**JOBSHEET 4**

**Version Control System**

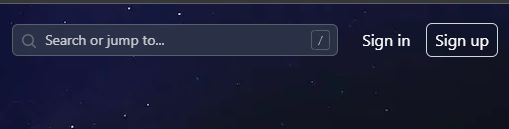
1. **Learning Outcome**

* Students are capable to create a repository account.
* Students are capable to use basic command of GitHub.

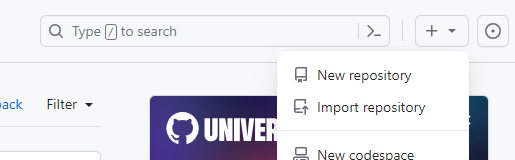
1. **Lab Activity**
   1. **Experiment 1: Getting Started with Github**

**Experiment time : 60 minutes**

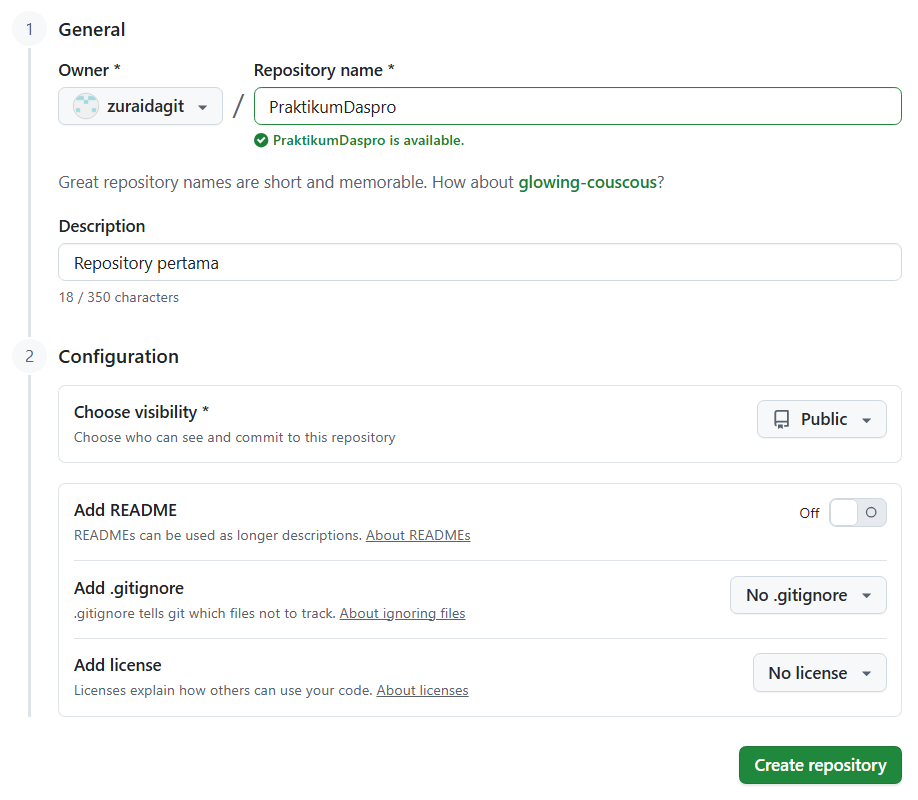
1. Open GitHub website at **https://github.com**.
2. Click the "**Sign up**" button to create a GitHub account.



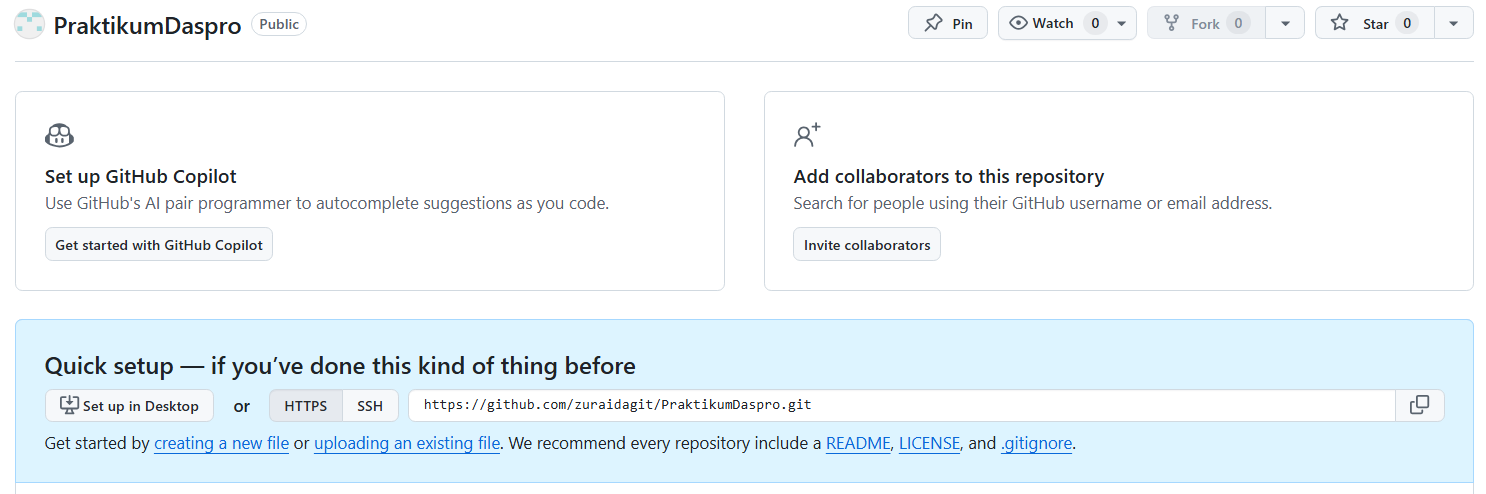
1. Follow the registration steps by providing information needed and continue with email verification.
2. After finishing the registration, log in to your GitHub account, and now you are ready to create a new repository by clicking **"+"** and **"New repository"**



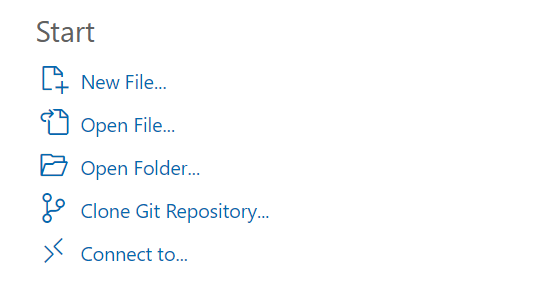
1. Fill in **repository name, description (optional),** and **other needed configurations**.
2. You can set up your repository to be **public** or **private** based on your needs. To complete the you can click “**Create repository”**.



