Agenda

- Setting up a Github account
- Installing and configuring Git
- Local and Remote Repos
- Deploying React App on Github Pages

What Is Git?

It is a free, high-quality distributed version control system suitable for tracking modifications in source code in software development. It was original created as an open-source system for coordinating tasks among programmers, but today it is widely used to track changes in any set of files. The ke objectives of Git are as follows:

- Speed and efficiency
- Data integrity
- Support for distributed and non-linear workflows

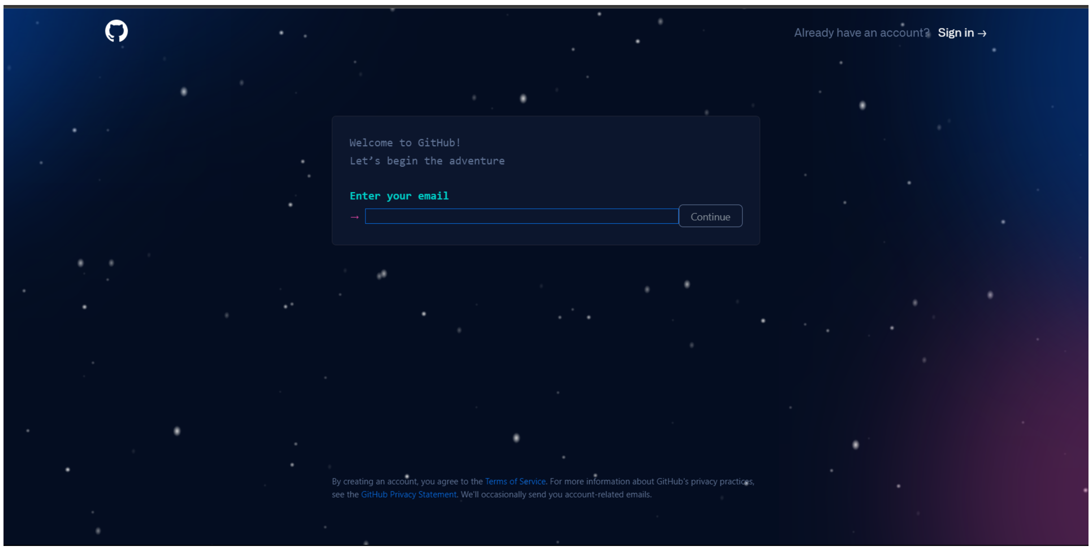
What Is GitHub?

It is a web-based Git repository. This hosting service has cloud-based storage. GitHub offers all distributed version control and source code manageme functionality of Git while adding its own features. It makes it easier to collaborate using Git.

Additionally, GitHub repositories are open to the public. Developers worldwide can interact and contribute to one another's code, modify or improve making GitHub a networking site for web professionals. The process of interaction and contribution is also called social coding.

Setting up a Github account

If you do not already have a github account you can create one by going to https://github.com/signup

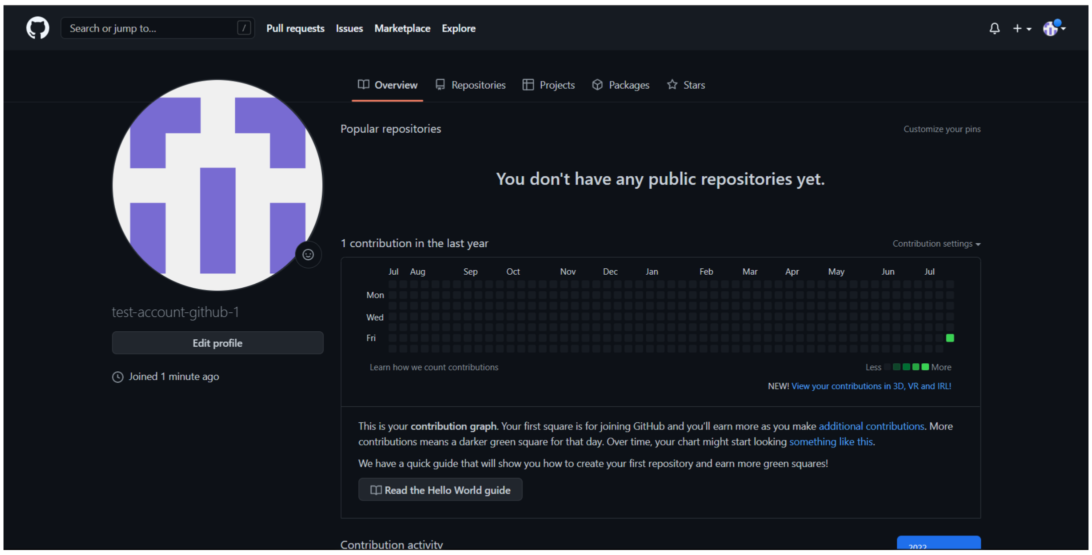


Welcome to GitHub!
Let's begin the adventure

Enter your email

By creating an account, you agree to the Terms of Service. For more information about GitHub's privacy practices, see the GitHub Privacy Statement. We'll occasionally send you account-related emails.

After you complete the setup you can open your profile.



