STEP 1: Create a new GitHub repository & push Flask project

Thatʼs a **great project** to learn Git and GitHub, especially for beginners! Since youʼd like a **step-by-step guide with clear explanations**, Iʼll break down each instruction into easy-to-follow steps, including the Git commands and screenshots to take.

Iʼll also suggest **when to take screenshots** so you can include them in your document.

Letʼs go step by step:

# 📝 STEP 1: Create a new GitHub repository & push Flask project

## Create a new repository on GitHub

 Go to [GitHub](https://github.com/).

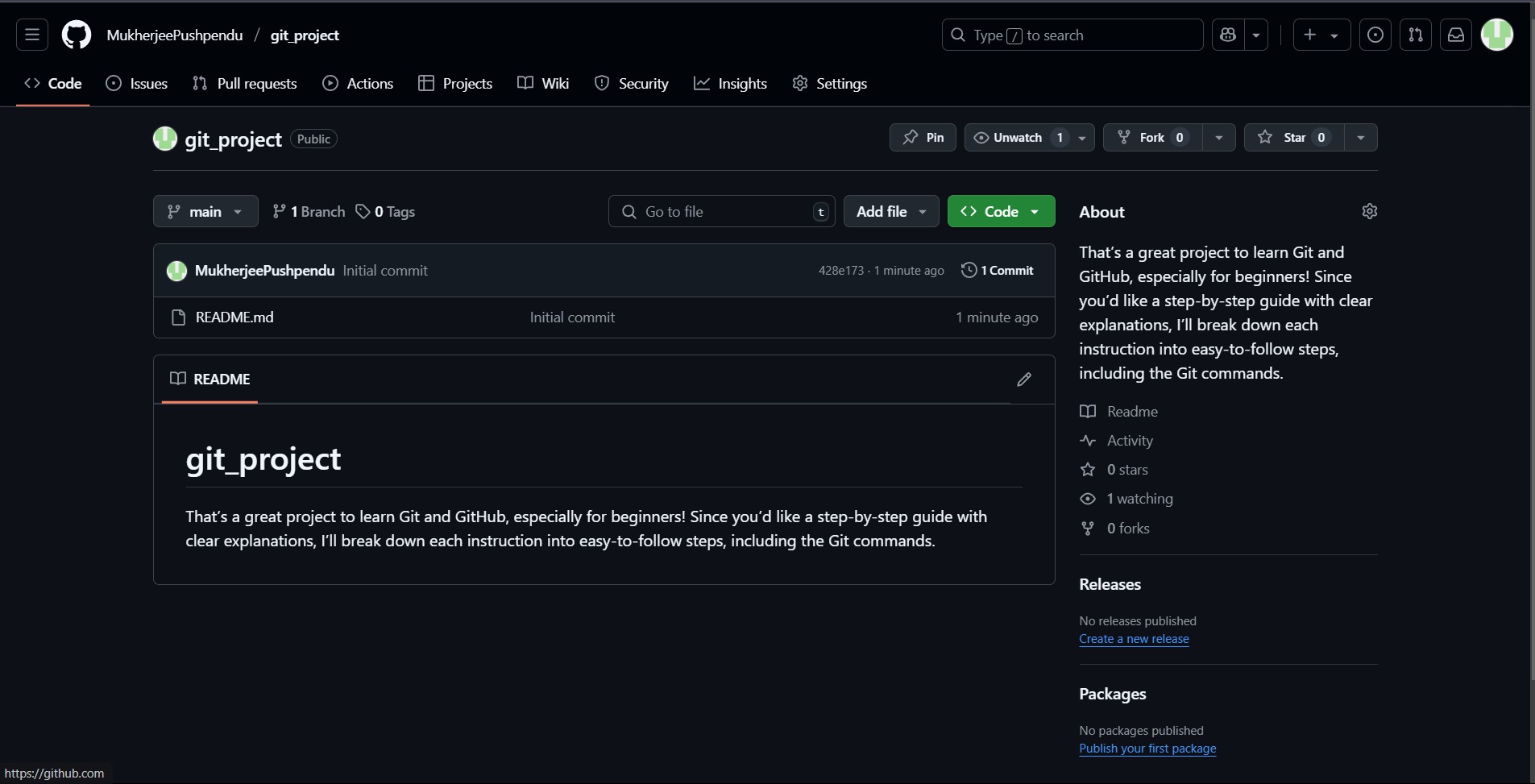
 Click **“Newˮ** → **“New Repositoryˮ**.

Enter a repository name (e.g., ), keep it **public** or **private**.

git\_project

 **Do not initialize** with README, .gitignore, or license.  Click **Create Repository**.

✅ **Screenshot to take:** GitHub repo page after creation.



## Clone the repository using SSH

➡ If you already have an SSH key on GitHub, skip SSH setup and go to cloning.

## If you donʼt have SSH keys:

1. Run this command in **Git Bash** or terminal:

ssh-keygen -t ed25519 -C ["your\_email@example.com"](mailto:your_email@example.com)

1. Press **Enter** through the prompts to accept defaults.
2. Copy your public SSH key:

cat ~/.ssh/id\_ed25519.pub

1. Go to GitHub → **Settings  SSH and GPG keys  New SSH key** → paste your key → **Save**.

## Now clone the repo:

git clone https://github.com/MukherjeePushpendu/git\_project.git cd git\_project

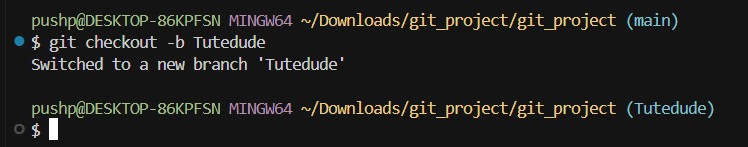
✅ **Screenshot to take:** Terminal showing successful clone.

