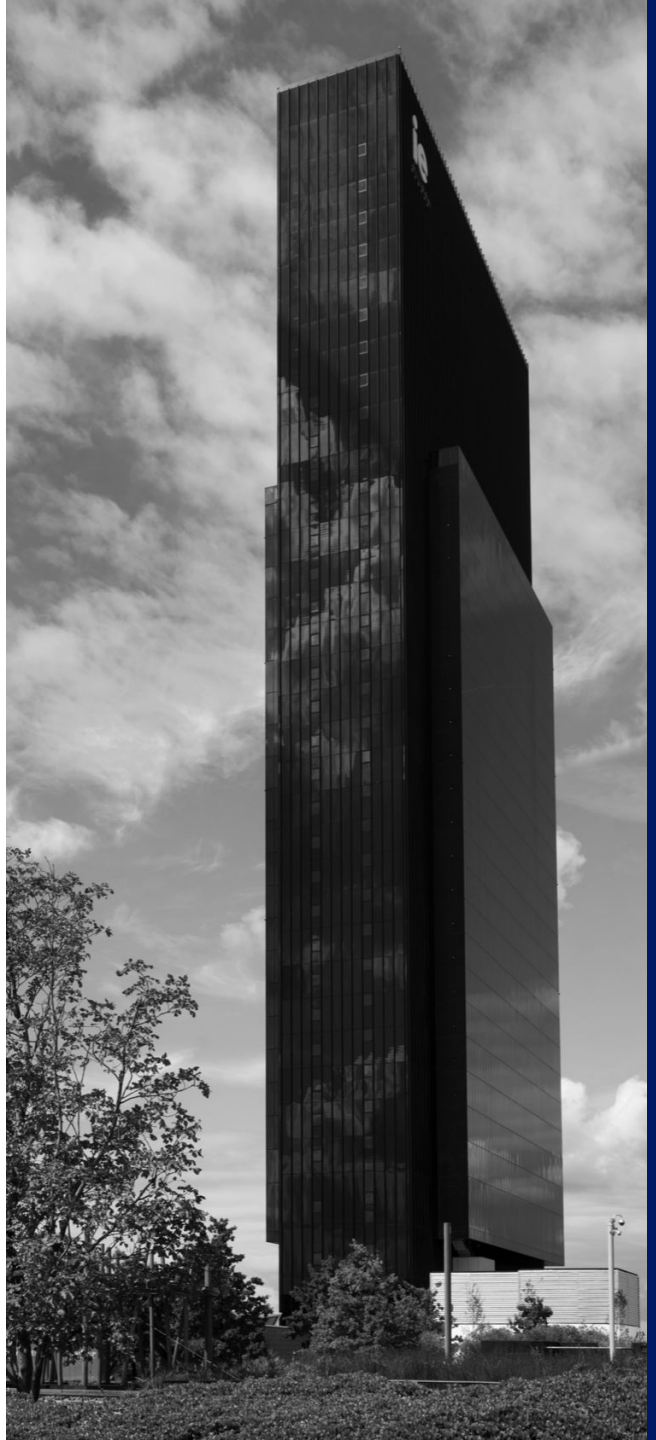


Code Compass

Final Review

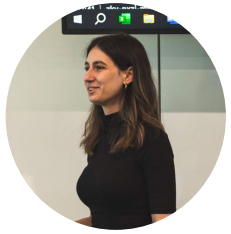
ie UNIVERSITY

Agile Team

Product Manager + Data Engineer
Gabriel de Olaguibel

Data Scientist + ML Ops
Miranda Drummond

Data Engineer + Data Scientist + ML Engineer
Luca Cuneo



Data Scientist + MLOps
Keti Sulamanidze

Data Engineer + Data Scientist
Maud Hovland

Data Scientist + ML Engineer
Félix Gómez-Guillamón

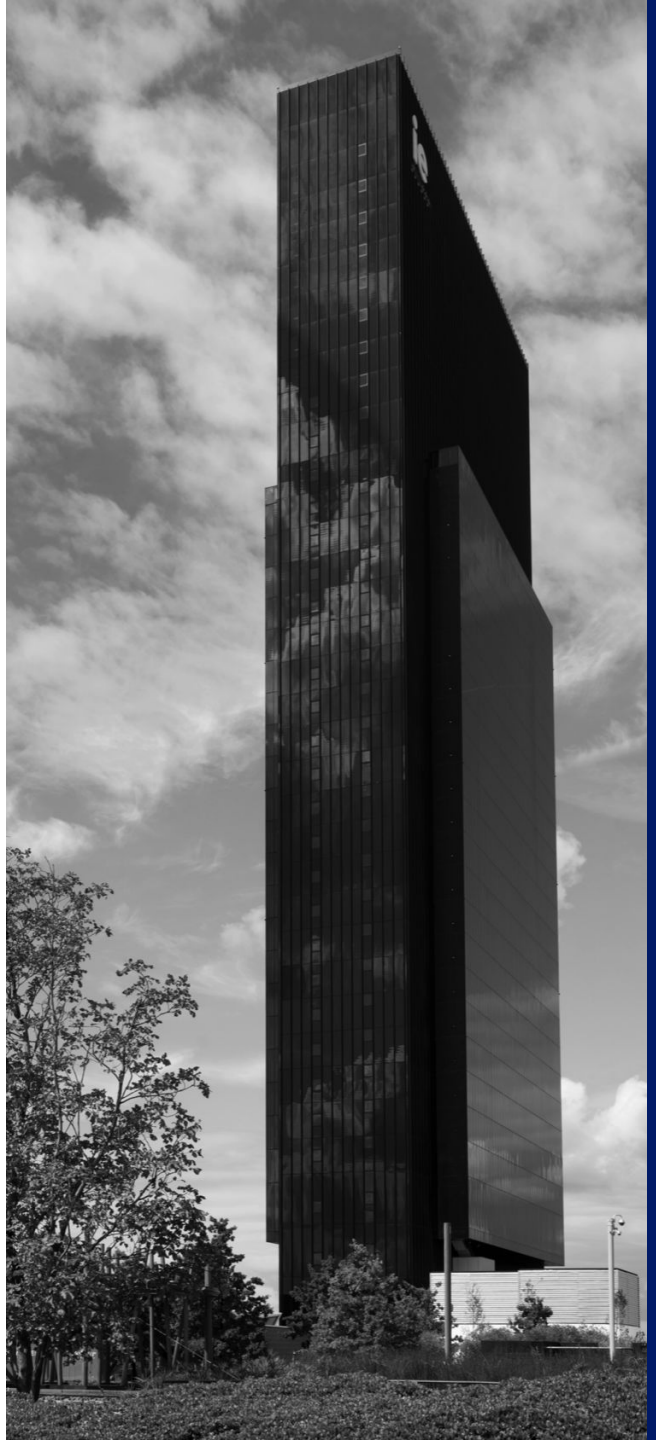
Code Compass

Navigating Developers to Personalized GitHub exploration and Insights

Agenda

- Product Goals
- Demo
- Data Engineering
- Recommender EDA
- Recommender Model
- Chatbot
- Tests/MLOps

