

# GitHub

# Basics



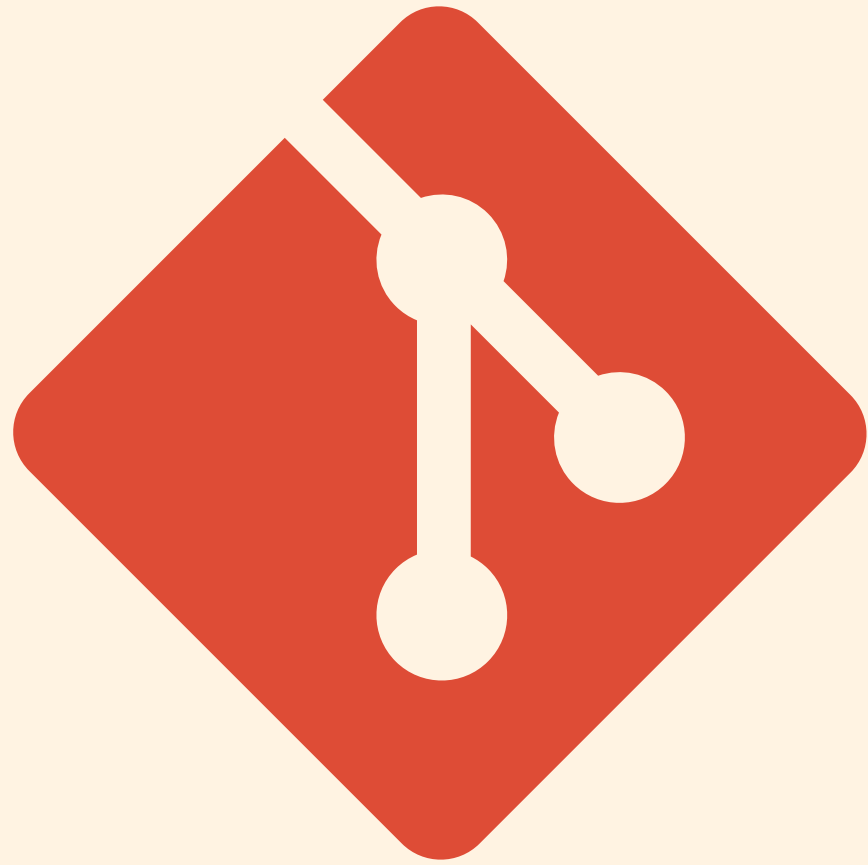


# What Is Github?

Github is a hosting platform for git repositories. You can put your own Git repos on Github and access them from anywhere and share them with people around the world.

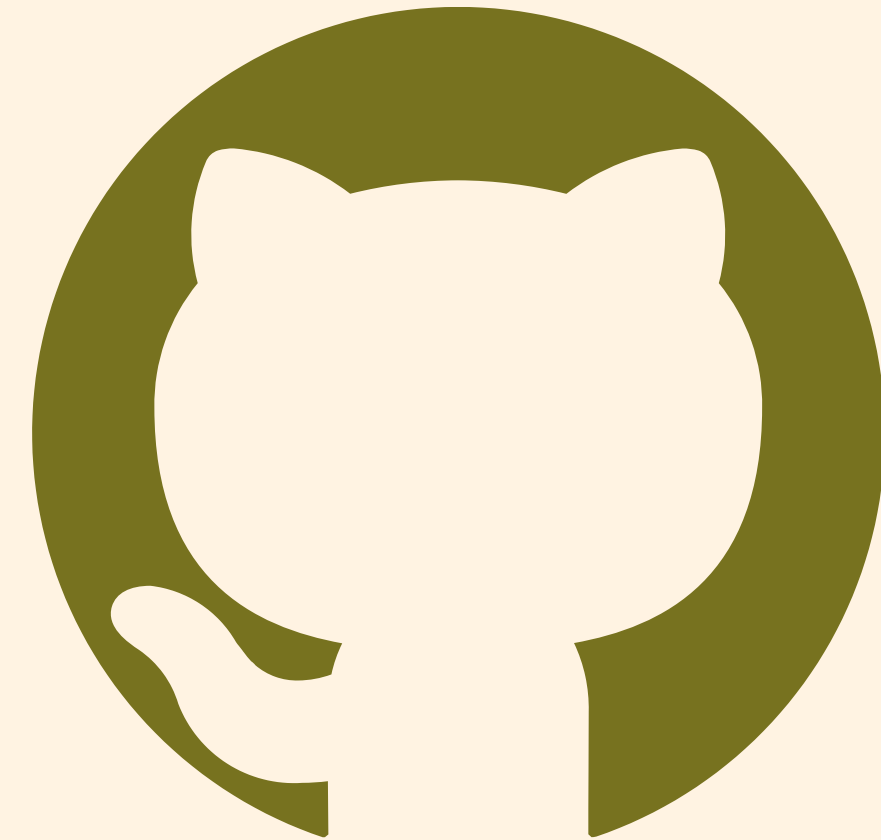
Beyond hosting repos, Github also provides additional collaboration features that are not native to Git (but are super useful). Basically, Github helps people share and collaborate on repos.





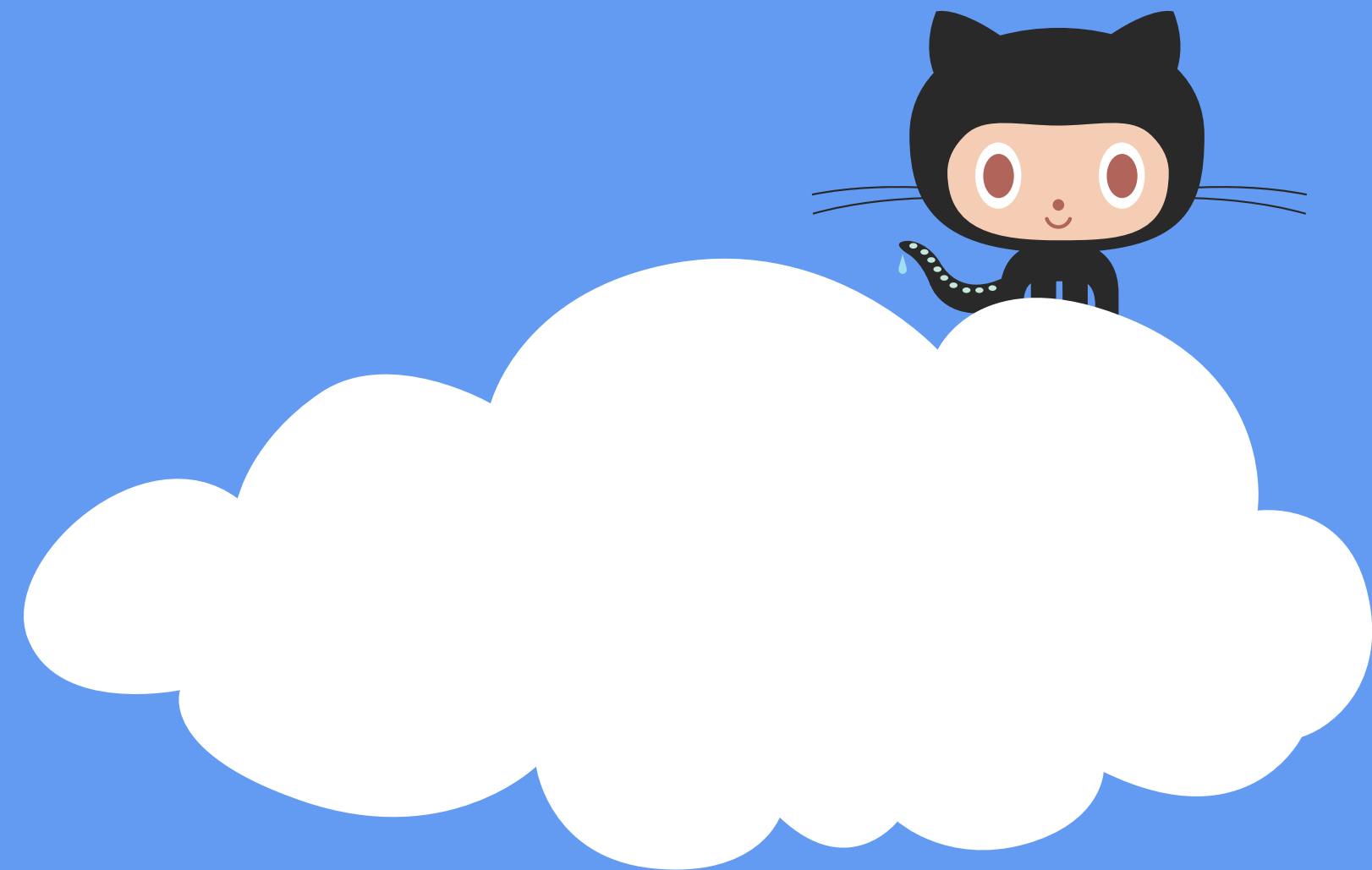
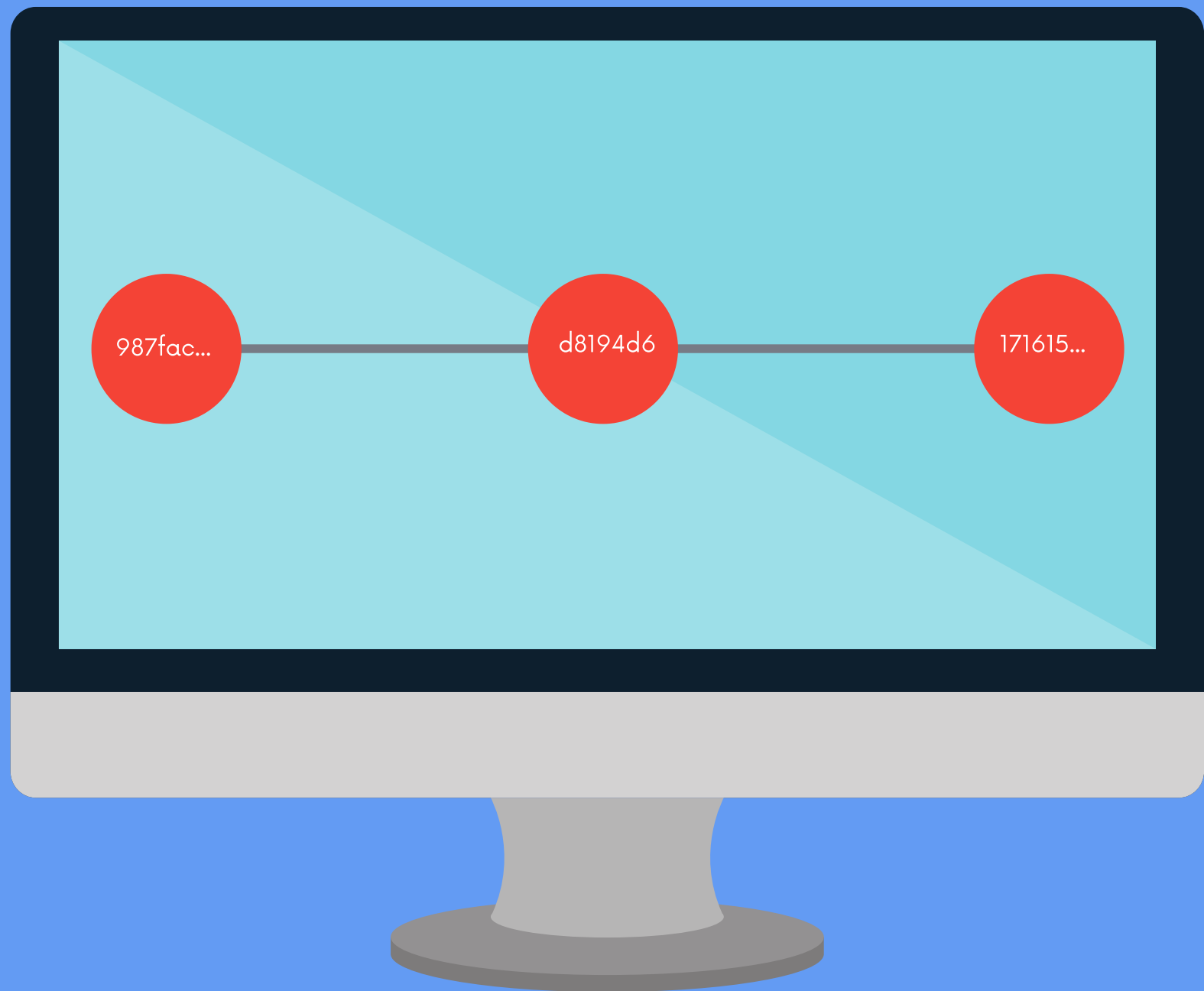
# Git

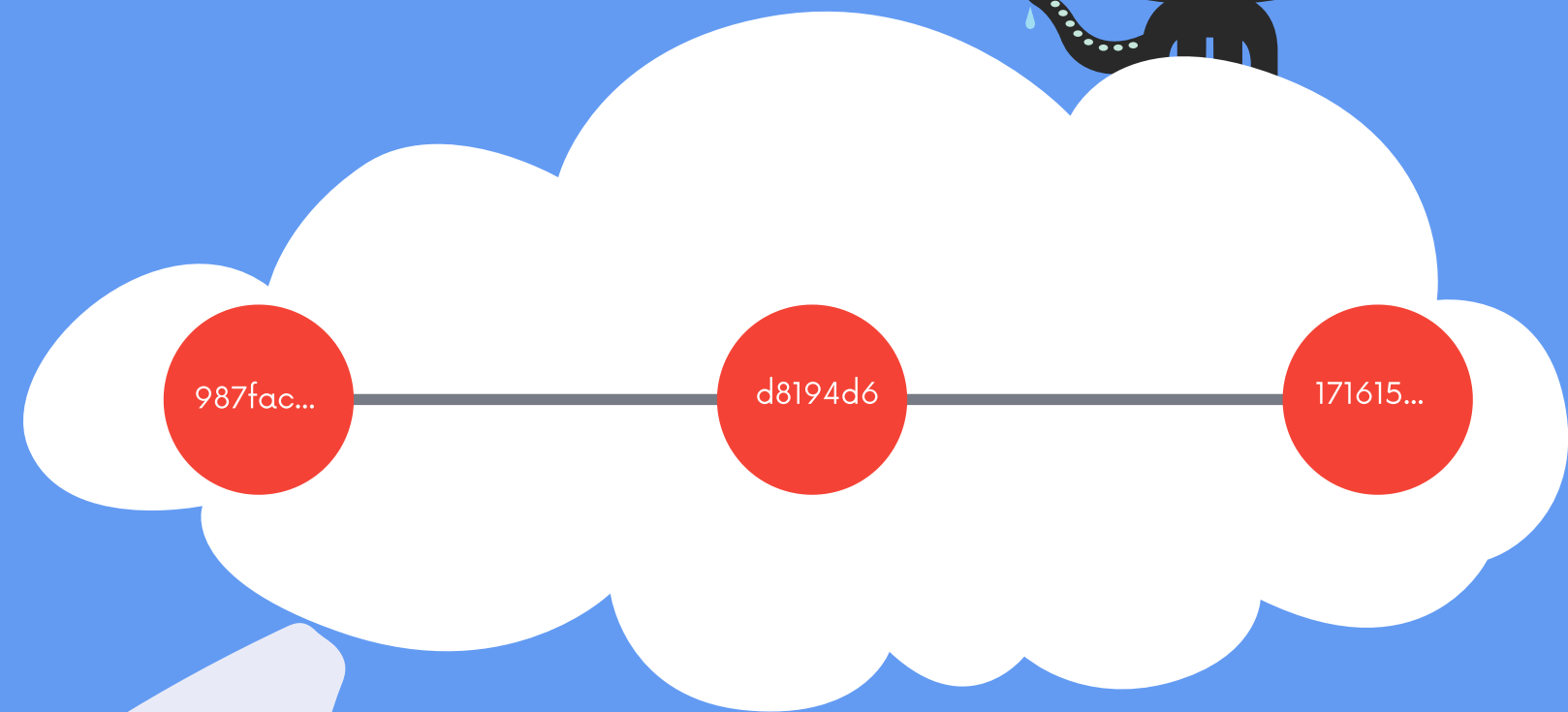
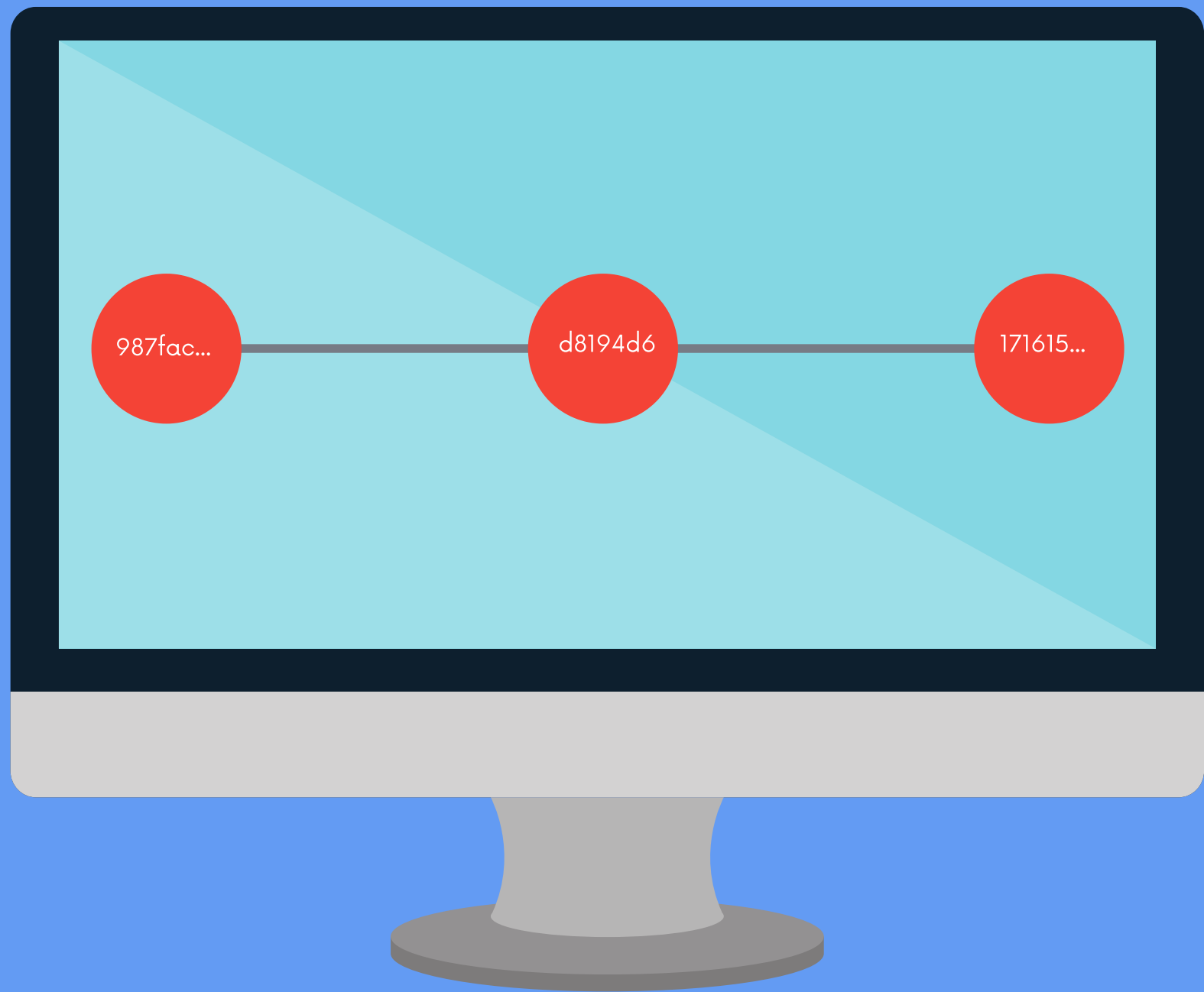
Git is the version control software that runs locally on your machine. You don't need to register for an account. You don't need the internet to use it. You can use Git without ever touching Github.

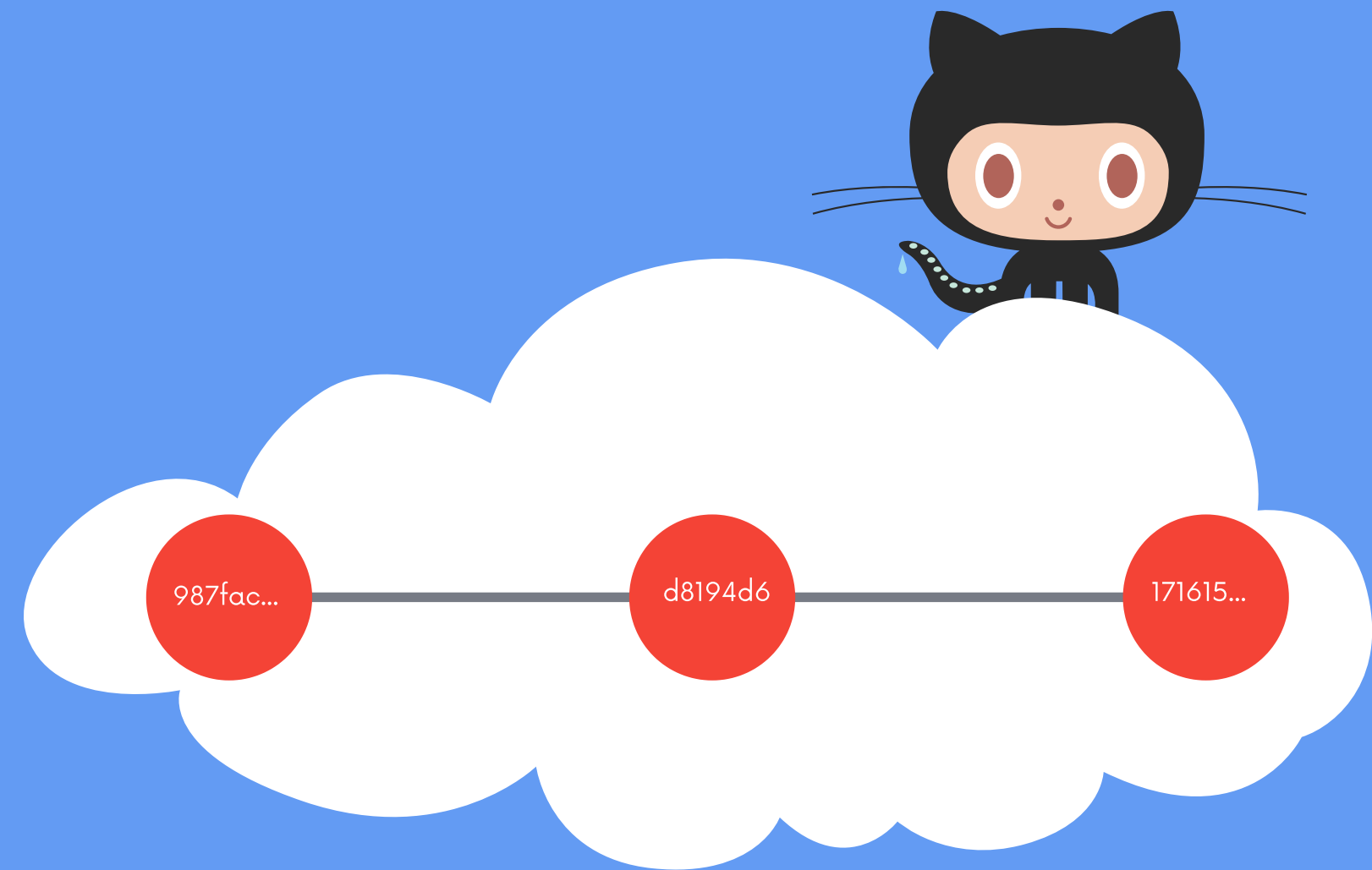
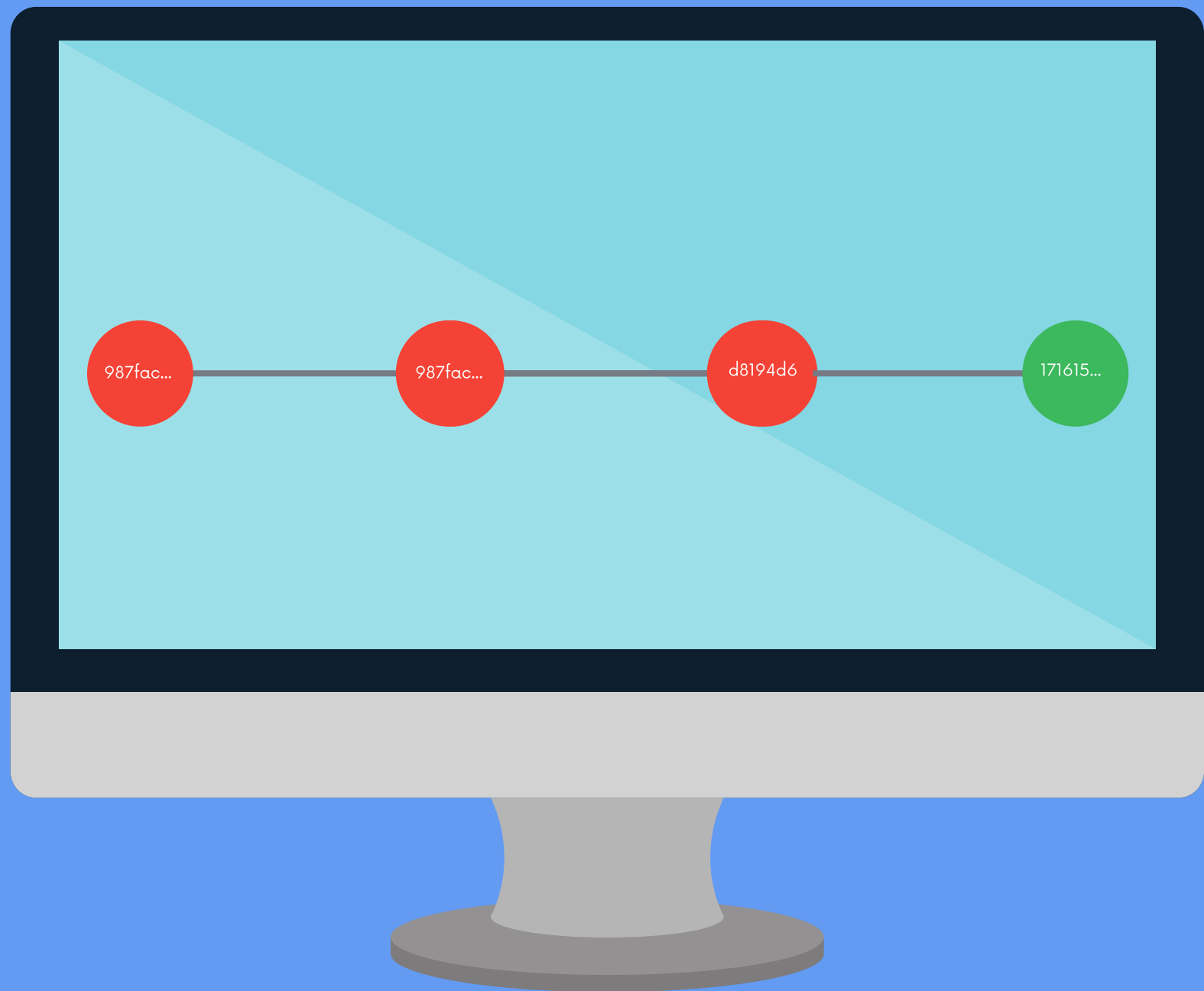


