

**A Project  
report on  
“SneakFreak - Sneakers”  
with  
Source Code Management  
(CS181)**

Submitted by

<b>Team Member 1</b>	<b>Priyanshu Vashishtha</b>	<b>2110992074</b>
<b>Team Member 2</b>	<b>Aashirwad</b>	<b>2110992099</b>
<b>Team Member 3</b>	<b>Kuber Bansal</b>	<b>2110992020</b>
<b>Team Member 4</b>	<b>Jasjot Singh</b>	<b>2110992022</b>



**Department of Computer Science & Engineering**

Chitkara University Institute of Engineering and Technology, Punjab

Jan- June  
(2021-22)



Institute/School Name	<b>Chitkara University Institute of Engineering and Technology</b>		
Department Name	<b>Department of Computer Science &amp; Engineering</b>		
Programme Name	<b>Bachelor of Engineering (B.E.), Computer Science &amp; Engineering</b>		
Course Name	<b>Source Code Management</b>	Session	<b>2021-22</b>
Course Code	<b>CS181</b>	Semester/Batch	<b>2<sup>nd</sup>/2021</b>
Vertical Name	<b>Zeta</b>	Group No	<b>G27</b>
Course Coordinator	<b>Dr. Neeraj Singla</b>		
Faculty Name	<b>Mr. Sachendra</b>		

**Submission**

**Name:**

**Signature:**

**Date:**



## Table of Content

<b>S. No.</b>	<b>Title</b>	<b>Page No.</b>
1	Version control with Git	3-5
2	Problem Statement	6
3	Objective	6
4	Resources Requirements – Frontend / Backend	7
5	Concepts and commands, Workflow	8-20

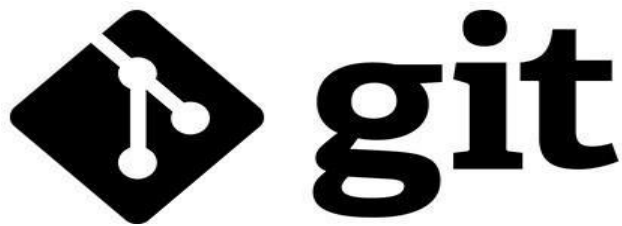
## 1. Version control with Git

### What is GIT and why is it used?

Git is a version control system that is widely used in the programming world. It is used for tracking changes in the source code during software development. It was developed in 2005 by Linus Torvalds, the creator of the Linux operating system kernel.

Git is a speedy and efficient distributed [VCS](#) tool that can handle projects of any size, from small to very large ones. Git provides cheap local branching, convenient staging areas, and multiple workflows. It is free, open-source software that lowers the cost because developers can use Git without paying money. It provides support for non-linear development. Git enables multiple developers or teams to work separately without having an impact on the work of others.

Git is an example of a distributed version control system (DVCS) (hence Distributed Version Control System).



### What is GITHUB?

It is the world's largest open-source software developer community platform where the users upload their projects using the software Git.



### What is the difference between GIT and GITHUB?

GIT VS GITHUB	
GIT	GITHUB
Git is a distributed version control system which track changes to source code over time.	Github is a web based hosting service for Git repository to bring teams together.
Git is a command line tool which requires an interface to interact with the world.	Github is a graphical interface and a development platform created for millions of developers.
It creates local repository to track changes locally rather than store them on a centralized server.	It is open source which means code is stored on a centralized server.
It stores and catalog changes in code in a repository.	It provides a platform as a collaborative effort to bring teams together.

## What is Repository?

A repository is a directory or storage space where your projects can live. Sometimes GitHub users shorten this to “repo.” It can be local to a folder on your computer, or it can be a storage space on GitHub or another online host. You can keep code files, text files, image files, you name it, inside a repository.

## What is Version Control System (VCS)?

A version control system is a tool that helps you manage “versions” of your code or changes to your code while working with a team over remote distances. Version control keeps track of every modification in a special kind of database that is accessible to the version control software. Version control software (VCS) helps you revert back to an older version just in case a bug or issue is introduced to the system or fixing a mistake without disrupting the work of other team members.

