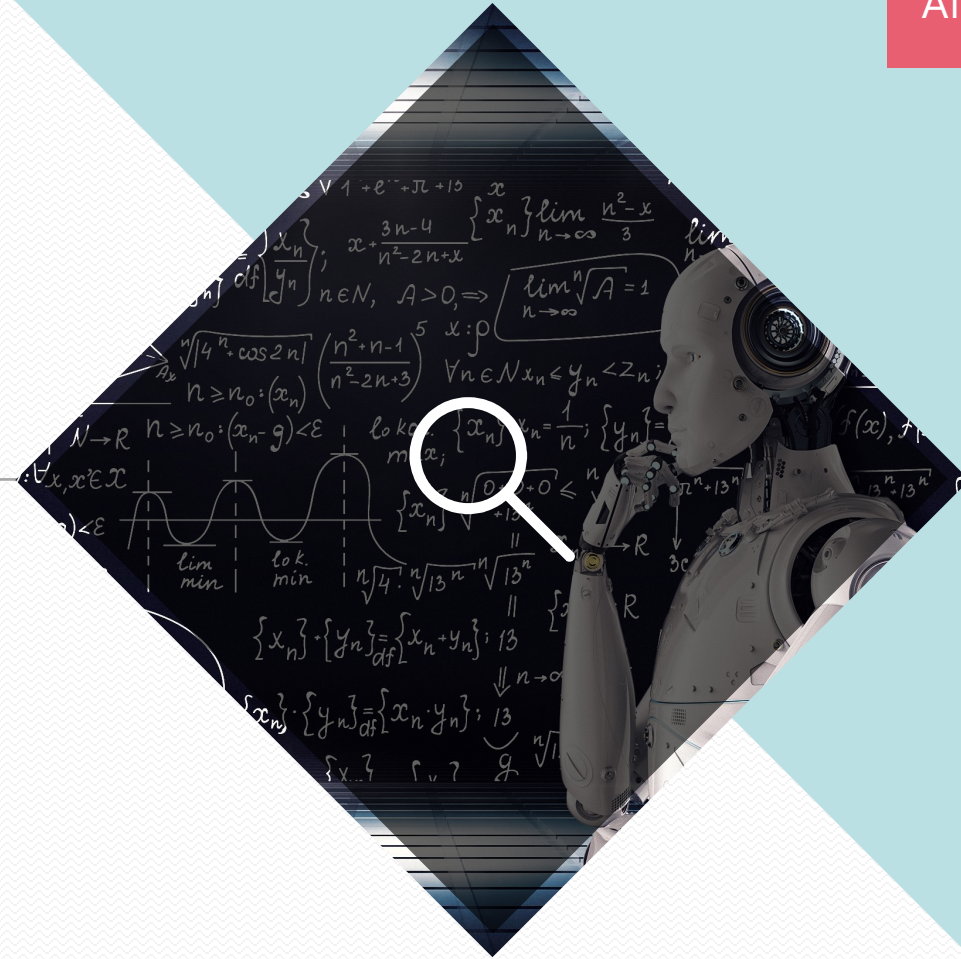


GitHub



“As more and more artificial intelligence is entering into the world”

CONTENTS

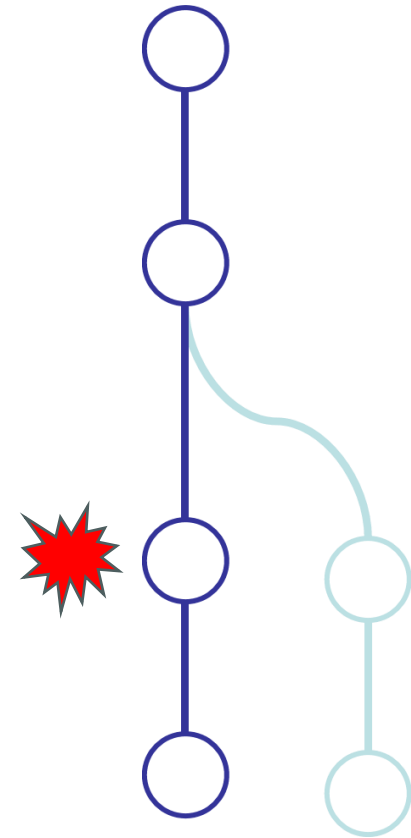
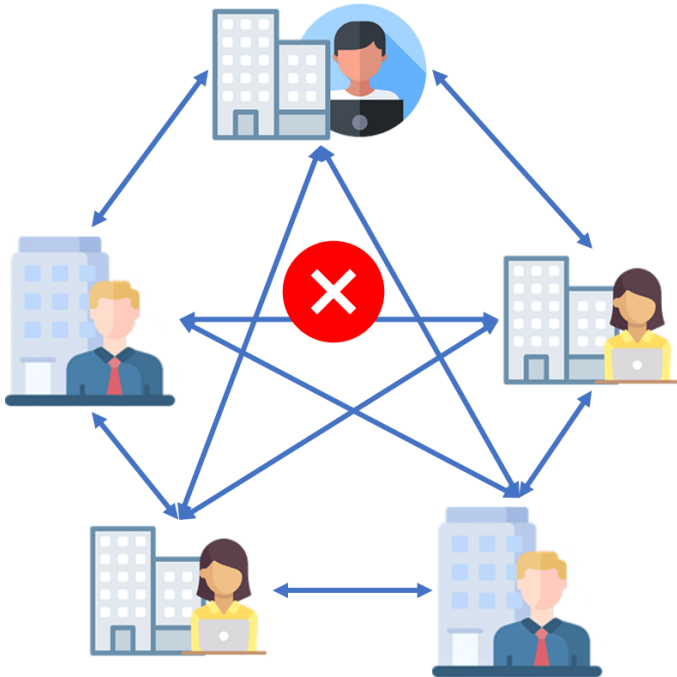
- *Before Start*
 - Why GitHub?
- *GitHub*
 - What is GitHub?
 - Getting Started
 - Setting Up a GitHub Repository
 - Creating Issue
 - Creating Pull Request
 - Ground Rule for Intel Curriculum
 - Homework Workflow



Why GitHub?

Why is Software Management needed?

- Collaboration
- Storing/Restoring
- Figure out what happened



Git vs GitHub : What's the difference?

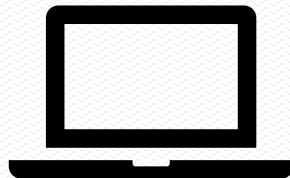


Git

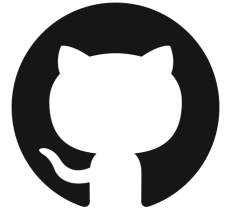
Used for
version
control

Tracks
changes
made to a
file

Installed
locally on
computer



GitHub



Cloud-
based



Used for
hosting Git
repository

Provides a
web
interface to
view file
changes

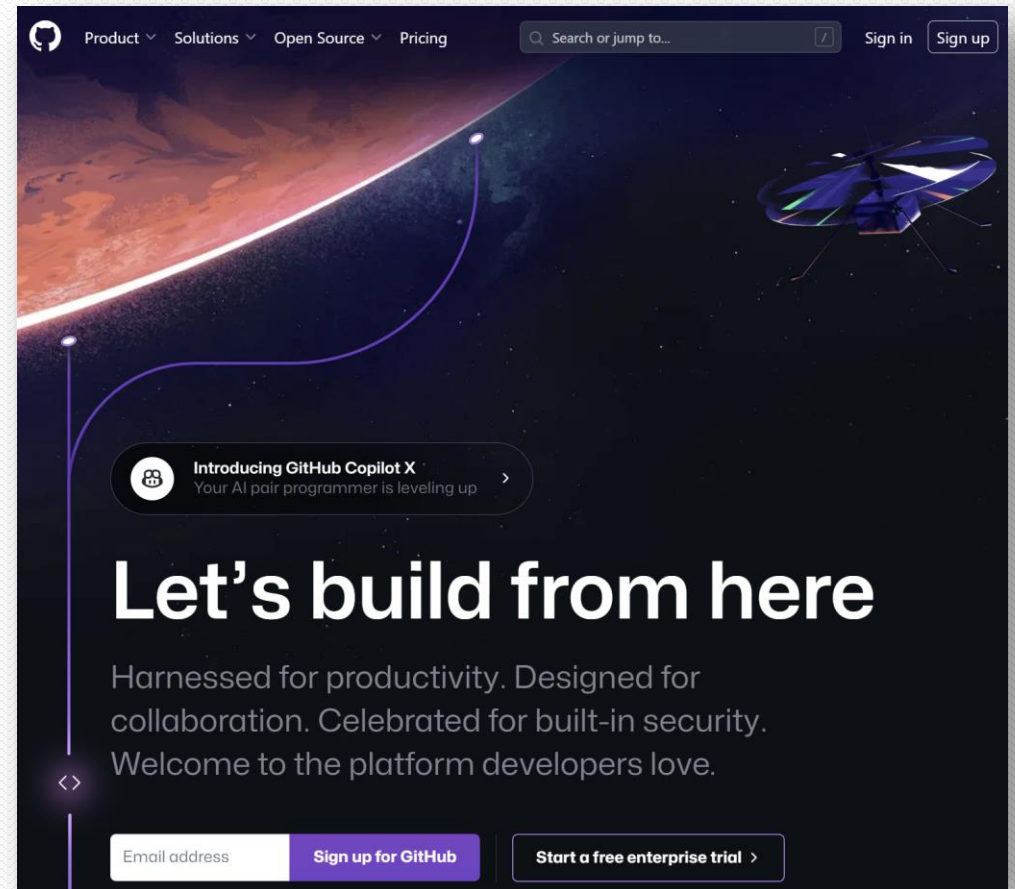


GitHub

What is GitHub?

- A platform and cloud-based service for software development and version control **using Git**, allowing developers to store and manage code.
- It is commonly used to host open source software development projects.
- Link : <https://github.com/>

- Without GitHub? You can setup a remote Git repository in your local server for collaboration.



Getting Started

- GitHub Desktop

- An application that enables to interact with GitHub using a GUI instead of the command line or a web browser.
- Use GitHub Desktop to complete most Git commands from local machines with visual confirmation of changes.
- Supported OS
 - Windows
 - macOS
- Link : <https://desktop.github.com/>

- GitHub CLI

- An open source tool for using GitHub from your computer's command line.
- Supported OS
 - Windows
 - macOS
 - Linux
- Link : <https://github.com/cli/cli>
- Command to authenticate

```
$ gh auth login
```


Getting Started

- Ensure the email address that will be associated with the commits you push from the command line as well as web-based Git operations you make

1 Settings

서울기술교육센터 (kccistc)
Your personal account

Public profile
Account
Appearance
Accessibility
Notifications

Access
Billing and plans
Emails
Password and authentication
Sessions
SSH and GPG keys
Organizations
Enterprises
Moderation

Emails

wckang@korcham.net – Primary

This email will be used for account-related notifications and can also be used for password resets.

- Not visible in emails
This email will not be used as the 'from' address for web-based Git operations, e.g., edits and merges. We will instead use 148169295+kccistc@users.noreply.github.com.
- Receives notifications
This email address is the default used for GitHub notifications, i.e., replies to issues, pull requests, etc.

