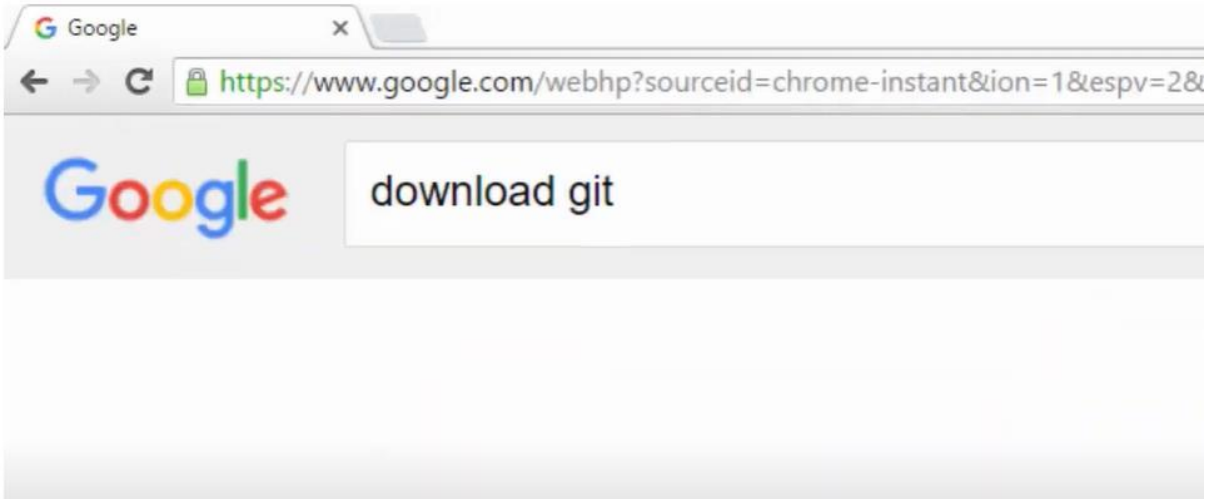
<u>Experiment 1.</u> Installation of Git and creating repository.

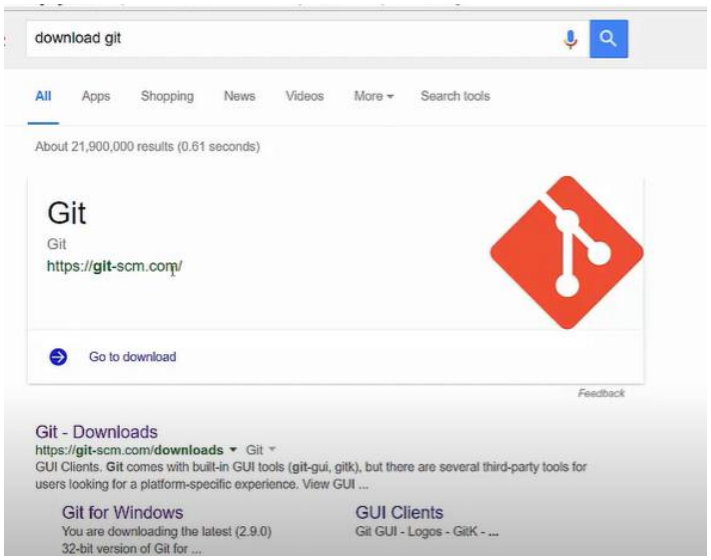
- Git is a version control system used for tracking changes in computer files. It is generally used for source code management in software development.
- Git is used to tracking changes in the source code
- The distributed version control tool is used for source code management
- It allows multiple developers to work together
- It supports non-linear development through its thousands of parallel branches
- **Git** is an **open-source distributed version control system**. It is designed to handle minor to major projects with high speed and efficiency. It is developed to co-ordinate the work among the developers. The version control allows us to track and work together with our team members at the same workspace.
- Git is foundation of many services like **GitHub** and **GitLab**, but we can use Git without using any other Git services. Git can be used **privately** and **publicly**.
- Git was created by **Linus Torvalds** in **2005** to develop Linux Kernel. It is also used as an important distributed version-control tool for **the DevOps**.
- Git is easy to learn, and has fast performance. It is superior to other SCM tools like Subversion, CVS, Perforce, and ClearCase.

