CodeBase buddy

Raghavan Muthuregunathan https://www.linkedin.com/in/raghavanmit/

*In the demo, **We use open interpreter to ask questions about open interpreter code base :-) (Slide 12/13)**

Problem statement

- Imagine you are trying to contribute to a **new code base** (a github repository) for a beginner task.
- Knowing **which** file to change, **where** to make the change can be time consuming.

We've all been there. You're enthusiastic about contributing to a new GitHub repository, but you're overwhelmed. Which file do you modify? Where do you start? For newcomers, the maze of a new codebase can be truly daunting.

What if **Code-Interpreter** solves the problem for you ?

CodeBase buddy

- The "CodeBase Buddy" system leverages the power of retrieval augmented generation technique + GPT + code interpreter’s capabilities to guide users through their first tasks in a new codebase.
- By analyzing the repository's structure and files, it builds an **vector index.**
- Users can input a task description, and the system would subsequently **offer step-by-step guidance**, akin to having an experienced developer by their side.
- When user asks a question, it provides specific pointers on
  - which files to modify and
  - how to go about those changes

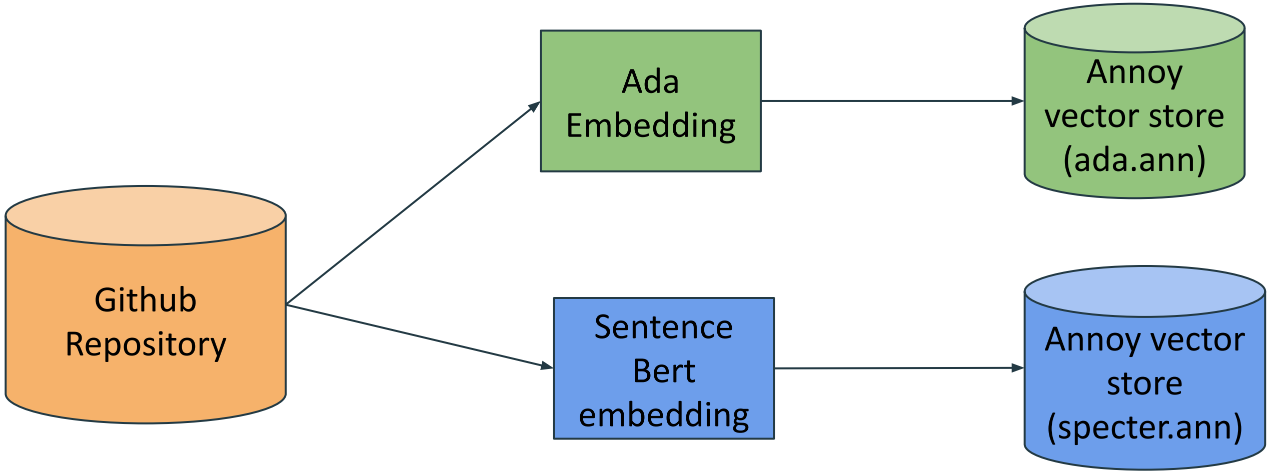
Technical solution

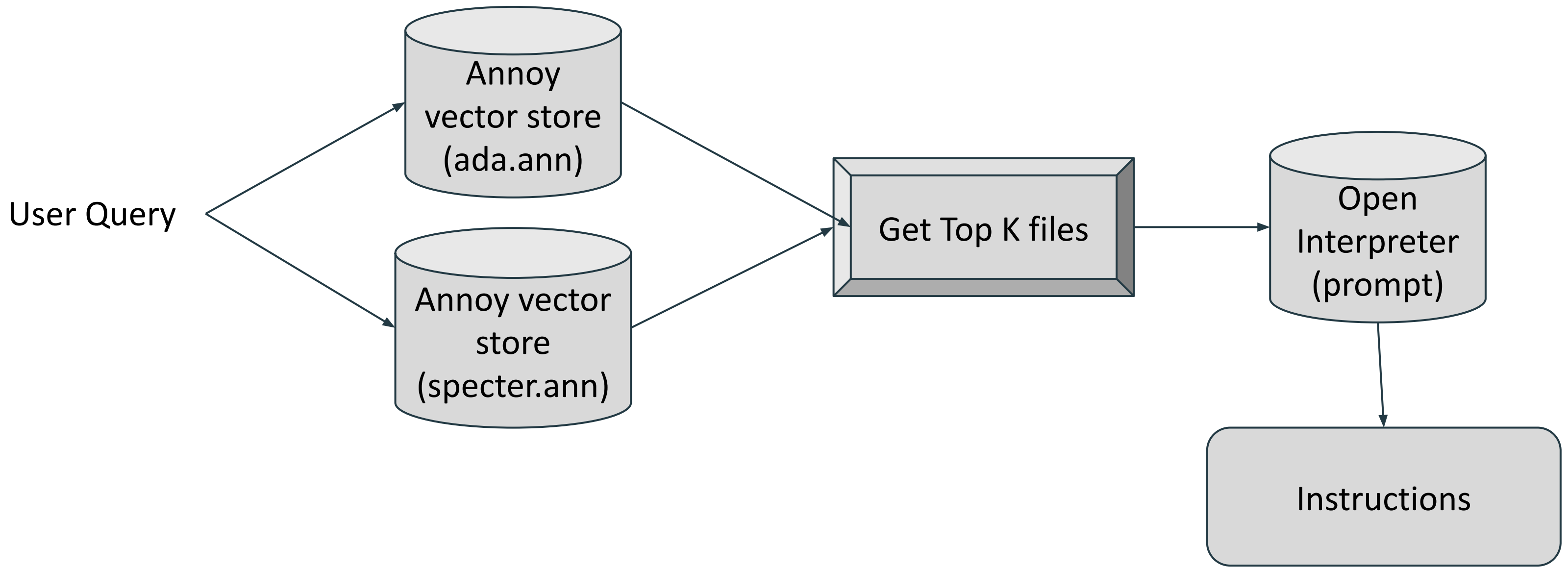
The technical solution is based on the following 3 steps.

1. Build a vector index generating embedding for every file (.py .java)
2. Query the vector index
3. Leverage Code interpreter to provide instructions

Build embeddings

- Traverse every file matching *.py
- Read the content and generate an embedding
  - Using OpenAI's Ada embedding
  - Using Sentence BERT embedding
- Build 2 annoy indices one for each embedding (vector store)


Github Repository
Ada Embedding
Sentence Bert embedding
Annoy vector store (ada.ann)
Annoy vector store (specter.ann)

