

# Tìm hiểu sơ lược về Git, Github, Gitlab

**Trần Tuấn Vũ \_ 21002182 \_ K66A5**

## I. Git

### Khái niệm :

- Là một hệ thống quản lý phiên bản phân tán (Distributed Version Control System – DVCS),
- Là một trong những hệ thống quản lý phiên bản phân tán phổ biến nhất hiện nay.
- Cung cấp cho mỗi lập trình viên kho lưu trữ (repository) riêng chứa toàn bộ lịch sử thay đổi.

### Lợi ích khi dùng Git :

- Các dự án thực tế thường có nhiều lập trình viên làm việc song song. Vì vậy, một hệ thống kiểm soát phiên bản như Git là cần thiết để đảm bảo không có xung đột code giữa các lập trình viên.
- Ngoài ra, các yêu cầu trong các dự án như vậy thay đổi thường xuyên. Vì vậy, một hệ thống kiểm soát phiên bản cho phép các nhà phát triển revert và quay lại phiên bản cũ hơn của code.
- Cuối cùng, đôi khi một số dự án đang được chạy song song liên quan đến cùng một cơ sở code. Trong trường hợp như vậy, khái niệm phân nhánh trong Git là rất quan trọng.
- Tính năng của Git :
  1. Dễ sử dụng, thao tác nhanh, gọn, lẹ và rất an toàn.
  2. Dễ dàng kết hợp các phân nhánh (branch), có thể giúp quy trình làm việc code theo nhóm đơn giản hơn rất nhiều.
  3. Chỉ cần clone mã nguồn từ kho chứa hoặc clone một phiên bản thay đổi nào đó từ kho chứa, hoặc một nhánh nào đó từ kho chứa là bạn có thể làm việc ở mọi lúc mọi nơi.
  4. Deployment sản phẩm của bạn một cách không thể nào dễ dàng hơn.

### Các lệnh Git cơ bản:

#### **1) git config**

Tác dụng : Để set user name và email trong main configuration file.

Cách dùng :

- Để kiểm tra tên và kiểu email trong cấu hình dùng `git config --global user.name` và `git config --global user.email`.
- Để set email hoặc tên mới: `git config --global user.name = "vux-66a5"` và `git config --global user.email = "trantuanvu_t66@hus.edu.vn"`

## 2) git init

Tác dụng : Khởi tạo 1 git repository 1 project mới hoặc đã có.

Cách dùng: `git init` trong thư mục gốc của dự án.

## 3) git clone

Tác dụng: Copy 1 git repository từ remote source.

Cách dùng: `git clone <:clone git url:>`

## 4) git status

Tác dụng: Để check trạng thái của những file bạn đã thay đổi trong thư mục làm việc. VD: Tất cả các thay đổi cuối cùng từ lần commit cuối cùng.

Cách dùng: `git status` trong thư mục làm việc.

## 5) git add

Tác dụng: Để đưa một tập tin vào Staging Area

Cách dùng: `git add <tên_file>` hoặc muốn thêm hết file của thư mục thì `git add .`

## 6) git commit

Tác dụng: commit nghĩa là một action để Git lưu lại một snapshot của các sự thay đổi trong thư mục làm việc. Và các tập tin, thư mục được thay đổi đã phải nằm trong Staging Area. Mỗi lần commit nó sẽ được lưu lại lịch sử chỉnh sửa của code kèm theo tên và địa chỉ email của người commit. Ngoài ra trong Git

cũng có thể khôi phục lại tập tin trong lịch sử commit của nó để chia cho một branch khác, vì vậy bạn sẽ dễ dàng khôi phục lại các thay đổi trước đó.

Cách dùng: `git commit -m "First commit"`

### 7) git push/git pull

Tác dụng: Push hoặc Pull các thay đổi đến remote. Nếu đã add và commit các thay đổi và muốn đẩy nó lên hoặc remote đã update và apply tất cả thay đổi đó trên code của mình.

Cách dùng: `git pull <:remote:> <:branch:>` and `git push <:remote:> <:branch:>`

### 8) git branch

Tác dụng: liệt kê tất cả các branch (nhánh).

Cách dùng: `git branch` hoặc `git branch -a`

### 9) git merge

Tác dụng: Merge 2 branch lại với nhau.

