Techionista Academy

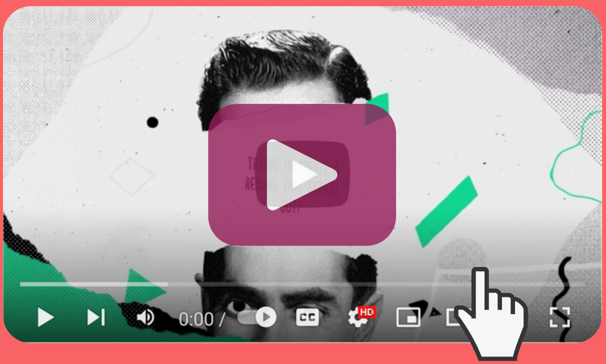
Career Skills

Becoming a GitHub Pro

April 2023

Table of Contents

Videos



If you see anything that has a play button, it surely has a link to a video that you want to watch, so just click play and you'll have a new window with the video opened.

Links

Bold underlined text in Plum color is a clickable link. **I am a link, click me!**

Pointer

This pointing hand icon invites you to click the element it's pointing to.

Becoming a GitHub Pro

GitHub 101

GitHub is an important and widely used professional tool and the field of Technology. In this guide, we will cover why using GitHub is an important skill to acquire as well as how to set up and utilize your profile.

GitHub 101

What is GitHub?

Git and GitHub

GitHub is a program that lets you work together with other GitHub users on various projects. GitHub uses Git, a popular open-source version control software, to track every contribution and contributor. GitHub hosts your files and allows you and your team to (simultaneously) work together on projects from anywhere in the world.

Why you should use GitHub

GitHub allows you to show your work and your general engagement with the tool to the wider public. GitHub is therefore not just used as a collaboration tool, but also frequently referred to by recruiters as a means of building your data science portfolio.

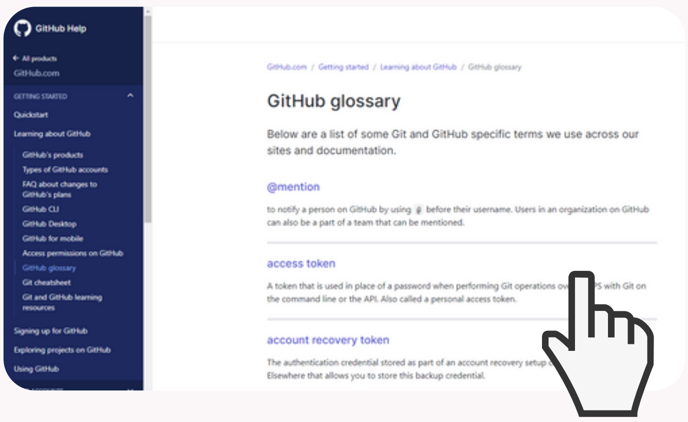
Moreover, it is one of the largest coding communities around, so using it can provide wide exposure for your projects and for you.

This animated video above shows how GitHub can be used along an easy-to-understand real-life example.

The GitHub Lingo

GitHub may seem overwhelming at first. It uses a lot of technical jargon that you may not be aware of yet and uses its own unique language that they have created for some of GitHub's features. Do not worry, we are not expecting you to become experts at GitHub during your time at Techionista. However, we do encourage you to learn at your own pace and take your time getting to know how to utilize GitHub.

Luckily, GitHub has its own glossary that is very extensive. We have provided a link to it below and recommend bookmarking it so you can have it handy whenever you come across something new.



GitHub glossary

Below are a list of some Git and GitHub specific terms we use across our sites and documentation.

@mention

to notify a person on GitHub by using @ before their username. Users in an organization on GitHub can also be a part of a team that can be mentioned.

access token

A token that is used in place of a password when performing Git operations ov PS with Git on the command line or the API. Also called a personal access token.

account recovery token

The authentication credential stored as part of an account recovery setup Elsewhere that allows you to store this backup credential.

Click on the above photo to find the GitHub Glossary.

In addition to this resource, we will provide some brief explanations of the terminology needed to get you started with the basics on the next page.

These definitions should help get you through the rest of this guide and get you started on your own GitHub journey!

GitHub Glossary

Defining some vocabulary

Repository

A repository (repo) is "the main folder" of a single project. It is the place where you organize all folders, files, and data sets relevant to the project.

Projects

You are able to create a project within a repository to organize and prioritize work. Creating a project provides you with project management tools like a Kanban board where you can create custom workflows that meet your needs and keep your project on track! We will not be going too deep into project in this guide, but we recommend checking it out.

README file

A README file is a file in a repository that provides important information about the project for others to read. This file gives additional context to what you are working on that others may not know by just looking at your repository. The README file also is displayed on the main page of your repo.

Commits

When you want to make a change to a project, you create a commit on a feature branch. This will submit a change you want to make to the project. Commits also capture the history of all changes so that contributors can see who made the change and why through a series of "commit messages". Once the commit is ready for discussion with other contributors you will be able to open a pull request.

Pull Request

You can open a pull request to propose your changes you created with your commit. Then, you can request other contributors to review and pull in your contribution and merge them into their branch once the changes have been discussed.

Branches

Branching is the way to work on different versions of a repository at a time. By default, your repository has one master branch which is considered to be the definitive branch. We use branches to experiment and make edits before committing them to master.

Main Branch
The main branch or master branch is the branch that will result in your final project. You do not want to experiment and make changes to this branch or you could make too many changes to the project that end up not working.

Feature Branch
A feature branch is a copy of your main branch, in which you are able to make changes without it affecting the main branch. There can be multiple feature branches being worked on at a time and they are not permanent. Feature branches are meant to be deleted once they are done being worked on and have been merged with the main branch.

⚠ *GitHub is currently undergoing to process of erasing the terminology "master branch" from their content as it has been deemed to be inappropriate. GitHub is replacing it with "main branch" in the future. We introduce it as both in this guide, so that you know to recognize these terms as the same thing. However, for the remainder of the guide we will be using the term main branch.*

Markdown

Markdown is a simple language used to format files within GitHub. It is not something you need to be concerned about learning right away because it is not crucial to the functionality of GitHub, but is important to make things look nice. We will not go over markdown in-depth in this guide, however, it is mentioned a few times. To learn more you can use this **cheatsheet** made by GitHub.

Getting Started

How to set started with your profile

Getting started

Step 1. Sign in to your GitHub account, or click here to create a free GitHub account if you don't already have one. Follow the on-screen instructions to create your account. If you already have created an account you can skip to step 3.

Step 2. While creating your account it may ask you questions like "How many people will you be collaborating with?" and "Are you joining as a student or teacher?" what you select will not greatly change your profile, but we recommend selecting '2 - 5' and 'student.' You will then be asked some questions about your occupation, programming skills, and how you plan to use GitHub. You can choose whichever option best fits your goals. After you complete these steps you will be brought to your profile.

Step 3. You should now be logged in and on your homepage. From here you can do many things like personalizing your profile, start collaborating on projects and jump into exploring GitHub and learning how to use this great tool. In this guide, we will cover how to start doing all three.

Step 4. We will start with your profile. We will cover how to upload a project and the GitHub Learning Lab in other sections of this guide. To get to your profile click the user icon in the upper righthand corner of your screen. A drop-down menu will appear and can click 'Your profile.'

How to create a convincing profile

Employers want to see both a great skill set and a strong aptitude for data science in their new hires. Instead of telling them, why not show them what you can do? Having a portfolio is great public evidence of your skills and interest in the matter. GitHub is a very popular tool to do so and starts with your profile.

Tips

The basics. You can pin your best projects, allowing you to present yourself from your best side (as seen on the second profile in the examples on the next page). The best projects are those where the code is visible and well-documented and where a great README file is attached explaining what you did, why you did it, how you did it, and what you learned.

Be active. If you compare the example profiles on the next page, you can see that one is the profile of someone who just started and one is that of an experienced data scientist. The green checkboxes clearly show which is which by highlighting the contributions throughout the year(s). Recruiters are known to use this tracker as an indicator of your interest in coding, therefore you want to have as many green checkboxes as possible.

Contribute to other projects. A great way to be active is by contributing to other projects. A great guide on how to do so can be found here.

Fill in your personal details. Include your personal details on your profile and make sure they are in line with other social profiles like LinkedIn as recruiters may cross-check. Don't forget to upload a professional picture.

Keep editing. Feel free to edit a project after you published it. In general, sharing work that is still in process is better than sharing nothing. Keep updating your projects.

Getting Started

Tips to creating a great profile

Upload projects relevant to the job you desire. The pin function is not just great to cherry-pick the projects that you excelled at, it is also a great way of highlighting the type of work that you want your future employer to look at. Two areas you should consider are the type of work and sector of your projects:

Type of work
The type of work you highlight may also be motivated by the type of job you desire. For instance, if you want to become a data analyst, data cleaning and data storytelling should be a key part of your portfolio. On the other hand, if you want to be working with customer-facing systems, it is important to show that you can build operational systems, thus you will have to show end-to-end projects.

Sector
Simultaneously the sector may be of interest: If you strive to work for a commercial organization, you could think of working with customer or marketing data. Whereas if you would like to work for a governmental organization social data will be more relevant.

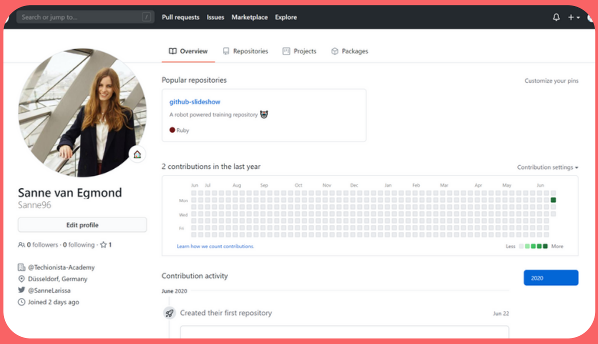
Examples

To the right, there are three examples of profiles. The first is a starter profile, which yours will most likely closely resemble. The other two are professional profiles that have been using GitHub for years and in their everyday life.

Now, do not start getting stressed over your profile. It will take months and years for your profile to look like these professional profiles. Your goal is to start exploring GitHub by starting your profile and learning how to use it at your own pace. We have provided you tips to develop your profile over time, so within the next day or even weeks.

You will have more guidance on how to get started learning GitHub in the following sections of this guide.

Beginners Profile



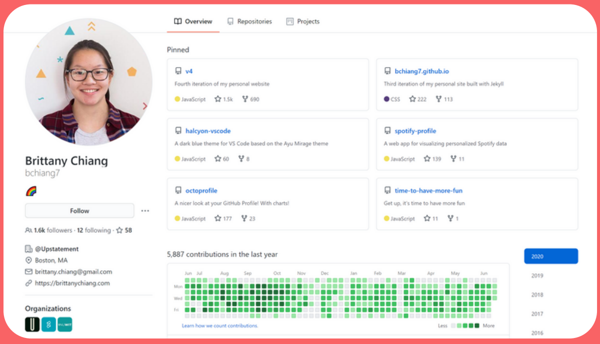
Sanne van Egmond
Sanne96
0 followers · 0 following · 1
@Techionista-Academy
Düsseldorf, Germany
@SanneLarissa
Joined 2 days ago

Popular repositories
github-slideshow
A robot powered training repository
Ruby

2 contributions in the last year

Contribution activity
June 2020
Created their first repository Jun 22

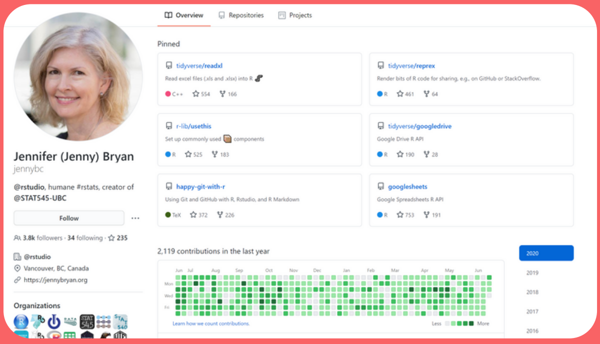
Professional Profiles



Brittany Chiang
bchiang7
1.6k followers · 12 following · 58
@Upstatement
Boston, MA
brittany.chiang@gmail.com
https://brittanychiang.com
Organizations

