

**Subject→ Software Engineering**

**TA→ Execution Based Assessment.**

**Under the Guidance of→ Heena Agrawal Ma'am**

---

**Group Members→**

14. Aniket Pardhi

43. Harsh Agrawal

47. Jatin Jangir

---

## **Performing Testing using Postman and Jenkins.**

**Postman→** Postman is one of the most popular software testing tools which is used for API testing. With the help of this tool, developers can easily create, test, share, and document APIs.

### **Introduction to Postman**

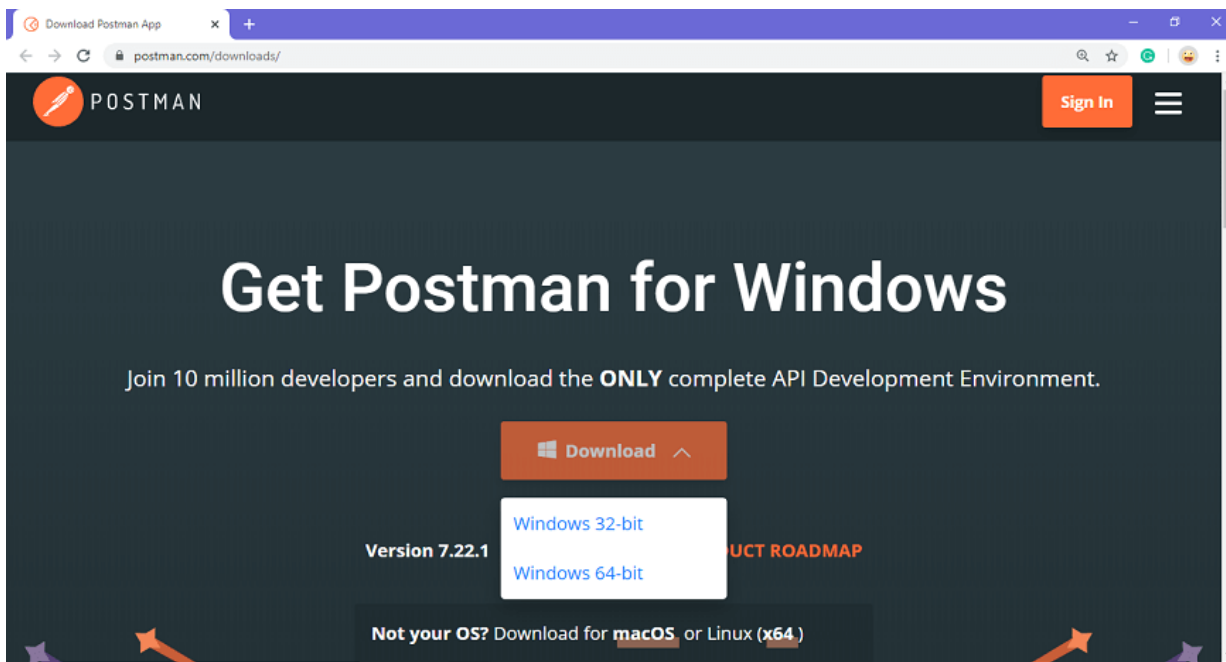
- Postman is a standalone software testing API (Application Programming Interface) platform to build, test, design, modify, and document APIs. It is a simple Graphic User Interface for sending and viewing HTTP requests and responses.
- While using Postman, for testing purposes, one doesn't need to write any HTTP client network code. Instead, we build test suites called collections and let Postman interact with the API.
- In this tool, nearly any functionality that any developer may need is embedded. This tool has the ability to make various types of HTTP requests like GET, POST, PUT, PATCH, and convert the API to code for languages like JavaScript and Python.

## **INSTALLATION→**

### **Steps to download and install the native Postman application**

**Step-1:** Go to the link <https://www.postman.com/downloads/>

and click download for Mac or Windows or Linux based on your operating system.

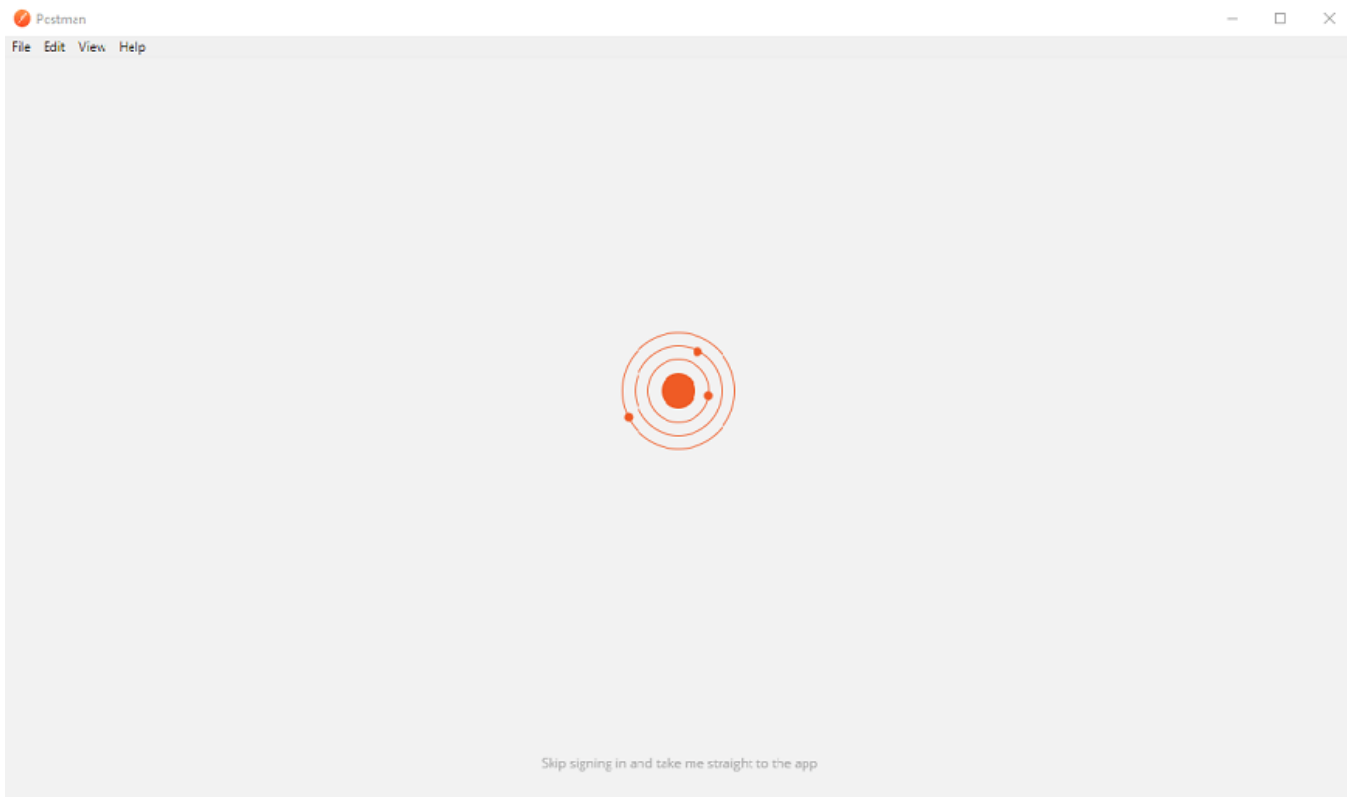


**Step-2:** For downloading the app for Windows, click on the download button and select the particular version. I opted for the 64-bit version. If you are using a 32-bit OS, you can choose the 32 bit, as shown in the above image.

**Step-3:** You can check the download progress on the bottom left if you are using the Chrome browser. Once the .exe file is downloaded, you need to install the application, as shown in the below image.



**Step-4:** Once the installation completes, you will be redirected to a window as shown in the image where you can click on Stop signing in and take me straight to the app (as this app can also be used without logging in) or otherwise you will get a new window to sign up.

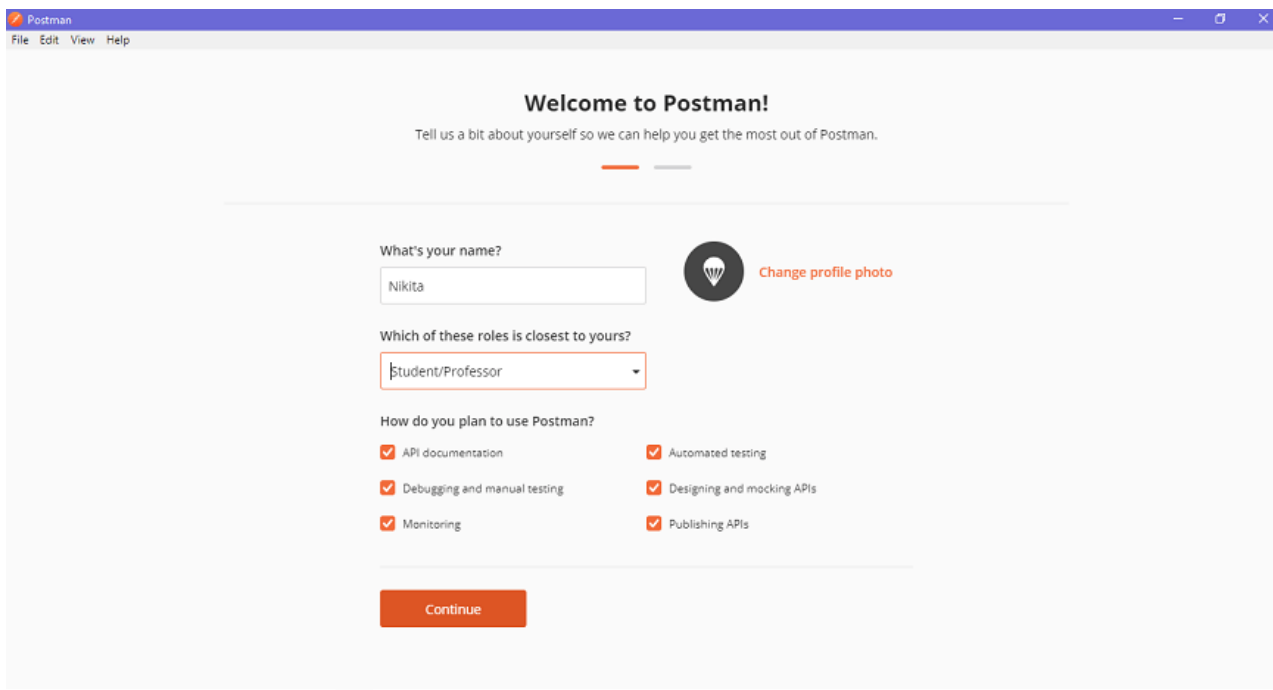


It is better to create an account as this will help you to save the work you do within the Postman, and with this, you won't lose any work.

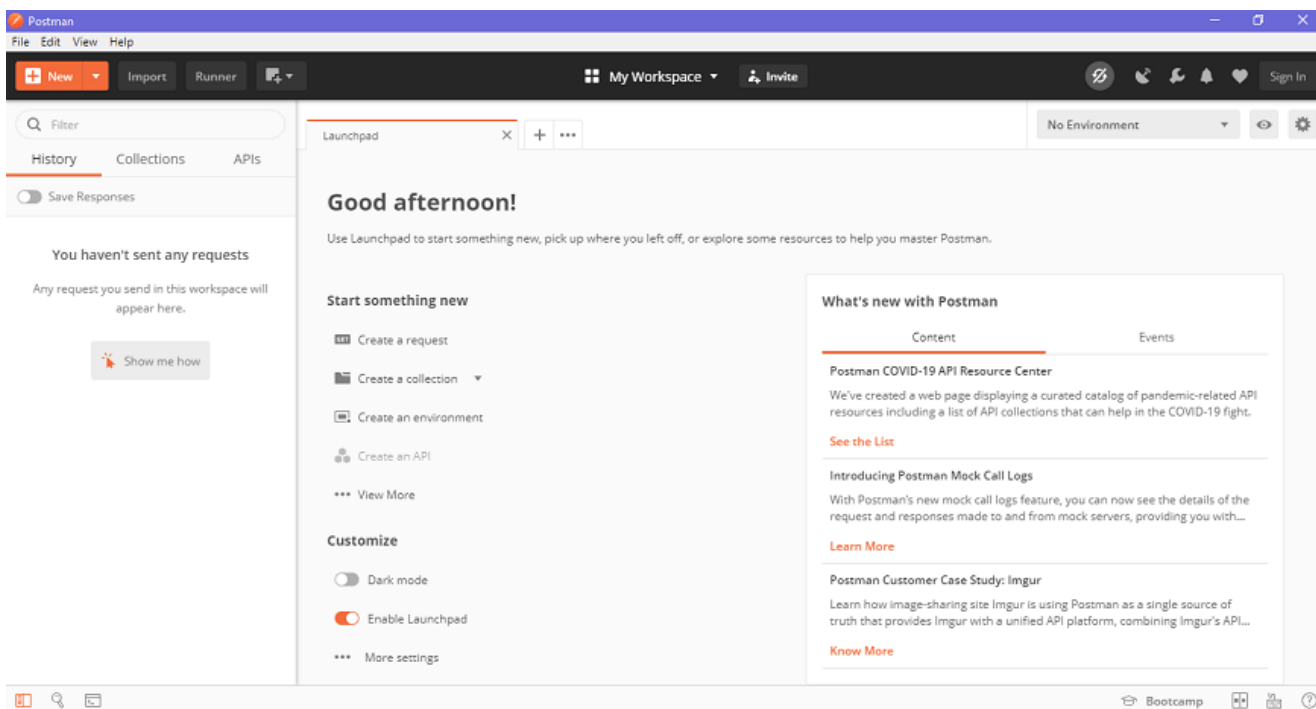
**Step-5:** Create your account with all the required details, or you can also signup with Google, as shown in the image.

The image shows the Postman application window with the 'Create Account' screen. The window has a menu bar with 'File', 'Edit', 'View', and 'Help'. The main area is divided into two sections. On the left, there is a 'Why Sign Up?' section with the Postman logo and a list of benefits: 'Organize all your API development within Postman Workspaces', 'Sync your Postman data across devices', 'Backup your data to the Postman cloud', and 'It's free!'. Below the list is an illustration of two astronauts in space. On the right, there is a 'Create Account' form with fields for 'Email', 'Username', and 'Password'. The 'Email' field contains 'nikita\*\*\*\*\*@gmail.com'. The 'Username' field contains 'Nikita'. The 'Password' field is masked with dots. There are checkboxes for 'I agree to the Terms of Use.' and 'Keep me signed in'. Below the form is a 'Create free account' button. At the bottom, there is a link to 'Sign in / Sign up through email instead'.

**Step-6:** After signing in, select the workspace tools as per your requirement, and then click on, continue to get the startup screen.



**Step-7:** You will see the following page, and then you are ready to use Postman.



## **Sending Request Through Postman→**

Sending a request is as easy as posting a URL into your web browser. We can easily send requests to APIs in Postman. An API request helps you to access, or send, data from a data source.

To send the API request, we need an **HTTP** method. Some commonly used methods are POST, GET, DELETE, PUT, and PATCH.

GET: This **HTTP** method is used to access the data from an API.

POST: This method transmits new data.

DELETE: This is used to remove or delete the existing data.

PATCH: This method is used to update the existing data.

PUT: This method is used to update the existing data.

---

## **Performing API Testing using postman→**

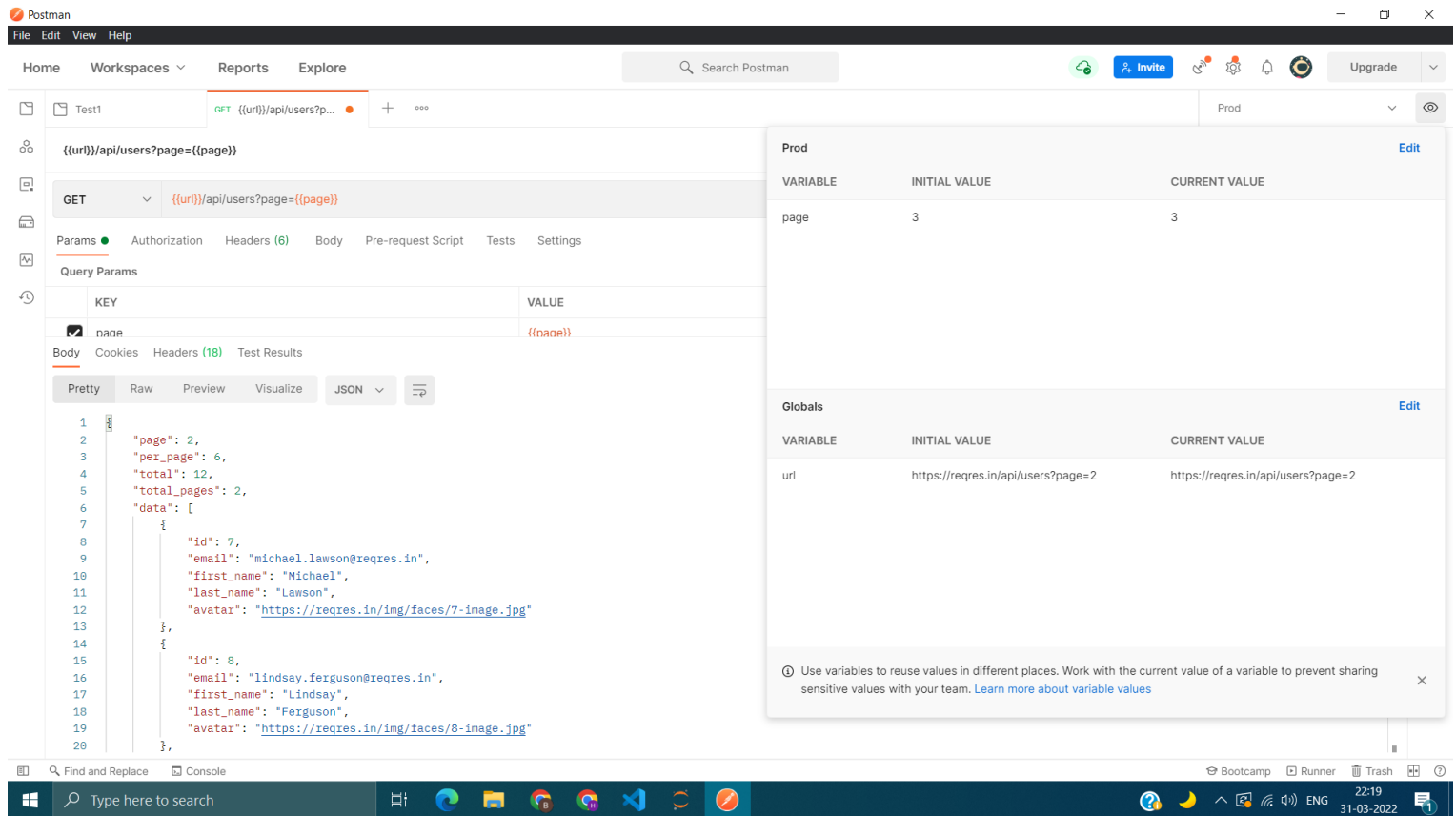
API testing primarily aims to test the business logic layer of the system's architecture. API testing is primarily handled by the QA team, which requires them to run any API on top of a particular build meant to serve a specific purpose.

*API testing also tests the unit as part of a system, while unit testing typically tests the unit in relative isolation from the rest of the system.* Hence API testing is also end to end testing. This simply means when we test the complete software in API testing then the modules which make that software are also tested, obviously. But when we do unit testing, we focus only on the functionality of that module and see its working which is completely isolated from the rest of the modules/software.

---

## **Setting Environment Variables→**

- A variable is a symbolic representation of data that enables you to access a value without having to enter it manually wherever you need it. This can be useful if you are using the same values in multiple places. Variables make your requests more flexible and readable, by abstracting the detail away.
- For example, if you have the same URL in more than one request, but the URL might change later, you can store the URL in a variable `base_url` and reference it in your requests using `{{base_url}}`. If the URL changes, you can change the variable value and it will be reflected throughout your collection, wherever you've used the variable name.

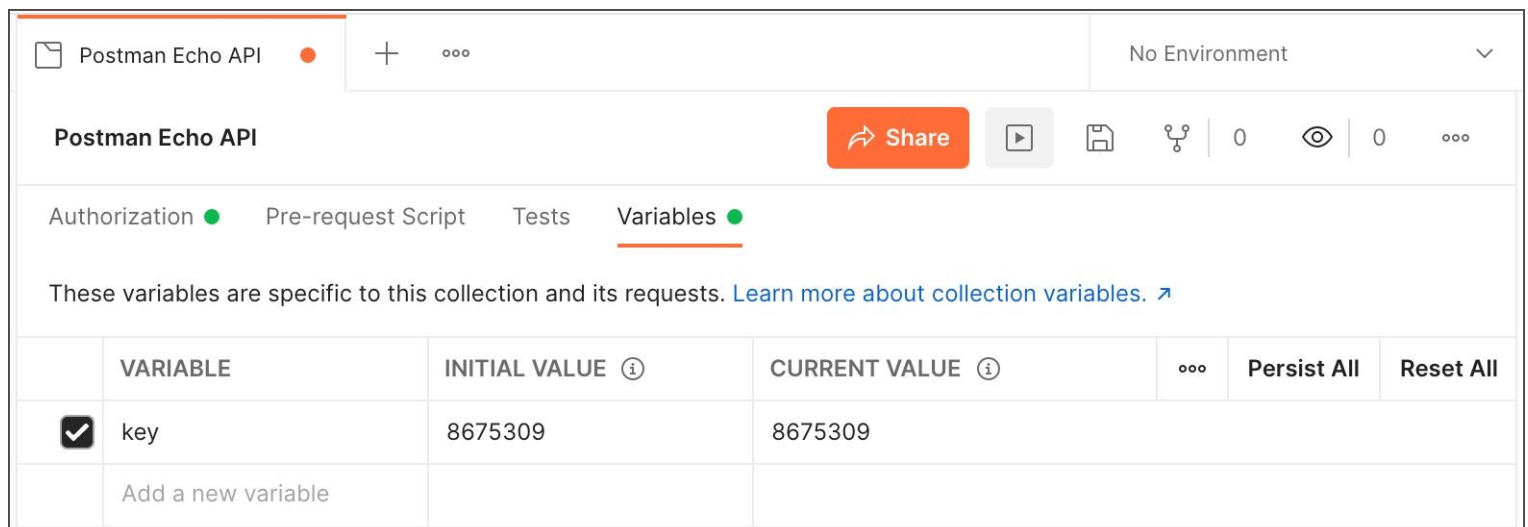


# Defining collection variables

You can add collection variables when you create the collection or at any time after that.

To create or edit a variable for an existing collection:

1. Select Collections in the left sidebar.
2. Select a collection, and then select the Variables tab.



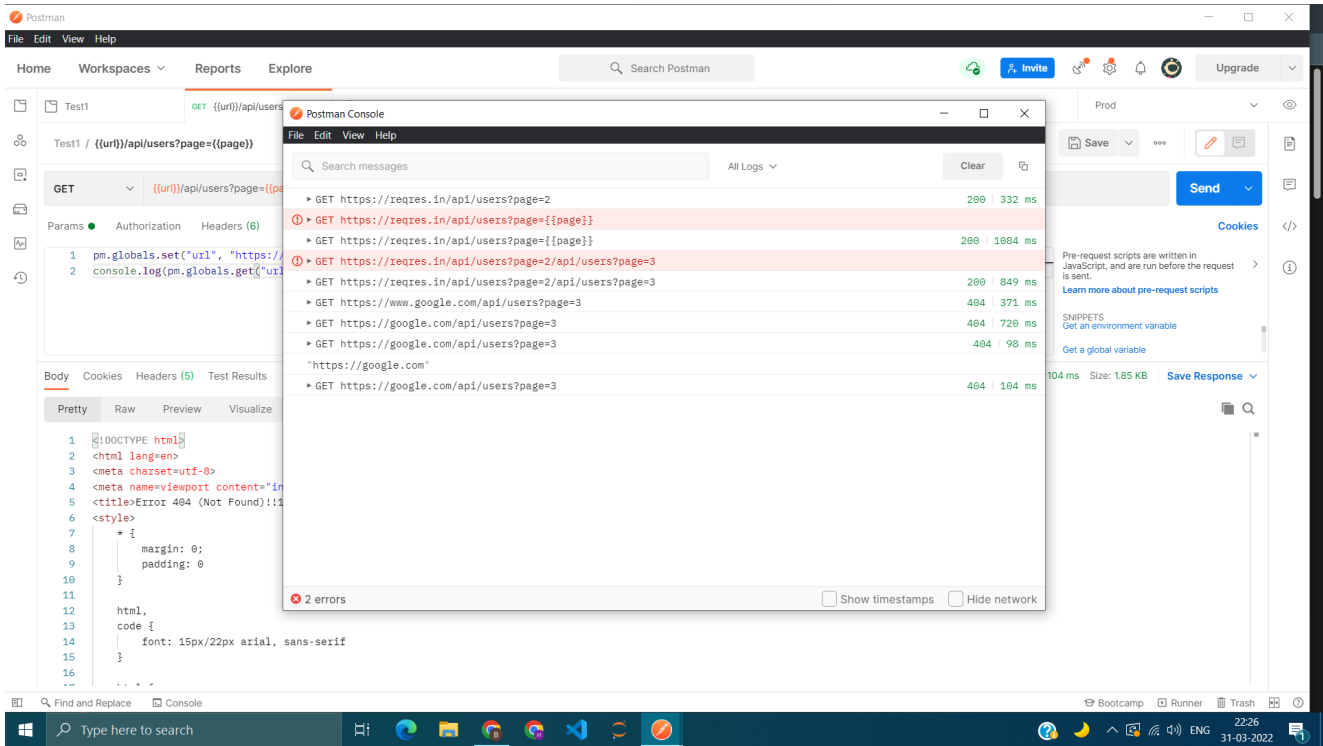
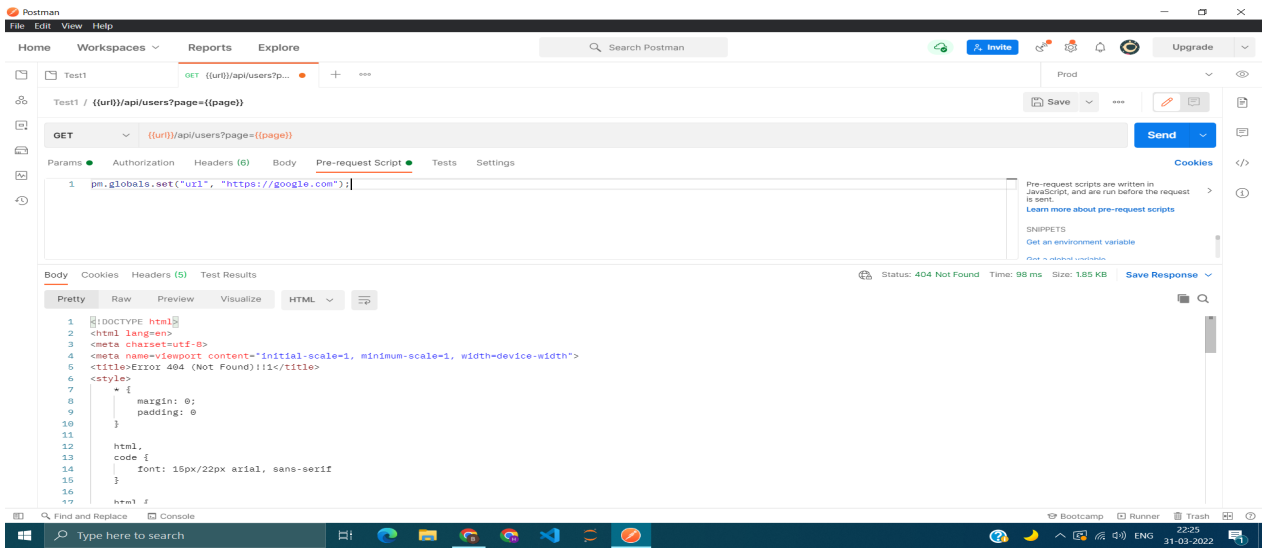
If you don't have Editor access to a collection, you will see a Request Access button. You won't be able to add new collection variables, update initial values, or persist values. You can edit the current value for local use, override the collection variable by using an environment variable with the same name, or [request Editor access](#) to the collection.

