SCM-PROJECT
BY

Name-Haraksh Singh
Roll no-2110990541
Group-08

PRACTICAL-1

AIM: Setting up linux git client.

Installing Git on Linux :

From your terminal, enter the following commands:

Step 1: install git using apt-get:

```
m2m-aksh@mind2minds-aksh:~$ sudo apt-get update
[sudo] password for m2m-aksh:
```

After Entering the command- '$ sudo apt-get update' , enter the password of your computer.For installing git type the following command:

```
m2m-aksh@mind2minds-aksh:~$ sudo apt-get install git
```

Step2: verify the installation was successful by typing git --version:

```
m2m-aksh@mind2minds-aksh:~$ git --version
git version 2.34.1
```

Step 3: Configure your Git username and email using the following commands. These details will be associated with any commits that you create:

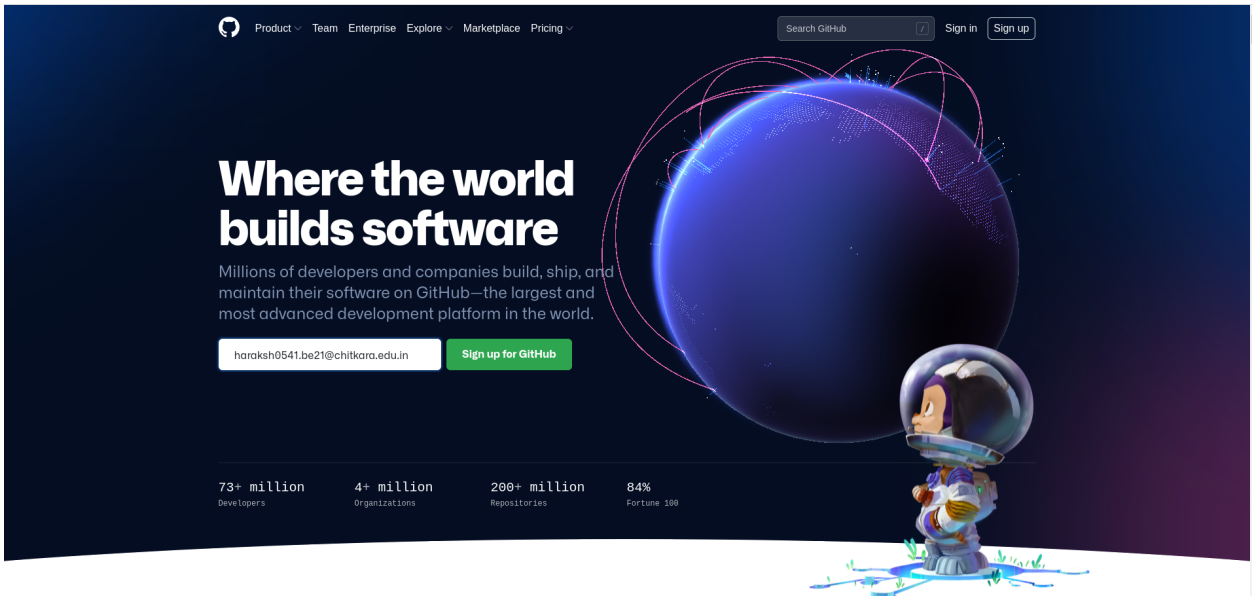
```
m2m-aksh@mind2minds-aksh:~$ git config --global user.name "haraksh singh"
m2m-aksh@mind2minds-aksh:~$ git config --global user.email "haraksh0541.be21@chi
tkara.edu.in"
```

