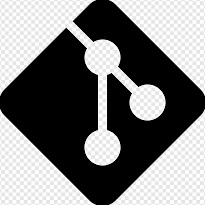
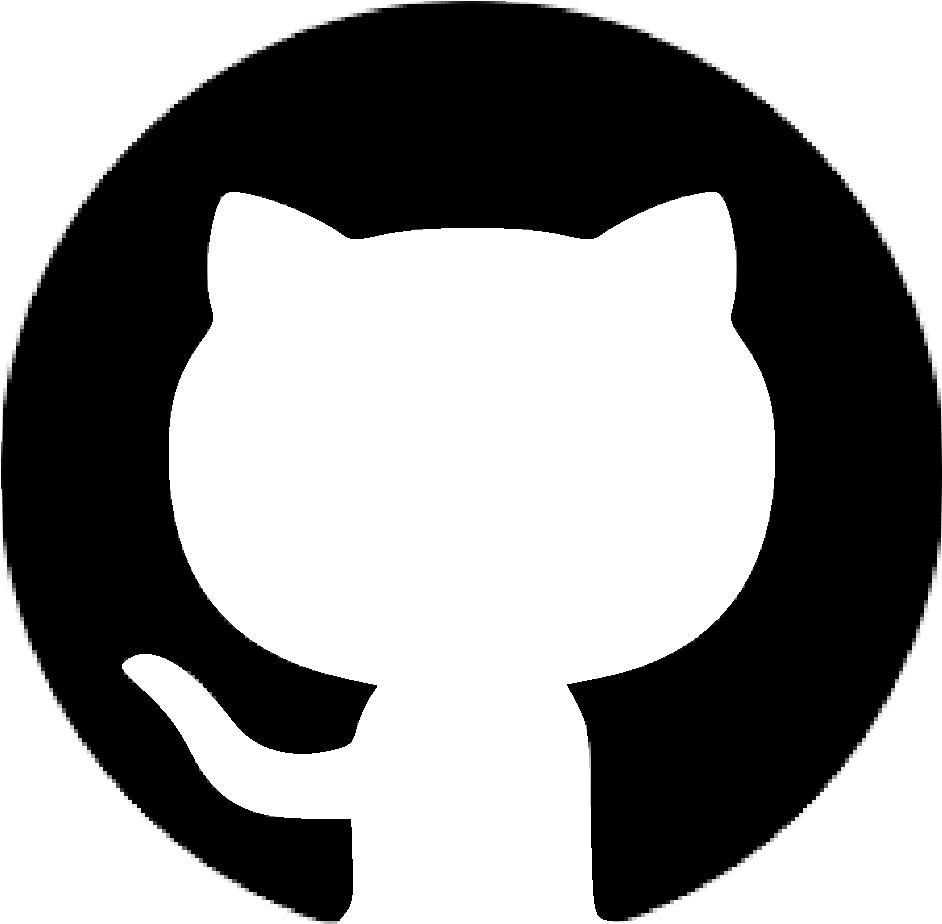
**Source Code Management**

**Course Code: CSE 2015**

**Slot: L14 L15**



**Name: Yashwanth S R**

**SEN No.: A86605224241**

**Faculty: Dr Monit Kapoor**

# Lab Session 1: Git Fundamentals

## Computer

A **computer** is a device that performs calculations, whether they are logical or mathematical.

## Program/Code

A **program** (or **code**) is a set of coded instructions that a computer executes to perform specific tasks.

## Need for Managing Source Code

Modern applications, such as Spotify, consist of multiple programs working together on both the frontend and backend to deliver a smooth user experience.

Regular updates are essential for:

* **Fixing Bugs:** Quickly resolving errors that may occur.
* **Improving UI/UX:** Enhancing the user interface and overall experience.
* **Optimising Performance:** Addressing and refining issues for better performance.

For programmers, effective management of source code is crucial because:

* It ensures that all files remain in context throughout the lifecycle of the program.
* It facilitates collaboration, allowing multiple developers to work together on a shared codebase.

## Tools for Source Code Management

1. **Git:**

A version control system that runs locally on your computer. Git helps track changes and manage versions of your project.

1. **GitHub:**

A global, cloud-based platform that hosts Git repositories, enabling developers to share, collaborate, and contribute to projects from anywhere in the world.