Cách dùng: Chuyển tới branch muốn merge rồi dùng: `git merge <:branch_muon_merge:>`

### 10) git remote

Tác dụng: Để check remote/source bạn có hoặc add thêm remote

Cách dùng: `git remote` để kiểm tra và liệt kê.

`git remote add <: remote_url:>` để thêm.

`git remote remove origin` để xóa remote đã tồn tại

## II. Github và Gitlab

### 1. GitLab

- Là một công cụ quản lý lưu trữ kho lưu trữ được phát triển bởi GitLab Inc và được sử dụng cho quy trình phát triển phần mềm.
- Cung cấp nhiều cách quản lý để chúng ta có thể sắp xếp hợp lý quy trình làm việc hợp tác của mình để hoàn thành vòng đời phát triển phần mềm.
- Cho phép nhập kho lưu trữ từ Google Code, Bitbucket, v.v.
- Một số tính năng của GitLab:
  1. Nền tảng quản lý kho lưu trữ phiên bản cộng đồng mã nguồn mở.
  2. Dễ dàng duy trì kho lưu trữ trên máy chủ.
  3. Cung cấp các công cụ như Group Milestones, Time Tracking and Issue Tracker, v.v. để phát triển hiệu quả.
  4. Nhiều tính năng giao diện người dùng và xác thực tự phát hơn.
  5. Quyền của người dùng và việc bảo vệ Branch được tăng cường.

## 2. GitHub

- Là một công cụ dịch vụ lưu trữ kho lưu trữ có tính năng cộng tác và kiểm soát truy cập.
- Là một nền tảng để các lập trình viên cùng nhau sửa lỗi và lưu trữ các dự án mã nguồn mở.
- Được thiết kế cho các nhà phát triển và để giúp họ theo dõi những thay đổi của họ trong một dự án thông qua kho lưu trữ.
- Một số tính năng của GitHub:
  1. Chỉ định các cột mốc và nhãn cho các dự án.
  2. Cho phép xem và so sánh giữa các Branches.
  3. GitHub Pages cho phép xuất bản và lưu trữ các trang web trong GitHub.
  4. Tính năng đánh dấu cú pháp.
  5. Nó cho phép tích hợp API của bên thứ ba để theo dõi lỗi và lưu trữ đám mây.

### **Điểm khác nhau cơ bản giữa GitHub và GitLab:**

- **Open-sourced:** GitLab phiên bản cộng đồng là mã nguồn mở nhưng GitHub thì không
- **Private Repository:** GitLab cũng cung cấp kho lưu trữ riêng miễn phí. GitHub cũng cho phép người dùng có kho lưu trữ riêng miễn phí nhưng với tối đa ba cộng tác viên.
- **Project Analysis:** GitLab cung cấp cho người dùng xem biểu đồ phát triển dự án. GitHub chưa có tính năng này nhưng họ có thể kiểm tra lịch sử cam kết.
- **Security:** GitLab an toàn hơn Github. Github kém an toàn hơn vì thiếu Bảng điều khiển bảo mật(security Dashboard), Tuân thủ giấy phép(License Compliance).

