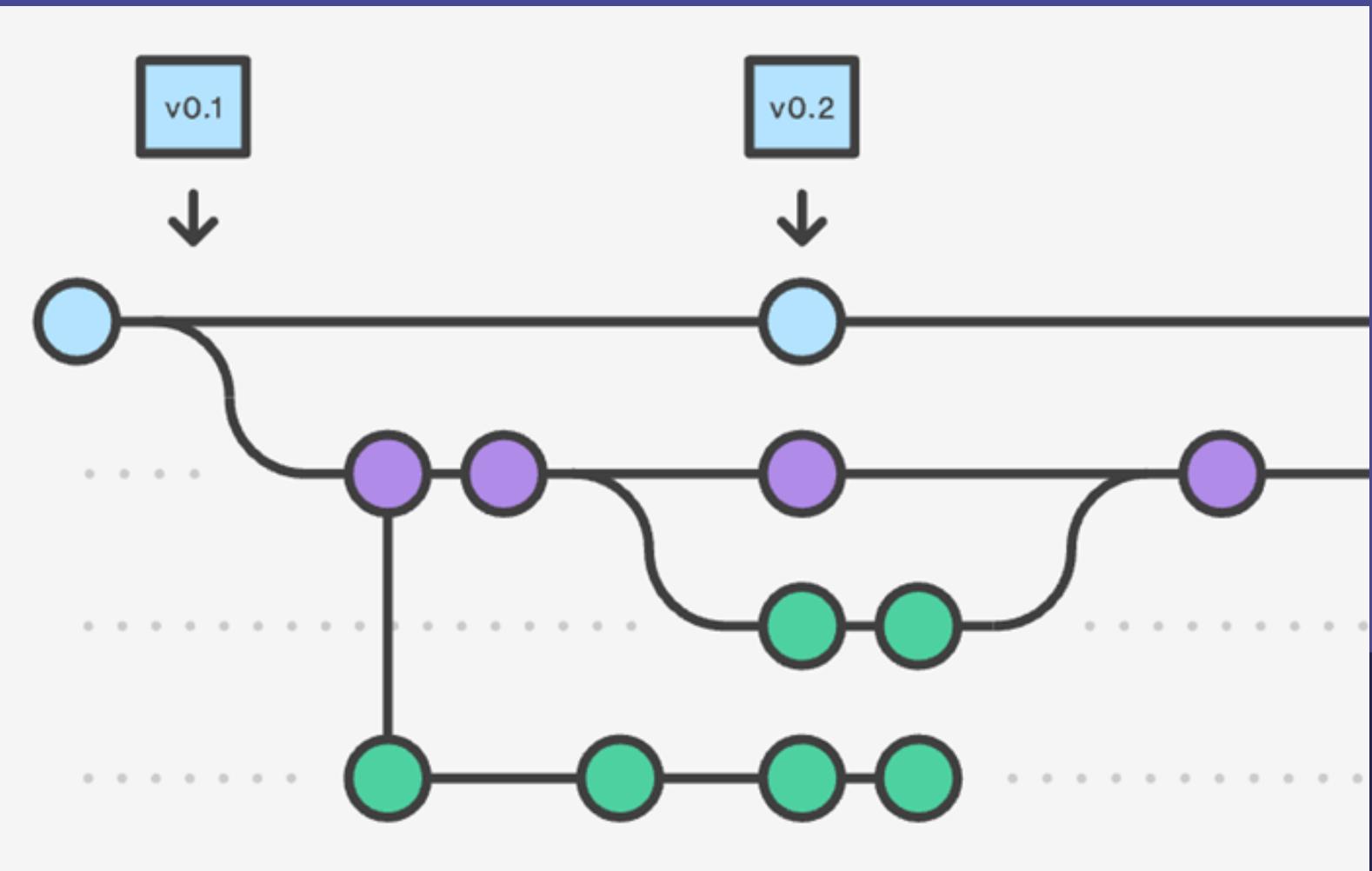


# What is Git?



# Source Control

- Source control, or version control, is a way of tracking your files progress over time.
- It is usually saved in a series of snapshots and branches, which you can move back and forth between.



...

# What you can do

Source control allows you to:

- Distribute your file changes over time
- Prevent against data loss/damage by creating backup snapshots
- Manage complex project structures



# About Git

- Git is a source control software, similar to many others out there.
- It follows all the same rules and concepts that any source control follows.

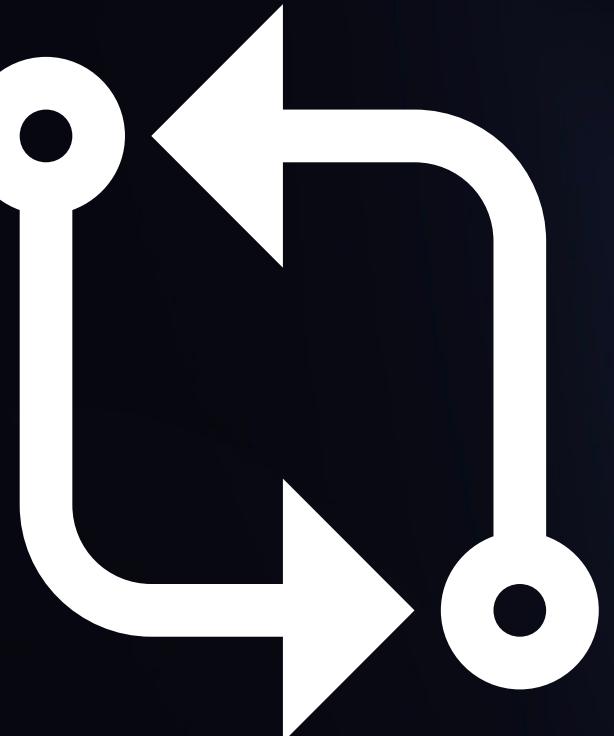


# Why use Git?

- The most popular source control software in the world
- Lots of documentation and support
- Lots of integration with other applications (SourceTree, Heroku, GitHub)



...



# Summary

- Git is a source/version control software.
- It is the most popular source control software in the world.
- It has lots of integration with other programs, giving it lots of value and demand.



# Git vs GitHub



# What GitHub Is

- GitHub is an application allowing you to store remote repositories. (it's in the name!)
- You can interact with your GitHub repository through the push/pull system on your local machine.
- GitHub is used primarily to allow other people to add to the project (ex. Open source projects)
- GitHub allows more people than just yourself to see and interact with the repository.

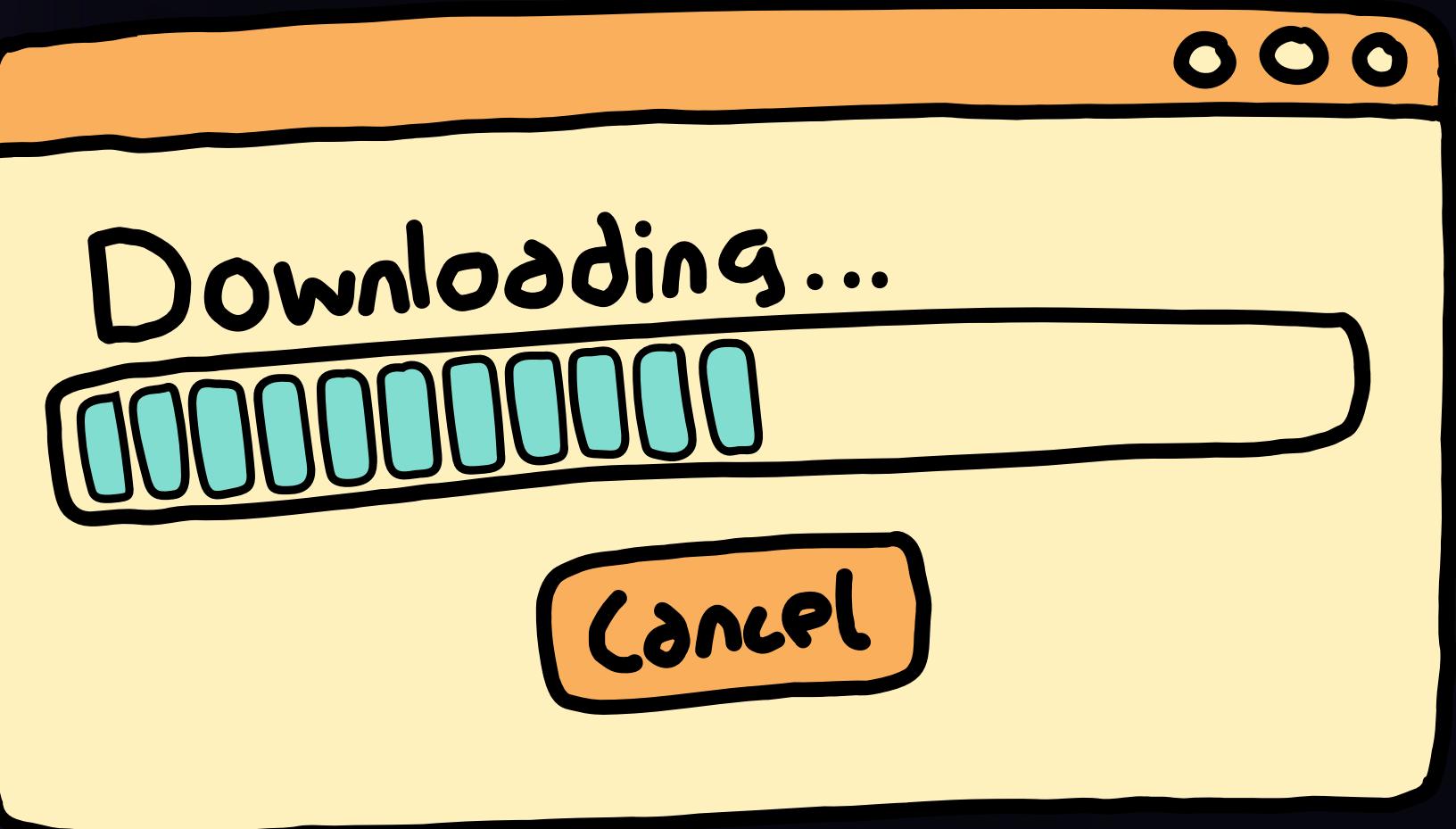


# The Difference

- Git is a source control software allowing you to take snapshots and distribute your creations & modifications over time.
- GitHub is an application allowing you to store and interact with your repository on a remote server, as well as adding more features (eg. Publicity, licensing, collaborators)
- Git is the bones and flesh of source control, while GitHub gives you the platform to work with your repository easier.



# Installing Git



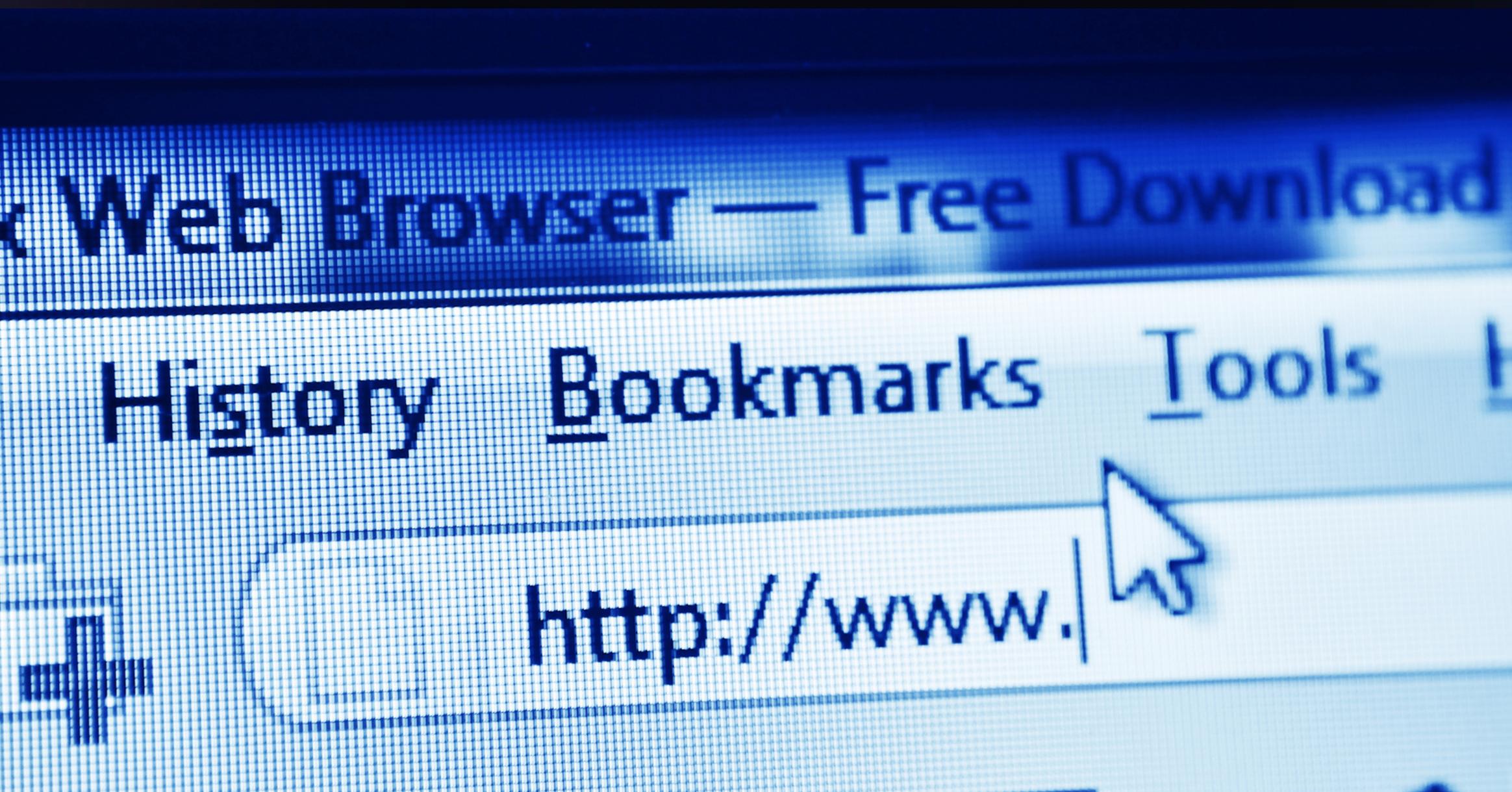
# Installing Git

- open your web browser



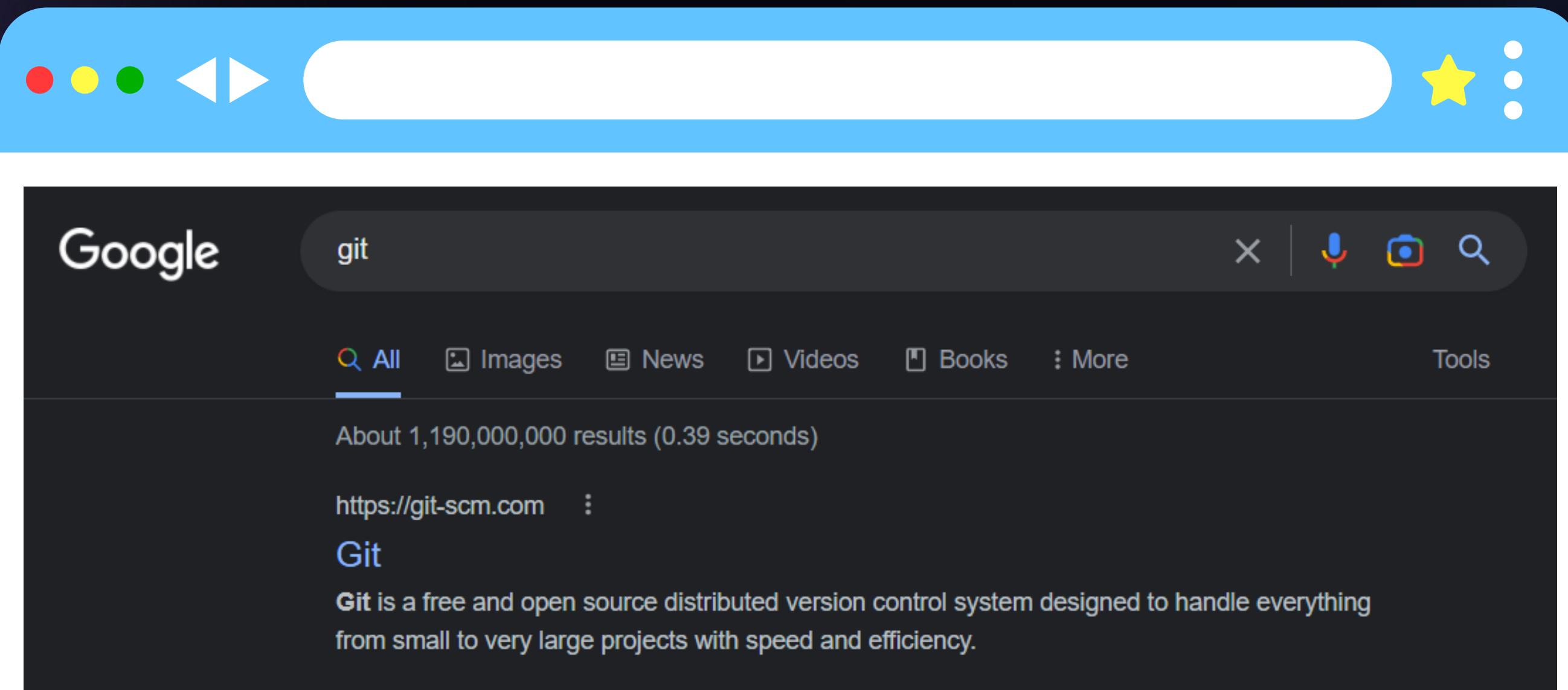
# Installing Git

- search git



# Installing Git

- go to the first search result



# Installing Git

- now go to the inside of that web site <https://git-scm.com/>

The screenshot shows the homepage of the Git website (<https://git-scm.com/>). At the top, there are three colored circles (red, orange, green) on the left and a search bar on the right. The main header features the Git logo (a red diamond with a white 'g') and the text "git --fast-version-control". Below the header, a brief introduction states: "Git is a **free and open source** distributed version control system designed to handle everything from small to very large projects with speed and efficiency." A second paragraph highlights Git's performance: "Git is **easy to learn** and has a **tiny footprint with lightning fast performance**. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like **cheap local branching**, **convenient staging areas**, and **multiple workflows**." To the right of the text is a diagram illustrating a distributed network of seven white server stacks connected by red and yellow lines on a grid background.

# Installing Git

- now scroll down and click on download button

The screenshot shows the official Git website's 'Downloads' section. It features four main links: 'About' (with a gear icon), 'Documentation' (with a book icon), 'Downloads' (with a download arrow icon), and 'Community' (with a speech bubble icon). Below these is a section for the 'Pro Git' book. To the right, a computer monitor displays a teal box with the text 'Latest source Release 2.39.2' and a link to 'Release Notes (2023-02-06)'. At the bottom, there are links for 'Windows GUIs', 'Tarballs', 'Mac Build', and 'Source Code'.