ie UNIVERSITY



Product

ie UNIVERSITY

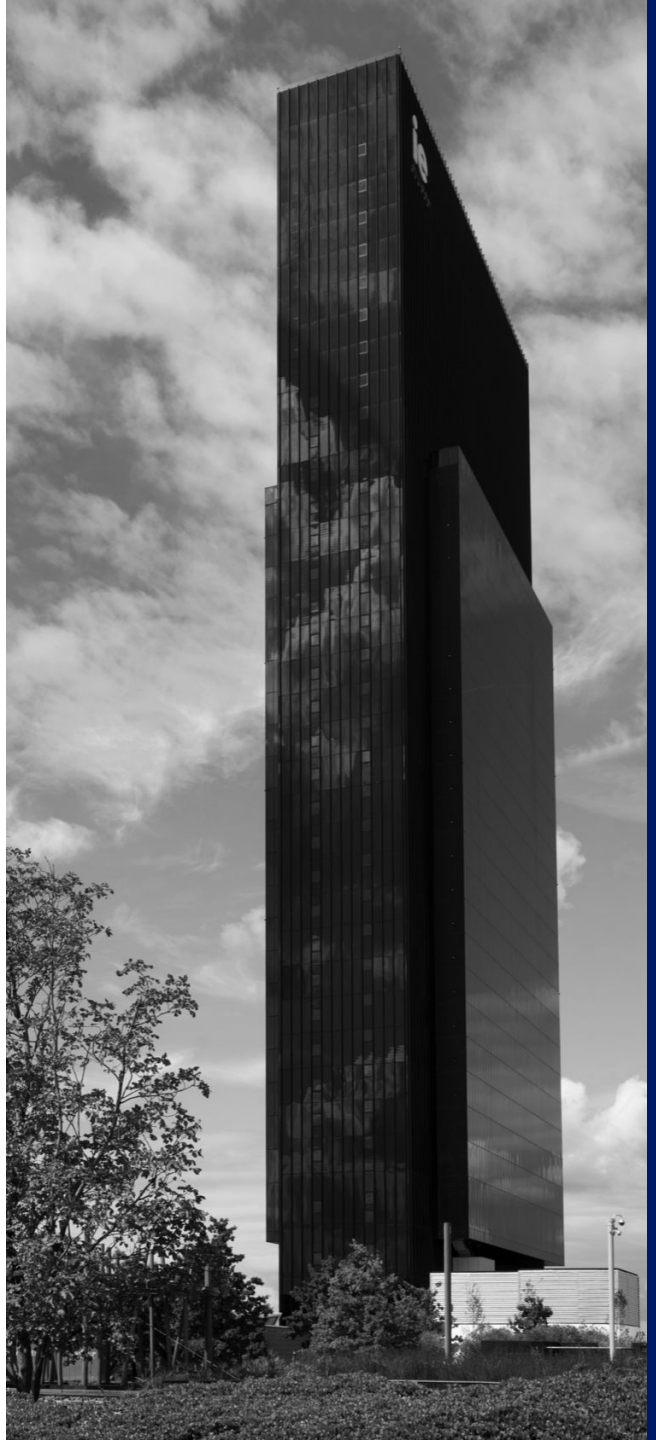
Vision

Enhance the GitHub experience for developers and learners by providing personalized experience.

Solution

*A combination of a **recommendation system** and a **chatbot**:*

- ***Personalized Recommendations**: Utilizing user-specific, Code Compass will recommend relevant GitHub projects.*
- ***Interactive Learning and Exploration**: The chatbot will serves as an interactive guide, helping users to delve deeper into repositories , understand technology applications, and navigate more effectively.*



Demo

ie UNIVERSITY

Chatbot

Github Repository Chatbot

Exit Chat

Tell me about the following repository: https://github.com/recommenders-team/recommenders

The repository "recommenders" contains a wide range of files and directories. Some key files and directories include:

- Main Files:
  1. README.md
  2. LICENSE
  3. CODE_OF_CONDUCT.md
  4. CONTRIBUTING.md
  5. SETUP.md
  6. SECURITY.md
  7. AUTHORS.md
  8. MANIFEST.in

Recommender

GitHub Repo Recommendation System

Enter the target user's username:

miguelgfierro

Get Recommendations

Recommendations

1 - Repo ID: 537923335.0, Owner: khizarsiddiqui, Link

Description: A Python program that generates ASCII art from graphical images using Pillow, a fork of the Python Imaging Library (PIL) and numpy.

2 - Repo ID: 403642804.0, Owner: shazron, Link

Description: Adobe I/O Photoshop API SDK

3 - Repo ID: 251512047.0, Owner: Ewenwan, Link

Description: MindSpore is a new open source deep learning training/inference framework that could be used for mobile, edge and cloud scenarios.

4 - Repo ID: 317381518.0, Owner: noahgift, Link

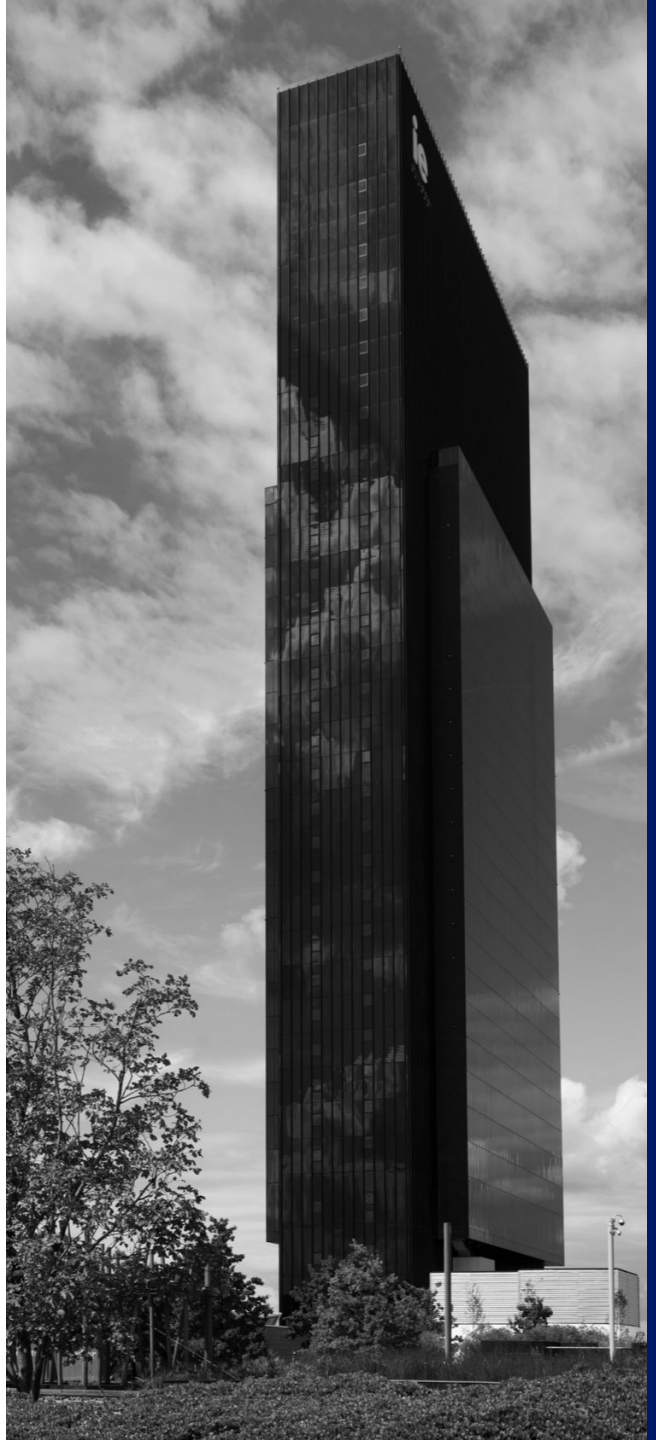
Description: Some recipes for doing with serverless technologies