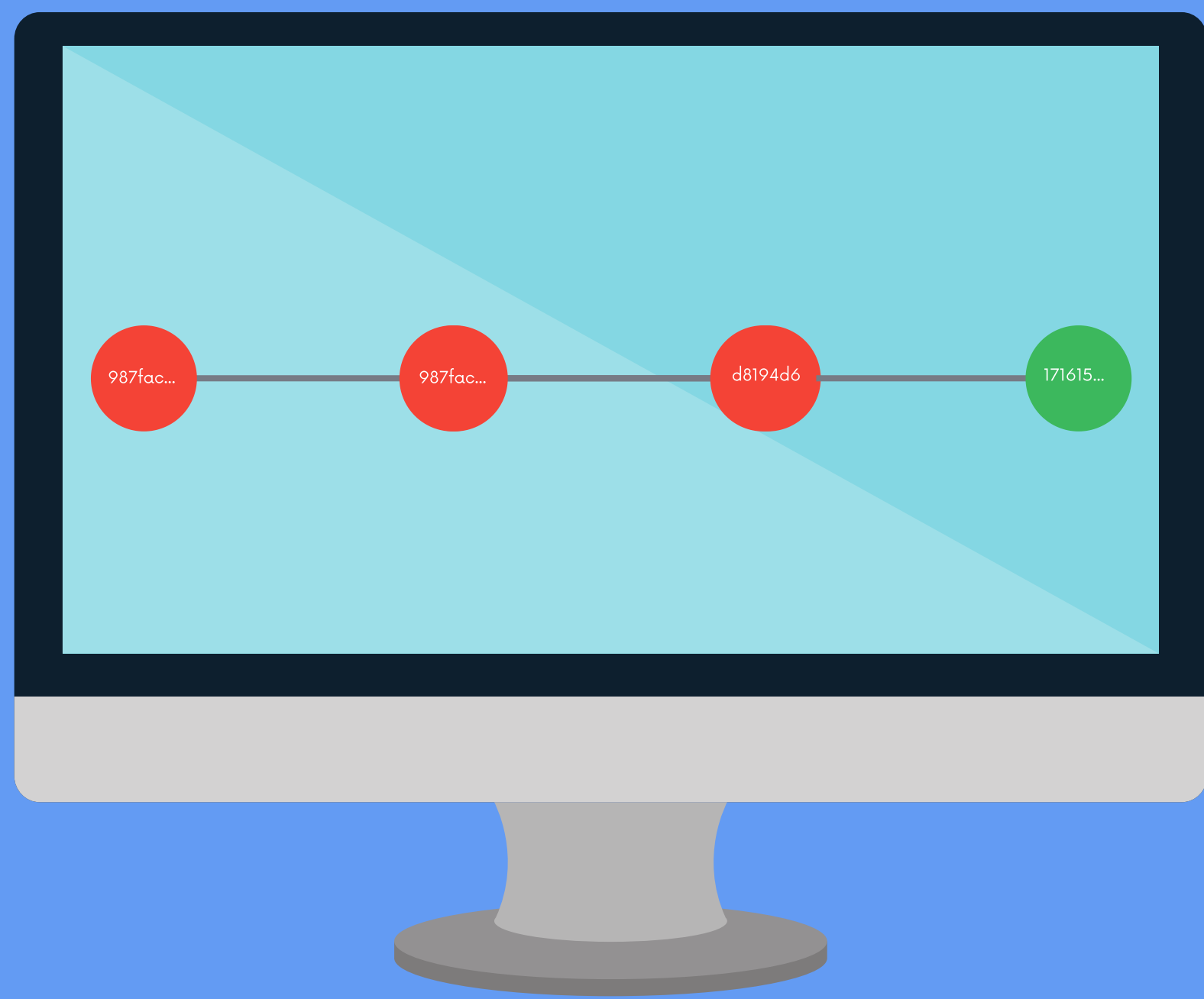
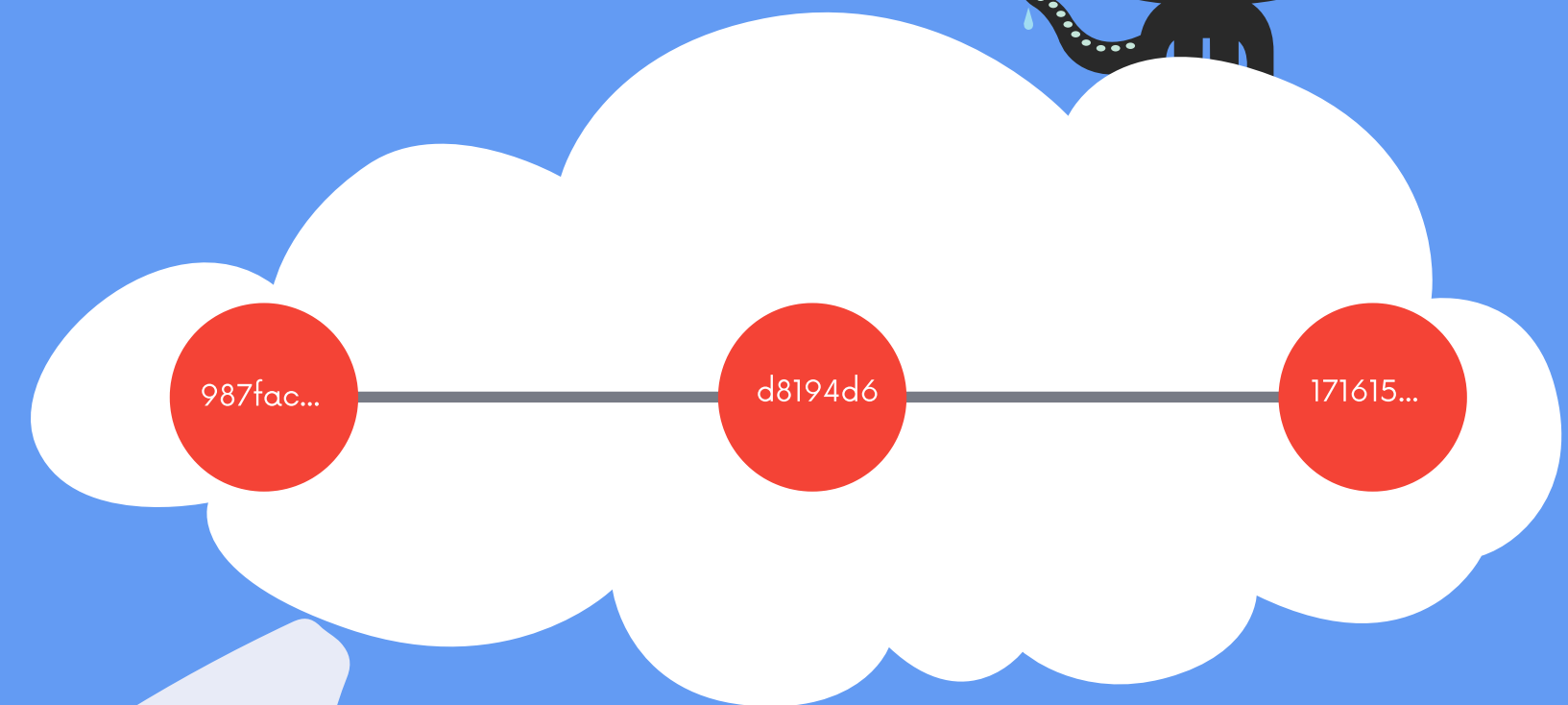
# Github

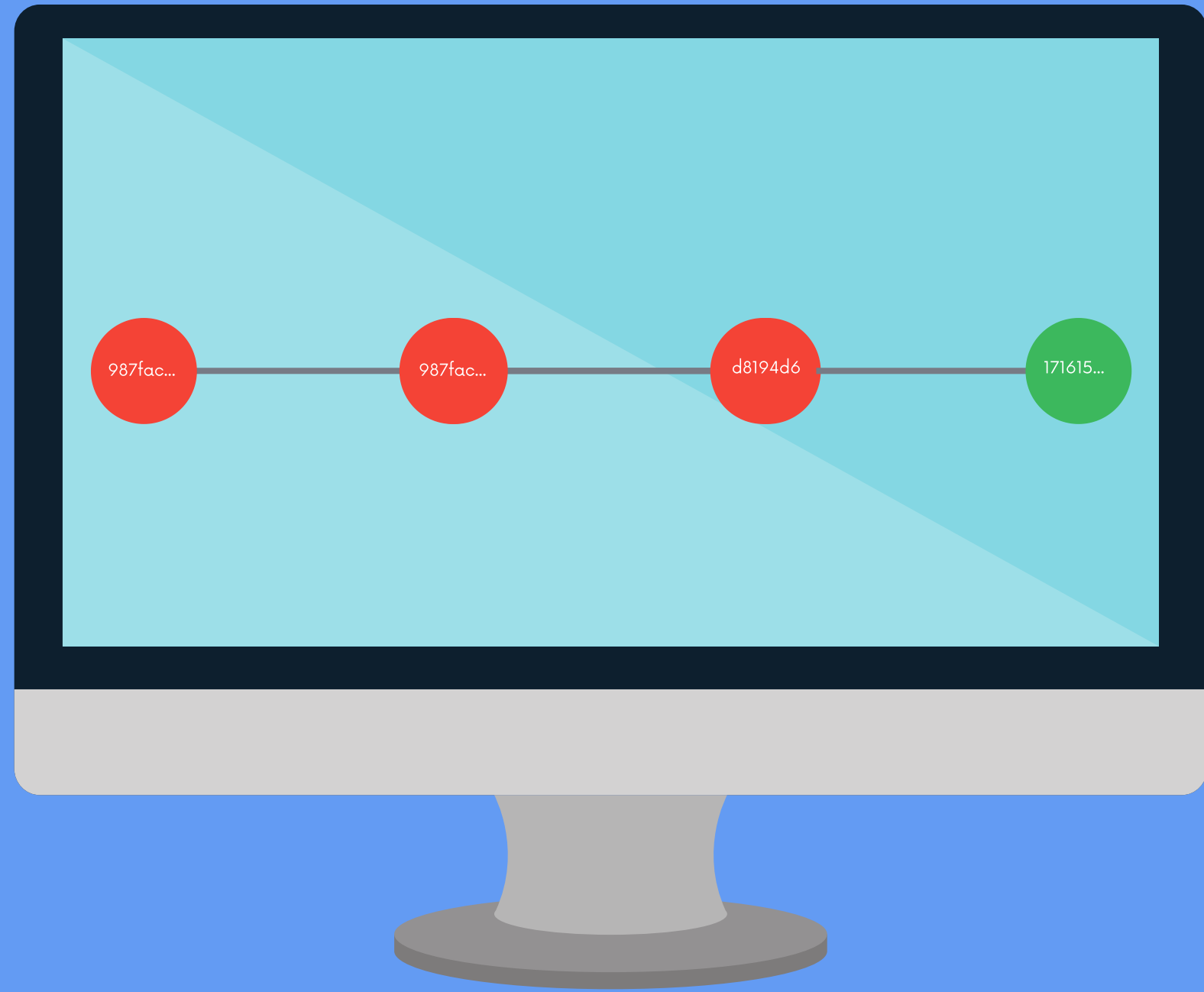
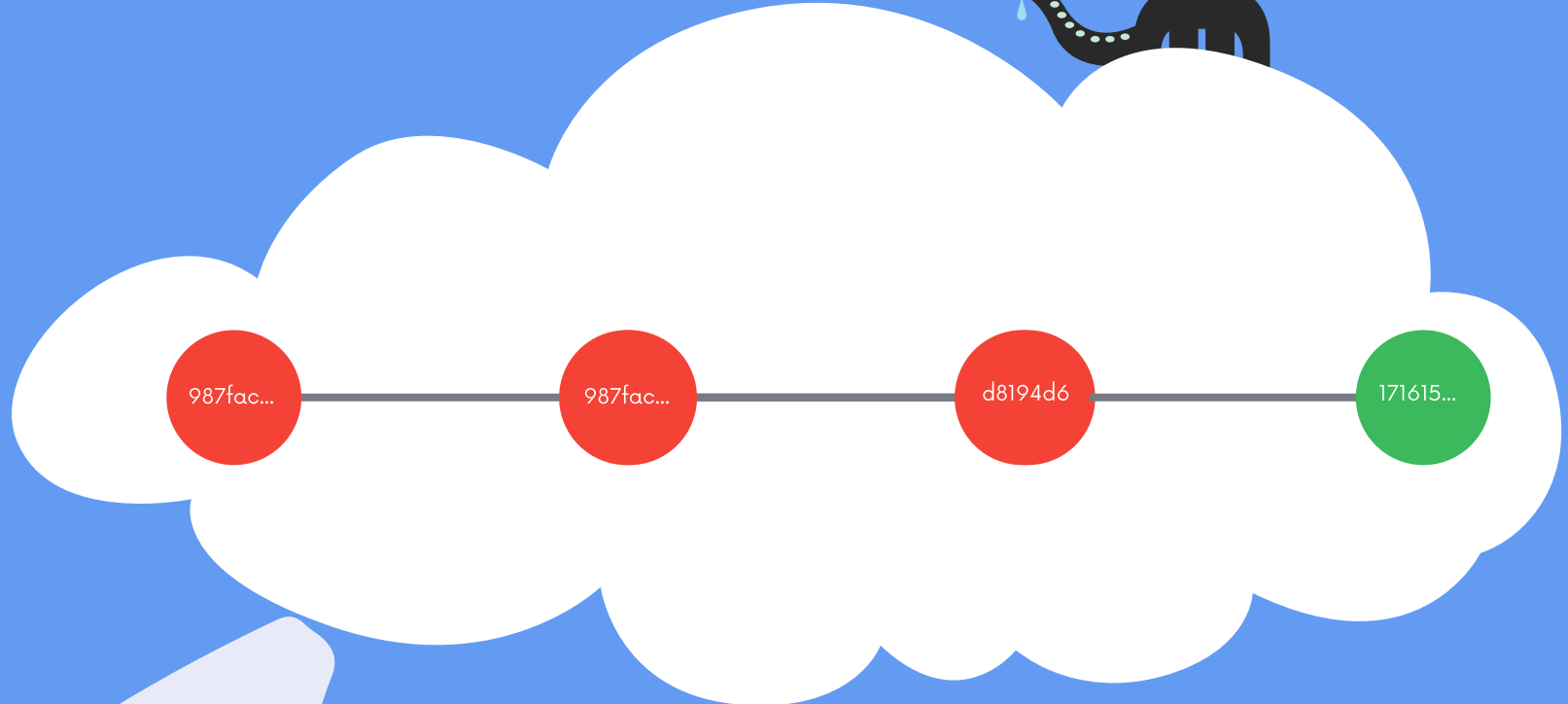
Github is a service that hosts Git repositories in the cloud and makes it easier to collaborate with other people. You do need to sign up for an account to use Github. It's an online place to share work that is done using Git.





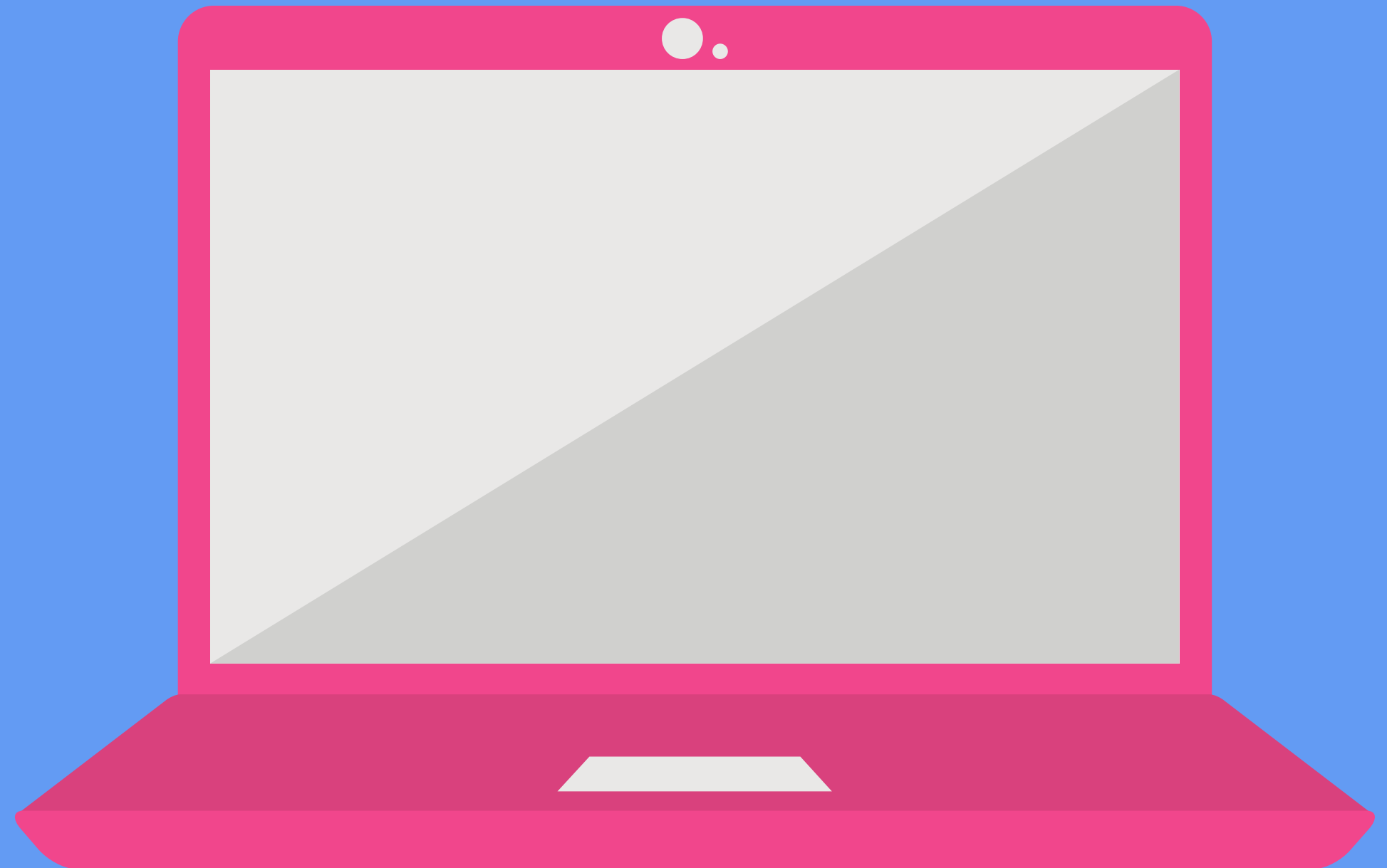


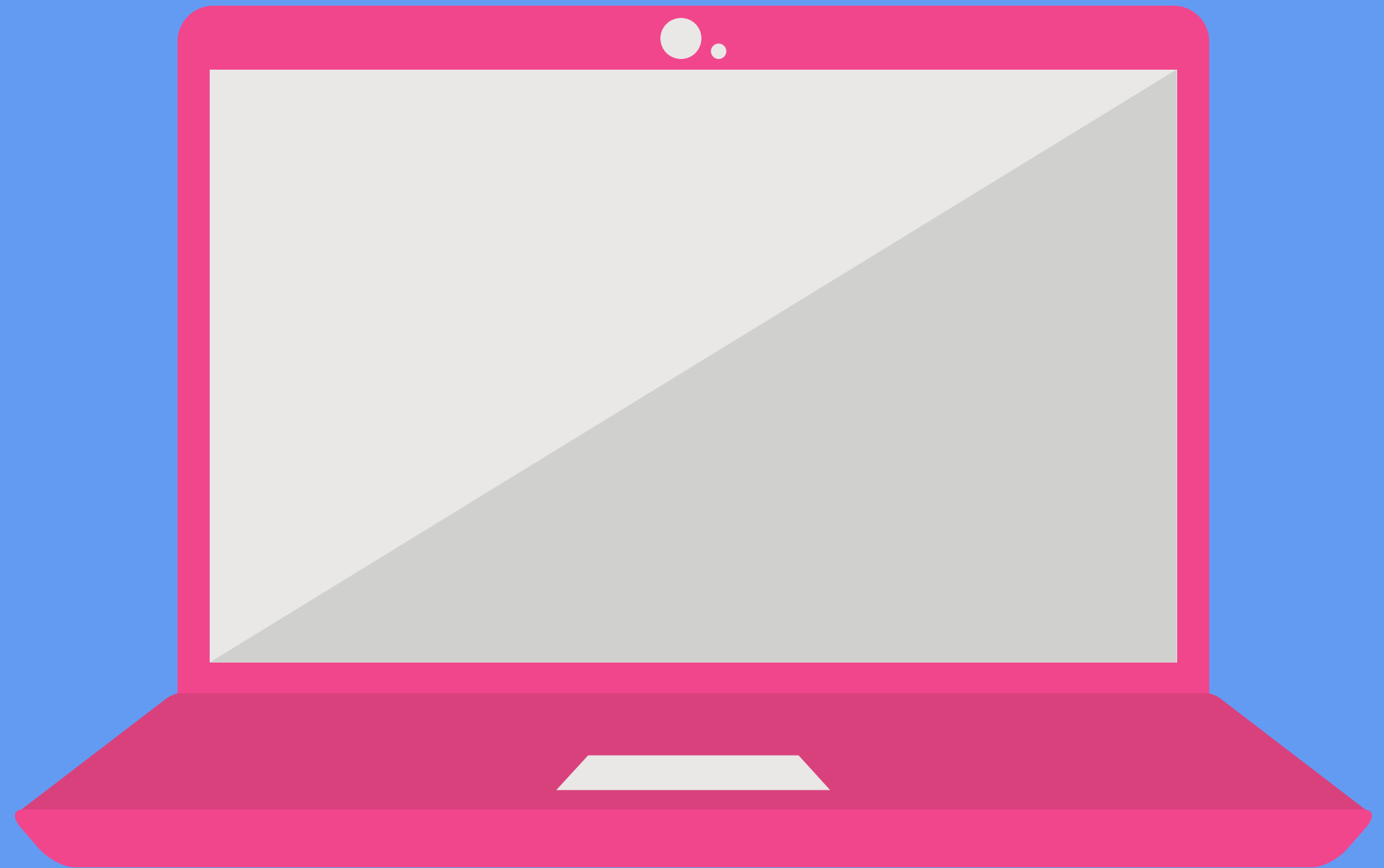
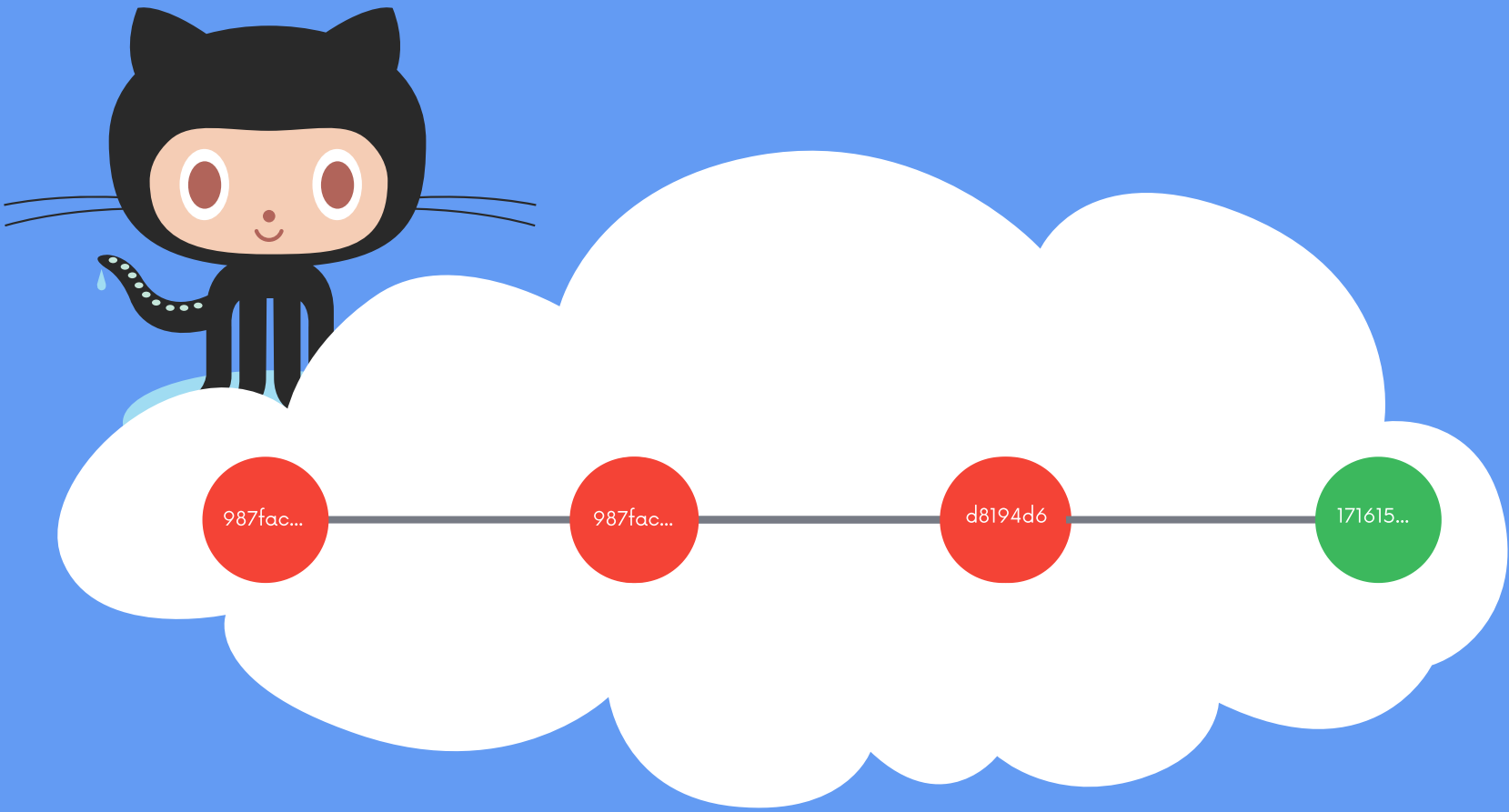


