## 

Week 2 Git Push and Pull

## **📘 Week 1 Recap – Git Essentials (Completed)**

### **🧱 1. Initialize a Local Repository**

*git init*

* Creates a .git folder in your current directory
* Begins tracking your files using Git

✅ *Use this in any new or existing project folder*

### **🕵️ 2. Check File Status**

git status

* Shows the current state of your files:  
  + Red = untracked
  + Green = staged
  + White = unchanged

✅ *Use this before every commit*

### **3. Stage Files for Commit**

git add .

# or

git add filename.txt

* Moves changes into the **Staging Area**
* You choose which changes will be committed

### **4. Commit Your Changes**

git commit -m "Added homepage layout"

* Saves a **snapshot** of your work with a message
* Commits represent checkpoints in your project

✅ *Keep messages short but clear*

### **🔍 5. View Commit History**

git log

# or simplified:

git log --oneline

* Shows previous commit history
* Can be used to locate specific changes

### **🧪 6. Basic File Operations (Via Bash)**

touch filename.txt # Create file

mkdir myfolder # Create folder

rm file.txt # Remove file

* Helps manage content directly in the terminal

### **🌿 7. Create & Switch Branches (Previewed)**

git branch feature-1

git checkout -b feature-1

* Helps isolate new features without affecting main
* You’ll go deeper into branching in Week 3

## **📦 Week 2: Remote Collaboration and GitHub Integration**

### **🌐 1. Create a Remote Repository on GitHub**

Go to GitHub → New Repository

**Don’t add a README** if you already have one locally

Copy the repo **URL (HTTPS or SSH)**

### **🔗 2. Link Local Repo to GitHub (Remote)**

git remote add origin https://github.com/yourname/project.git

git remote -v

* Connects your local repo to GitHub
* origin is the default alias for your GitHub repo

### **📤 3. Push Your Project to GitHub**

git push -u origin main

* First-time push adds all commits to GitHub
* The -u flag sets origin/main as the default upstream

📌 *Now your project is officially backed up in the cloud*

### **📥 4. Pull Latest Changes**

git pull origin main

* Retrieves updates from GitHub
* Merges them into your local code
* Prevents **conflicts** before your next push

### **📥 5. Clone a Repository**

git clone https://github.com/peer/repo.git

* Creates a full local copy of any public repo
* Useful for team collaboration and open-source projects

### **📘 6. Inspect Your Remote Setup**

git remote -v

* Verifies where your code will be pushed/pulled from

### **🧰 7. Optional but Helpful**

git log --oneline --graph --all

* See a visual history of branches and merges
* Especially helpful later when branching is introduced

## **🧑‍💻 🧪 Hands-On Activities**

* Create GitHub account (if not done)
* Create a new repository (empty, no README)
* Connect your local repo via git remote add origin
* Push local code using git push -u origin main
* Clone a peer’s repo using git clone
* Pull changes using git pull
* Take screenshots of terminal output

## **🎯 Deliverables**

* ✅ Link to your GitHub repo
* ✅ Screenshot of:  
  + git remote -v
  + git push and git pull in terminal
  + git clone with peer’s repo
* ✅ README.md created and committed

## **🧠 Real-World Note**

**In professional teams**, code is never shipped without using:

* git push to trigger CI/CD workflows (build, test, deploy)
* git pull to sync and prevent merge conflicts
* GitHub for tracking **issues**, **PRs**, and **collaboration**

Next week you’ll:

* Dive into **branching**, **conflict resolution**, and **pull requests**