Add email address

Email address Add

2 Primary email address

Because you have email privacy enabled, wckang@korcham.net will be used for account-related notifications as well as password resets. 148169295+kccistc@users.noreply.github.com will be used for web-based Git operations, e.g., edits and merges.

wckang@korcham.net Save

Getting Started

- The first thing you should do when you install Git is to set your user name and email address.

```
$ git config --global user.name "FirstName LastName"  
$ git config --global user.email "email@example.com"
```

- Check git setting

```
$ git config --list  
  
user.name=FirstName LastName  
user.email=email@example.com  
color.status=auto  
color.branch=auto  
...
```

- Another way to check git setting
 - Open **.gitconfig**
 - Windows
 - C:\Users\<user name>\.gitconfig
 - Ubuntu
 - /home/<user name>\.gitconfig

Practice : Getting Started

- Goal
 - Create GitHub account
 - Install GitHub CLI : <https://github.com/cli/cli>

```
$ gh auth login
```

```
? What account do you want to log into? GitHub.com
```

```
? What is your preferred protocol for Git operations? HTTPS
```

```
? Authenticate Git with your GitHub credentials? Yes
```

```
? How would you like to authenticate GitHub CLI? Login with a web browser
```

```
! First copy your one-time code: D8FF-6166
```

```
Press Enter to open github.com in your browser...
```

```
✓ Authentication complete.
```

```
- gh config set -h github.com git_protocol https
```

```
✓ Configured git protocol
```

```
! Authentication credentials saved in plain text
```

```
✓ Logged in as mokiya
```

Practice : Getting Started

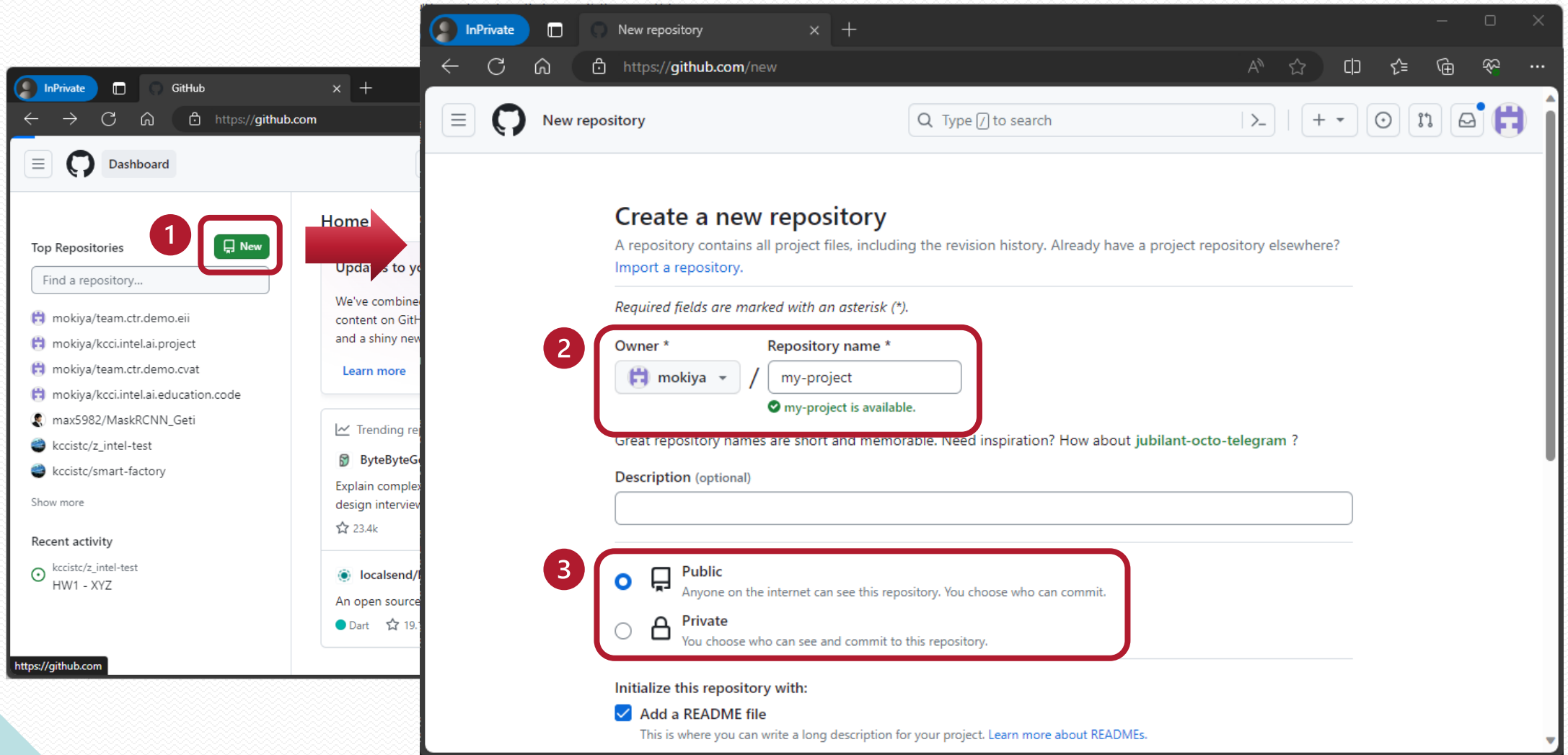
- Goal
 - Set primary email in GitHub
 - Set name & email in local machine

```
$ sudo apt-get update
$ sudo apt-get install git

$ git config --global user.name "FirstName LastName"
$ git config --global user.email "email@example.com"
```

Setting Up a GitHub Repository

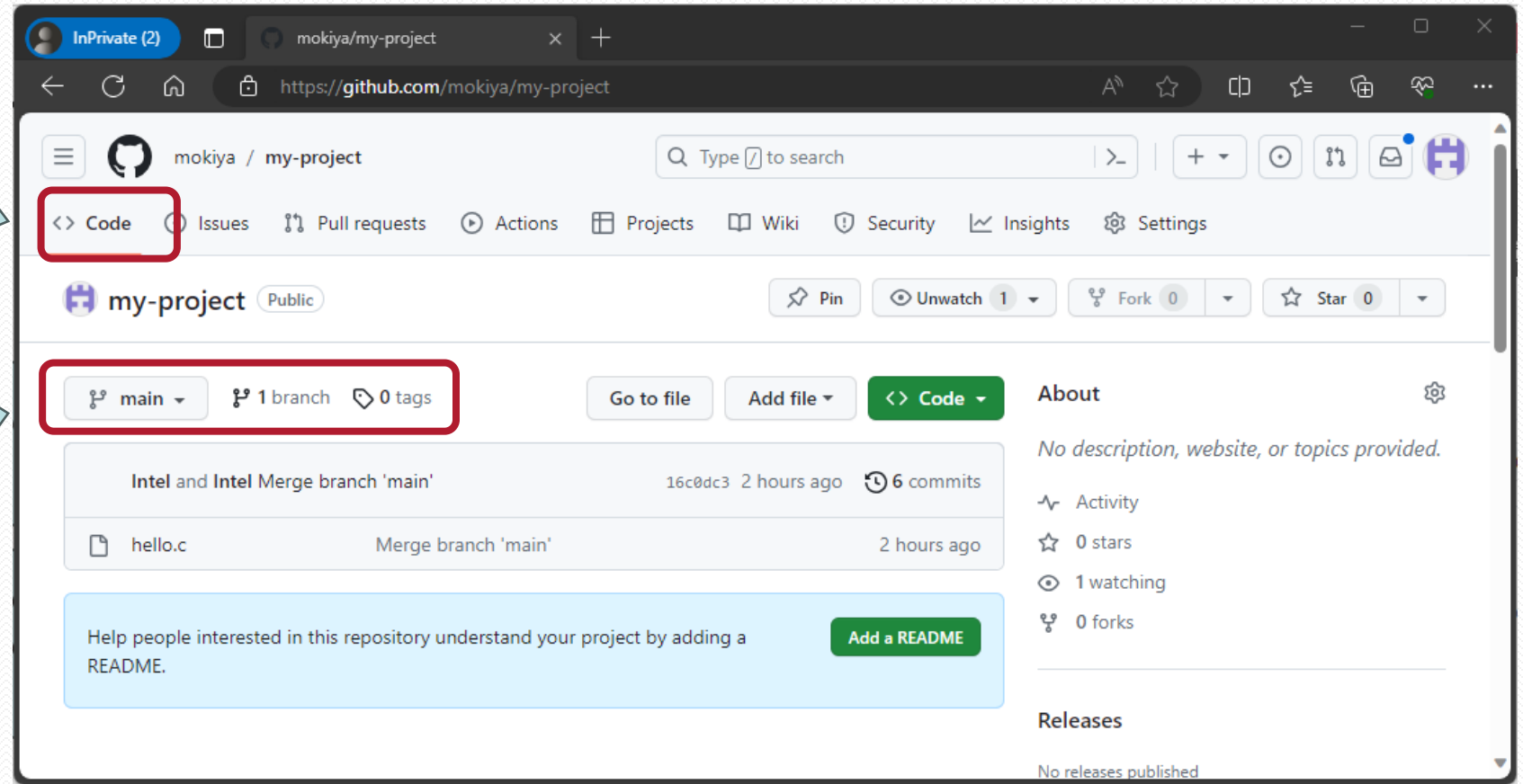
- Create new project in the GitHub



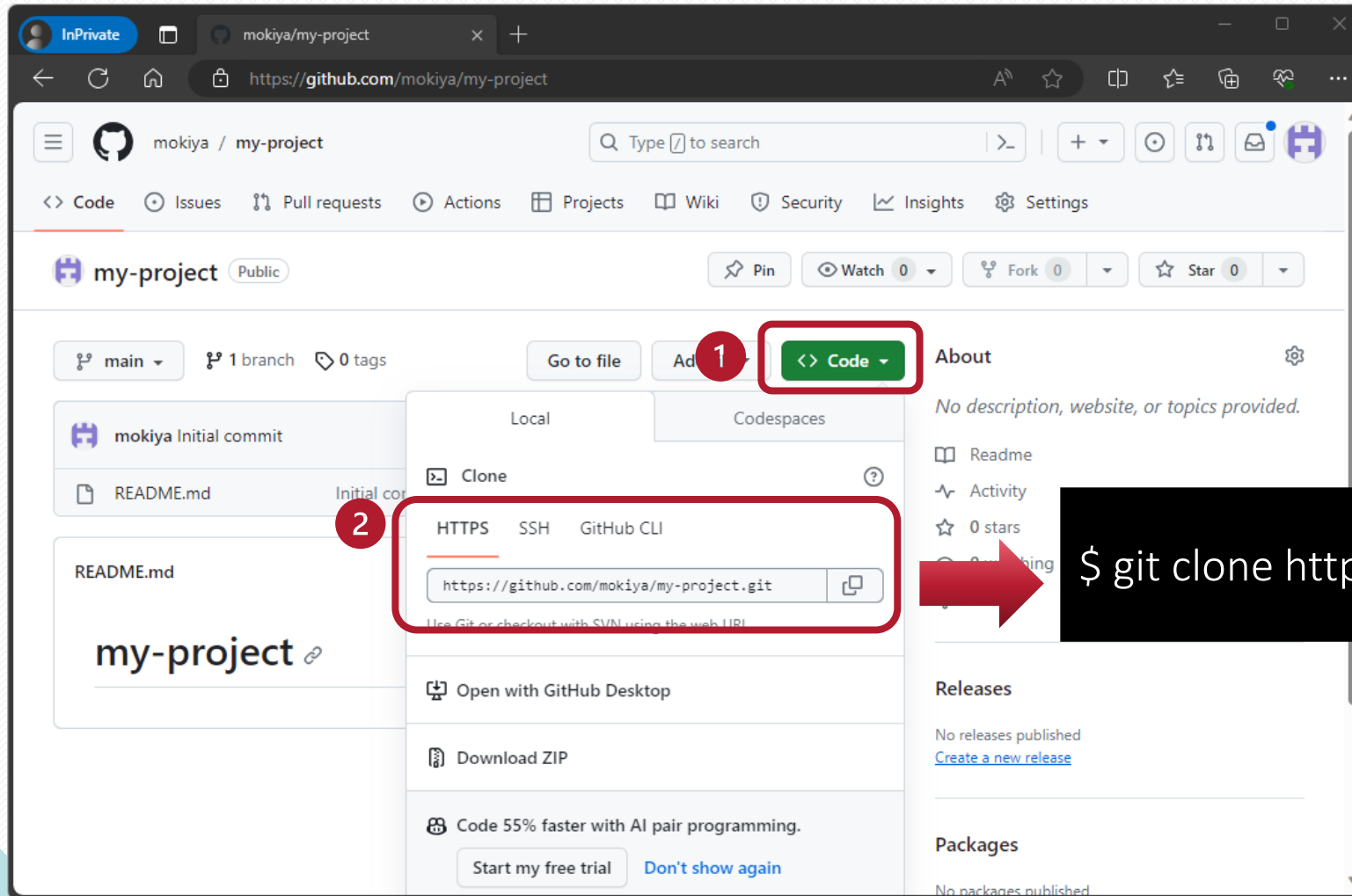
Setting Up a GitHub Repository

Web Interface to see project source codes

Choose branch or tag what you want to a specific snapshot



Setting Up a GitHub Repository



```
$ git clone https://github.com/mokiya/my-project.git
```

Setting Up a GitHub Repository

- Once you successfully created GitHub repository, you can start adding commits from scratch and pushing commits to new GitHub repository.
- If you already have an existing repository, you can push an existing repository.

```
$ cd ~/existing-repo
```

```
$ git remote add github https://github.com/new-repo-url.git
```

```
$ git branch -M main
```

```
$ git push -u github main
```


Practice : Setting Up a GitHub Repository

- Goal
 - Create new GitHub repository
 - Practice same way as we did last time, but try to practice using github.
 - Hints)
 - git clone
 - git remote
 - git push

Practice : Setting Up a GitHub Repository

- An existing repository

```
$ cd ~/git-training  
$ git clone ~/git-training/my-project.git -b main project-z  
$ cd ./project-z  
  
$ git remote add github https://github.com/mokiyu/my-project.git  
$ git branch -M main  
$ git push -u github main
```

- Clone new Github repository to check, or visit GitHub web.

```
$ mkdir ~/github-training  
  
$ cd ~/github-training  
$ git clone https://github.com/mokiyu/my-project.git -b main  
$ cd ./my-project  
$ git log
```

Intel Curriculum GitHub Repository

- Materials, sample codes, homework, team projects and so on for Intel curriculum will be managed in Intel Curriculum GitHub Repository.

Intel Curriculum GitHub Repository : <https://github.com/kccistc/intel-03/>

The screenshot shows the GitHub repository page for 'intel-03' by user 'kccistc'. The repository is public and has 3 watchers, 2 forks, and 0 stars. The main branch is 'main'. The repository contains a patch 1 (#1) with 3 commits. The file list includes 'class01', 'class02', 'doc', '.gitattributes', '.gitignore', '.gitmodules', and 'README.md'. The README.md file is selected, showing the title '상공회의소 서울기술교육센터 인텔교육 3기' and a 'Clone code' button. The right sidebar shows the repository's activity, including a list of files, a 'Readme' link, and a 'Releases' section.

intel-03 Public

Watch 3 Fork 2 Star 0

main 1 branch 0 tags

Go to file Add file <> Code

kccistc Kccistc patch 1 (#1) da817bc 2 days ago 3 commits

File	Commit	Time
class01	Kccistc patch 1 (#1)	2 days ago
class02	Kccistc patch 1 (#1)	2 days ago
doc	Intel Edge AI Curriculum	2 days ago
.gitattributes	Intel Edge AI Curriculum	2 days ago
.gitignore	Intel Edge AI Curriculum	2 days ago
.gitmodules	Intel Edge AI Curriculum	2 days ago
README.md	Kccistc patch 1 (#1)	2 days ago

README.md

상공회의소 서울기술교육센터 인텔교육 3기

Clone code

About

No description, website, or topics provided.

Readme

Activity

0 stars

3 watching

2 forks

Report repository

Releases

No releases published

Packages

No packages published

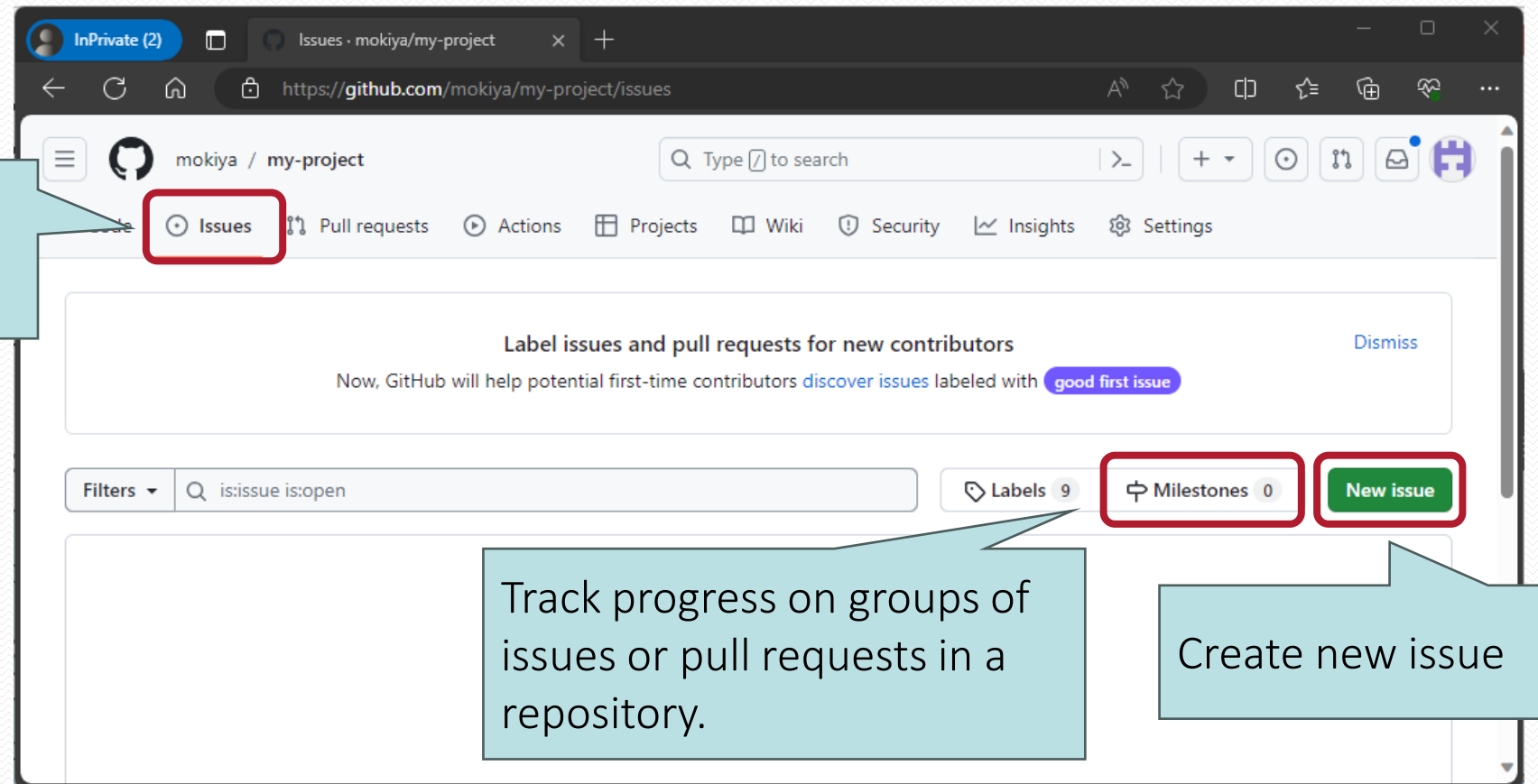
Languages

Python 60.5% Jupyter Notebook 39.4%

Creating Issue

- GitHub Issues to track ideas, feedback, tasks, or bugs for work on GitHub.

Web Interface to see project source codes



Practice : Creating Issue

- Goal
 - Submit new issue with following contents in Intel Curriculum Repository

1 [김영희] My practice repository

2

- * Name : 김영희
- * Github account : abcde
- * My practice repository URL : <https://github.com/abcde/my-project.git>

3 Submit new issue

[English Name] My practice repository

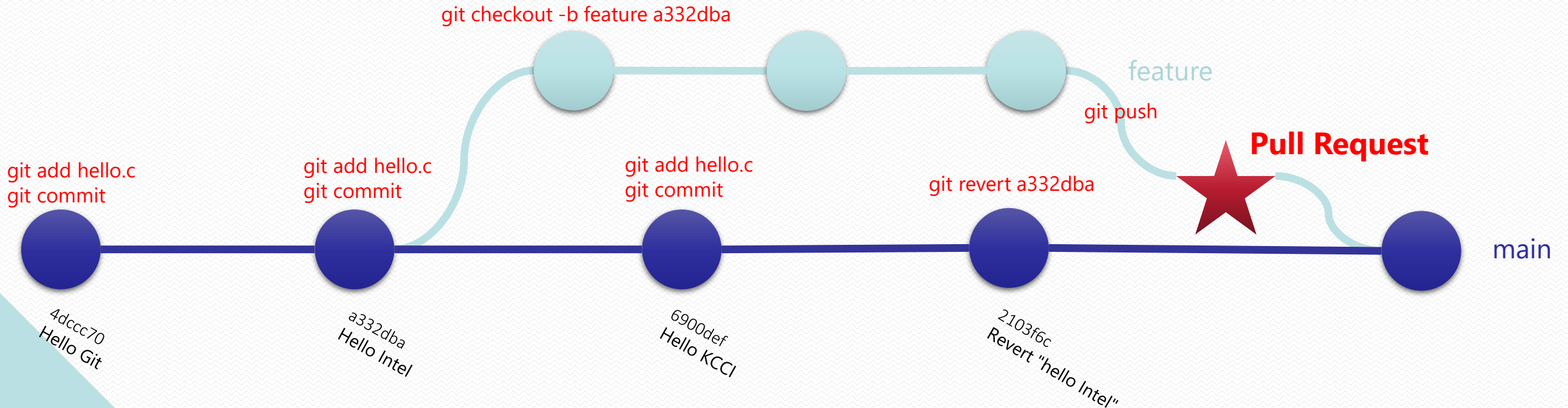
- * Name : Your English name
- * GitHub account : Your GitHub account
- * My practice repository URL : <https://github.com/abcde/my-project.git>

Practice : Creating Issue

- Goal
 - Based on your created issue, teacher will ...
 - Give Intel Curriculum Repository permission to your GitHub account
 - Visit your practice repository and review your practice.
- Please check teacher's comment in your created issue, then follow up teacher's feedback.

