**Task**

**Creating an Auto-scaling group (ASG) with custom VPC using terraform modules and deploying it with the help of jenkins pipeline:**

So First to do this tasks the requirements are:

**1. AWS**

**2. Terraform**

**3. GitHub**

**4. GitBash**

**5. Jenkins**

**1. AWS:**

- AWS is abbreviated as Amazon Web Services it is a cloud provider.

- It provides on-demand cloud computing platforms and APIs to individuals, companies, and governments, on a metered, pay-as-you-go basis. Clients will often use this in combination with autoscaling.

**2. Terraform:**

- It is an IAC i.e., Infrastructure as A Code where we can deploy the resources in the AWS through code

- It is a quick deployment method of the Resources

**3. GitHub:**

- GitHub is a Source code Management, means the code which we write will be stored into GitHub.

- We have concept called Repository where we actually store the Code.

- It can be also used for version control in the organizations.

- Instead of saving the code in local machine it is the best practice to push it to the GitHub Repository

**4. GitBash:**

- GitBash is a tool used to push the code to the repository.

- We have many tools for pushing the code into the repository.

**5. Jenkins:**

- Jenkins is used for CI/CD pipeline

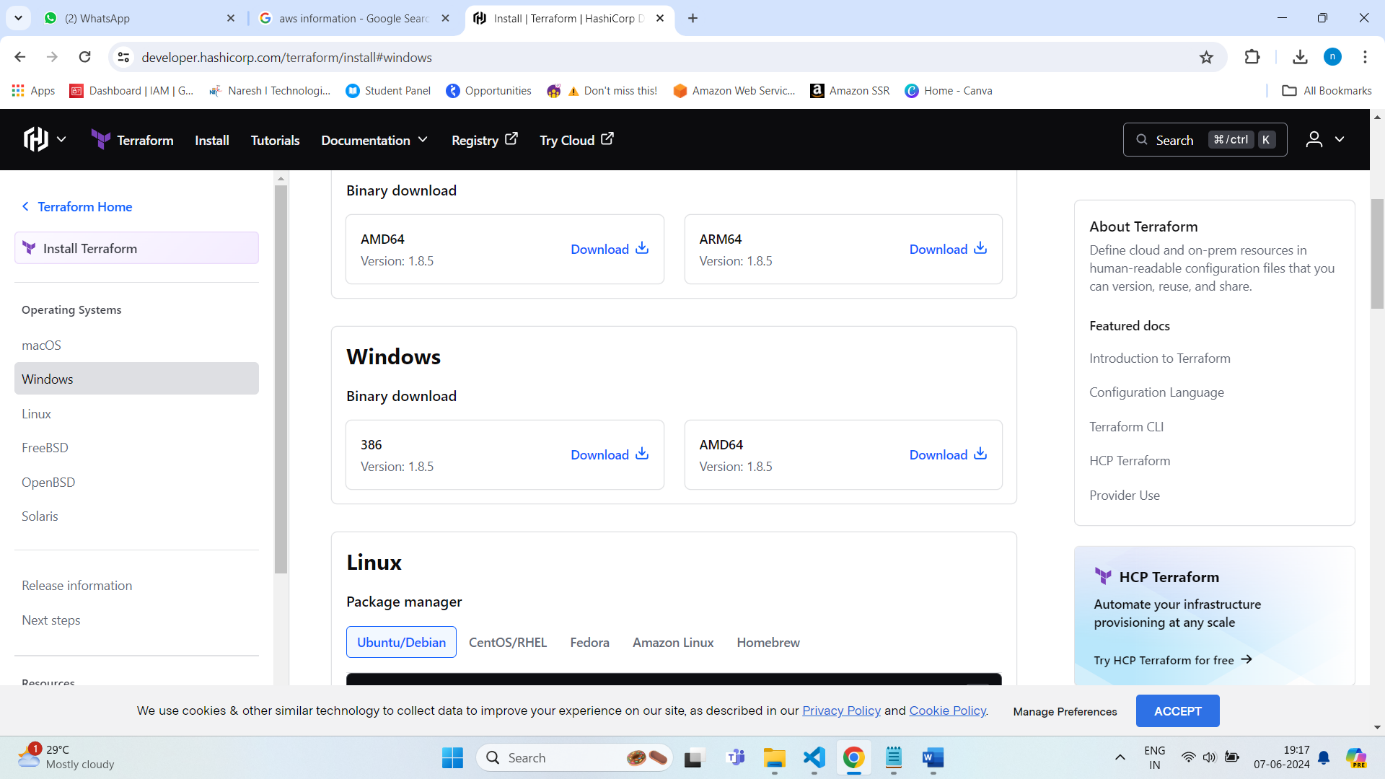
- It is used to create the pipeline from the GitHub Repository to Jenkins

- When the jenkins are triggered then the code which we written in the repository will get Build

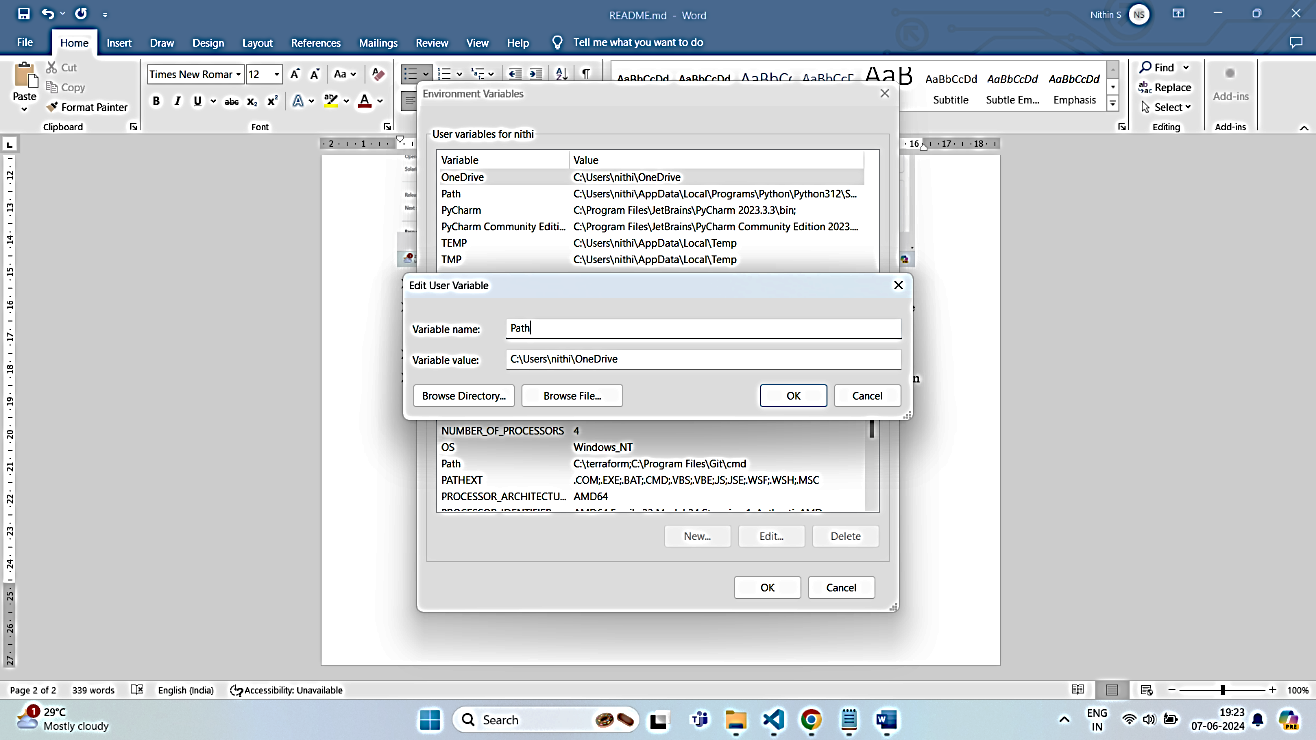
Steps we followed to complete these task is are below:

First thing we should do is install Terraform from the chrome which is devoloped by the company called **Hashicorp**

**Link :** [**https://developer.hashicorp.com/terraform/install#windows**](https://developer.hashicorp.com/terraform/install#windows)

****

* Download and Install the Terraform
* After installing we should declare the Environment variables, because it allows to execute the Terraform from anywhere in the computer
* Here we have to give the path of the Terraform.exe file



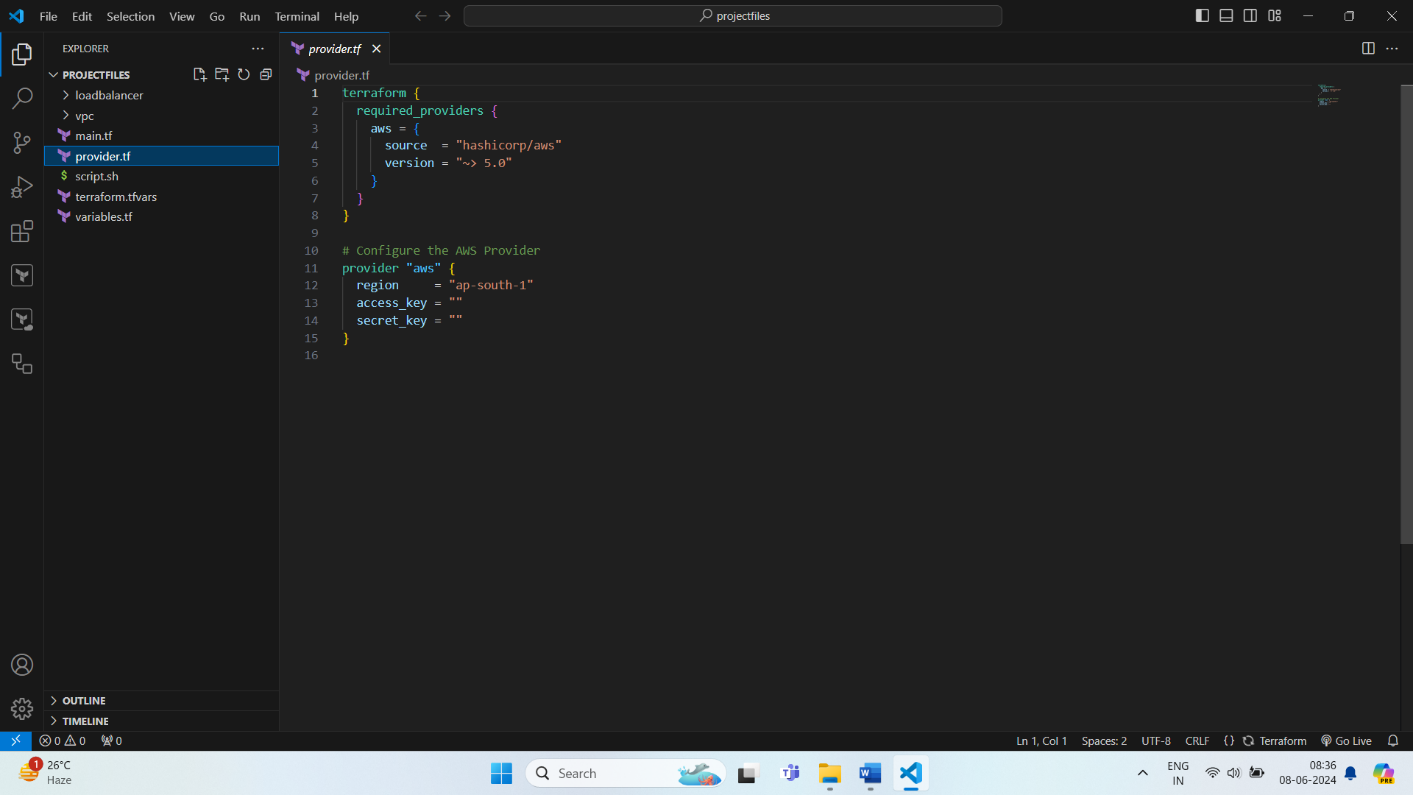
* If we didn’t give the environment variables then every folder we write code should contain Terraform.exe file.
* Then we open the VS Code and the which is in this repository.
* Terraform mainly uses three commands :

1. terraform init – used to install the required pluggins

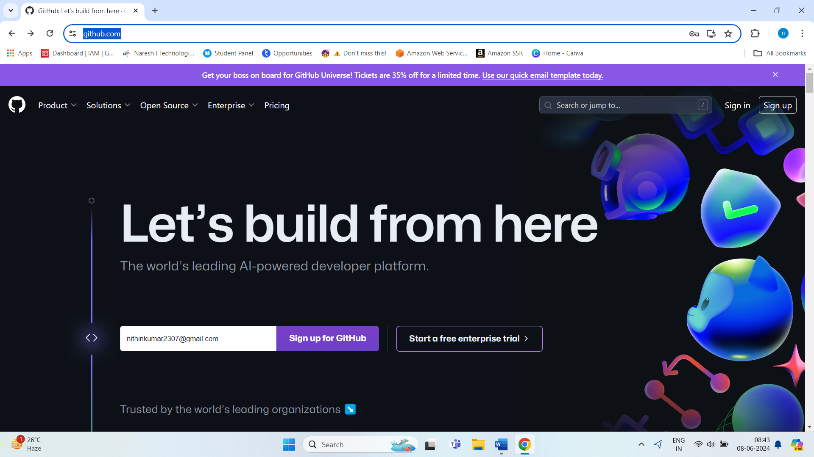
2. terraform plan – used to show the which resources are created, changed or derstroyed.

