Installing Docker from the Official Repository

Install Docker from the official Docker repository to ensure you get the latest stable program version. To access the official Docker repository, add the new package source to Ubuntu and then install Docker. Follow the steps below:

Update the Package Repository

Run the following command to update the system's package repository and ensure the latest prerequisite packages are installed:

sudo apt update

When prompted, enter your root password and press Enter to proceed with the update.

```
jenkins@jenkins-server:~$ sudo apt update
[sudo] password for jenkins:
Hit:1 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:2 http://in.archive.ubuntu.com/ubuntu noble InRelease
Get:3 http://in.archive.ubuntu.com/ubuntu noble-updates InRelease [89.7 kB]
Hit:4 http://in.archive.ubuntu.com/ubuntu noble-backports InRelease
Ign:5 https://pkg.jenkins.io/debian-stable binary/ InRelease
Hit:6 https://pkg.jenkins.io/debian-stable binary/ Release
Fetched 89.7 kB in 5s (19.9 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
10 packages can be upgraded. Run 'apt list --upgradable' to see them.
jenkins@jenkins-server:~$
```

Install Prerequisite Packages

The apt package manager requires a few prerequisite packages on the system to use packages over HTTPS. Run the following command to allow Ubuntu to access the Docker repositories over HTTPS:

sudo apt install apt-transport-https ca-certificates curl software-properties-common -y

Installing the prerequisite packages for Docker on Ubuntu.

```
jenkins@jenkins-server:~$ sudo apt install apt-transport-https ca-certificates curl software-properties-common -y
[sudo] password for jenkins:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240203).
ca-certificates set to manually installed.
curl is already the newest version (8.5.0-2ubuntu10.1).
curl set to manually installed.
software-properties-common is already the newest version (0.99.48).
software-properties-common set to manually installed.
The following NEW packages will be installed:
  apt-transport-https
0 upgraded, 1 newly installed, 0 to remove and 10 not upgraded.
Need to get 3,974 B of archives.
After this operation, 35.8 kB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu noble/universe amd64 apt-transport-https all 2.7.14build2 [3,974 B]
Fetched 3,974 B in 1s (6,223 B/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 98678 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.7.14build2_all.deb ...
Unpacking apt-transport-https (2.7.14build2) ...
Setting up apt-transport-https (2.7.14build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
jenkins@jenkins-server:~$
```

The command above:

- Allows apt to transfer files and data over https.
- Allows the system to check security certificates.
- Installs curl, a data-transfer utility.

- Adds scripts for software management.

Add GPG Key

A GPG key verifies the authenticity of a software package. Add the Docker repository GPG key to your system by running:

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

Adding the Docker GPG key to verify package authenticity.

```
jenkins@jenkins-server: ~
jenkins@jenkins-server:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
jenkins@jenkins-server:~$
```

The output should state OK, verifying the authenticity.

Add Docker Repository

Run the following command to add the Docker repository to apt sources:

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"

Adding the Docker official repository to the apt package manager.

```
jenkins@jenkins-server: ~
jenkins@jenkins-server:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Get:1 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Ign:2 https://pkg.jenkins.io/debian-stable binary/ InRelease
Hit:3 http://in.archive.ubuntu.com/ubuntu noble InRelease
Hit:4 https://pkg.jenkins.io/debian-stable binary/ Release
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:6 http://in.archive.ubuntu.com/ubuntu noble-updates InRelease [89.7 kB]
Hit:7 http://in.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:8 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [6,952 B]
Get:10 http://in.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [77.1 kB]
Get:11 http://in.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [35.7 kB]
Fetched 258 kB in 5s (50.5 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
jenkins@jenkins-server:~$
```

The command adds the official Docker repository and updates the package database with the latest Docker packages.

Specify Docker Installation Source

Execute the apt-cache command to ensure the Docker installation source is the Docker repository, not the Ubuntu repository. The apt-cache command queries the package cache of the apt package manager for the Docker packages we have previously added.

Run the following command:

apt-cache policy docker-ce

Specifying the Docker installation source.

```
jenkins@jenkins-server: ~
jenkins@jenkins-server:~$ apt-cache policy docker-ce
docker-ce:
  Installed: (none)
  Candidate: 5:26.1.3-1~ubuntu.24.04~noble
  Version table:
     5:26.1.3-1~ubuntu.24.04~noble 500
        500 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages
     5:26.1.2-1~ubuntu.24.04~noble 500
        500 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages
     5:26.1.1-1~ubuntu.24.04~noble 500
        500 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages
     5:26.1.0-1~ubuntu.24.04~noble 500
        500 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages
     5:26.0.2-1~ubuntu.24.04~noble 500
        500 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages
     5:26.0.1-1~ubuntu.24.04~noble 500
        500 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages
     5:26.0.0-1~ubuntu.24.04~noble 500
        500 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages
jenkins@jenkins-server:~$
```

The output states which version is the latest in the added source repository.

Install Docker

Install Docker by running:

sudo apt install docker-ce docker-ce-cli containerd.io -y

Installing Docker on Ubuntu using the official repository.

```
jenkins@jenkins-server: ~
jenkins@jenkins-server:~$ sudo apt install docker-ce -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
0 upgraded, 10 newly installed, 0 to remove and 10 not upgraded.
Need to get 122 MB of archives.
After this operation, 434 MB of additional disk space will be used.
Get:1 https://download.docker.com/linux/ubuntu noble/stable amd64 containerd.io amd64 1.6.32-1 [30.0 MB]
Get:2 http://in.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu noble/main amd64 libltdl7 amd64 2.4.7-7build1 [40.3 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu noble/main amd64 libslirp0 amd64 4.7.0-1ubuntu3 [63.8 kB]
Get:5 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-buildx-plugin amd64 0.14.0-1~ubuntu.24.04~noble [29.7 MB]
Get:6 http://in.archive.ubuntu.com/ubuntu noble/universe amd64 slirp4netns amd64 1.2.1-1build2 [34.9 kB]
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce-cli amd64 5:26.1.3-1~ubuntu.24.04~noble [14.6 MB]
Get:8 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce amd64 5:26.1.3-1~ubuntu.24.04~noble [25.3 MB]
Get:9 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce-rootless-extras amd64 5:26.1.3-1~ubuntu.24.04~noble [9,319 kB]
Get:10 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-compose-plugin amd64 2.27.0-1~ubuntu.24.04~noble [12.5 MB]
Fetched 122 MB in 10s (12.1 MB/s)
Selecting previously unselected package pigz.
(Reading database ... 98682 files and directories currently installed.)
Preparing to unpack .../0-pigz_2.8-1_amd64.deb ...
Unpacking pigz (2.8-1) ...
Selecting previously unselected package containerd.io.
Preparing to unpack .../1-containerd.io_1.6.32-1_amd64.deb ...
Unpacking containerd.io (1.6.32-1) ...
```

Wait for the installation process to complete.

Set docker to start automatically

To start docker automatically when the instance starts, you can use the below command:

sudo systemctl enable docker

Start docker

You can use the below command:

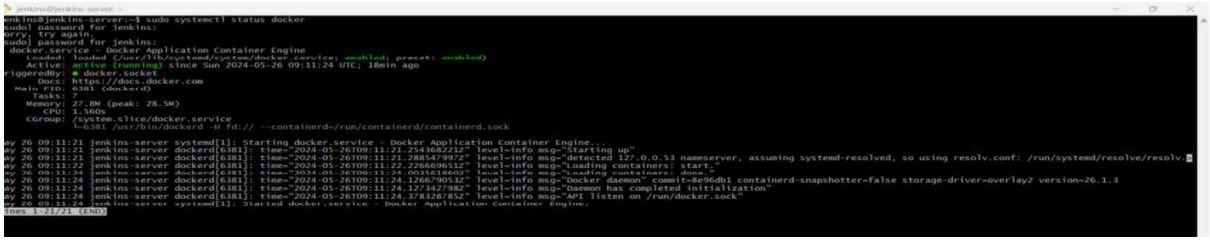
sudo systemctl start docker

Check Docker Status

Check if Docker is installed, the daemon started, and the process is enabled to start on boot. Run the following command:

sudo systemctl status docker

Checking the Docker daemon status.



The output states that the Docker daemon is up and running.

Set required permission for your Ubuntu user id to use Docker

Run the below command to add the user “devops” (it can be anything as you wish i.e. you had created in Ubuntu guest OS) to the “docker” group -

sudo usermod -a -G docker devops

How to restart docker

Do only if needed.

