**Experiment**

**Working with Git and GitHub**

1. Git Installation
2. Creating a GitHub account
3. Git Commands - Working with Local Repositories
4. Working with Local and Remote Git Repositories

**AIM**

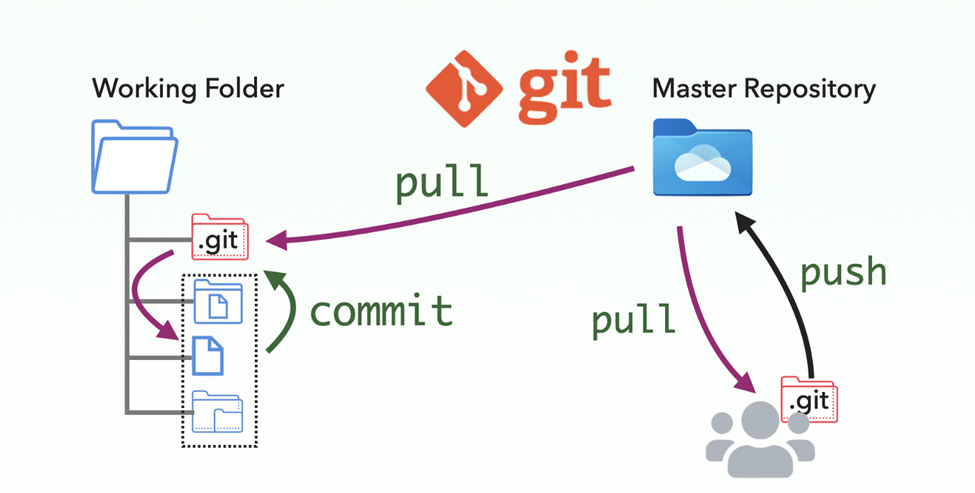
To experience the version control system, Tracking code changes, and code collaboration.

**Introduction:**

Git is a popular version control system. It was created by Linus Torvalds in 2005. It is designed to make it easier to have multiple versions of a code base, sometimes across multiple developers or teams

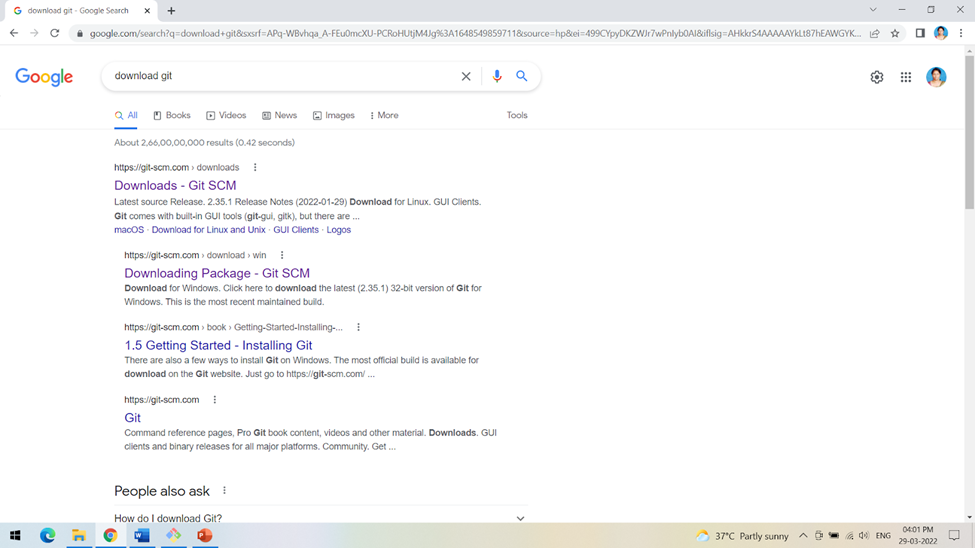
**Functions of Git**

* Manage projects with Repositories
* Clone a project to work on a local copy
* Control and track changes with Staging and Committing
* Branch and Merge to allow for work on different parts and versions of a project
* Pull the latest version of the project to a local copy
* Push local updates to the main project



1. **Git Installation on Windows:**

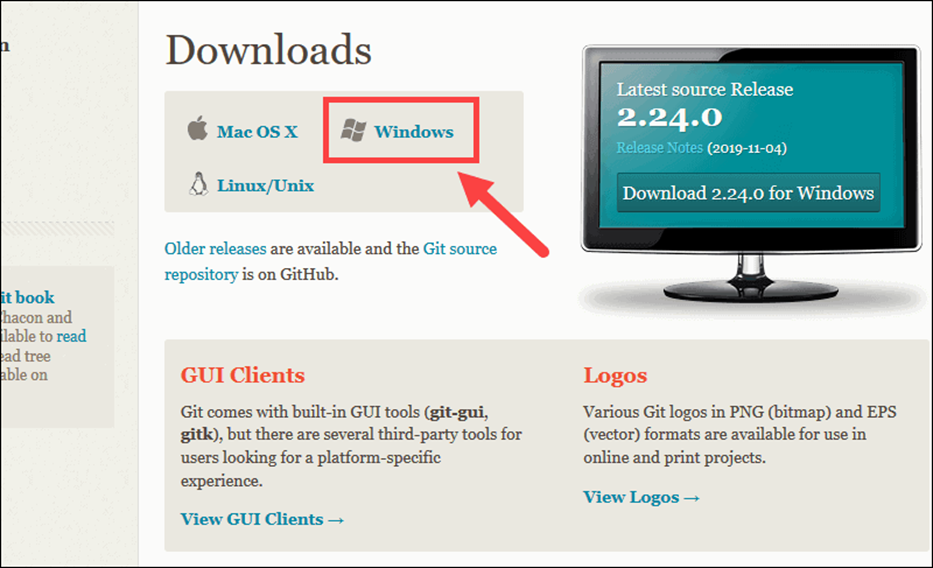
* Step 1: Search for **Git download** in google.
* Click on **Downloads - Git SCM**



**Step 2:**

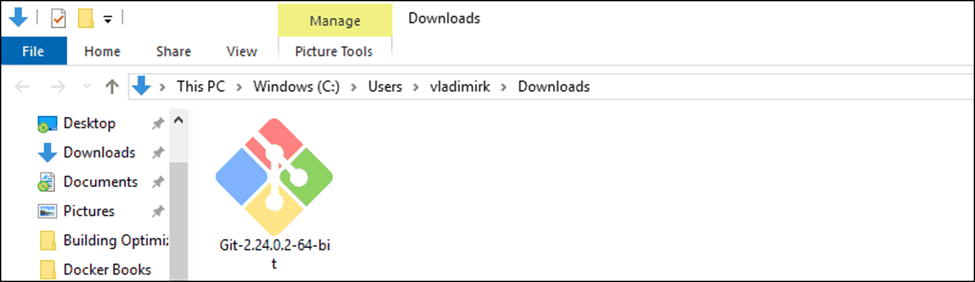
Browse to the official Git website: <https://git-scm.com/downloads>

Click the download link for Windows and allow the download to complete



**Step 3:**

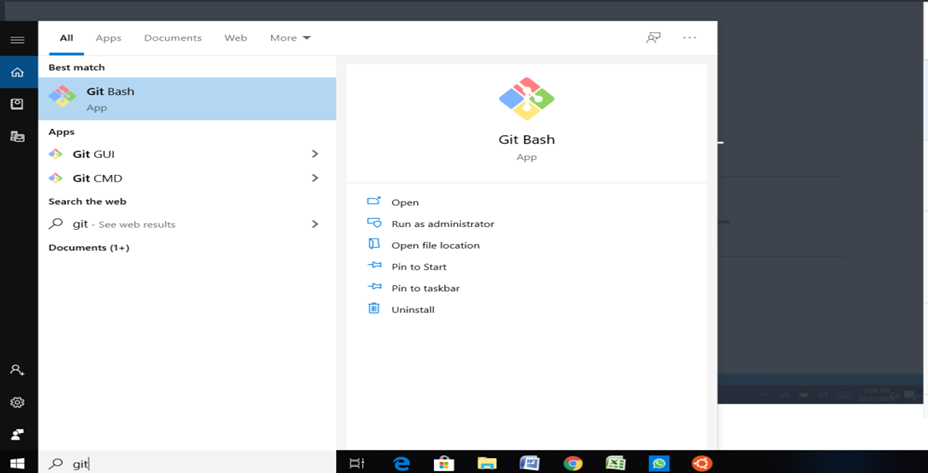
Double click the git-installer to install git on your system by following the instructions.



**Step 4:**

In all the steps leave the defaults unless you have a specific need to change them and click Next.

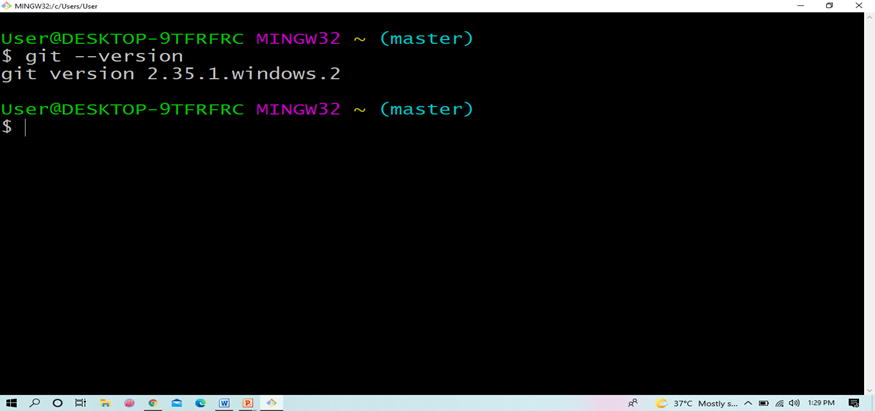
Finally, Git is installed and Git Bash is in the interface to Git.



Step 5: Ensure git is installed. Click on Git Bash to start the command interface.