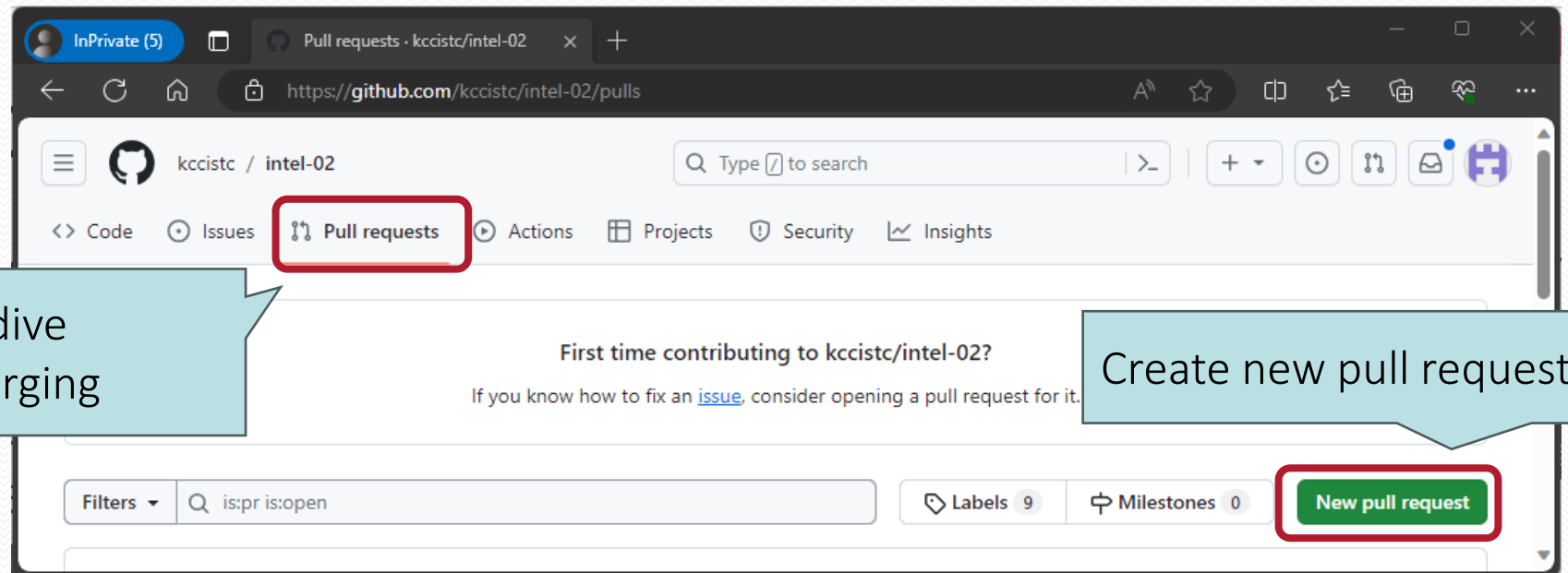
Creating Pull Request

- GitHub is a web-based hosting service for Git repositories.
- In simple terms, you can use Git without GitHub, but you CANNOT use GitHub without Git.
- GitHub offers Pull Request as web interface to discuss changes for merging.

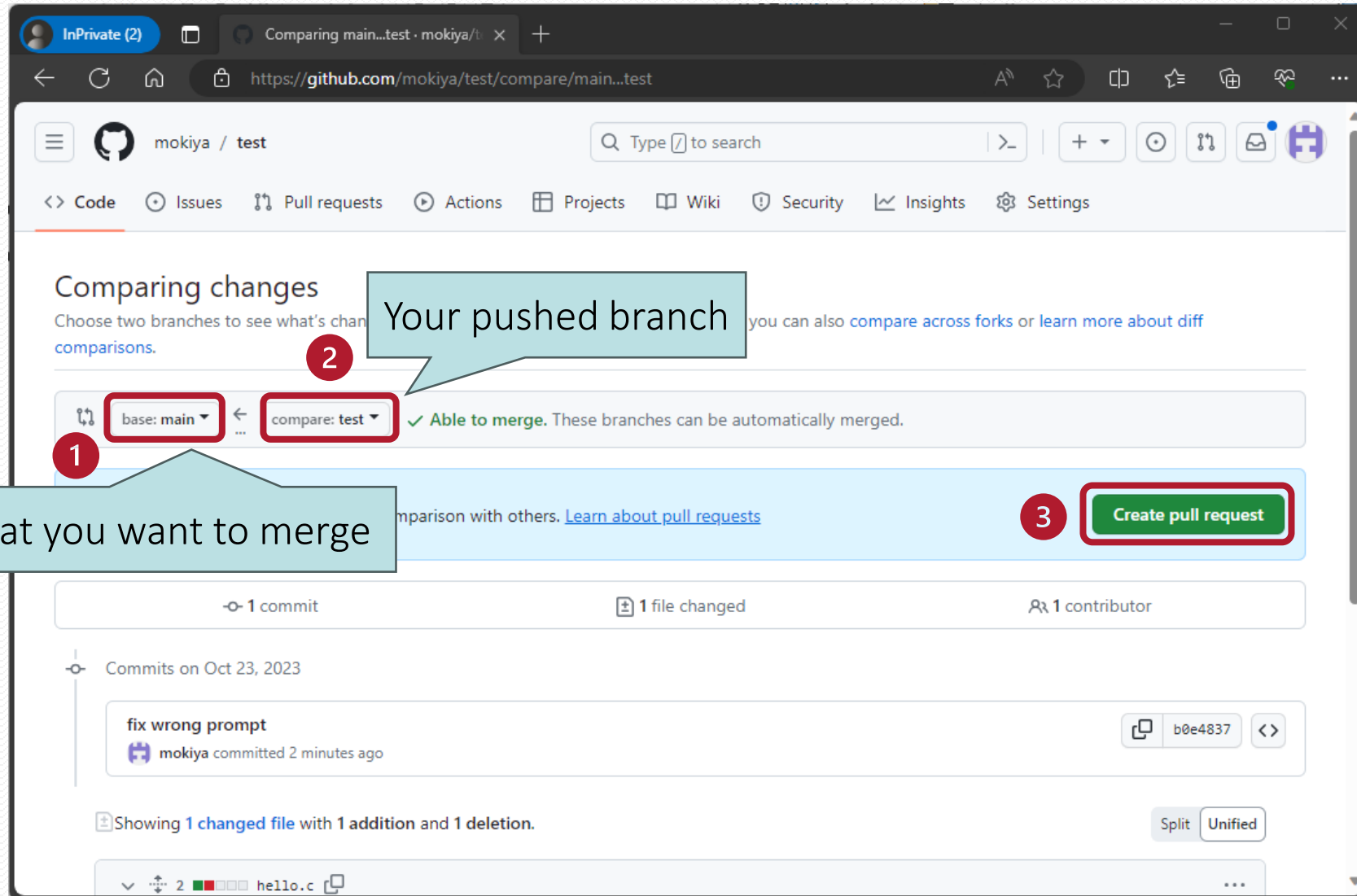


Creating Pull Request

- Pull Request tells others about changes you've pushed to a branch in a repository on GitHub.
- Once a pull request is opened, you can discuss and review the potential changes with collaborators and add follow-up commits before your changes are merged into the base branch.



Creating Pull Request



Creating Pull Request

The screenshot shows the GitHub interface for creating a pull request. The browser address bar shows the URL `https://github.com/mokiya/test/compare/main...test`. The page title is "Open a pull request" with a subtitle "Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).".

At the top, there are tabs for "Code", "Issues", "Pull requests", "Actions", "Projects", "Wiki", "Security", "Insights", and "Settings". Below these, the "Pull requests" tab is selected.

The main content area shows a comparison between "base: main" and "compare: test". A green status bar indicates "Able to merge. These branches can be automatically merged.".

The pull request title is "fix wrong prompt". Below the title, there are two tabs: "Write" and "Preview". The "Write" tab is active, showing a text area with the content "I'd appreciate it, if you can review and give me any feedback.".

On the right side, there are several sections for configuring the pull request:

- Reviewers:** A section with a gear icon and the text "No reviews". A callout bubble labeled "Choose reviewer" points to this section.
- Assignees:** A section with a gear icon and the text "No one—[assign yourself](#)".
- Labels:** A section with a gear icon and the text "None yet".
- Projects:** A section with the text "None yet".
- Milestone:** A section with a gear icon and the text "No milestone".

At the bottom right, there is a green button labeled "Create pull request" with a dropdown arrow. A callout bubble labeled "Choose patch owner" points to this button.

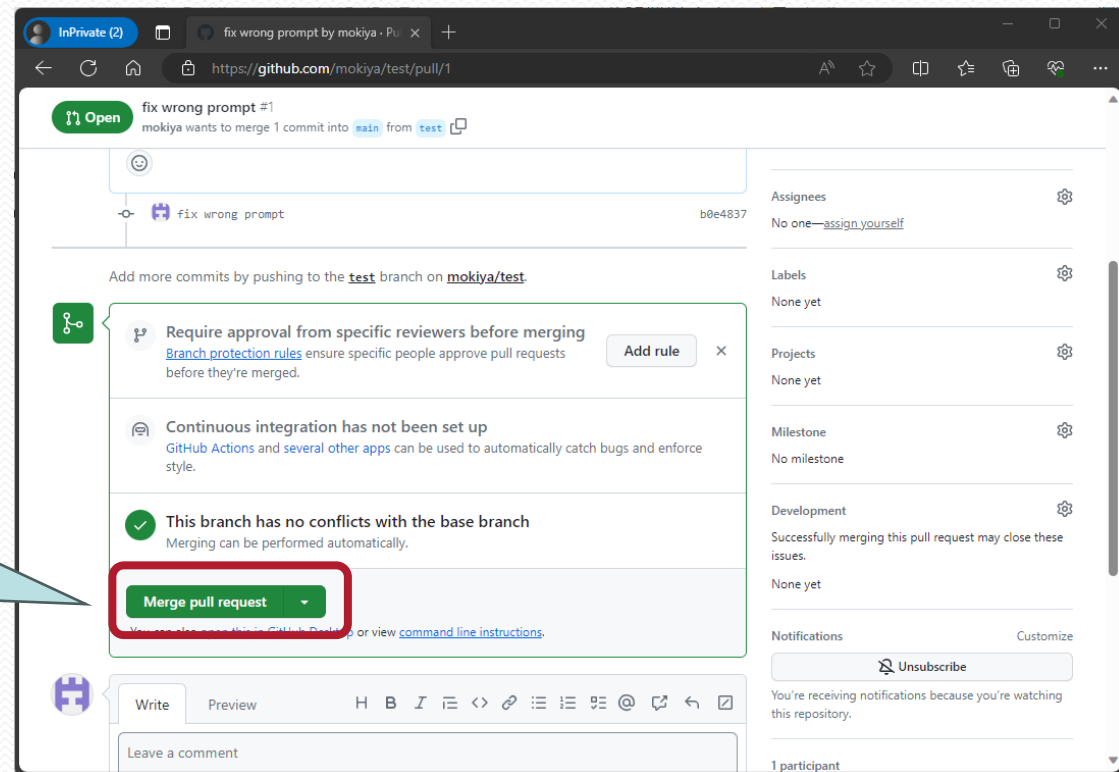
Four red circles with numbers 1 through 4 are placed on the page to highlight specific elements:

- 1: Points to the text area where the pull request description is written.
- 2: Points to the pull request title.
- 3: Points to the "Assignees" section.
- 4: Points to the "Create pull request" button.