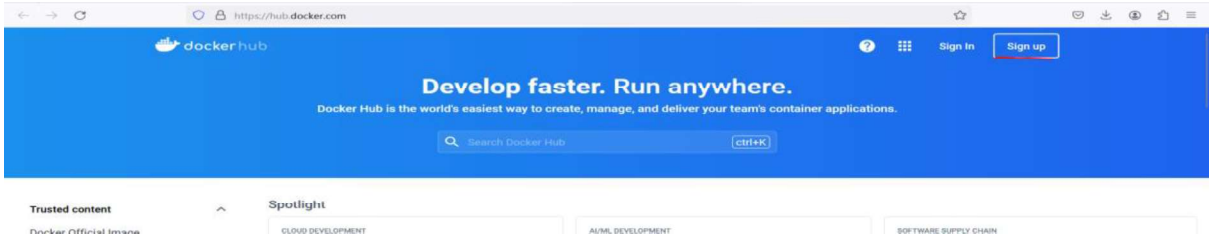
sudo systemctl daemon-reload

sudo systemctl restart docker

Create / manage Docker Hub account

Sign Up for Docker Hub account

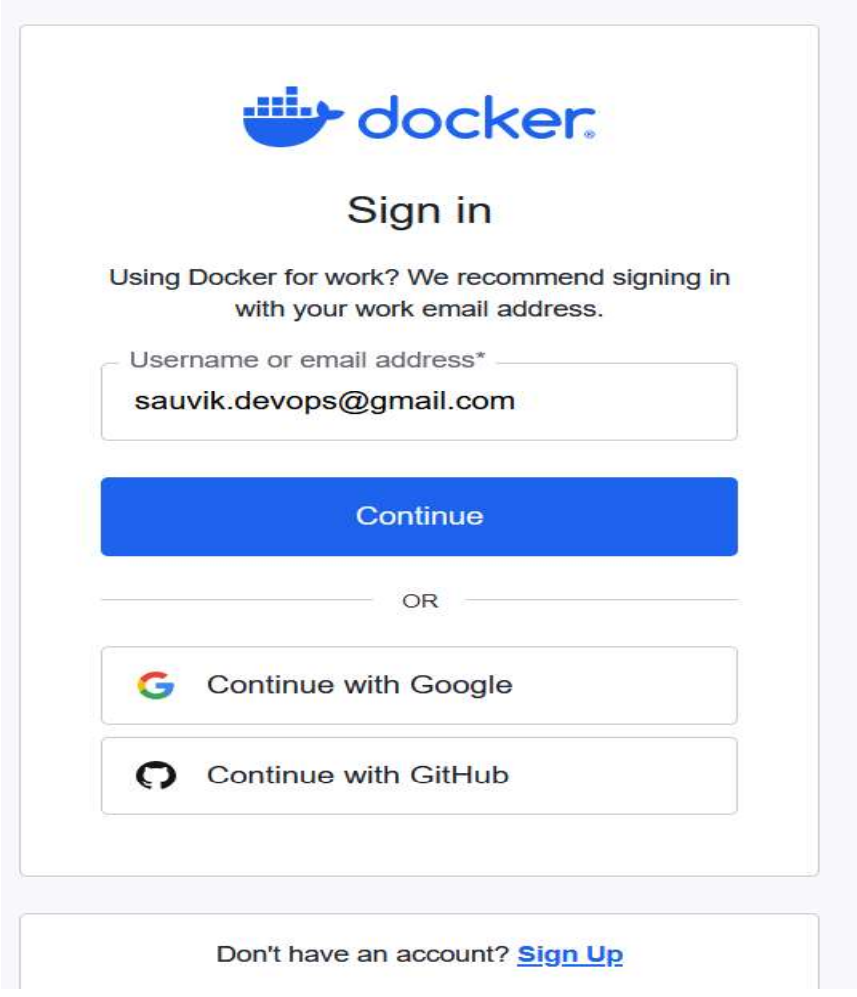
Visit https://hub.docker.com/ and click on “Sign up”



You can use your existing gmail account to signup for Docker Hub.

Sign In to Docker Hub account

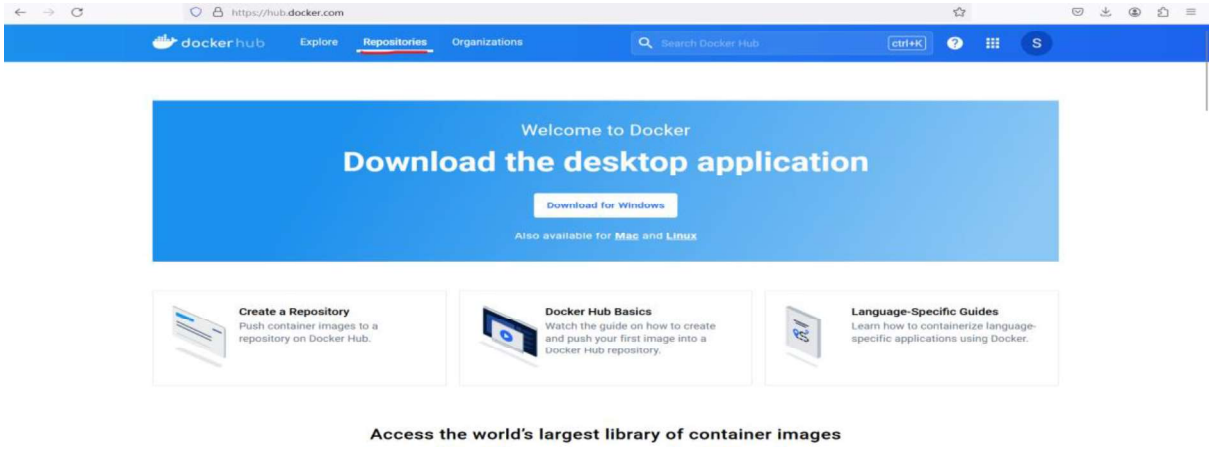
Once signup is done, please login to Docker Hub account using your gmail account.



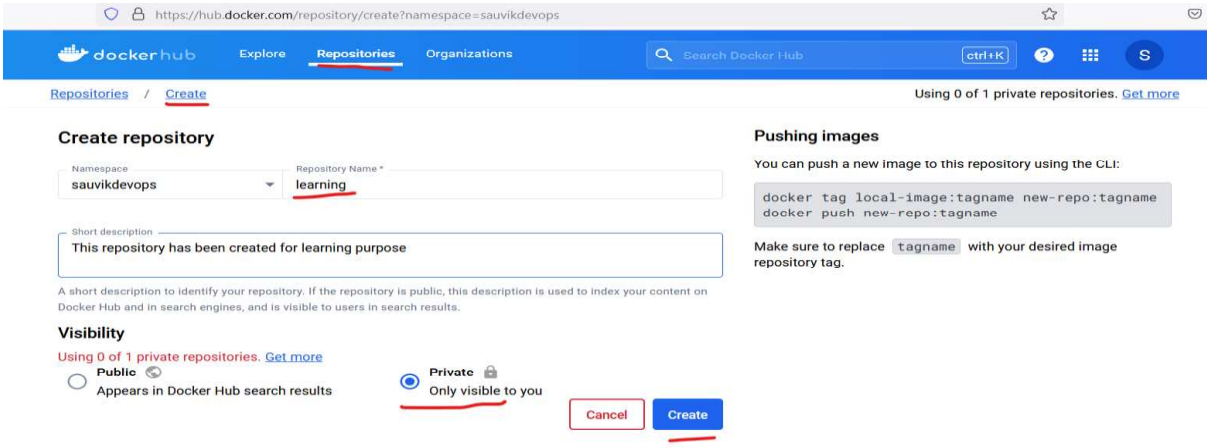
After sign in you will be able to see the home page.

Create repository in Docker Hub

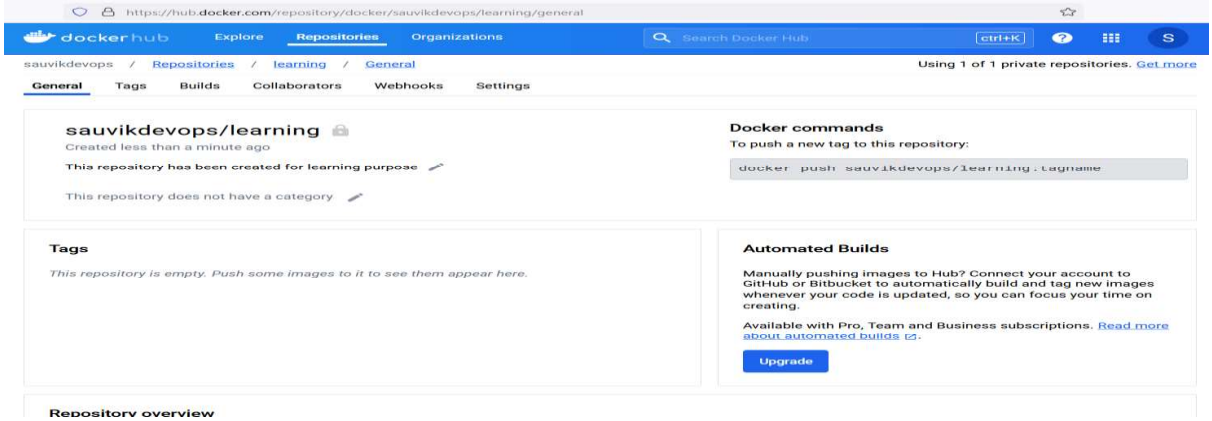
From Docker Hub home page after you login, click on '**Repositories**'



Give a name of your repository and select 'private' so that no anonymous login can happen. Now click on create to proceed.



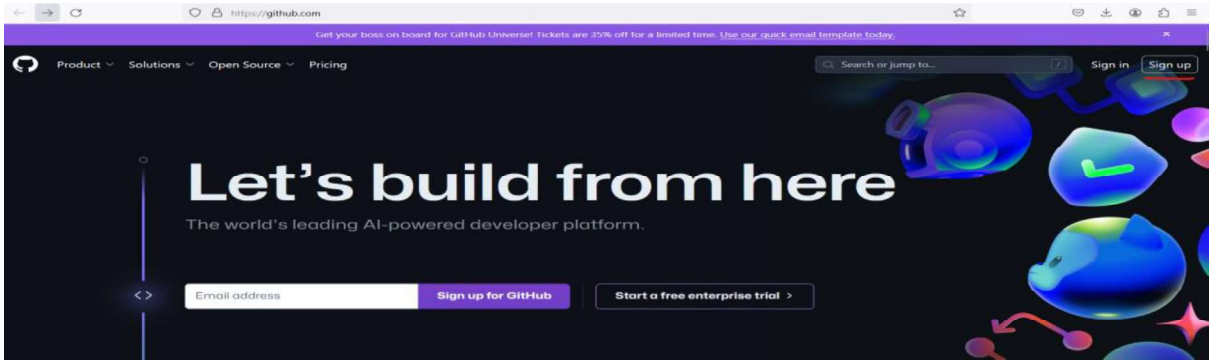
Your container registry would be created like below



