CSC 3380 Project

What is the Project

The major project for this class is an opportunity to work in a group setting and learn something new. The goal for this project is to practice with technologies that may be new to you or to try and master a technology you already know. The scope of the project naturally must be at least the duration of the semester. However, in the scope meetings that I will require in the early weeks of the semester, groups are allowed to look at scopes that expand for two semesters. This intention and scope must be known by me, so I can grade accordingly.

Parts

The project will be graded in four parts listed below. Each makeup a portion of the grade and are all equally important to having a smooth development cycle.

1. Codebase 50%
2. Weekly standups 10%
    a. Lowest 2 will be dropped
3. Bi-weekly Role Report 15%
    a. Lowest will be dropped
4. Idea pitch presentation 5%
5. Final presentation / Live demo 20%

Limitations on Languages/Project

NOTE: Using a forbidden language will result in a 0 (ZERO) on the codebase portion of the project

1. Must support classes (i.e. no scripting languages)
2. Must use explicit typing (No JavaScript [Because TypeScript exists], Python only allowed with type hinting)
3. No purely AI projects

Important Dates

- Every Friday, there will be a standup that the Communication Lead will submit all group members' responses.
- September 17 and 19 will be the Idea Pitch Presentation
    - The weeks before, groups are required to schedule a meeting with me and bring a few ideas to discuss with me for scope.
- November 12 – November 26 will be for the Final Presentations / Live Demos
- December 6 or 10 will be final due date of project

Tasks

Every week, each member needs to complete at least 4 points worth of tasks. How these tasks are defined and how many points they are worth are detailed specifically in the Project Manager section because that is the role assigning tasks. Completion of these tasks will be documented in the weekly standups submitted by the Communications Lead. In addition, each week, members can earn bonus for up to an additional 2 points worth of tasks. Note, this can also be achieved by completing two 3-point tasks, for example, because this meets the 4 point task minimum and the 2 extra points can be used towards the bonus.

Also, the optional homework for this class will each have an assigned point value to them that can be used towards the number of task points needed for the week of the homework submission. These homeworks are design to be minor challenges of a skill or topic. As such, these will likely only be worth 1 or 2 task points.

Roles

Every group must have a member performing each role. Though there are designated roles, everyone in the group is still responsible for contributing to the codebase.

Recommendations

- If you do not posses a computer, it is possible to borrow one from the Middleton Library at LSU, at room 241. For policy and when you can borrow, check https://lib.lsu.edu/.
- When someone claims a task, it is theirs. The only way a task can change hands is if the person who claims it communicates to the group that they are relinquishing it. It might also be beneficial to establish a kind of “lease” on the tasks, say a week or two, that after said lease, if the task isn’t done yet, others may begin to claim the task simply by letting the group know their intention and claiming ownership of the task on the kanban board.
- Use this opportunity to learn something new. It can be terrifying to try and learn a new language or do something you have no idea if you can do. THIS IS OK. You have the semester to work on this project, and you would be surprised at how well you can pick up new technologies. Regarding programming languages, basically all of the have some way for functions, if-statements, loops, variables, numerics, strings, and custom data types. Recognizing this, it the becomes a matter of figuring out how to “translate” into this other language, and before you know it, you can speak it without needing to translate.
- Always commit whatever work you are doing at the end of the day / work session. Think of commits as saving your work. It isn’t until you submit a pull request that your task is being marked as “submitted”.
- Identify when in the week you can work on the project. The task system, while simulating the real world, is also meant to help keep work on the project going at a steady pace and keep a minimum amount of weekly progress.
- Account for debugging and research time. This is likely the first major group project you have done in your degree. Even if you plan that the time to write the code should only take about 3 hours, debugging and researching any errors / knowledge gaps can exponentially add lots of time.
- Communicate with your group members often. It can be all too easy to shy away from talking to others and staying “in your lane”. However, to get the most out of this project and grow the most, be upfront and communicate with your team as often as you can.
- A rule of thumb for asking for help: spend about 30 – 45 minutes trying to solve it yourself, then alert your group members to your issue and see if they can help. Everyone has different knowledge areas, and the most benefit of them comes from open and consistent communication.
- Take it one step at a time. One benefit of the task system is that is seeks to break down big milestones into manageable tasks that build up over time. It can be very daunting to start, but I can assure you that if you work consistently, grand things wait for you at the end.