Creating Pull Request

- Once a pull request is opened, issue discussion will start.
- You might need to rework/re-upload changes based on reviewer feedback.
- You might need to rework/re-upload changes, if there are conflicts.

If reviewer approved to merge patches, patch owner can merge changes.



Practice : Creating Pull Request

- Goal
 - Clone Intel Curriculum GitHub.
 - Create commits with following changes
 - Create directories named as your English name under `class01/homework/` and `class01/homework/`
 - Push commits to a new branch following the naming convention below
 - Naming convention : `class01-hw2-[github account]`
 - Example) if GitHub account is abcdef, branch name should be `class01-hw2-abcdef`.
 - Create Pull Request
 - Base branch should be “`main`”
 - Reviewer should be “`mokiya`”
 - Once reviewer approve your changes, merge your changes in created Pull Request

Ground Rules for Intel Curriculum

Create an Issue with mandatory fields below using GitHub Issue in Intel Curriculum GitHub Repository

- Title / Comment
- Milestone
- Assignee
- Development

Create a Pull Request with mandatory fields in Intel Curriculum GitHub Repository

- Base branch : main
- Reviewer

Based on feedback or conflicts, might need to re-work & re-push changes.

Create an
issue



Clone/Commit



Push



Create a Pull
Request



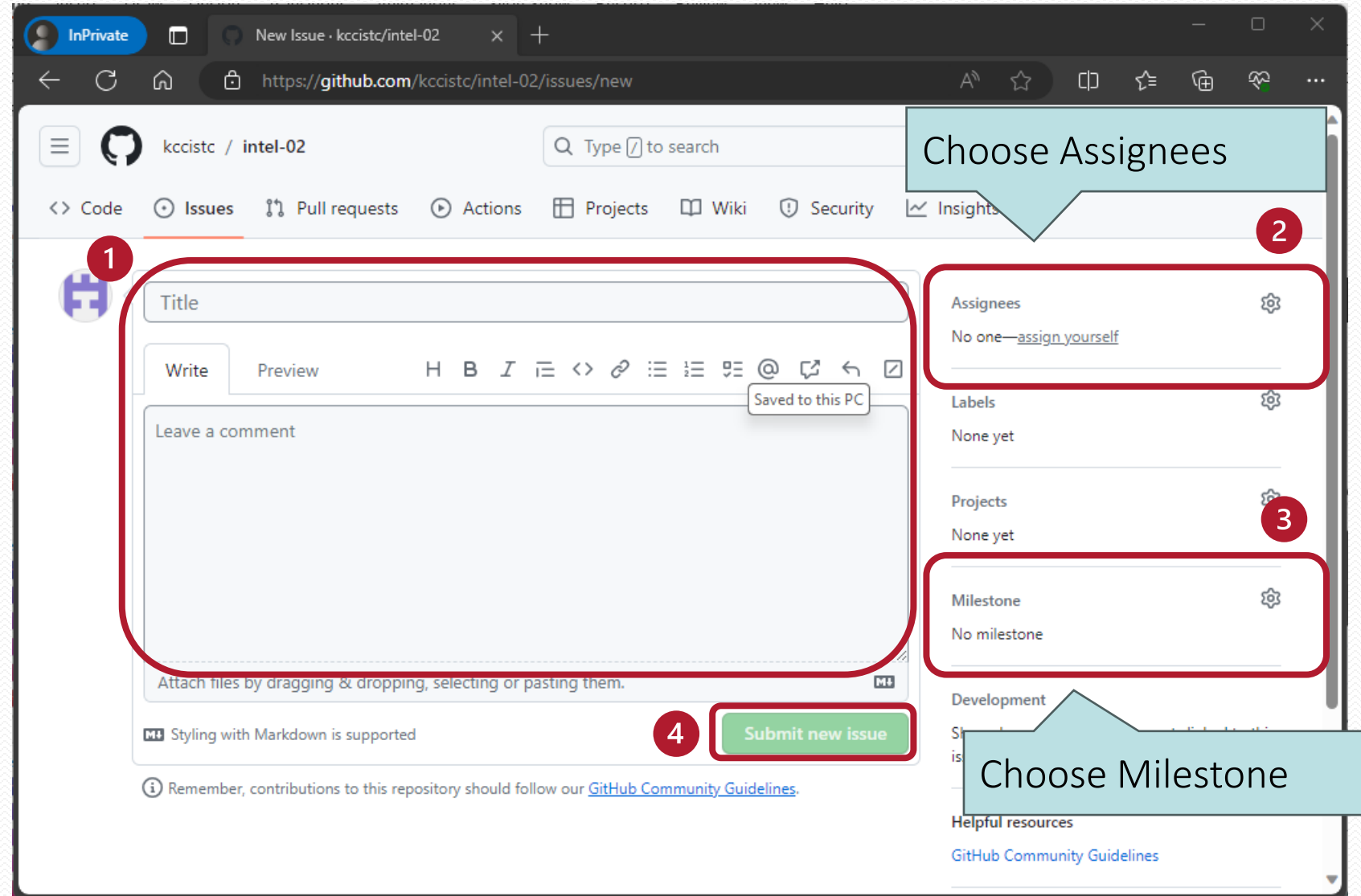
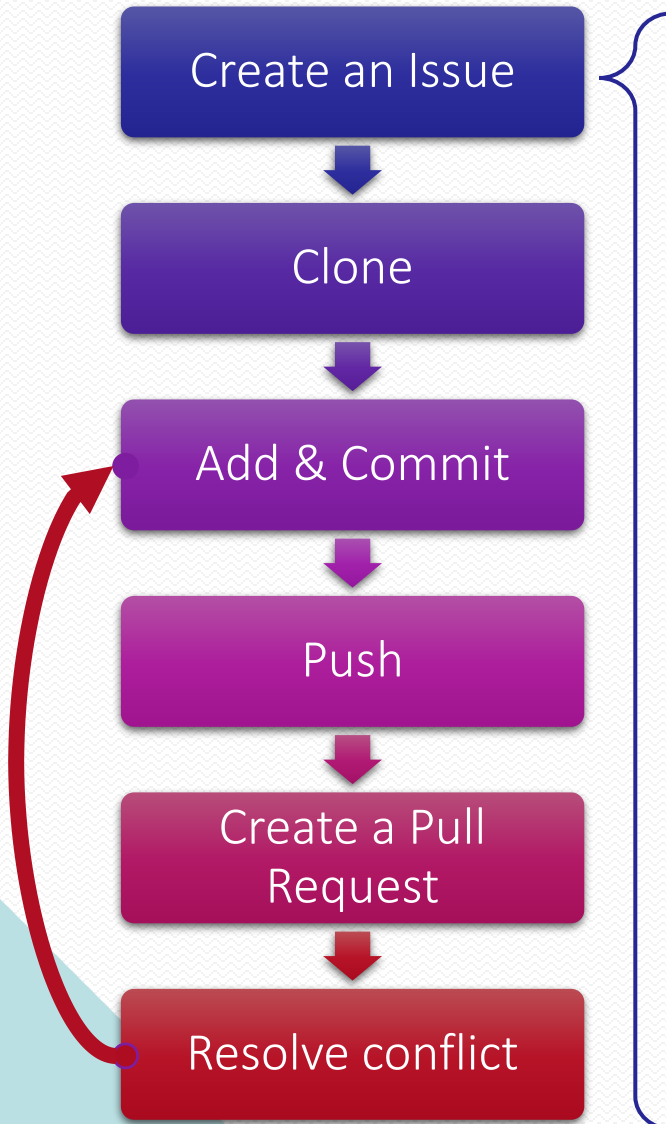
Merge

Push commits to a new branch following the naming convention below

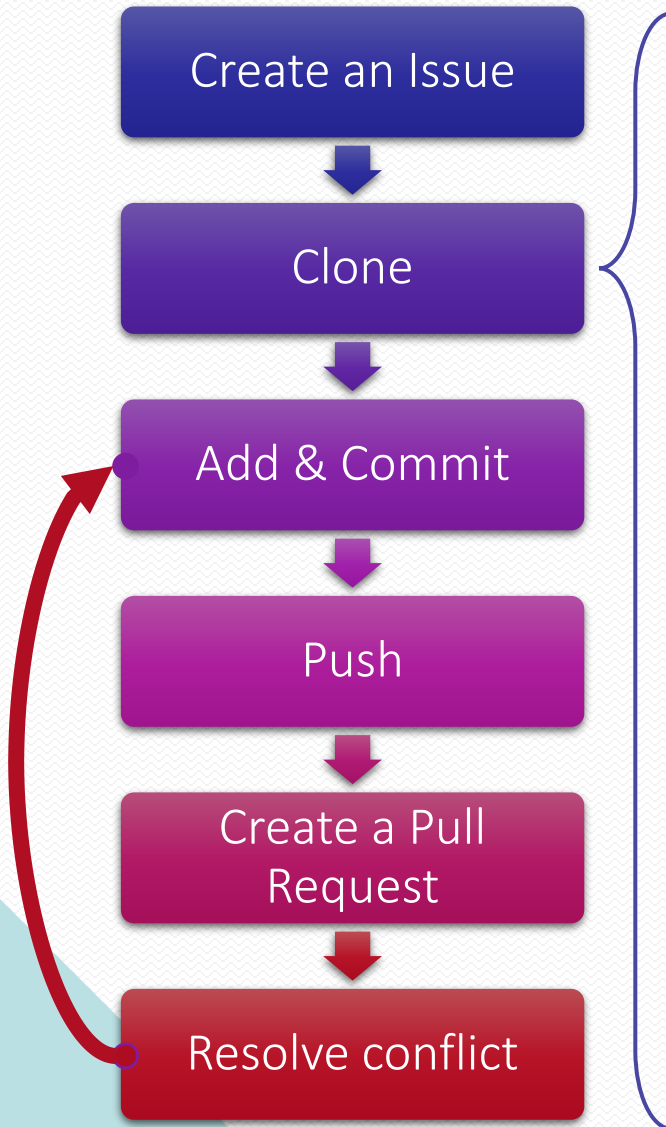
- Naming convention : class[xx]-hw[y]-[GitHub account]
- Example) GitHub account : abcdef | Course : class01 | Homework 2
Branch name should be class01-hw2-abcdef.