Create / manage Git Hub account

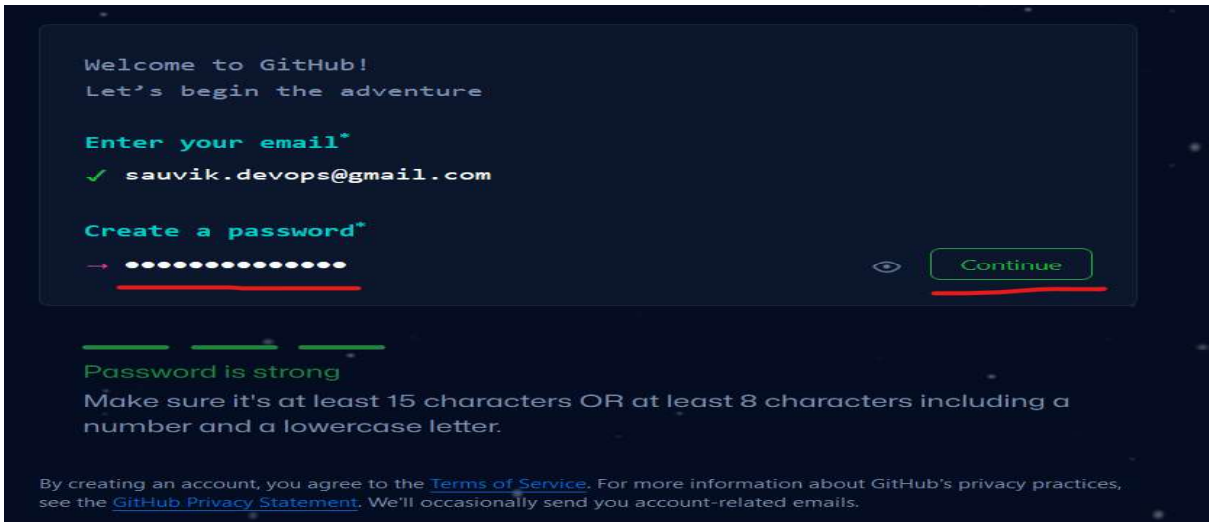
Signup for GitHub account

Visit https://github.com/ and click on “Sign up”



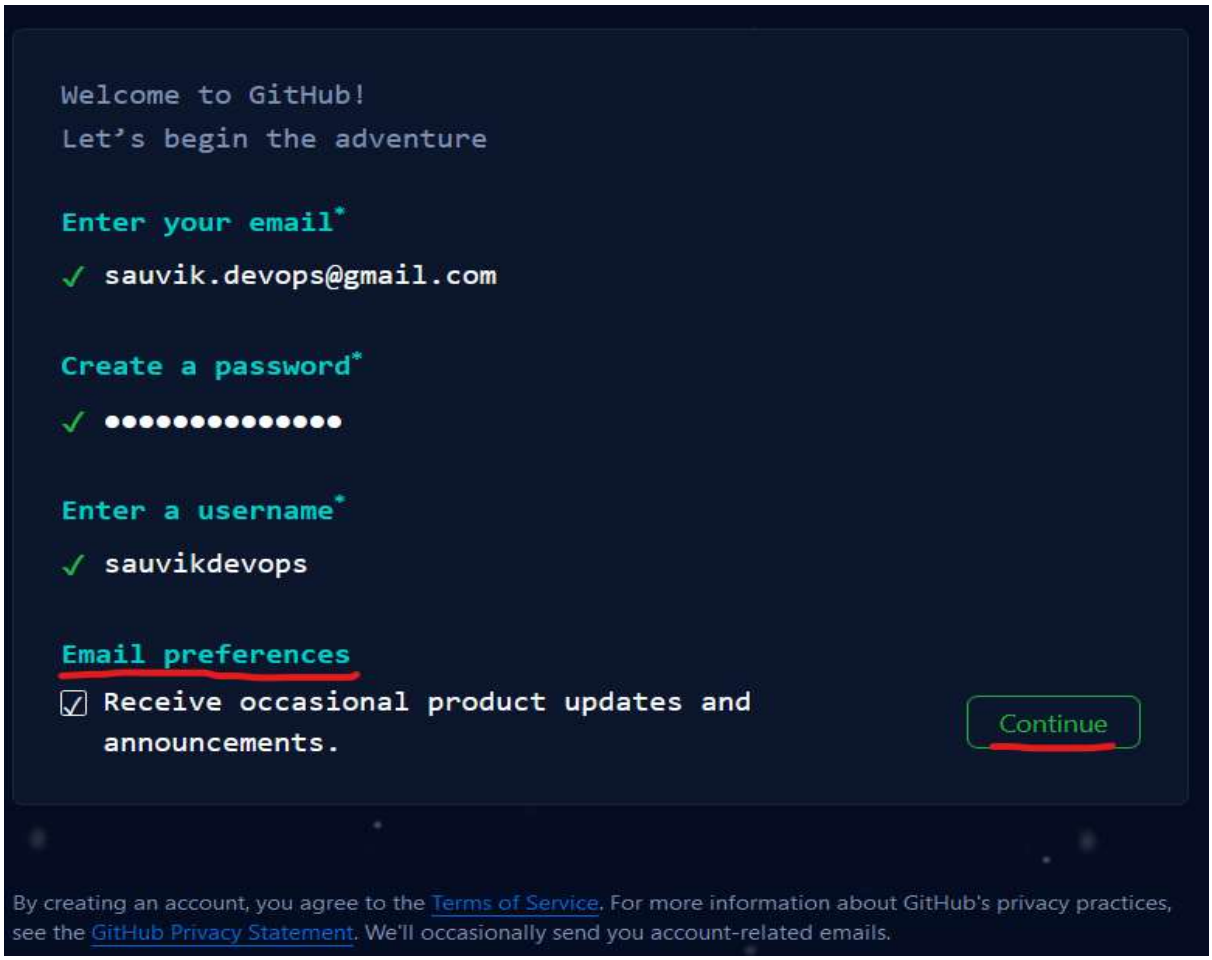
Put your e-mail ID and click on continue

Set a strong password and click on Continue.

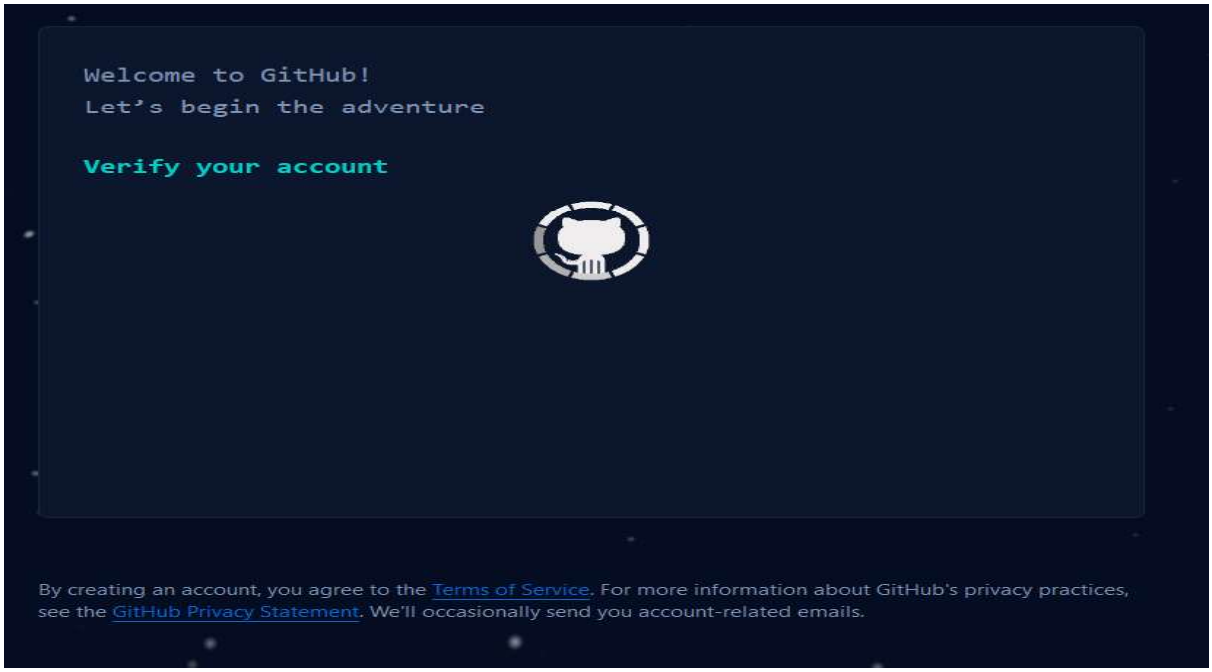


Pick a username for Docker Hub account and click on Continue.

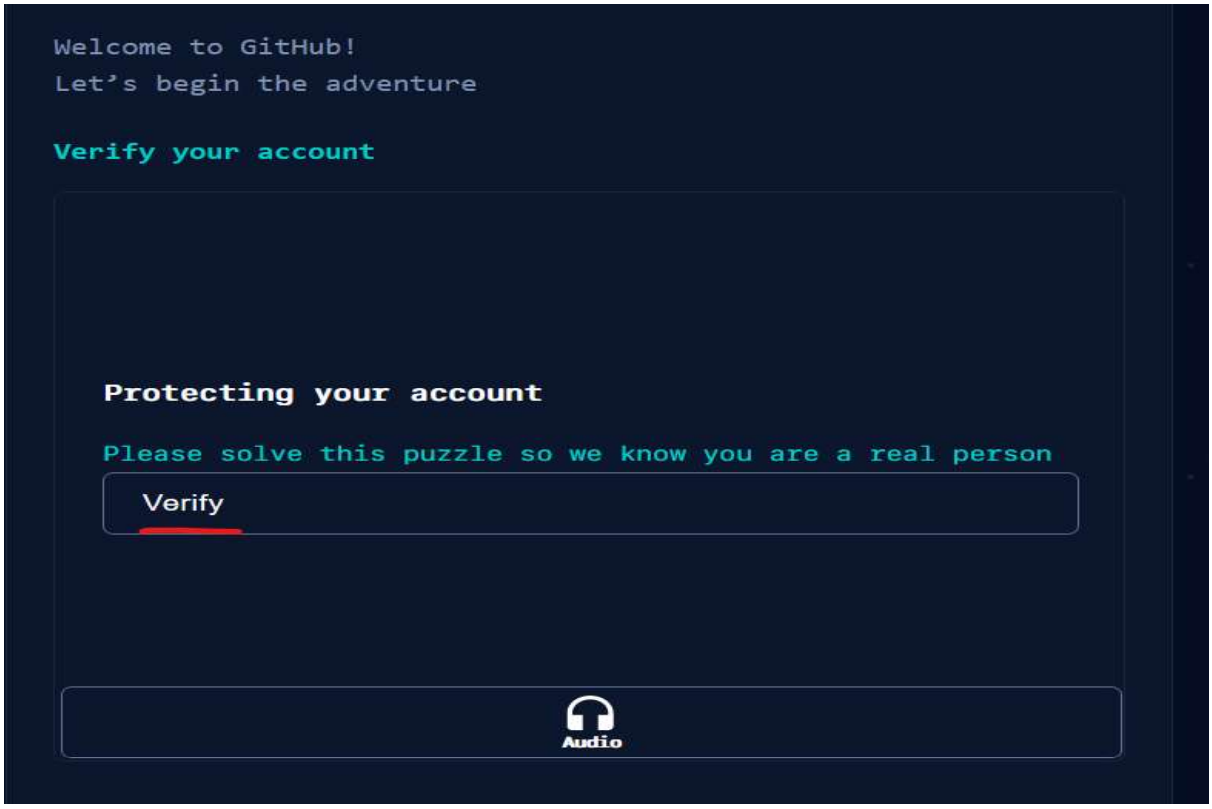
Click on Continue to proceed.