You can also [define collection variables in scripts](#).

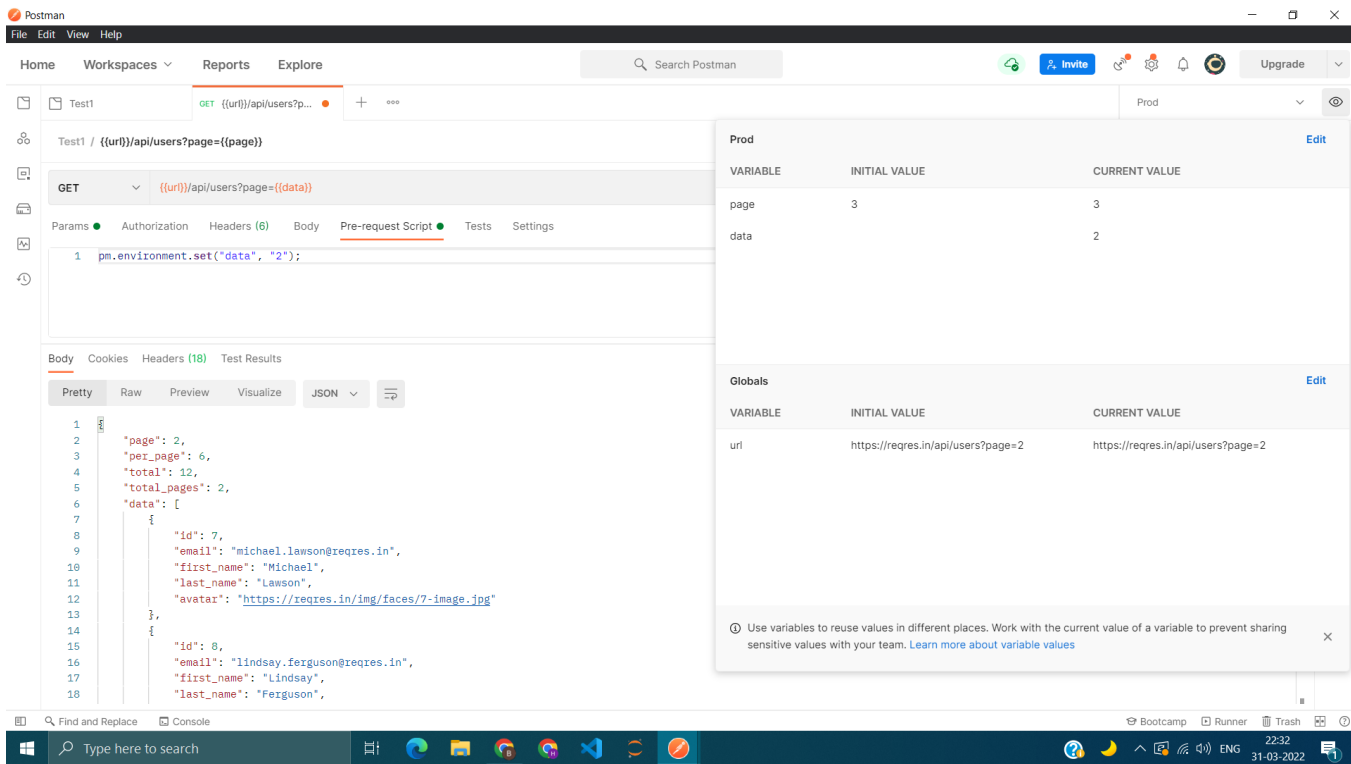
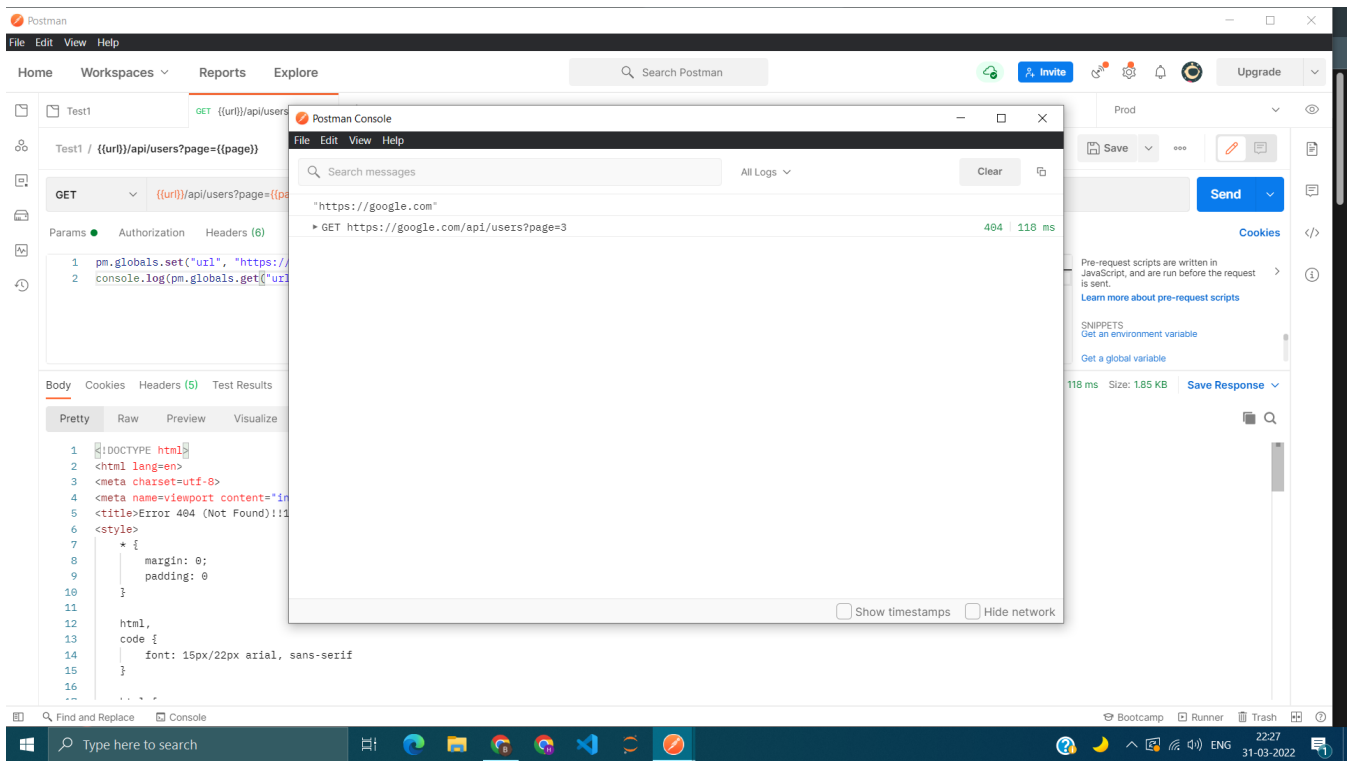
## Defining variables in scripts

You can set variables programmatically in your request scripts.

Method	Use-case	Example
<b>pm.globals</b>	Use to define a global variable.	<code>pm.globals.set("variable_key", "variable_value");</code>
<b>pm.collectionVariables</b>	Use to define a collection variable.	<code>pm.collectionVariables.set("variable_key", "variable_value");</code>
<b>pm.environment</b>	Use to define an environment variable in the currently selected environment.	<code>pm.environment.set("variable_key", "variable_value");</code>
<b>pm.variables</b>	Use to define a local variable.	<code>pm.variables.set("variable_key", "variable_value");</code>
<b>unset</b>	You can use unset to remove a variable.	<code>pm.environment.unset("variable_key");</code>








To check if the variable is available and in scope for the request:

1. Select one of the collection or global links. To turn on an environment, use the \*\*select an environment link.
2. Turn on or make the necessary changes to the value of the variable.