**About**  
The advantages of Git compared to other source control systems.

**Documentation**  
Command reference pages, Pro Git book content, videos and other material.

**Downloads**  
GUI clients and binary releases for all major platforms.

**Community**  
Get involved! Bug reporting, mailing list, chat, development and more.

**Pro Git** by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

**Latest source Release**  
**2.39.2**  
[Release Notes \(2023-02-06\)](#)

[Download for Windows](#)

[Windows GUIs](#) [Tarballs](#)  
[Mac Build](#) [Source Code](#)

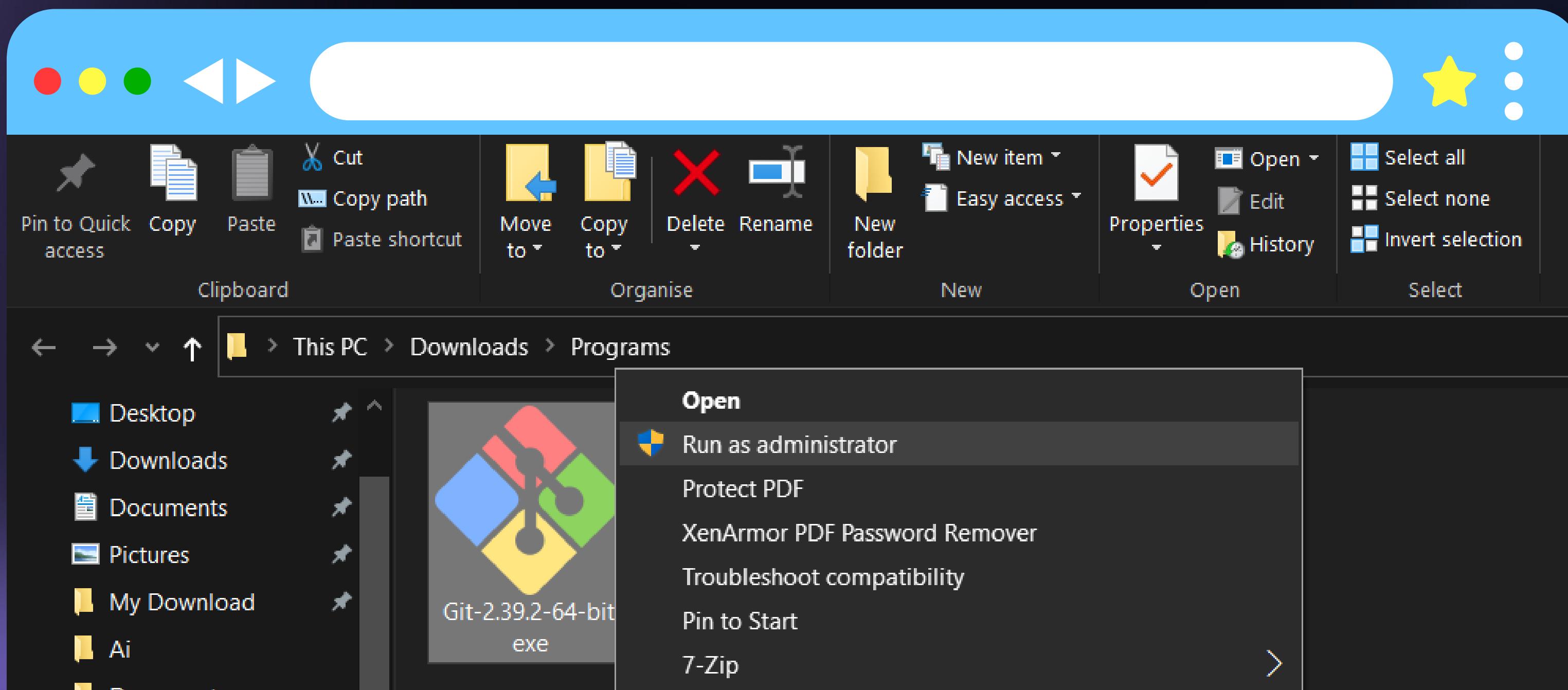
# Installing Git

- select your OS and download git setup of your PC



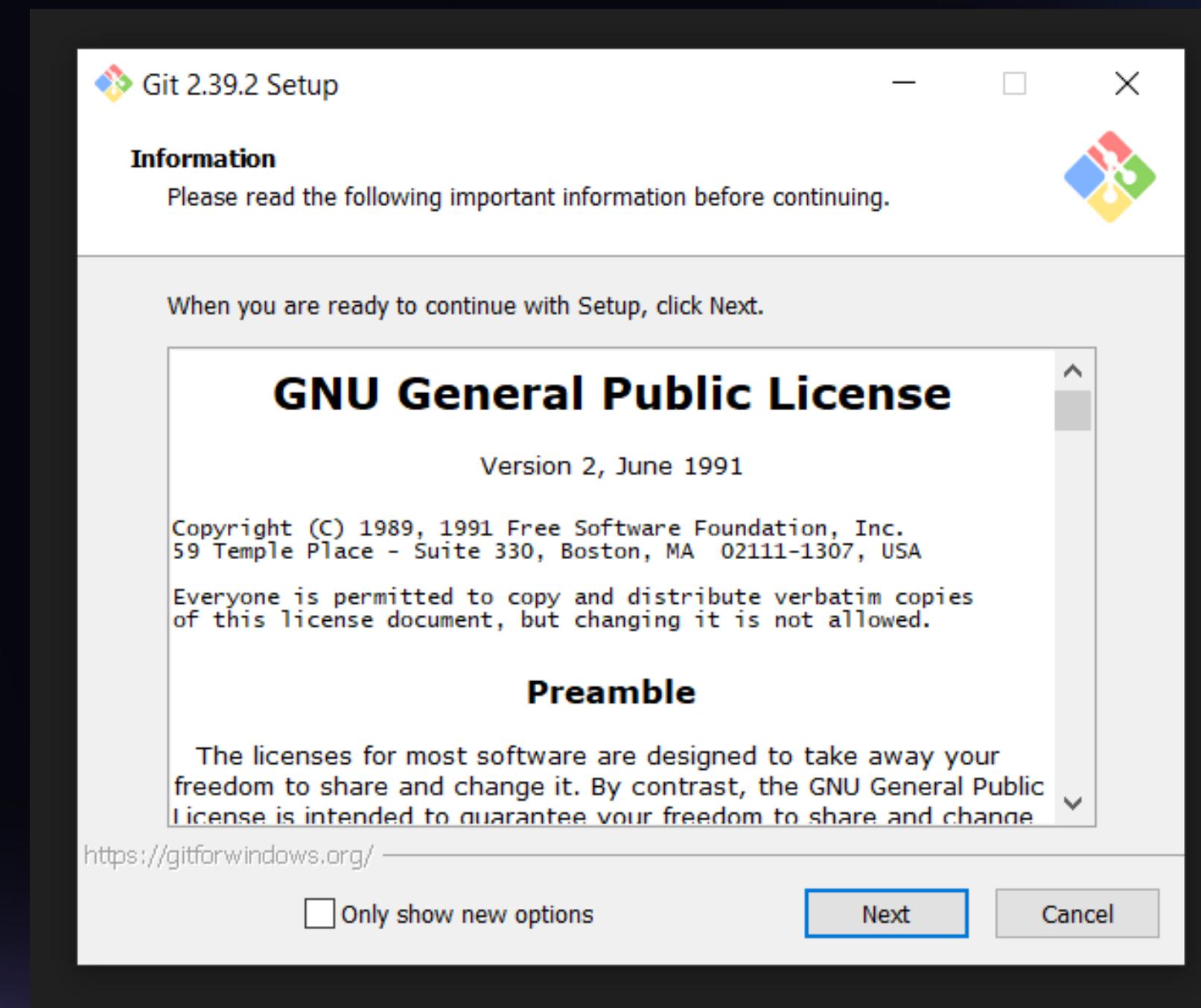
# Installing Git

- after completed the download run setup as Administrator



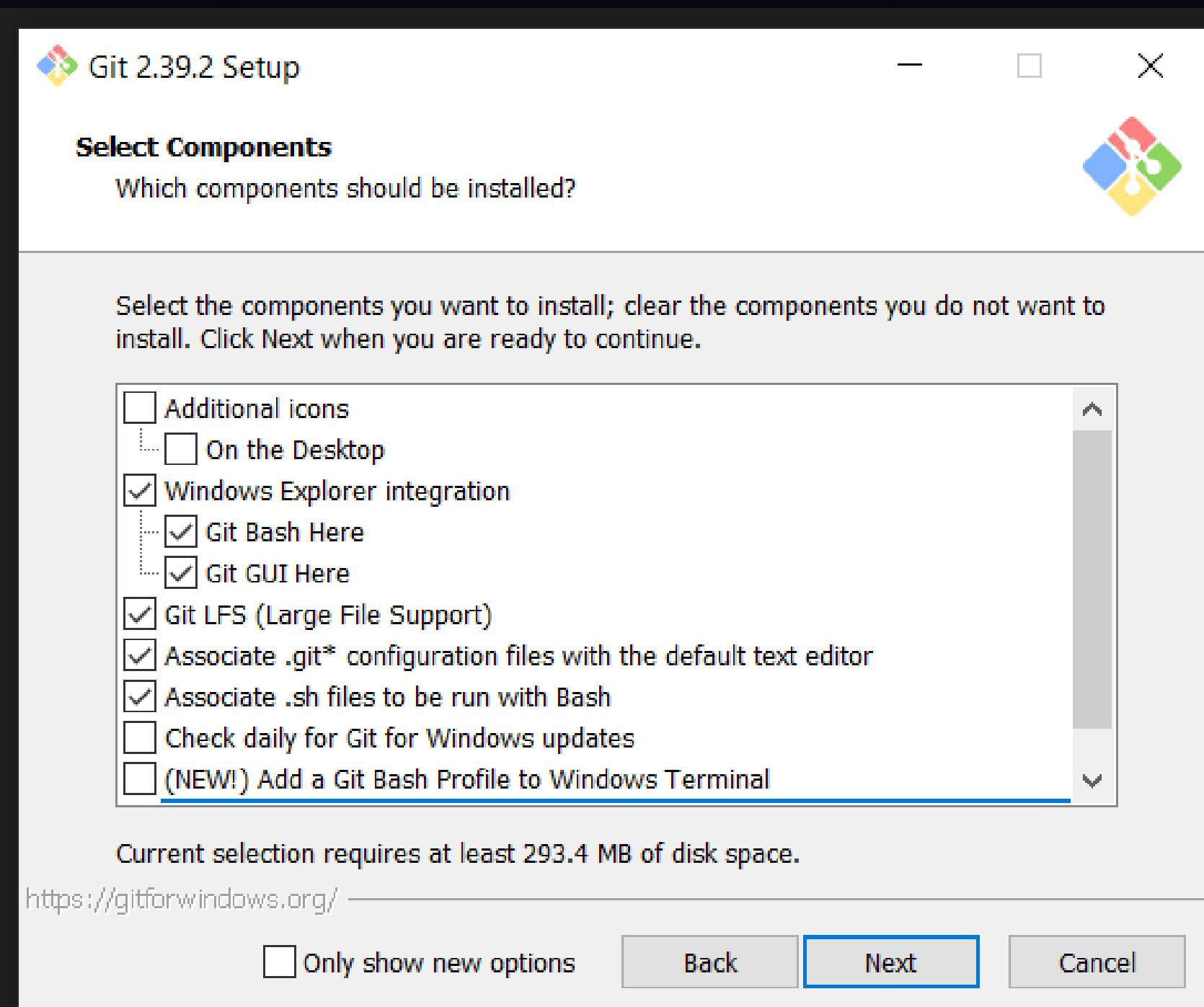
# Installing Git

- in first window you can see the GNU General Public License details, click next button