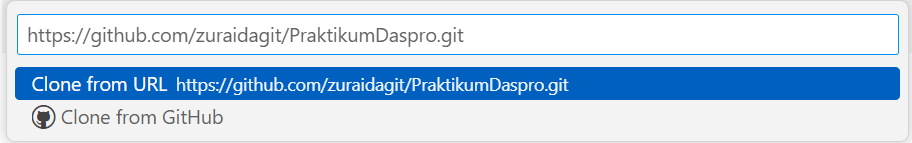
1. Copy the repository **URL** by clicking the Copy **icon** which will later be used in the cloning process.



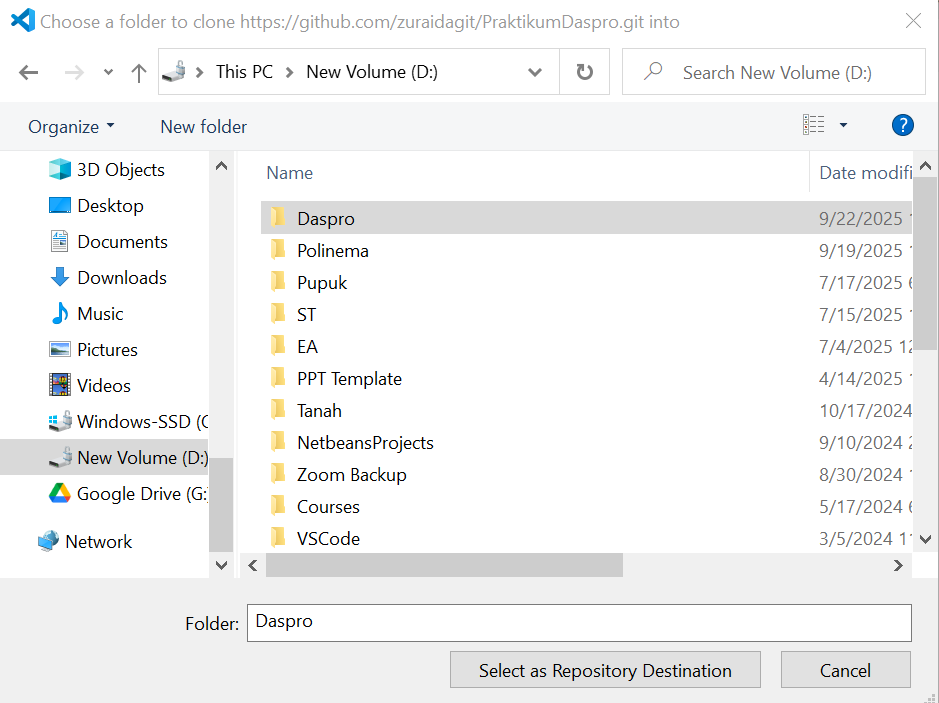
1. The repository that is created in the previous steps, is saved in the Github server. You can manage your repository locally as well. To do so, then you will need to clone the repository into your local computer. First, the Github Client must be installed on your local computer, the installer is provided here <https://git-scm.com/downloads>, Follow the steps to finish the installation.
2. To clone, open VS Code. Select **Clone Git Repository**



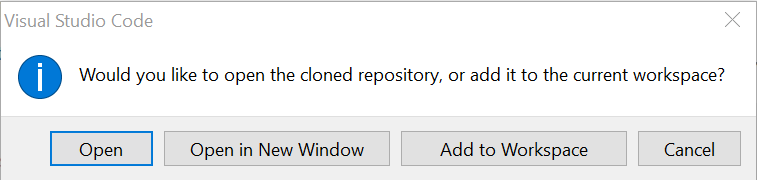
1. Paste repository Url from Github, then enter



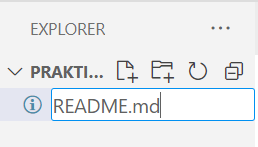
1. Select “Repository Destination”



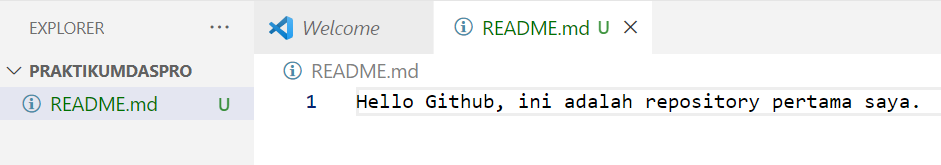
1. Select Open or Open in New Window



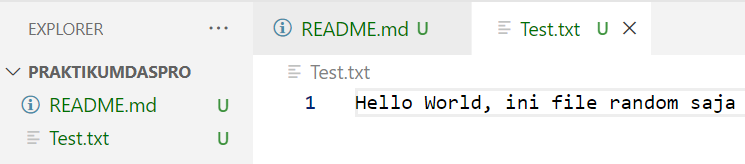
1. You can create new files or modify files in the repository as needed. Experiment by adding new files **“README.md”.**



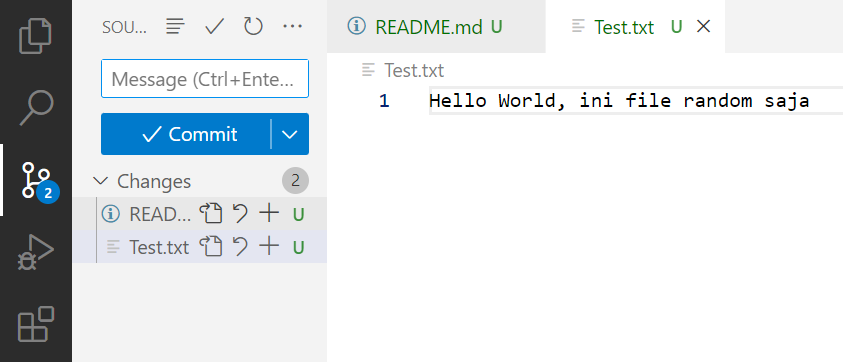
1. Fill the “**README.md**” file with the following text. Note that the U icon indicates that the file is “Untracked” or not yet tracked by Git..