PRACTICAL-2

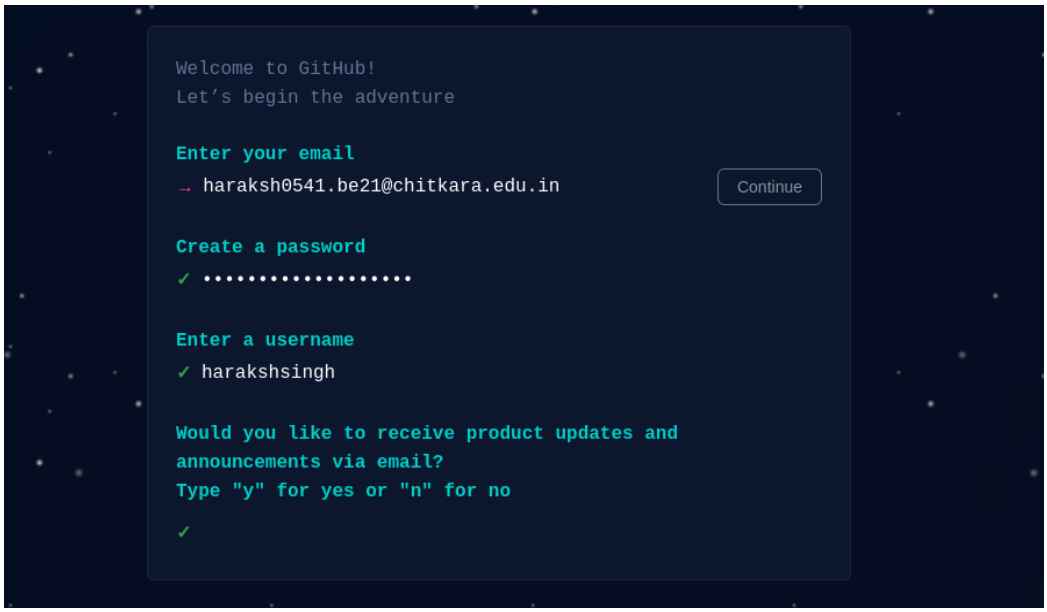
Aim: Setting up a github account.

Steps for creating an account: choose product according to one needs.verify your email, setup two factor authentication,,and view your profile.

1. To set a github account go to the link: https://github.com and follow the prompts.



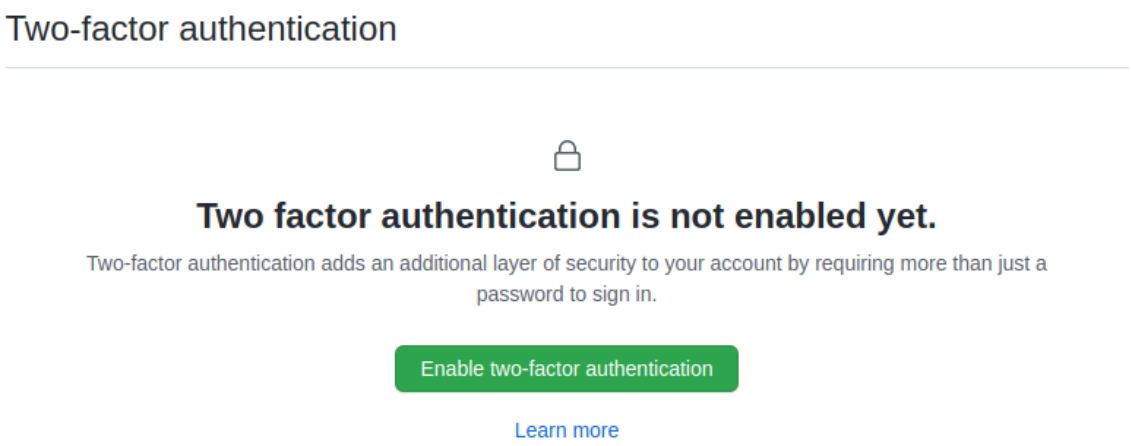
2. In order to secure your github account use a strong and unique password.



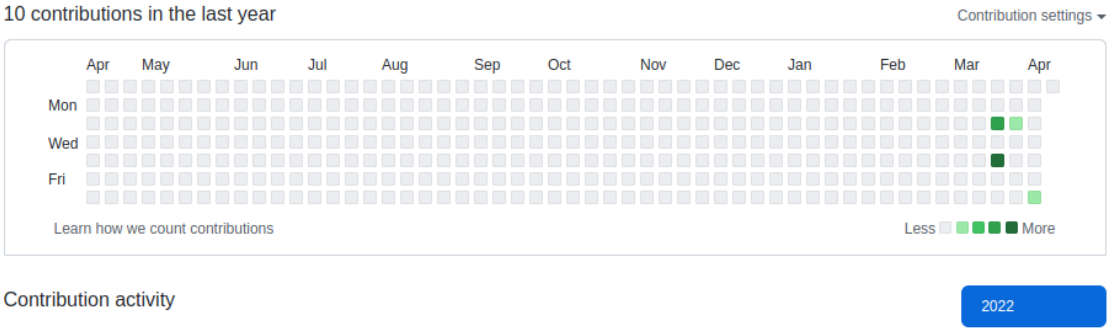
3.**choosing github product:** There are two options for github product-github free and github pro. Both have their own features. One can upgrade at any time.For detailed description of github plans refer to ”https://docs.github.com/en/enterprise-cloud@latest/get-started/learning-about-github/githubs-products”.