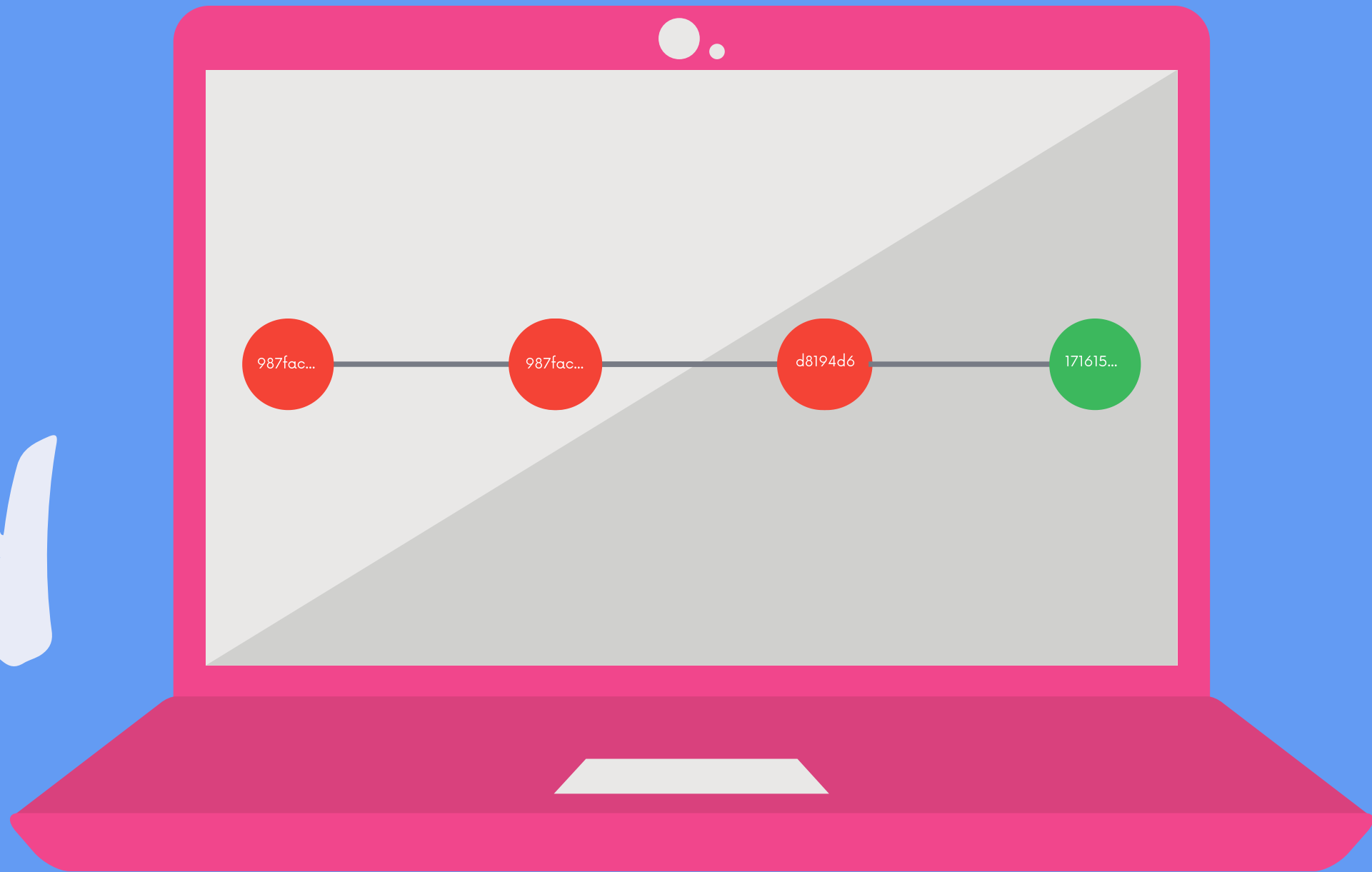
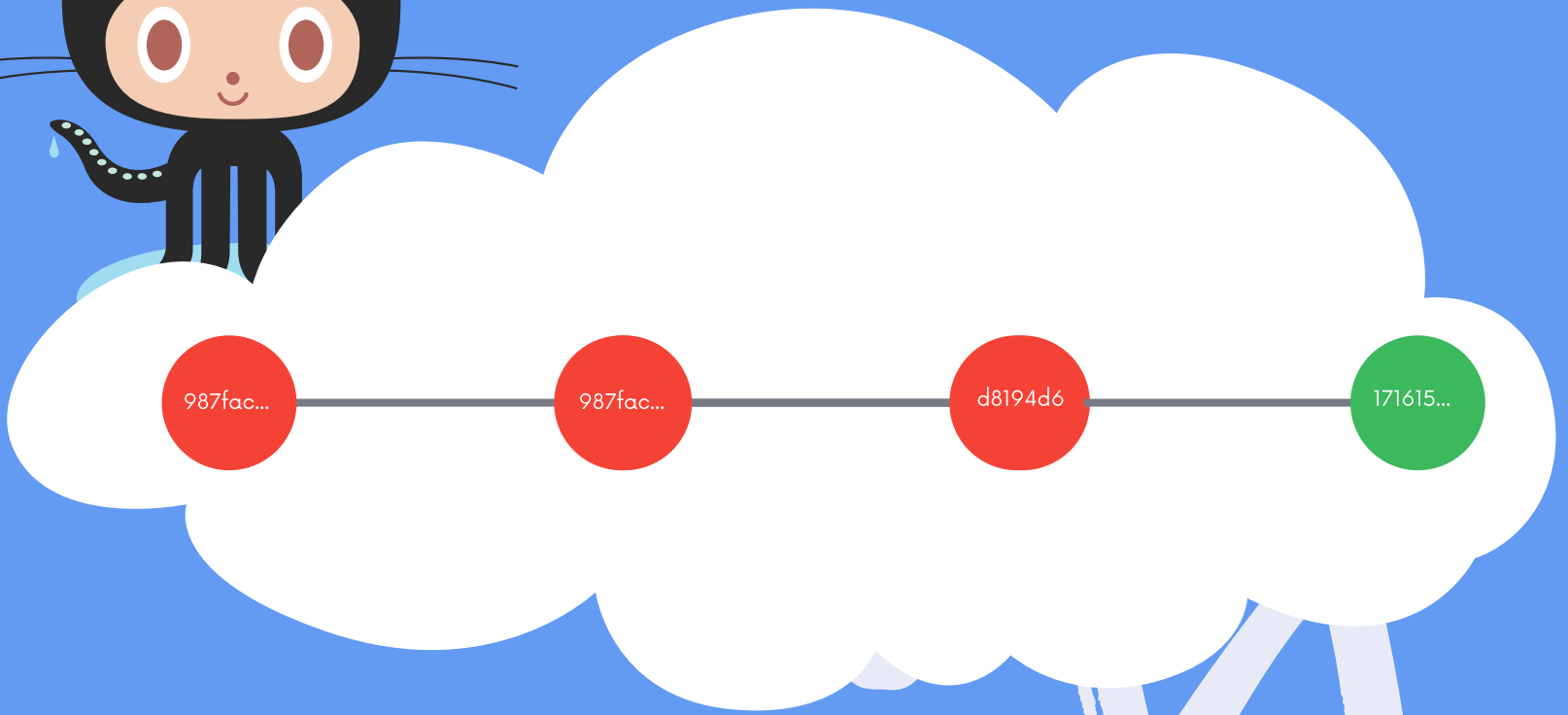


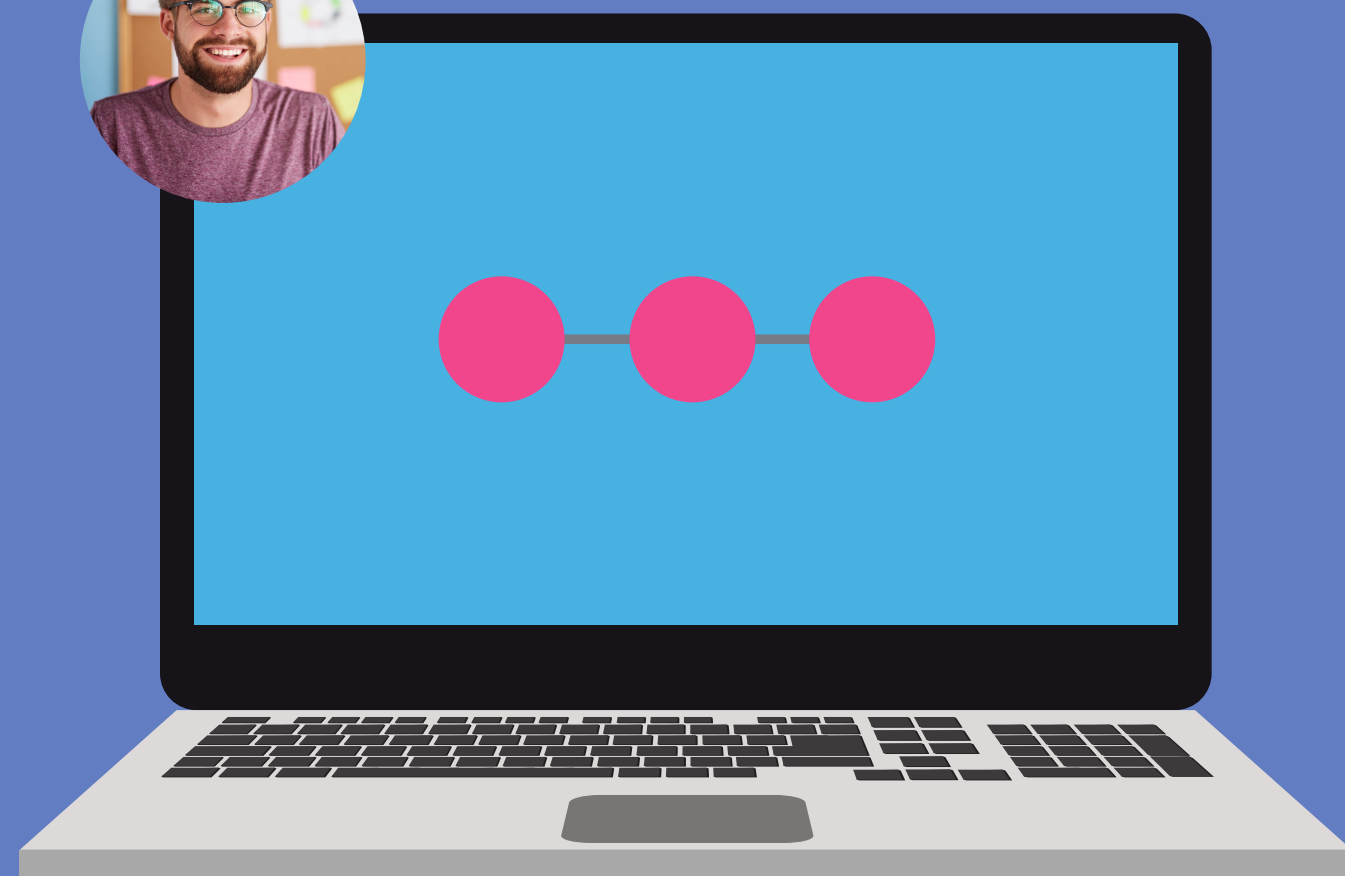
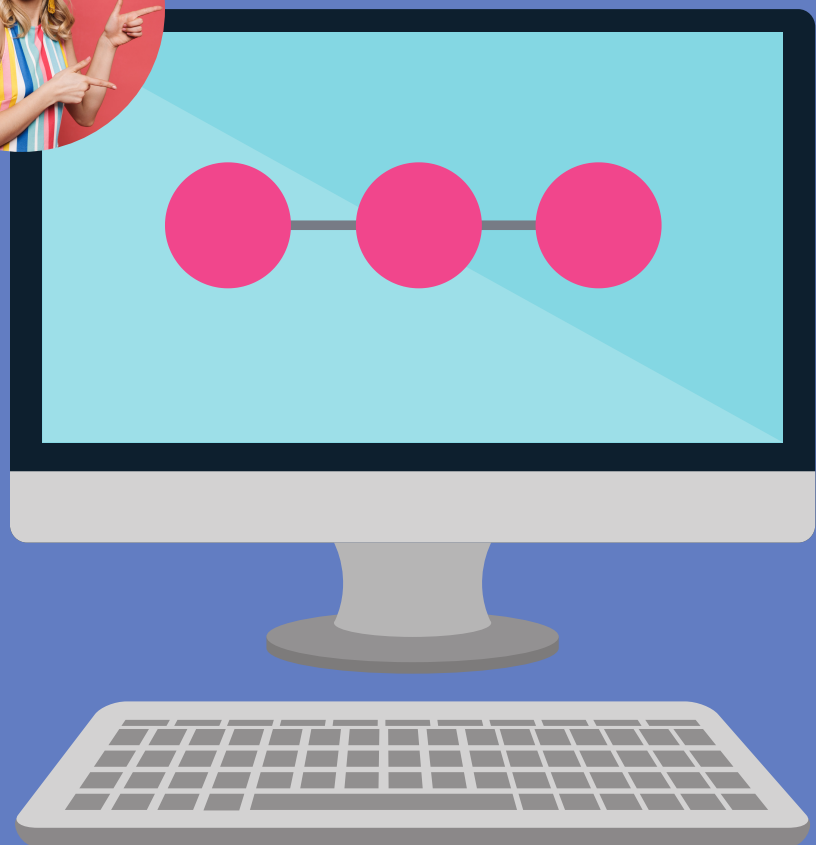
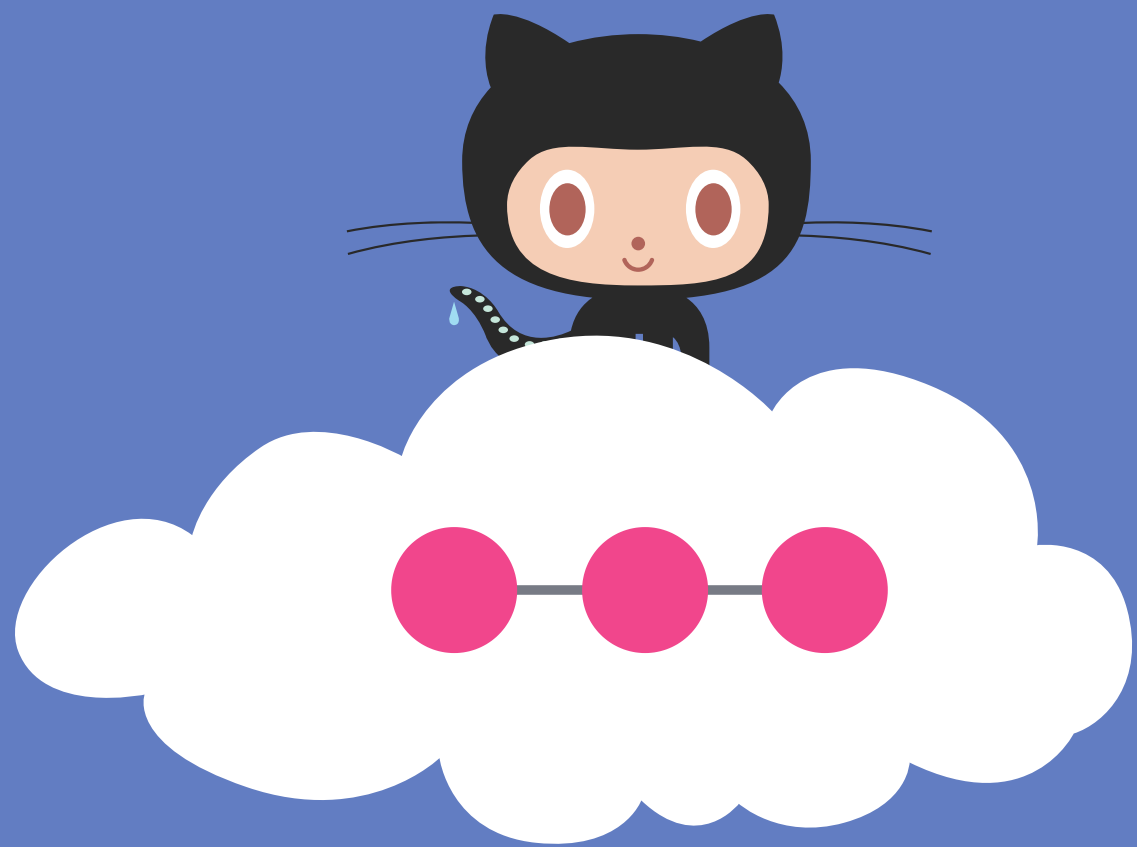


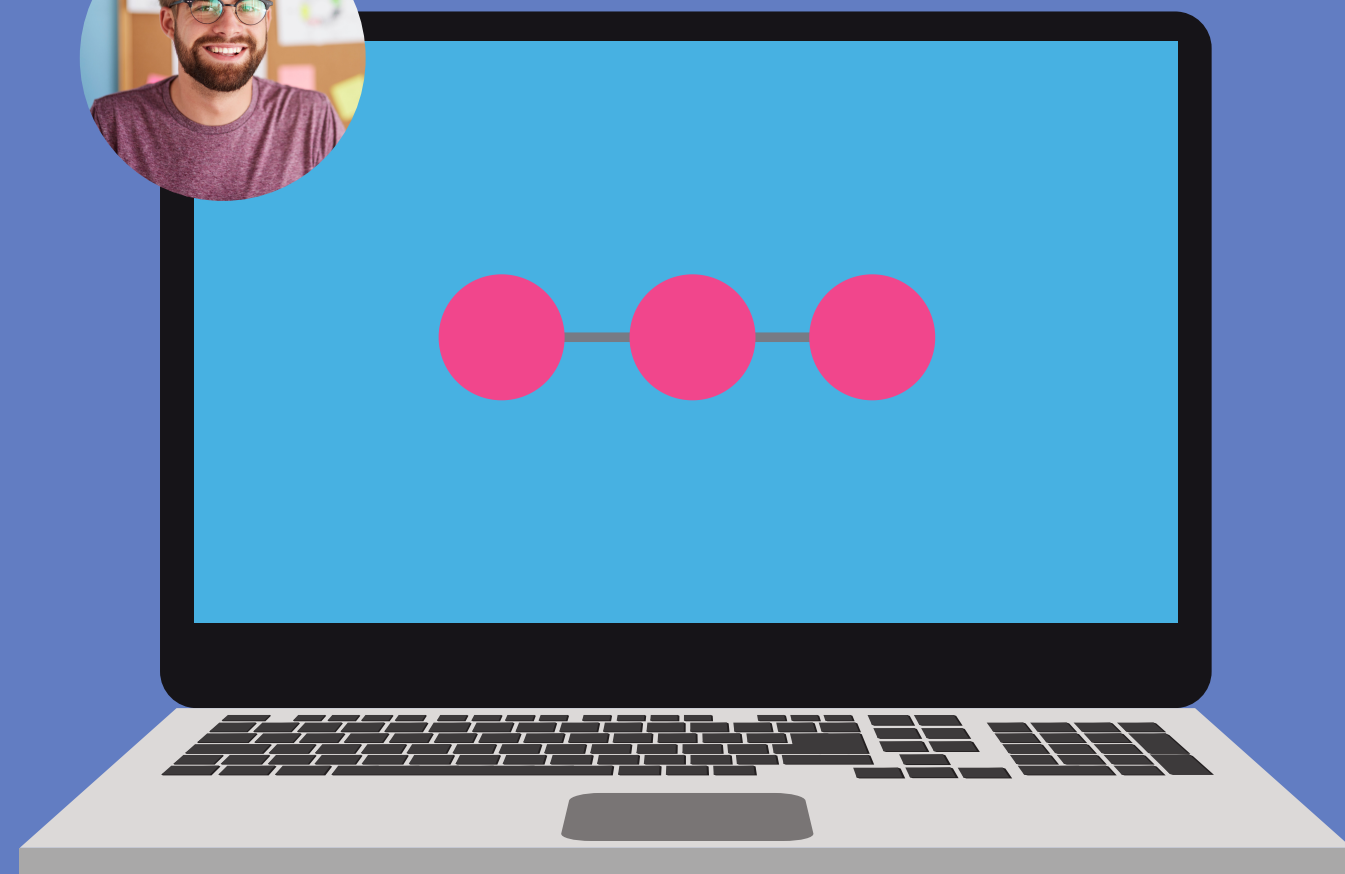
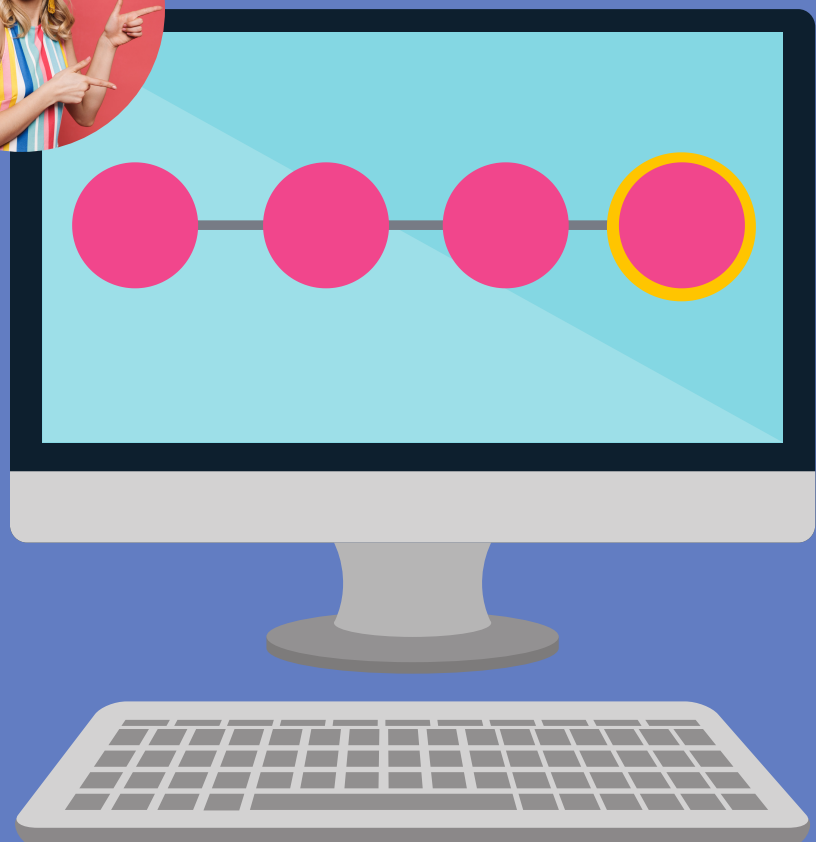
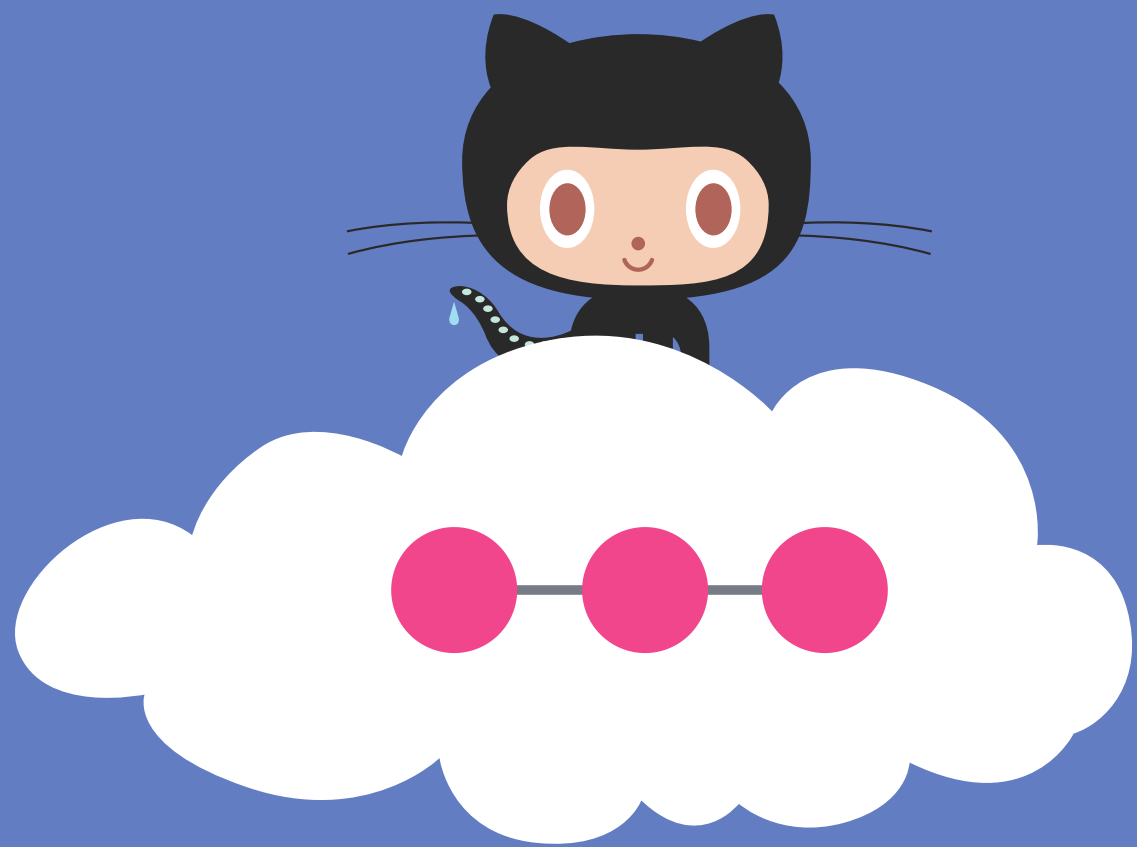
I GOT A  
NEW LAPTOP!  
Now I need my code

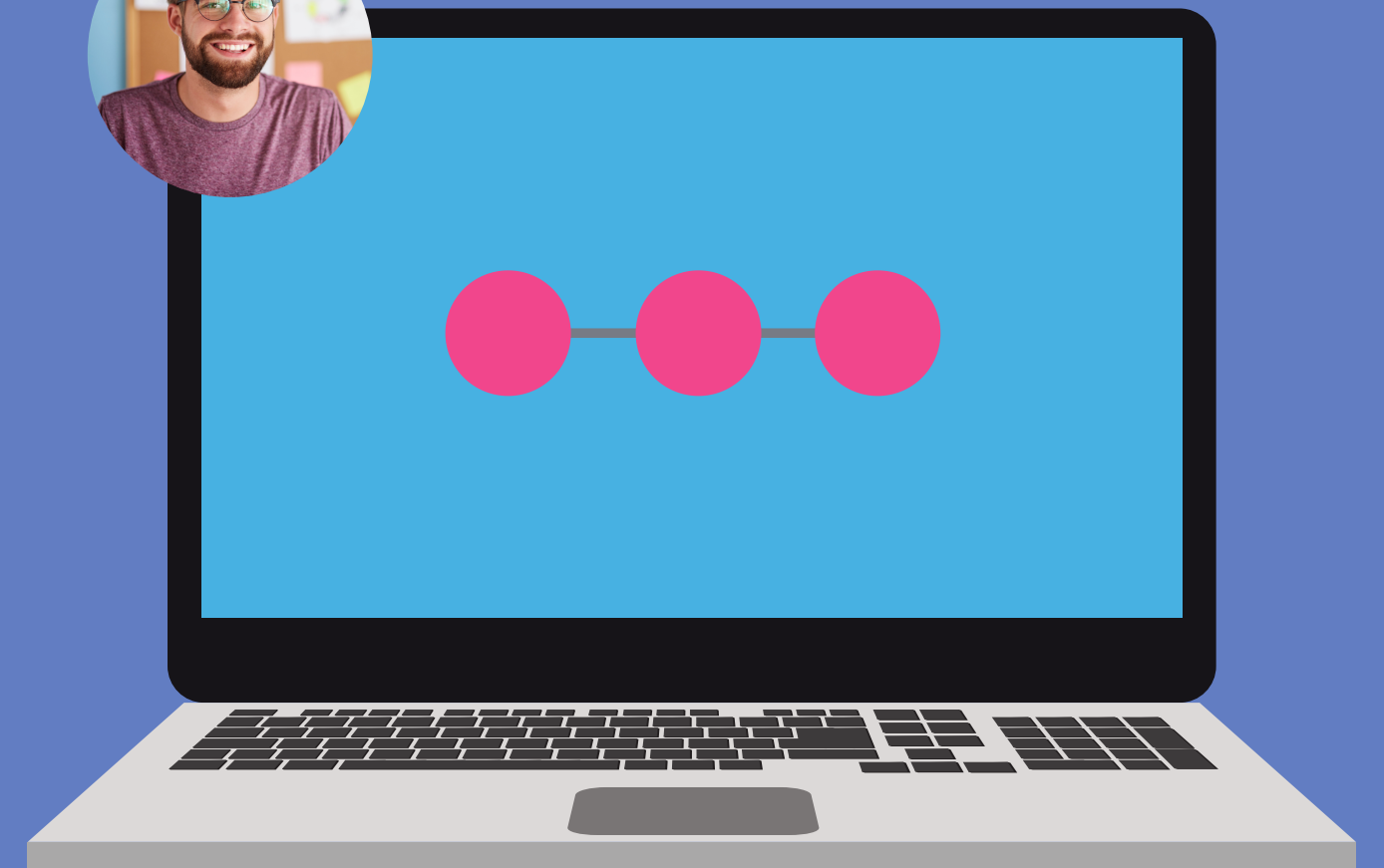
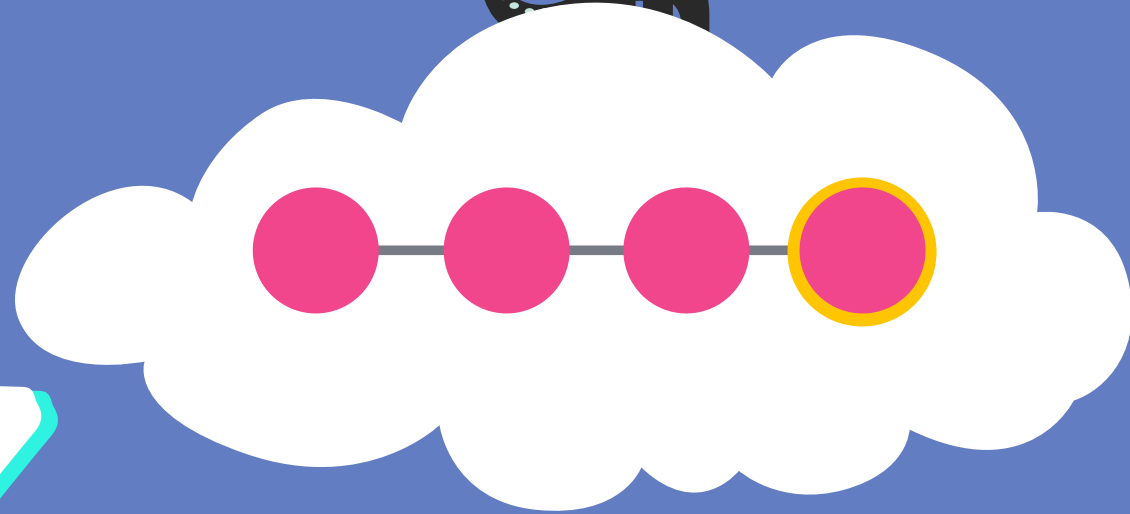