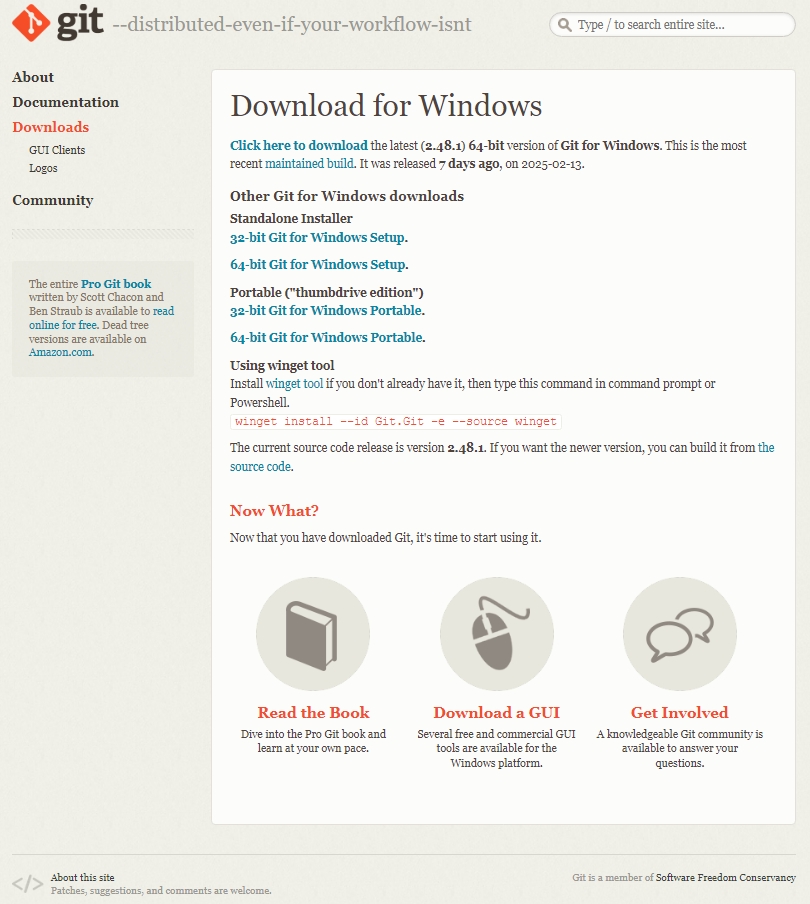
## Version

A **version** in version control represents a snapshot of your project at a specific moment in time. This snapshot allows you to review, revert, or compare changes made throughout the development process.

# Lab Practical 1

## Installing Git Using Microsoft Edge

Step 1: Visit section 1.5 of pro git document and navigate to windows section



Step 2: Install by clicking 64-bit for windows setup (if Ur laptop supports 64-bits):

**Description**: Downloads git for your laptop.

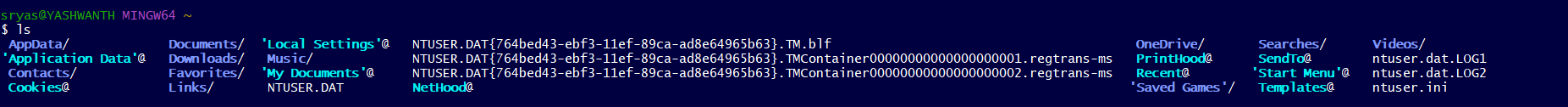
Step 3: Click on download button and click all yes and next buttons without changing anything!

**Description:** Installs git bash or git to your laptop.

## 2.Basic CLI Commands

## 1) Command: ls

**Description:** Lists all files and directories in the current directory.



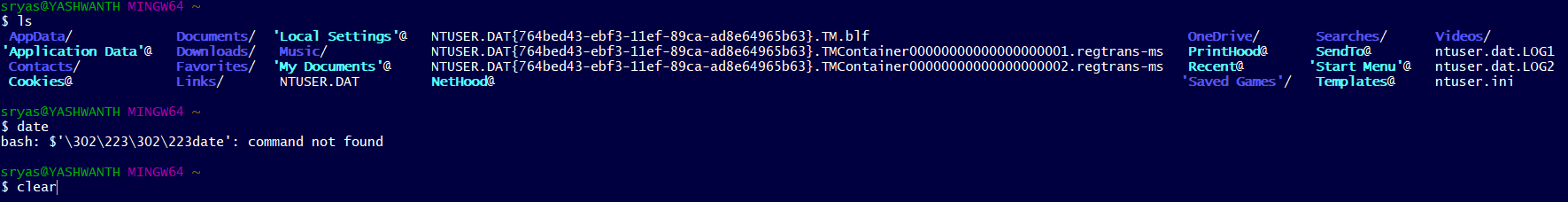
## 2) Command: date

Description: shows the current date and time in a standard format



## 3) Command: clear

Description: The clear command in the CLI is used to clear all the current text and output displayed in the terminal window.



## 4) Command: time

Description: The time command in the CLI is used to measure the execution time of a command or program.



**5) Command**: **rm hello.txt**

**Description:** Removes the file hello.txt from the current directory. 

## 6) Command: cat hello.txt

Description: The cat command (short for **concatenate**) is used to display the contents of a file.