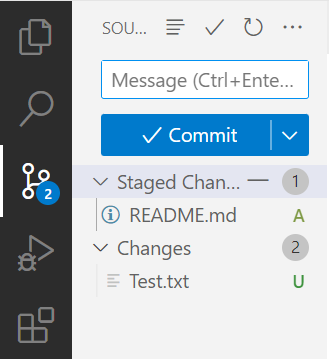
1. To save changes to the local repository, the first step is to **Add**. Adding files to the **staging area**, a temporary area where changes are prepared before being permanently saved to the local repository via **Commit**. In other words, **Adding** means you are selecting which changes to commit.
2. Add the Test.txt file for comparison.



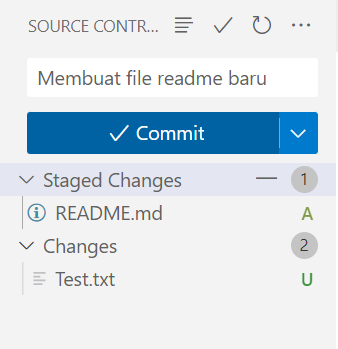
1. Select the Source Control icon on the left, then click the + icon to **add** a **README**.**md** file. Leave the Test.txt file untracked.



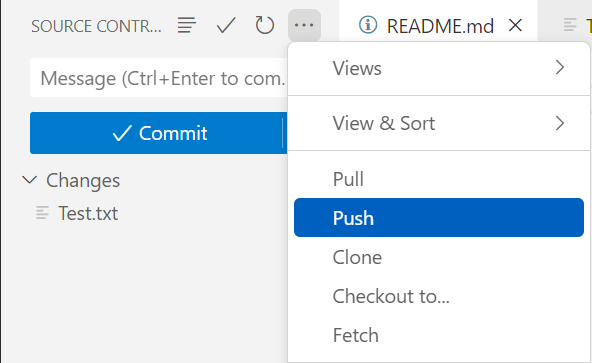
1. The status of the README.md file will change to **A** or **Added**



1. To save changes (create a new version) in the local repository, fill in the **Commit** **Message** and then click **Commit**. Make sure to fill in the commit message with clear notes, such as details about the changes made.



1. At this point, a new version has been created, but only in your local repository. To update the GitHub repository with the version you've saved in your local repository, do a **Push**.



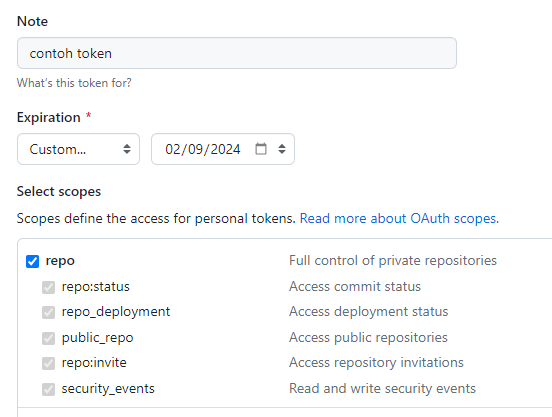
1. When pushing to the repository, if Git asks for a username or password, please use your Git username and password.

If an error like the following appears:

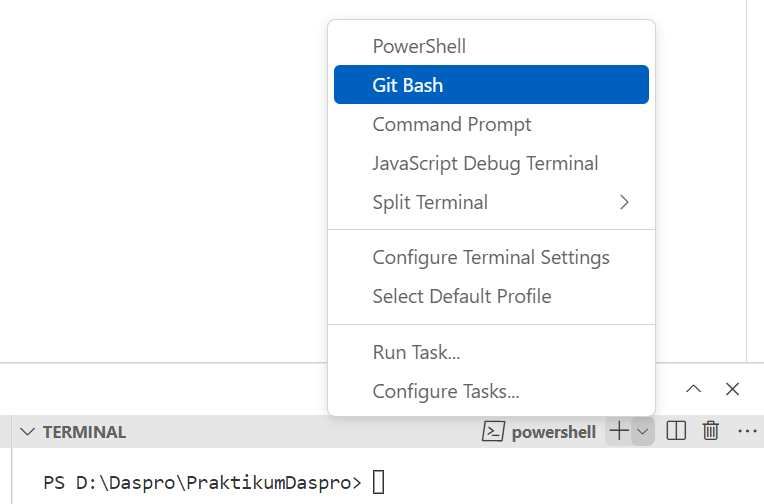
|  |
| --- |
| remote: Permission to <username>/<nama-repository>.git denied  fatal: unable to access ‘https://github/<usename>/<nama-repository>.git: The requested URL returned error: 403 |

Then, all you need to do is create a token to push to the repository. **If the error doesn't appear, please skip to step 25**.

1. To create a token, click your account on GitHub 🡪 Settings 🡪 Developer Settings 🡪 Personal access tokens 🡪 Tokens (classic) 🡪 Generate a personal access token. Fill in the Note, Expiration, and Select scopes fields. Once done, click the **Generate token** button. **Save the token as it cannot be viewed again for use in subsequent pushes..**



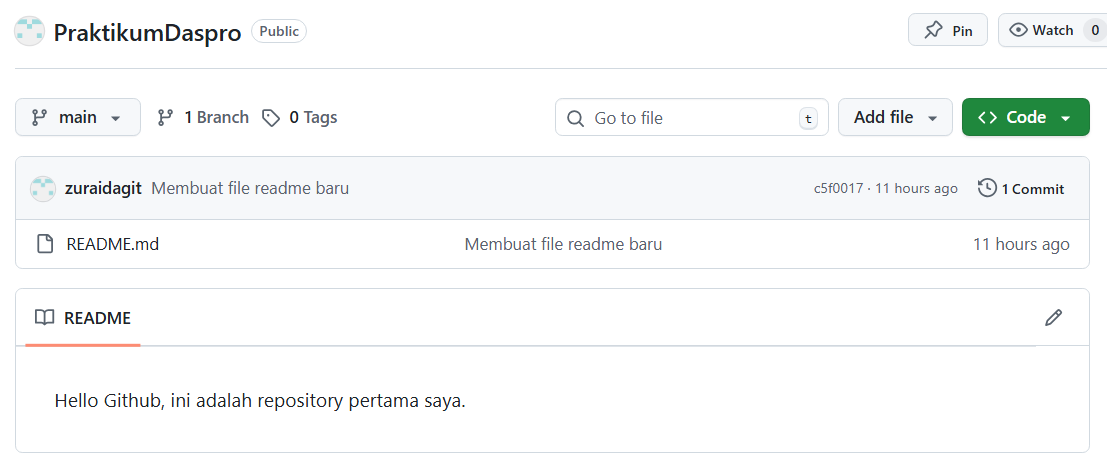
1. Open Git Bash terminal



1. In the git bash terminal, run command:

**git push** [**https://<token>@github.com/<username>/<nama-repository>.git**](https://%3ctoken%3e@github.com/%3cusername%3e/%3cnama-repository%3e.git)

1. Check your GitHub page. The README.md file you created locally is now available in your GitHub repository. Click the filename to view its contents.