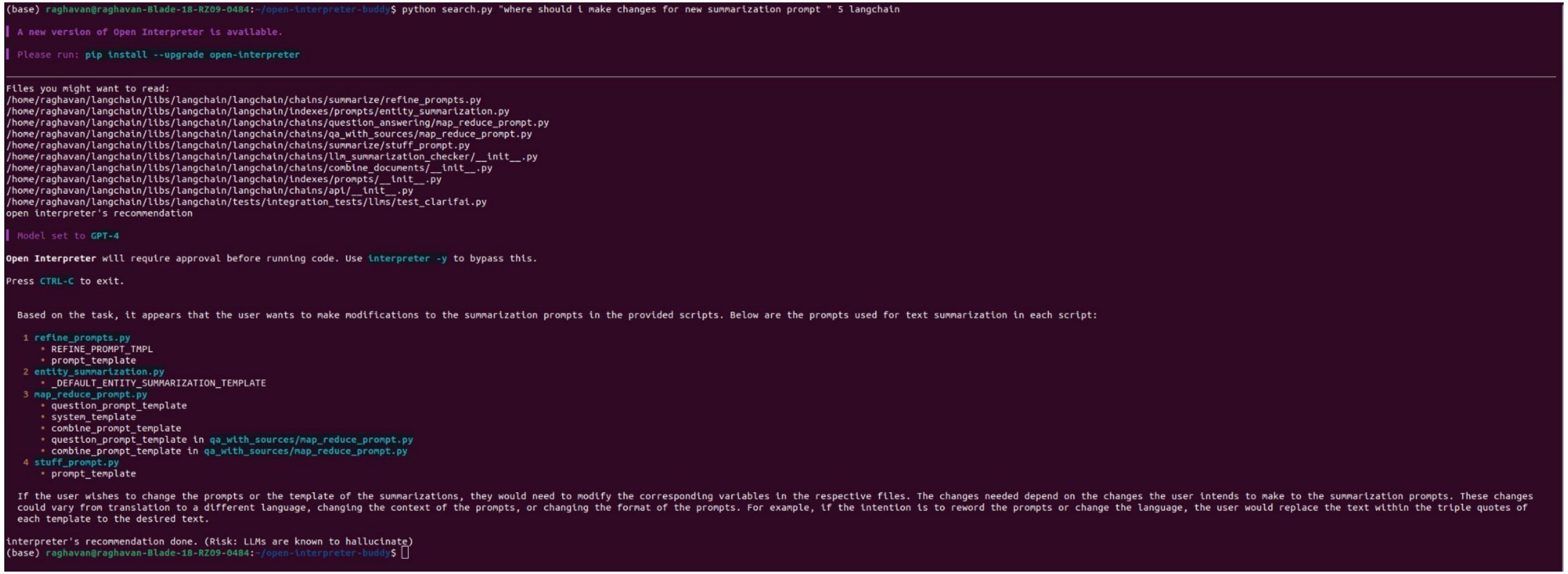

User Query
Annoy vector store (ada.ann)
Annoy vector store (specter.ann)
Get Top K files
Open Interpreter (prompt)
Instructions

For this hackathon demo, I chose 4 repositories

- Built 4*2 indices (1 using Ada, 1 use SBERT)
- Queried both
- Consolidated the results
- Used open interpreter to recommend changes

Github repository	**Number of python files**
LangChain	1983
Llama index	779
Our very own **open-interpreter** :-)	55
localGPT	9