## Types of VCS

1. Local Version Control System
  2. Centralized Version Control System
  3. Distributed Version Control System
- I. **Local Version Control System:** Local Version Control System is located in your local machine. If the local machine crashes, it would not be possible to retrieve the files, and all the information will be lost. If anything happens to a single version, all the versions made after that will be lost.
  - II. **Centralized Version Control System:** In the Centralized Version Control Systems, there will be a single central server that contains all the files related

to the project, and many collaborators checkout files from this single server (you will only have a working copy). The problem with the Centralized Version Control Systems is if the central server crashes, almost everything related to the project will be lost.

- III. **Distributed Version Control System:** In a distributed version control system, there will be one or more servers and many collaborators similar to the centralized system. But the difference is, not only do they check out the latest version, but each collaborator will have an exact copy of the main repository on their local machines. Each user has their own repository and a working copy. This is very useful because even if the server crashes we would not lose everything as several copies are residing in several other computers

## 2. Problem Statement

Looking and buying new shoes , sneakers can sometimes become difficult and inaccessible owing to repetitive and distracting advertisements, and occasionally due to the paid policy of these programs.

These websites' user interfaces and user experiences are both poor (UX). Apart from this there are many websites on which you can look for Sneakers but this only factor makes the decision more confusing.

## 3. Objective

PROJECT NAME: SneakFreak

Our goal is to create a web app for our users that will provide them with a variety and trending sneakers for all categories whether it is male, female , kids.

Our prime objective is to create a lightweight online app with a very basic yet classic UI/UX that allows our customers to search for their best and Fitting Shoes/Sneakers without any sort of disturbances like advertisements.

## 4. Resources Required.

Frontend – HTML5, CSS

Backend – NodeJS

### Languages





## 5. Concepts , Commands, Workflow and Discussions.

***Aim: Create a distributed Repository and add members in project team***

- Login to your GitHub account and you will land on the homepage as shown below. Click on the button shown in the menu bar and then click on New Organization.

The screenshot shows the GitHub web interface. At the top, there is a search bar and navigation links for Pull requests, Issues, Marketplace, and Explore. Below this, the repository path 'priyanshu996 / SneakFreak' is displayed, along with a 'Private' label and an 'Unwatch' button. A dropdown menu is open in the top right corner, showing options: 'New repository', 'Import repository', 'New gist', and 'New organization' (which is highlighted in blue). Below the repository header, there are tabs for Code, Issues, Pull requests, Actions, Projects, Security, Insights, and Settings. The 'Code' tab is active, showing a file list with columns for file name, upload action, and commit information. The file list includes 'Mens.html' (uploaded 3 days ago) and 'SNEAKFREAK.zip' (uploaded 4 days ago). On the right side, there is an 'About' section for the repository, showing 'IWT Project', '0 stars', '1 watching', and '0 forks'.

- Set Up Your Organization. Fill Your Organization's Name and Other Details. .

Tell us about your organization

## Set up your organization

**Organization account name \***

SneakFreak - Sneakers ✓

This will be the name of your account on GitHub.  
Your URL will be: <https://github.com/SneakFreak-Sneakers>.

**Contact email \***

priyanshu2074.be21@chitkara.edu.in ✓

**This organization belongs to: \***

☒ **My personal account**  
I.e., priyanshu996

☐ **A business or institution**  
For example: GitHub, Inc., Example Institute, American Red Cross

**Verify your account**

- After creating the repository, we have to create a Repository.

ch or jump to...

Pull requests Issues Marketplace Explore

**SneakFreak-Sneakers** Follow

Overview Repositories Projects Packages Teams People 1 Settings

**Create your first SneakFreak-Sneakers repository.**

A new home for your code where you can start collaborating with your team.

Create a new repository

View as: Public ▼  
You are viewing this page as a public user.  
You can [create a README file](#) visible to anyone.

People

Invite someone

- Create a New Repo by Pressing the New repository button. Fill in the required details.



## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?

[Import a repository.](#)

Owner \*

Repository name \*



SneakFreak-Sneakers ▾

/

Sneakers ✓

Great repository names are short and memorable. Need inspiration? How about [supreme-lamp](#)?

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☒ Add a README file

This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: None ▾

Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

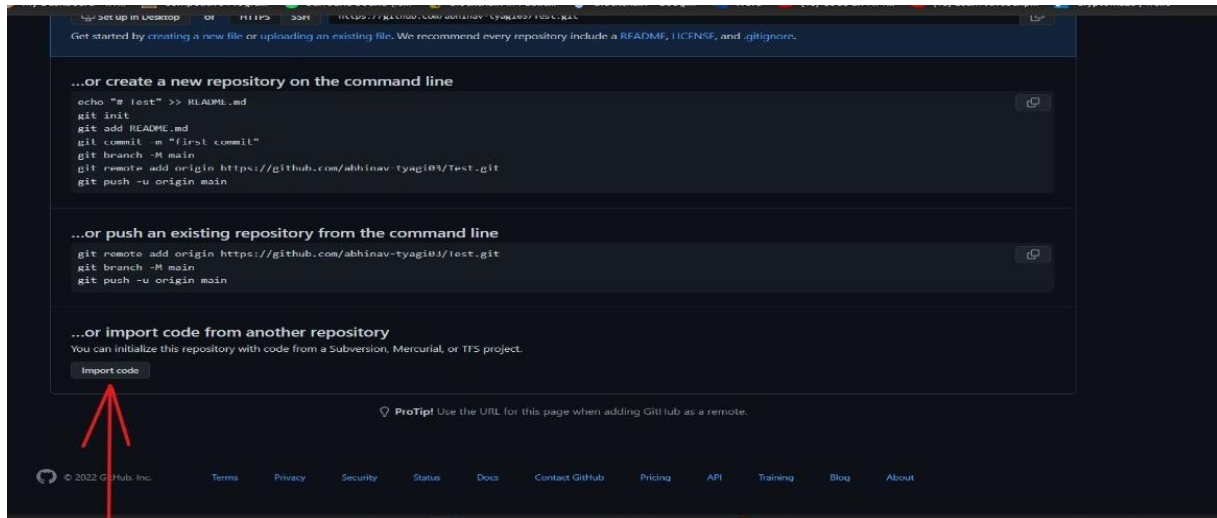
License: None ▾

This will set main as the default branch. Change the default name in SneakFreak-Sneakers's [settings](#).

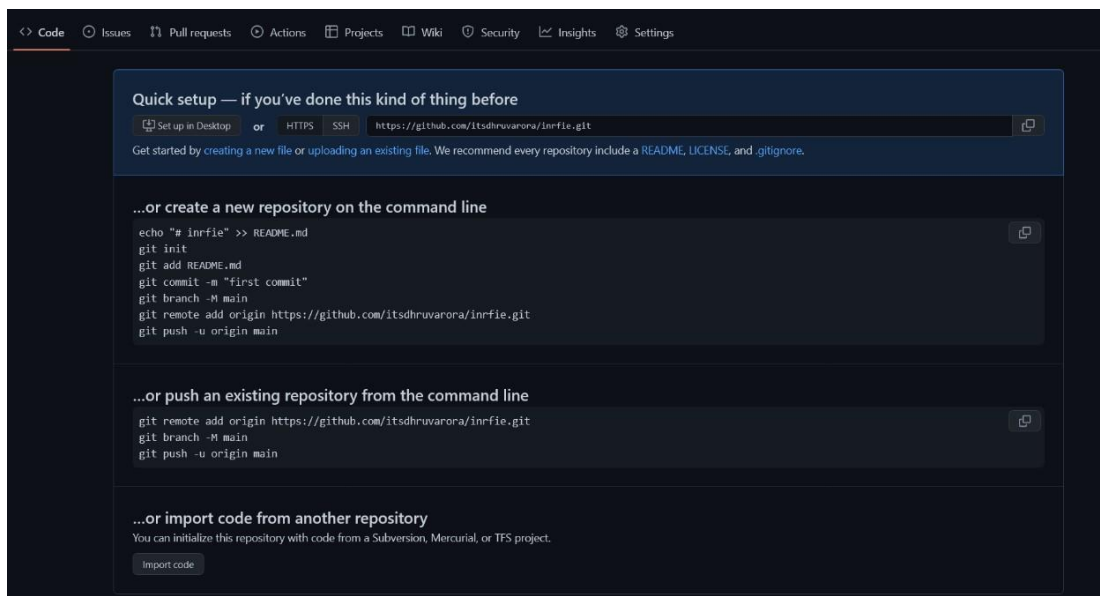
You are creating a public repository in the SneakFreak-Sneakers organization.