3. terraform apply- used to deploy the resources

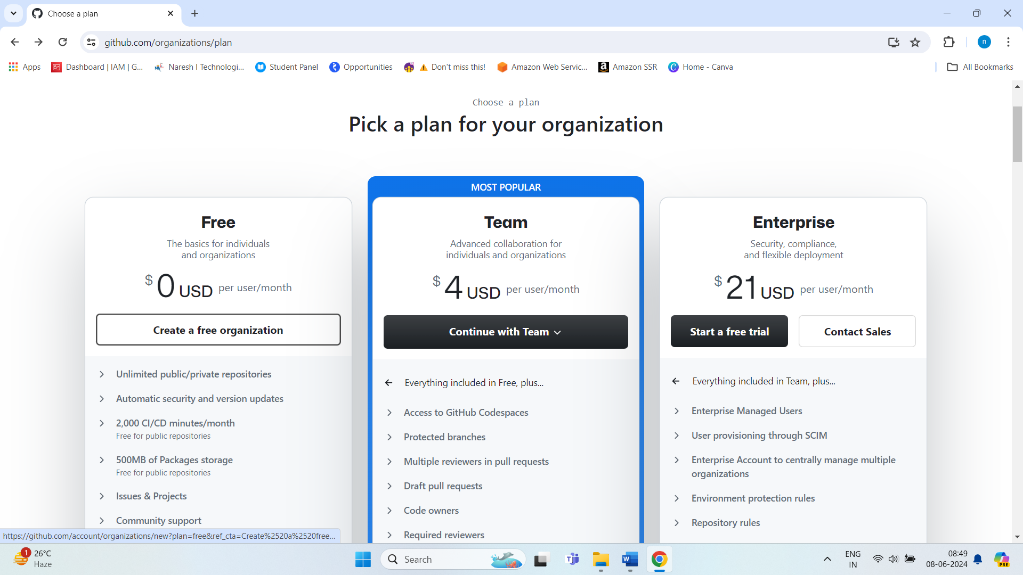
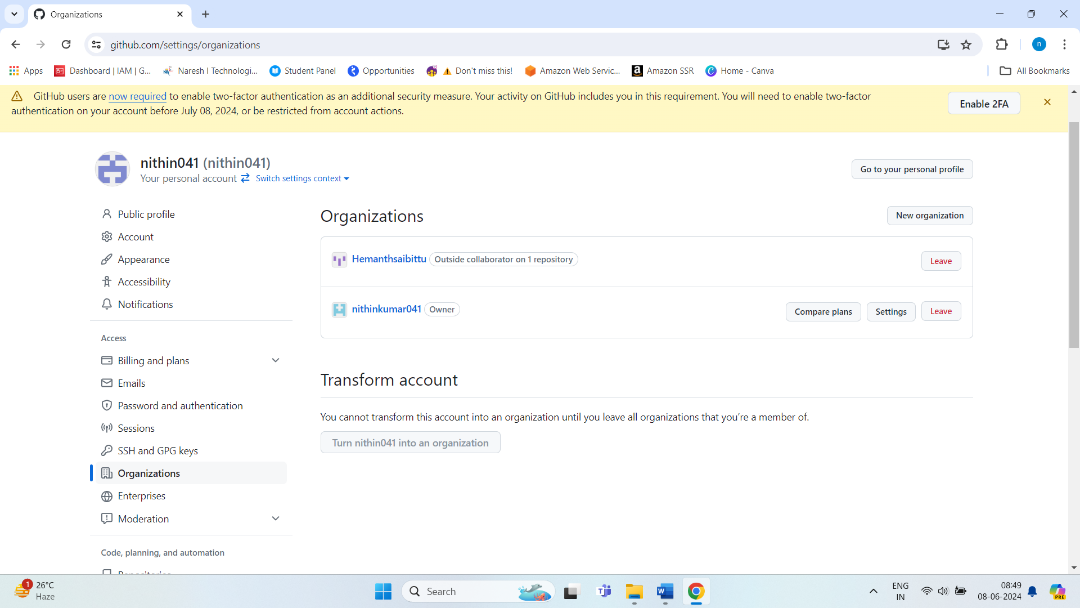
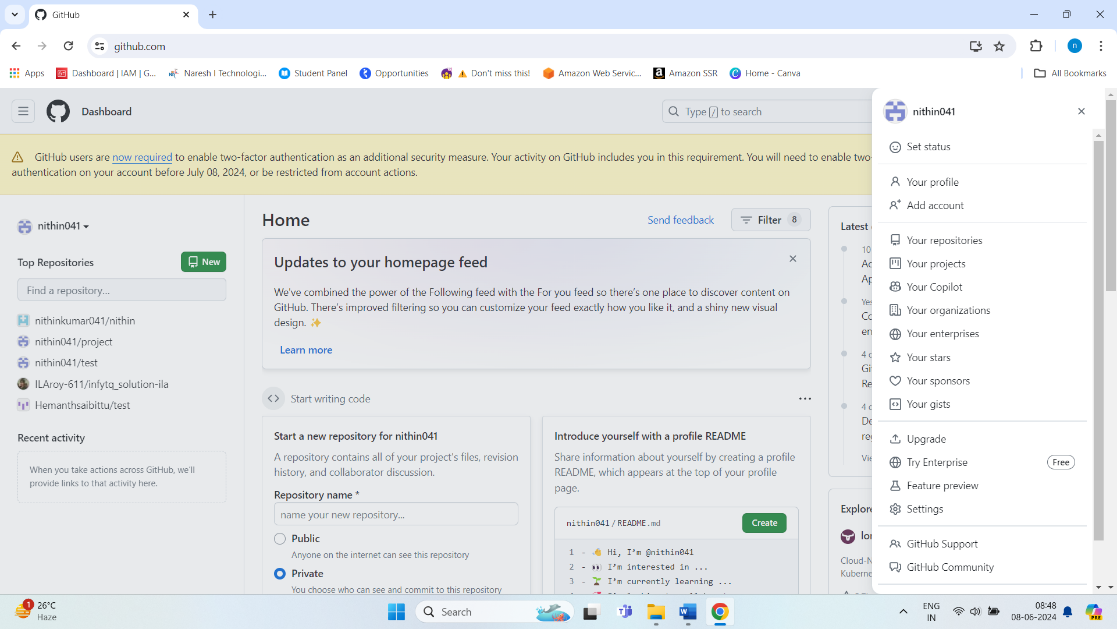
* For Authentication of AWS to Terraform we use Access keys and Secret Keys in a file called provider.tf

