INTRODUCTION TO GITHUB

WHAT IS GITHUB???

Imagine you and your friends are working on a big project together, like creating a Website. Each of you writes different parts of the code. To make sure everything comes together nicely, you need a way to keep track of all the changes and see who made what. Here GitHub comes into picture.

GitHub is like a collaborative bookshelf for code and projects. It keeps track of changes, helps you work with others, and makes sure your (code) is safe and organized. It's an essential tool for developers and teams to work together and create amazing projects!

Where do we use GITHUB??

GitHub is a place on the internet where programmers/individual store their code and work together on software projects. It's like a virtual workspace where developers can collaborate and keep track of all the changes they make to the code.

It is used by developers, teams, open-source projects, companies, and individuals to manage, collaborate on, and share code, making it an integral part of modern software development workflows.

Some more use-case of GITHUB

- **Code Storage :** It acts as a storage , where we generally store our codes. It provides a secure and reliable platform to keep code safe and accessible from anywhere.
- **Learning and Collaboration:** GitHub fosters a strong developer community where individuals can learn from others, contribute to projects they find interesting, and collaborate with developers from around the world.
- **Portfolio and Showcasing Work:** For individual developers, GitHub can serve as a portfolio to showcase their coding skills and contributions to various projects.
- **Version Control:** It is primarily used for version control, which means tracking changes in code over time. It allows developers to work on the same project simultaneously, keeping track of who made what changes and when. This ensures that the codebase remains organized and enables easy collaboration.

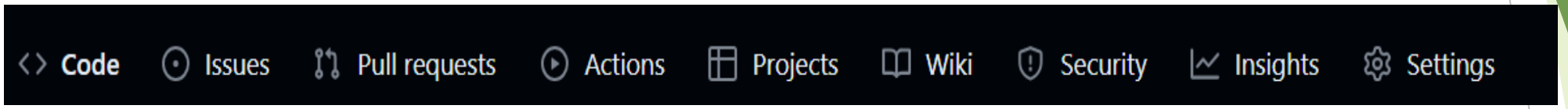
COMPONENTS OF GITHUB

The main component of Github is **REPOSITORIES.**

A repository, often referred to as a "repo," is a central storage space on GitHub where you can keep all the code and related files for a specific project. It acts as a container that holds everything needed to build and maintain the software application or project.

This can be public, allowing anyone to see the code, or private, restricted to a specific group or organization.

Other Components are:-



Code: IT is the core element of a repository that stores the actual source code and related files for a software project.

Issues : It is a feature that helps developers and teams track, manage, and discuss tasks, bugs, feature requests, and other project-related items. It serves as a centralized system for communication and collaboration around the development process.

Pull Request: It is a crucial feature that facilitates collaboration, code review, and the merging of changes from one branch to another within a repository. Pull requests are commonly used when developers want to propose their code changes to be integrated into the main branch of a project.

Project : Project component in GitHub is a powerful tool for task and project management. It helps teams visualize their work, organize tasks, and collaborate effectively, enhancing overall productivity and project success.

Security: It is a set of features and tools designed to help developers and repository owners enhance the security of their code and identify potential vulnerabilities in their projects. The Security component focuses on providing insights and actions to address security concerns in software development.

Wiki: It is a feature that allows developers and project contributors to create and maintain documentation, user guides, and other informative content related to a repository.

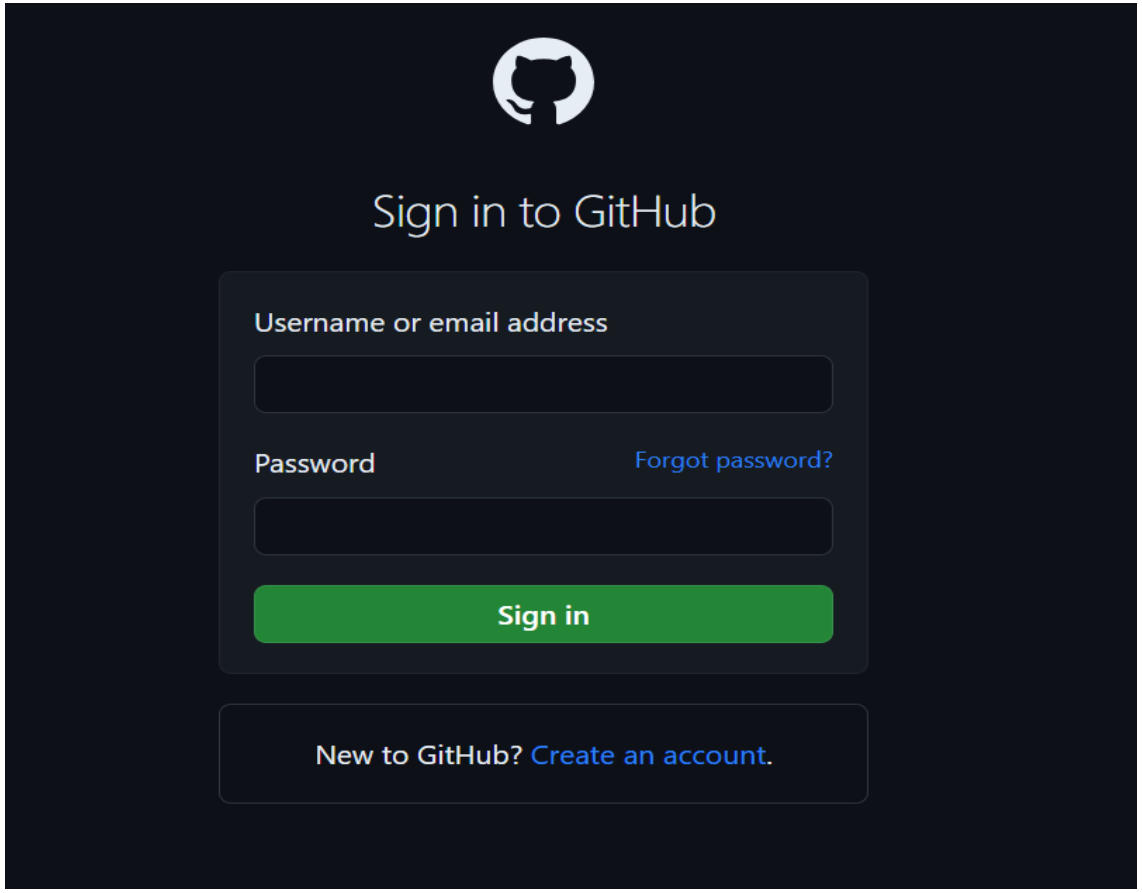
Insights: It offer detailed metrics and visualizations that enable better decision-making and understanding of how the project is progressing.

Settings: This allows repository owners and administrators to manage various aspects of the repository, set access permissions, configure integrations, and customize the repository's behavior.

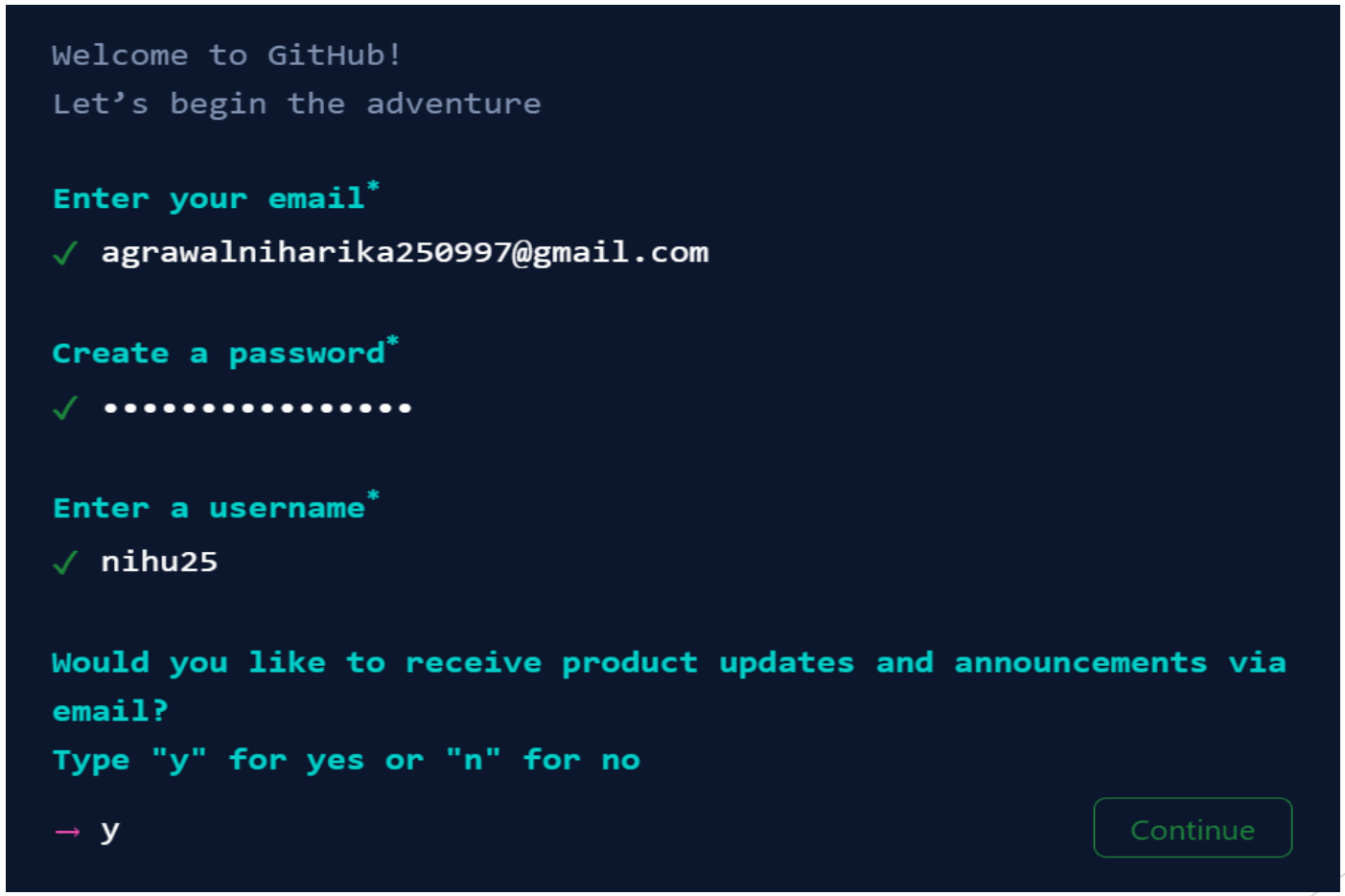
CREATING GITHUB ACCOUNT

1. Login to https://github.com/

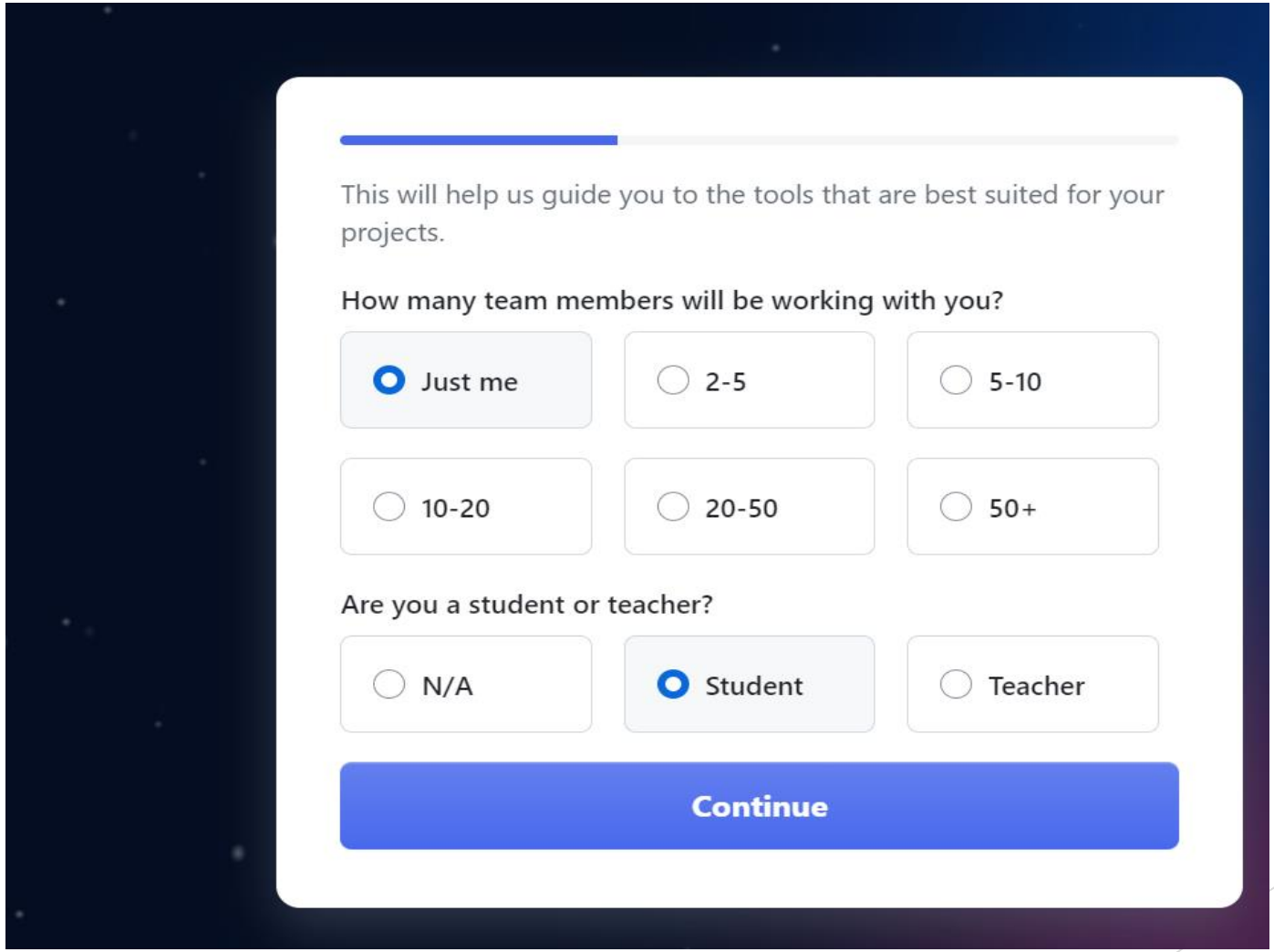
2. Click on Create Account.