5 - Repo ID: 604684135.0, Owner: giswqs, Link

Description: A Python package demo for interactive mapping

6 - Repo ID: 10714550.0, Owner: petrounias, Link

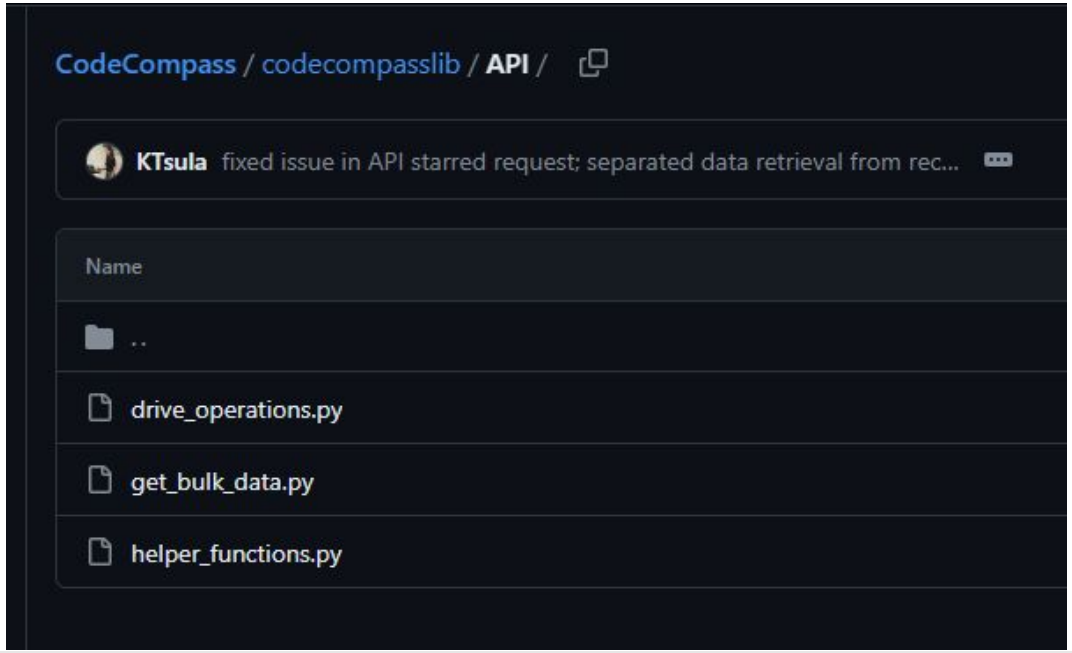
Description: Programmatic building of JSON schemas (document and field mappings) with validation.