Create repository

- If you want to import code from an existing repository select the import code option.



- To create a new file or upload an existing file into your repository select the option in the following box.




- Now, you have created your repository successfully.
- To add members to your repository, open your Organization and select People option in the navigation bar.
- Click on Collaborators option under the access tab.

- To add members click on the add people option and search the id of your respective team member.

## Manage access

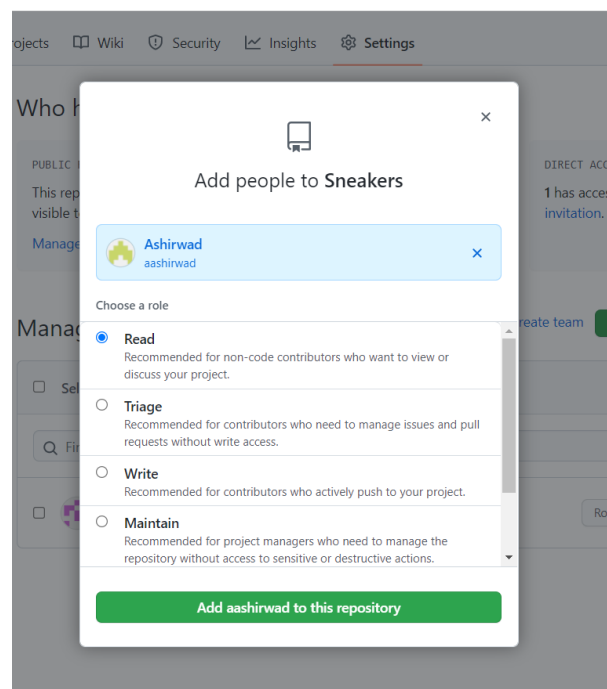
[Create team](#)



**You haven't added any teams or people yet**

Organization owners can manage individual and team access to the organization's repositories. Team maintainers can also manage a team's repository access. [Learn more about organization access](#)

Add people
Add teams



- To accept the invitation from your team member, open your email registered with GitHub.
- You will receive an invitation mail from the repository owner. Open the email and click on accept invitation.
- You will be redirected to GitHub where you can either select to accept or decline the invitation.

- Next, Open the desired Repository in the Organisation. Look and click on Settings -> Collaborators and Teams. Here you can Manage the role of each collaborator.

The screenshot shows the GitHub repository settings page for 'SneakFreak-Sneakers/Sneakers'. The 'Settings' tab is selected, and the 'Collaborators and teams' section is active. The page displays three access methods: PUBLIC REPOSITORY (public and visible to anyone), BASE ROLE (All 1 members can access this repository, Read), and DIRECT ACCESS (3 have access to this repository, 3 invitations). Below these, the 'Manage access' section shows a list of collaborators with their roles and pending invites.

