



# Working with GitHub

---

## Content

What is Git? .....	2
GitHub .....	2
What is Github? .....	2
Introduction to GitHub Terminology .....	3
Getting Started with GitHub .....	4
Markdown.....	6
Using Markdown on GitHub .....	6
Creating an Issue .....	7
What makes a good issue?.....	8
Repository Structure .....	8
Pull Requests .....	12
Making a Pull Request .....	12
Merging a Pull Request.....	14
Other Functionalities .....	16
GitHub Pages .....	16
Turning your GitHub Repository into a Website .....	16
Projects.....	18
Further Resources.....	19



## What is Git?

Git is a version control system that helps you keep track of file changes. It is also possible to track who made the changes. This makes Git perfect for working with others in writing code for example. Git is used on platforms such as GitHub, Bitbucket and GitLab. Git and GitHub are two different things.

To learn more about Git, follow the [online course on W3Schools](#).

## GitHub

### What is Github?

[GitHub](#) is a platform that allows anyone to create, store, manage, and share content. Oftentimes this is code. But that is not the only thing present in GitHub. Here are some other use cases for GitHub:

- Software development
- Fanfiction
- Zines
- Community/crowdsourcing projects
- Finding file samples
- Educational resources
- Reading lists
- Datasets
- Blog posts
- Documentation

So why is GitHub great to use? GitHub allows people to work collaboratively on any project, with tracking and control settings favouring non-linear workflows and multiple contributions at a time. By tracking everything, there is also a lot of data integrity. Additionally, you can use GitHub to let the developers know you have an issue/something is not working.

To start, read more about GitHub [here](#). One thing to bear in mind is that GitHub, while it is an amazing tool, does come with a lot of jargon. A great explanation for some of the terminology around GitHub can be found in this guide.

As digital archivists, we often use open-source tools to aid us with e.g. identifying and validating digital objects or harvesting websites. Here are some useful examples of GitHub repositories used in the field that you can find:

- <https://github.com/openpreserve/jhove>
- <https://github.com/digital-preservation/droid>
- <https://github.com/webrecorder/browsertrix-crawler>
- <https://github.com/noord-hollandsarchief> (see the repositories in this account)

There are a lot of people working with digital archives that use GitHub. This is the reason that it is important for everyone to learn how to use it. Additionally, to show you that it is not just a scary programmer platform. A full introductory tutorial on GitHub can be found [here](#).



## Introduction to GitHub Terminology

A great resource for finding GitHub terminology can be found [here](#). However, for the purposes of this guide, here is a quick table of basic terms that may help:

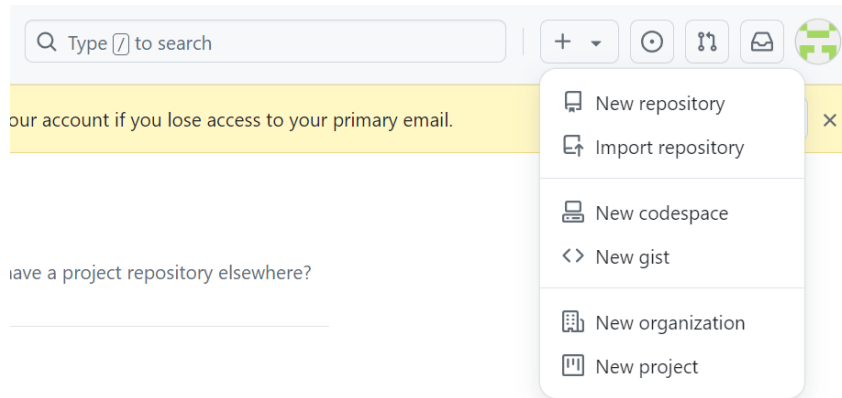
Term	Definition
<b>Repository</b>	A repository is the most basic element of GitHub. They're easiest to imagine as a project's folder. A repository contains all of the project files (including documentation), and stores each file's revision history. Repositories can have multiple collaborators and can be either public or private.
<b>Issue</b>	Issues are suggested improvements, tasks or questions related to the repository. Issues can be created by anyone (for public repositories), and are moderated by repository collaborators. Each issue contains its own discussion thread. You can also categorise an issue with labels and assign it to someone.
<b>Commit</b>	A commit, or "revision", is an individual change to a file (or set of files). When you make a commit to save your work, Git creates a unique ID (a.k.a. the "SHA" or "hash") that allows you to keep record of the specific changes committed along with who made them and when. Commits usually contain a commit message which is a brief description of what changes were made.
<b>Branch</b>	A branch is a parallel version of a repository. It is contained within the repository, but does not affect the primary or main branch allowing you to work freely without disrupting the "live" version. When you've made the changes you want to make, you can merge your branch back into the main branch to publish your changes.
<b>Fork</b>	A fork is a personal copy of another user's repository that lives on your account. Forks allow you to freely make changes to a project without affecting the original upstream repository. You can also open a pull request in the upstream repository and keep your fork synced with the latest changes since both repositories are still connected.
<b>Push</b>	To push means to send your committed changes to a remote repository on GitHub.com. For instance, if you change something locally, you can push those changes so that others may access them.
<b>Pull</b>	Pull refers to when you are fetching changes and merging them. For instance, if someone has edited the remote file you're both working on, you'll want to pull in those changes to your local copy so that it's up to date.
<b>Markdown</b>	Markdown is an incredibly simple semantic file format, not too dissimilar from .doc, .rtf and .txt. Markdown makes it easy for even those without a web-publishing background to write prose (including with links, lists, bullets, etc.) and have it displayed like a website.



## Getting Started with GitHub

To create your first GitHub account you will first need to register at [www.github.com](https://www.github.com) and follow the instructions provided [here](#). You will need an email address, password and potentially 2-Factor Authentication.