Popular repositories

Customize your pins

You don't have any public repositories yet.

test-account-github-1

Joined 1 minute ago

1 contribution in the last year

Contribution settings

Learn how we count contributions

Less More

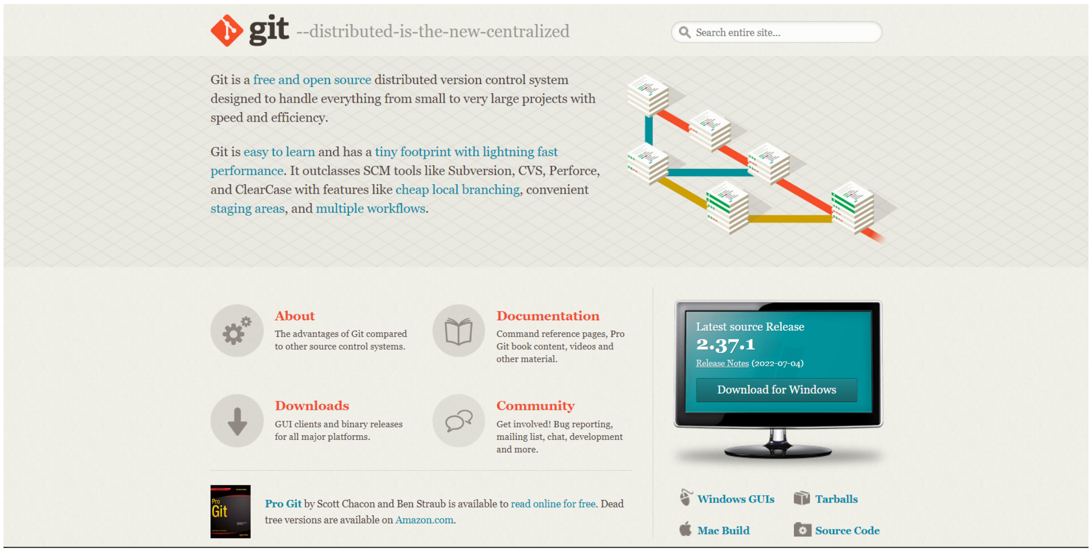
NEW! View your contributions in 3D, VR and IRL!

This is your **contribution graph**. Your first square is for joining GitHub and you'll earn more as you make additional contributions. More contributions means a darker green square for that day. Over time, your chart might start looking something like this.

We have a quick guide that will show you how to create your first repository and earn more green squares!

Contribution activity

Once you are done with setting up the github you have to navigate to https://git-scm.com/ . This will be where you install git.



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.

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.

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

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

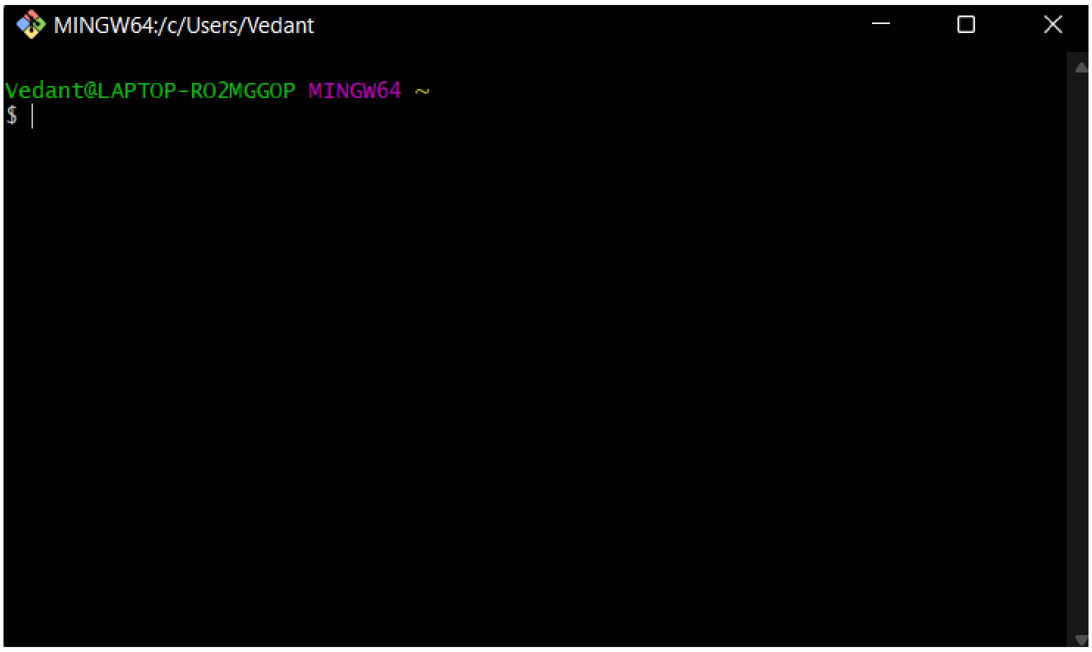
Downloads
GUI clients and binary releases for all major platforms.

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

Latest source Release
2.37.1
Release Notes (2022-07-04)

Pro Git by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

Once you are done installing git it will install a program called git bash in your machine.You can open that.



```
MINGW64:/c/Users/Vedant

Vedant@LAPTOP-RO2MGGOP MINGW64 ~
$ |
```

What you can do is connect the git bash to your github account using the following commands

```
git config --global user.name "your_user_namer"
git config --global user.email "your_email"
```