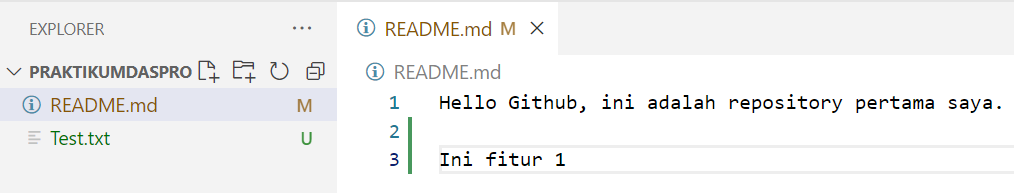


1. If Git asks you to log in every time you **push**, use the command:

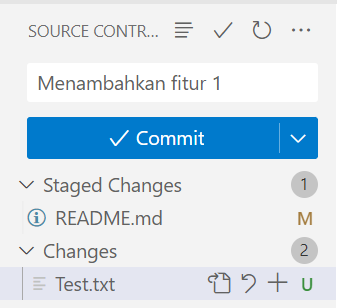
**git config --global user.email “<**[**email@example.com**](mailto:email@example.com)**>”**

**git config --global user.name “<username>”**

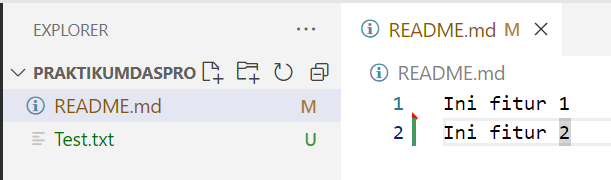
1. To clarify Add, Commit, and Push, try making changes to the README.md file again. If the file is saved, the status of the README.md file will change to **M** or **Modified**, meaning there are changes to the file, but they have not yet entered the staging area (not yet added). Newly added lines are marked with a green line on the line number.



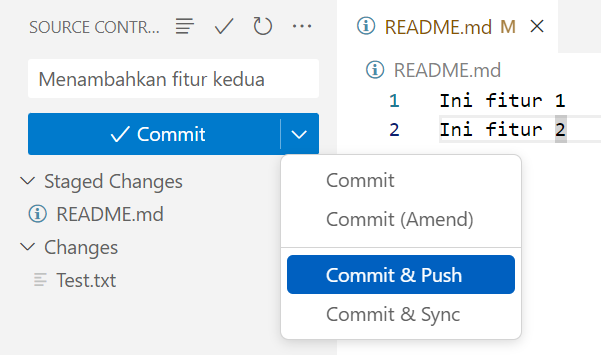
1. Add and Commit the README.md file in the same way as before..



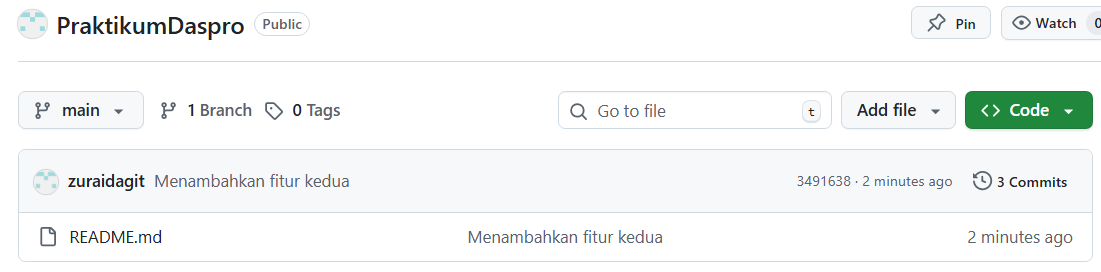
1. Make further changes to the README.md file.



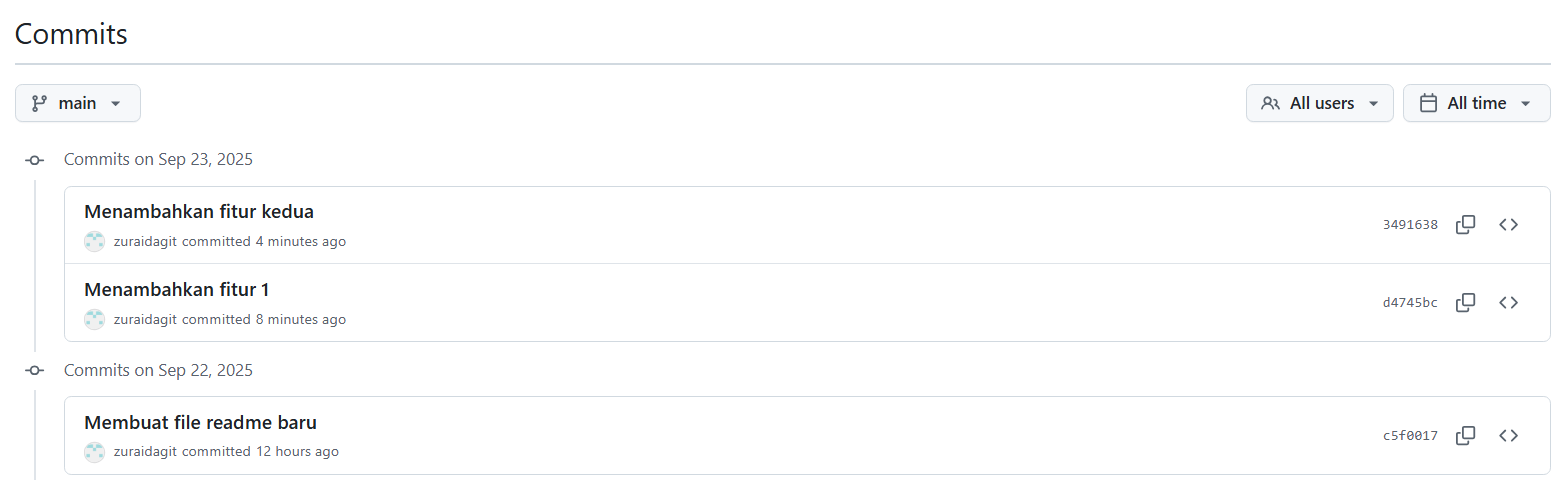
1. Add and Commit again. This time, try to **commit** and **push** at the same time. Commit and push should only be done if you are managing a private repository (not collaborating with others)