Pinned
v4 — Fourth iteration of my personal website — JavaScript 1.5k 690
bchiang7.github.io — Third iteration of my personal site built with Jekyll — CSS 222 113
halcyon-vscode — A dark blue theme for VS Code based on the Ayu Mirage theme — JavaScript 60 8
spotify-profile — A web app for visualizing personalized Spotify data — JavaScript 139 11
octoprofile — A nicer look at your GitHub Profile! With charts! — JavaScript 177 23
time-to-have-more-fun — Get up, it's time to have more fun — JavaScript 11 1

5,887 contributions in the last year



Jennifer (Jenny) Bryan
jennybc
@rstudio, humane #rstats, creator of @STAT545-UBC
3.8k followers · 34 following · 235
@rstudio
Vancouver, BC, Canada
https://jennybryan.org
Organizations

Pinned
tidyverse/readxl — Read excel files (.xls and .xlsx) into R — C++ 554 166
tidyverse/reprex — Render bits of R code for sharing, e.g., on GitHub or StackOverflow. — R 461 64
r-lib/usethis — Set up commonly used components — R 525 183
tidyverse/googledrive — Google Drive R API — R 190 28
happy-git-with-r — Using Git and GitHub with R, Rstudio, and R Markdown — TeX 372 226
googlesheets — Google Spreadsheets R API — R 753 191

2,119 contributions in the last year

GitHub Desktop

Here you will learn about more about the GitHub Desktop Application

GitHub Desktop Application

Once you have created an account and set up your profile you will want to download the GitHub Desktop Application.

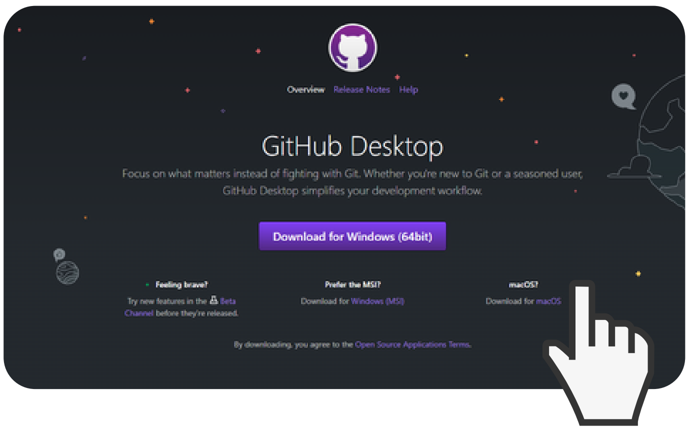
You will notice that during the courses in the Learning Lab you were able to make changes to files directly in GitHub through the browser. Although this is possible, this is not how most people work. This is mainly because there is no way to know if someone else is working in the same exact branch or file as you in this online environment.

The GitHub Desktop Application allows you to download any repository to your own local computer and make changes in the file directly from your computer and in offline settings. You then are able to locate these files in your computer on the Desktop Application and push changes to the online repository that others have access to.

To summarize, using the GitHub Desktop Application allows you to:

- Work offline and from your own local computer
- Download all files from a repository
- Create commits, pull requests, and merge branches from your local computer
- Work in an environment safely, without fear that another collaborator is also working in the same branch or file as you

You can download the desktop application from the GitHub website or by clicking on the photo below.



Overview Release Notes Help

GitHub Desktop

Focus on what matters instead of fighting with Git. Whether you're new to Git or a seasoned user, GitHub Desktop simplifies your development workflow.

Download for Windows (64bit)

Feeling brave?
Try new features in the Beta Channel before they're released.

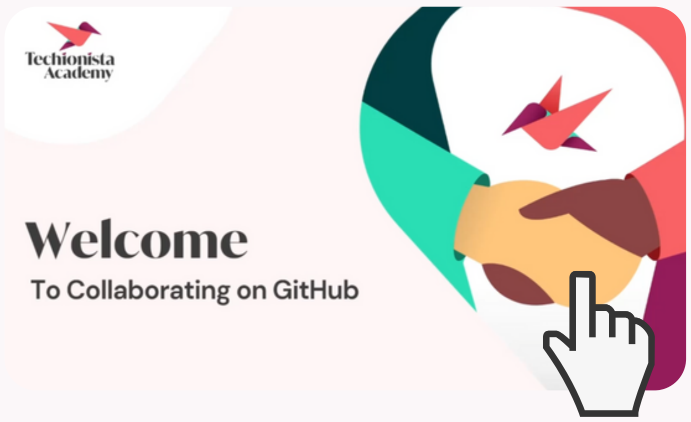
Prefer the MSI?
Download for Windows (MSI)

macOS?
Download for macOS

By downloading, you agree to the Open Source Applications Terms.

Collaborating on GitHub

As mentioned many times throughout this guide GitHub is a powerful tool that helps people collaborate on awesome projects. We have prepared a video that show exactly how it looks and feels to collaborate with one another using GitHub.



Becoming a GitHub Pro

GitHub's Skills Lab

In this section, we will show you how to get started exploring GitHub.

The GitHub Learning Lab

How to start using GitHub

Introduction to the Skills Lab

GitHub has its own Skills Lab with interactive tutorials that will teach you how to do some of the basic and most commonly used features on the platform. There are many different courses available. Some we have required, as they will help you learn the basic functionality of GitHub. We would recommend, as you begin getting comfortable with the platform to continue doing some additional Skills Lab courses that continue to increase in difficulty so that you can become more familiar with the more complex features that it offers.

Accessing the Skills Lab

Accessing the Skills Lab is easy. Let's walk through it.

Step One. You can go to skills.github.com or click here. If you are already signed in it will take you directly to the homepage. If you are not signed in, click the 'sign in' button on the top right-hand corner and log in with your GitHub account information.