To verify it you can type

```
git config --global --list
```

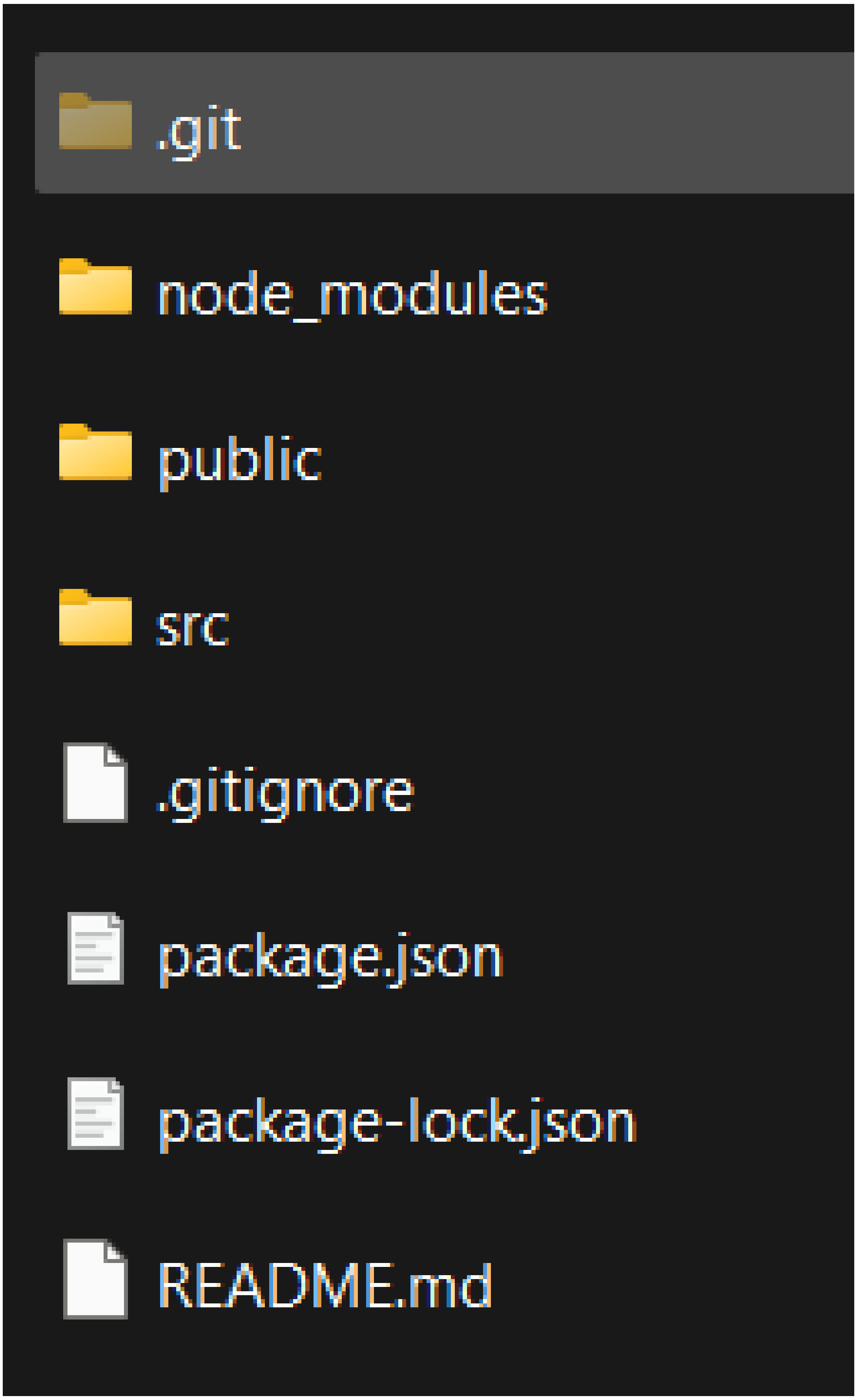
Local and Remote Repositories

The local repository is a Git repository that is **stored on your computer.**

The remote repository is a Git repository that is **stored on some remote computer.**

The remote repository is usually used by teams as **a central repository** into which everyone pushes the changes from his local repository and fro which everyone pulls changes to his local repository.

Once you decide it's a good time to share the changes, you push the changes from your local repository to the **remote repository**. This copies tl changes from **.git folder on your local computer** to **.git folder on the remote computer.** From this moment, your changes are visible to people wl have access to the remote repository.



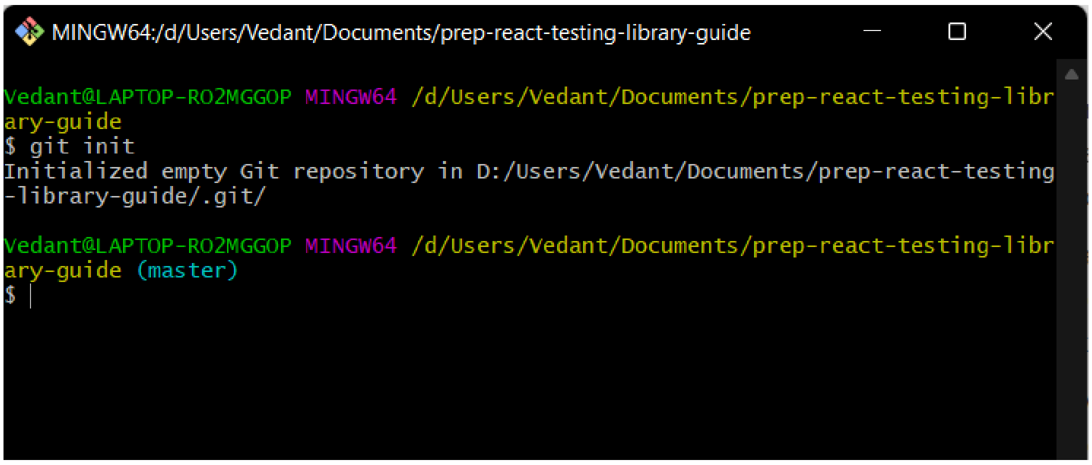
```
.git
node_modules
public
src
.gitignore
package.json
package-lock.json
README.md
```

- The **.git** folder contains all information that is necessary for the project and all information relating commits, remote repository address, etc. It also contains a log that stores the commit history. This log can help you to roll back to the desired version of the code.
- The **.git** folder will contain details of every single change made to the code base. All snapshots of the modifications will be recorded in this folder like a database, which makes it possible to undo the changes and rollback to the desired version of the code.
- The **.git** folder is hidden to prevent accidental deletion or modification of the folder. The version history of the code base will be lost if this folder is deleted. This means, we will not be able to rollback changes made to the code in future.

Creating Local Git Repo

So now we will navigate to the app we used last time for testing and initialize a git repo.(If you already had git installed the repo would be create automatically at the time of creation of react app)

In git bash we will navigate to our project folder and type `git init` . This will initialize a **local git repository**:

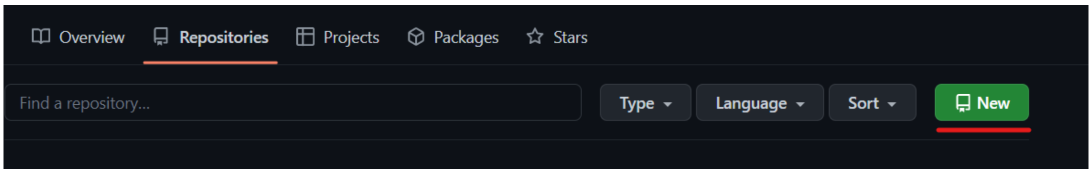


```
MINGW64:/d/Users/Vedant/Documents/prep-react-testing-library-guide

Vedant@LAPTOP-RO2MGGOP MINGW64 /d/Users/Vedant/Documents/prep-react-testing-libr
ary-guide
$ git init
Initialized empty Git repository in D:/Users/Vedant/Documents/prep-react-testing
-library-guide/.git/

Vedant@LAPTOP-RO2MGGOP MINGW64 /d/Users/Vedant/Documents/prep-react-testing-libr
ary-guide (master)
$
```

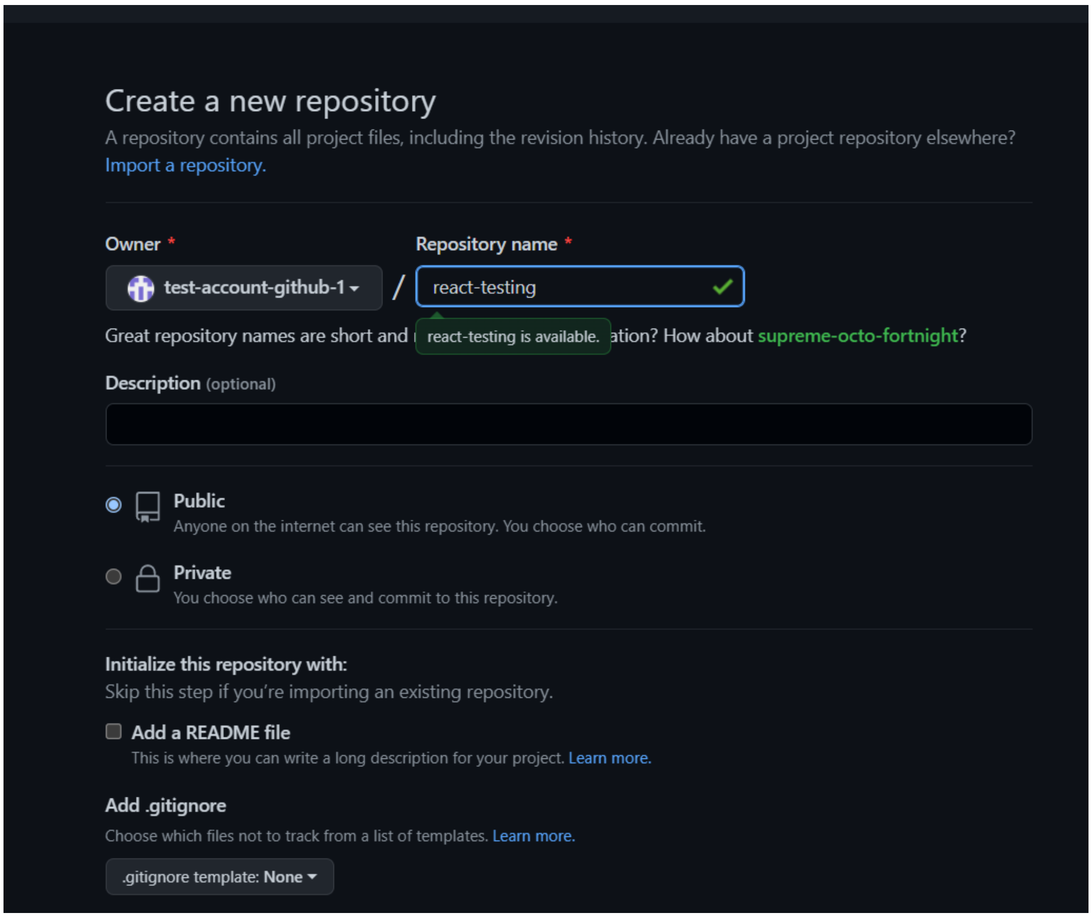
To know the changes in repo type git status

Creating Remote Repo on Github

To create a repository on github you can go to your profile and select New option and fill up the info:



You can choose to keep the repo public or private, add a .gitignore file (A .gitignore file is in which we specify the file/folder names from our project w don’t want to include in the repo)