Type	Role
<input type="checkbox"/> Select all	
Find a team, organization member or outside collaborator...	
<input type="checkbox"/> Ashirwad Awaiting ashirwad's response	Pending Invite  Role: Read Remove
<input type="checkbox"/> jasjot16 Awaiting jasjot16's response	Pending Invite  Role: Read Remove
<input type="checkbox"/> kuber Awaiting kuber's response	Pending Invite  Role: Read Remove

## Experiment No. 02

### Aim: Open And Close a Pull Request

- To Open a Pull Request, First of All, it will be required to fork the repository and commit changes into your own.

```

MINGW64/c
Hp@LAPTOP-0R4R5FV2 MINGW64 ~/OneDrive/Desktop
$ cd C:

Hp@LAPTOP-0R4R5FV2 MINGW64 /c
$ mkdir SneakFreak project

Hp@LAPTOP-0R4R5FV2 MINGW64 /c
$ cd SneakFreak project
bash: cd: too many arguments

Hp@LAPTOP-0R4R5FV2 MINGW64 /c
$ git init
Initialized empty Git repository in C:/git/

Hp@LAPTOP-0R4R5FV2 MINGW64 /c (master)
$ git clone https://github.com/SneakFreak-Sneakers/Sneakers.git
Cloning into 'Sneakers'...
remote: Enumerating objects: 17, done.
remote: Counting objects: 100% (17/17), done.
remote: Compressing objects: 100% (15/15), done.
remote: Total 17 (delta 2), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (17/17), 194.24 KiB | 1.24 MiB/s, done.
Resolving deltas: 100% (2/2), done.

Hp@LAPTOP-0R4R5FV2 MINGW64 /c (master)
$
  
```

- Add and commit the changes to the local repository.

```

Hp@LAPTOP-0R4R5FV2 MINGW64 /c (master)
$ git add
Nothing specified, nothing added.
hint: Maybe you wanted to say 'git add .'?
hint: Turn this message off by running
hint: "git config advice.addEmptyPathsSpec false"

Hp@LAPTOP-0R4R5FV2 MINGW64 /c (master)
$ ls
'$mfeDeepRem'/'      'Program Files'/'      Users/
'$Recycle.Bin'/'     'Program Files (x86)'/'  Windows/
'$WinREAgent'/'      'ProgramData/'          hiberfil.sys
'Documents and Settings'@  Recovery/              hp/
DumpStack.log.tmp         SWSetup/               hpswsetup/
Intel/                    SneakFreak/            pagefile.sys
MinGW/                   Sneakers/              project/
OneDriveTemp/             'System Volume Information'/'  swapfile.sys
PerfLogs/                 System.sav@

Hp@LAPTOP-0R4R5FV2 MINGW64 /c (master)
$ cd SneakFreak

Hp@LAPTOP-0R4R5FV2 MINGW64 /c/SneakFreak (master)
$ ls

Hp@LAPTOP-0R4R5FV2 MINGW64 /c/SneakFreak (master)
$ vi index.html

[1]+  Stopped                  vi index.html

Hp@LAPTOP-0R4R5FV2 MINGW64 /c/SneakFreak (master)
$ git add index.html
fatal: pathspec 'index.html' did not match any files

Hp@LAPTOP-0R4R5FV2 MINGW64 /c/SneakFreak (master)
$ |

