Introduction to Programming/ Installation and running Python / Getting Started with GitHub

Getting started with GitHub

GitHub is a free program and website which we will use to manage code. You do not need to pay any money to GitHub.

What is happening

Git is a program called a *version-control system*, which is used for tracking changes in source code during software development.

GitHub is a website which hosts software source code and uses Git. GitHub is based around repositories, which you can think of a folder for a project.

We will use GitHub in this module for you to share code you have written with me. Specifically, we will use a system called GitHub Classroom.

This section explains how to get started with GitHub and run a task on GitHub Classroom. There are a lot of things to authorise and register, but this won't need doing every time you use GitHub.

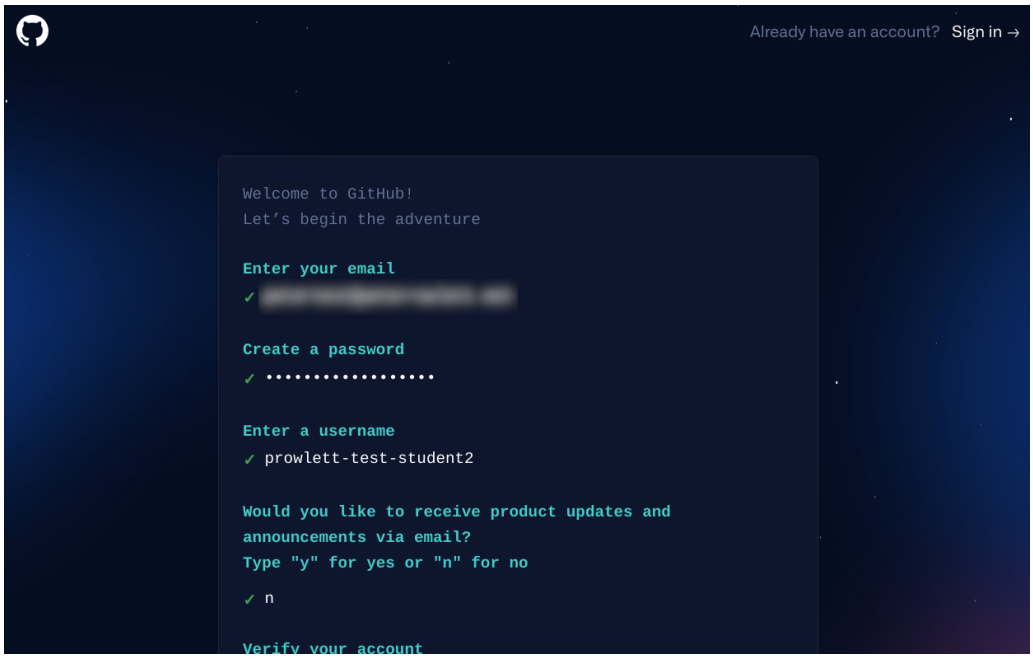
There are instructions here to use the GitHub Desktop app or GitHub via its website. If you can't install software on the computer you are using, you will have to use the website. There are organisational advantages to using the Desktop app because it will copy files to and from GitHub and keep them organised for you. Use whichever you feel more comfortable with.