## Create a branch with your username (e.g., )

git checkout -b Tutedude

**Tutedude**

✅ **Screenshot:** terminal showing branch switched.



## Add your Flask project files

 Copy your Flask project files into the repo folder. Then:

git add .

git commit -m "Added Flask project files" git push origin Tutedude

✅ **Screenshot:** terminal after .

git push

## Merge your branch into main

git checkout main git merge Tutedude

✅ **Screenshot:** terminal showing merge success.

# 📝 STEP 2: Create a JSON

**<yourname>\_new**

**Create a new branch**

git checkout -b Tutedude\_new

# branch & update

git push origin main

✅ **Screenshot:** terminal after switching.

## Update JSON file

 Open your JSON file (used in

route).

 Modify the content (add, change or update keys/values).  Save the file.

/api

Then:

git add data.json

git commit -m "Updated JSON data for /api route" git push origin Tutedude\_new

✅ **Screenshot:** updated JSON file + terminal after .

push

## Merge into main

git checkout main

git merge Tutedude\_new

🟢 **If thereʼs a merge conflict:**

 Git will tell you which files have conflicts.

Open those files → look for

<<<<<<<

,

,

markers.

 Keep the version from Then:

=======

>>>>>>>

Tutedude\_new

→ delete markers.

# 📝 STEP 3: Create

**master\_1**

**master\_2**

## Create branches from main

git checkout main

git checkout -b master\_1 git checkout main

git checkout -b master\_2

**&**

# branches

git add conflicted\_file.json

git commit -m "Resolved merge conflict using Tutedude\_new changes" git push origin main

✅ **Screenshot:** terminal showing both branches created.

## In : Build frontend To-Do page

**master\_1**

Add an HTML form with:

input

Item Name

input

Item Description

 Submit button Example:

<form action="/submittodoitem" method="POST">

<label>Item Name:</label><input type="text" name="itemName"><br>

<label>Item Description:</label><input type="text" name="itemDescriptio n"><br>

<button type="submit"Submit</button>

</form>

Then:

git add templates/todo.html

git commit -m "Added To-Do page form" git push origin master\_1

✅ **Screenshot:** form code + terminal after .

push

**In**

**master\_2**

## : Build backend

**route**

In your Flask app:

from flask import request

from pymongo import MongoClient

@app.route('/submittodoitem', methods=['POST']) def submit\_todo\_item():

itemName = request.form['itemName']

itemDescription = request.form['itemDescription'] client  MongoClient('mongodb://localhost:27017/') db = client['todo\_db']

collection = db['items']

collection.insert\_one({'itemName': itemName, 'itemDescription': itemDescri ption})

return "Item added!"

**/submittodoitem**

Then:

git add app.py

git commit -m "Added /submittodoitem route to store item in MongoDB" git push origin master\_2

✅ **Screenshot:** code + terminal after .

push

## Merge both branches into main

git checkout main git merge master\_1 git merge master\_2 git push origin main

✅ **Screenshot:** both merges.

# 📝 STEP 4: Enhance form & Git operations

## In : Add new fields step-by-step

**master\_1**

➡ Add **Item ID field** to form:

<label>Item ID:/label><input type="text" name="itemId"><br>

git add templates/todo.html

git commit -m "Added Item ID field" git push origin master\_1

✅ **Screenshot:** HTML + terminal.

➡ Add **Item UUID field**:

<label>Item UUID:/label><input type="text" name="itemUUID"><br>

✅ **Screenshot.**

git add templates/todo.html

git commit -m "Added Item UUID field" git push origin master\_1

➡ Add **Item Hash field**:

<label>Item Hash:</label><input type="text" name="itemHash"><br>

✅ **Screenshot.**

git add templates/todo.html

git commit -m "Added Item Hash field" git push origin master\_1

# 📝 Quick command summary for switching branches:

👉 Already created → just switch:

CopyEdit

git checkout master\_1 # do work, commit

No need for



-b

**Merge**

git checkout main git merge master\_1 git push origin main

again!

## into main

**master\_1**

git checkout master\_2

# do work, commit

git checkout main

# merge master\_1 and master\_2 into main

✅ **Screenshot.**

## Git Reset in main to rollback to first commit (Item ID only)

git log

# (copy commit hash of "Added Item ID field") git reset --soft <commit-hash>

Then recommit:

git commit -m "Rollback to Item ID only" git push origin main --force

✅ **Screenshot:** log + reset + commit.

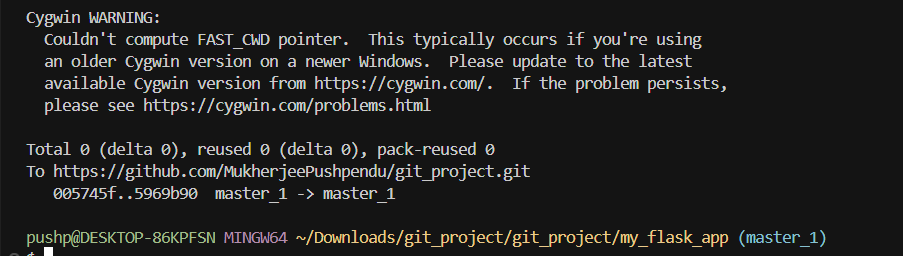


## Rebase changes back to

git checkout master\_1 git rebase main

git push origin master\_1 --force

**master\_1**

✅ **Screenshot:** terminal after rebase.

