# Objective

The purpose of this report is to demonstrate the integration of GitHub and Jenkins for continuous integration (CI). This setup ensures that every time code is pushed to a GitHub repository, Jenkins automatically pulls, builds, and executes configured tasks.

# Application Code

The application used for this integration is a simple Java-based 'Hello World' project. The repository includes the following:

HelloWorld.java:

public class HelloWorld {

public static void main(String[] args) { System.out.println("Hello, World!");

}

}

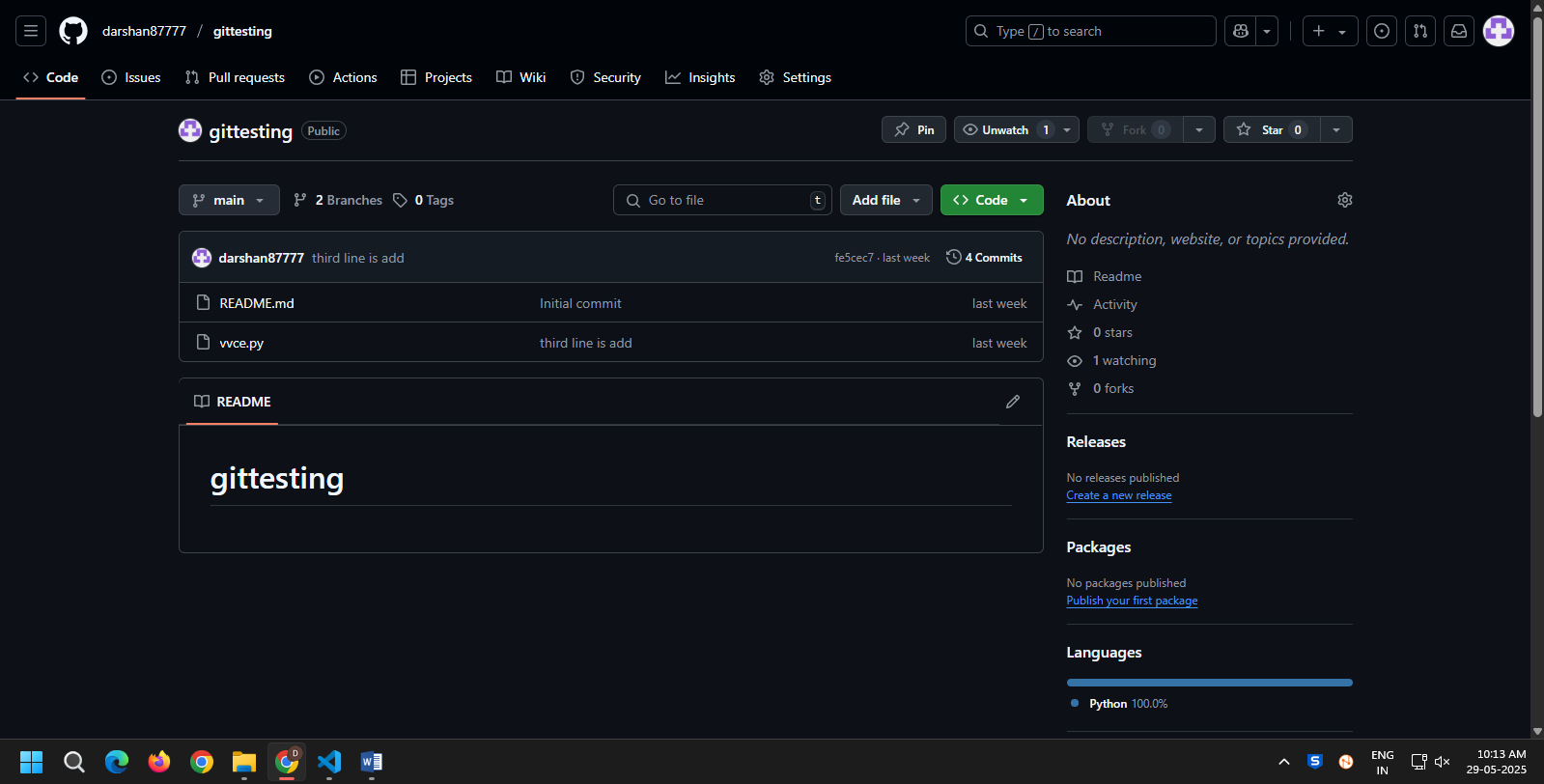
build.bat:

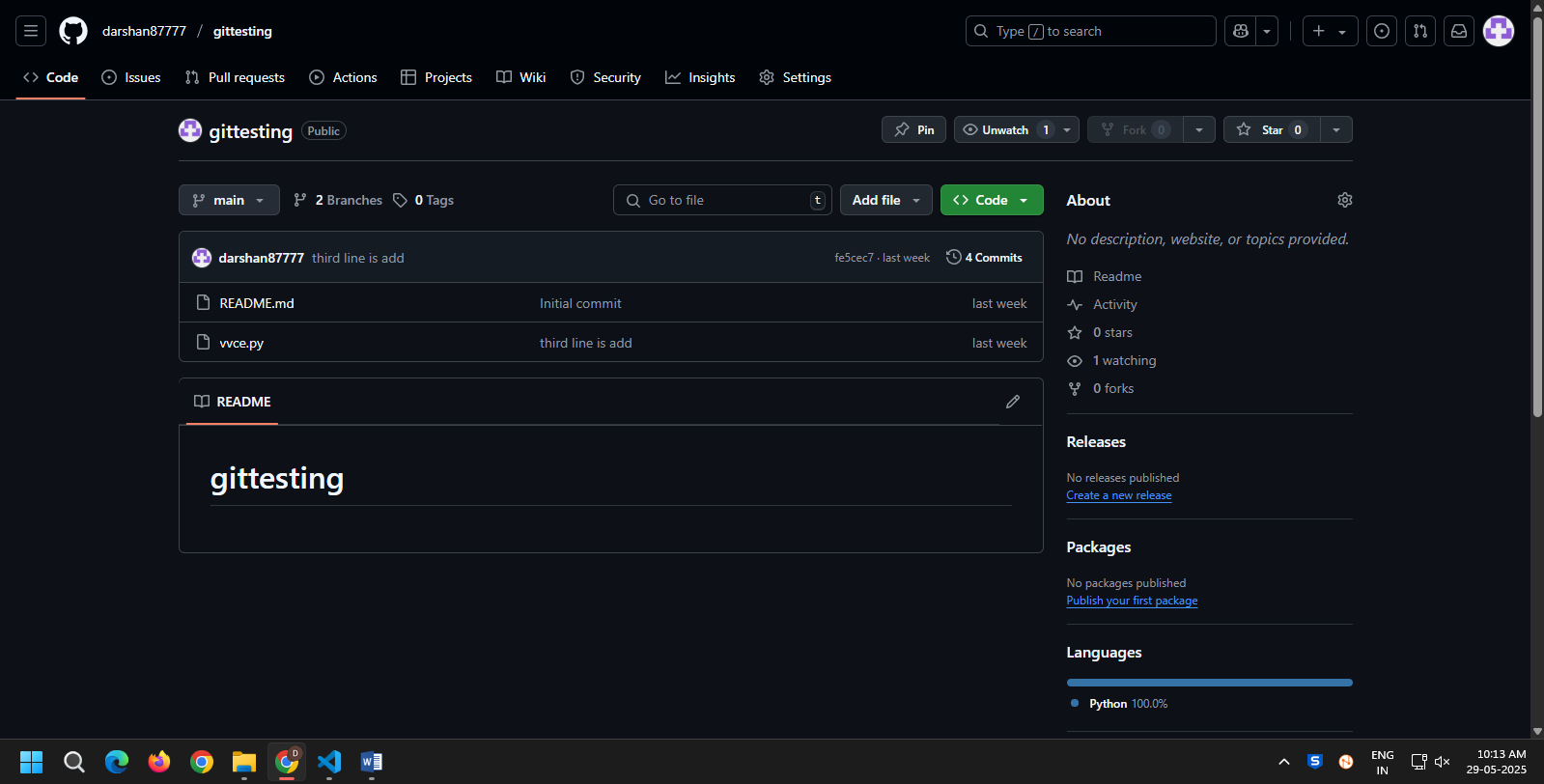
@echo off

javac HelloWorld.java

if %errorlevel% neq 0 exit /b %errorlevel% java HelloWorld

# GitHub Project Screenshot





# Jenkins Project Configuration

* 1. GitHub URL Configuration:
* Repository URL: https://github.com/darshan87777/gittesting.git
  1. Branch Configuration:
* Branch to build: \*/main
  1. Windows Batch Command (Build Step): echo Building Java Project

call build.bat

* 1. Post-build Actions:
* GitHub hook trigger for GITScm polling (optional)
* Publish JUnit test result report (if applicable)
* Email Notification (if configured)

# Result and Validation

After a successful push to GitHub:

* Jenkins pulls the latest code.
* Executes the build.bat script.
* Outputs "Hello, World!" in the Jenkins console.

# Conclusion

The integration of GitHub and Jenkins enables automated builds and streamlined development workflows. By configuring Jenkins to pull from GitHub and execute build scripts, developers receive quick feedback on code changes, reducing errors and improving productivity.