UNIVERSITY OF ALBERTA

GitHub

Software Development Platform

UNIVERSITY OF ALBERTA

Git versus GitHub.

Git is a version control system. We can use it to keep track of changes and versions of a project as we develop it.

GitHub is a web application, capable of storing git repositories. It also offers a host of other collaborative, sharing, and development pipeline features.

About GitHub.

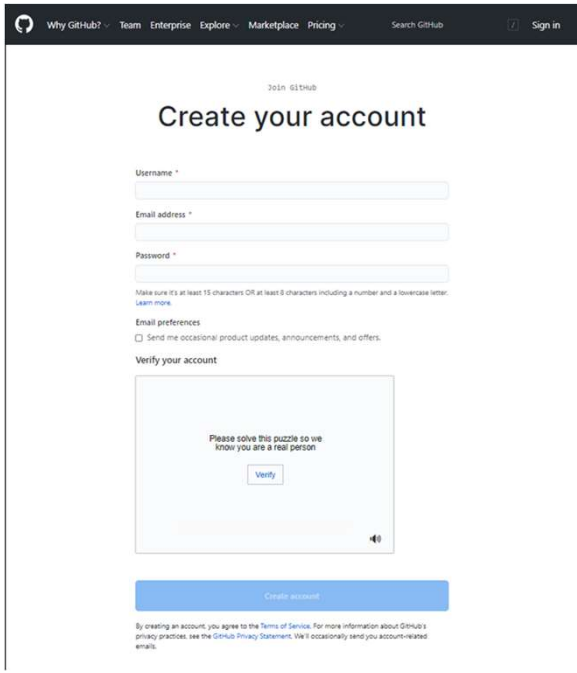
GitHub is a Ruby on Rails website application developed and launched in 2008—originally—by Scott Chacon, P. J. Hyett, Chris Wanstrath, and Tom Preston-Werner.

By the end of 2018, it was acquired by Microsoft, as their investment in open-source software and tools increase.

Make an account.

In order to make use of GitHub and its services, you'll first need to go through their sign up process and create an account:

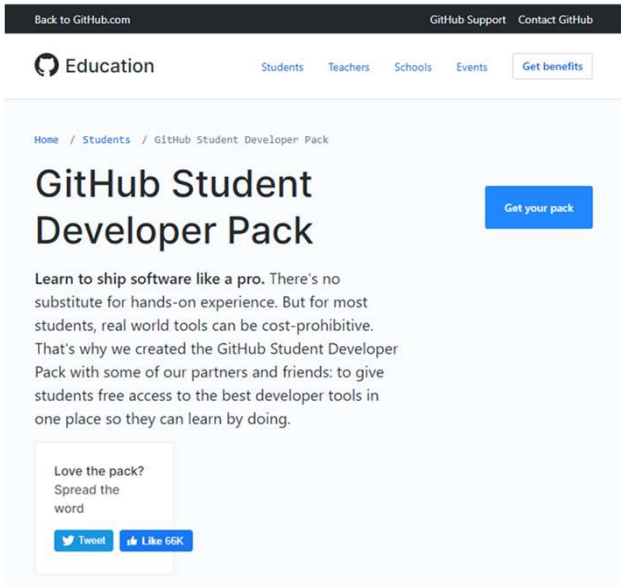
https://github.com/join



The GitHub Student Developer Pack.

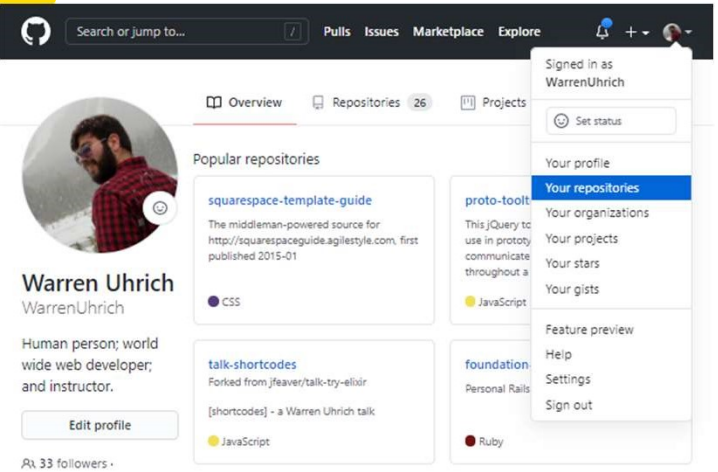
As a student, you may qualify for GitHub's Student Developer Pack. Look into upgrading your account to take advantage of premium features at no extra charge:

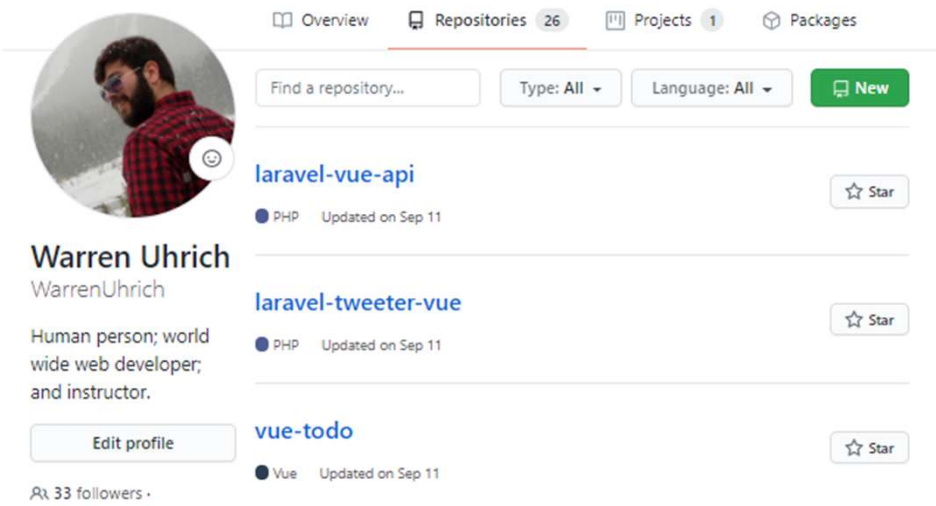
https://education.github.com/pack



Navigating to your repositories on GitHub.

Once signed in, click your profile image in the top-right corner of the page. A dropdown will appear, with an option labeled: “Your repositories.” Click this to visit a page listing your repositories. If you’ve just made a new account, it will be empty for now.





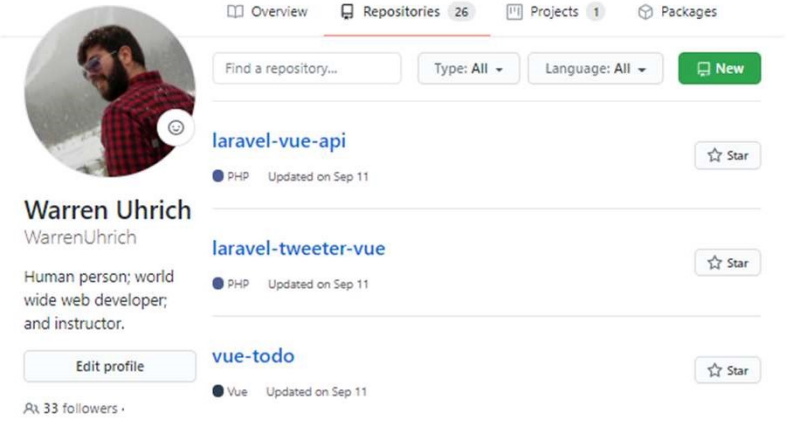
UNIVERSITY OF ALBERTA

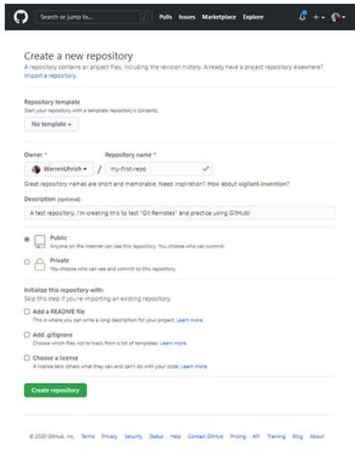
Creating space for a new Git repository on GitHub.

Click the “New” button to create a new repo.

Fill in a repo name, and a description if you’d like.

When you’re ready, click the “Create repository” button.

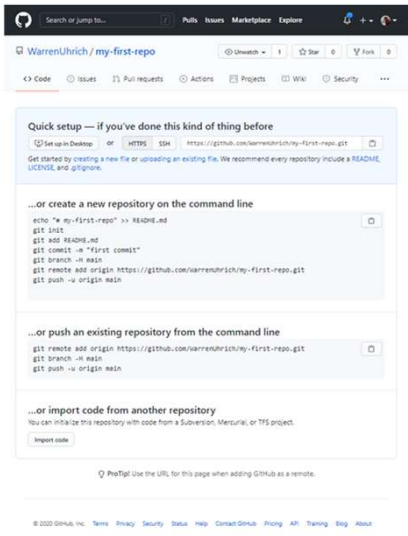




Your new GitHub repo.

You are now the proud owner of your very own GitHub repository!

Make a note of the HTTPS URL, we'll need this for linking our local repository to this remote one on GitHub.



Helpful commands.

GitHub's "New Repository" screen is helpful and includes instructions to help you:

These are commands we can use to connect a repository on our personal computer, to the remote repository on GitHub. Let's give it a try!

Get where you need to be.

Let's break down the steps. First, remember, you'll need to make sure you've navigated to the correct directory on your system before running any Git commands.

If you're following along from the "Git" slides, you'd get there by opening the Git Bash terminal and entering the following…

```
cd ~/
cd projects
cd my-first-git-project
```

UNIVERSITY OF ALBERTA

Ensure you're in the correct folder.

You can enter "pwd" into your terminal, and it will print the path where you are.

You can also try "git status". If you are not in an existing repository folder, it will let you know that Git has not been initialized in your current location.

If you find yourself in the wrong place, you may have to navigate around a bit to get to your local repo!

Adding a remote.

In order for you to easily copy your local repository files and changes over to GitHub, we will set up the GitHub repo location as a Git "remote."

git remote add origin https://YOUR-GITHUB.URL/REPO.git

For this repo, it will now treat the name "origin" as the URL you have specified. This will be important when you get to the command that copies your repo over to that site—so that you don't have to type out the whole URL every time!

UNIVERSITY OF ALBERTA

About remotes.

git remote add NAME URL

You can have more than one remote on a repo, if you'd like. We called this remote "origin" in the last slide, but the name you use to represent a remote could be anything you'd like.

Avoid spaces or other special characters in the name.

The name you choose will represent the URL that follows during this "add" command.

Use the main branch.

The Git branch we use to represent the authoritative version of our project is most-often called “main”.

Follow this convention by renaming your current branch to “main” via the following:

```
git branch -M main
```

Push the repo to your remote.

Now that we have that set-up out of the way, it is time to push our changes to GitHub from our local repository…

git push -u origin main

In this command, “u” means “upstream.” This sets the remote and branch that follow as the default for this repo.

Next is the name of your remote—in our case we called it “origin.”

And finally is the branch name we chose: “main”.

Signing in.

When you try to push, GitHub will need to validate that you are the correct user to be updating this repo.

This is a security feature to ensure that no one else can overwrite your project, rename things, or delete things!

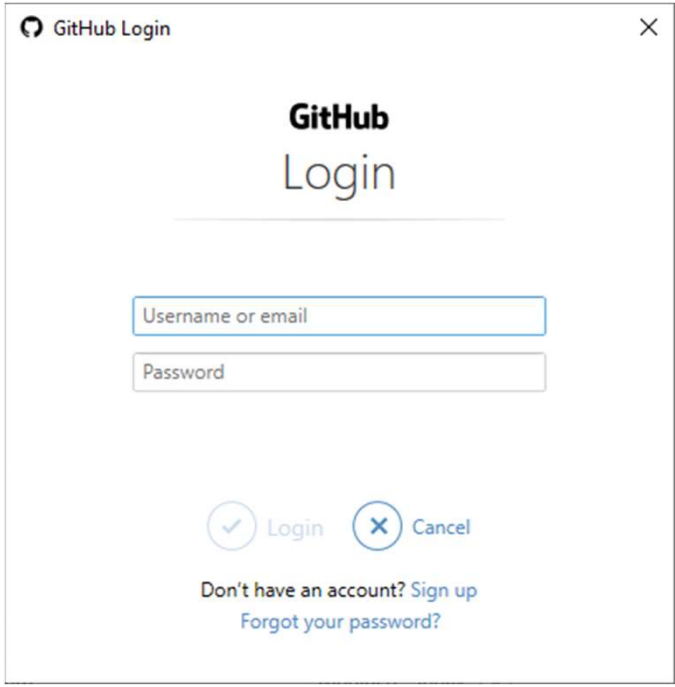
You'll either have a prompt in the terminal, asking for your GitHub username and password, or a pop-up representation of the same.

Fill in whichever prompt appears, and your push should go through.

Authentication prompt.

