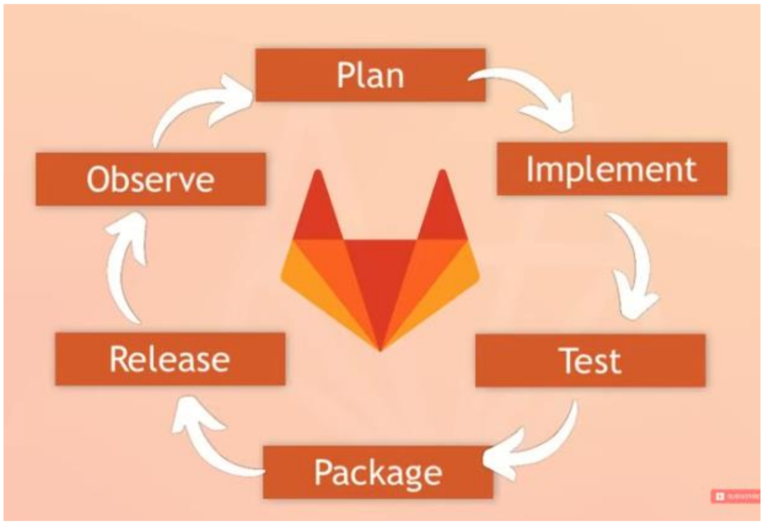
# Lab # 20

# Getting Familiar with GitLab

## What is GitLab?

- ✓ GitLab was originally a **fully free** and open-source software distributed under **the MIT License**.
- ✓ It was split into two distinct versions - **GitLab CE** (Community Edition) and **GitLab EE** (Enterprise Edition) in **July 2013.**
- ✓ **In 2017**, GitLab announced that their code would become fully open-sourced under an MIT License.
- ✓ GitLab is a **web-based DevOps** platform that helps developers manage their code and the entire software development lifecycle in one place.
- ✓ It provides Git repository hosting (like GitHub), along with built-in tools for version control, **CI/CD pipelines**, code review, issue tracking, and project management
- ✓ GitLab helps teams reduce product **lifecycles** and increase productivity, which in turn creates value for customers.
- ✓ The application doesn't require users to manage authorizations for each tool. If permissions are set once, then everyone in the organization has access to every component.
- ✓ Customers can opt for the paid version of GitLab if they want to access more functionalities. For example, the Premium version costs $19 per user/month.

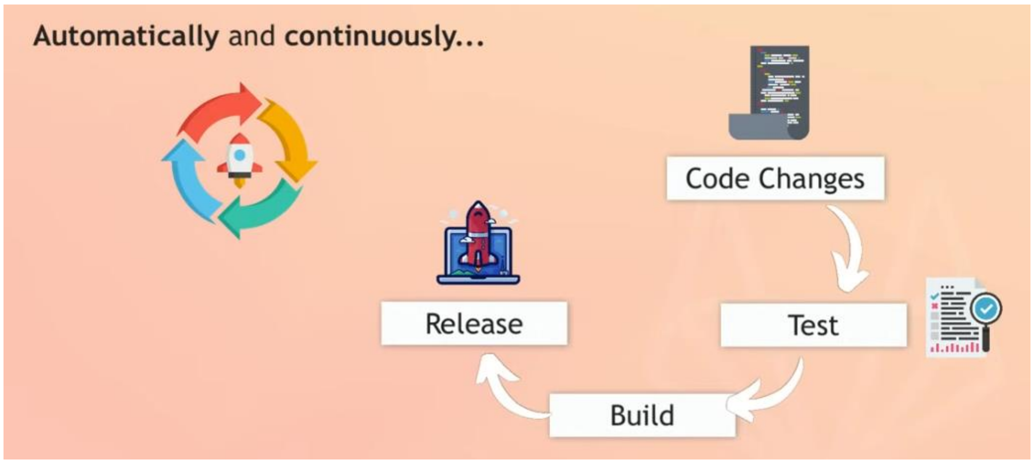


https://about.gitlab.com/

# Lab # 20

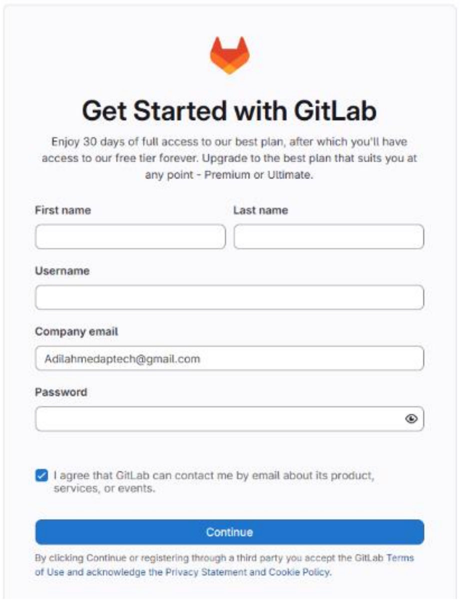
**Basic Terminologies:**

- **Git Repository:** A storage location that contains project files and their complete version history.
- **Issue Tracking:** A GitLab feature used to create, assign, and track tasks, bugs, and feature requests.
- **Wiki:** A shared space in GitLab for maintaining project documentation and guidelines.
- **Merge Requests (MRs):** A mechanism to propose, review, and merge code changes into the main branch.
- **CI/CD Pipelines:** Continuous Integration (CI) / Continuous Delivery/Deployment (CD), an automated process for building, testing, and deploying code using.gitlab-ci.yml.
- **GitLab Runners:** Agents that execute CI/CD pipeline jobs on different environments or platforms.
- **Groups and Projects:** A structure in GitLab used to organize repositories, manage access, and enable team collaboration.



