Class 14 – GitHub Activity Sheet – Software Tools 2024

Developed by Jacqueline May and Jessica Castellanos-Labarcena, updated by Karl Cottenie

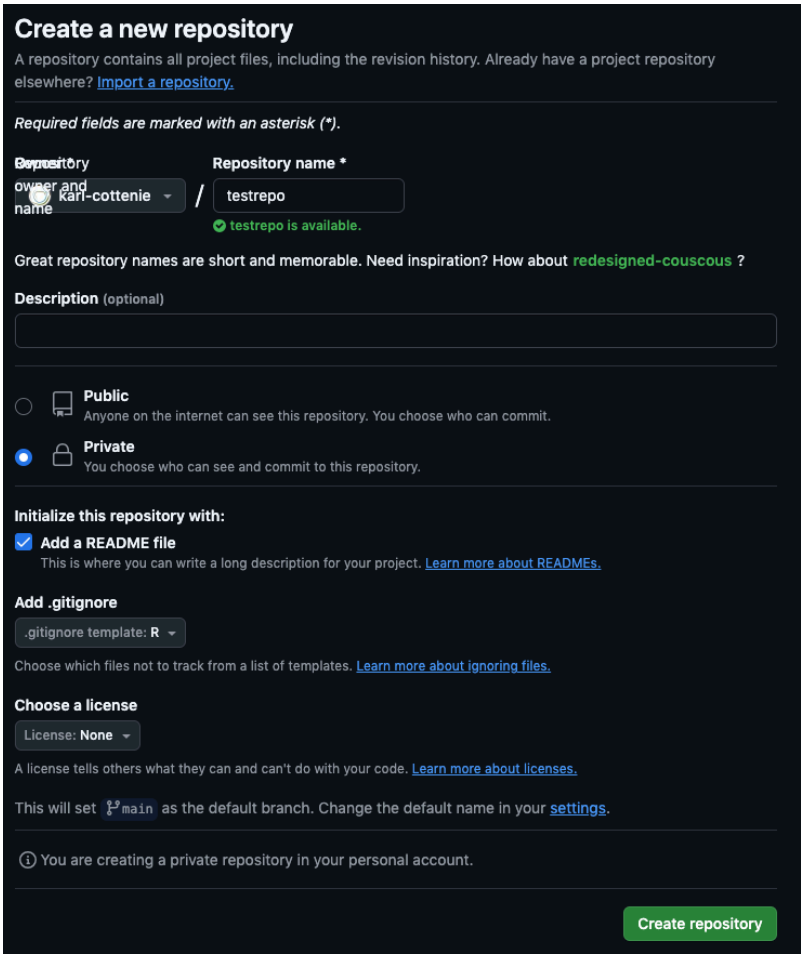
Learning Outcomes

By the end of completing this activity, you should be able to:

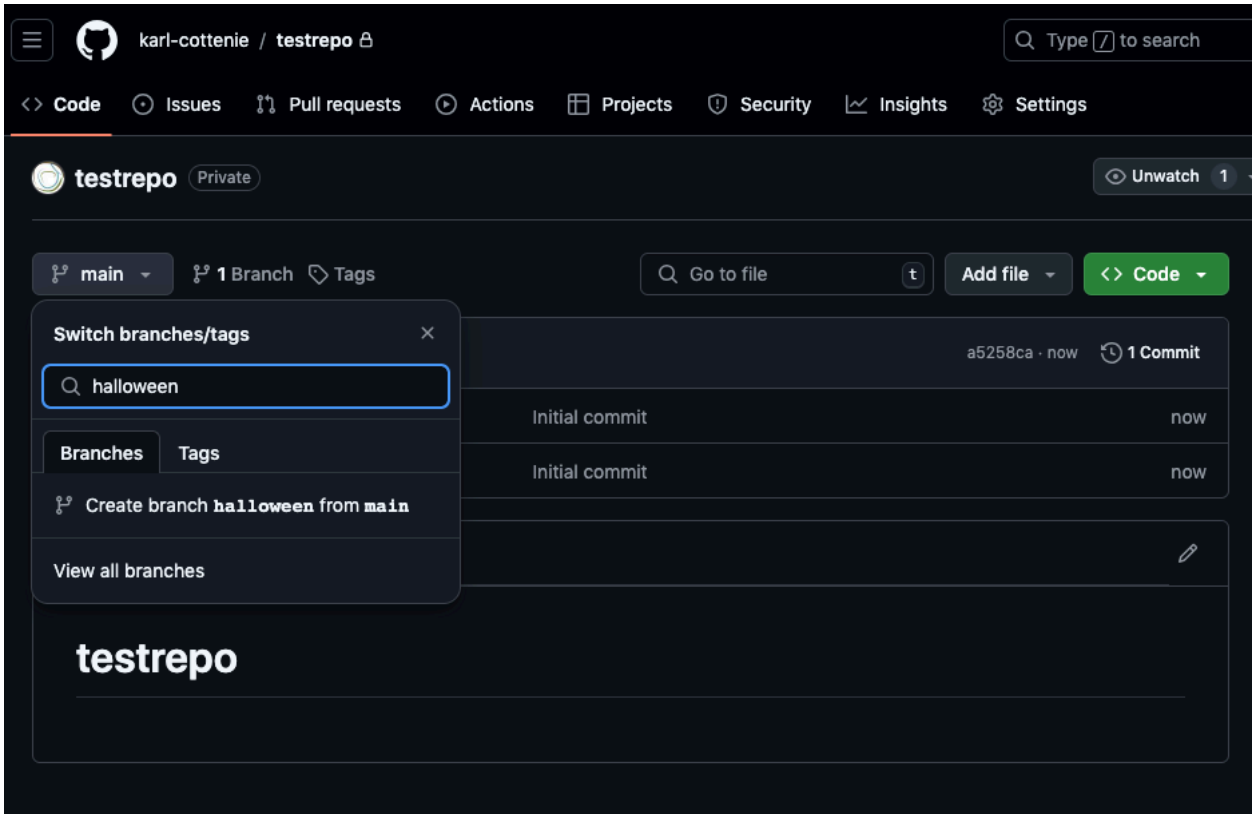
- Use the GitHub browser to create your own repository, learn how to commit changes, and open and merge pull requests.
- Perform all of these steps from within RStudio.

Part 1: GitHub Browser

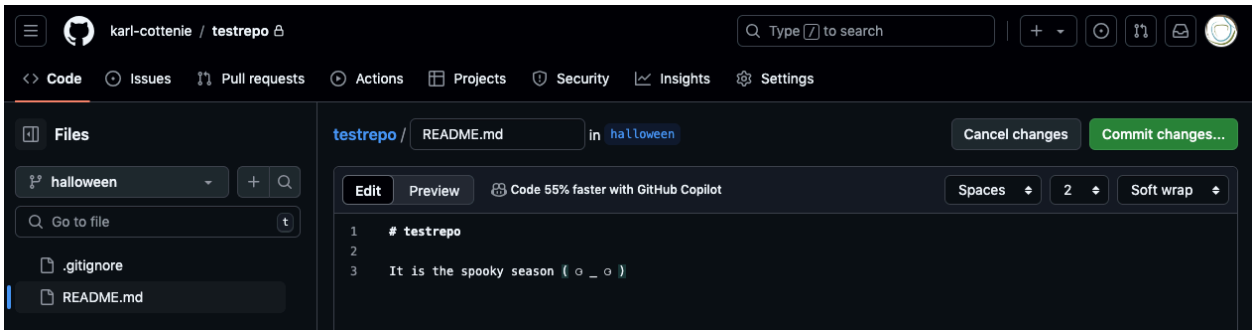
1. Login to your GitHub account on www.github.com.
2. Create a repository
    a. In the upper right corner, click the **"+"** symbol and then select **New repository**.
    b. Give a name to your repository.
    c. Write a short description about the repository.
    d. Select whether you want the repository to be **Public** (anyone can see it) or **Private** (you can choose who sees it).
    e. Under "Initialize this repository with:", select "**Add a README file**".