```



- Use git push origin branch name option to push the new branch to the main repository.

```
Hp@LAPTOP-0R4R5FV2 MINGW64 /c/SneakFreak (master)
$ git commit -m "changed title"
Author identity unknown

*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

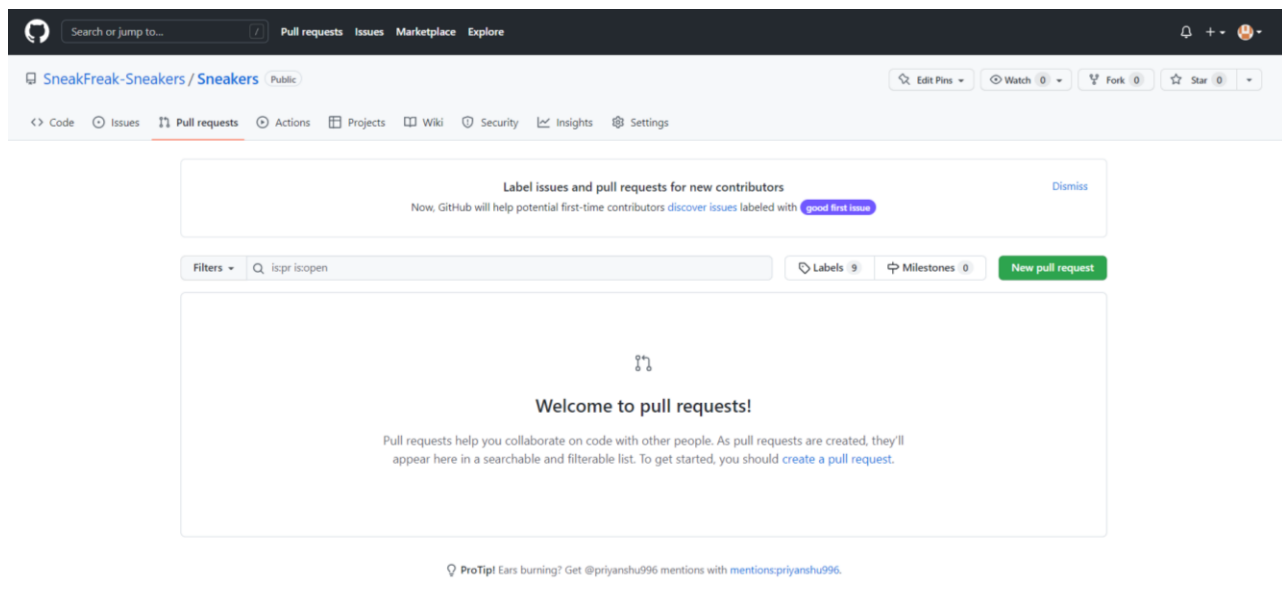
to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'Hp@LAPTOP-0R4R5FV2.(none)')

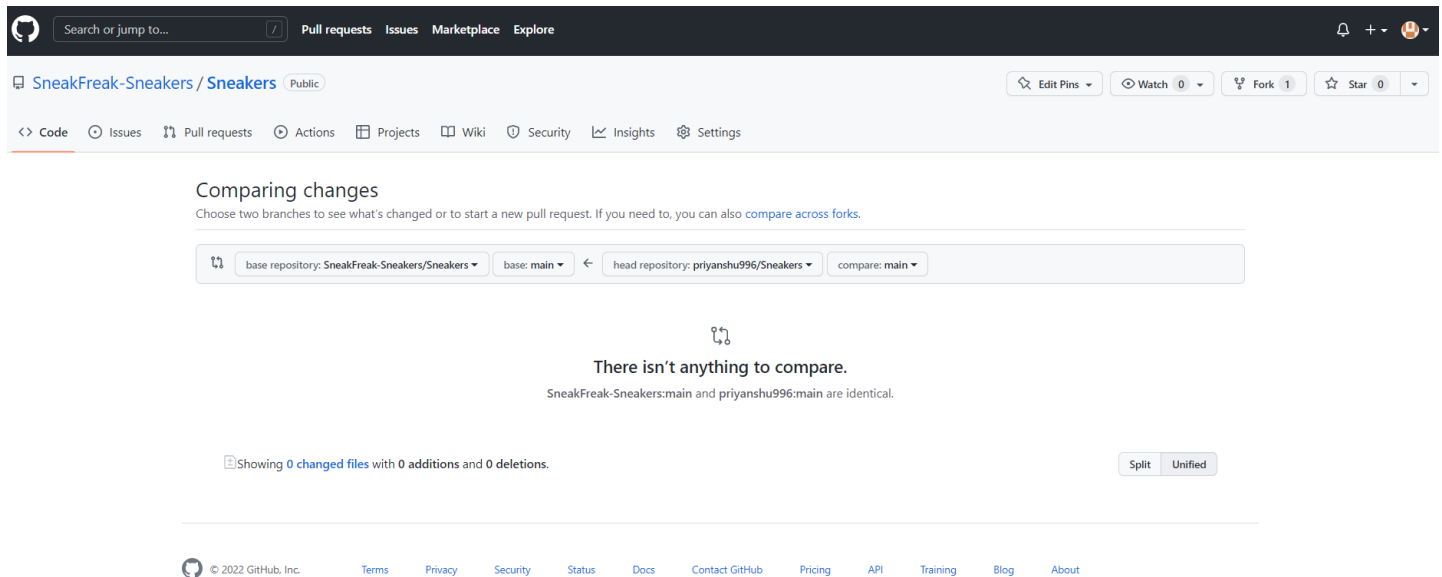
Hp@LAPTOP-0R4R5FV2 MINGW64 /c/SneakFreak (master)
$ git remote add SneakFreak https://github.com/SneakFreak-Sneakers/Sneakers.git

Hp@LAPTOP-0R4R5FV2 MINGW64 /c/SneakFreak (master)
$ git push SneakFreak
error: src refspec refs/heads/master does not match any
```

