Used Technologies :

tools for seamless collaboration and sharing.
Visual Studio: It is a widely used integrated development environment created by
Microsoft, popular among developers for its powerful features and tools that make it easy to create high-quality applications for various programming languages.
Gitlab: It is a web-based Git repository manager that provides a complete platform for software development. It offers powerful features such as version control, continuous
integration and deployment, and issue tracking. GitLab is known for its intuitive interface, extensive documentation, and open-source community, making it a popular choice for teams of all sizes.
Git: It is a free and open-source distributed version control system designed to help developers manage changes to their code over time. It provides powerful tools for
tracking changes, collaborating with other developers, and managing complex projects, making it an essential tool for modern software development.
FastAPI: It is a web framework for building high-performance APIs with Python.
It provides request and response classes, automatic generation of API documentation,
and support for asynchronous programming.
Streamlit: A framework that allows developers to create and deploy interactive web applications. Streamlit provides components for data visualization, machine learning, and other data-driven applications. Its real-time update functionality is useful for exploratory data analysis, and it is optimized for performance with large datasets and complex visualizations.
os: A Python built-in module that provides a way of interacting with the underlying operating system that Python is running on. It can an be used to manipulate the file system, environment variables, process management, and more.

glob: A Python module that is particularly useful when working with large sets of files or when dealing with files that have similar names or patterns. The glob module
allows you to retrieve a list of files or directories that match a specified pattern.
subprocess: The subprocess module is a Python built-in module that provides a way to spawn new processes, connect to their input/output/error pipes, and obtain their return codes. This module allows you to run system commands and interact with the resulting output or error messages from within your Python program.
Matplotlib: It is a data visualization library in Python that provides a range of tools for creating high-quality plots, charts, and graphs.
Pandas: It is a data analysis Python library that allows users to work with various types of data, including CSV, Excel, SQL databases, and more. It provides a range of data manipulation tools, including filtering, and reshaping data, which make it easy to clean and transform data for analysis.
re: Is a Python library that provides powerful support for regular expressions, allowing developers to easily search for and manipulate text based on specific patterns or rules. It contains a variety of pattern matching, replacement, splitting, and grouping
methods, as well as other sophisticated capabilities.
openai: Is a library that provides access to state-of-the-art models that can perform
natural language tasks, including machine translation, and question-answering. The library also includes tools for fine-tuning these models on specific tasks and custom
datasets, allowing developers to create tailored solutions.
Requests: The requests library is a Python standard for making HTTP requests. It
hides the intricacies of making requests behind a simple API, allowing you to concentrate on interacting with services and consuming data in your application.
transformers: Is an open-source library developed by Hugging Face for NLP tasks. It has pre-trained models and includes tools for model

evaluation and interpretation, has an intuitive API, and is widely used in industries like finance, healthcare, and customer service.

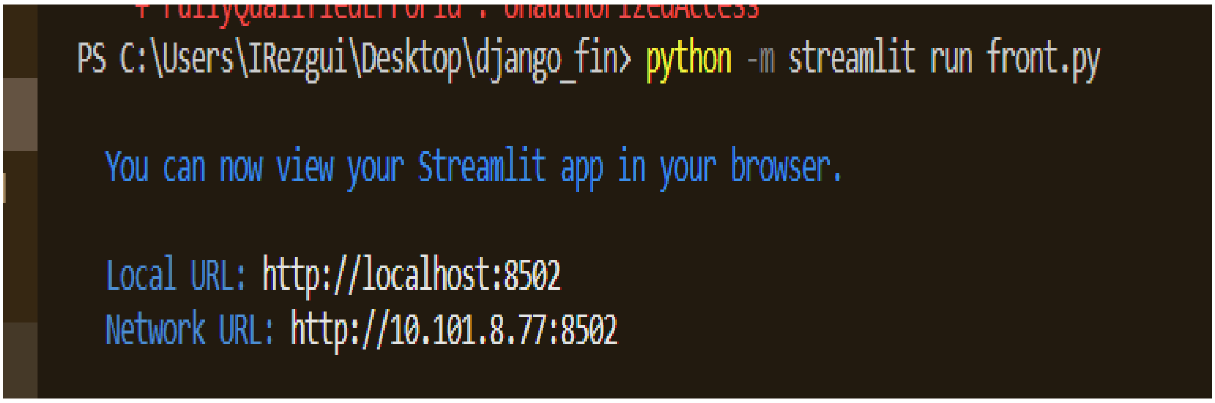
Exploitation :

To launch the front-end of the developed app all you need to do is to execute the command specified bellow under the path of the directory you're working with :

```
python -m streamlit run front.py
```

It is worth noting that front.py is the file that contains the code that will launch the front-end. You can also launch the front-end from CMD by writing :

```
python -m streamlit run "path_of_the _file_front.py"
```