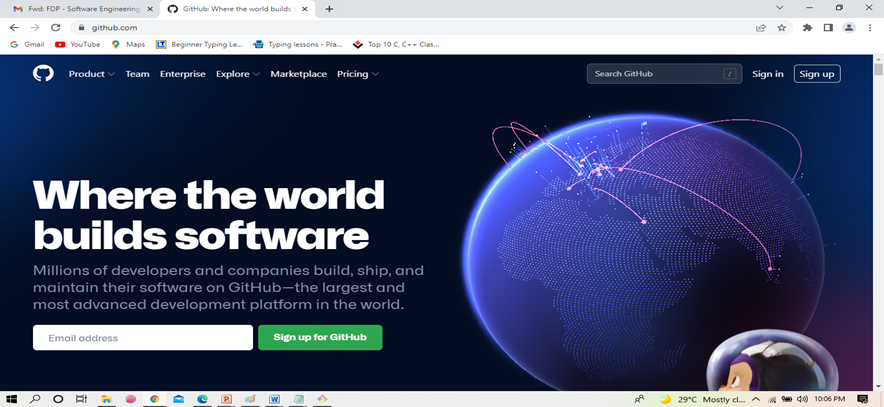
$ git --version

Which displays the installed version of git as follows

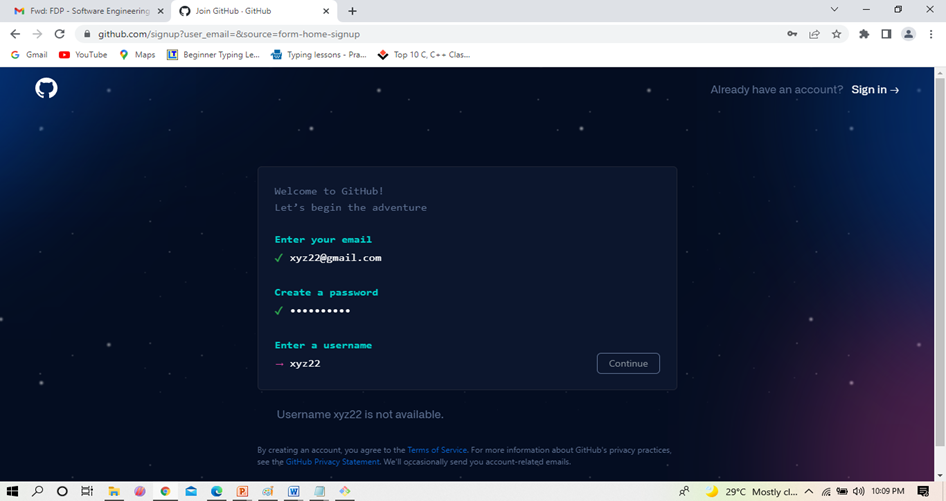
****

1. Creating a Git Hub Account

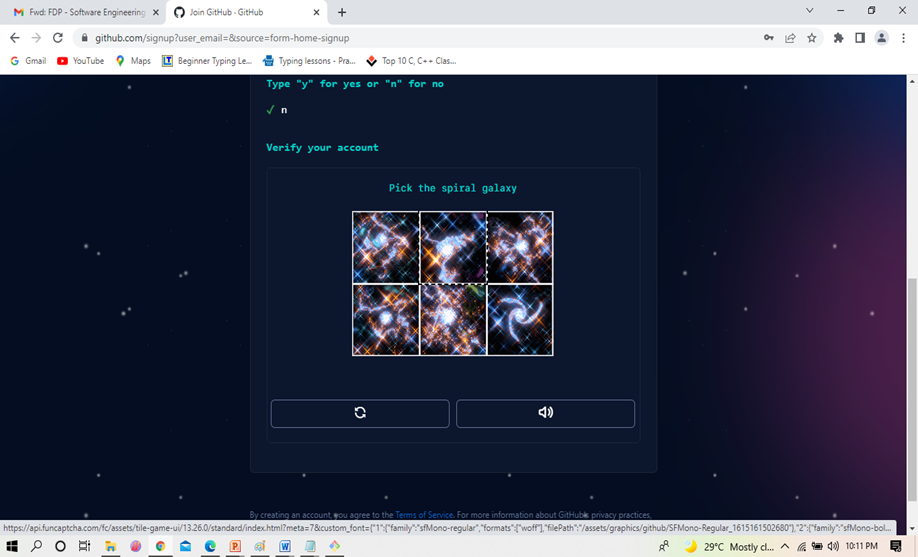
Step 1: Browse the website [https://github.com](https://github.com/) and click on Signup



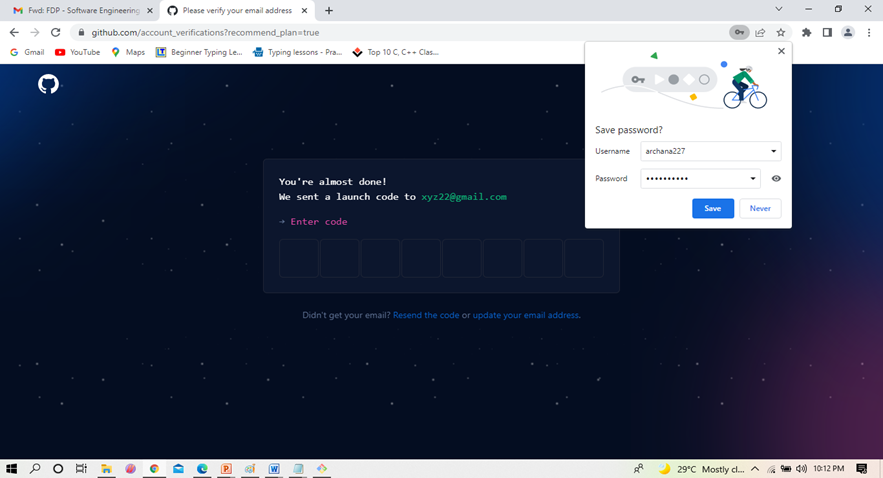
Step 2 : Enter your email address with which you want to access github and also create a password for the same.



Step 3 : Complete the verification process by selecting the spiral galaxy in the given set of images.

****

Step 4: Complete the account creation by entering the launch code that is sent to your email address as a part of confirmation.

****

1. **Working with Git Commands: Local Repository**

**Introduce yourself to Git:**

For the first time when you start with Git you need to enter the username and email address with which the GitHub account is linked. This is to access the remote repository from your local repository.

• git config --global user.name “USER NAME"

• git config --global user.email USER@MAILID

•  **mkdir <dirname>** Create a new directory for the project

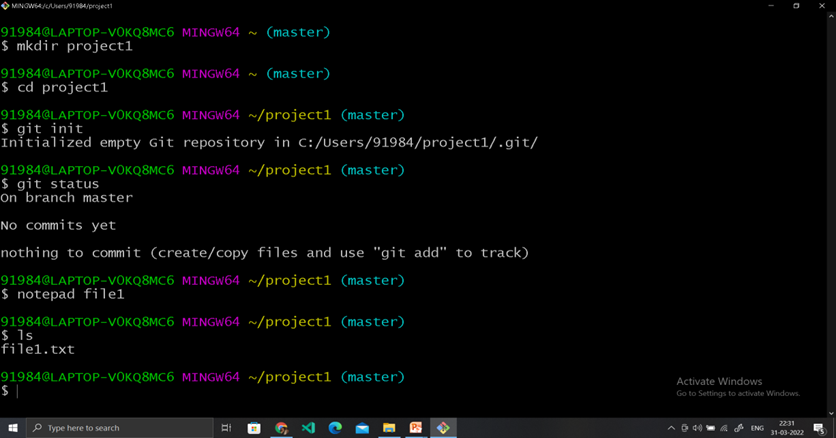
• **cd <dirname>** Change to that directory

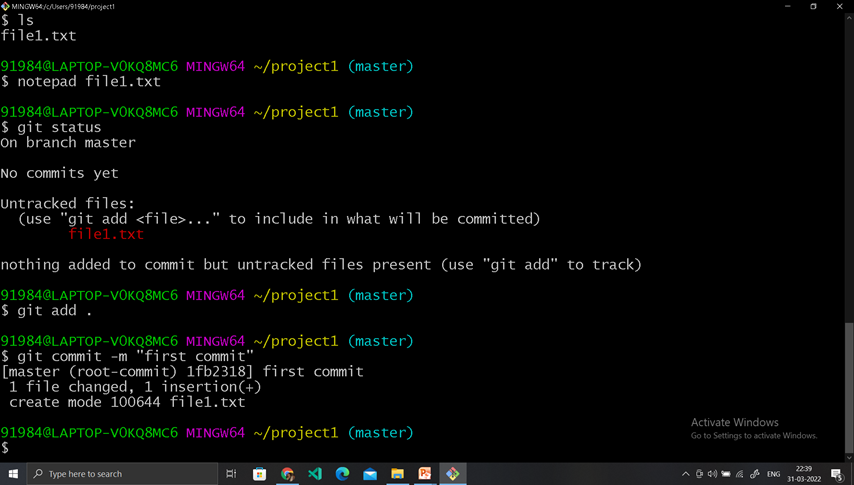
•  **git init:** The git init command is used to create an empty git repository.

•  **git status:** displays the state of the working directory and the staging area

• **git add** . Or git add filename: is used to add all the new or modified files or a specific file to the staging area to be committed further.

•  **git commit -m “message”:**  captures a snapshot of the project's currently staged changes. Committed snapshots can be thought of as “safe” versions of a project—Git will never change them unless you explicitly ask it to.



