Github Workshop
Anna Schultze, LSHTM Electronic Health Records group
February 7 2023

Welcome to the EHR group github training!

In this practical there are two tasks, one simpler which will show you how to work with git on your own, and one slightly more complex to show the basics of collaborative coding. If you have some github experience you can probably skip straight to the second part.

This document contains detailed instructions, but we recommend you ask questions to 'git' as much as possible out of the workshop.

Before starting, you will need to:

- Have a github account (Go to https://github.com/ and create a new account using the sign up to GitHub box). It's recommend to set up 2FA straight away for added security.
- Have installed github desktop (https://desktop.github.com/). If you prefer you can use the mac terminal/git bash instead of github desktop, but you'd have to ensure you have git installed and configured ahead of time.

These instructions and the exercises live on this repo: https://github.com/ehr-lshtm/github-training, where you can access the latest versions and get all the datasets.

Happy Git-ing!

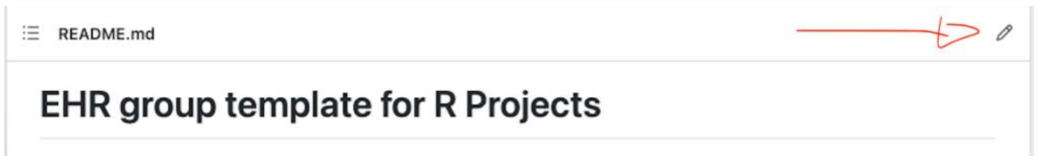
Task 1: Working on your own

In this task, you'll create a new repository from a template and upload a file.

1A. Create a new repository from a template

The first thing we're going to do is create a new repository. I always do this online in Github, and today we're going to be using the EHR groups templates.

1. Navigate to https://github.com/ehr-lshtm
2. Find the template you want to use and click on it, either template-r or template-stata
3. Click "use this template", and "create new repository"
   - Change the "Owner" by using the dropdown menu. This allows you to choose where the repository should be created, and will likely list all organizations you are a member of as well as your own private github. Choose your own private github.
   - Choose a name for your repository. This should be something short and informative that's easy to type
     - "my-test-repo" is good
     - "lshtm-ehr-github-workshop-training-test-repository-feb2023' is bad
   - Add a short description if you like
   - Check "private" to set the repository to private
   - Include a brief description, and select " create"
     - This should now create the repository on your own github page
     -
4. The repository has been created with the default README text. Click the edit button to change the text



5. Edit the text to something descriptive
6. Commit the changes as prompted by Github

- It's fine to commit edits to README straight to the main branch
- Github has proposed an automatic commit message for you: this is the title box where it says "Create README.md"
- You can include an optional description if you want

At the end of these steps, you should have created a new repository on your own Github profile, based on the EHR R or Stata template, with a new README.

1B. Clone the repository

The repository you've just made lives online. To be able to interact with it you need to create a local copy, which is called to clone the repository.

1. Open Github Desktop (ensure you are logged in)
2. Select File → Clone Repository

- This should bring up a list of repositories that exist on your online profile and/or as part of organisations you are part of. Your new repository should be listed.
- If your new repository is not listed, you can still add it by just clicking on the URL option and pasting the URL

3. Remember to select where you want your local repo to live by changing the local path.
4. Select clone

- If you forgot to update the repo location, you can clone it again into a different folder. In fact, you can clone the repository as many times as you want.

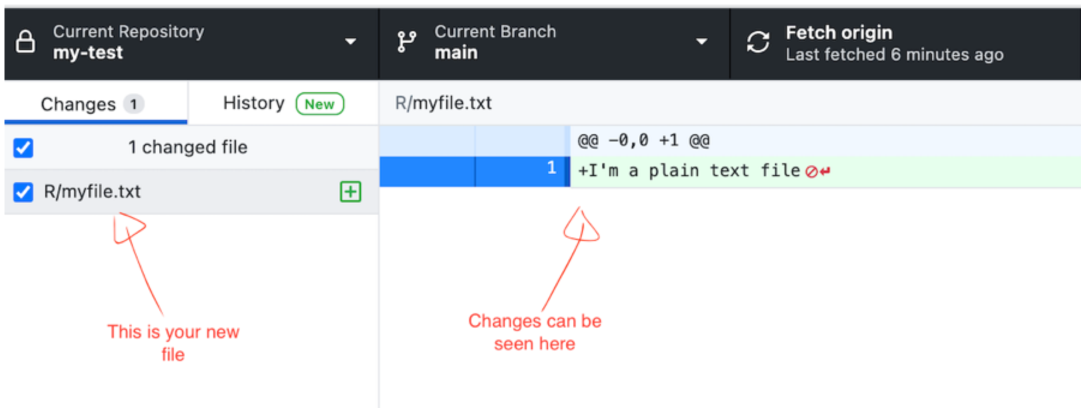
You should now have created a local copy of the repo in the location you've specified. Confirm this by opening the location on your computer.

