Update GitHub Bio Instructions

Jacob Jeffries & Liam Myhill

March 25, 2024

1 Introduction

This set of instructions will guide members of MUEXLY who may or may not be new users of git through the process of updating their bio information within the organization web site, which can be found at *https://muexly.github.io/*. While knowing git and its capabilities is incredibly useful for development of computational tools, these instructions aim to simplify the process for individuals who are not familiar with utilizing a command line to interface with their computers. For this reason, the following instructions are divided into two sections, the first intended to be the simplest method for updating your organization bio, and the second for users who are comfortable or familiar with git commands.

2 Updating Bio through GitHub.com

2.1 GitHub Account Creation & MUEXLY Membership

If the user has not done so already, upon joining MUEXLY it is recommended that one creates an account at GitHub.com, which can be linked to your @clemson.edu email. Once the account has been created, forward your account username to an appropriate party who can invite you to the MUEXLY organization When

you have accepted the invite, and naviagte to the organization screen, you should be greeted by the landing page presented in Figure 1 (a). Here, you can see the many repositories that members of MUEXLY have shared with the organization. People interested in learning how to program should frequent the repositories of other members to see various algorithm implementations and learn different code styles.

2.2 Navigating the MUEXLY Organization Page

Navigate to the **MUEXLY.github.io** repository, as highlighted in Figure 1 (b). This repository specifically contains all of the code and supplemental materials found on the organization's webpage. A Hugo site is deployed at the url provided in the introduction such that the information contained in this repository *(Figure 1 (c))* is displayed as it appears on the organization home page and tab pages. To specifically access where the files containing bio information are located, navigate to the **content** directory as highlighted in Figure 1 (d).