Langchain: where should i make changes for summarization prompt

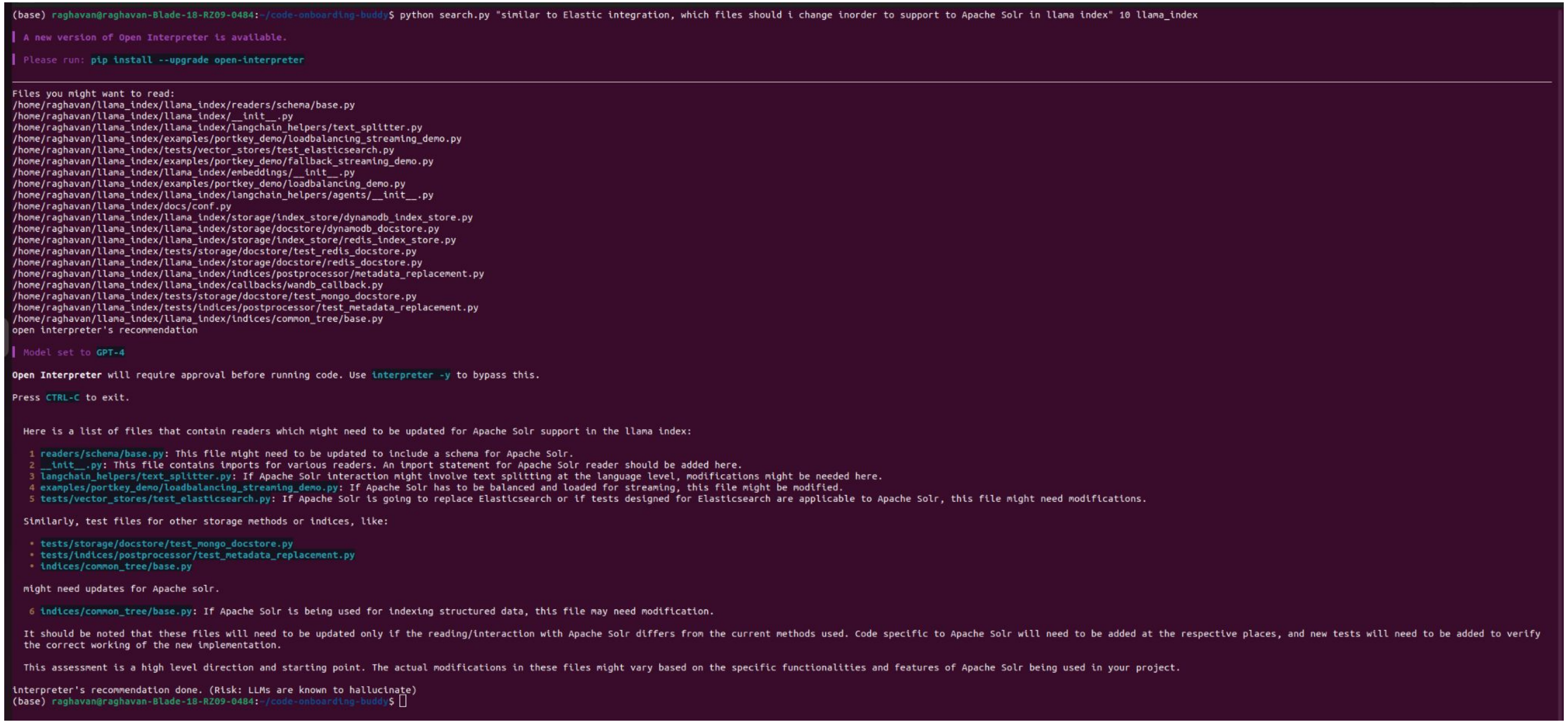


```
(base) raghavan@raghavan-Blade-18-RZ09-0484:~/open-interpreter-buddy$ python search.py "where should i make changes for new summarization prompt " 5 langchain

▌ A new version of Open Interpreter is available.

▌ Please run: pip install --upgrade open-interpreter

────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────
Files you might want to read:
/home/raghavan/langchain/libs/langchain/langchain/chains/summarize/refine_prompts.py
/home/raghavan/langchain/libs/langchain/langchain/indexes/prompts/entity_summarization.py
/home/raghavan/langchain/libs/langchain/langchain/chains/question_answering/map_reduce_prompt.py
/home/raghavan/langchain/libs/langchain/langchain/chains/qa_with_sources/map_reduce_prompt.py
/home/raghavan/langchain/libs/langchain/langchain/chains/summarize/stuff_prompt.py
/home/raghavan/langchain/libs/langchain/langchain/chains/llm_summarization_checker/__init__.py
/home/raghavan/langchain/libs/langchain/langchain/chains/combine_documents/__init__.py
/home/raghavan/langchain/libs/langchain/langchain/indexes/prompts/__init__.py
/home/raghavan/langchain/libs/langchain/langchain/chains/api/__init__.py
/home/raghavan/langchain/libs/langchain/tests/integration_tests/llms/test_clarifai.py
open interpreter's recommendation

▌ Model set to GPT-4

Open Interpreter will require approval before running code. Use interpreter -y to bypass this.

Press CTRL-C to exit.