Step Two. Once logged in you will find yourself on the Learning Lab homepage. Here you can see all the different courses offered. We recommend starting with the Introduction to GitHub course. Click this course to begin.

Step Three. Once you begin the course a bot will walk you through it step-by-step and provide you with feedback. Below is a video that walks you through the Introduction to GitHub course and explains some of the topics more in-depth.

Learning Lab Courses

Below is a list of some of the Learning Lab courses we recommend trying. We would recommend doing them in the order listed below.

Your goal should be to understand GitHub and how to use it. Do not do these courses to memorize topics and vocab. It is a tool that will benefit you greatly, so take your time when using it for the first time.

- First Day on GitHub: This module can take about 60 minutes and is a great start to learning how to use GitHub!

- Introduction to GitHub: This module takes about 60 minutes and helps you to familiarize yourself with the GitHub jargon.

- First Week on GitHub: This is a collection of 4 learning materials which takes about 2.5 hours to complete. It contains:
    - A GitHub pages module
    - A Reviewing pull requests module
    - A Managing merge conflicts module
    - A Securing your workflows module

Becoming a GitHub Pro

The GitHub Flow

Now that we know some terminology and have a profile ready to start we can learn more about how to utilize GitHub.

Creating a Repository

How to upload your own project

What is a Repository?

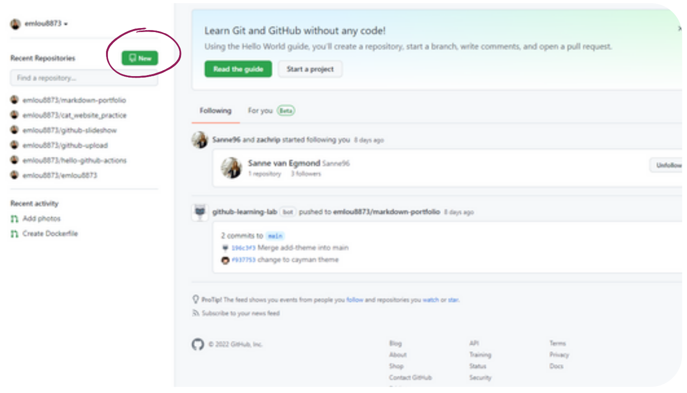
Before delving into simply doing things, let's first see what a repository is and why we must use it.

Repositories (repo) in GitHub represent various different projects and programming languages you have worked on, and showcasing this to potential employers can give them an insight into the skill set you possess. The usual convention is to create a new repo for a new project that you wish to showcase or save on GitHub. Different repos correspond to different things that you have worked on, and there is a nice separation instead of just saving all things in one place.

Now that we know a little bit about why repositories are important, let us see how to create one. Make sure you have a GitHub account and are logged in first, and then follow the steps below.

Uploading your Project

Step One. Once you have logged in, you will be on the homepage. Here you can see all the past repos you have created. Click the "New" button on the lefthand menu to start creating a new repo. If you have never uploaded a project or created a repo through the Learning Lab your screen looks a little different and has a 'Create repository" button instead of "New" on the lefthand menu.



emlou8873 ▾

Recent Repositories — New

Find a repository...

- emlou8873/markdown-portfolio
- emlou8873/cat_website_practice
- emlou8873/github-slideshow
- emlou8873/github-upload
- emlou8873/hello-github-actions
- emlou8873/emlou8873

Recent activity

- Add photos
- Create Dockerfile

Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

Read the guide — Start a project

Following — For you (Beta)

Sanne96 and zachrip started following you 8 days ago

Sanne van Egmond Sanne96 — 1 repository · 3 followers — Unfollow

github-learning-lab (bot) pushed to emlou8873/markdown-portfolio 8 days ago

2 commits to main

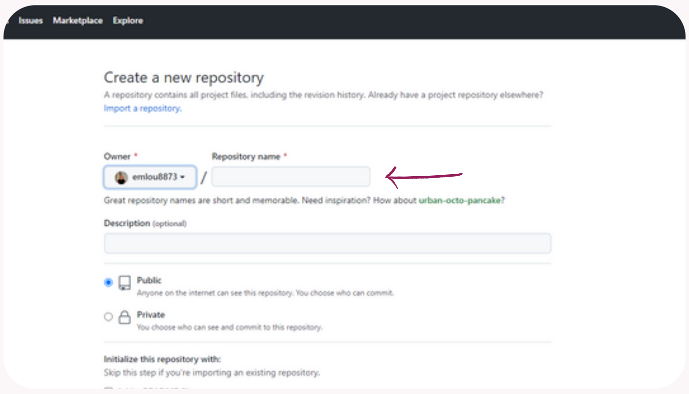
- 196c3f3 Merge add-theme into main
- f937753 change to cayman theme

ProTip! The feed shows you events from people you follow and repositories you watch or star.

Subscribe to your news feed

© 2022 GitHub, Inc. — Blog · About · Shop · Contact GitHub · API · Training · Status · Security · Terms · Privacy · Docs

Step Two. Now you will land on a page like the one below. Here you can name your repository. Typically this should be something related to the topic you are working on. For example, since I want to showcase my hackerrank progress, I called the repo SQL-Project. You can optionally also provide a little description for the repo.



Issues · Marketplace · Explore

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner * — emlou8873 ▾ / Repository name *

Great repository names are short and memorable. Need inspiration? How about urban-octo-pancake?

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.

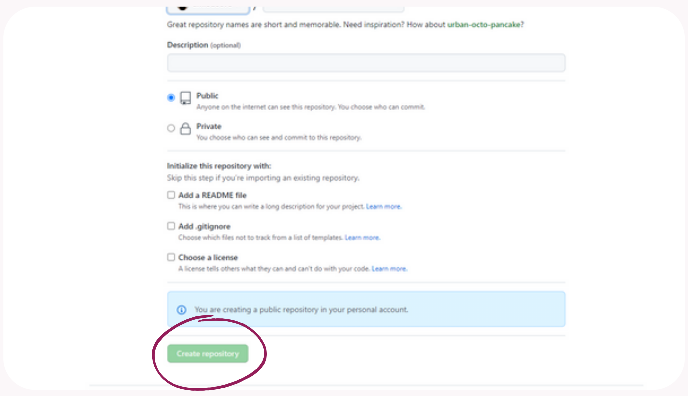
Initialize this repository with:

Skip this step if you're importing an existing repository.