Using the GitHub website

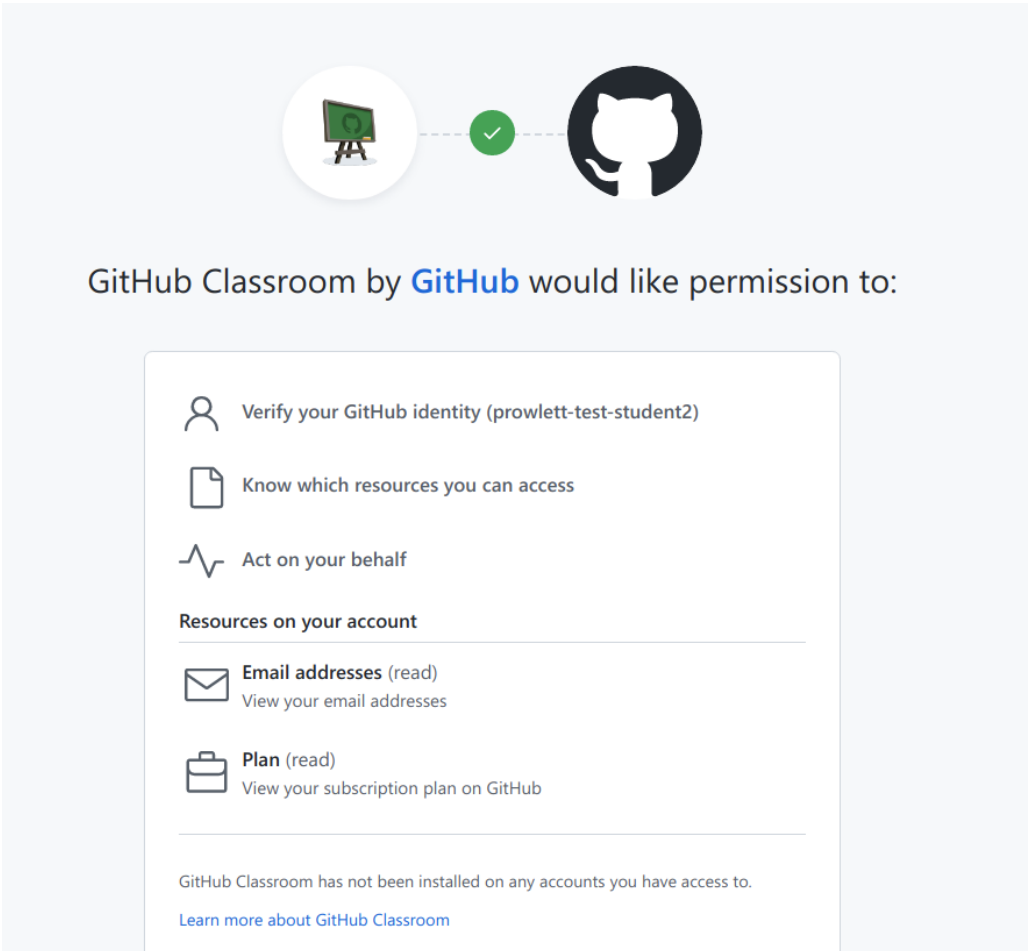
You will be downloading and uploading files via your web browser. Try to keep your files sensibly arranged and not get in a muddle about which version

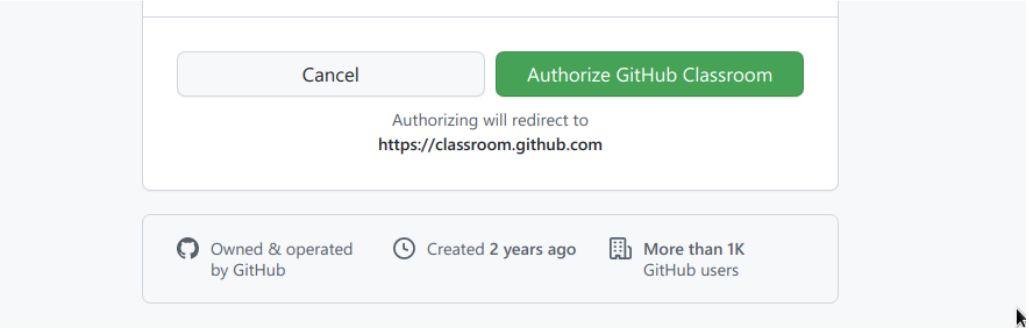
of which program you are editing or uploading.