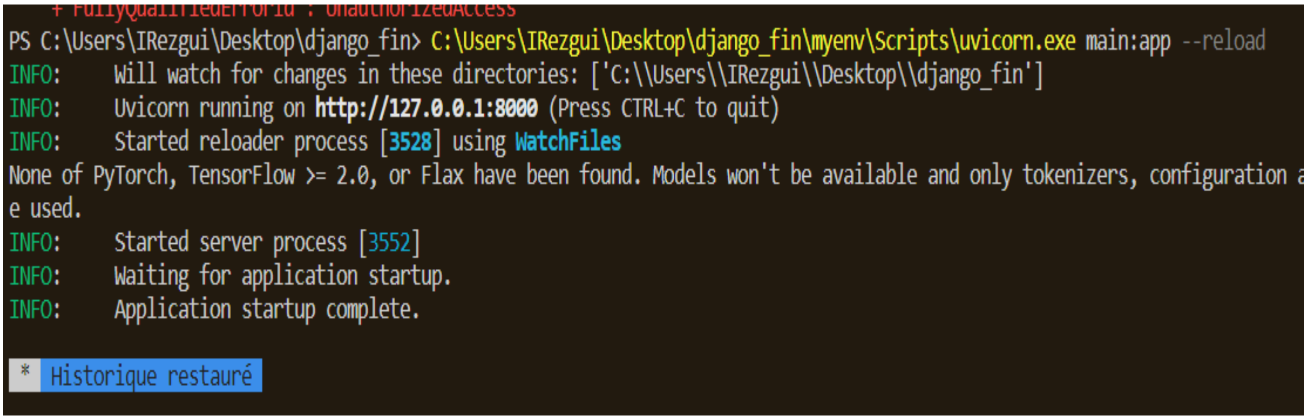
```
    + FullyQualifiedErrorId : UnauthorizedAccess
PS C:\Users\IRezgui\Desktop\django_fin> python -m streamlit run front.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8502
  Network URL: http://10.101.8.77:8502
```

To launch the back-end of the developed app all you need to do is to execute the command specified bellow:

```
Path_of_of_the_directory_of_code\myenv\Scripts\uvicorn.exe main: app --reload
```

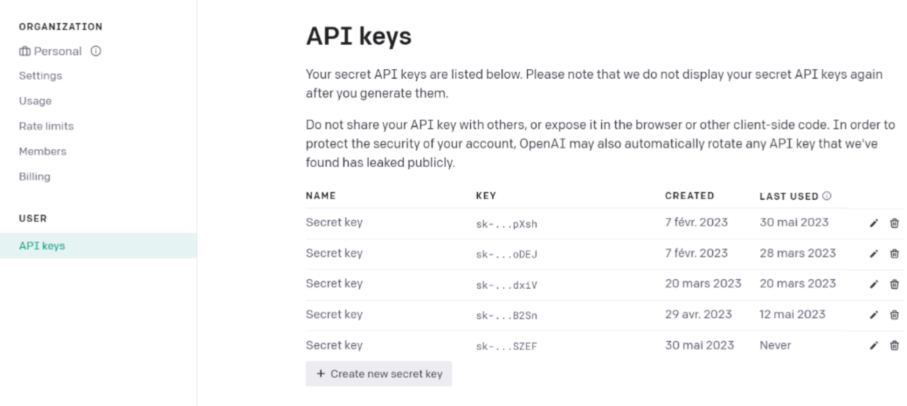


```
    + FullyQualifiedErrorId : UnauthorizedAccess
PS C:\Users\IRezgui\Desktop\django_fin> C:\Users\IRezgui\Desktop\django_fin\myenv\Scripts\uvicorn.exe main:app --reload
INFO:     Will watch for changes in these directories: ['C:\\Users\\IRezgui\\Desktop\\django_fin']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [3528] using WatchFiles
None of PyTorch, TensorFlow >= 2.0, or Flax have been found. Models won't be available and only tokenizers, configuration a
e used.
INFO:     Started server process [3552]
INFO:     Waiting for application startup.
INFO:     Application startup complete.

 * Historique restauré
```

It is also worth noting that you should add your API key from the OpenAi website so that the translation could work.

This is the url from where you can get your key : https://platform.openai.com/account/api-keys



API keys

Your secret API keys are listed below. Please note that we do not display your secret API keys again after you generate them.

Do not share your API key with others, or expose it in the browser or other client-side code. In order to protect the security of your account, OpenAI may also automatically rotate any API key that we've found has leaked publicly.