Steps to install Git:

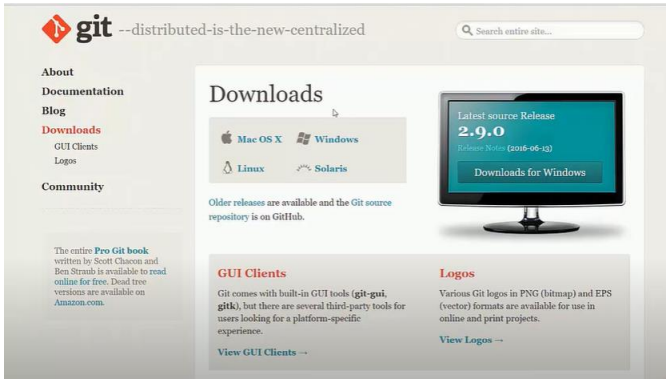
Step 1- Open browser and type Download Git

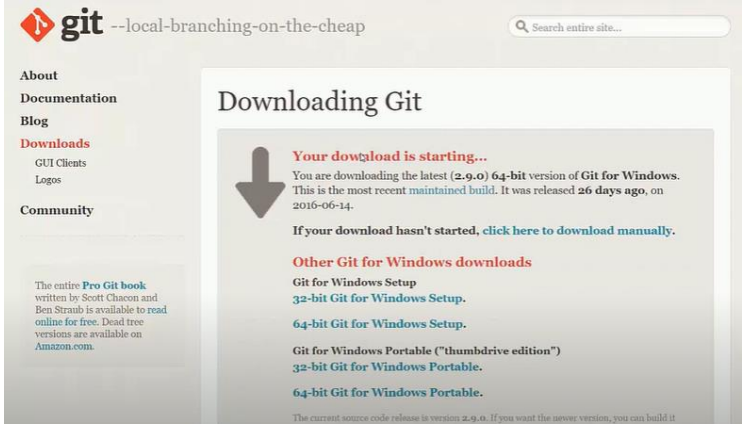


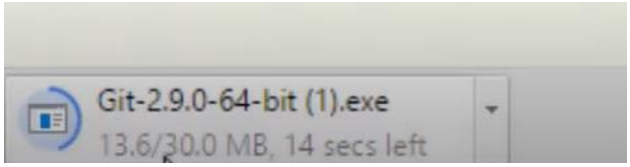
Step 2- Git-scm is the official website, you can click that or just click on Git downloads



Step3- Click the download link for Windows and allow the download to complete.

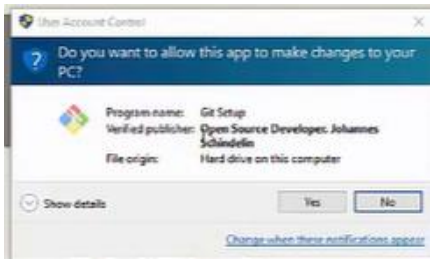






Step 4- Browse to the download location (or use the download shortcut in your browser). Double-click the file to extract and launch the installer.

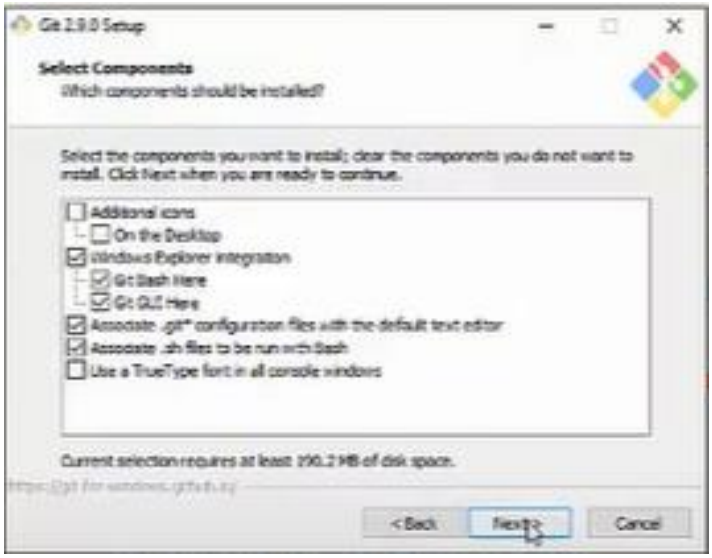
Step 5- Allow the app to make changes to your device by clicking **Yes** on the User Account Control dialog that opens.