Once reviewer agrees to merge,
merge your patches in GitHub

Homework Workflow based Ground Rules



Homework Workflow based Ground Rules

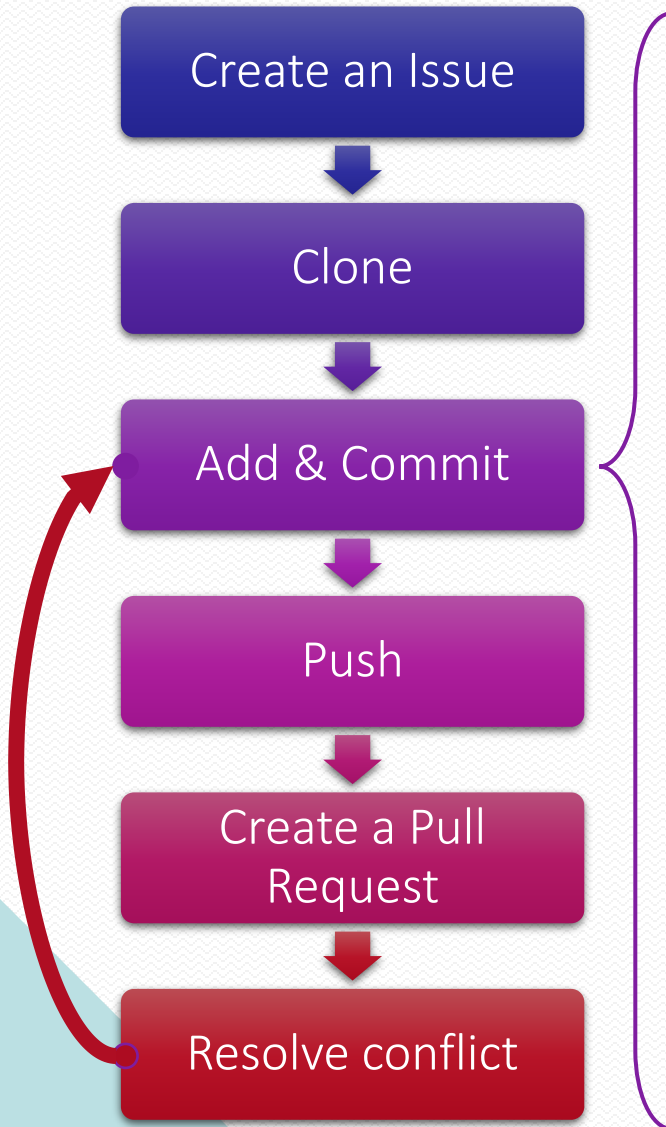


- A Git command line utility which is used to target an existing repository and create a clone, or copy of the target repository.

```
$ git clone <Repository GitHub URL> -b <Branch or Tag Name> <Directory>
```

- Clone Branch called <Branch or Tag Name> from the repository located at <Repository GitHub URL> into the folder called <Directory> on the local machine.

Homework Workflow based Ground Rules



- A Git command adds a change in the working directory to the staging area.

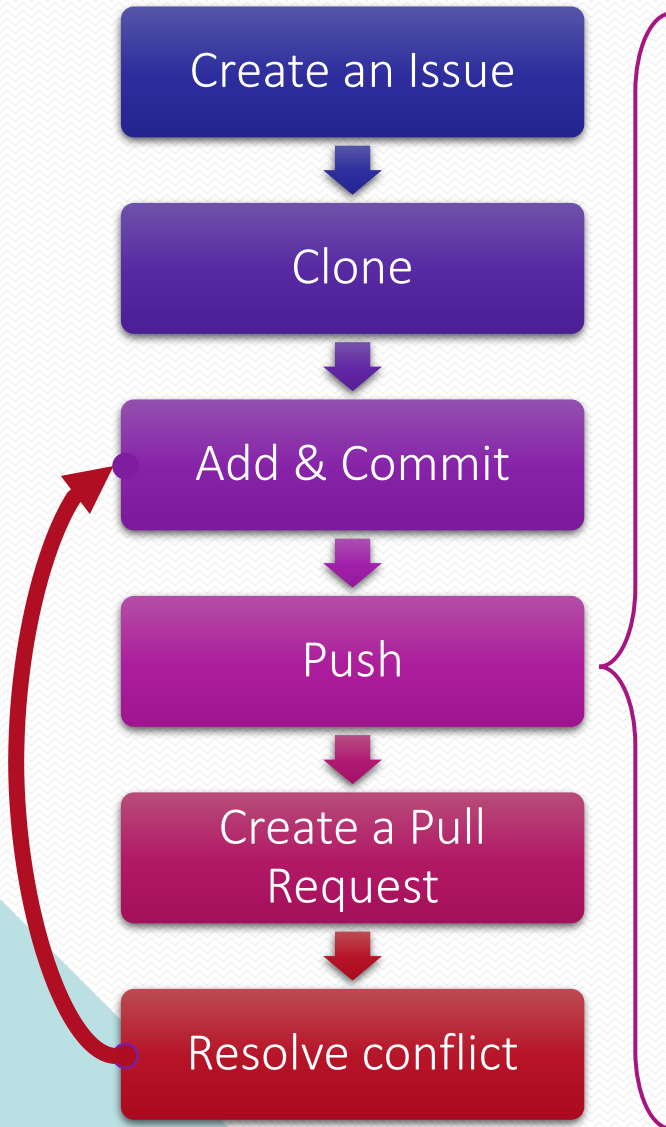
```
$ git add <File name 1> <File name 2> <Directory 1>
```

- Allow <File name 1>, <File name 2>, <Directory> to stage files to be committed.
- A Git command creates a commit, which is like a snapshot of your repository. These commits are snapshots of your entire repository at specific times.

```
$ git commit -m <Comment>
```

- Without -m option, default text editor will be opened to create the commit message.

Homework Workflow based Ground Rules

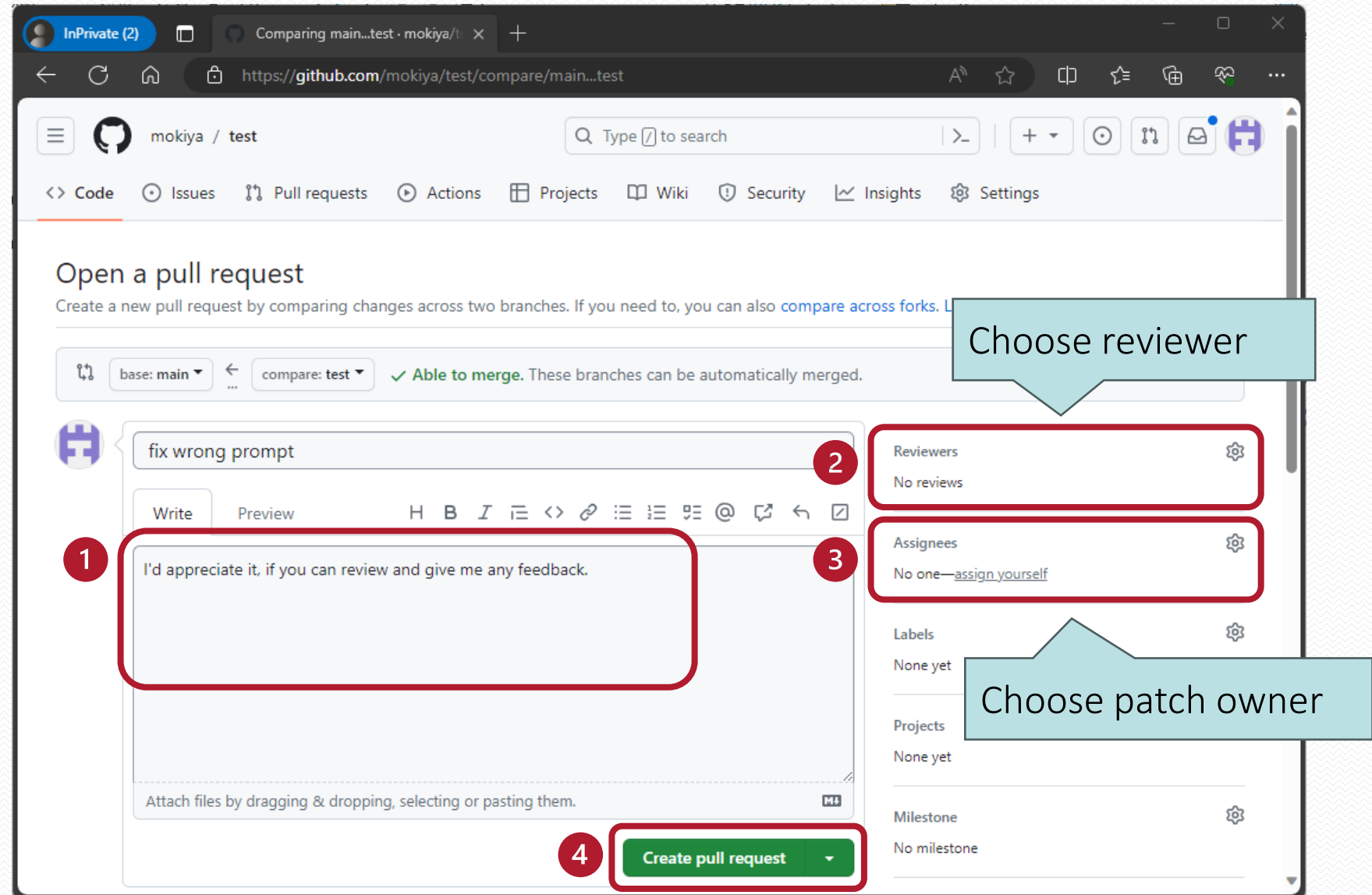
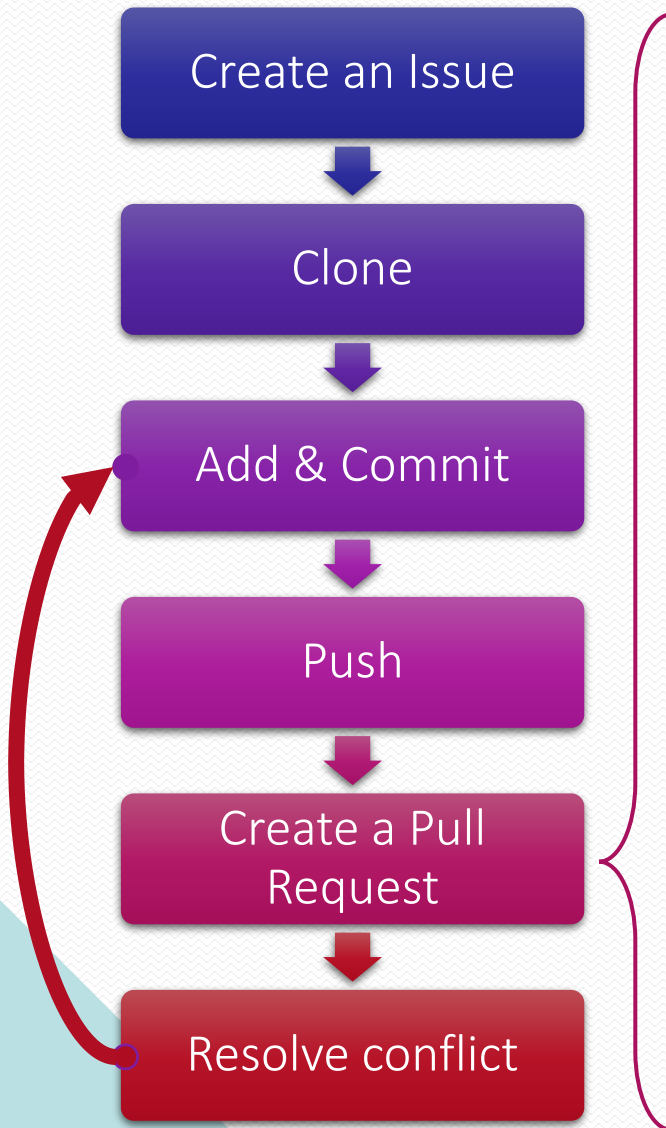