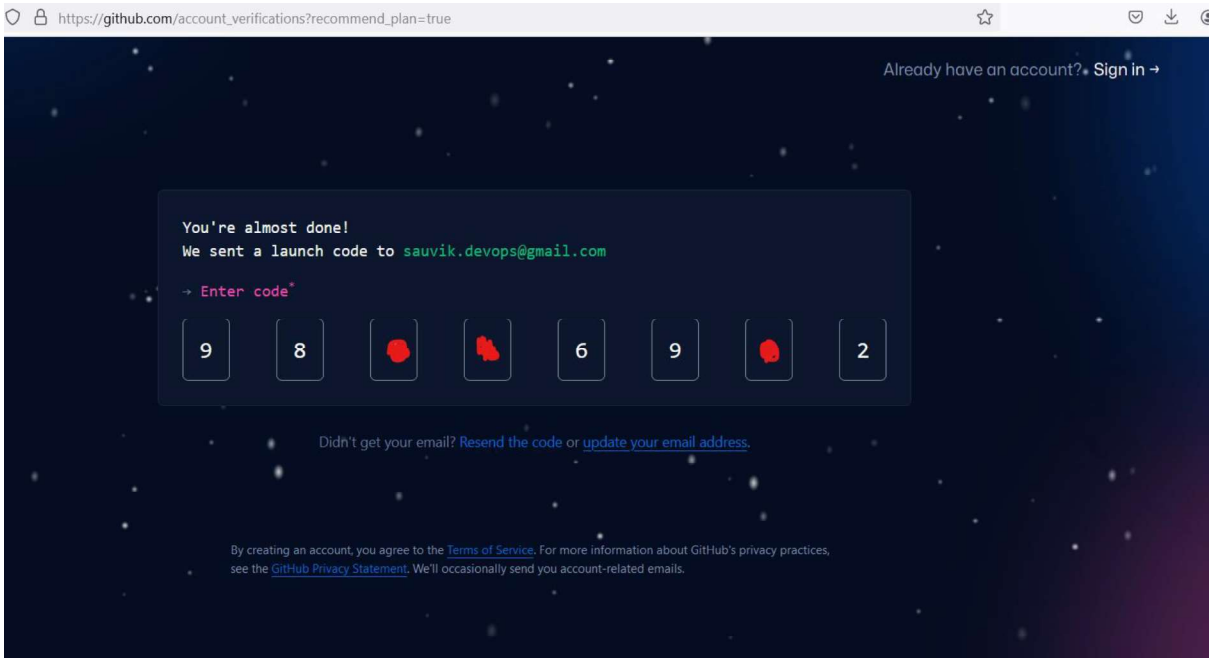
Your Github account will now get created.



Solve the puzzle for "Verify your account"



Check your email for launch code received from GitHub and then put on the GitHub account creation screen.

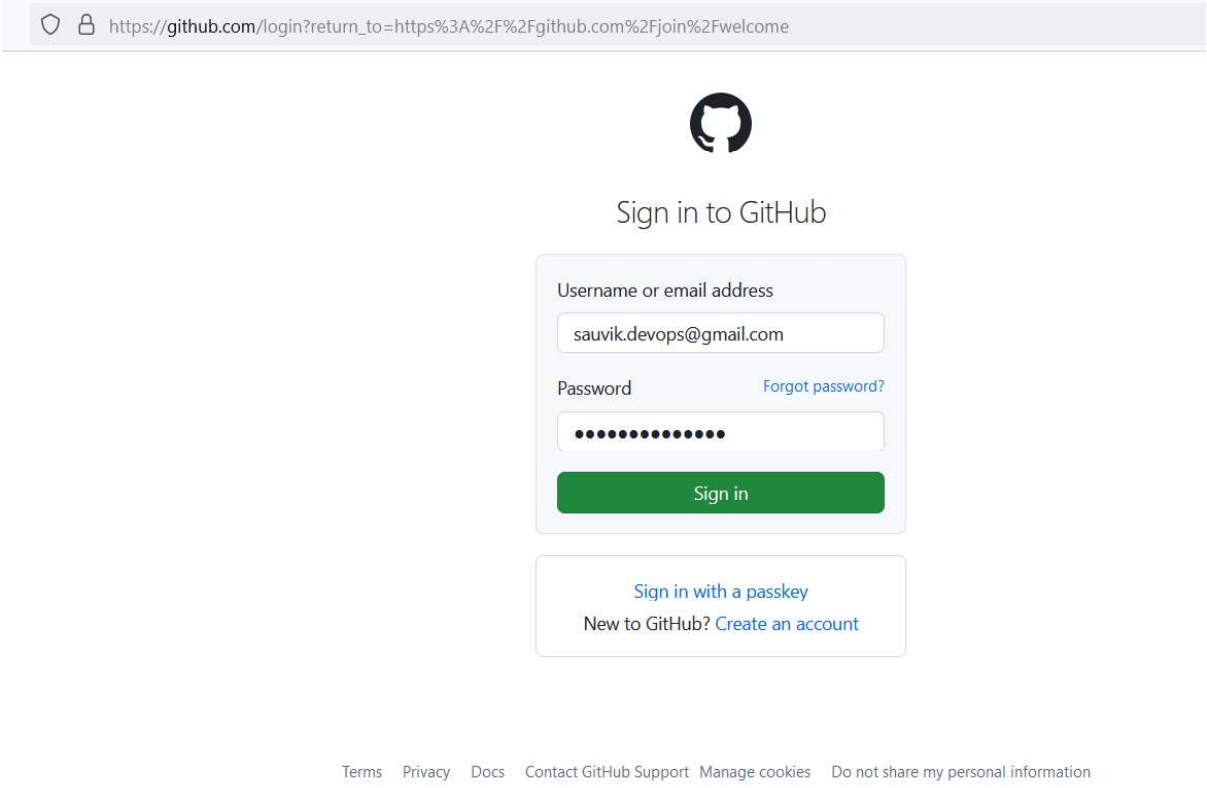


You are all set now.

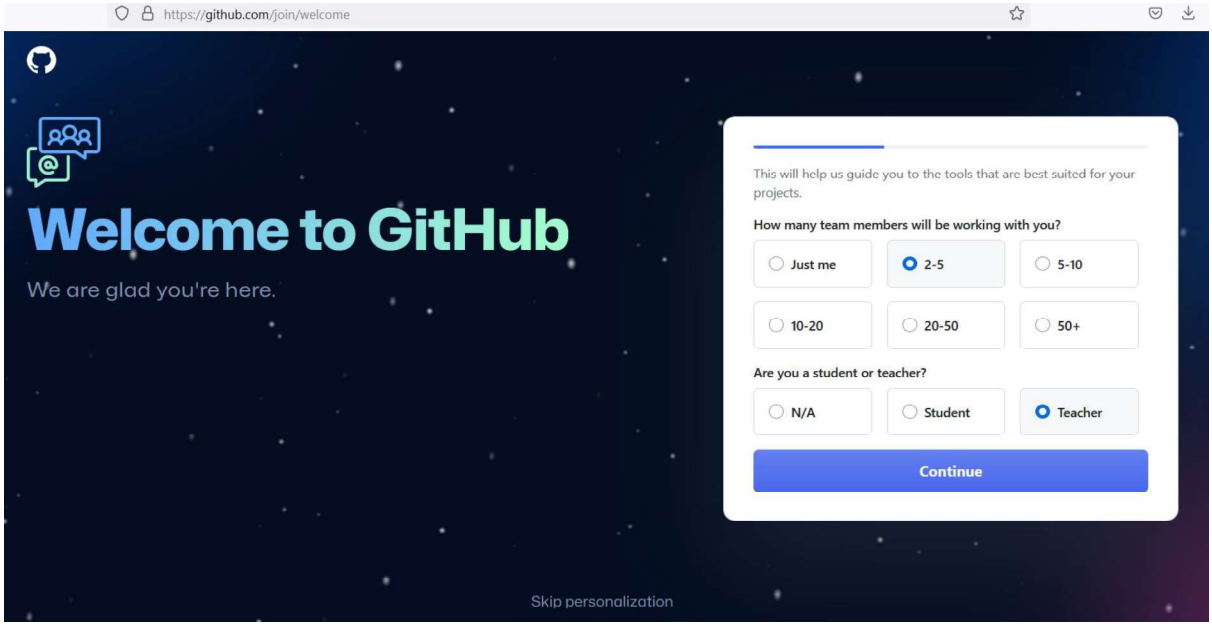
Login to GitHub account

Visit https://github.com/ and click on “Sign In”