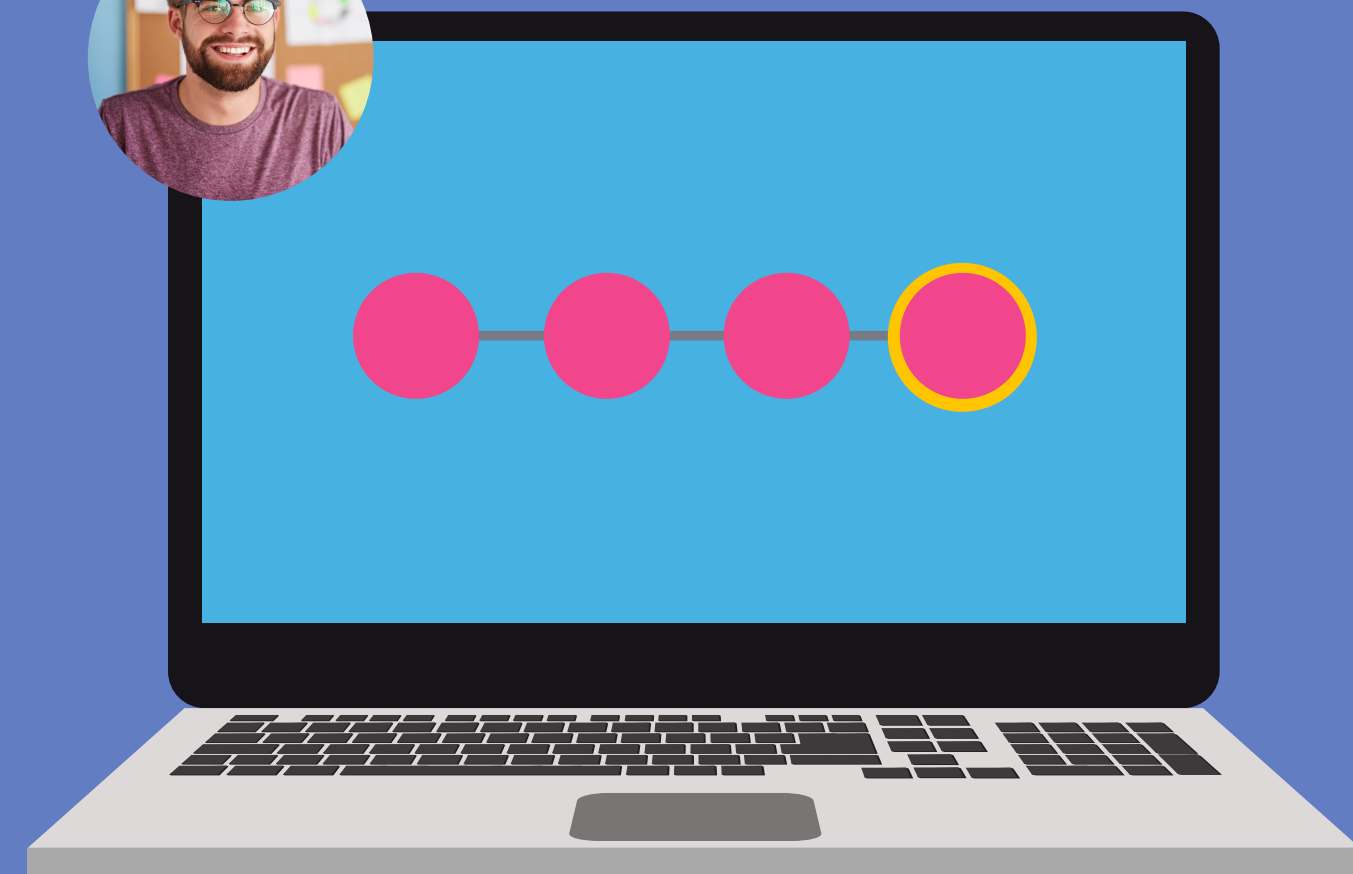
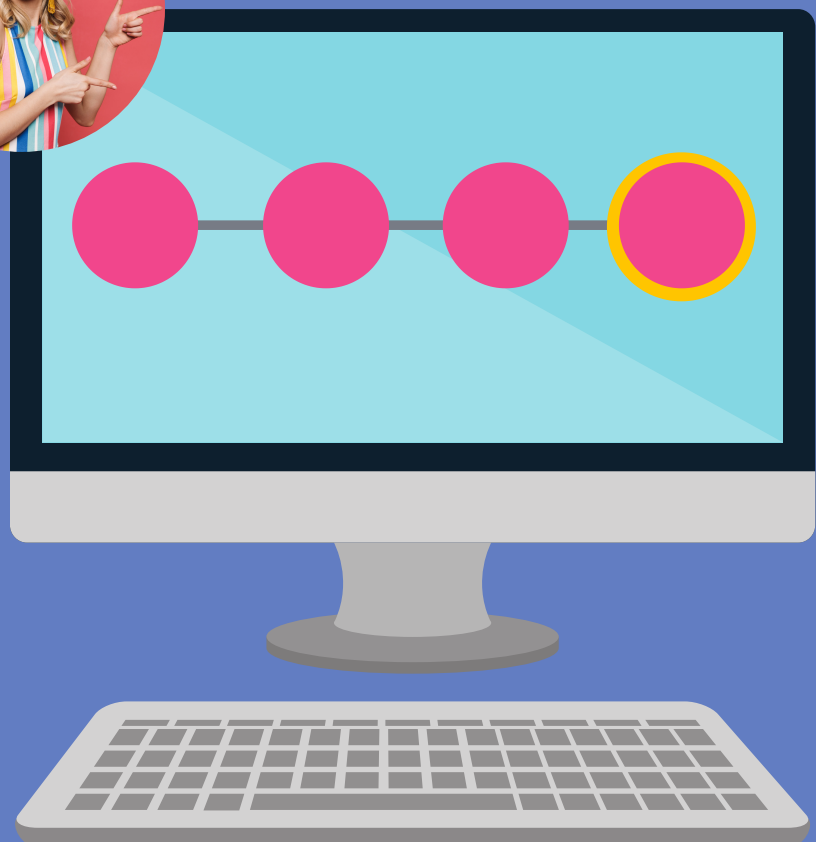
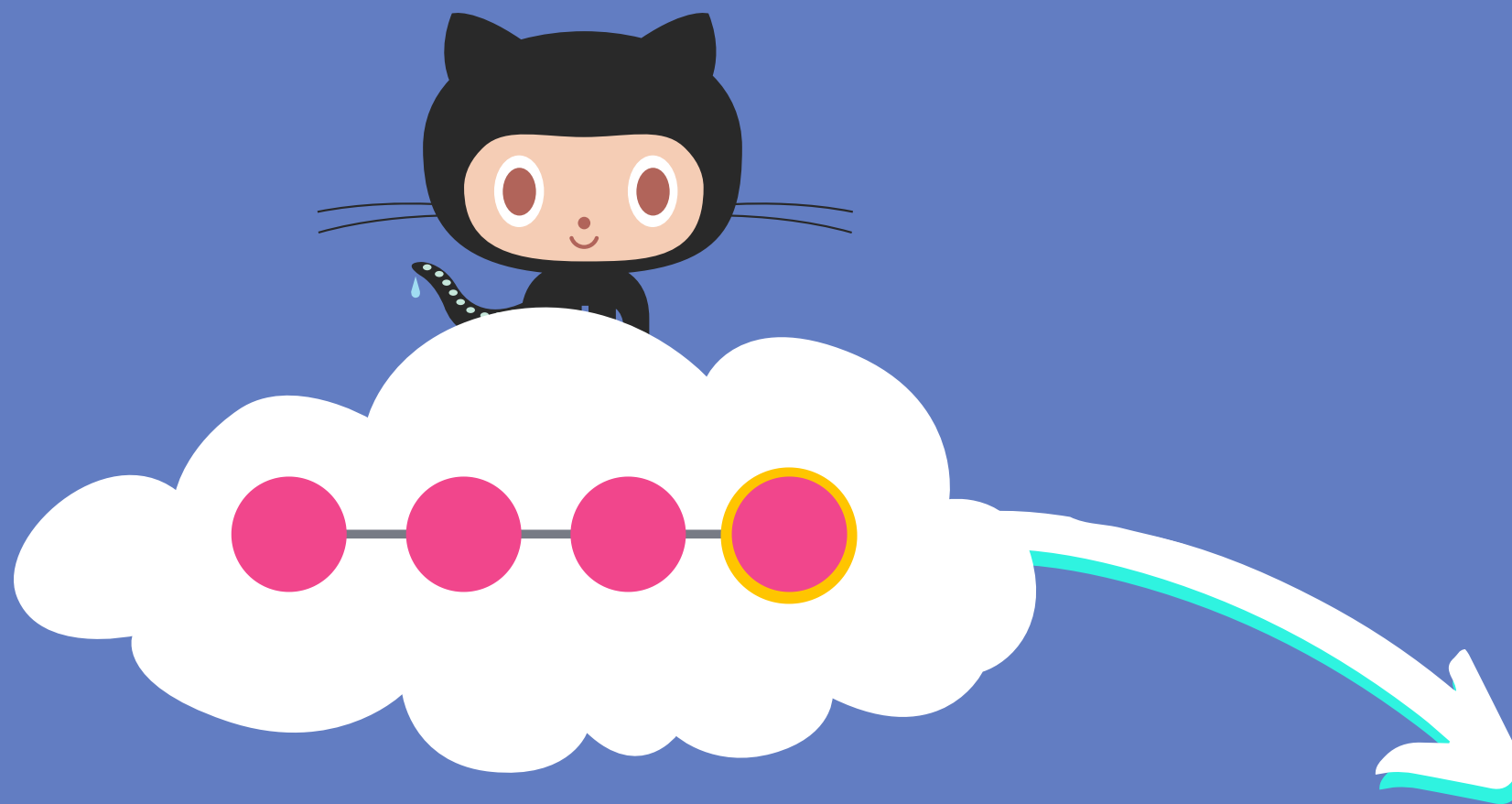


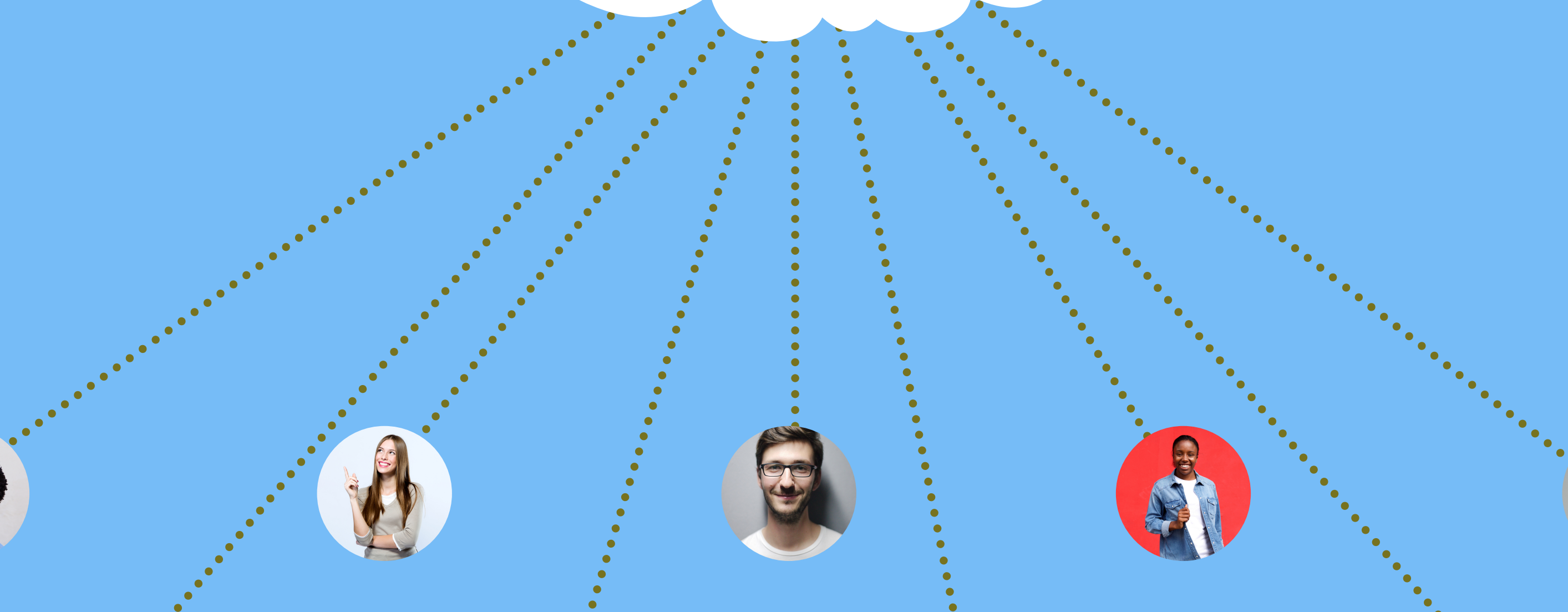
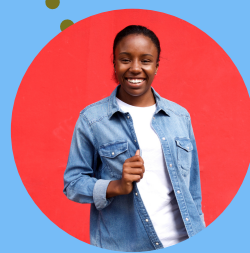
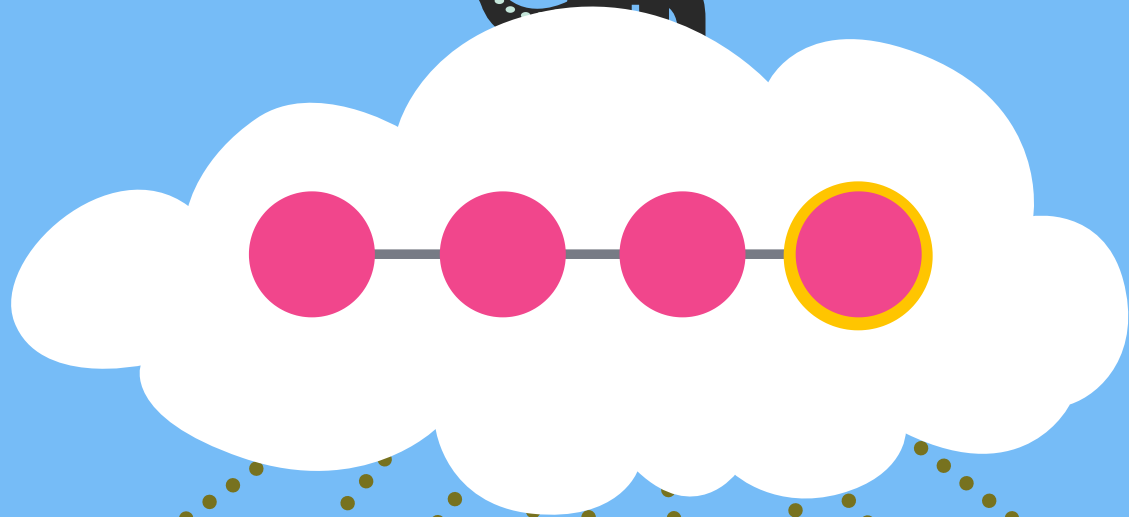












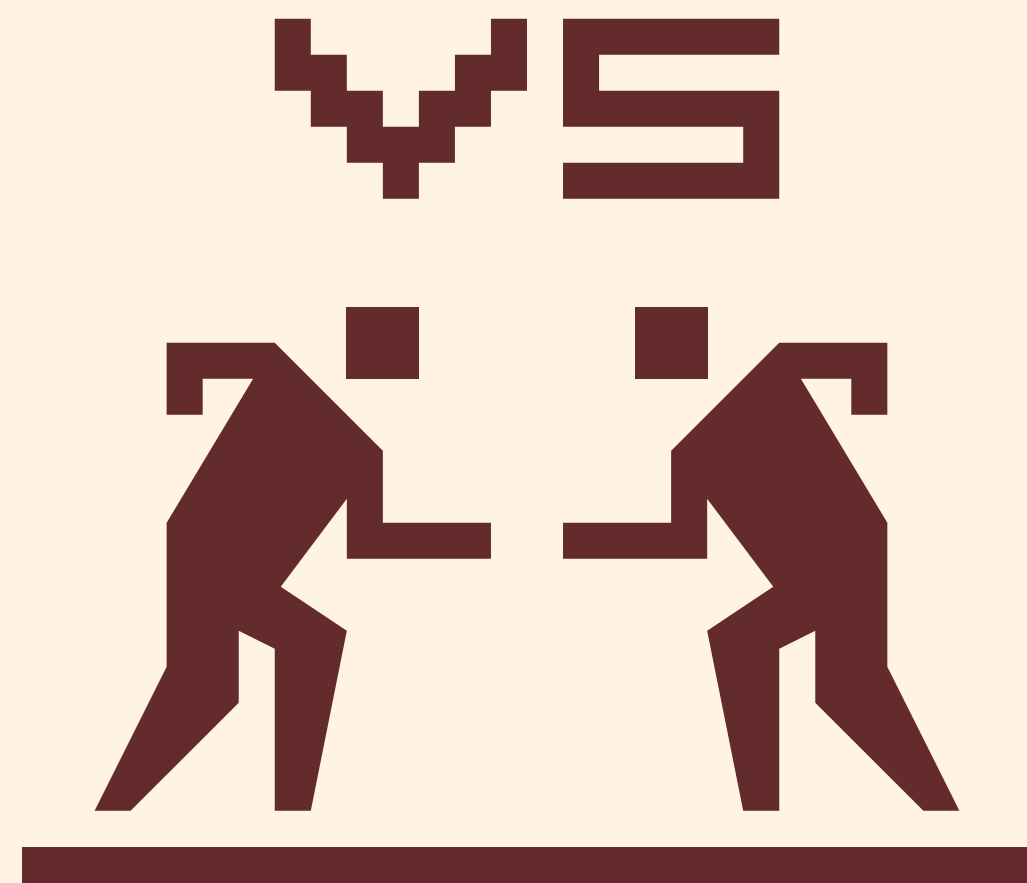




# Github is not your only option...

There are tons of competing tools that provide similar hosting and collaboration features, including GitLab, BitBucket, and Gerrit.

With that said....





# It's very popular!

Founded in 2008, Github is now **the world's largest host of source code**. In early 2020, Github reported having over 40 million users and over 190 million repositories on the platform.





# It's Free!

Github offers its basic services for free! While Github does offer paid Team and Enterprise tiers, the basic Free tier allows for unlimited public and private repos, unlimited collaborators, and more!





# Why You Should Use Github

(or at least know how to use it)





# Collaboration

If you ever plan on working on a project with at least one other person, Github will make your life easier! Whether you're building a hobby project with your friend or you're collaborating with the entire world, Github is essential!





# Open Source Projects

Today Github is THE home of open source projects on the Internet. Projects ranging from React to Swift are hosted on Github.

If you plan on contributing to open source projects, you'll need to get comfortable working with Github.





# Exposure

Your Github profile showcases your own projects and contributions to others' projects. It can act as a sort of resumé that many employers will consult in the hiring process. Additionally, you can gain some clout on the platform for creating or contributing to popular projects.





# Stay Up To Date

Being active on Github is the best way to stay up to date with the projects and tools you rely on. Learn about upcoming changes and the decisions/debate behind them.



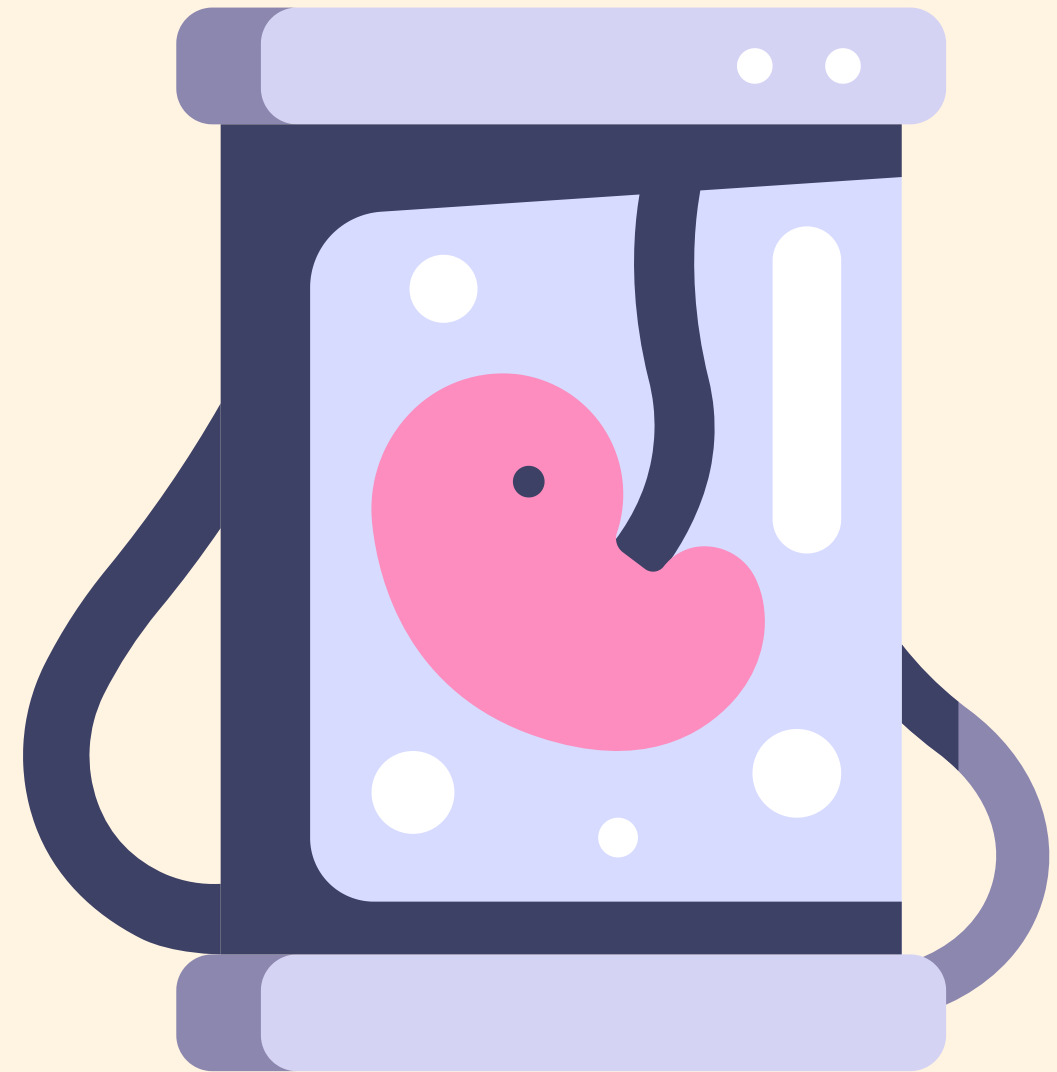




# Cloning

So far we've created our own Git repositories from scratch, but often we want to get a **local copy of an existing repository** instead.

To do this, we can clone a remote repository hosted on Github or similar websites. All we need is a URL that we can tell Git to clone for use.





# git clone

To clone a repo, simply run `git clone <url>`.

Git will retrieve all the files associated with the repository and will copy them to your local machine.

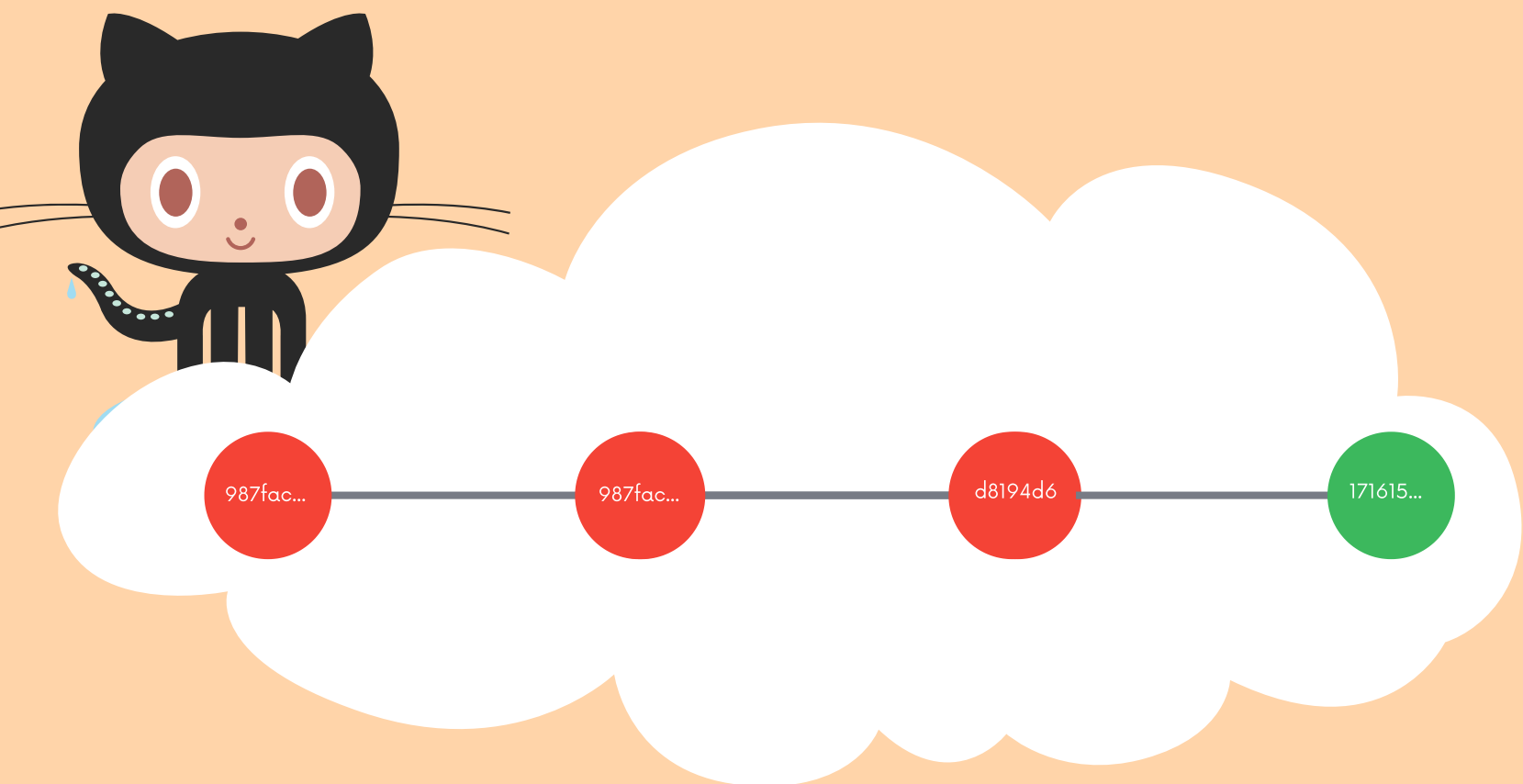
In addition, Git initializes a new repository on your machine, giving you access to the full Git history of the cloned project.

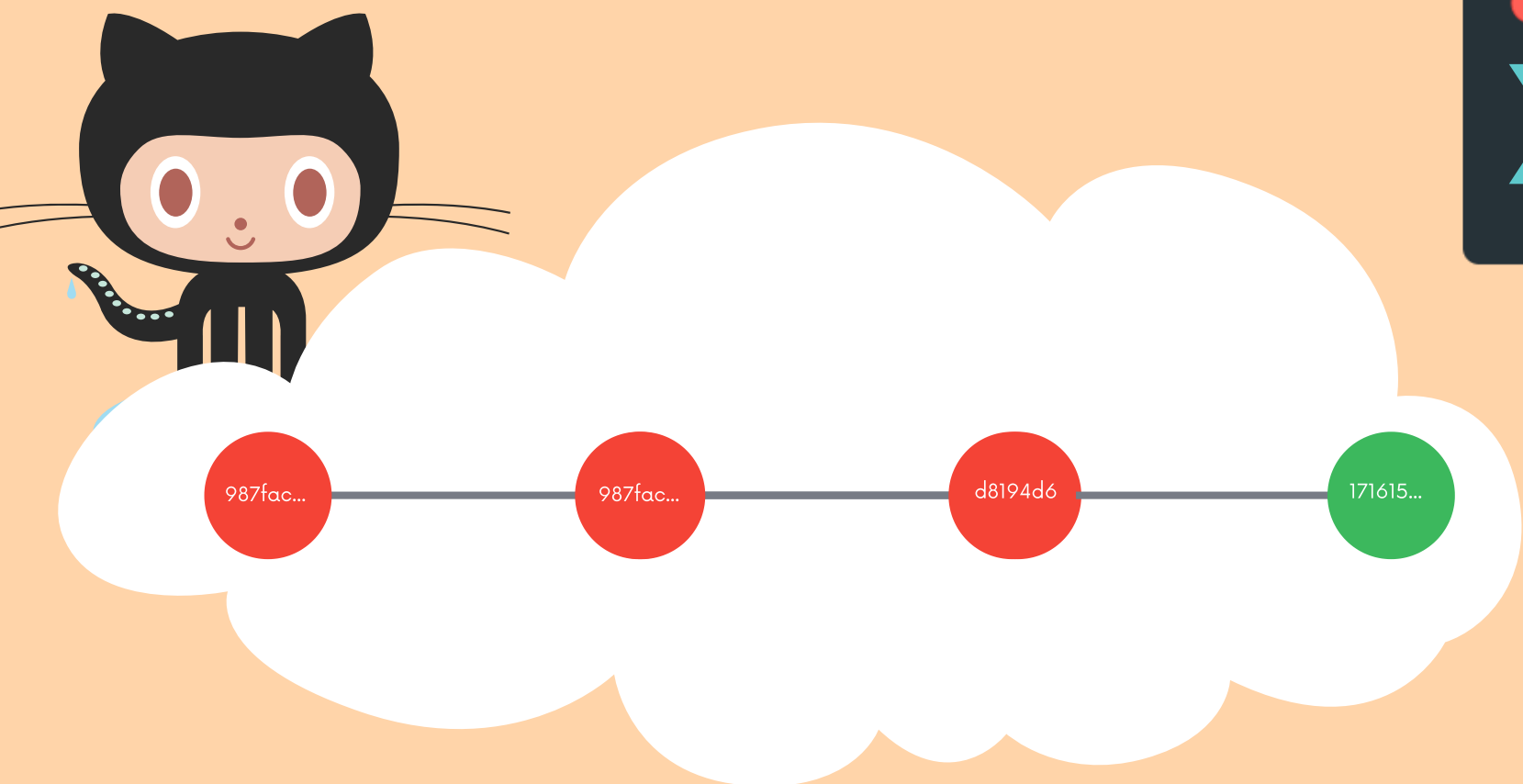
A dark-themed terminal window with three colored window control buttons (red, yellow, green) in the top-left corner. A light blue prompt character is followed by the text `git clone <url>` in white monospace font.

```
> git clone <url>
```

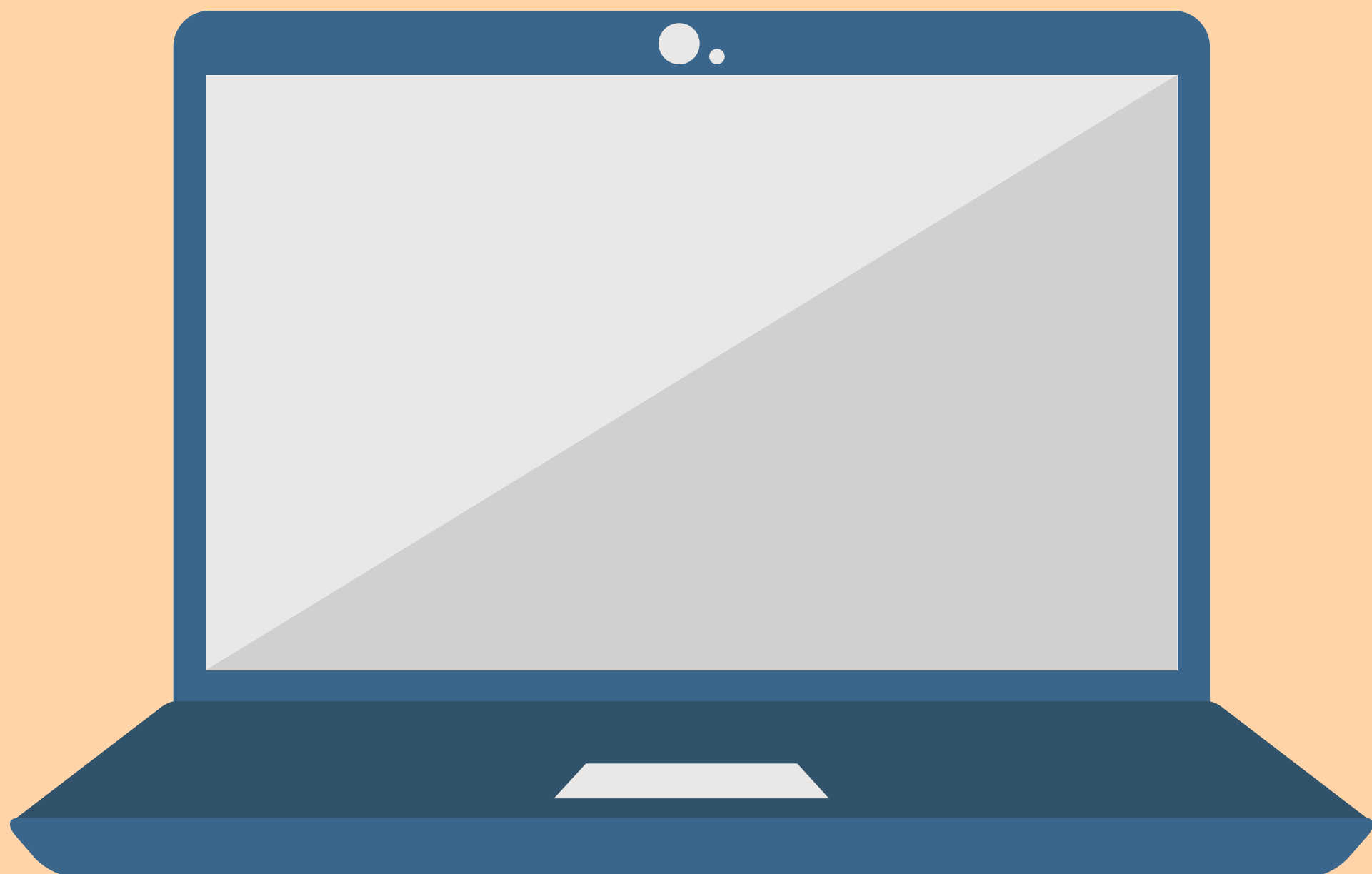
Make sure you are not inside of a repo when you clone!

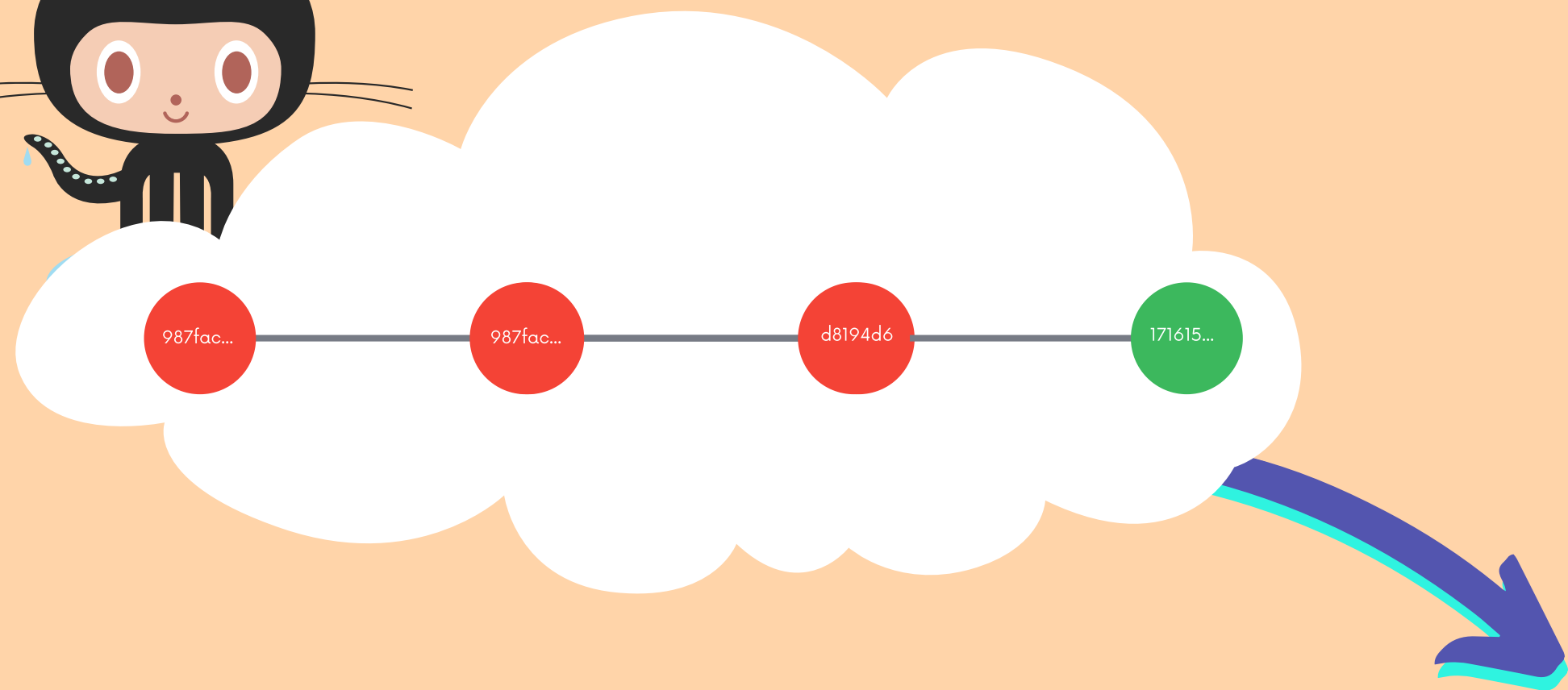






```
> git clone https://github.com/blah
```





WE NOW HAVE A LOCAL  
COPY OF THE REPO!





# Permissions?

Anyone can clone a repository from Github, provided the repo is public. You do not need to be an owner or collaborator to clone the repo locally to your machine. You just need the URL from Github.

Pushing up your own changes to the Github repo...that's another story entirely! You need permission to do that!





# We are not limited to Github Repos!

`git clone` is a standard git command.

It is NOT tied specifically to Github. We can use it to clone repositories that are hosted anywhere! It just happens that most of the hosted repos are on Github these days.



**Let's Register!**





# Configuring SSH Keys





# SSH Keys

You need to be authenticated on Github to do certain operations, like pushing up code from your local machine. Your terminal will prompt you every single time for your Github email and password, unless...

You generate and configure an SSH key! Once configured, you can connect to Github without having to supply your username/password.





# How Do I Get My Code On Github?





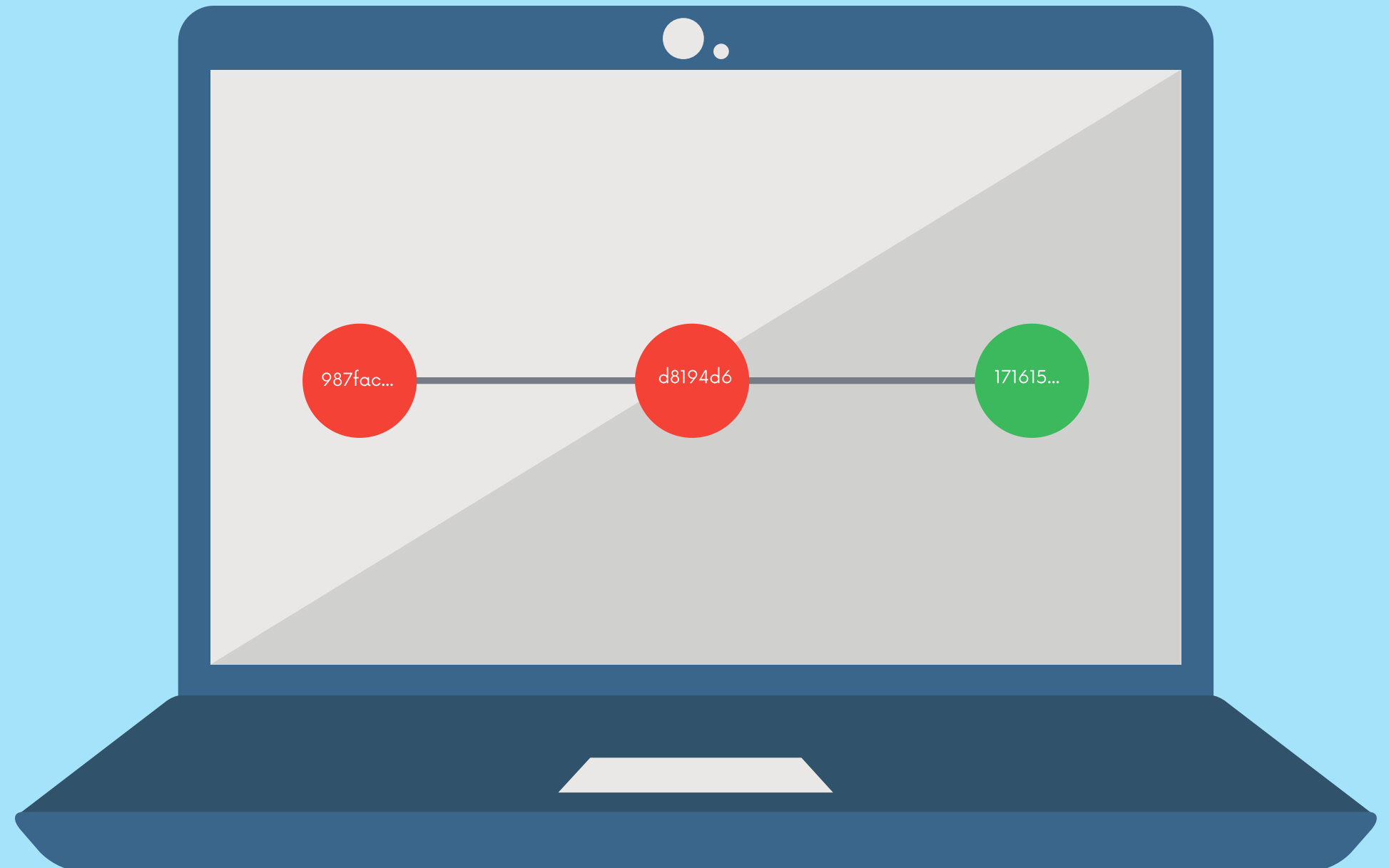
# Option 1: Existing Repo

If you already have an existing repo locally that you want to get on Github...

- Create a new repo on Github
- Connect your local repo (add a remote)
- Push up your changes to Github

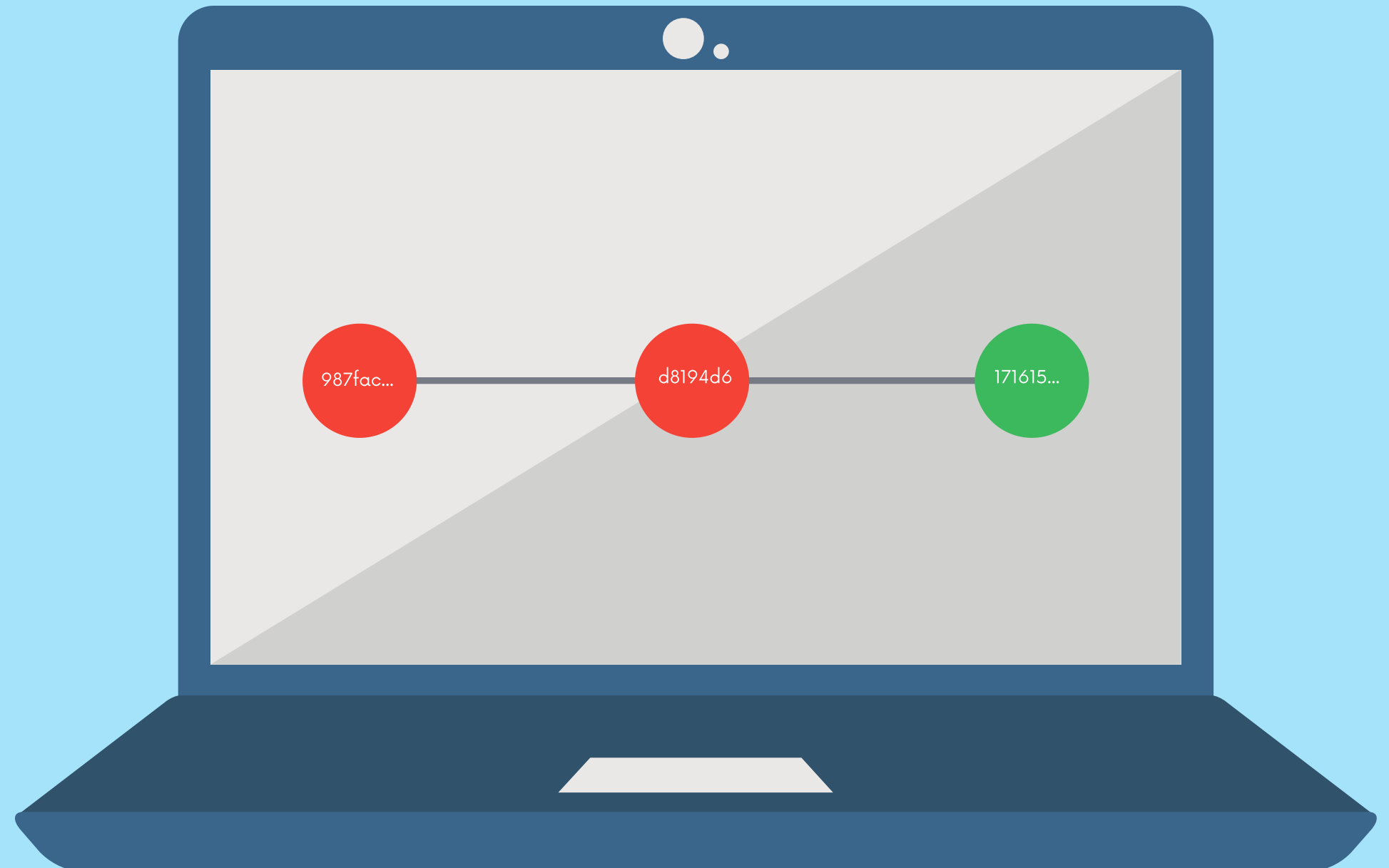


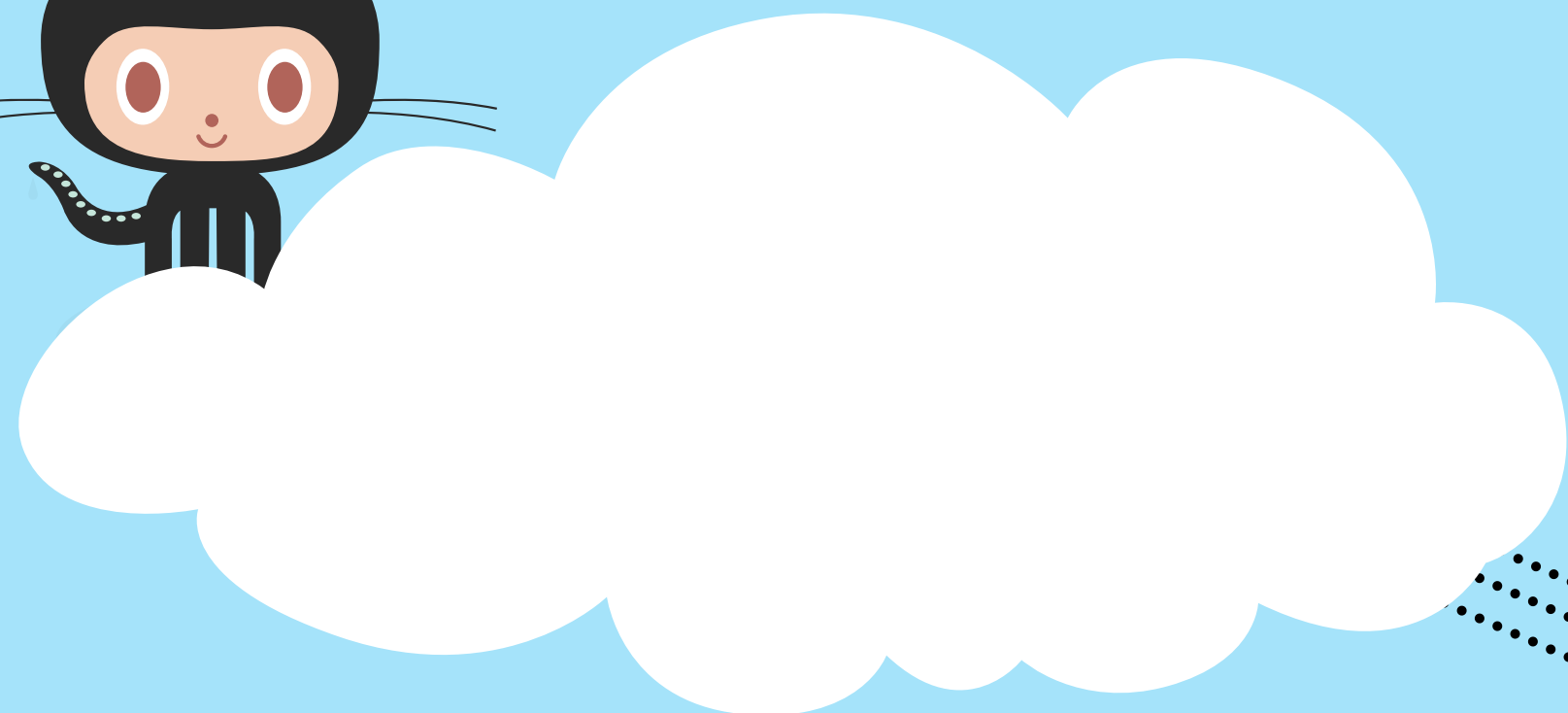
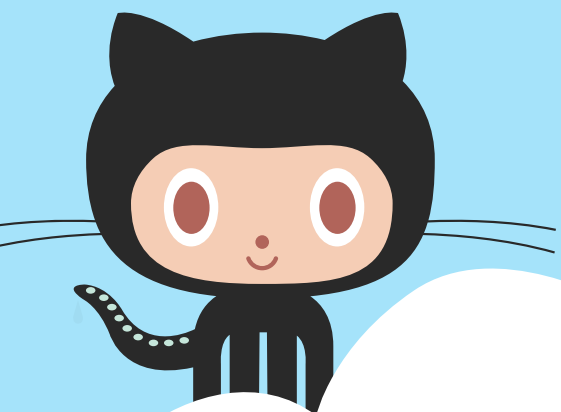
This lovely repo only exists on the laptop:



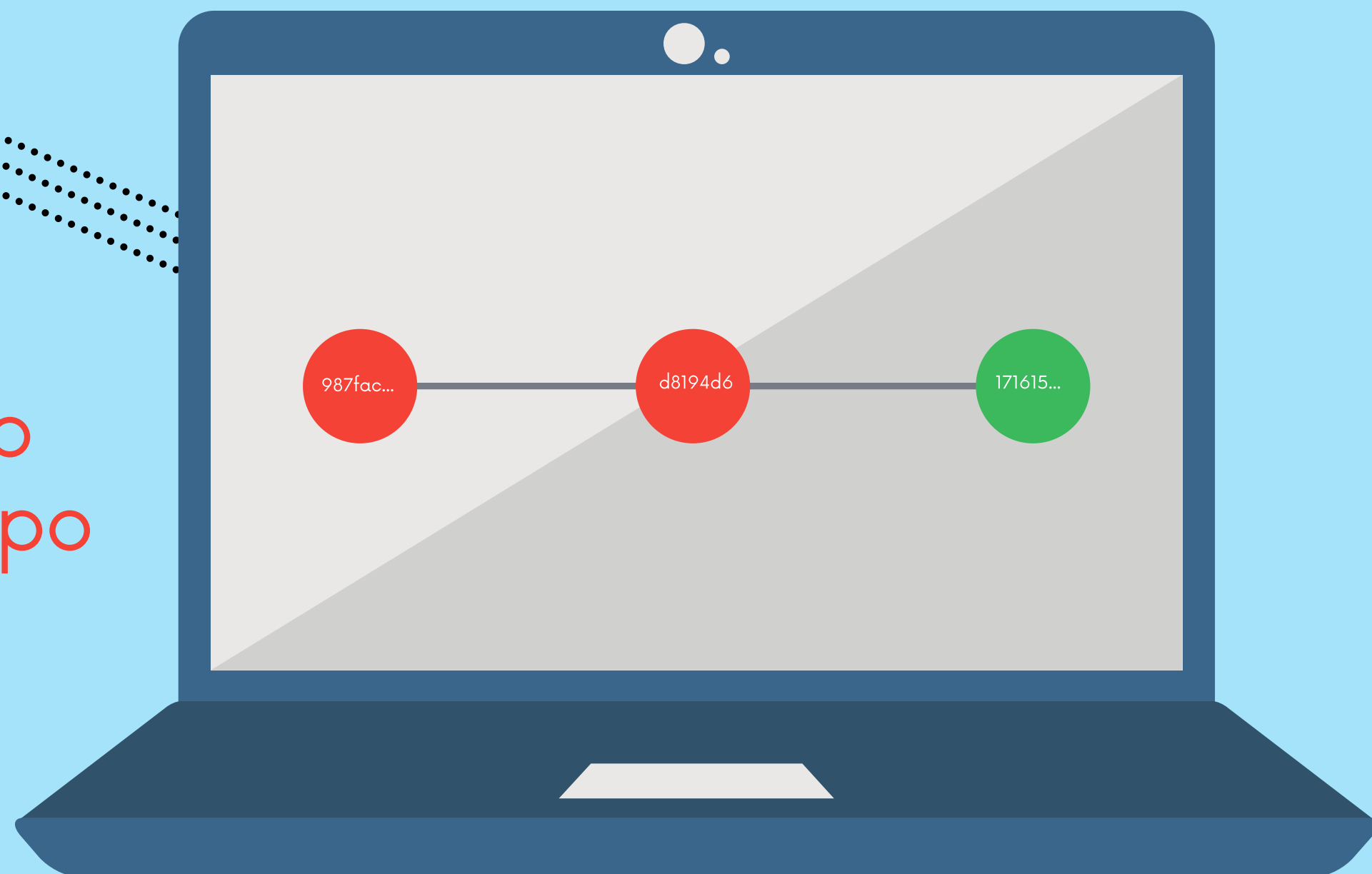


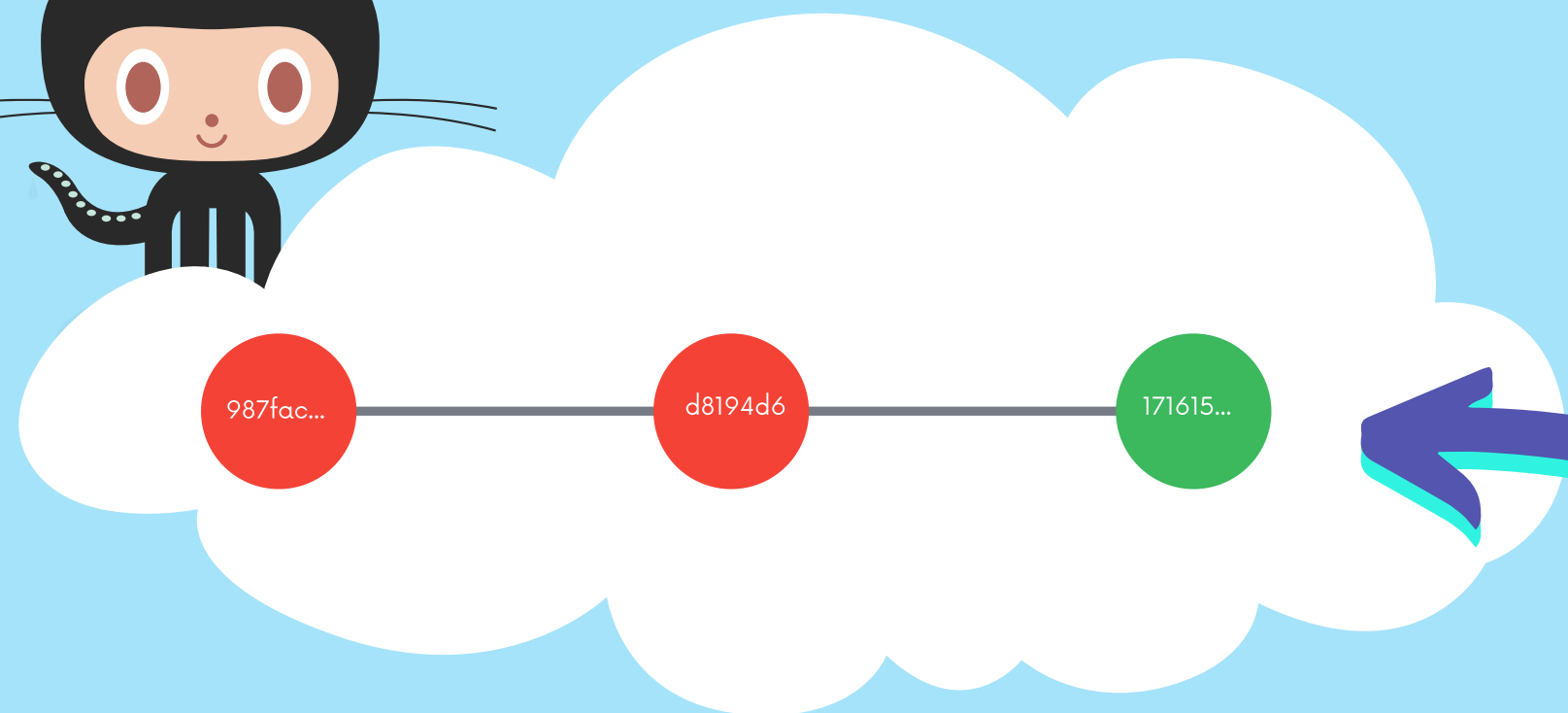
Create Empty  
Github Repo!



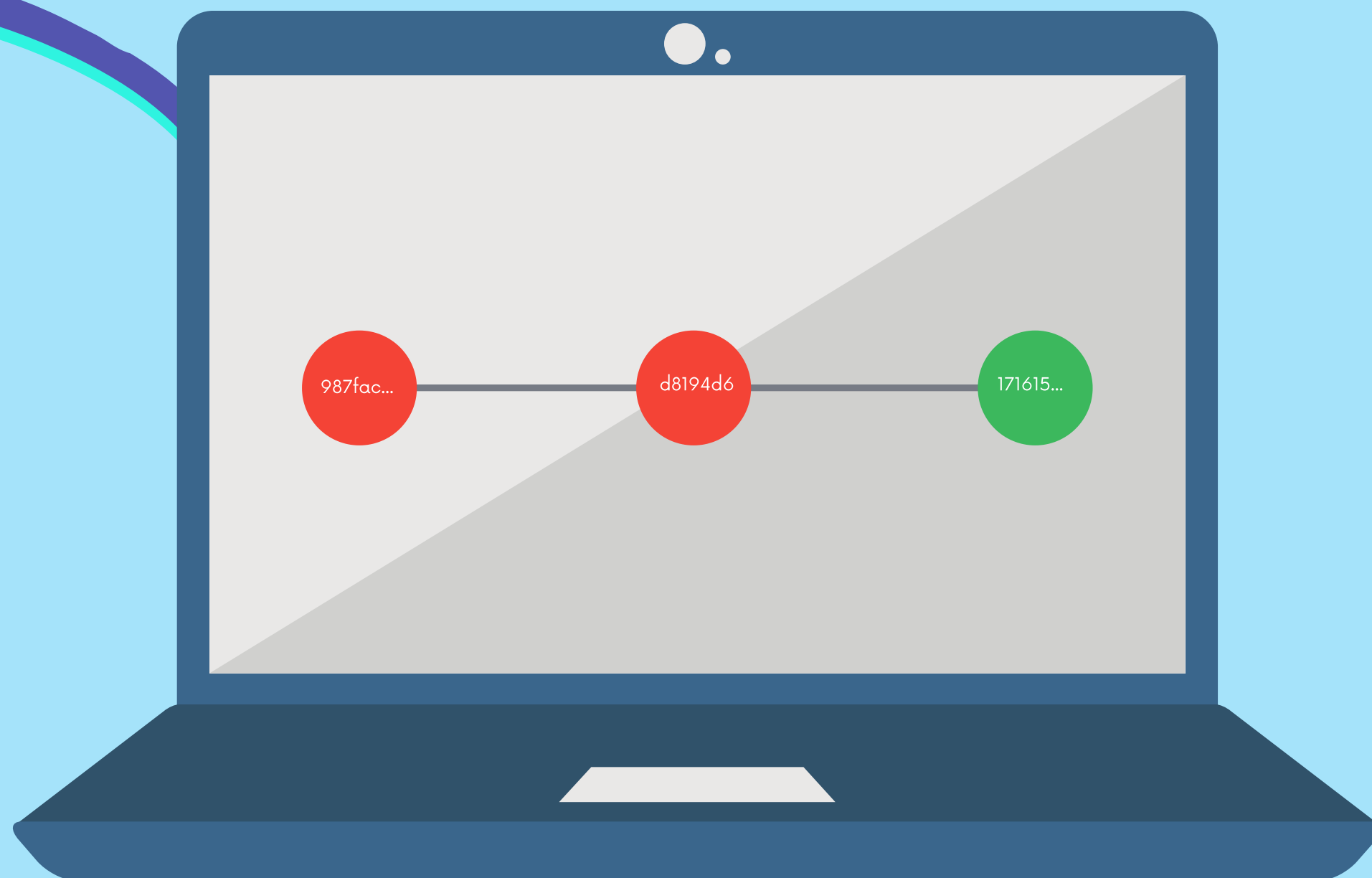


Tell your local repo  
about the Github repo





Push from local repo to the new Github repo







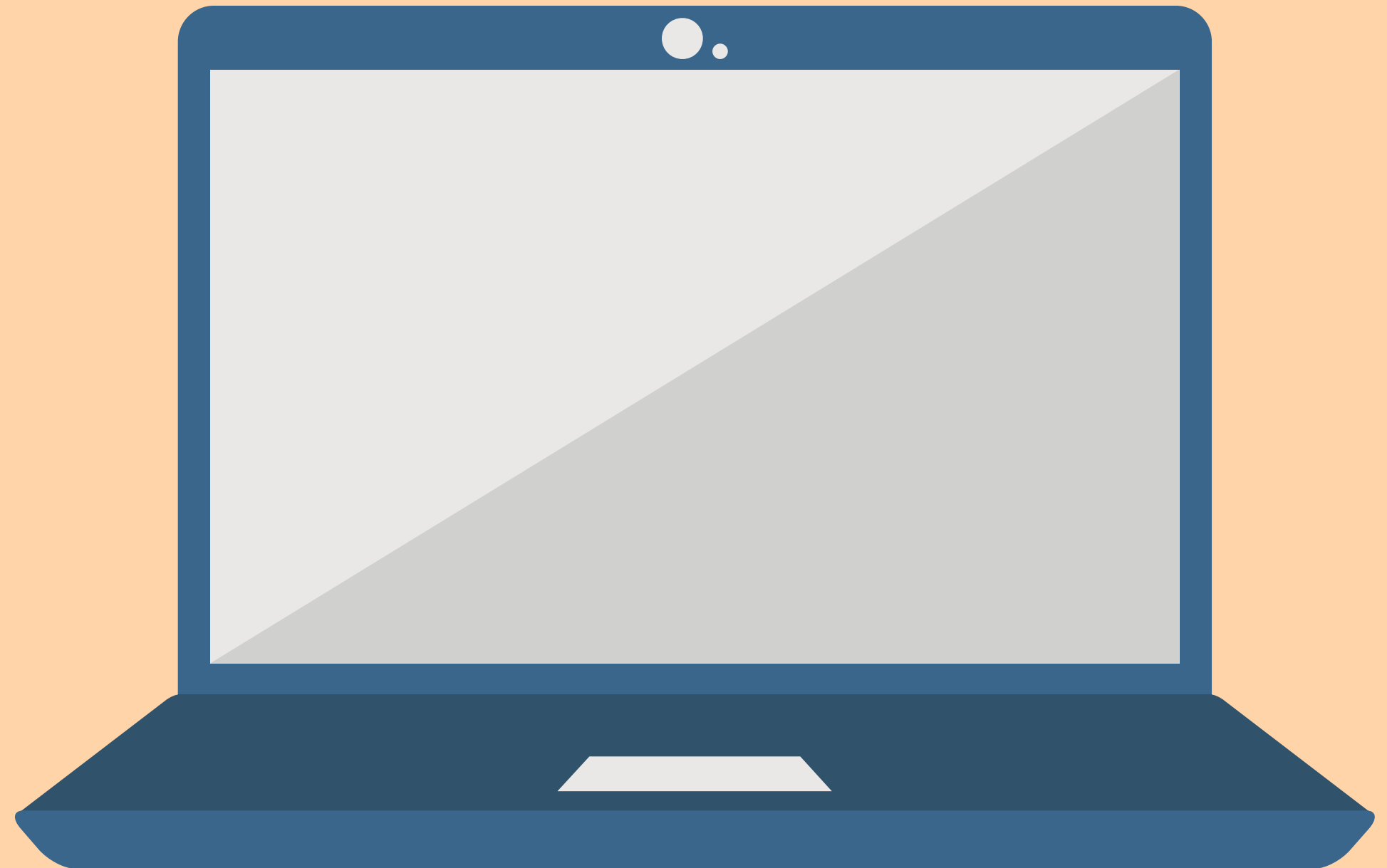
# Option 2: Start From Scratch

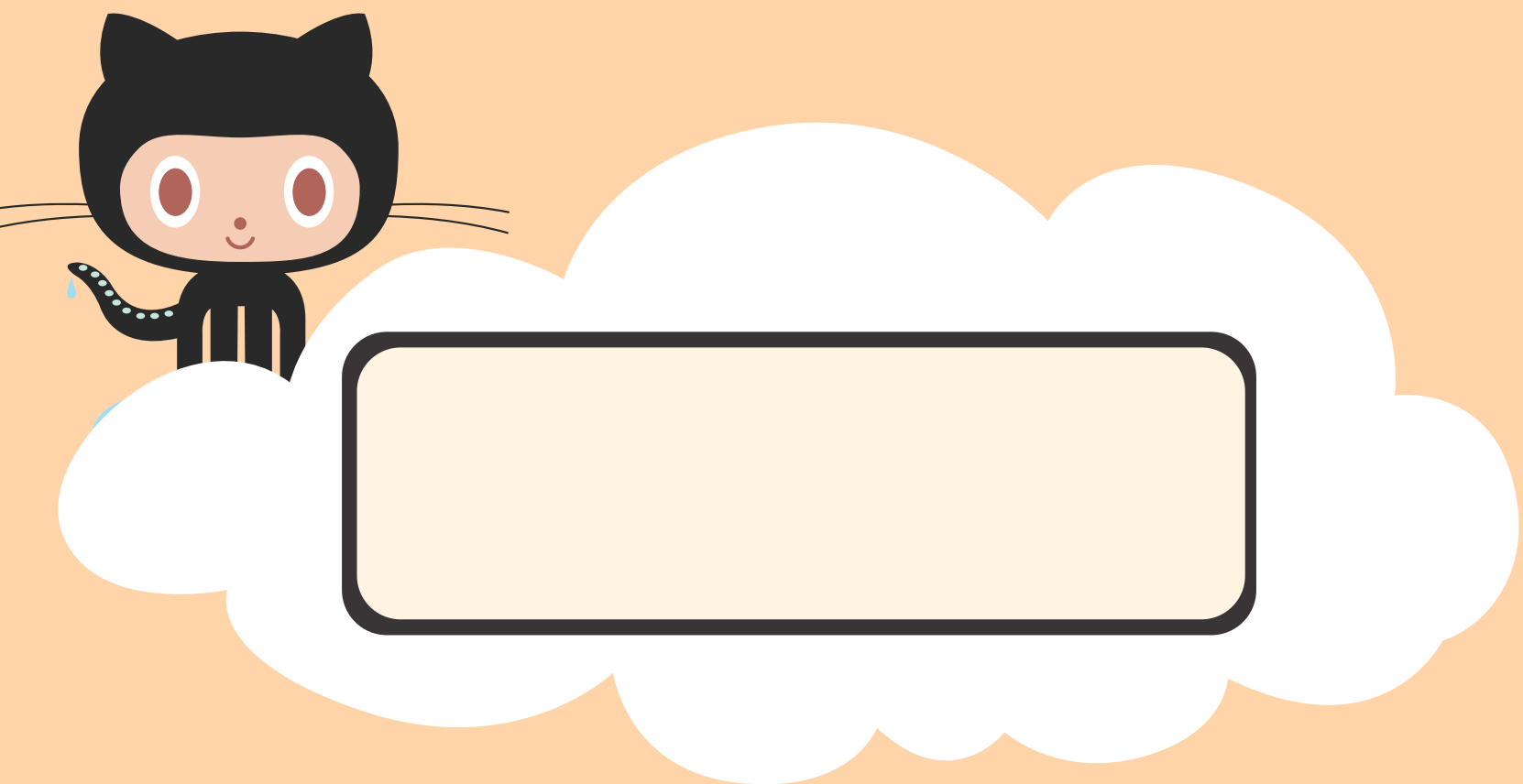
If you haven't begun work on your local repo, you can...

- Create a brand new repo on Github
- Clone it down to your machine
- Do some work locally
- Push up your changes to Github

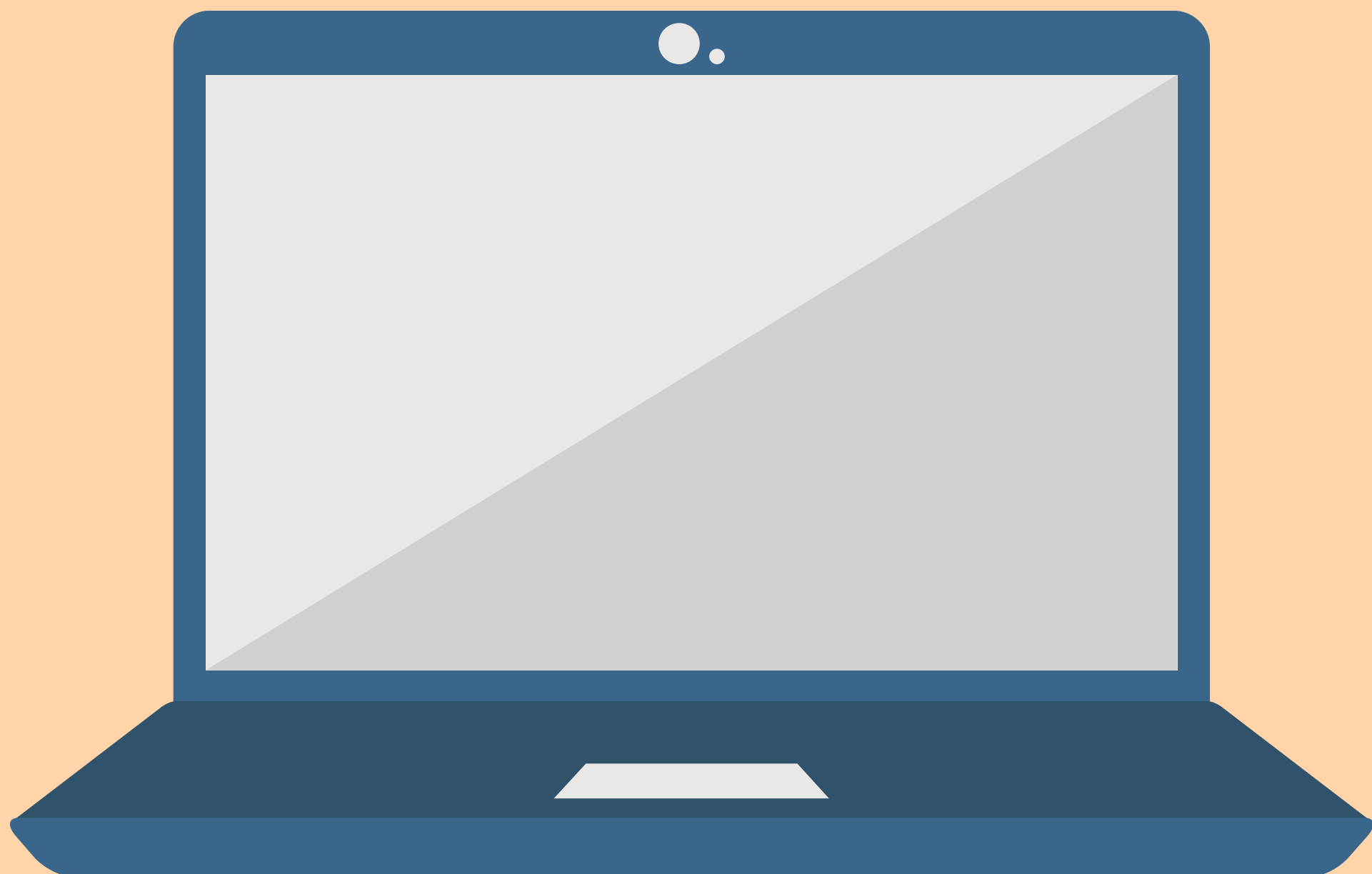


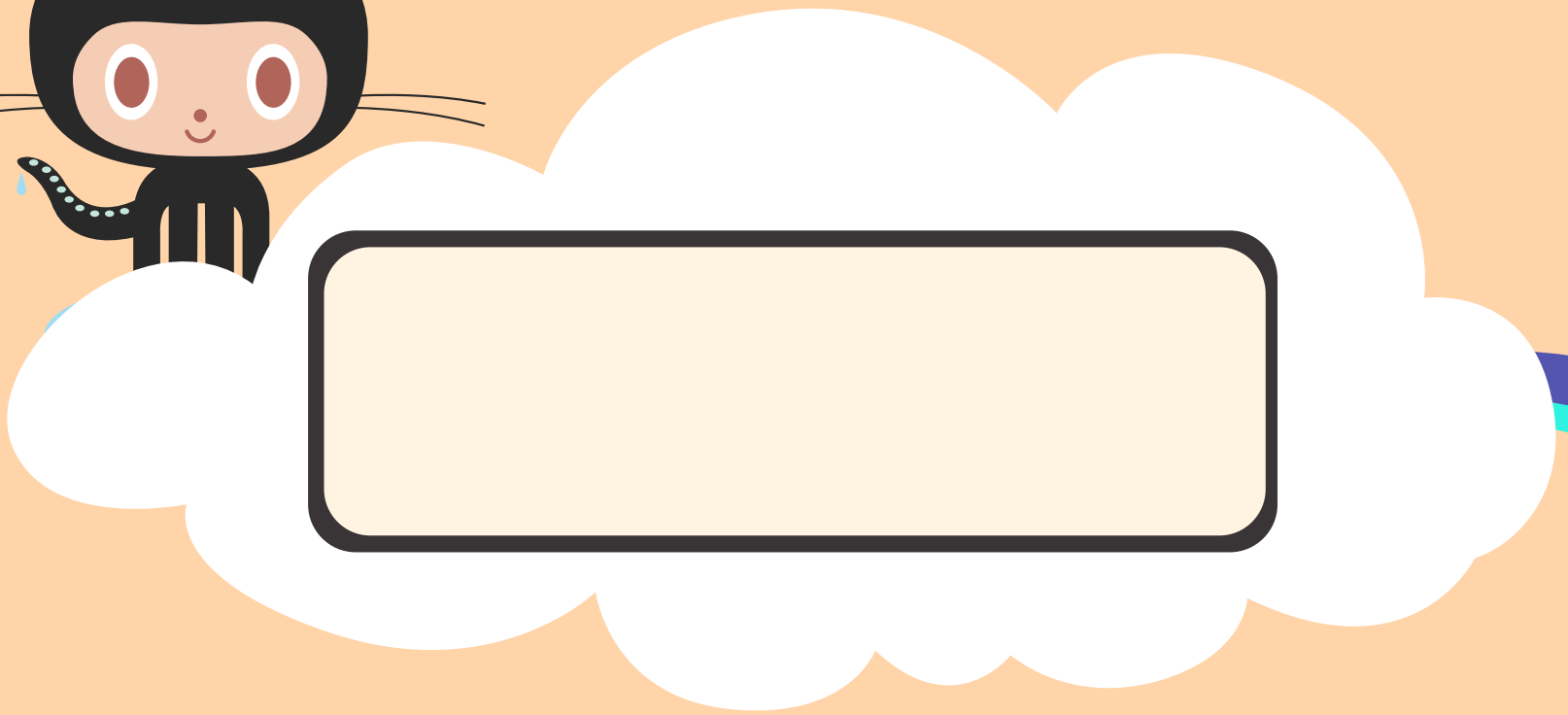
No repo exists yet!



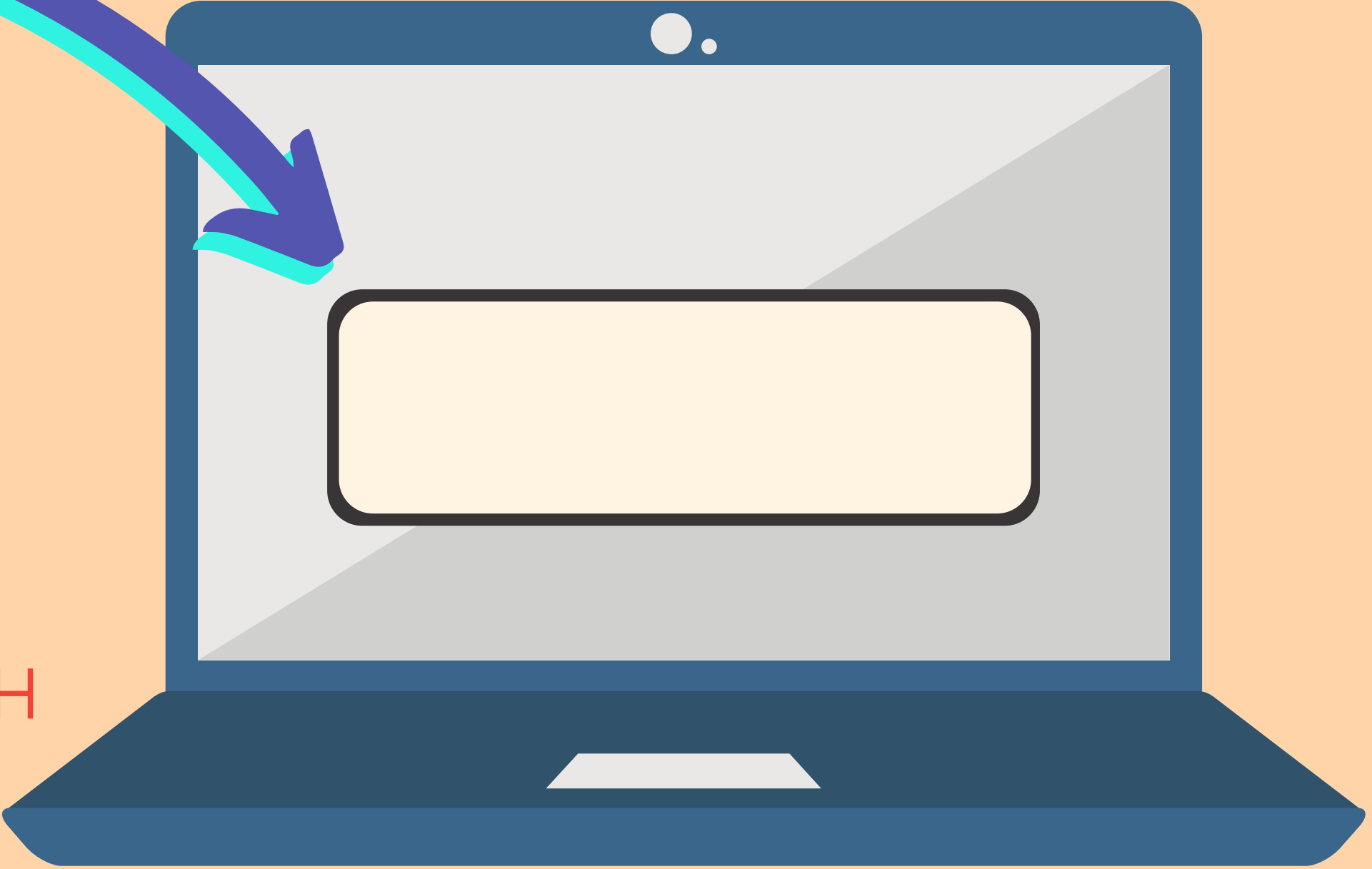


Create a new empty  
repo on Github

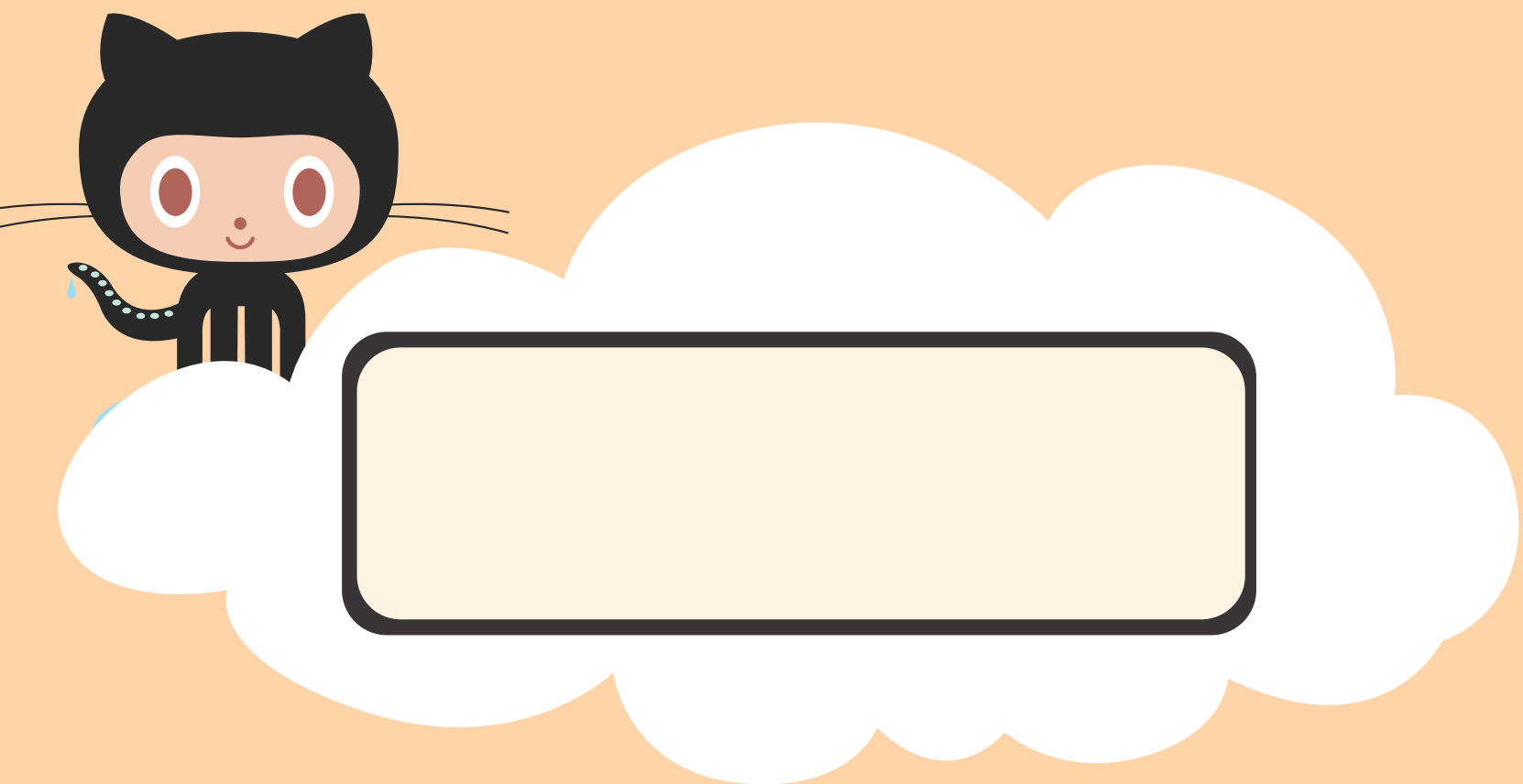




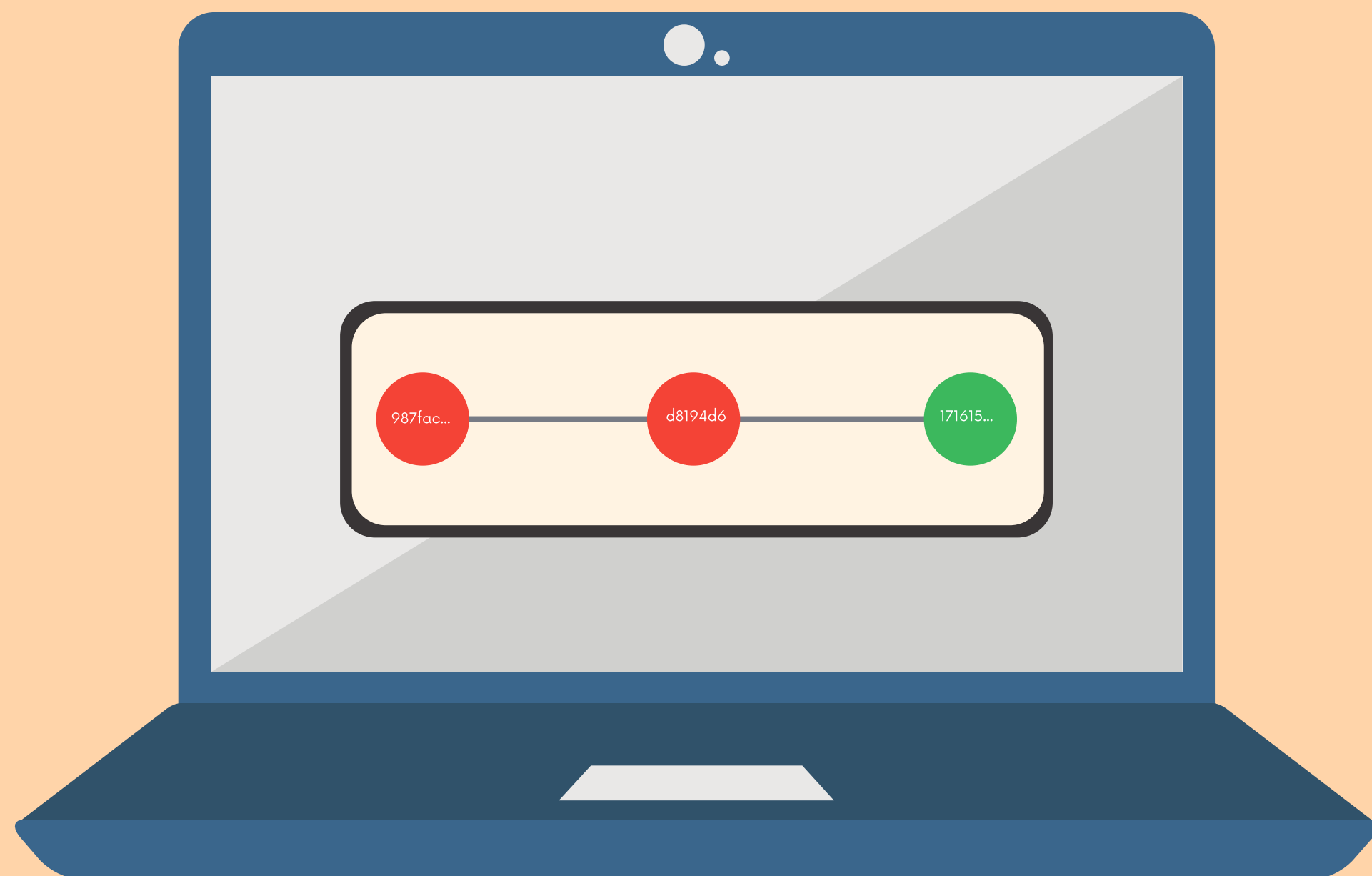
Clone the Github repo  
to your local machine

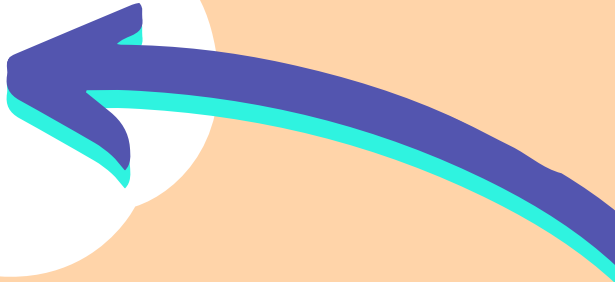
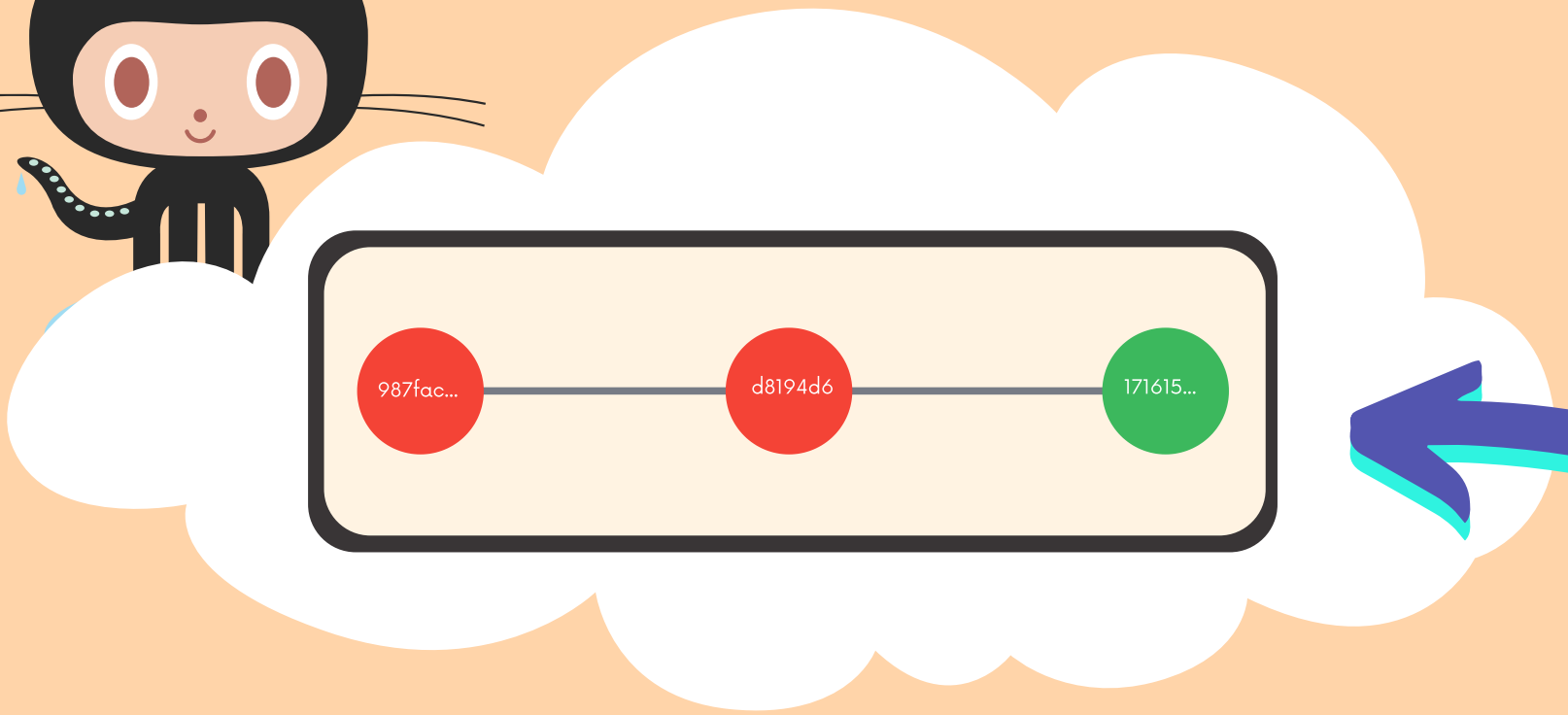


The local repo is  
automatically  
"connected" to GH

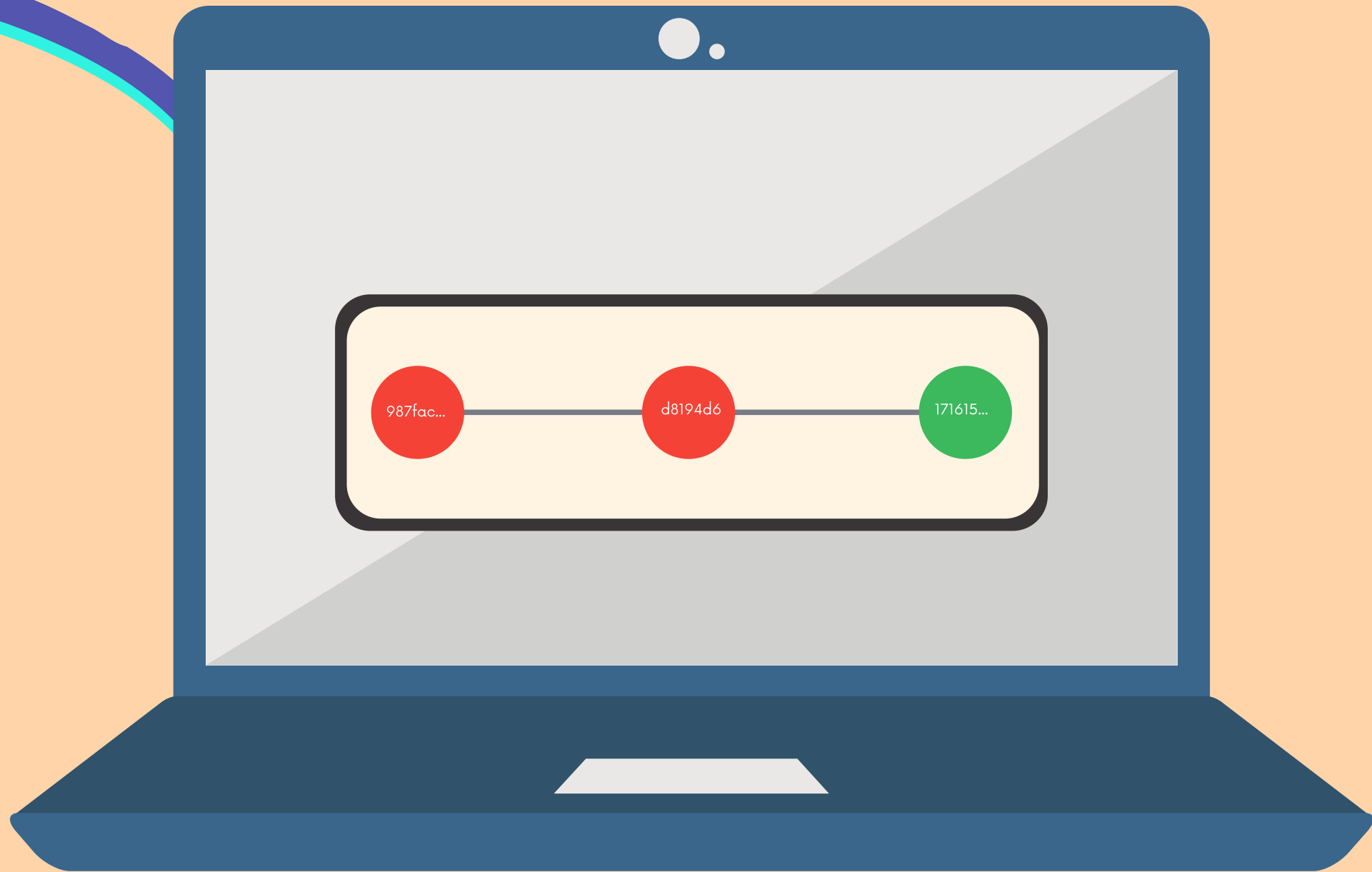


Do some work and make  
some commits locally





Push that new work  
up to Github!





# Pushing

To get your own changes and Git history up on Github, we need to PUSH them up. The typical workflow looks something like this:

- Make some changes locally
- Add and commit those changes
- Repeat...
- **Push new commits up to Github**



**First, We Need To  
Make a Repo On Github**





# Remote

Before we can push anything up to Github, we need to tell Git about our remote repository on Github. We need to setup a "destination" to push up to.

In Git, we refer to these "destinations" as remotes. Each remote is simply a URL where a hosted repository lives.

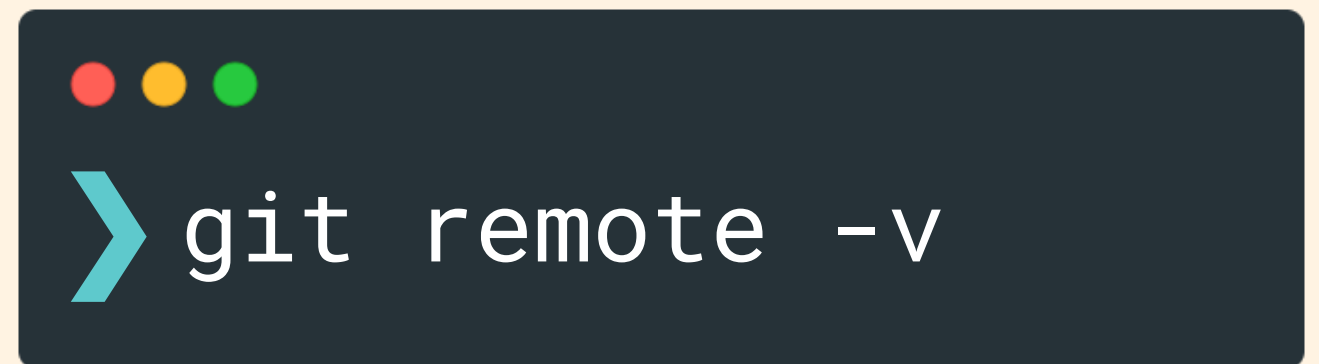




# Viewing Remotes

To view any existing remotes for your repository, we can run `git remote` or `git remote -v` (verbose, for more info)

This just displays a list of remotes. If you haven't added any remotes yet, you won't see anything!



```
> git remote -v
```





# Adding A New Remote

A remote is really two things: a URL and a label.  
To add a new remote, we need to provide both to Git.

```
 >git remote add <name> <url>
```





# Adding A New Remote

```
git remote add origin  
https://github.com/blah/repo.git
```

Okay Git, anytime I use the name "origin", I'm referring to this particular Github repo URL.





# Origin?

**Origin** is a conventional Git remote name, but it is not at all special. It's just a name for a URL.

When we clone a Github repo, the default remote name setup for us is called origin. You can change it. Most people leave it.





# Adding A New Remote

```
git remote add mygithuburl  
https://github.com/meh/repo.git
```

Okay Git, anytime I use the name "mygithuburl", I'm referring to this particular Github repo URL.

This is not a commonly used remote name.

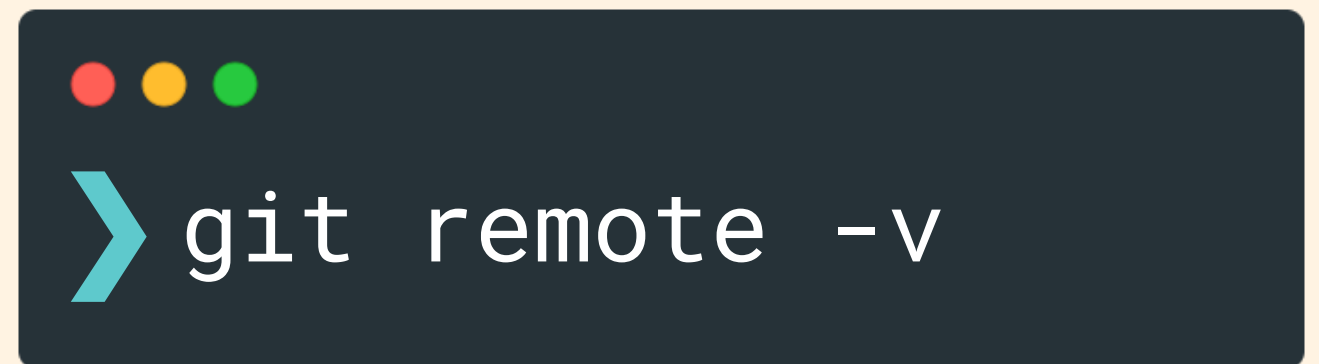




# Checking Our Work

Try viewing your remotes with `git remote -v`, and you should now see a remote showing up!

Remember, by setting up a remote we are just telling Git about a remote repository URL. We have not "communicated" with the Github repo at all yet.

A dark-themed terminal window with three colored window control buttons (red, yellow, green) in the top left corner. A light blue prompt character is followed by the command `git remote -v` in white text.

```
> git remote -v
```



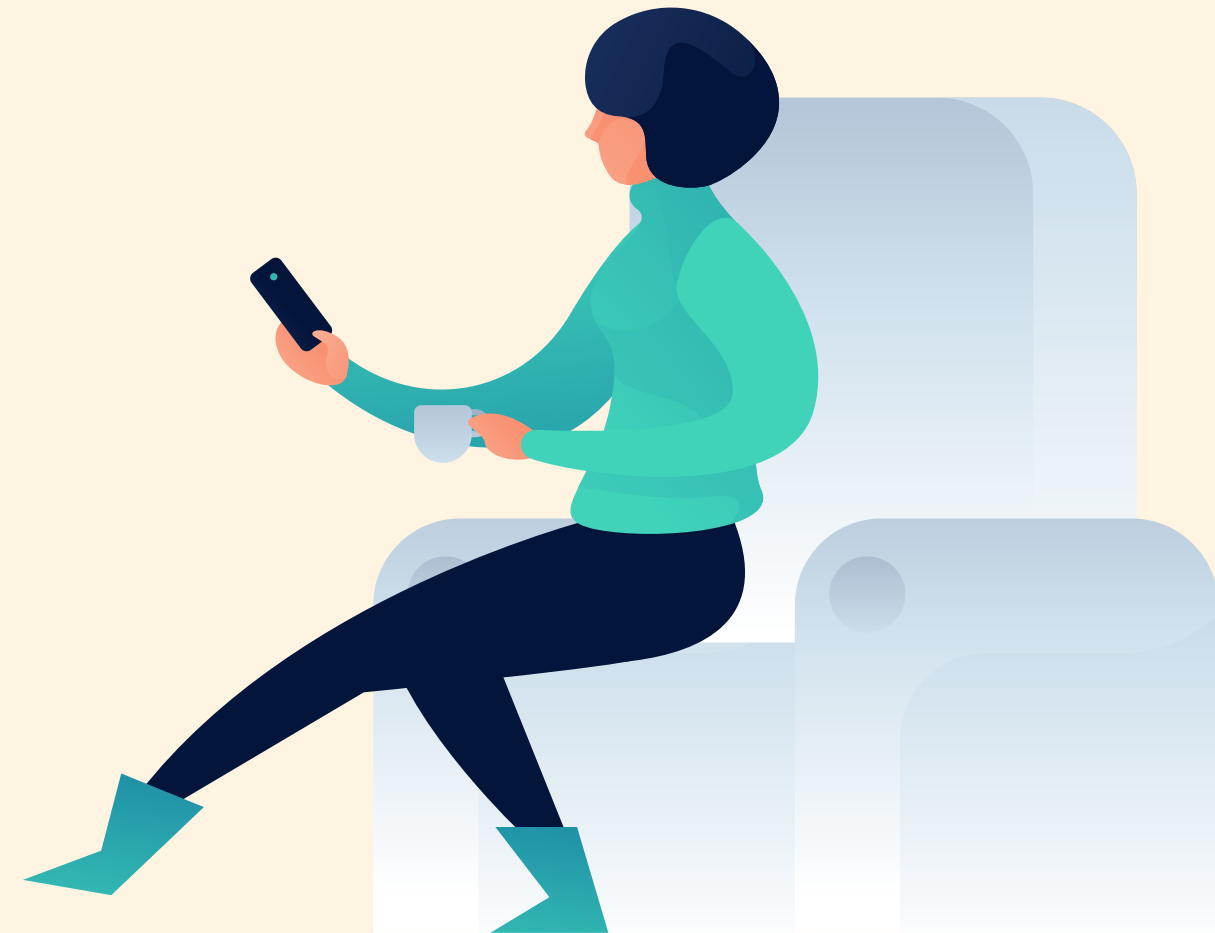


# Other commands

They are not commonly used, but there are commands to rename and delete remotes if needed.

```
git remote rename <old> <new>
```

```
git remote remove <name>
```



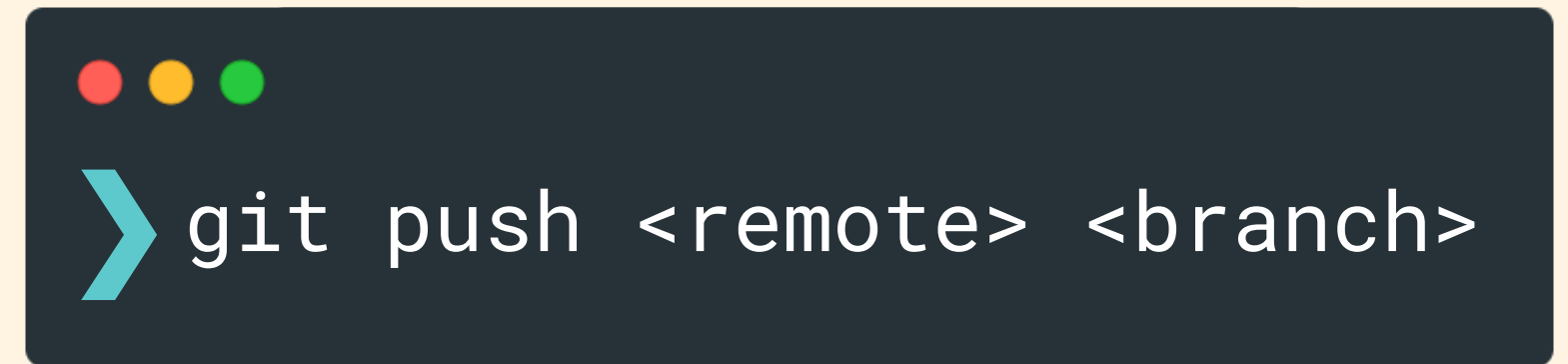




# Pushing

Now that we have a remote set up, let's push some work up to Github! To do this, we need to use the **git push** command.

We need to specify the remote we want to push up to AND the specific local branch we want to push up to that remote.



```
> git push <remote> <branch>
```





# An Example

`git push origin master` tells git to push up the master branch to our origin remote.



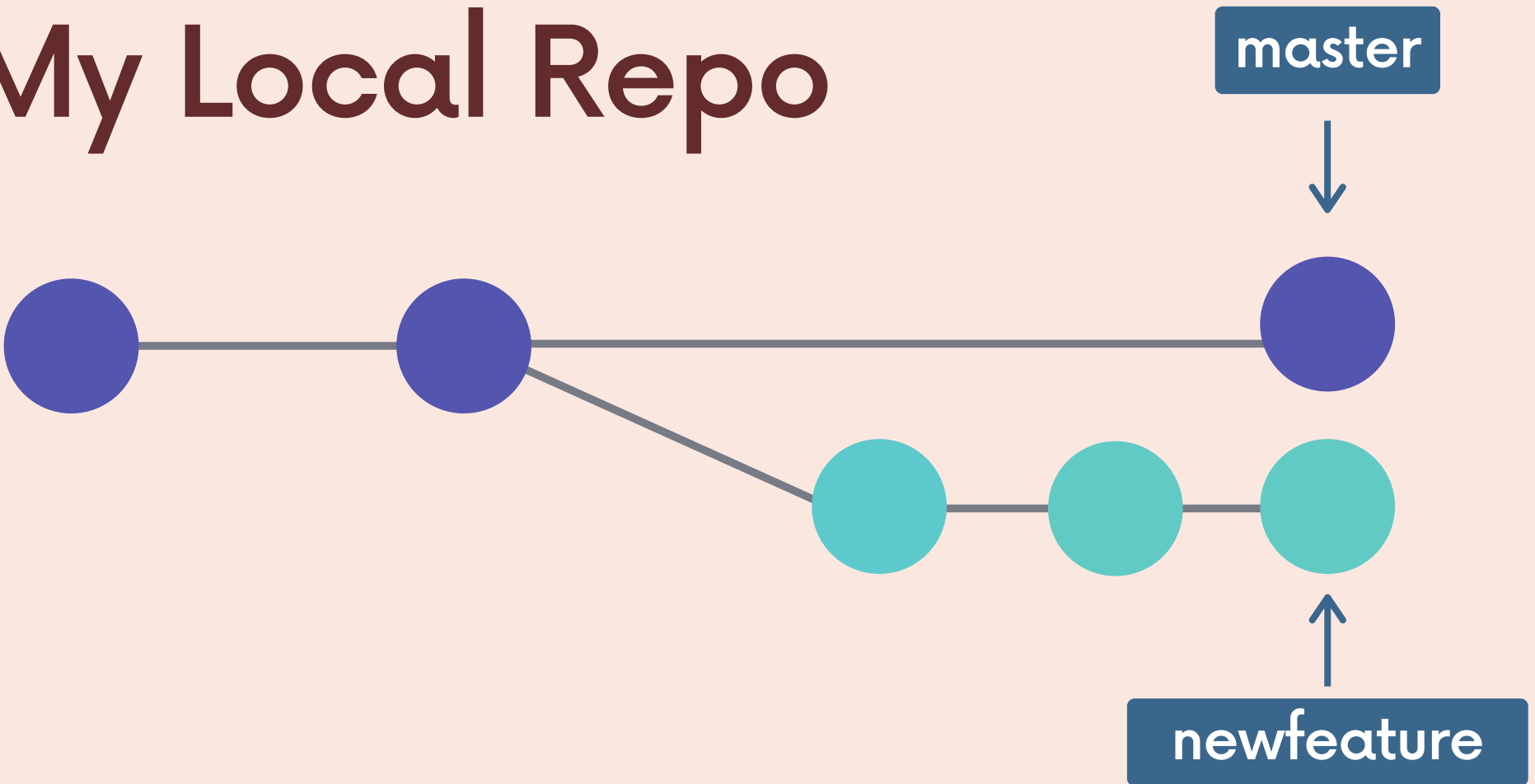
```
> git push origin master
```



# Github Repo

nothing to see here...