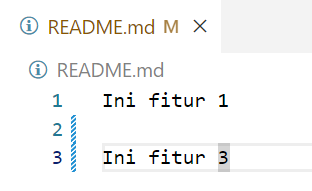
1. Check the Github repository, select the link in the red box to display the commit history..



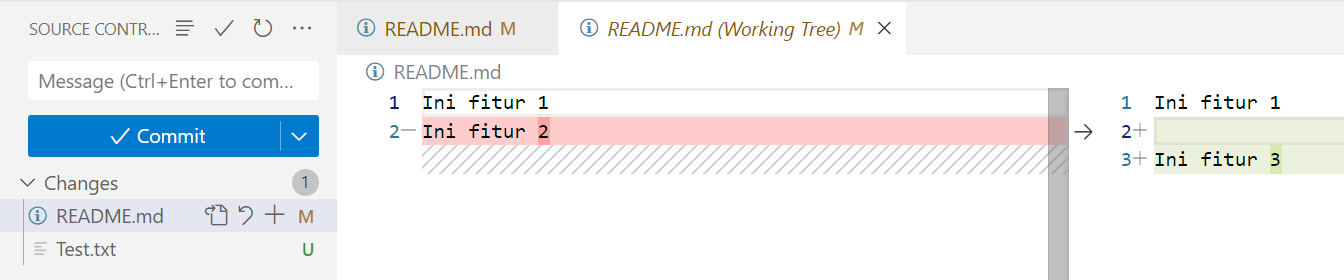
1. You can see every version created every time a commit is made..



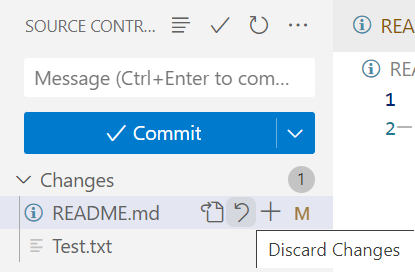
1. Please go back to VS Code and try making changes again to the README.md file as in the following example.:



1. To compare the newly made changes with the last commit, click the file name in Source Control and a comparison will be displayed. Red lines represent deleted lines, while green lines represent added lines.

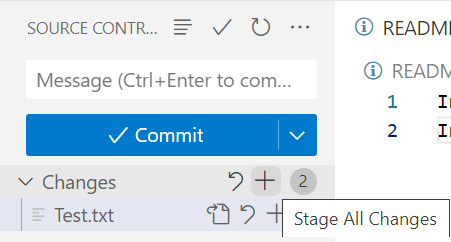


1. For example, if you want to undo changes to the last committed version, click the Undo icon. Select Discard Changes if a confirmation pop-up appears.



**Pertanyaan**

1. Check if the Test.txt file exists on GitHub. If not, why not?
2. Explain the functions of Add, Commit, and Push.
3. Add Test.txt. To add all changes made, use the + icon to the right of Changes. Stage all changes, then Commit and Push..

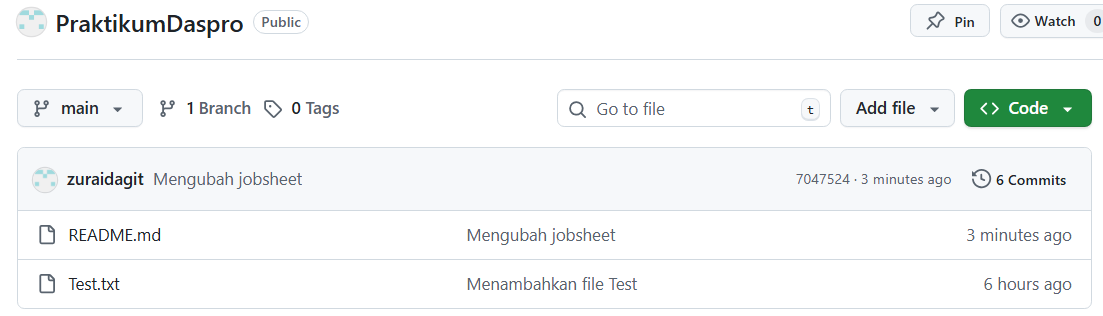


* 1. **Experiment 2 : Basics of Collaboration on Github**

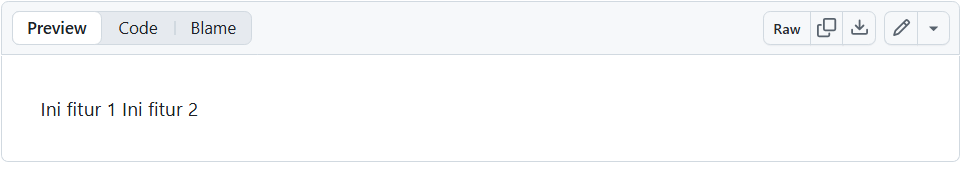
**Experiment time : 30 minutes**

Besides being a personal repository, GitHub is often used as a collaborative tool for working on projects with a team. Try the following experiment to see how collaboration works with GitHub.

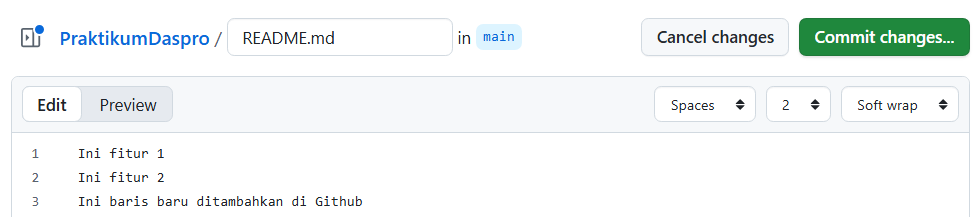
1. Note that when you collaborate with a team on a project, perform a Pull to update the project version in your local repository with the latest version on GitHub. Best practice: perform a Pull every time you start work and every time you push.
2. For example, make changes to the README.md file on GitHub. Click on the file name.