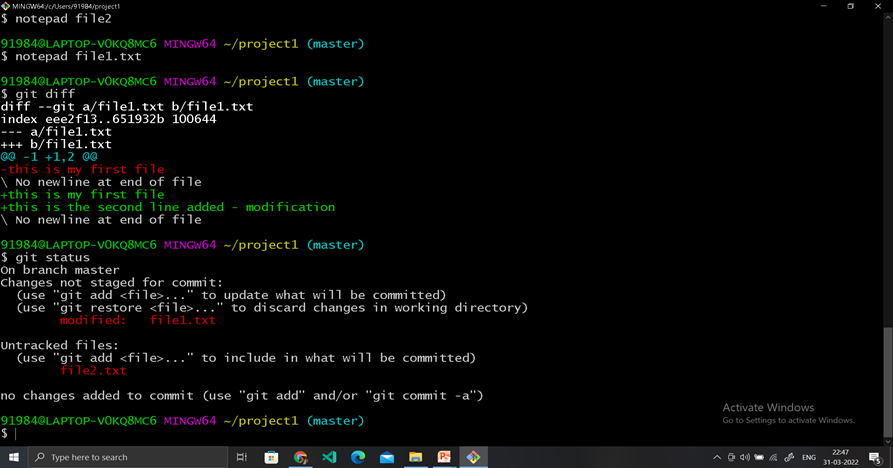


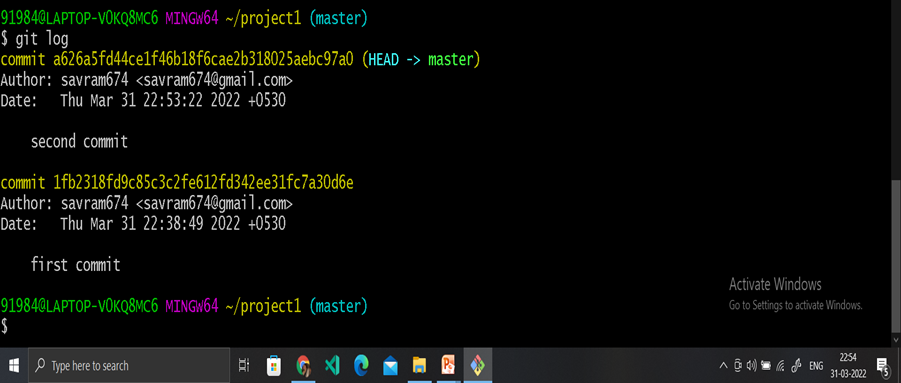
• **git diff** is used to show changes between commits, commit, and working tree, etc.

• **git log** command displays all of the commits in a repository's history. By default, the command displays each commit's: Secure Hash Algorithm (SHA) author. date.

• **git reset** is a command that is used to undo local changes to the state of a Git repo.

• **git log** - showing commit history with hash code for each commit





A branch in Git is simply a lightweight movable pointer to one of these commits.

The default branch name in Git is **master.**

• **git branch <branchname>** It will create a new branch with the given name

• **git checkout <branchname>** command lets you navigate between the branches created by the git branch.

• **git checkout –b <branchname>** will create and checkout to a new branch automatically.

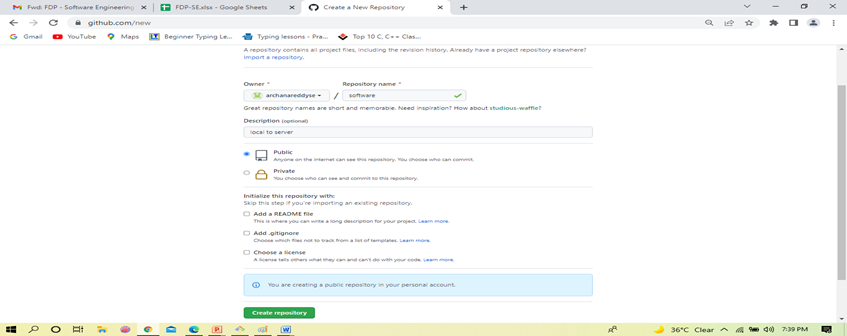
• **git branch –d <branchname>** will delete the branch locally.

• **git merge** is used to merge the sub branch with the main after confirmation on changes.

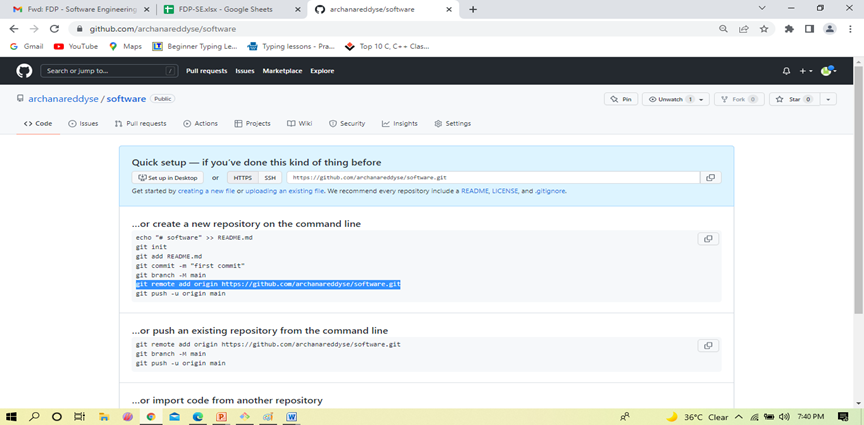
1. **Working with Git Commands: Remote Repository**

• Sign-in to your git account at github.com

• Create a new remote repository to store your files from local space.

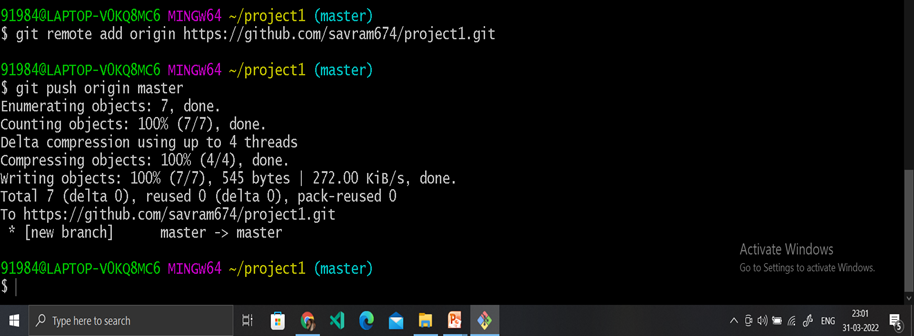


• Use the http link to the remote repository for pushing the files from the local to the remote repository.

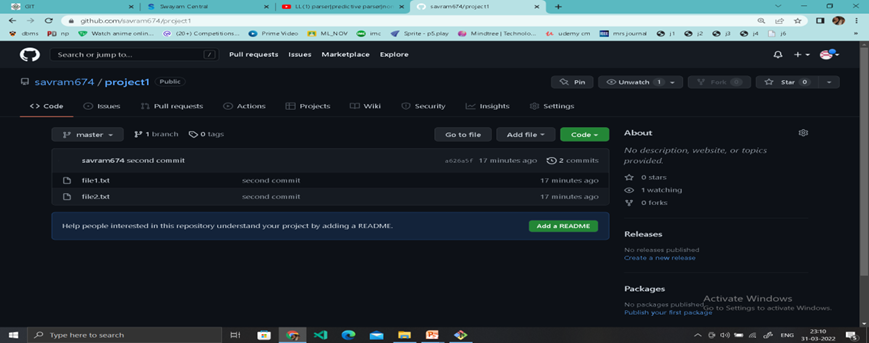


• **git remote add origin <address>** - git remote command is used to create, view and delete connections to other repositories. The connection here is used as book marks that serve as convenient names to be used as a reference.

• **git push –**  the command is used to transfer (push) the committed contents from local repository to remote repository.

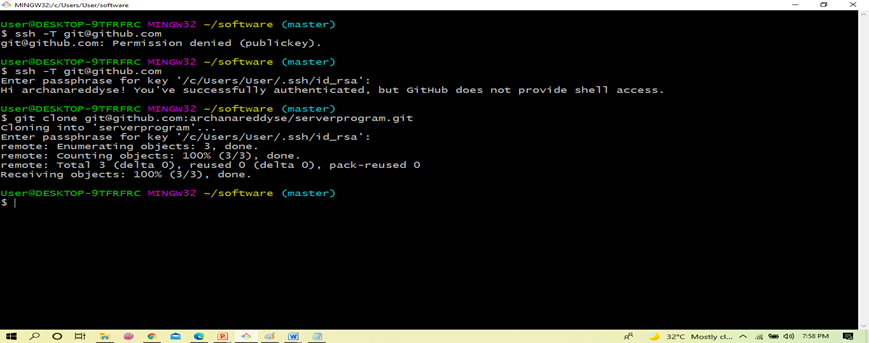


**Refresh the Github account to see the newly pushed contents**



• **git clone <remote URL>** - git is used to creating a local working copy of an existing remote repository. It downloads the remote repository to the local system.

• **git pull <branch name> <remote URL> –**  the command is used to fetch and merge the changes from the remote repository to the local repository.