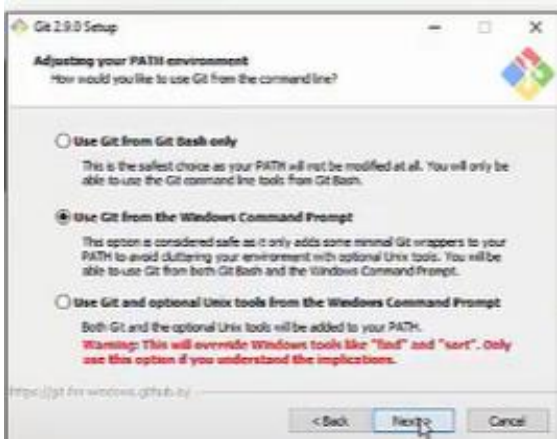
Step 6- Review the GNU General Public License, and when you're ready to install, click **Next**.

Step 7- The installer will ask you for an installation location. Leave the default, unless you have reason to change it, and click **Next**.

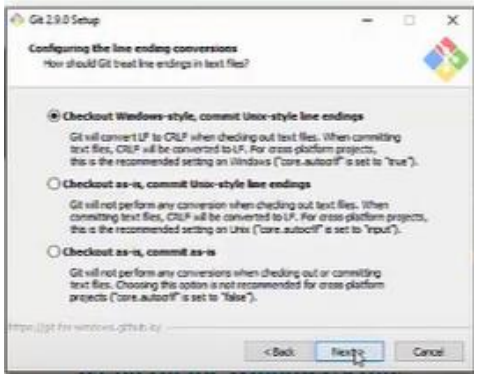
Step 8- A component selection screen will appear. Leave the defaults unless you have a specific need to change them and click **Next**.



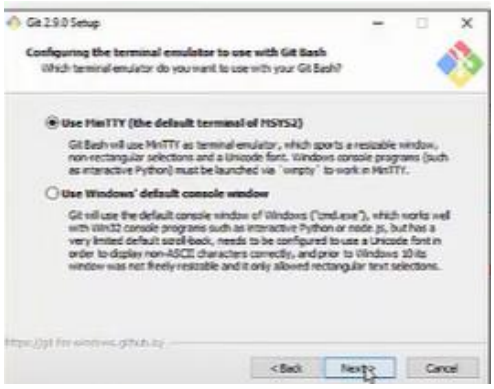
Step 9- This installation step allows you to change the **PATH environment**. The **PATH** is the default set of directories included when you run a command from the command line. Leave this on the middle (recommended) selection and click **Next**.



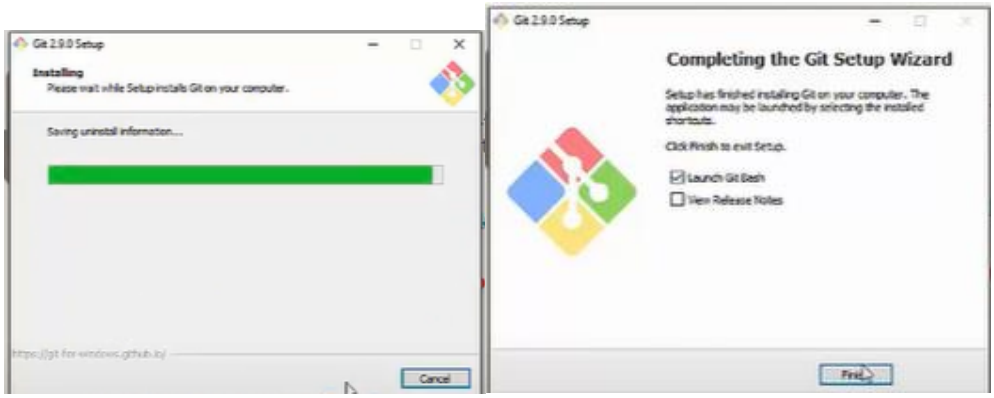
Step 10- The next selection converts line endings. It is recommended that you leave the default selection. This relates to the way data is formatted and changing this option may cause problems. Click **Next**.



Step 11- Choose the <u>terminal emulator</u> you want to use. The default MinTTY is recommended, for its features. Click **Next**.



Step 12- Once the installation is complete, tick the boxes to view the Release Notes or Launch Git Bash, then click **Finish**.

