Prerequisites

When arriving on the platform, we should first be prompted with a few inputs to set up the LLM as well as the repository that we want to work with. From that point on, we have different sections that represents

Start APP

Frontend

Inputs:
Repository Link
Branch Name
Language (only one) (Python by default; C++, js, etc)
LLM to use (openLLM, or HuggingFace)

Backend

Instantiates:
RAG
LLM
Save/Retriever/Executor Tools
Sends:
A signal when everything is ready

Use cases

Frontend

Backend

Generate a test for a specific file or function

Inputs:
- File or function in question
- Option or button to save file ?

Output:
In conversation chain, so we can have upcoming queries

Inputs:
- Question: Generate test
- RagWrapper
- filePath to save the test file in (generated ?)

Output:
- LLM output
- File

CONV_CHAIN

Evaluate a test and correct or complete if necessary

Inputs:
- File or function in question
- Dynamically detect all test file then choose a file

Output:
In conversation chain, so we can have upcoming queries

Inputs:
- Question: Evaluate test
- RagWrapper
- Code executor tool --> put the output back in conversation **loop** and reevaluate
Save tool

Output:
- LLM output at each execution stage

Loop, execute and correct code until done

CONV_CHAIN

Log reader: read a log and get inforrmation

Inputs:
- Filepath of logs in question
- file format (txt)
- Query

Output:
In conversation chain, so we can have upcoming queries

Inputs:
- Question: Explore log and answer to query
- RagWrapper

Output:
- LLm output

QA_CHAIN

Code reader: Check if a code respects specification

Inputs:
- File or function
- Specification query

Output:
Bullet point with X or V on specification

Inputs:
- Question: Check if code is correct as in specification
- RagWrapper

Output:
- LLm output

QA_CHAIN

Code reader: Detect bug in a file

Inputs:
- File or function
- Error output of an execution

Output:
- Bullet points of potential bugs (with rag)
- Bullet points of potential causes (if we provide an execution error)
- Corrected code in response

Inputs:
- Question: Check if code is correct as in specification
- RagWrapper
- BugNet RagWrapper

Output:
- LLm output

CONV_CHAIN

General Chatbot:
- Explain code
- Check if code is aligned with specification
- All other queries

Inputs:
- Question (with file, specification etc)

Output:
- LLM output

Inputs:
- Question: Check if code is correct as in specification
- RagWrapper

Output:
- LLm output

CONV_CHAIN