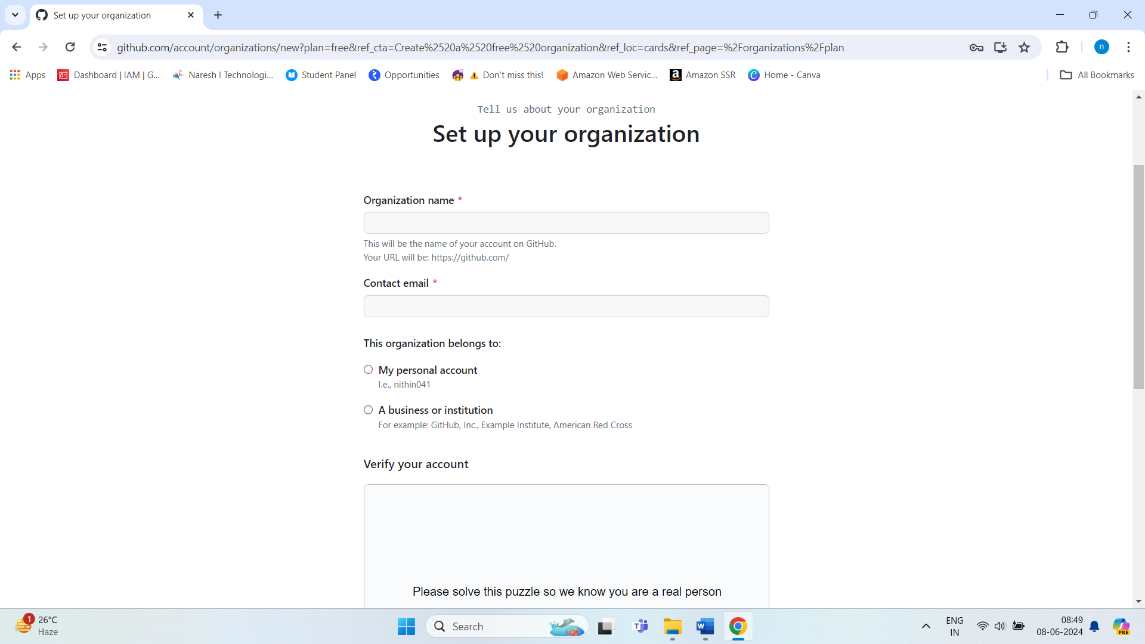


* For any code to push into the GitHub shouldn,t contain any secrets and large file.
* Before Pushing we need to delete all the terraform file.
* So now we see how to create a repository
* To create a repository we should first have a GitHub Account
* So for that we should go to a website called <https://github.com/>

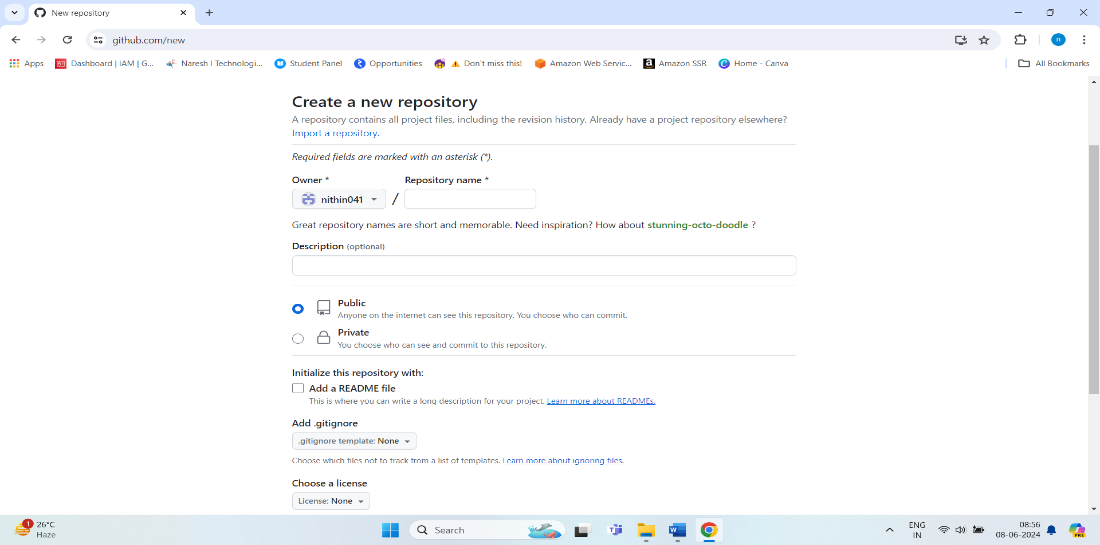
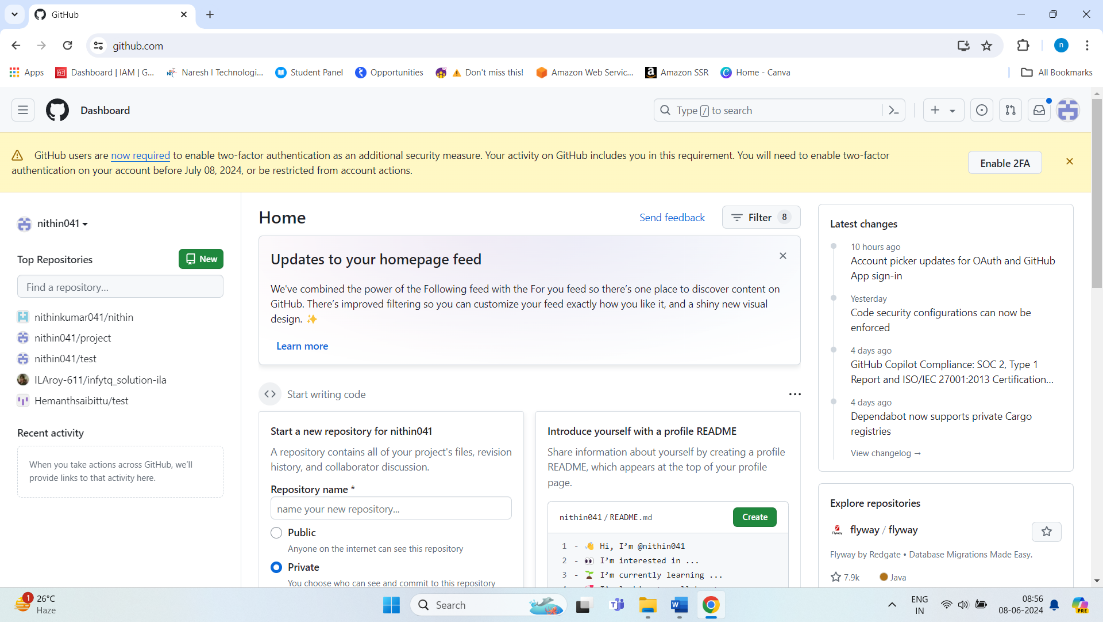


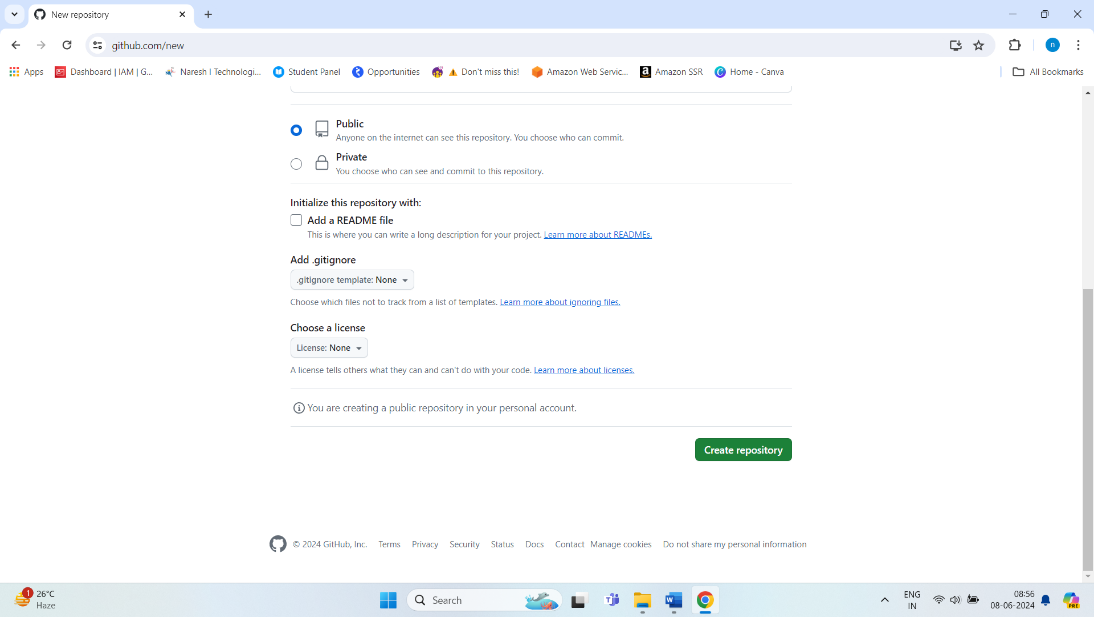
* Then we Click Sign up
* Enter the details and create an account
* To create a repository we should create organization

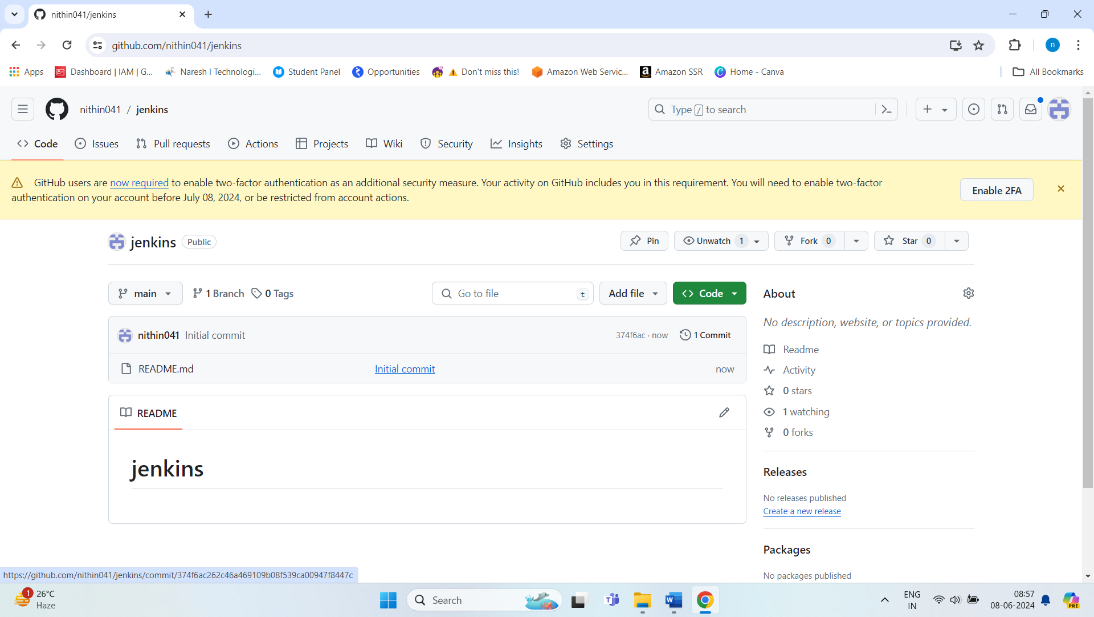




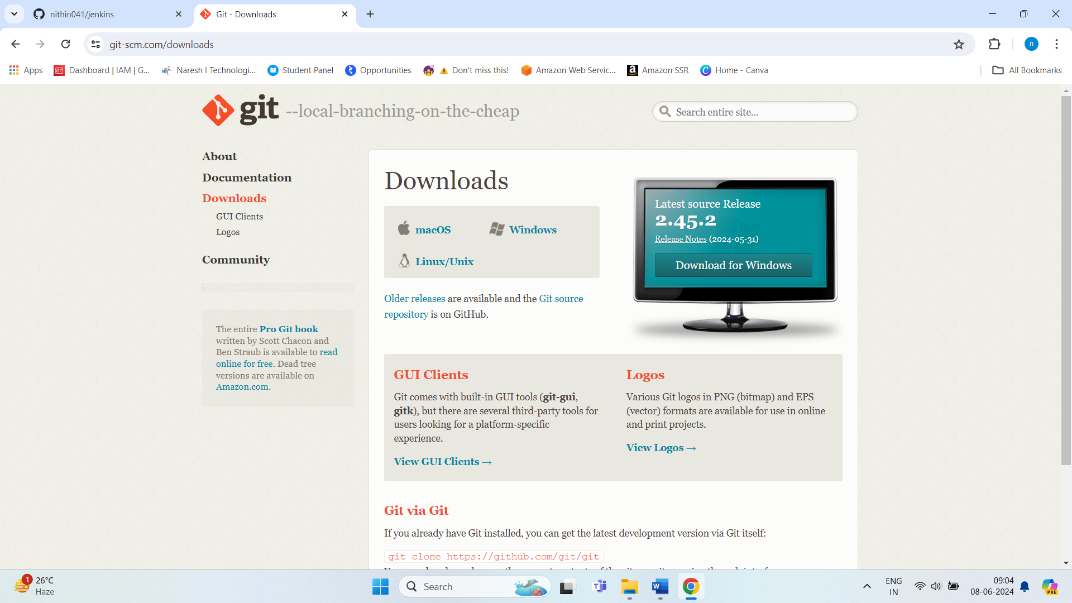
* The above are the steps to create an organisation.
* In that organisation we create a repository
* So below are the steps to create a repository



