Let’s “Git” Jiggy!

Game Set-Up:

- There should be 4-5 people in every group.
- Ensure that everyone’s gitbash/terminal is working fine.
- Ensure that everyone has their gitbash/terminal connected to their github
  - Note: If it’s not, that would be a great ‘challenge’ to figure out as a group (I have a helpful link in this ‘Git Refresher’ repo); but, please reach out and ask for help if needed!!

Instructions -

1. Choose 1 person to start the game. (This person is Person A.) Person A will then select Persons B-E as the game continues.
2. Person A - create a new repository in Github. The name of the repo can be called ‘Git Exercise’ - Add a README file to the repo then click ‘create repository’.
3. Before moving forward - Person A should add everyone in the group as a collaborator on the git repo they just created. Open Repository > Settings > Collaborators. Excellent!
4. Person A - open up the README file on github and add a sentence to the file telling us your weekend plans. Commit the changes in github.
5. Person A - share the link to your github repo. You can simply grab the link from the navigation bar at the top of your webpage.
6. Everyone else in the group - open up the link to the ‘Git Exercise’ repository Person A created.
7. Person B - fork the repository. What’s different about the repo you’re looking at now? How do we know this repo is the forked repo?
8. Person B - open up the README file of the forked repo (in github). Add in your plans for the weekend and the color of the pants you’re wearing. Commit those changes in github. What happened to Person A’s repo?
9. Person C - using the link Person A sent. Open the README file in github and add in your plans for the weekend and the last thing you watched on TV; then commit the changes. What do we notice? How was person B’s forked repo affected? How was Person A’s repo affected? What do we think that means?
10. Person D - using the link Person A sent, open up to the Git Exercise repo and copy the link you’re going to use to clone this repo to your local machine. (Hint: Use the green ‘Code’ button and copy the URL under the

HTTPS option.) Open up your terminal and clone the repo to a folder that is not already under git source control.

11. Person D - Once you have cloned the repo to your local machine - go into the folder (Hint: maybe run 'cd GitExercise' in the terminal) and to see what branch you're on. It should be the main or master branch (depending on your settings). You can run 'git status' to see that you're all up to date..
12. Person D - Open up the README file on your local machine. Skip a line or 2 and then add in the plans you have for the weekend as well as what you ate for dinner last night. Depending on what IDE you used to open the REAMDE, save the changes you made - if it doesn't auto-save.
13. Person D - navigate back to your terminal, make sure you are in the right directory. Type 'git status' to see the changes. Commit your changes with a concise message; Then push the changes to github.
14. What happened??? - Everything seems okay, right?
    a. In this case Person D is the only one that has this repo on their local machine. All of the other changes were made on github itself so no harm is done here. However making changes on the main/master branch is VERY DANGEROUS!! **Don't do dat!**
15. Person E - Just like Person D just did, clone this repo to your local machine.
16. Person E - Once you've cloned the repo go into the folder and type git status to see that you're all up to date.
17. Person E - create a new branch (Hint: There's a git cheat sheet in the Git Refresher repo. Go find it!!!!)
18. Person E - Check to see that your branch was created by running 'git branch'; type 'q' in the terminal if the terminal gets stuck after viewing the branches.
19. Person E - switch over to the branch you just created. (Hint: you can use 'git switch nameOfBranch'). Once you've switched over to the new branch, open up the README file and add in your plans for the weekend as well as the Golden Rule.
20. Person E- Stage the commits (hint: git add)
    Commit the changes(hint: git commit)
    Push your changes(git push)
21. Something happened??? Uh Oh…Well, this is a brand new branch so we have to set up the upstream to let Github know something on our local machine is coming to the remote world. The terminal should give you a command to copy and paste into the terminal. (hint: looks somethings like

this `git push --set-upstream origin <branch name>)` Copy the command you're given in the terminal, paste it into the terminal and run it. Once you do that, you should have successfully pushed to github once .

22. Person E - Go back to github . You should see that your branch is updated and you're ready to create a pull request. Create a pull request; you can add in whatever comment/description you want for the pull request.
23. Person A - review the changes and merge the changes Person E's branch to the code on the main branch.
24. Person E - in your terminal You should switch back to your local main branch. As a group, compare the difference between what's in the README on Person E's local main branch and the main branch on Github. How can Person E get their main branch and the remote main branch to look the same? (hint: think of it as "pulling" down the code from github into your local machine) Also, look at Person B's forked repo. What do you notice??

25. Persons A-D are welcomed to follow the steps that Person E did and practice. Take turns merging the pull requests . Remember only 1 person should be merging code at a time. Once code is merged to the main branch everyone in the group should be 'pulling' those changes to their machines. Branching and communication is literally all you need to prevent merge conflicts!
26. And Just like that… Games Over

27. We Love Being Extra! - Let's create a merge conflict and see if we can resolve it!!! No?? Okay ………