1. Go to github.com and click 'Sign up'.
2. Create an account. Try to choose a professional-sounding username.

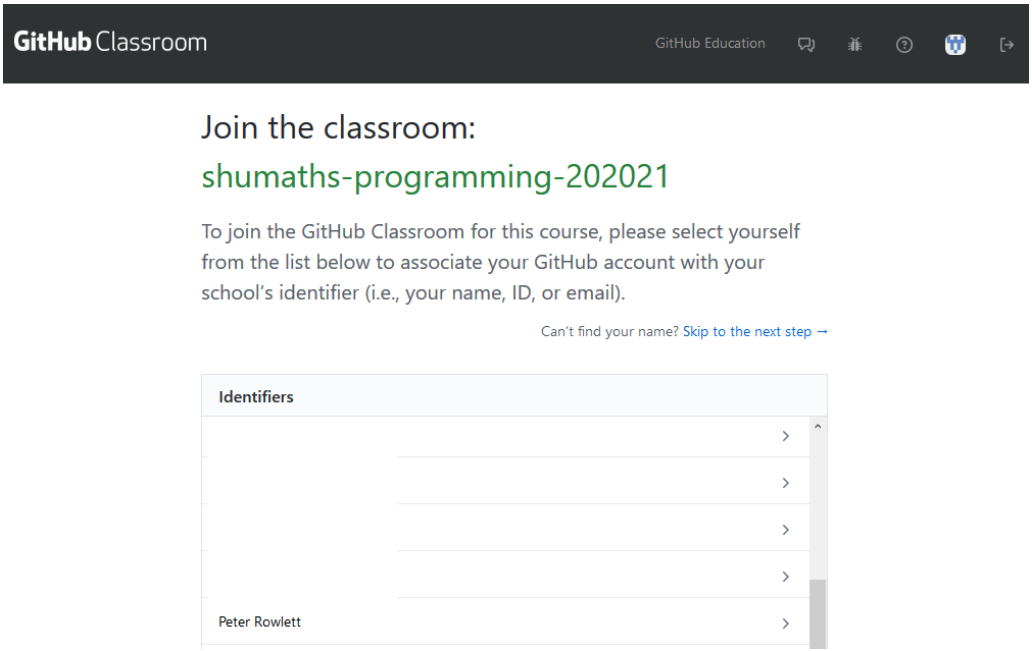


3. Go through the process to verify your email.
4. Go to classroom.github.com/a/Xcvhb1sS. Click to authorise GitHub Classroom to access your GitHub account.

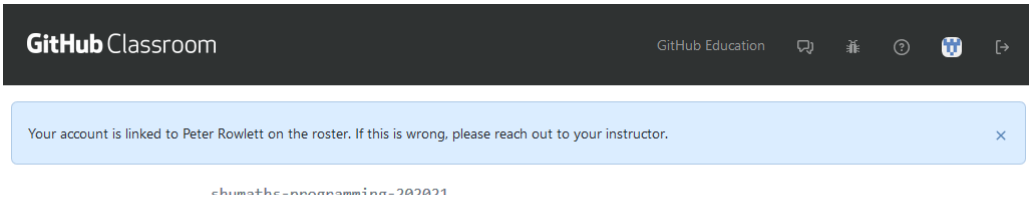

GitHub Classroom by GitHub would like permission to:
Verify your GitHub identity (prowlett-test-student2)
Know which resources you can access
Act on your behalf
Resources on your account
Email addresses (read)
View your email addresses
Plan (read)
View your subscription plan on GitHub
GitHub Classroom has not been installed on any accounts you have access to.
Learn more about GitHub Classroom