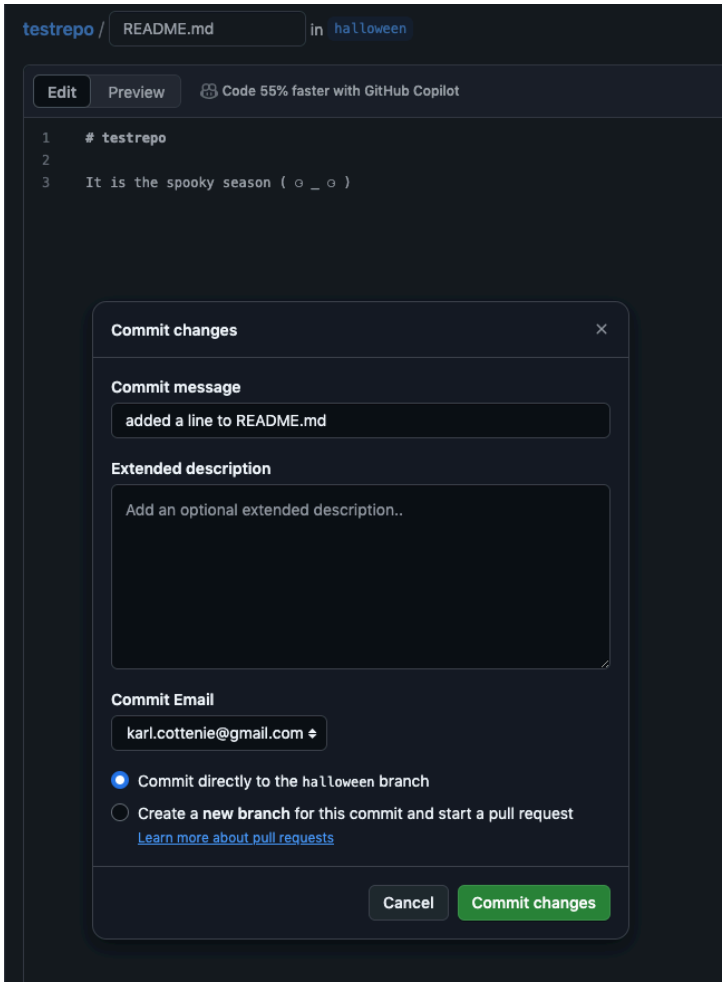
3. Create a new branch
    a. Navigate to your new repository (it should bring you to its main page after you create it).
    b. Click on the drop down menu that says "**main**".
    c. In the text box where it says "*Find or create a branch…*", type the name of your new branch. Under the **Branches** tab, click "*Create branch: feature from 'main'*".



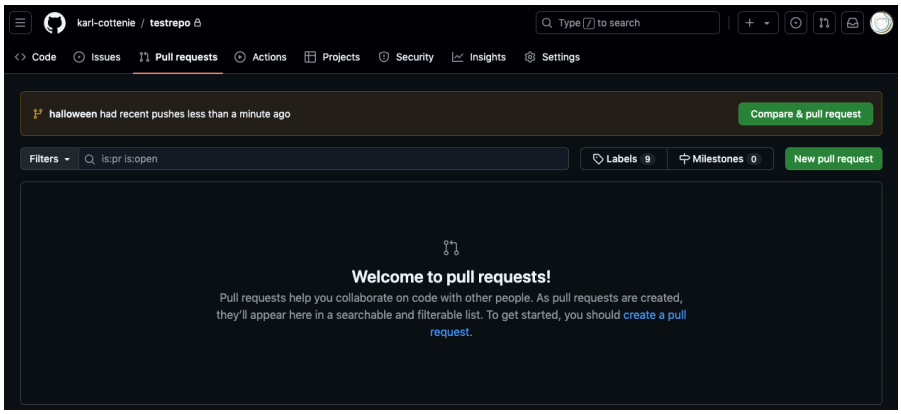
4. You should have two branches now! Let's edit the README.md file and commit this change.
    a. Click the README.md file
    b. Click the **pencil icon** in the right corner to edit this file.
    c. Add a new line to the file.



    d. Click on the green **Commit changes** button.
    e. Write a commit message that describes your edit. Note there is an option for adding an extended description.
    f. The edit should now be present in your README.md file on your new branch (but NOT in the README.md file in your main branch!).

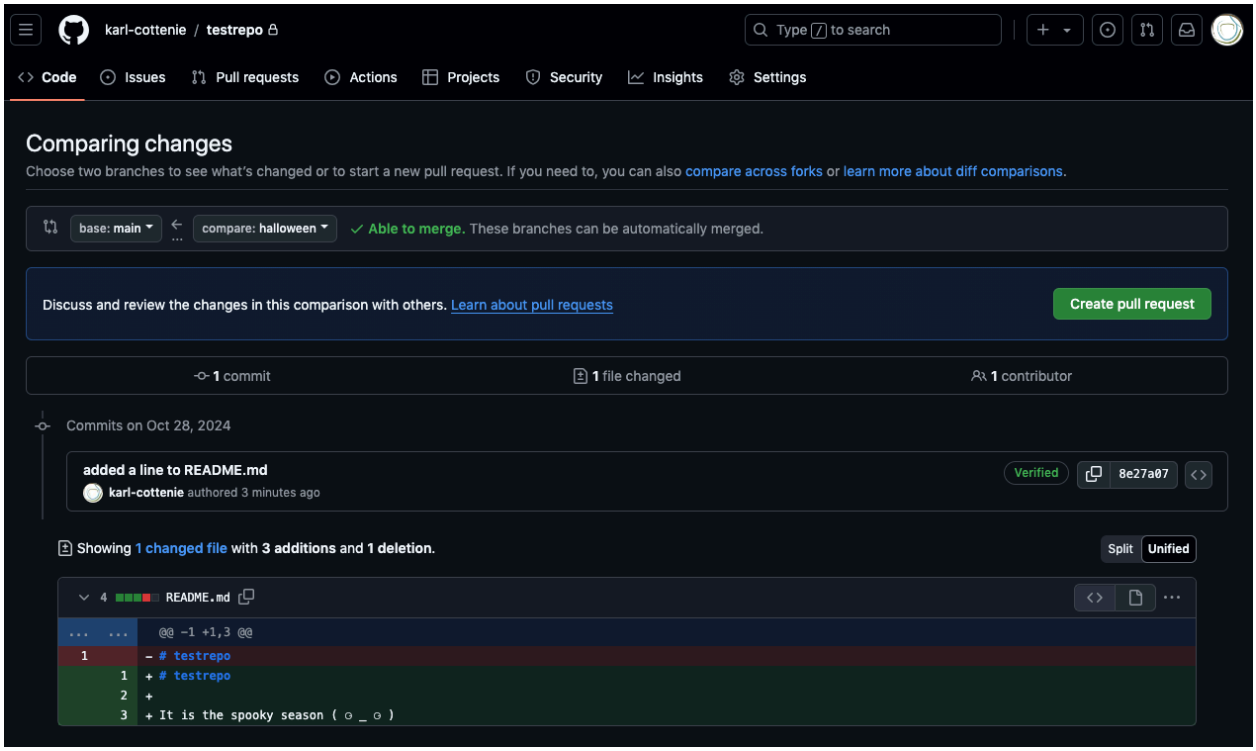


5. Let's open a **Pull Request** now so we can merge our edits in our new branch with our main branch. Note that in this case, we are opening a pull request in our own repository for demonstration purposes.
    a. Click on the **Pull requests** tab.
    b. Click on the green **New pull request** button.

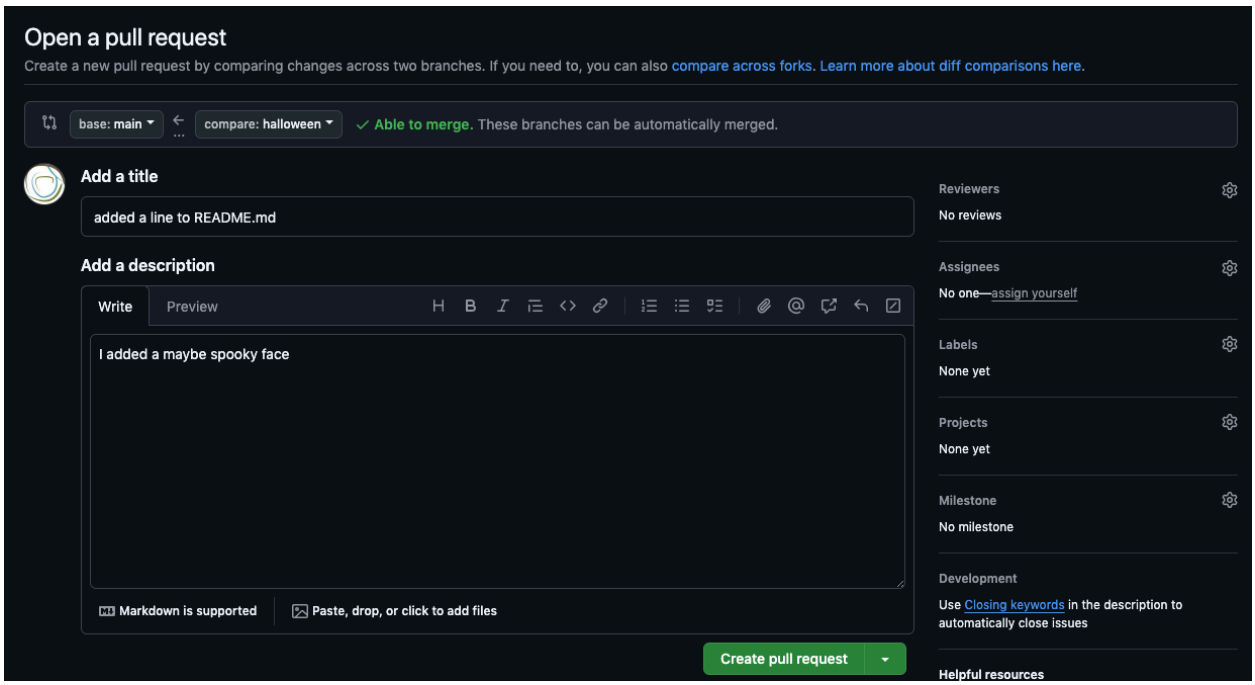


    c. Make sure it is set to **base: main** and **compare: your new branch**.

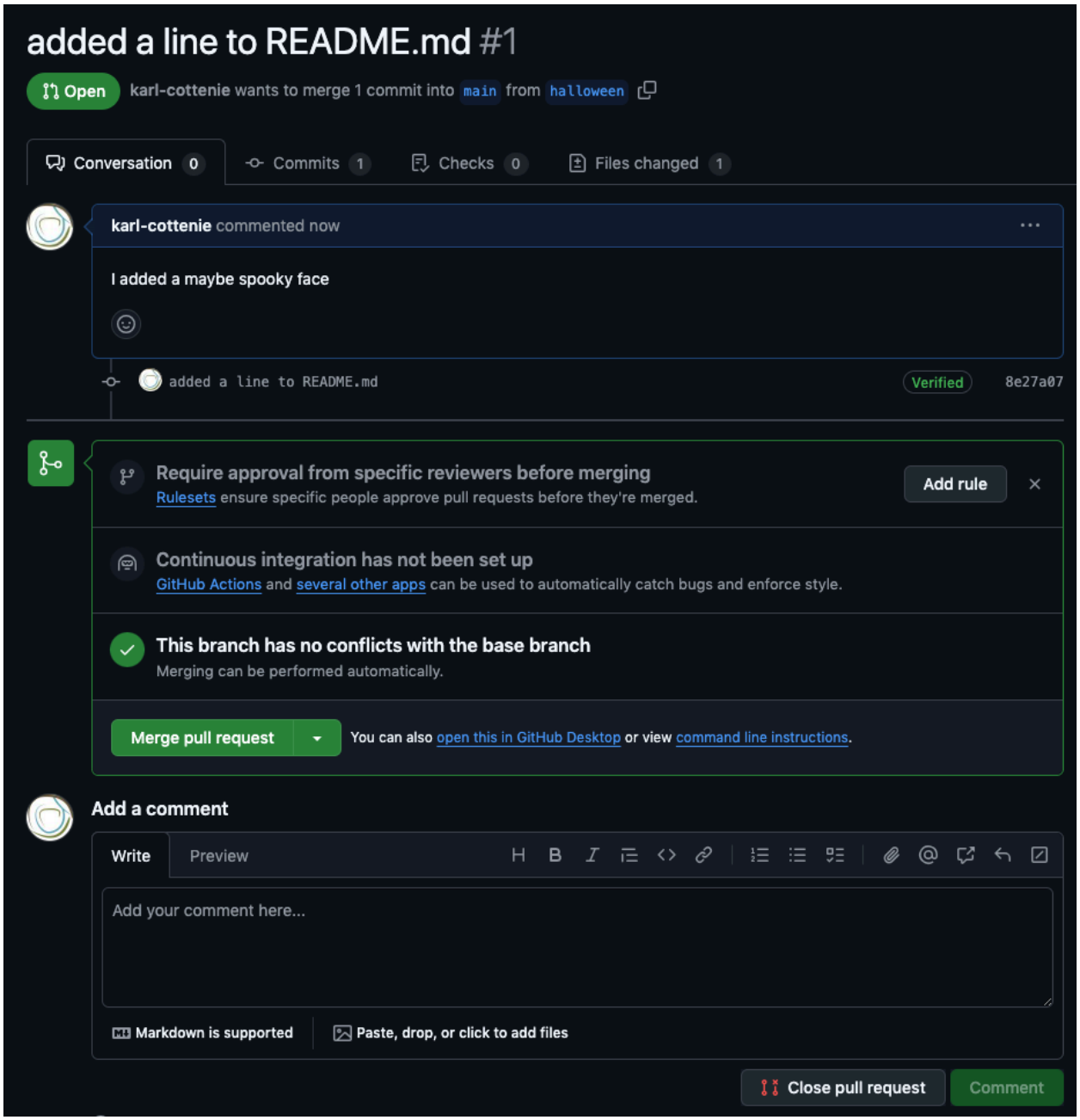
d. Look over the changes and make sure they're what you want to submit. Additions to the file are highlighted in green. Deletions would be in red.
e. When you're ready, click on the green **Create pull request** button



f. Under "Open a pull request," you can add a description of your changes.
g. Finally, click the green **Create pull request** button.



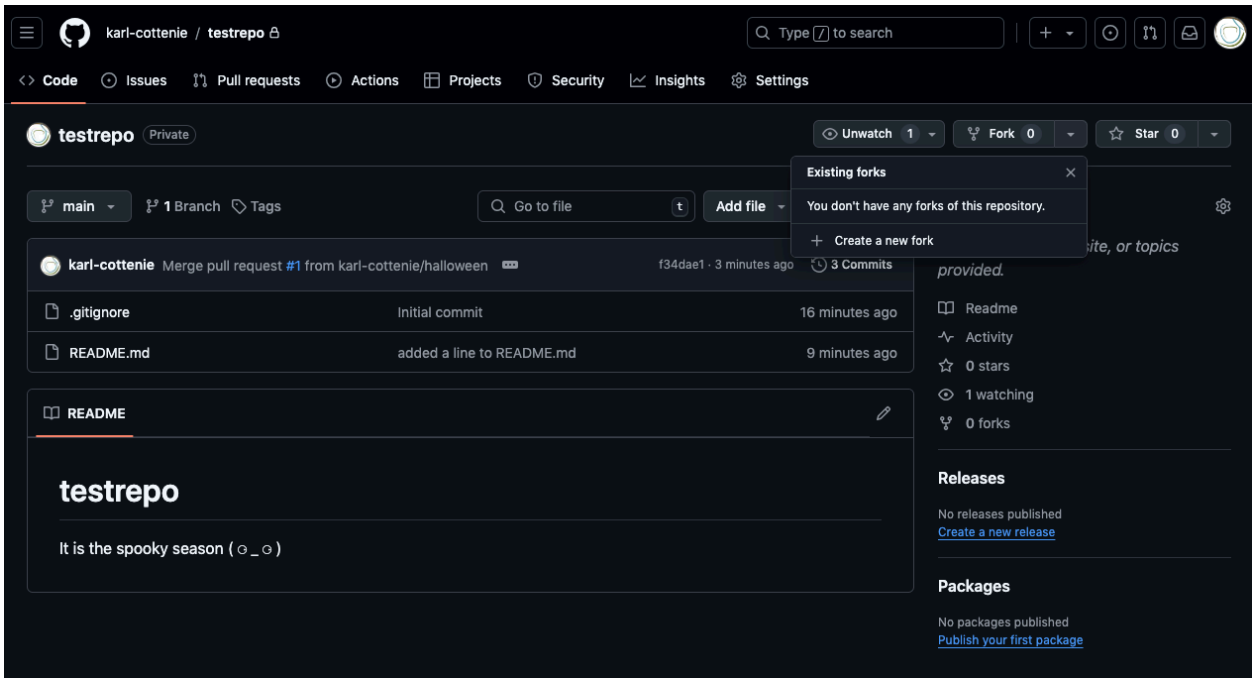
6. Now that we have opened a pull request, let's review it and merge it with our main branch.
    a. Click the green **Merge pull request** button to merge changes on your new branch with the main branch



b. Click the green **Confirm merge** button.
c. Finally, click the **Delete branch** button (you no longer need this branch since the changes have been merged with your main branch!).

Part 2: Forking your group members' repositories

1. Let's add your group members as collaborators to your repository. To do this, first navigate to the main page of your repository you want to add them to.
2. Click on **Settings**.
3. On the left hand side, click on **Collaborators.**
4. Click on the green **Add people** button.
5. Search for your group member's username or email and send them an invite to your repository. Once they accept the invite, they will have access to your repository even if it is private.
6. Now, let's fork our group members' repositories (make a personal copy of their repository).
    a. Navigate to their main repository page.
    b. Click on the **Fork** button in the top right.

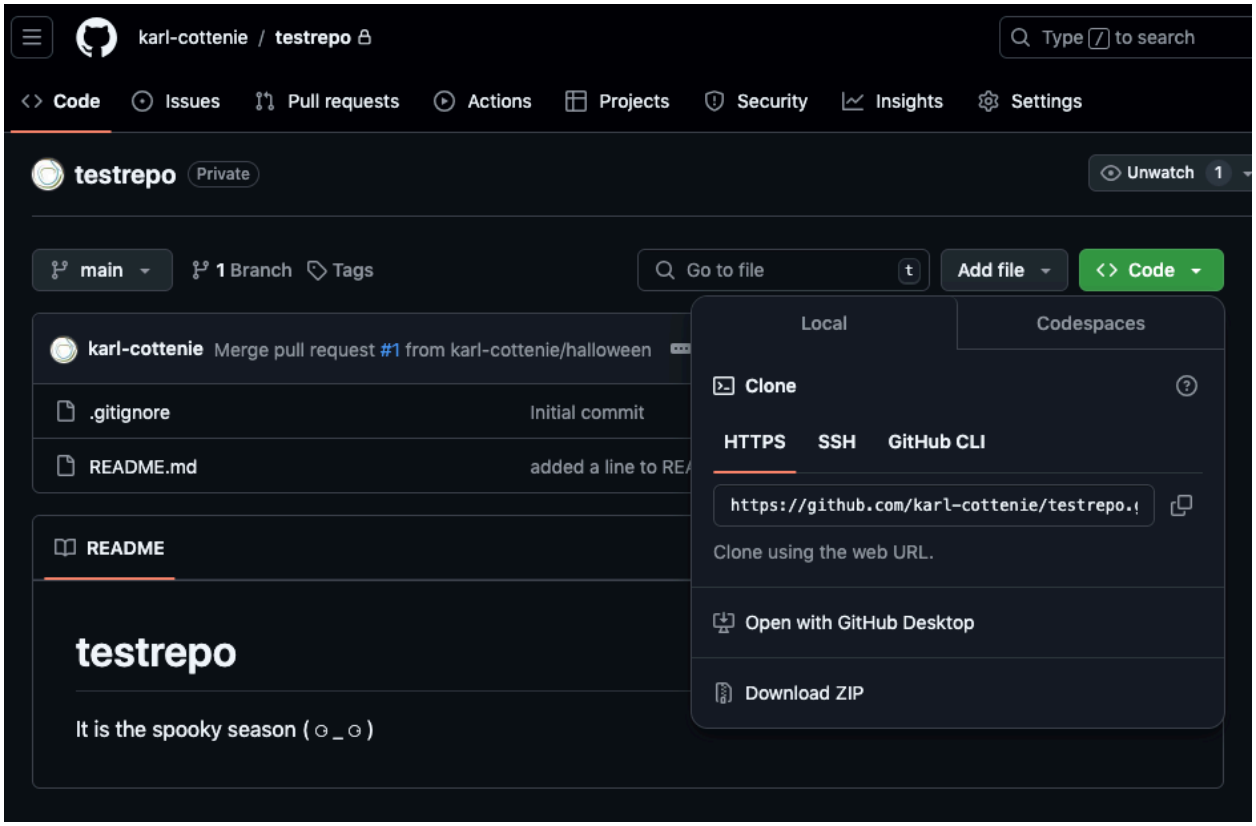


7. You should now have your own personal copy of your group member's repository!

Part 3: Setting up Git in RStudio

We can link our GitHub to RStudio. This is convenient as we can submit changes through RStudio and they will appear on our GitHub accounts online!

1. Navigate to the main page of your repository on GitHub.
2. Click on the green **Code** button.
3. Copy the URL.



4. Open RStudio.
5. Let's tell Git who we are. Click Tools/Shell. You need Git so you can use it at the command line and so RStudio can call it. Enter which git to request the path to your Git executable:

```
[18:23 karlcottenie@iMac:~
$ which git
/usr/bin/git
[18:23 karlcottenie@iMac:~
$ git --version
git version 2.21.1 (Apple Git-122.3)
[18:23 karlcottenie@iMac:~
```

If you are successful, that's great! You have Git already. There is no need to install! Move on. If, instead, you see something more like git: command not found, find instructions for installing git at the end of the document.

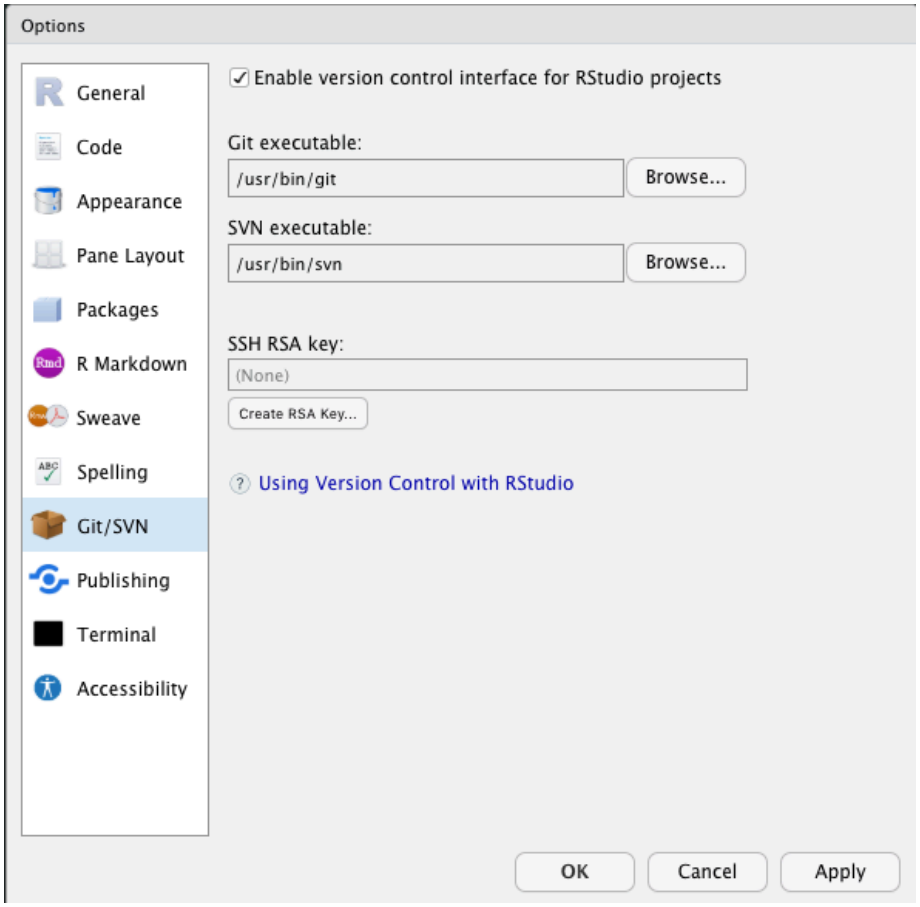
6. In the shell, enter (note the double dashes):

   git config --global user.email "YourEmail.com"

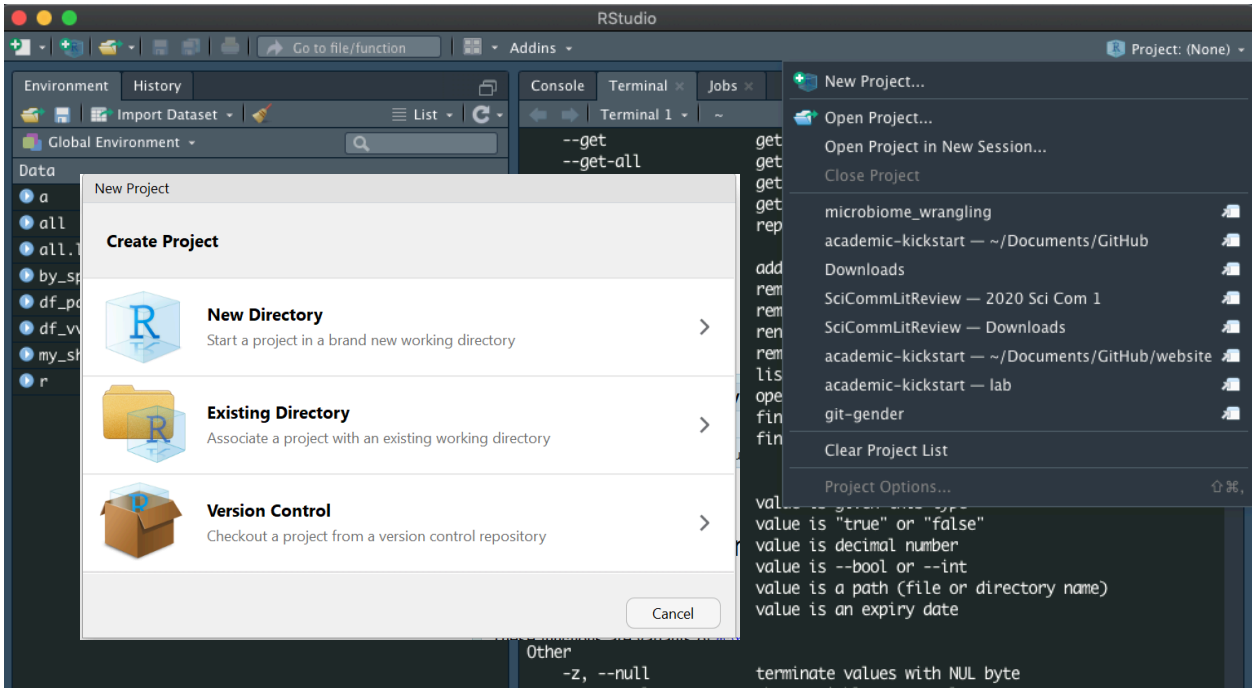
   git config --global user.name "YourUserName"

What username should you give to Git? Although this does not have to be your GitHub username, it can be. Another good option is your actual first name and last name. Your commits will be labelled with this username, so make it informative to potential collaborators and future you. What email should you give to Git? This **must be the email associated with your GitHub account**.

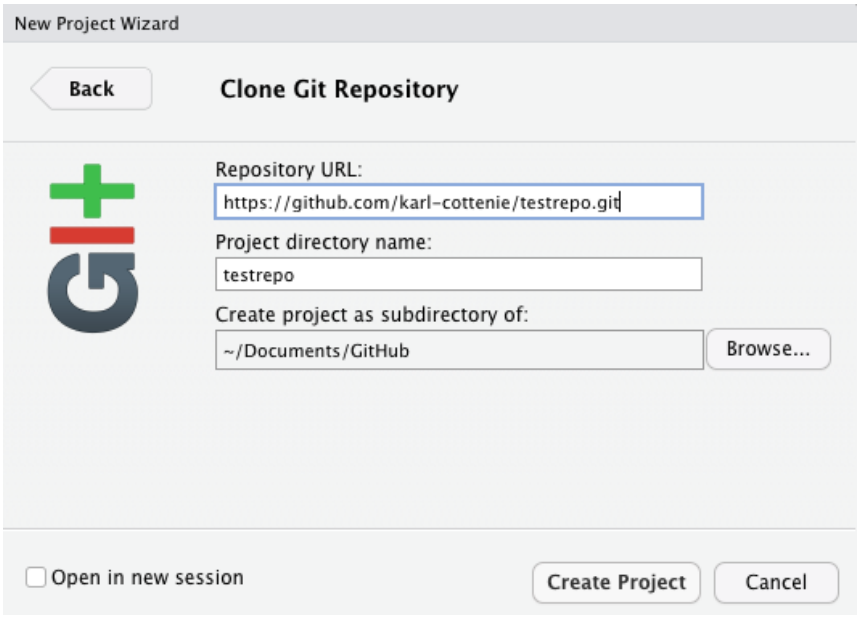
7. Make sure RStudio can find Git:
8. In RStudio, go to: Tools/Global Options -Git/SVN. Make sure the correct path is indicated for Git executable from step 5.



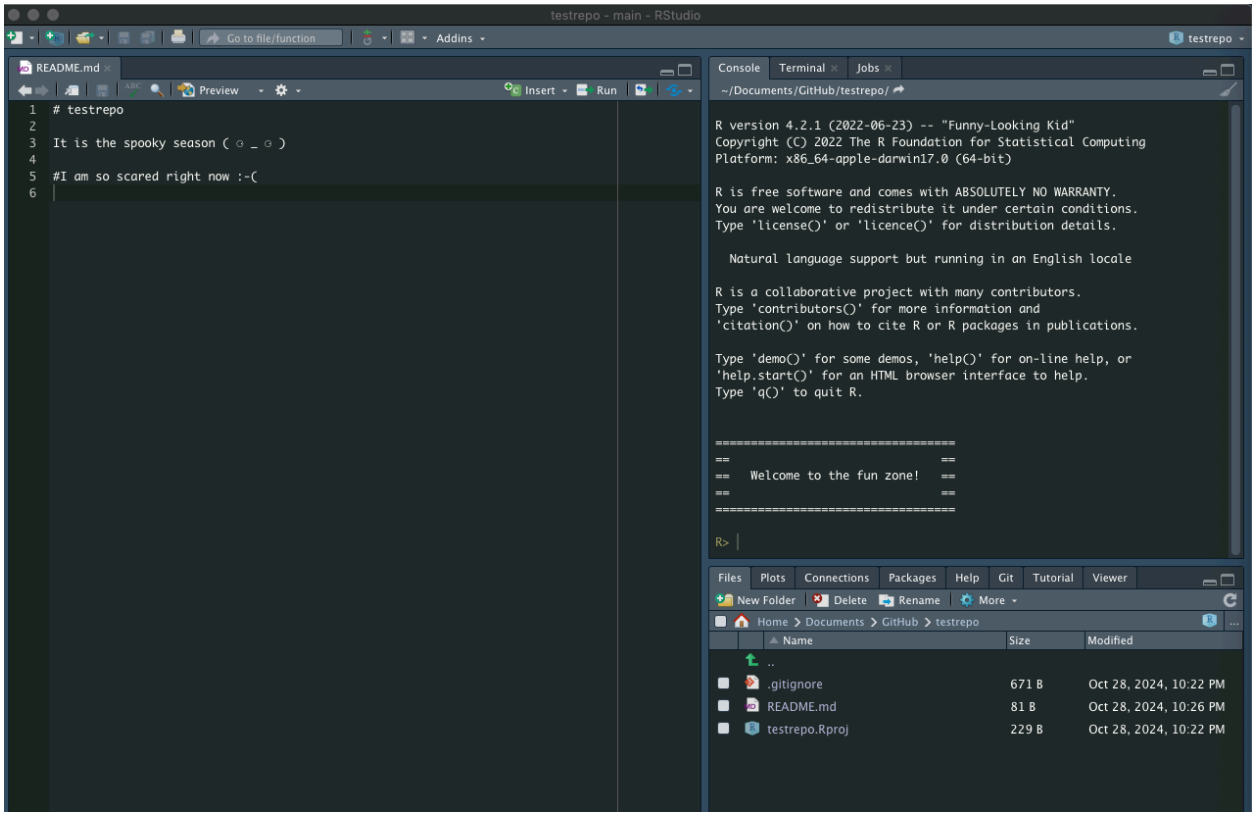
9. Follow the instructions here (https://happygitwithr.com/https-pat) to set your personal access token.
10. In RStudio, click on the **Project** dropdown menu in the top right corner and select **New Project**.



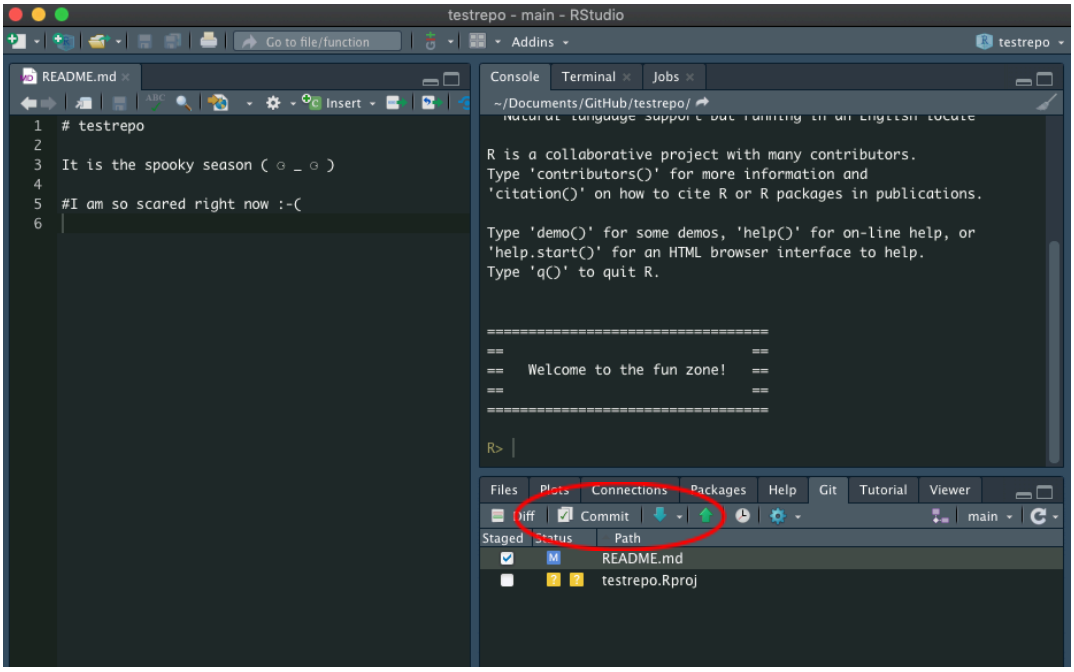
11. Click on **Version Control.**
12. Click on **Git**.
13. Enter the repository URL. This allows you to **clone** the repository into your local R environment.



14. Click **Create Project**. You will be prompted for your username and password. After entering those, all of the files associated with the repository should soon be present in your local R project.
15. Open the README.md file in RStudio and add a line to the file. Save the file.

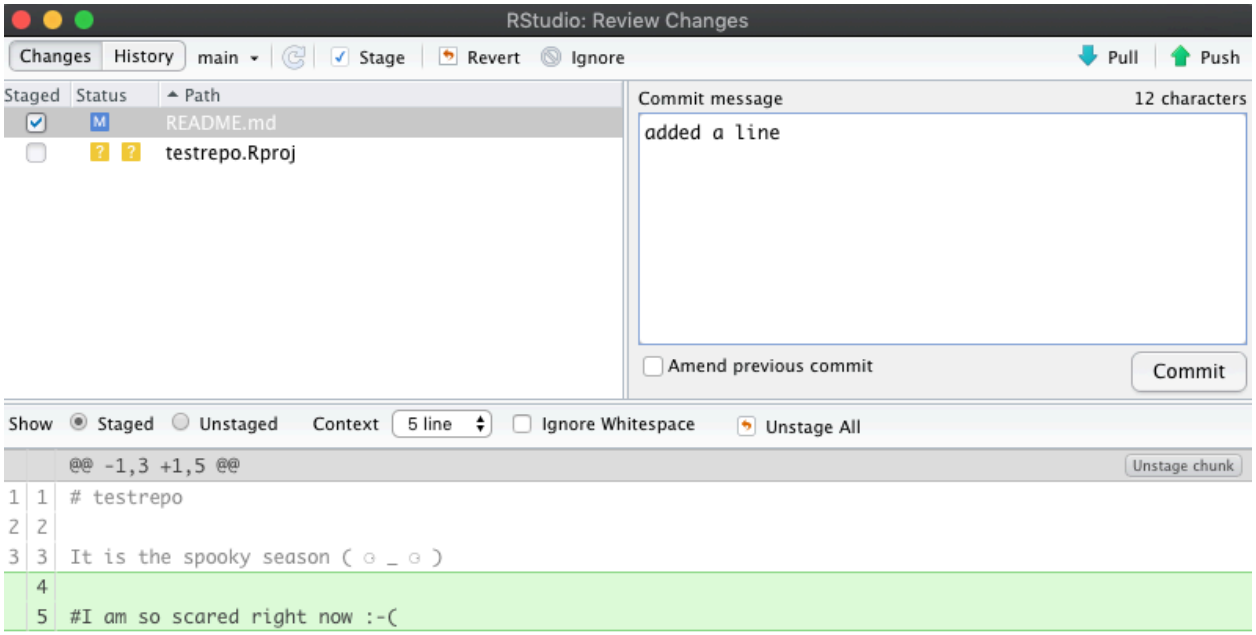


16. Now, let's commit the changes to this local repository. Click the **Git** tab in the upper right pane.
17. Click the **Staged** box for the README.md file you modified.
18. Click **Commit.**



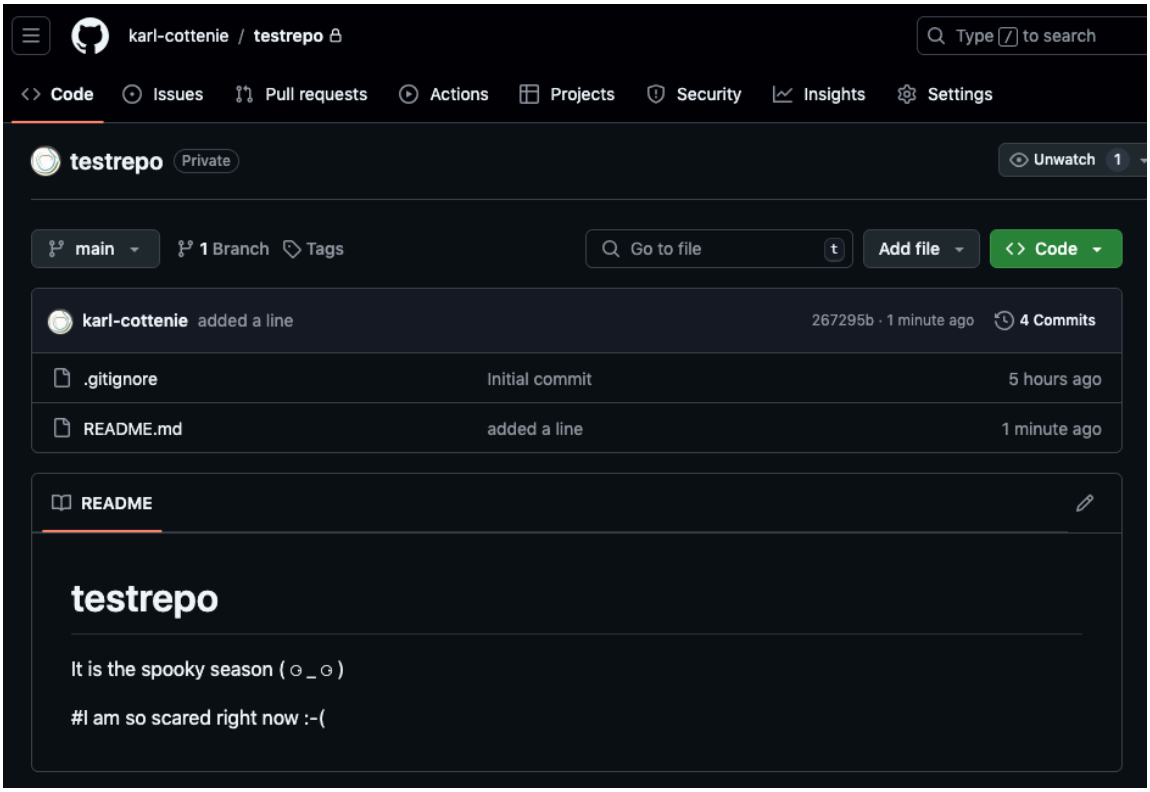
19. Add a description of the change you made to the **Commit message** textbox.

20. Click **Commit**.

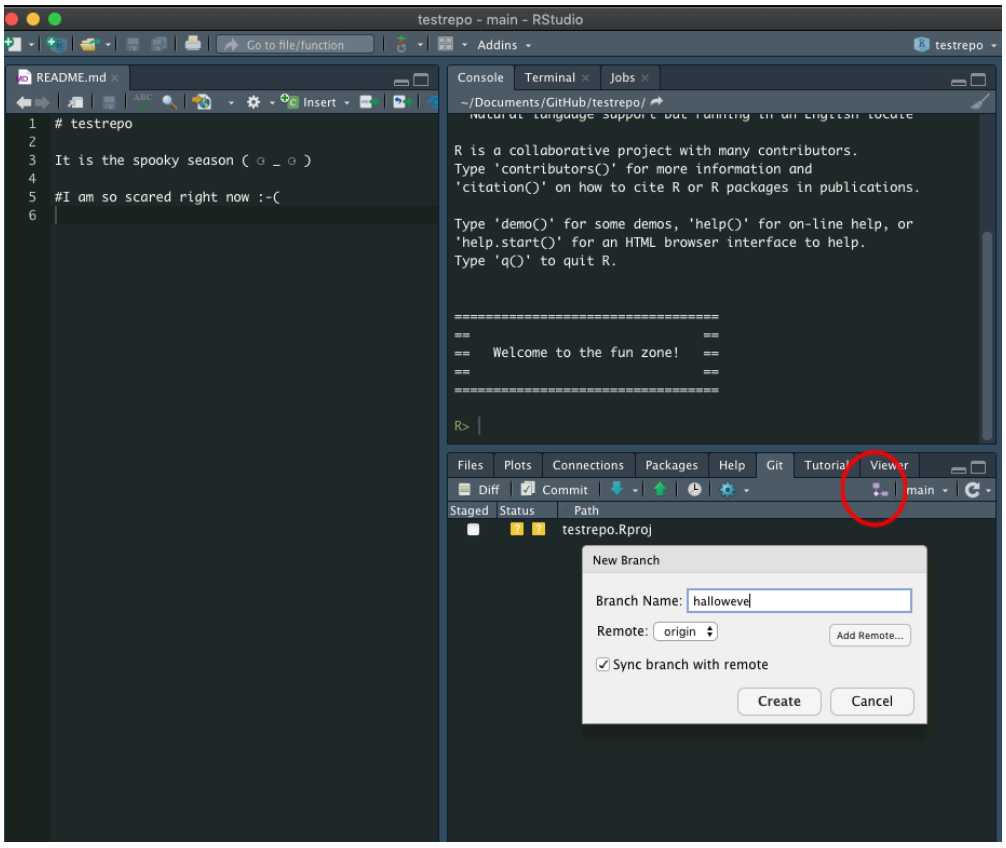


21. To send these changes to the remote repository (i.e. the repository on GitHub), click the green **Push** button.

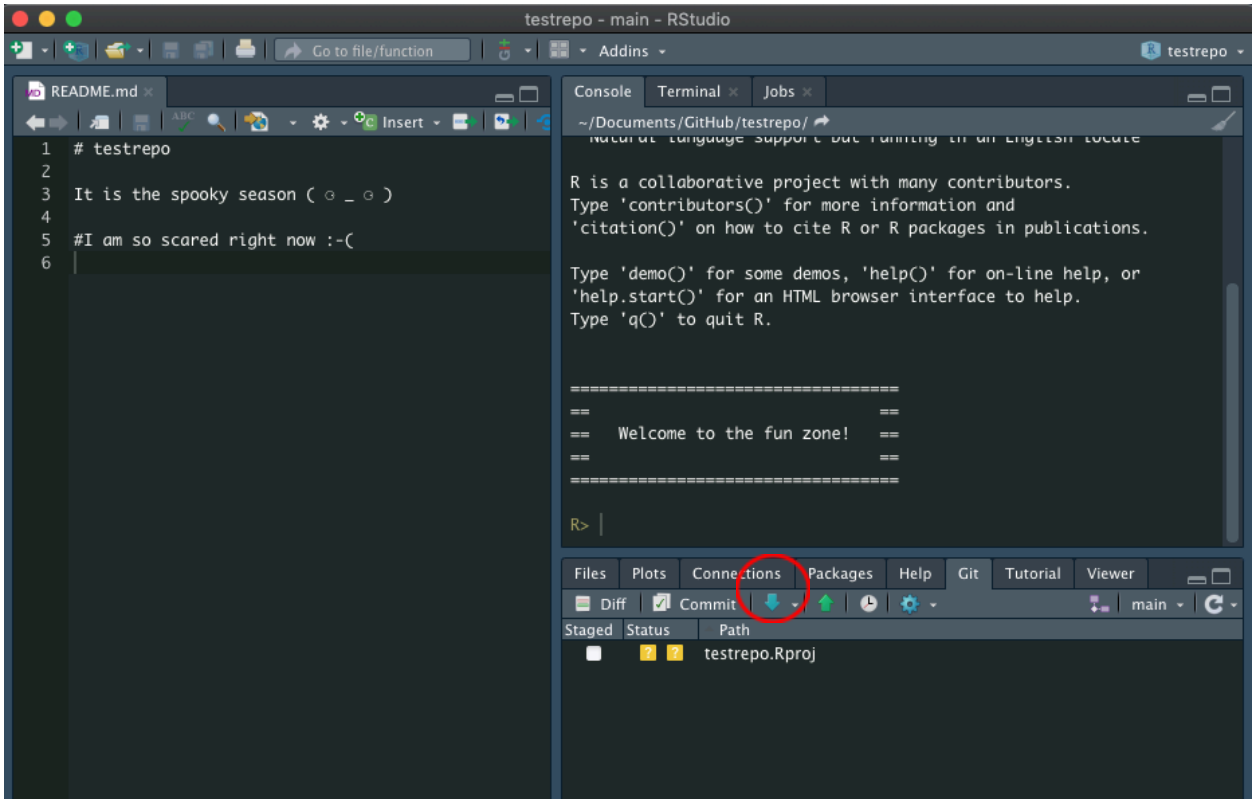
22. Navigate back to your repository on your GitHub account and refresh the page. Confirm that the changes have appeared in your file online. **NOTE**: If this doesn't work for you, that means that you're having issues syncing RStudio to GitHub. Refer to the end of the document for instructions to solve this problem.



23. Note: you can also create branches in RStudio by clicking on this icon:



24. Note: if you wanted to update your local (RStudio) repository to include changes that were made on your remote repository, you would click the blue **Pull** arrow.



Instructions for installing Git.

Please follow the steps recommended in this tutorial (https://happygitwithr.com/install-git)

If you are struggling on Windows, consider there are different types of shell, and you might be in the wrong one. You want to be in a “Git Bash” shell instead of Power Shell or the legacy cmd.exe command prompt.

If you're having issues syncing RStudio to GitHub

When we interact with a remote Git server, such as GitHub, we have to include credentials in the request. This proves we are a specific GitHub user with access to make changes. Git can communicate with a remote server using one of two protocols, HTTPS or SSH, and the different protocols use different credentials.

The https:// clone URLs are available on all repositories, regardless of visibility. https:// clone URLs work even if you are behind a firewall or proxy.

When you git clone, git fetch, git pull, or git push to a remote repository using HTTPS URLs on the command line, Git will ask for your GitHub username and password. When Git prompts you for your password, enter your personal access token. Find below instructions on how to create a PAT. When you create a fine-grained personal access token, you grant it a set of permissions. Set all to write/read.

https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token

If the PAT is not working for you, consider setting up a key for SSH (this requires command-line experience). More details on this site. (https://happygitwithr.com/ssh-keys)