Assignment 4: Working Together

Overview

Last week, you created a Git repository and practiced committing changes and pushing them up to GitHub. Time to spice things up a bit! For this assignment, you and a partner will be working together and making changes to the same remote GitHub repository. As this course goes on, you will learn better practices for working together with others, but to start you will practice with a simple taking turns strategy.

Assignment Instructions

You ***must*** work with an assigned partner for this assignment. Only one group member should to submit the assignment to Blackboard, and should add who they worked with in the comment box of the submission.

Your group is tasked with creating an application that imitates a vending machine. Both members of your group will need to contribute to this application. One group member will create the remote repo on GitHub, and both group members must have a copy of the repo from GitHub on their own computers in order to commit and push changes. No group member is allowed to commit twice in a row. This means that when looking at the commit history for your group's GitHub repo, I should not see the same person committing sequentially.

Below are the features the vending machine must have.

- A large variety of snacks are available for purchase.
- Each snack has a price associated with it.
- The user is welcomed and given usage instructions at the beginning of the transaction.
- The user can pick which snack they want to buy.
- The user can insert as much money as they would like to the machine, and only $1 bills are accepted.
- Exact change must be given back to the user if they inserted too much money into the machine after making their purchase.
- The user is asked for confirmation on the transaction before it is carried out, and if the user decides to cancel the transaction, they are given a full refund.
- The user is given a departure message that makes them feel good about their snack choice.

You may carry out these features in the order that you see fit, and features must be divvyed up amongst both group members in a logical manner. You need to split up the features of this app into at least four pieces, meaning each group member will have to make at least two commits. Think about the requirements of this application in terms of "tasks" – which tasks are best to start on, which tasks should be done at the same time, which tasks can be saved until later. I highly recommend that each group writes down their task breakdown plan before starting to work on the application.

As mentioned before, after one group member creates a commit (finishes a task) and pushes their changes to GitHub, it will then be the other group member's turn to do the next task. In order for a

group member to pull down changes from GitHub (remote repo) that they do not yet have on their computer (local repo), they must run a **git pull** command. Before a group member starts their "turn", they will need to run a **git pull** to sync up their local repo with the remote repo, otherwise they will lose out on their group member's recent changes. You can read about pulling down each other's changes in the **Pull_Remote_Changes.pdf** document.

I recommend the following workflow cycle:

1. First group member works on a feature
2. First group member commits and pushes their changes
3. Second group member pulls down the new changes from GitHub (remote repo) down to their own computer's Git repo (local repo).
4. Second group member works on a feature
5. Second group member commits and pushes their changes
6. First group member pulls down the new changes from GitHub (remote repo) down to their own computer's Git repo (local repo).
7. Go back to Step 1.

In the future you will learn how to work in an asynchronous fashion where multiple developers can work on the same project simultaneously. For now, this assignment is focusing on just getting used to working on the same repo with others. Baby steps!

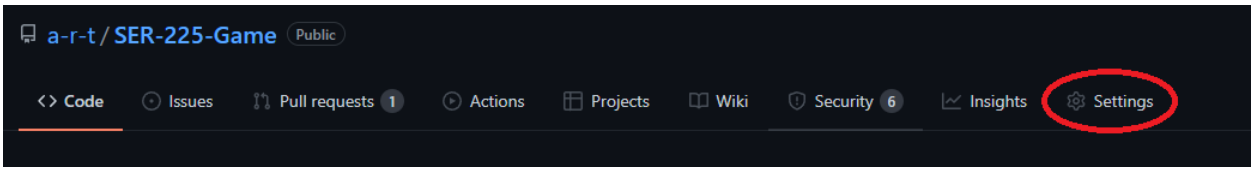
Repo Setup

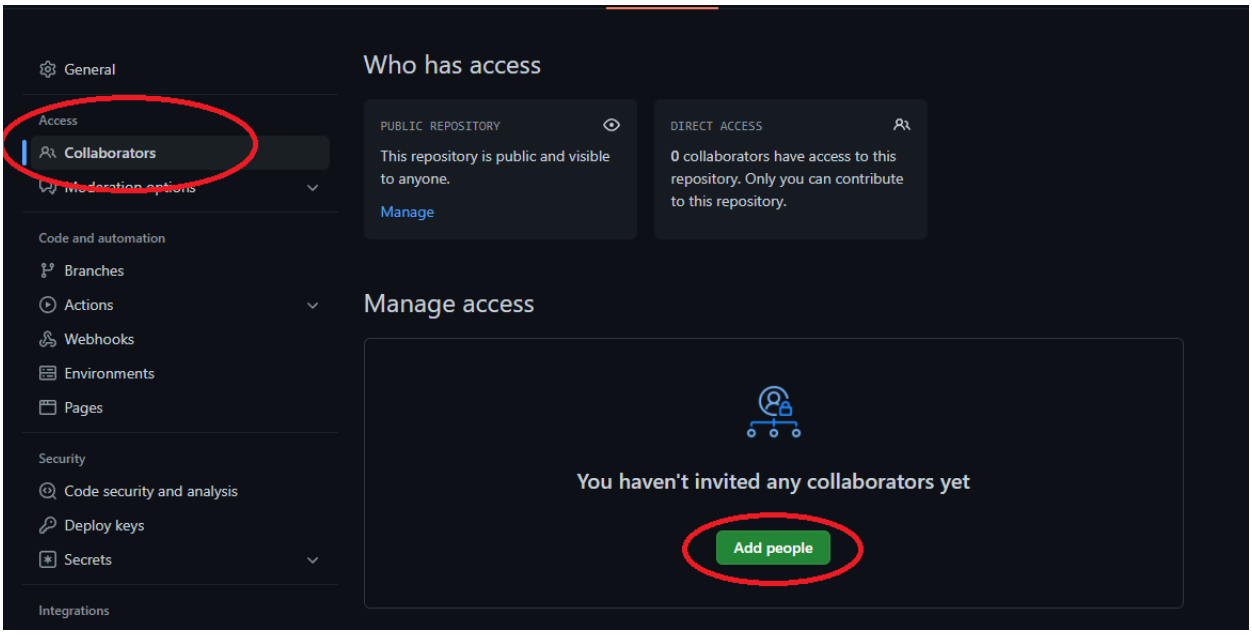
Only one group member needs to create and setup the repository in their GitHub account. Since both group members will be working from the same GitHub repo, only one GitHub repo needs to be created. It does not matter which group member elects to create the repo. When creating the repo, the group member should also include a basic "initial commit" to get things started.