Once you have created your account in the top right corner you will see a number of buttons. Click on the + sign and then the **New Repository** tab.



A screen will then appear with different options to set up your repository, you will need to keep the repository public to turn it into a website. The main parts to fill out are the name of the repository and a brief description of it. You can add a license if you wish to.

### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?  
[Import a repository.](#)

*Required fields are marked with an asterisk (\*).*

**Repository template**  
No template  
Start your repository with a template repository's contents.

**Owner \*** tnafrancesca / **Repository name \***

Great repository names are short and memorable. Need inspiration? How about [cautious-succotash](#) ?

**Description (optional)**

☒ **Public**  
Anyone on the internet can see this repository. You choose who can commit.

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

**Initialize this repository with:**  
☐ **Add a README file**  
This is where you can write a long description for your project. [Learn more about READMEs.](#)

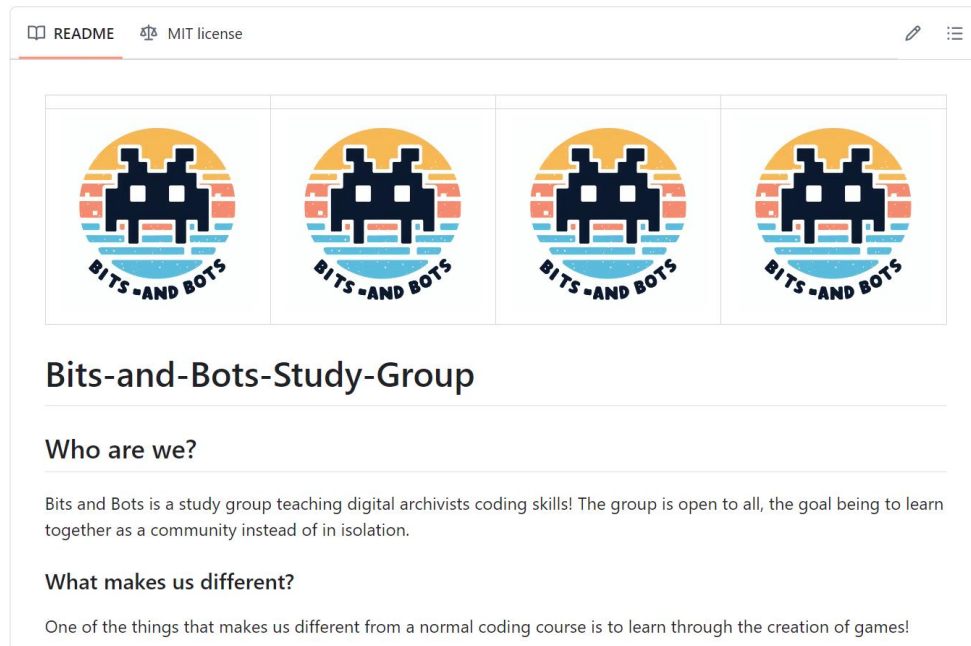
**Add .gitignore**  
.gitignore template: None  
Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

**Choose a license**  
License: None  
A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

🔔 You are creating a public repository in your personal account.



Tick the box to set up the README, this will tell everyone what your website is about when they come to your page. After clicking **create repository** then you can create your README. A README for a GitHub repository looks a bit like this below:



For a basic README you can just type some text in the box to tell everyone what the project is about. However, a README uses a type of programming language called markdown. It is possible to use styling options and add pictures to your README using the Markdown language.



## Markdown

Markdown is a lightweight markup language with a plain text formatting syntax. It is a tool used to convert text to HTML for easy content management. You can use it to write README pages about your repositories in GitHub as well as format GitHub pages. It is simpler and quicker to write than HTML, if you aren't looking to do anything too complicated.

To style words in Markdown, you simply need to surround those words with some special characters. You can use those around one or multiple words. Below are some examples:

Markdown	Style	Example	Output
<code>_</code>	Italic	<code>_Bacon_</code>	<i>Bacon</i>
<code>*</code>	Italic	<code>*More Bacon*</code>	<i>More Bacon</i>
<code>**</code>	Bold	<code>**Spam**</code>	<b>Spam</b>
<code>__</code>	Bold	<code>__Eggs__</code>	<b>Eggs</b>
<code>**_</code>	Italic and bold	<code>**_Python_**</code> <code>_**Python**_</code>	<b><i>Python</i></b>
<code>~~</code>	Strikethrough	<code>~~Crossed off~~</code>	<del>Crossed off</del>

In HTML, there are six headings that you can define with the `<h>` tag. In Markdown, you simply use a hash mark (`#`). For heading 1 you use one `#`, for heading 2 two `##` et cetera, until you are at heading 6: `#####`

You can also create a heading in bold or italics. For example:

```
#### This is heading 4 with _one_ word in italics, written in Markdown.
```

## Using Markdown on GitHub

With Markdown you can create README files in your GitHub repository. These files can be used to tell other people why your project is useful, what others can do with your project, and how they can use it. The README file is often the first item a visitor will see when visiting your repository.

You can define relative links and image paths in your rendered files to help readers navigate to other files in your repository.