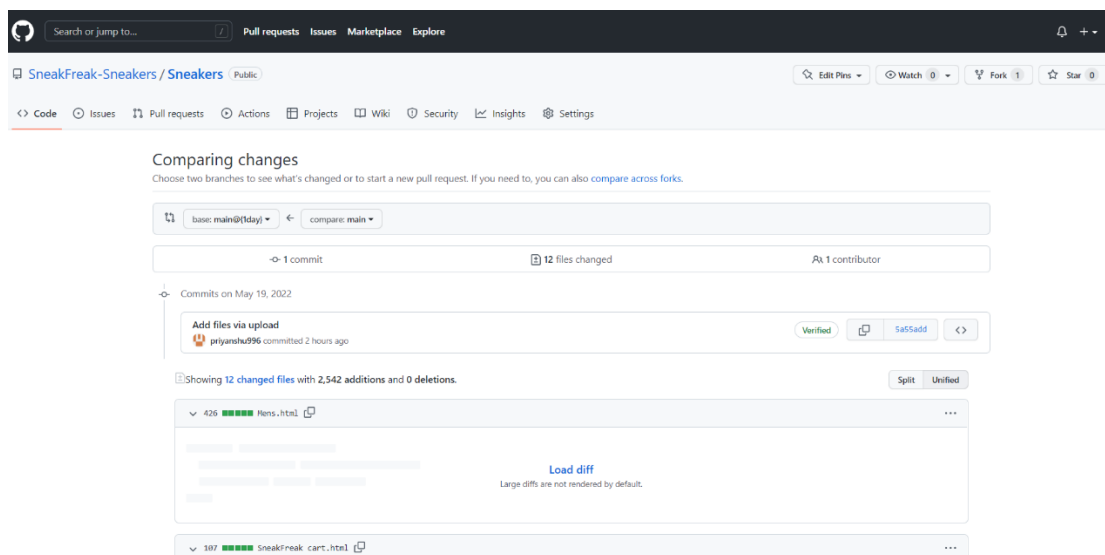
- After pushing new branch GitHub will either automatically ask you to create a pull request or you can create your own pull request.
- To create your own pull request click on pull request option.



- GitHub will detect any conflicts and ask you to enter a description of your pull request.



- After opening a pull request all the team members will be sent the request if they want to merge or close the request.



- If the team member chooses not to merge your pull request they will close you're the pull request.

- To close the pull request simply click on close pull request and add comment/ reason why you closed the pull request.
- You can see all the pull request generated and how they were dealt with by clicking on pull request option.