3. Select  Save to confirm your changes.

If the variable is unresolved because it doesn't exist:

1. Select Add new variable.

2. Enter a Name, set a Value for the variable, and select the appropriate Scope (global, collection, or environment) from the dropdown.
  3. Select Set variable.
- 

### **Data Driven Test Case using Postman→**

#### **Using CSV file:**

1. Create a CSV file.
  2. Add email, password in columns of CSV file
  3. Add possible test cases of email and password.
  4. Create a collection named Data-driven test
  5. To get variable from CSV file, run this collection on Runner, click Run button
  6. Collection Runner window will appear.
  7. Iterations are the no. of data rows in CSV file
  8. Add CSV file in the Data option. Preview button will display the preview of CSV file
  9. Click on Run button in blue color
  10. Result window will be displayed
- 

#### **Sample Runner Window → Data Driven test.**

- In this “Data-Driven Testing in Postman” article, I will be demonstrating how you can implement this concept and get a tight grip over this.
- Data-driven testing is when we have one test that we run multiple times with different data variables. It’s useful for things like, if we have a certain range of characters that we’re supporting in our names, to make sure that all those characters are supported in different tests.
- Data-driven testing can be a very effective approach in testing an API against different data-sets. Postman supports CSV and JSON files to get data for the test scripts. The data-driven approach is useful when we need to execute a test with multiple sets of Data. Also, modifying or adding any data fields will just need updating the data files which is easier than going to test script and updating test data.

Postman

File Edit View Help

HomeWorkspacesReportsExplore

Search Postman

Invite

Upgrade

Testing

NewImport

GET {{url}}/api/users?p...

Test1

Test1

Runner

Prod

Collections

Execution Based

Test1

GET {{url}}/api/users?page={{page}}

RUN ORDER

Deselect AllSelect AllReset

GET {{url}}/api/users?page={{page}}

Iterations

5

Delay

0ms

Data

Select Filedata.csv

Data File Type

text/csvPreview

Save responses

Keep variable values

Run collection without using stored cookies

Save cookies after collection run

Run Test1

Find and ReplaceConsole

Type here to search

BootcampRunnerTrash

23:1231-03-2022

Postman

File Edit View Help

HomeWorkspacesReportsExplore

Search Postman

Invite

Upgrade

Testing

NewImport

GET {{url}}/api/users?p...

Test1

Test1

Runner

Prod

Collections

Execution Based

Test1

GET {{url}}/api/users?page={{page}}

RUN ORDER

Deselect AllSelect AllReset

GET {{url}}/api/users?page={{page}}

Iterations

5

Delay

0ms

Data

Select Filedata.csv

Data File Type

text/csvPreview

Save responses

Keep variable values

Run collection without using stored cookies

Save cookies after collection run

Run Test1

PREVIEW DATA

Iteration	id	name	job
1	1	"promod"	"dutta"
2	2	"amit"	"singh"
3	3	"varun"	"pk"
4	4	"dheeraj"	"joshi"
5	5	"Manish"	"Gill"

Find and ReplaceConsole

Type here to search

BootcampRunnerTrash

23:1331-03-2022

Postman

File Edit View Help

HomeWorkspacesReportsExplore

Search Postman

Testing

NewImport

GET {{url}}/api/users?p...

Runner

Prod

Collections

Execution Based

Test1

GET {{url}}/api/users?page={{page}}

RUN ORDER

Deselect AllSelect AllReset

GET {{url}}/api/users?page={{page}}

Iterations

5

Delay

0

ms

Data

Select Filedata.csv

Data File Type

text/csv

Preview

Save responses

Keep variable values

Run collection without using stored cookies

Save cookies after collection run

Run Test1

Find and Replace

Console

BootcampRunnerTrash

23:18

31-03-2022

Postman

File Edit View Help

HomeWorkspacesReportsExplore

Search Postman

Testing

NewImport

GET {{url}}/api/users?p...

Runner

Prod

Test1 / {{url}}/api/users?page={{page}}

Save

Send

GET

{{url}}/api/users...

Params

Authorization

Headers (10)

Body

Pre-request Script

Tests

Settings

Headers

8 hidden

KEY	VALUE	DESCRIPTION
Accept		
Abc	{{randomInt}}	
Key	Value	Description

Body

Cookies

Headers (18)