A relative link is a link that is relative to the current file. For example, if you have a README file in root of your repository, and you have another file in docs/CONTRIBUTING.md, the relative link to CONTRIBUTING.md in your README might look like this:

```
[Contribution guidelines for this project](docs/CONTRIBUTING.md)
```

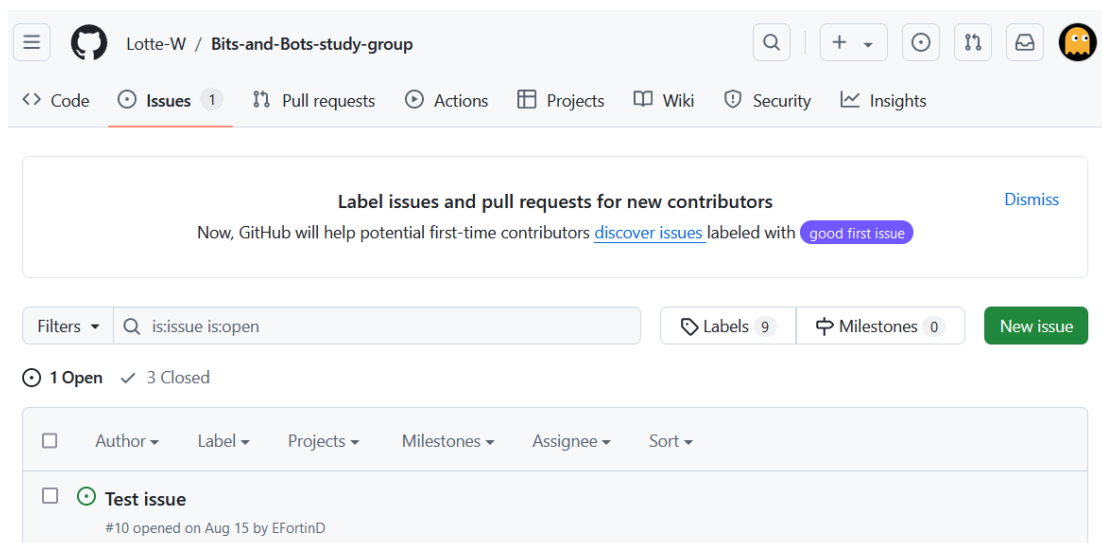


GitHub will automatically transform your relative link or image path based on whatever branch you're currently on, so that the link or path always works. The path of the link will be relative to the current file. Links starting with / will be relative to the repository root. You can use all relative link operands, such as ./ and ../.

Relative links are easier for users who clone your repository. Absolute links may not work in clones of your repository - we recommend using relative links to refer to other files within your repository.

## Creating an Issue

To raise an issue simply go to a repository, for example the Bits-and-Bots-study-group, and click on the issues tab.



You'll then see a list of all the existing issues, check that nothing similar exists that you can't already comment on or add to. If that is the case then you can click the Submit New Issue button.

Add a title

Add a description

Write Preview

Markdown is supported

Paste, drop, or click to add files

Submit new issue



## What makes a good issue?

Issues should be:

- Very clear, give enough information so that other people can understand the issue as well.
- Try and add detail. The more detail, the better.
- If possible, also try to provide the solution as well as the problem.

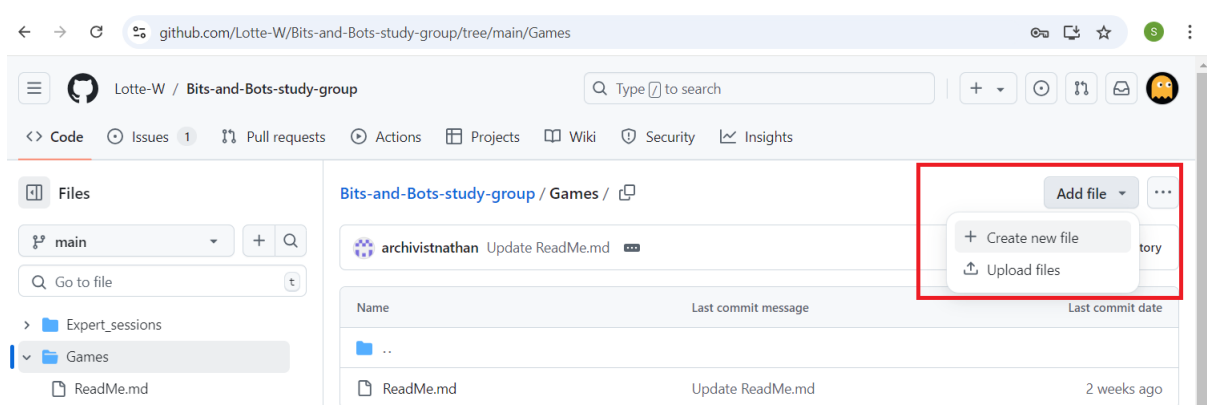
This [example of a bad issue](#) gives insight why we consider this to be a ‘bad’ issue. Someone commented on it that more context should be added, because it is unclear what the issue is exactly about. This other [example of a good issue](#), shows that the issue is more clear now.

## Repository Structure

In this section, we explain a bit more on the structure of your repository and how to add, rename and delete files and folders.

You can create new files directly on GitHub in any repository you have writing access to. If you don’t have access to it, you can fork the project to your personal account and send a pull request to the original repository after you commit your change. This is not only the case for adding files, but for all the changes you want to make to a repository you don’t have access to.

To add a file on GitHub of a repository you have writing rights to, navigate to the main page of the repository and browse to the folder where you want to create a file. On the top right, you can select an ‘Add file’ dropdown menu, and then click ‘Create new file’.