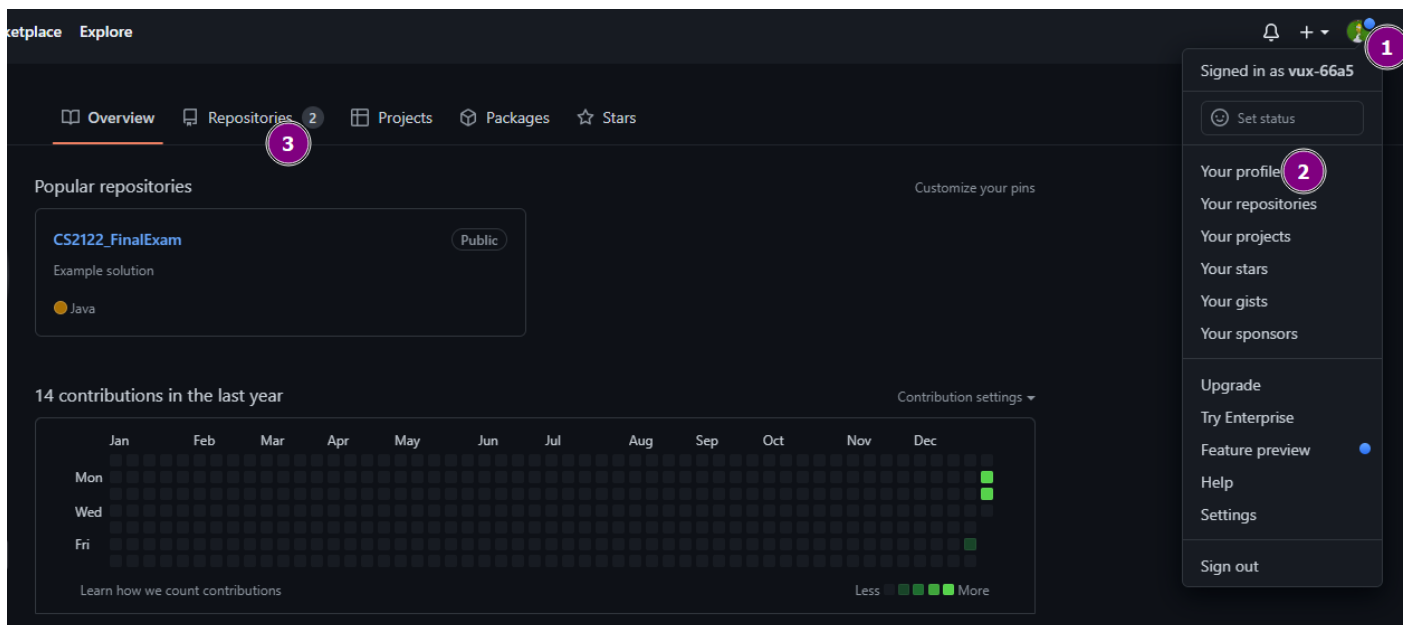
# III. Sử dụng Git

Git là một VCS, GitHub và GitLab,... chỉ là các trang web cung cấp dịch vụ lưu trữ Git. Nên các câu lệnh Git được sử dụng trên mọi nền tảng lưu trữ là như nhau.

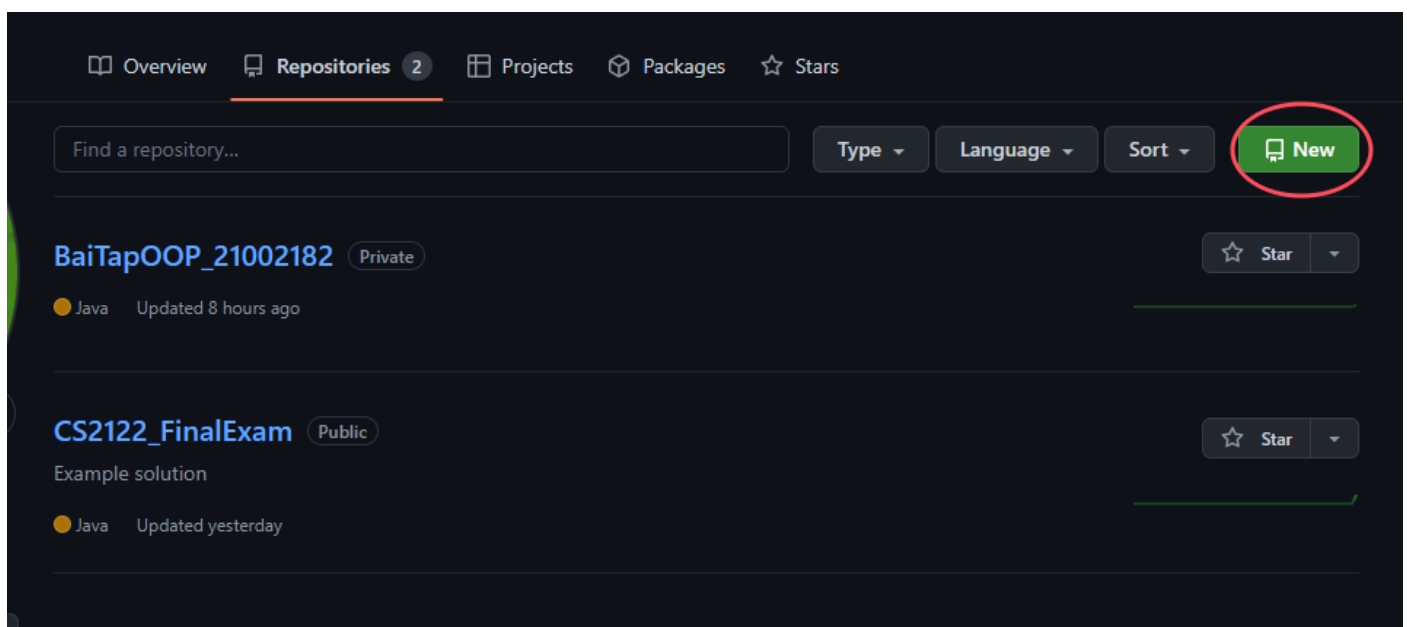
Hướng dẫn sử dụng Git để đẩy code lên GitHub ( Tương tự với GitLab)