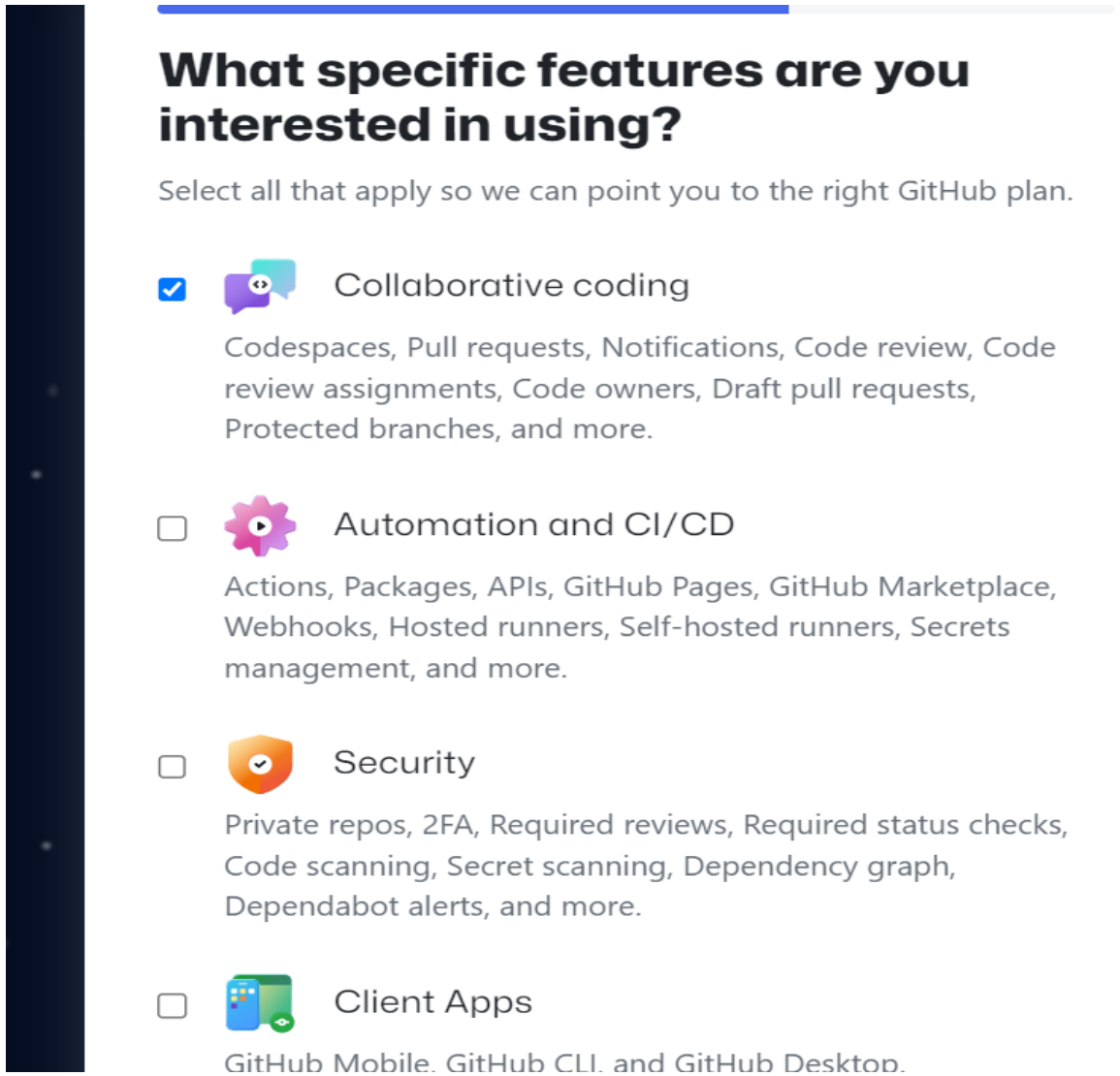
3. Enter Email ID , Password , Username and then click on continue.