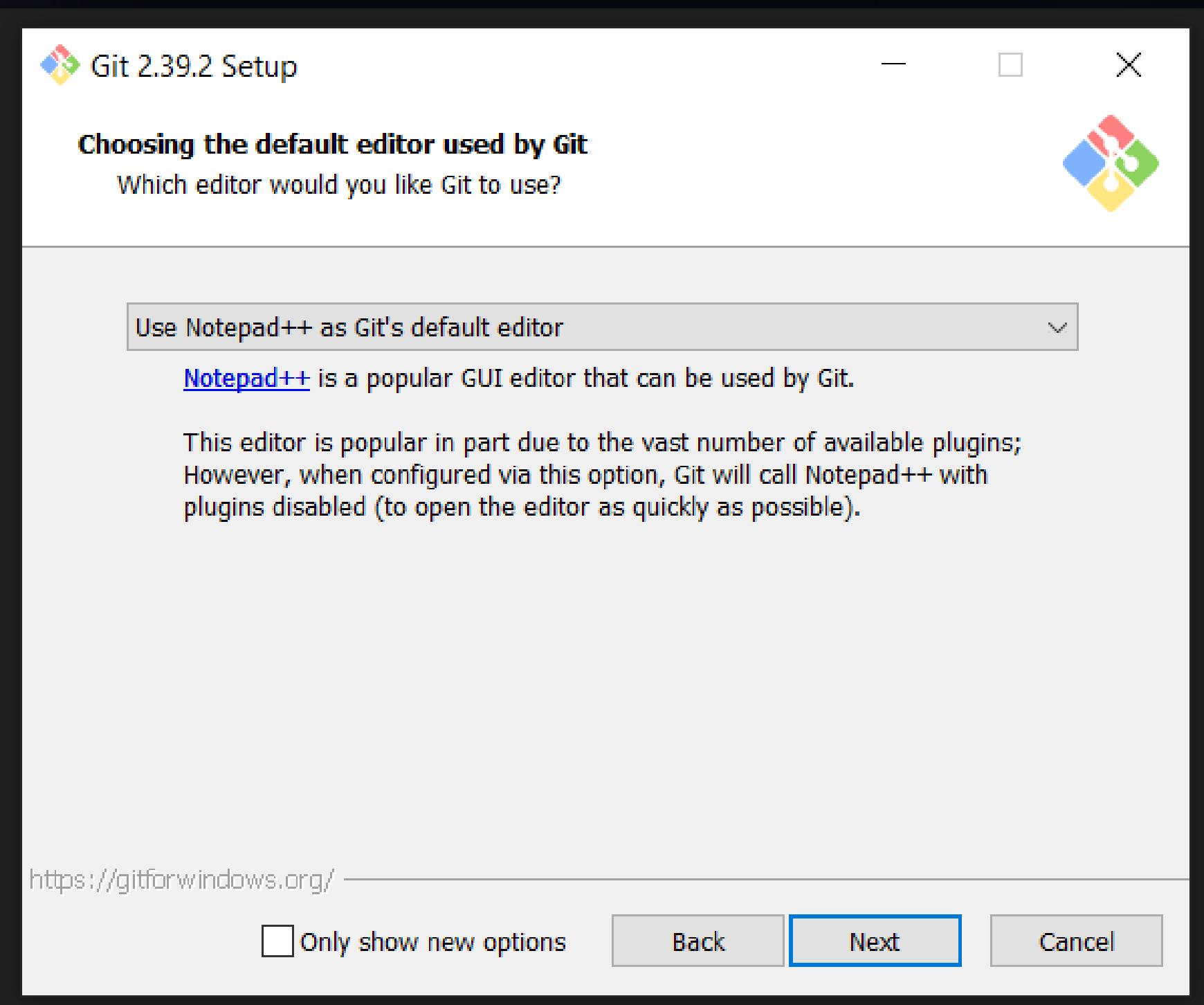
# Installing Git

- select components what should be need to installed



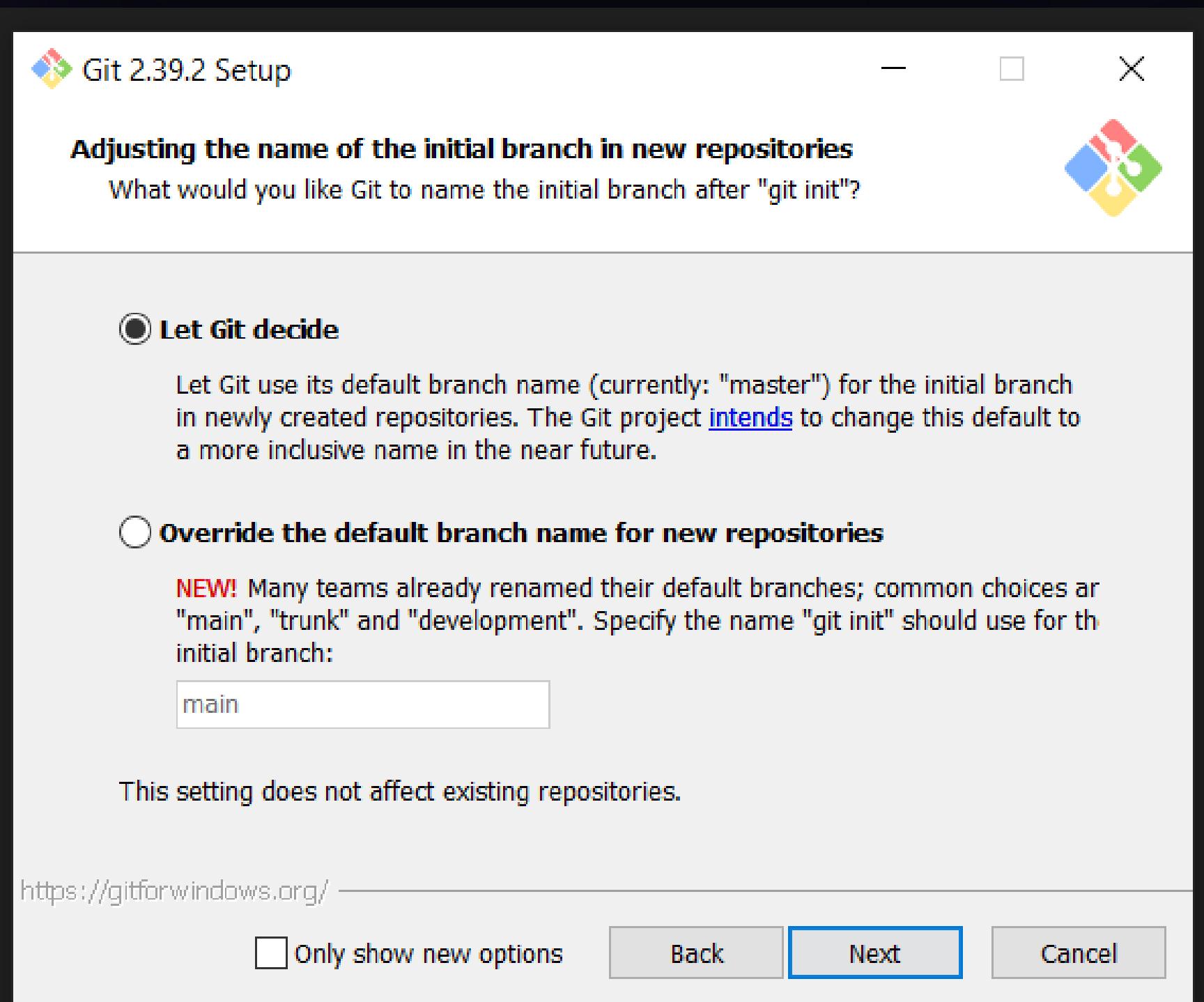
# Installing Git

- select default editor for GIT



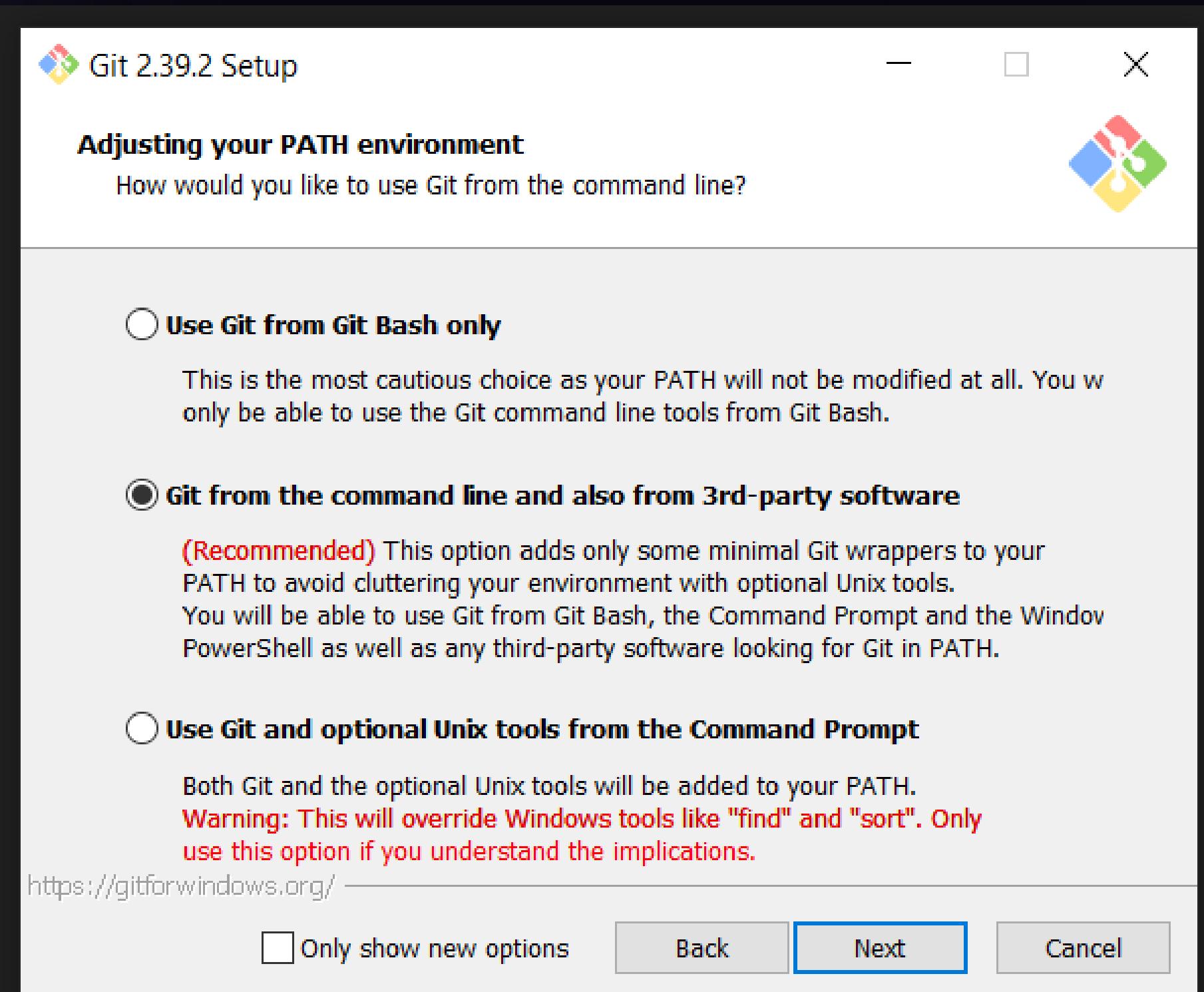
# Installing Git

- No need to change simply click next



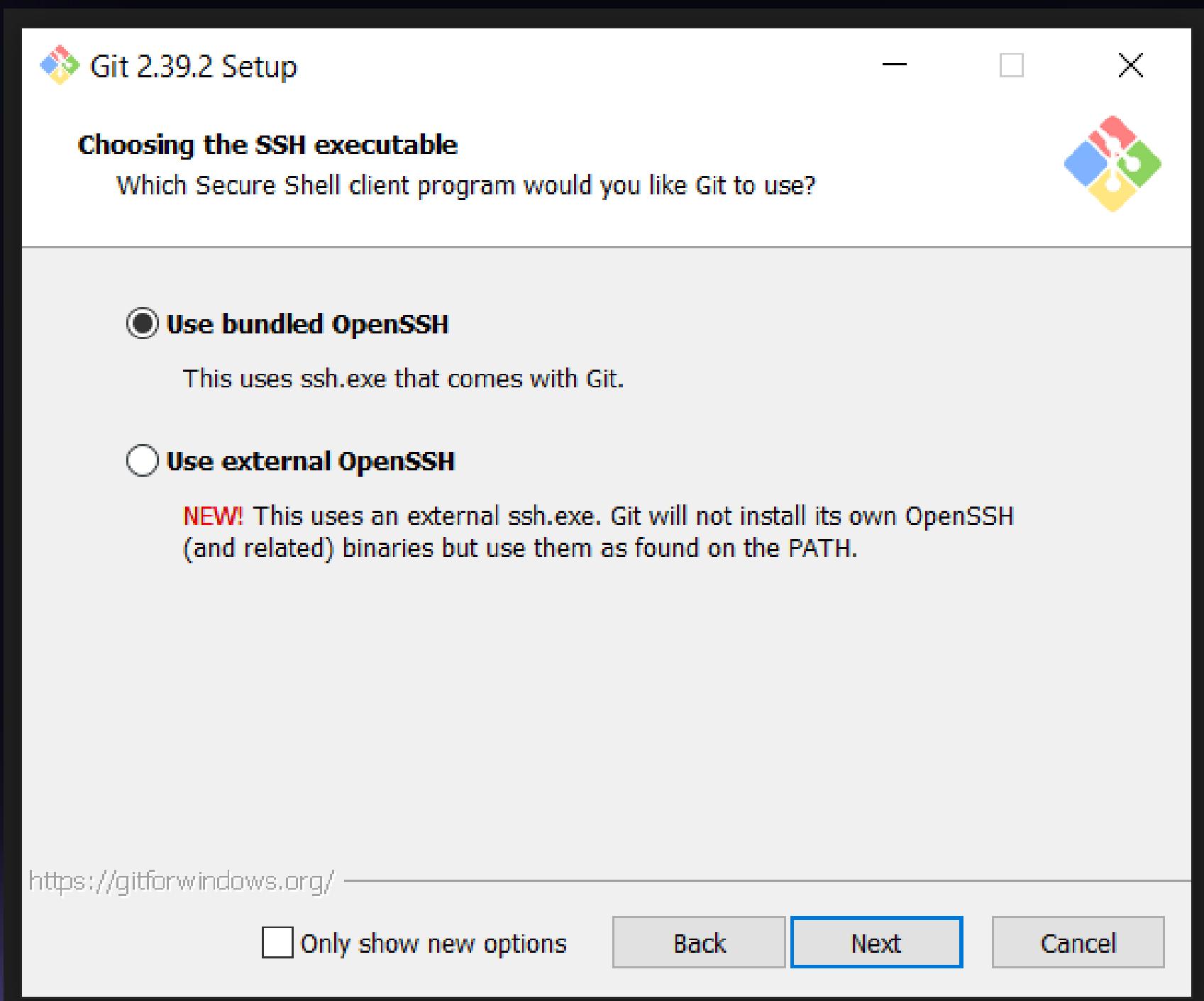
# Installing Git

- No need to change simply click next



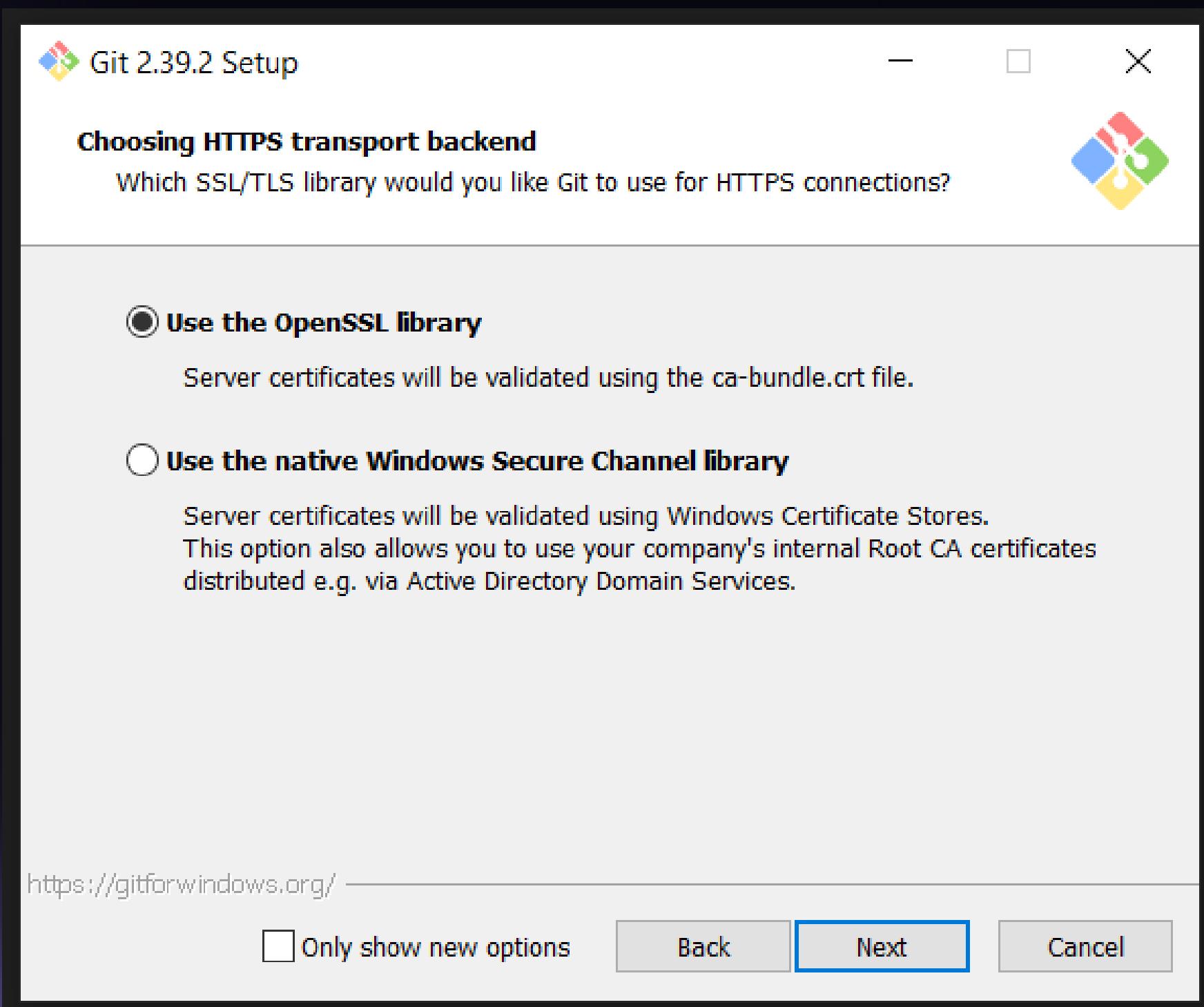
# Installing Git

- No need to change simply click next



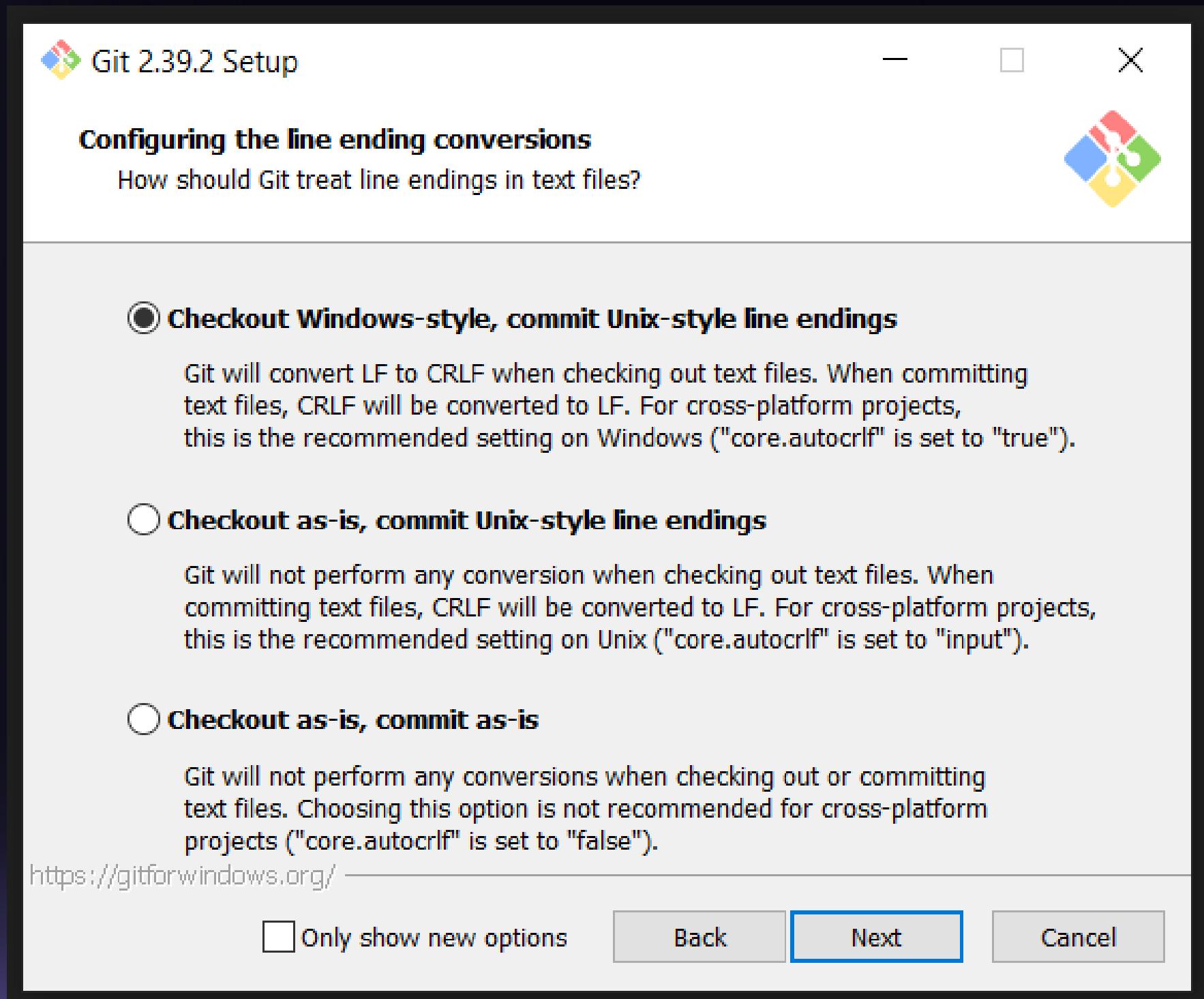
# Installing Git

- No need to change simply click next



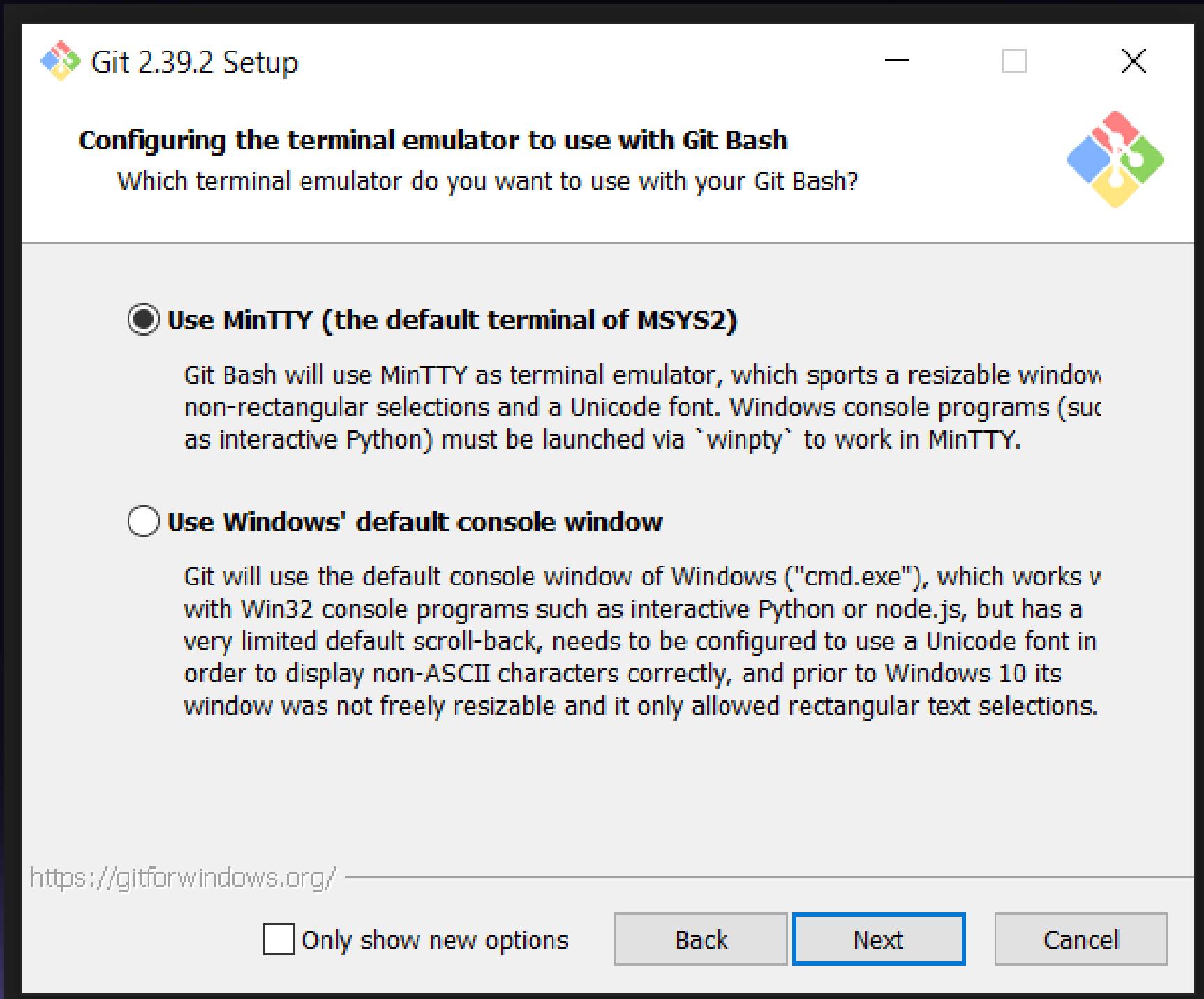
# Installing Git

- No need to change simply click next



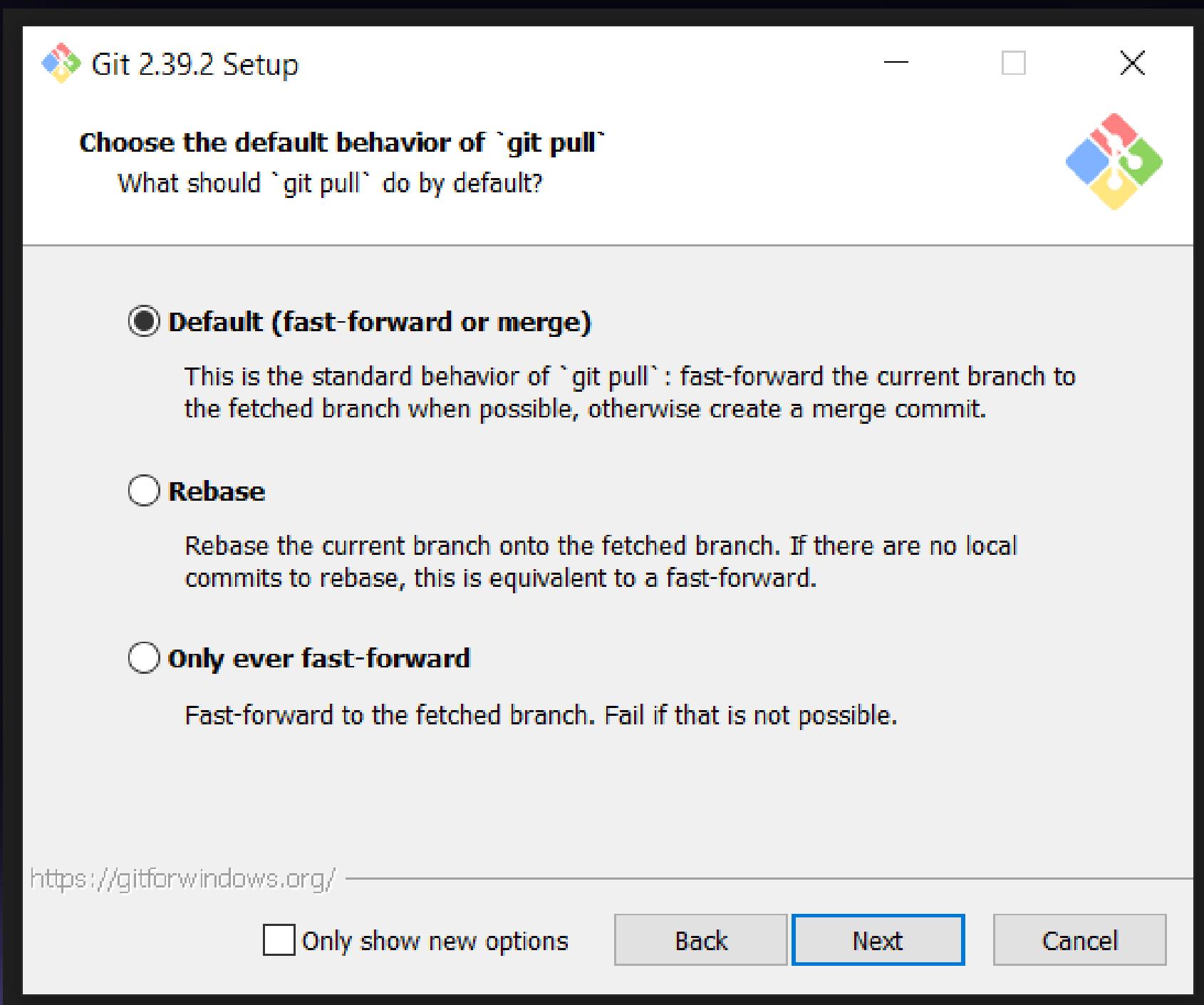
# Installing Git

- No need to change simply click next



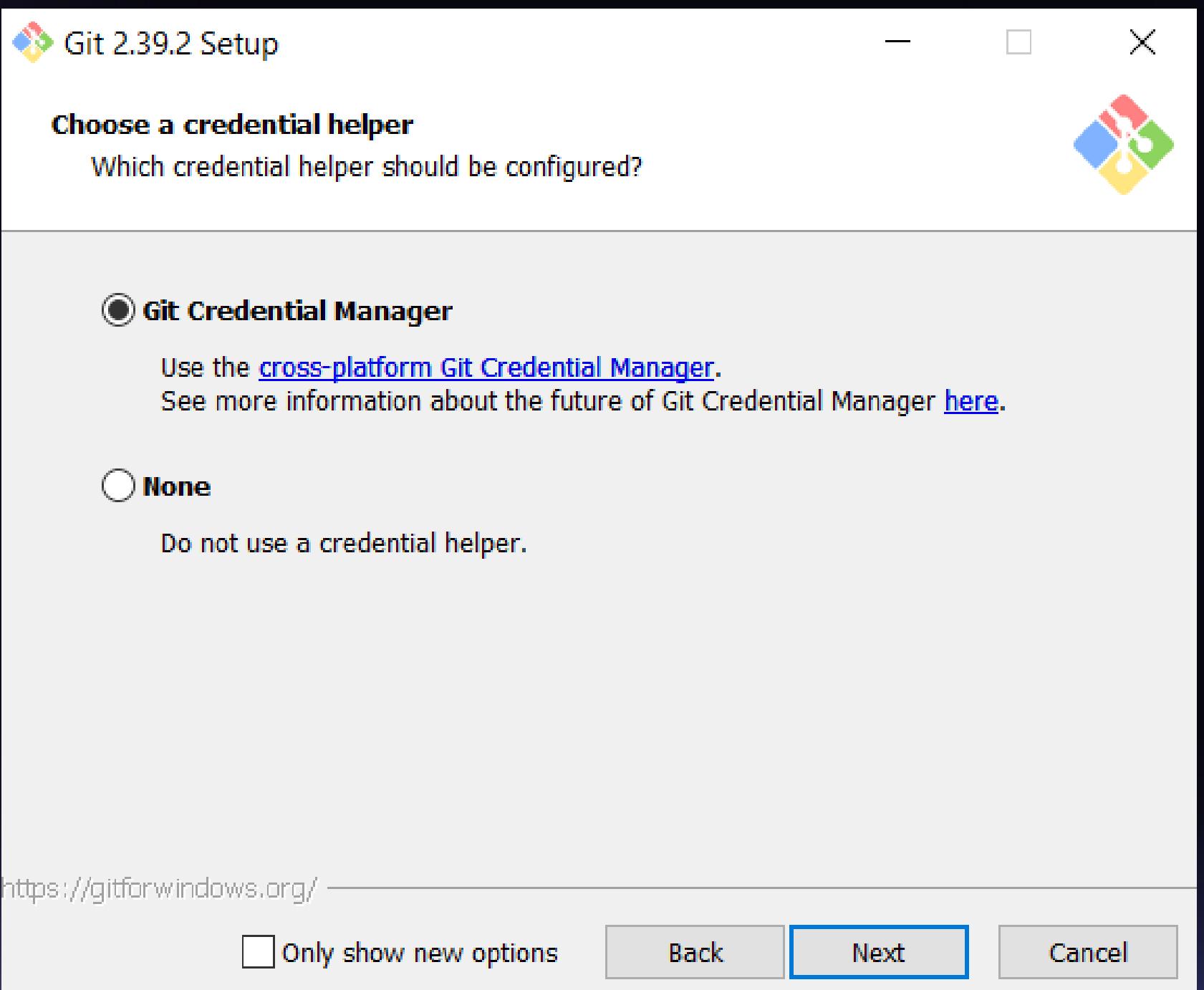
# Installing Git

- No need to change simply click next



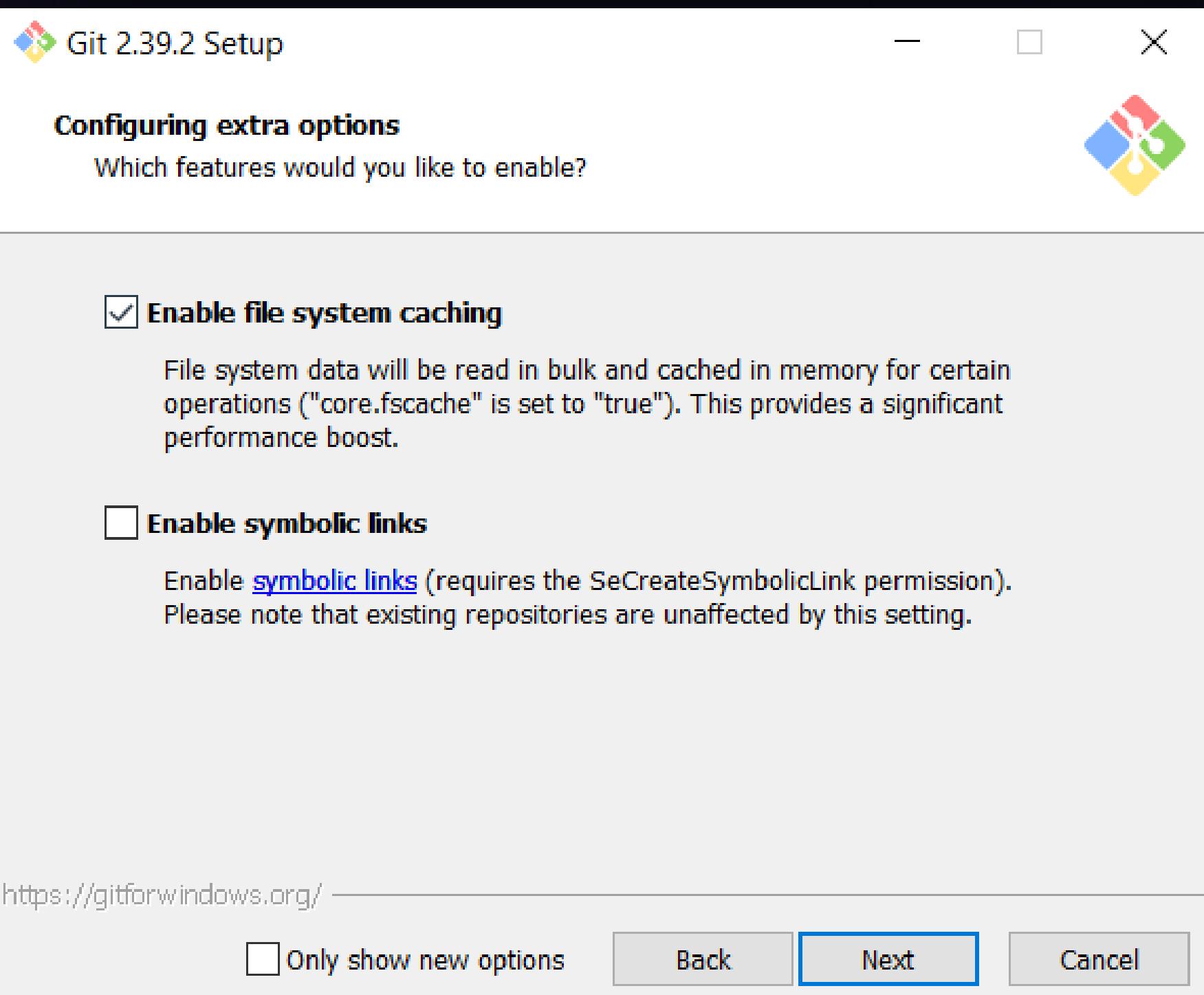
# Installing Git

- No need to change simply click next



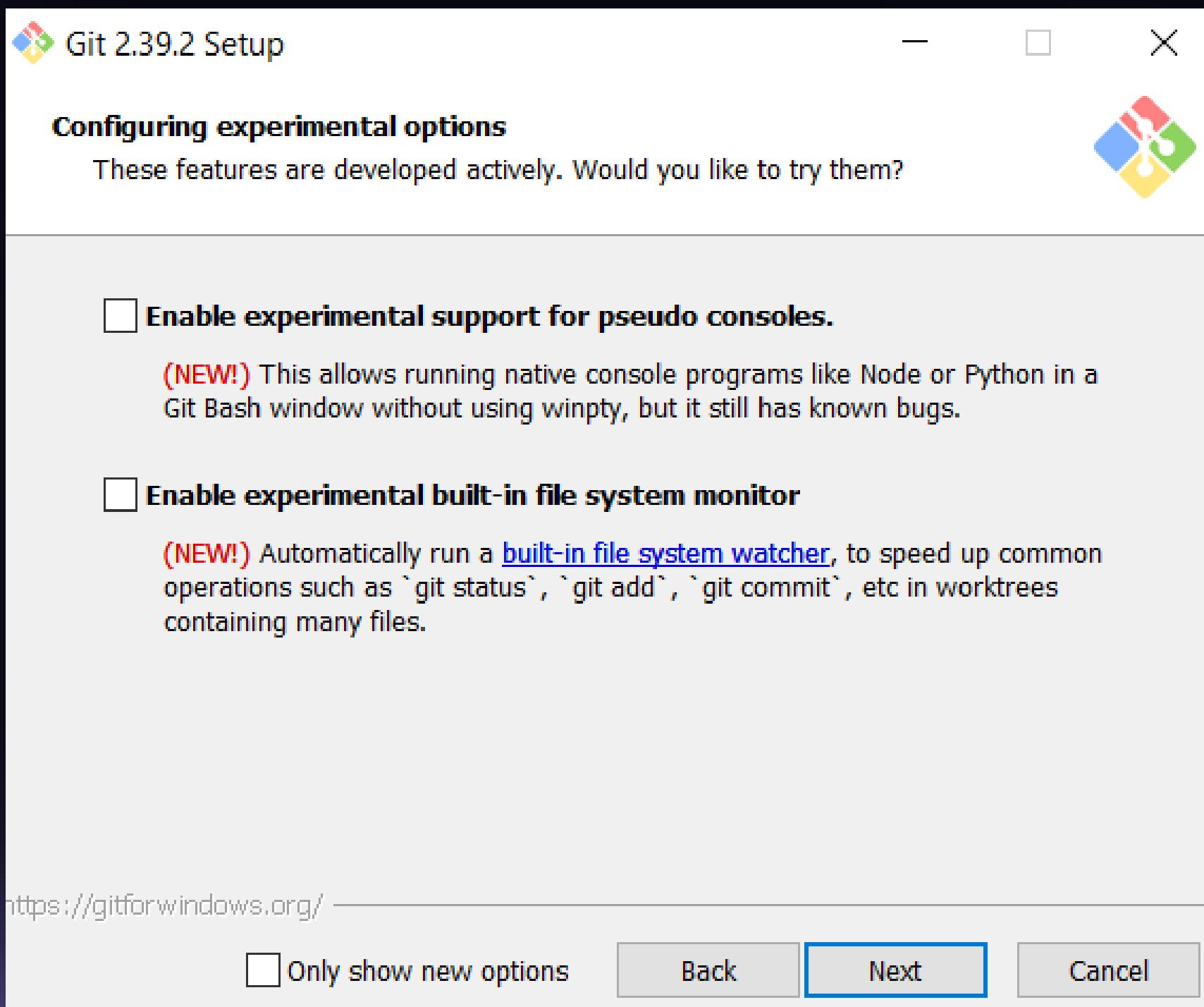
# Installing Git

- No need to change simply click next



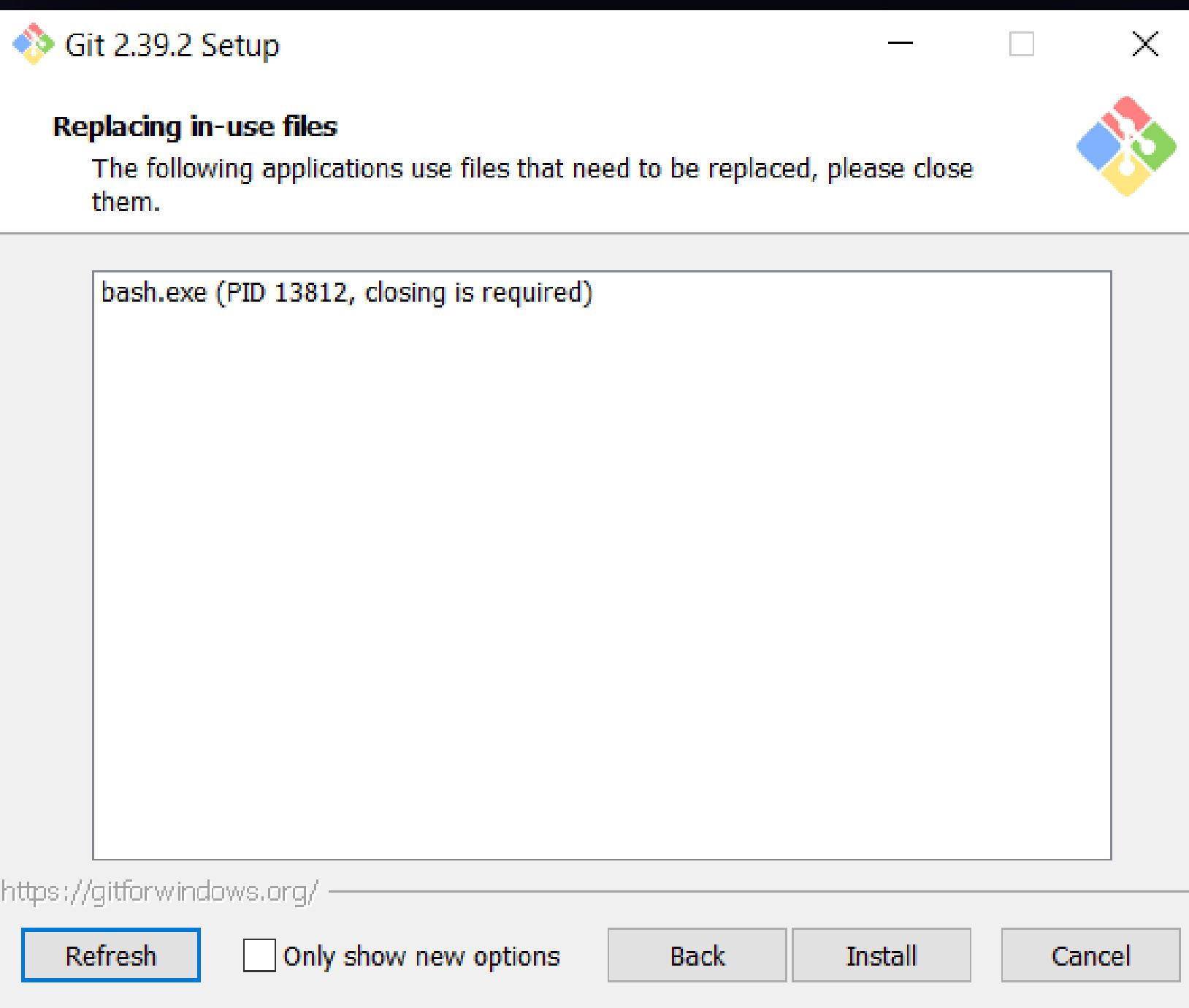
# Installing Git

- No need to change simply click next



# Installing Git

- Now click Install

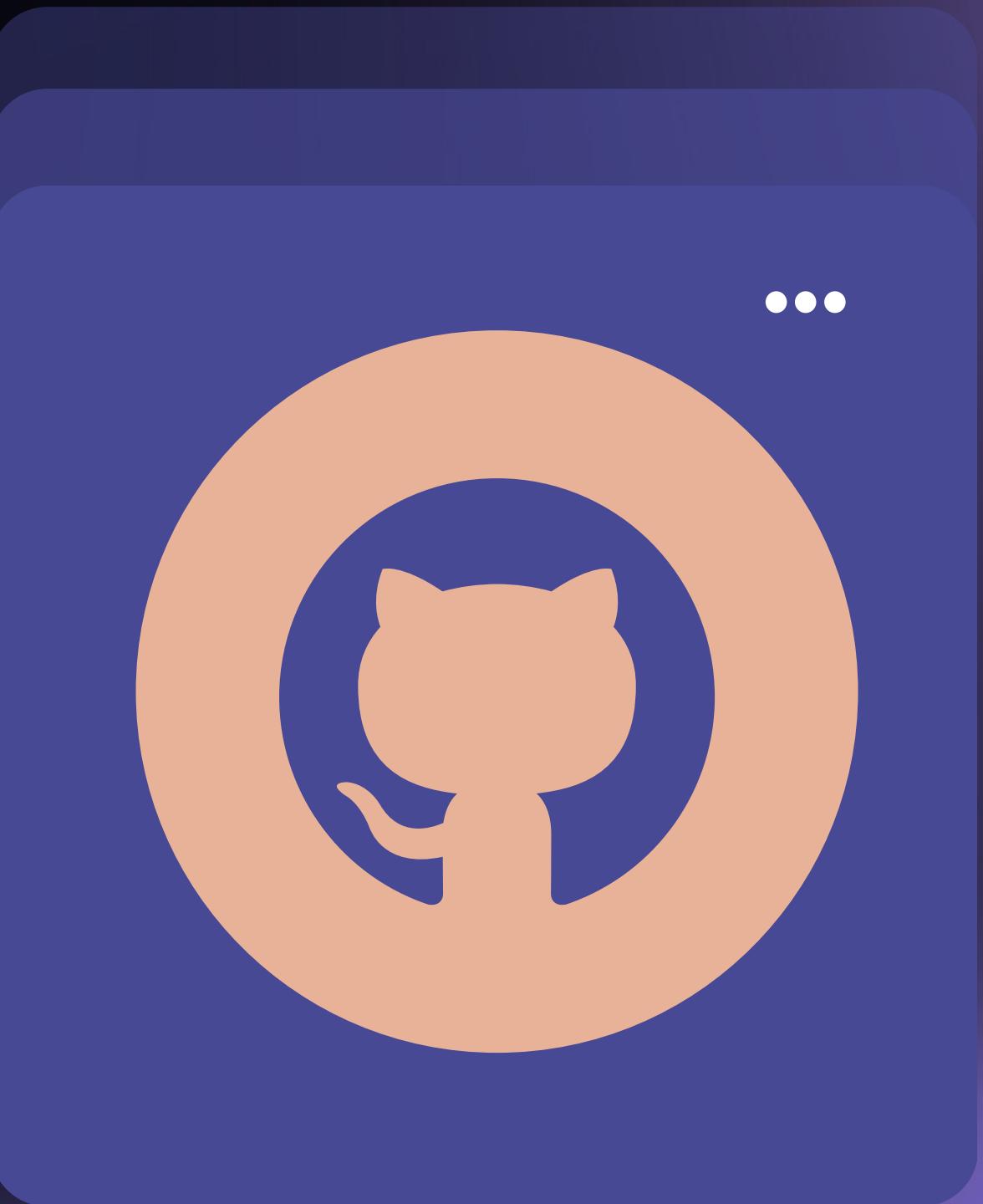


**CONGRATULATIONS**



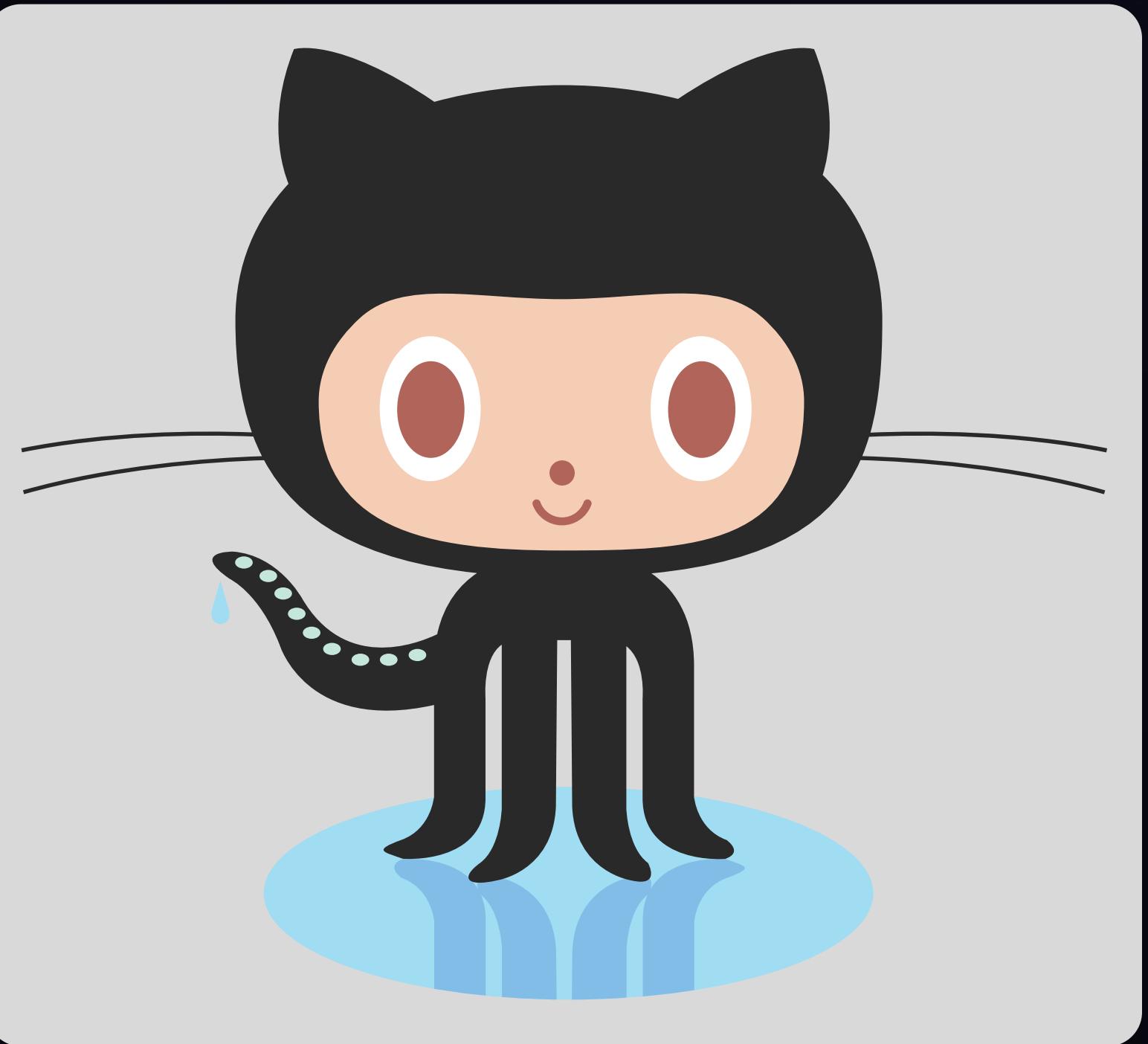
You have successfully Installed Git

# What is GitHub?



# Introduction

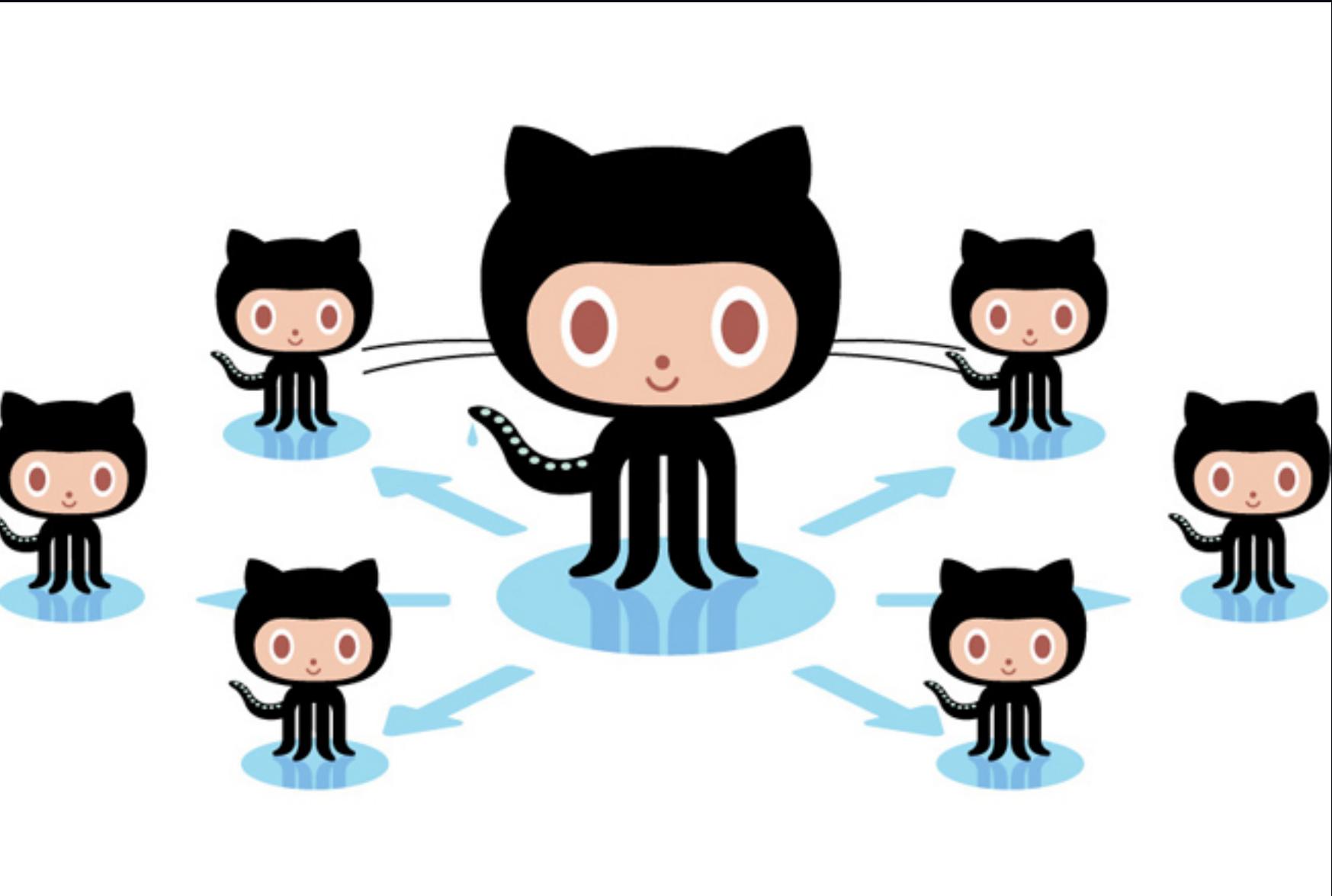
- GitHub is an application that allows you to store remote repositories on their servers.
- They also provide a user-friendly platform to interact with & manage your repositories.
- GitHub is public, allowing millions to share their projects to the world.



