

**Source Code Management File**

**Submitted by:  HARSHITA BATRA**

**ROLL NO. - 2110990587**

**Group- 8 - B**

# Index

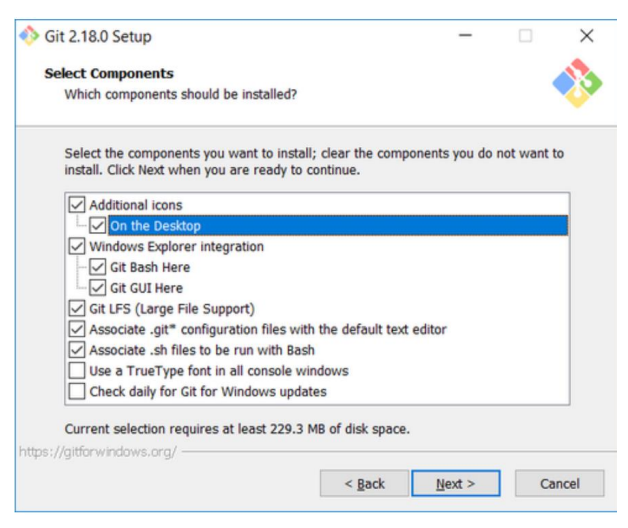
|  |  |  |
| --- | --- | --- |
| **S.No** | **Aim of Experiment** | **Remarks by teacher** |
| 1 | Setting up the git client |  |
| 2 | Setting up GitHub Account |  |
| 3 | Program to generate logs |  |
| 4 | Create and visualize branches in Git |  |

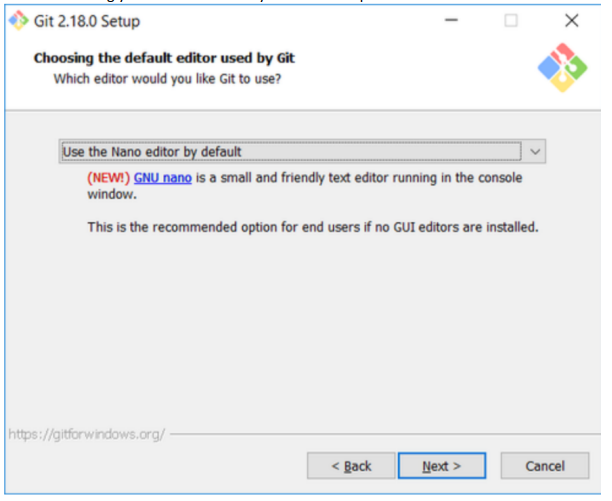
EXPERIMENT – 1

Aim: Setting up the git client.

Git Installation: Download the Git installation program

In the Select Components screen, Windows Explorer Integration is selected as shown:



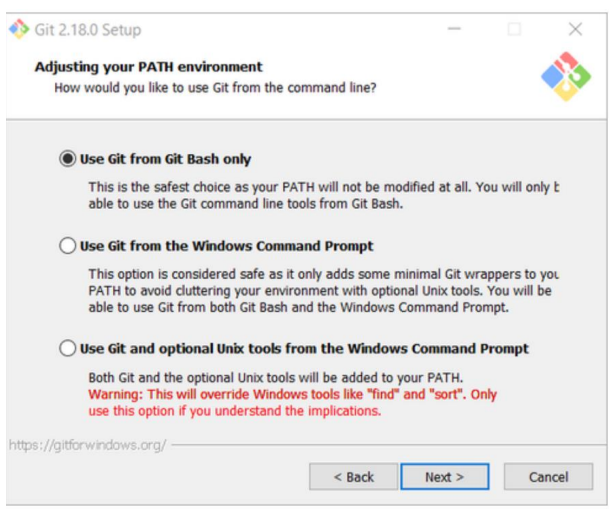


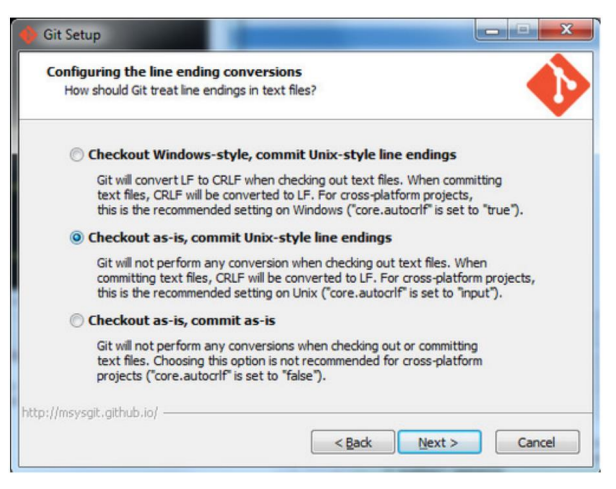
Three options are acceptable In the Adjusting your PATH screen

1. Use Git from Git Bash only: no integration, and no extra command in your command path.

2. Use Git from the windows Command Prompt: add flexibility – you can simply run git from a windows command prompt, and is often the setting for people in industry – but this does add some extra commands.