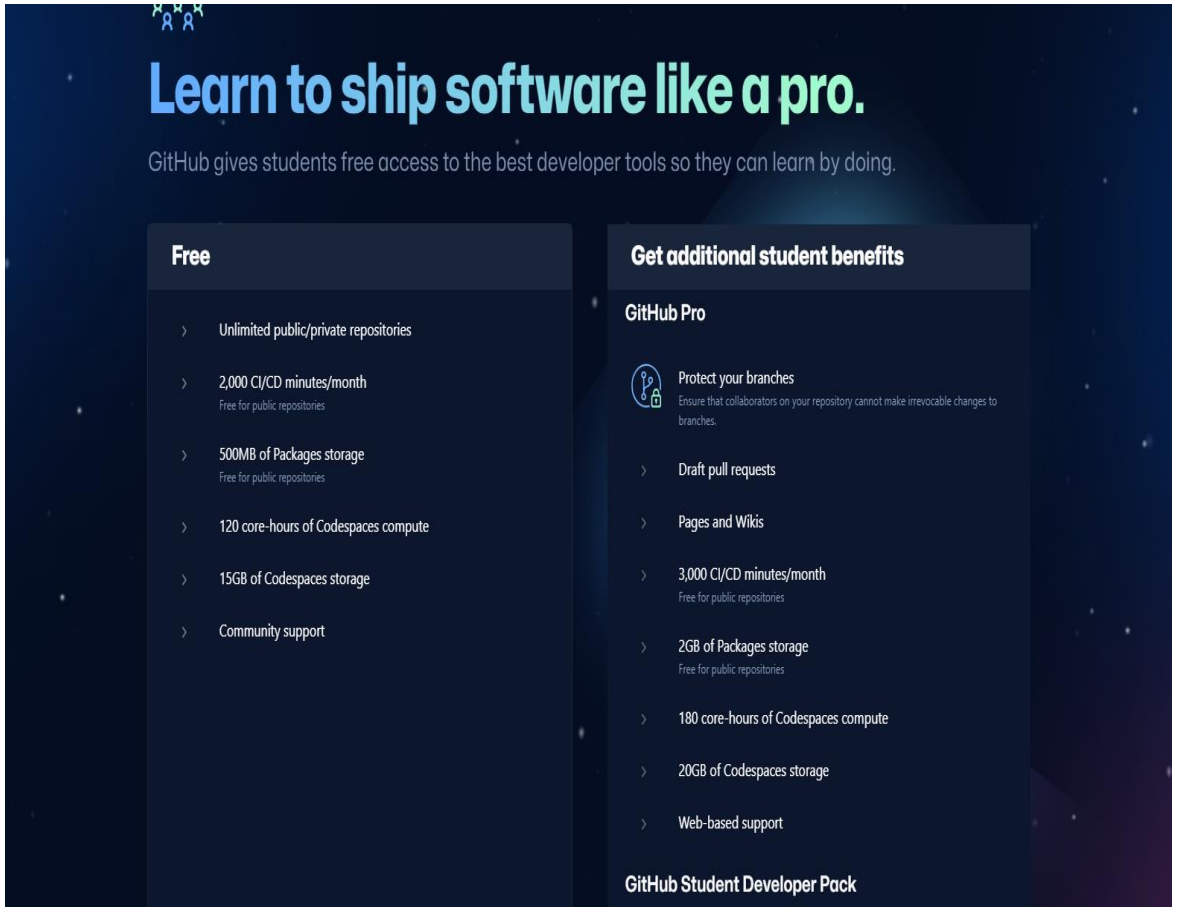
4. Select the below option and click on continue.

