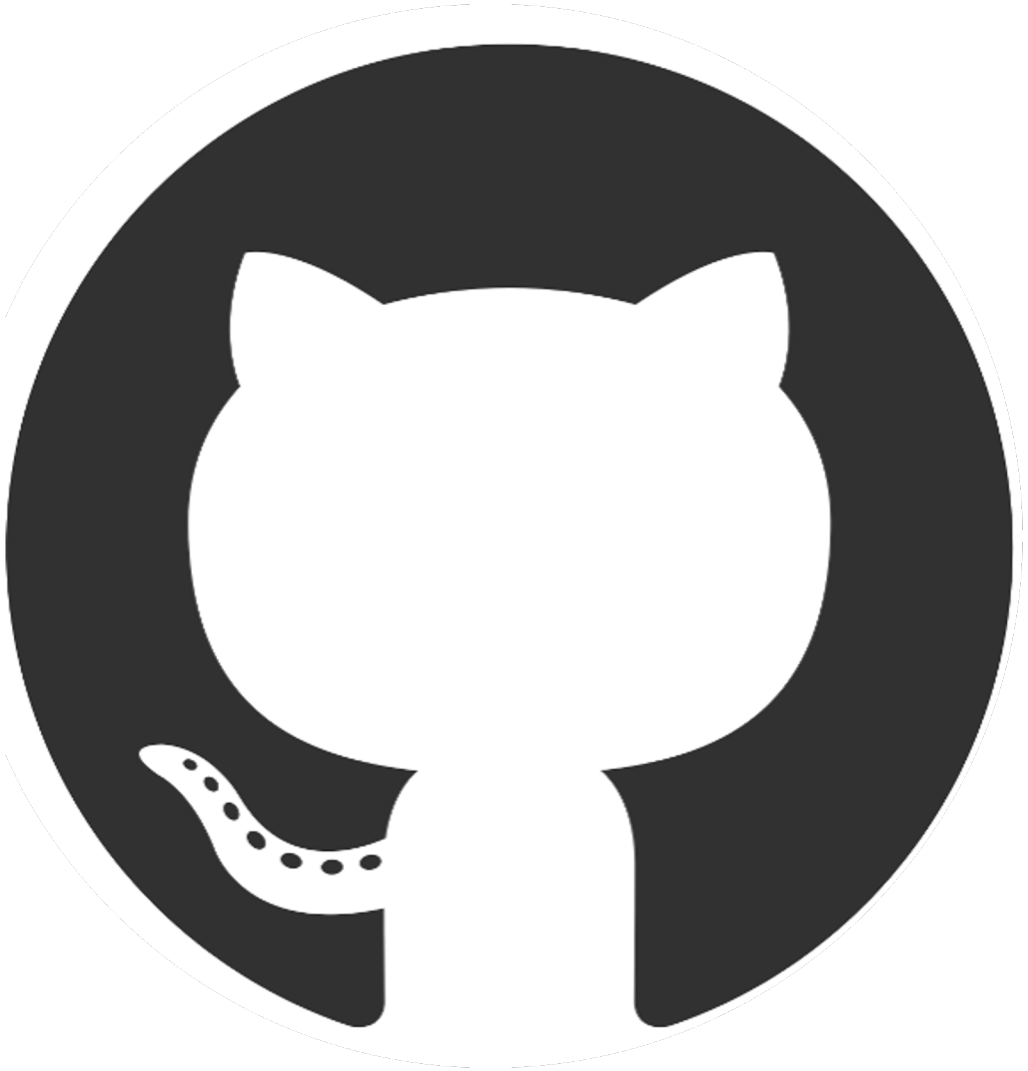
GitHub

COMP 3104 - DevOps



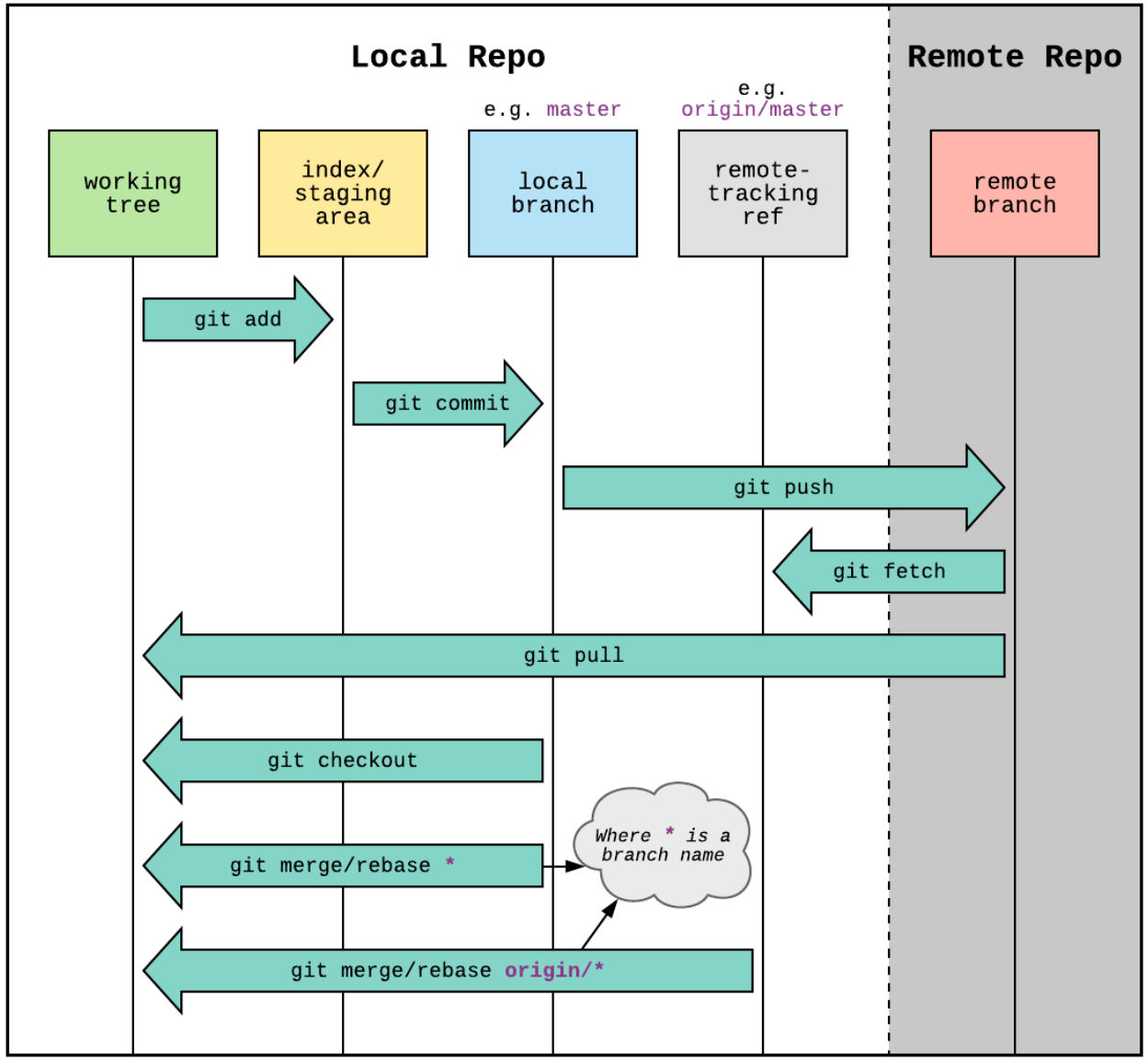
What is GitHub ?

- Git hub is an open source project that is considered to be a version control system where the user can store and share their repositories

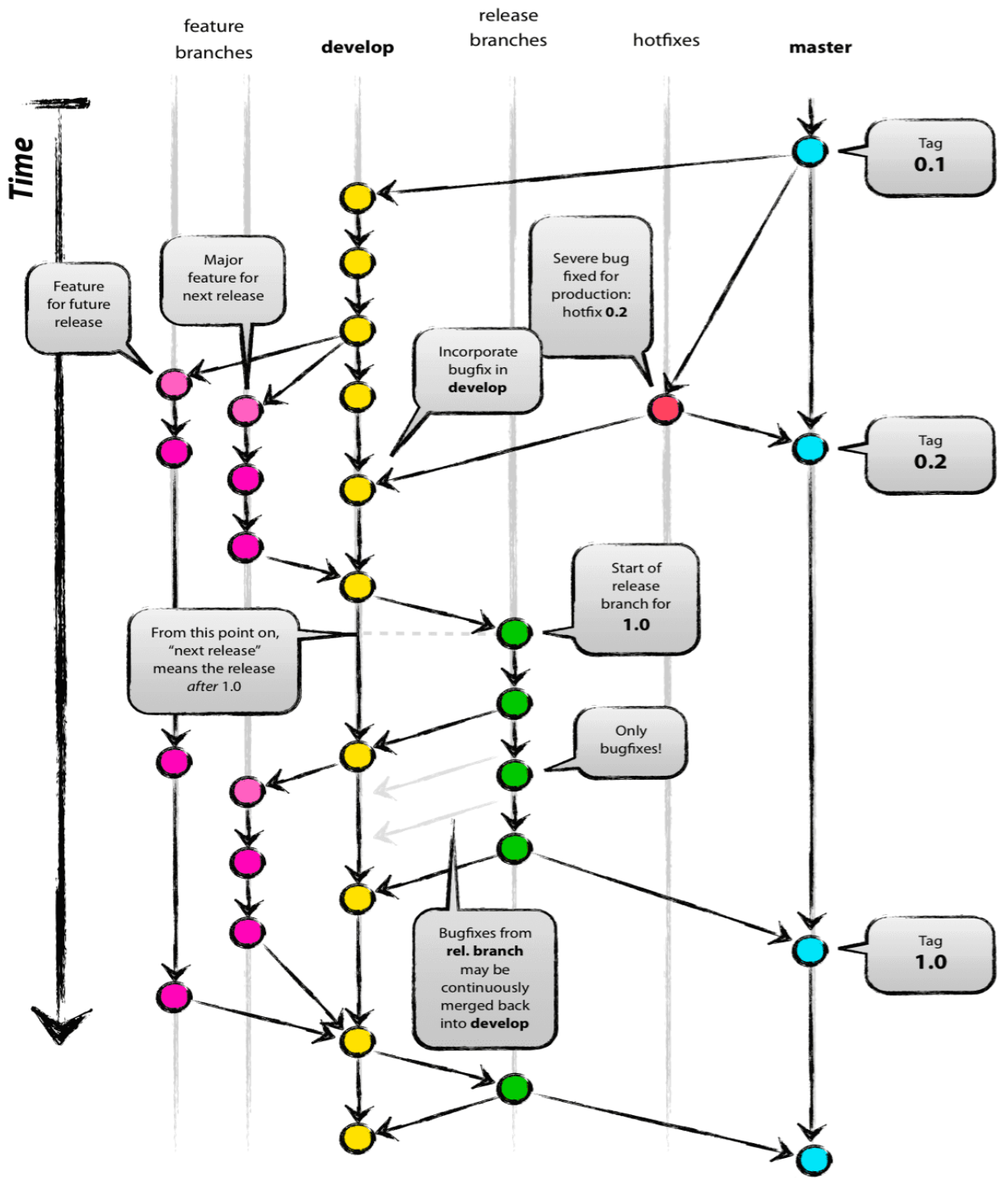
- Git hub can be considered as a networking site for the programmers which will keep them all connected

https://github.com/

GitHub Workflow I



GitHub Workflow II



Step 1. Create a Repository

- A **repository** is usually used to organize a single project.
- Repositories can contain folders and files, images, videos, spreadsheets, and data sets – anything your project needs.
- We recommend including a *README*, or a file with information about your project.
- Your hello-world repository can be a place where you store ideas, resources, or even share and discuss things with others.

To create a new repository

1) In the upper right corner, next to your avatar or identicon, click and then select **New repository**.
2) Name your repository hello-world.
3) Write a short description.
4) Select **Initialize this repository with a README**.

Web Reference Link

Owner **Repository name**

PUBLIC hubot / hello-world

Great repository names are short and memorable. Need inspiration? How about **petulant-shame**.

