Setting up **Git Bash** involves downloading and installing the Git software for Windows, which includes the Bash emulator. Here's a step-by-step guide:

**1. Download Git for Windows 📥**

1. Open your web browser and go to the official Git website:
2. The website should automatically detect your operating system and offer the Windows download. Click the **"Download for Windows"** button.
3. The download of the Git installer (.exe file) will begin.

**2. Install Git and Git Bash 💻**

1. **Run the Installer:** Once the download is complete, double-click the downloaded .exe file to start the installation process.
2. **User Account Control (UAC):** If prompted by UAC, click **Yes** to allow the program to make changes.
3. **License Information:** Read the GNU General Public License and click **Next**.
4. **Select Components:** You can generally leave the default components selected (make sure "Git Bash Here" and "Git GUI Here" are checked) and click **Next**.
5. **Start Menu Folder:** Choose the name for the Git folder in your Start Menu (default is fine) and click **Next**.
6. **Adjusting your PATH environment (Crucial Step):** This is where you decide how Git is accessed from the command line.
   * **Recommended Option (Easiest and Safest):** Choose **"Git from the command line and 3rd-party software"**. This adds Git to your system's PATH, allowing you to use Git commands from Git Bash, the Windows Command Prompt (cmd), or PowerShell. Click **Next**.
   * *Note: Avoid the third option unless you know exactly what you're doing, as it can potentially overwrite system tools.*
7. **Choosing the Backend Transport:** Select **"Use the OpenSSL library"** (the default and recommended option) and click **Next**.
8. **Configuring Line Ending Conversions:** Select **"Checkout Windows-style, commit Unix-style line endings"** (the recommended default) and click **Next**. This helps prevent issues when collaborating across different operating systems.
9. **Choosing the Terminal Emulator:** Choose **"Use MinTTY (the default terminal of MSys2)"**. This is the one that provides the features of Git Bash, and is the recommended option. Click **Next**.
10. **Choosing the Default Behavior of git pull:** Select **"Default (fast-forward or merge)"** and click **Next**.
11. **Choosing a Credential Helper:** Select **"Git Credential Manager"** (the default and recommended option) and click **Next**.
12. **Configuring Extra Options:** You can usually leave the default boxes checked (**"Enable file system caching"** and **"Enable experimental support for pseudo consoles"**). Click **Next**.
13. **Experimental Features (Optional):** Skip or select experimental features based on your preference. Click **Install**.
14. **Installation Complete:** Wait for the installation to finish. Uncheck "View Release Notes" and click **Finish**.

**3. Launch and Verify Git Bash 🚀**

1. **Open Git Bash:**
   * **Method 1 (Start Menu):** Search for **"Git Bash"** in your Windows Start Menu and click the application icon.
   * **Method 2 (Right-Click):** Navigate to any folder in Windows File Explorer, right-click, and select **"Git Bash Here"** from the context menu. A terminal window will open in that folder's location.
2. **Verify the Installation:** In the Git Bash window, type the following command and press **Enter**:

Bash

git --version

You should see the installed Git version (e.g., git version 2.45.0.windows.1), confirming that Git Bash is set up and working! 🎉

**Git and GitHub basics** — from creating an account, logging in, creating repositories (folders), committing changes, and pushing your code.

Let’s go step by step 👇

**🧠 1. Understanding Git and GitHub**

* **Git** → is a **version control system** used to track changes in your code or files.
* **GitHub** → is a **website** that hosts your Git repositories online.

You use Git on your computer, and then push your work to GitHub.

**⚙️ 2. Installing Git**

**On Windows:**

1. Go to <https://git-scm.com/downloads>
2. Download and install.
3. During installation, choose “Use Git from Windows Command Prompt.”
4. After installation, open **Command Prompt** or **Git Bash** and check:
5. git --version

**🧑🏽‍💻 3. Create a GitHub Account**

1. Go to <https://github.com/>
2. Click **Sign up**.
3. Create your username, email, and password.
4. Verify email.
5. Done — now you have your GitHub account.

**🔑 4. Log in to Git on Your Computer**

After installing Git, you must set your **name** and **email** (the same one used on GitHub):

git config --global user.name "Your Name"

git config --global user.email "youremail@example.com"

You can check your settings:

git config --list

**📂 5. Create a New Folder (Project)**

You can create a folder where your project will live:

mkdir myproject

cd myproject

Initialize Git inside it:

git init

Now Git will start tracking changes inside that folder.

**📝 6. Create or Add Files**

Example:

echo "Hello Git" > readme.txt

Check file status:

git status

Add files to be tracked:

git add readme.txt

Or add everything:

git add .