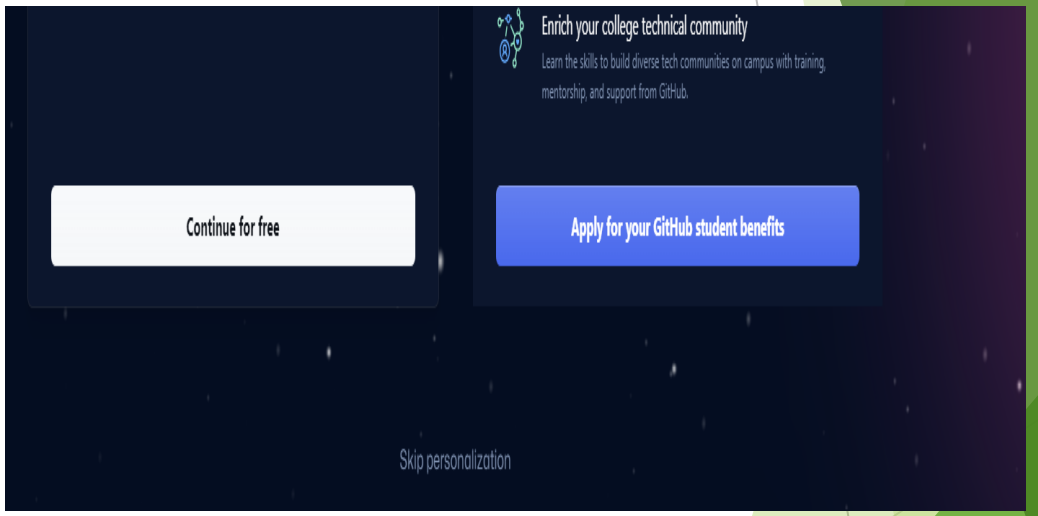


5. Select the below option and click on Continue.

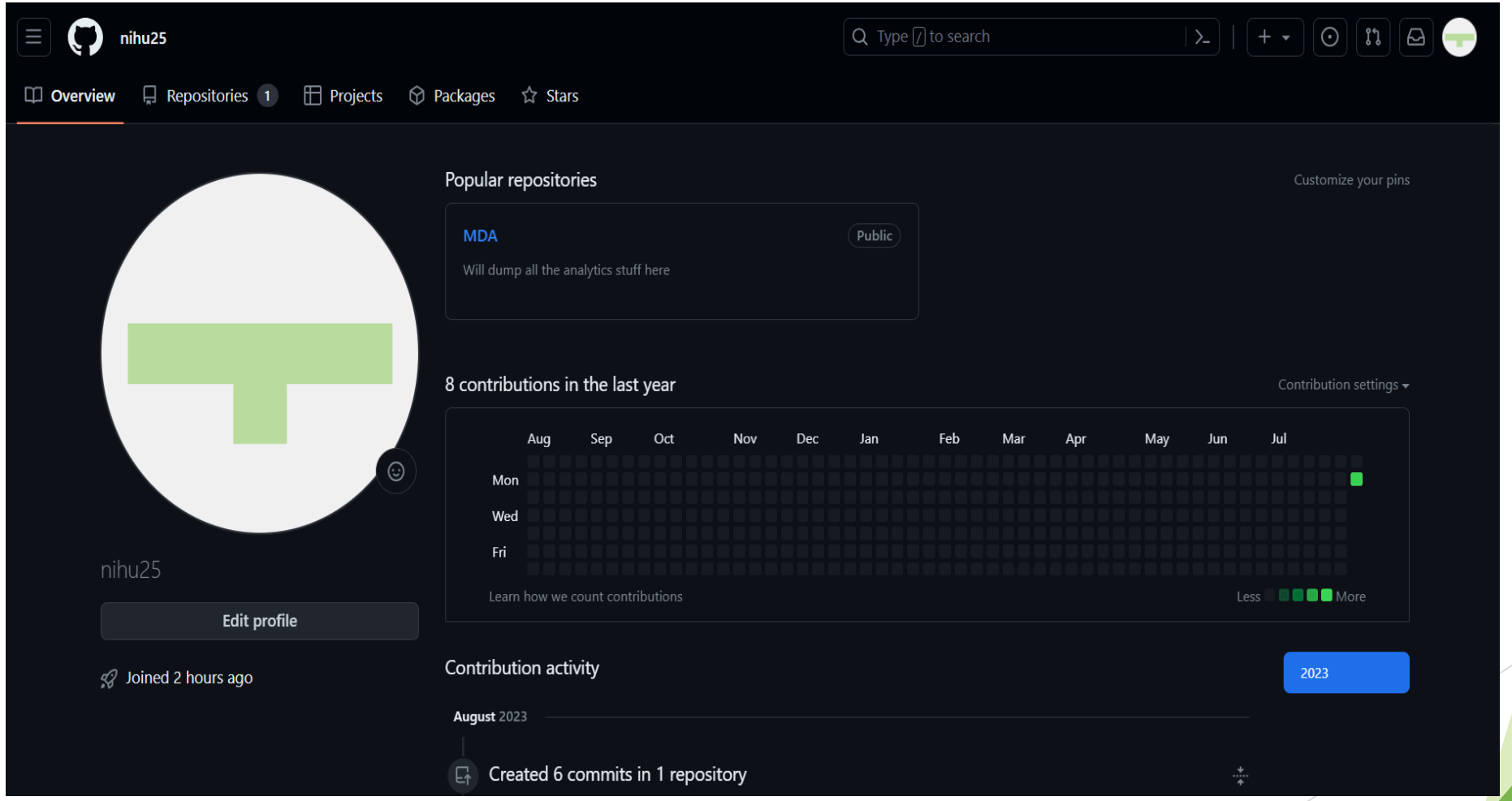


6. Next choose the Free-Trial Option and Click on Continue for Free.

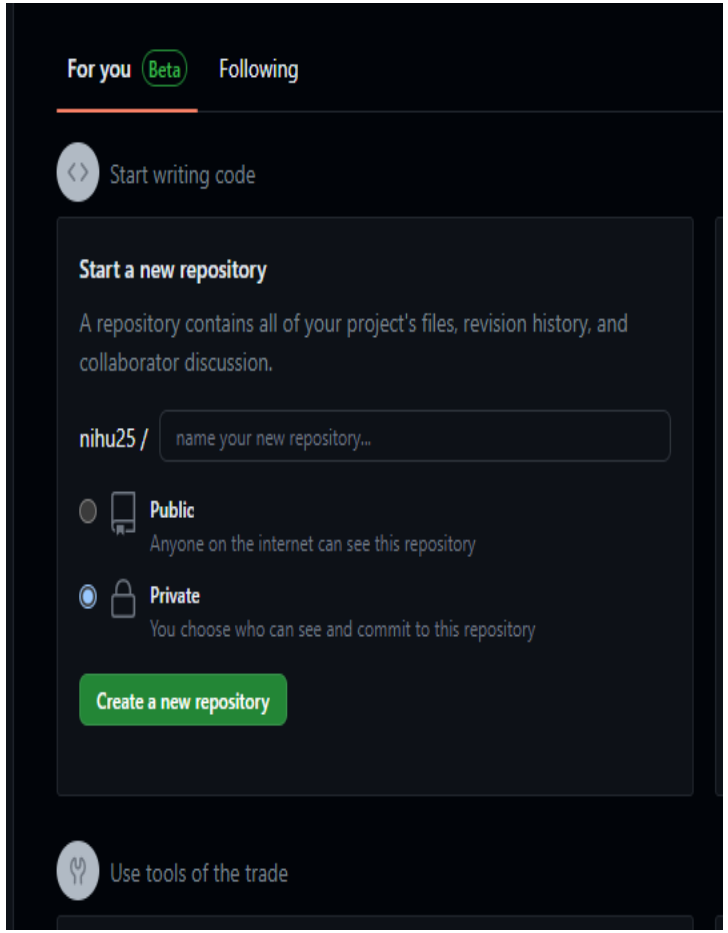


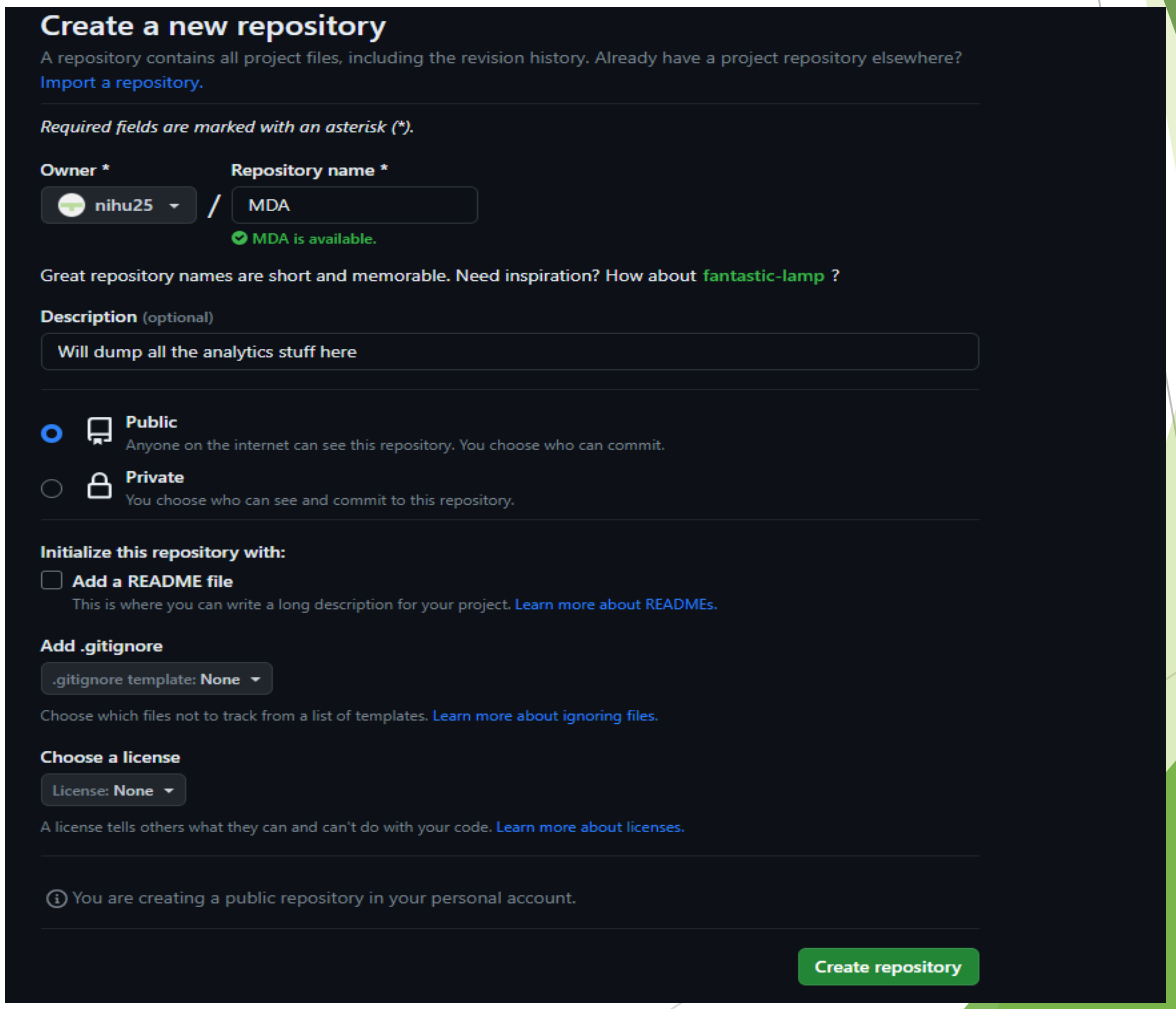