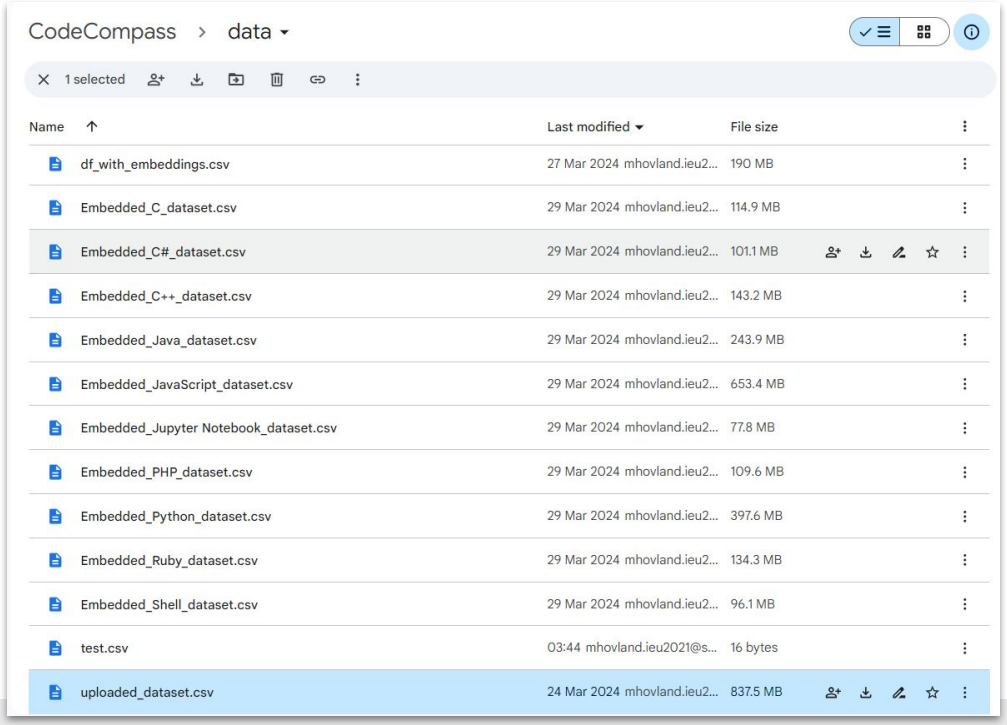


Data Engineering

ie UNIVERSITY

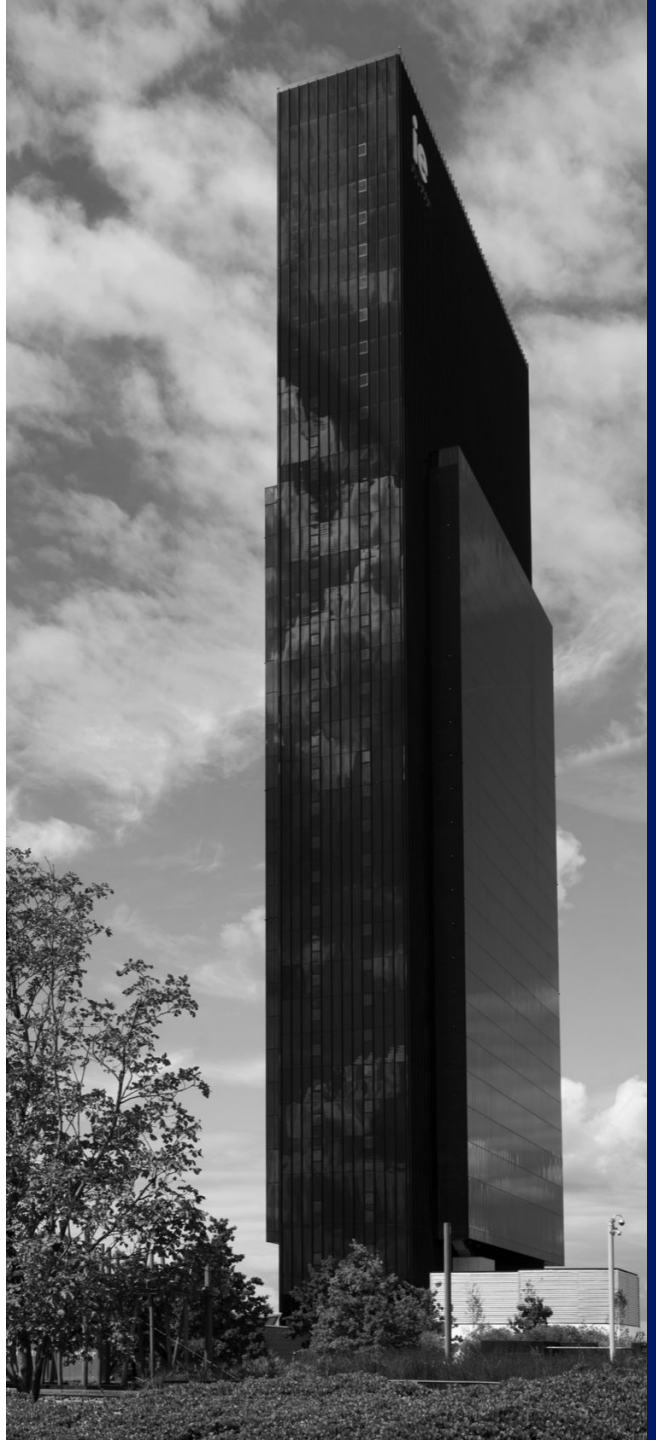
Data Engineering





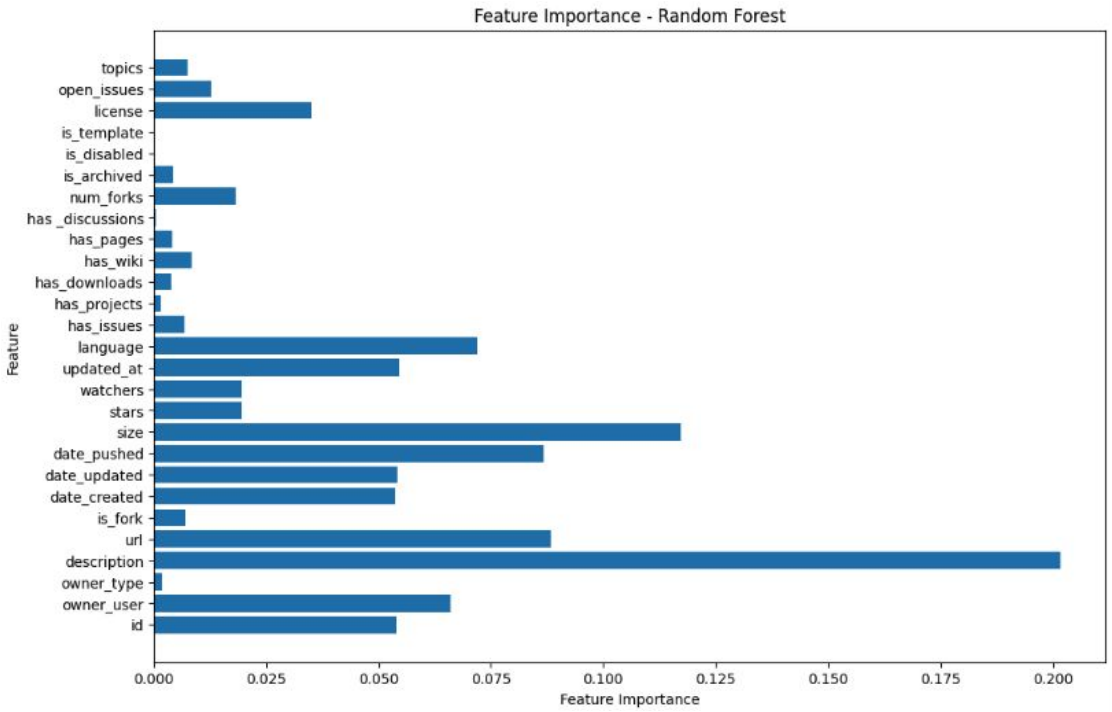
- *Utilizing Github API to retrieve repository and profile information*
- *All code files contained within codecompass/API folder*
  - *get_bulk_data.py:*
    - *Contains the logic for building our dataset (this has been improved since last review)*
  - *helper_functions.py:*
    - *Contains functions to load secrets, or extract useful information from repositories*
  - *drive_operations.py*
    - *Contains the functions to upload our data to our google drive.*

ie UNIVERSITY



Recommender EDA

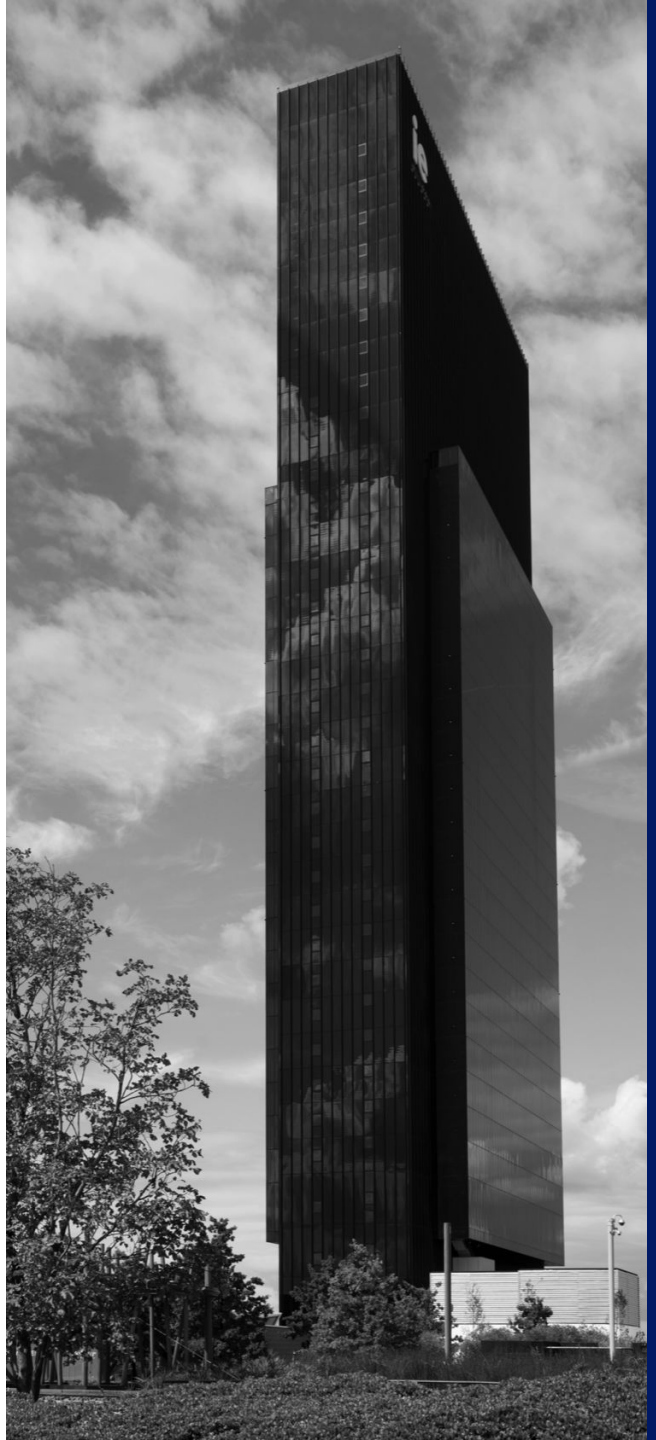
EDA / Feature Engineering



EDA and Feature Engineering:

- ***Data Preparation**: Loads and cleans data, removing unnecessary columns and handling missing values.*
- ***Visualization and Correlation**: Utilizes plots to visualize data and performs correlation analysis to identify key relationships.*
- ***Encoding and Vectorization**: Applies one-hot encoding to categorical data and uses TfidfVectorizer for textual data to prepare it for machine learning.*
- ***Normalization**: Scales numerical features to a uniform range using techniques like MinMaxScaler.*
- ***Data Preparation for ML**: Converts processed data into tensors, making it suitable for training with PyTorch models.*

ie UNIVERSITY



Recommender Model

ie UNIVERSITY

Model Embeddings

OpenAI

```python
def generate_openAI_embeddings(strings_to_embed, client):
    """
    Generates OpenAI embeddings for the given strings using the specified OpenA
    Args:
        strings_to_embed (list): A list of strings to generate embeddings for.
        client: The OpenAI client object used to make API requests.
    Returns:
        dict: A dictionary containing the embeddings generated by OpenAI.
    Raises:
        OpenAIException: If there is an error while making the API request.
    """
    response = client.embeddings.create(
        input=strings_to_embed,
        model="text-embedding-3-large",  # You can choose the model you prefer
        dimensions=256  # You can choose the number of dimensions you prefer
    )
    return response
```

Word2Vec model trained in SWE domain

```python
word_vect = KeyedVectors.load_word2vec_format("./codecompasslib/PretrainedModels/SO_vectors_200.bin", binary=True)
return word_vect
```