  Based on the task, it appears that the user wants to make modifications to the summarization prompts in the provided scripts. Below are the prompts used for text summarization in each script:

   1 refine_prompts.py
      • REFINE_PROMPT_TMPL
      • prompt_template
   2 entity_summarization.py
      • _DEFAULT_ENTITY_SUMMARIZATION_TEMPLATE
   3 map_reduce_prompt.py
      • question_prompt_template
      • system_template
      • combine_prompt_template
      • question_prompt_template in qa_with_sources/map_reduce_prompt.py
      • combine_prompt_template in qa_with_sources/map_reduce_prompt.py
   4 stuff_prompt.py
      • prompt_template

  If the user wishes to change the prompts or the template of the summarizations, they would need to modify the corresponding variables in the respective files. The changes needed depend on the changes the user intends to make to the summarization prompts. These changes
  could vary from translation to a different language, changing the context of the prompts, or changing the format of the prompts. For example, if the intention is to reword the prompts or change the language, the user would replace the text within the triple quotes of
  each template to the desired text.

interpreter's recommendation done. (Risk: LLMs are known to hallucinate)
(base) raghavan@raghavan-Blade-18-RZ09-0484:~/open-interpreter-buddy$ 
```

- Open interpreter used Annoy index and retrieved the relevant python files
- Later used GPT-4 to give the recommendation to change the files of
    - Refine_prompts.py
    - Stuff_prompts.py
    - Entity_summarization.py
    - Mapreduce prompt.py

This will improve the productivity of beginner to expert of the code base.