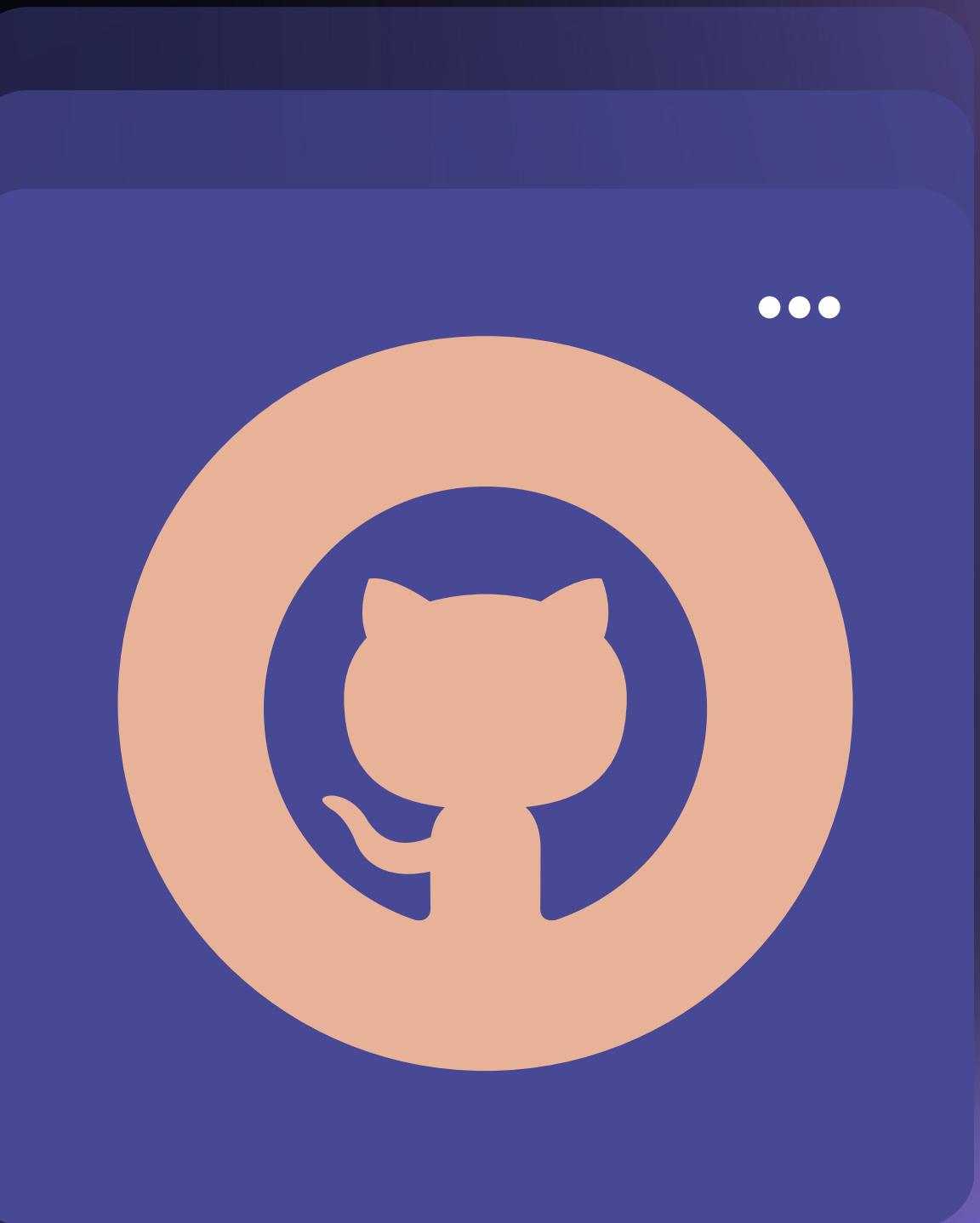
# Why use GitHub?

GitHub allows you to:

- Have a portfolio of your best work.
- Open opportunities you might not have with a local repository
- Easy access from any device in any location.
- Industry standard for hosting Git repositories