Description (optional)

Just another repository

Public
Anyone can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with a README
This will allow you to `git clone` the repository immediately. Skip this step if you have already run `git init` locally.

Add .gitignore: **None** | Add a license: **None**

Create repository

Step 2. Create a Branch

- **Branching** is the way to work on different versions of a repository at one time.
- By default your repository has one branch named **main/master** which is the definitive branch.
- When you create a branch off the master branch, you're making a copy, or snapshot, of main as it was at that point in time.
- If someone else made changes to the master branch while you were working on your branch, you could pull in those updates.
- This diagram shows:
  - ✓ The master branch
  - ✓ A new branch called feature (because we're doing 'feature work' on this branch)
  - ✓ The journey that feature takes before it's merged into main

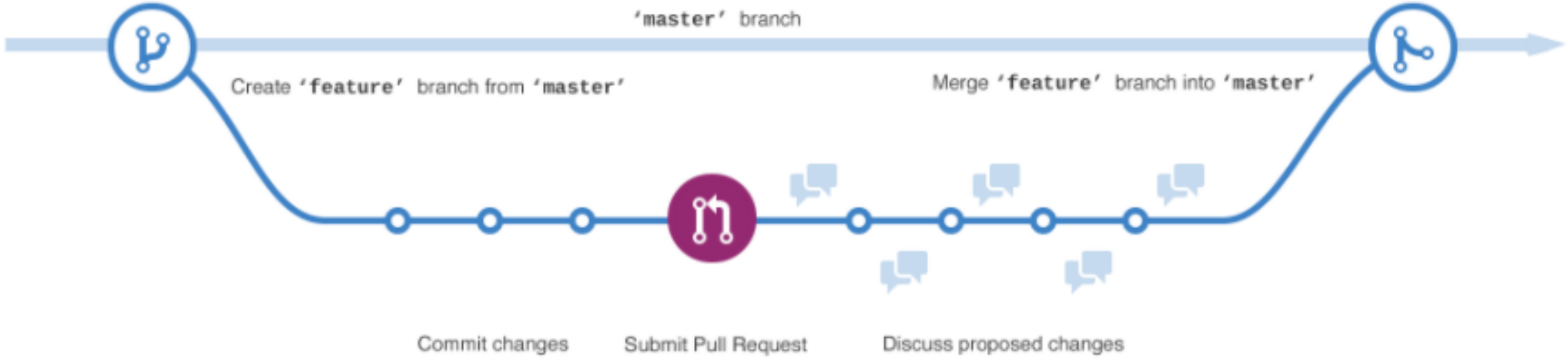
Git Branch

Have you ever saved different versions of a file? Something like:

- story.txt
- story-joe-edit.txt
- story-joe-edit-reviewed.txt

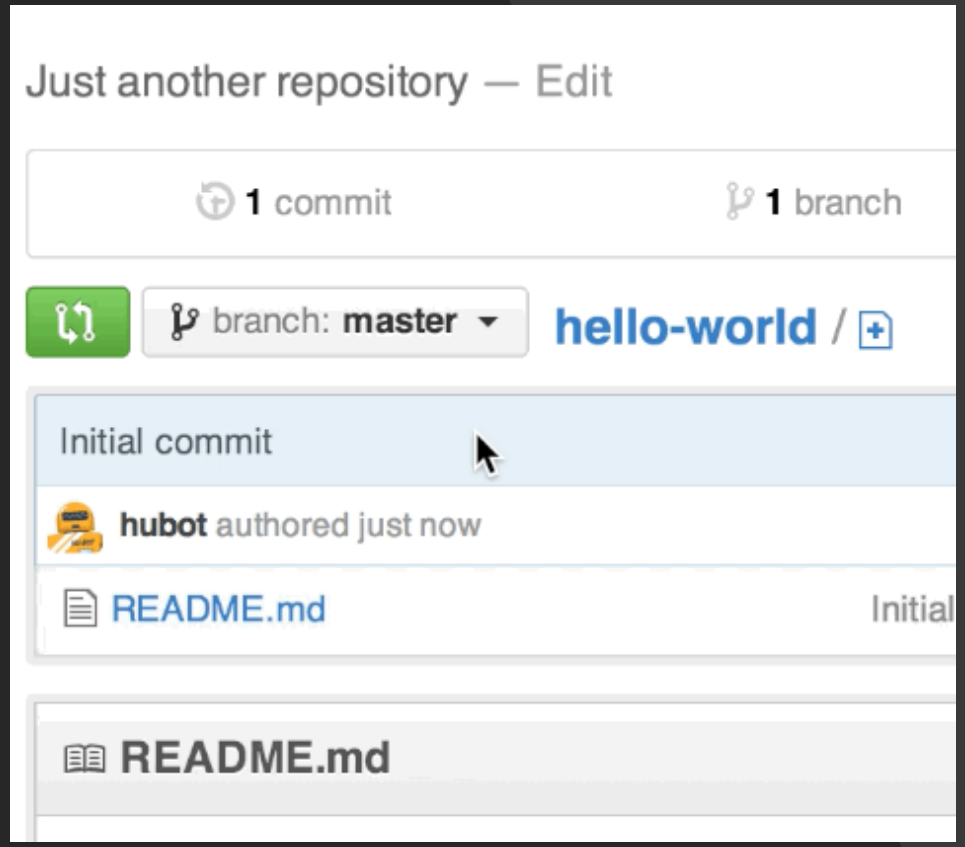
Branches accomplish similar goals in GitHub repositories.

Here at GitHub, our developers, writers, and designers use branches for keeping bug fixes and feature work separate from our main (production) branch. When a change is ready, they merge their branch into main.



To **create** a new branch

1) Go to your new repository hello-world.
2) Click the drop down at the top of the file list that says **branch: master**.
3) Type a branch name, readme-edits, into the new branch text box.
4) Select the blue **Create branch** box or hit “Enter” on your keyboard.

