Software Assignment

1. Explain the fundamental concepts of version control

* With version control, every change made to the code base is tracked. This allows software developers to see the entire history of who changed what at any given time — and roll back from the current version to an earlier version if they need to.

1. Why GitHub is a popular tool for managing versions of code

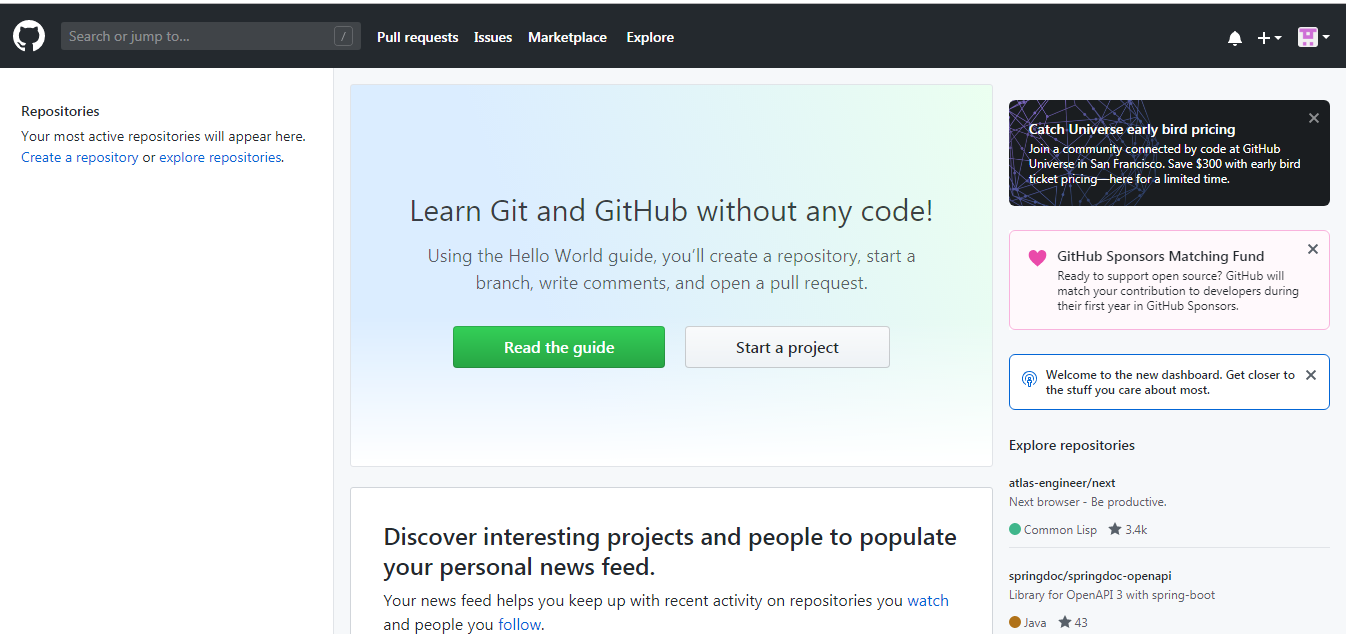
* Businesses use GitHub as version control systems, letting development team members track changes to source code as developers collaborate on it.
* GitHub is a cloud-based hosting service that lets you manage Git repositories.

1. How does version control help in maintaining project integrity?

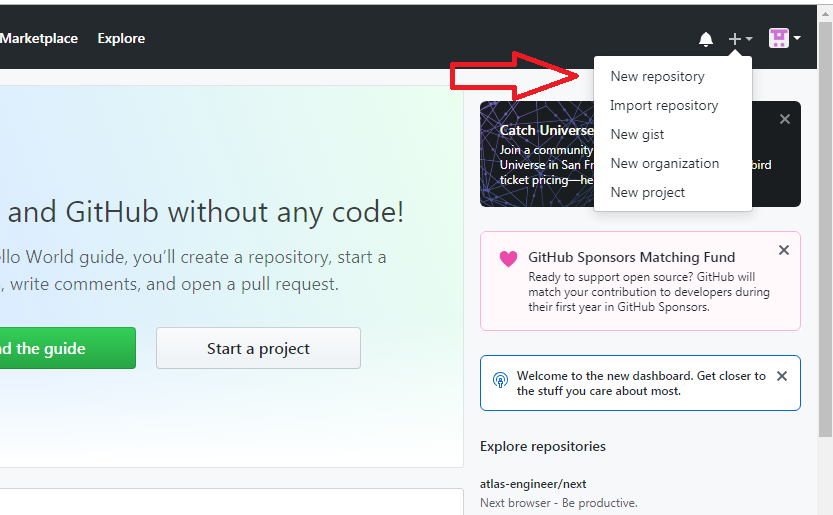
* Version control allows the developer "orchestra" to see every commit and access, review, collaborate, experiment, compare, and undo changes to ensure code integrity and faster releases.