Codellama locally

```python
def generate_codellama_embeddings(text):
    """
    Generates embeddings for the given text using the OllamaEmbeddings model.

    Args:
        text (str): The input text for which embeddings need to be generated.

    Returns:
        query_result (list): A list of embeddings for the input text.
    """
    embeddings_model = OllamaEmbeddings(model='codellama:7b', device='gpu') #
    query_result = embeddings_model.embed_query(text)
    return query_result
```

Sentence transformer | roberta-base

```python
def generate_sentence_transformer_embeddings(text):
    """
    Generates Sentence Transformer embeddings for the given text.

    Parameters:
    text (str): The input text to generate embeddings for.

    Returns:
    embedding (numpy.ndarray): The Sentence Transformer embeddings for
    """
    # Load a pre-trained Sentence Transformer model
    model_name = 'stsb-roberta-base'
    model = SentenceTransformer(model_name)
    embedding = model.encode(text)
    return embedding
```

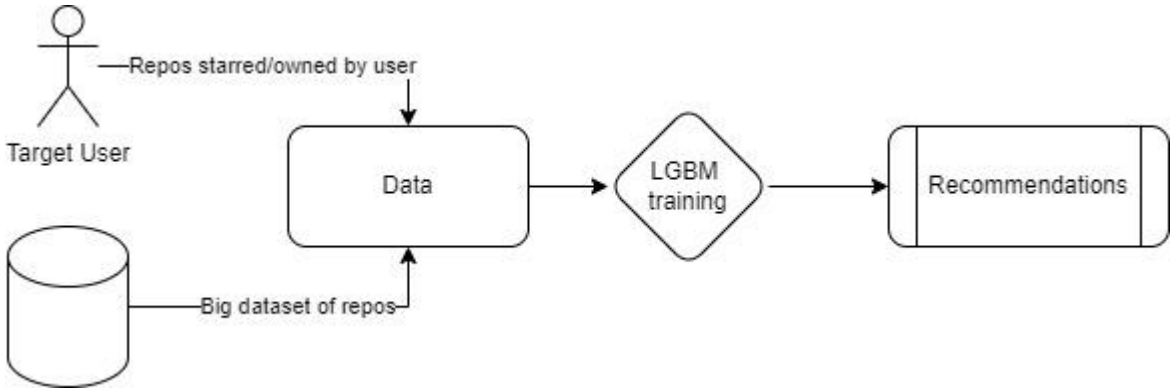
- Word2Vec - the fastest | not advanced enough.
- Roberta-base - Slightly more advanced sentence transformer | quite slow and space consuming
- Codellama - Much more advanced and applicable | very slow and space consuming
- OpenAI text embeddings 3-large - Advanced, applicable, adjustable is size | not done locally

ie UNIVERSITY

Final Model

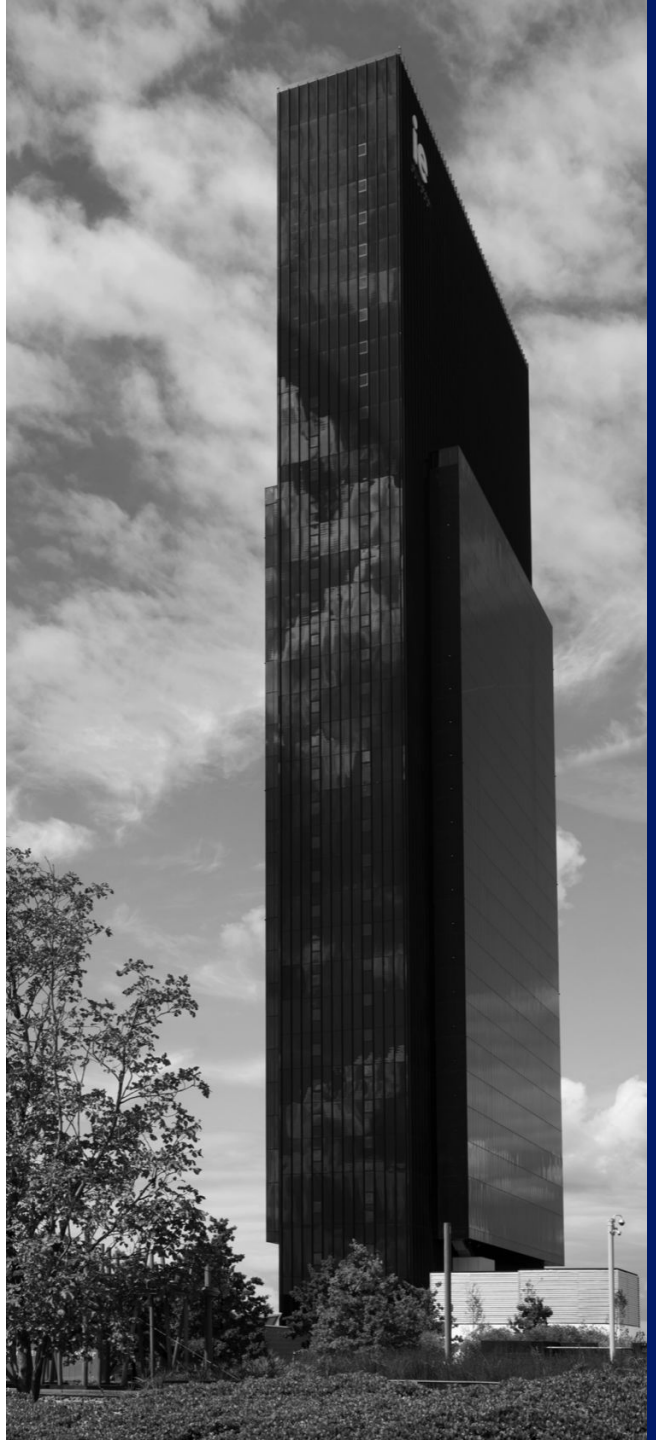
LightGBM- why?

- Efficiency and Scalability
- Gradient Boosting Strengths
- Categorical Feature Handling