- A Git command is used to upload local repository content to a remote repository.

```
$ git push <Remote> HEAD:<Branch Name>
```

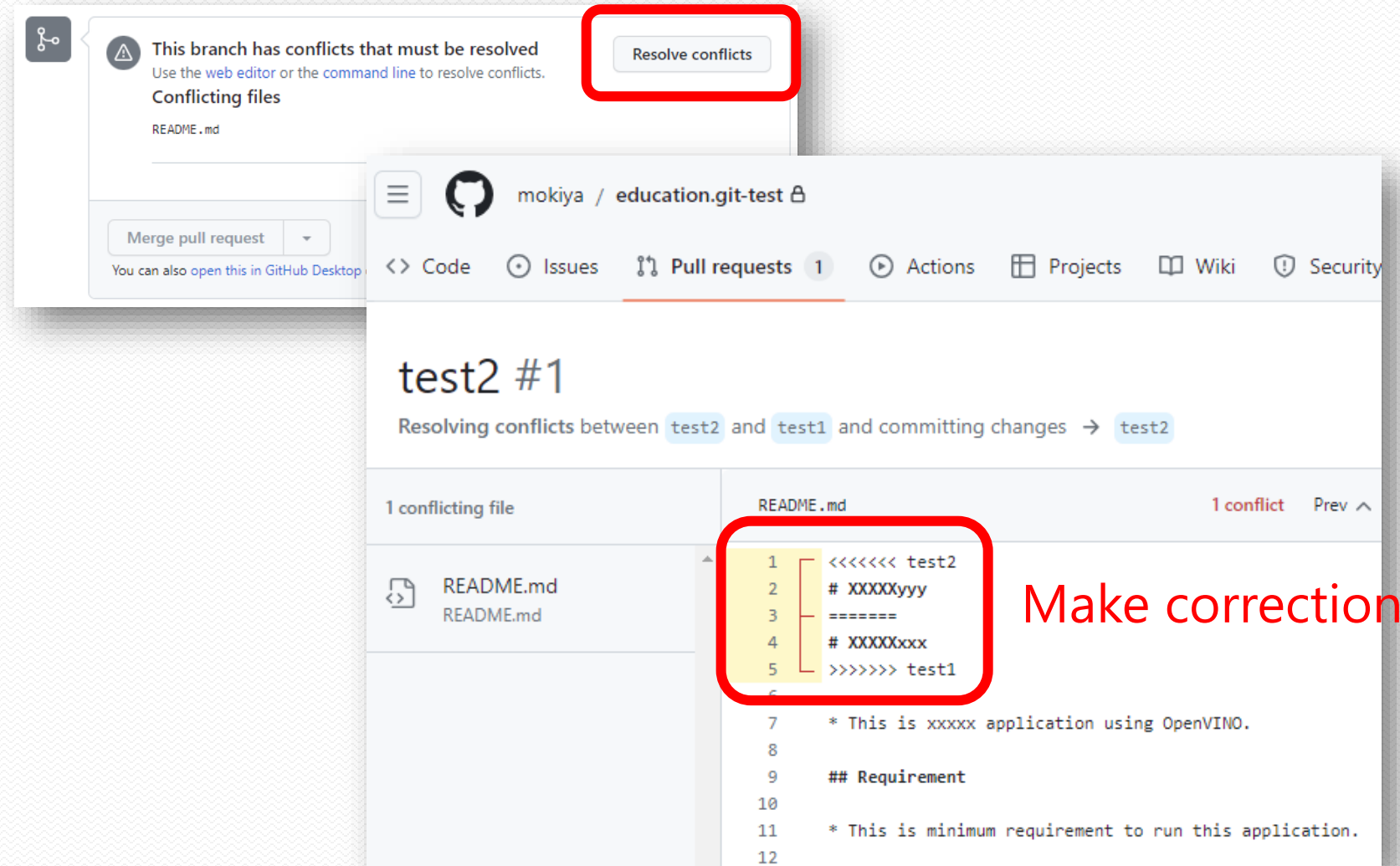
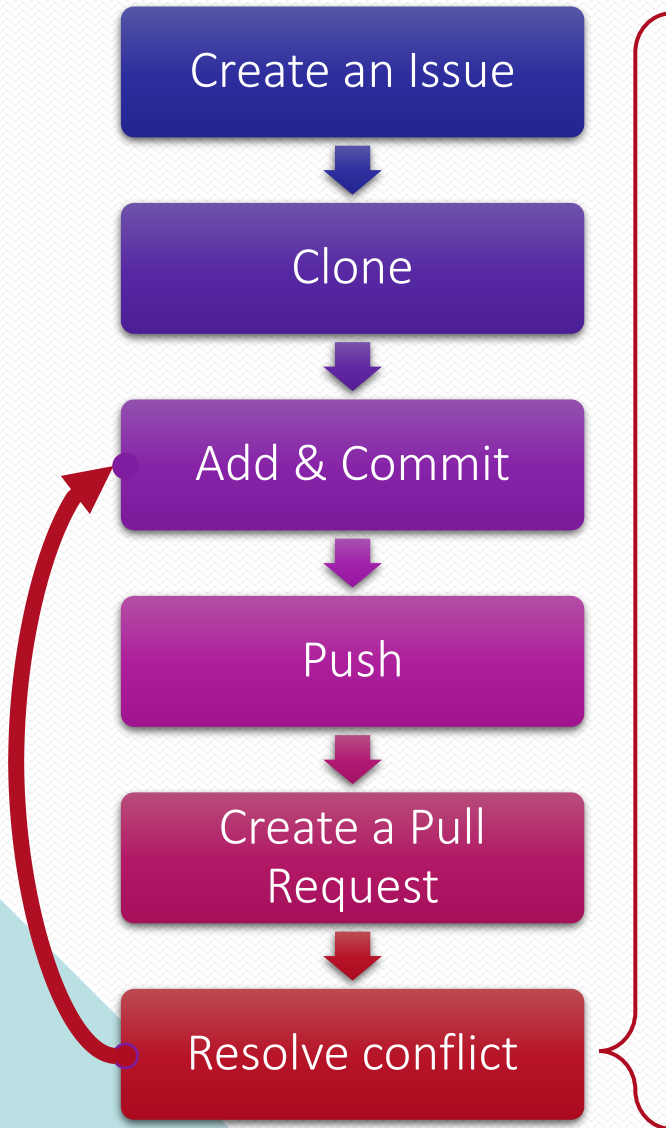
- Push your changes called **HEAD:** to specific branch (**<Branch Name>**) in the remote repository (**<remote repository>**)
- You can use commit id or local branch name instead of **HEAD:**
- Naming convention : **class[xx]-hw[y]-[GitHub account]**
Ex) GitHub account : **abcdef**
Course : **class01**
Homework 2 : **hw2**
Branch name should be **class01-hw2-abcdef**

Homework Workflow based Ground Rules

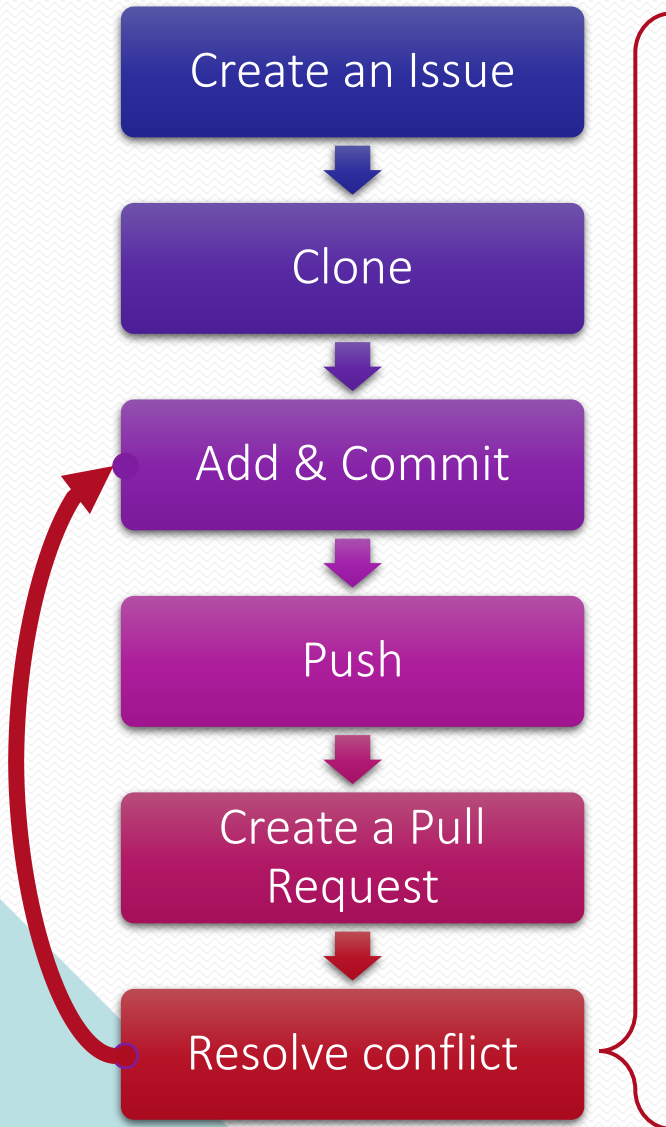


Homework Workflow based Ground Rules

- Resolution #01 : Resolve conflicts in the GitHub



Homework Workflow based Ground Rules



- Resolution #02 : Resolve conflicts in local machine
 - Step 1: Clone the repository or update your local repository with the latest changes.

```
$ git pull origin test1
```

- Step 2: Switch to the head branch of the pull request.

```
$ git checkout test2
```

- Step 3: Merge the base branch into the head branch.

```
$ git merge test1
```

- Step 4: Fix the conflicts and commit the result.
- Step 5: Push the changes.