4.**verifying email:** Verify your email address and your account is ready.

5. **Configuring two-factor authentication:** Two-factor authentication adds an additional layer of security to your account by requiring more than just a password to sign in.



6.**viewing your contribution graph in your profile:**



Graph tracks your contributions to the github platform datewise.

Repositories:
You can make many repositories and push your source code in that repository. In github Repository is a storage location for storing your files, source code and other resources.A repository is a main folder for each of your projects.

Organizations: you can make an organization in which you can have a team and repositories also and work in collaboration.

PRACTICAL-3

AIM: Generate log.

COMMANDS:

1. Basic git init

   Syntax:

   ```
   m2m-aksh@mind2minds-aksh:~/codeblocks$ git init
   ```

   You have to run this command in the root folder of the files that you want to track. By running this command, an empty git repository is initialized. All other git commands cannot be executed if you do not have a git repository in the working directory.This is the first command for every new project.It can be used to convert an existing, unversioned project to a Git repository or initialize a new, empty repository.

2. Basic git status

   syntax:

   ```
   m2m-aksh@mind2minds-aksh:~/codeblocks$ 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 aren't tracked by git. Status output does not show any information regarding the committed project history.

3. Basic git add command

   syntax:

   ```
   m2m-aksh@mind2minds-aksh:~/codeblocks$ git add array-1.cpp
   ```

   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 affect the repository in any significant way—changes are not actually recorded until you run git commit.

```
m2m-aksh@mind2minds-aksh:~/codeblocks$ git add array-1.cpp
m2m-aksh@mind2minds-aksh:~/codeblocks$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   array-1.cpp

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        2d-array.cpp
        Untitled1
        Untitled1.cpp
        Untitled1.o
        array-1
        array-1.o
        assignment-1
        assignment-1.cpp
        assignment-1.o
        assignment-2
        assignment-2.cpp
```

4. Basic git commit
   Syntax:
   $ git commit -m “message for 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 frequentl used.

5. Basic git log
   Git log command is one of the most common 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 be used as:

```
m2m-aksh@mind2minds-aksh:~/codeblocks$ git log
commit be936ee067562203e504490d55214db758614ac9 (HEAD -> main, origin/main)
Author: haraksh singh <haraksh0541.be21@chitkara.edu.in>
Date:   Tue Mar 22 14:44:56 2022 +0530

