MODULE 1 DOCUMENT

- **Project setup:**

Git checkout master

What it Does:

`git checkout master` command is used to switch your working directory to the `master` branch.

Why it's Used:

When you're working with multiple branches in a Git project, you may want to:

- **Move back to the main code (`master`)**
- **Merge new code into `master`**
- **Test or deploy from `master`**

This command helps you exit your current branch and move into the `master` branch.

I understand there are more branches ,we set all to one called master,so it gives flow of project

Feed back - I dont know meaning for its up to date.

2. pull origin master

What it Does :

`git pull origin master` means:

> "From the **remote repository** named `origin`, take the latest changes from the `master` branch and **merge** them into my **local current branch**."

This is a **two-in-one** command:

1. `git fetch` – fetches the latest code from GitHub.

2. `git merge` – merges that code into your local branch.

Why it's Used:

- To **update your local code** with the latest changes made by others (or yourself) on GitHub.

- Before starting new work, or before pushing your changes, it's good practice to run this so your local branch is **up to date**.

I learned- we need to work on the updated branch,we need to get updated code from the master athunala merging process nadakuthu

Feedback-repository name vanthu puthusa iruku

git checkout -b your-assignment-branch-name

What it is,Technical Meaning:

This command does two things at once:

Creates a new branch called your-assignment-branch-name

Switches you to that new branch immediately

This is a shortcut for:

git branch your-assignment-branch-name
git checkout your-assignment-branch-name

Why it's Used:

When you want to start new work (like a new feature, assignment, or bug fix), you create a new branch to keep your changes separate from the main code.

This keeps the master or main branch clean and safe.Later, you can merge this branch into master after testing.

I understand-each branch is created separately for separate work then the branches are merged with master it is the process of creating new branch and naming it.

Feedback-good explanation

- **Install Dependencies**

```
cd app/frontend
```

Technical Meaning:

> **"Change directory" into the `frontend` folder** (which contains your frontend code like React or HTML/CSS).
>
> I learned-terminalah where i was it is the answer so i am in frontend ,if i give any modifications in frontend in my project it is done here after using this command.

Feedback-cd frontend is wrong,cd app/frontend is correct.not definition given for this command.

```
npm install
```

Technical Meaning:

This command will:

> **Read the `package.json` file** and **install all the dependencies** (like React, Axios, Tailwind, etc.) that your project needs.

It will create a `node_modules/` folder (which holds all installed libraries).

I understand-step you project to all tools needed

Feedback-good explanation

`cd ../backend`

Technical Meaning:

"Go one folder back (`..`) and then enter the `backend` directory."

I understand-go one level back if I am in frontend,then switch to backend

`python -m venv venv`

Technical Meaning:

Create a **Python virtual environment** inside the current folder (named `venv`).

- `venv` = virtual environment (safe Python space)
- `python -m venv venv` → This creates a folder `venv/` with Python + pip isolated for this project

I understand- it is a separate space for our project python code saving
feedback-no t definition for this command is given

`source venv/bin/activate`

Technical Meaning:

Activate the virtual environment so that any `pip install` goes into this `venv` only.

After activation:

- `(venv)` prefix will appear in terminal
- Python and pip will point to this isolated environment

I understand-it is used to activate venv,then when we install dependencies it is installed in venv,not system.
Feedback-not definition given

pip install -r requirements.txt

Technical Meaning:

This command tells Python's package manager (**pip**) to:

> **Read the file `requirements.txt`**, and install **all the Python packages** listed inside it.

Why it's Used:

- Ensures **everyone on your team uses the same versions** of libraries.
- Makes it easy to **recreate** the same environment on a new system/server.

I learned-it install all requirements neede for a backend
Feedback-not definition given

`cd app/backend`

- **Technical Meaning**
  "Change directory into the `app/backend` folder" — where your Flask project files (like `app.py` or `main.py`) are located.

I understand-it moves to backend folder.
feedback-I dont know why suddenly we running backed,some times frontend

`flask run`

Technical Meaning:

This command starts a **local development server** using Flask, and your backend API becomes live at:

I not understand well

Feedback-project flow is ok,but definition lacks

Frontend Implementation

Prompt is good,I received output,but I faced while i try to type it is in background white colour,initally I faced signup/login failed issue

Feedback-frontpage is created I am happy,but initially I dont know how to check output

`cd frontend`

Technical Meaning:

Move into the `frontend` folder where your React or other frontend code lives.

`npm run dev`

Technical Meaning:

Run the **development server** defined in `package.json` (most likely `vite`, `webpack`, or `next`).

I understand it is a used to generate link ,which moves to output page.
Feedback- defenition not given

`git status`

Technical Meaning:

This command shows the **current state of your working directory** and your Git stage area.

It tells you:

- Which files have been changed
- Which files are staged for commit
- Which files are not tracked by Git yet

I understand-it gives update about codes
Feedback-not mention what does “status” means here

`git add .`

Technical Meaning:

This command tells Git to **stage (prepare) all** the modified or new files in the current folder (and subfolders) for the next commit.

- `.` = means "everything in this folder"
- After this, files are marked as ready for `git commit`

I understand-it makes all the files ready which are modified
Git commit-note locked and saved.
Feedback-good

git commit -m "Assignment X: Brief description of changes"

Technical Meaning:

- This command **records (saves)** your staged changes in Git with a **message**.
  The -m flag stands for **message** – it's a short summary of what you changed.

This will save your changes with that description. Later, if you or your team see the Git history, this message will help everyone understand what you changed.
Good

git checkout master

Technical: Switches to the master branch on your local system

going from your personal page to the **main notebook**

git pull origin master

Technical: Gets the latest updates from the **remote master** branch (like GitHub) and updates your **local master**.

`git merge assignment-1-feature`

Technical: Combines the code from `assignment-1-feature` branch **into your master** branch.

`git push origin master`

Technical: Uploads your updated **master branch** from local to the **remote (GitHub)**.
Send everything online
feedback - it is hard to understand firstly

Overall feedback-defenition for some commands neede,with simple english.

- S.Jayashree