Project Manager

This role is responsible for making the tasks everyone performs and oversees assigning new tasks to people when asked. The Project Manager ensures there is always work to be done and knows what areas of the project need to be worked on.

The Project Manager is responsible for maintaining a kanban board that is accessible to all members of the team. There should be at least 4 columns: Not Started, In Progress, Waiting for Merge, and Finished. I highly recommend using the software called Jira. This software is free to use and has nice features to aid in development like generating git commands to use to create new branches.

Every task will start in the Not Started column and be unassigned. Here, team members can claim tasks made by the Project Manager. The task entries will have the following format, at minimum:

- Name of the task
- Detailed description of task
    - State scope and condition for completion
    - Include any links/images to designs or references
- A point value of 1-4
    - 1: Should be able to be completed in about 1-2 days
    - 2: Should be able to be completed in about 3-4 days
    - 3: Should be able to be completed in about 5-6 days
    - 4: Probably will take a week to complete

Bi-weekly report: Kanban Update

Every two weeks, a report will need to be generated regarding at least 5 of the tasks completed during those two weeks, one per member. For each task reported here, it will take the following form:

- Name
- Date added
- Date completed
- Who completed the task
- Point value assigned
- Point value reassessment
    - This is what you feel the point value of the task should have been if this were to be assigned again
- Reflection on why you would reassign the task to the new point value
    - Note: sometimes point values for the same task will differ between teams. Be honest in your reflection because this is to better gauge what your team is capable of and what their weak points are.

Communications Lead

This role is responsible for the communication of the group. From being the one to submit group assignments to leading the standups, the Communication Lead is to ensure everyone is aware of where all group members are at in their progress.

Every week, the Communication Lead is responsible for submitting the weekly standup (detailed below). If for whatever reason the Communication Lead communicates they are unable to turn in the compiled standup, the pecking order for submission follows:

- Communication Lead
- Project Manager
- Git Master
- Design Lead
- Quality Assurance Tester

Weekly Standups

Requirements

- Communication Lead submits a document containing the response of each team member

Standup(3 parts)

- What did I work on from last time?
- What am I currently working on?
- Are there any issues holding me back for completing my tasks?

Task completion(Not required for the first 2 standups)

- List the specific tasks you completed for the week
    - Format
        - Task Name
        - Date completed
        - Point value
- To get full marks for this part, you must complete at least 4 points worth of tasks
    - Tasks are detailed more in the Project Manager section.
    - Opportunity of up to 2 bonus points for completing extra tasks.

Bi-weekly report: Group Meeting Summary

Give a paragraph or two detailing the meetings that took place. I highly encourage meeting more than once a week, but once a week should be a minimum for how often everyone meets up to discuss the project and progress. However, this report only requires the minimum of an all-hands group meeting at least once every two weeks. I plan to give sometime on the final class day of the week to hopefully complete a standup, but this is not guaranteed nor is it enough to conduct a meaningful group meeting.

Recommend topics of discussion include but are not limited to:

- Helping a group member find resources to solve an issue they have been having
- Group members trading tasks or a group member relinquishing a task
- Detailing a timeline of when specific tasks should be done
- Group members openly communicating what tasks they intend to claim
- Sharing what new knowledge they have gained to help solve a task / problem
- Working with the Git Master to conduct their code review

Git Master

The Git Master is the ruler of the repository. Whilst everyone knows to push and pull from branches, the Git Master is tasked with managing merges to ensure all the code that goes to main works and is functioning as intended.

Proper branch management is a requirement of this project, and it is the responsibility of the Git Master to ensure this is maintained. In addition, it is the responsibility of the Git Master to ensure there is a baseline of standards in the code pushed. Every task will have its own branch from Dev. It is the role of the Git Master to resolve all pull requests from these task branches into Dev, and when the project is reaching an end, the Git Master is responsible for pulling stable versions of Dev into Main/Prod.