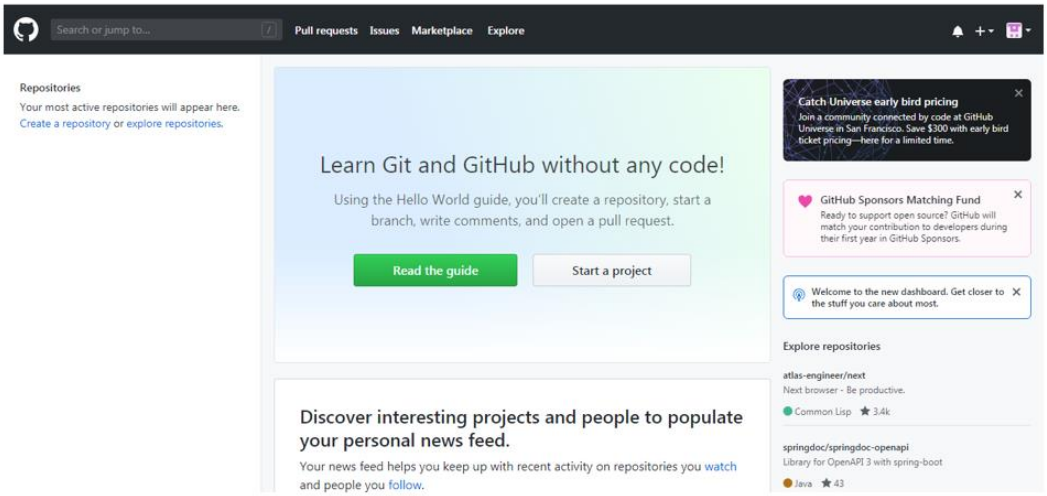


Step 13- To launch **Git Bash** open the **Windows Start** menu, type ***git bash*** and press **Enter** (or click the application icon).

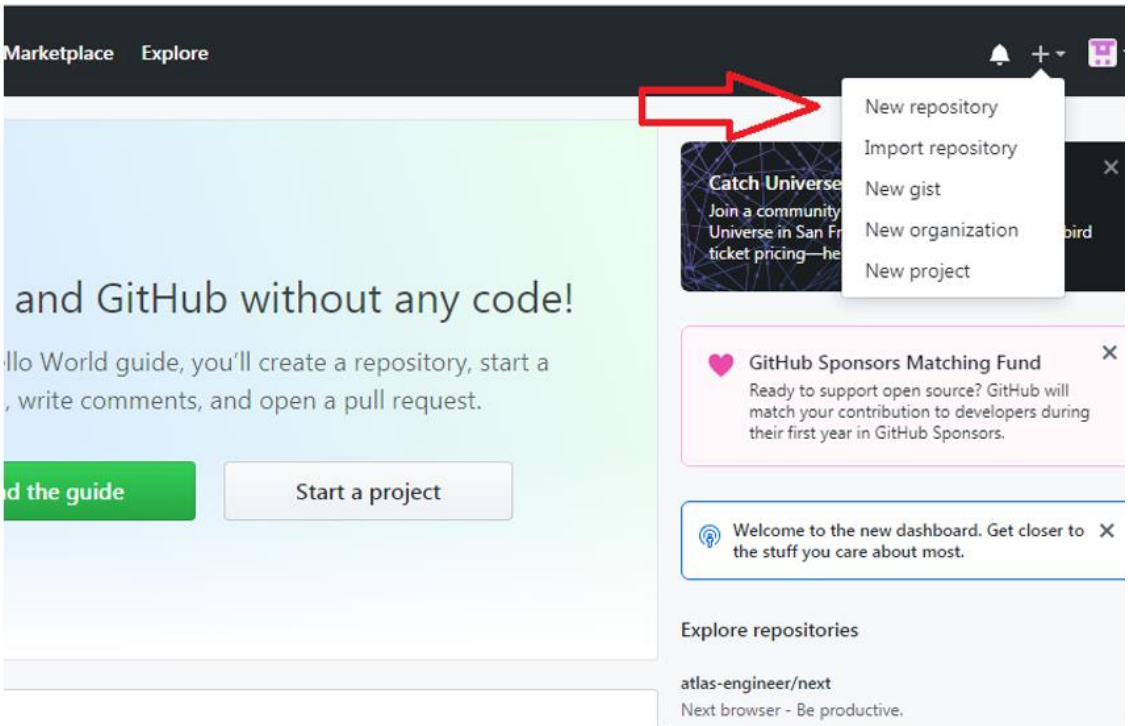

```
ASUS@LAPTOP-SNVS57GF MINGW64 ~
$ git
usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           [--super-prefix=<path>] [--config-env=<name>=<envvar>]
           <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
   clone     Clone a repository into a new directory
   init      Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
   add       Add file contents to the index
   mv        Move or rename a file, a directory, or a symlink
   restore   Restore working tree files
   rm        Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
   bisect    Use binary search to find the commit that introduced a bug
   diff      Show changes between commits, commit and working tree, etc
   grep      Print lines matching a pattern
   log       Show commit logs
   show      Show various types of objects
   status    Show the working tree status

grow, mark and tweak your common history
   branch    List, create, or delete branches
   commit    Record changes to the repository
   merge     Join two or more development histories together
   rebase    Reapply commits on top of another base tip
   reset     Reset current HEAD to the specified state
   switch    Switch branches
   tag       Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
   fetch     Download objects and refs from another repository
   pull      Fetch from and integrate with another repository or a local branch
   push      Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
See 'git help git' for an overview of the system.
```