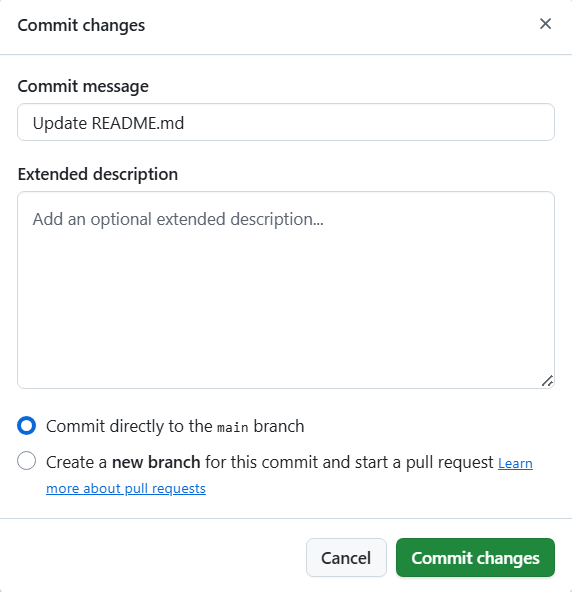
1. Select Edit icon



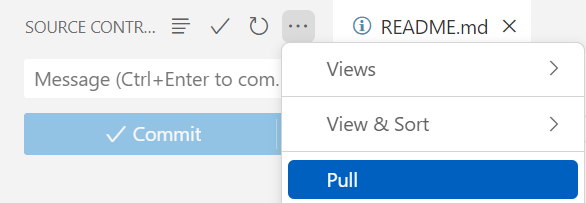
1. Change file then **Commit changes**



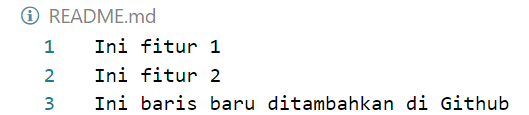
1. Click **Commit changes**



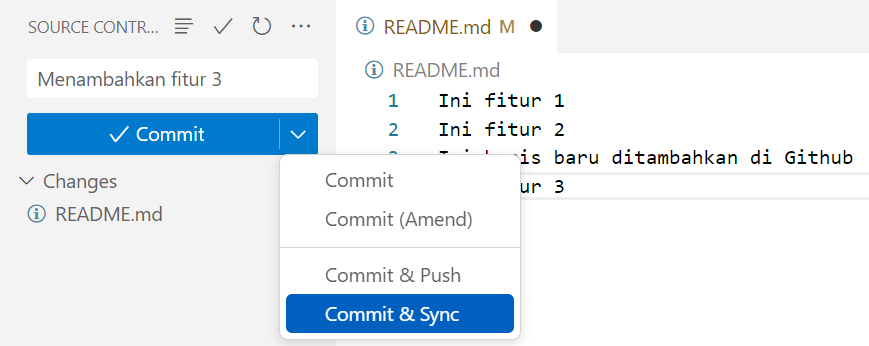
1. To fetch/ download changes on Github, click More (…) 🡪 **Pull**.



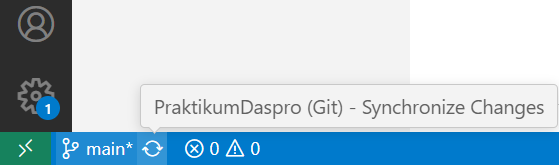
1. After the Pull is done, the README.md file in the local repository will also be updated.



1. You can also select the Commit & Sync option. Sync means synchronizing your local repository and GitHub, so you'll pull from GitHub and then push to GitHub directly. To try it out, add a new line to README.md, click **Add** then **Commit & Sync**.