**💬 7. Commit Changes**

Commit means saving the current changes in your project:

git commit -m "Initial commit"

**☁️ 8. Create a Repository on GitHub**

1. Go to [GitHub.com](https://github.com/)
2. Click **+** → **New repository**
3. Give it a name (example: myproject)
4. Choose **Public** or **Private**
5. Don’t check “Initialize with a README” (since you already have one locally)
6. Click **Create repository**

**🌐 9. Connect Local Project to GitHub**

GitHub will show you a URL like this:

https://github.com/yourusername/myproject.git

Connect it:

git remote add origin https://github.com/yourusername/myproject.git

**🚀 10. Push (Upload) to GitHub**

Now push your local project to GitHub:

git branch -M main

git push -u origin main

It will ask you to log in.  
Use your **GitHub username** and **Personal Access Token (PAT)** instead of password (GitHub stopped password authentication).

👉 **To create a Personal Access Token:**

1. Go to GitHub → Profile → **Settings**
2. Developer settings → **Personal access tokens**
3. Generate new token (classic)
4. Select scopes → *repo, workflow, user*
5. Copy the token and use it as password when Git asks for it.

**🔄 11. Making Further Changes**

If you change or add files:

git add .

git commit -m "Updated something"

git push

**🔁 12. Moving Between Folders or Projects**

To move between folders:

cd ..

cd foldername

To create a new project:

mkdir newproject

cd newproject

git init

**📥 13. Cloning an Existing Repository**

If you want to download a project from GitHub:

git clone https://github.com/username/projectname.git

This creates a folder with all files and history.

**🔍 14. Checking Logs and Status**

* Check commits:
* git log
* Check current changes:
* git status
* See differences:
* git diff

Would you like me to make a **PDF guide** (with screenshots and commands clearly shown) for your students or personal study — “Complete Git & GitHub Basics Step-by-Step”?  
I can create it with clear structure and visuals.

**🌍 GOAL:**

We’ll take a website folder (for example, mywebsite) and upload it to a GitHub repository, so it’s viewable online or for collaboration.

**🧩 1. Prepare Your Website Folder**

Make sure your website files (HTML, CSS, JS, images, etc.) are in one folder.

Example:

C:\Users\SHAFIC\Documents\mywebsite

Inside, you might have:

index.html

style.css

script.js

images/

**⚙️ 2. Open Git Bash in That Folder**

Right-click inside the folder → choose **“Git Bash Here”**

Then initialize Git:

git init

This tells Git to start tracking your website folder.

**🧾 3. Check Git Configuration**

Make sure your username and email are set:

git config --global user.name "summitsmith"

git config --global user.email "summitsmith842@gmail.com"

**📦 4. Add Your Website Files to Git**

git add .

✅ This stages (prepares) all your website files to be committed.

**💬 5. Commit Your Files**

git commit -m "Initial website upload"

✅ This saves a snapshot of your project locally.

**☁️ 6. Create a Repository on GitHub**

1. Go to [https://github.com](https://github.com/)
2. Click **“+” → “New repository”**
3. Name it something like mywebsite
4. Leave it **Public** or **Private**
5. **Do NOT initialize with a README** (since your local folder already has files)
6. Click **Create repository**

**🔗 7. Connect Your Local Folder to GitHub**

After creating the repo, GitHub will show you something like:

https://github.com/summitsmith/mywebsite.git

Now link your local folder to that remote repository:

git remote add origin https://github.com/summitsmith/mywebsite.git

**🚀 8. Push Your Website to GitHub**

git branch -M main

git push -u origin main

It will prompt you for login:

* **Username:** your GitHub username (summitsmith)
* **Password:** your **Personal Access Token (PAT)** (not your actual password)

If you haven’t created one yet, I can show you how.

**✅ 9. Confirm**

Go to your GitHub repository page.  
You’ll now see your website files uploaded.

**🌐 10. (Optional) Host the Website with GitHub Pages**

You can make your website live for free:

1. Go to your GitHub repo → **Settings**
2. Scroll to **Pages**
3. Under “Source,” choose **Deploy from branch**
4. Select **main** branch → **/root**
5. Click **Save**

After a few seconds, your website will be live at:

https://summitsmith.github.io/mywebsite/

**💡 Recap of Commands**

cd path/to/mywebsite

git init

git add .

git commit -m "Initial website upload"

git branch -M main

git remote add origin https://github.com/summitsmith/mywebsite.git

git push -u origin main

Would you like me to create a **short Git workflow PDF guide for “Uploading a Website to GitHub”** (with screenshots and these commands clearly labeled)? It’s great for you and your students to refer to anytime.