B1 : Truy cập <https://github.com/> và tạo tài khoản

B2 : Truy cập profile cá nhân, chọn Repositories



B3 : chọn New



B4: Đặt tên repo, thêm mô tả cho repo ( khuyến khích ) , chọn trạng thái repo ( Public hoặc Private ) rồi ấn Create repository

## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner \*

Repository name \*

1

vux-66a5 ▾

/


OOPHomework ✓

Great repository names are short and memorable. Need inspiration? How about [urban-pancake?](#)


Description (optional)

2

upload bài tập oop

☒  **Public**

Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**

You choose who can see and commit to this repository.

3

**Initialize this repository with:**

Skip this step if you're importing an existing repository.

☐ **Add a README file**

This is where you can write a long description for your project. [Learn more.](#)

**Add .gitignore**


Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: None ▾

**Choose a license**

A license tells others what they can and can't do with your code. [Learn more.](#)

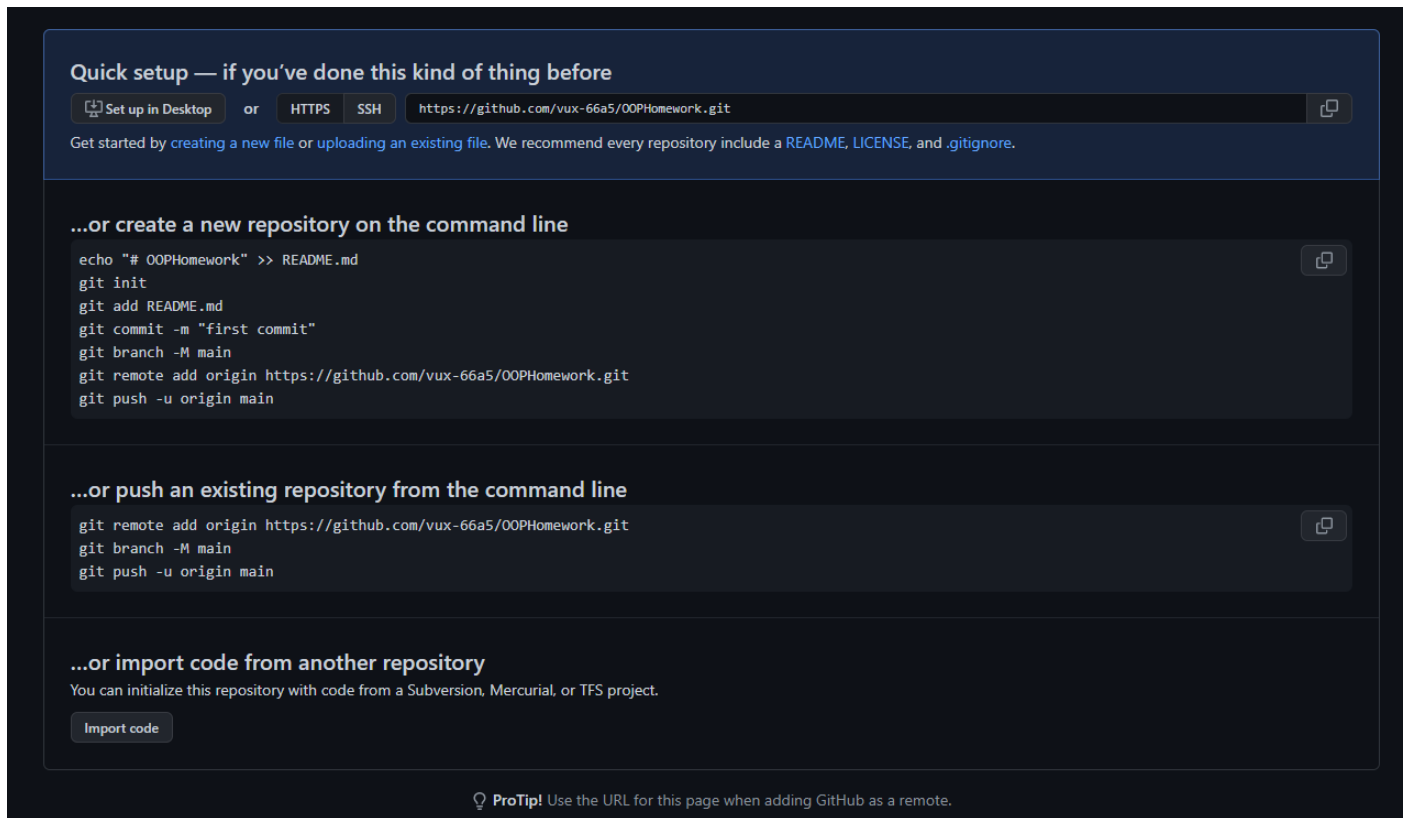
License: None ▾

 You are creating a public repository in your personal account.

