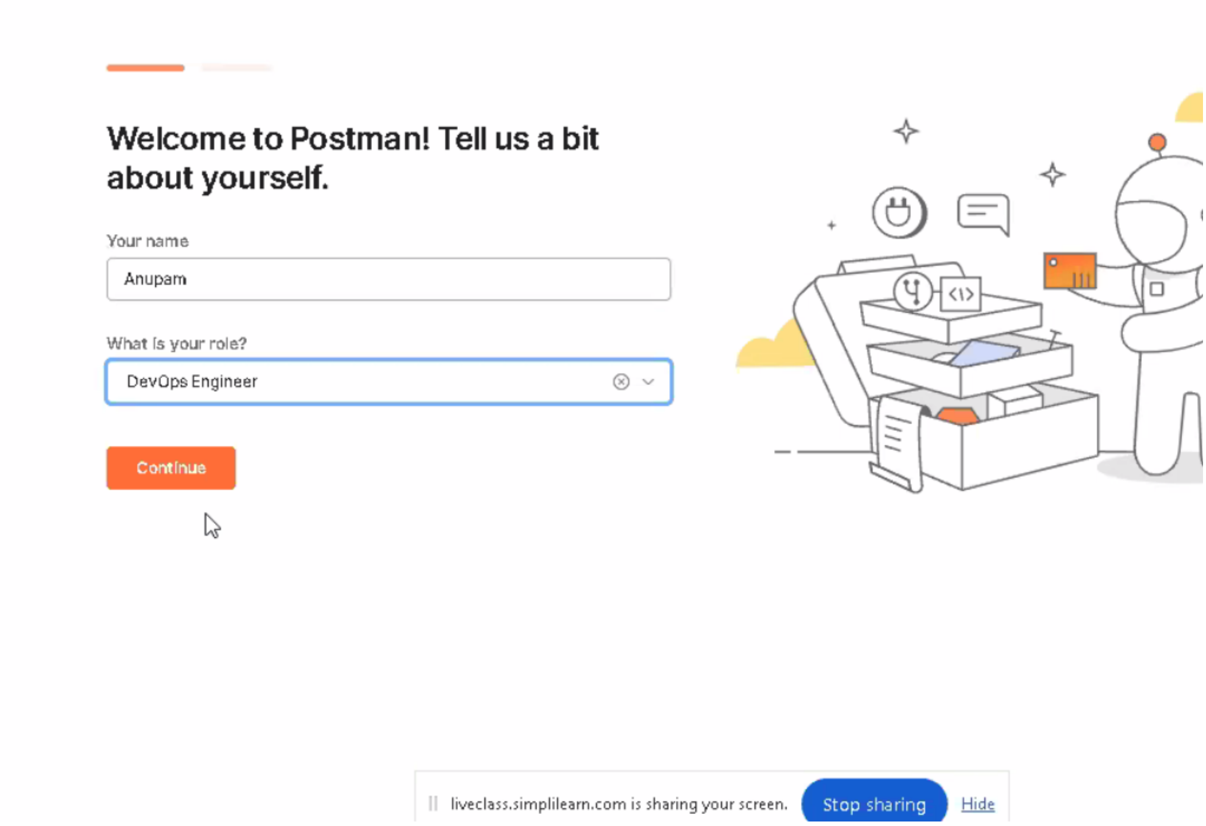
Publisher Postman

18)

19)

20)Click on Allow

21)



Welcome to Postman! Tell us a bit about yourself.

Your name: Anupam

What is your role?: DevOps Engineer

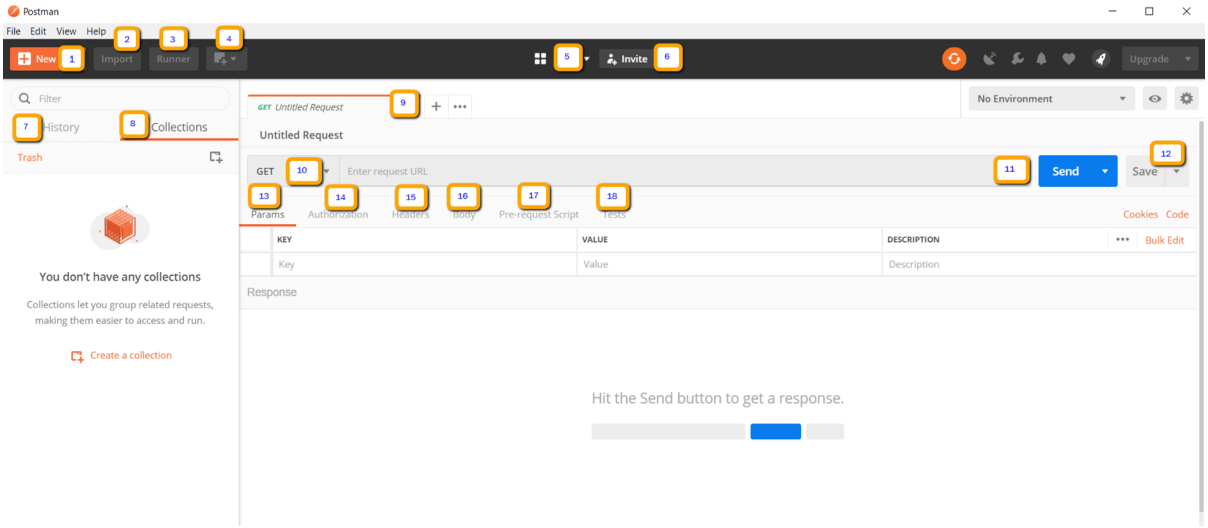
22)

23)Click on continue and you will be on the postman dashboard

2) Demonstrate how to create the first API request.

Setting up API request on Postman

- The screenshot below shows the Postman Workspace.



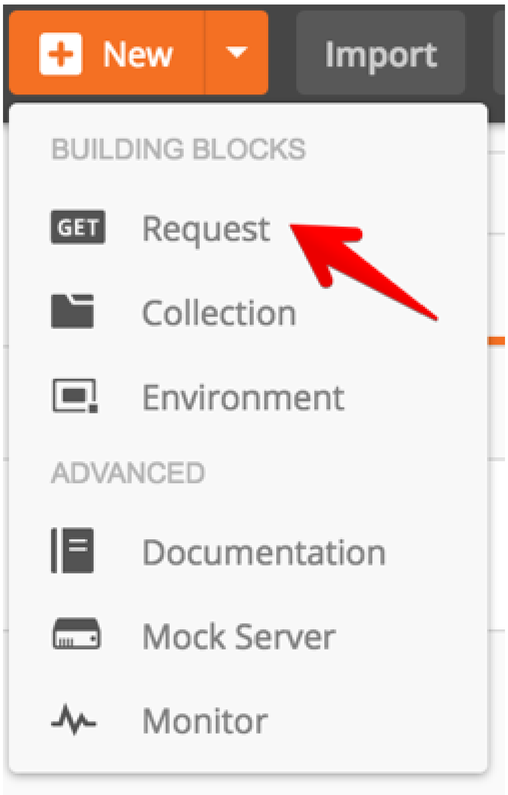
You don't have any collections

Collections let you group related requests, making them easier to access and run.