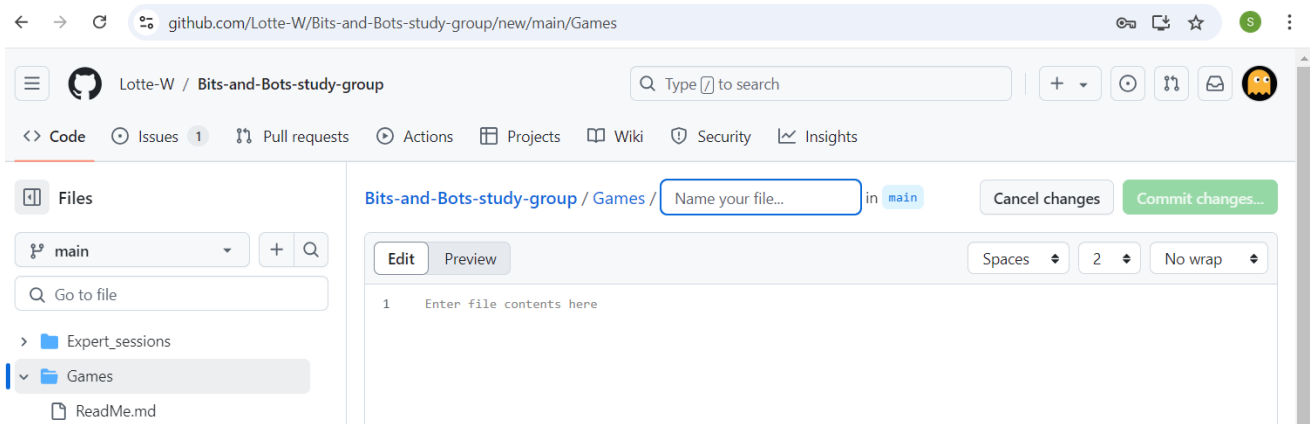






Type the name and extension for the file in the name field. Content for the file can be typed in the contents text box. If you want to review your content first, click the 'Preview' tab on the top left, next to the 'Edit' tab. If you're finished, click on 'Commit changes...' (green button on the top right).

If you want to place your newly created file in a new folder, you can use the filename field. Type the name for the folder and end with a / and type the name of the file after the slash in the newly appeared filename field. In the screenshot above you see the name of the directory (Bits-and-Bots-study-group) and the name of the folder (Games) before the filename field. Note how every part is divided by forward slashes.



A new window will appear for the changes (see screenshot below). Type a brief message that describes the change you made to the file. Below the commit message field, you can choose to add your commit to the current branch or to a new branch. If you create a new one, you can start a pull request.

Commit changes

Commit message

Create example

Extended description

Add an optional extended description..

☒ Commit directly to the main branch

☐ Create a new branch for this commit and start a pull request [Learn more about pull requests](#)

Cancel Commit changes



It is also possible to upload an existing file. To upload a file, navigate to the main page of the repository and browse to the folder where you want to upload the file. On the top right, you can select an 'Add file' dropdown menu, and then click 'Upload files'. You can choose your files or drag and drop them to add them to the repository. After you've done so, you can add a commit message as well, to describe what you have done and choose where you want to commit your changes, just like when you would create a new file.

github.com/Lotte-W/Bits-and-Bots-study-group/upload/main/Games

Lotte-W / Bits-and-Bots-study-group

Drag files here to add them to your repository  
Or [choose your files](#)

**Commit changes**

Add files via upload

Add an optional extended description...

☒ Commit directly to the `main` branch.

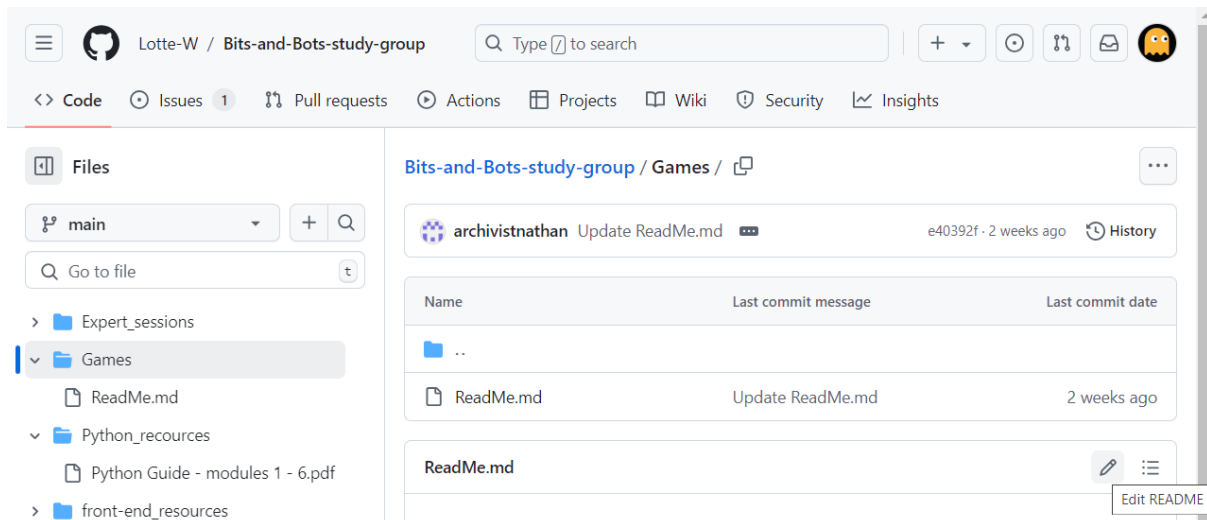
☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

[Commit changes](#) [Cancel](#)