**CI / CD Pipeline**

**Step # 01: Sign Up or Install GitLab**



Enjoy 30 days of full access to our best plan, after which you'll have access to our free tier forever. Upgrade to the best plan that suits you at any point - Premium or Ultimate.

First name

Last name

Username

Company email: Adilahmedaptech@gmail.com

Password

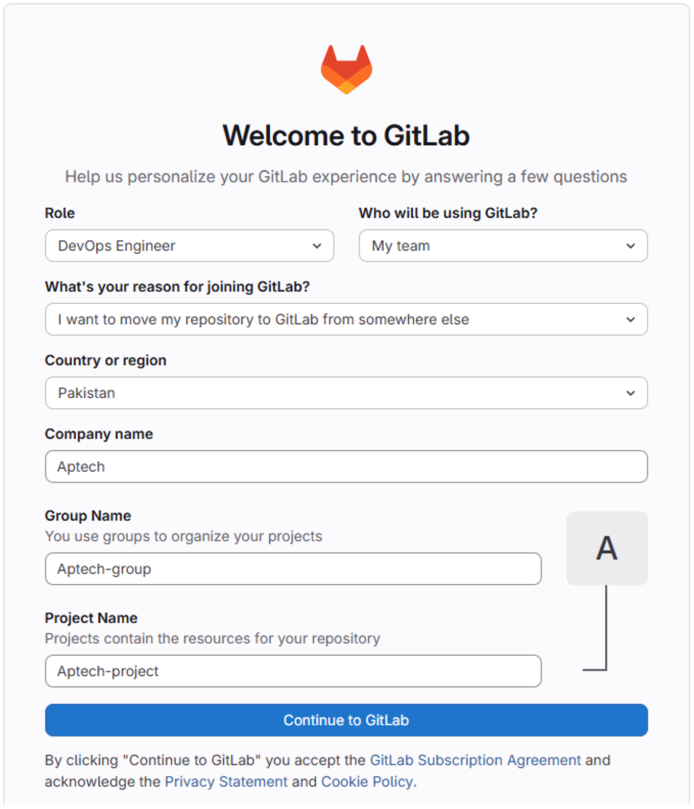
☑ I agree that GitLab can contact me by email about its product, services, or events.

By clicking Continue or registering through a third party you accept the GitLab Terms of Use and acknowledge the Privacy Statement and Cookie Policy.

# Lab # 20

## Step # 02:

After logging in, Set an environment.



Welcome to GitLab

Help us personalize your GitLab experience by answering a few questions

Role: DevOps Engineer

Who will be using GitLab?: My team

What's your reason for joining GitLab?: I want to move my repository to GitLab from somewhere else

Country or region: Pakistan

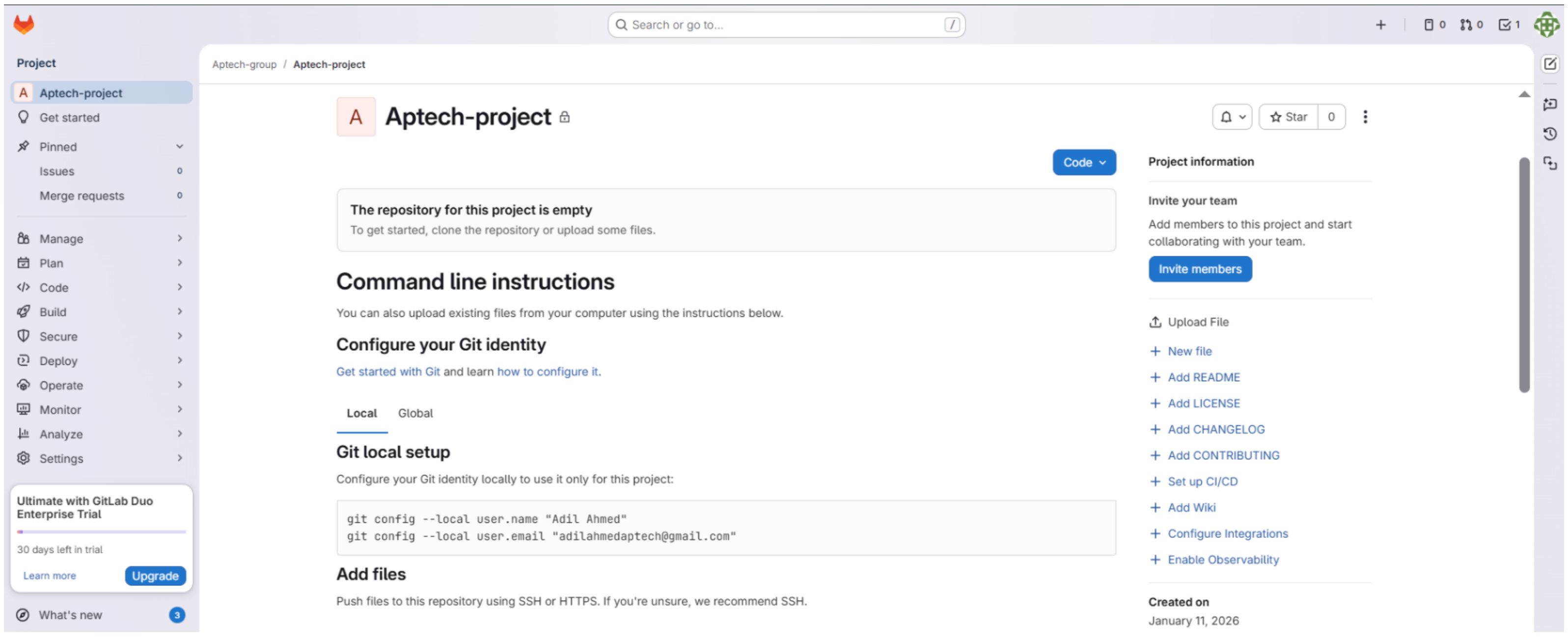
Company name: Aptech

Group Name
You use groups to organize your projects
Aptech-group

Project Name
Projects contain the resources for your repository
Aptech-project

By clicking "Continue to GitLab" you accept the GitLab Subscription Agreement and acknowledge the Privacy Statement and Cookie Policy.

**After Creating a blank project, Interface will look like this.**



The repository for this project is empty
To get started, clone the repository or upload some files.

Command line instructions

You can also upload existing files from your computer using the instructions below.

Configure your Git identity

Get started with Git and learn how to configure it.

Local Global

Git local setup

Configure your Git identity locally to use it only for this project:

```
git config --local user.name "Adil Ahmed"
git config --local user.email "adilahmedaptech@gmail.com"
```

Add files

Push files to this repository using SSH or HTTPS. If you're unsure, we recommend SSH.

Project information

Invite your team

Add members to this project and start collaborating with your team.

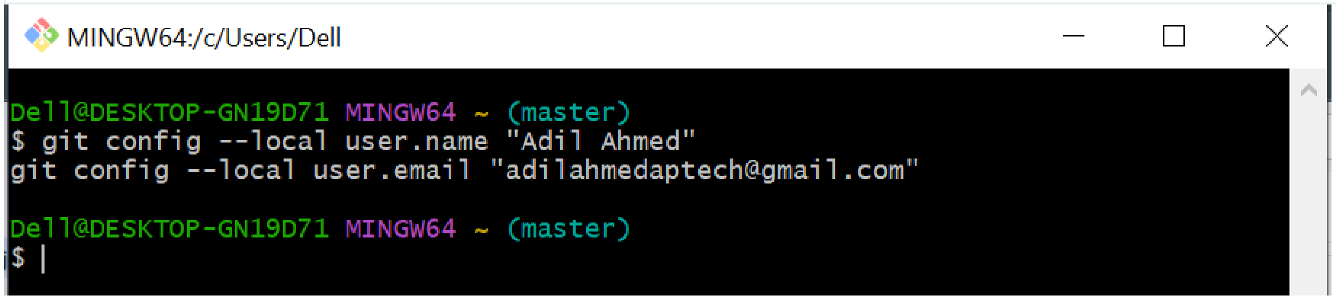
Created on
January 11, 2026

# Lab # 20

### Step # 03 Configure your Git Identity

- Open your Git Bash
- For the first step, to configure your username and email ID.
- To configure, use the following commands:

```
▪ git config --local user.name "User_ID"
▪ git config --local user.email "Email_ID"
```



MINGW64:/c/Users/Dell

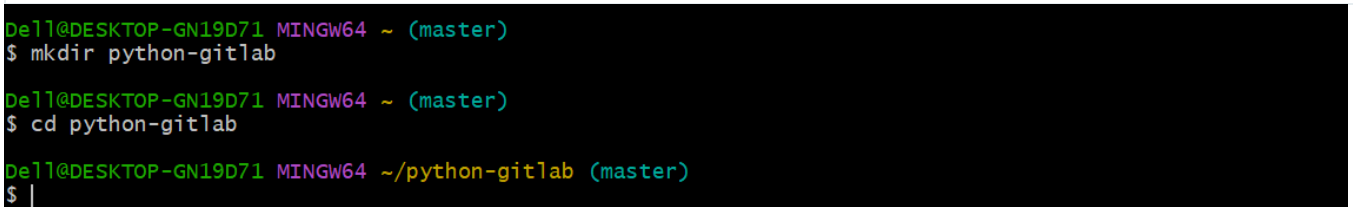
```
Dell@DESKTOP-GN19D71 MINGW64 ~ (master)
$ git config --local user.name "Adil Ahmed"
git config --local user.email "adilahmedaptech@gmail.com"

Dell@DESKTOP-GN19D71 MINGW64 ~ (master)
$ |
```

**Note:** You will notice that something called the "master" appears on the screen. Whenever a Git repository is created for the first time, it creates a branch, and it's called the master

### Step # 04: Create the Project Folder:

To create a repository in the working directory, use the following commands:

```
mkdir python-gitlab-lab
cd python-gitlab-lab
```



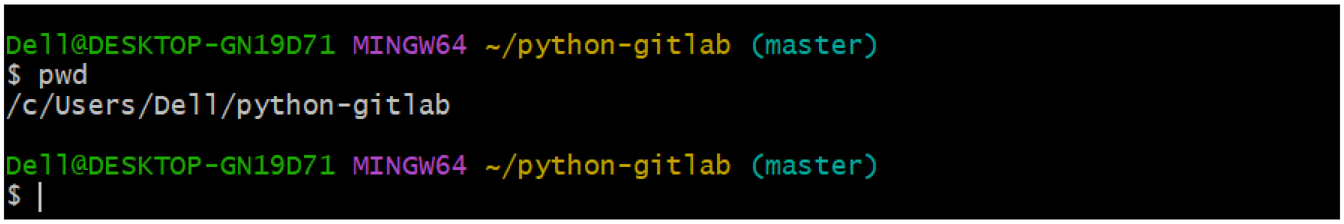
```
Dell@DESKTOP-GN19D71 MINGW64 ~ (master)
$ mkdir python-gitlab

Dell@DESKTOP-GN19D71 MINGW64 ~ (master)
$ cd python-gitlab

Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ |
```

### Step # 04: Check location:

Now you can navigate to this repository, using the following command:

```
Pwd
```

# Lab # 20

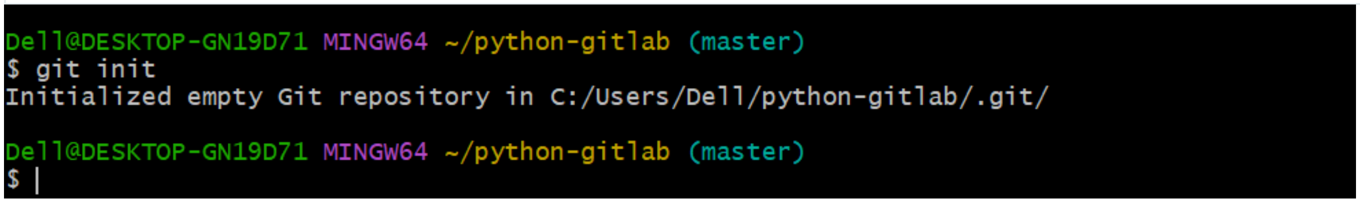


```
Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ pwd
/c/Users/Dell/python-gitlab

Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ |
```

**Step # 05: Initialize Git repository:**

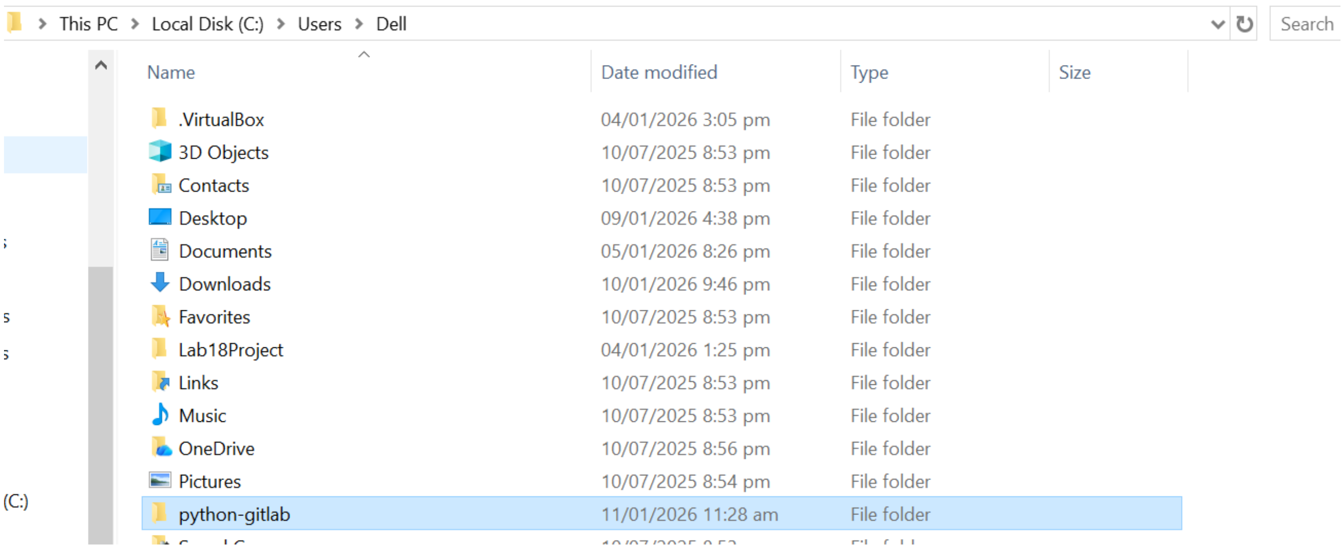
Now it's time to initialize a git repository. To initialize a repository, use the following command:

```
git init
```



```
Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ git init
Initialized empty Git repository in C:/Users/Dell/python-gitlab/.git/

Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ |
```

**Step # 06: Navigate to the folder.**



This PC > Local Disk (C:) > Users > Dell

| Name | Date modified | Type | Size |
|---|---|---|---|
| .VirtualBox | 04/01/2026 3:05 pm | File folder | |
| 3D Objects | 10/07/2025 8:53 pm | File folder | |
| Contacts | 10/07/2025 8:53 pm | File folder | |
| Desktop | 09/01/2026 4:38 pm | File folder | |
| Documents | 05/01/2026 8:26 pm | File folder | |
| Downloads | 10/01/2026 9:46 pm | File folder | |
| Favorites | 10/07/2025 8:53 pm | File folder | |
| Lab18Project | 04/01/2026 1:25 pm | File folder | |
| Links | 10/07/2025 8:53 pm | File folder | |
| Music | 10/07/2025 8:53 pm | File folder | |
| OneDrive | 10/07/2025 8:56 pm | File folder | |
| Pictures | 10/07/2025 8:54 pm | File folder | |
| python-gitlab | 11/01/2026 11:28 am | File folder | |

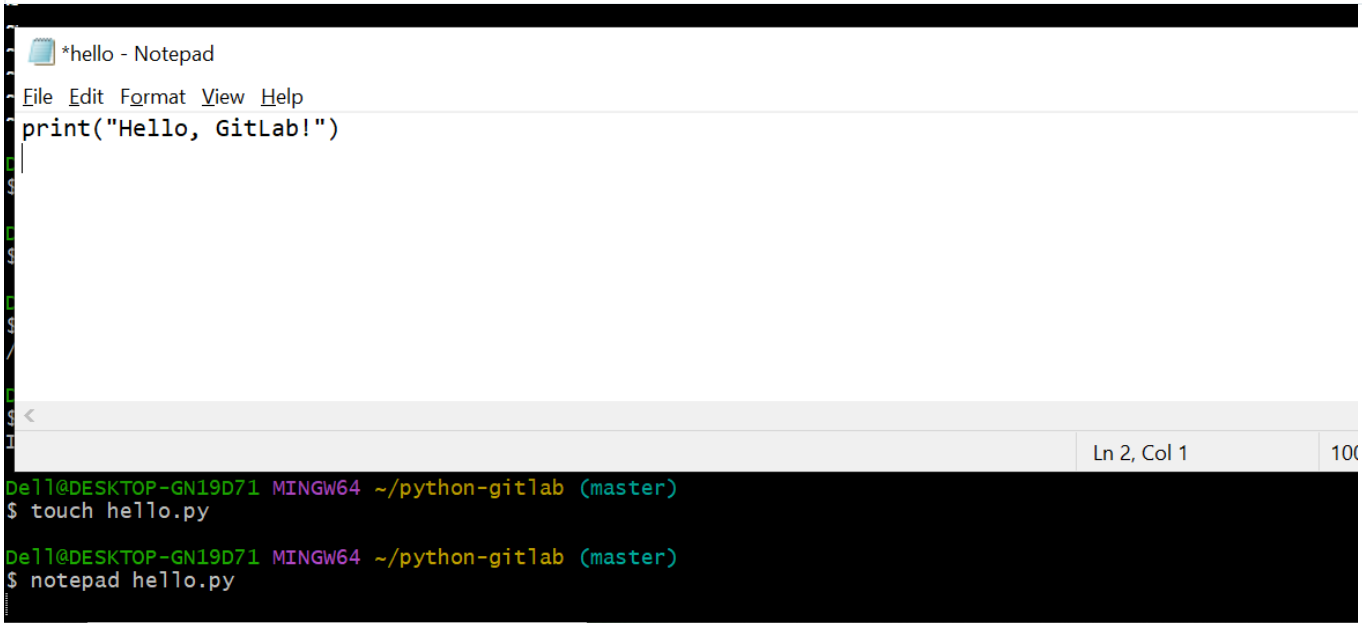
**Step # 07: Create a Python file**

Now, create a notepad for the repository. Later on, you can push that file onto the GitLab repository.

# Lab # 20

To create a notepad, use the following commands:

```
touch hello.py
notepad hello.py
```



```
print("Hello, GitLab!")
```

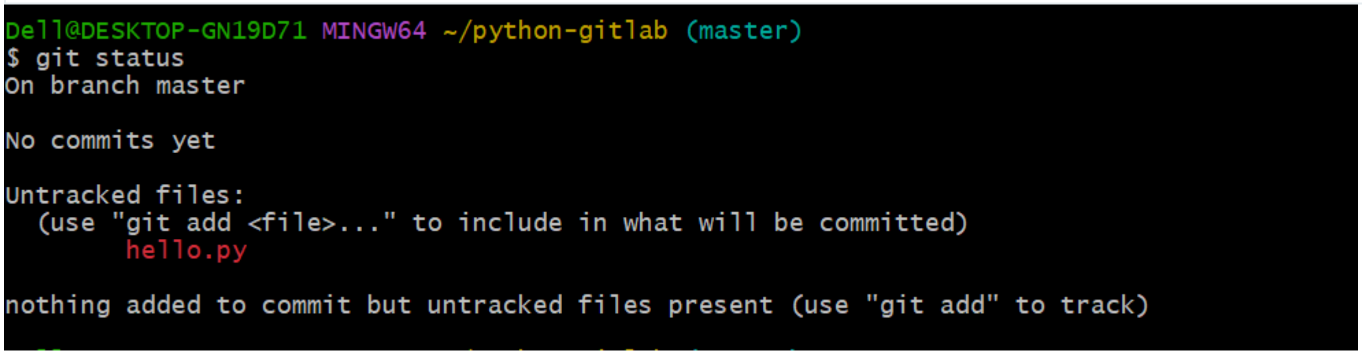
```
Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ touch hello.py

Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ notepad hello.py
```

Notepad will appear on the screen. Type anything inside it, then save and close it.

**Step # 08:**

The next step is to check the status of the file.

```
git status
```



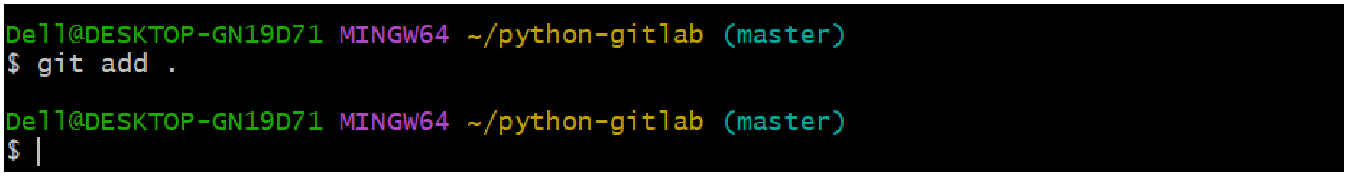
```
Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        hello.py

nothing added to commit but untracked files present (use "git add" to track)
```

It shows that there isn't a file committed yet, and there are untracked files. The untracked files can be seen in red. Now, add the file to the staging area with the following command:

# Lab # 20

**Step # 08:**

```
git add.
```

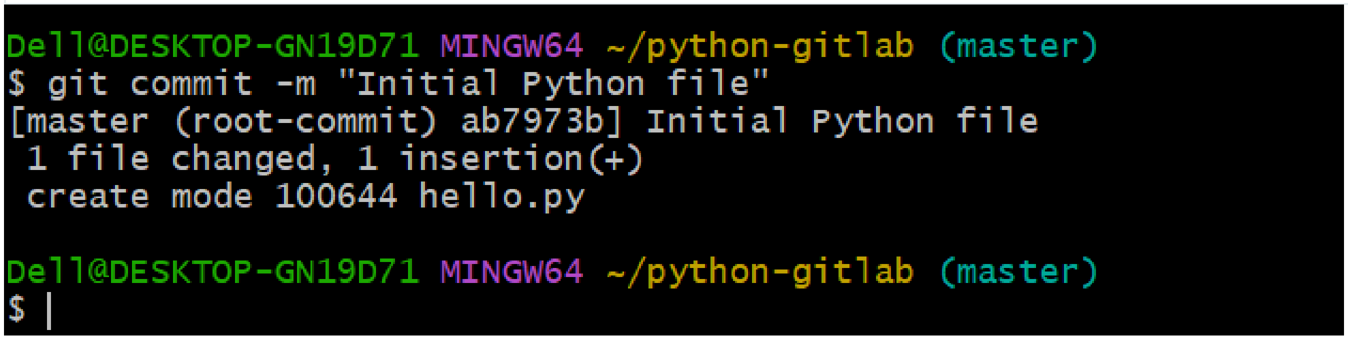


```
Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ git add .

Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ |
```

**Step # 09:**

The next step is to commit the file. To commit, use the commit command.

```
git commit -m "Initial Python file"
```

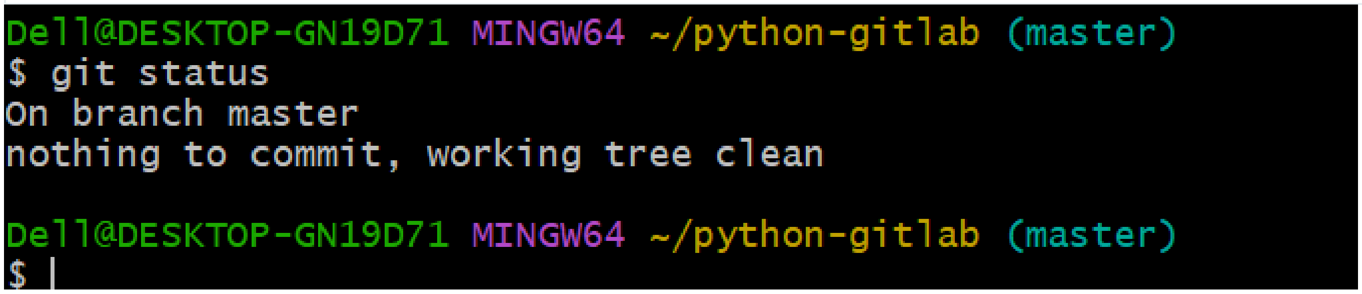


```
Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ git commit -m "Initial Python file"
[master (root-commit) ab7973b] Initial Python file
 1 file changed, 1 insertion(+)
 create mode 100644 hello.py

Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ |
```

**Step # 10**

Recheck the status of the file.

```
git status
```



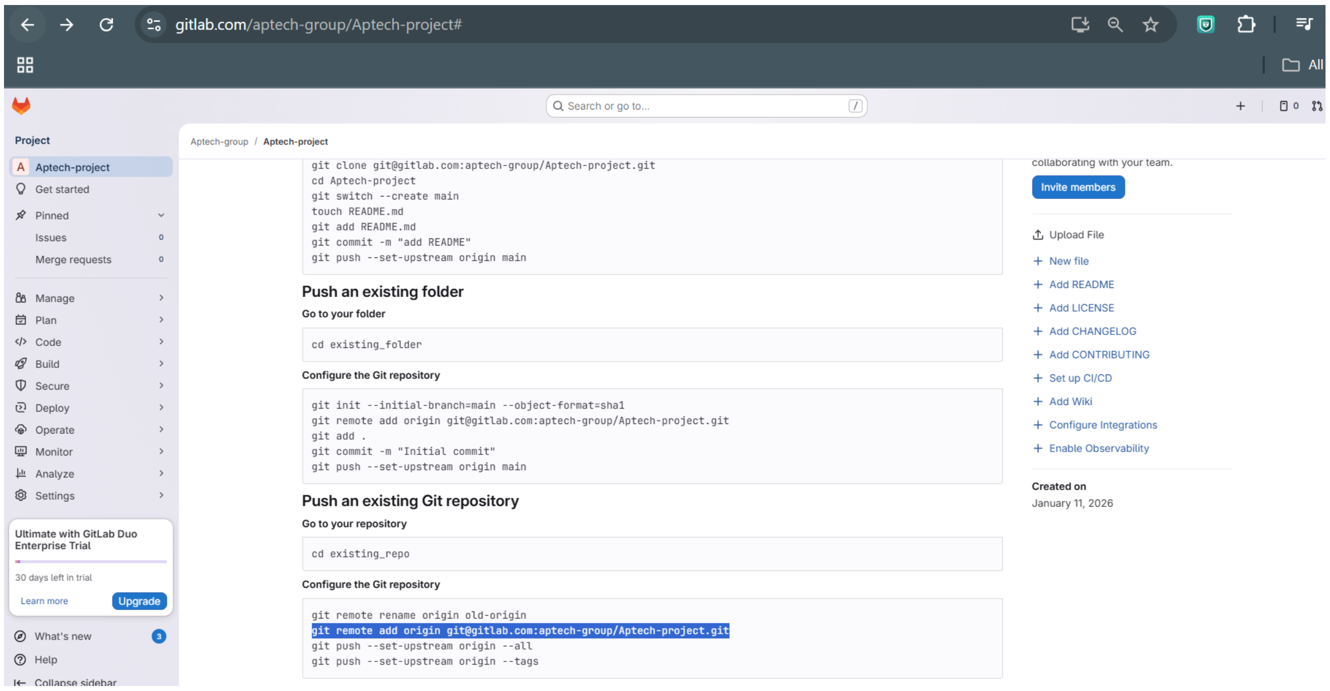
```
Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ git status
On branch master
nothing to commit, working tree clean

Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ |
```

You'll notice that there are no more commits to be made, as there was a single notepad, and that was committed in the previous step.