Create a new repository

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

Owner * test-account-github-1 / Repository name * react-testing

Great repository names are short and react-testing is available. ation? How about supreme-octo-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

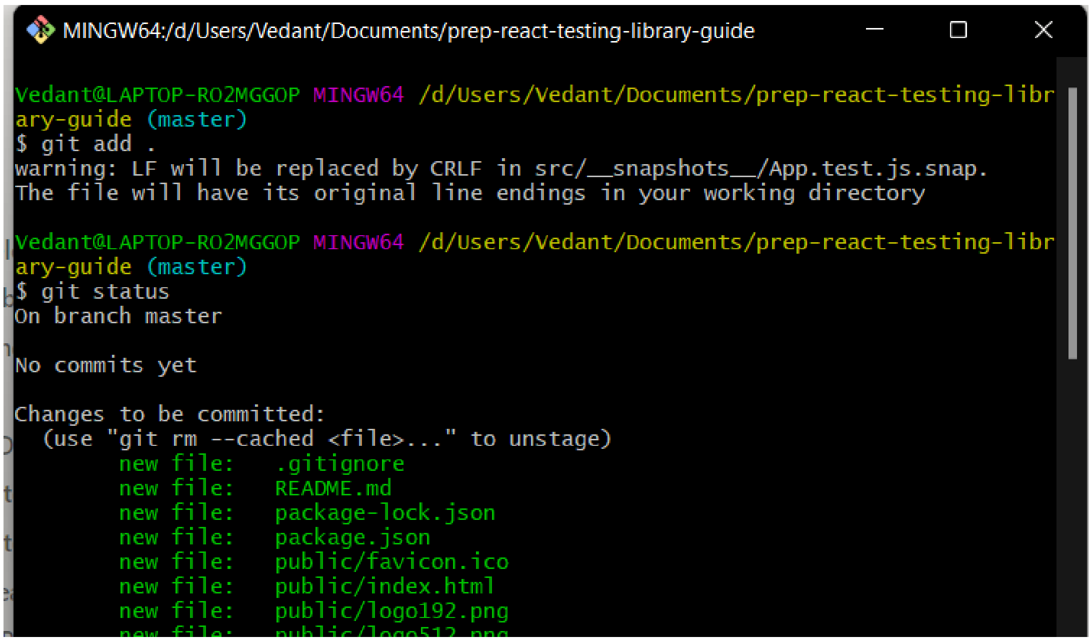
Now lets push the local repo to the remote repo we just created

In our project directory we will first add a .gitignore file(If you already had git installed and created the rect app using create-react-app the file will b already present with following contents)

`.gitignore`

```
# See https://help.github.com/articles/ignoring-files/ for more about ignoring files.

# dependencies
/node_modules
/.pnp
.pnp.js

# testing
/coverage

# production
/build

# misc
.DS_Store
.env.local
.env.development.local
.env.test.local
.env.production.local

npm-debug.log*
yarn-debug.log*
yarn-error.log*
```

After that to push all the code to local repo we type `git add .`

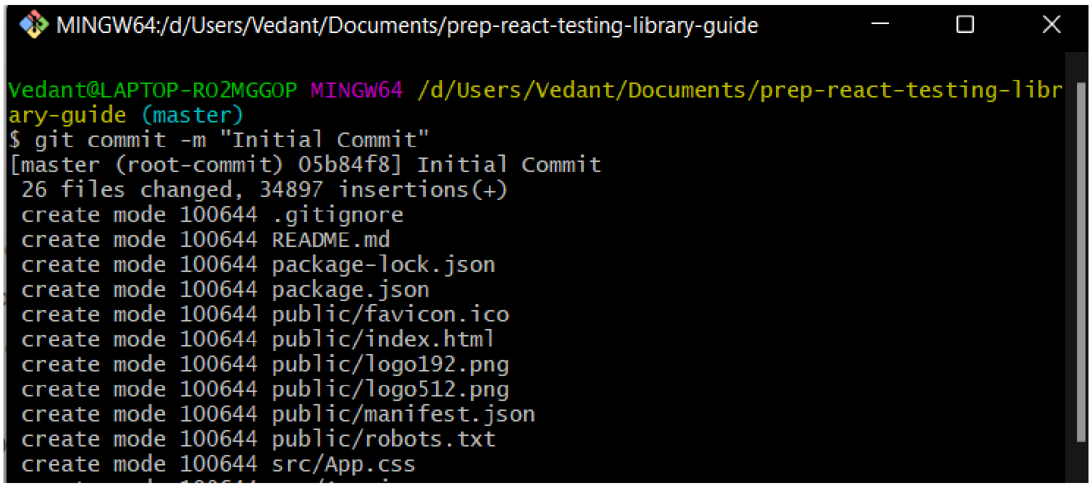


```
MINGW64:/d/Users/Vedant/Documents/prep-react-testing-library-guide

Vedant@LAPTOP-RO2MGGOP MINGW64 /d/Users/Vedant/Documents/prep-react-testing-libr
ary-guide (master)
$ git add .
warning: LF will be replaced by CRLF in src/__snapshots__/App.test.js.snap.
The file will have its original line endings in your working directory

Vedant@LAPTOP-RO2MGGOP MINGW64 /d/Users/Vedant/Documents/prep-react-testing-libr
ary-guide (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   .gitignore
        new file:   README.md
        new file:   package-lock.json
        new file:   package.json
        new file:   public/favicon.ico
        new file:   public/index.html
        new file:   public/logo192.png
```