**Experiment 8:**

Creating static pages of the project and committing using Git and GitHub

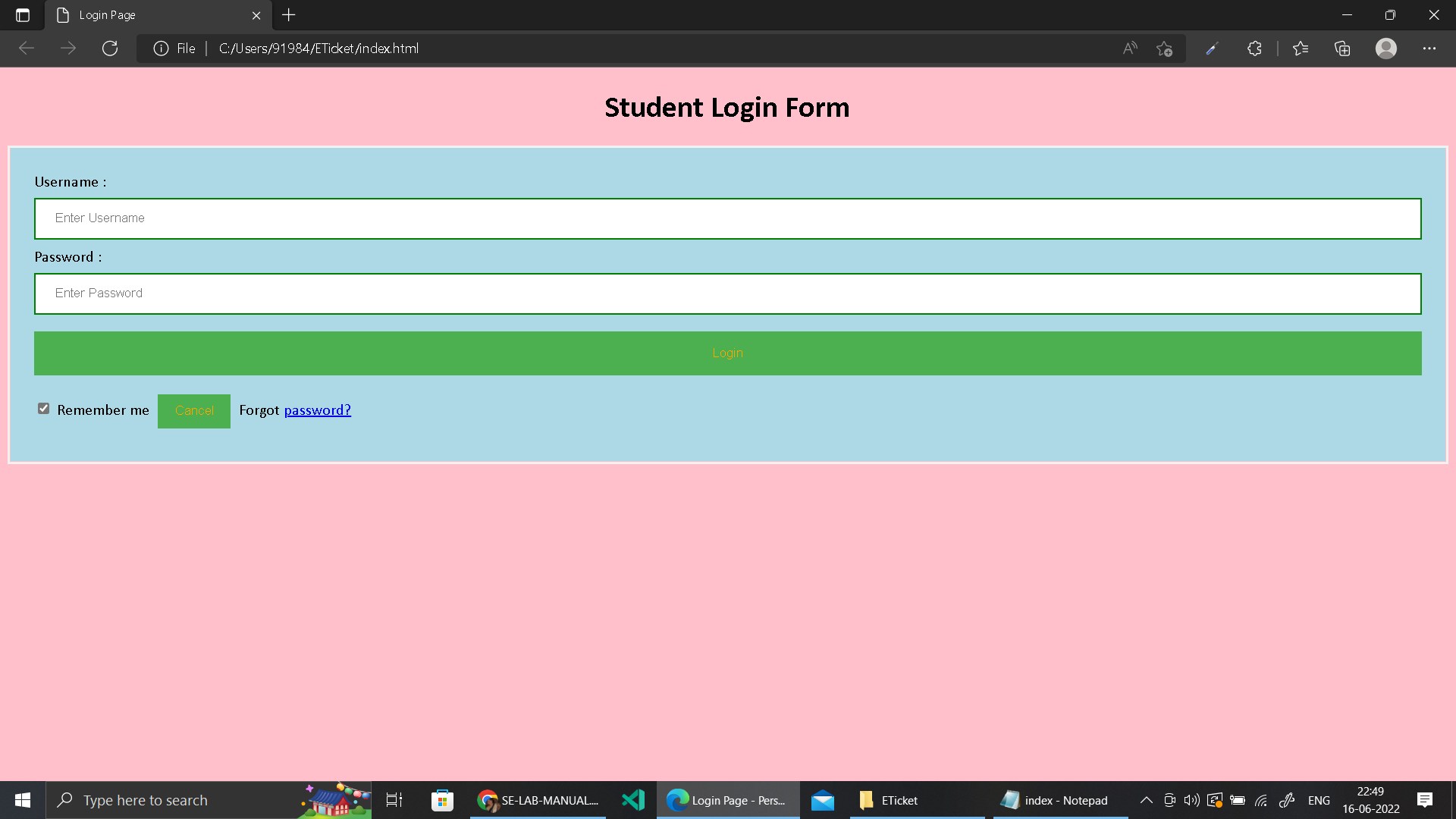
**AIM:** To experience the real-time scenario of collaborative coding. When multiple developers work on the same project all the team members will be provided with reading/write access. This facilitates tracking of the modification made to the code by everyone and maintains different versions of the developed modules.

**Procedure:**

* Create a directory in the local system for working on the project
* Change to that directory
* Create an empty git repository by running - the **git init**  command.



* Create any static page for the project using HTML.
* Here is the sample code for the student login form as shown below.

****

**HTML code for creating the above form.**

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width, initial-scale=1">

<title> Login Page </title>

<style>

Body {

font-family: Calibri, Helvetica, sans-serif;

background-color: pink;

}

button {

background-color: #4CAF50;

width: 100%;

color: orange;

padding: 15px;

margin: 10px 0px;

border: none;

cursor: pointer;

}

form {

border: 3px solid #f1f1f1;

}

input[type=text], input[type=password] {

width: 100%;

margin: 8px 0;

padding: 12px 20px;

display: inline-block;

border: 2px solid green;

box-sizing: border-box;

}

button:hover {

opacity: 0.7;

}

.cancelbtn {

width: auto;

padding: 10px 18px;

margin: 10px 5px;

}

.container {

padding: 25px;

background-color: lightblue;

