Code Comments Specification

Motivation

Code comments is a platform to help you take any repository and add comments to it, which can be used for code reviews, feedback, or any other purpose. It is designed to be simple and easy to use, with a focus on clear and consistent commenting so you can help your students improve their code quality.
It provides an independent commenting interface which is not tied to the process of pull requests or issues, allowing for a more flexible and streamlined commenting experience. This ensures that the commenting process is fluid as it is the main focus of the platform rather than being a secondary feature of a larger system.

Core Features

- **Project Management**: The platform allows reviewers to manage their projects, which are essentially Github repositories. Reviewers can add, edit, or delete projects as needed.
  - **Repository Support**: The platform supports Github repositories, allowing reviewers to easily access and comment on code hosted on Github.
  - **Branch Support**: Reviewers can select a specific branch of the repository to comment on. This allows for targeted feedback on specific versions of the code.
- **Commenting**: Reviewers can add comments in various ways:
  - **Singleline Comments**: Reviewers can leave comments on individual lines of code.
  - **Multiline Comments**: Reviewers can leave comments on multiple lines of code.
  - **File Comments**: Reviewers can leave comments on entire files. This means that reviewers can provide feedback on the overall structure or content of a file.
  - **Folder Comments**: Reviewers can leave comments on entire folders. This may allow reviewers to provide feedback on the whole project organization or structure.
- **Editor-like Feel**: The commenting interface should feel like an editor, allowing reviewers to easily navigate and comment on code.
- **Code Navigation**: The platform provides a user-friendly interface for navigating through the codebase, including files and folders. Reviewers can easily browse through the code to find the specific lines they want to comment on.

- **Comment Extraction**: Comments should be extractable to a separate file, such as a Markdown file or a JSON file for easy sharing and review (raw comments). The comments can be exported in a structured format, making it easy to share feedback with others or keep a record of the comments made.

User Roles & Permissions

User Roles

- **Reviewers**: Add comments, edit or delete their comments, and manage their projects (full access). They are the primary users of the platform and have full control over the comments they leave.
- **Students**: They are provided limited access by the reviewer to the application where they can view comments on their code, but cannot edit or delete them (read-only access). No registration is required for students to view the comments.

Stakeholders

- **Reviewers**: They are the primary users of the platform, responsible for adding comments and managing projects.
- **Students**: They are the recipients of the comments and feedback provided by the reviewers.
- **Developers**: They are responsible for maintaining and improving the platform, ensuring it meets the needs of reviewers and students.

Workflow

Specific workflows for reviewers and students are designed to ensure a smooth commenting experience. The following examples outline the typical workflows for both roles where the reviewer is a teacher or mentor and the student is a learner.

Teacher Workflow Example

1. A teacher receives a link to a Github repository from a student.
2. The teacher opens the app and provides the repository url and the branch he wants to comment on.
3. The teacher navigates through the codebase, viewing files and folders.
4. The teacher adds comments on specific lines of code, files or folders as needed.
5. The teacher provides access to the student for the commented code.

Student Workflow Example

1. A student receives limited access to their commented code from their teacher.
    - The limited access is a read-only view of the comments made by the teacher.
2. The student views the comments on their code, mainly using the summary page where they are provided in a structured format.
3. The student can navigate to the specific lines of code where the comments were made and see the context of the comments.

Potential Additional Features

- **Threaded Commenting**: Allow multiple replies to a comment for better discussion. Goes hand in hand with user accounts (highly recommended).
- **User Accounts**: Allow users to create accounts to manage their comments and projects (optional).