You can see all the files that were pushed to local repo. Now we need to commit this changes we type:

```
git commit -m "Initial Commit"
```

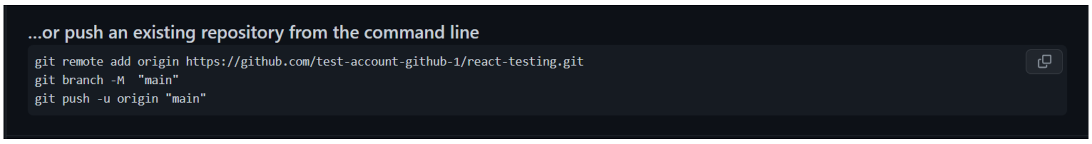
`-m` is for the message we want to associate with that commit



```
MINGW64:/d/Users/Vedant/Documents/prep-react-testing-library-guide

Vedant@LAPTOP-RO2MGGOP MINGW64 /d/Users/Vedant/Documents/prep-react-testing-libr
ary-guide (master)
$ git commit -m "Initial Commit"
[master (root-commit) 05b84f8] Initial Commit
 26 files changed, 34897 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 README.md
 create mode 100644 package-lock.json
 create mode 100644 package.json
 create mode 100644 public/favicon.ico
 create mode 100644 public/index.html
 create mode 100644 public/logo192.png
 create mode 100644 public/logo512.png
 create mode 100644 public/manifest.json
 create mode 100644 public/robots.txt
 create mode 100644 src/App.css
```

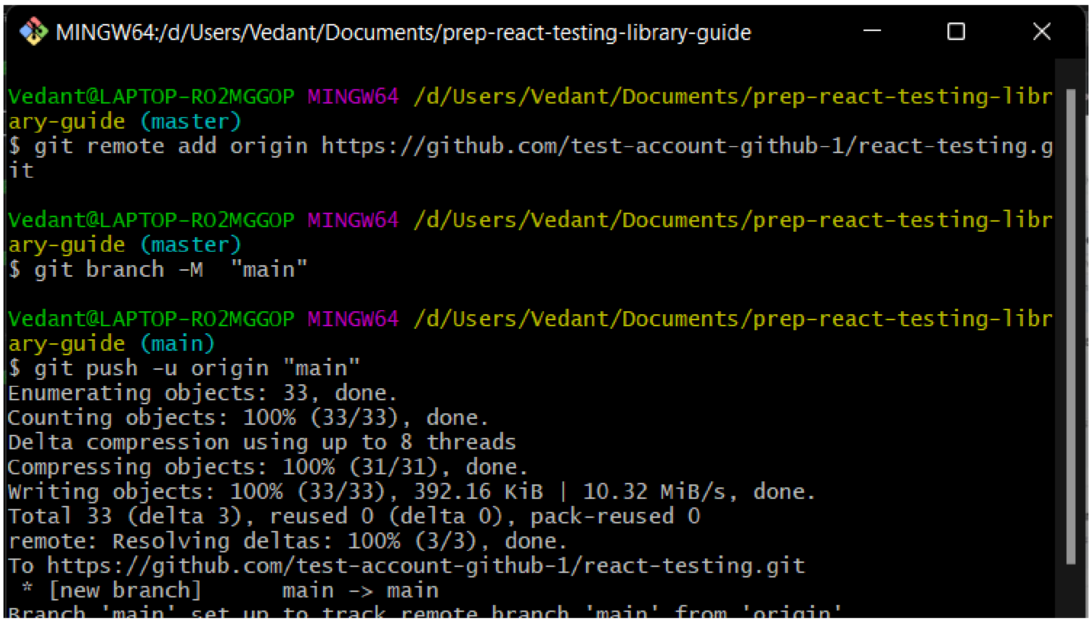
Now our local repo setup is completed! Now we will push this local repo to remote repo we created on github.

On the home page of your github repo you can see the commands to do this:



...or push an existing repository from the command line

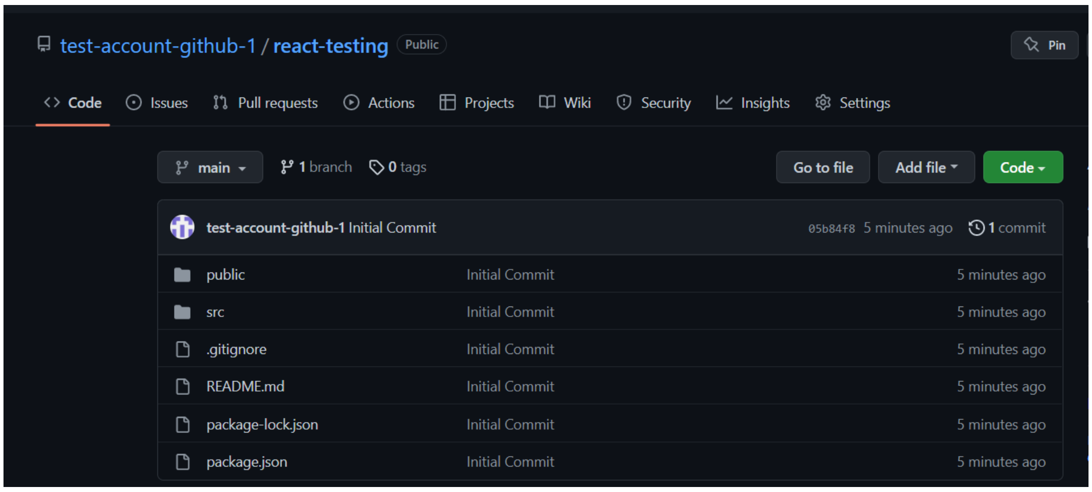
```
git remote add origin https://github.com/test-account-github-1/react-testing.git
git branch -M "main"
git push -u origin "main"
```



```
MINGW64:/d/Users/Vedant/Documents/prep-react-testing-library-guide

Vedant@LAPTOP-RO2MGGOP MINGW64 /d/Users/Vedant/Documents/prep-react-testing-libr
ary-guide (master)
$ git remote add origin https://github.com/test-account-github-1/react-testing.g
it

Vedant@LAPTOP-RO2MGGOP MINGW64 /d/Users/Vedant/Documents/prep-react-testing-libr
ary-guide (master)
$ git branch -M  "main"

Vedant@LAPTOP-RO2MGGOP MINGW64 /d/Users/Vedant/Documents/prep-react-testing-libr
ary-guide (main)
$ git push -u origin "main"
Enumerating objects: 33, done.
Counting objects: 100% (33/33), done.
Delta compression using up to 8 threads
Compressing objects: 100% (31/31), done.
Writing objects: 100% (33/33), 392.16 KiB | 10.32 MiB/s, done.
Total 33 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), done.
To https://github.com/test-account-github-1/react-testing.git
 * [new branch]      main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'
```