1C. Add a plain text file and push it back up

We're now going to add a file and upload it.

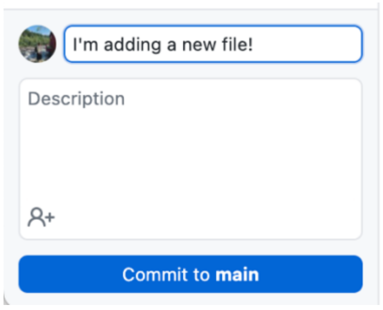
1. Create a new file in some plain text format (.do, .R, .txt – it doesn't matter which one) and save it into an appropriate folder of your local repo
2. This file should now appear in Github desktop, on the left hand pane

- Github desktop will also show you the content of the file and any changes
- Green is an addition compared to what was there before, red is a removal
- The blue tick indicating that the file is "checked" means that it is staged, that is, it will be included in the next commit
- **Always check this list, and uncheck files you do not want to upload**
- **All files listed in the .gitignore will not show up in Github Desktop**



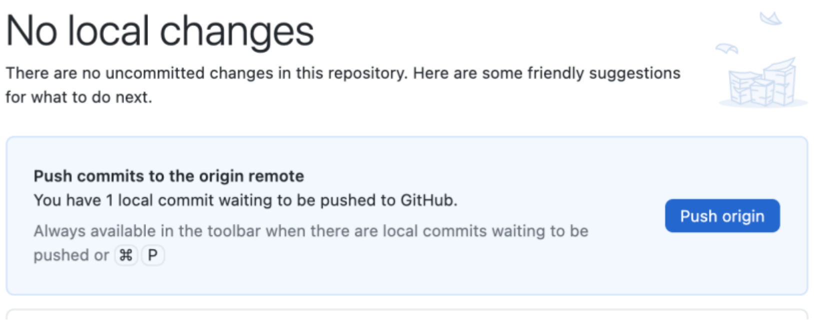
```
R/myfile.txt
@@ -0,0 +1 @@
1 +I'm a plain text file
```

3. To upload this file to Github we'll have to add a commit message, which you do in the bottom box on the left. Github Desktop suggests a message, but you can also add your own. Add a message (or use Github's automatic one) and choose commit.



- This will also show ***where*** you are committing your changes. Here, we are working alone so we are just committing to the main branch.
- It is annoying to move commits between branches (you have to undo them, then add them again), so make sure you're happy with the branch choice before you commit.

4. To upload the file to Github, select the "push origin" button. This means you "push" all the files that are checked in Github desktop to your "origin" repository.

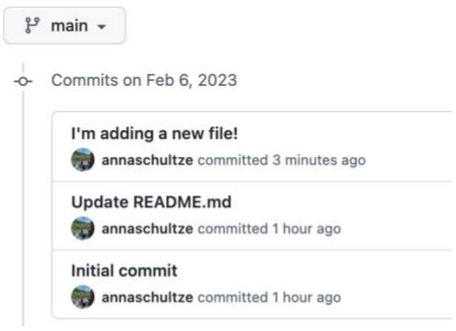


No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Push commits to the origin remote
You have 1 local commit waiting to be pushed to GitHub.
Always available in the toolbar when there are local commits waiting to be pushed or ⌘ P

5. Confirm this worked by going to your github repository and checking for the new file. You should see it in the same folder, and you can also see the time you uploaded this. If you press the "commit" history button, you can go through your repo and see all historical versions of your code.



Commits on Feb 6, 2023

I'm adding a new file!
annaschultze committed 3 minutes ago

Update README.md
annaschultze committed 1 hour ago

Initial commit
annaschultze committed 1 hour ago

These are all the steps needed to use github in the most basic sense – for code sharing, version control and code back-ups, on your own. You don't need to know anything else, but these tools are really most useful for working together with others. The next exercise will show you how this is done.

Task 2: Collaborative Coding

In this task, you'll correct some of Anna's code (in either R or Stata) and open a pull request with some of those changes.