Create a collection

Hit the Send button to get a response.

- Open Postman Workspace.
- On the top left corner, click on the **New option** button.



- In the drop-down options available, click on Request option.
- Adda name for the request.

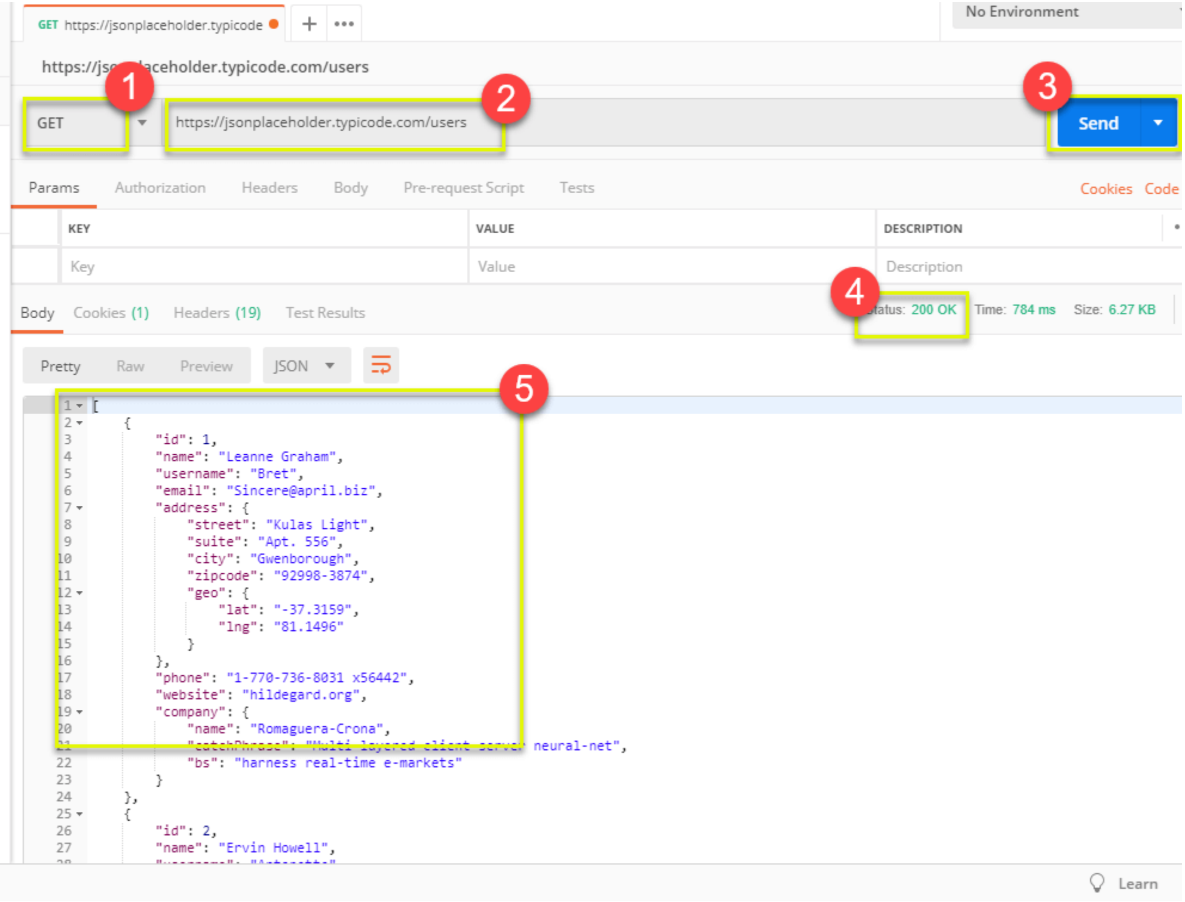
- Scroll down and click on create a collection option and add a name for the collection.
- Click on the Save button.
- Your Request is ready now.

Step 2.2.2: Getting a response of API request on Postman

- Take one Sample URL as shown below.

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

- In the request URL field, input the above link.
- Click on the Send button.
- You will see 200 OK Messages.



```
[
    {
        "id": 1,
        "name": "Leanne Graham",
        "username": "Bret",
        "email": "Sincere@april.biz",
        "address": {
            "street": "Kulas Light",
            "suite": "Apt. 556",
            "city": "Gwenborough",
            "zipcode": "92998-3874",
            "geo": {
                "lat": "-37.3159",
                "lng": "81.1496"
            }
        },
        "phone": "1-770-736-8031 x56442",
        "website": "hildegard.org",
        "company": {
            "name": "Romaguera-Crona",
            "catchPhrase": "Multi-layered client-server neural-net",
            "bs": "harness real-time e-markets"
        }
    },
    {
        "id": 2,
        "name": "Ervin Howell",
        "username": "Antonette",
```

3) Demonstrate how to use Postman with SOAP.

This guide has three subsections, namely:

2.3.1 Adding SOAP request in postman

2.3.2 Running and validating the response

2.3.3 Pushing the code to your GitHub repositories

Step 2.3.1: Adding SOAP request in Postman

- The screenshot below shows the Postman Workspace.

- Open Postman WorkSpace.
- On the top left corner, click on the New option button.

- In the drop-down options available, click on the Request option.
- Your Request is ready now.
- Take the given URL as the Soap request URL.

http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wso?WSDL

- In the request URL field, input the above link.
- Then set the method as Post.
- Set Body as raw and set text as XML.
- Provide data in the body.

<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">

- <Body>
- <CapitalCity xmlns="http://www.oorsprong.org/websamples.countryinfo">
- <sCountryISOCode>IN</sCountryISOCode>
- </CapitalCity>
- </Body>
- </Envelope>

- IN keyword in the above code block is the Country code for India.
- Click on the Send.

- Scroll down and click on the Body and verify the Response.

Step 2.3.2: Running and validating the response

- Click on Tests.
- Select Status code: Code is 200
- Click on Send again.
- Scroll down and click on TestResults and verify the response.
- You will see 200 OK Messages.

4) Explain how to work on GET requests, with an example.

Development Environment:

- Postman Tool
- URL

This guide has two subsections, namely:

2.4.1 Working with GET Request

2.4.2 Pushing the code to your GitHub repositories

Step 2.4.1: Working with Get Request

- We will use the following sample URL for all examples in this tutorial:

  https://reqres.in/api/users/2

- Set your HTTP request to GET
- In the request URL field, input link
- Click Send
- You will see the **200 OK** Message
- There should be 10 user results in the body

- There may be cases where the Get request will be unsuccessful. It can be due to an invalid request URL or missing authentication.