Step Three. Now you want to specify some other things for creating the repo. You might want to start out by making it public, so companies and other users can see what you have done. Private repos are made when working on sensitive projects with information you do not want to share with others.

You will also want to add a README file (this is a description page for your project and people will see it when they open the repository)

Finally click on "Create repository."



Great repository names are short and memorable. Need inspiration? How about urban-octo-pancake?

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:

Skip this step if you're importing an existing repository.

- ☐ Add a README file — This is where you can write a long description for your project. Learn more.
- ☐ Add .gitignore — Choose which files not to track from a list of templates. Learn more.
- ☐ Choose a license — A license tells others what they can and can't do with your code. Learn more.

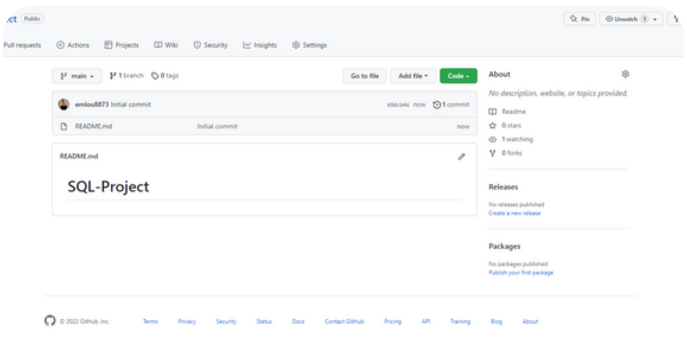
You are creating a public repository in your personal account.

Create repository

GitHub Flow

Learning how to utilize GitHub

Step Four. Congratulations you have created your very own new repository! Now you can add files, code, and other things to it. You can also use it to display your stars and achievements for a particular programming language you are learning.



emlou8873 Initial commit

README.md	Initial commit	now

README.md

SQL-Project

What is the GitHub Flow?

The GitHub Flow is a workflow that makes collaboration and making changes to a project easy. If you have already watched the "Introduction to GitHub" video or completed the Learning Lab course you have already been exposed to this workflow. In this section, we will go over the workflow.

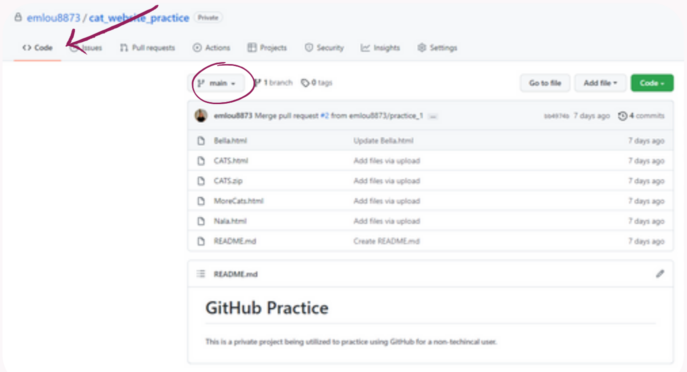
It is important to know that although the workflow seems to follow a specific step-by-step process that the process is not linear. You may spend time going back-and-forth between two sections before moving forward in the flow.

Exploring the Code View

After you upload your project you will see your repo from the 'Code' page, which is also called the "Code View." This page will house all of your files and code for your project.

One thing that is important to notice where is shows what branch you are working on. It should default to the 'main' branch. You can see where this is located in the next screenshot.

A branch is a version of your project. The main branch is your main copy and one that you do not want to make edits directly to.



emlou8873 Merge pull request #2 from emlou8873/practice_1 — 7 days ago — 4 commits

Bella.html	Update Bella.html	7 days ago
CATS.html	Add files via upload	7 days ago
CATS.zip	Add files via upload	7 days ago
MoreCats.html	Add files via upload	7 days ago
Nala.html	Add files via upload	7 days ago
README.md	Create README.md	7 days ago

README.md

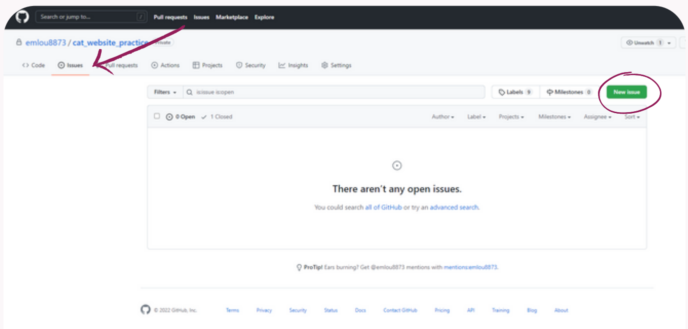
GitHub Practice

This is a private project being utilized to practice using GitHub for a non-techincal user.

Issues

The next page in your repo is "Issues." Here you can discuss improvements, changes, and new features that could be made to the project.

You are able to assign issues to specific users so that you have a clear plan of who is responsible for specific changes being made.



There aren't any open issues.

You could search all of GitHub or try an advanced search.

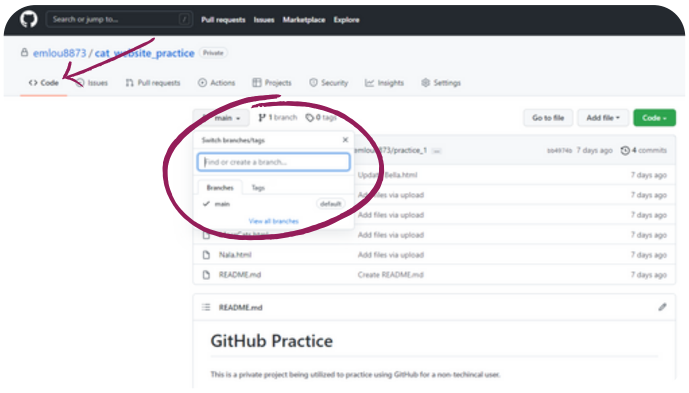
Creating a New Branch

When you have discussed an issue and are ready to start making changes you are able to create a new "feature branch" to work from. This feature branch allows you a safe environment to make changes and try new things without affecting the main branch. Here are the steps to creating a feature branch:

Step One. On the Code View of your repo, locate where the branch indicator is located. It should say "main." Click this and a dropdown menu should occur. This is pictured in a screenshot on the next page.

