CS455 2024-25 SEMESTER-1
Homework-2

Problem Statement:

Building on your work from Homework-1, your task is to enhance your existing web-based game by improving code quality, ensuring maintainability, and adding comprehensive unit tests. This homework focuses on adding code quality metrics to your CI pipeline, refactoring your codebase, and increasing test coverage.

Requirements:

1. Add Code Quality Metrics:

- Integrate tools to assess code quality within your CI pipeline. Examples include ESLint for JavaScript, Pylint for Python, or any other language-appropriate tool (including for any server side code you might have).

- Set up / change conventions as you see fit for your project (e.g., do you want {} for single statement blocks or not?).

- Ensure that your CI pipeline fails if the code does not meet the set quality standards.

- Include metrics like code complexity, test coverage, and code duplication.

2. Refactor Your Code:

- Review your codebase from Homework-1 and identify areas where code can be improved.

- Specifically, make your previous build go green, by reducing complexity, refactoring out duplicates, improving readability and cleaning up other code smells.

- Document any major changes made during the refactoring process.

3. **Add Unit Tests:**

- Write unit tests for your game's core functionalities.

- Ensure that **at least 50% of your codebase is covered** by these tests.

- Integrate these tests into your CI pipeline so that they are run automatically on every commit.

- Ensure that your CI pipeline includes these tests and that it fails if any test does not pass.

- Ensure coverage metrics are part of your code quality metrics. Also, add linting to your test code (no duplicates, for example).

4. Continuous Integration:

- Continue using CI/CD to deploy your game automatically on every commit.

- Ensure that the deployed game remains fully functional after all changes.

5. Documentation:

- Update your README file to include information about the code quality tools used, how to run them, and a summary of the refactoring and testing changes.

- Ensure your GitHub repository remains public.

- Add a folder with the initial (say, from homework-1) code quality and test coverage metrics in a folder, and then a final report when you feel done. That way, we can all see how far you have come.

- Anything else for the TAs to know where to find/evaluate.

Grading:

- Individual contributions will be graded based on the amount of code you refactor, the quality improvements you make, and the unit tests you write. Additionally, we'll also look into who has added the metrics to the pipeline, etc. Everyone does basic devops!

- As always, break everything down to tasks first, assign tasks to yourself and track as you complete. Gathering precise data on time tasks take is important for practical software engineering.

- The team will be graded on the overall improvement in code quality, test coverage, and the successful implementation of the CI pipeline.

- The latest version of the game (after refactoring, etc) should still be hosted and working

How to turn in:

- Adds a submission (PDF/word/text) to HelloIITK, with link to github repository and the hosted game.
- Tell us where to find the reports for code quality (for both the code and the test suite), test run and pass/fail reports, test coverage, etc. on the CI server. If you add it to README, say so and tell us exactly where to find them.
- Turn in one submission per team, duplicate submissions will be simply ignored.

When is it due?

- 12th September, 11:59 pm.

Word of advice:

1. Start early, writing tests will require significant refactoring and therefore take time.
2. Making things clear is your responsibility, lack of clarity / documentation on where to find something will be penalized.