Creating Repository

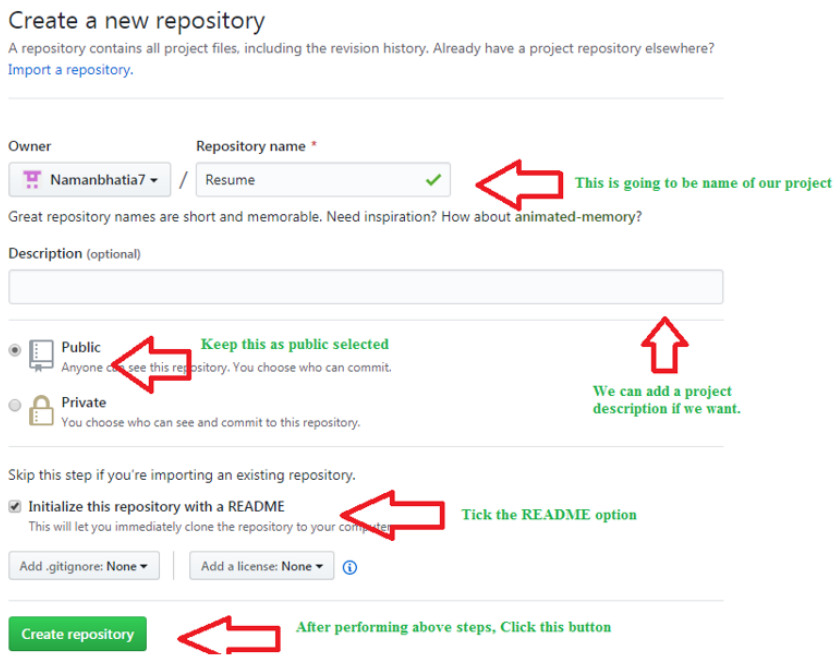
step 1: After successfully setting up GitHub account login to your account. You will see the screen as below.



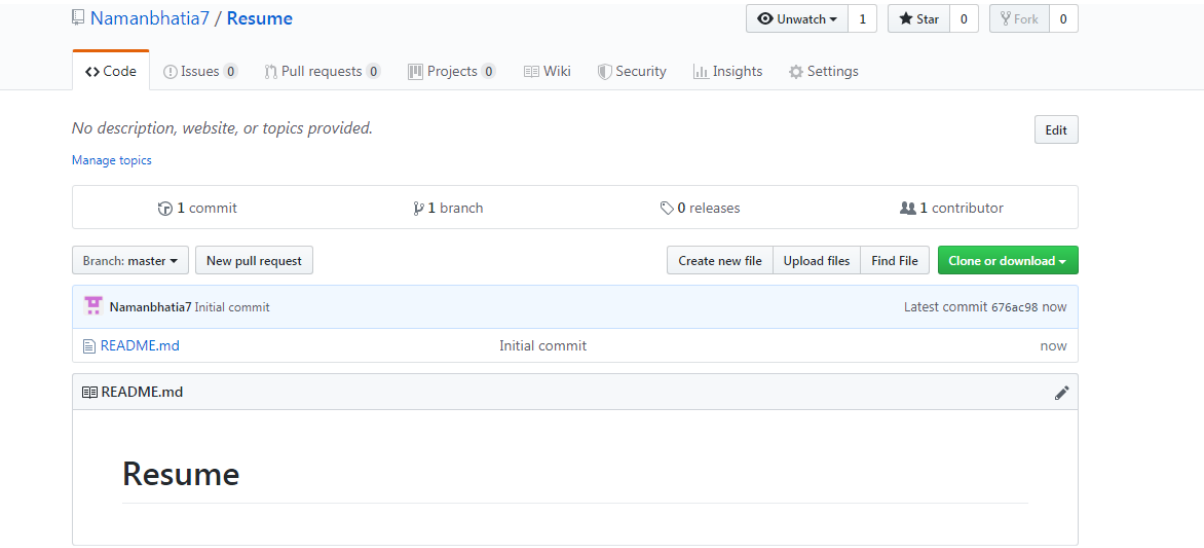
step 2: Click on the new repository option.