2A. Fork the training repository

1. Navigate to https://github.com/ehr-lshtm/github-training
2. We're going to ***fork*** this repository. To fork the repository, click "fork" in the right hand corner.

   - Make sure the "owner" displayed is your own github name. You can keep the repository name as github-training and the description as is.
   - Click "create fork"
   - The reason we are forking rather than cloning it directly is because a) I want you all to work from your own "version" of this, and not automatically be able to push your changes to the same origin repository (because you're all going to be doing the same changes), and b) I'm not 100% sure everyone has the right permissions to clone this directly from the EHR group.

3. You should now have this repo on your own Github page, and it'll have a little notice saying where you've forked it from.

2B. Clone the repository

1. To be able to add changes to this repository, we now want to download a local copy by ***cloning*** the repo. We follow the same steps as in [1], also outlined below.

2. Go to github desktop, select file → clone repository

   - This should bring up a list of repositories that exist on your online profile and/or as part of organisations you are part of. Your new repository should be listed.
   - If you're a member of the EHR group, be careful to clone your forked repository and not the original one from the EHR group. Your own one will start with your github username, i.e, `annaschultze/github-training`
   - If your new repository is not listed, you can still add it by just clicking on the URL option and pasting the URL. As above, use the URL from your own github repo.

3. Remember to select where you want your local repo to live by changing the local path

4. Select clone

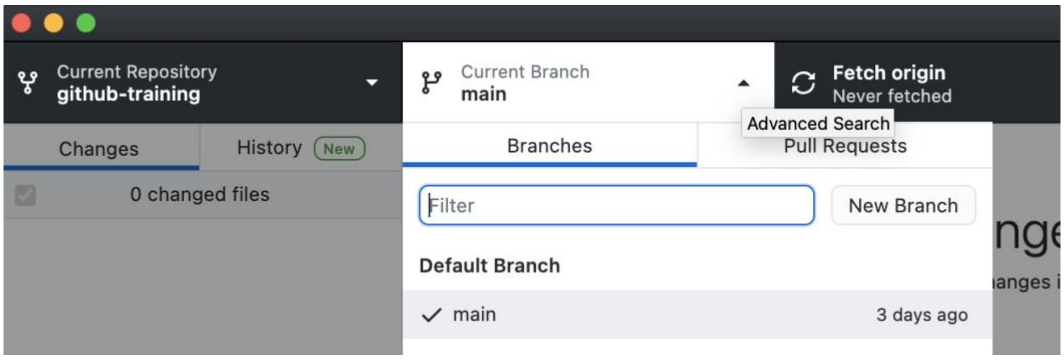
   - This might give you a little pop-up telling you that you have changes on the branch (even if you don't have changes on the branch), and prompting you to select what you want to do with them.
   - Select "to contribute to the parent project", unless you are not a member of the EHR group, in which case choose use for your own purposes.
   - Git is here asking if you want your default pull requests to be opened in the EHR group version of the repo, or your own.

2C. Create a New Branch

Because we are multiple people collaborating on this project, we can't all work on the main branch, so to do any work we need to checkout a new branch.

1. In the top bar of Github Desktop, click "current branch" to bring up the branch menu

2. Select New Branch



3. Type in a name for your new branch

- Each project can only have one branch with the same name
- Because lots of us are working on this project, I suggest you create a branch with your name in it, ie, “anna-code-review” or similar, whereas normally you would call it something relating to your feature i.e, “fix_error” or “code_review”

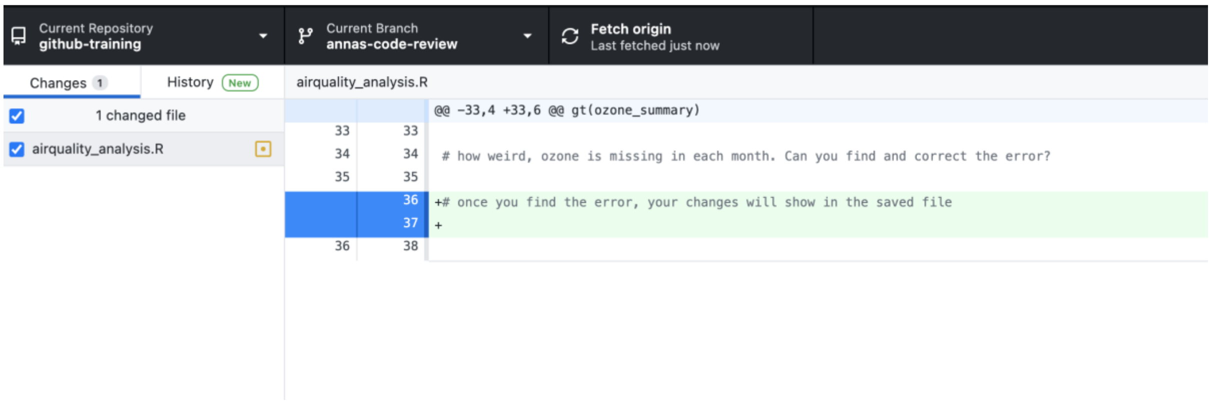
4. Click “create branch”. This should update the name on your “current branch” at the top in Github Desktop, and also give you the option to publish your branch if you like.