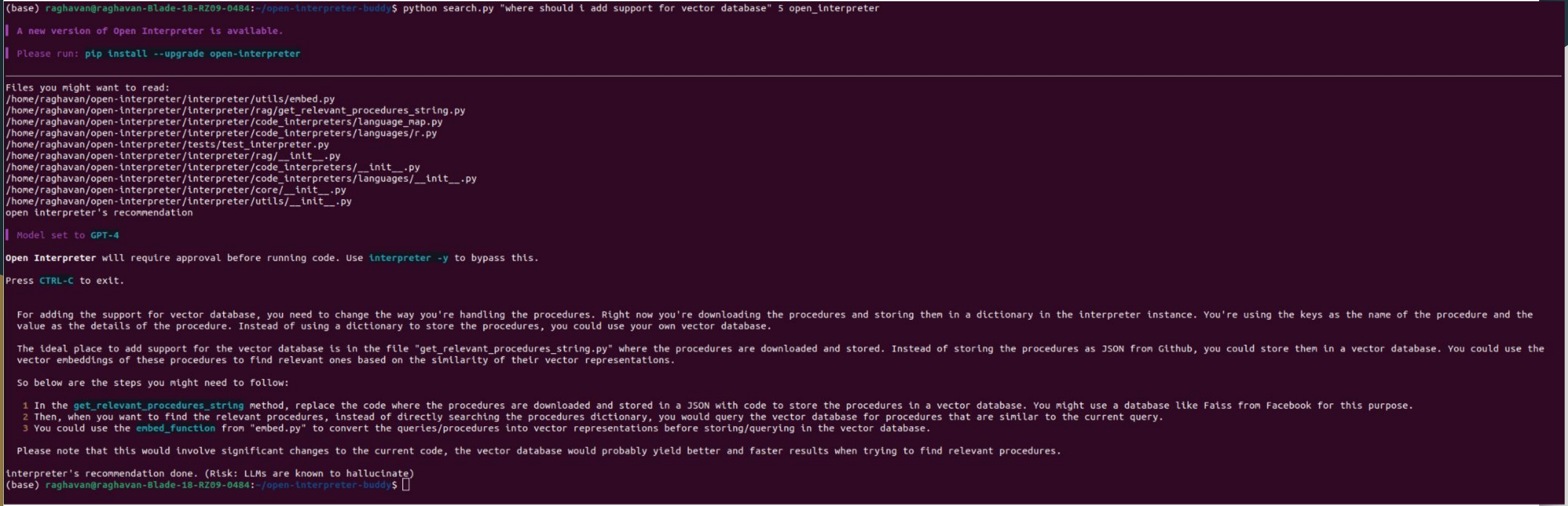
Llama_index: User wants to add support for Apache Solr similar to Elastic



```
(base) raghavan@raghavan-Blade-18-RZ09-0484:~/code-onboarding-buddy$ python search.py "similar to Elastic integration, which files should i change inorder to support to Apache Solr in llama index" 10 llama_index

▎ A new version of Open Interpreter is available.

▎ Please run: pip install --upgrade open-interpreter

────────────────────────────────────────────────────────────────────────────────

Files you might want to read:
/home/raghavan/llama_index/llama_index/readers/schema/base.py
/home/raghavan/llama_index/llama_index/__init__.py
/home/raghavan/llama_index/llama_index/langchain_helpers/text_splitter.py
/home/raghavan/llama_index/examples/portkey_demo/loadbalancing_streaming_demo.py
/home/raghavan/llama_index/tests/vector_stores/test_elasticsearch.py
/home/raghavan/llama_index/examples/portkey_demo/fallback_streaming_demo.py
/home/raghavan/llama_index/llama_index/embeddings/__init__.py
/home/raghavan/llama_index/examples/portkey_demo/loadbalancing_demo.py
/home/raghavan/llama_index/llama_index/langchain_helpers/agents/__init__.py
/home/raghavan/llama_index/docs/conf.py
/home/raghavan/llama_index/llama_index/storage/index_store/dynamodb_index_store.py
/home/raghavan/llama_index/llama_index/storage/docstore/dynamodb_docstore.py
/home/raghavan/llama_index/llama_index/storage/index_store/redis_index_store.py
/home/raghavan/llama_index/tests/storage/docstore/test_redis_docstore.py
/home/raghavan/llama_index/llama_index/storage/docstore/redis_docstore.py
/home/raghavan/llama_index/llama_index/indices/postprocessor/metadata_replacement.py
/home/raghavan/llama_index/llama_index/callbacks/wandb_callback.py
/home/raghavan/llama_index/tests/storage/docstore/test_mongo_docstore.py
/home/raghavan/llama_index/tests/indices/postprocessor/test_metadata_replacement.py
/home/raghavan/llama_index/llama_index/indices/common_tree/base.py
open interpreter's recommendation

▎ Model set to GPT-4