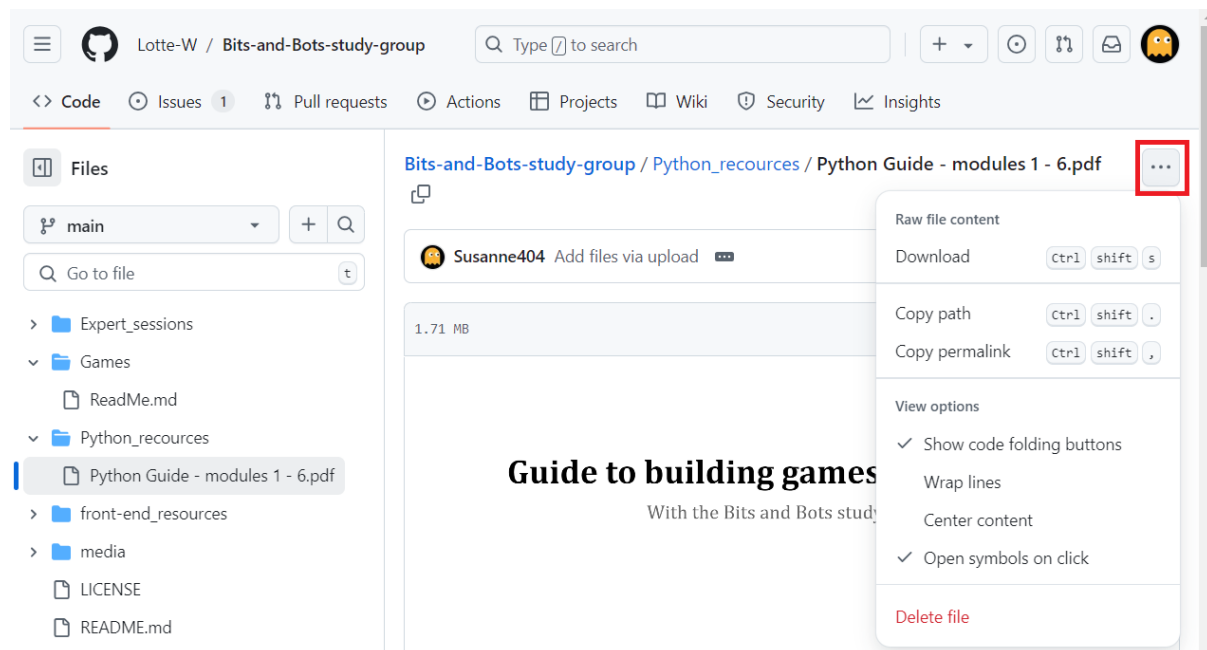
© 2024 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact](#) [Manage cookies](#) [Do not share my personal information](#)



You can rename any file in your repository directly in GitHub. Simply browse to the file you want to rename, and click on the pencil on the right to edit the file. You can change the name of the file in the filename field, and also update the contents of your file if you want. Click on ‘Commit changes...’ and describe what you’ve done and where you want to commit the changes.



Lastly, you can delete an individual file or an entire directory in your repository on GitHub. Browse to the file or the directory you want to delete and select the dropdown menu in the top right corner (with the three dots) and choose ‘Delete file’ or ‘Delete directory’.



More on working with files can be found here:

<https://docs.github.com/en/repositories/working-with-files>

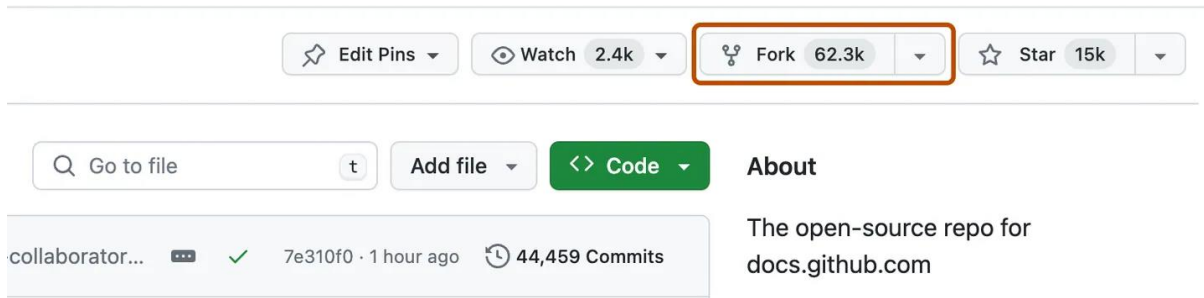


## Pull Requests

A pull request is a proposal to merge a set of changes from one branch into another.

### Making a Pull Request

To make a pull request, you first need to fork your repository. At the top of your repository, you can see how many times a repository has been forked.




After clicking on this, you get the following:

### Create a new fork

A *fork* is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project.

Required fields are marked with an asterisk (\*).

Owner \*  
 Lotte-W / Repository name \*  
File\_Format\_Frenzy  
✔ File\_Format\_Frenzy is available.


By default, forks are named the same as their upstream repository. You can customize the name to distinguish it further.

Description (optional)

An interactive fiction game about file format identification and the dire consequences of getting it wrong

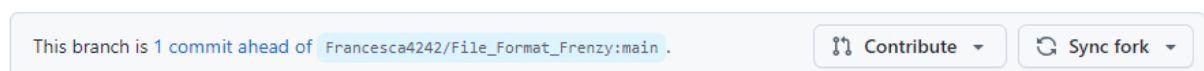
☒ Copy the `main` branch only

Contribute back to Francesca4242/File\_Format\_Frenzy by adding your own branch. [Learn more.](#)

 You are creating a fork in your personal account.

Create fork

Now that you have forked the repository, you can make changes such as adding to the README, adding files, etc. While doing this, you can see how many changes you have made by seeing how many commits you are ahead of the original repository. In this example, one sentence has been added to the README.





By selecting contribute, it gives you the option to open a pull request. When selecting this, you can add a title to your pull request along with a description. You can also see what you have changed and how that compares to the original repository.

## Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#) or [learn more about diff comparisons](#).

base repository: Francesca4242/File\_Format\_Fre...
base: main
head repository: Lotte-W/File\_Format\_Frenzy
compare: main

✓ Able to merge. These branches can be automatically merged.