After executing these commands your local repo is now pushed to your remote repo on github

You can refresh your github repo to see the changes:



test-account-github-1 / react-testing Public

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

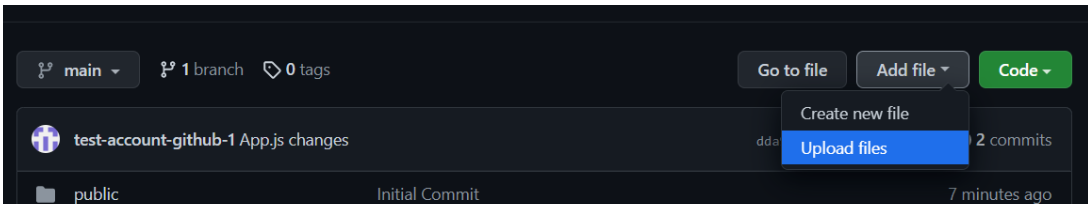
main · 1 branch · 0 tags · Go to file · Add file · Code

test-account-github-1 Initial Commit · 05b84f8 5 minutes ago · 1 commit

public	Initial Commit	5 minutes ago
src	Initial Commit	5 minutes ago
.gitignore	Initial Commit	5 minutes ago
README.md	Initial Commit	5 minutes ago
package-lock.json	Initial Commit	5 minutes ago
package.json	Initial Commit	5 minutes ago

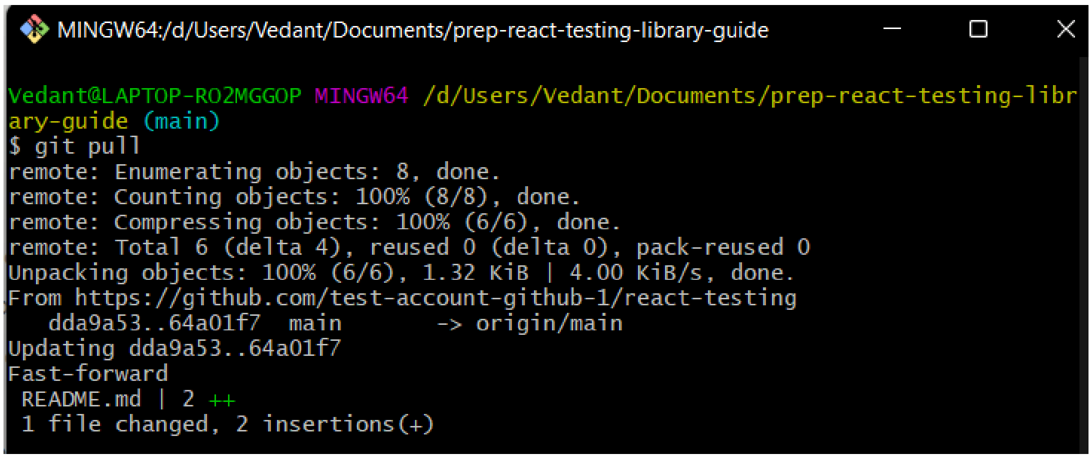
Now whenever you push changes on local repo you just have to push those on remote repo that's it!!

You can also add files right from github itself:



Now if some changes are made only on remote repo and you want to get those changes in remote repo you can pull those. For Example let's edit th readme file on the github repo.

To pull the changes we use the command `git pull`


MINGW64:/d/Users/Vedant/Documents/prep-react-testing-library-guide

```
Vedant@LAPTOP-RO2MGGOP MINGW64 /d/Users/Vedant/Documents/prep-react-testing-libr
ary-guide (main)
$ git pull
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 6 (delta 4), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (6/6), 1.32 KiB | 4.00 KiB/s, done.
From https://github.com/test-account-github-1/react-testing
   dda9a53..64a01f7  main       -> origin/main
Updating dda9a53..64a01f7
Fast-forward
 README.md | 2 ++
 1 file changed, 2 insertions(+)
```

That's it. This is how you can manage your project using git and github!!

Deploying React App on GitHub Pages

A project that just uses JavaScript, HTML and CSS is simple to host on GitHub Pages. Projects that are built in React, Vue or Angular require son configurations, though.

We will deploy the same app that we just pushed on our remote github repo.

In order for us to be able to upload our built application to GitHub Pages, we first need to install the gh-pages package.

```
npm i gh-pages
```

This package will help us to deploy our code to the gh-pages branch which will be used to host our application on GitHub Pages.