5) Demonstrate how to work with POST requests.

Step 2.5.1: Creating a POST Request

- Click a new tab to create a new request

- Set your HTTP request to POST
- Input the same link in request url:https://reqres.in/api/users
- Switch to body tab

Step 2.5.2: Writing the body of Request

- Click raw
- Click JSON
- In the code editor, add the following data to the file:

- Copy and paste just one user result from the previous get request like below. Ensure that the code has been copied correctly with paired curly braces and brackets. Change id to 11 and name it to any desired name. You can also change other details like the address.

```
  [ {
  "name": "morpheus",
  "job": "leader",
  "id": "463",
  "createdAt": "2019-08-28T10:51:59.088Z"
}
]
```

Step 2.5.3: Checking response

- Click Send
- Status: 201 Created should be displayed
- Request body data is displayed in the response body

6) Demonstrate how to parameterize requests.

This guide has four subsections, namely:

2.6.1 Checking response before creating Variable

2.6.2 Creating Variable

2.6.3 Getting the Response

2.6.4 Pushing the code to GitHub repositories

Steps 2.6.1: Checking response before creating Variable

- Set your HTTP request to GET
- Input this link: https://reqres.in/api/users?page=2. Replace the first part of the link with a parameter, such as {{url}}. Request url should now be {{url}}/users.
- Click Send

Note: There should be no response since we have not set the source of our parameter.

Steps 2.6.2: Creating a Variable

- Click the eye icon
- Click edit to set the variable to a global environment which can be used in all collections

- Set the name to the URL, which is https://reqres.in/api
- Click Save

- Click Close if you see the next screen

- Change the Environment

Steps 2.6.3: Getting the Response

- Always ensure that your parameters have a source, such as an environment variable or data file to avoid errors

7) Demonstrate how to create a Collection.

Steps 2.7.1: Creating a Collection

- Click on the NEW button
- Then you get a pop up like below:

Steps 2.7.2: Inputting the Desired name

- Click on the Collection button
- Then you can pass your desire name
- Click the Create button

Steps 2.7.3: Adding a new request in Collection

- Click on New
- Give a name

- Pass sample url (https://reqres.in/api/users?page=2)
- Save it

Steps 2.7.4: Running the Collection

- Go to the Collection option
- Click on Arrow
- Click on Run

- After that, you will get a new console
- Click on Run Collection1

Steps 2.7.5: Checking the Output

- The output will look like:

8) Demonstrate how to run a Collection using Collection Runner.

Working with Collection Runner

There are two ways to run a collection which is the Collection Runner and Newman.

Let's begin by executing the collection in Collection Runner.

- Click on the Runner button found at the top of the page next to the Import button.

- Collection Runner page should appear as below. The following is a description of the various fields:

Step 2.8.2: Run Collection Runner

- Choose Collection1: Set iterations as 3
- Set delay as 2500 ms
- Click on Run Collection1 button

- Run Results page should be displayed after clicking the Run button. Depending on the delay, you should see the tests as they execute.
- Once the tests have finished, you can see the test status if it is Passed or Failed and the results per iteration.
- You see Pass status for the Get Requests.
- Since we did not have any tests for Post, there should be a message that the request did not have any tests.

9) Demonstrate how to work with variables in Postman.

Step 2.10.1: Creating a variable

- Create a Collection

- Create two Requests

- Use these two URLs:

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

https://reqres.in/api/users/2

- Save both the Requests

- Go to Collection (three-dot mark) and click on it
- Click Edit

- Go to the Variable option
- Create a Variable

Step 2.10.2: Using the variable

- Use your variable and click Send

- The output will look like:

10) Create a request, and collections and variables using Postman.

Step 2.1.1: Problem statement for creating requests, collections, and variables using Postman.

- Objective: Create a collection, add a request, create respective variables in Postman using any sample url, and check the response.

- Steps Involved:

1. Open Postman
2. Create a collection
3. Add requests by using sample url
4. Create variable

5. Validate the response

Step 2.1.2: Solution for the problem statement.

- Steps to create a collection

1. Open Postman
2. Go to New option
3. Create a collection

- Steps to set requests

1. Go to New option
2. Click on Request
3. Save it
4. Use the sample url:

https://reqres.in/api/users/2

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

- Steps to set variables

1. Go to collection
2. Click on the Triple dot mark
3. Click Edit

4. Click on Variable

5. Add variable

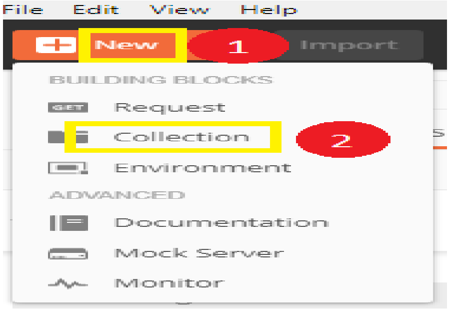
- Validate the response

1. Go to the request

2. Use the variable that was created earlier

11) Demonstrate how environments are used in Postman.
Creating an environment

- Create a Collection

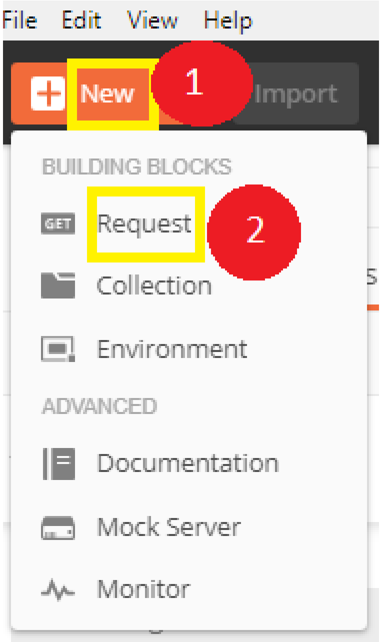


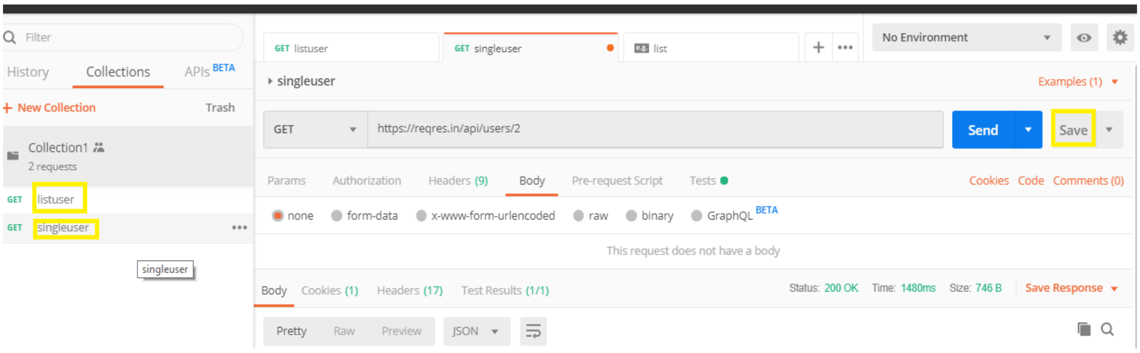
- Create two new Requests:

https://reqres.in/api/users/2

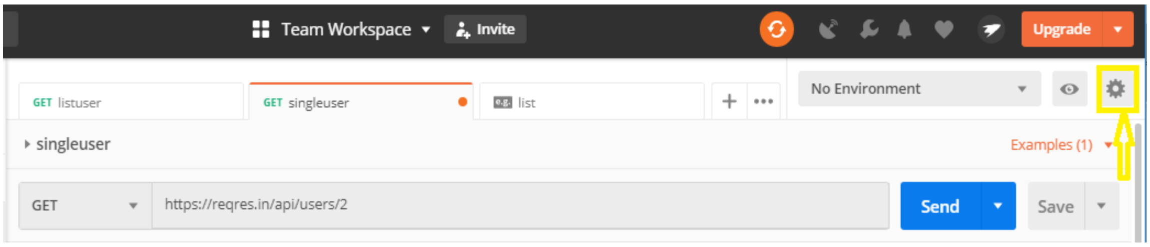
https://reqres.in/api/users?page=2

- Save it

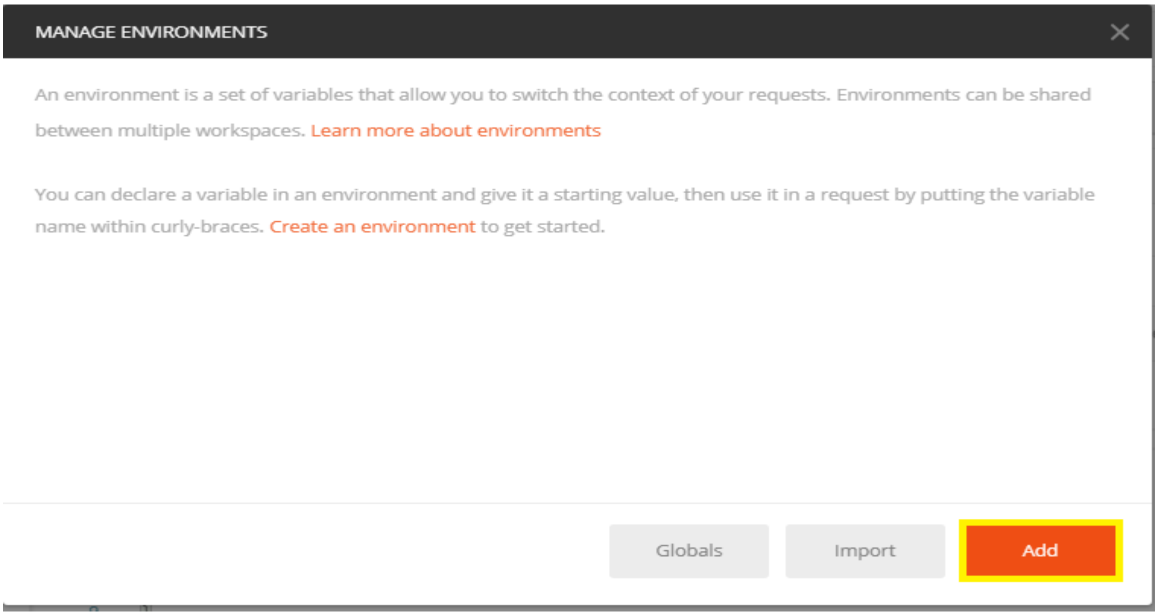




- Click on the gear icon which says Manage Environment



- Click on Add

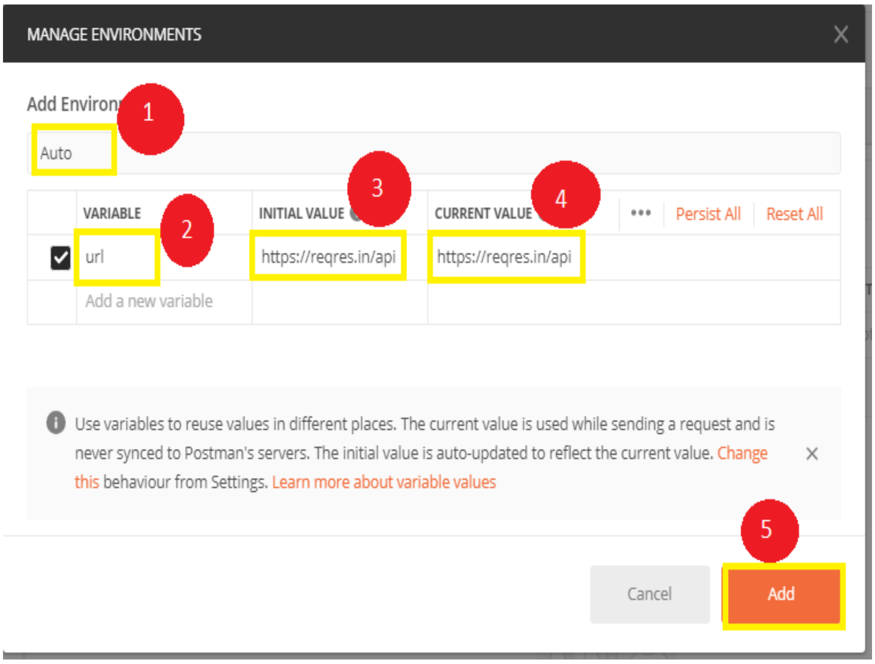


MANAGE ENVIRONMENTS

An environment is a set of variables that allow you to switch the context of your requests. Environments can be shared between multiple workspaces. Learn more about environments

You can declare a variable in an environment and give it a starting value, then use it in a request by putting the variable name within curly-braces. Create an environment to get started.