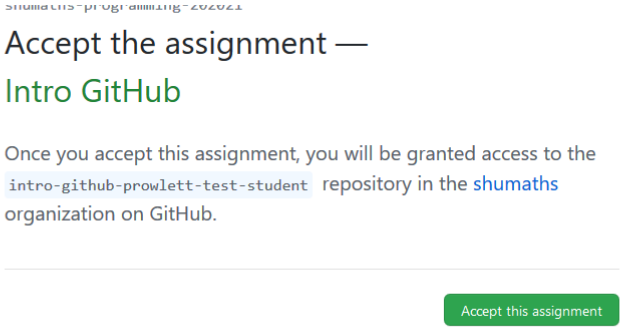


5. When you are shown a list of students, choose your name.

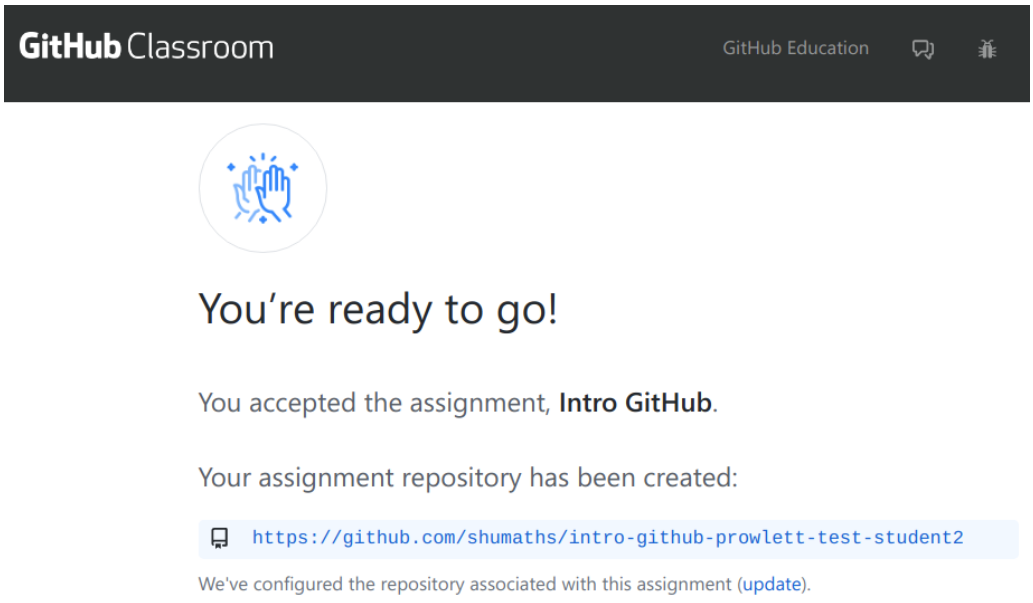


6. Click to accept the assignment.



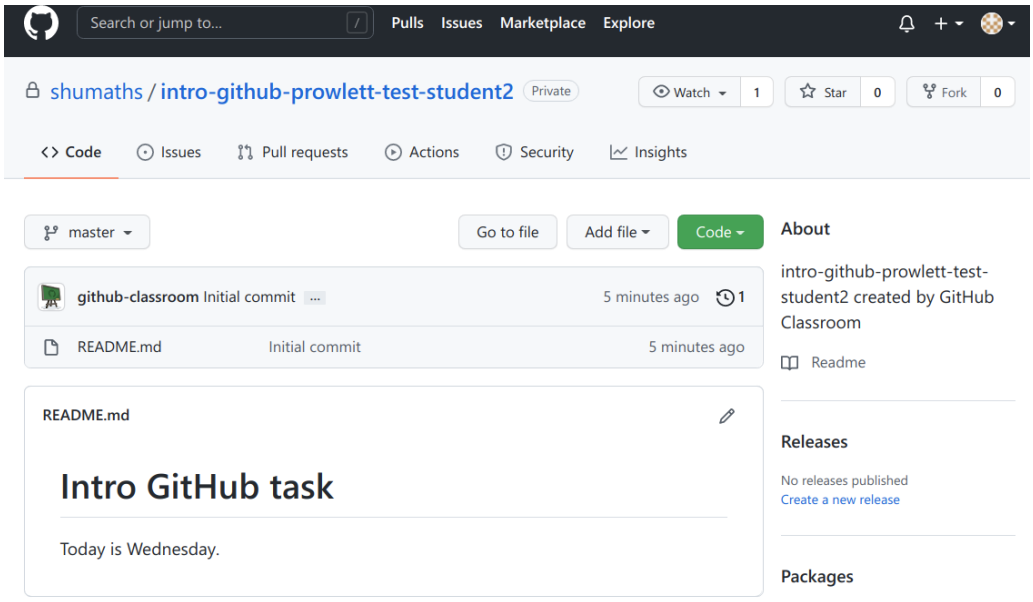


7. You might need to refresh the page after a minute or two, but you should eventually get to a confirmation page. Click the link to access your new repository.