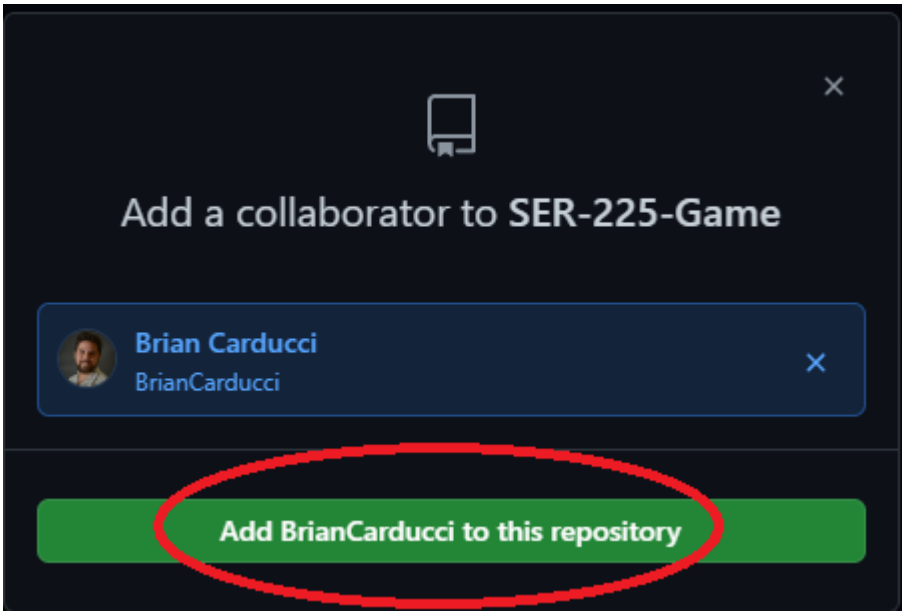
After the repo has been created, the other group member should clone their repo. Cloning a repo means copying the repo from GitHub on to a different computer. Git will handle doing this for you through the use of the **git clone** command. While you cloned a repo in the past for assignment 1, that was a while ago, so feel free to follow the **Cloning_a_GitHub_Repo.pdf** document for a refresher.

While anyone can clone a public repo on GitHub, special permissions must be given to a GitHub user in order to allow them to push changes to a repo that is not theirs. In order for other group members outside of the repo owner to push changes, they must be explicitly granted access.

The repo owner can give access to their other group member by going into the **settings** of their repo, going to the **Collaborators** tab, clicking the **Add people** button, and entering the GitHub username of the group member they are working with. After hitting the **Add to this repository** button, that group member will be sent an email telling them they are invited to be a collaborator, and upon accepting, will be authorized to push code to that repo from that point on. This only applies for this one GitHub repo, and will not be applied to any other repos. Images of this process are below:


a-r-t / SER-225-Game
Public
Code
Issues
Pull requests 1
Actions
Projects
Wiki
Security 6
Insights
Settings


General
Access
Collaborators
Code and automation
Branches
Actions
Webhooks
Environments
Pages
Security
Code security and analysis
Deploy keys
Secrets
Integrations
Who has access
PUBLIC REPOSITORY
This repository is public and visible to anyone.
Manage
DIRECT ACCESS
0 collaborators have access to this repository. Only you can contribute to this repository.
Manage access
You haven't invited any collaborators yet
Add people


Add a collaborator to SER-225-Game
Brian Carducci
BrianCarducci
Add BrianCarducci to this repository

Submission

Your final submission to Blackboard should include:

- A link to the group's GitHub repository

Grading Rubric

In order to get an A on this assignment (20/20):

- Successfully created a Git repository on GitHub that both group members can access and push changes to: **5pts**
- Each group member contributed commits to the repository (work should be split up evenly), and neither group member committed more than one time in a row: **10pts**
- Project on GitHub compiles and runs successfully and includes all the required features: **5pts**

Note: I'm not going to be analyzing your vending machine program and intentionally checking for errors, but I want it to reasonably work and I do not want to see straight spaghetti code.