Project Proposal

1. Overview

The purpose of the project is to develop an automated code review tool, mainly used for analyzing code problems in Python projects and tracking changes in code quality. The tool detects CWE issues (format, warning, error) in the project by integrating Pylint and calculates the overall code quality score. The final goal of the project is to provide an easy-to-use interface to help users identify and fix code problems automatically, thereby enhancing code quality and displaying the project's progress through visual charts.

After the user uploads a Python project, the system will automatically analyze the code and generate CWE problem reports, including specific instructions regarding code quality issues, warnings, and errors. The tool will provide English descriptions for each issue and its location. Then, the system will transfer these English tips and code to ChatGPT, generating Chinese modification suggestions and the modified version of the corresponding code block through intelligent conversation. Users can check these instructions and suggestions in the interface and choose to save these analysis outcomes (including the number of CWE problems and code quality score) into the database.

The system supports users in logging into their accounts to check the project's history. The number of CWE problems and code quality scores will be visualized in various types of charts, such as line charts, bar charts, and pie charts, helping users track the project's development process. Moreover, users can export historical data, generating reports in CSV and PDF formats. This tool is mainly used to simplify the code review process, especially suitable for teams to manage their project quality and achieve continuous improvement.

Through this project, users can quickly identify their code problems and check the real-time improvement status, thereby enhancing development efficiency, reducing code errors, and ultimately improving the overall quality of the project.

1. Requirements
   1. Functional Requirements
      1. Project Upload

Users can upload a Python project through a visual interface.

* + 1. CWE Problems detection

The system would analyze CWE problems in the project automatically, including code format problems, warnings, and errors. For each problem, the system would provide English tips and their locations to help users understand the basic information about the problem.

* + 1. Code Quality Score

The system will calculate and display the code quality score of the uploaded project to help users evaluate the overall code quality.

* + 1. Integrating ChatGpt

The system would detect CWE problems and transfer the English tips, code locations, and relevant code to ChatGPT to get modification ideas and the modified version of the corresponding code blocks in Chinese through intelligent conversation.

* + 1. Database Document

User can choose to store the number of CWE problems into database for follow-up checks.

* + 1. History View

The system allows users to view their submission history after logging in, including the number of CWE problems and code quality scores for each submission.

* + 1. Data Visualization

The system provides several types of charts to display the changing trends of the number of CWE problems and code quality scores, as well as the relationship between them, to help users track the progress of the project.

* + 1. Upload Custom Rules

Users can submit custom CWE problem classification criteria in the interface to cover the default Pylint rules.

* 1. Non-Functional Requirements
     1. Security

The system should protect private information and data and prevent unauthorized access through account password login.

* + 1. Usability

The system should provide a user-friendly visual interface, enabling users to use the tools conveniently, quickly understand its functions, and reduce their learning curve.

* 1. Demand Tracking

All functional and non-functional requirements should be tracked during the subsequent stages of the project, ensuring they can be achieved during the development process. The team needs to use a requirement tracking matrix to connect requirements with system design and implementation, ensuring each requirement is checked and conforms to users’ expectations after development.

1. Team Roles and Responsibilities
   1. Project Manager
2. Be responsible for the overall project plan, progress control, and resource coordination.
3. Ensure smooth communication between team members and organize project progress meetings regularly.
4. Handle risks and problems in the project and adjust the project plan in a timely manner.
   1. System Architect
5. Design the overall architecture of the system, ensuring effective interaction between modules.
6. Be responsible for technology selection, and make the development specifications and best practices.
7. Evaluate and instruct the technical schemes followed by the team during the implementation process.
   1. Frontend Developer
8. Be responsible for the design and implementation of the visual interface, ensuring the friendliness and usability of the interface.
9. Realize data visualization functions, including the display of charts and the design of user interactions.
10. Collaborate with Backend Developers, ensuring smooth data interaction between front and back ends.
    1. Backend Developer
11. Be responsible for the core function implementation of the system, including project analysis, CWE problem detection, and interaction with databases.
12. Integrate Pylint and ChatGPT, ensuring their functionality operates smoothly.
13. Design and maintain the API interface, providing data support to the front end.
    1. Database Administrator
14. Design and manage the integrity and security of the data.
15. Be responsible for read and write operations, as well as storage and querying of historical data.
16. Optimize database performance, ensuring stability when the system is under high load.
    1. Quality Assurance Engineer
17. Design test plans and test cases.
18. Be responsible for functional and non-functional testing of the system.
19. Conduct automated testing, ensuring the system's stability under various conditions.
20. Collect test feedback, collaborate with the development team, ensuring problems can be fixed in a timely manner.
    1. Documentation Specialist
21. Be responsible for project documentation writing, including user manuals, technical documents, and project reports.
22. Ensure the completeness, accuracy, and readability of the documents to assist users and team members.
23. Coordinate with team members to provide relevant information to update the documents in a timely manner.