Open Interpreter will require approval before running code. Use interpreter -y to bypass this.

Press CTRL-C to exit.


  Here is a list of files that contain readers which might need to be updated for Apache Solr support in the llama index:

   1 readers/schema/base.py: This file might need to be updated to include a schema for Apache Solr.
   2 __init__.py: This file contains imports for various readers. An import statement for Apache Solr reader should be added here.
   3 langchain_helpers/text_splitter.py: If Apache Solr interaction might involve text splitting at the language level, modifications might be needed here.
   4 examples/portkey_demo/loadbalancing_streaming_demo.py: If Apache Solr has to be balanced and loaded for streaming, this file might be modified.
   5 tests/vector_stores/test_elasticsearch.py: If Apache Solr is going to replace Elasticsearch or if tests designed for Elasticsearch are applicable to Apache Solr, this file might need modifications.

  Similarly, test files for other storage methods or indices, like:

   • tests/storage/docstore/test_mongo_docstore.py
   • tests/indices/postprocessor/test_metadata_replacement.py
   • indices/common_tree/base.py

  might need updates for Apache solr.

   6 indices/common_tree/base.py: If Apache Solr is being used for indexing structured data, this file may need modification.

  It should be noted that these files will need to be updated only if the reading/interaction with Apache Solr differs from the current methods used. Code specific to Apache Solr will need to be added at the respective places, and new tests will need to be added to verify
  the correct working of the new implementation.

  This assessment is a high level direction and starting point. The actual modifications in these files might vary based on the specific functionalities and features of Apache Solr being used in your project.

interpreter's recommendation done. (Risk: LLMs are known to hallucinate)
(base) raghavan@raghavan-Blade-18-RZ09-0484:~/code-onboarding-buddy$ 
```