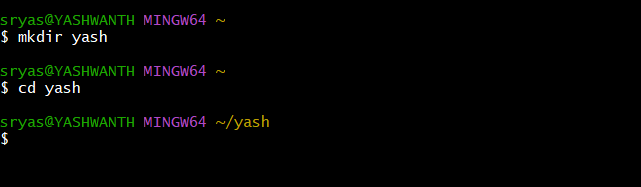


**8) Command: mkdir yash**

Description:To create a new directory named yash

**8)Command: cd yash**

**Description:** Changes the current working directory to the yash directory.



# 3. Vim Text Editor

## 1) Command: vi hello.txt

**Description:** Opens (or creates) the file hello.txt in the Vim text editor.





## 2) Command: i (Insert Mode)

**Description:** Enters insert mode in Vim to allow text input.





## 3) Command: esc

Description: Used to exit insert mode



## 4) Command: :wq

**Description:** Saves the changes and exits the Vim editor.





# 4. Git Commands

## 1. Command: git - - version

**Description:** The git --version command is used to check the installed version of Git on your system.

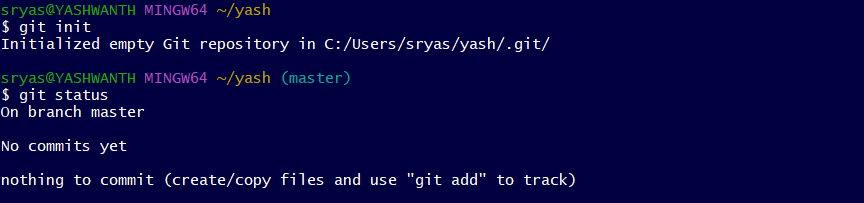


The below files are not currently tracked by git



## 2. Command: git init

**Description:** Initializes a new Git repository in the current directory.

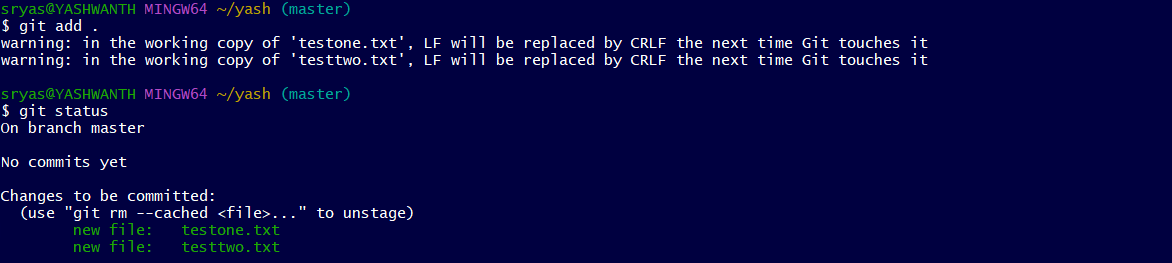


**4. Command: git status**

**Description:** Displays the current status of the working directory and staging area.

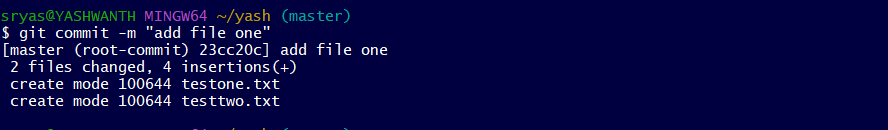
## 3.Command: git add .

**Description:** Adds our txts to the staging area in preparation for a commit.



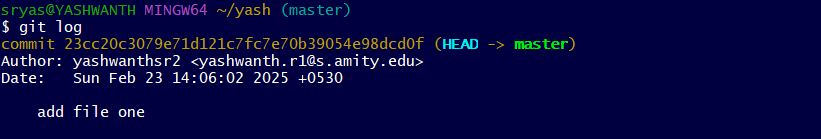
## [5.Com](http://5.Com/)mand:git commit -m "add file one"

**Description:** Commits the staged changes with the message "add file one”



## 6.Command: git log

**Description:** Displays the commit history of the repository.



## 7. Command: git clone

Description: to obtain a copy of an existing Git repository

## 8. Command git config --global user.name “yashwanthsr2”

Description: used to set up user name which will be linked to future commits



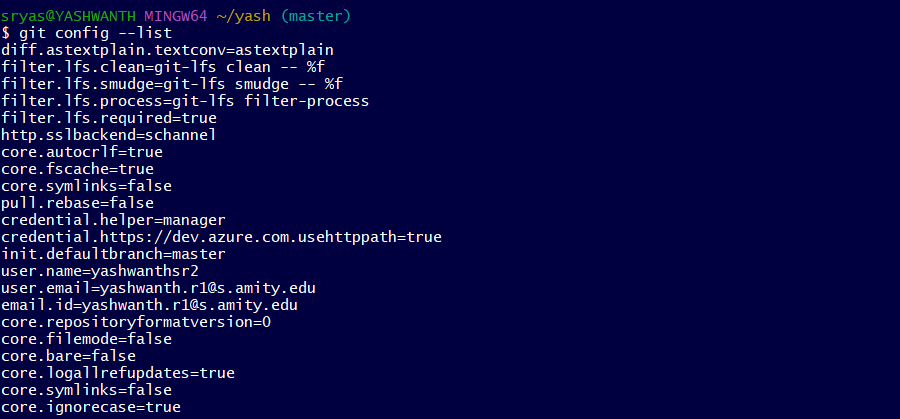
## 9. Command git config --global email.id “yashwanthr.

Description: used to set up email Id which will be linked to future commits.



