Codebase Rubric

Deliverables

- GitHub-hosted project in the class repository
- README describing how to run your project after initial cloning of the repository
- INDIVIDUAL submission of a writeup detailing what each member was responsible for and rate on a scale of 10 how well you feel each member contributed

Percentage Breakdown

- 20% Proper branch management(detailed below)
- 10% README gives proper details for running code
- 10% Writeup
- 10% Peer-rating
- 50% Code

Code

For your codebase, I am looking to see the progress made toward implementing the features the group has agreed upon. Whether a feature is fully implemented factors in both frontend and backend integration. While I do encourage being over-ambitious with the scope of the project, I also would prefer a few features implemented in full over a bunch of features being hard-coded to appear implemented. Whilst these may seem contradictory hopes, I see it as having many ideas to always have work but continue only once on idea has been seen to the end. By fully seeing ideas to the end, it results in more learning about the full-stack and better development in the future.

The Code section will be roughly graded as 12 points. 4 points are given for the full implementation of a feature. This means the feature works as intended and is fully functional.

NOTE: Everyone is required to contribute to the codebase. Since I can reliably assume that there will be seven weeks of tasks being required and everyone will have at least 1 regular task point a week, I expect to see a minimum of 7 pull requests made by each member of the group to mark their task completion. In addition, there will be a 5% penalty to the **Codebase** grade for **each** missing pull request below 7.

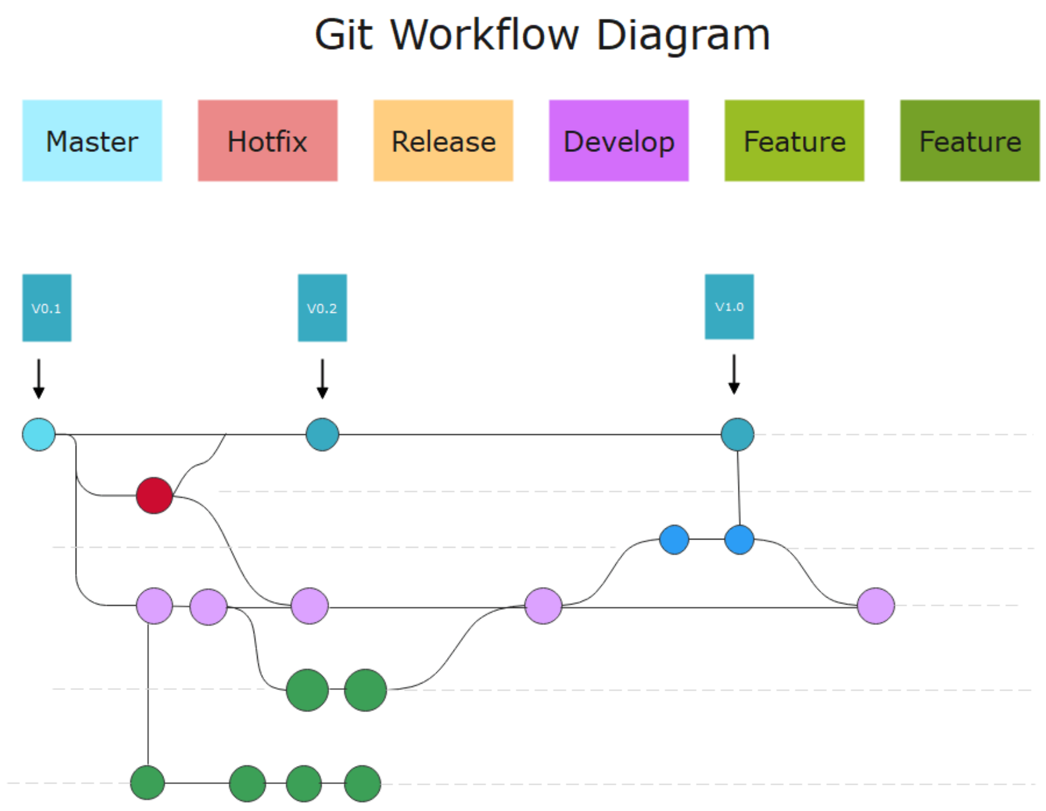
NOTE: All code must be your own. NO USING GENERATIVE AI. If AI is suspected to have been used to create the code in the project, the entire group will receive a 0 for the code section. The reason for this harshness is due to the overwhelming affects reckless use of AI has on codebases. Not only will the code only barely function in the context it is written, but it also has the major tendency of being hard to work with outside the original context of its creation, making overall development harder.

README

A README.md is a Markdown file commonplace to repositories that explains the repo. In addition, these also give detail to how someone new to the project can start up development after cloning the repository. This is REQUIRED as this is how the grader and myself will be able to look at the project on our own time.