}

</style>

</head>

<body>

<center> <h1> Student Login Form </h1> </center>

<form>

<div class="container">

<label>Username : </label>

<input type="text" placeholder="Enter Username" name="username" required>

<label>Password : </label>

<input type="password" placeholder="Enter Password" name="password" required>

<button type="submit">Login</button>

<input type="checkbox" checked="checked"> Remember me

<button type="button" class="cancelbtn"> Cancel</button>

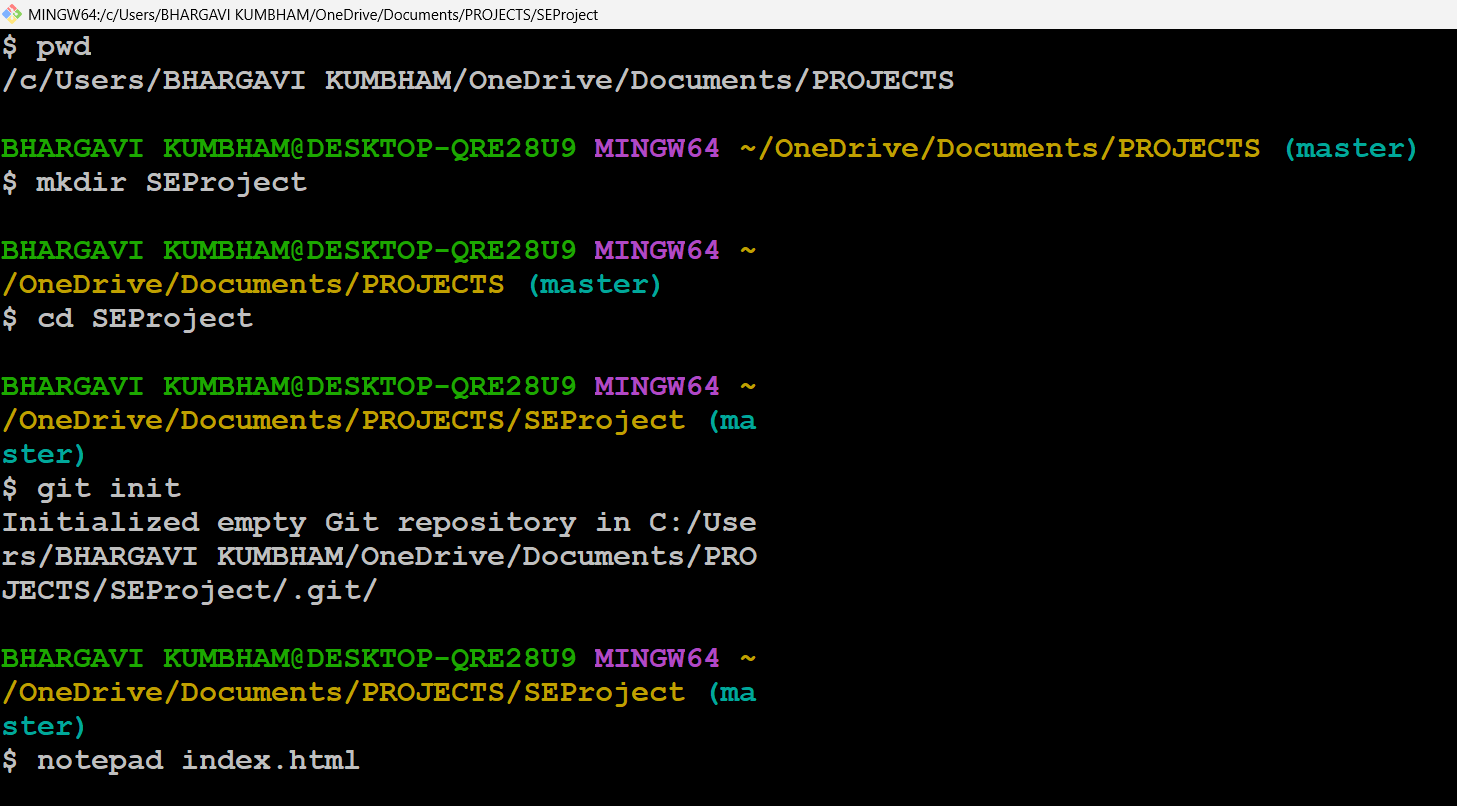
Forgot <a href="#"> password? </a>

</div>

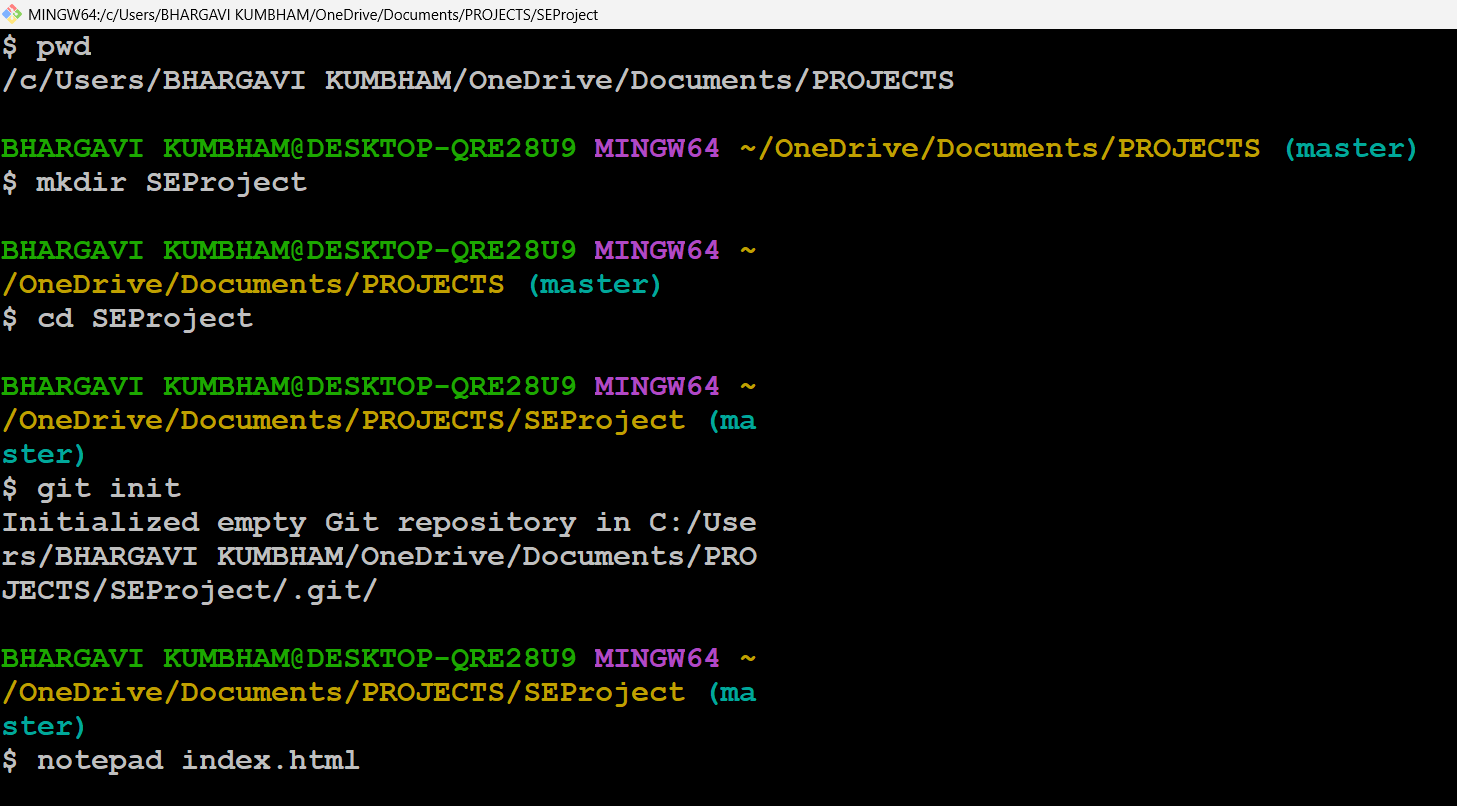
</form>

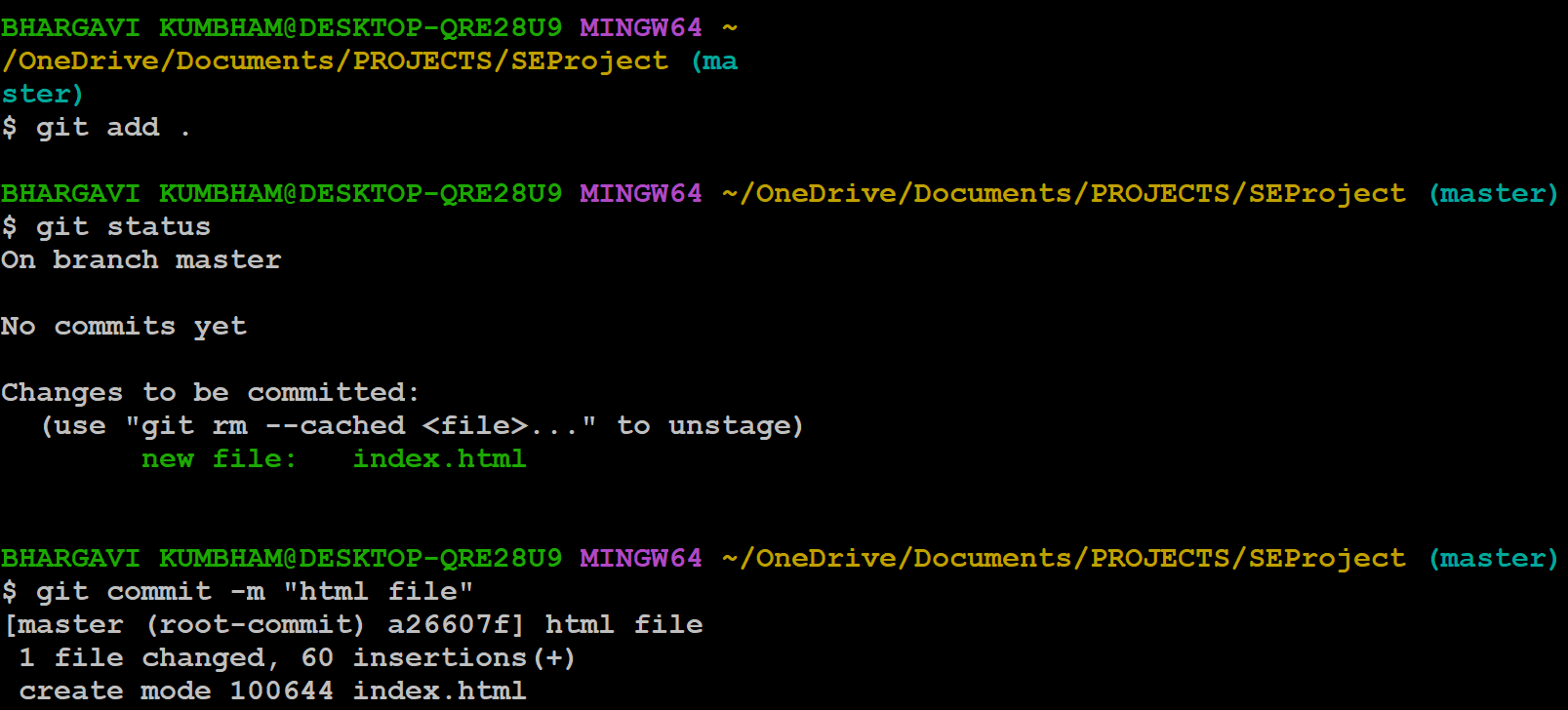
</body>

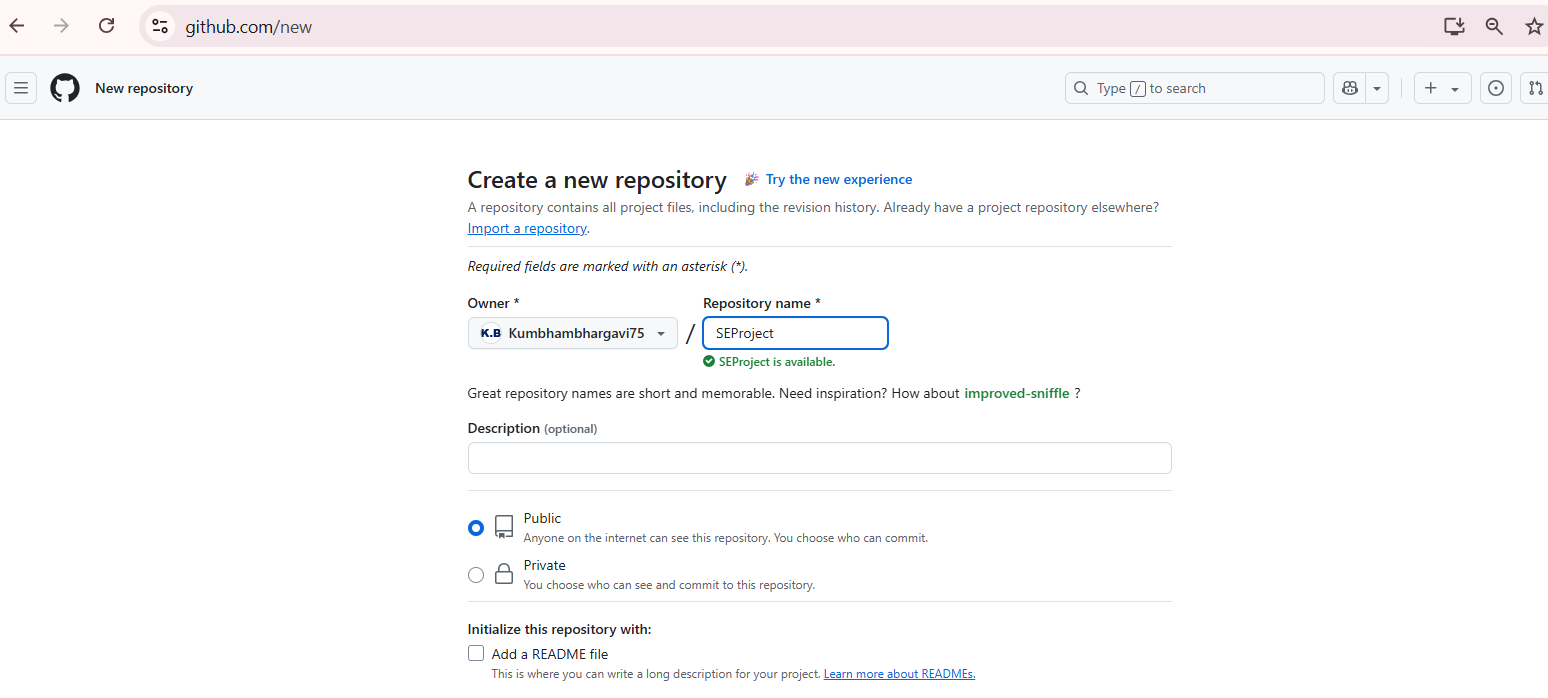
</html>



* Commit the changes made to the new file. You can also view the status and observe the changes before and after commit.





* Create a new repository on GitHub
* 
* Link the remote repository and the local repository using the git remote command

A screenshot of a computer

AI-generated content may be incorrect.

* Push the local repository contents on tp the remote repository using command
  + **git push -u origin master**

****

**A computer screen with colorful text

AI-generated content may be incorrect.**

* Refresh GitHub to check the uploaded content

A screenshot of a computer

AI-generated content may be incorrect.