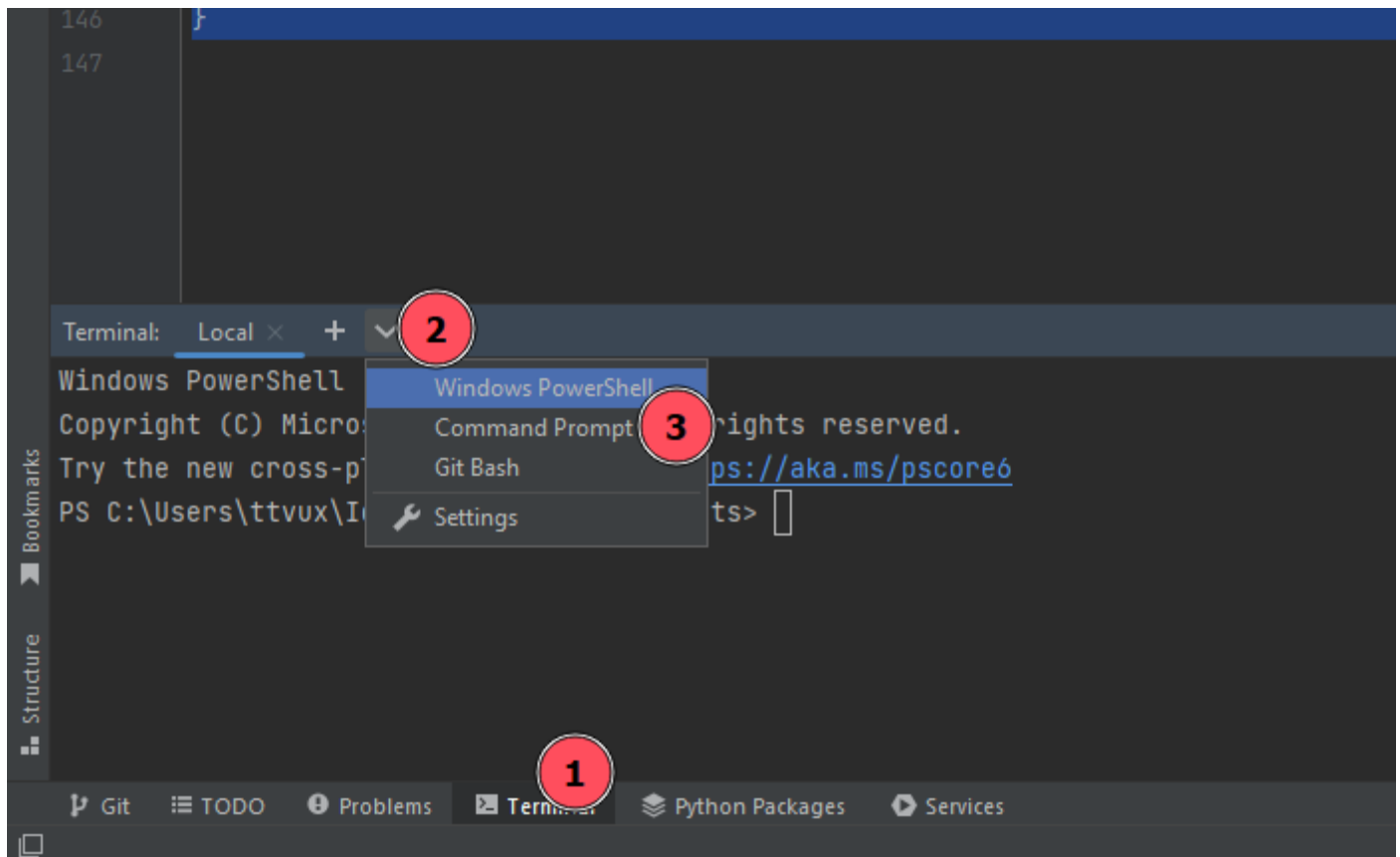
Create repository

4

Kết quả thu được :



B5 : Mở IntelliJ IDEA, mở CMD



B6: Lần lượt nhập các câu lệnh :

### Thêm .git folder vào folder đang làm việc

```
C:\Users\ttvux\IdeaProjects\JavaProjects>git init
Reinitialized existing Git repository in C:/Users/ttvux/IdeaProjects/JavaProjects/.git/
```

### Add toàn bộ file hiện có của folder đang làm việc vào folder .git

```
C:\Users\ttvux\IdeaProjects\JavaProjects>git add .
warning: in the working copy of 'src/BreadManager/Bread.java', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/BreadManager/BreadStore.java', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/BreadManager/Cheese.java', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/BreadManager/Olives.java', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/BreadManager/ThickcrustBread.java', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/BreadManager/ThincrustBread.java', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/BreadManager/ToppingDecorator.java', LF will be replaced by CRLF the next time Git touches it