Once you're signed in, you should see "push" status updates in your terminal.

```
$ git remote add origin https://github.com/WarrenUhrich/my-first-repo.git
$ git branch -M main
$ git push origin main
```

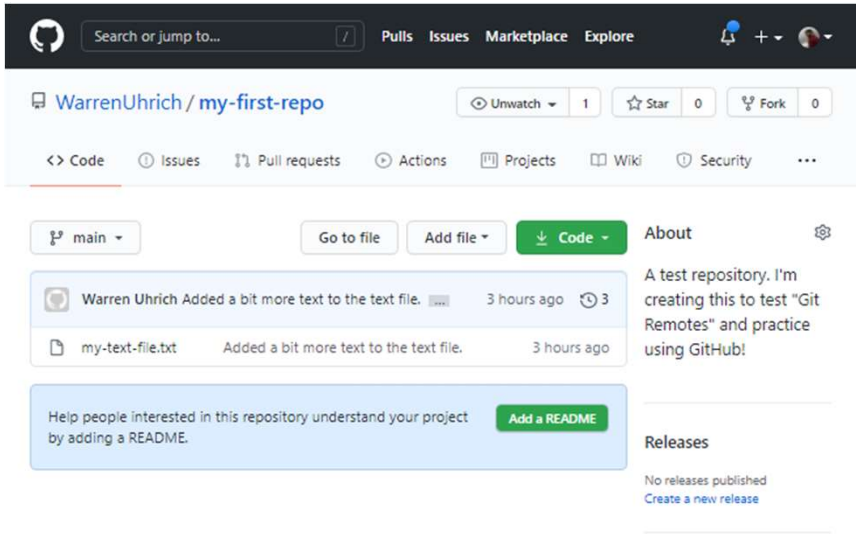


Checking your remote repository.

If the push was successful, refresh the repository webpage in your web browser.

It will now be populated with your files and commits!

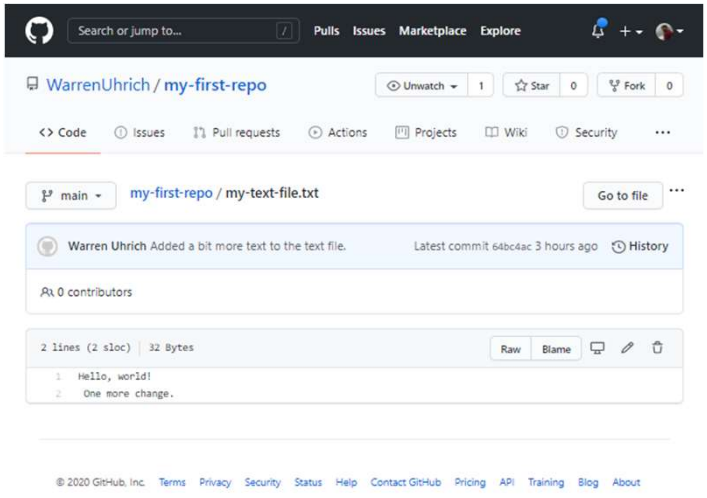
```
$ git push origin main
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (9/9), 737 bytes | 368.00 KiB/s, done.
Total 9 (delta 0), reused 0 (delta 0)
To https://github.com/WarrenUhrich/my-first-repo.git
 * [new branch]      main -> main
```



Explore.

You can view your commit history, individual files, different branches, and much much more on GitHub.

Have a look around and get comfortable with the site!



Working on your repository.

If you're working with remotes, especially as a team, you will start incorporating them into your local process and steps.

When making changes, ensure you keep the following in mind.

The Git workflow.

Remember our original Git workflow?

1. git add --all
2. git commit -m “Your message.”

When working with remotes like GitHub, we’ll be adding one additional step to your process.

The GitHub workflow.

Make commits often, and ensure your message is descriptive. The new workflow ends up looking like so…

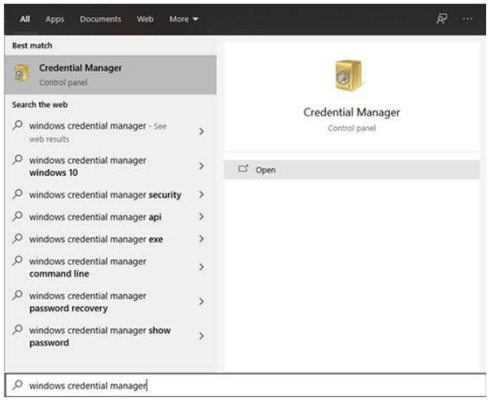
1. git add --all
2. git commit -m “Your message.”
3. git push origin main

The third step will update the remote with your local changes!

Managing Windows credentials.