General Notes for the Git Master

This role, more so than others, requires active participation each week. Many tasks will require the Dev branch to be up to date with all work of the previously completed tasks because these new tasks will require the foundation already built.

The Git Master will also likely be the first person relied upon whenever issues with Git arise. It will be tougher at first as everyone gets used to using Git, likely for the first time. But, like I stated in the syllabus, feel free to ask me any questions you may have.

Bi-weekly report: Code Review

NOTE: The first time submitting a Code Review will instead be a document of the code standards the group will be using. Each language is different, and examples can be found online. A good example is looking at how Amazon documents their code quality: https://docs.aws.amazon.com/wellarchitected/latest/devops-guidance/dl.cr.1-standardize-coding-practices.html.

Now, I don't expect the document to be nearly as verbose. However, some key points that should be included as a minimum will be variable naming convention, max nesting depth, tolerance for repeated code / making functions, function naming convention, class naming convention, and file organization.

For every submission thereafter, the code review will consist of listing a pull request submitted by each member and noting any changes made to be consistent with the code standards. In addition, work with the Communications Lead to get with the group member(s) that need their code fixed to limit future instances.

Design Lead

From code to UI, the Design Lead is responsible for working with everyone to ensure the vision is met. Throughout the entire project, there will be a need to plan out and envision what will be made, and this role is the primary planner for how the Project Manager will base their tasks (at least for the front-end).

Software to use to aid in designs will vary depending on the project, but for most projects, the software known as Figma should suffice. This is a free platform for the purposes it would be used for this project. Additionally, it is possible to get a license for Adobe Creative Cloud through LSU. Check the following link for details: https://grok.lsu.edu/Article.aspx?articleid=20276.

The Design Lead is responsible for iterating designs for the group, helping to give a visual of what the end user will interface with. Now, the expectation of the involvement for designs will be different based on the type of project decided upon by the group. The expectations are as follows:

- Games
    - Prototypes for level maps
    - Must support monitors with aspect ratios of either 16:9 or 4:3
    - Custom 2D sprites must be made for characters and items (if a 2D game)
        - Sprites must also include animation frames
- Mobile Apps
    - Page designs for both iOS and Android
    - App must include at least 6 pages
- Web Apps / Websites
    - Responsive web design for all pages for (choose 2) desktop, tablet, and mobile resolutions
        - Mobile: 375 x 812 (iPhone 11 Pro)
        - Tablet Resolution: 810 x 1080 (iPad)
        - Desktop Resolution:
            - 1440 x 900 (MacBook Air)
            - 1920 x 1080 (Standard monitor)
    - Site must include at least 6 pages

Bi-weekly report: Design Update

Submit screenshots or a link to view the designs in their current state. In addition, write a few sentences about the motivation for the current designs.

Quality Assurance Tester

If in a group of 4, all members share in this responsibility equally. This person is dedicated to finding bugs in the software and figuring out how to fix them. Do make sure to communicate to the Project Manager to possibly make tasks or track down the member responsible for the code. Work with them to ensure the final demo is as clean as possible.

As Quality Assurance Tester, the goal is to identify any and all bugs in the software. It should be noted that bugs are simply unexpected execution of code. These can range from minor issues like small visual flickers to major issues like the program crashing. This role is very critical to ensuring the final product runs as smoothly as possible.

Every time a bug is found, it is up to the Quality Assurance Tester to submit a task to the Kanban board with the following information:

- SEVERITY plus name of bug
    - Severity has 3 rankings
        - Severe: Results in crashes or makes the software otherwise unusable
        - Moderate: Results is a degraded user experience. Examples include, but are not limited to, deleting user input, long load times,
        - Minor: Results in annoyances on the user's end, but doesn't completely hamper the functions of the software
- Description of bug plus steps to recreate
- Screenshots of bug in action (required if possible to get in the first place)
- Point value between 1-4 (see Project Manager section for the meaning of each point)
    - Usually, most bugs can be fixed in under 4 days, so it will be common for bug reported tasks to only be 1 or 2 points

If need be, communicate with the Project Manager about how to properly add tasks to the Kanban Board. Since these bug reports are tasks, these will count towards the points needed for weekly task completion. However, to those thinking to game the system on purposely leaving in bugs to fix for easy task points, every bug left in the software is marks against the final code grade. Also, the more time spent fixing bugs is less time finishing features and the final product.