1. Describe the process of setting up a new repository on GitHub.

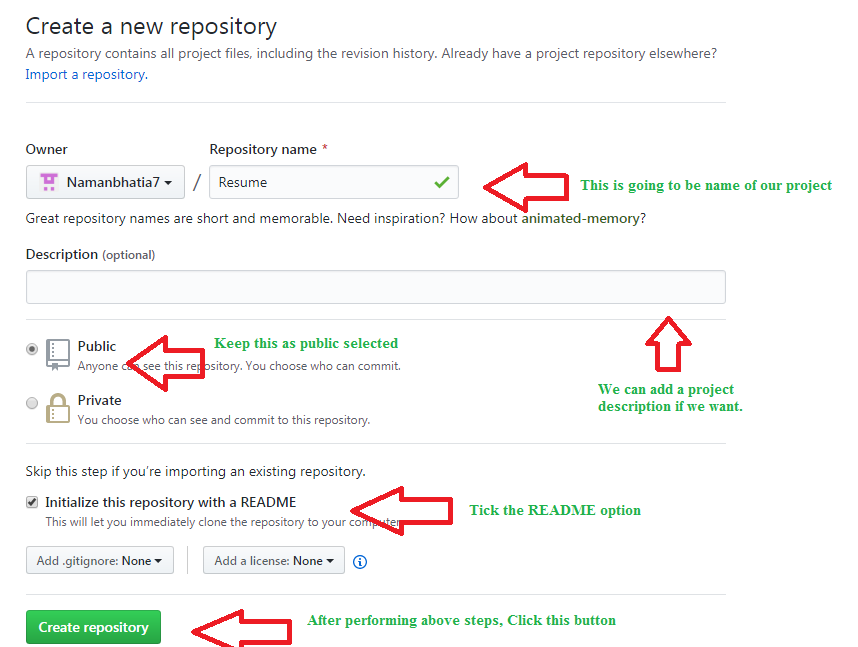
* **Step 1:** After successfully setting up GitHub account login to your account.



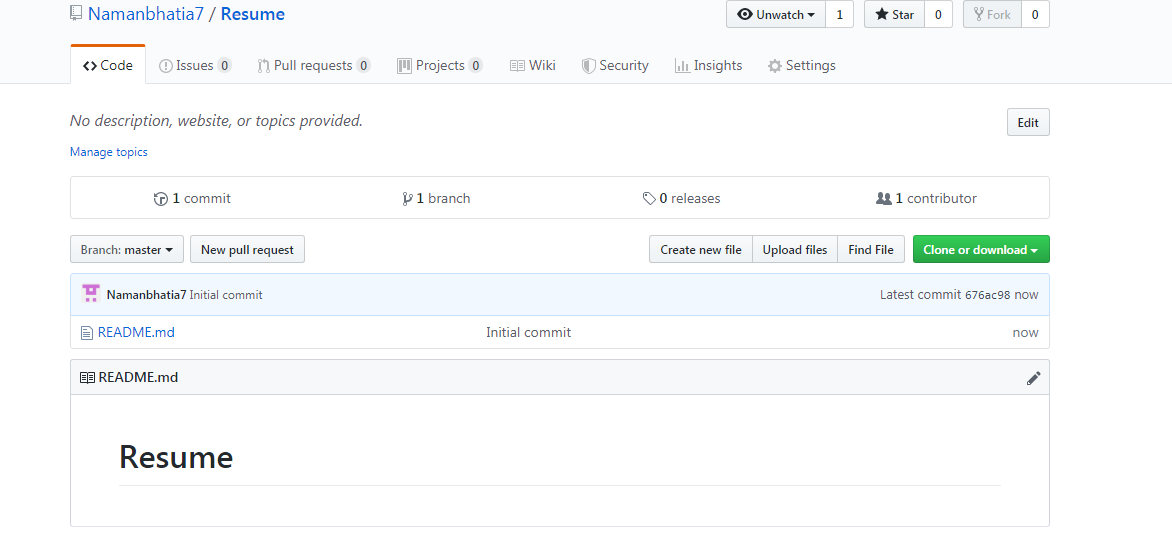
* **Step 2:** Click on the **new repository** option.



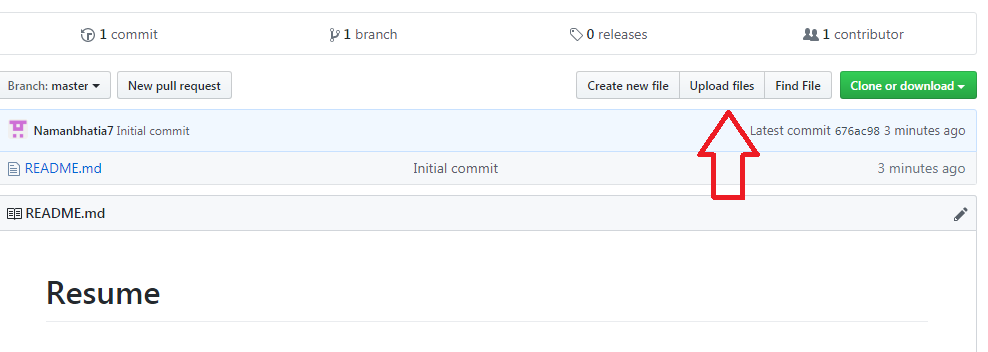
* **Step 3:** After clicking **new repository** option, we will have to initialize some things like, **naming our project**, choosing the **visibility** etc. After performing these steps click **Create Repository** button.



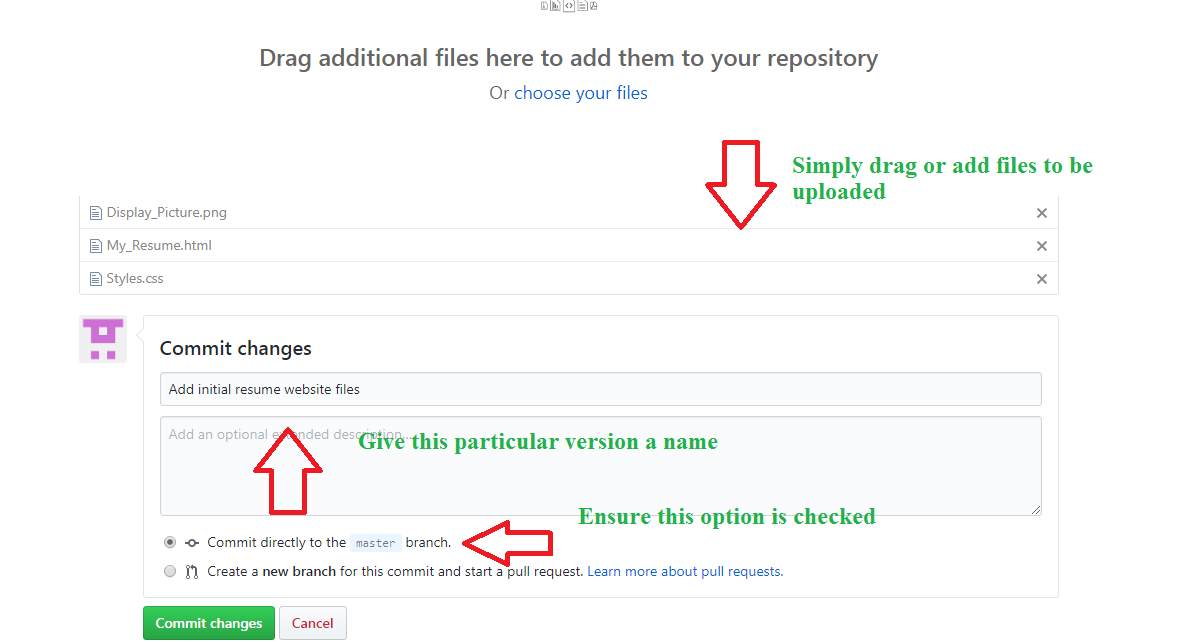
* **Step 4:** After clicking the button, we will be directed to below page. Right now the only file we have is a readme file.



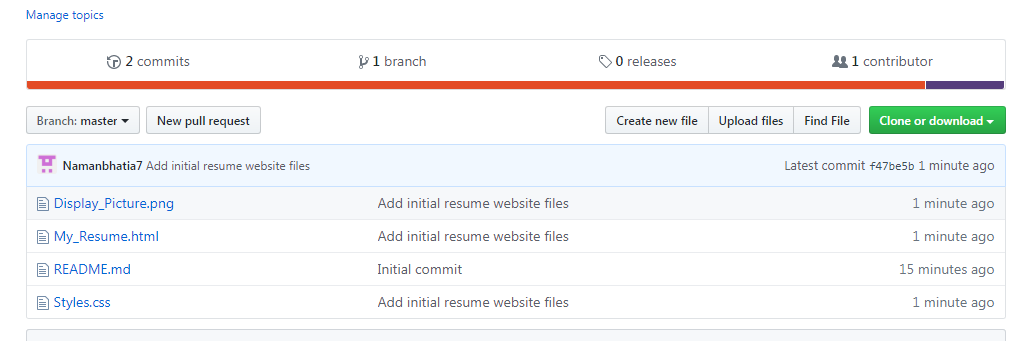
* **Step 5:** Now click on the “Upload files” button



* **Step 6:** Follow the steps mentioned in the Picture below and click “commit changes”



* **step 7:** Now you will see that all of our files uploaded in our GitHub



1. Discuss the importance of README file in a GitHub repository.

* Adding a README file to a repository to communicate important information about the project.

