Lab: Development Tools - Git and SSH Key Integration

This lab will guide you through setting up an SSH key, integrating it with GitHub, creating and managing a Python project repository, and performing basic Git operations such as branching, committing, and pushing changes.

Objectives:

By the end of this lab, you will be able to:

1. Create an SSH key on an Ubuntu virtual machine (VM).
2. Copy the SSH key to your GitHub account.
3. Create a GitHub repository and clone it onto your VM.
4. Set up a basic folder structure for a Python project.
5. Perform Git operations: committing as master, branching, and managing versions.

Steps:

Step 1: Create an SSH Key

1. Open the terminal on your Ubuntu VM.
2. Generate an SSH key
3. Press Enter to accept the default file location and provide a passphrase (optional).
4. Display the generated public key:
5. Copy the entire output to your clipboard.

Step 2: Add the SSH Key to GitHub

1. Log in to your GitHub account.
2. Navigate to Settings > SSH and GPG Keys > New SSH Key.
3. Provide a title (e.g., "Ubuntu VM") and paste the copied SSH key into the "Key" field.
4. Click Add SSH Key.

Step 3: Create a GitHub Repository

1. On GitHub, click on the + icon in the top-right corner and select New Repository.

2. Provide a repository name (e.g., `python_project_lab`) and an optional description.

3. Select Public or Private, and check Add a README file.

4. Click Create Repository.

Step 4: Clone the Repository on Your VM

1. Copy the repository's SSH URL from GitHub.

2. On your Ubuntu VM, run:

3. Navigate to the cloned directory:

Step 5: Set Up a Python Project Folder Structure

1. Create the following folder structure

```
python_project_lab/
       ├── src/
            │  ├── __init__.py
            │  ├── data_loader.py
            │  ├── preprocessing.py
            │  ├── model.py
            │  └── evaluation.py
       ├── tests/
            │  ├── __init__.py
            │  ├── test_data_loader.py
            │  ├── test_preprocessing.py
            │  └── test_model.py
       ├── README.md
       └── requirements.txt
```

2. Create a basic `README.md` file

3. Step 6: Commit Initial Changes as Master
    a. Stage the changes:

    b. Commit the changes:

    c. Push to the `master` branch:

Step 7: Create a New Branch

1. Create and switch to a new branch (e.g., `feature-branch`)

2. Step 8: Create First Code Version (V1)

    a. Add a simple Python script in the `src` folder (e.g., `src/main.py`):

    b. Stage and commit the changes:

    c. Push to GitHub:

Step 9: Edit Code and Commit as V2

1. Edit the `src/main.py` file to include a new feature

2. Stage and commit the changes:

Step 10: Create a New Branch for Further Development

3.
    1. Create a new branch (e.g., `experimental-branch`):

    2. Push the branch to GitHub: