2.1.5 Create a README File

Now that our development environment is set up, let's add some documentation to give people a quick overview of the Run Buddy project. In GitHub, this is handled by a `README.md` file. READMEs are typically written in Markdown, which offers some lightweight HTML and is widely used on GitHub, but they can also be regular text files.

DEEP DIVE

DEEP DIVE

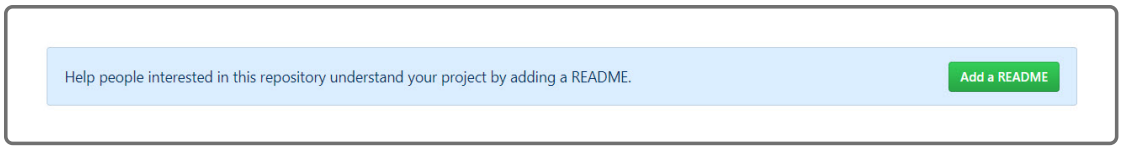
For a closer look, check out the **GitHub Guides on documenting your projects (https://guides.github.com/features/wikis/)** .

The `README.md` file is located in the top level or root directory of the repo. This allows GitHub to display it on the repo's landing page.

REWIND

Like HTML, Markdown uses syntax and HTML notation for styling. For more information, see this **Markdown guide on basic syntax (https://www.markdownguide.org/basic-syntax)** or check out the **Wikipedia page on the Markdown language (https://en.wikipedia.org/wiki/Markdown)**.

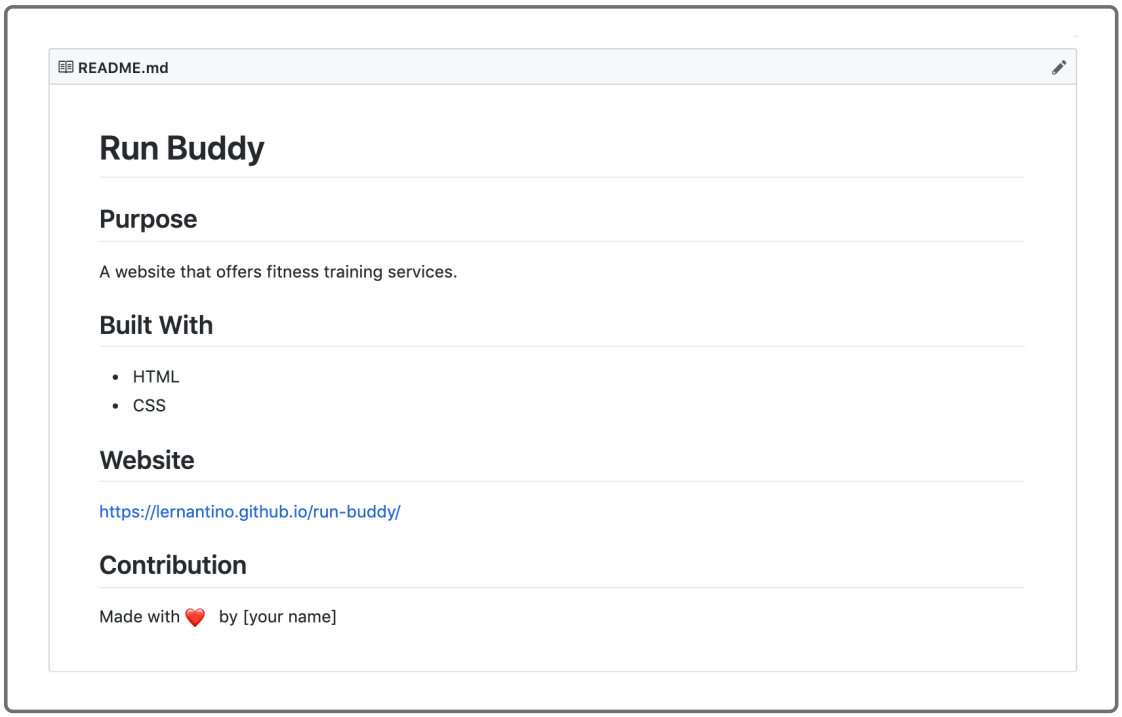
Let's go ahead and create a `README.md` file for Run Buddy. Navigate to your repo's homepage in GitHub. You'll see a call to action to add a README, as the following image shows:



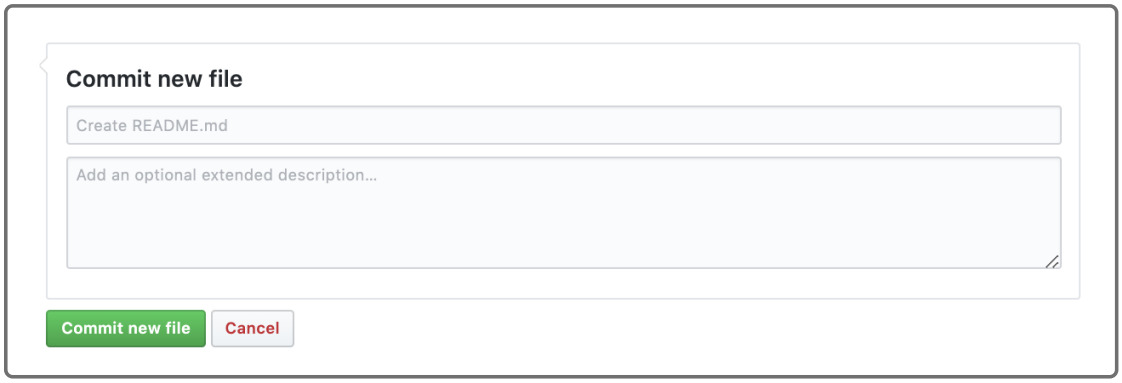
Click on the "Add a README" button to open a text editor within GitHub. In this editor, type the following Markdown code:

```
# Run Buddy

## Purpose
A website that offers fitness training services.

## Built With
* HTML
* CSS

## Website
https://lernantino.github.io/run-buddy/

## Contribution
Made with ❤ by [your name]
```

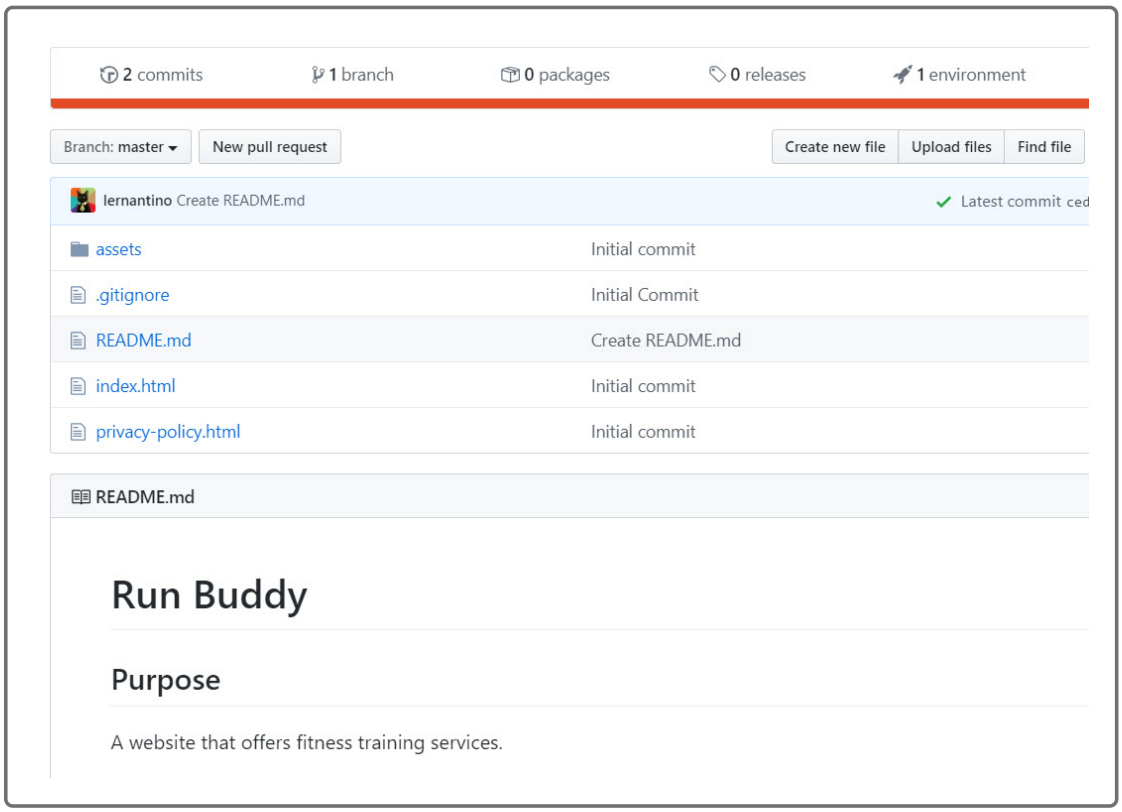
Click the preview tab to view the stylized version:



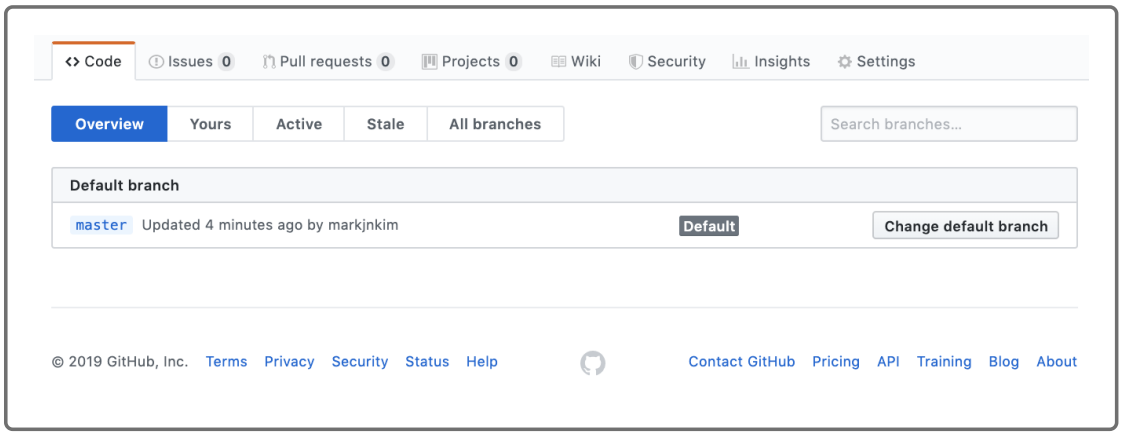
Add a message for the commit (you can keep the default "Create README.md") and then click the "Commit new file" button:



To check if the `README.md` file was created properly, see if it's visible on the repo's landing page:



At the top of this image, we see the branches tab. This is a subset of tabs located in the Code tab. If you don't see the branches option, the Code tab may not be selected. Click on this option to look at our branches to see the following:



PAUSE

Why do you see the `master` branch but not the `develop` branch we just created?

The `develop` branch was created locally on your computer, but this change hasn't been updated yet on GitHub. The repo on GitHub is considered the **remote repo**. As we did in the past module, we'll need to `git push` our local changes up to the remote repo in GitHub.

Hide Answer

Pushing our branch to GitHub creates a secure backup and provides a collaborative and communal development branch that team members can branch from. When pushing a branch, though, the syntax will be different, as seen here:

```
git push origin <remote-branch-name>
```

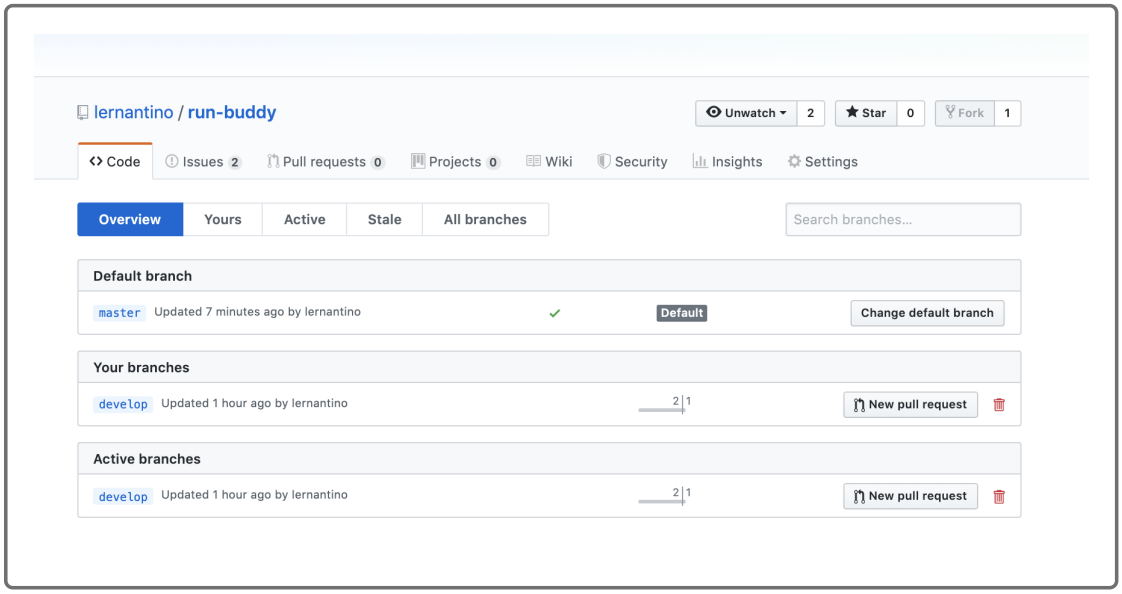
Did you notice something new here? In Module 1, we pushed our changes to the remote `master` branch. But now we're creating and pushing to a new `develop` branch, so we added the branch name (`<remote-branch-name>`).

This command does two things: creates a new remote branch and then pushes the code from the local active branch into that remote branch. The word `origin` simply to refers the source repository where the cloning originally occurred.

In our scenario, the Git command to create and push to a remote `develop` branch is:

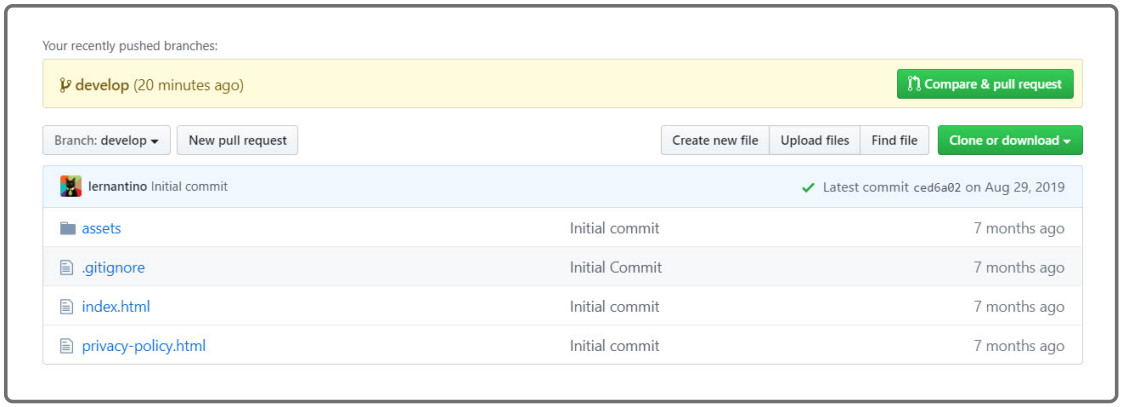
```
git push origin develop
```

After you run that command, look at the branch tab in the run-buddy repo on GitHub to see if it worked:



Yup! There's our new `develop` branch.

If you click on the `develop` link, you can navigate to the `develop` branch of the repo, shown here:



Do you see the notification (highlighted in yellow) that the new `develop` branch was created? Also notice that the "Branch:" dropdown menu now has the ability to select `develop` or `master`.

This remote version serves as the communal or collaborative branch that we will update with our enhancements. It's our shared and stable

development environment. It also serves as a great way for team members to update their local development environment to the latest stable version—they simply need to pull down the `develop` branch from GitHub.

Next, we'll learn how to communicate updates between the remote repo (on GitHub) and the local repo (on our computer).