5. Click “publish branch”. This should make the branch appear on your own github profile.

2D. Find the coding error

Now, there are two analytical scripts here, a do file and an R script. Open whichever you are most comfortable with and run the code. They use inbuilt datasets for Stata and R and I’m hoping they will just run as intended, but let me know if they do not. There is some kind of error or mistake in both of the files. Find the error, correct it, and save the R and/or do file.

Once you’ve done this this should show up in github desktop as a change.



```
@@ -33,4 +33,6 @@ gt(ozone_summary)

 # how weird, ozone is missing in each month. Can you find and correct the error?

+# once you find the error, your changes will show in the saved file
+
```

(if you can’t find the error, ask Anna)

2E. Commit the change, push and open a pull request

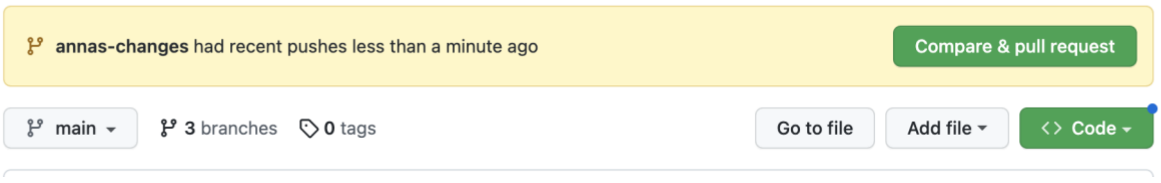
1. Commit your changes to upload them to Github using Github Desktop

- Check that the file you want to upload is “checked” in the left hand panel. This technically means it is staged and ready to be committed.
- Add an informative commit message (git will suggest one for you, but you can edit it)
- Check that you are commiting to the branch you want

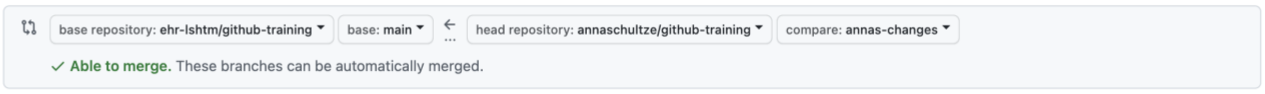
2. Select “push origin” to send these changes up to your own Github profile

3. Now, you want to flag to someone that you’ve done these changes and that you want to add them to the project. This is done through pull requests. I always open pull requests through Github online, so we’re going to ignore that Github Desktop asks you about pull requests and navigate to your online repository.

4. There, you should have a yellow box stating that you have a branch with recent pushes. Click “compare and pull request”

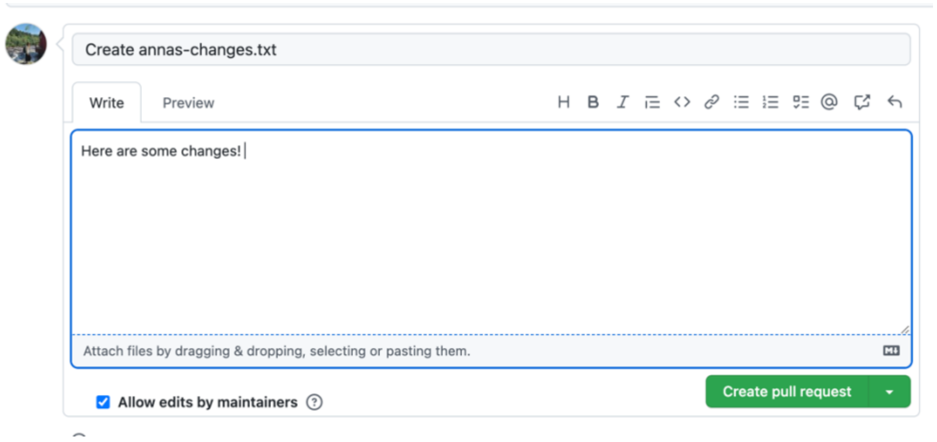

annas-changes had recent pushes less than a minute ago

5. This takes you to the option to open pull requests. The branch you’re merging into is called the “base”, and the one with your changes is shown on the left and is called the “compare”