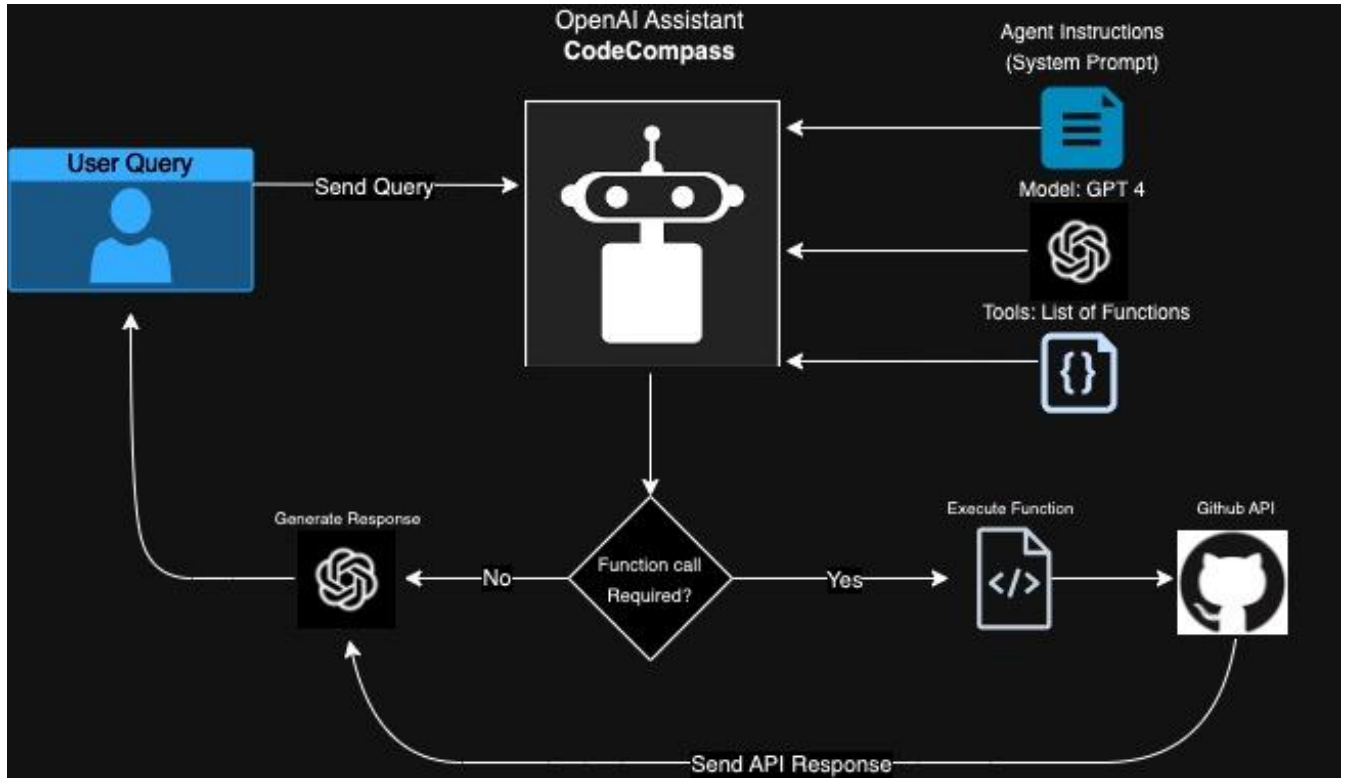
Models Considered

- Cosine similarity - assigns similarity scores to user language choices
- Differentiating repos (couldn’t find related papers suggested by others)
- LSTUR (no timestamps for user interaction data) -> could be considered for future.
- RAG - could be used to fill in missing descriptions in repos



Chatbot Architecture

Chatbot Architecture



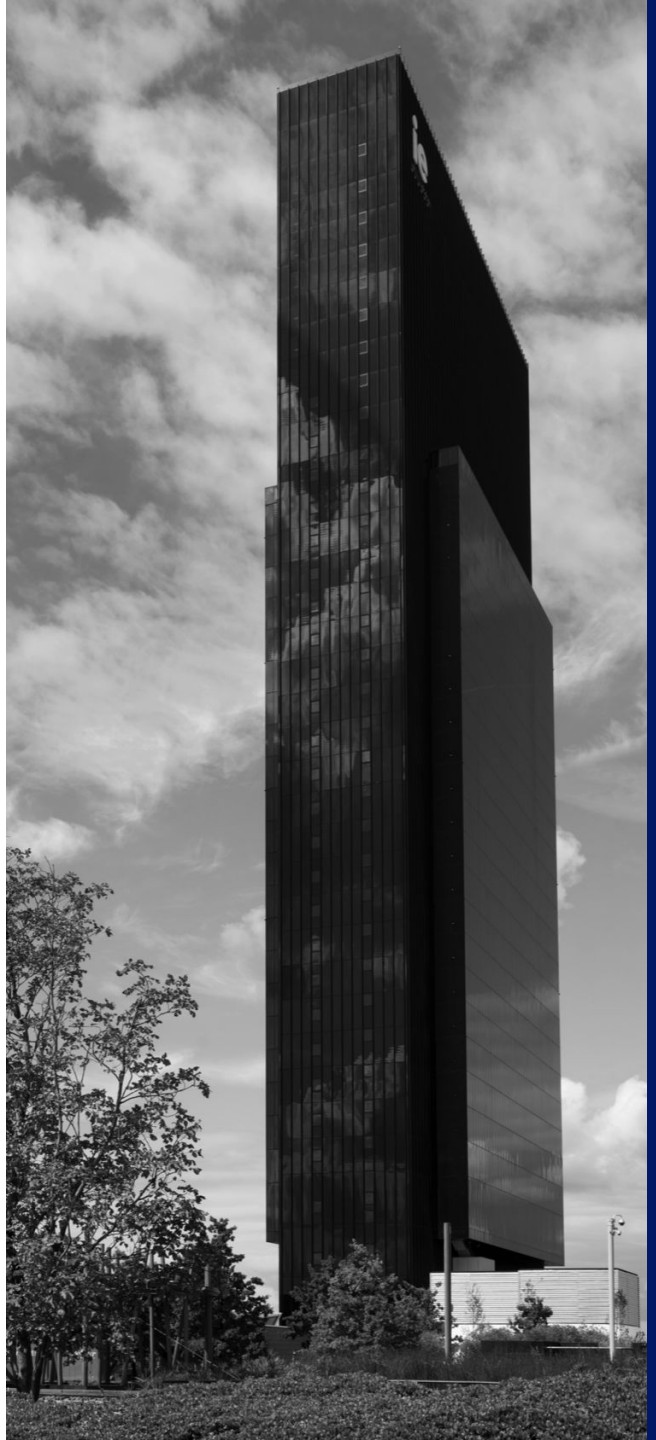
Based on OpenAI Assistants (Beta)

- Performs Function calling to External API's when Repository Information is needed
- Function Calling Capabilities:
  - Fetch repo file tree
  - Fetch Contents of specific files
  - Fetch branches in the repo
  - Fetch commit history
  - Search by keyword
  - Search commits by keyword
  - Find repositories by keyword

Development Iterations

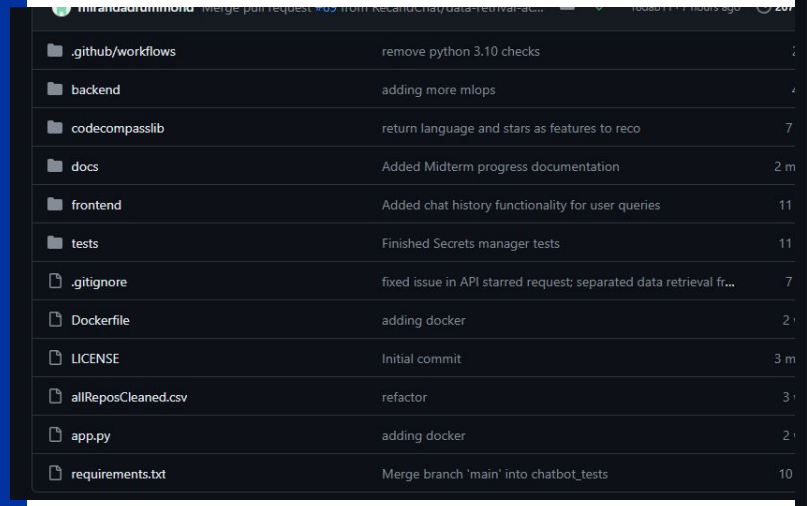
- Vanilla OpenAI API + "openplugin"
- OpenAI Chat Completion
- OpenAI Assistant (Final)