- Give a name to your environment



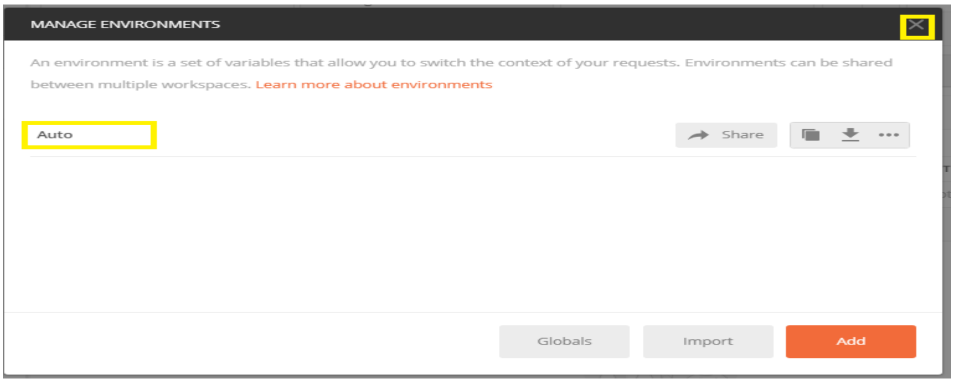
Add Environ

Auto

	VARIABLE	INITIAL VALUE	CURRENT VALUE
☑	url	https://reqres.in/api	https://reqres.in/api
	Add a new variable		

Use variables to reuse values in different places. The current value is used while sending a request and is never synced to Postman's servers. The initial value is auto-updated to reflect the current value. Change this behaviour from Settings. Learn more about variable values

- After clicking Add, we get a new window
- Close that Environment window

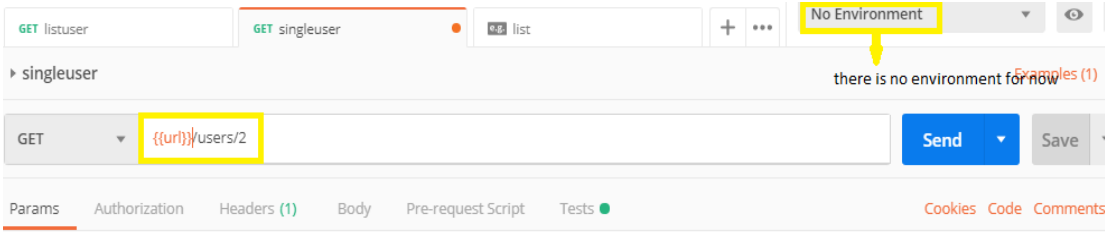


An environment is a set of variables that allow you to switch the context of your requests. Environments can be shared between multiple workspaces. Learn more about environments

Auto

Step 2.11.2: Running the Request using Environment

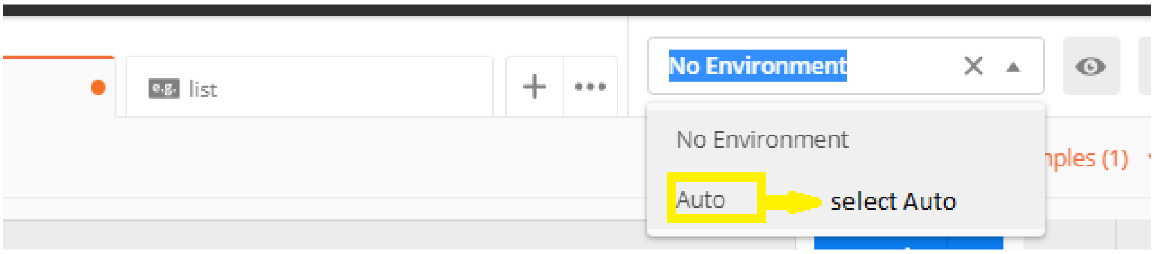
- Open your Requests and write your URL like the below image and use your Environment



there is no environment for now

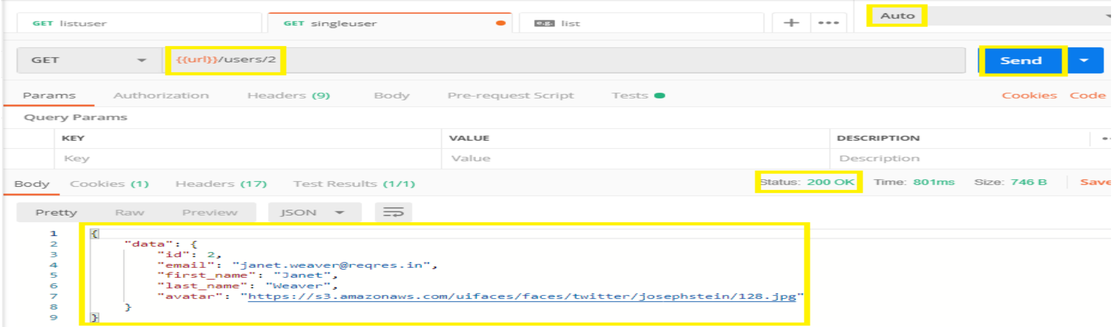
GET {{url}}/users/2

- Change your environment



select Auto

- Then run it and the output will look like this:



GET {{url}}/users/2

Status: 200 OK Time: 801ms Size: 746 B

```
{
    "data": {
        "id": 2,
        "email": "janet.weaver@reqres.in",
        "first_name": "Janet",
        "last_name": "Weaver",
        "avatar": "https://s3.amazonaws.com/uifaces/faces/twitter/josephstein/128.jpg"
    }
}
```

12) Demonstrate how test scripts are written in Postman.

Writing a script using snippet

- Create a new Request.

https://reqres.in/api/users/2

- Switch to the tests tab. On the right side are snippet codes.
- From the snippets section, click on "Status code: Code is 200."

- Now click Send. The test result should now be displayed.

Steps 2.12.2: Running the script and checking Response

- Go back to the Test tab and let's add another Test. This time we will compare the expected result to the actual result.

- From the snippets section, click on "Response time less than 200ms.”

- Click Send. There should now be two passed test results for your request.

13) Demonstrate how Postman is used with Jenkins.

: Installing Jenkins

- Jenkins is already installed in your practice lab in /usr/share/jenkins directory. Refer to QA to QE lab guide -- Phase 2 for more information.

- Open cmd goto jenkins.war location and run a command:

Java -jar jenkins.war (you need to install Java before running this command)

- Then open localhost:8080 on browser

- Go back to the terminal and use the following command to find the Admin Password:

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

- Type the password in the browser

- You will get a proper Jenkins window after setting up your admin password

Steps 2.13.2: Exporting Collection to desktop

- Open Postman
- Create a Collection
- Export the Collection

- Always choose Collection V2.1 (recommended)
- Save it on your desired location

Steps 2.13.3: Building that postman file in Jenkins

- Open Jenkins
- Create a job with the name Newman

- Pass two commands in your Execute windows batch command

```
cd C:\Users\Prakat-L-004\Desktop\Postman
newman run Collection1.json
```

- Apply
- Save

- Go to build

- Click on the build tab
- Go to console
- View as plain text
- Output: Finished Success

14) Illustrate workspaces in Postman using an example.

Creating a workspace

- Open Postman
- Go to the top of the page of Postman
- Click on the arrow mark
- Click on Create
- Manage Workspace

15) Demonstrate how monitors are used.