GitHub Flow

Learning how to utilize GitHub



Step Two. You are able to create a new branch by typing in a name for the new branch and pressing "enter." It is recommended that you name the branch something related to the issue you are working on or a change you are intending to make.

That is all there is to create a new branch. Any changes you make to this branch will not impact the main branch until you are ready to merge the two.

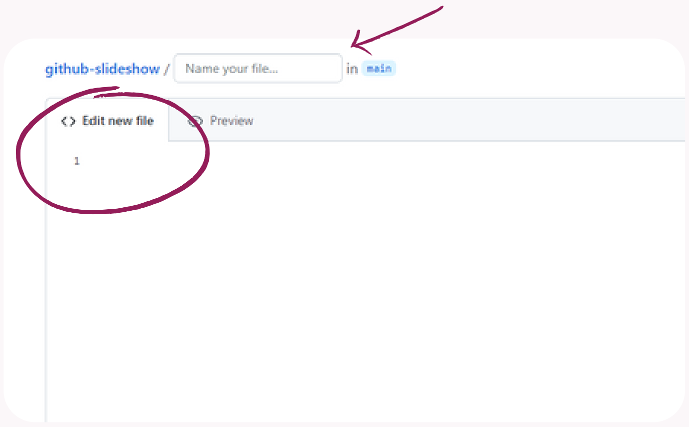
It is important to know that there can be many different branches in-progress. It is recommended that you create a new branch for every issue you are working on.

Commits

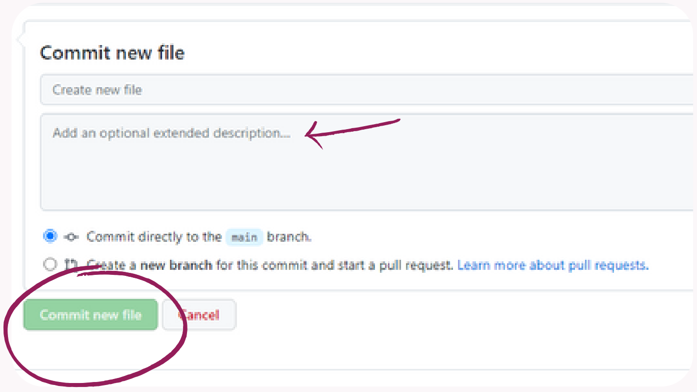
Now that you have created a new branch you can start making changes. To do this you will want to add a new file or "commit a file." Once you have added this to your repo you can work on the changes you want to make. To commit a file you follow these steps:

Step One. Start on your Code View of the repo. Make sure you are working on a feature branch and not the main branch. Once you have ensured you're working on the right branch you can click "Add file" and then "Create new file."

Step Two. Once you click "Create new file" you will be brought to a new page that looks like the next screenshot. Here you can name your file, as well as add the changes you would like to make in the large text section.



Step Three. Before you submit your commit you will want to add a commit message. This is a brief message used to communicate what change you intend to make. This message is usually written in the active voice, is less than 50 characters, and never ends with a period. Once you have added your commit message you can click "commit new file". Now you have submitted your changes!


Commit new file
Create new file
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 new file
Cancel

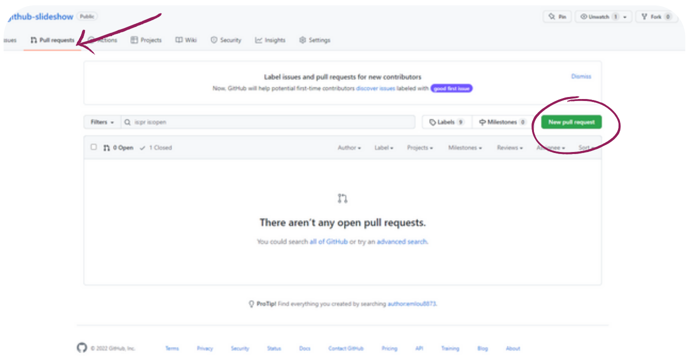
Pull Requests

Like issues, pull requests are used to collaborate and discuss changes being made to a project. The difference is, that in issues we are discussing general changes that could be made to the overall project, and pull requests are discussing a work-in-progress change being made through a new commit. When you create a pull request, you link it to a committed file and open the floor for discussion on a specific action that you are trying to move forward. On the next page, we will explain step-by-step how to open a pull request.

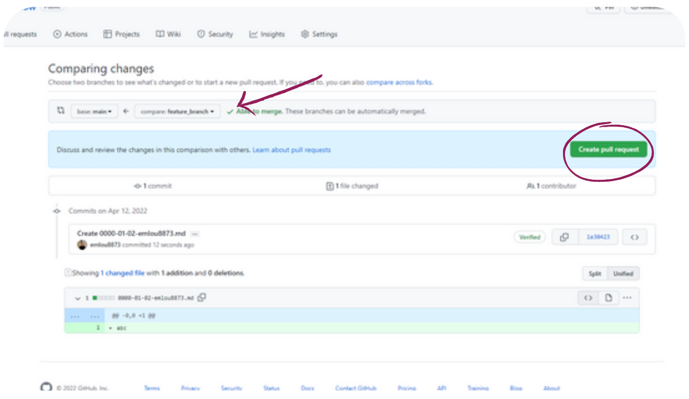
GitHub Flow

Learning how to utilize GitHub

Step One. In your repo click the "Pull requests" tab on the top menu bar. Once you find yourself on the pull request page you can click "New pull request."