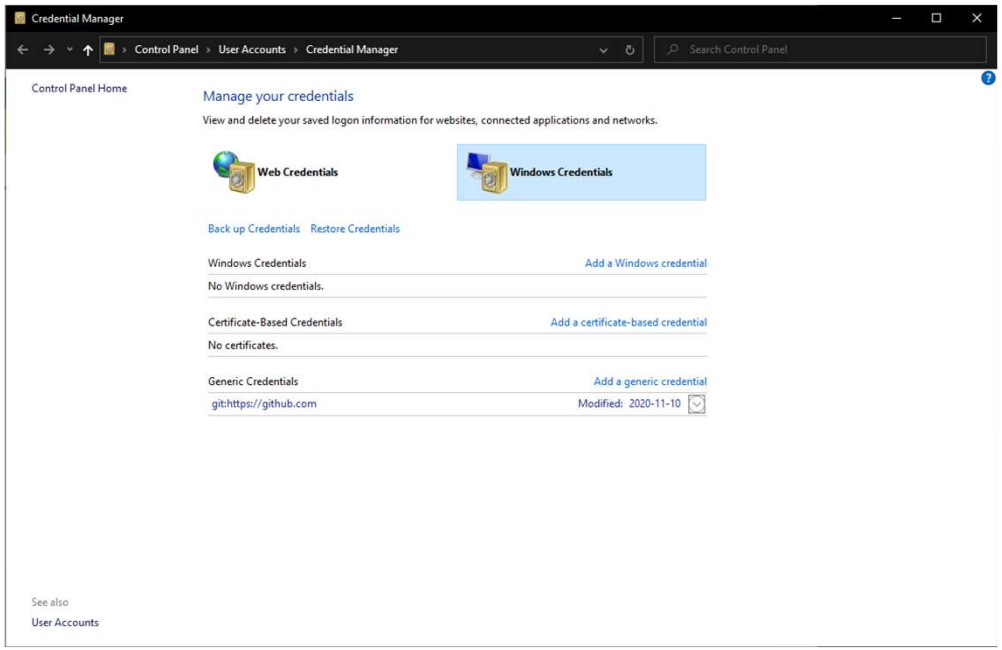
If you ever need to update or remove Windows credentials, like those you entered via your terminal, you can manage them via the Credential Manager.

Windows key + S will open Window Search. Enter “Credential Manager” and open the application.



Managing Windows credentials.

Here you'll find your Git credentials. You can "Edit" or "Remove" the credentials as needed.

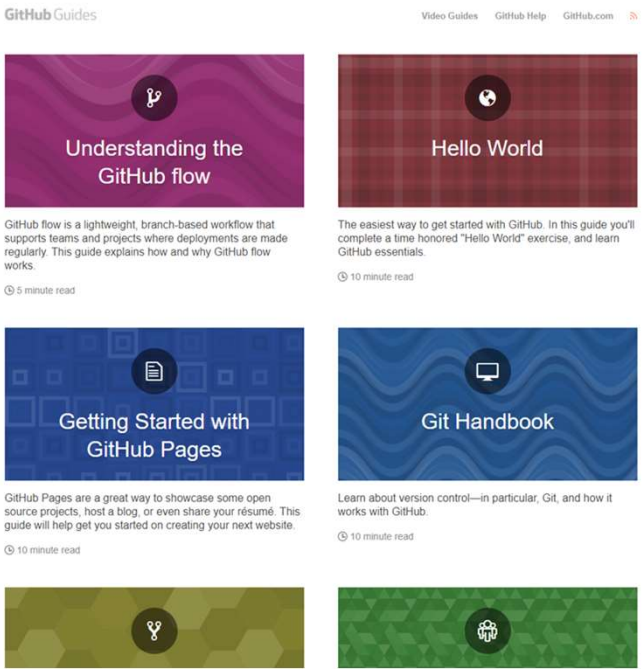


Read more.

GitHub offers comprehensive documentation, tutorials, and guides on the features they provide as well as the Git version control system.

Check them out!

https://guides.github.com/



Recommended Reading

If reading through a book is more your style, try out the following:

- Loeliger, J., McCullough, M. (August 2012). *Version Control with Git, 2nd Edition.* O'Reilly Media, Inc.
- Tsitoara, M. (November 2019). *Beginning Git and GitHub: A Comprehensive Guide to Version Control, Project Management, and Teamwork for the New Developer.* Apress.

To learn about writing markdown (used for READMEs), try this book:

- Mailund, T. (August 2019). *Introducing Markdown and Pandoc: Using Markup Language and Document Converter.* Apress.


UNIVERSITY OF
ALBERTA