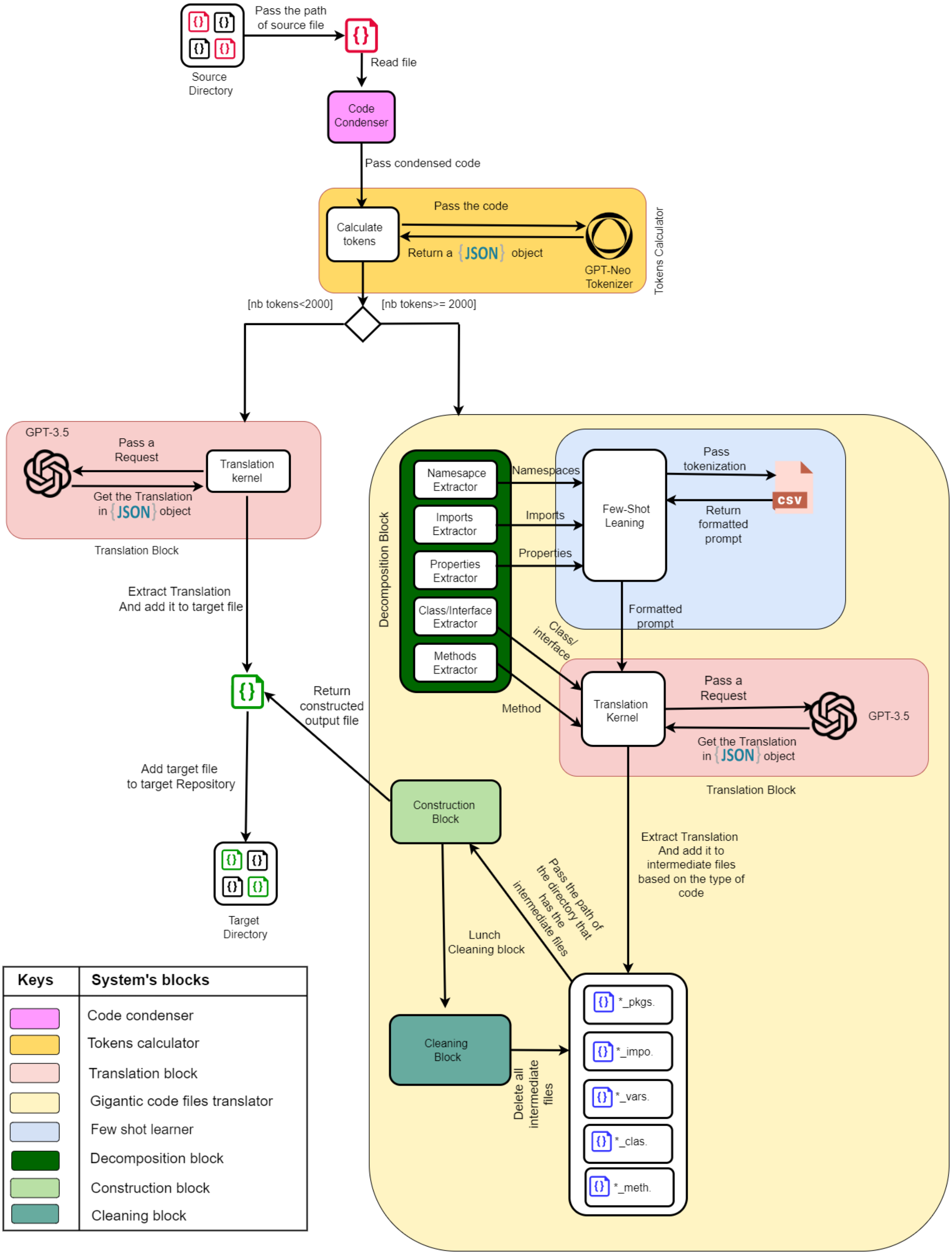
NAME	KEY	CREATED	LAST USED
Secret key	sk-...pXsh	7 févr. 2023	30 mai 2023
Secret key	sk-...oDEJ	7 févr. 2023	28 mars 2023
Secret key	sk-...dxiV	20 mars 2023	20 mars 2023
Secret key	sk-...B2Sn	29 avr. 2023	12 mai 2023
Secret key	sk-...SZEF	30 mai 2023	Never

One thing you can do to make your work easier is to set your key as a local variable. You can use these commands to do so , but first create a python file and save it in your working environment :

```
import os

os.environ["OPENAI_API_KEY"] = "your_openai_key_here"
```

Proposed Approach (architecture):



Keys	System's blocks
🟪	Code condenser
🟨	Tokens calculator
🟥	Translation block
🟨	Gigantic code files translator
🟦	Few shot learner
🟩	Decomposition block
🟩	Construction block
🟩	Cleaning block

- The code condenser is asked to delete all comments from a given file . it is worth noting that both java and c# has the same format of comments .
- Whenever you work with a file you need to calculate the number of tokens in inside of it. Approximately 1000 token is equal to 750 words.
- We have used the GPT-NEO tokenizer to calculate the number of tokens since it is free, and you can load it from the hugging face library .
- For the translation we have used the GPT-3.5 model and specifically the text-davinicii-003 model. Its context vector is equal to 4096 tokens in total ; you can send approximately 2000 token as your source text and expect the model to respond with its translations (also 2000 tokens)
- The decomposer block's goal is to decompose a given file into different blocks . it is going to extract at each time a specific block from your code, send it to the OpenAi gpt-3.5 model for translation, get the translations, saves them in intermediate files.
- The construction block is asked to construct a final file out of the created intermediate files . it follows a specific template for both java and c#.
- The cleaning block is then asked to delete all of the intermediate files after the creation of the final file is done