- Interpreter converted the query into embedding
- fetched the relevant file from both indices
- Used GPT4 to make the recommendation to first look into reader/schema/base.py

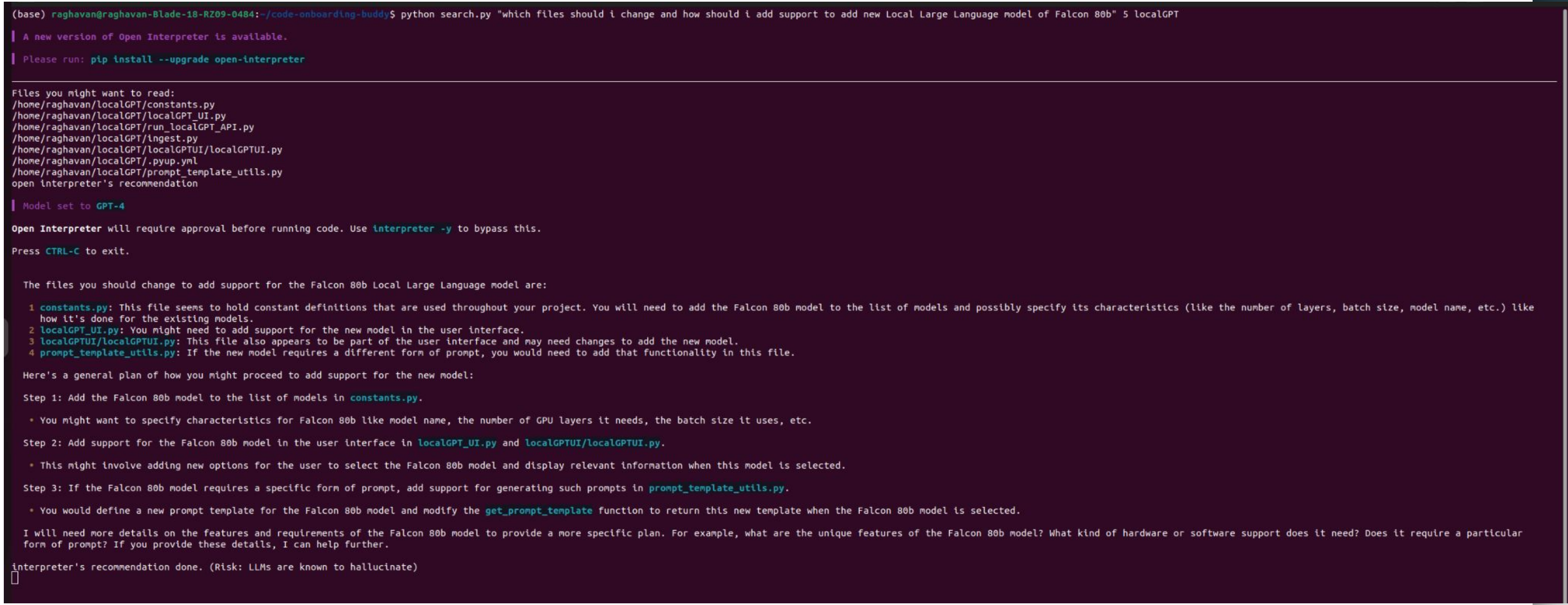
What if you want to make changes to the awesome **open_interpreter and raise a PR ? :-)**



```
(base) raghavan@raghavan-Blade-18-RZ09-0484:~/open-interpreter-buddy$ python search.py "where should i add support for vector database" 5 open_interpreter

▌ A new version of Open Interpreter is available.

▌ Please run: pip install --upgrade open-interpreter

────────────────────────────────────────────────────────────────────────────────
Files you might want to read:
/home/raghavan/open-interpreter/interpreter/utils/embed.py
/home/raghavan/open-interpreter/interpreter/rag/get_relevant_procedures_string.py
/home/raghavan/open-interpreter/interpreter/code_interpreters/language_map.py
/home/raghavan/open-interpreter/interpreter/code_interpreters/languages/r.py
/home/raghavan/open-interpreter/tests/test_interpreter.py
/home/raghavan/open-interpreter/interpreter/rag/__init__.py
/home/raghavan/open-interpreter/interpreter/code_interpreters/__init__.py
/home/raghavan/open-interpreter/interpreter/code_interpreters/languages/__init__.py
/home/raghavan/open-interpreter/interpreter/core/__init__.py
/home/raghavan/open-interpreter/interpreter/utils/__init__.py
open interpreter's recommendation

▌ Model set to GPT-4

Open Interpreter will require approval before running code. Use interpreter -y to bypass this.

Press CTRL-C to exit.


  For adding the support for vector database, you need to change the way you're handling the procedures. Right now you're downloading the procedures and storing them in a dictionary in the interpreter instance. You're using the keys as the name of the procedure and the
  value as the details of the procedure. Instead of using a dictionary to store the procedures, you could use your own vector database.

  The ideal place to add support for the vector database is in the file "get_relevant_procedures_string.py" where the procedures are downloaded and stored. Instead of storing the procedures as JSON from Github, you could store them in a vector database. You could use the
  vector embeddings of these procedures to find relevant ones based on the similarity of their vector representations.

  So below are the steps you might need to follow:

   1 In the get_relevant_procedures_string method, replace the code where the procedures are downloaded and stored in a JSON with code to store the procedures in a vector database. You might use a database like Faiss from Facebook for this purpose.
   2 Then, when you want to find the relevant procedures, instead of directly searching the procedures dictionary, you would query the vector database for procedures that are similar to the current query.
   3 You could use the embed_function from "embed.py" to convert the queries/procedures into vector representations before storing/querying in the vector database.

  Please note that this would involve significant changes to the current code, the vector database would probably yield better and faster results when trying to find relevant procedures.

interpreter's recommendation done. (Risk: LLMs are known to hallucinate)
(base) raghavan@raghavan-Blade-18-RZ09-0484:~/open-interpreter-buddy$ 
```