7. Once your get account created, you will land into your Github Profile.

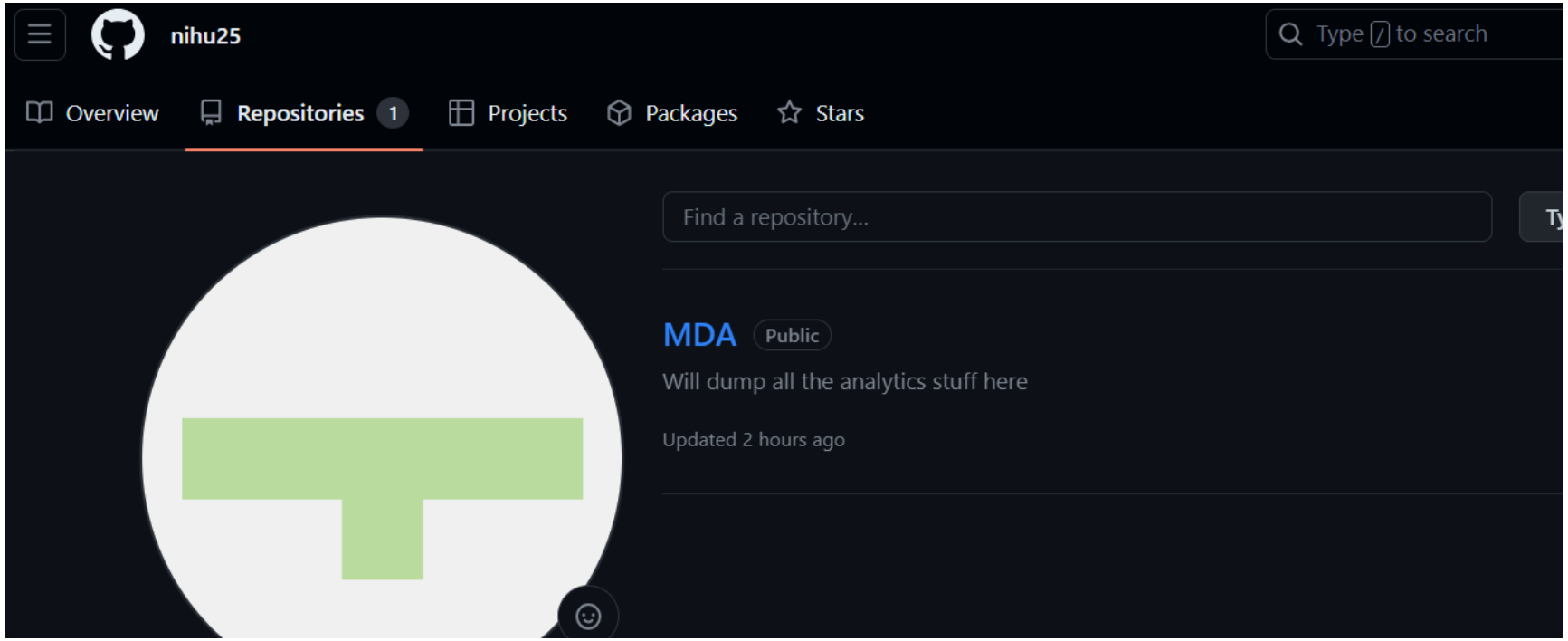


8.Now will start with creation of Repositories.
Click On Create a new repository (Fig .1). Enter the Repository name and then Click on Create Repository(Fig 2)