Changing the view

- Open Postman
- Go to the Setting option
- Click on Themes

Step 2.15.2: Managing Collection

- Create a new Collection
- Scroll down and find the option Browse

- Click on Browse
- Click on Collections
- Click on Monitor Collection

Step 2.15.3: Managing environments and workspace

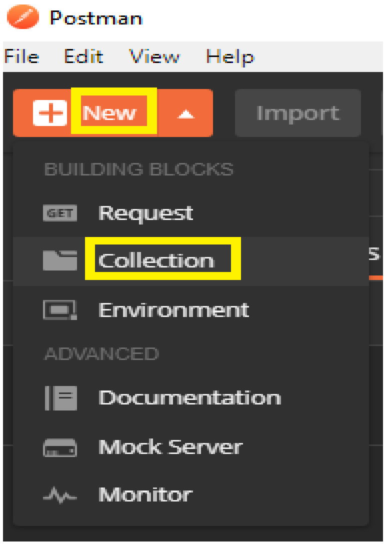
- Click on Add to workspace option

- Go to Environment

16) Demonstrate how to perform API documentation.

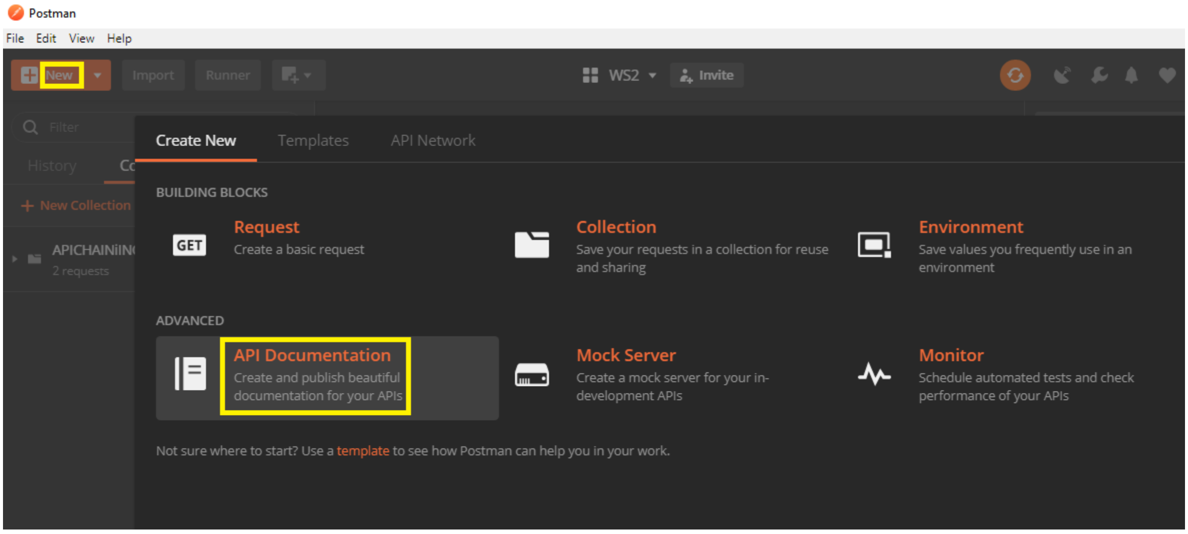
Creating a Collection

- Open Postman
- Go to New
- Click on Collection



Step 2.16.2: Creating API documentation

- Go to New
- Click on API Documentation



BUILDING BLOCKS

Request
Create a basic request

Collection
Save your requests in a collection for reuse and sharing

Environment
Save values you frequently use in an environment

ADVANCED

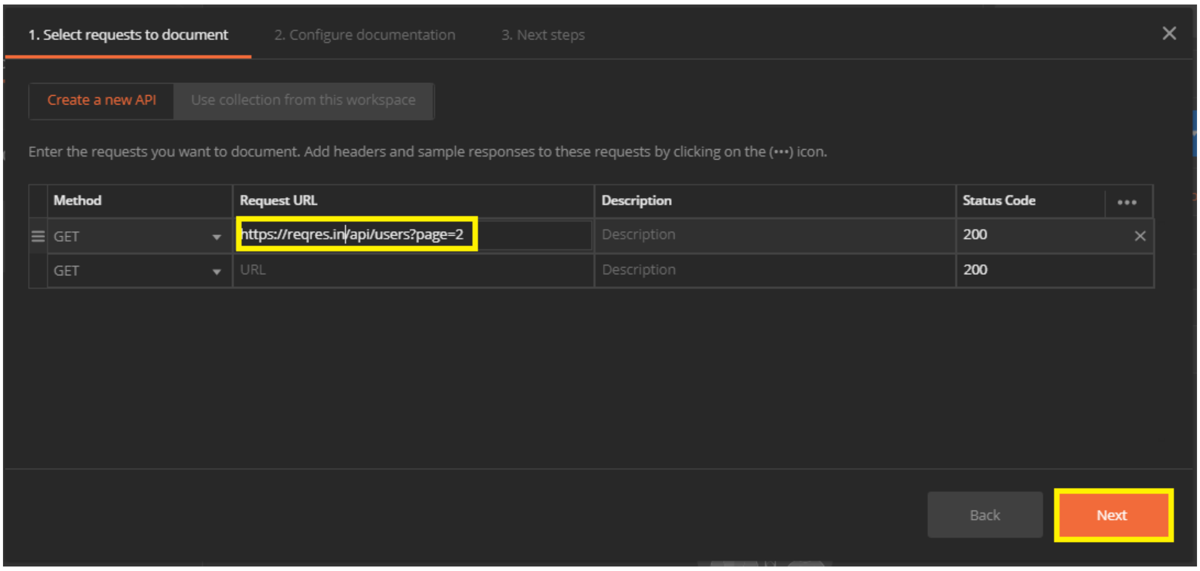
API Documentation
Create and publish beautiful documentation for your APIs

Mock Server
Create a mock server for your in-development APIs

Monitor
Schedule automated tests and check performance of your APIs

Not sure where to start? Use a template to see how Postman can help you in your work.

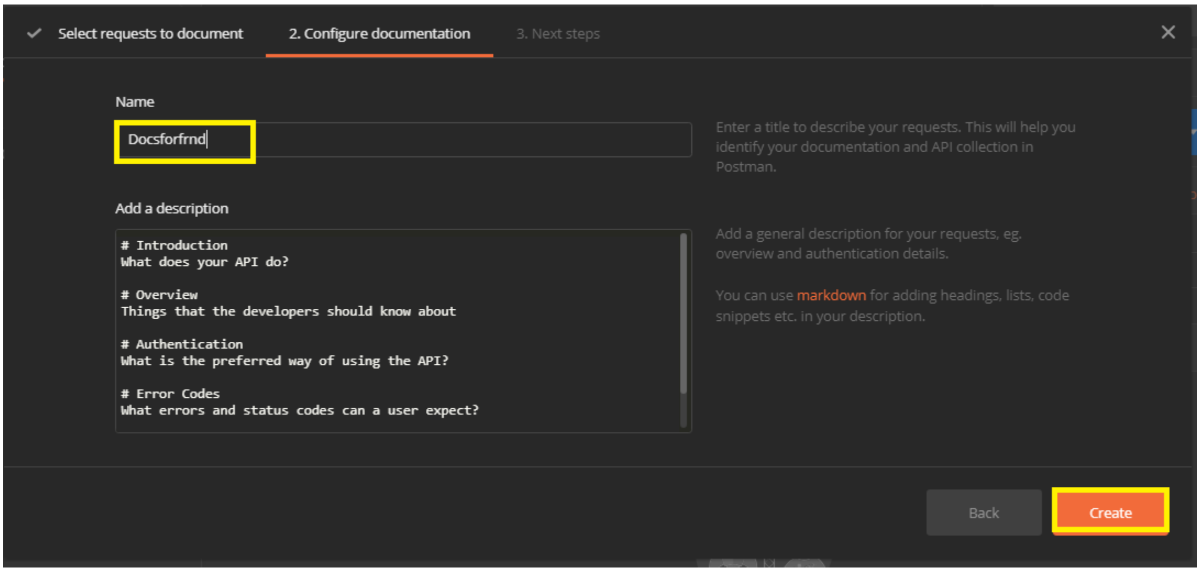
- Set your HTTP request as Get request
- Give request url as: https://reqres.in/api/users?page=2