Once accessing the **content** directory *(Figure 2 (a)*, select the **team** directory *(Figure 2 (b)* to see where all the information for team members is stored separately. If a directory has not been made for you already, you can either request one be made for you, or refer to a brief section in the documentation included in Section 3.2. Doing this will require you to follow all previous steps outlined in Section 3. Assuming a member folder exists, navigate into it to find a file titled *index.md*. This is your primary markdown file which populates the information displayed on the group website via the Hugo deployment. Modifying the contents of this file will update the text displayed on the group website, and any modifications made should be well thought out and intentional. Suggestions for how to modify your organization biographies are provided in Section 2.3.
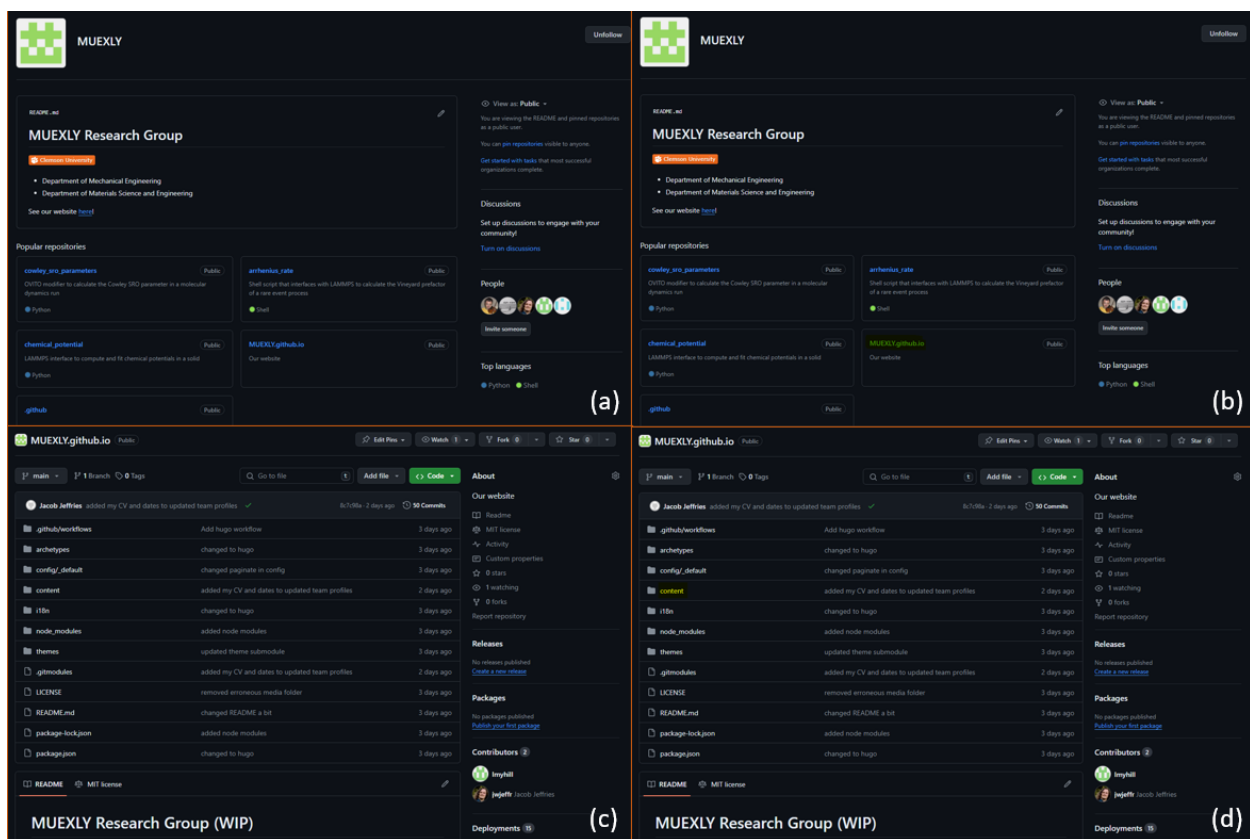
Figure 1: Images of *(a)* the landing page displaying all member repositories, *(b)* a highlighted guide directing the user to the **MUEXLY.github.io** repository, *(c)* the subdirectories of the **MUEXLY.github.io** repository, *(d)* a highlighted guide directing the user to the **content** subdirectory
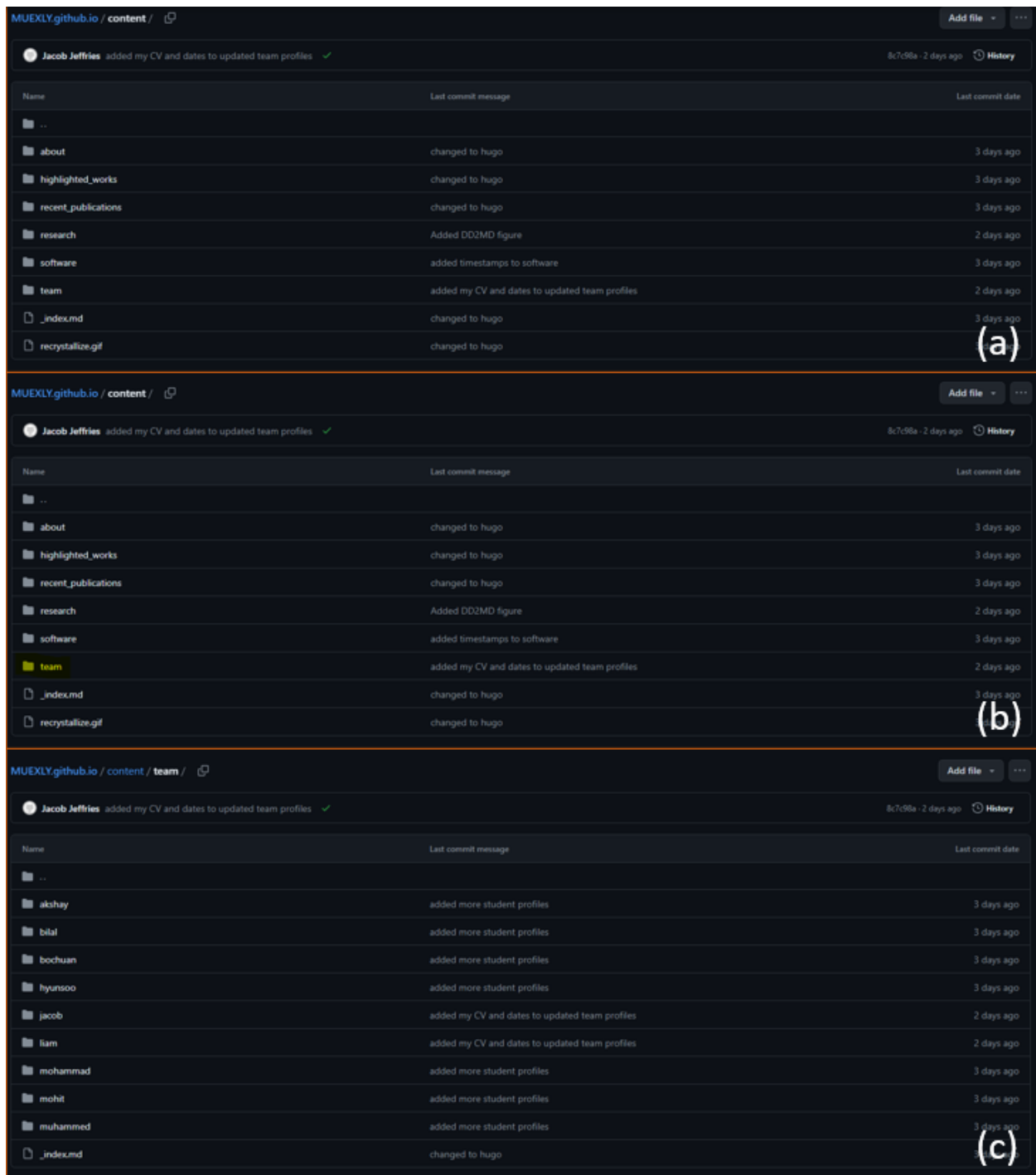
Figure 2: Images of *(a)* the subdirectories of the **contents** directory, *(b)* a highlighted guide to the **team** subdirectory, *(c)* an image of the team subdirectories

2.3   Modifying User Markdown File

Once you have a directory created for your organization bio, you can either modify or create an **index.md** markdown file to modify the contents of your biography page. An example of one such file is provided in Figure 3, which also highlights the ability to modify the markdown script directly on the GitHub website. If the **index.md** file does not already exist, you can add a file into your directory using the file-tree shown in Figure 4. Hugo only recognizes files named **index.md** by default, so it is recommended the user maintain this naming convention. Adding supplemental materials can also be added into your user directory, and linked to your organization bio. To link added files into your markdown file see the example in Figure 3. Also, note that graduate student profiles are listed on the website according to when they've been updated. This was designed to encourage regularly updating your bio. To specify the date, change the **date** flag in the header in **index.md** to the current datetime in ISO-8601 format, as seen in Figure 3. To see the current datetime in ISO-8601 format, you can navigate to `https://www.utctime.net/`.

There are many other examples of ways you can improve your bio, and feel free to make it as simple or complex as you desire. Your organization web page is a great place to market yourself, so feel free to add your previous work or perhaps an updated resume to show off your capabilities. Formatting markdown files is relatively simple, but the end result is a well formatted summary of your work out there for the world to see! For basic markdown syntax, the author recommends following the guidelines provided at `https://www.markdownguide.org/cheat-sheet/`.

3 Updating Bio through Git Terminal Commands

3.1 Cloning the GitHub Repository

It is expected that individuals who are more comfortable modifying their organization bio via the command line and git will be more concerned with pertinent filepaths and repository structure. This information can be gleaned from the tutorial provided in Section 2; however, to streamline the process, all necessary commands are provided here as documentation.

To clone the GitHub repository containing the website development workspace:

```
$ git clone https://github.com/MUEXLY/MUEXLY.github.io.git
```

To access the team directory:

```
$ cd /MUEXLY.github.io/content/team/
```

3.2 Creating a Member Directory

If a directory does not already exist for your information, one can be created within the team directory using the following command

```
$ mkdir ${YOUR_NAME_HERE}$
```

An **index.md** script can be initialized within this directory using a text editor of choice (assumed vim).

```
$ vim /${YOUR_NAME_HERE}$/index.md
```

All files you want to store in your organization folder for linking to the **index.md** can be copied into the member directory

```
$ cp ${WHATEVER_YOU_WANT}$ /root_path/MUEXLY.github.io/content/team/${YOUR_NAME_HERE}$
```

3.3 Committing Changes

If you are given permission to commit changes by an organization owner or moderator, you can update any code repository using the following commands. An overview of committing git changes is provided for those unfamiliar with the process. Executing any of the following lines will require git authentication, so provide the username and password to your GitHub account when prompted.

To add files to a commit queue,

```
$ git add *
```

To commit changes to a revision that is ready for a push (records changes locally),
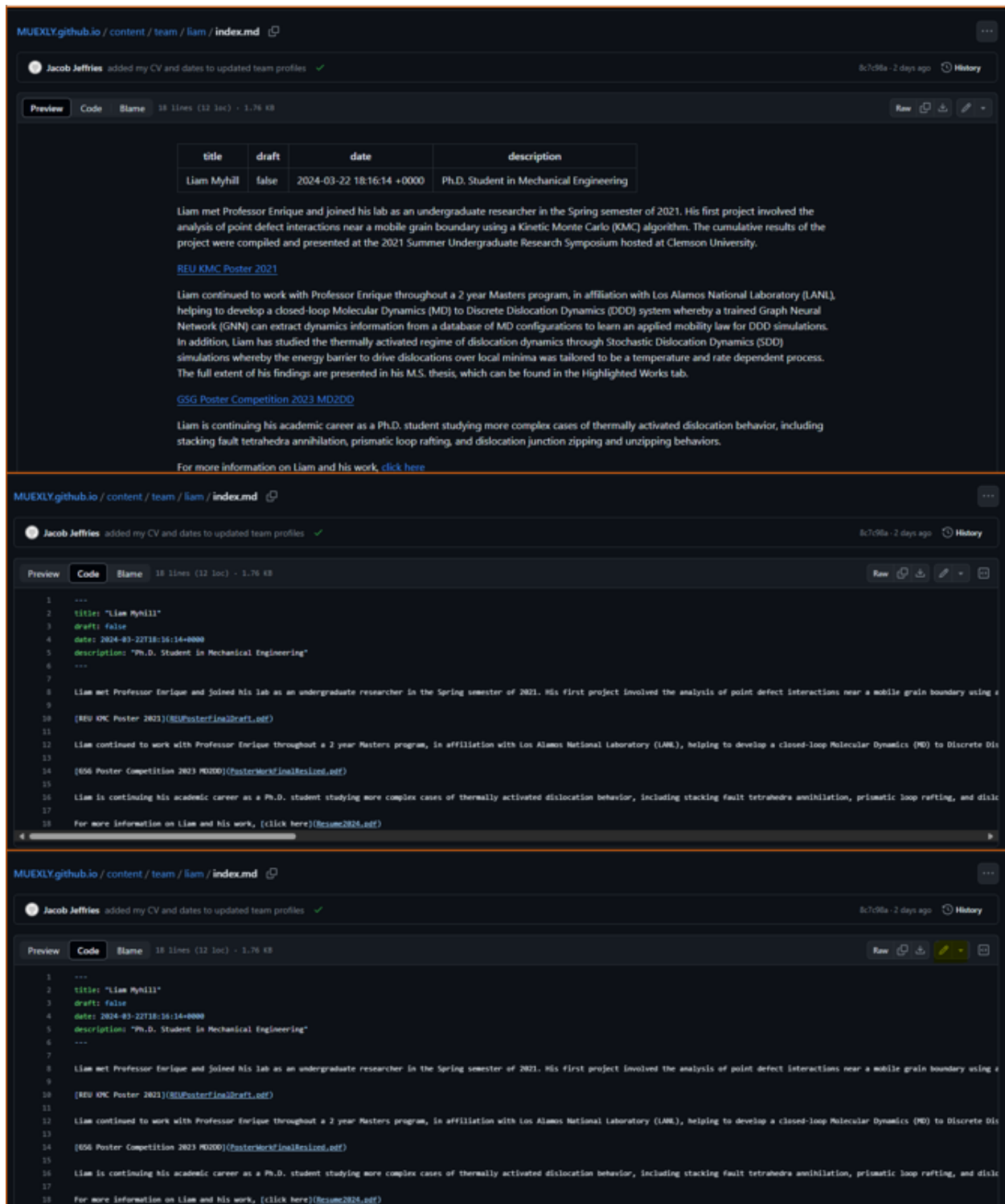
```
$ git commit -a *
```