## 10. Command git config - - list

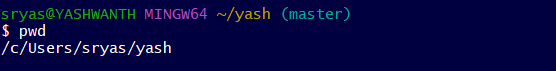
Description: used to view all the settings



# Lab Session 2: Git Commands

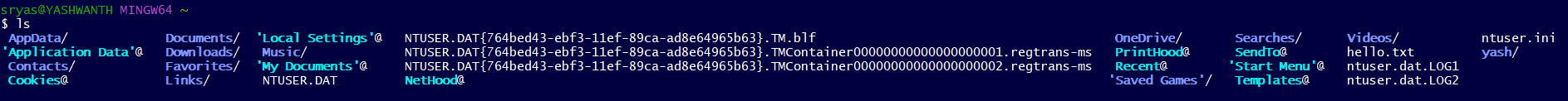
## 1. Command pwd

Description: returns the present working directory



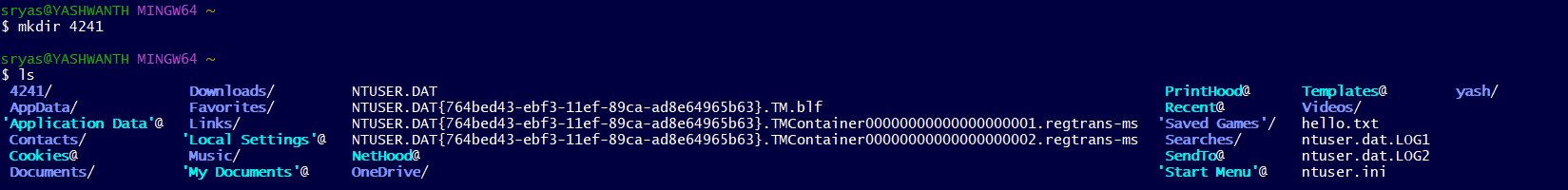
## 2. Command ls

Description: lists all files and folders in the current directory



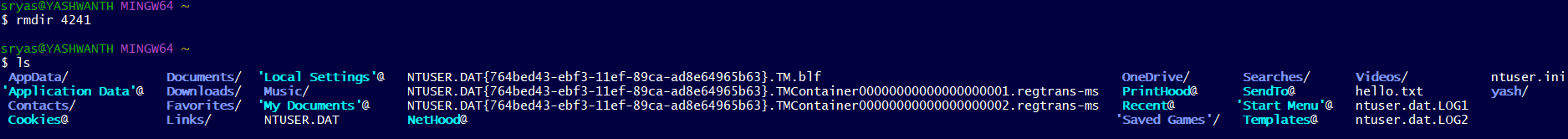
## 3. Command mkdir

Description: used to to make new directory/folder



## 4 .Command rmdir

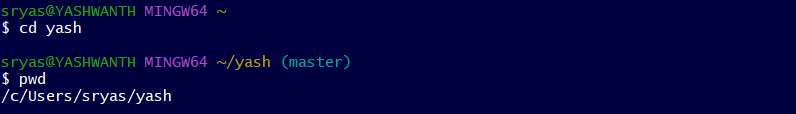
Description: used to remove a directory



## 

## 5. Command cd

Description: used to change current directory



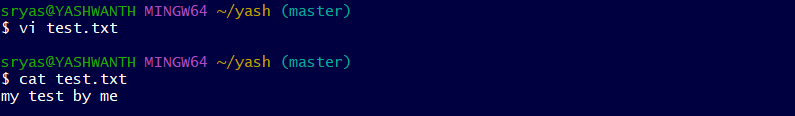
**6. Command cd ..**

Description: used to exit the current sub directory

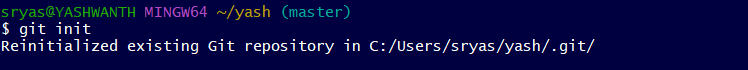


# Git commit

## Step 1: create a file in the present working directory and add content

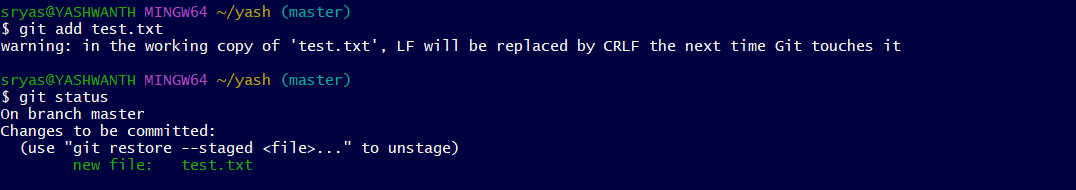


## Step 2: using git init initialise (if it is firstly not installed) a hidden git repository for tracking the files iam already installed so its reinstalled