✕ Clear current search query, filters, and sorts

<input type="checkbox"/>	0 Open ✓ 9 Closed	Author ▾	Label ▾	Projects ▾	Milestones ▾	Reviews ▾	Assignee ▾	Sort ▾
<input type="checkbox"/>	<b>Priyanshu's pull request</b> #9 by Priyans-hu was merged yesterday							
<input type="checkbox"/>	<b>changed title</b> #8 by itsdhravarora was merged yesterday							
<input type="checkbox"/>	<b>Delete app.js</b> #7 by itsdhravarora was merged yesterday							
<input type="checkbox"/>	<b>Add files via upload</b> #6 by itsdhravarora was merged yesterday							
<input type="checkbox"/>	<b>all commit</b> #5 by PRINGGUPTA07 was merged yesterday							
<input type="checkbox"/>	<b>commit 3</b> #4 by PRINGGUPTA07 was merged yesterday							
<input type="checkbox"/>	<b>Add files via upload</b> #3 by itsdhravarora was merged yesterday							
<input type="checkbox"/>	<b>Create sample</b> #2 by kalpana0521 was merged yesterday							
<input type="checkbox"/>	<b>added homepage</b> #1 by itsdhravarora was merged yesterday							

**Experiment No. 03****Aim: Publish and print network graphs**

The network graph is one of the useful features for developers on GitHub. It is used to display the branch history of the entire repository network, including branches of the root repository and branches of forks that contain commits unique to the network.

A repository's graphs give you information on traffic, projects that depend on the repository, contributors and commits to the repository, and a repository's forks and network. If you maintain a repository, you can use this data to get a better understanding of who's using your repository and why they're using it.

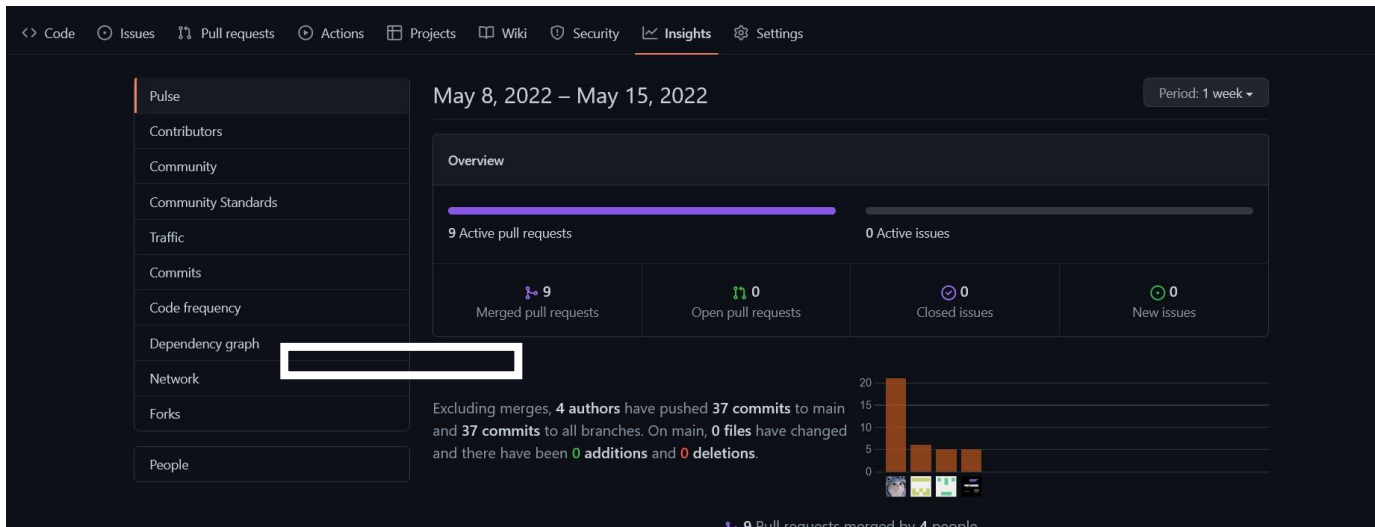
Some repository graphs are available only in public repositories with GitHub Free:

- Pulse
- Contributors
- Traffic
- Commits
- Code frequency
- Network

**Steps to access network graphs of respective repository**

1. On GitHub.com, navigate to the main page of the repository.
2. Under your repository name, click Insights.

3. At the left sidebar, click on Network.



You will get the network graph of your repository which displays the branch history of the entire repository network, including branches of the root repository and branches of forks that contain commits unique to the network.