NOTE: For websites, make sure to test on at minimum, a Chromium browser, a Mozilla browser and Safari as these 3 browsers have slightly different styling compilers.

NOTE: If the group only has 4 people, the bi-weekly report won't be for a grade, but I do recommend going through the motions for it.

Bi-weekly report: Bug Report

NOTE: the Quality Assurance Tester will not have to give a bug report for the first two weeks of the project as groups are just starting to code, and there likely won't be anything to report. If there are no bugs to report, write a paragraph about your process for looking for bugs for those 2 weeks.

This is a list of all bugs found and / or fixed within the past 2 weeks, up to 5. For each bug on the report, report them in the following form:

- Name of bug
- Severity
- Date found
- Date fixed
- Underlying issue that caused the bug

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.

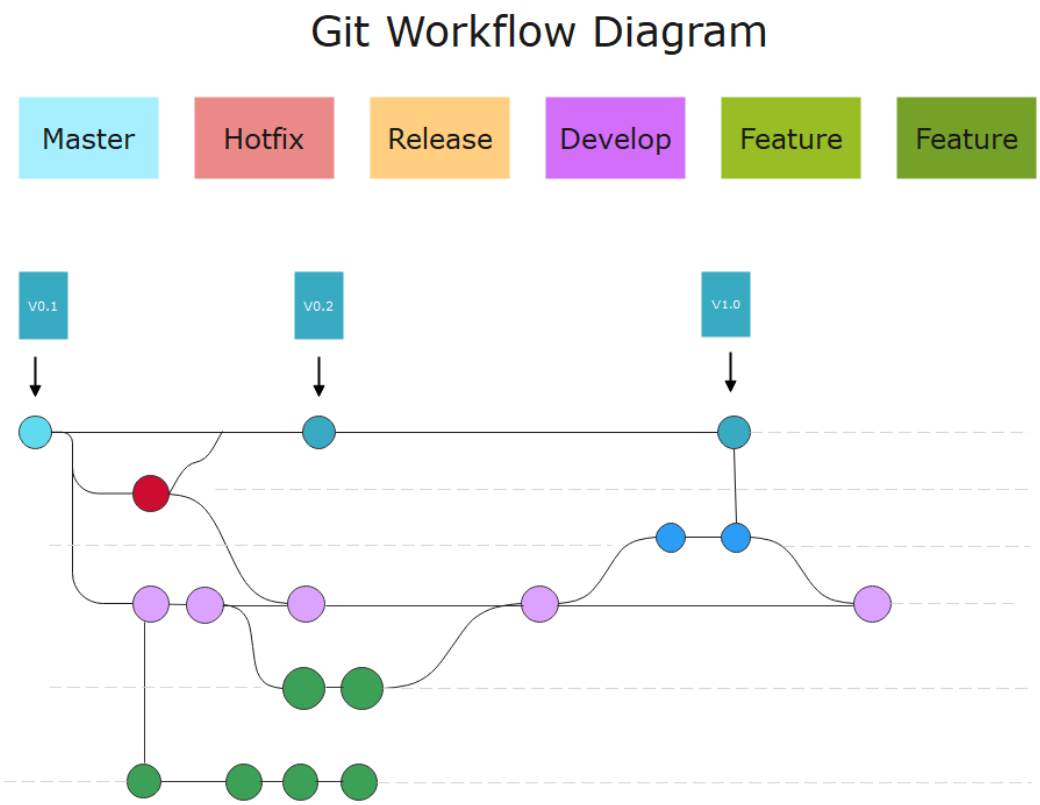
NOTE: All code must be your own. NO USING GENERATIVE AI. If AI is suspected to have been used to create the code in the project, the entire group will receive a 0 for the code section. The reason for this harshness is due to the overwhelming affects reckless use of AI has on codebases. Not only will 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.

In addition, **every** commit made must have a meaningful message accompanying the commit. And, **every** pull request made must have a detailed message.

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.