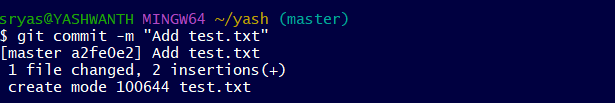


**Step 3: using git add move the file to staging area**

## Step 4: check git status for confirmation

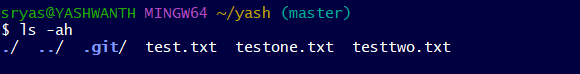


## Step 5: commit the file to a local repository



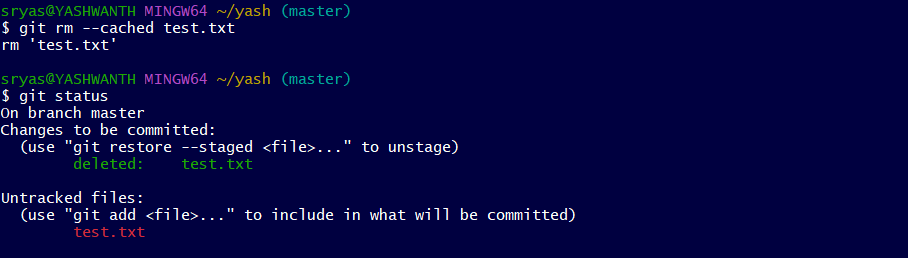
## Command ls-ah

Description: used to check hidden files in a directory



## Command git rm --cached <file>

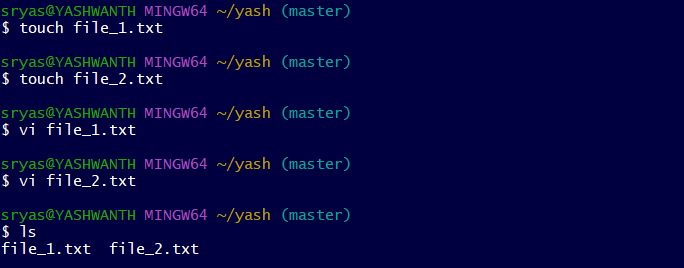
Description : used to remove file from staging area



# Lab Session 3: Git Diff

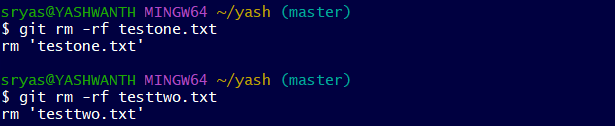
## Command touch <file\_name>

Used to create a file without any content ( iam firstly created some more files ) now more two files are added named file\_1.txt , file\_2.txt.



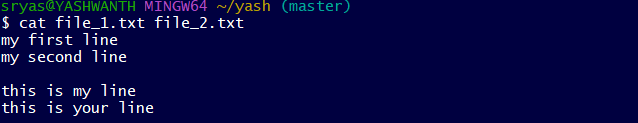
## Command git rm -rf <file\_name>

Used to remove a file from git tracking

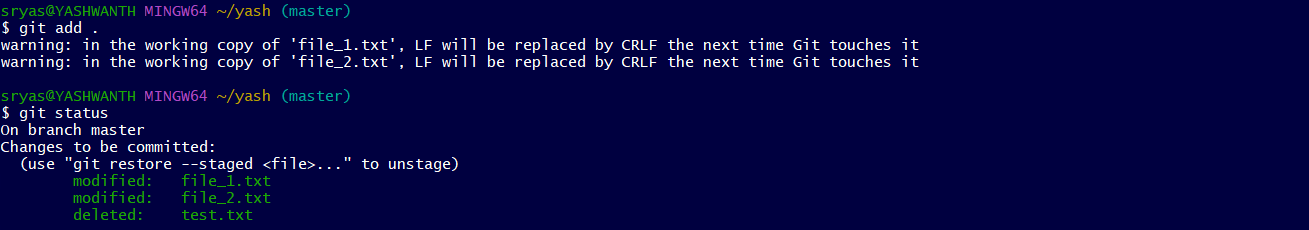


## Task 1: make two commits in a directory

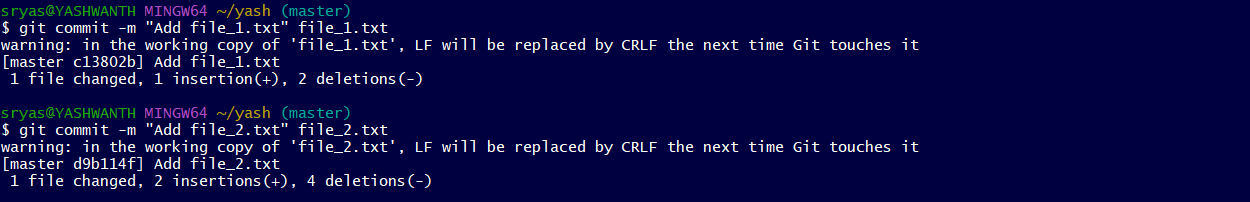
### Step 1: create two files with content in a directory



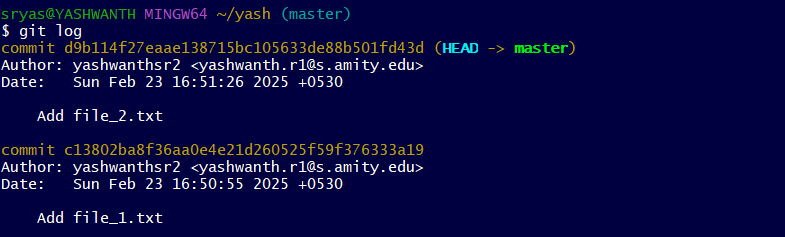
### Step 2: use git add. Command to add those files in the staging area



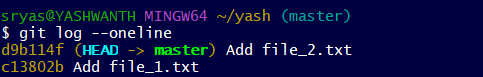
### Step 3: use git commit -m to commit both files to local repository