- **We use open interpreter to ask questions about open interpreter code base**
- User asked “how to add support to a new vector store” in open interpreter code base
- Open interpreter recommended to look into **embed.py** and asked to change
  - Get_relevant_procedures_string method
  - embed method

LocalGPT: User asks “**which files should i change and how should i add support to add new Local Large Language model of Falcon 80b**”



```
(base) raghavan@raghavan-Blade-18-RZ09-0484:~/code-onboarding-buddy$ python search.py "which files should i change and how should i add support to add new Local Large Language model of Falcon 80b" 5 localGPT

▎ A new version of Open Interpreter is available.

▎ Please run: pip install --upgrade open-interpreter

Files you might want to read:
/home/raghavan/localGPT/constants.py
/home/raghavan/localGPT/localGPT_UI.py
/home/raghavan/localGPT/run_localGPT_API.py
/home/raghavan/localGPT/ingest.py
/home/raghavan/localGPT/localGPTUI/localGPTUI.py
/home/raghavan/localGPT/.pyup.yml
/home/raghavan/localGPT/prompt_template_utils.py
open interpreter's recommendation

▎ Model set to GPT-4

Open Interpreter will require approval before running code. Use interpreter -y to bypass this.

Press CTRL-C to exit.


  The files you should change to add support for the Falcon 80b Local Large Language model are:

   1 constants.py: This file seems to hold constant definitions that are used throughout your project. You will need to add the Falcon 80b model to the list of models and possibly specify its characteristics (like the number of layers, batch size, model name, etc.) like
     how it's done for the existing models.
   2 localGPT_UI.py: You might need to add support for the new model in the user interface.
   3 localGPTUI/localGPTUI.py: This file also appears to be part of the user interface and may need changes to add the new model.
   4 prompt_template_utils.py: If the new model requires a different form of prompt, you would need to add that functionality in this file.

  Here's a general plan of how you might proceed to add support for the new model:

  Step 1: Add the Falcon 80b model to the list of models in constants.py.

   • You might want to specify characteristics for Falcon 80b like model name, the number of GPU layers it needs, the batch size it uses, etc.

  Step 2: Add support for the Falcon 80b model in the user interface in localGPT_UI.py and localGPTUI/localGPTUI.py.

   • This might involve adding new options for the user to select the Falcon 80b model and display relevant information when this model is selected.

  Step 3: If the Falcon 80b model requires a specific form of prompt, add support for generating such prompts in prompt_template_utils.py.

   • You would define a new prompt template for the Falcon 80b model and modify the get_prompt_template function to return this new template when the Falcon 80b model is selected.

  I will need more details on the features and requirements of the Falcon 80b model to provide a more specific plan. For example, what are the unique features of the Falcon 80b model? What kind of hardware or software support does it need? Does it require a particular
  form of prompt? If you provide these details, I can help further.

interpreter's recommendation done. (Risk: LLMs are known to hallucinate)
```

- Open interpreter identifies the files to be changed and gives specific step by step instruction
- Add Falcon 80 b model to list of models in **constants.py**
- Add support in User interface of **localGPT_UI.py**
- For specific prompt templates, it recommends to modify the method **get_prompt_template** in **prompt_template_utils.py**

Advantages of CodeBase Buddy

1. **Accelerated Onboarding:** New contributors can quickly get up to speed with the codebase, reducing the onboarding time.
2. **Reduced Errors:** With specific guidance, newcomers are less likely to make mistakes or introduce bugs.
3. **Increased Engagement:** A supportive tool can encourage more contributions from the community, especially those hesitant due to unfamiliarity with the codebase.
4. **Continuous Learning:** Even for experienced developers, the tool can be a means to discover and learn about lesser-known parts of the codebase.
5. **Documentation Aid:** The system can also help identify areas where the repository's documentation might be lacking or outdated.

Thank you

Github: https://github.com/Raghavan1988/CodeBaseBuddy

Linkedin: https://www.linkedin.com/in/raghavanmit/

Lablab.ai discord: rm3844

Please reach out to my linkedin profile or in lablab.ai discord.

Demo Loom:
https://www.loom.com/share/7f89bcacef6547669cad6c66a4c0a2db?sid=4f6da485-2084-4dd2-83e1-25d3dae836bb