base repository: ehr-lshtm/github-training
base: main
head repository: annaschultze/github-training
compare: annas-changes
✓ Able to merge. These branches can be automatically merged.

- You can change the base repository to be your own, or the EHR group if you’re a member.
- It’s important to check the repositories and branches to make sure these are correct
- If you are a member of the EHR group, you can select this as the base. If you’re not you’ll likely only see your own main as the base option. The reason this matters is just because to assign a reviewer, the person needs edit permissions for your repository. By default all people in the same organisation have permissions to do code review.

6. Select a title for your pull request, add an informative description and click “create pull request”


Create annas-changes.txt

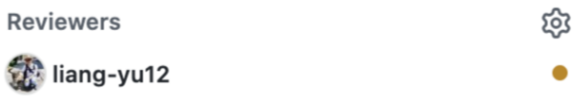
Here are some changes!

Write
Preview
Attach files by dragging & dropping, selecting or pasting them.
Allow edits by maintainers
Create pull request

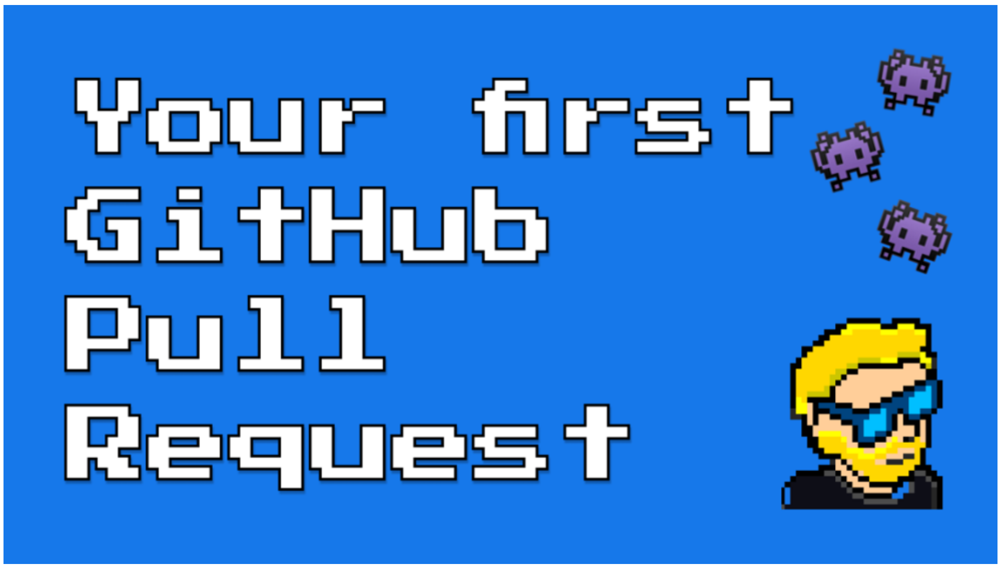
- The description should ideally contain some instructions for your reviewer

- For example, it can contain some information about parts of a program you are unsure about, and what files to focus on.

7. Select a reviewer in the right hand corner by clicking the settings wheel and typing the name of your reviewer. Typically, this is someone on your project – but today, ask the person next to you what their github name is, and assign them as a reviewer.



Congratulations, you have opened your first PR!



If this was a real project, you would merge the PR after review, but we are only taking it to this step today.

After your PR has been merged, it's important to remember to update your local main. Without any further changes, Github Desktop still thinks you're on your existing branch. If you wanted to add some more code through a new feature, it would be important for you to switch back to the main branch, and then pull your changes down (in Github desktop, fetch origin, followed by pull). **This is essentially syncing your local repository, and is a very easy step to forget.**

2F. If time, complete code review for one of your colleagues

Hopefully, one of your colleagues has assigned you to review their code. Complete the code review for them by clicking on "files changed" and "review changes". Please do not merge in the changes, as it'll make for an unhappy main branch on the EHR github.