### Step 4: use git log command to verify the commit history

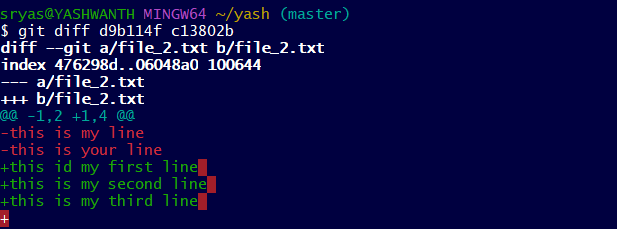


### Step 5: use git log --oneline for generating shorter commit id



## Task 2: compare two commits in a directory

### Use git diff along with the commit id generated from git log - - online

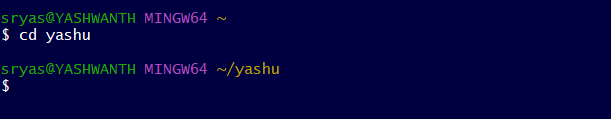


**Lab session 4 : Git Diff**

### Same as 3rd session we going to compare to commits of two codes python or it may be c programming codes

1. **Command : cd yashu**

**Description :** changes the current working directory to the yashu directory.



1. **Command : touch <codes name>**

**Description :** The touch command is used to create an empty file update the time spam of an existing file



3.**Command : vi <file name>**

**Description :** vi command is used to open and edit file using the vim text editor









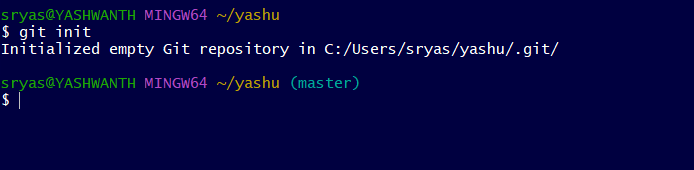
1. **Command : cat <file name>**

**Description :** The cat command is used to display the contents of a file .



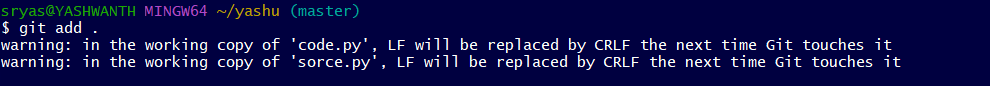
1. **Command : git init**

**Description :** git init is used to inistalise a new git repository in a project dictory.



1. **Command : git add .**

**Description :** git add command in git bash is used to stage (new, modified, or deleted files) for the next commit

****

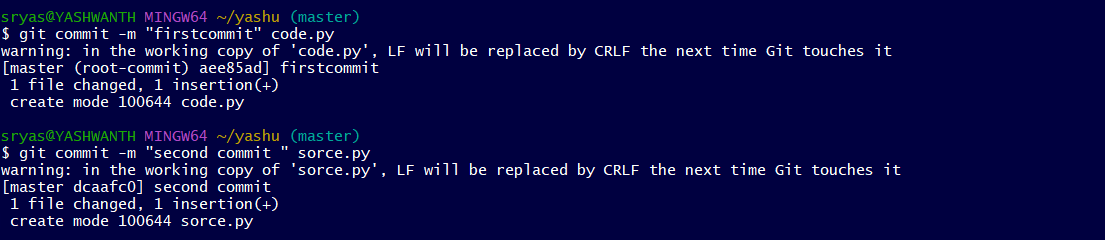
1. **Command : git status**

**Description :** git status command is used to show the current state of the working directory and staging area



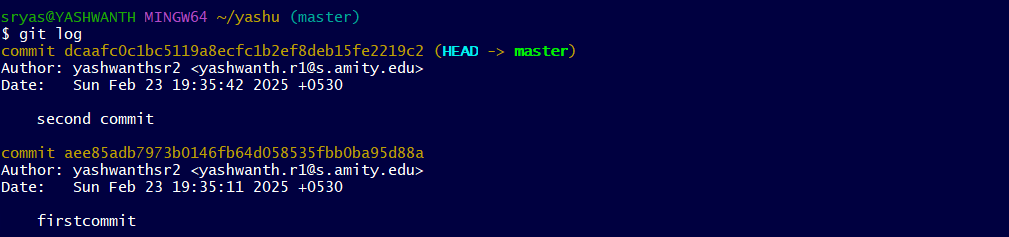
7.**Command : git commit -m “our ref name” < filename>**

**Description :** it is used to save staged changes with a descriptive message ina single step.



1. **Command : git log**

**Description :** It is used to display the commit history of a repository.



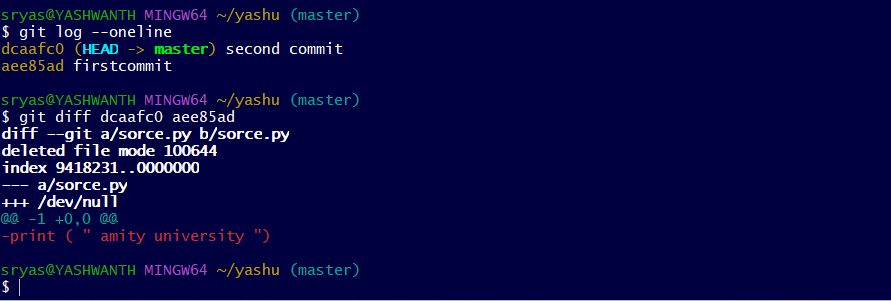
**10)command : git log --oneline**

**Description :** It is used to display a compact commit history with one commit per line or its make it in very simplest form



**11)Command : git diff**

**Description :** Use git diff along with the commit id generated from git log - - oneline



12. Command: git remote add “origin”

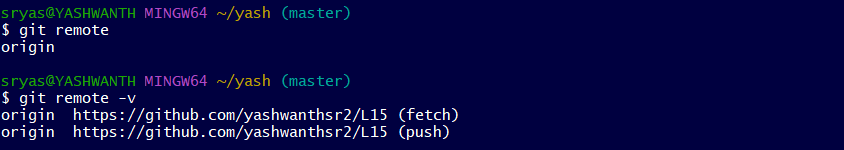
### Description: To connect with the Users GitHub account.