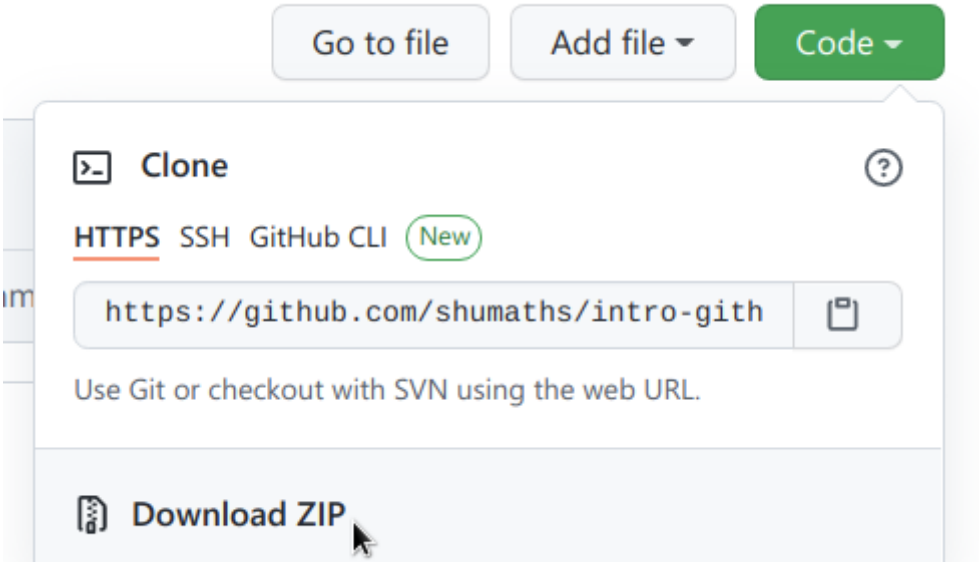
Using GitHub to edit a file and submit your changes

1. A repository is where your code is stored.

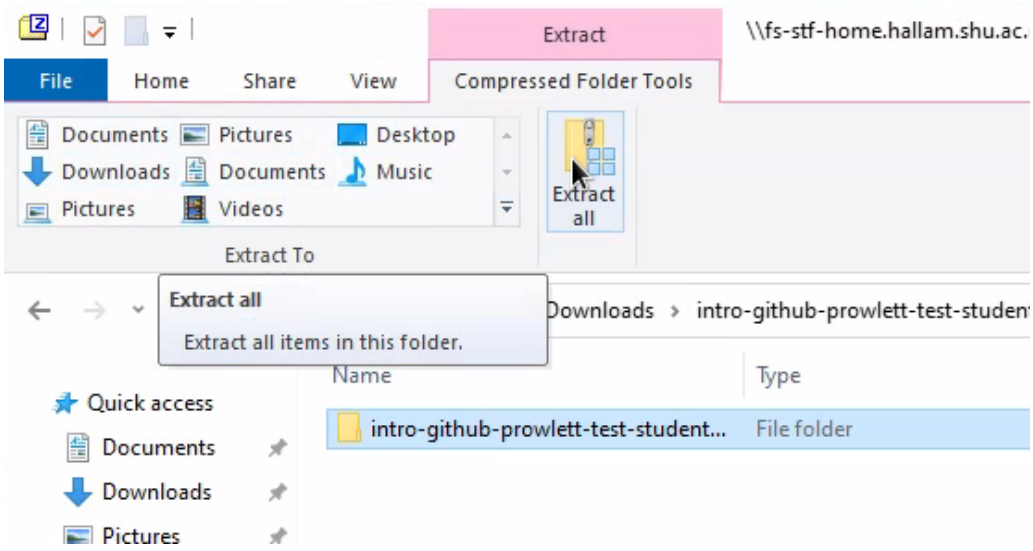


2. You can edit files on the GitHub site, but you should get in the habit of downloading a copy of the file you are going to edit. This is because we will be mostly editing Python code, and you need a copy of the file on your computer to be able to run the code.

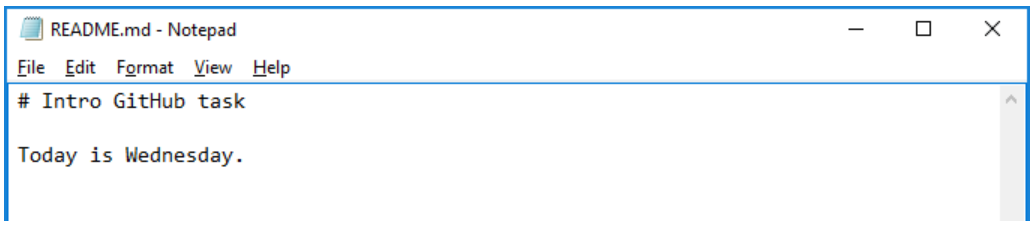
   The easiest thing to do here is to click the 'Code' button and choose 'Download ZIP'. This will download a zip file containing the whole repository to your computer.