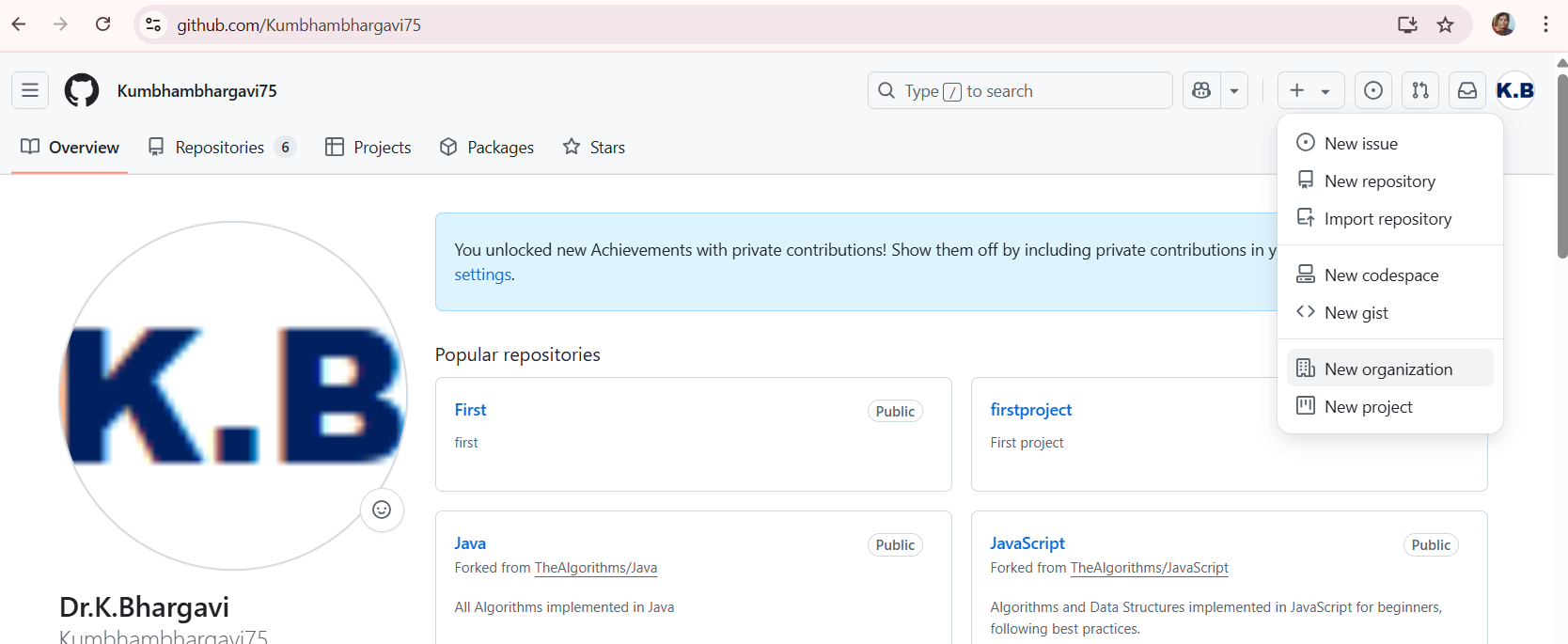
* The co-developers can clone this to their local repository, view, edit, and contribute their inputs.

A screenshot of a computer

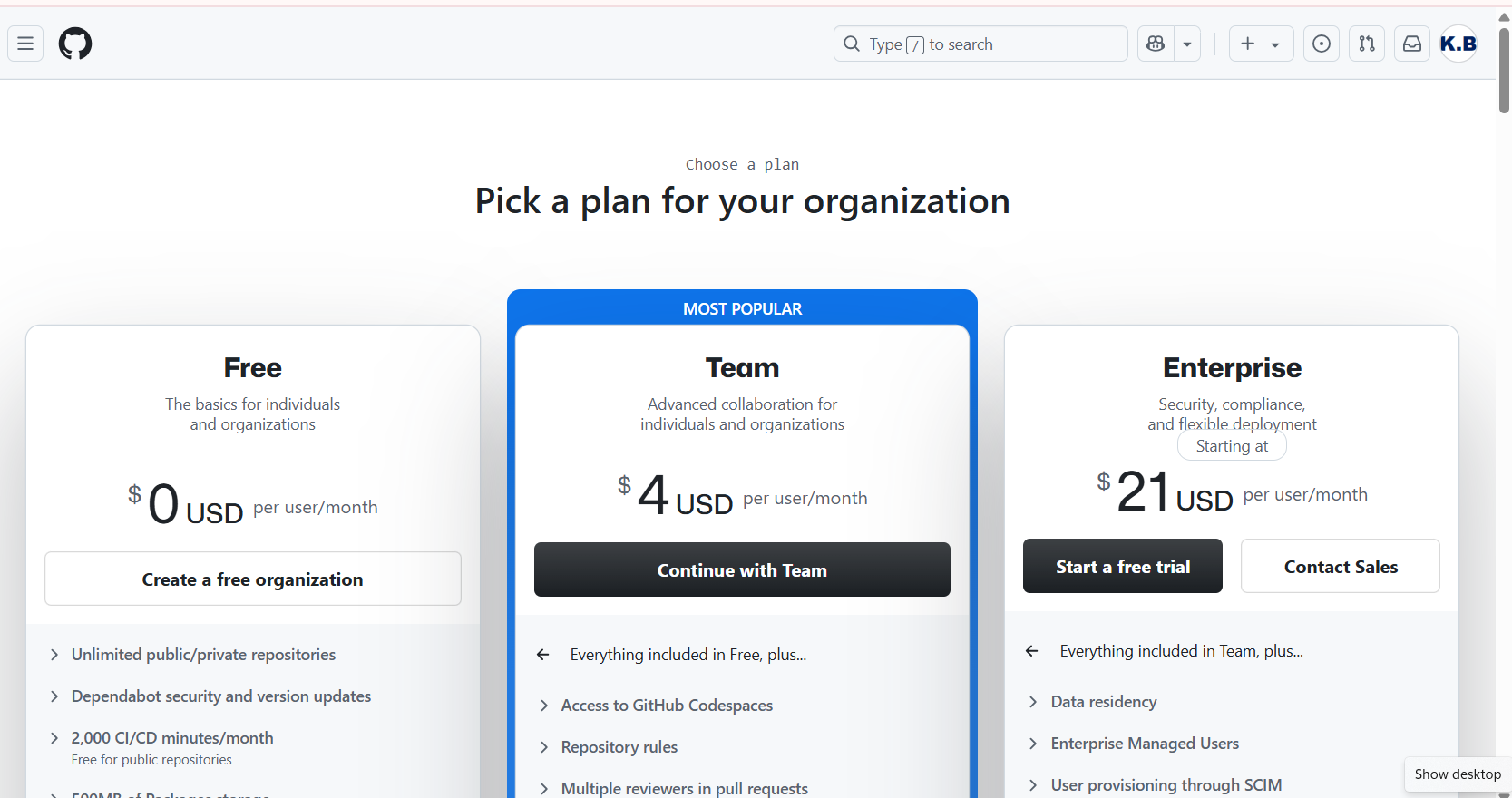
AI-generated content may be incorrect.

**Facilitating Collaborative Work**

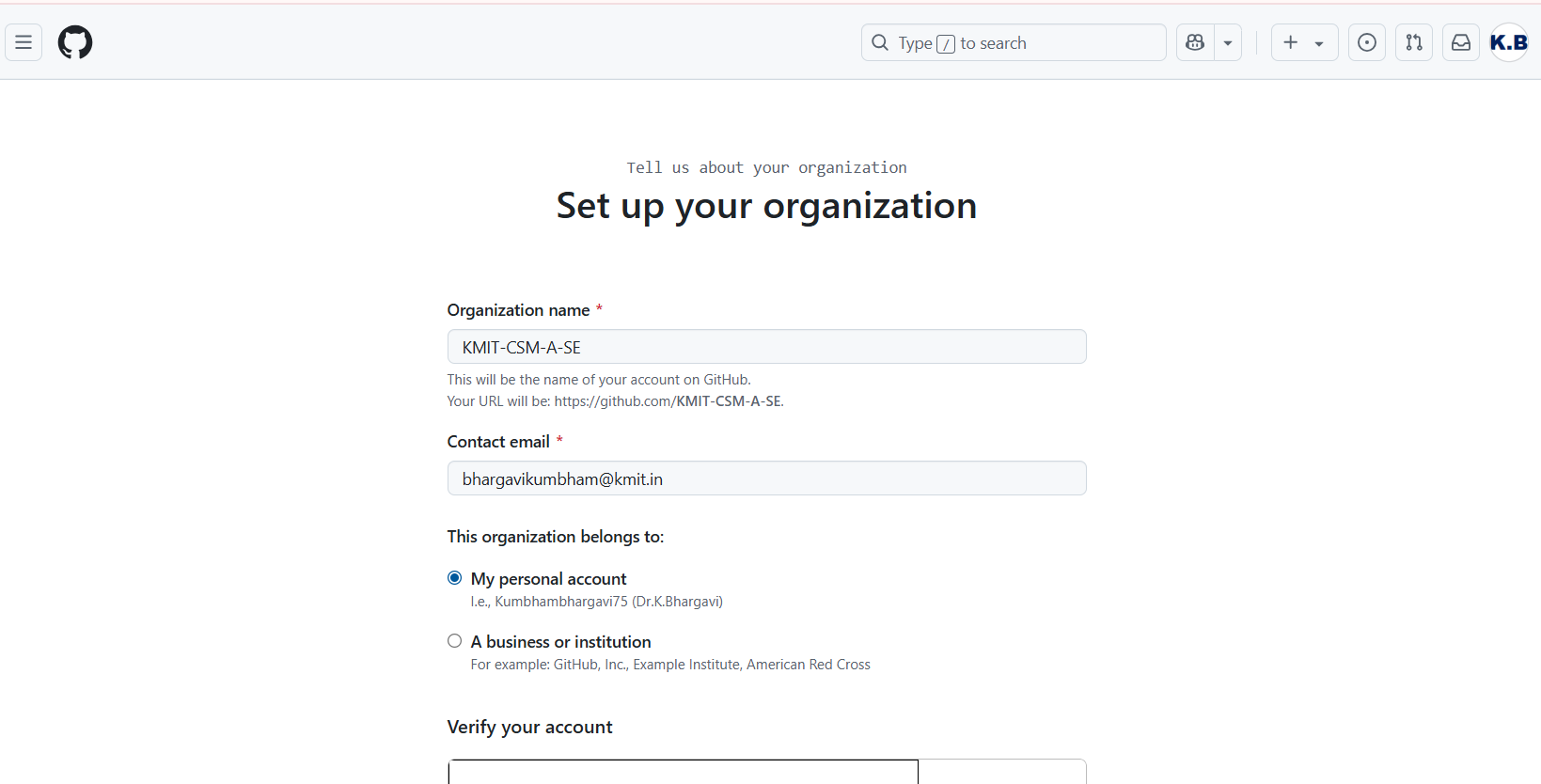
**Step 1:** Create a new Organization - Organizations are shared accounts where businesses and open-source projects can collaborate across many projects at once. Owners and administrators can manage member access to the organization's data and projects with sophisticated security and administrative features.



**Select the free organization**



**Step 2:** Set up your organization by entering the details like name, associated email, account verification, inviting or adding members to the organization, etc.

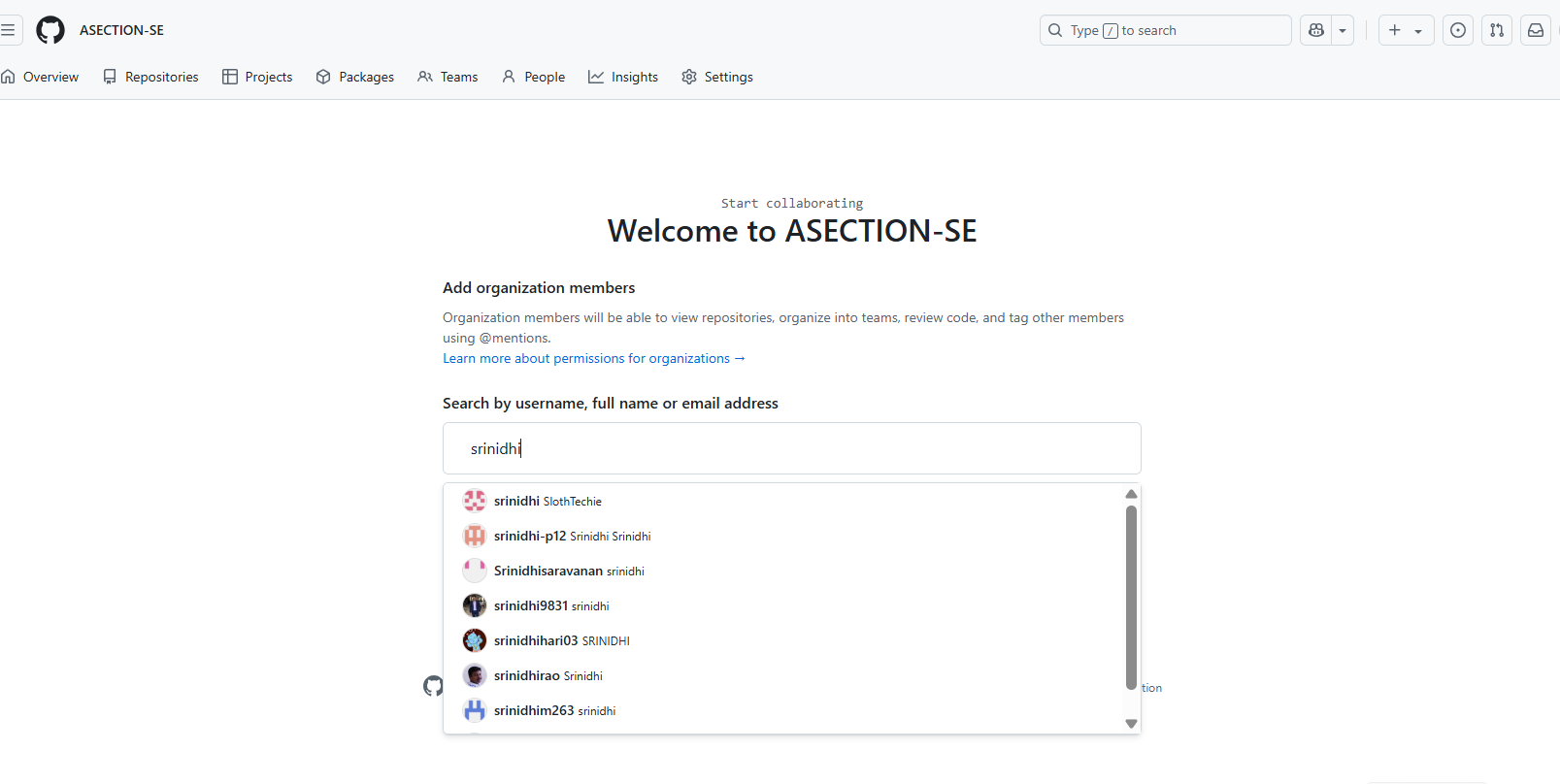


Click on NEXT

A screenshot of a computer

AI-generated content may be incorrect.

**Step 3** : Complete the setup by adding members to the group



A screenshot of a computer

AI-generated content may be incorrect.

Type each student's GitHub username

Click Send Invitation

Students will receive an invite and must accept it.