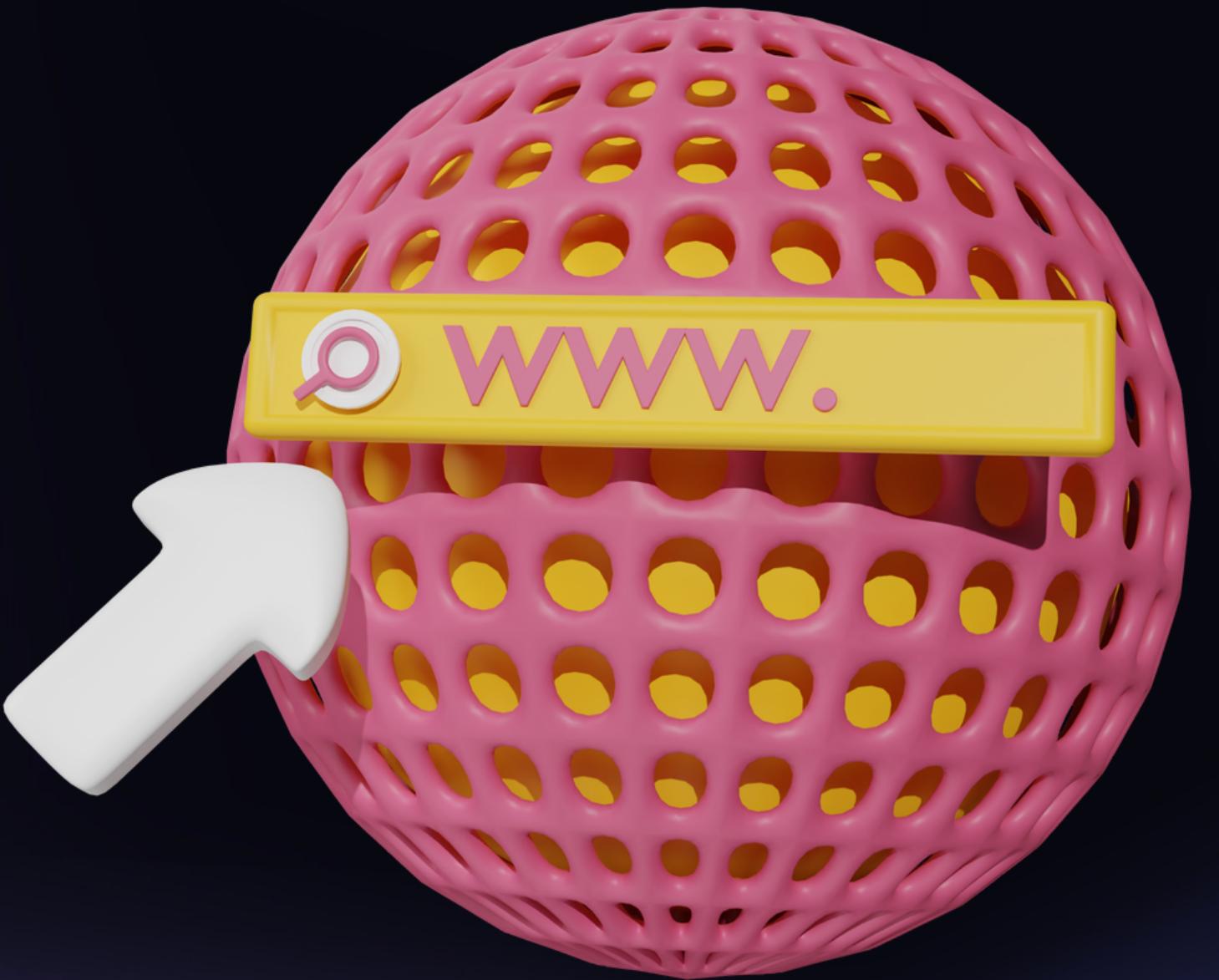


# How to Create GitHub Account



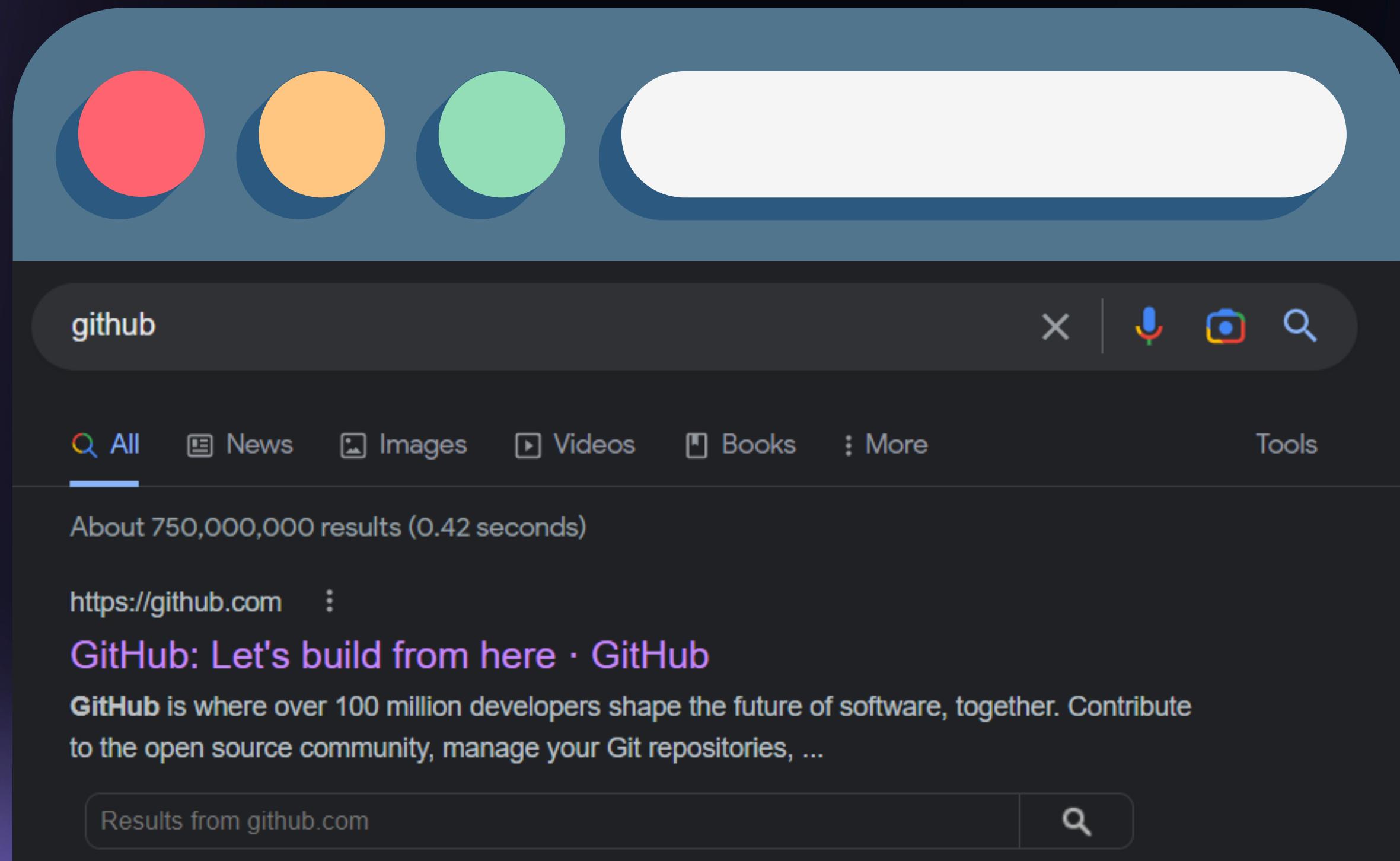
# How to Create GitHub Account

- open your web browser and search GitHub



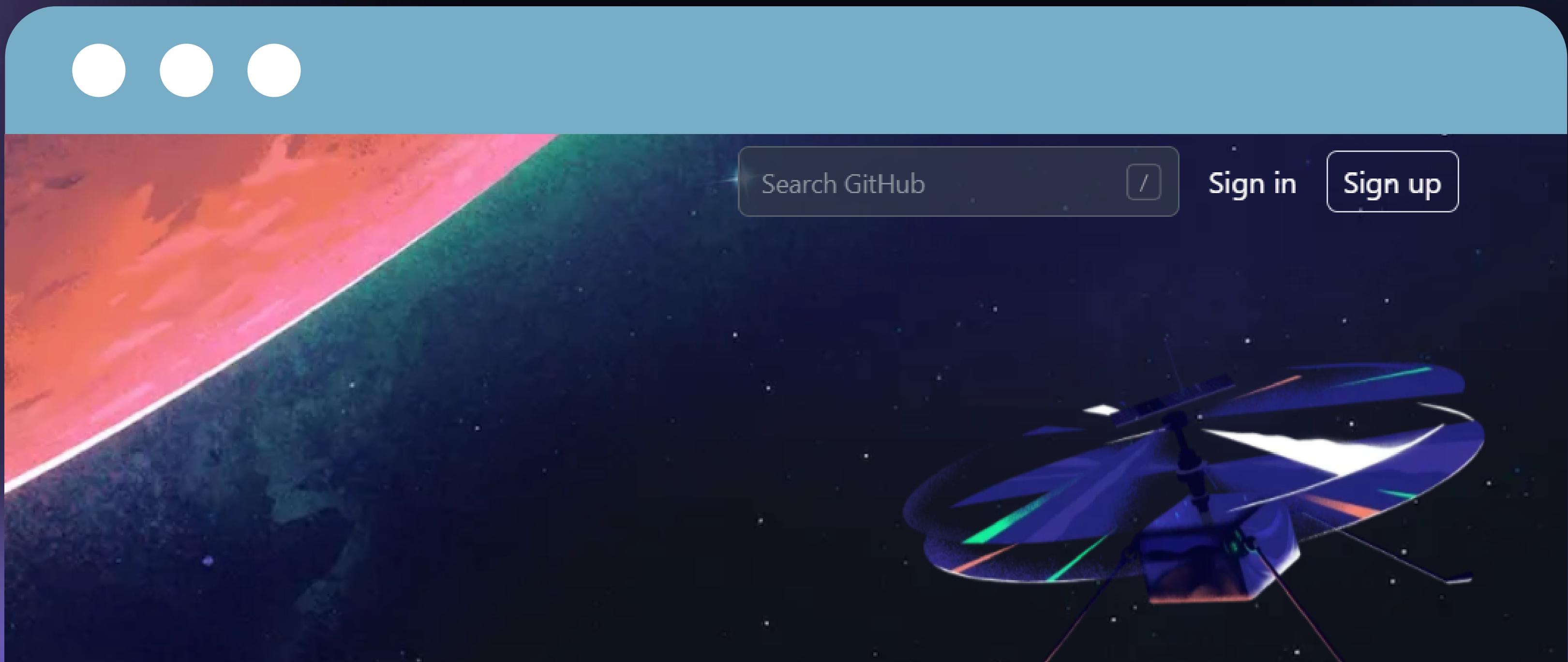
# How to Create GitHub Account

- click on first search result on your web browser



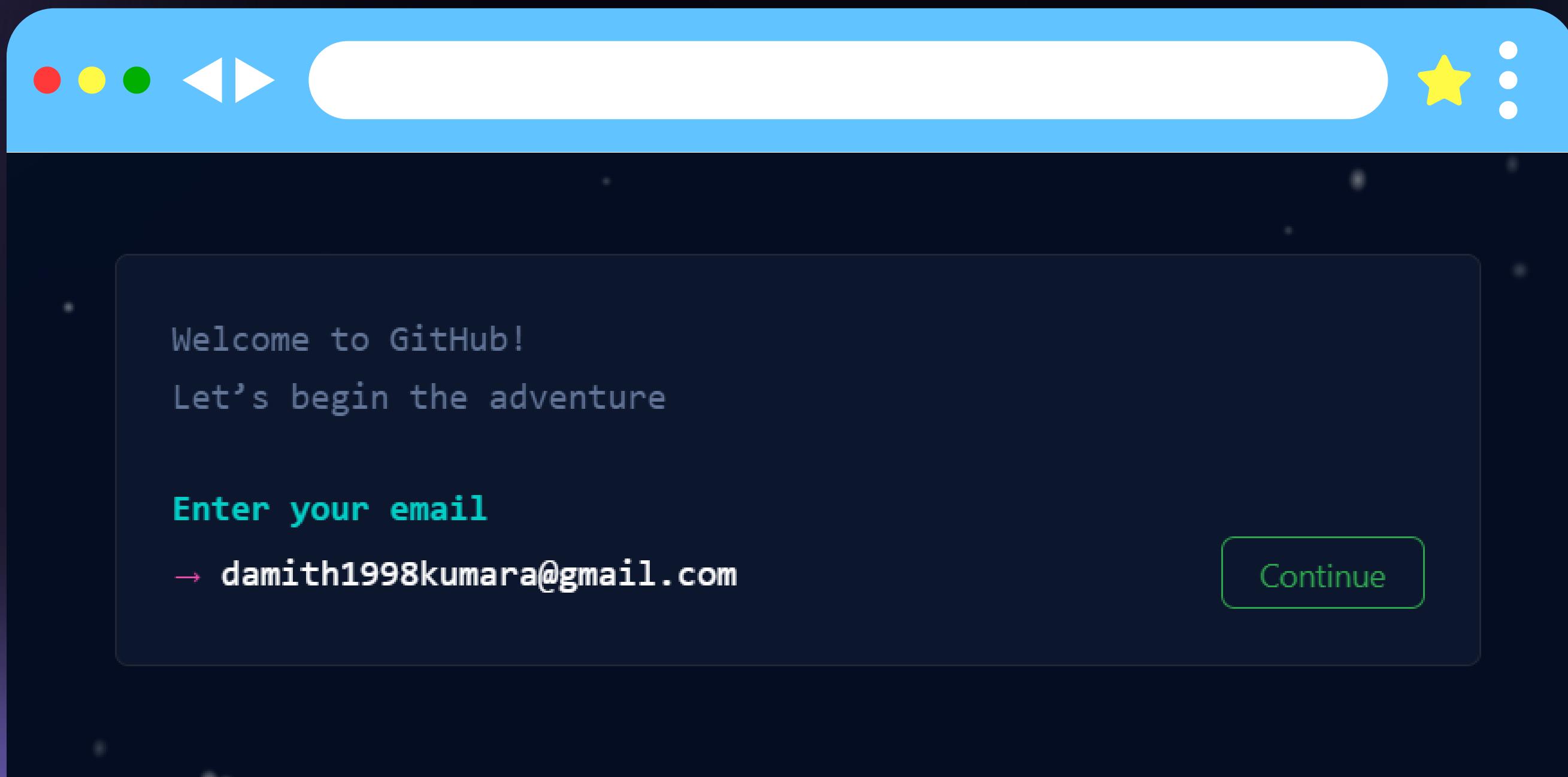
# How to Create GitHub Account

- now click on Signup button



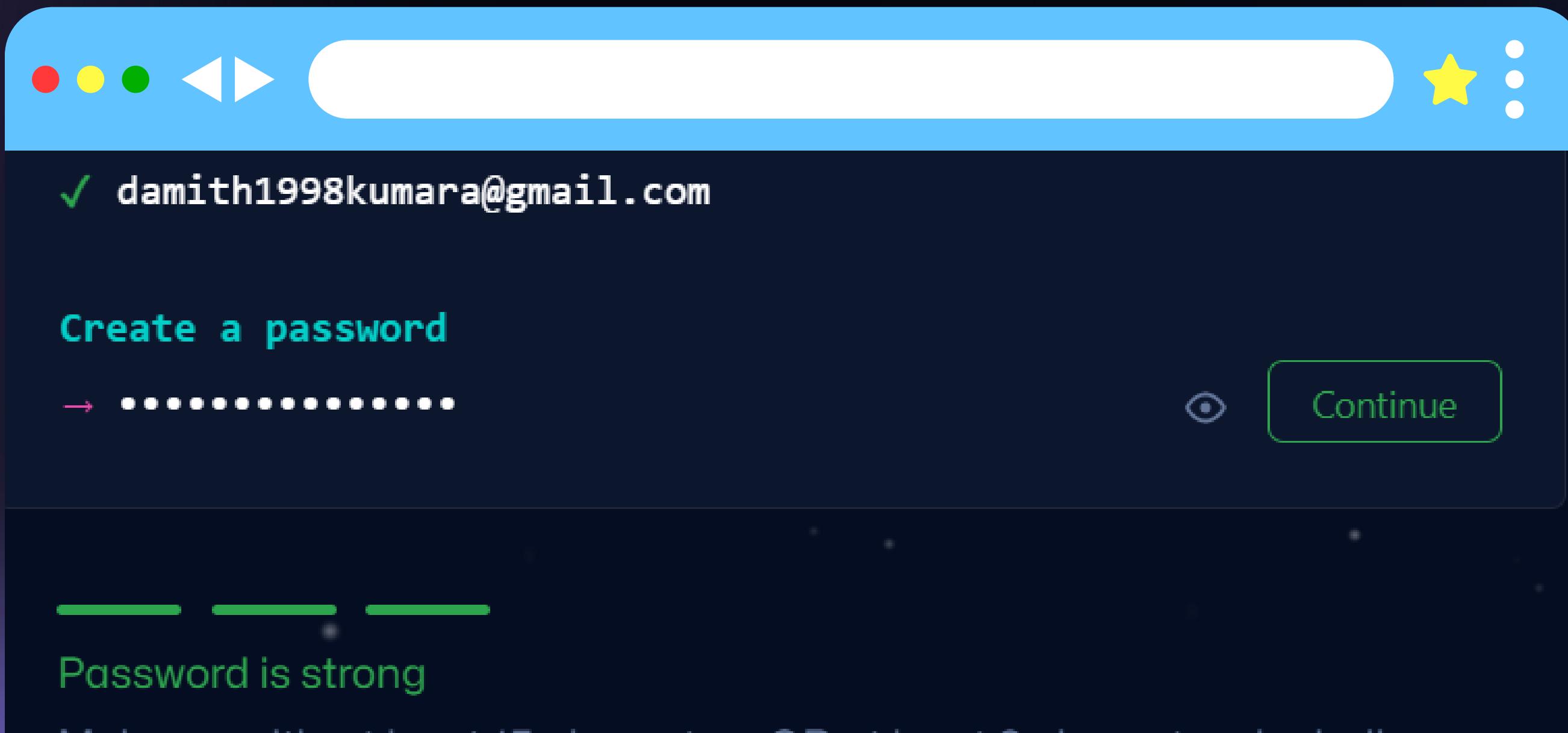
# How to Create GitHub Account

- now enter your valid email address and click Continue



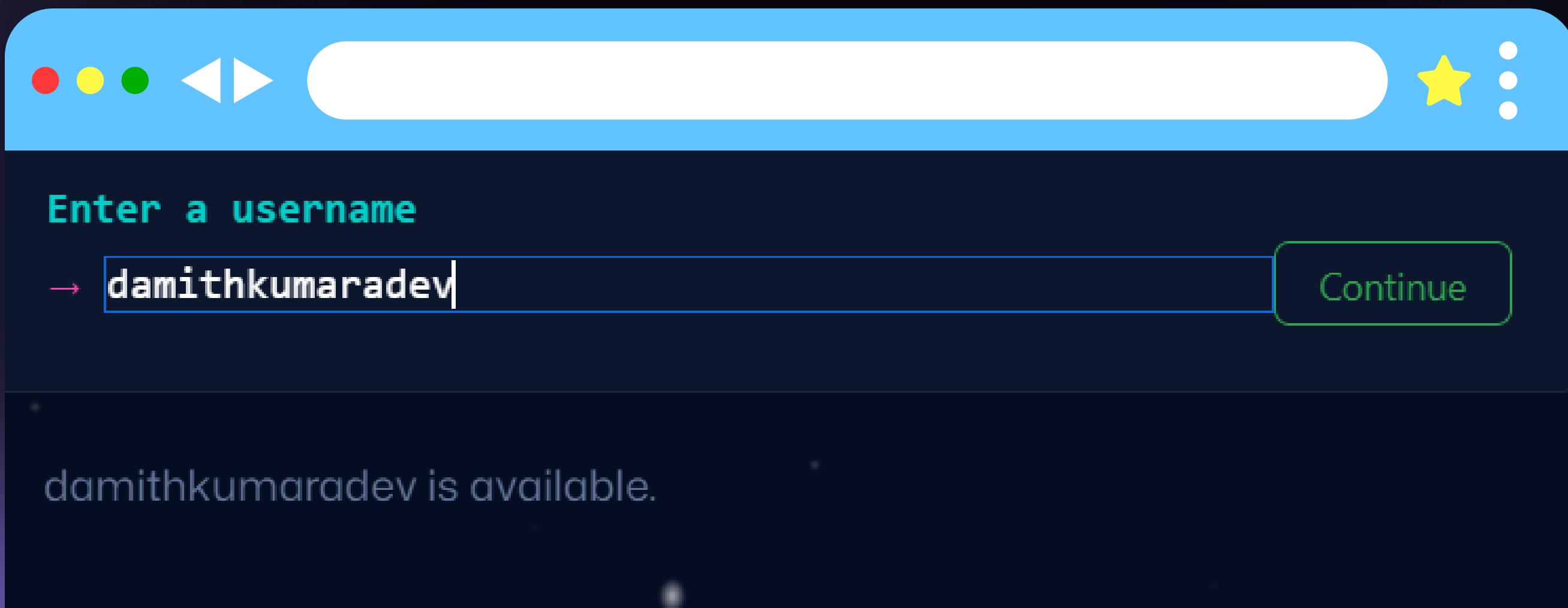
# How to Create GitHub Account

- create strong password and click continue



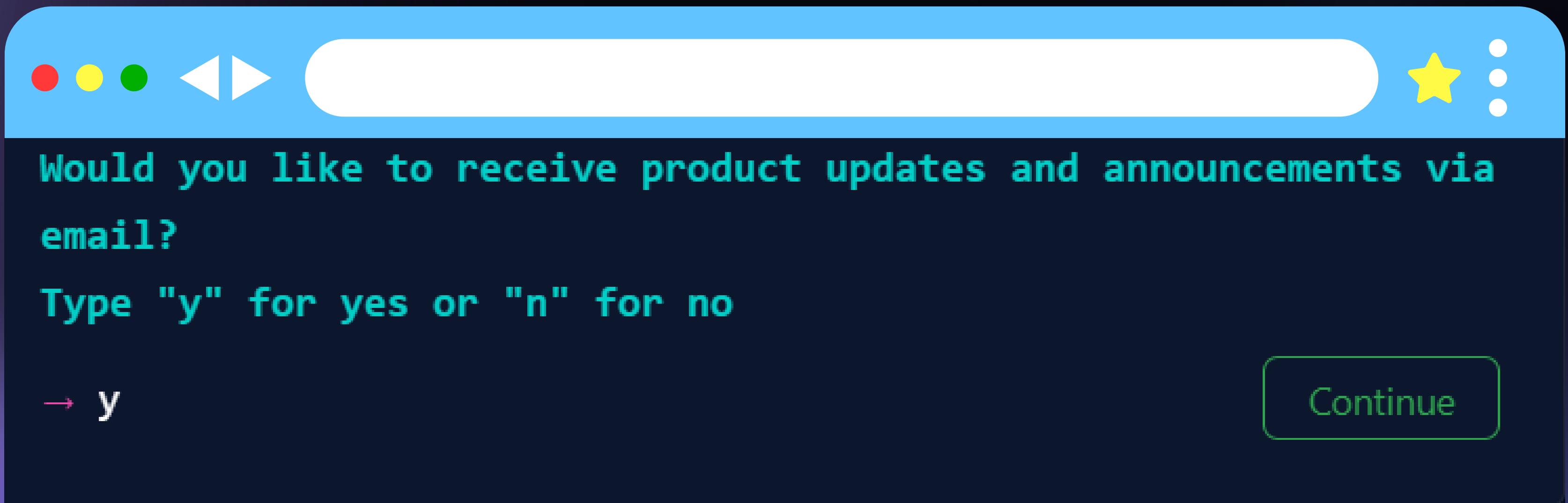
# How to Create GitHub Account

- set your username for github and check that is available or not , if that available click continue



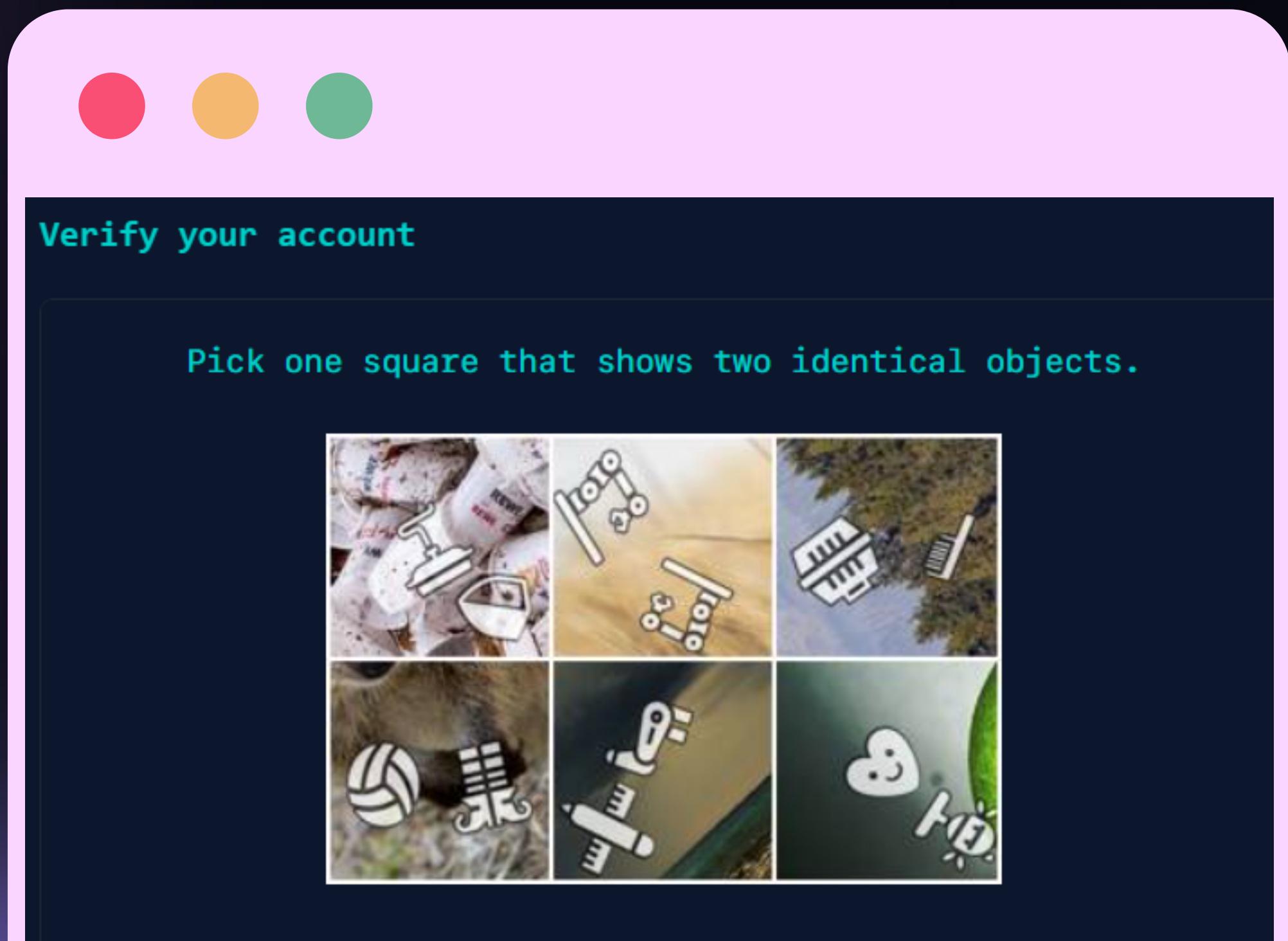
# How to Create GitHub Account

- in here you can type y for yes and n for no, after that click on Continue button



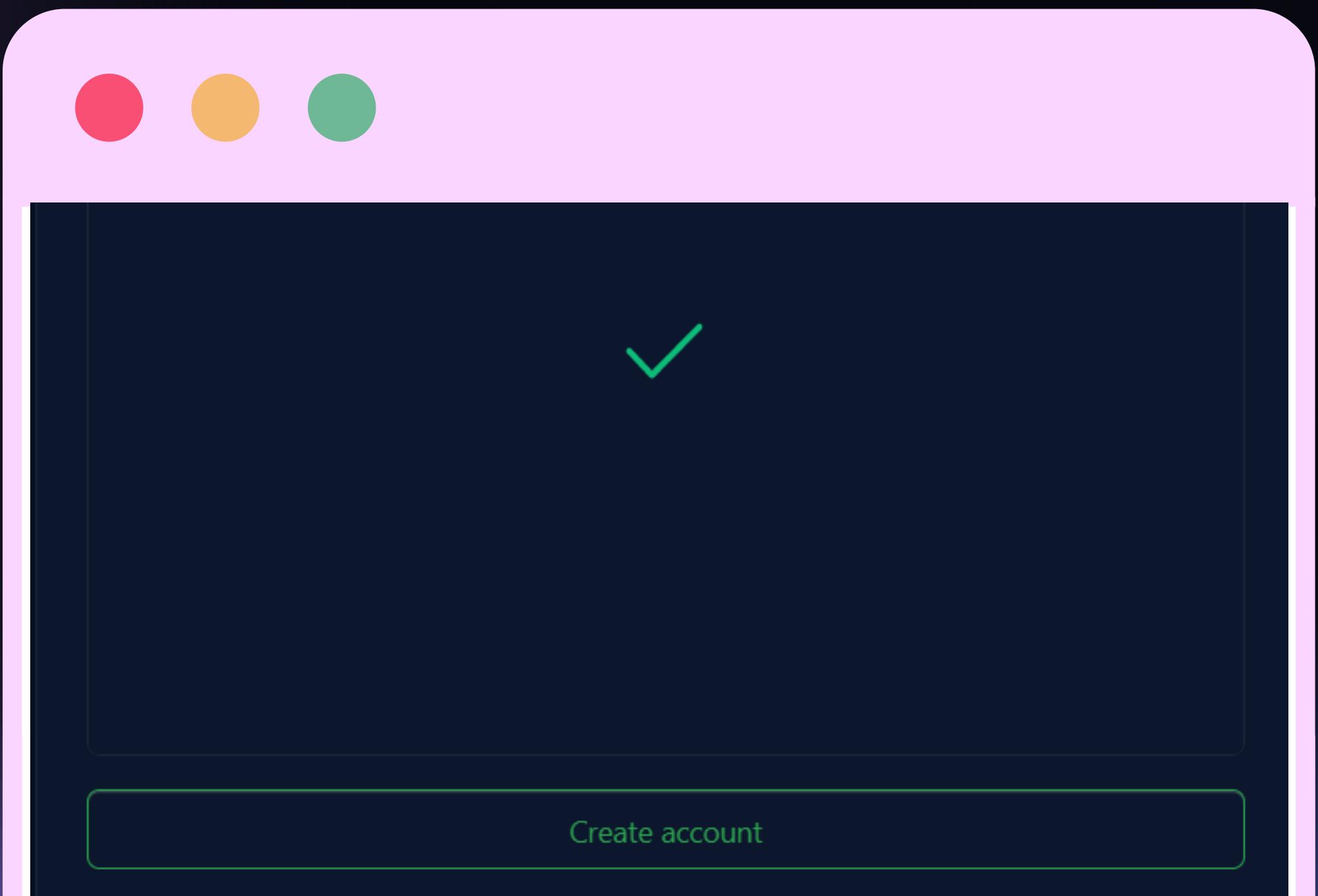
# How to Create GitHub Account

- now you need to verify your account by solving the puzzle



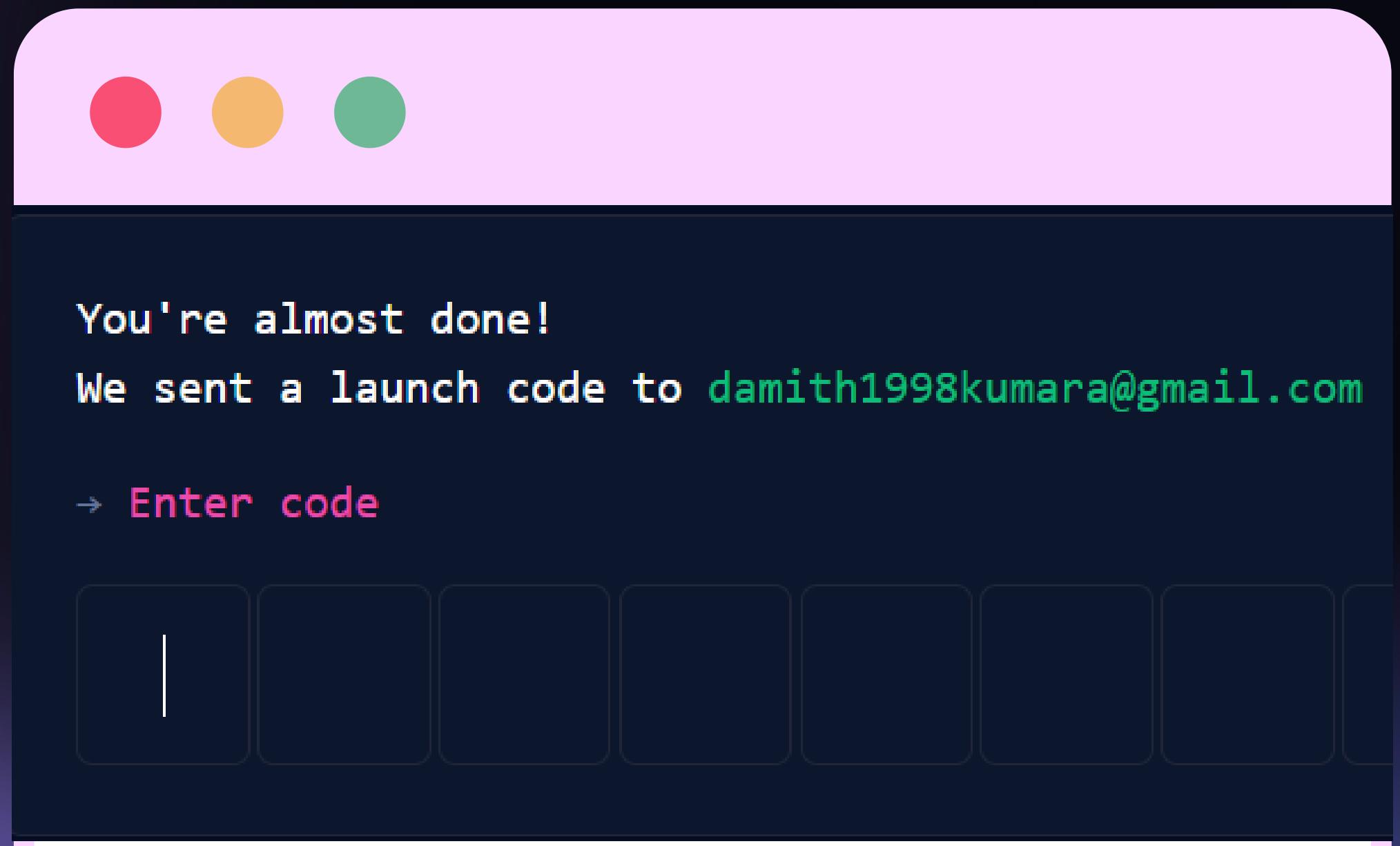
# How to Create GitHub Account

- after solving the puzzle simply click on the create account



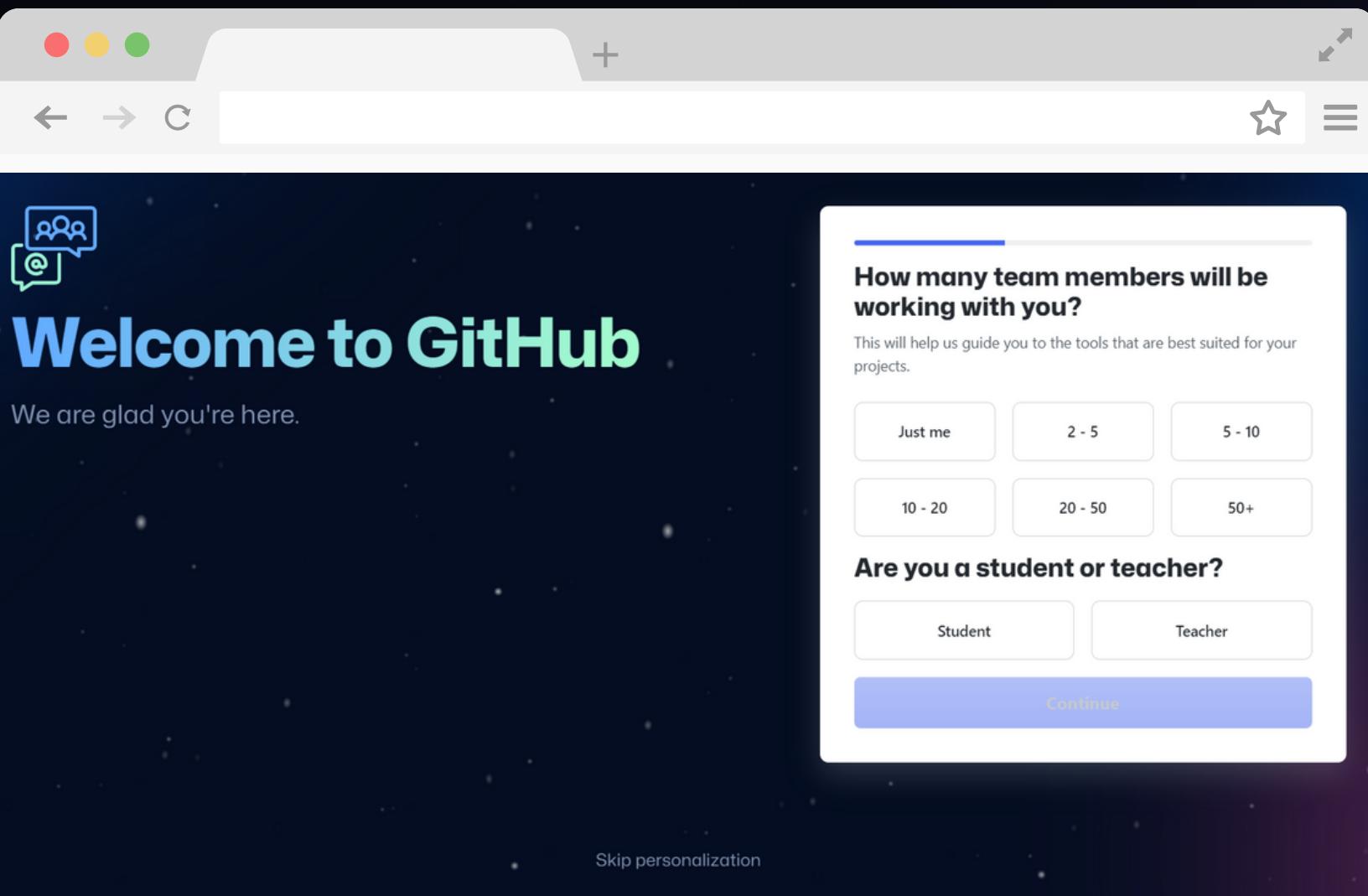
# How to Create GitHub Account

- now you need to enter the verification code, you can find that in your email address



# How to Create GitHub Account

- if you have successfully entered your pin you can see below window in that they are asking some questions regarding you, if you wish you can fill that otherwise you can skip that part.