1. What should be included in a well-written README, and how does it contribute to effective collaboration.

* **Project Title and Description**

Begin with a concise title that succinctly captures the essence of your project. It should be followed by a detailed description of the project. What does your project do? What problems does it solve? Why is it valuable?

* **Installation Instructions**

Your README should provide clear installation instructions. If the project has any dependencies, make sure to include them in the steps. A code snippet showing the installation commands could be quite helpful.

* **Usage**

Show examples of how your project should be used. The user should understand how to implement your project with minimal effort. Include code snippets or screenshots to illustrate.

* **Contributing**

If your project is open-source and you want to encourage others to contribute, include a section about how one can contribute. Direct them to the necessary resources and processes.

* **Documentation**

If your project has additional documentation, provide links to these resources.

* **License**

Include a section for the license of your project. This informs users about their legal rights and restrictions when using your software.

* **Contact Information**

Make sure to provide contact details or ways to support the project if there are any questions or concerns.

* **Acknowledgments**

If your project uses code from other open-source projects, it’s professional to provide acknowledgment and credit where it’s due.

1. Compare and contrast the differences between a public repository and a private repository on GitHub.

* Public repositories are accessible to everyone on the internet.
* Private repositories are only accessible to you.

1. What are the advantages and disadvantage of each, particularly in the context of collaborative projects?
2. Detail the steps involved in making your first commit to a GitHub repository.

* Create a new branch called example-tutorial-branch

git checkout -b example-tutorial-branch

* In a text editor like Visual Studio Code, Sublime, vi, or any other editor, open the README.md file and add this text:

Hello world! I'm using Git!

* Save the file.
* Git keeps track of changed files. To confirm which files have changed, get the status.

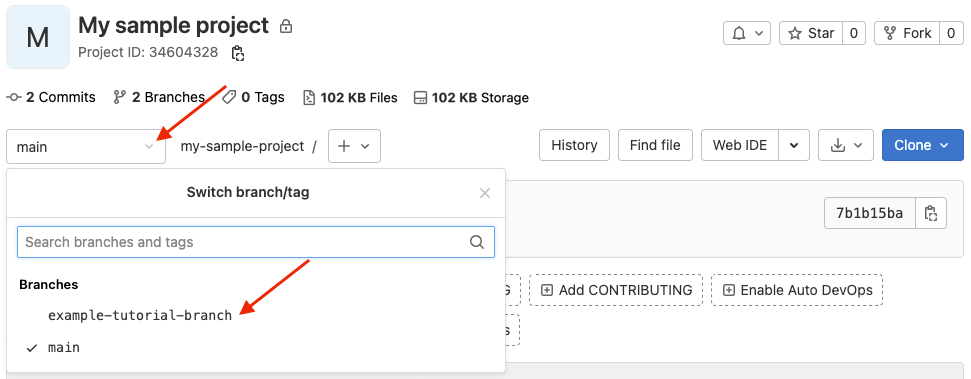
git status

* You should get output similar to the following:
* On branch example-tutorial-branch
* Changes not staged **for** commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes **in** working directory)

* modified: README.md
* no changes added to commit (use "git add" and/or "git commit -a")
* **Commit and push your changes**

[](https://docs.gitlab.com/ee/tutorials/make_first_git_commit/img/branches_dropdown_v14_10.png)

1. Explore the role of pull requests in the GitHub workflow.

* A pull request is a proposal to merge a set of changes from one branch into another. In a pull request, collaborators can review and discuss the proposed set of changes before they integrate the changes into the main codebase. Pull requests display the differences, or diffs, between the content in the source branch and the content in the target branch.

1. Discuss the concept of “forking” a repository on GitHub.

* A fork is a new repository that shares code and visibility settings with the original “upstream” repository.

1. How does forking differ from cloning.

* **Forking** creates your own copy of a repository in a remote location (for example, GitHub). Your own copy means that you will be able to contribute changes to your copy of the repository without affecting the original repository.
* **Cloning** makes a local copy of a repository, not your own copy. Think of it as downloading a repository onto your local hard drive. Unlike forks, clones have references to their original repositories.

1. Examine the importance of issues and project boards on GitHub. How can they be used to track bugs, manage tasks, and improve project