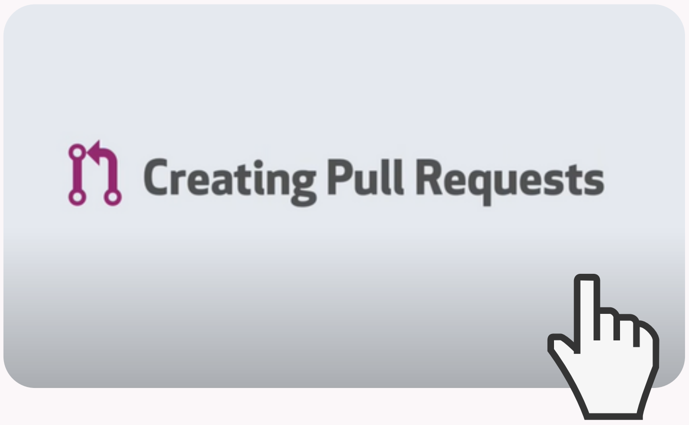

Pull requests
Label issues and pull requests for new contributors
New pull request
There aren't any open pull requests.

Step Two. On the next page, you will be able to select your base branch and a compare branch. The base branch is the branch you would merge your changes to and is most commonly your main branch. The compare branch is a feature branch that you made your commit on. It is important to note that you cannot do a pull request unless a commit has already been completed. Once you select your base and compare branches it will show the commits that have been created. You select which commit you want link to the pull request and then click "create pull request."



Step Three. Now you will create a title and add a description of the pull request, so other collaborators understand what you are trying to accomplish with your changes. Once you add your message you can click "create pull request" one last time. This can be seen in the next screenshot. Once you have opened a pull request other contributors can discuss the changes and if everything looks good you will be able to merge your commit to the main branch!

Pull requests can be on the more complex parts of the GitHub flow. GitHub has a helpful and quick video to explain pull requests further, which you can find below.

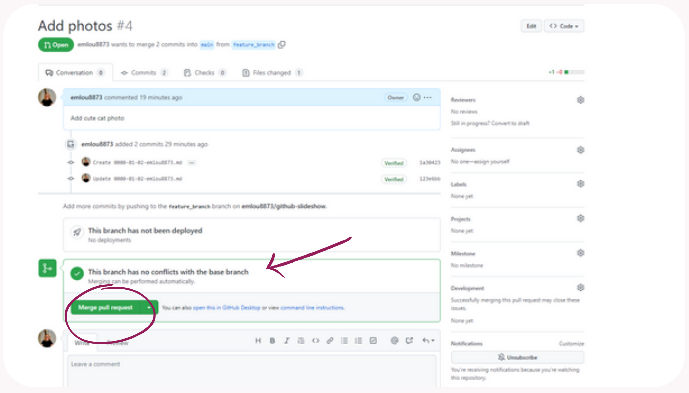


Merging Pull Requests

After the pull request has been discussed and the changes have been proven to be successful it is time to merge the pull request into the main branch.

Step One. In your repo go to the "Pull requests" page and click the pull request that is ready to be merged.

Step Two. Once you are in the pull request you will see the green button that says "Merge pull request" click this when you are ready to fully commit your changes to the main branch.


Add photos #4
This branch has not been deployed
This branch has no conflicts with the base branch
Merge pull request

It will say right above the "Merge pull request" button if there are any conflicts with merging this pull request to the main branch. You did it! You merged a pull request and have made a change to the main branch and closed the "workflow loop." The last step is to delete this feature branch and start a new one!

GitHub Flow

Adding Stars to your README file

What is a README file?

A README file is the first page that anyone will see when they open a repository, hence you want to fill it with the most important information that is representative of what your repository is about. You are also able to add stars and badges to this. Below we will walk you through the steps to add the stars you gained from the hackerrank exercise that you may have already completed in the course.

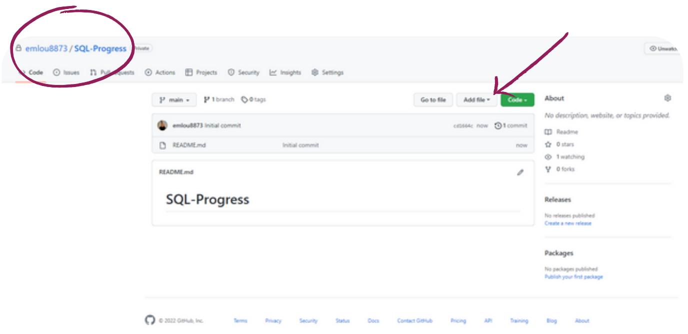
If you have not completed the exercise, this will still serve as a great resource and tool for future use.

Hackerrank Stars

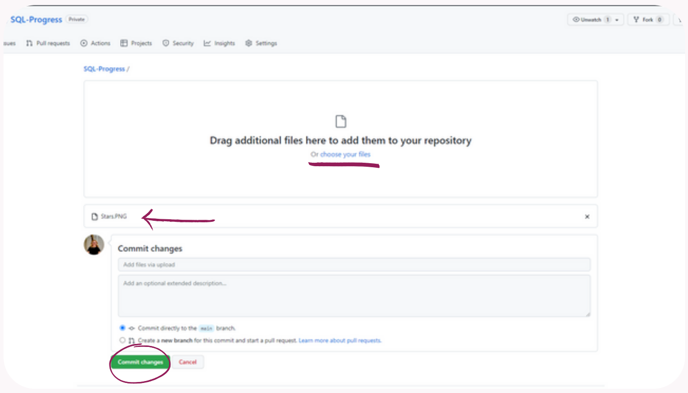
In order to go through the steps of this mini tutorial to showcase the stars you have obtained on hackerrank, first, make sure you have a GitHub account and you have created a repository already. If you have not created a repo, follow the tutorial for creating a repo from earlier on in this guide, and come back.