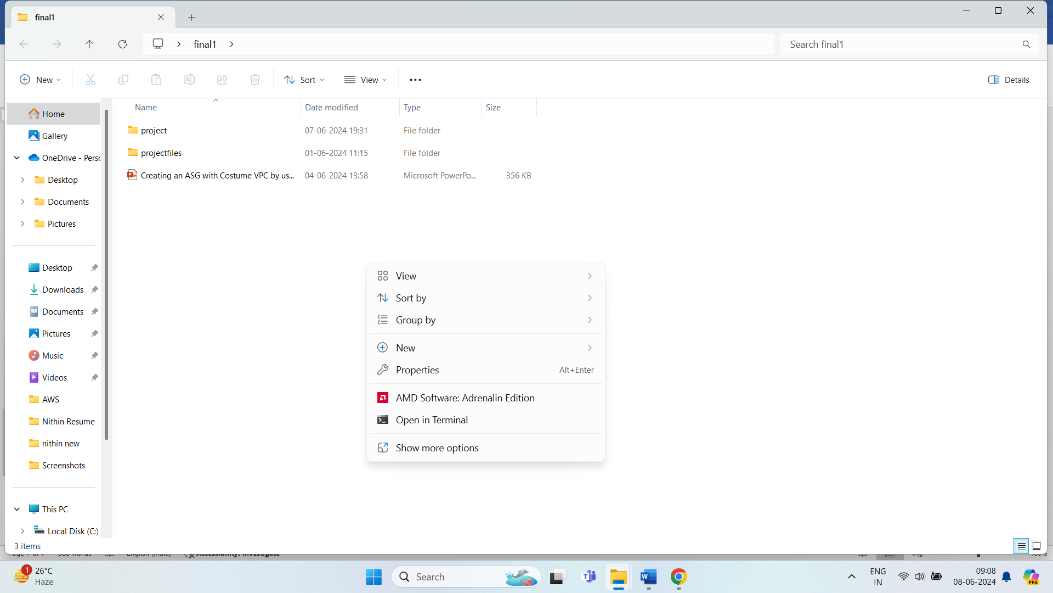


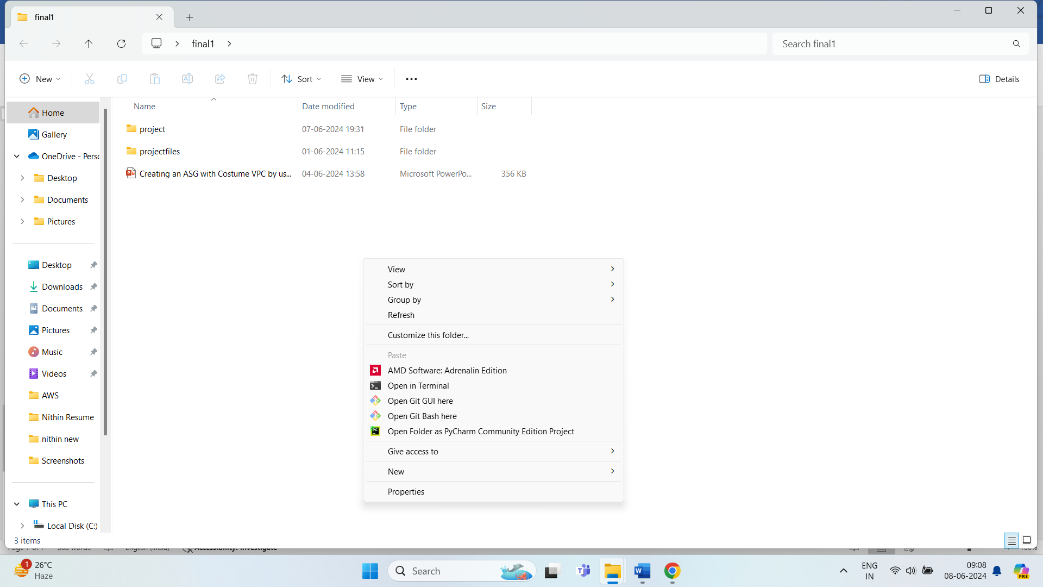


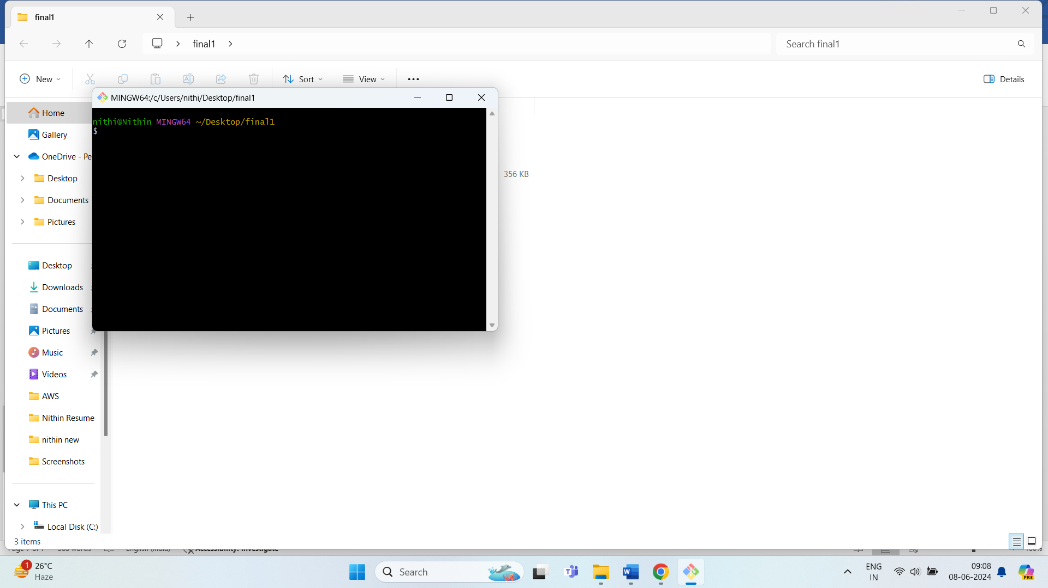
* So after creating the repository we should install GitBash to push our code to repository
* The download link of GitBash for windows is <https://www.git-scm.com/downloads>



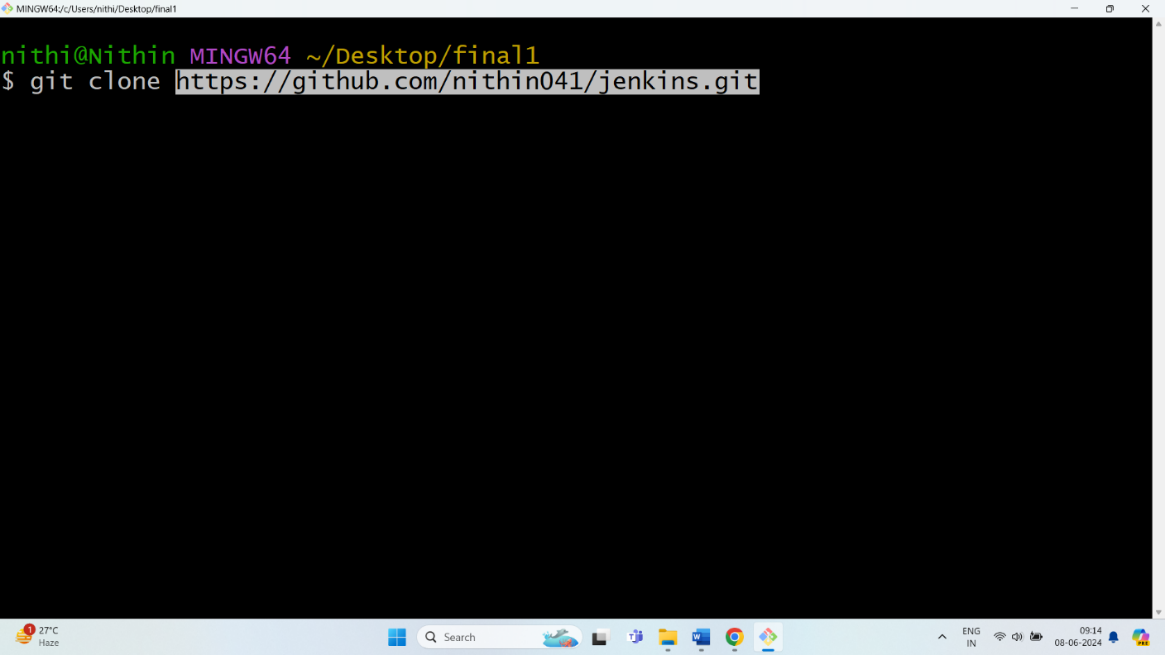
**GitBash**

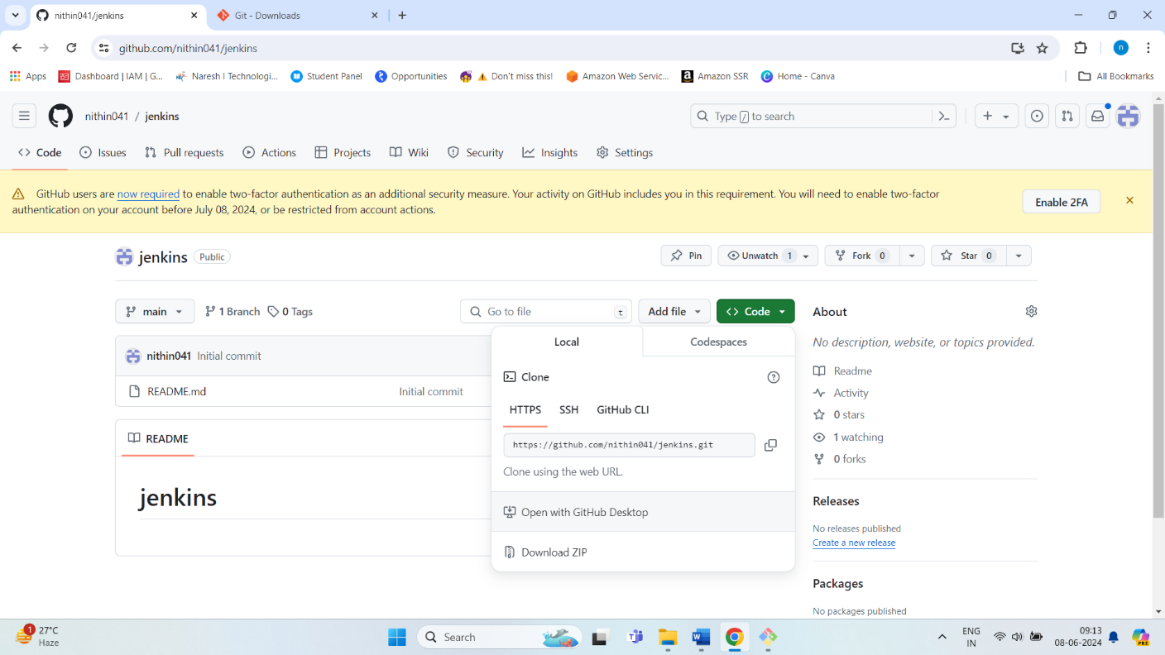
* The Next step is to open the GitBash.
* So to open GitBash first we should go to the folder where our code is present
* Then right click there, click show more options and the click open GitBash Here.



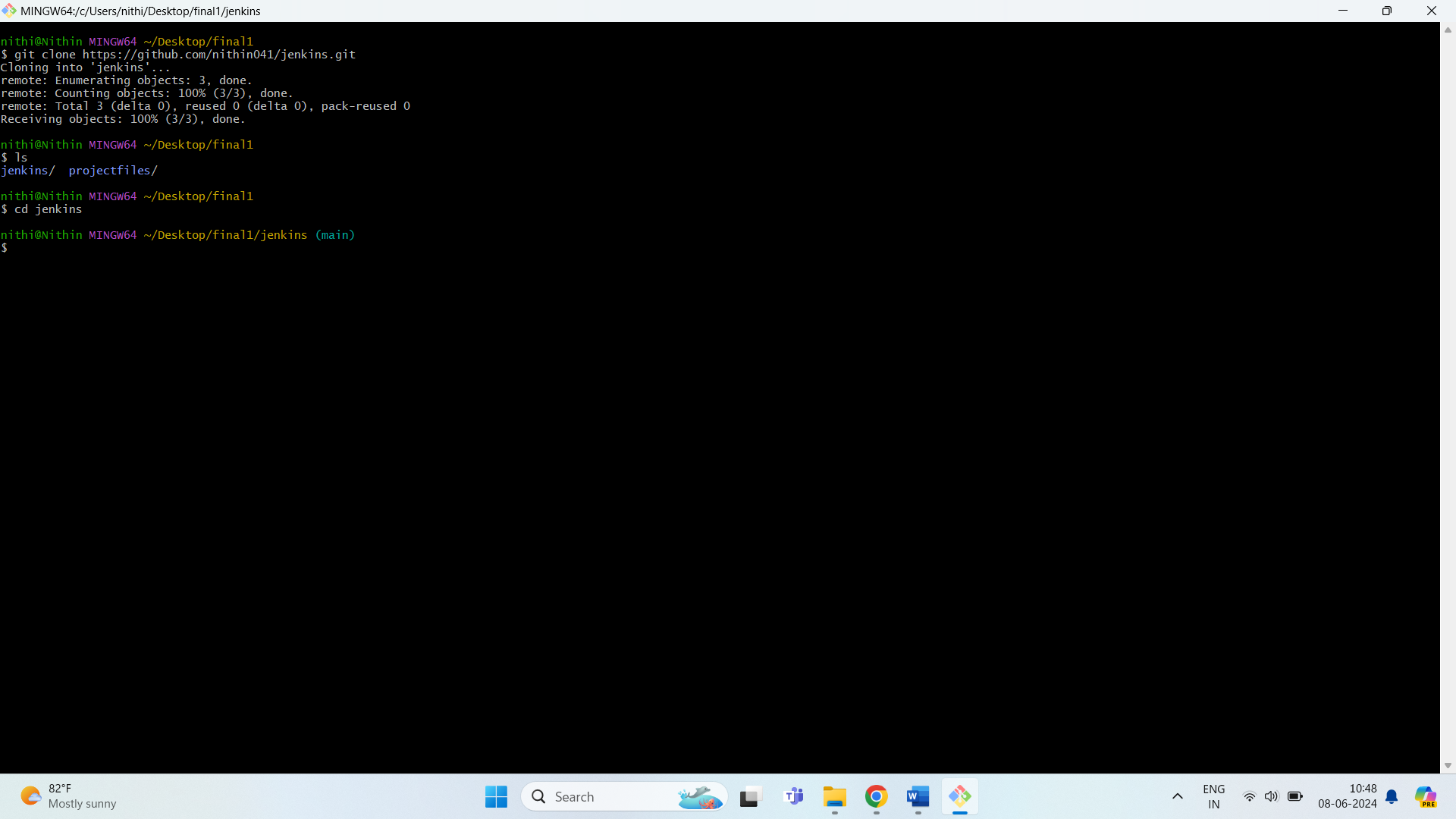


* This is how the GitBash terminal is opened.
* Next we should clone the GitHub Repository.
* To clone the repository we use the command “git clone <repository\_link>”





