1. Create github account
2. Install git
3. Install github-desktop

### **🔹 Step 1: Check if Git is Installed**

Open **Terminal** and type:

bash

CopyEdit

git --version

* If it returns something like git version 2.30.1, you're all set.
* If not, it will prompt you to install Apple's Command Line Tools — just follow the instructions to install.

### **🔹 Step 2: Create a Folder for Your Project**

In Terminal:

bash

CopyEdit

mkdir my-first-repo

cd my-first-repo

Replace my-first-repo with any folder name you like.

### **🔹 Step 3: Initialize a Git Repository**

Inside the folder:

bash

CopyEdit

git init

This creates a new empty Git repository in that folder.

You’ll see a message like:

sql

CopyEdit

Initialized empty Git repository in /Users/yourname/my-first-repo/.git/

### **🔹 Step 4: Add Some Files**

For example, create a simple README file:

bash

CopyEdit

echo "# My First Git Repo" > README.md

Check the status:

bash

CopyEdit

git status

### **🔹 Step 5: Stage and Commit the Files**

Stage your new file:

bash

CopyEdit

git add README.md

Then commit:

bash

CopyEdit

git commit -m "Initial commit"

### **🔹 Step 6: Check the History**

To see your commits:

bash

CopyEdit

git log

Press q to exit the log view.

## **🗂 Optional: Connect to a Remote GitHub Repository**

If later you want to push this to GitHub:

1. Go to [GitHub.com](https://github.com) → Log in → Click **New Repository**
2. Name your repo (e.g., my-first-repo) and create it without README (you already have one locally)
3. Copy the remote URL (e.g., https://github.com/yourname/my-first-repo.git)

Then in your terminal:

bash

CopyEdit

git remote add origin https://github.com/yourname/my-first-repo.git

git branch -M main

git push -u origin main

## **🧠 Bonus: Useful Git Commands Cheat Sheet**

| **Command** | **Purpose** |
| --- | --- |
| git status | Check current changes |
| git add <file> | Stage file |
| git commit -m "msg" | Save snapshot |
| git log | View commit history |
| git diff | See unstaged changes |
| git push | Upload to GitHub |
| git pull | Download from GitHub |

—

**Github desktop**

Here’s a **step-by-step guide** to install and start using **GitHub Desktop for Mac**:

## **✅ Step-by-Step: Install & Use GitHub Desktop on Mac**

### **🔹 Step 1: Download GitHub Desktop**

1. Go to<https://desktop.github.com>
2. Click **Download for macOS**
3. Once downloaded, open the .dmg file
4. Drag **GitHub Desktop** into the **Applications** folder

### **🔹 Step 2: Launch GitHub Desktop**

* Open Applications → Find **GitHub Desktop** → Open it
* On first launch, it may prompt you: "Do you want to open it?" → Click **Open**

### **🔹 Step 3: Sign in to GitHub**

1. Click **Sign in to GitHub.com**
2. Your browser will open → Log in with your GitHub account
3. Authorize GitHub Desktop to connect to your account
4. It will return to the GitHub Desktop app after success

### **🔹 Step 4: Configure Git (First-Time Setup)**

When prompted, set your identity:

* **Name**: your name (e.g., *Hari Shanker*)
* **Email**: your GitHub email address

Click **Continue**.

### **🔹 Step 5: Create a New Local Repository**

1. Click **File → New Repository** (or click "Create New Repository" on the welcome screen)
2. Fill in the form:  
   * **Name**: e.g., my-first-genai-app
   * **Description**: *(optional)*
   * **Local Path**: where the folder will be created
   * **Initialize this repository with:** ✅ **README**
3. Click **Create Repository**

This creates a local folder and a Git repo with README.md in it.

### **🔹 Step 6: Add Files and Make a Commit**

1. Open the repo folder in Finder (click “Repository → Show in Finder”)
2. Add a file — for example, create main.py and write some Python code
3. Go back to GitHub Desktop — it will show the new file as **Uncommitted Changes**
4. Write a summary like: Add main.py with basic GenAI setup
5. Click **Commit to main**

### **🔹 Step 7: Publish to GitHub**

1. Click the **"Publish repository"** button at the top
2. Choose:  
   * **Name**: keep as is
   * ✅ Keep it public or private as you want
   * Optionally select your organization (**GenAITinkerers**) if it exists
3. Click **Publish Repository**

Now your code is live on GitHub.com!

## **🔄 Common Actions in GitHub Desktop**

| **Action** | **How to Do It** |
| --- | --- |
| Commit changes | Make edits → GitHub Desktop detects them → Write a commit message → Click "Commit" |
| Push to GitHub | Click "Push origin" (top bar) |
| Pull latest changes | Click "Fetch origin" (top bar) |
| Create a branch | Click “Current Branch” → “New Branch” |
| Switch repos | File → Open Recent or click the name dropdown top-left |

**Gitignore related:**

## **✅ What .gitignore Does**

### **🔍 Purpose:**

The .gitignore file tells **Git** which files and folders **not to track** in your repository. This helps avoid:

* Accidentally uploading temporary or sensitive files (like API keys)
* Clutter from system or editor-generated files
* Keeping your repo clean and focused only on code and essential files

## **🧠 Typical Files You Should Ignore in a Python + VS Code Project on macOS**

| **Category** | **Examples** |
| --- | --- |
| Python bytecode | \_\_pycache\_\_/, \*.pyc |
| Virtual environments | venv/, .env/ |
| VS Code settings | .vscode/ |
| macOS system files | .DS\_Store |
| Output artifacts | \*.log, \*.tmp, artifacts/ |
| Secrets & configs | .env, \*.secret, config.json (if personal) |

## **📝 Step-by-Step: Add .gitignore in Your Project**

### **🔹 Step 1: Open Your Project in VS Code**

Use Terminal:

bash

CopyEdit

code path/to/your/project

Or open it from GitHub Desktop → “Repository” → “Open in Visual Studio Code”

### **🔹 Step 2: Create the .gitignore File**

In the root of your project (same folder as main.py or README.md):

1. Right-click → **New File**
2. Name it: .gitignore

*(Make sure it starts with a dot!)*

### **🔹 Step 3: Add These Lines to .gitignore**

gitignore

CopyEdit

# Byte-compiled / cache files

\_\_pycache\_\_/

\*.py[cod]

\*.so

# Virtual environment

env/

venv/

.venv/

# VS Code settings

.vscode/

# macOS system files

.DS\_Store

# Logs and outputs

\*.log

\*.tmp

artifacts/

# Python environment files

\*.egg-info/

.eggs/

# Environment variables or secrets

.env

\*.secret

config.json

apikey.json

### **🔹 Step 4: Save and Commit**

Once added:

In GitHub Desktop or Terminal:  
  
 bash  
CopyEdit  
git add .gitignore

git commit -m "Add .gitignore to exclude unnecessary files"

Push your changes:  
  
 bash  
CopyEdit  
git push

## **✅ Result**

Now Git will **ignore** any files or folders listed in .gitignore, keeping your repo clean and professional.

## **🧠 Bonus Tips**

If you've already added files that should be ignored, use:  
  
 bash  
CopyEdit  
git rm -r --cached venv/

git commit -m "Remove ignored files from repo"

* You can find good templates here:<https://github.com/github/gitignore>