Legacy-Archeolog

Team Edgerunners: Olaf Werner, Bogdan Jastrzębski

Plan of presentation

1. Problem solved
2. Main Algorithm
3. Example usage
4. Technologies used
5. Future development
6. Road map

Problem solved: searching for relevant files

When new programmer needs to implement an feature and he do not have an contact to the people (or AI) who implemented it, he may have to go through a number of files before he finds something useful. This process can take a long time for the programmer.

To help with this issue the Legacy-Archeolog was developed. This framework reads an repository and embeds all files, folders and commits to enable semantic search. After defining task user needs to solve, program will give an sorted list of all relevant files which are needed for task.

Main Algorithm

1. Collect all commits and embed their summaries generated by an GPT-4.
2. Recursively embed all files in an catalogue with catalogue itself having an sum of all embeddings.
3. When program receives an new task it searches for semantic similarity between task and among all other embedded entities and sorts them.
4. Files with highest similarity are the ones which programmer should focus on first.

Example usage

```
--------------------------------------------------------------------------------
task>  My task is to add a new field to students.
Related commits:
................................................................................
Added CSV student upload

................................................................................
Commit ID: ce91348a46ea669a7a835f1ab358c87ccf04ce04
Message: corrected upload template
Purpose: This commit was made to correct the students upload template. The previous code did not completely represent the required fields and behavior of the upload form leading to an incomplete form layout. To fix this, several
 lines of code have been added to include instructions on uploading a CSV file and to declare the needed fields.

Changes:
The upload form template 'students/templates/students/students_upload.html' has changed significantly. There were 31 lines of code deleted, and 6 new lines added for further enhancement and correction.

List of changes:

* students/templates/students/students_upload.html -- Introduced instructions to guide the user while uploading a CSV file. The required fields such as registration number, surname, firstname, other names, gender, parent number,
 and address have been explicitly mentioned for clarity. This inclusion will aid users in preparing the CSV file correctly before uploading.

................................................................................
```

Technologies used

1. Programming language: Python
2. Embedding framework: openai API, model=text-embedding-ada-002
3. Generative AI: openai API, model=gpt-4
4. Git parsing: GitPython
5. Vector search: hnswlib

Future development

Because of time constraints embedding was done on file and folders level. Finer granularity could be achieved but parsing an AST turned out to be more complex, especially in projects which are using multiple languages. Embedding strategy could also be more sophisticated.

Also the generative AI could be used to solve to some of the user tasks immediately, but most important confabulation/hallucination prevention technique Self-Reflection (actually running live code to see if it gets an errors) also was hard to implement during single hackathon.

Road map

1. AST parsing (2 weeks)
2. Specialized embedding strategy using GNN (4 weeks)
3. Vector database (1 week)
4. Self-reflection framework for generative AI (6 weeks)
5. Dynamic finetuning for for frameworks used in the projects (3 weeks)
6. Automatic integration with git (1 week)
7. GUI for the user (5 weeks)

Github repo

Our project is available at https://github.com/wernerolaf/Legacy-Archeolog under MIT License.