1. Another option is to use the Sync icon in the status bar.

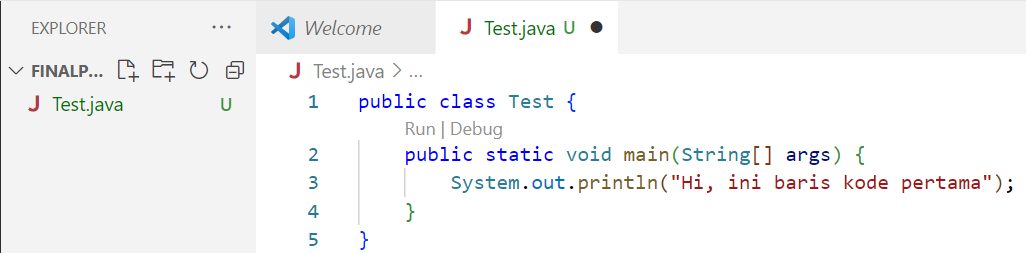


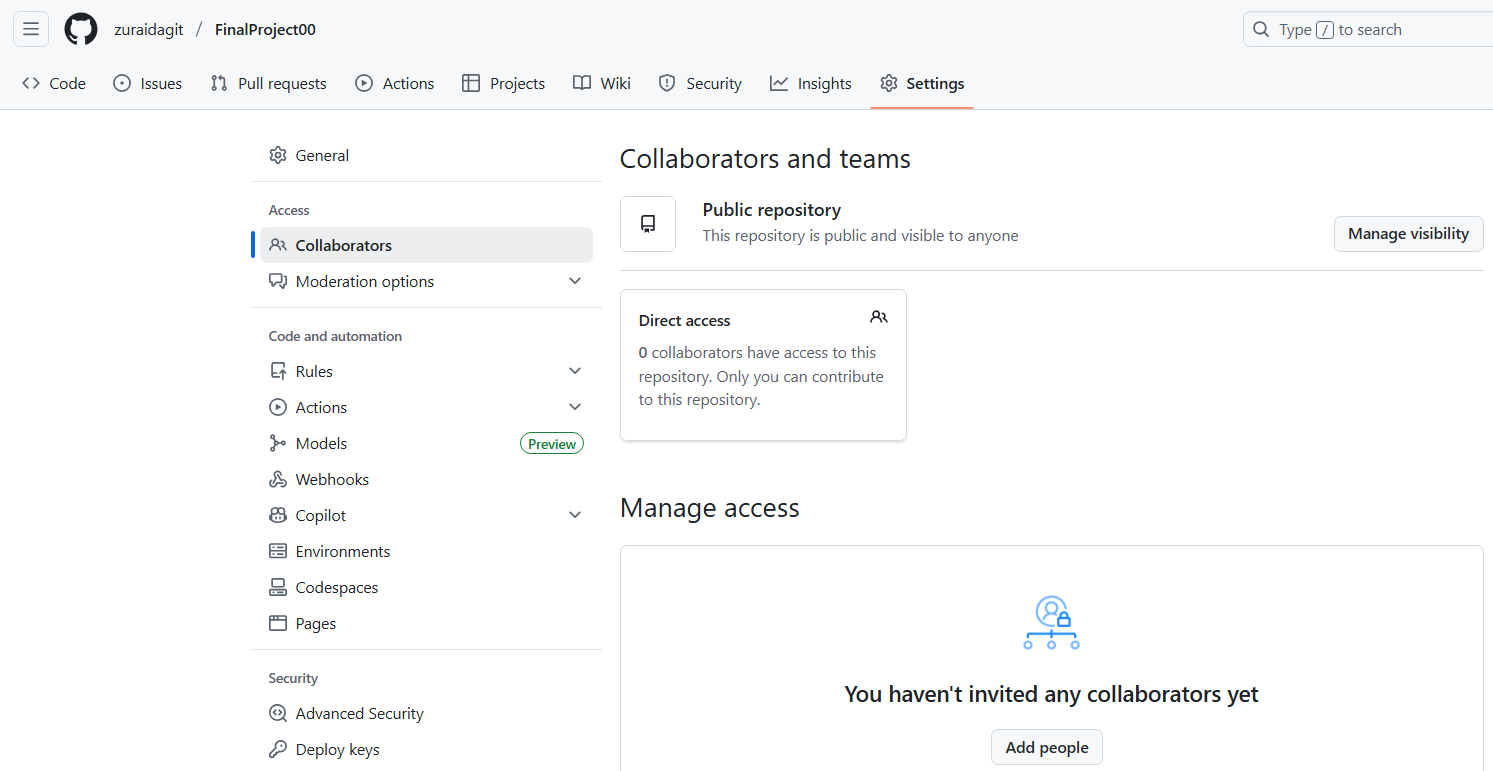
1. Sync should also only be performed if the repository is managed privately. If working with a team, perform **Pull** and **Push** operations separately to better handle potential conflicts.

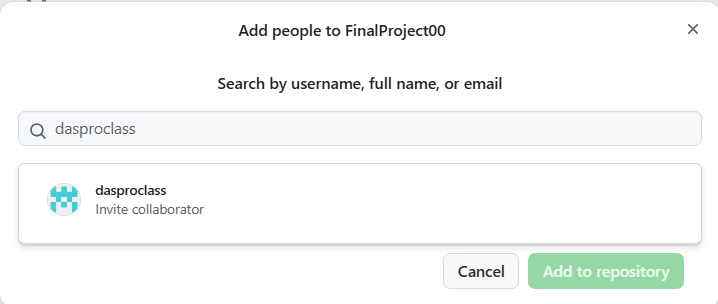
**Assignment**

**Time : 60 minutes**

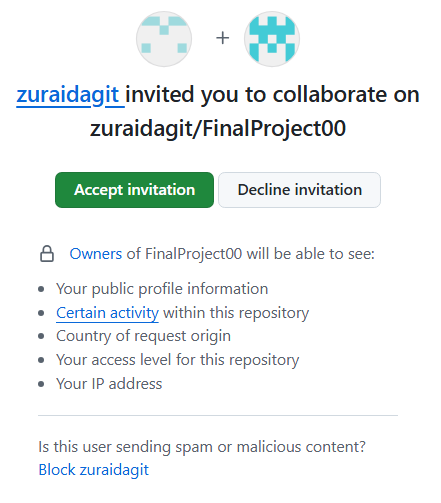
1. To try collaborating, create a new repository named FinalProject<NoAbsen>.
2. Clone it to your local repository as in Experiment 1.
3. Try adding a new file, for example, Test.java, containing the main() function.



1. Do Add, Commit, dan Push.
2. You can collaborate with team members by selecting Settings → Collaborators → Add people.
3. For example, add 1 of your friends based on their Github username. Invite collaborators then **Add**.



1. Ask your friend to open the email sent by GitHub. Click the “Accept or decline this invitation” link. Select “Accept invitation”.

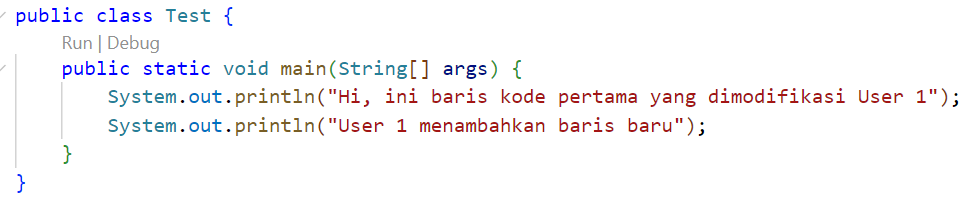


1. 8. Ask your friend to clone the repository to their laptop/PC, then make changes to the Test.java file or add new files. Add, Commit, and Push the changes to save them to the GitHub repository.
2. 9. Do a Pull so that your local repository also stores the latest version of the project.
3. Take turns cloning your friend's repository, then do some more experimenting with GitHub (add, commit, push, pull) to get used to it. Screenshot of commit history from GitHub.

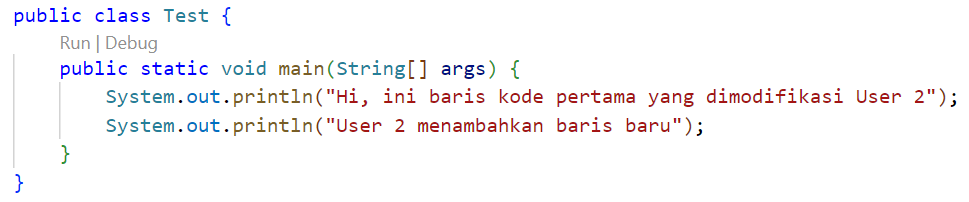
**Notes: Merge Conflict**

Merge conflict occurs when Git cannot automatically merge two versions, for example, when two people make changes to the same (or adjacent) lines in a file. Merge conflicts can also occur when changes are made to a file that was later deleted by someone else..