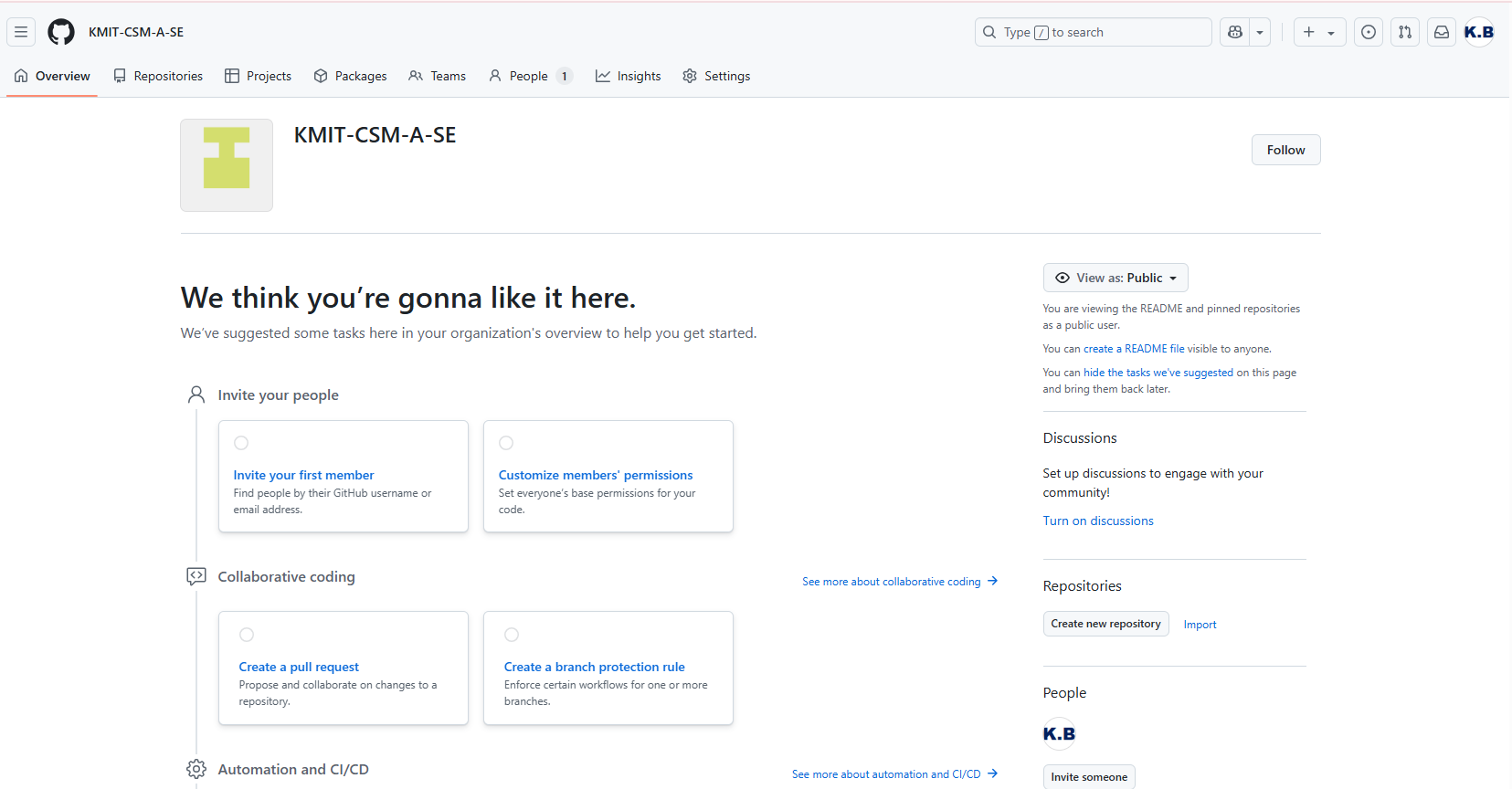
Or can be teams KMIT-CSM-A-SE

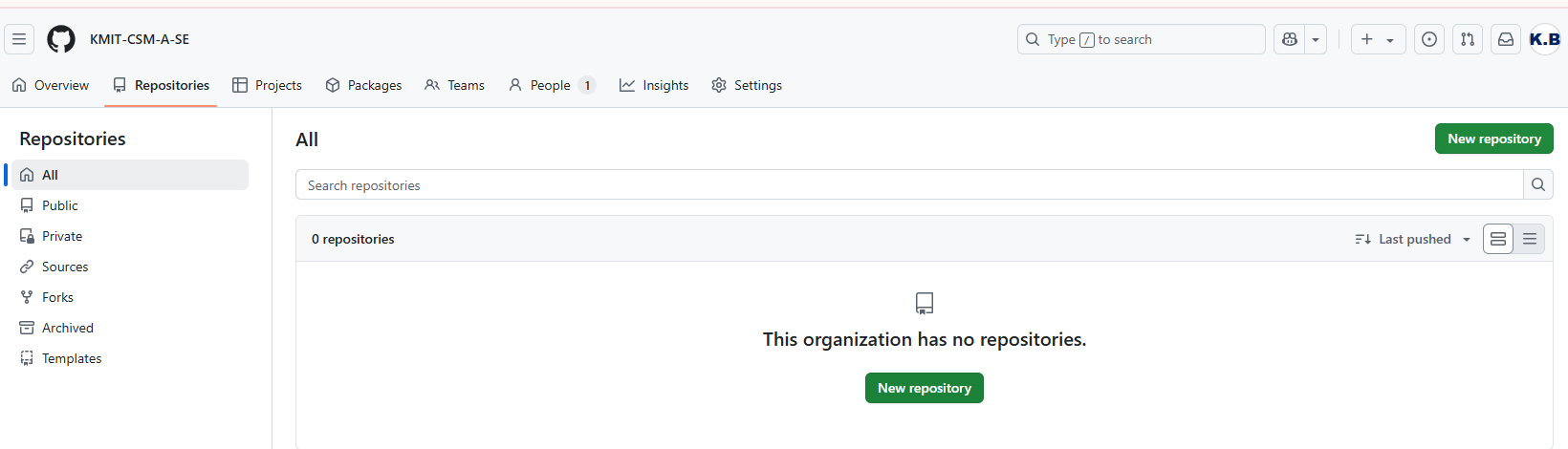
├── SEProject-TeamA

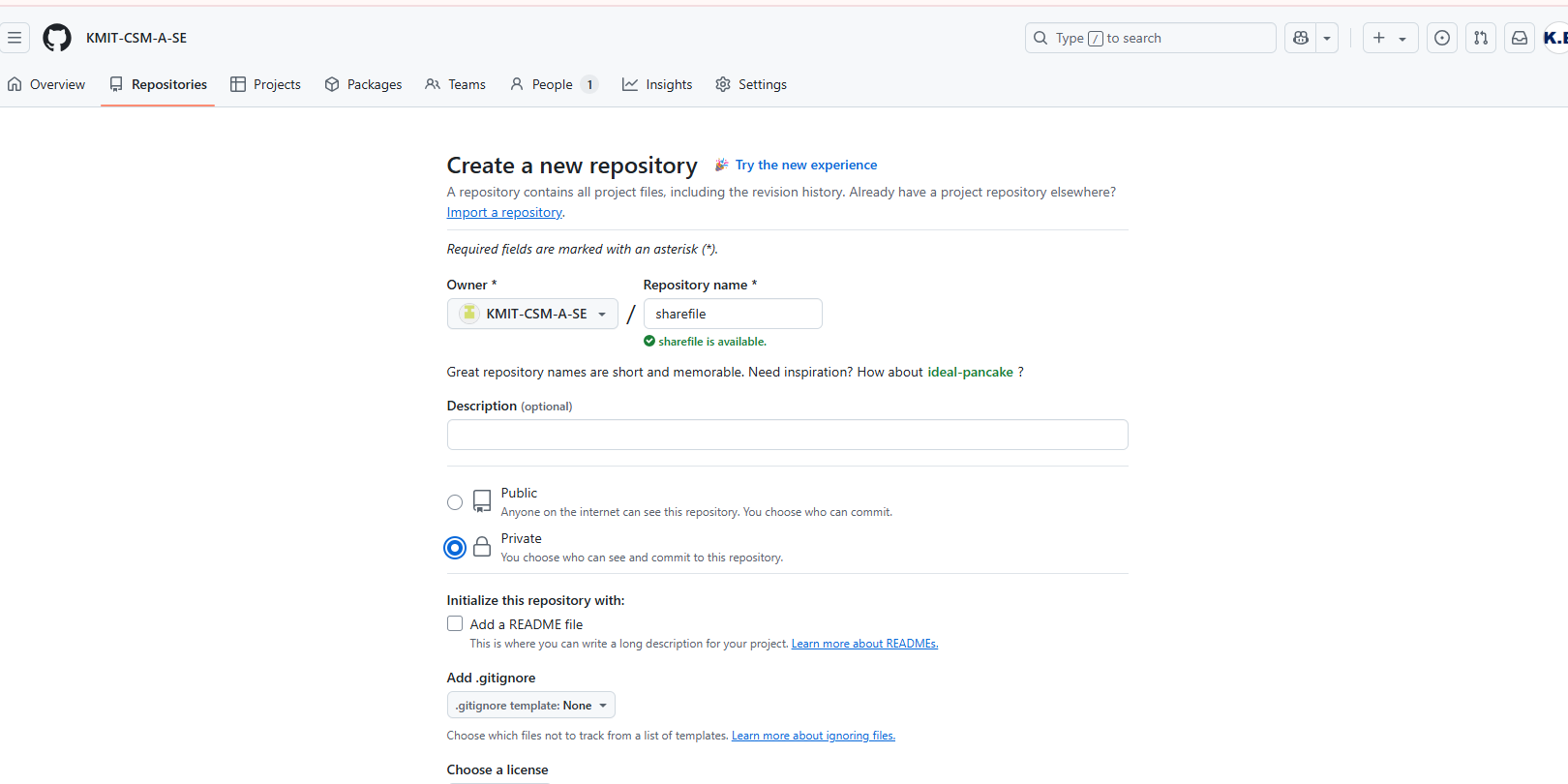
├── SEProject-TeamB

└── SEProject-TeamC

**Step 4** : Create the remote repository for storing the project related files. This repository is accessible to every member of the team as per the permissions given.

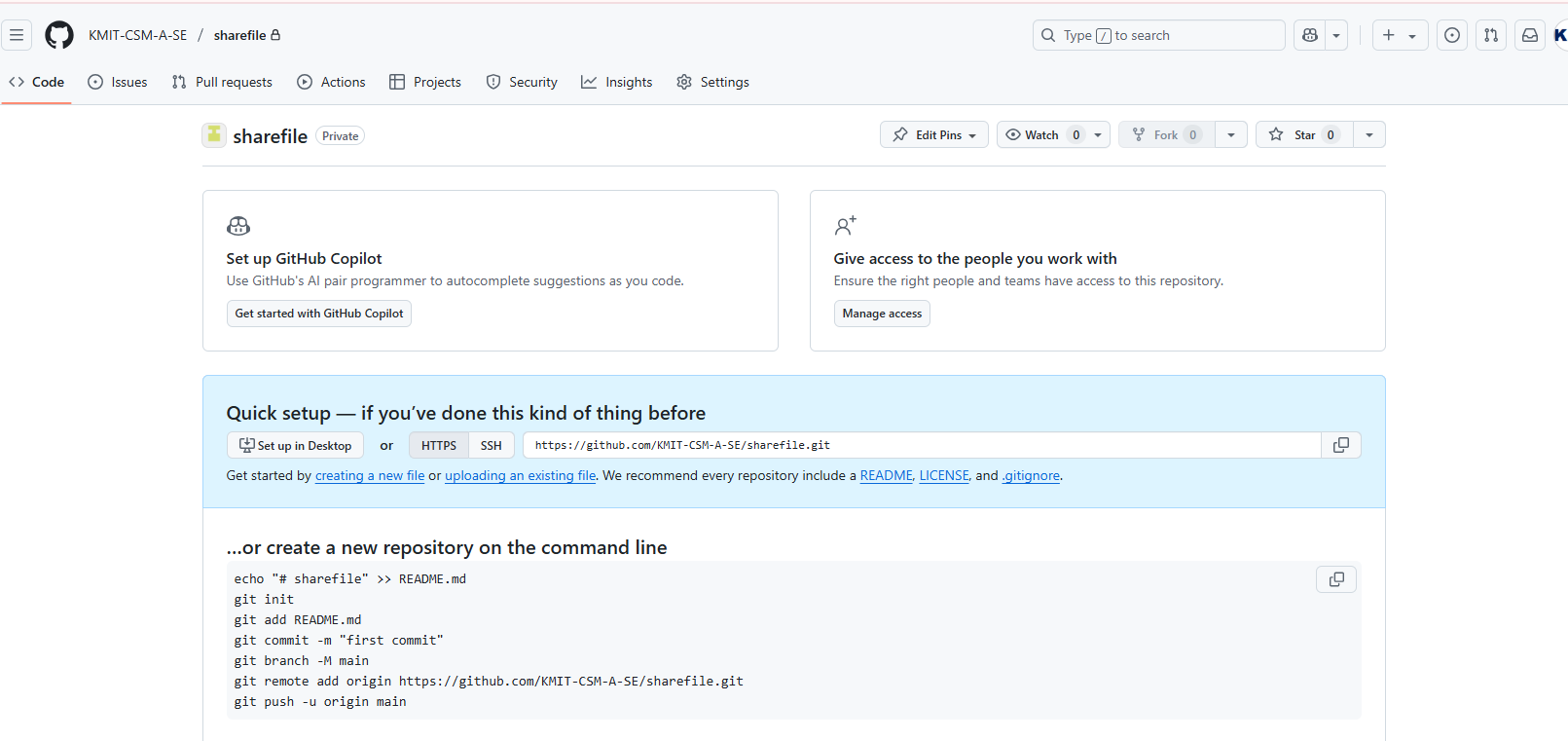






The repository can be made private so that it is accessible only to the group members rather than being in a public domain.

Now all the members of the team can contribute to the development of the project and the different files with all the versions and modification notices will be available in the respective repositories and is accessible to all the members



**Student Workflow**

git clone https://github.com/KMIT-SE-2025/SEProject-TeamA.git

**# Create a branch (optional)**

**git checkout -b feature-branch**

**# Make changes, then push**

**git add .**

**git commit -m "Added login feature"**

**git push origin feature-branch**