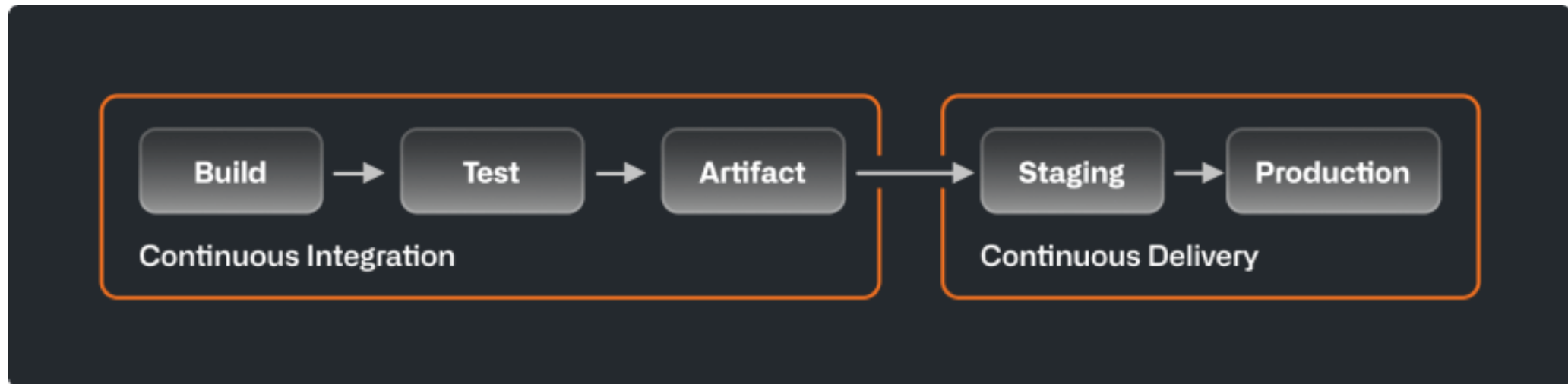
```
$ git push -u origin test2
```

Practice : Homework Workflow

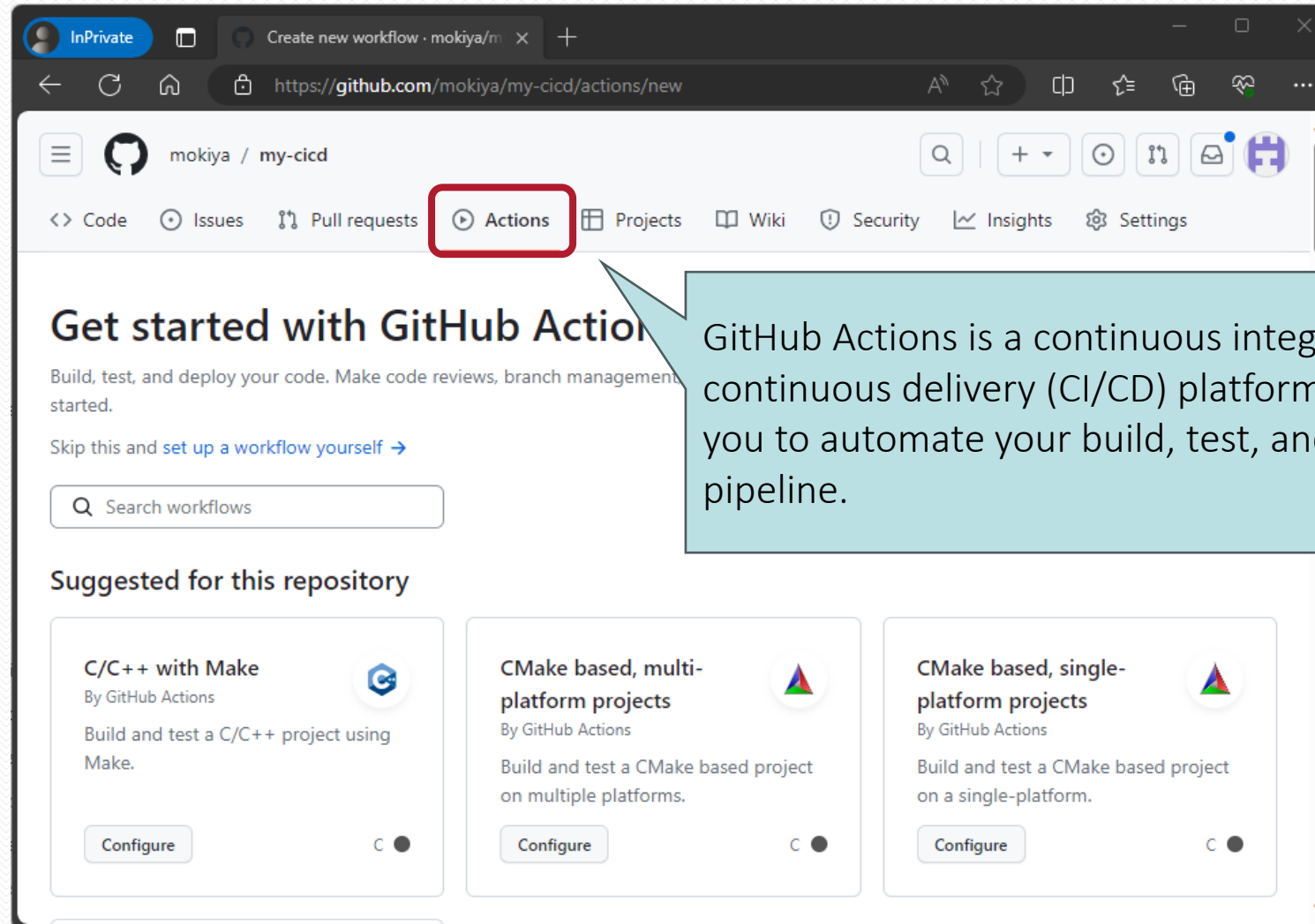
- Goal
 - Enter your name into the table in the two files below.
 - ./class01/README.md
 - ./class02/README.md
 - Be sure to work according to ground rules and create a PR.

CI / CD (Continuous Integration & Delivery)

- CI (Continuous Integration)
 - Automatically builds, tests, and integrates code changes within a shared repository
- CD (Continuous Delivery)
 - Automatically delivers code changes to production-ready environments for approval, then deploys code changes to customers directly.



GitHub Actions



GitHub Actions is a continuous integration and continuous delivery (CI/CD) platform that allows you to automate your build, test, and deployment pipeline.

Self-hosted Runner for GitHub Actions

- A self-hosted runner is a system that you deploy and manage to execute jobs from GitHub Actions on GitHub.com

The image illustrates the process of adding a self-hosted runner to GitHub Actions through three sequential screenshots, with red boxes and numbers highlighting key steps:

- Step 1:** The first screenshot shows the GitHub Actions 'Runners' page for the repository 'mokiya / my-cicd'. The 'Settings' tab is selected, and the 'Runners' section is visible. A red box highlights the 'New self-hosted runner' button.
- Step 2:** The second screenshot shows the 'Add new self-hosted runner' page. The 'Runner image' section is visible, and the 'Linux' option is selected. A red box highlights the 'Linux' option.
- Step 3:** The third screenshot shows the 'Runners' page after the runner has been added. A new runner named 'gram' is listed with the status 'Idle'. A red box highlights the 'gram' runner.

Workflow in GitHub Actions

The image shows two overlapping browser windows illustrating GitHub Actions workflow configuration. The left window displays the repository page for 'mokiya / my-cicd' with the 'Actions' tab selected (marked with a red circle and '1'). Below the 'Get started with GitHub Actions' section, the 'Suggested for this repository' area is highlighted with a red box and a red arrow (marked with a red circle and '2'). The right window shows the workflow editor for 'c-cpp.yml' with the following YAML content:

```
2
3 on:
4   push:
5     branches: [ "main" ]
6   pull_request:
7     branches: [ "main" ]
8
9 jobs:
10  build:
11
12    runs-on: self-hosted
13
14    steps:
15      - uses: actions/checkout@v3
16      - name: make
17        run: gcc ./hello.c -o ./hello
18      - name: test
19        run: ./hello
20      - name: deploy
21        run: echo "deploy"
22
```

The 'runs-on: self-hosted' line and the 'steps' section are highlighted with a red box. The right sidebar of the editor shows the 'Marketplace' tab with a search bar and featured actions: 'Upload a Build Artifact' (2.5k stars), 'Setup Java JDK' (1.3k stars), and 'Close Stale Issues' (1.1k stars).

Demo : CI / CD

The image displays two screenshots of the GitHub Actions interface, illustrating a CI/CD workflow.

Left Screenshot (Workflow List):

- URL: <https://github.com/mokiya/my-cicd/actions>
- Page Title: mokiya / my-cicd
- Navigation: Code, Issues, Pull requests, **Actions**, Projects, Wiki
- Message: Workflow run deleted successfully.
- Section: Actions (New workflow)
- Filter: All workflows
- Workflow: C/C++ CI
- Management: Caches, Runners (Beta)
- Section: All workflows (Showing runs from all workflows)
- Workflow run: 1 workflow run
- Workflow: **Update c-cpp.yml** (C/C++ CI #3: Commit 663822f pushed by mokiya)

Right Screenshot (Workflow Run Details):

- URL: <https://github.com/mokiya/my-cicd/actions/runs/6612651512>
- Page Title: mokiya / my-cicd
- Navigation: Code, Issues, Pull requests, **Actions**, Projects, Wiki, Security, Insights, Settings
- Section: C/C++ CI
- Workflow: **Update c-cpp.yml #3** (Re-run all jobs)
- Summary: Triggered via push 2 minutes ago, Status: **Success**, Total duration: 25s, Artifacts: -
- Jobs: build (Success)
- Run details: Usage, Workflow file
- Job: **build** (Success, 15s)

Demo : CI / CD

The image shows a GitHub Actions workflow being edited and its execution results. A red arrow points from the workflow definition to the execution logs.

Workflow Definition (Left):

```
2
3 on:
4   push:
5     branches: [ "main" ]
6   pull_request:
7     branches: [ "main" ]
8
9 jobs:
10  build:
11    runs-on: self-hosted
12
13    steps:
14      - uses: actions/checkout@v3
15      - name: make
16        run: gcc ./hello.c -o ./hello
17      - name: test
18        run: ./hello
19      - name: deploy
20        run: echo "deploy"
21
22
```

Execution Logs (Right):

Summary

Jobs

- ✓ build

Run details

- Usage
- Workflow file

build

succeeded 4 minutes ago in 15s

Search logs

- ✓ Set up job 2s
- ✓ Run actions/checkout@v3 0s
- ✓ make 0s
 - 1 Run gcc ./hello.c -o ./hello
 - 2 gcc ./hello.c -o ./hello
 - 3 shell: /usr/bin/bash -e {0}
- ✓ test 0s
 - 1 Run ./hello
 - 4 Hello Git
 - 5 Hello KCCI
- ✓ deploy 0s
 - 1 Run echo "deploy"
 - 4 deploy
- ✓ Post Run actions/checkout@v3 0s
- Complete job