Test Results

Status: 200 OKTime: 949 msSize: 1.88 KB

Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1  {
2    "page": 2,
3    "per_page": 6,
4    "total": 12,
5    "total_pages": 2,
6    "data": f
```

Find and Replace

Console

BootcampRunnerTrash

23:18

31-03-2022

Collection Runner

File Edit View Help

Collection RunnerRun Results

My Workspace

Run In Command LineDocs

8 PASSED

1 FAILED

Data Driven test

No Environment

just now

Run Summary

Export Results

Retry

New

Iteration 1

POST Register https://reqres.in/api/regi...

Data Driven test / Register

201 Created 2533 ms 571 B

Status code is 201

Check email abc

Check password pi

Iteration 2

POST Register https://reqres.in/api/regi...

Data Driven test / Register

201 Created 866 ms 578 B

Status code is 201

Check email abc@gmail

Check password 123 | AssertionError: expected '123' to deeply equal 123

Iteration 3

POST Register https://reqres.in/api/regi...

Data Driven test / Register

201 Created 216 ms 587 B

Status code is 201

Check email abc@gmail.com

Check password 123#hdsgh

**Jenkins** → Jenkins is a self-contained, open source automation server which can be used to automate all sorts of tasks related to building, testing, and delivering or deploying software.

Jenkins can be installed through native system packages, Docker, or even run standalone by any machine with a Java Runtime Environment (JRE) installed.

Jenkins

Search

1

Aniket Pardhi

log out

Dashboard

New Item

People

Build History

Manage Jenkins

My Views

Lockable Resources

New View

Build Queue

Build Executor Status

built-in node (0 of 2 executors busy)

All

+

S

W

Name

Last Success

Last Failure

Last Duration

⊗

☁

API testing

N/A

7 days 22 hr

#1

0.77 sec

▶

⊗

☁

SE\_TA

7 days 23 hr

#1

7 days 2 hr

#25

0.91 sec

▶

⊗

☁

testingJenkins

N/A

24 min

#4

2.8 sec

▶

⊗

☁

trial

N/A

7 days 1 hr

#6

1.4 sec

▶

Icon: S W L

Icon legend

Atom feed for all

Atom feed for failures

Atom feed for just latest builds

## Installing Jenkins

Jenkins is typically run as a standalone application in its own process with the built-in Java servlet container/application server (Jetty).

Jenkins can also be run as a servlet in different Java servlet containers such as Apache Tomcat or GlassFish. The simplest way to install Jenkins on Windows is to use the Jenkins Windows installer. That program will install Jenkins as a service using a 64 bit JVM chosen by the user. Keep in mind that to run Jenkins as a service, the account that runs Jenkins must have permission to login as a service.

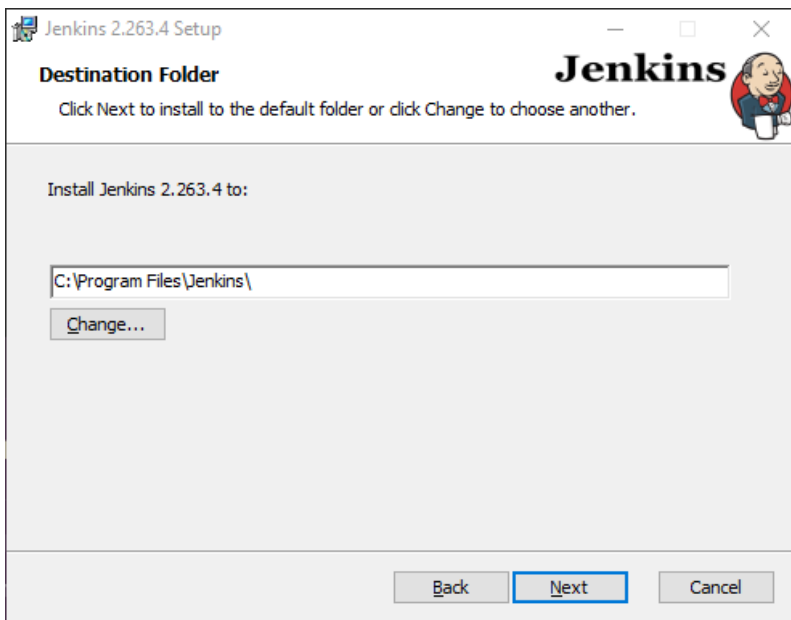
### Step 1: Setup wizard

On opening the Windows Installer, an Installation Setup Wizard appears, Click Next on the Setup Wizard to start your installation.



### Step 2: Select destination folder

Select the destination folder to store your Jenkins Installation and click Next to continue.



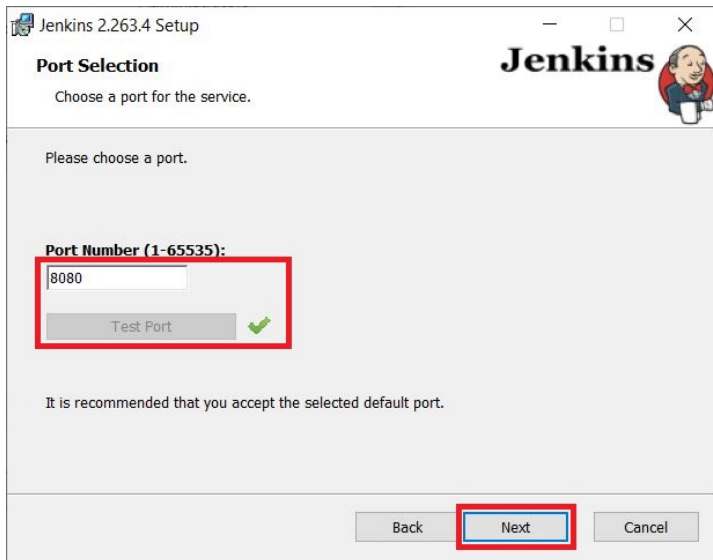
### Step 3: Service logon credentials

When Installing Jenkins, it is recommended to install and run Jenkins as an independent windows service using a local or domain user as it is much safer than running Jenkins using LocalSystem(Windows equivalent of root) which will grant Jenkins full access to your machine and services.

To run Jenkins service using a local or domain user, specify the domain username and password with which you want to run Jenkins, click on Test Credentials to test your domain credentials and click on Next.

### Step 4: Port selection

Specify the port on which Jenkins will be running, Test Port button to validate whether the specified port is free on your machine or not. Consequently, if the port is free, it will show a green tick mark as shown below, then click on Next.

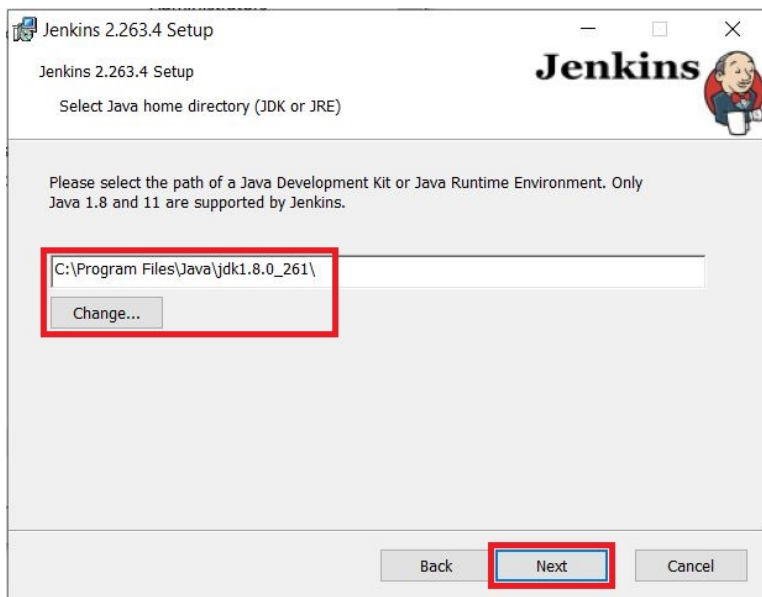


The image shows the 'Port Selection' window of the Jenkins 2.263.4 Setup. The window title is 'Jenkins 2.263.4 Setup'. The main heading is 'Port Selection' with the instruction 'Choose a port for the service.' Below this, it says 'Please choose a port.' There is a text input field labeled 'Port Number (1-65535):' containing the value '8080'. To the right of the input field is a 'Test Port' button, and to its right is a green checkmark icon. Below the input field, it says 'It is recommended that you accept the selected default port.' At the bottom of the window are three buttons: 'Back', 'Next', and 'Cancel'. The 'Next' button is highlighted with a red rectangle.

### Step 5: Select Java home directory

The installation process checks for Java on your machine and prefills the dialog with the Java home directory. If the needed Java version is not installed on your machine, you will be prompted to install it.

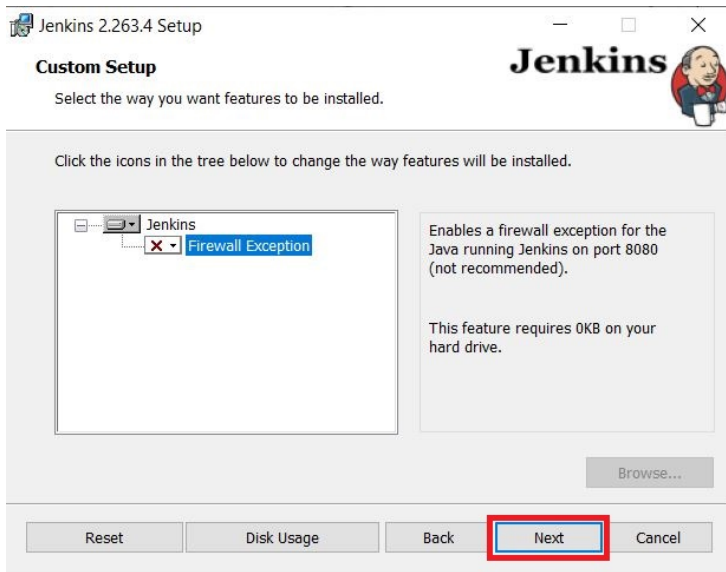
Once your Java home directory has been selected, click on Next to continue.



The image shows the 'Select Java home directory (JDK or JRE)' window of the Jenkins 2.263.4 Setup. The window title is 'Jenkins 2.263.4 Setup'. The main heading is 'Select Java home directory (JDK or JRE)'. Below this, it says 'Please select the path of a Java Development Kit or Java Runtime Environment. Only Java 1.8 and 11 are supported by Jenkins.' There is a text input field containing the path 'C:\Program Files\Java\jdk1.8.0\_261\'. To the left of the input field is a 'Change...' button. At the bottom of the window are three buttons: 'Back', 'Next', and 'Cancel'. The 'Next' button is highlighted with a red rectangle.

## Step 6: Custom setup

Select other services that need to be installed with Jenkins and click on Next.



## Step 7: Install Jenkins

Click on the Install button to start the installation of Jenkins.

### Unlocking Jenkins

When you first access a new Jenkins instance, you are asked to unlock it using an automatically-generated password.



## Step 1

Browse to <http://localhost:8080> (or whichever port you configured for Jenkins when installing it) and wait until the Unlock Jenkins page appears.



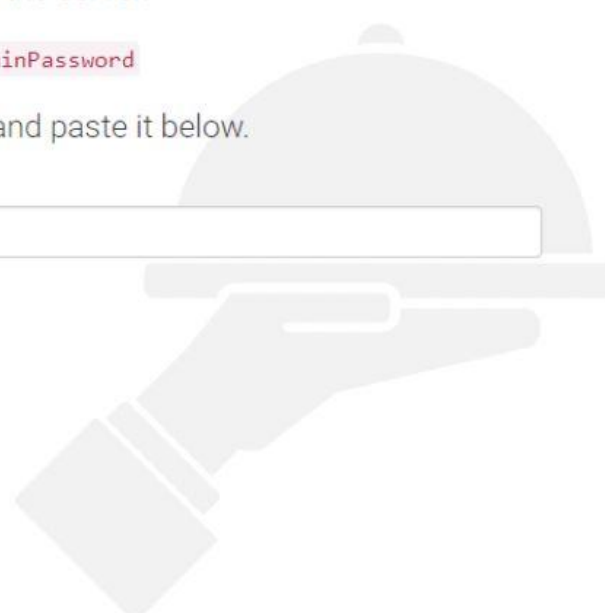
# Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

```
C:\Program Files (x86)\Jenkins\secrets\initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password