Add a title

Update README.md

Helpful resources  
[GitHub Community Guidelines](#)

Add a description

Write
Preview

H B I ≡ <> 🔗 ⌨️ @ 🗨️ ↩️

Add your description here...

Markdown is supported
Paste, drop, or click to add files

☒ Allow edits by maintainers
Create pull request

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

1 commit
1 file changed
1 contributor

Commits on Oct 2, 2024

Update README.md
Verified
4dcd212

Lotte-W authored 11 minutes ago

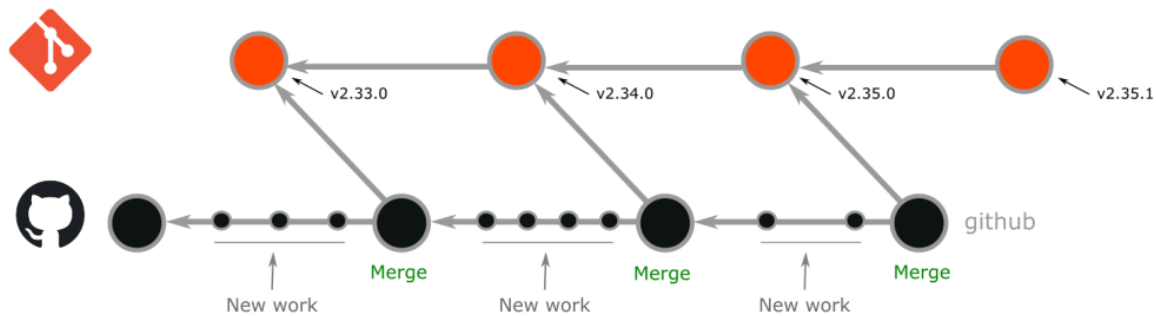
Showing 1 changed file with 2 additions and 0 deletions.
Split Unified

2 README.md

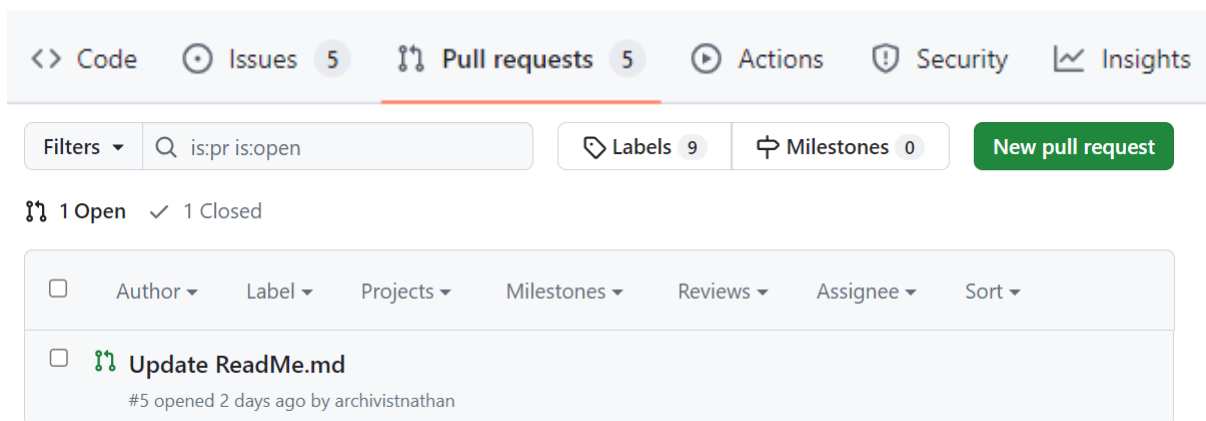
After making sure everything is correct, you can create the pull request. The owner of the repository can look at your request and decide to merge it (accepting your changes).



## Merging a Pull Request



After a pull request has been made, the owner of a repository can merge a pull request to accept the changes that have been proposed by the maker of the request. Go to the pull request tab to see all the requests.



Click on a pull request to see the changes that are proposed. There are separate tabs for conversation (comments on what has changed), Commits (where you can see the changes), and Files Changed (where you can see which files are affected and also the main changes).



Conversation 0 Commits 2 Checks 0 Files changed 1 +2 -0

Changes from all commits File filter Conversations Jump to 0 / 1 files viewed Review changes

Games/ReadMe.md

```

@@ -6,6 +6,8 @@ Below is a list of games that have their own repository or webpage elsewhere
6      6
7      7  ## Python Games
8      8
9      9 + - [ArchiveWordPy](https://github.com/archivistnathan/archiveswordpy): Guess the archives word in
10     10 + this terminal-based game inspired by Wordle
11     11
12     12  ## Front-end Games
13     13
  
```

Under conversation, if you wish to accept the changes you can press Merge pull request. If you are unsure you can comment, and if you wish to reject the pull request then you can use the Close pull request button.

+ New changes since you last viewed View changes

Merge pull request #1 from archivistnathan/archivistnathan-patch-1 Verified 511c04f

Add more commits by pushing to the **main** branch on [archivistnathan/Bits-and-Bots-study-group](#).

**This branch has no conflicts with the base branch**  
Merging can be performed automatically.

Merge pull request You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

Add a comment

Write Preview H B I ≡ <> @ Paste, drop, or click to add files

Add your comment here...

Markdown is supported

Close pull request Comment

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

ProTip! Add [.patch](#) or [.diff](#) to the end of URLs for Git's plaintext views.

You will notice that GitHub has checked to make sure that the pull request does not conflict with any other code you have written in the main branch.