### 

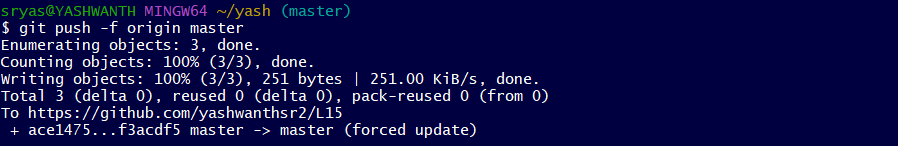
### 13. Command: git remote

### Description: To check the status of the repositories connected with the Users account.



14. Command: git push -u “origin” master

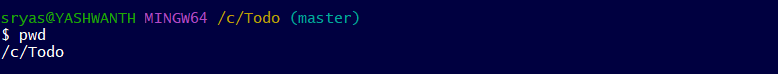
### Description: To push all the files to the Users account.



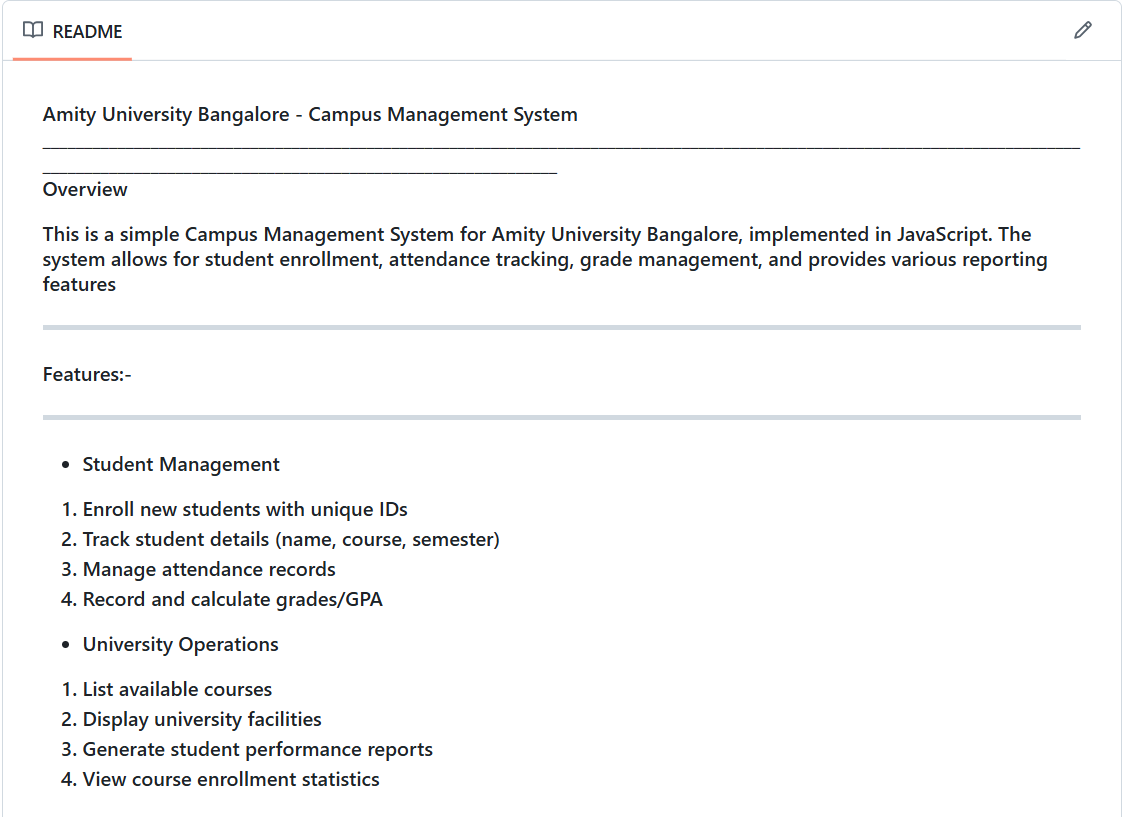
### SCM PROJECT

### The project is to make a repository in our own git hub account and also in git bash then add our project files then commit it and push those all fies to git hub account and access all 4 team-mate’s repositories, fork it, clone it, make some changes and merge them.