3. Use Git and optional Unix tools from the Windows Command Prompt: this is also a robust choice and useful if you like to use Unix like commands like grep



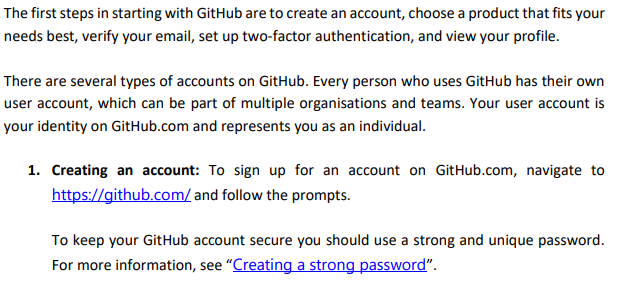


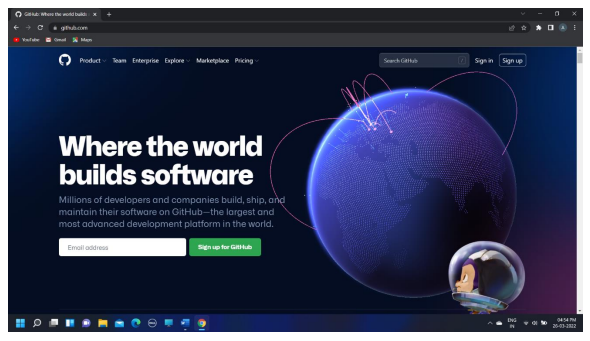
Once Git is installed, there is some remaining custom configuration we must do. Follow the steps below:

1. From within File Explorer, right-click on any folder. A context menu appears containing the commands " Git Bash here" and "GitGUI here". These commands permit you to launch either Git client. For now, select Git Bash here.
2. b. Enter the command (replacing name as appropriate) git config -- global core.excludesfile c:/users/name/. gitignore This tells Git to use the .gitignore file you created in step 2 NOTE: TO avoid typing errors, copy and paste the commands shown here into the Git Bash window, using the arrow keys to edit the red text to match your information.
3. c. Enter the command git config --global user.Email "name@msoe.edu" This links your Git activity to your email address. Without this, your commits will often show up as "unknown login". Replace name with your own MSOE email name.
4. d Enter the command git config --global user.name "Your Name" Git uses this to log your activity. Replace "Your Name" by your actual first and last name. e. Enter the command git config --global push.default simple This ensures that all pushes go back to the branch from which they were pulled. Otherwise pushes will go to the master branch, forcing a merge.

EXPERIMENT – 2

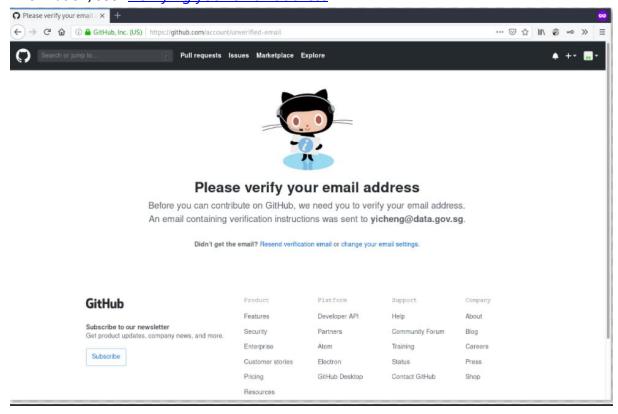
Aim: Setting up the git hub account.



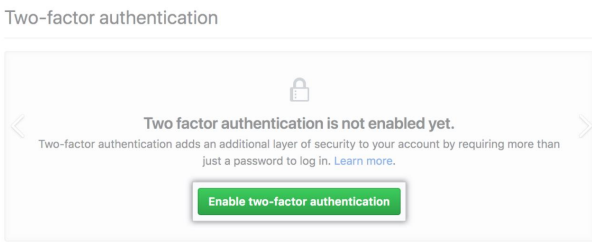


2. Choosing your GitHub product: You can choose GitHub Free or GitHub Pro to get access to different features for your personal account. You can upgrade at any time if you are unsure at first which product you want. For more information on all GitHub’s plans, see “GitHub's products”.