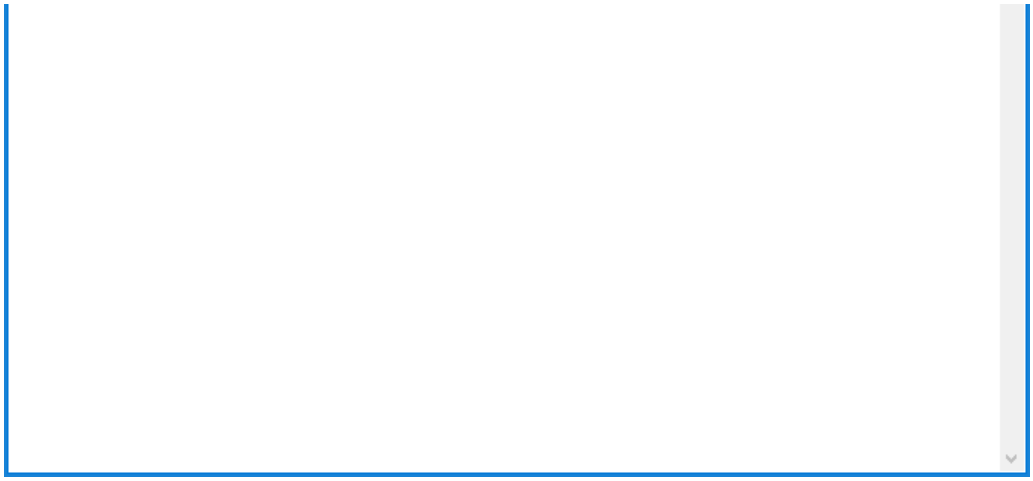


3. Once you have downloaded the zip file, please unzip it - don't just double-click to look at the contents. You should unzip the repository into a sensible place such as your folder for this week's work. Right click or look in the menu for an option to "extract" the files.

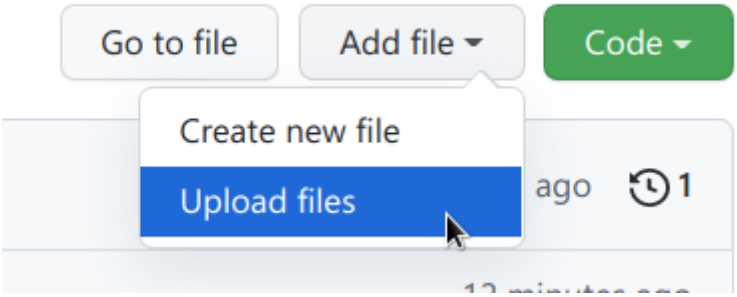


4. Edit the file README.md in a text editor. This is a Markdown file, a simple language for writing structured documents. The line starting with # is a heading. Edit the file so it says "Today is Friday." Save the file.