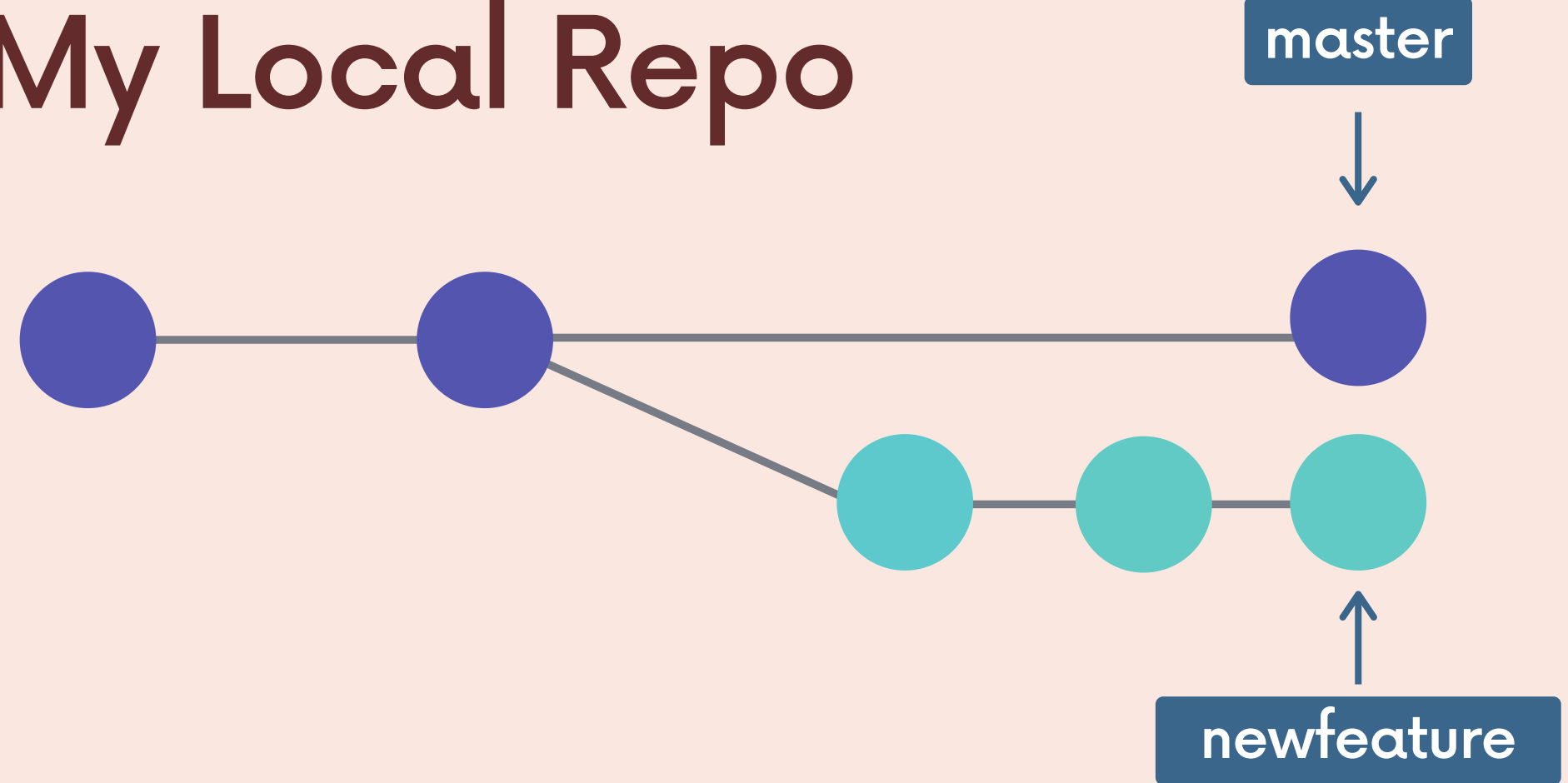
## My Local Repo



# Github Repo



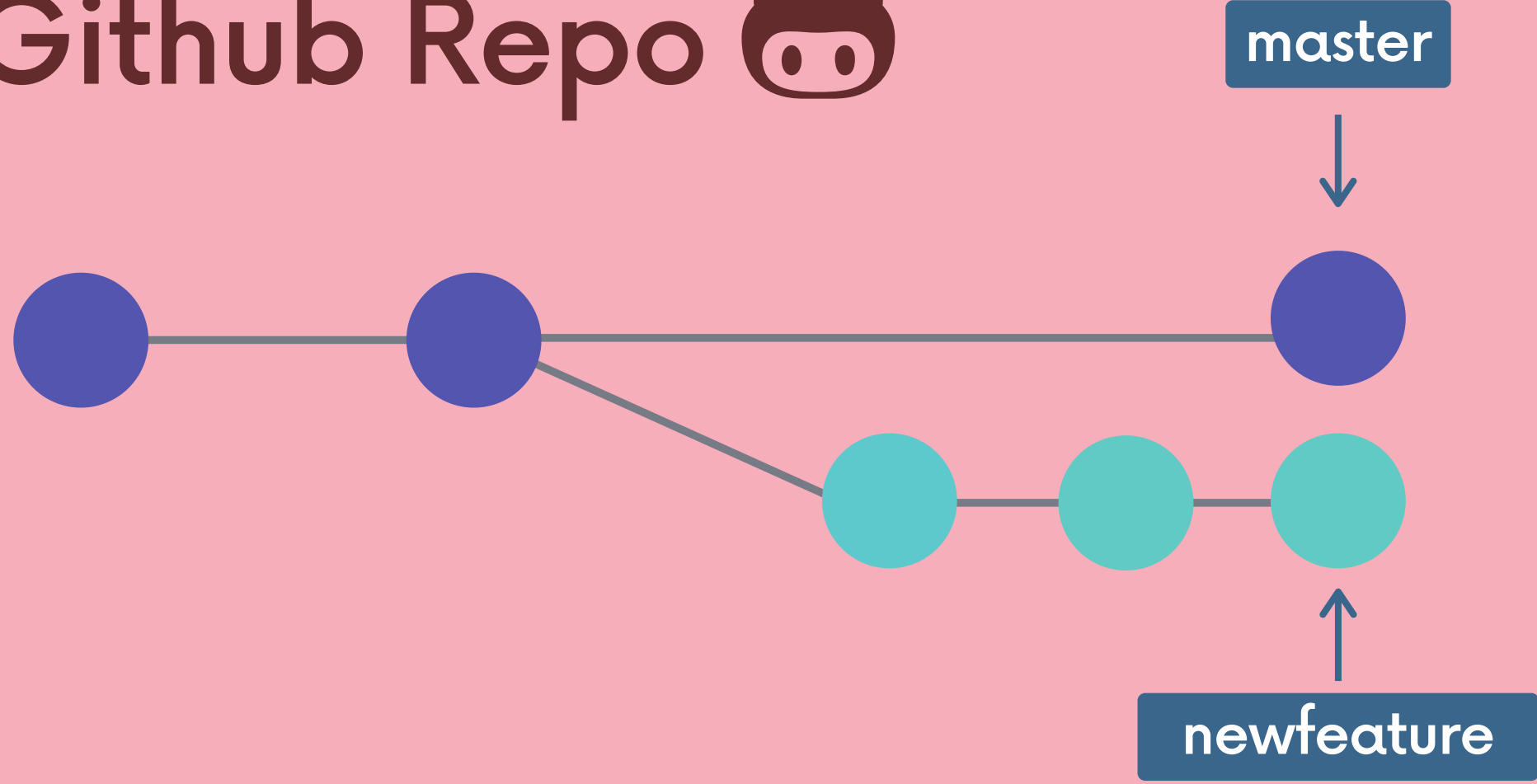
# My Local Repo



```
> git push origin master
```

To push up my master branch to my Github repo (assuming my remote is named origin)

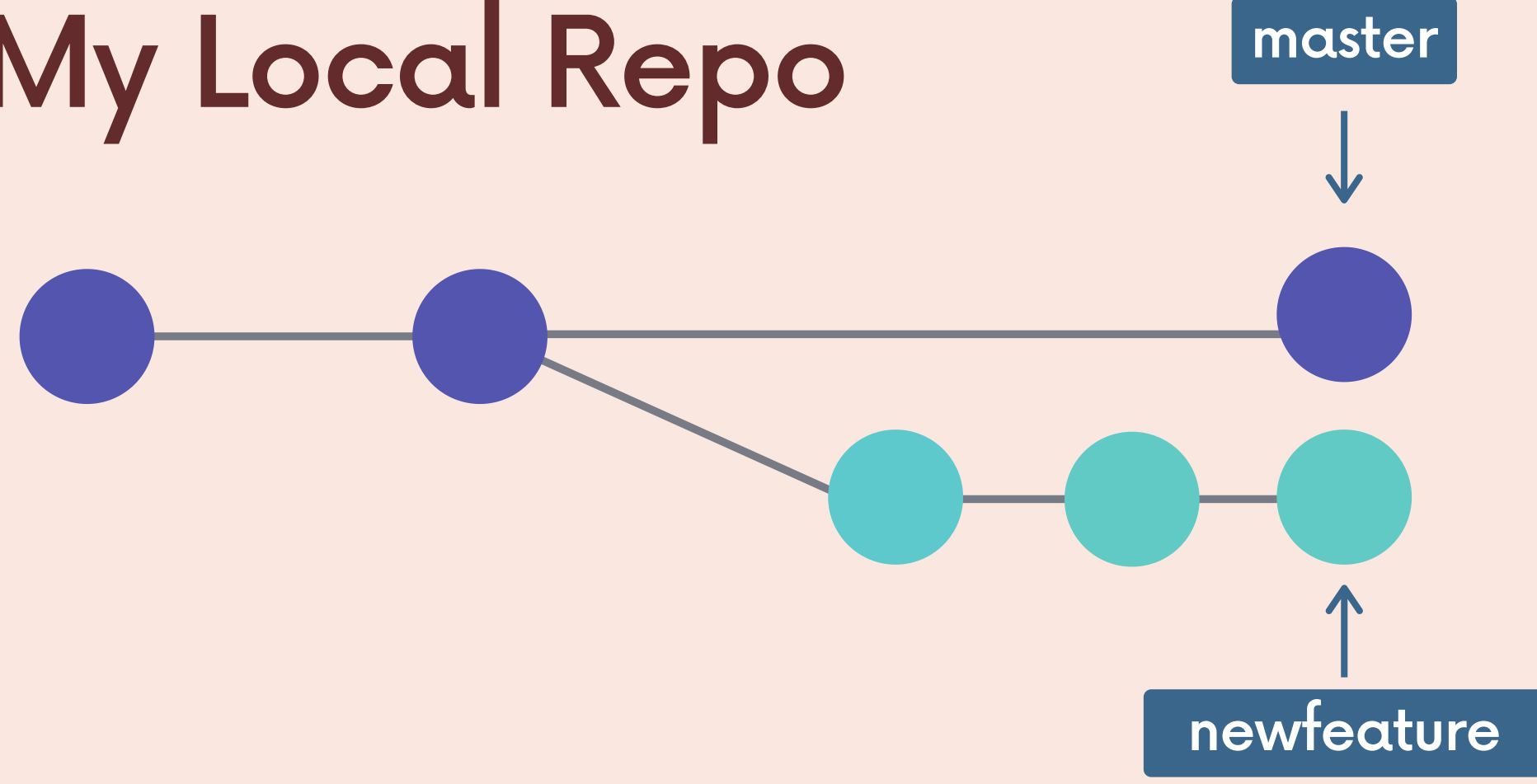
# Github Repo



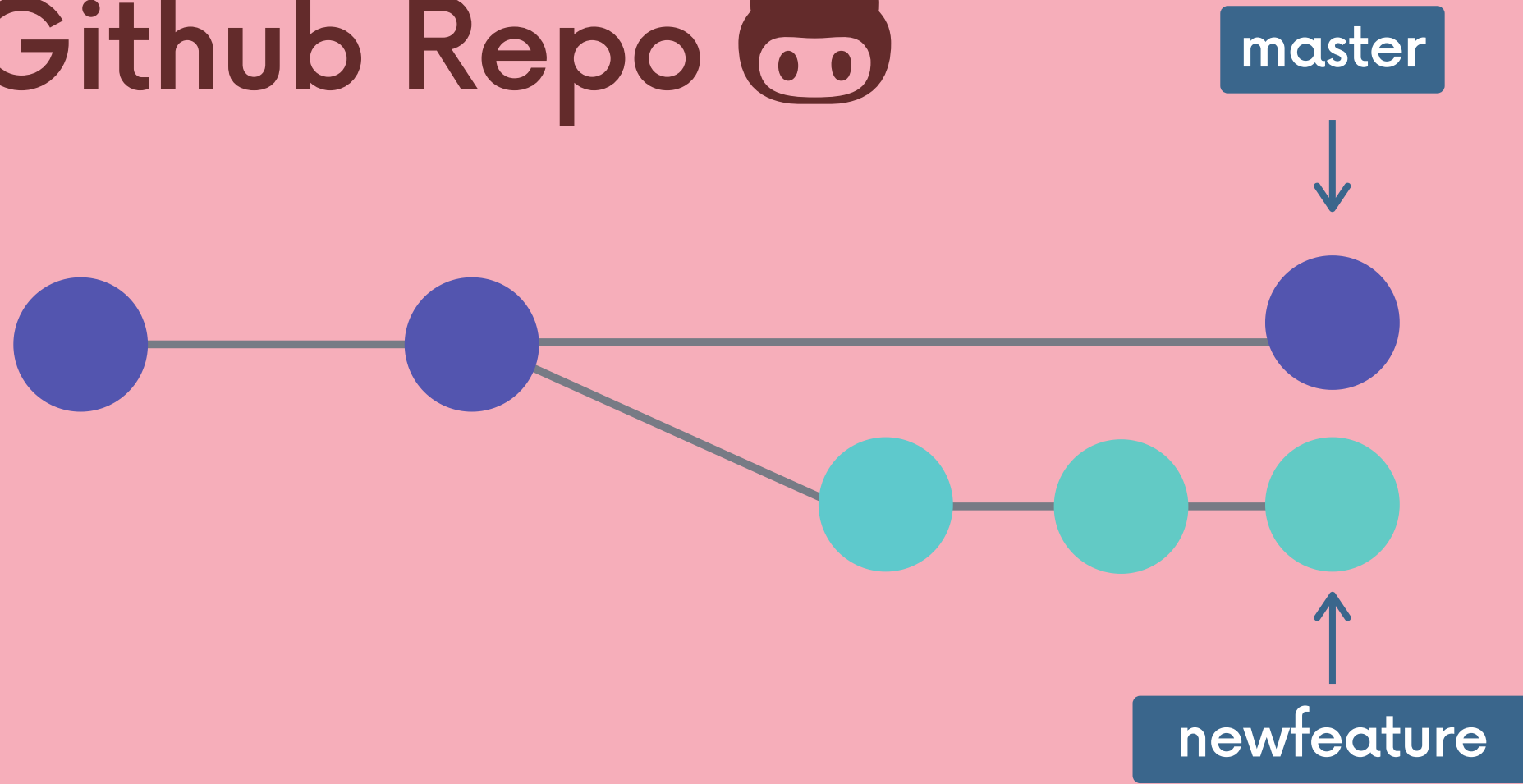
To push up the newfeature branch to Github...

```
> git push origin newfeature
```

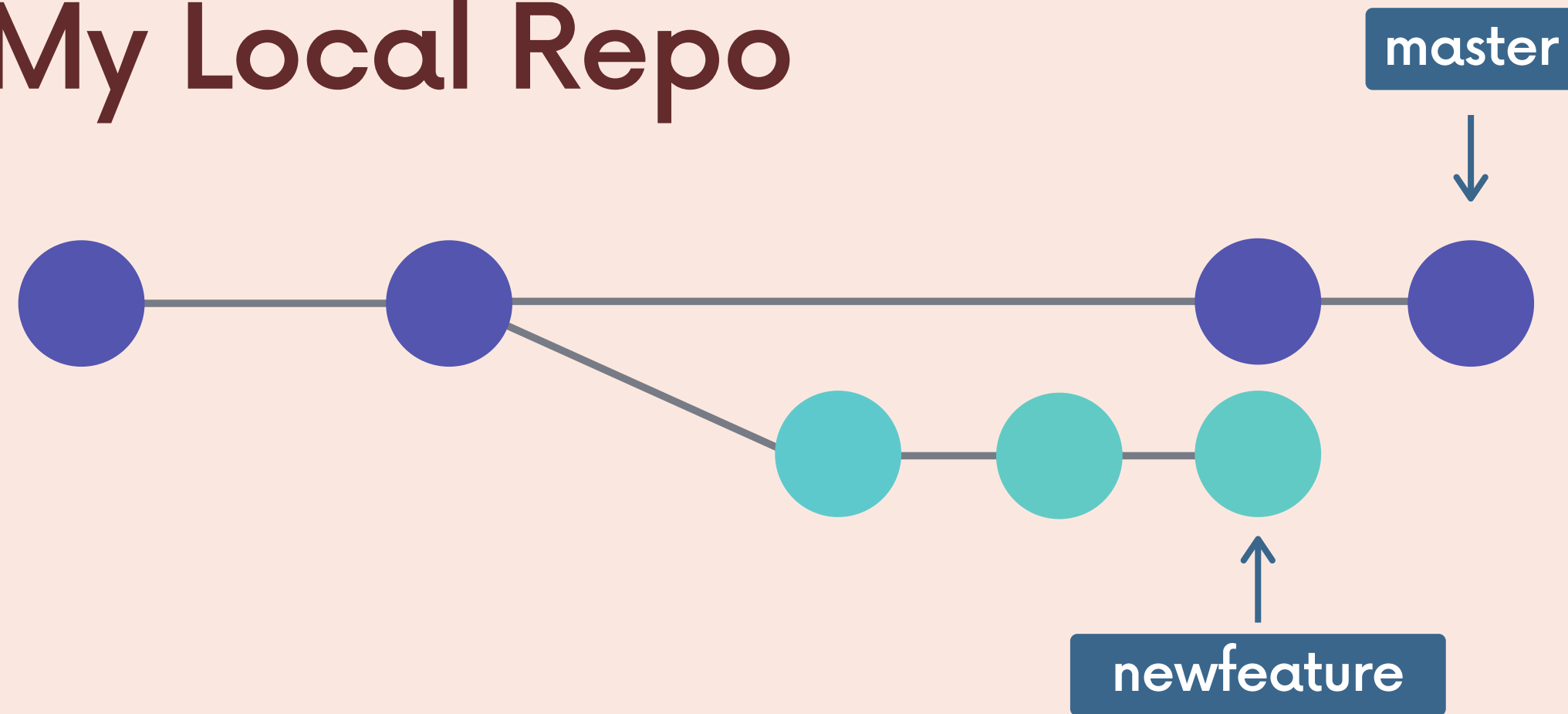
# My Local Repo



# Github Repo

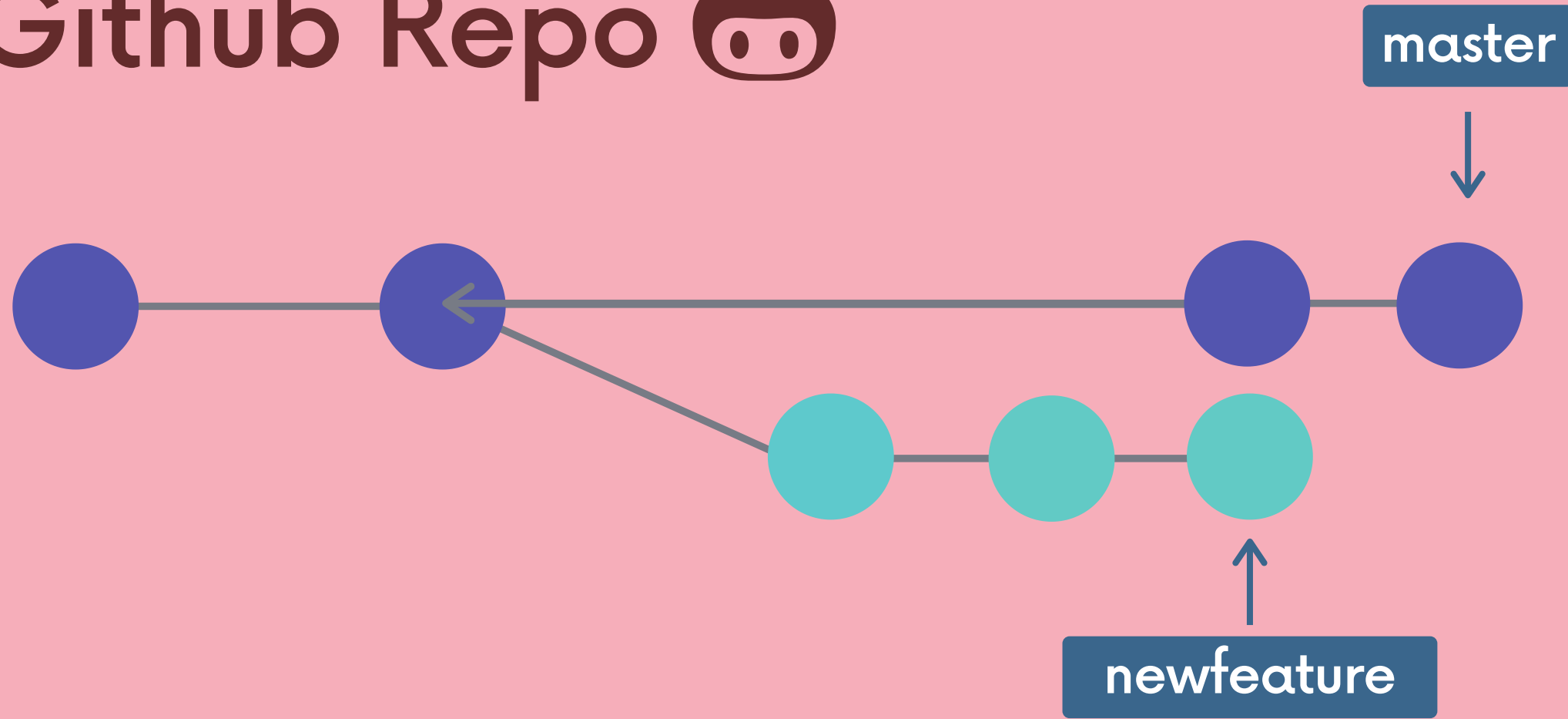


# My Local Repo



I make some new commits locally.  
My Github repo has no idea!

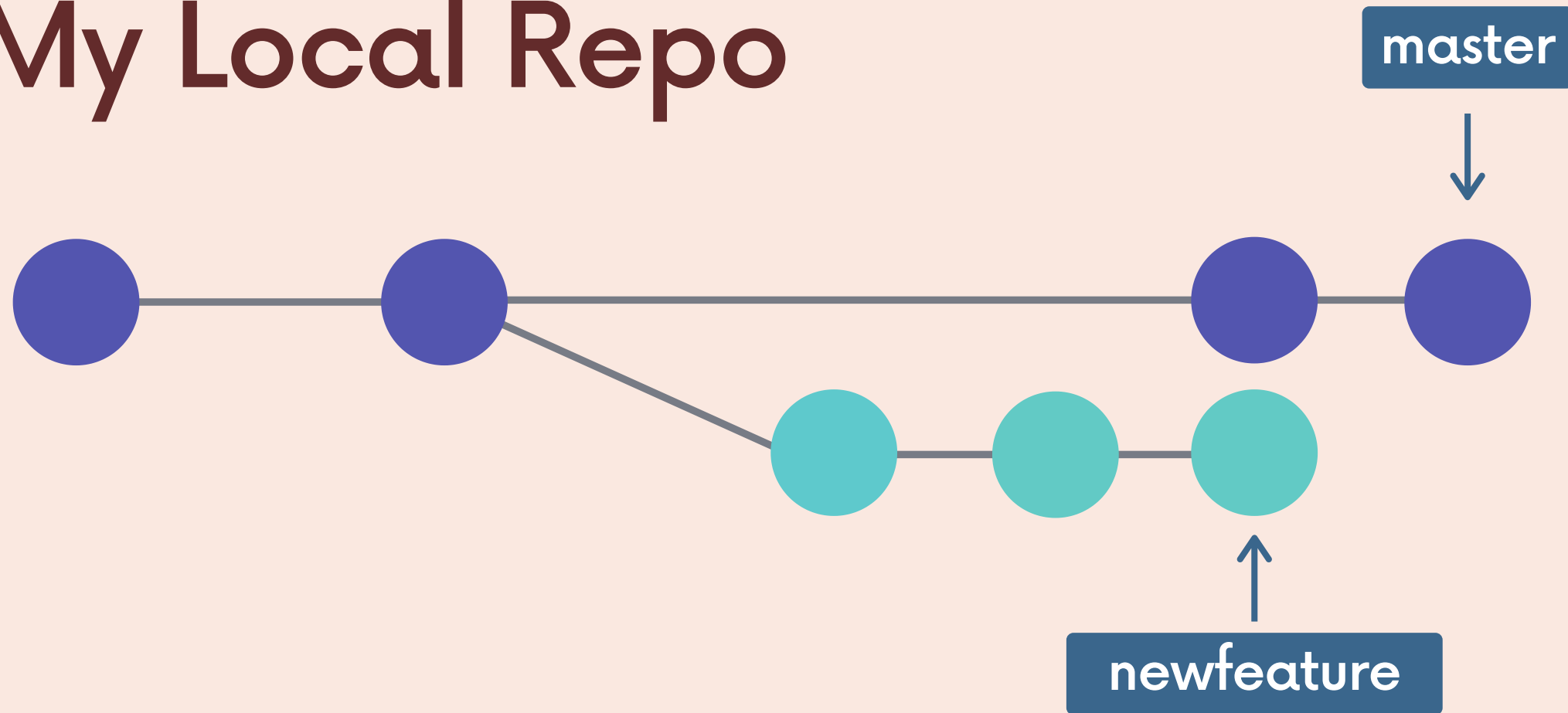
# Github Repo



Push up the master branch again, to make sure the Github repo has the new commits

```
  
> git push origin master
```

# My Local Repo





# Workflow Recap

Remember to follow these basic steps:

- Create a new empty repo on Github
- Copy the repo URL
- Add a remote to your local repo, using the URL
- Push your changes up to the remote

```
> git remote -v
```