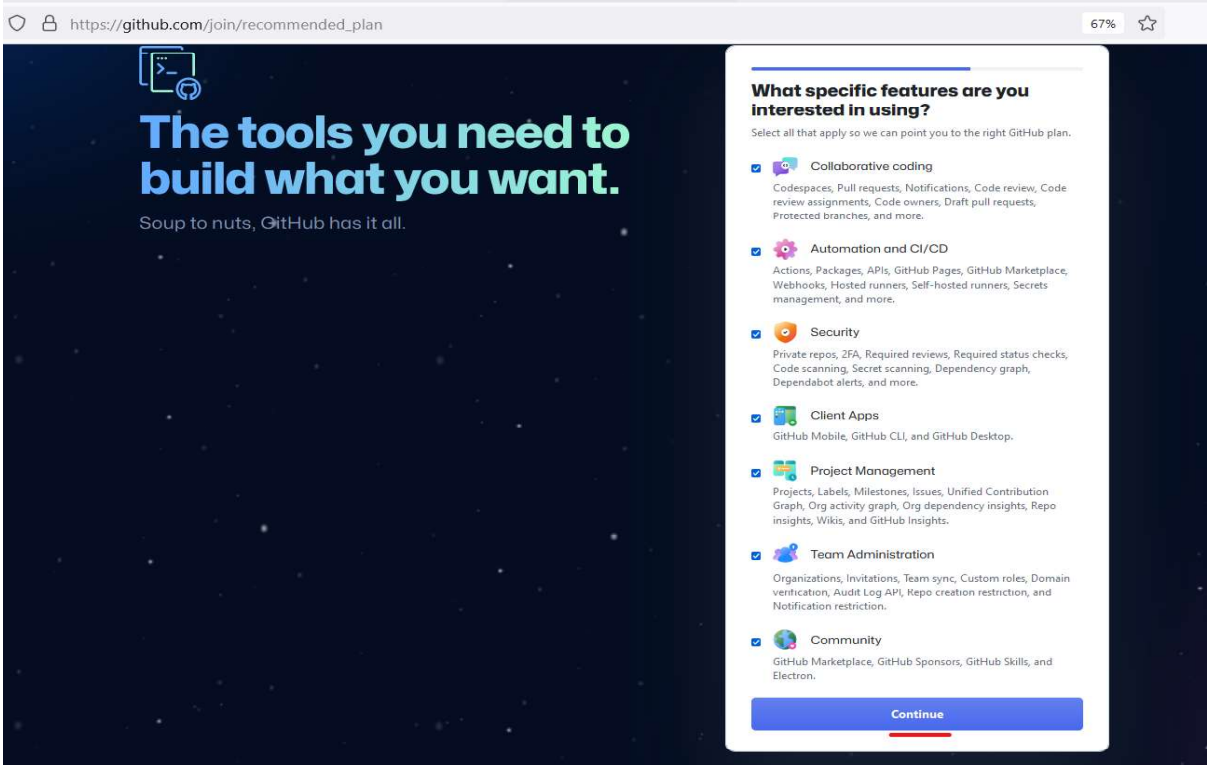
Put your GitHub account credential to login



Just fill the few questionaries to set the account as shown in below screen prints.

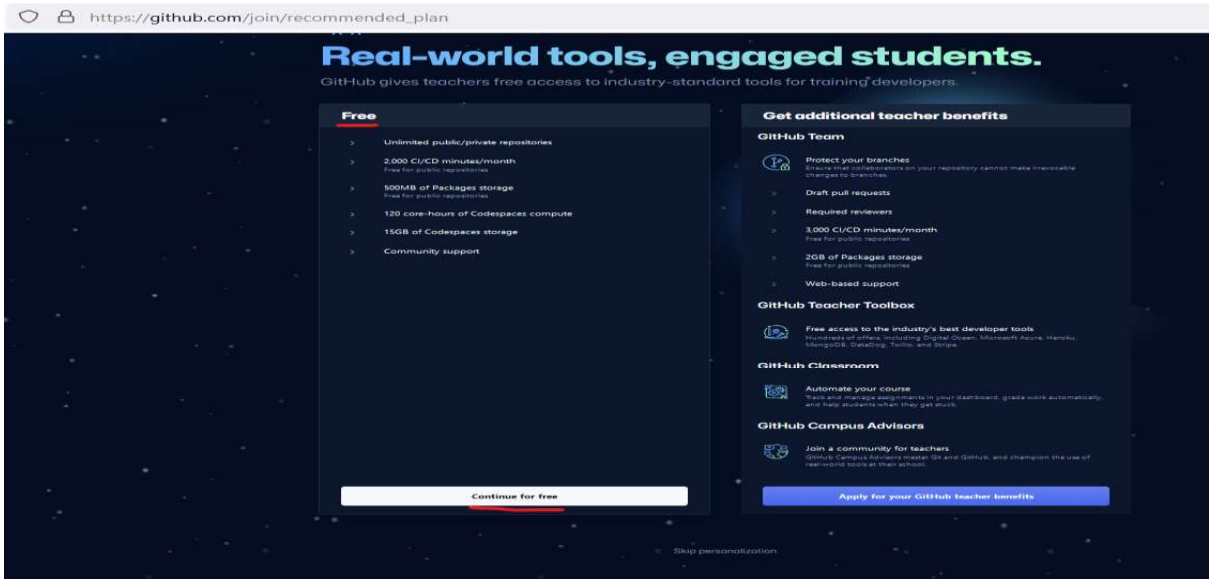


Click on continue

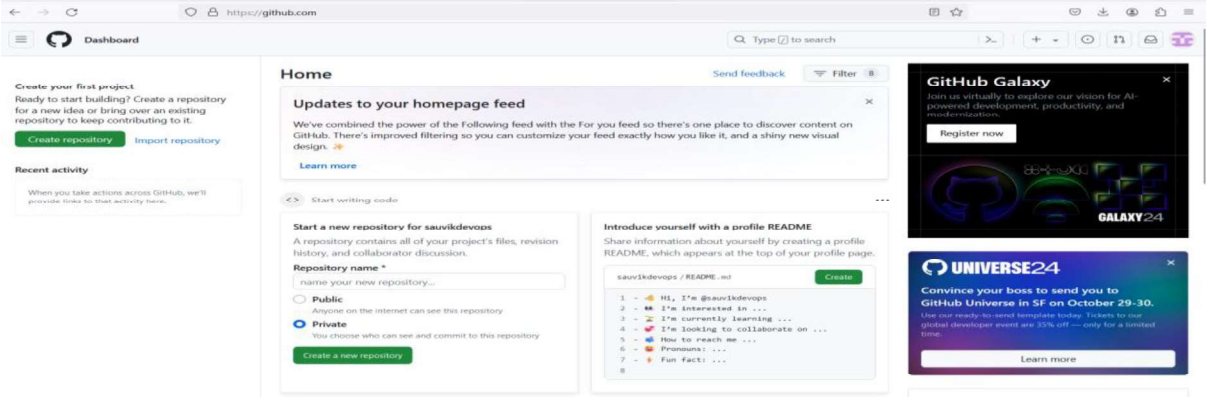


Select all options and click on – Continue

In the next screen, select “Free” account type.



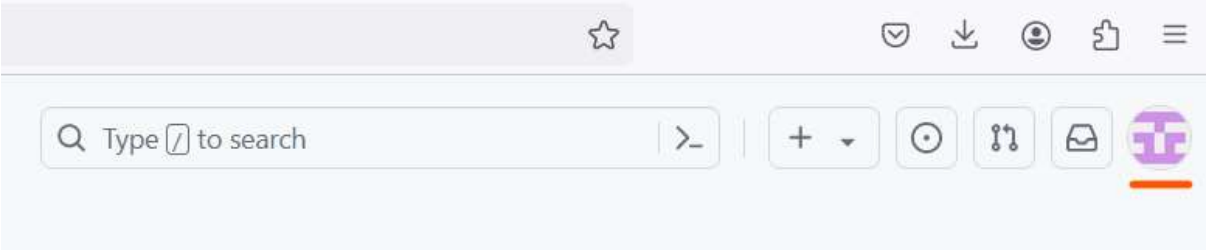
Your GitHub account setup is done and you will see the blow screen.



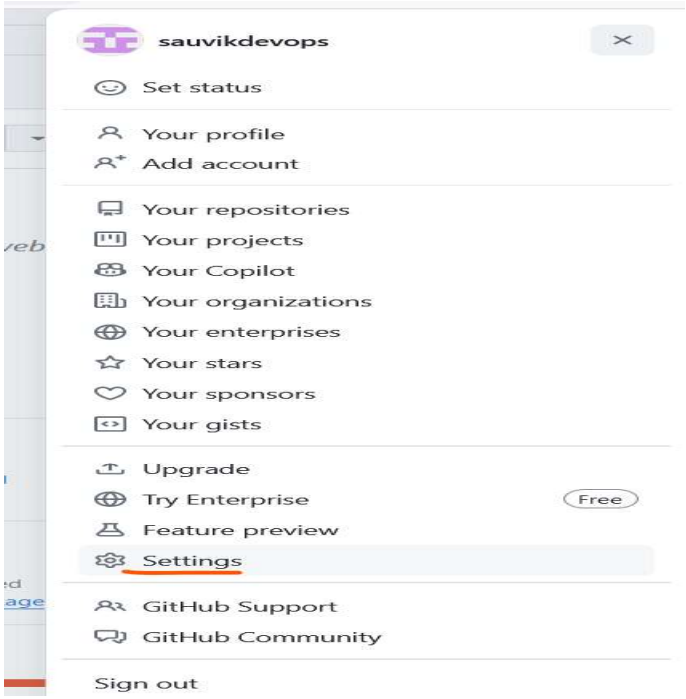
Generate Personal Access Token in GitHub

Personal access tokens are an alternative to using passwords for authentication to GitHub when using the GitHub API or the command line. Personal access tokens are intended to access GitHub resources on behalf of yourself.

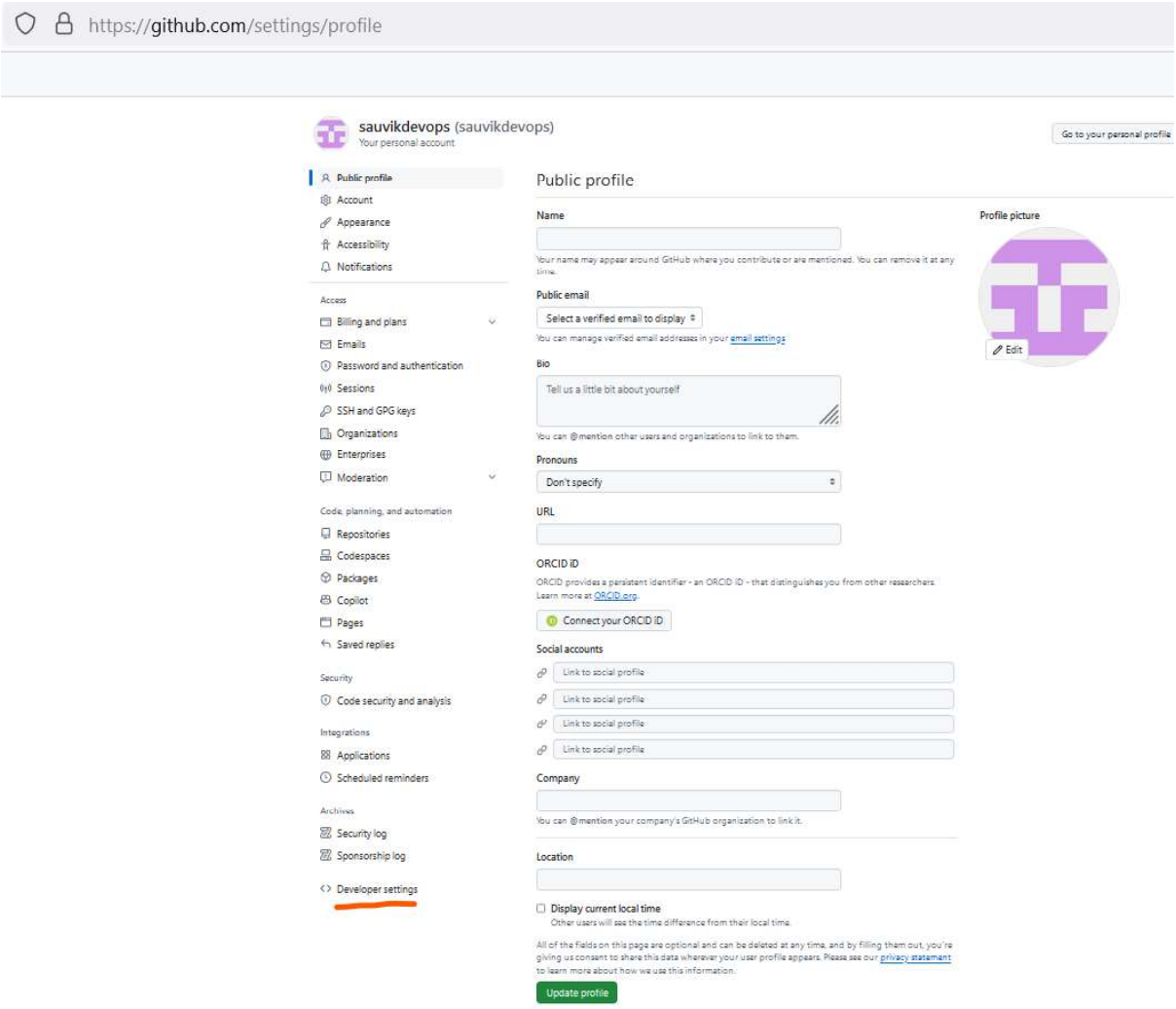
After you login to Click on the profile image (top right corner) of your GitHub account



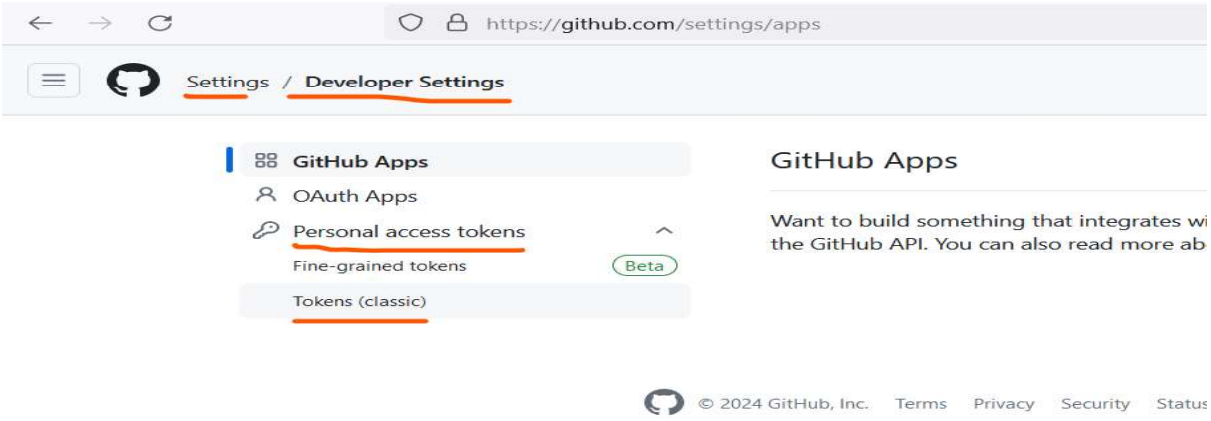
Now scroll down and click on Settings



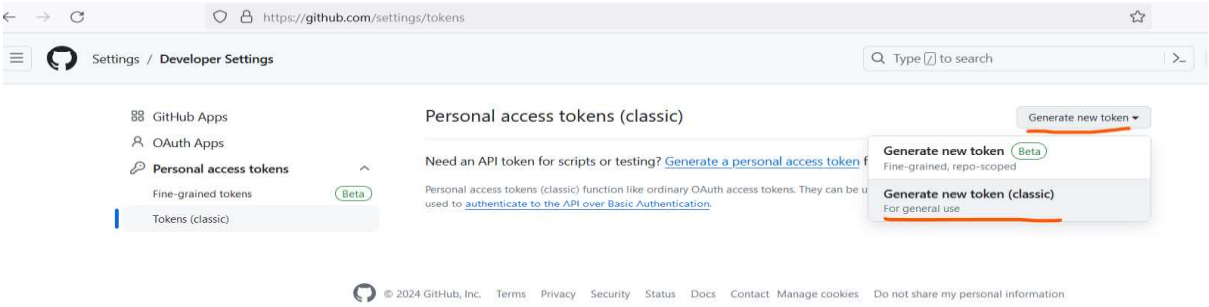
Scroll down and at bottom left Click on <> Developer Settings



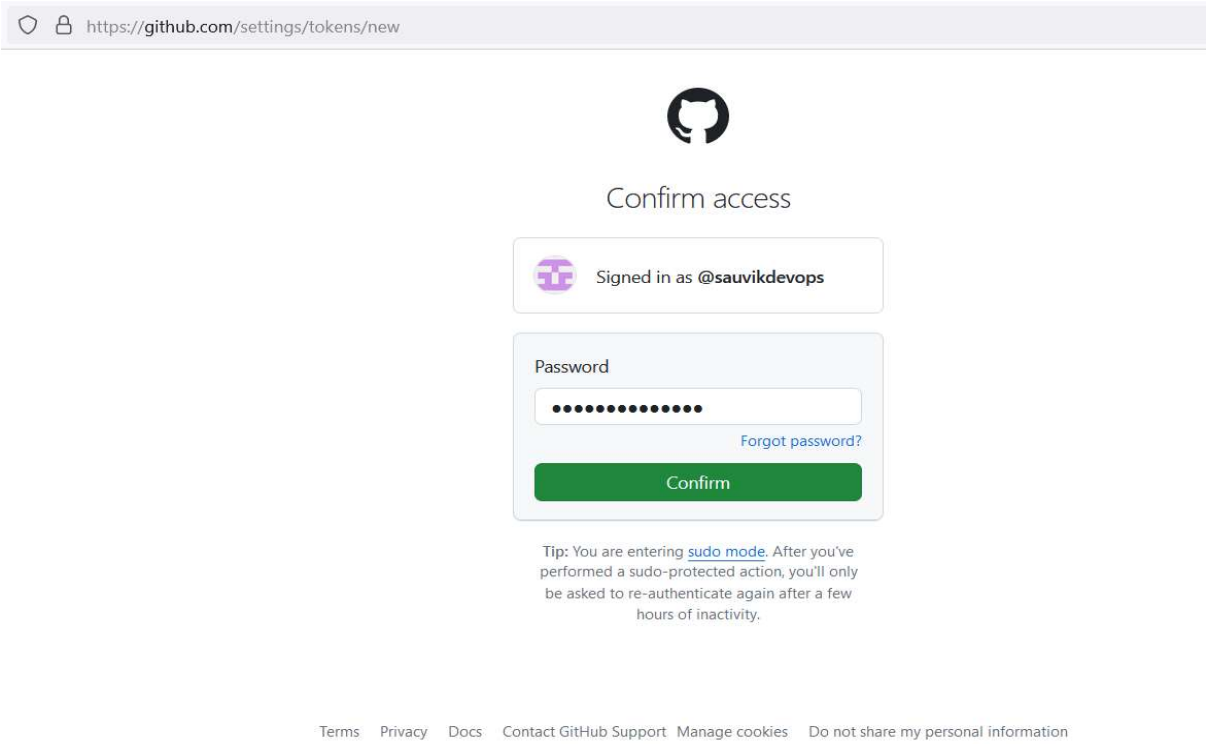
Click on Personal access tokens → Tokens (classic)



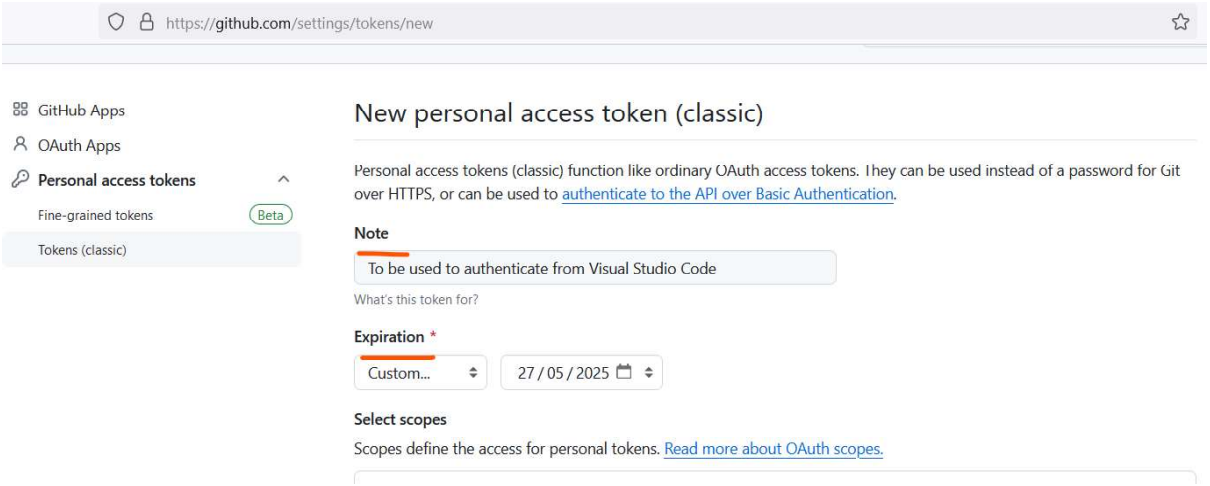
Click on Generate new token → Generate new token (classic)



