

## SGQ-CAG-LLM

This repository contains the pilot for a service that uses a Cache Augmented Retrieval (CAG) system to analyze student questionnaires and output a report. At the end of each semester, each student is invited to complete a questionnaire for each course. With the inclusion of an open question, it becomes imperative to use smart services to summarize the thousands of reports into an easily digestible report.

The repository contains two distinct pieces of code: i) the service and ii) the client. The service implements the previously mentioned service. The client is one possible example of how to interact with the service. It acts as a template for further integrations.

### Service

#### Description

The service was implemented in FastAPI and uses LlamaIndex as the baseline for the CAG service. It builds an in-memory Vector Index and, using it as context, questions a Large Language Model (LLM) to summarize the required information for the report.

#### Usage

For ease of deployment, the service can easily be deployed using the provided docker compose file. Simply run the following commands:

```
docker compose build
docker compose up -d
```

The service is available through port 80 and a REST API.

To stop the service, run the following commands:

```
docker compose down
```

To check the logs of the service, execute the following commands:

```
docker compose logs
```

#### API

The service only has one endpoint: POST /report.

#### POST: /report

#### Parameters

name	type	data type	description
None	required	JSON	{“course”:< str >, “year”:< int >, “observations”:[< str >]}

## Responses

http code	content-type	response
200	application/json	{“positive”:< str >, “negative”:< str >, “sentiment”: {“Positivo”: < float >, “Neutro”: < float >, “Negativo”: < float >}}

## Client

### Description

A template for a simple client that can be used to call the service. It sends a data sample (repostas\_alunos) and receives the response. It also generates a sample report.

### Usage

To use the client, first it is necessary to install the necessary requirements. It is recommended to use a virtual environment. Execute the following commands:

```
python3 -m venv venv
source venv/bin/activate
pip install -r requirements_client.txt --upgrade
```

After installing the requirements, you can execute the client. Execute the following commands:

```
python -m src.client -i data/repostas_alunos.csv
```

## TODO

1. Add an environmental variable to control the remote LLM from the service
2. Add documentation (pdoc) to the service
3. Optimize the prompts used for the LLM
4. Optimize the chunk size from the LLM

## Authors

- **Rafael Teixeira** - rgtzths
- **Mário Antunes** - mariolpantunes

## License

This project is licensed under the MIT License - see the LICENSE file for details