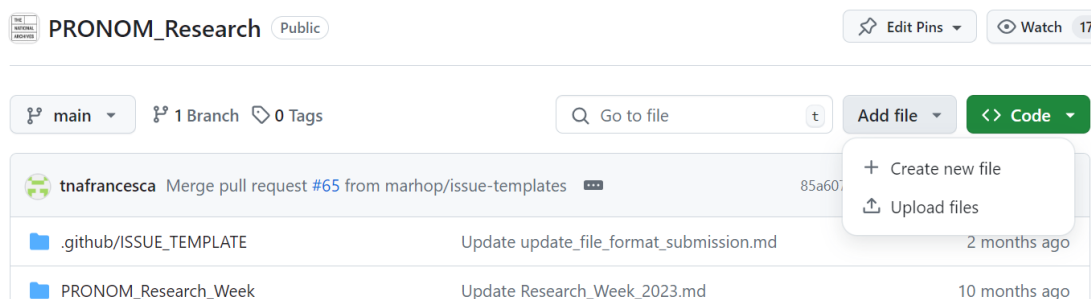
## Other Functionalities

### GitHub Pages

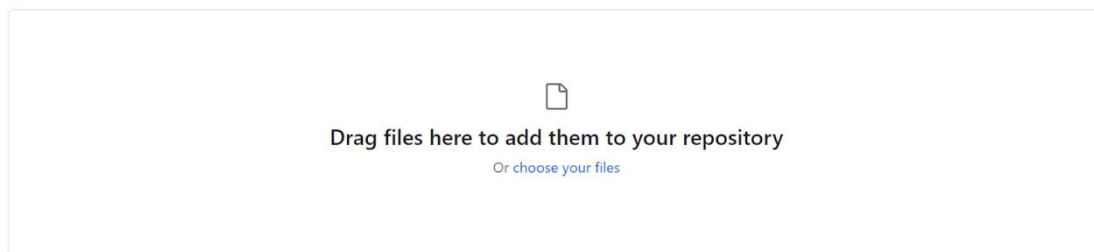
By adding an `index.html` file or a markdown file to your repository you can create your own webpage. By enabling GitHub pages within the settings of the repo GitHub will generate a webpage for you that can be shared online. The address of your repository could be, for example: <https://github.com/Francesca4242/SecretSanta>. Once turned into a webpage, this will become: <http://francesca4242.github.io/SecretSanta>

### Turning your GitHub Repository into a Website

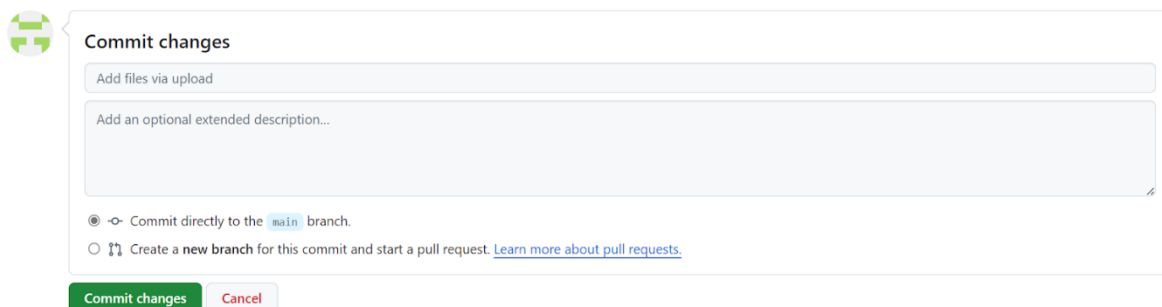
Go back to your `.html` file and change the name of the file (whatever it is called) to `index.html`. Upload this file to your repository by going to the repository page you just created, clicking **add files** and then **upload files**.



Upload the `index.html` file that you just renamed to the repository by dragging and dropping it into the area or navigating to the file in file explorer.



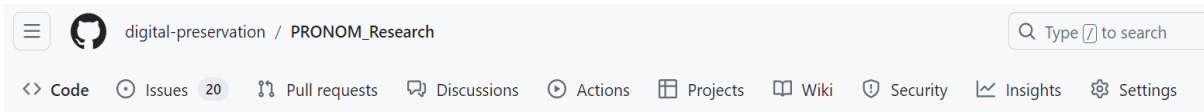
Once done at the bottom type the changes you have made (for example 'added `index.html` file') and press the green **commit changes** button.



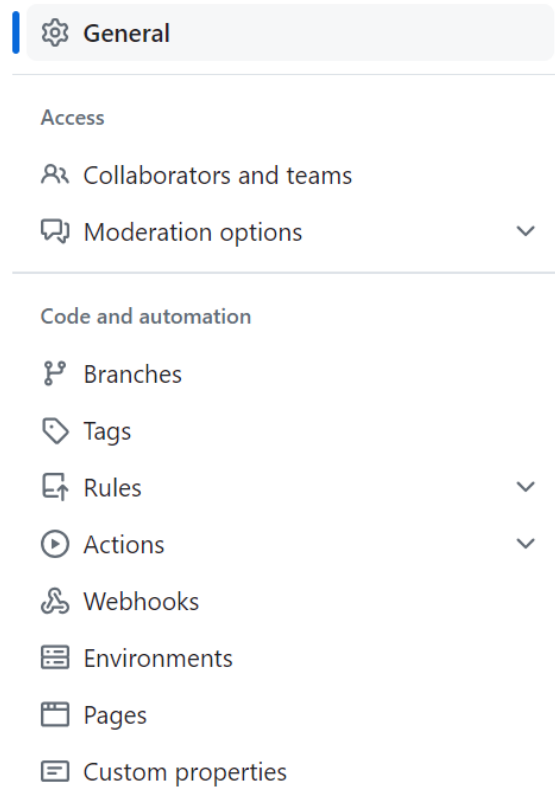




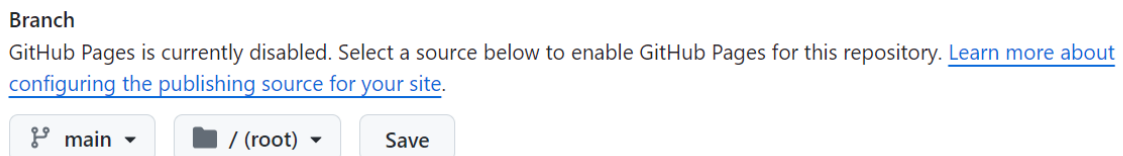
You should now be back at the repository main page, go to settings in the bar at the top on the right hand side.