NOTE: Detail the README in such a way that anyone can run the project with minimal knowledge. For example, list every extension used, list the versions of software downloaded, give detailed steps on how to run both backend and frontend, etc. The grader and I will follow these READMEs to the letter. Read the note at the end to remind yourself after reading the rest that **failure to do so will result in a grade of 0 for the code section**.

Branch Management



To help enforce/heavily encourage proper project management in Git, **branch management will be a major part of the project grade**. As mentioned in class, your main/master branch is where only the dev branch can merge its code. The main branch must always be able to run without errors. The dev branch is where new features are tested together to see if every new thing works. Branching off of dev, new features are made and tested. After testing, these feature branches are merged into the dev branch.

When working on the project, every task must have a branch off the dev branch. Once completed with the task, make a pull request to merge that branch into the dev branch. DO NOT delete these branches after resolving pull requests.

In addition, <u>**every**</u> commit made must have a meaningful message accompanying the commit. And, <u>**every**</u> pull request made must have a detailed message. What I mean by meaningful is that the message should succinctly explain what progress has been made in the code within the commit or pull request. Someone else should be able to have a solid grasp of what was changed from the message without needing to see the code. The following comic is a good example of good vs bad messages.

COMMENT
DATE
CREATED MAIN LOOP & TIMING CONTROL 14 HOURS AGO
ENABLED CONFIG FILE PARSING 9 HOURS AGO
MISC BUGFIXES 5 HOURS AGO
CODE ADDITIONS/EDITS 4 HOURS AGO
MORE CODE 4 HOURS AGO
HERE HAVE CODE 4 HOURS AGO
AAAAAAAA 3 HOURS AGO
ADKFJSLKDFJSDKLFJ 3 HOURS AGO
MY HANDS ARE TYPING WORDS 2 HOURS AGO
HAAAAAAAANDS 2 HOURS AGO
AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.

Writeup

This is where you can detail all your opinions about the project. In addition, include details of what everyone in the group was responsible for and what they did. These will be submitted individually since I want honest feedback about your group members and your thoughts on the project: the good, the bad, and the ugly.

Peer-Rating

As a part of the Writeup, for each member of your group, rate them on a scale of 0 – 10.

- 0 is no contribution
- 1 is contributed next to none of their responsibilities
- 5 is contributed the bare minimum
- 10 is contributed to all their responsibilities and played a role in ensuring others met their responsibilities.

IMPORTANT NOTES

If the project code does not work on the main branch, the CODE aspect of the project will be a 0.

If there is not a README that accurately details how to get the project running after a clone, both the CODE and README section will receive a 0.

All code must be your own. NO USING GENERATIVE AI. If AI is suspected to have been used to create the code in the project, the entire group will receive a 0 for the code section. The reason for this harshness is due to the overwhelming affects reckless use of AI has on codebases. Not only will the code only barely function in the context it is written, but it also has the major tendency of being hard to work with outside the original context of its creation, making overall development harder.

If a banned language is used, the CODE section will receive a 0. Regarding scripting, blueprint scripting in game engines counts as a purely scripting language and is, thus, banned.

If any retaliatory actions take place by the students, the offender will get a 0 on the project, and I will work with the rest of the group to recover against what happened.

Everyone is required to contribute to the codebase. Since I can reliably assume that there will be seven weeks of tasks being required and everyone will have at least 1 regular task point a week, I expect to see a minimum of 7 pull requests made by each member of the group to mark their task completion. In addition, there will be a 5% penalty to the CODEBASE grade for each pull request below 7.

Contacting Me

If you have any questions or need any help, don't hesitate to ask me for help. I am available outside of class on Discord and by email.

Email: jdenny3@lsu.edu
Discord: mr.jld

Links

Homework

Class GitHub: https://github.com/orgs/CSC-3380-Class-Code

Role resources

Jira: https://www.atlassian.com/software/jira

GitHub Projects: https://docs.github.com/en/issues/planning-and-tracking-with-projects/learning-about-projects/about-projects

When2meet: https://www.when2meet.com/

Amazon's Code Standards: https://docs.aws.amazon.com/wellarchitected/latest/devops-guidance/dl.cr.1-standardize-coding-practices.html

Google's Code Standards: https://google.github.io/styleguide/

Code Review Best Practices: : https://www.atlassian.com/blog/add-ons/code-review-best-practices

Figma: https://www.figma.com

Adobe Creative Cloud: https://www.adobe.com/creativecloud.html

LSU Resources

LSU Library Rentals: https://lib.lsu.edu/

LSU Access to Adobe Creative Cloud: https://grok.lsu.edu/Article.aspx?articleid=20276