```

### Cài đặt danh tính ( khớp với username và tài khoản email đăng kí github )

```
C:\Users\ttvux\IdeaProjects\JavaProjects>git config --global user.email "trantuanvu_t66@hus.edu.vn"
```

```
C:\Users\ttvux\IdeaProjects\JavaProjects>git config --global user.name "vux-66a5"
```

### Commit

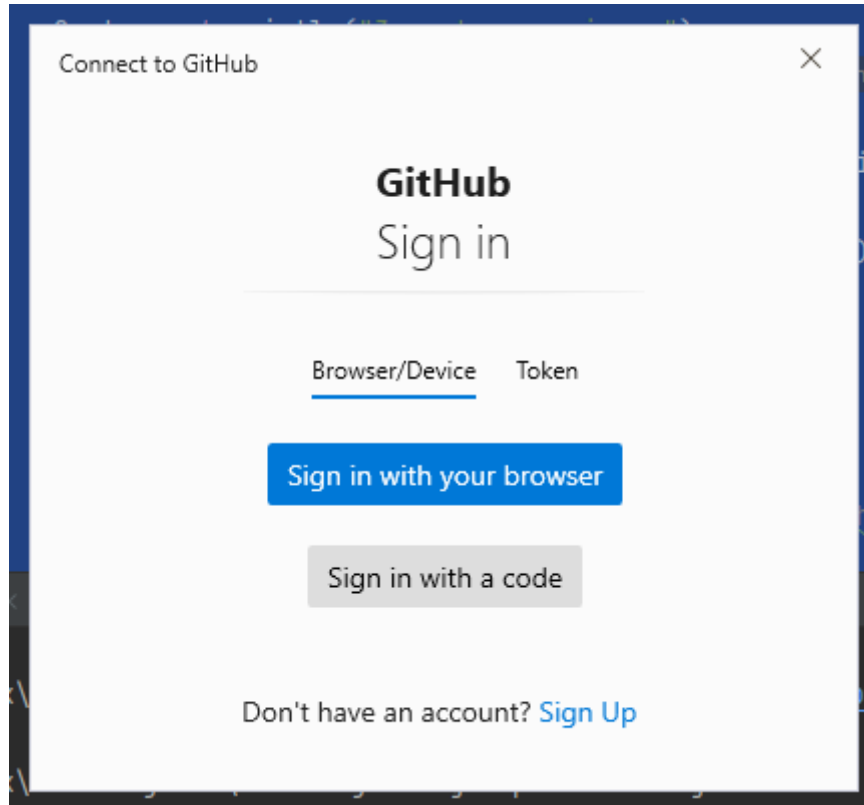
```
C:\Users\ttvux\IdeaProjects\JavaProjects>git commit -m "lan dau dung git"
[master (root-commit) f4cd77b] lan dau dung git
41 files changed, 1022 insertions(+)
create mode 100644 .idea/.gitignore
create mode 100644 .idea/misc.xml
create mode 100644 .idea/modules.xml
create mode 100644 .idea/uiDesigner.xml
create mode 100644 .idea/vcs.xml
create mode 100644 JavaProjects.iml
create mode 100644 out/production/JavaProjects/BreadManager/Bread.class
create mode 100644 out/production/JavaProjects/BreadManager/BreadStore.class
create mode 100644 out/production/JavaProjects/BreadManager/Cheese.class
create mode 100644 out/production/JavaProjects/BreadManager/Olives.class
```