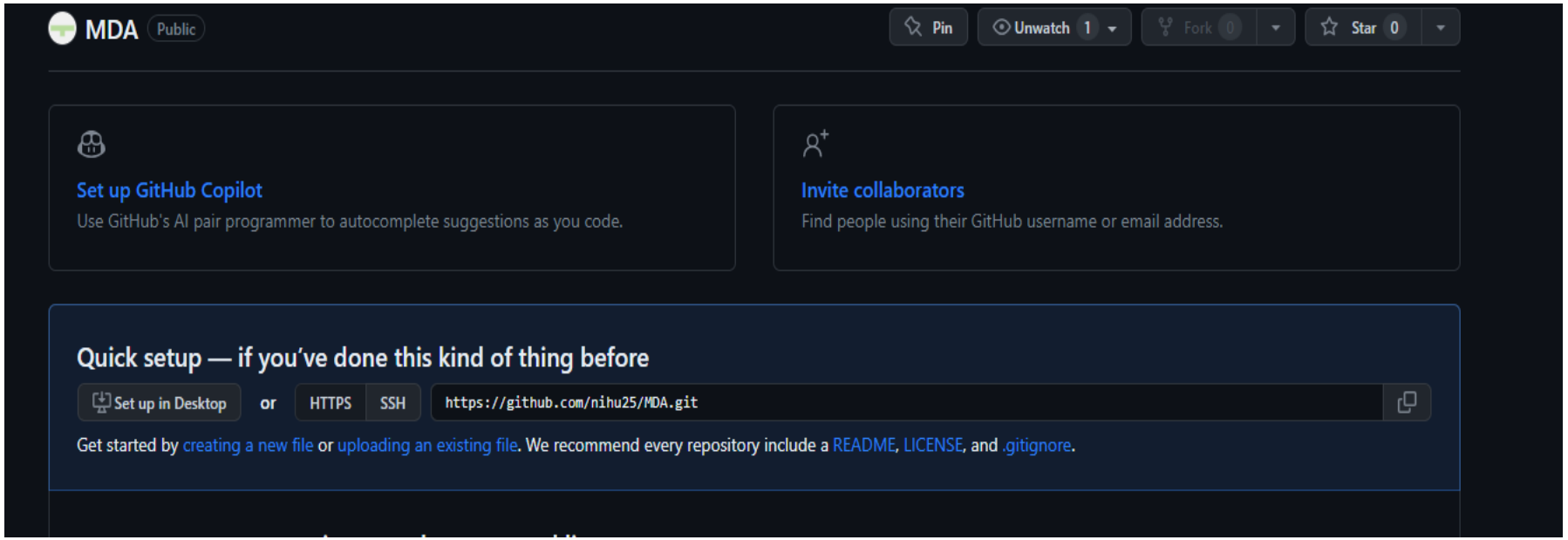


9. Once it is created it will display on your profile.



Now the next step is to create sub folders inside that repository.

10. Click on the Repository you created and there you will get a option to create a new file.



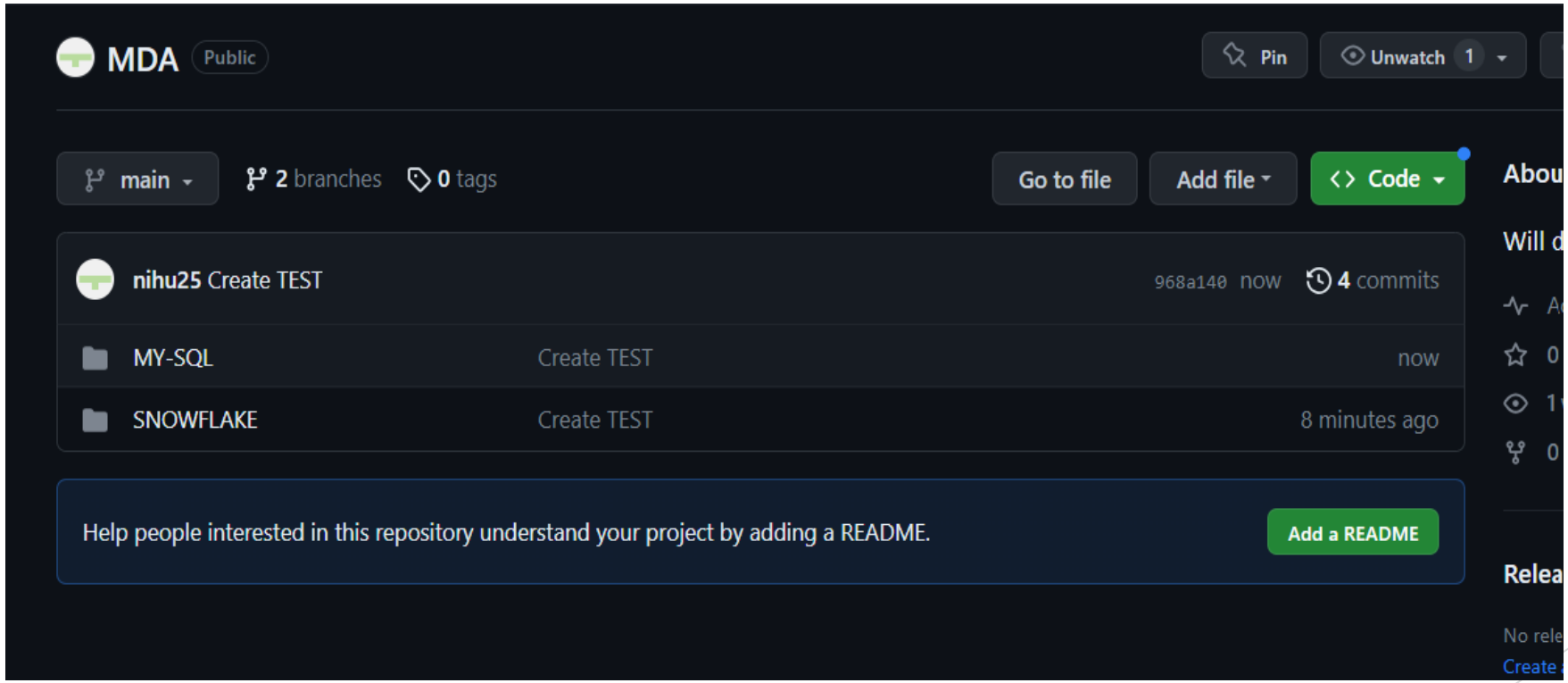
11. There is no direct option to create folder inside repository , for creating folder will click on Create new file, will give file name with slash (/). Folder will get created.Here MY-SQL folder.

After that enter the file name you want to create inside that folder(Creating TEST File here) and click on COMMIT.