To allow us to use the gh-pages package properly, we need to add two keys to our scripts value in the package.json file:

```
"scripts": {
    "start": "react-scripts start",
    "predeploy": "npm run build", <----------- #1
    "deploy": "gh-pages -d build", <---------- #2
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
```

Next, we need to modify our package.json file by adding the **homepage** field. This field is used by React to figure out the root URL in the built HTML fil In it, we will put the URL of our GitHub repository.

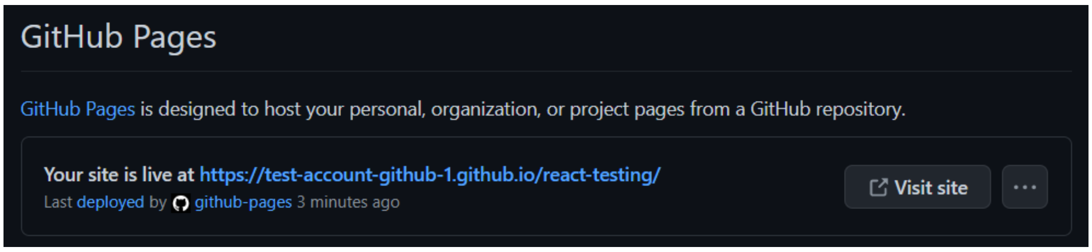
```
{
  "name": "starter-project",
  "homepage": "https://github.com/test-account-github-1/react-testing", <----
  "version": "0.1.0",
  /....
}
```

To deploy our application, type the following in the terminal:

```
npm run deploy
```

Running the command above takes care of building your application and pushing it to a branch called gh-pages, which GitHub uses to link with GitHu Pages.

You will know that the process was successful if at the end of it you see the word **Published**. We can now head to our GitHub repository under Settinc and scroll down to the GitHub Pages section.


GitHub Pages

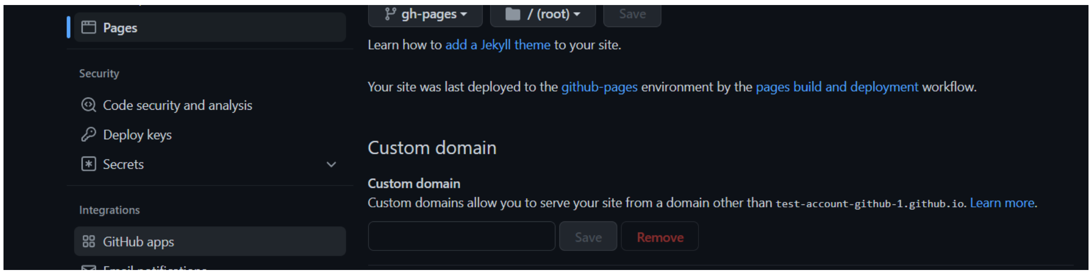
GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Your site is live at https://test-account-github-1.github.io/react-testing/
Last deployed by github-pages 3 minutes ago

Deploying to a GitHub custom subdomain

1. Purchase a domain name from a domain service provider of your choosing (e.g., Namecheap or GoDaddy)

2. Connect the custom domain to GitHub Pages. To do so, click on the **Pages** menu on the **Settings** tab. Next, scroll down to the **Custom domain** field and enter the newly purchased domain name. This will automatically create a commit with a `CNAME` file at the root of your repository



Learn how to add a Jekyll theme to your site.

Your site was last deployed to the github-pages environment by the pages build and deployment workflow.

Custom domain

Custom domain
Custom domains allow you to serve your site from a domain other than `test-account-github-1.github.io`. Learn more.

3. Ensure the `CNAME` record on your domain service provider points to the GitHub URL of the deployed website (in the case of this example, nelsonmic.github.io/logdeploy/). To do so, navigate to the **DNS management** page of the domain service provider and add a `CNAME` record that points to `username.github.io` where `username` is your GitHub username

Interview Questions

- **What does git status command do?**

`git status` command is used for showing the difference between the working directory and the index which is helpful for understanding git in-dep and also keep track of the tracked and non-tracked changes.

- **What does git add command do?**

1. This command adds files and changes to the index of the existing directory.

2. You can add all changes at once using `git add .` command.

3. You can add files one by one specifically using `git add <file_name>` command.

4. You can add contents of a particular folder by using `git add /<folder_name>/` command.

- **Why do we not call git “pull request” as “push request”?**

1. “Push request” is termed so because it is done when the target repository requests us to push our changes to it.

2. “Pull request” is named as such due to the fact that the repo requests the target repository to grab (or pull) the changes from it.

- **What is a conflict?**

1. Git usually handles feature merges automatically but sometimes while working in a team environment, there might be cases of conflicts such as:1. When two separate branches have changes to the same line in a file2. A file is deleted in one branch but has been modified in the other.

2. These conflicts have to be solved manually after discussion with the team as git will not be able to predict what and whose changes have to be given precedence.

- **How will you resolve conflict in Git?**

1. Conflicts occur whenever there are multiple people working on the same file across multiple branches. In such cases, git won't be able to resolve it automatically as it is not capable of deciding what changes has to get the precedence.

2. Following are the steps are done in order to resolve git conflicts:1. Identify the files that have conflicts.2. Discuss with members who have worked on the file and ensure that the required changes are done in the file.3. Add these files to the staged section by using the git add command.4. Commit these changes using the git commit command.5. Finally, push the changes to the branch using the git.

Additional Resources

- https://education.github.com/pack

Thank You !