    git array-1.cpp
m2m-aksh@mind2minds-aksh:~/codeblocks$
```

PRACTICAL-4

AIM: Create and visualize branches in git

How to create branches?

The main branch in git is called master branch. But we can make branches out of this main master branch. All the files present in master can be shown in branch but the files which are created in branch are not shown in master branch. We also can merge both the parent(master) and child (other branches).

1. For creating a new branch: git branch “name of branch”
2. To check how many branches we have : git branch
3. To change the present working branch: git checkout “name of the branch”

Visualizing Branches:

To visualize, we have to create a new file in the new branch “activity” instead of the master branch. After this we have to do three step architecture i.e. working directory, staging area and git repository.

After this I have done the 3 Step architecture which is tracking the file, sending it to the staging area and finally we can rollback to any previously saved version of this file.

After this we will change the branch from activity1 to master, but when we switch back to master branch the file we created i.e “hello” will not be there. Hence the new file will not be shown in the master branch. In this way we can create and change different branches. We can also merge the branches by using the git merge command.

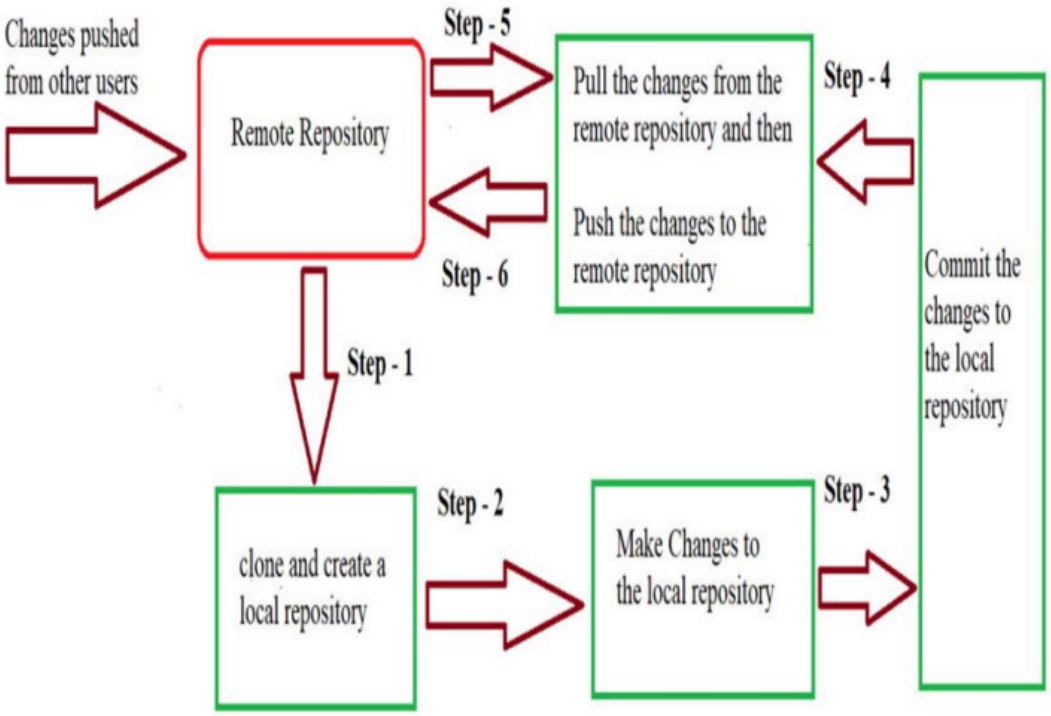
In this way we can create and change different branches. We can also merge the branches by using git merge command.

```
m2m-aksh@mind2minds-aksh:~/various-git-branch-demo$ git branch activity
m2m-aksh@mind2minds-aksh:~/various-git-branch-demo$ git branch
  activity
* master
m2m-aksh@mind2minds-aksh:~/various-git-branch-demo$ git checkout activity
Switched to branch 'activity'
m2m-aksh@mind2minds-aksh:~/various-git-branch-demo$ git status
On branch activity
nothing to commit, working tree clean
m2m-aksh@mind2minds-aksh:~/various-git-branch-demo$ git branch
* activity
  master
m2m-aksh@mind2minds-aksh:~/various-git-branch-demo$ touch hello
m2m-aksh@mind2minds-aksh:~/various-git-branch-demo$ git status
On branch activity
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        hello

nothing added to commit but untracked files present (use "git add" to track)
m2m-aksh@mind2minds-aksh:~/various-git-branch-demo$ git add .
m2m-aksh@mind2minds-aksh:~/various-git-branch-demo$ git commit -m "new file added in activity branch"
[activity 0c8694b] new file added in activity branch
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 hello
m2m-aksh@mind2minds-aksh:~/various-git-branch-demo$ git log
commit 0c8694b0ad4dcb18b2e9d8044f24199e668bbab0 (HEAD -> activity)
Author: haraksh singh <haraksh0541.be21@chitkara.edu.in>
Date:   Sun Apr 10 23:44:26 2022 +0530

    new file added in activity branch

commit 659ee49114bd2778371121b66de6ec3d8ea58030 (master)
Author: haraksh singh <haraksh0541.be21@chitkara.edu.in>
Date:   Sun Apr 10 23:40:39 2022 +0530