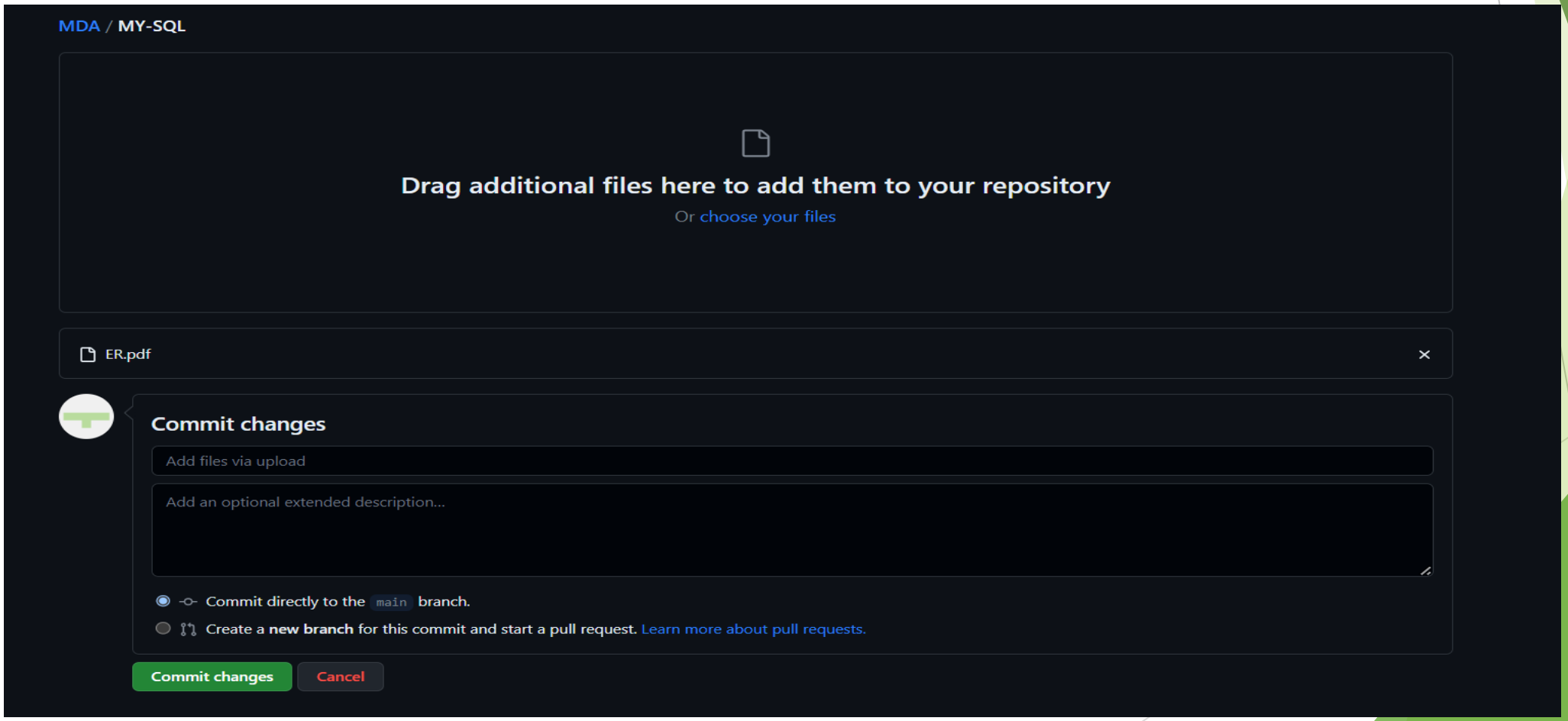
WHY COMMIT ??

Whenever any changes we make in repository (creation of file, uploading file etc) we commit those changes so that it get saved.

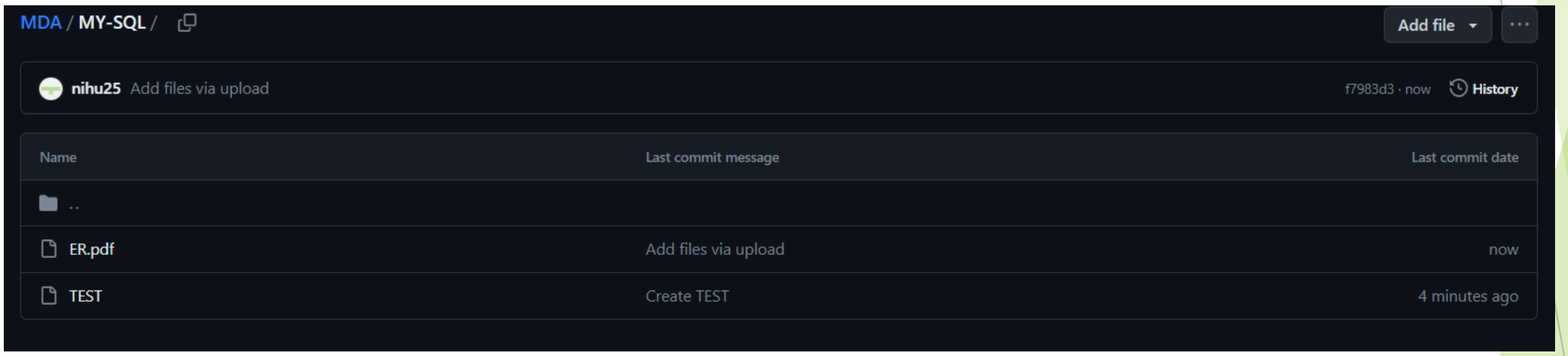
12. Two folders created in repository and inside every folder TEST file created.
Similarly you need to create folders.



13. Uploading existing file : Go inside the folder where you want to upload any file. Here ER.pdf.
Choose the file and then click on Commit. It will get uploaded.



Here we can see we have MDA repository , we have MY-SQL folder and then we have one TEST file and other we uploaded an existing ER.pdf file.



Similarly, you can create other folders and upload the file.

GIT CLI

We can also perform all the github operation using command line. For that we have Git CLI.

Git CLI (Command Line Interface) is a command-line tool that provides access to the Git version control system. It allows developers to interact with Git repositories and manage their codebase using text commands typed into a terminal or command prompt. The Git CLI is available on various operating systems, including Windows, macOS, and Linux.

Some special Git Cli Command

- **git clone [repository_url]:** Clone (download) a repository from GitHub to your local machine. Replace [repository_url] with the URL of the repository you want to clone.
- **git commit :** Create a new commit .
- **git status:** Check the status of your working directory and see which files are modified, staged, or untracked.
- **git push:** Push your local commits to the remote repository on GitHub.
- **git pull:** Fetch changes from the remote repository and merge them into your local branch.
- **git init:** Initialize a new Git repository in the current directory.

THANK YOU