5. Back at the GitHub website, under 'Add file' choose 'Upload files'.

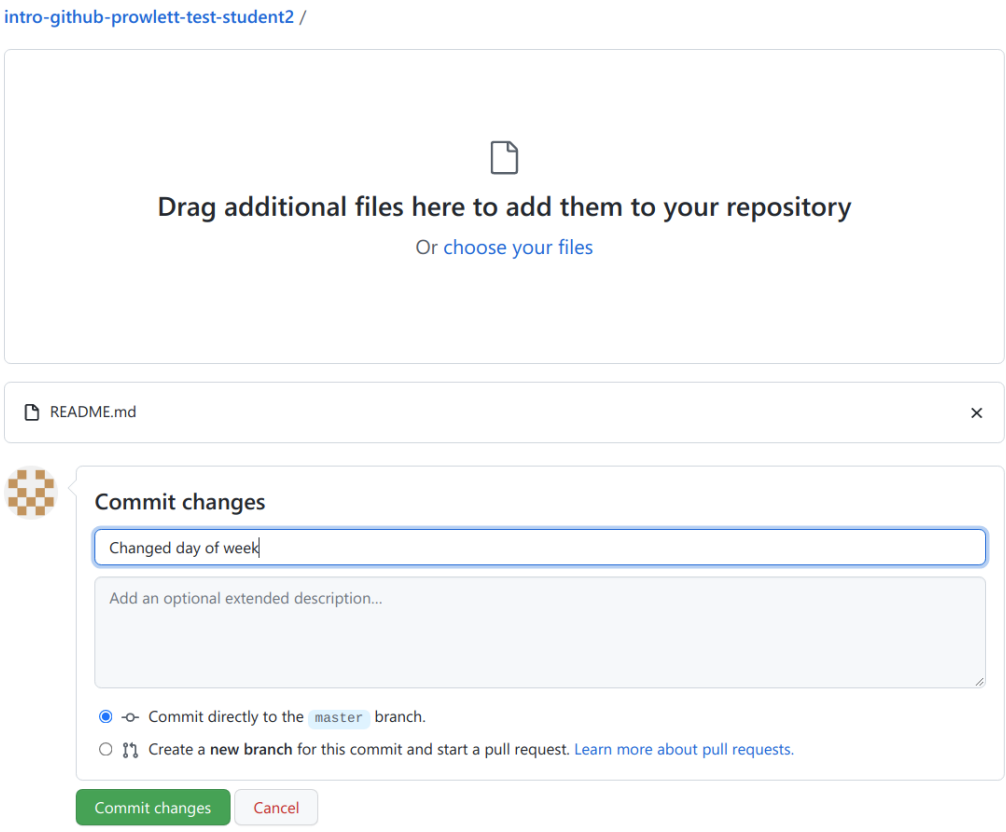


6. Add your file to the page that opens and under 'Commit changes' write a brief summary of your change in the message box (there's no need to put anything under Description).

   Remember Git is a version-control system? What ***commit*** does is makes a version of your repository with that change, and labels this with what you write. This means if you made further edits you could look through the history of the README.md file and see what changes you made when.

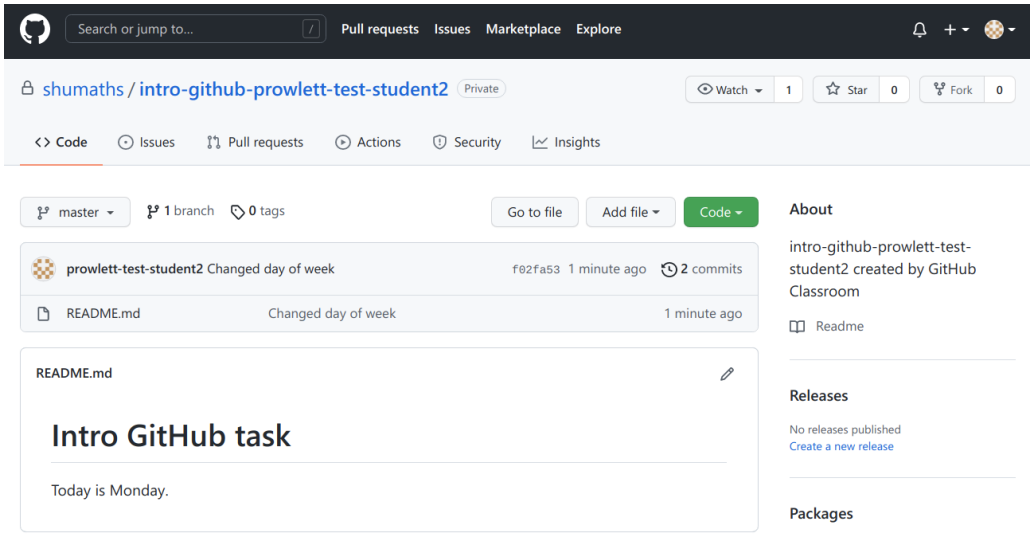
   It is best not to just leave the default message "Add files via upload" for each commit, because when you are looking later through the history of a file's edits, it is good if you have given yourself a hint what changes are being made. This need only be a short note to yourself.

You can commit your changes here directly (which should be selected as the default option). You only need worry about branches if you are working on a more complicated project or collaborating with others.



7. Click Commit changes and you should be returned to your repository with your changes implemented. Here, the README.md file will show

with your changes implemented. Here, the `README.md` file will show the changed contents, and the most recent change at the top should show the message you typed.



In summary, the process is:

- download the files to your local system and edit them - here we edited the markdown file using a text editor, but typically you might edit a Python file using your Python editor (e.g. IDLE).
- upload the files back to the GitHub website and commit the changes directly to GitHub. This is so that I can see your code too and give you feedback.

Next time - weekly exercises

In future, the weekly exercises are in the form of a GitHub Classroom link which you will click to create your own repository. You can then edit the files or add your own (depending on the exercise) and upload these changes to your GitHub repository in the same way as you did here.