Creating a GitHub Repository and Pushing Code from Linux

Background

Version control is an essential part of modern software development, and Git is the most widely used system for this purpose. GitHub provides a cloud-based platform to store, share, and collaborate on code. Using Git from Linux offers developers a highly efficient environment due to its powerful command-line interface, flexibility, and automation support. With Git and GitHub, you can maintain code history, work collaboratively, and ensure your projects are securely backed up online.

Benefits of Using Git on Linux

Working with Git on Linux provides several advantages:

- Native Command-Line Support: Linux integrates seamlessly with Git commands, making operations smooth and fast.
- Automation Friendly: Linux scripting allows developers to automate Git workflows for efficiency.
- Open-Source Ecosystem: Many development tools on Linux are designed to work directly with Git.
- Security: Linux offers strong support for SSH authentication, ensuring secure communication with GitHub servers.
- Stability and Performance: Git operations on Linux often perform faster and with fewer compatibility issues compared to other platforms.

Prerequisites

Before setting up a repository, ensure the following prerequisites are met:

1. Working knowledge of basic commands in linux
2. Linux Environment: A Debian/Ubuntu-based distribution (or equivalent).
3. GitHub Account: Create an account on https://github.com.

Here's a step-by-step guide for creating a GitHub repository and pushing your code from Linux:

1. **Install Git (if not already installed)**
   sudo apt update && sudo apt install git -y # For Ubuntu/Debian

   Check version:
   git –version

2. **Configure Git (one-time setup)**
   git config --global user.name "Your Name"
   git config --global user.email "your_email@example.com"

   Verify:
   git config --list

3. **Create a Repository on GitHub**

   a. Go to GitHub → log in.
   b. Click the "+" (top right) → New repository.
   c. Enter:
      - Repository name (e.g., my-project)
      - Description (optional)
      - Choose Public or Private
   d. Click Create repository.
   e. You will see instructions with a remote URL like:
      https://github.com/username/my-project.git

4. **Initialize Git in Your Project Folder**
   cd ~/path/to/your/project
   git init

   ---

5. **Add Files and Commit**
   git add .
   git commit -m "Initial commit"

6. **Connect Local Repo to GitHub**
   git remote add origin https://github.com/username/my-project.git
   git remote -v

7. **Push Code to GitHub (via SSH)**

   Use SSH Keys (recommended for long-term use)

   a. **Generate an SSH key**:
      ssh-keygen -t ed25519 -C your_email@example.com
      (Press Enter for defaults, set a passphrase if you want.)
   b. **Start the SSH agent and add the key**:
      eval "$(ssh-agent -s)"
      ssh-add ~/.ssh/id_ed25519
   c. **Add the public key to GitHub**:
      cat ~/.ssh/id_ed25519.pub

      Copy the output, then go to:
      GitHub → Settings → SSH and GPG keys → New SSH key → Paste it.
   d. **Change the remote URL to SSH**:

git remote set-url origin git@github.com:openalgorithmdevelopers/tbd.git

(openalgorithmdevelopers/tbd.git is the name of the repository under your user name, which you have created in the git)

   e. **Push using SSH**:

      git push -u origin main

Finish

SSH stands for Secure Shell.

It's a protocol that lets you securely connect and exchange data between your computer and another machine (like GitHub's servers).

With SSH, instead of typing your password/token each time, you authenticate using a pair of cryptographic keys:
- Private key → stays only on your computer (secret).
- Public key → uploaded to GitHub (safe to share).

Analogy: SSH is like a lock-and-key system where GitHub has the lock (public key) and your computer has the key (private key).