# Lab # 20

## Step # 11: Connect Local Repo to GitLab

Now, it's time to push the notepad onto the GitLab repository. To accomplish this, go to your GitLab and copy the git remote origin command, as shown below.



```
git clone git@gitlab.com:aptech-group/Aptech-project.git
cd Aptech-project
git switch --create main
touch README.md
git add README.md
git commit -m "add README"
git push --set-upstream origin main
```

**Push an existing folder**

Go to your folder

```
cd existing_folder
```

Configure the Git repository

```
git init --initial-branch=main --object-format=sha1
git remote add origin git@gitlab.com:aptech-group/Aptech-project.git
git add .
git commit -m "Initial commit"
git push --set-upstream origin main
```

**Push an existing Git repository**

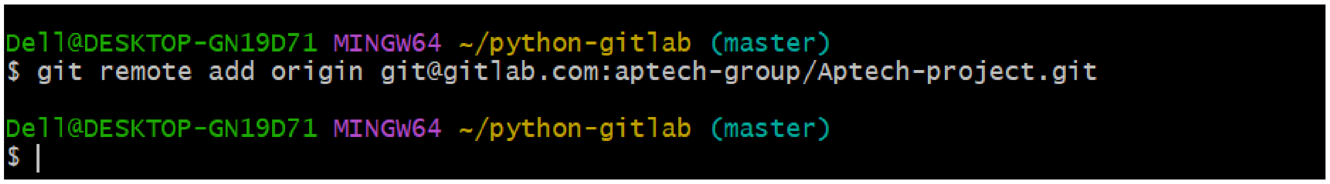
Go to your repository

```
cd existing_repo
```

Configure the Git repository

```
git remote rename origin old-origin
git remote add origin git@gitlab.com:aptech-group/Aptech-project.git
git push --set-upstream origin --all
git push --set-upstream origin --tags
```

After you have done this, go back to your Git Bash and paste the command.

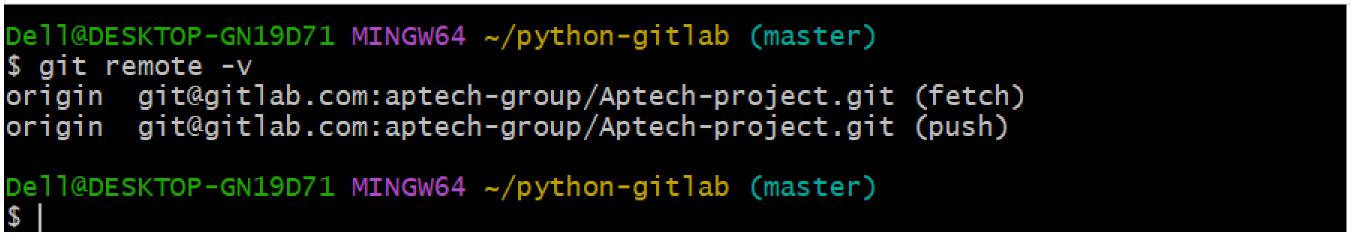


```
Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ git remote add origin git@gitlab.com:aptech-group/Aptech-project.git

Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ |
```

Now use the remote command, followed by the push command, to push the file to the remote repository.

## Step # 12: Verify remote:

git remote -v

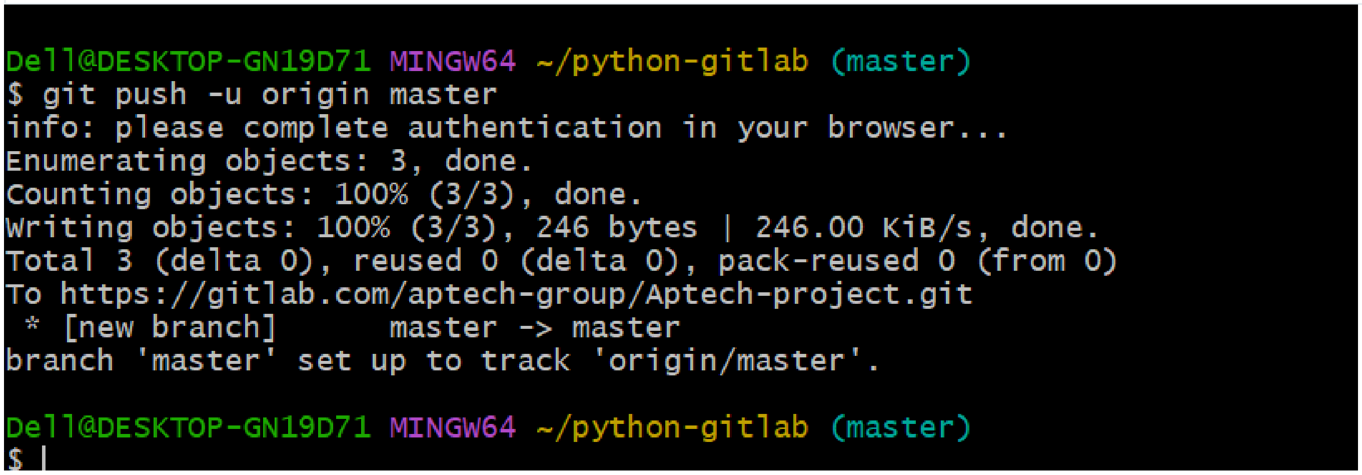


```
Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ git remote -v
origin  git@gitlab.com:aptech-group/Aptech-project.git (fetch)
origin  git@gitlab.com:aptech-group/Aptech-project.git (push)

Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ |
```