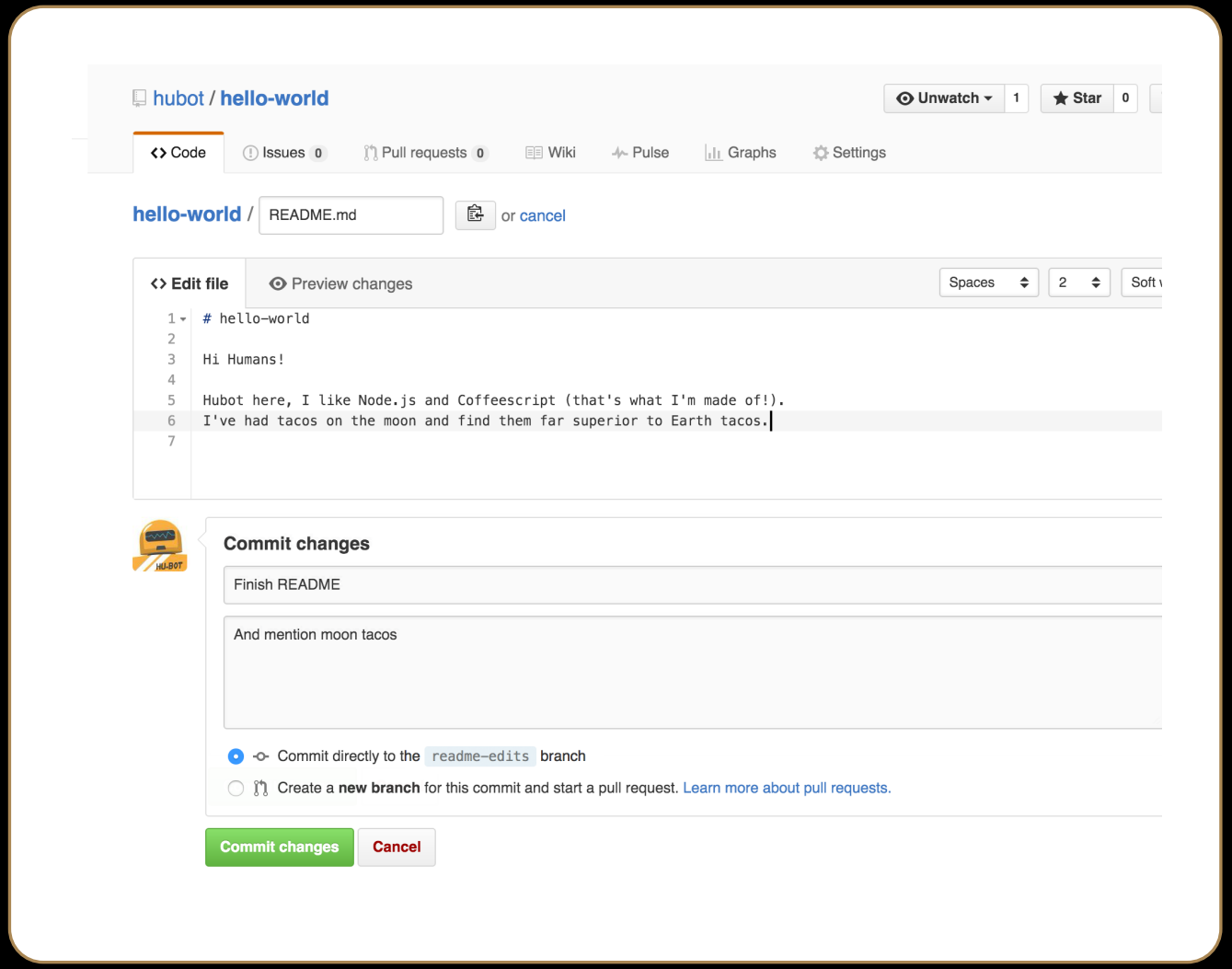


Step 3. Make and commit changes

- On GitHub, saved changes are called ***commits***.
- Each commit has an associated ***commit message***, which is a description explaining why a particular change was made.
- Commit messages capture the history of your changes, so other contributors can understand what you've done and why.
- **Make and commit changes**
  - ✓ Click the README.md file.
  - ✓ Click the pencil icon in the upper right corner of the file view to edit.
  - ✓ In the editor, write a bit about yourself.
  - ✓ Write a commit message that describes your changes.
  - ✓ Click **Commit changes** button.

ReadMe File

- These changes will be made to just the README file on your readme-edits branch, so now this branch contains content that's different from main.



Step 4. Open a Pull Request

- **Pull Requests** are the heart of collaboration on GitHub.
- When you open a *pull request*, you're proposing your changes and requesting that someone review and pull in your contribution and merge them into their branch.
- Pull requests show *diffs*, or differences, of the content from both branches.
- The changes, additions, and subtractions are shown in GREEN and RED.
- As soon as you make a commit, you can open a pull request and start a discussion, even before the code is finished.
- You can even open pull requests in your own repository and merge them yourself.

To better understand we have learned the GitHub flow on slide#3.

Step 5. Merge your Pull Request

- In this final step, it's time to bring your changes together – **merging** your readme-edits branch into the main branch.
- Click the green Merge pull request button to merge the changes into master.
- Click Confirm merge.
- Go ahead and delete the branch, since its changes have been incorporated, with the Delete branch button in the purple box.

Git Bash Commands

- **Git Bash** is a source control management system for Windows.
- It allows users to type Git commands that make source code management easier through versioning and commit history.
- **Bash** is a Linux-based command line (that has been ported over to Windows) while **Shell** is a native Windows command line.

Students Note: Exercise of Git installed is completed in Week-01

Git Commands list

- git config
- git init
- git clone
- git add
- git commit
- git diff
- git reset
- git status
- git rm
- git log
- git show
- git tag
- git branch
- git checkout
- git merge
- git remote
- git push
- git pull
- git stash

✓ Click Here for complete list of Git Bash Commands
✓ Git Cheatsheet

About SSH

- **Secure Shell (SSH)** is a cryptographic protocol and interface for executing network services, shell services and secure network communication with a remote computer.

- Secure Shell enables two remotely connected users to perform network communication and other services on top of an unsecured network.

- It was initially a Unix-based command but is now supported on Windows-based systems as well.

Refer Docx file for more details under week-02/lecture folder on BB

Click here to see SSH Crash Course

Thank You