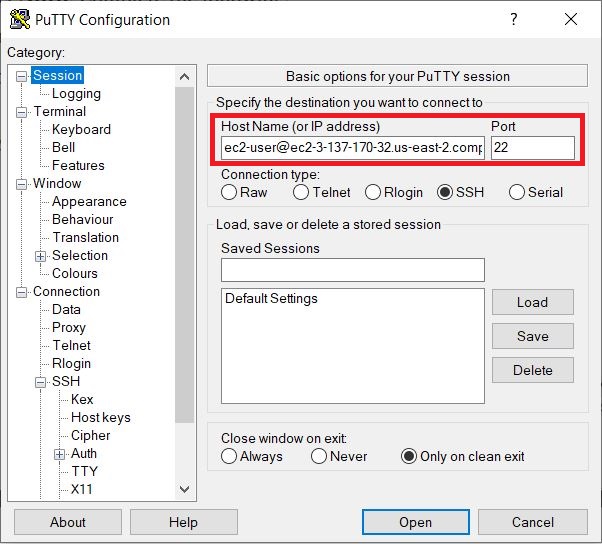
* After cloning we the repository into our folder and should run the commands as follows



* After running the above command copy the folder to the repository manually and run the below commands
* git config user.name “username”
* git config user.email “your GitHub email”
* git add .
* git commit -m “update”
* git push
* After running these commands the code will be pushed to the GitHub and we see the code in our repository.

**Jenkins Server**

* After all this process complete then we should create the Jenkins server.
* For this server we launch an EC2 instance in the our AWS with t2.medium instance type and Amazon Linux OS with Security Group allowing SSH
* After this we should login to the Instance using putty.exe



### Downloading and installing Jenkins

1.Ensure that your software packages are up to date on your instance by using the following command to perform a quick software update:

[ec2-user ~]$ sudo yum update –y

2.Add the Jenkins repo using the following command:

[ec2-user ~]$ sudo wget -O /etc/yum.repos.d/jenkins.repo \

<https://pkg.jenkins.io/redhat-stable/jenkins.repo>

3.Import a key file from Jenkins-CI to enable installation from the package:

[ec2-user ~]$ sudo rpm --import <https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key>

[ec2-user ~]$ sudo yum upgrade

4. Install Java (Amazon Linux 2023):

[ec2-user ~]$ sudo dnf install java-17-amazon-corretto -y

5.Install Jenkins:

[ec2-user ~]$ sudo yum install jenkins -y

Enable the Jenkins service to start at boot:

[ec2-user ~]$ sudo systemctl enable jenkins

Start Jenkins as a service:

[ec2-user ~]$ sudo systemctl start jenkins

You can check the status of the Jenkins service using the command:

[ec2-user ~]$ sudo systemctl status Jenkins

**Installing Terraform on Amazon Linux**

Install yum-config-manager to manage your repositories.

sudo yum install -y yum-utils

2. Use yum-config-manager to add the official HashiCorp Linux repository.

sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo

3. Install.

sudo yum -y install terraform

4. Version

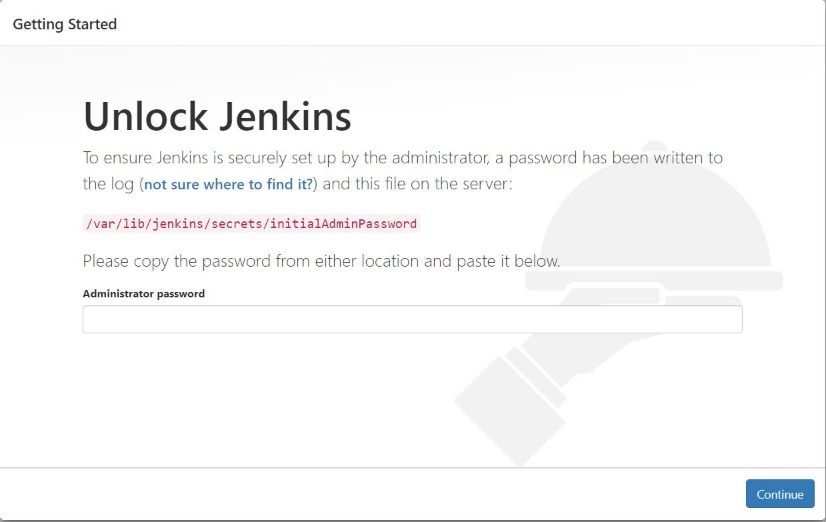
terraform –version

* And also we should install git in the instance the command is “yum install git”

**Configuring Jenkins**

Jenkins is now installed and running on your EC2 instance. To configure Jenkins:

Connect to http://<your\_server\_public\_DNS>:8080 from your browser. You will be able to access Jenkins through its management interface:



1. As prompted, enter the password found in **/var/lib/jenkins/secrets/initialAdminPassword**.

Use the following command to display this password:

**[**ec2-user ~]$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword

1. The Jenkins installation script directs you to the **Customize Jenkins page**. Click **Install suggested plugins**.
2. Once the installation is complete, the **Create First Admin User** will open. Enter your information, and then select **Save and Continue**.



1. On the left-hand side, select **Manage Jenkins**, and then select **Manage Plugins**.
2. Select the **Available** tab, and then enter **Terraform plugin** at the top right.
3. Select the checkbox next to **Amazon EC2 plugin**, and then select **Install without restart**.
4. Next go to credentials add the GitHub Credentials
5. For authentication of Jenkins to AWS we create a role with admin access and attach it to the Jenkins EC2 Instance.
6. Then return to dashboard and click on the new item to create the pipeline.
7. After clicking new item give some name and click pipeline and click ok
8. Next give the description and in the source select the GitHub project and the give the repository link
9. Then scroll down we find a space for writing script
10. The script will be available in this repository.
11. Only just change the repository link and directory name.
12. Then Click Save.
13. After All these open the pipeline and in the right side we find **build now** option
14. Then all the resources will be deployed in the AWS.