Figure 3: Images of *(top)* An example preview of a markdown file, *(middle)* The source code which generates the text and formatting, *(bottom)* how to edit the source code via the GitHub webpage IDE
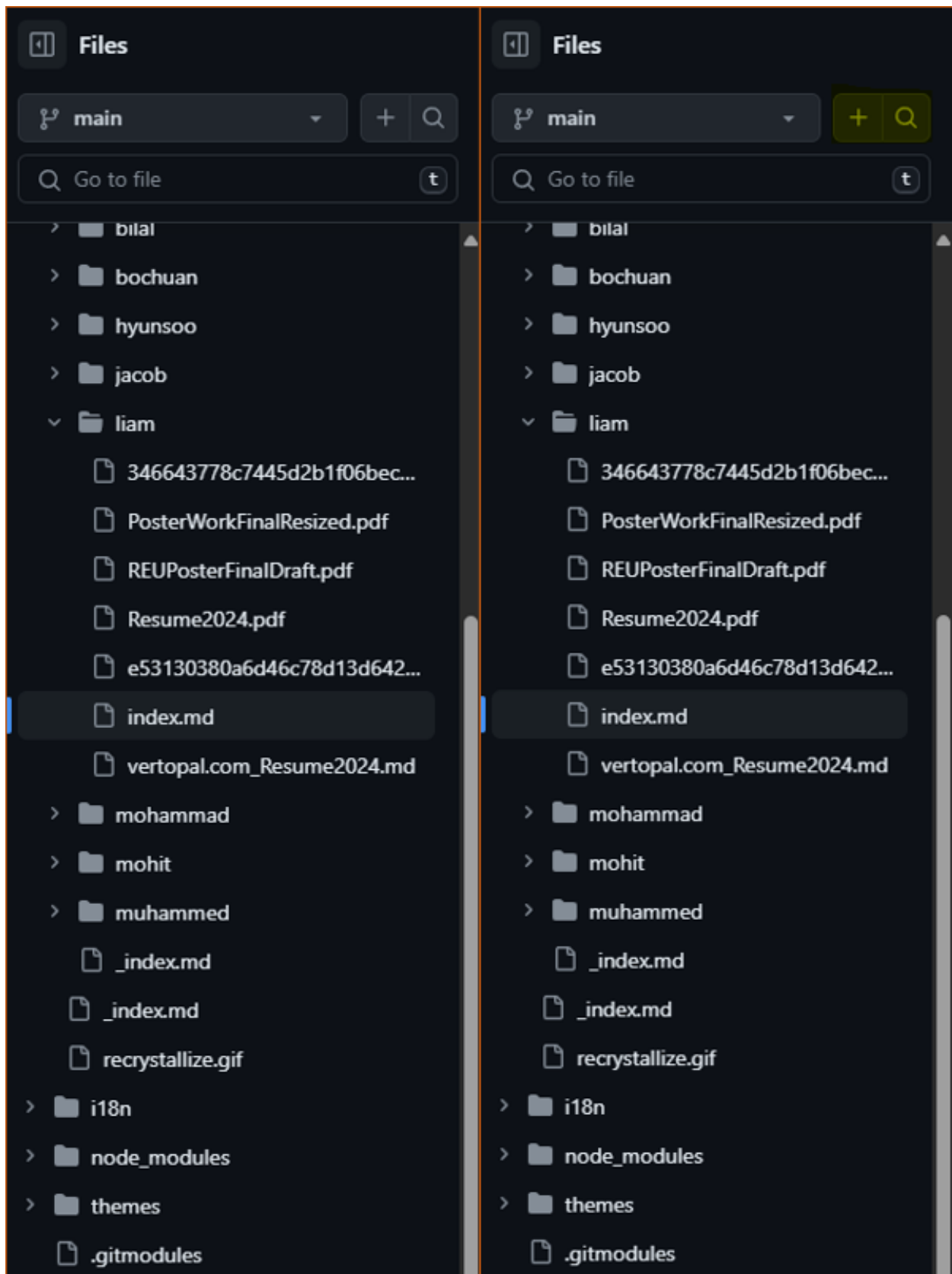
Figure 4: Images of *(left)* The file-tree as visualized using the GitHub website interface, *(right)* Highlighted options to add new files or search for existing files in the file-tree

Upon committing, you can choose to note what changes were made and why. Good record keeping can help with future debugging, so be kind to your future self.

To push changes to the repository,

```
$ git push *
```

Be sure that you commit prior to a push, otherwise the changes you make will not be reflected in the online repository. For an overview of the git work-flow, see Figure 5, which comes from `https://stackoverflow.com/questions/6143285/git-add-vs-push-vs-commit`.

Figure 5: Graphic depiction of the git workflow from clone to push

the curves indicate that the command on the right is usually executed after the command on the left. This gives an idea of the flow of commands someone usually does with Git.

CREATE
init
clone

BROWSE
status
log
show
diff
branch

CHANGE

REVERT
reset
checkout
revert

UPDATE
pull
fetch
merge
am

BRANCH
checkout
branch

COMMIT
commit

PUBLISH
push
format-patch