ie UNIVERSITY

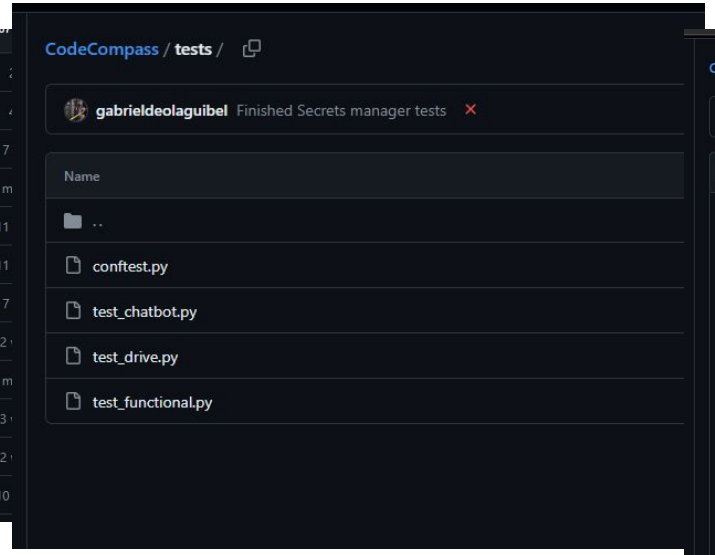


MLOps/Tests

Testing/Workflows



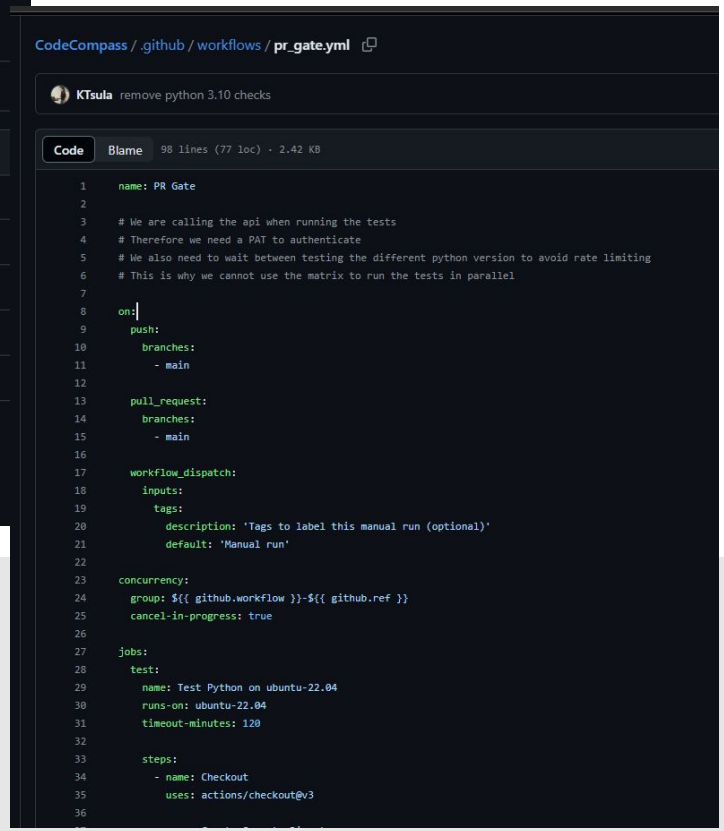
.github/workflows	remove python 3.10 checks	2
backend	adding more mlops	4
codecompasslib	return language and stars as features to reco	7
docs	Added Midterm progress documentation	2 m
frontend	Added chat history functionality for user queries	11
tests	Finished Secrets manager tests	11
.gitignore	fixed issue in API starred request; separated data retrieval fr...	7
Dockerfile	adding docker	2
LICENSE	Initial commit	3 m
allReposCleaned.csv	refactor	3
app.py	adding docker	2
requirements.txt	Merge branch 'main' into chatbot_tests	10



CodeCompass / tests /

gabrieldeolaguibel Finished Secrets manager tests

Name
..
conftest.py
test_chatbot.py
test_drive.py
test_functional.py



CodeCompass / .github / workflows / pr_gate.yml

KTsula remove python 3.10 checks

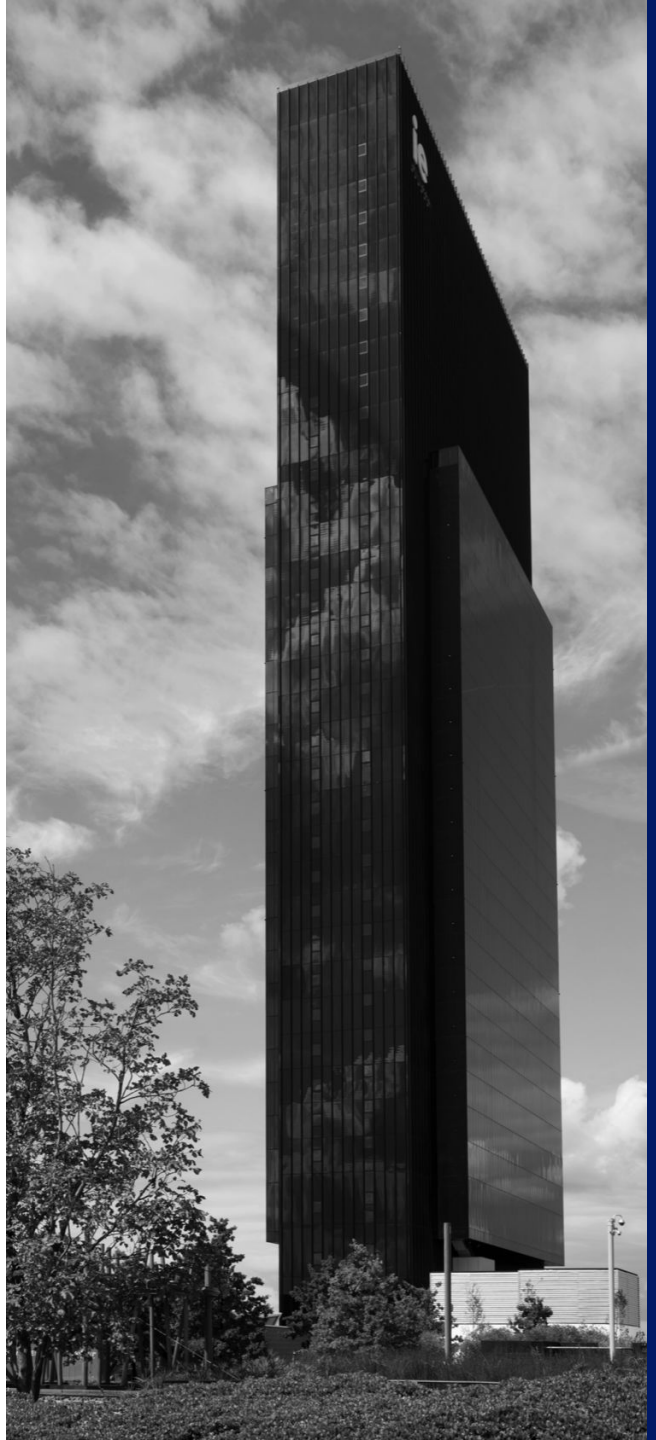
Code Blame 98 lines (77 loc) · 2.42 KB

```yaml
name: PR Gate

# We are calling the api when running the tests
# Therefore we need a PAT to authenticate
# We also need to wait between testing the different python version to avoid rate limiting
# This is why we cannot use the matrix to run the tests in parallel

on:
  push:
    branches:
      - main

  pull_request:
    branches:
      - main

  workflow_dispatch:
    inputs:
      tags:
        description: 'Tags to label this manual run (optional)'
        default: 'Manual run'

concurrency:
  group: ${{ github.workflow }}-${{ github.ref }}
  cancel-in-progress: true

jobs:
  test:
    name: Test Python on ubuntu-22.04
    runs-on: ubuntu-22.04
    timeout-minutes: 120

    steps:
      - name: Checkout
        uses: actions/checkout@v3
```

Tests can be found on the Tests folder at the root of the project, which are then executed by the workflow when making a PR to merge to main.
We have three main test files:

- *test_chatbot.py*
- *test_drive.py*
- *test_functional.py*

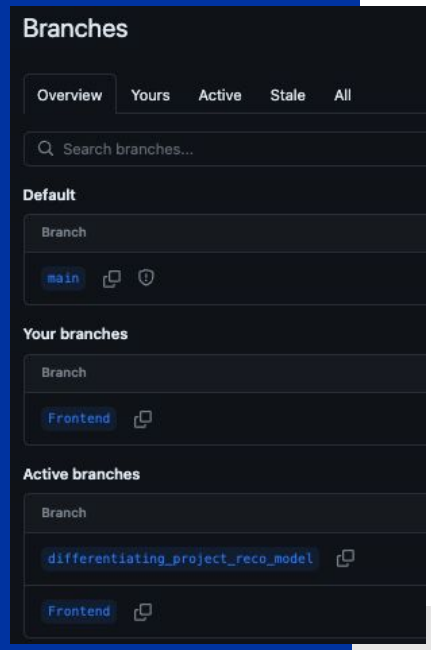
ie UNIVERSITY

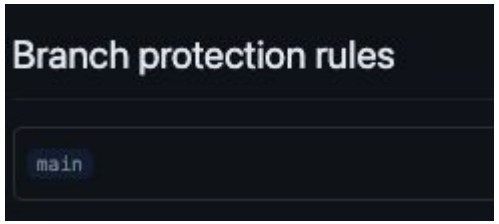


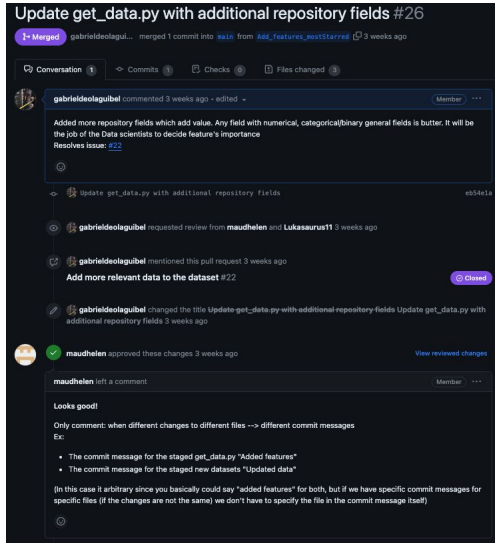
Development Methodology

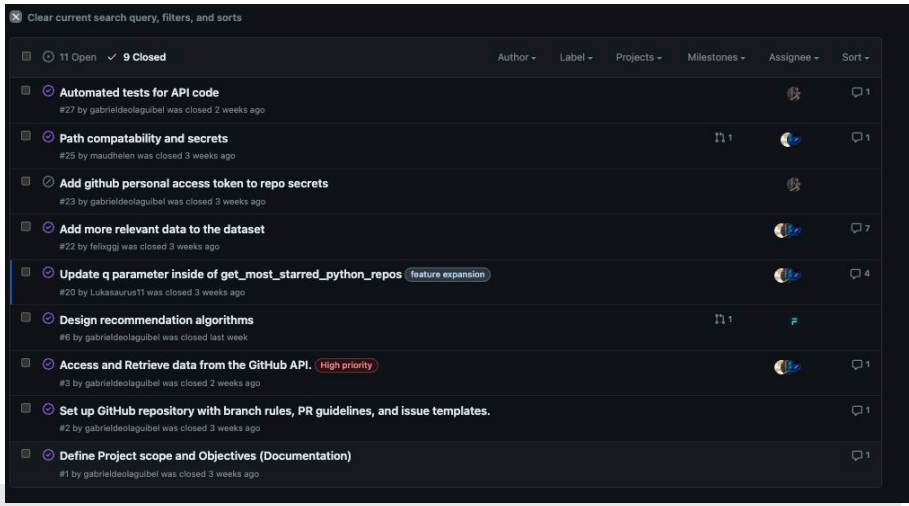
Development Methodology

Good Coding Practices









Required development in short-lived branches:

- *No Member is allowed to push to main*
- *Branches are deleted after merges to main*

Add code via PRs:

- *Requires another member to review and comment.*
- *Requires automated workflow tests to pass*

Issues:

- *Issues are part of the github project backlog*
- *Members submit issue with what need to be fixed/done and get feedback from others on the ideas*
- *PRs directly address issues*

Libraries and Repository design:

- *Notebooks and scripts call on codecompasslib library or external libraries*
- *Code is neatly organized in its respective folders following “Project template” design by Miguel.*

ie UNIVERSITY

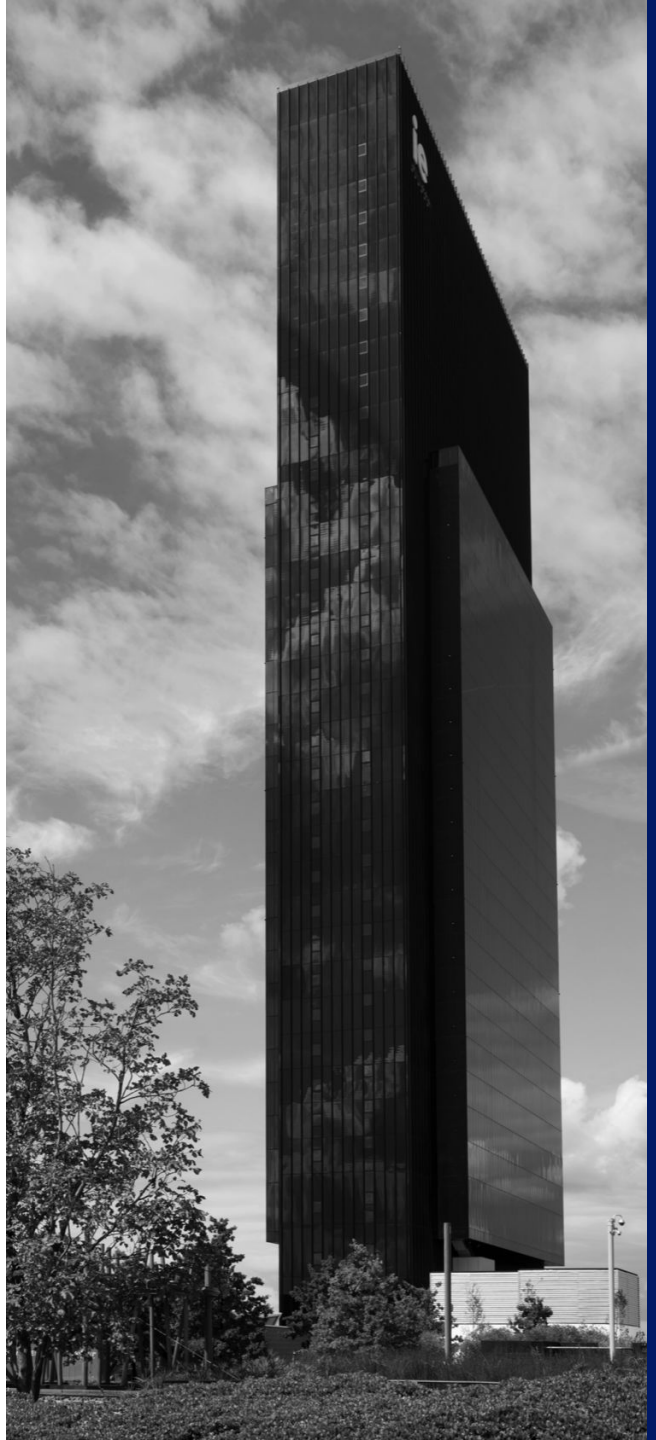
Further Improvements

To be Implemented by Sunday April 7

- **Full Code Documentation**: Explanation/Math behind the algos, Demo videos/gifs
- **Finalize Library (codecompasslib)**: Perform release and move a few files around/cleanup
- **Further Tests**: Improve code coverage for (Reco Model functions)

Longer term Improvements

- **Implement different Reco Models:**
  - Collaborative filtering based models;
  - Time considerate models (ex. LSTUR to capture short term and long term representations)
- **Improve chabot functionality:** Implement more repository data retrieval
- **Cloud Deployment**: Host fontends on Cloud
- **Improve Datastore:** Try Vector Databases to store embeddings for faster Data Retrieval
- **Should work for any user:** Implement function that embeds and adds new user's repos to dataset automatically



Feedback / Questions

ie UNIVERSITY