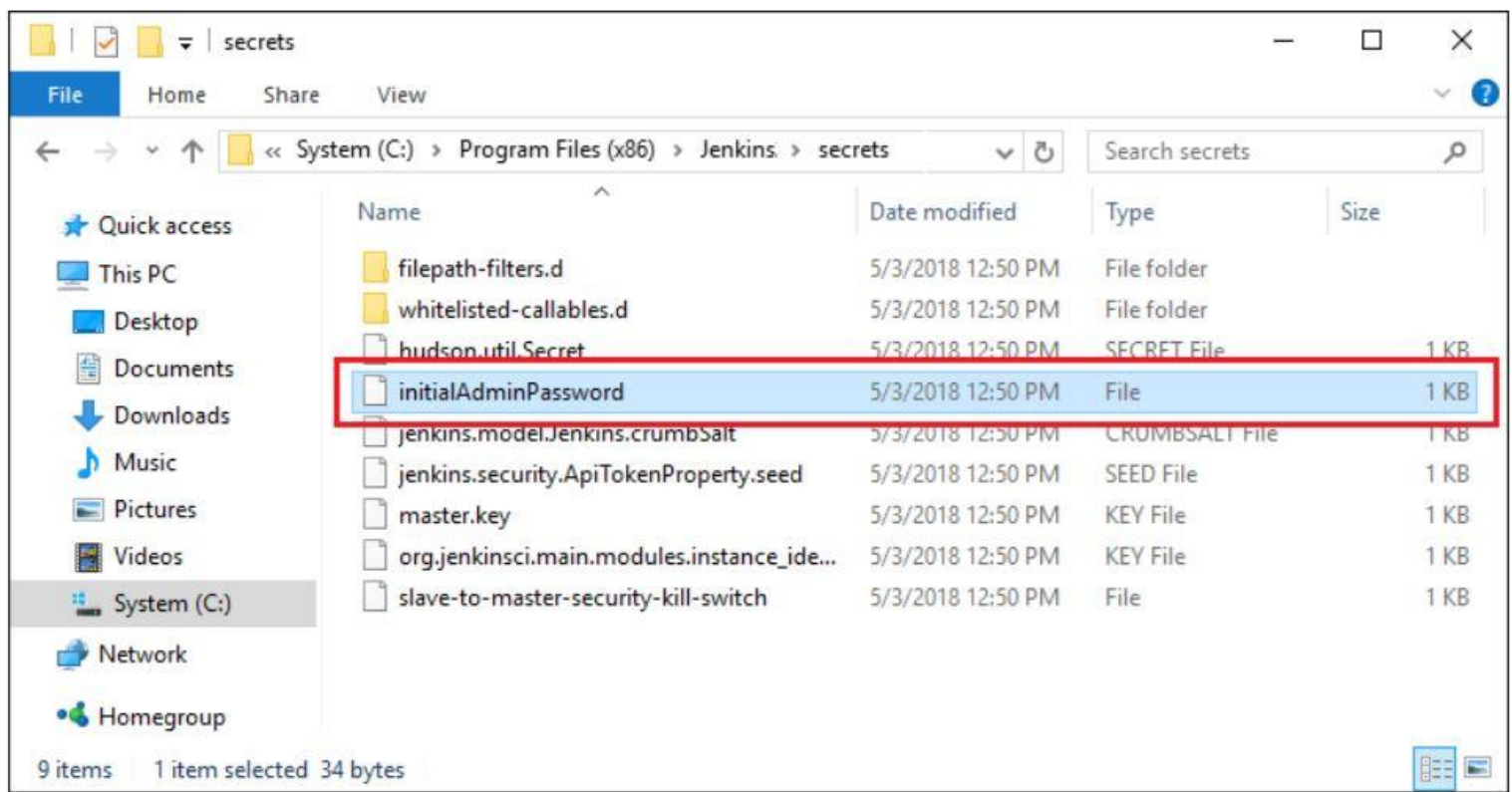
Continue

## Step 2

The initial Administrator password should be found under the Jenkins installation path (set at Step 2 in Jenkins Installation).

For default installation location to C:\Program Files\Jenkins, a file called initialAdminPassword can be found under C:\Program Files\Jenkins\secrets.

However, If a custom path for Jenkins installation was selected, then you should check that location for the initialAdminPassword file.



### Step 3

Open the highlighted file and copy the content of the initialAdminPassword file.

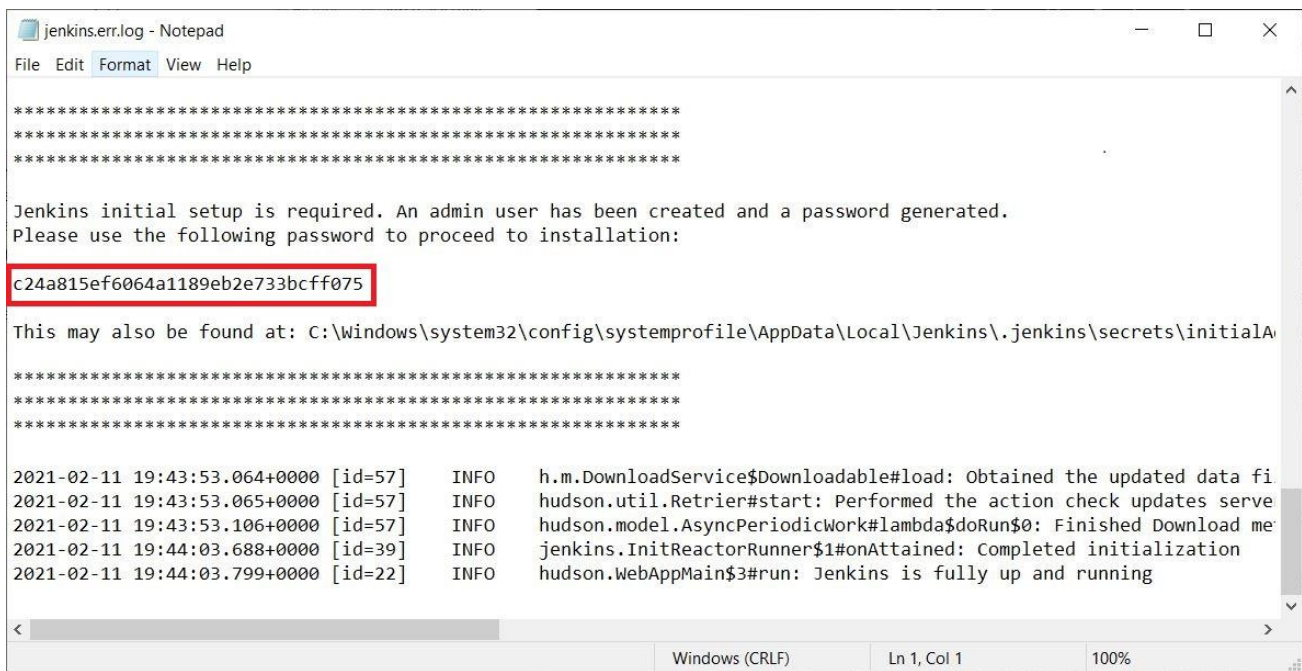


### Step 4

On the Unlock Jenkins page, paste this password into the Administrator password field and click Continue.

Notes:

- You can also access Jenkins logs in the jenkins.err.log file in your Jenkins directory specified during the installation.
- The Jenkins log file is another location (in the Jenkins home directory) where the initial password can also be obtained.



```
jenkins.err.log - Notepad
File Edit Format View Help

*****
*****
*****

Jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:

c24a815ef6064a1189eb2e733bcff075

This may also be found at: C:\Windows\system32\config\systemprofile\AppData\Local\Jenkins\.jenkins\secrets\initialAdminPassword

*****
*****
*****

2021-02-11 19:43:53.064+0000 [id=57] INFO h.m.DownloadService$Downloadable#load: Obtained the updated data fi
2021-02-11 19:43:53.065+0000 [id=57] INFO hudson.util.Retrier#start: Performed the action check updates serve
2021-02-11 19:43:53.106+0000 [id=57] INFO hudson.model.AsyncPeriodicWork#lambda$doRun$0: Finished Download me
2021-02-11 19:44:03.688+0000 [id=39] INFO jenkins.InitReactorRunner$1#onAttained: Completed initialization
2021-02-11 19:44:03.799+0000 [id=22] INFO hudson.WebAppMain$3#run: Jenkins is fully up and running

Windows (CRLF) Ln 1, Col 1 100%
```

This

password must be entered in the setup wizard on new Jenkins installations before you can access Jenkins's main UI. This password also serves as the default administrator account's password (with username "admin") if you happen to skip the subsequent user-creation step in the setup wizard.

## Customizing Jenkins with plugins

After unlocking Jenkins, the Customize Jenkins page appears. Here you can install any number of useful plugins as part of your initial setup.

Click one of the two options shown:

- **Install suggested plugins** - to install the recommended set of plugins, which are based on most common use cases.
- **Select plugins to install** - to choose which set of plugins to initially install. When you first access the plugin selection page, the suggested plugins are selected by default.

# Getting Started

✓ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding	OWASP Markup Formatter
✗ Timestampers	✗ Workspace Cleanup	✓ Ant	✗ Gradle	** Structs
✗ Pipeline	✗ GitHub Branch Source	✗ Pipeline: GitHub Groovy Libraries	✗ Pipeline: Stage View	** Token Macro
✗ Git	✗ SSH Build Agents	✗ Matrix Authorization Strategy	🔄 PAM Authentication	Build Timeout
🔄 LDAP	🔄 Email Extension	✓ Mailer		** Credentials
				** Trilead API
				** SSH Credentials
				** Pipeline: Step API
				** Plain Credentials
				Credentials Binding
				** SCM API
				** Pipeline: API
				Timestampers
				** Caffeine API
				** Script Security
				** Plugin Utilities API
				** Font Awesome API
				** Popper.js API
				** JQuery3 API
				** - required dependency

Jenkins 2.332.1

🏠 Back to Dashboard

⚙️ Manage Jenkins

🧩 Manage Plugins

## Installing Plugins/Upgrades

### Preparation

- Checking internet connectivity
- Checking update center connectivity
- Success

Cucumber json test reporting	✓ Success
cucumber-slack-notifier	✓ Success
Cucumber Trend Reports	✓ Success
cucumber-perf	✓ Success
Cucumber reports	✓ Success
Loading plugin extensions	✓ Success
Javadoc	✓ Success
Maven Integration	✓ Success
Loading plugin extensions	✓ Success

➡ [Go back to the top page](#)  
(you can start using the installed plugins right away)

➡ ☐ Restart Jenkins when installation is complete and no jobs are running

## Creating the first administrator user

Finally, after customizing Jenkins with plugins, Jenkins asks you to create your first administrator user.

