HyiKi / **ai-review-code** Public

main Branches Tags Go to file Go to file Code

2 Commits

.gitignore

READM…

READM…

requirem…

review.py

review_s…

review_s…

About

AI Automated Code Review For GitLab

#python #git #gitlab #ai #openai #codereview #chatgpt #chatgpt-bot #codereview-bot

Readme

Activity

0 stars

1 watching

0 forks

Report repository

ublished

Packages

No packages published

Languages

● **Python** 100.0%

README

AI CODE REVIEWER

English/[illegible]

Introduction

The importance of AI-powered code review is introduced in this project. Code review is an essential part of daily development work. Leveraging AI for automated code review can enhance our development efficiency. AI can help us detect potential errors in code, evaluate the overall code quality, and provide specific improvement suggestions.

Setup

This project is designed for the GitLab platform. To get started, you will need to prepare your OPENAI key, GitLab URL, and GitLab token.

Step 1: Install Project Dependencies

Install project dependencies by running:

```
pip install -r requirements.txt
```

Step 2: Configure Basic Settings

Set your OPENAI key, GitLab URL, and token as environment variables:

```
export OPENAI_API_KEY=your_openai_api_key
export GITLAB_URL=your_gitlab_url
export GITLAB_PRIVATE_TOKEN=your_gitlab_private_token
```

Step 3: Optional - Proxy Setup

If GitLab is in an intranet environment, set up a proxy by adding an additional environment variable:

```
export GITLAB_PROXY=http://your_proxy_server:port
```

Step 4: Perform AI Code Review

To conduct an AI code review, follow these steps:

```
python review_script.py <project_id> <merge_request_id>
```

Demonstration

Let's demonstrate the code review process with the following code snippet in Python:

```
a = 1/0
```

Review Result

The provided code snippet consists of a single line that attempts to divide 1 by 0, which will lead to a ZeroDivisionError in most programming languages that perform arithmetic operations. This represents a logical error and a lack of error handling.

Feedback:

- **Error Handling**: The code attempts to perform a division by zero, which is undefined and will result in a runtime error (specifically `ZeroDivisionError` in languages like Python). It is crucial to add error handling around this operation to manage such cases gracefully. For example, wrapping the operation in a try-except block (in Python) would allow the program to catch the error and handle it appropriately.
- **Code Quality**: Given the simplicity of the code, there isn't much to evaluate in terms of complexity or redundancy. However, the attempt to divide by zero indicates a lack of understanding or a mistake in the logic that should be addressed.
- **Bug Detection**: The primary issue with this code is the division by zero error, which is a significant bug that will cause the program to crash.

Recommendation: To improve this code, you should ensure that divisor is never zero before attempting division or catch and handle the potential exception that could be thrown due to a division by zero. Here's a simple example of handling this in Python:

```python
try:
    a = 1 / 0
except ZeroDivisionError:
    a = None  # or handle appropriately
```

This code doesn't impact signatures of exported functions, global data structures, variables, or the external interface of any broader software system, given its simplicity and standalone nature. However, without proper correction and error handling, it's not suitable for merging into a codebase.

Usage Scenario

Set up a webhook in your GitLab project to automate AI code reviews for merge requests.

Feedback

If you encounter any issues during the usage or have any suggestions, feel free to reach out for discussion at any time.