Login if asked –

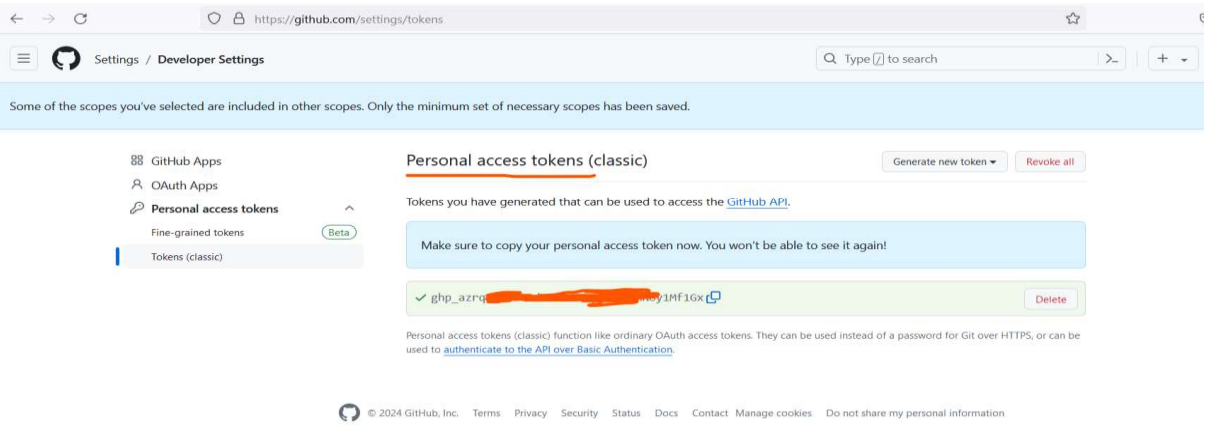


Add Note and Expiration for your token



Select the required scopes (if not sure, select all scopes) and click on Generate Token (at bottom of page)

Your Personal access token is created successfully



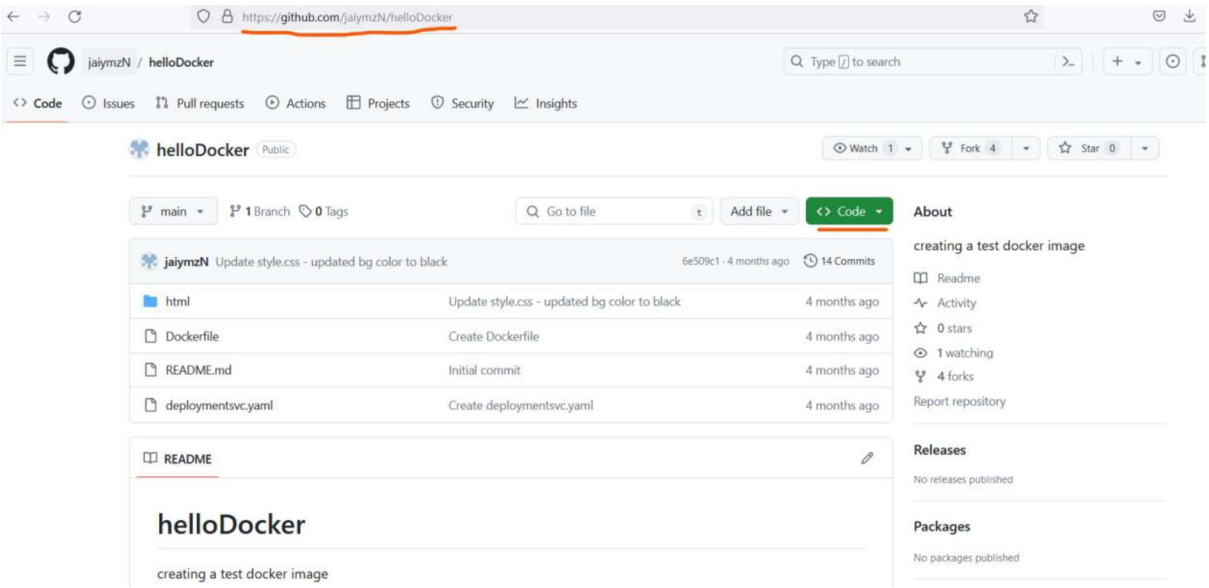
Note: Make sure to copy your personal access token now. You won't be able to see it again!

Clone a Git repository

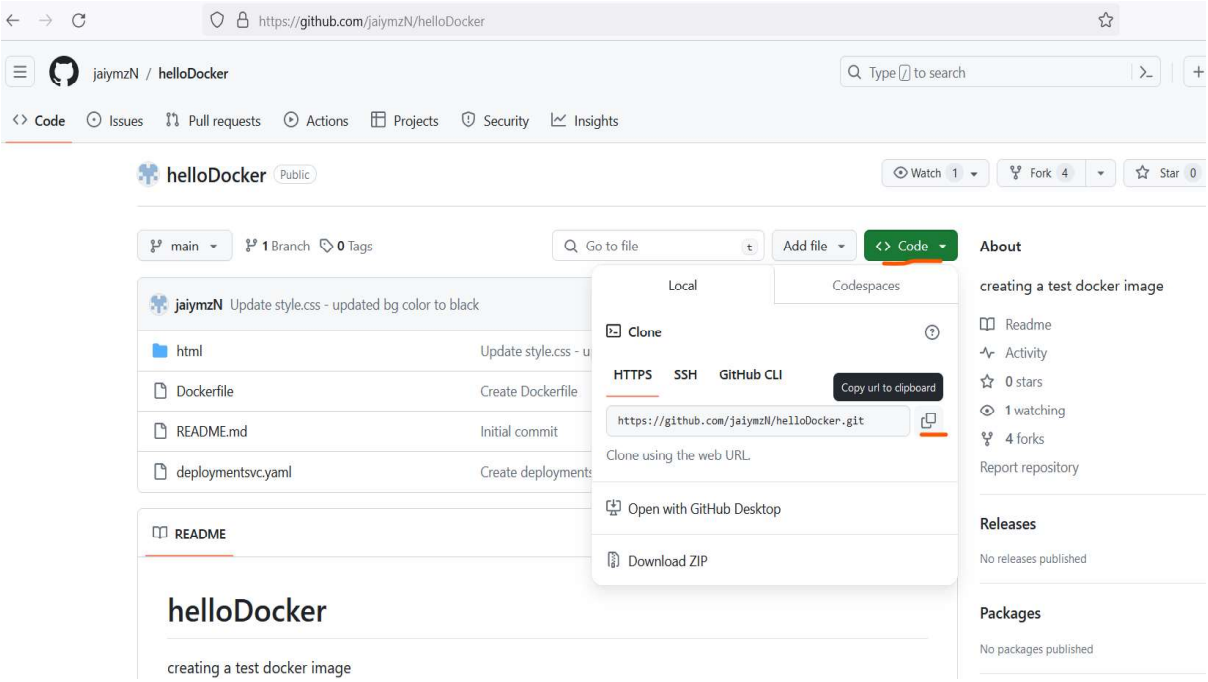
Go to any Git repo (in case of public repo, you would not need any credential of source repo, otherwise it is needed. So ask for credential / token for source repo to the owner).

For example, to clone the below public repo (we would not need any credential for this), open this URL in browser:

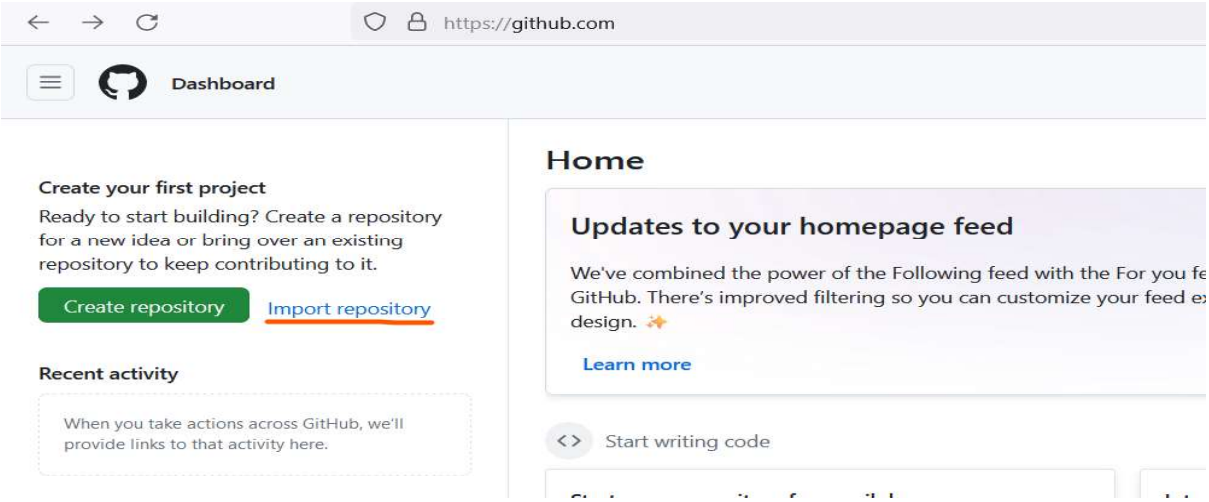
https://github.com/jaiymzN/helloDocker



Click on “Code” followed by copy sign →

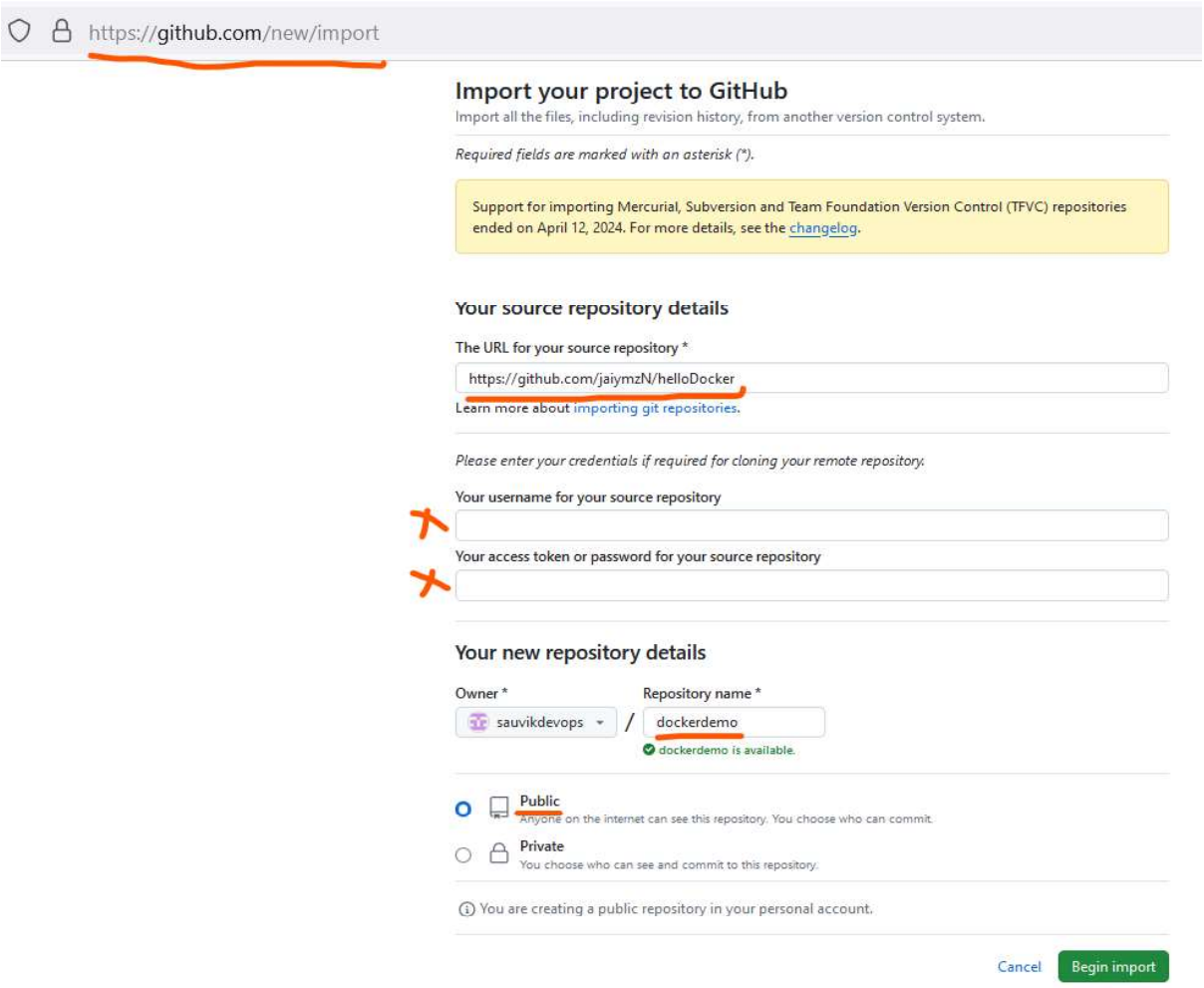


Now, login to your GitHub account (https://github.com/), and click on **“Import repository”**

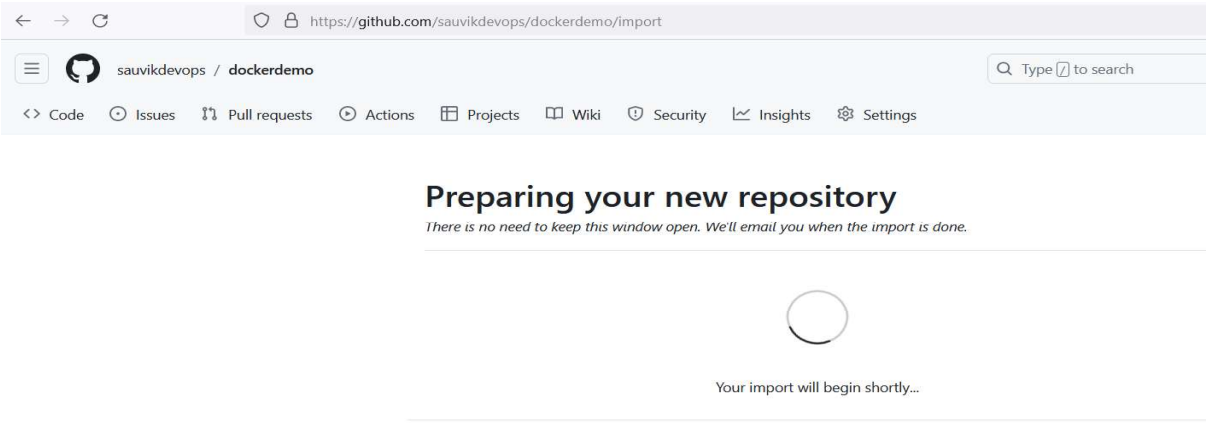


Now, do the following –

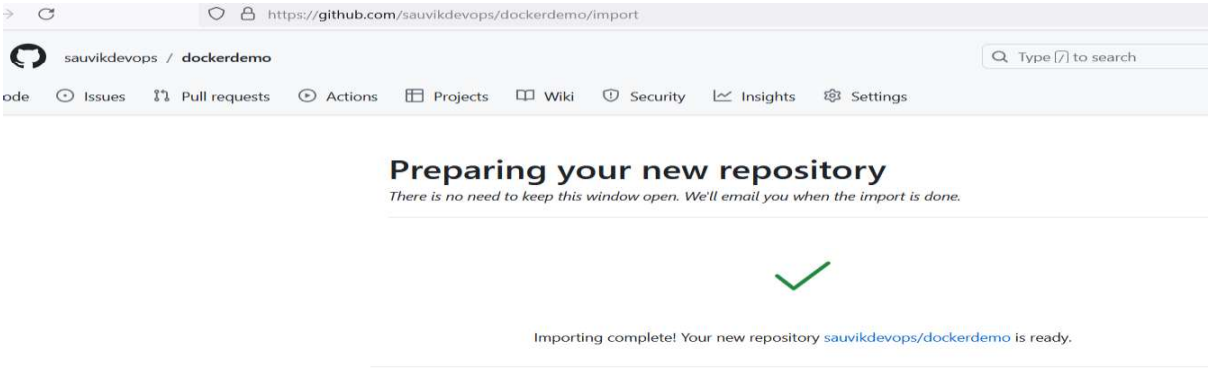
1. put the copied URL of your source repository. In this case, it is https://github.com/jaiymzN/helloDocker
2. Remove username and password of source repository since this is a public repo
3. Give a new "repo name" inside your GitHub account
4. Select the repo type as "Public"



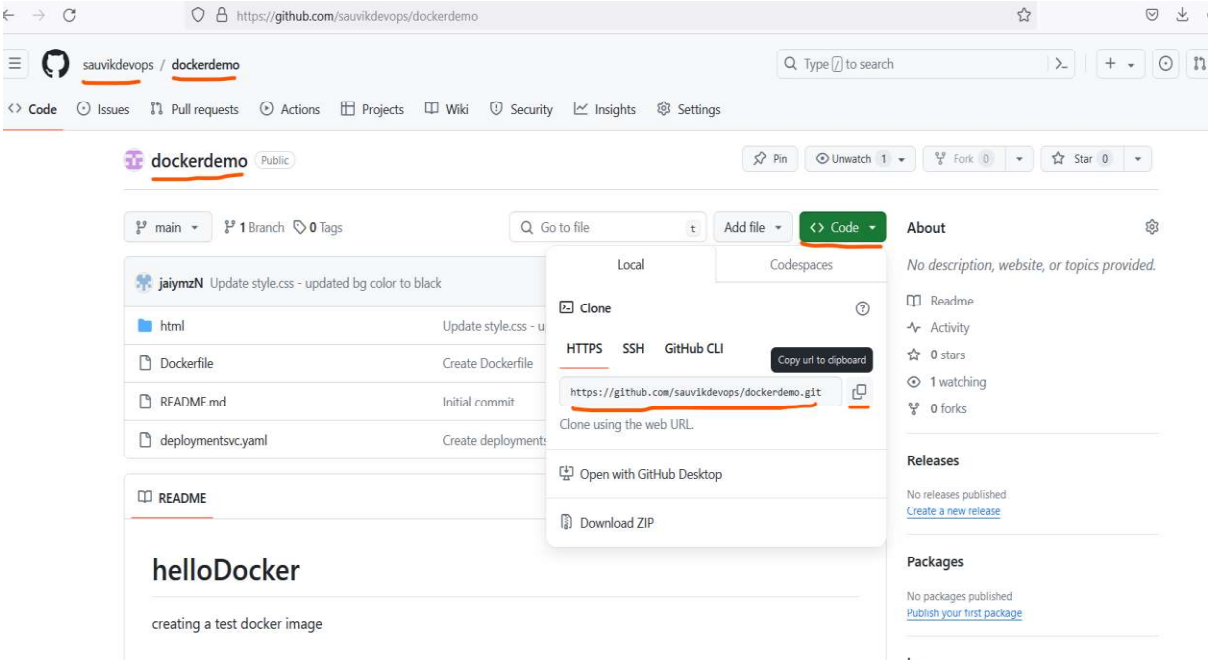
5. Click on "Begin Import"



Once import is complete, you will get confirmation as below:



Click on the newly created repo link (for example sauvikdevops/dockerdemo here) to go to there.



Note the repo URL which we would need later for CICD pipeline.