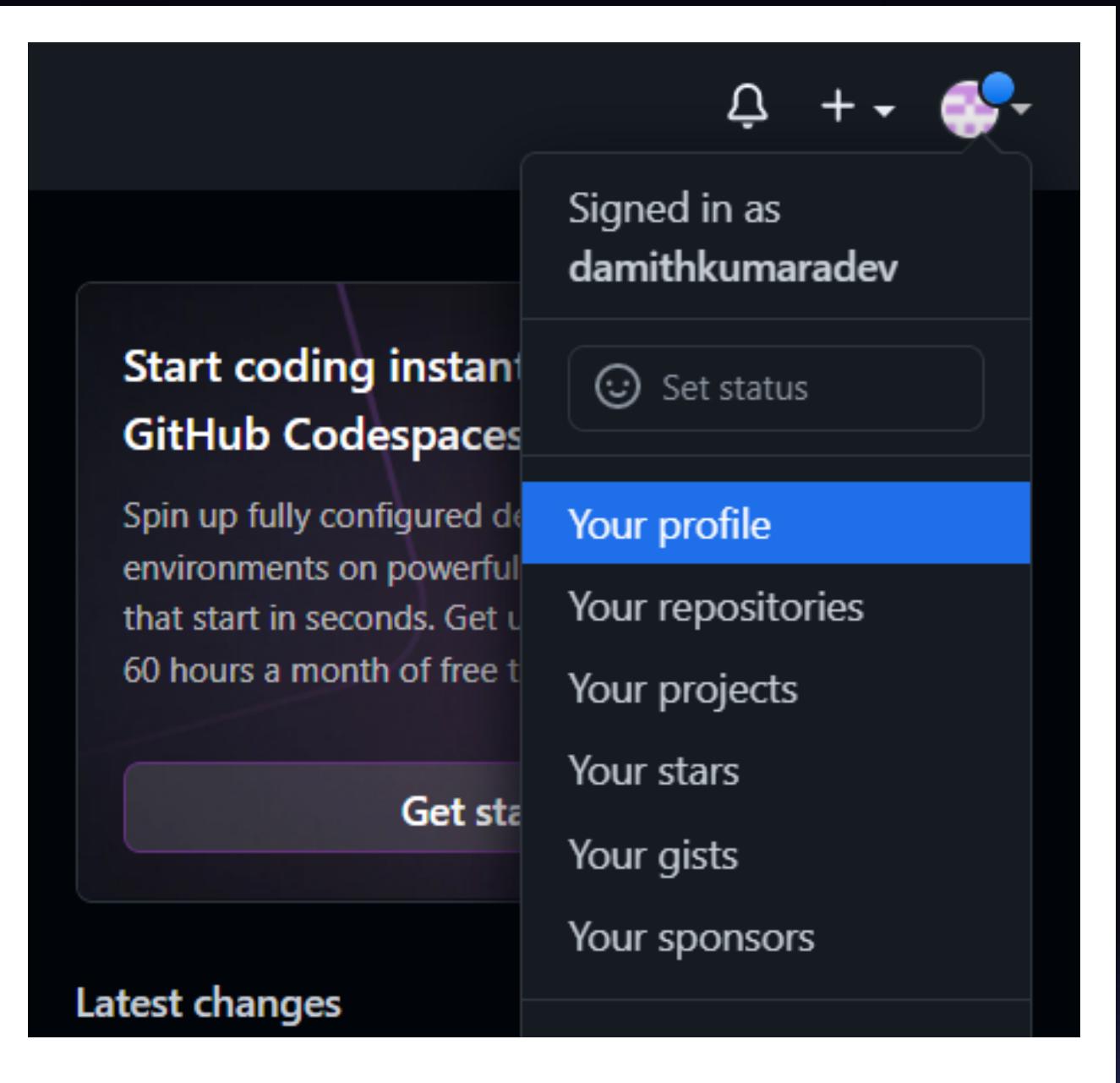


**CONGRATULATIONS**

You have successfully created your  
Github Account

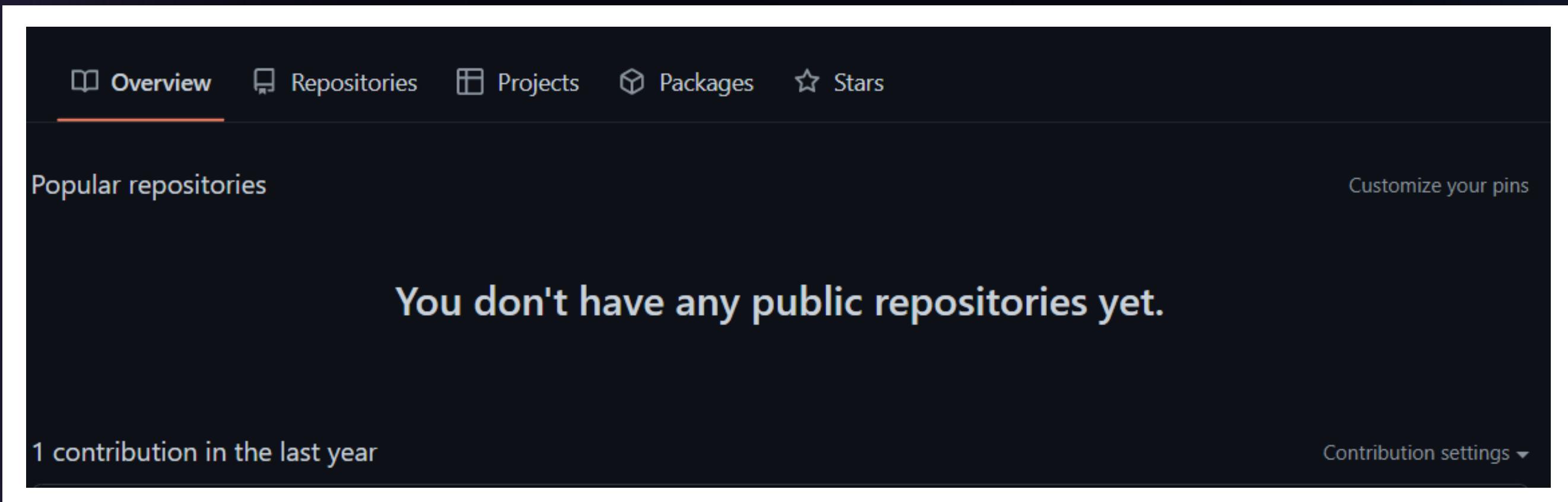
# Create Your First repository

- first go to the top right corner in Github and go to your profile



# Create Your First repository

- now click on repositories



# Create Your First repository

- now click on New button after that you can see below window

Create a new repository

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

Owner \* Repository name \*

 damithkumaradev /

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

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](#).

.gitignore template: [None ▾](#)

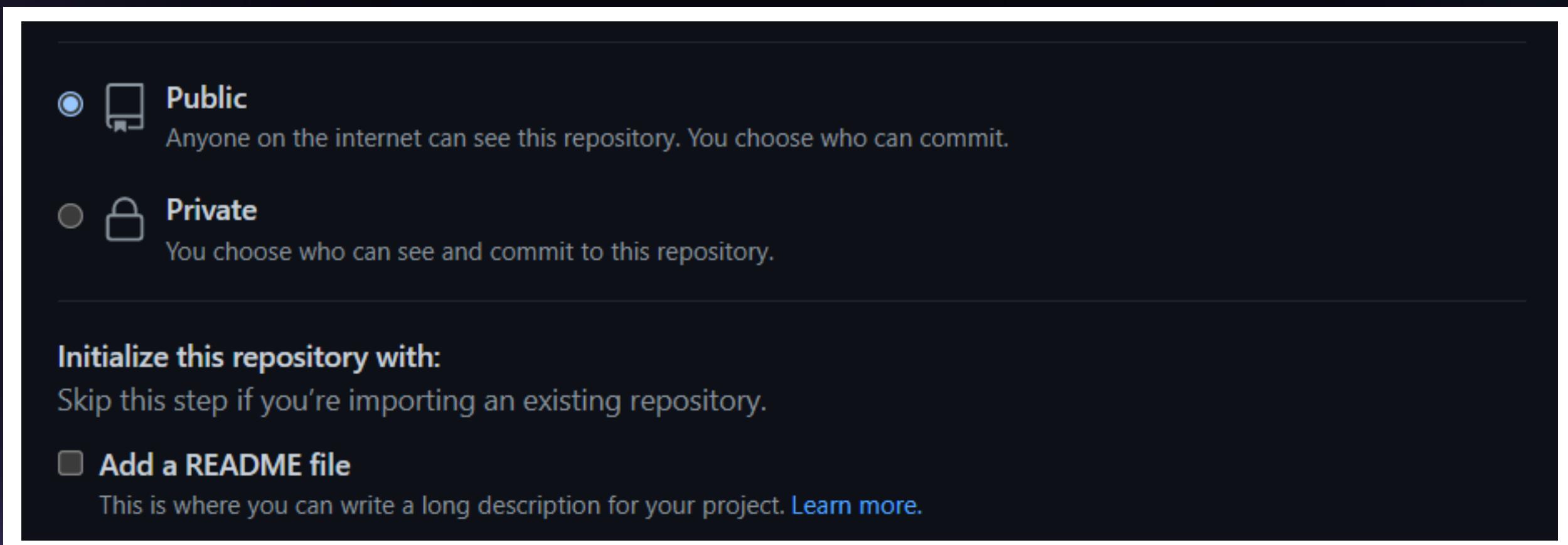
# Create Your First repository

- now you need to set the Owner of your repository and need to give the repository name and need to give some description about this repository

The screenshot shows a dark-themed GitHub repository creation interface. At the top left, there's a dropdown menu labeled "Owner \*". Inside the dropdown, a user icon and the text "damithkumaradev" are visible, followed by a downward arrow. To the right of the dropdown is a separator line with a slash character. To the right of the slash is a text input field labeled "Repository name \*". Below these fields, a note reads: "Great repository names are short and memorable. Need inspiration? How about [didactic-fortnight](#)?" At the bottom left, there's a section labeled "Description (optional)" with a text input field below it.

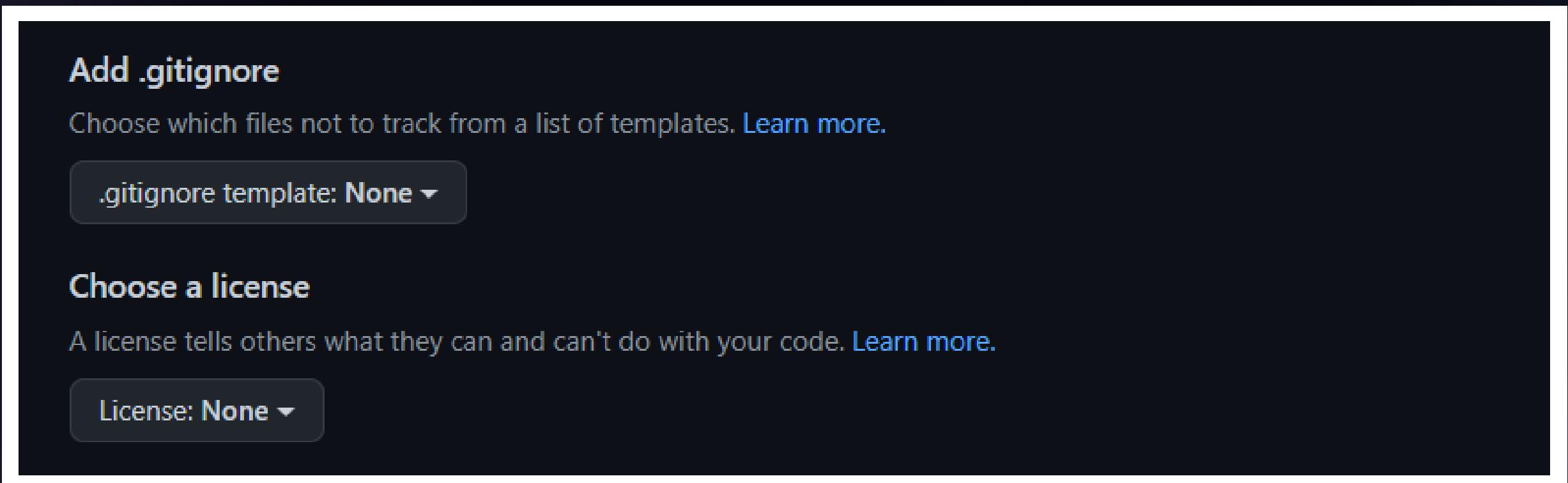
# Create Your First repository

- need to select visibility of your repository, that is public or private. after that you need do add the README file for your repository



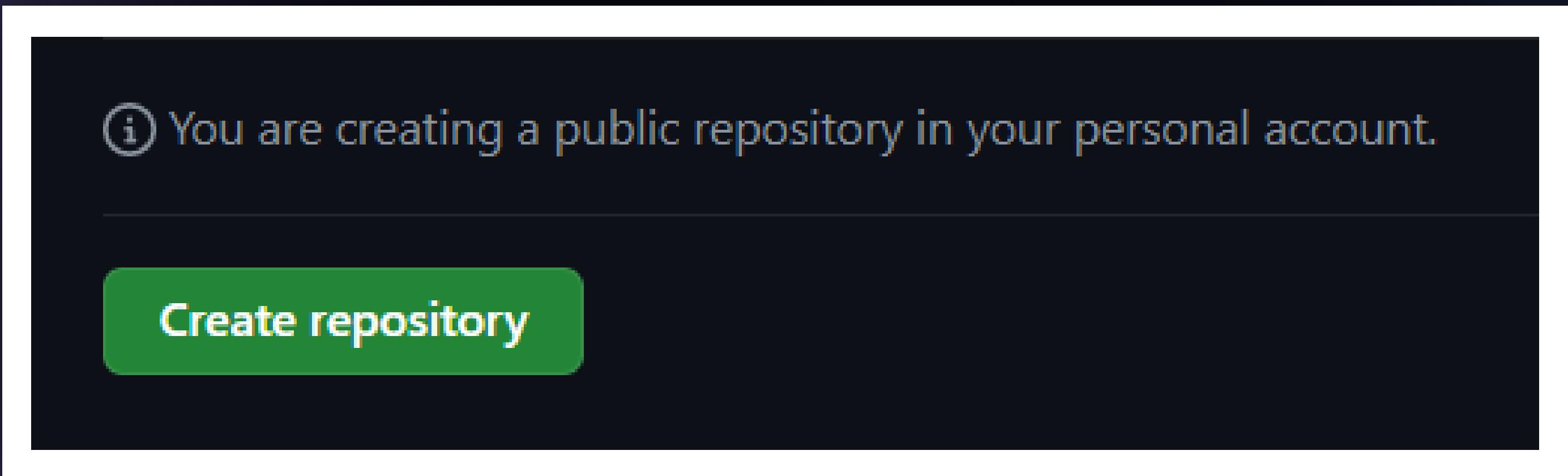
# Create Your First repository

- now add .gitignore file to your repository and Choose a license for your project



# Create Your First repository

- after that click on create repository



Owner \* Repository name \*

 damithkumaradev / mytest ✓

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

Description (optional)

this is sample project for github

 **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](#).

.gitignore template: [None](#) ▾

**Choose a license**  
A license tells others what they can and can't do with your code. [Learn more](#).

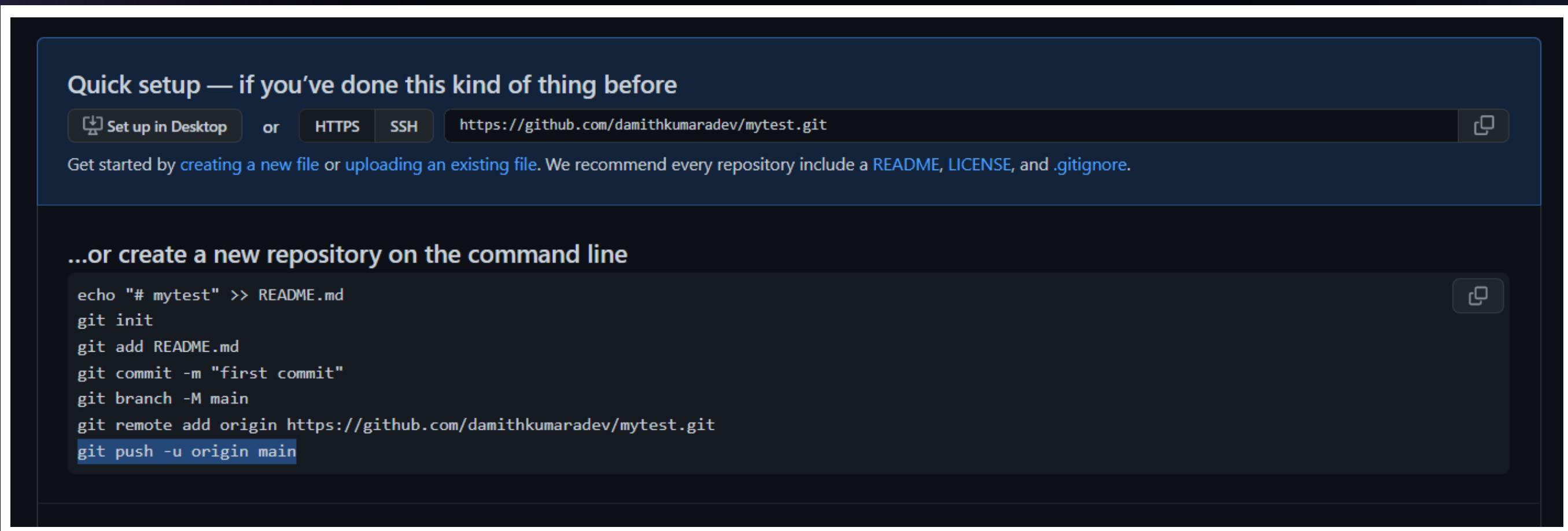
License: [None](#) ▾

 You are creating a public repository in your personal account.

