SbBot: Generate keyword and Rflint fix

專題編號　　:114-CSIE-S014
報行期限　　:113 年第 1 學期至 114 年第 1 學期
指導教授　　:陳香君
專題參與人員：111590058 陳軒元
　　111590059 楊君威

一, 摘要

This project develops a web-based chatbot system using HTML, CSS, JavaScript, and Python. The chatbot integrates Retrieval-Augmented Generation (RAG) to perform keyword and code generation based on PDF rule documents and JSON datasets. It also includes an RFLint checking feature for Robot Framework, which automatically cleans and verifies code formatting. The system provides an interactive interface for viewing, comparing, and managing code revisions directly on the website.

關鍵詞: sbBot, keyword generation, chatbot.

二, 緣由與目的

When developing Sunbird test cases, several problems were found, such as the need to follow strict keyword naming rules, the difficulty of understanding existing code, and the manual process of fixing RFLint formatting. These challenges showed the need for a smarter and more efficient solution. This project aims to build an AI-based support system that helps generate correct keyword names, create code from existing examples, and automatically find and fix RFLint format errors based on company rules. By using this system, the test case writing process can become more efficient, accurate, and easier to manage.

三, 研究報告內容

This project aims to generate the most suitable keyword names, improving readability and clarity. To achieve the desired results, we will pursue the following approach:

1. Generate keyword name:

This chatbot's feature generates clean keyword names by retrieving similar existing keywords from a FAISS index and prompting an LLM with predefined rules. It supports generating one or more suggestions based on user input and returns concise, rule-compliant names.

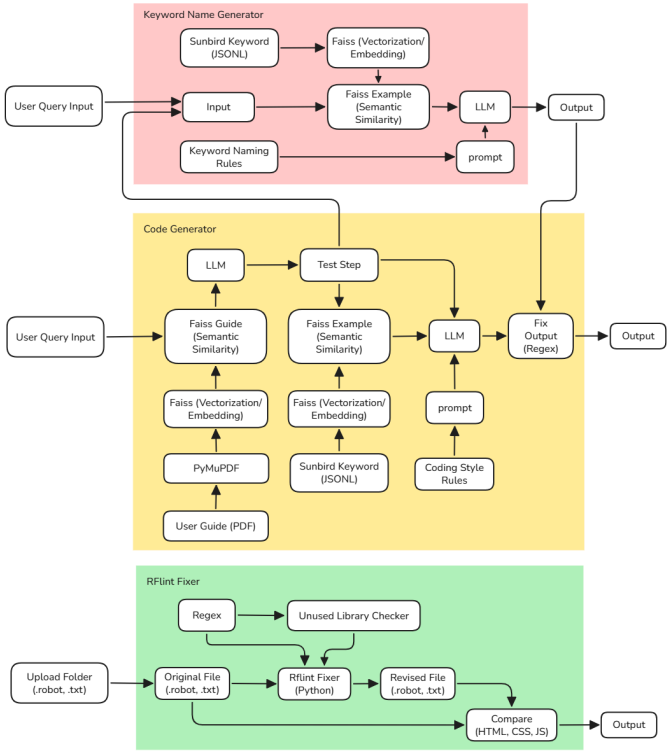
2. Generate keyword code:

This chatbot's tool generates code using embedded documents. User queries are parsed into step-by-step scenarios, used to retrieve similar existing keywords. It then generates a clean keyword name and code by prompting an LLM with predefined rules, and the results are further refined for consistency.

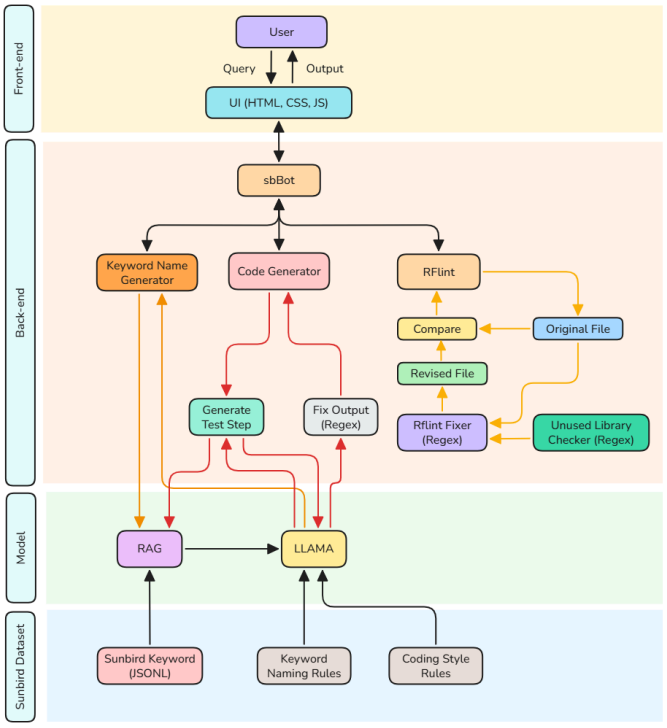
3. Rflint Format Fixer:

This feature automatically corrects Robot Framework code formatting based on company standards using Python and the re library. The system generates a properly formatted version and compares it with the original file through a web interface built with HTML, CSS, and JavaScript, allowing users to accept or ignore suggested changes. Additionally, the Unused Library Checker analyzes each file to verify whether imported libraries and resources are actively used, ensuring cleaner and more efficient code management.

四, 使用技術方法



五, 架構流程



六, 實驗結果

1. Generate keyword name :

Metric	Result
Attempt	Few, mostly single pass
Output consistency	High (85%)
Semantic accuracy	High (85%)

2. Generate keyword code :

Metric	Result
Attempt	Often requires retries
Output consistency	Moderate (70%)
Semantic accuracy	Mixed (60%)

2. Rflint Format Fixer :

Metric	Manual Fix	RFlint Fixer
Complexity	O(n . m)	O(n)
Output consistency	Varies by user	Consistent

七, 結論

In conclusion, this chatbot system effectively addresses the main challenges in test case development by automating keyword naming, code generation, and RFLint checking. These features improve efficiency and reduce human errors, allowing team members to focus on more essential testing tasks. Through the use of AI and RAG methods, the system also maintains consistent code formatting in line with company standards. In the future, further enhancements can be made to expand its functions and improve its performance for larger-scale projects.

八, 參考文獻

[1] A. Mezhlumyan, “Building a Retrieval-Augmented Generation (RAG) System for Academic Papers,” B.S. thesis, Zaven and Sonia Akian College of Science and Engineering, American University of Armenia, Yerevan, Armenia, May 2024. [Online]. Available: https://cse.aua.am/files/2024/05/Building-a-Retrieval-Augmented-Generation-RAG-System-for-Academic-Papers.pdf

[2] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, et al., “LLaMA 2: Open Foundation and Fine-Tuned Chat Models,” Technical report, Meta AI, Menlo Park, CA, USA, Jul. 18, 2023. [Online]. Available: https://arxiv.org/abs/2307.09288

[3] PyMuPDF, “PyMuPDF Documentation,” *Read the Docs*, Oct. 2025. [Online]. Available: https://pymupdf.readthedocs.io/en/latest/.