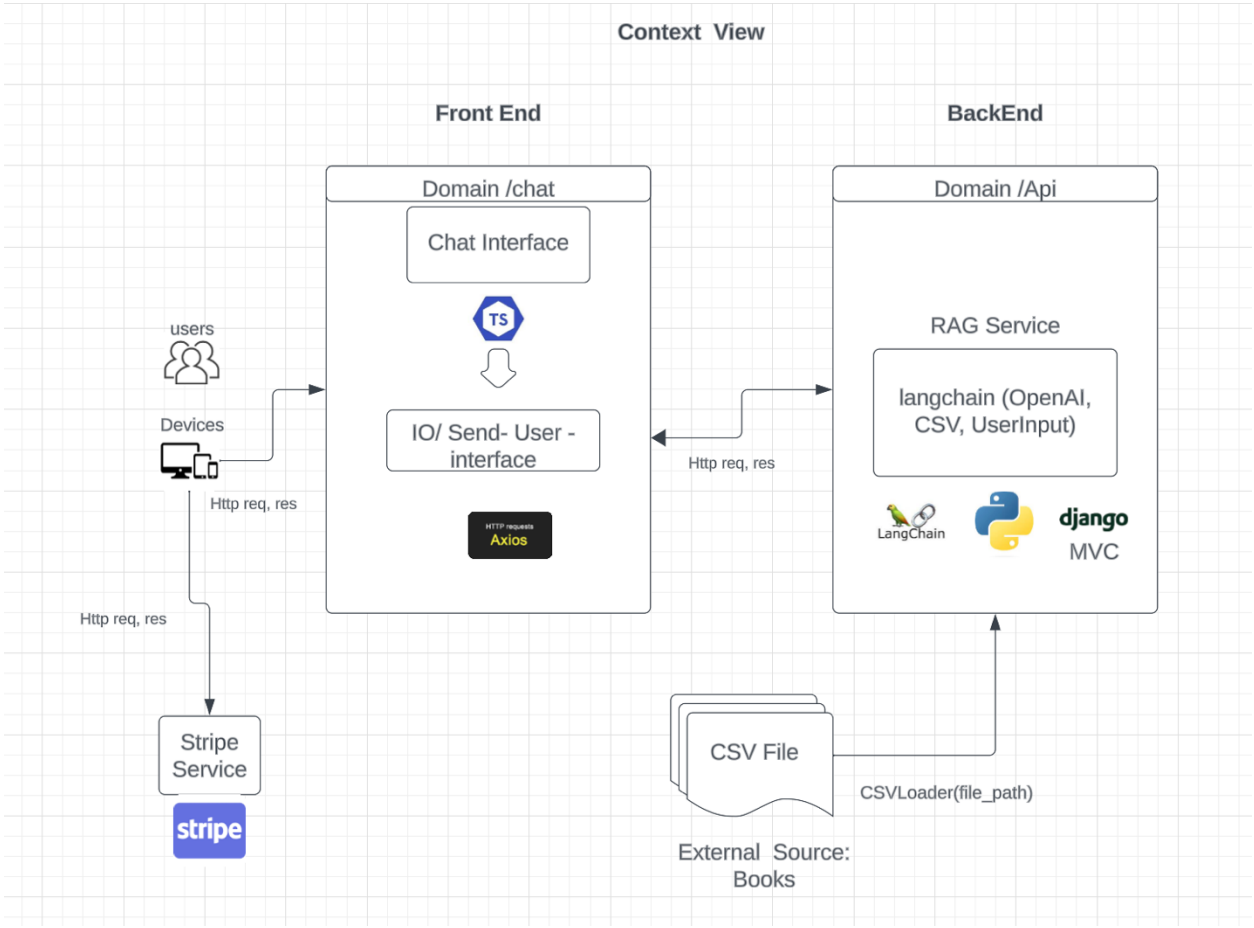
Architectural view

Chat Bot with LLMs and RAG for enhancing responses in an online bookstore



This Architecture has three main components: the back-end, front-end and third party integration.

This solution aims to enhance the user experience when customers buy books in an online store. Using a chatbot interface, the system displays a prompt asking the user questions related to books. Once the interface receives an answer, it is sent to the backend, which is built in Python. The system processes the user message using a RAG service (Langchain). This service takes into account an external CSV file to personalize the response according to the specific context. For the backend operation, the system uses the MVC pattern, and to support this process, we utilize the Django Framework. The communication between the back-end and front-end is facilitated by Axios, which provides a framework for HTTP connections. Finally, when the front-end receives the response, it is displayed in the chatbot and is also parsed to list the book, price, rating, and description. This enables the user to pay through the checkout button.

The charBot use the following technologies to support its operation:

Front-end:

Axios: Axios is a JavaScript library for making HTTP requests in browsers and Node.js. It simplifies asynchronous data fetching, supporting various HTTP methods, request/response interceptors, and automatic JSON data transformation. With a concise and promise-based API, Axios enhances code readability and facilitates error handling. It is commonly used in web development, interfacing with APIs in frameworks like React or Vue.js, and server-side environments such as Node.js or Python.

Back-end:

LangChain

LangChain is a framework for developing applications powered by language models. It enables applications that:
-Are context-aware: **c**onnect a language model to sources of context (prompt instructions, external sources of data, etc.)
-Reason: rely on a language model to reason (about how to answer based on provided context, what actions to take, etc.)

Django
Django is a high-level Python web framework designed for rapid development and clean, maintainable design. It follows the Model-View-Controller (MVC) architectural pattern and includes an Object-Relational Mapping (ORM) system for database management. Django simplifies tasks such as URL routing, form handling, and database migrations, promoting efficient and scalable web application development. It is widely used for building robust and secure web applications, providing built-in features for authentication, admin interfaces, and template engines.

Retrieval-Augmented Generation (RAG)
is a natural language processing model that integrates information retrieval with text generation. It combines a retriever module to extract relevant information from a large dataset or knowledge base, and a generator module to create coherent and contextually relevant responses. RAG is often employed in question-answering systems and conversational AI applications, enhancing the model's ability to provide accurate and context-aware responses by leveraging information retrieval before generating a reply. This approach improves the overall performance and knowledge incorporation of language models.

Third parties:

This solution uses stripe to simulate the payment gateway. It is displayed after the chatbot recommends a book to the user.

On the other hand, the LLM uses an external CSV to enhance the responses. The source of this file is: https://www.kaggle.com/datasets/jalota/books-dataset

This is the description of the dataset:

Title: The title of the book.

Category: category of the book.

Price: the price of the book.

Price After tax: the cost of the books including tax.

Tax amount: tax on the book.

Availability: the quantity available in the stock.

Number of reviews: number of people who reviewed the book.

Book Description: description of the book.

Image Link: link where you can see the image of the book.

Stars: star rating for each book out of 5.