### Go to the directory on your computer



### The initial commit README.md has been commited in git hub account itself



### Create a file and type the code in it the add it commit it

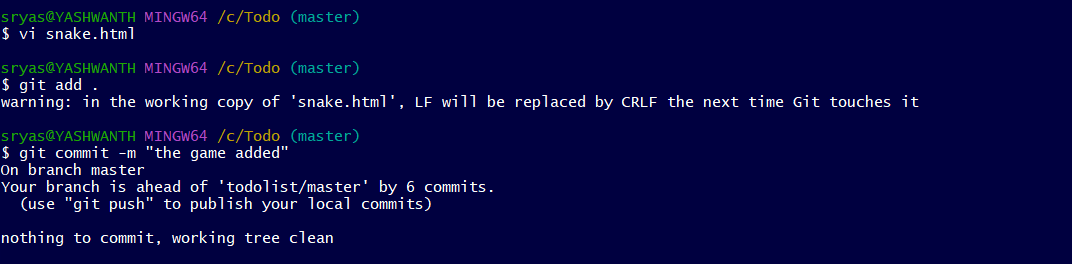
### 

### Like this create the same type our project files and same way add them and commit them

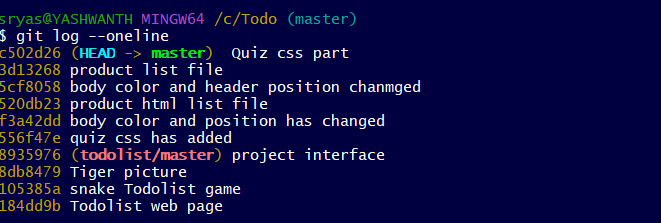
### 

### 

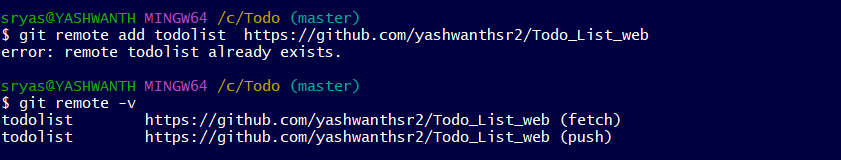
### 



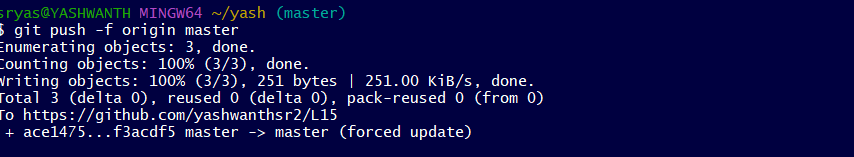
### Using “Git log –oneline” to check all commits



### Using git remote to add GitHub repository

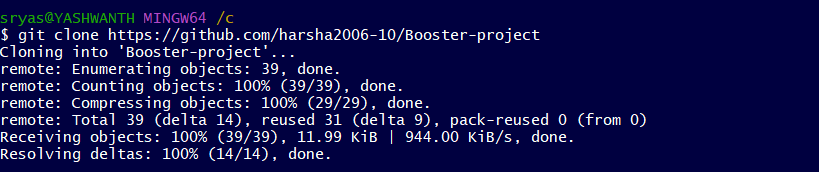


### use git push origin master to push all files to our git hub account

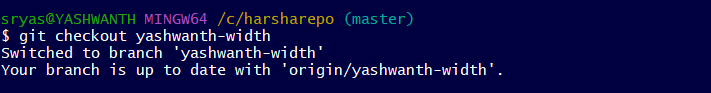


### \*(Now we will clone the repository of the rest of my Team Members, make some changes in their repositories, and send pull requests.)

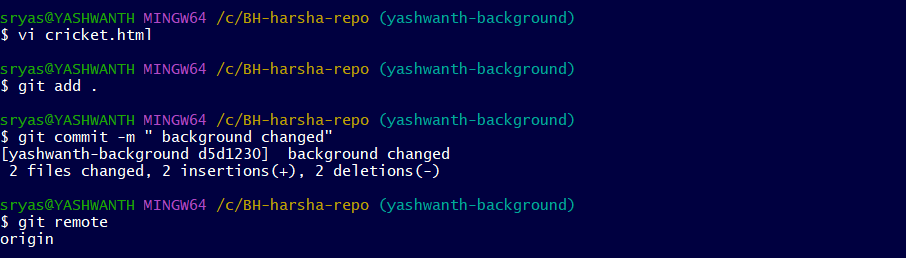
### 9) Fork the repo that your friend pushed the files and clone it in git bash and it will be saved in the name of forked repo

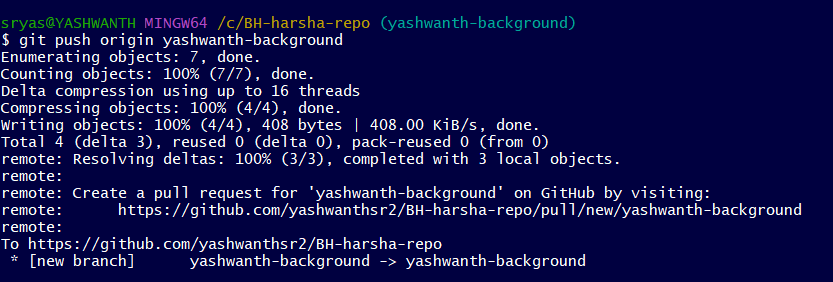


### Create a new branch and check out to it



### 11) Make some changes in the repository. | Add the edited files. | Commit the files. | Push it to the Forked Repository for pull request.





### As the same we have to do with our other team members also

### 12)Finally after all the changes made and pull requests we use “git pull” to sync all the changes in our local folder