    master branch
m2m-aksh@mind2minds-aksh:~/various-git-branch-demo$ git branch
* activity
  master
m2m-aksh@mind2minds-aksh:~/various-git-branch-demo$ ls
2d-array.cpp  array-1.cpp  assignment-2.cpp  assignment-3.cpp  hello
m2m-aksh@mind2minds-aksh:~/various-git-branch-demo$ git checkout master
Switched to branch 'master'
m2m-aksh@mind2minds-aksh:~/various-git-branch-demo$ ls
2d-array.cpp  array-1.cpp  assignment-2.cpp  assignment-3.cpp
m2m-aksh@mind2minds-aksh:~/various-git-branch-demo$
```

PRACTICAL-5

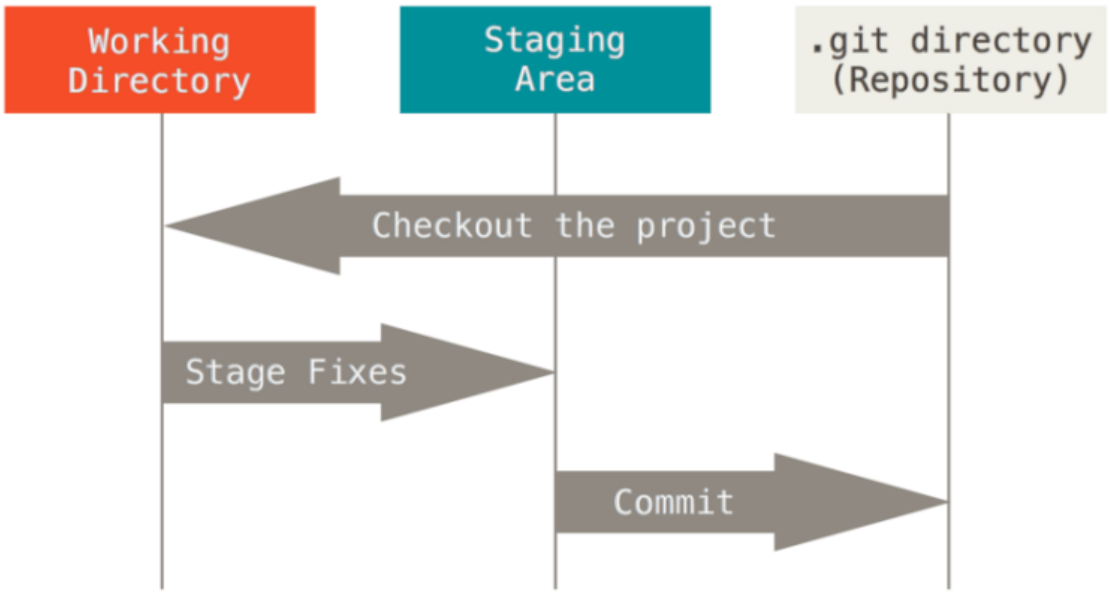
Aim: Git lifecycle description

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-



• **Step 1**- We first clone any of the code residing in the remote repository to make our won local repository.
• **Step 2**- We edit the files that we have cloned in our local repository and make the necessary changes in it.
• **Step 3**- We commit our changes by first adding them to our staging area and committing them with a commit message.
• **Step 4 and Step 5**- We first check whether there are any of the changes done in the remote repository by some other users and we first pull those changes.
• **Step 6**- If there are no changes we push our changes to the remote repository and we are done with our work. Experiment No. 05 Page 15 of 16 CS181 When a directory is made a git

repository, there are mainly 3 states which make the essence of Git version Control System. The three states are



1. Working Directory

Whenever we want to initialize aur local project directory to make a Git repository, we use the git init command. After this command, git becomes aware of the files in the project although it does not track the files yet. The files are further tracked in the staging area.

2. Staging Area

Now, to track files the different versions of our files we use the command git add. We can term a staging area as a place where different versions of our files are stored. git add command copies the version of your file from your working directory to the staging area. We can, however, choose which files we need to add to the staging area because in our working directory there are some files that we don't want to get tracked, examples include node modules, temporary files, etc. Indexing in Git is the one that helps Git in understanding which files need to be added or sent. You can find your staging area in the .git folder inside the index file. git add <file name>.

3. Git Directory

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 <message>