Firstly, make sure you already have a screenshot of your badge/star saved on your computer because you will be asked to upload this. Now let us follow the steps:

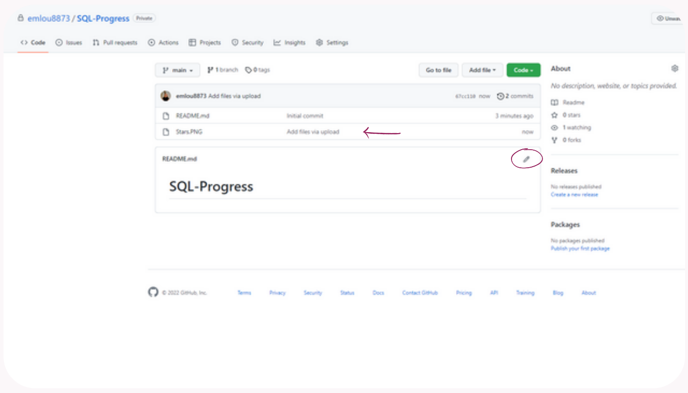
Step One. Go to your desired repository, which in my case will be called SQL-Progress which I already created before. Now click on "Add file" in the top right corner and from the dropdown select the "Upload files" button.



Step Two. Once you are on this new page, you can select "choose your files", where you can browse your computer and select the location in which you saved the screenshot to your star/badge. Once you have added the picture it should show up at the bottom, for example, you can see that I added "stars.png". After you have added all the files you wanted, click "Commit changes" to save them to the repo.


SQL-Progress
Drag additional files here to add them to your repository
Or choose your files
Stars.PNG
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.
Commit changes
Cancel

Step Three. Now you will be redirected to the homepage of your repository, and here you can see the image that you just added! Now we want to display this image on the README file. In order to edit the README file, click the little pencil icon on the top right of the README.md heading.



GitHub Flow

Adding Stars to your README file

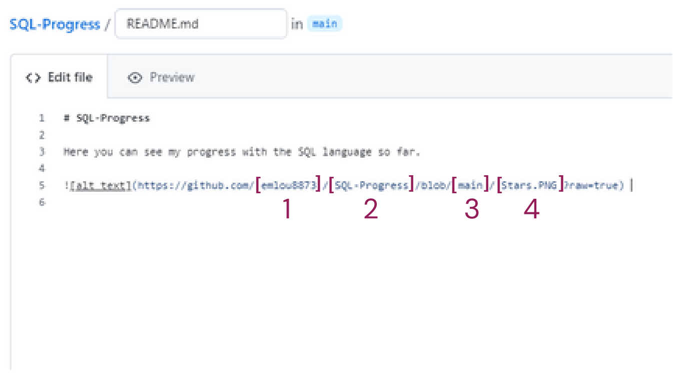
Step Four. Now we have the interface for the README file in front of us. Readme files are written in a markup language called markdown (this is not something we will cover in this guide, but it is covered in the Learning Lab).

To add the photo you will want to type the following image URL into your README file and fill in the numbers with the corresponding information below:

![alt text](https://github.com/1/2/blob/3/4?raw=true)

1. Your GitHub username
2. The name of the repo you are working in (remember if there are spaces or capital letters in this, don't forget to also add it here)
3. The branch you are working on (in our case, main)
4. The name of the file you uploaded (Make sure to also use the correct file extension name)

Below is an example of what this should look like using [] brackets to indicate where to plug in the information above. You should not use these same brackets as these are only being used to show you where the information should be inserted. There is a second screenshot that shows the information as it should be saved to properly show the stars.

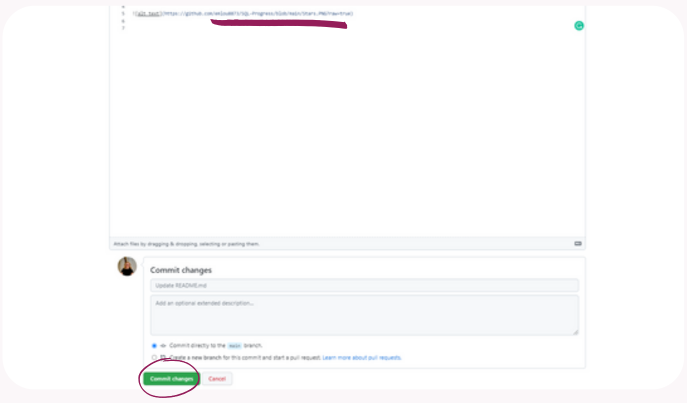


```
# SQL-Progress

Here you can see my progress with the SQL language so far.

![alt text](https://github.com/[emlou8873]/[SQL-Progress]/blob/[main]/[Stars.PNG]?raw=true)
```

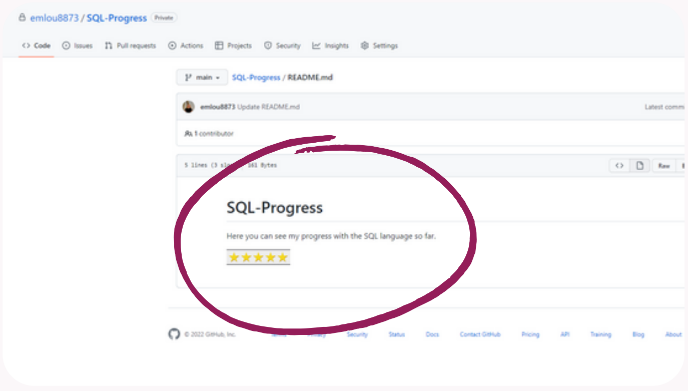
Step Five. So now that you have written the correct path, remove all the square brackets [ ] because they are not contained in the actual URL of the image. Now it should look something like the picture below. Once you have ensured that the image URL is correct you can click "Commit changes."



```
![alt text](https://github.com/emlou8873/SQL-Progress/blob/main/Stars.PNG?raw=true)
```

Attach files by dragging & dropping, selecting or pasting them.
Commit changes
Update README.md
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

Step Six. Now on your homepage, you should see the image you have uploaded and added to your README right away! If the image does not show, go back and check the URL of the image you put in your README file for the typos.



SQL-Progress

Here you can see my progress with the SQL language so far.

★★★★★

Becoming a GitHub Pro

Guide Feedback

Thank you for taking the time to review this guide. Whether you read it completely through, skimmed it for information, or only ready a part or two we would appreciate your feedback! Any feedback will help us ensure we are sharing the best information with our talents and can help us improve our guides for the future. You have the option to share your feedback anonymously. If you would like to provide feedback click the button below:

Provide Feedback Here