1. When the Create First Admin User page appears, specify the details for your administrator user in the respective fields and click Save and Finish.
2. When the Jenkins is ready page appears, click Start using Jenkins.  
Notes:
  - This page may indicate Jenkins is almost ready! instead and if so, click Restart.
  - If the page does not automatically refresh after a minute, use your web browser to refresh the page manually.
3. If required, log in to Jenkins with the credentials of the user you just created and you are ready to start using Jenkins!

## Using credentials

There are numerous 3rd-party sites and applications that can interact with Jenkins, for example, artifact repositories, cloud-based storage systems and services, and so on.

A systems administrator of such an application can configure credentials in the application for dedicated use by Jenkins. This would typically be done to "lock down" areas of the application's functionality available to Jenkins, usually by applying access controls to these credentials. Once a Jenkins manager (i.e. a Jenkins user who administers a Jenkins site) adds/configures these credentials in Jenkins, the credentials can be used by Pipeline projects to interact with these 3rd party applications.

Credentials stored in Jenkins can be used:

- anywhere applicable throughout Jenkins (i.e. global credentials),
- by a specific Pipeline project/item
- by a specific Jenkins user

Jenkins can store the following types of credentials:

- **Secret text** - a token such as an API token (e.g. a GitHub personal access token),
- **Username and password** - which could be handled as separate components or as a colon separated string in the format `username:password` (read more about this in Handling credentials),
- **Secret file** - which is essentially secret content in a file,
- **SSH Username with private key** - an SSH public/private key pair,
- **Certificate** - a PKCS#12 certificate file and optional password, or
- **Docker Host Certificate Authentication credentials.**

## How to Create a New Build Job in Jenkins

The freestyle build job is a highly flexible and easy-to-use option. You can use it for any type of project; it is easy to set up, and many of its options appear in other build jobs. Below is a step by step process to create a job in Jenkin.

=====


Procedure :-

1. Create a new item as a freestyle process


### Enter an item name

testing


» Required field



**Freestyle project**  
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.




**Maven project**  
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.





**Multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.


OK

Cancel


 **Jenkins**


 **1**


 **Aniket Pardhi**


 **log out**


Dashboard ▸ testingJenkins ▸


 Back to Dashboard


 Status


 Changes


 Workspace


 Build Now

 Configure


 Delete Project

 GitHub


 Rename


 **Build History** trend ^

## Project testingJenkins

 Add description

Disable Project

 Workspace

 Recent Changes

### Permalinks

- Last build (#4), 50 min ago
- Last failed build (#4), 50 min ago
- Last unsuccessful build (#4), 50 min ago
- Last completed build (#4), 50 min ago

2. Add configuration to project :

General
 Source Code Management
 Build Triggers
 Build Environment
 Build
 Post-build Actions

**Description**
 [Plain text] [Preview](#)

☐ Discard old builds ?
 ☒ **GitHub project**

**Project url** ?


Save

Apply

Advanced...

3. Select the github **project** and add your repository link in it .

Created Project Repository on github **harsh2308-agr** with name as JenkinsTesting



 Pull requests
 Issues
 Marketplace
 Explore

harsh2308-agr / JenkinsTesting
 Public
 Pin
 Unwatch 1
 Fork
 Star 0

<> Code
 Issues
 Pull requests
 Actions
 Projects
 Wiki
 Security
 Insights
 Settings

master 1 branch 0 tags
 Go to file
 Add file
 Code

harsh2308-agr my project
 dfa717e 17 minutes ago 1 commit

Drivers	my project	17 minutes ago
src	my project	17 minutes ago
testdata	my project	17 minutes ago
.gitignore	my project	17 minutes ago
README.md	my project	17 minutes ago

README.md
 

robust-selenium
 Selenium Automation
 UPDATE:: Version 85.0.4183.83 (Official Build) (64-bit) Chrome Driver has been uploaded. To Execute test successfully in chrome, please have your latest chrome browser downloaded.

**About**
 Testing using Jenkins
 Readme
 0 stars
 1 watching
 0 forks

**Releases**
 No releases published
 Create a new release

**Packages**
 No packages published
 Publish your first package

**Languages**

4. Enter Build as **Invoke top- level Maven project**

## Build



### Invoke top-level Maven targets

X?

#### Goals

[Advanced...](#)[Add build step ▾](#)

5. Add post-build Actions → Cucumber repots

## Post-build Actions



### Cucumber reports

X?[Advanced...](#)[Add post-build action ▾](#)[Save](#)[Apply](#)

6. Save the project and Build the project.

7. Now goto to console output.



## Console Output

```
Started by user robust_selenium
Building in workspace C:\Program Files (x86)\Jenkins\workspace\DemoTesting
Cloning the remote Git repository
Cloning repository https://github.com/testersMindset/robust-selenium/
> C:\Program Files\Git\bin\git.exe init C:\Program Files (x86)\Jenkins\workspace\DemoTesting # timeout=10
Fetching upstream changes from https://github.com/testersMindset/robust-selenium/
> C:\Program Files\Git\bin\git.exe --version # timeout=10
using GIT_ASKPASS to set credentials demo
> C:\Program Files\Git\bin\git.exe fetch --tags --progress https://github.com/testersMindset/robust-selenium/ +refs/heads/*:refs/remotes/origin/*
> C:\Program Files\Git\bin\git.exe config remote.origin.url https://github.com/testersMindset/robust-selenium/ # timeout=10
> C:\Program Files\Git\bin\git.exe config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
> C:\Program Files\Git\bin\git.exe config remote.origin.url https://github.com/testersMindset/robust-selenium/ # timeout=10
Fetching upstream changes from https://github.com/testersMindset/robust-selenium/
using GIT_ASKPASS to set credentials demo
> C:\Program Files\Git\bin\git.exe fetch --tags --progress https://github.com/testersMindset/robust-selenium/ +refs/heads/*:refs/remotes/origin/*
> C:\Program Files\Git\bin\git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
> C:\Program Files\Git\bin\git.exe rev-parse "refs/remotes/origin/origin/master^{commit}" # timeout=10
Checking out Revision 7cecbb7e221dfccfc1fc582a047040dd37733012 (refs/remotes/origin/master)
> C:\Program Files\Git\bin\git.exe config core.sparsecheckout # timeout=10
> C:\Program Files\Git\bin\git.exe checkout -f 7cecbb7e221dfccfc1fc582a047040dd37733012
```

8. You can later view cucumber reports generated as results

```
line : 1,
"elements": [
  {
    "before": [
      {
        "result": {
          "duration": 7647190644,
          "status": "passed"
        },
        "match": {
          "location": "CucumberHooks.initialize()"
        }
      }
    ],
    "line": 10,
    "name": "select dropdown options validations",
    "description": "",
    "id": "test-ability-to-checkout-cart-added-item;select-dropdown-options-validations;;2",
    "after": [
      {
        "result": {
          "duration": 649139341,
          "status": "passed"
        },
        "match": {
          "location": "CucumberHooks.tearDown()"
        }
      }
    ],
    "type": "scenario",
    "keyword": "Scenario Outline",
    "ctenc": [
```

9. Statistics of testing can also be seen

 Cucumber reports

# Features Statistics

The following graphs show passing and failing statistics for features



Feature	Steps					Total	Scenarios			Features	
	Passed	Failed	Skipped	Pending	Undefined		Passed	Failed	Total	Duration	Status
Test ability to checkout cart added item	31	0	0	0	0	31	11	0	11	28.412	Passed



Feature Test ability to checkout cart added item		
Tags: @modelTest		
Scenario Outline select dropdown options validations >		1.594
Tags: @modelTest		
Scenario Outline select dropdown options validations v		1.318
Hooks >		
Steps >		
Hooks >		
Tags: @modelTest		
Scenario Outline select dropdown options validations >		1.138
Tags: @modelTest		
Scenario Outline select dropdown options validations v		1.375
Hooks >		
Steps v		
Given I am logged into home page		0.409

## Conclusion→

Specifying with respect to **postman** end to end testing capabilities of postman are tested. It is able to generate accessible test reports. It can be easily integrated with jenkins. API testing is successfully done with proper results in terms of environment variables, data types, datapassed etc. Data driven testing is also done with proper results where data is taken in the form of csv dataset. We can use these procedures to make sure that tests run, and that reports are generated whenever you push updates to your API.

---

Continuous Integration is the most important part of DevOps that is used to integrate various DevOps stages. Jenkins is the most famous Continuous Integration tool. Jenkins is an open-source automation tool written in Java with plugins built for Continuous Integration purposes. Jenkins can be used to build and test your software projects continuously making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build. Continuous delivery of software by integrating with a large number of testing and deployment technologies can be done by Jenkins.

---

**THE END!!**