## Push

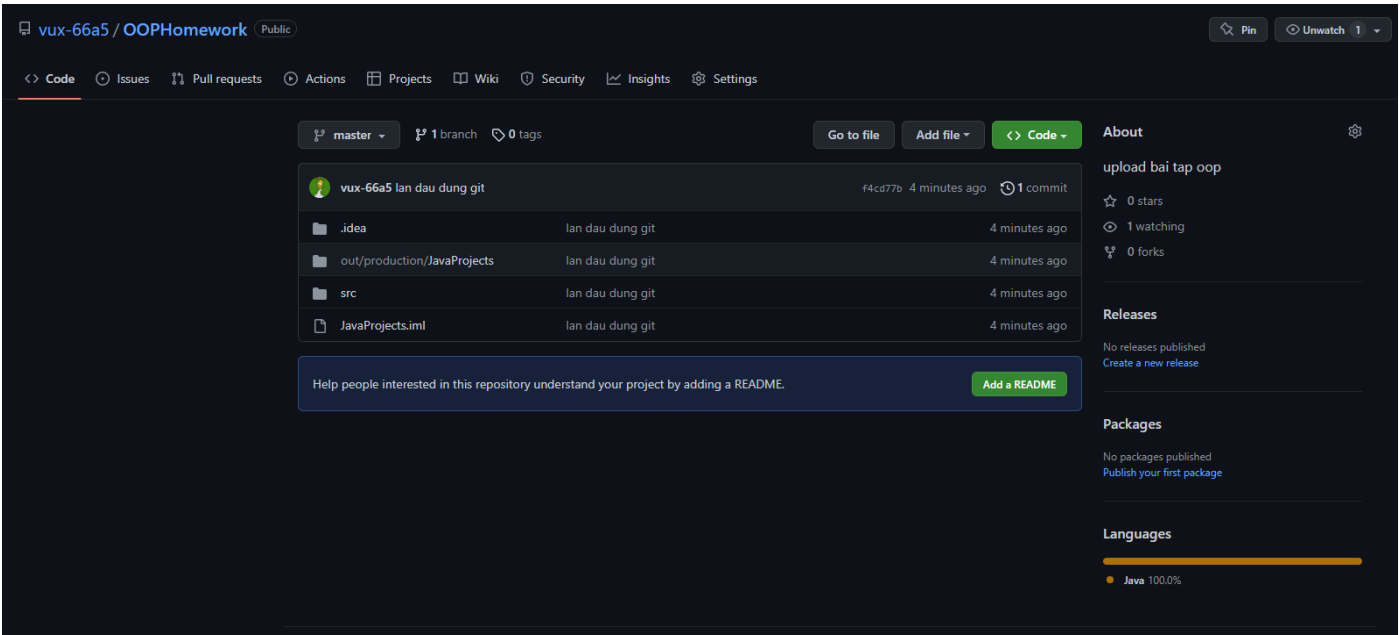
```
C:\Users\ttvux\IdeaProjects\JavaProjects>git push -u origin master  
info: please complete authentication in your browser...
```

### Xác thực qua UI (với Window)



**hoặc xác thực ngay trên Terminal** ( truy cập <https://github.com/settings/tokens> để tạo Token )

# Sau khi xác thực xong code đã được đẩy lên



Hoàn thành