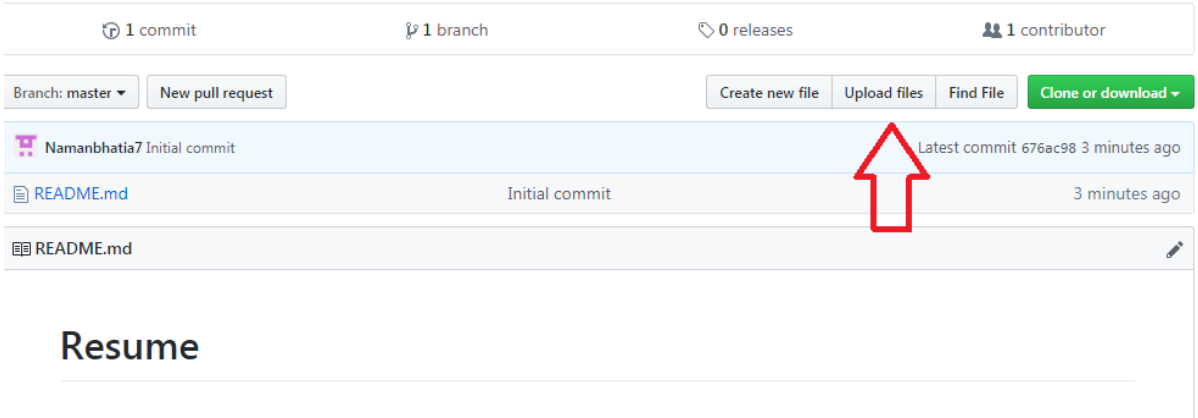
Step 3: After clicking new repository option, we will have to initialize some things like, naming our project, choosing the visibility etc. After performing these steps click Create Repository button.



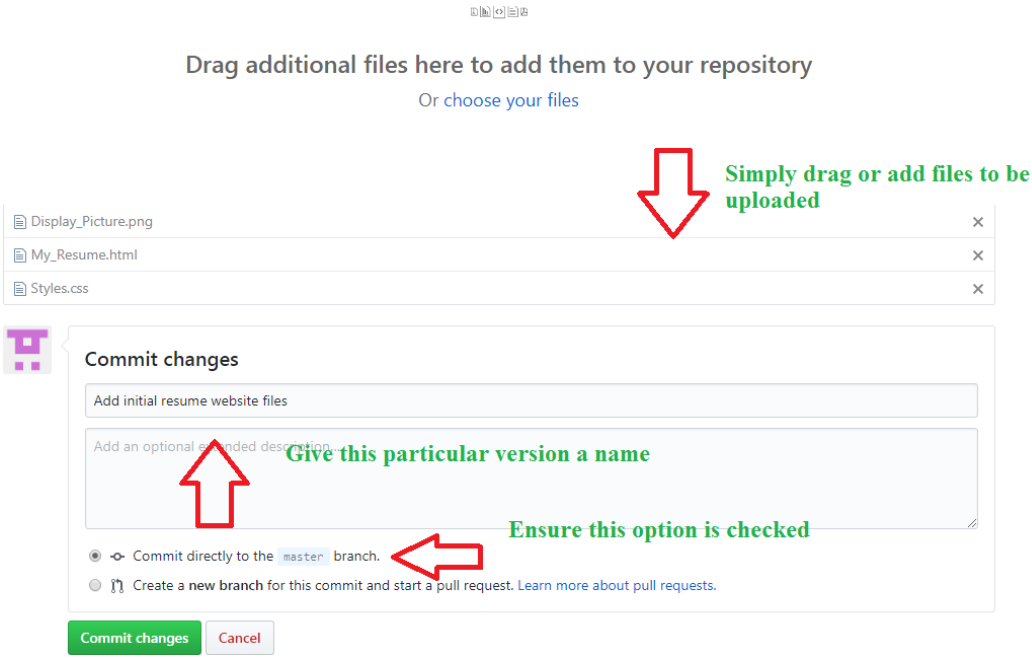
Step 4: After clicking the button, we will be directed to below page. Right now the only file we have is a readme file.



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