Merge Conflict

***If you do not have a GitHub account it is time to make one, it is easy and free. You also will need to have an IDE installed on your machine. (We recommend VS Code). For this exercise use any code you have previously created at Qwasar and make a few changes to it to complete the exercise. Instructions to set up GitHub locally are at the bottom.***

You Cannot use the GitHub App - All actions must be done in the terminal

—-------------------------------------------------------------------------------------------------

Objective: To learn and practice basic Git commands and collaborative workflows.

Prerequisites:

1. Have a GitHub account
2. Set up a shared repository
3. The files in your repo can be any exercise you’ve done in Docode

Step 1: Clone the Repo

1. One student creates the repository and shares the repository URL with the other two students.
2. Each student clones the repository to their local machine

Step 2: Branching and Committing Changes

1. Each student creates a new branch with their name as the branch name.
2. Each student makes changes to a specific file (e.g., adding a new function, modifying existing code, etc.) within their branch.
3. After making the changes, each student adds their changes to the staging area.
4. Each student commits their changes with a descriptive commit message

Step 3: Merging Changes

1. One student initiates the merge process by merging their branch into the main branch.
2. The other two students pull the changes made by the first student into their local repositories.

Step 4: Resolving Conflicts

1. Introduce conflicts intentionally by modifying the same lines of code in different branches.
2. One student attempts to merge their branch into the main branch, resulting in a conflict.
3. Each student resolves the conflicts by editing the conflicting files manually, removing the conflict markers (<<<<<<<, =======, >>>>>>>), and selecting the desired changes.
4. After resolving the conflicts, each student adds the modified files to the staging area and commits the changes.

Step 5: Pushing Changes

1. Each student pushes their local changes to the remote repository.

Step 6: Reviewing Changes

1. Each student reviews the changes made by the other two students in the shared repository.

Step 7: Pull Requests

1. One student creates a pull request to merge their branch into the main branch.
2. The other two students review the pull request, leave comments if needed, and approve the merge.
3. Merge the pull request to incorporate the changes into the main branch.

—------------------------------------------------------------------------------------------------------------------

**How to Setup GitHub Locally Using an SSH Key**

1. Generate an SSH Key Pair:

* Open a terminal or command prompt on your local machine.
* Use the following command to generate a new SSH key pair:

ssh-keygen -t rsa -b 4096 -C "[babythanos@snapkingdom.com](mailto:babythanos@snapkingdom.com)"

Replace "your\_email@example.com" with your actual email address associated with your GitHub account.

* Press Enter to accept the default file location.
* Two files will be generated: a public key file (usually id\_rsa.pub) and a private key file (usually id\_rsa).

1. Add the Public Key to Your GitHub Account:

* Log into GitHub
* Click on your profile picture in the top-right corner, then select "Settings" from the dropdown menu.
* In the left sidebar, click on "SSH and GPG keys."
* Click on the "New SSH key" button.
* Give your SSH key a title (e.g., "BabyThanos SSH Key").
* In the terminal or command prompt, use the following command to display your public key:

cat ~/.ssh/id\_rsa.pub (*remember the cat command from EX00?)*

* Copy the entire key that is displayed.
* Paste the copied key into the "Key" field on the GitHub website.
* Click the "Add SSH key" button to save your key.

1. Test the SSH Connection:

* In the terminal or command prompt, use the following command to test the SSH connection to GitHub:

ssh -T git@github.com

* You may see a warning message about authenticity; type "yes" to continue.
* If the connection is successful, you will see a message like "Hi username! You've successfully authenticated, but GitHub does not provide shell access."

1. Configure Git to Use SSH:

* In the terminal or command prompt, configure Git to use SSH instead of HTTPS for GitHub connections:

git config --global url."git@github.com:".insteadOf "<https://github.com/>"

1. Set Up a New Repository:

* Create a new repository on the GitHub website or navigate to an existing repository.
* Click on the "Code" button and select "SSH" to get the SSH URL of the repository (git@github.com:babythanos/repository.git).

1. Clone the Repository:

* In the terminal or command prompt, navigate to the directory where you want to clone the repository.
* Use the following command to clone the repository using SSH:

git clone [git@github.com](mailto:git@github.com):username/repository.git

* Replace username/repository.git with the actual repository URL you obtained in the previous step.