# Lab # 20

## Step # 13: Push Code to GitLab

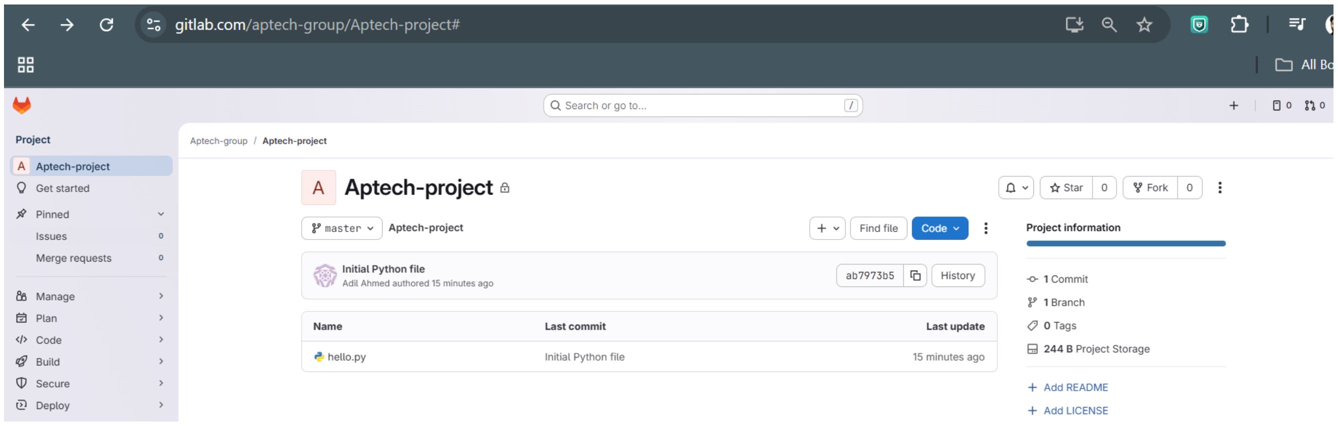
```
git push -u origin master
```



```
Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ git push -u origin master
info: please complete authentication in your browser...
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 246 bytes | 246.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://gitlab.com/aptech-group/Aptech-project.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.

Dell@DESKTOP-GN19D71 MINGW64 ~/python-gitlab (master)
$ |
```

## Step # 14:

Now go to your GitLab and check to see if there are any additions to the new project you initially created. You can see that the python file appears there. You can now open and check the contents of the file.

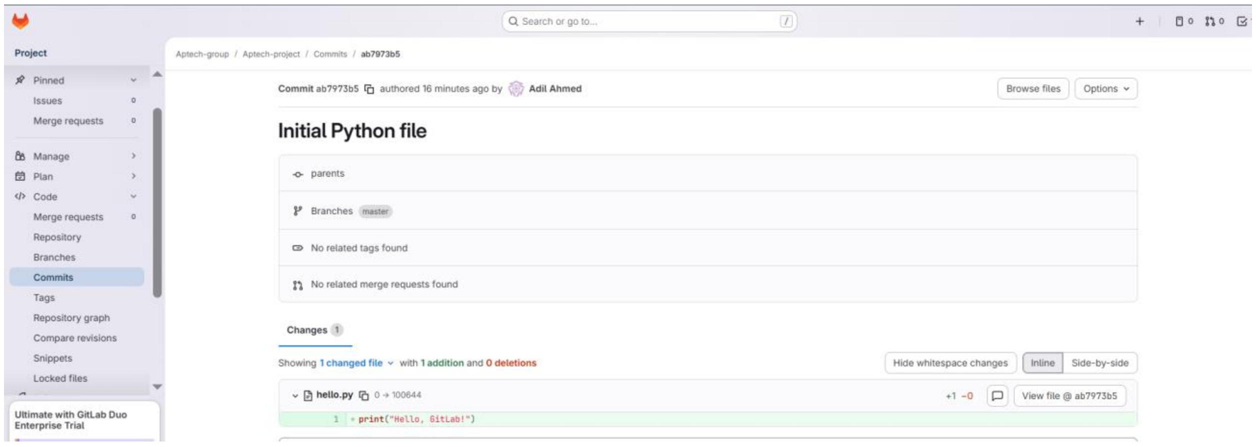


Initial Python file
Adil Ahmed authored 15 minutes ago
ab7973b5

| Name | Last commit | Last update |
|---|---|---|
| hello.py | Initial Python file | 15 minutes ago |

1 Commit
1 Branch
0 Tags
244 B Project Storage

# Lab # 20



Aptech-group / Aptech-project / Commits / ab7973b5

Commit ab7973b5 authored 16 minutes ago by Adil Ahmed

Initial Python file

parents

Branches master

No related tags found

No related merge requests found

Changes 1

Showing 1 changed file with 1 addition and 0 deletions

hello.py 0 → 100644 +1 −0

```
+ print("Hello, GitLab!")
```

**Task:**

1. Create a GitLab project and push a Python file.
2. Create a basic CI/CD pipeline to run the Python file automatically.