1. Select requests to document
2. Configure documentation
3. Next steps
Create a new API
Use collection from this workspace
Back
Next

Enter the requests you want to document. Add headers and sample responses to these requests by clicking on the (•••) icon.

Method	Request URL	Description	Status Code
GET	https://reqres.in/api/users?page=2	Description	200
GET	URL	Description	200

- Set a name
- Click on Create



Name

Docsforfrnd

Enter a title to describe your requests. This will help you identify your documentation and API collection in Postman.

Add a description

```
# Introduction
What does your API do?

# Overview
Things that the developers should know about

# Authentication
What is the preferred way of using the API?

# Error Codes
What errors and status codes can a user expect?
```

Add a general description for your requests, eg. overview and authentication details.

You can use markdown for adding headings, lists, code snippets etc. in your description.

17) Demonstrate how to get data from CSV and JSON.

Creating a new Request and setting the environment

- Go to Postman
- Click on the New menu. Choose Request as https://reqres.in/api/register

- Set that Request as post
- Set the Body as raw

```
{
  "email": "eve.holt@reqres.in",
  "password": "pistol"
}
```

- Set header content type as application/json
- Send the request

- Consider the request as: https://reqres.in/api/register/preprod

- Create an environment as

Endpoint=api/register/preprod

pass=pistol

email=eve.holt@reqres.in

Step 2.17.2: Getting the data from CSV file

- Go to desktop, create a text file, and name it as Data.csv
- Write your variables

- Run on collection runner
- Select the Data.csv file
- Data file type text/csv
- Click on Run

- The output is shown below:

Step 2.17.3: Getting the data from JSON file

- Go to desktop, create a text file, and name it as Data1.json
- Write your variables

- Run on Collection Runner
- Select the Data1.json file
- Select data file type as application/json
- Click on Run

18) Demonstrate how to run a collection remotely with URL.
2.18.1 Creating a collection
2.18.2 Creating a new request
2.18.3 Getting the Collection URL
2.18.4 Running the Collection URL remotely
2.18.5 Pushing the code to GitHub repositories

Step 2.18.1: Creating a collection

- Open Postman
- Click on **New** option
- Select Collection

Step 2.18.2: Creating a New Request

- Go to New
- Select Request

Step 2.18.3: Getting the Collection URL

- Go to Collection
-
- Click on the arrow mark next to Collection
- Click Share

- Click on Get link
- Copy that link

Step 2.18.4: Running the collection URL remotely

- Type the collection URL
- Install Newman:
  npm -g install newman
- Run that URL remotely using:

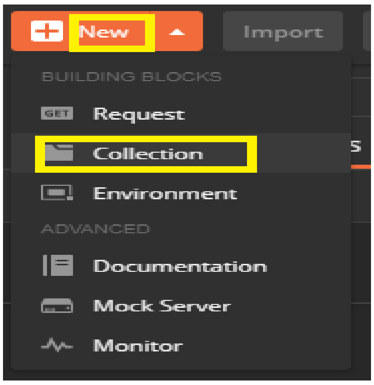
```
newman run https://www.getpostman.com/collections/9a3b8cbf5b3d161121bf
```

19) Demonstrate how to run SOAP requests in Postman.

2.19.1 Creating a Collection
2.19.2 Creating a New Request as POST
2.19.3 Providing request data in the body
2.19.4 Validating the request
2.19.5 Pushing the code to GitHub repositories

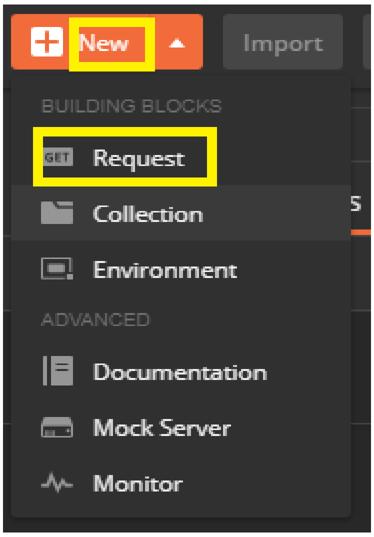
Step 2.19.1: Creating a collection

- Open Postman
- Click on **New**
- Select Collection



Step 2.19.2: Creating a New Request

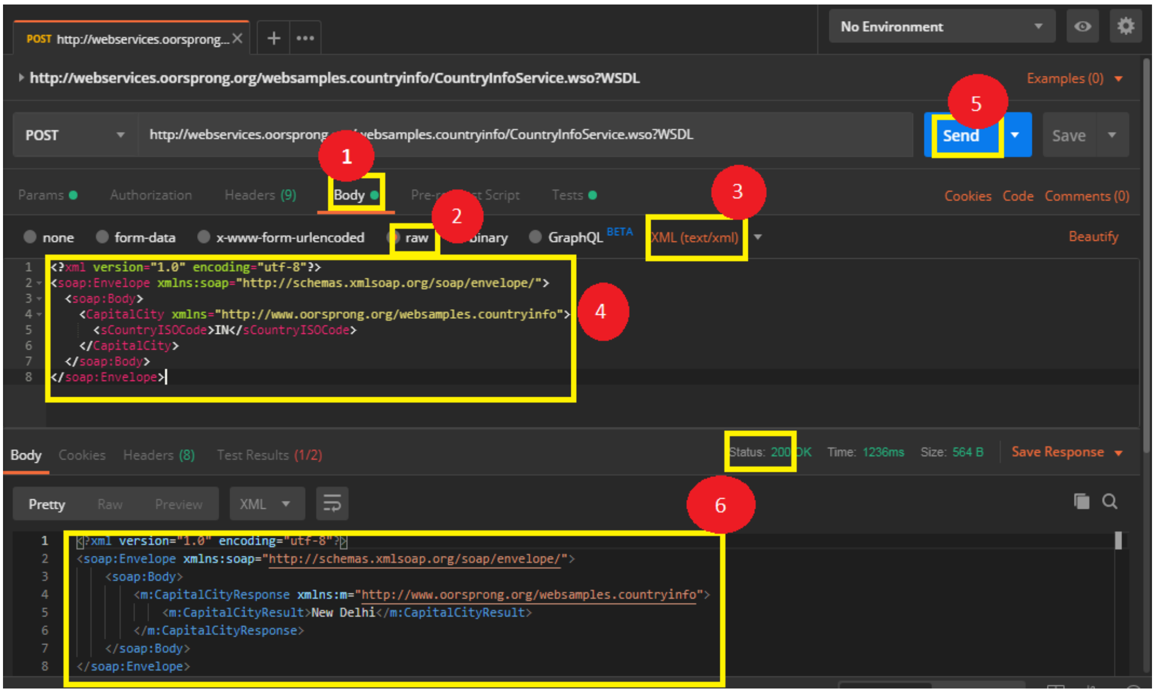
- Go to New
- Select Request



Step 2.19.3: Providing the Request data in body

- Go to Request body
- Click on raw
- Select **Xml(text/xml)**

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Body>
  <CapitalCity xmlns="http://www.oorsprong.org/websamples.countryinfo">
   <sCountryISOCode>IN</sCountryISOCode>
  </CapitalCity>
 </soap:Body>
</soap:Envelope>
```



http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wso?WSDL

POST http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wso?WSDL

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <CapitalCity xmlns="http://www.oorsprong.org/websamples.countryinfo">
      <sCountryISOCode>IN</sCountryISOCode>
    </CapitalCity>
  </soap:Body>
</soap:Envelope>
```

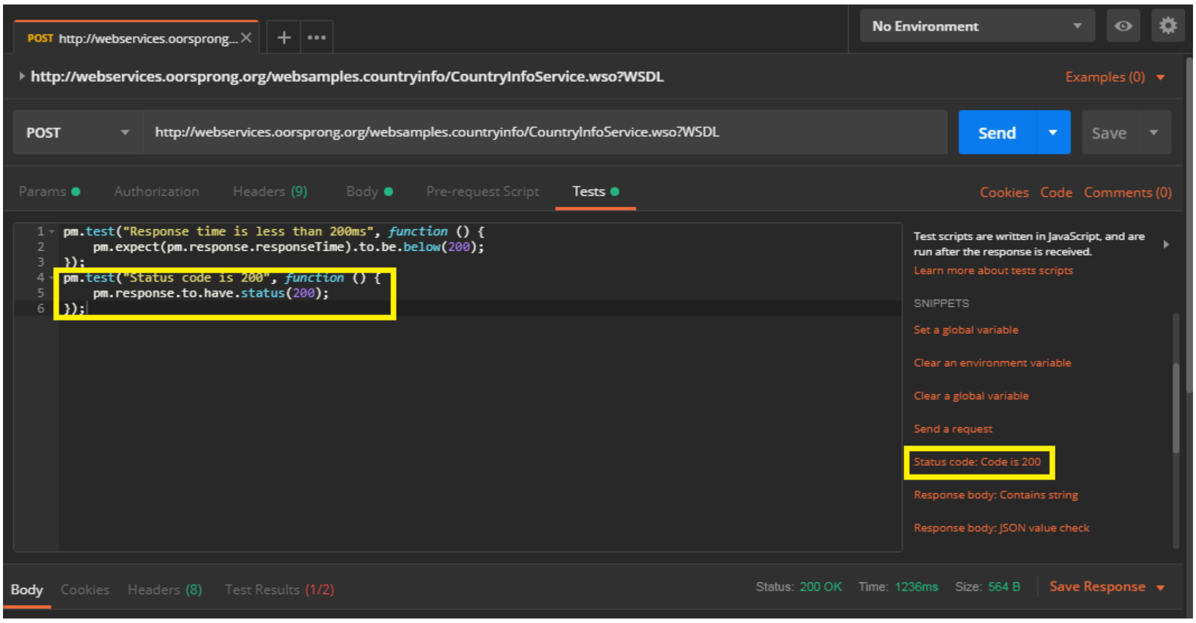
Body Cookies Headers (8) Test Results (1/2)

Status: 200 OK Time: 1236ms Size: 564 B

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <m:CapitalCityResponse xmlns:m="http://www.oorsprong.org/websamples.countryinfo">
            <m:CapitalCityResult>New Delhi</m:CapitalCityResult>
        </m:CapitalCityResponse>
    </soap:Body>
</soap:Envelope>
```

Step 2.19.4: Validating the request

- Go to **Test**
- Pass the snippet which shows status code as 200



http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wso?WSDL

POST http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wso?WSDL

```
pm.test("Response time is less than 200ms", function () {
    pm.expect(pm.response.responseTime).to.be.below(200);
});
pm.test("Status code is 200", function () {
    pm.response.to.have.status(200);
});
```

Test scripts are written in JavaScript, and are run after the response is received.
Learn more about tests scripts

Body Cookies Headers (8) Test Results (1/2)

Status: 200 OK Time: 1236ms Size: 564 B

Step 2.19.5: Pushing the code to GitHub repositories

Open your command prompt and navigate to the folder where you have created your files

20) Demonstrate API Chaining and REST in Postman.

Creating a Collection

- Open Postman
- Click on New
- Select Collection

Step 2.20.2: Creating two new requests

- Go to New
- Select Request
- Create GET Request(“https://api.coindesk.com/v1/bpi/currentprice.json”)
- Create POST Request(“https://api.twilio.com/2010-04-01/Accounts/{{twilioAccountSID}}/Messages.json”)

Step 2.20.3: Creating an Environment

- Go to the settings option
- Click on Add

Step 2.20.4: Checking the API chaining response

- Go to Test
- Write the body of the request

- After we hit Send, click on the Quick Look icon to see that we’ve extracted the data from the response body, and saved it as an environment variable. Now, we can use this variable in our next request.

- In the same manner, can you figure out how to extract the “time” from the response, and then save it as an environment variable?

Send a POST request using a stored environment variable.

- For this second request, we will use the Twilio API to send an SMS. Send a POST to https://api.twilio.com/2010-04-01/Accounts/{{twilioAccountSID}}/Messages.json where {{twilioAccountSID}} is your own Twilio Account SID. Remember to also save this second request to our collection.

- String substitution: We can use the double curly braces syntax anywhere we would normally input text, like {{variable-name}}, to use a variable. Postman will access the variable’s value and swap in the value as a string. Use the double curly braces in the URL, parameters, or any of the other data editors.

- More String Substitution: This request requires the authentication credentials we saved to our environment in the first step. We can use the same double curly braces syntax in Postman’s auth helper to handle Basic Auth for the Twilio API. Authentication (and scripts) are available for entire collections and folders too.