**Create repository**

# Create Your First repository

- Now you can see the new window like below

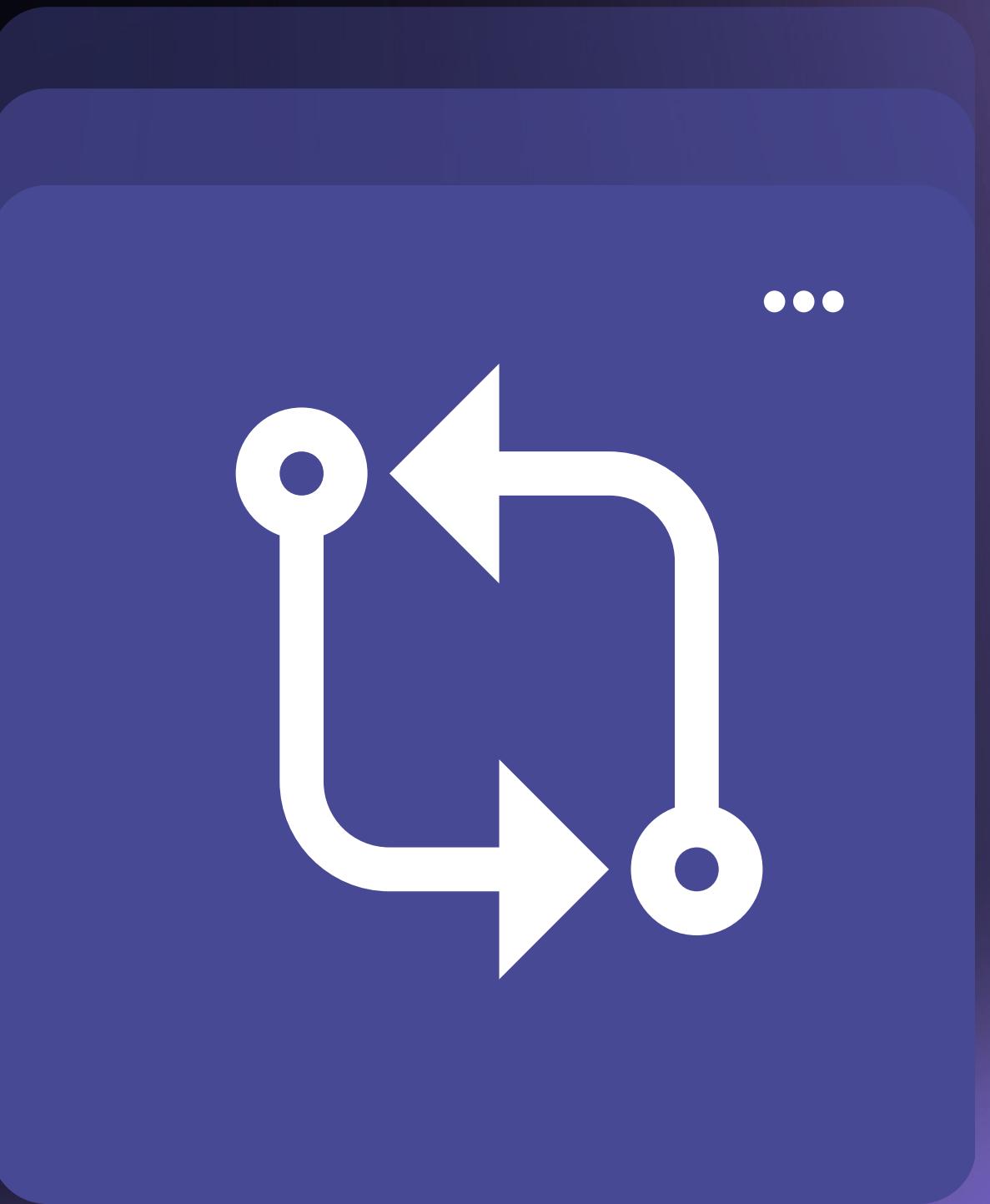




**CONGRATULATIONS**

You have successfully created your  
first repository

Upload your first  
Project to  
GitHub



# Upload your first Project to GitHub

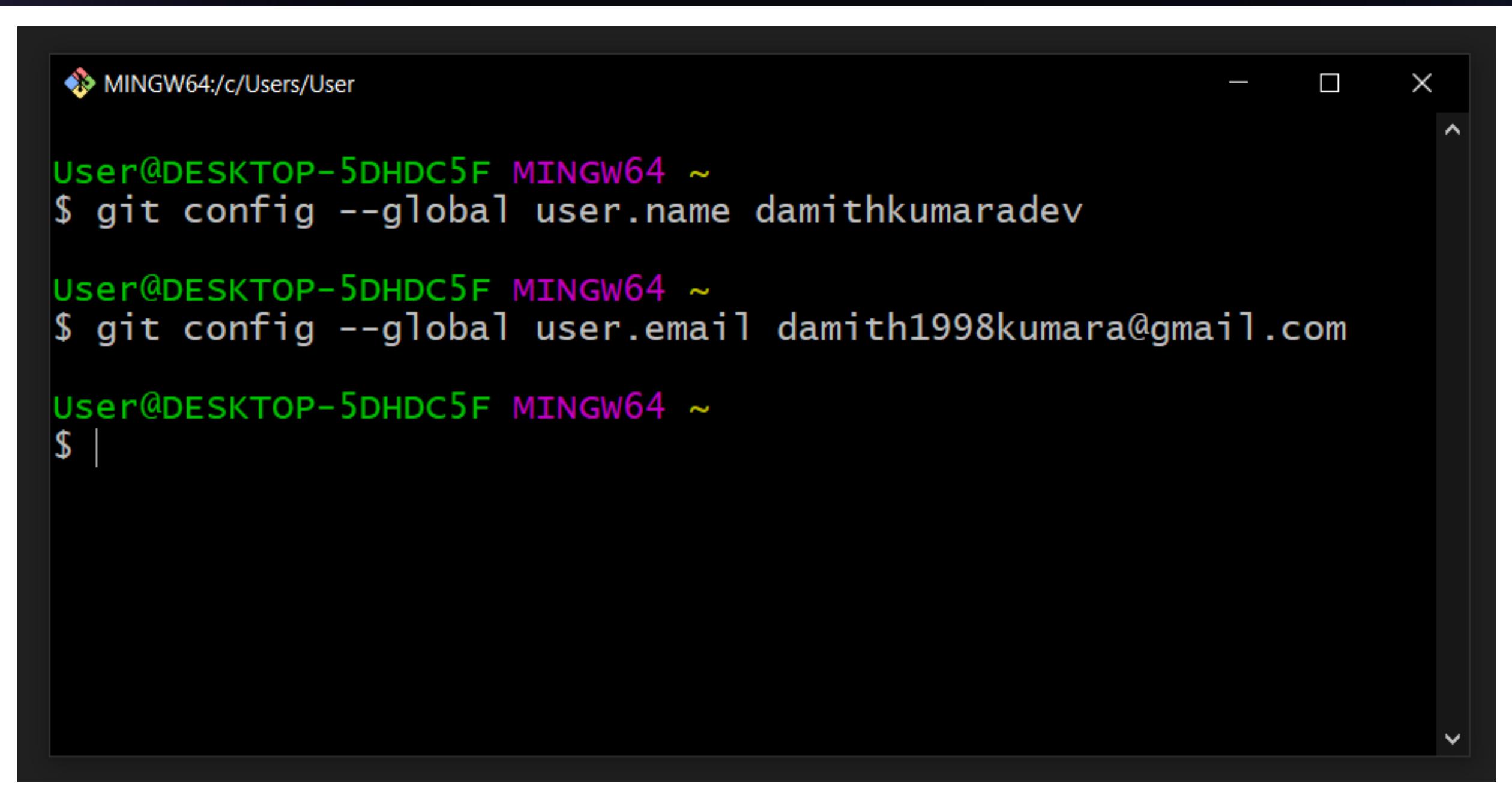
- In first time we need to configure your GitHub account with Git
- For that open Git Bash terminal in your PC
- We can use below code snipped to configure the GitHub account with Git

```
$ git config --global user.name "Your name here"
```

```
$ git config --global user.email "your_email@example.com"
```

# Upload your first Project to GitHub

- type that command and press the enter key



A screenshot of a terminal window titled "MINGW64:c/Users/User". The window contains the following text:

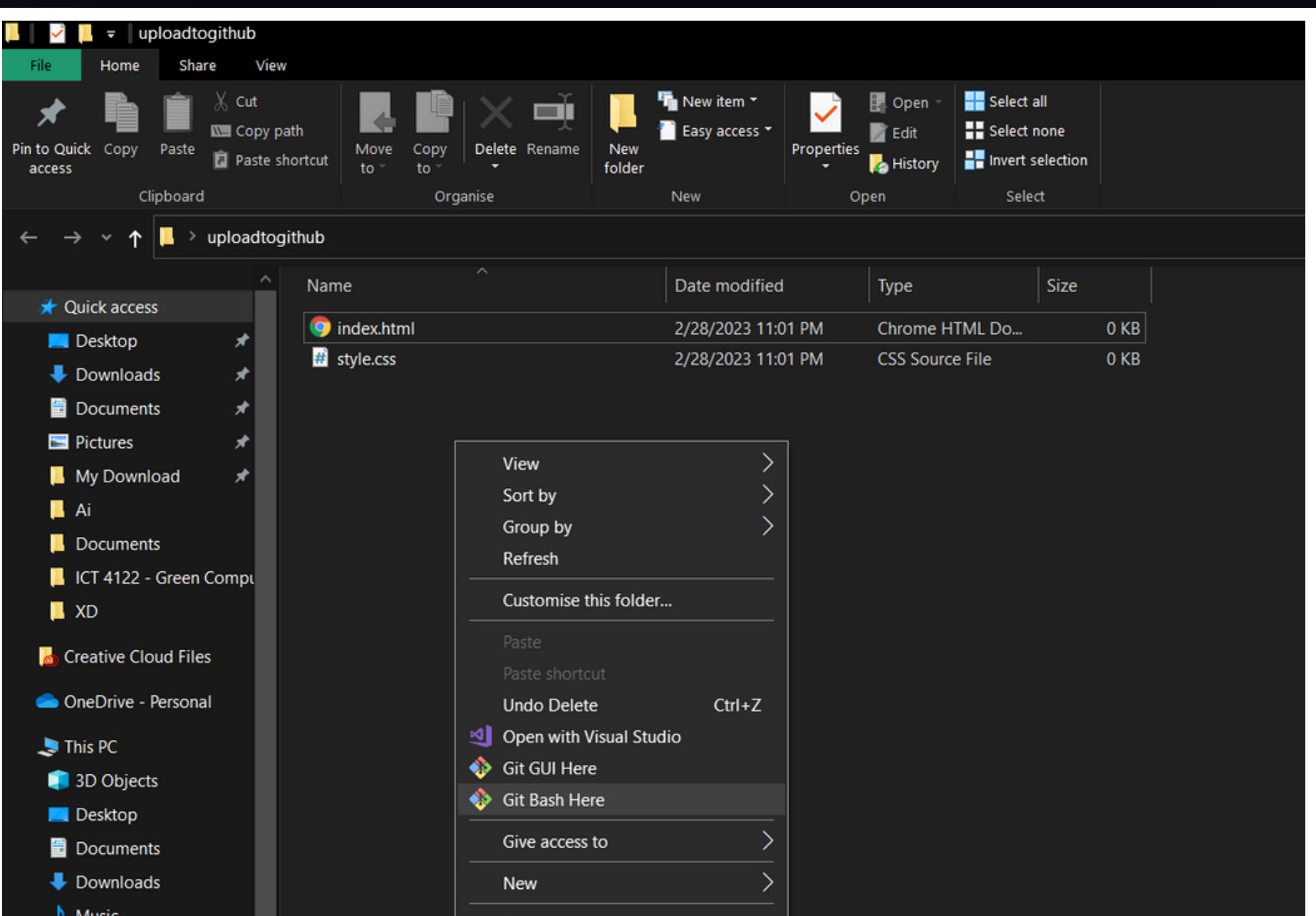
```
User@DESKTOP-5DHDC5F MINGW64 ~
$ git config --global user.name damithkumaradev

User@DESKTOP-5DHDC5F MINGW64 ~
$ git config --global user.email damith1998kumara@gmail.com

User@DESKTOP-5DHDC5F MINGW64 ~
$ |
```

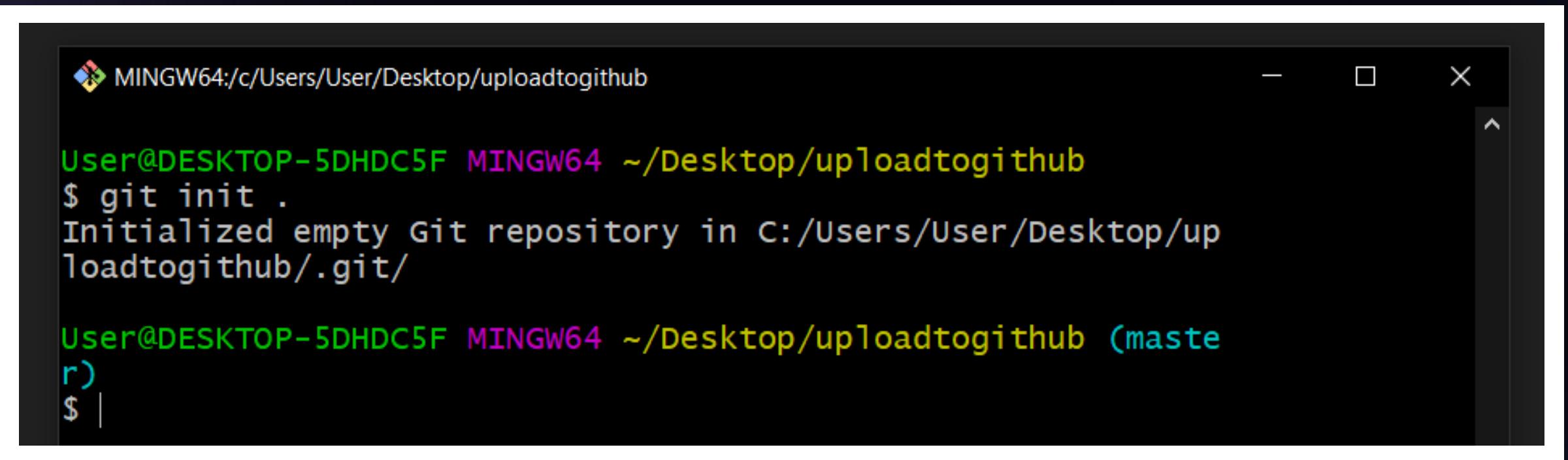
# Upload your first Project to GitHub

- Now go to your project folder and right click on that folder, select Git Bash Here



# Upload your first Project to GitHub

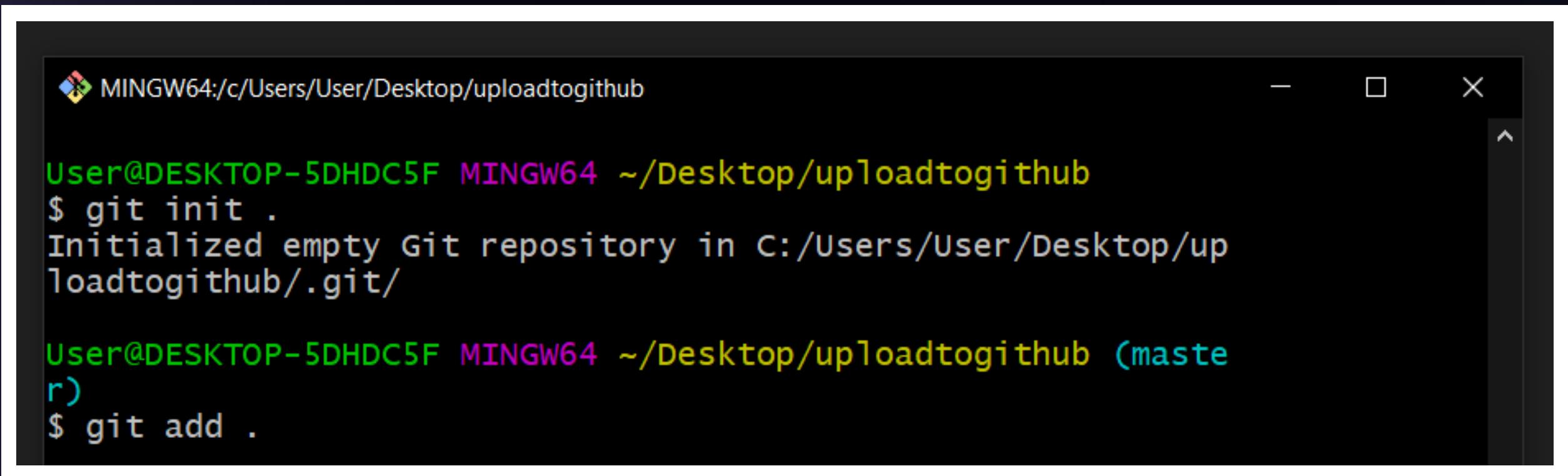
- Now enter the below code snipped and press enter  
git init .



```
MINGW64:/c/Users/User/Desktop/uploadtoghithub
User@DESKTOP-5DHDC5F MINGW64 ~/Desktop/uploadtoghithub
$ git init .
Initialized empty Git repository in C:/Users/User/Desktop/up
loadtoghithub/.git/
User@DESKTOP-5DHDC5F MINGW64 ~/Desktop/uploadtoghithub (maste
r)
$ |
```

# Upload your first Project to GitHub

- Now enter the below code snipped and press enter  
git add .



The screenshot shows a terminal window titled "MINGW64:/c/Users/User/Desktop/uploadtoghithub". The window contains the following text:

```
User@DESKTOP-5DHDC5F MINGW64 ~/Desktop/uploadtoghithub
$ git init .
Initialized empty Git repository in C:/Users/User/Desktop/up
loadtoghithub/.git/

User@DESKTOP-5DHDC5F MINGW64 ~/Desktop/uploadtoghithub (maste
r)
$ git add .
```

# Upload your first Project to GitHub

- Now enter the below code snipped and press enter  
git commit -m "first commit"

```
User@DESKTOP-5DHDC5F MINGW64 ~/Desktop/uploadtoghithub (master)
$ git commit -m "my first upload"
[master (root-commit) b78d196] my first upload
 2 files changed, 0 insertions(+), 0 deletions(-)
  create mode 100644 index.html
  create mode 100644 style.css
```

# Upload your first Project to GitHub

- Now enter the below code snipped and press enter  
git branch -M main

```
User@DESKTOP-5DHDC5F MINGW64 ~/Desktop/uploadtoghithub (master)
$ git branch -M main

User@DESKTOP-5DHDC5F MINGW64 ~/Desktop/uploadtoghithub (main)
$
```

# Upload your first Project to GitHub

- Now enter the below code snipped and press enter  
git remote add origin your\_repository\_url

```
User@DESKTOP-5DHDC5F MINGW64 ~/Desktop/uploadtoghithub (master)
$ git branch -M main

User@DESKTOP-5DHDC5F MINGW64 ~/Desktop/uploadtoghithub (main)
$ git remote add origin https://github.com/damithkumaradev/mytest.git

User@DESKTOP-5DHDC5F MINGW64 ~/Desktop/uploadtoghithub (main)
$ |
```

# Upload your first Project to GitHub

- Now enter the below code snipped and press enter  
git push -u origin main

```
User@DESKTOP-5DHDC5F MINGW64 ~/Desktop/uploadtoghithub (master)
$ git branch -M main

User@DESKTOP-5DHDC5F MINGW64 ~/Desktop/uploadtoghithub (main)
$ git remote add origin https://github.com/damithkumaradev/mytest.git

User@DESKTOP-5DHDC5F MINGW64 ~/Desktop/uploadtoghithub (main)
$ |
```

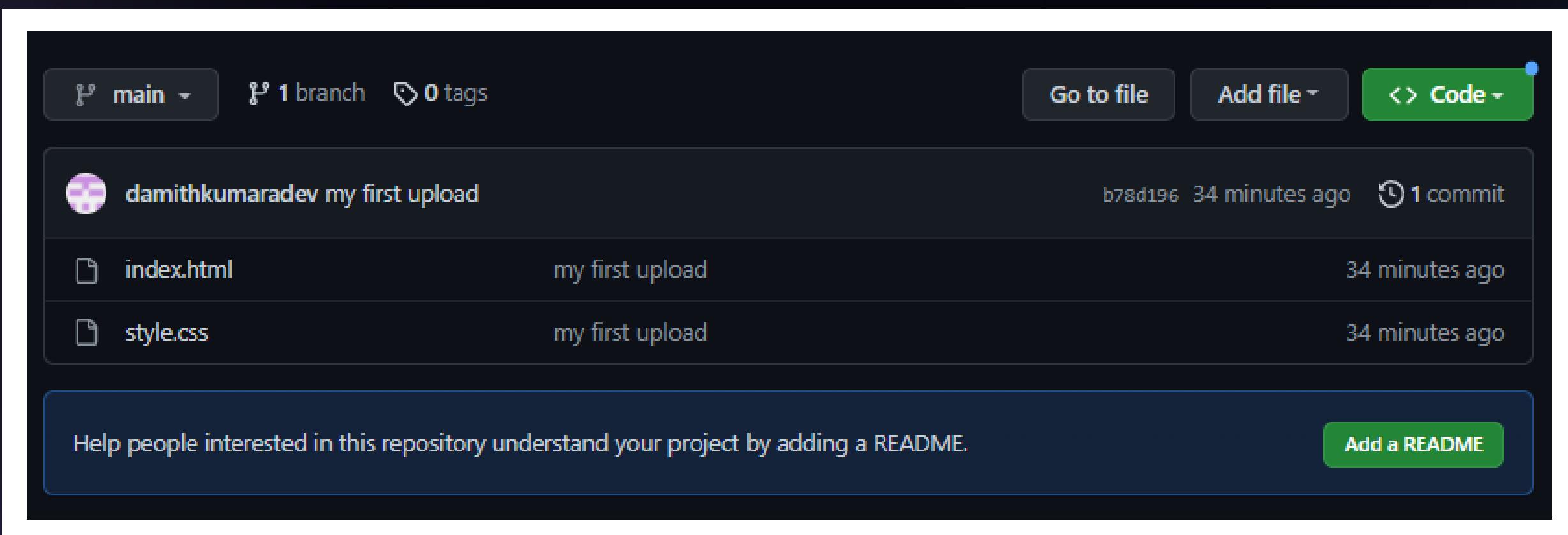
# Upload your first Project to GitHub

- Now enter the below code snipped and press enter  
git push -u origin main

```
User@DESKTOP-5DHDC5F MINGW64 ~/Desktop/uploadtoghithub (main)
$ git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 234 bytes | 234.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/damithkumaradev/mytest.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

# Upload your first Project to GitHub

- Now reload your Web browser and you can see the uploaded code in your GitHub account



# CONGRATULATIONS



You have successfully uploaded your  
first project to GitHub

# Bonus



# Find me on



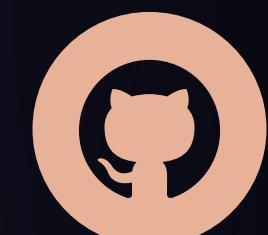
/bdhamithkumara



/bdhamithkumara



/bdhamithkumara



/bdhamithkumara



/bdhamithkumara



THANK  
... you