And then navigate to Pages in the side bar menu.



Second section down select the source as main and click save.



Your game should then appear at the website on your pages (it may take a while to compile). GitHub pages creates a website address based on your GitHub username and the name of your repository. For example, this could be:

[https://\[your GitHub username\].github.io/\[name of repository\]/](https://[your GitHub username].github.io/[name of repository]/)



If you can't find it then the page address should also be displayed on the screen- as below.

## GitHub Pages

[GitHub Pages](#) is designed to host your personal, organization, or project pages from a GitHub repository.

Your site is live at [https://francesca4242.github.io/Game\\_Game/](https://francesca4242.github.io/Game_Game/)

[Visit site](#)

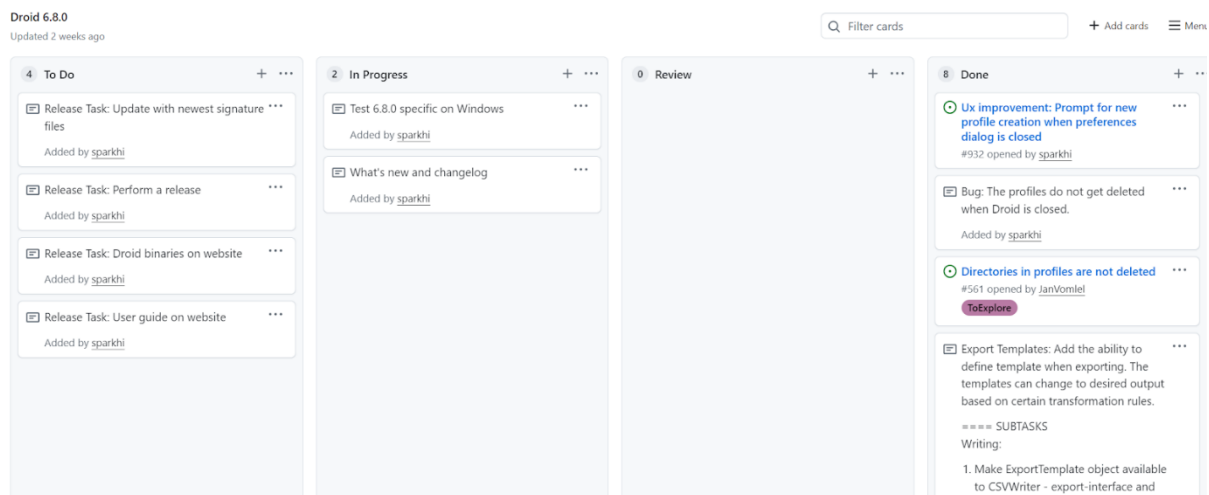


Do not forget to share the link so other people can look at it!

## Projects

GitHub allows the functionality of creating projects to show workflows and collaborate. A project is an adaptable collection of items that you can view as a table, a kanban board, or a roadmap and that stays up-to-date with GitHub data. Your projects can track issues, pull requests, and ideas that you note down.

You can create and customise multiple views by filtering, sorting, and grouping issues and pull requests, visualise work with configurable charts, and add custom fields to track metadata specific to your team. Rather than enforcing a specific methodology, a project provides flexible features you can customise to your needs and processes. You can find more information on starting a project [here](#).





## Further Resources

Skill	Link	Comment
Git	<a href="https://www.w3schools.com/git/default.asp">https://www.w3schools.com/git/default.asp</a>	This is an online course by W3Schools on Git.
GitHub	<a href="https://github.com/skills/introduction-to-github">https://github.com/skills/introduction-to-github</a>	Tutorial on GitHub skills page.
GitHub README files	<a href="https://docs.github.com/en/repositories/managing-your-repositorys-settings-and-features/customizing-your-repository/about-readmes">https://docs.github.com/en/repositories/managing-your-repositorys-settings-and-features/customizing-your-repository/about-readmes</a>	Explanation on GitHub README files: what are they and how to use them.
GitHub issues	<a href="https://docs.github.com/en/issues/tracking-your-work-with-issues/configuring-issues/quickstart">https://docs.github.com/en/issues/tracking-your-work-with-issues/configuring-issues/quickstart</a>	Quickstart for GitHub issues. Interactive guide to learn about GitHub Issues.
Markdown	<a href="https://www.markdowntutorial.com/">https://www.markdowntutorial.com/</a>	Interactive tutorial about Markdown.
Markdown	<a href="https://www.markdowntutorial.com/">https://www.markdowntutorial.com/</a>	Video tutorial about Markdown
Markdown	<a href="https://blog.webdevsimplified.com/2023-06/markdown-crash-course/">https://blog.webdevsimplified.com/2023-06/markdown-crash-course/</a>	Markdown crash course
Markdown	<a href="https://dillinger.io/">https://dillinger.io/</a>	Online Markdown editor, to see how it works and try it out!