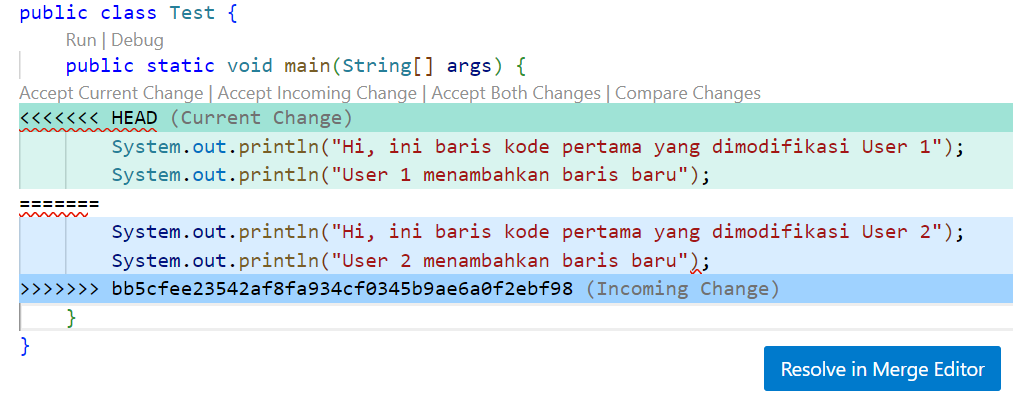
1. For example, suppose Test.java is modified by you as User1 as follows.



1. User2 also changed the Test.java file (against the same version) and has pushed it to Github.

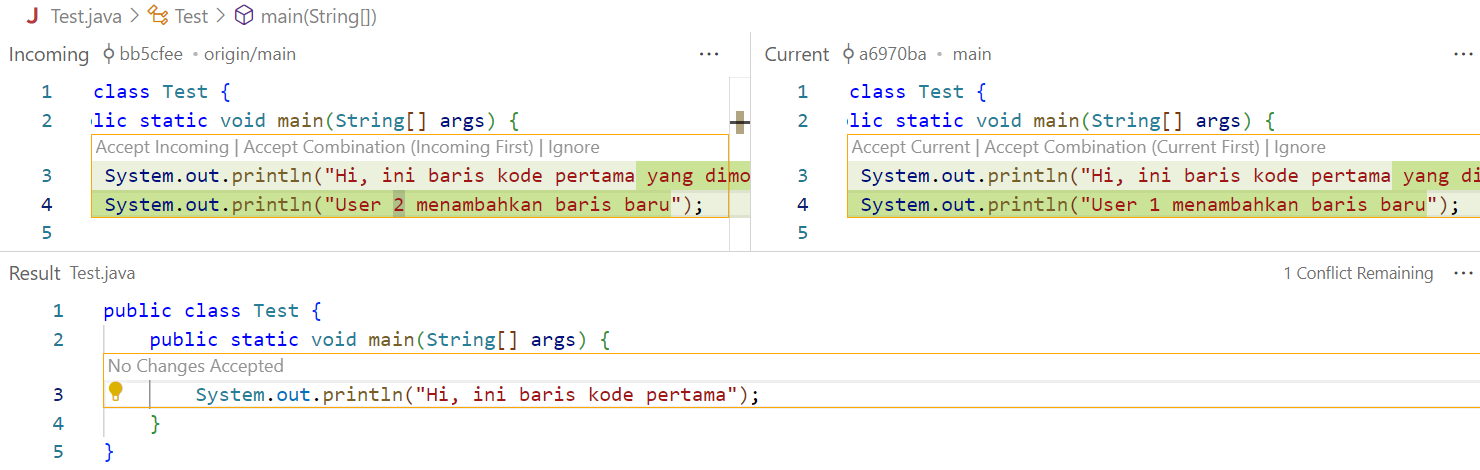


1. When pulling, a merge conflict will occur. The green Current Change represents the version in your local repository, while the blue Incoming Change represents the version on GitHub..

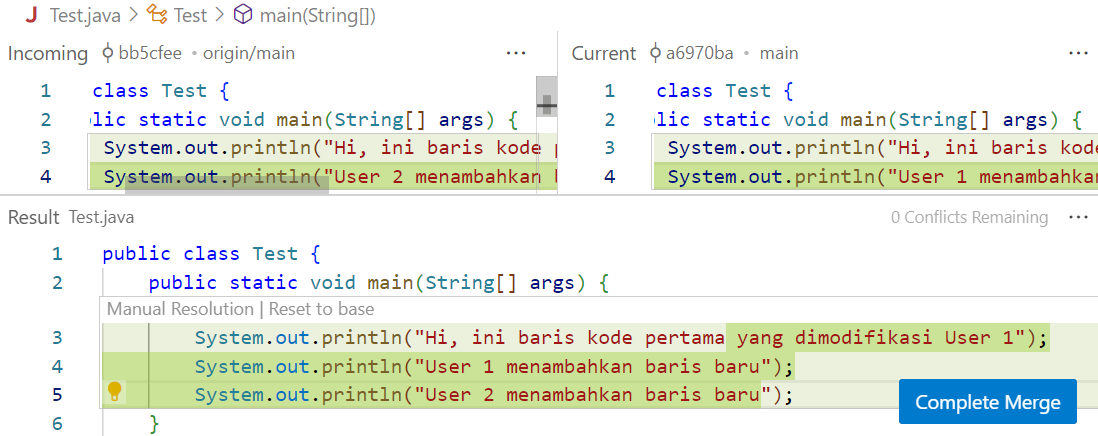


1. You can choose several options, namely:

* Accept Current Change: choose to keep your local repository version and ignore the GitHub version.
* Accept Incoming Change: choose the GitHub version and ignore your local repository version.
* Accept Both Changes: combine both versions. This option will usually cause errors and requires manual editing.
* Compare Changes: compare both versions before selecting.Another option is **Resolve in Merge Editor**.



1. You can select the desired version in the Incoming section (click Accept Incoming | Accept Combination(Incoming First) | Ignore) or the Current section (click Accept Current | Accept Combination(Current First) | Ignore) then modify it manually in the Result section if necessary.
2. If merge is as expected or correct, click **Complete Merge**.



1. Do **Commit** then **Push**.