Project Title: GenAI Test Bot

Supervisor Name: John Peng
Email: kongyijipeng@gmail.com
Organization: McMaster University
Description: We will build a bot that is integrated into Git that can write documentation and test suites for a submitted Git commit.

Goal: To create a bot integrated into a CI/CD workflow that can write documentation and test suites, which will be accurate > 90% of the time. Testing will be confined to testing Python and TypeScript code, using pytest.

Expected functionalities:

- The bot will be integrated into CI/CD pipeline and Git.
- The bot will have a feedback mechanism (not necessarily machine learning, can be manual input at the first stage) for when the generated test suites are not up to par.
- (Optional) The bot will take user feedback and use it to fine-tune a LLM, which would be specialized in generating code documentation and test suites.

Our Purpose & Stakeholders:
The GenAI Test Bot project aims to reduce intensive manual labor by building a bot that helps them document and test code with up to 90% accuracy. This will benefit developers by automating the creation of documentation and test cases, which is time-saving and helps enhance the quality of code written. As a result, project managers will enjoy improved code quality, faster development cycles, and streamlined project management. Furthermore, quality assurance teams will have access to comprehensive, automatically generated test suites, simplifying testing efforts. Lastly, end users will benefit from higher-quality software with fewer defects, ensuring a more reliable user experience.

Plan of Action:
As a starting point, a working set of test cases will already have been written, with a lot of scaffolding done and code commented. We will experiment with explicitly building code level semantics into generation, such as taking advantage of CFGs to predict execution paths. We will also look into potentially designing a system that takes user feedback and uses it to fine-tune a LLM, which will not only reach parity and beyond in terms of code generation performance, but also cost only a fraction in the long run, as opposed to being completely reliant on OpenAI GPT services. The performance of the bot can be improved iteratively in this way.

We will not be needing any data sets per se, but we will be needing data on the test cases, which will be provided to us by our supervisor.

Tools/programming languages/product management systems used will be:
Python, TypeScript, Git, Github APIs, OpenAI APIs, AWS.