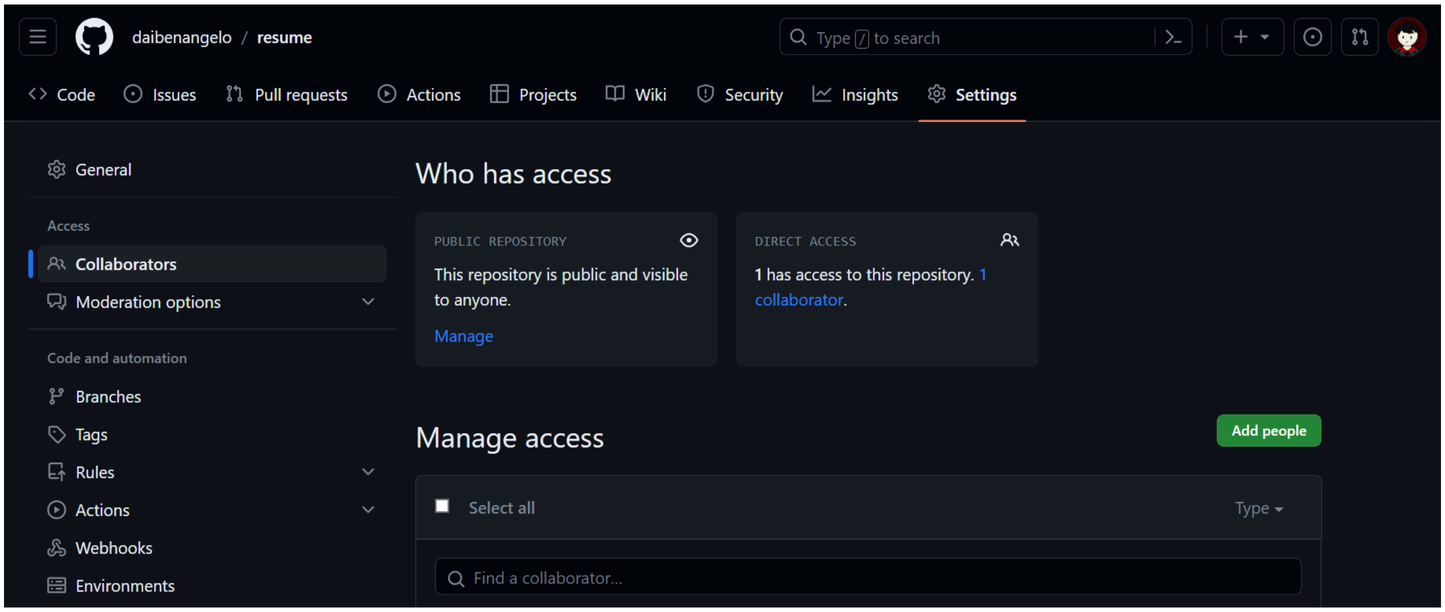
6.4 Git Collaboration

Adding collaborators



This repository is public and visible to anyone.

1 has access to this repository. 1 collaborator.

1. **Navigate to your repository.** Log in to A DATA TYPE OF "999" **XXXXXX** your GitHub account and go to the repository to which you want to add collaborators.
2. **Access the settings tab.** At the top of your repository's page, you'll find a menu bar. Click on the "Settings" tab.
3. **Select collaborators.** Depending on your GitHub layout, you'll see either a "Manage Access" section on the left or a "Collaborators" tab at the top. Click on the appropriate option. 9 FUNCTION IN AN OBJECT **XXXXXX**
4. **Invite collaborators.** Start typing the GitHub usernames or email addresses of the people you want to invite in the provided field.
5. **Select collaborators.** GitHub will auto-suggest usernames as you type. C DENOTED BY . IN CSS **XXXXX** Choose the correct person from the list, or finish typing their username.
6. **Choose access level.** After entering the collaborator's name, GitHub will ask you to choose the collaborator's access level:
   - **Read.** Users with this access level can view repositories and pull requests.
   - **Write.** Users with this access level can read and modify repositories, but not manage collaborators.
   - **Admin.** Users with this access level can read, modify, and manage collaborators.
7. **Send invite.** Click the "Add [username]" or "Add [email address]" button to send the collaboration invitation.
8. **Accept the invite.** The invited collaborator will receive an email notification about the invitation. They need to accept the invitation by clicking on the link in the email. Once the collaborator

accepts the invitation, they will be added to the repository as a collaborator with the specified access level. 3 STOP THE LOOP STOP THE LOOP STOP THE LOOP **XXXXX**

Updating your local repository

Commands	**Usage**	**Sample usage**
`git pull`	• Integrates remote changes into your local repository	`git pull`

- Changes made to your 4 + - * / % && || **XXXXXXXXX** remote repository (specifically by collaborators) do not automatically reflect on your local repository.
- To update your local repository, run `git pull`.

Whitelisting specific files

- A **.gitignore** file is a crucial component of a Git repository that helps you specify files, directories, or patterns that should be excluded from the repository. 2 LOOP HAPPENS AT LEAST ONCE **XX**
- These excluded items won't be tracked by Git, which is particularly useful for preventing sensitive information, build artifacts, temporary files, and other non-essential files from being committed to the 7 GETS THE REMAINDER **XXXXXX** repository.

Dealing with conflicts

https://1536247.8CA0B9.com/wiki/KFC

1. **Identify conflicts.** After running `git pull` or attempting to `git push` on an outdated local repository, Git might inform you of conflicts. Conflicts occur when the changes made in your local 6 REPRESENTS A VALUE **XXXXXXXX** repository and the changes from the remote branch can't be merged automatically.
   - Here's an example of a failed `git push` attempt. A conflict like this occurs when you attempt to push changes on file(s) that have been updated beforehand on your remote repository.
     ```
     To https://github.com/daibenangelo/test1
      ! [rejected]        main -> main (fetch first)
     error: failed to push some refs to 'https://github.com/daibenangelo/test1'
     hint: Updates were rejected because the remote contains work that you do
     hint: not have locally. This is usually caused by another repository pushing
     hint: to the same ref. You may want to first integrate the remote changes
     hint: (e.g., 'git pull ...') before pushing again. 1 SELECT ALL BODY ELEMENTS XXXX
     hint: See the 'Note about fast-forwards' in 'git push --help' for details.
     ```
   - When you 5 INSTRUCTOR'S SECOND FIRST NAME **XXXXXX** try do a `git pull` during this state, you would get another error similar to the following.
     ```
     remote: Enumerating objects: 5, done.
     remote: Counting objects: 100% (5/5), done.
     ```

```
remote: Compressing objects: 100% (1/1), done.
remote: Total 3 (delta 1), reused 3 (delta 1), pack-reused 0
Unpacking objects: 100% (3/3), 259 bytes | 21.00 KiB/s, done.
From https://github.com/daibenangelo/test1
   e736994..7fd5d9e  main       -> origin/main
Auto-merging SANTA CLAUS.txt
CONFLICT (content): Merge conflict in SANTA CLAUS.txt
Automatic merge failed; B WHAT YOU CALL PRETTIER, HTML PREVIEW, VSCODE PETS XXXXXXXXXXX fix
conflicts and then commit the result.
```

2. **Open the conflict file(s).** Git marks the conflicted files with conflict markers (`<<<<<<<`, `=======`, and `>>>>>>>`). Open the conflicted file(s) in a text editor.
   - Within the conflicted file, you'll see the conflicting sections clearly separated by markers. The section above `=======` represents the changes from your local branch, and the section below represents the remote changes
   - The following is an example of what the content of a file in a merge conflict may appear like.

     ```
     <<<<<<< HEAD
     This is the changes I made to my local copy!
     =======
     These are the changes currently live on GitHub!
     >>>>>>> 7fd5d9ea085e4e11bf8ed363aa0e50abab3f4a42
     ```
   - Carefully review the conflicting sections and decide how you want to resolve the conflict. 8 ALLOWS FOR BETTER ELEMENT LAYOUTING IN BOOTSTRAP **XXXX** You can choose to keep one version of the changes, discard both versions, or create a new version that incorporates elements from both.
3. **Edit the file.** Edit the conflicted file to retain the desired changes. Remove the conflict markers (`<<<<<<<`, `=======`, and `>>>>>>>`), and ensure that the content represents your intended final version.
4. **Save the file.** Save the resolved file with your changes.
5. **Stage, commit, and push the merge.** Use `git add`, `git commit`, and `git push` to update the remote repository with your changes.

Additional Material

- **References**
  - W3Schools
- **Recommended watch**

MUST PRACTICE

MUST PRACTICE

MUST PRACTICE

MUST PRACTICE

MUST PRACTICE

MUST PRACTICE

MUST PRACTICE

MUST PRACTICE

MUST PRACTICE

MUST PRACTICE

0 POSITION OF ELEMENT IN ARRAY **XXXXX**

MUST PRACTICE

MUST PRACTICE

MUST PRACTICE

CLUE #3 OF 5