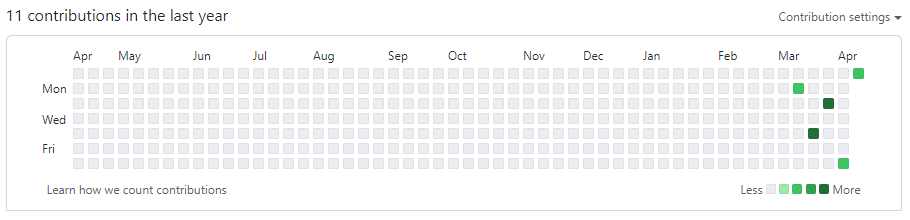
3. Verifying your email address: To ensure you can use all the features in your GitHub plan, verify your email address after signing up for a new account. For more information, see “Verifying your email address.



4. Configuring two-factor authentication: Two-factor authentication, or 2FA, is an extra layer of security used when logging into websites or apps. We strongly urge you to configure 2FA for safety of your account. For more information, see “About two factor authentication."



5. Viewing your GitHub profile and contribution graph: Your GitHub profile tells people the story of your work through the repositories and gists you’ve pinned, the organization memberships you’ve chosen to publicize, the contributions you’ve made, and the projects you’ve created. For more information, see “About your profile” and “Viewing contributions on your profile.”



EXPERIMENT – 3

Aim: Program to generate logs

Basic Git init

Git init command creates a new Git repository. It can be used to convert an existing, undersigned project to a Git repository or initialize a new, empty repository. Most other Git commands are not available outside of an initialize repository, so this is usually the first command you’ll run in a new project.

Basic Git status

The git status command displays the state of the working directory and the staging area. It lets you see which changes have been staged, which haven't, and which files aren't being tracked by Git. Status output does not show you any information regarding the committed project history.

Basic Git commit

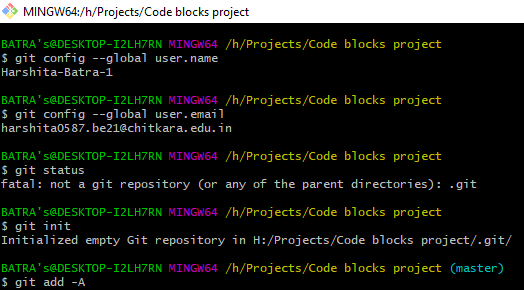
The git commit command captures a snapshot of the project's currently staged changes. Committed snapshots can be thought of as “safe” versions of a project—Git will never change them unless you explicitly ask it to. Prior to the execution of git commit, The git add command is used to promote or 'stage' changes to the project that will be stored in a commit. These two commands git commit and git add are two of the most frequently used

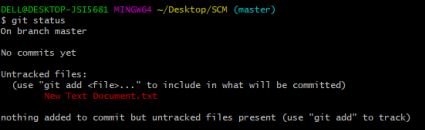
Basic Git add command

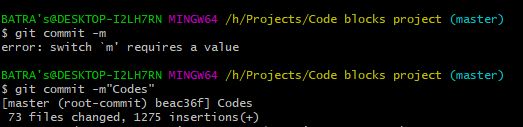
The git add command adds a change in the working directory to the staging area. It tells Git that you want to include updates to a particular file in the next commit. However, git add doesn't really affect the repository in any significant way—changes are not actually recorded until you run git commit.

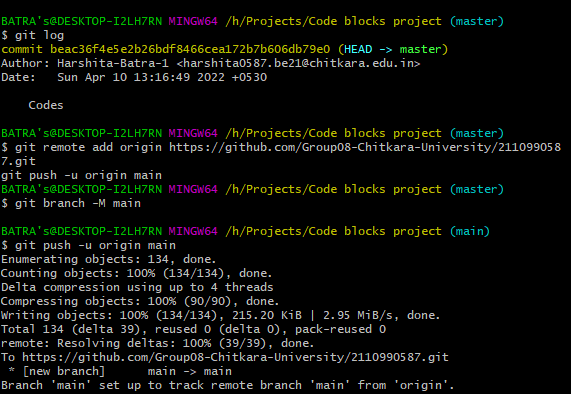
Basic Git log

Git log command is one of the most usual commands of git. It is the most useful command for Git. Every time you need to check the history, you have to use the git log command. The basic git log command will display the most recent commits and the status of the head. It will use as:



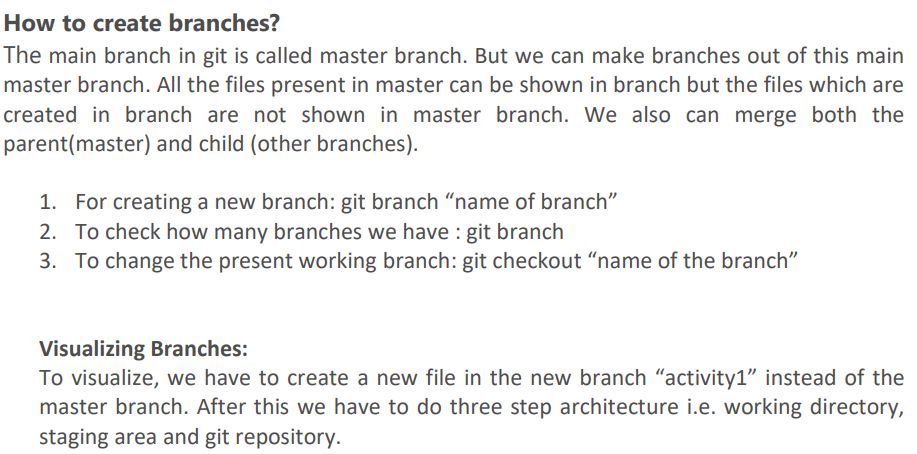


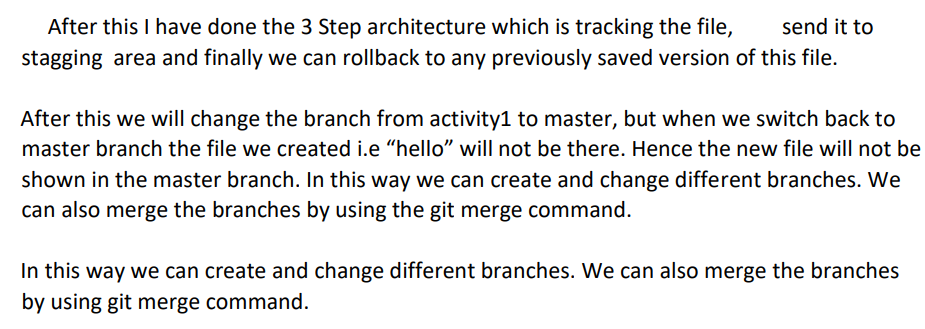




EXPERIMENT – 4

Aim: Create and visualize branches in Git



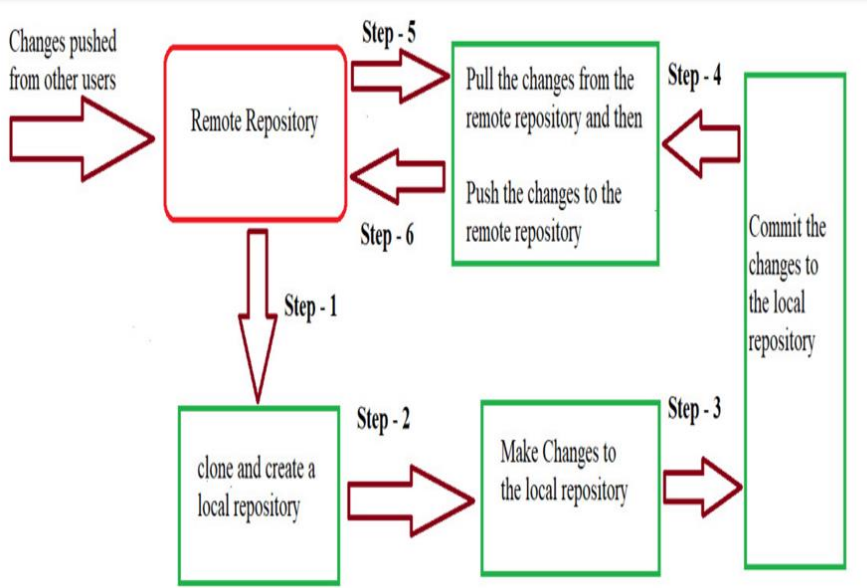


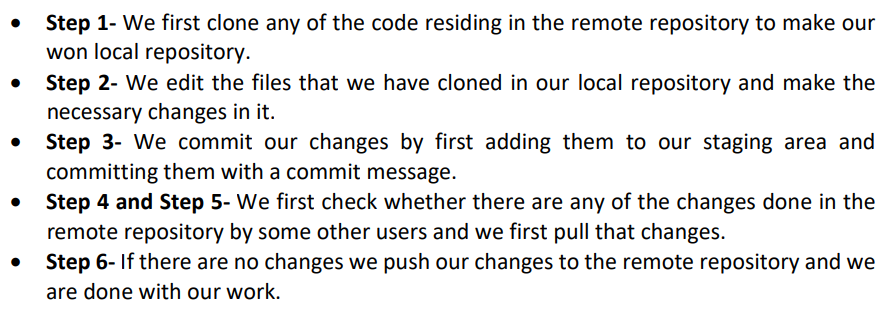


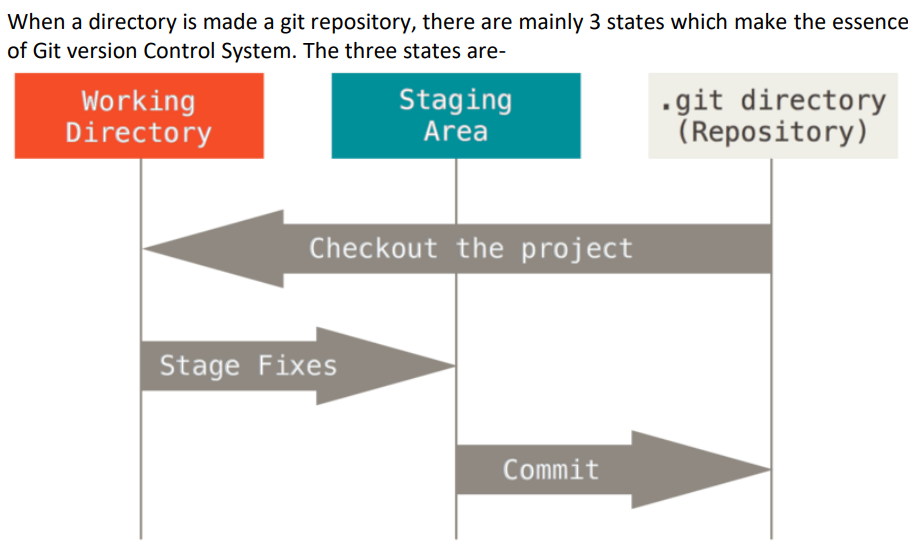
EXPERIMENT – 5

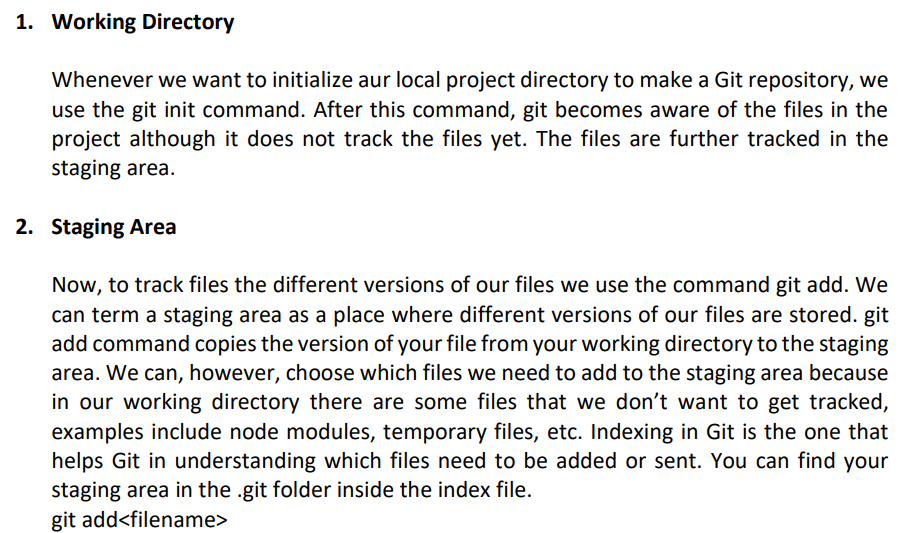
Aim: Git Lifecycle description

Git is used in our day-to-day work, we use Git for keeping a track of our files, working in a collaboration with our team, to go back to our previous code versions if we face some error. Git helps us in many ways. Let us look at the Lifecycle description that git has and understand more about its life cycle. Let us see some of the basic steps that we have to follow while working with Git-









Now since we have all the files that are to be tracked and are ready in the staging area, we are ready to commit aur files using the git commit command. Commit helps us in keeping the track of the metadata of the files in our staging area. We specify every commit with a message which tells what the commit is about. Git preserves the information or the metadata of the files that were committed in a Git Directory which helps Git in tracking files basically it preserves the photocopy of the committed files. Commit also stores the name of the author who did the commit, files that are committed, and the date at which they are committed along with the commit message. git commit -m