User / Tech Manual for CTU Chatbot

Download Python
The official home of the Python Programming
https://www.python.org/downloads/

Download Ollama on macOS
Download Ollama for macOS
https://ollama.com/download

One-Stop GenAI Stack to Build Production-Ready Apps | DataStax
Go from app idea to production with 20% higher relevance and 74x faster response on the industry-leading vector database, Astra DB. Get started for free!
https://www.datastax.com/

Install Langflow | Langflow Documentation
You can deploy Langflow either locally or as a hosted service with Datastax Langflow.
https://docs.langflow.org/get-started-installation

Chatbot

As part of an internship collaborating with the Capitol Technology University AICE Lab, we developed a chatbot to help students and visitors get answers to User questions.

To accomplish this task, a **retrieval-augmented generation** (RAG) model was created. A RAG combines information retrieval from a knowledge base with tailored responses, ensuring that answers are accurate and appropriate to the given context.

Here is a step-by-step guide to utilize and incorporate a RAG chatbot:

System Requirements

Table of Contents

- Chatbot
  - System Requirements
  - Technical Architecture
    - **Data Collection**
    - Text Splitting
    - Embedding
    - Testing and Tuning
  - Installation and Setup for Developers
    - Python
    - Data
    - Ollama
    - AstraDB / Faiss

- User Requirements:
  - Access to web browser/chatbot interface

Langflow Setup for Users

Technical Architecture

- RAG Workflow: Query → Embedding → Vector Store Retrieval → Response

- Components:
  - Data Source: CTU website data and CTU student handbook
  - Embedding Model: Ollama 3.2
  - Vector Store: Astra DB
  - LLM: Olamma

Data Collection

- Data Sources
  - Documents
    - 2024-2025 Student Handbook
    - 2024-2025 Clubs and Orgs Guide
    - Student guide to myCapitol
    - 2022-2023 Course Catalog
  - Websites (including sub-pages)
    - mycapitol.captechu.edu
    - captechu.edu

- Website Data

    - To obtain the website data, I used LangChain's WebBaseLoader to scrape from 18 Capitol Technology public sites with abundant text and modest formats. To clean the scraped data, I removed noisy headers and footers from the HTML and unnecessary white spaces around the data.

- Document Data
    - Every document was transformed into a Txt file and had unneeded whitespace removed.

Text Splitting

- A crucial step in the RAG process is text-splitting. Language models have strict limits on the number of tokens that can be processed at once. Splitting text into small, but meaningful; context-rich segments called chunks. This allows a language model to stay within its token limit while improving it's ability to locate specific and relevant information quickly.

- Text Splitter of Choice: Recursive Character Splitting is the current text-splitter method for all of the project's data. This method's primary objective is to retain contextual significance by keeping paragraphs, sentences, and words together, all with the goal of keeping related information in close proximity.

Embedding

- Embeddings: Machine learning models need data in a format that they can understand and interact

with. Embedding transforms text data into numerical data that the models can work with. The numerical data is a vector representation of text.

- Vector Store: Vectors must be stored in a database called a vector store. This allows speedy retrieval of necessary data to answer an asked question.
    - AstraDB is the vector store of choice because of its ease of use.

- Model: The embedding model being used for the project is Meta's Ollama 3.2 3B parameter model. Ollama 3.2 is a highly efficient model for its size that works exceptionally well with RAG data. Its performance is extremely close to Ollama 3.1 of 8B parameters, which is very impressive.

Testing and Tuning

- Testing: Ask the model common probing questions that a student may ask like, "What is the student life like at the school?" The answer can vary between the model not being able to retrieve adequate data to answer the query or giving an okay answer with an additional suggestion to search for the answer at a specific CapTechU office.

- Tuning
    - Adjusting chunk size
    - Adjusting chunk overlap
    - Adjusting retrieval parameters
    - Adjusting prompt

Installation and Setup for Developers

Python

- Install Python:
  - Download Python 3.8+ from the official Python website.
  - Ensure Python is added to your system's PATH during installation.
- Access through the command line or desired IDE like Visual Studio

Data

Download packaged University data files if available

If not available:

- Website
  - Install Langchain and LangChain's beautifulsoup4 (https://python.langchain.com/v0.1/docs/use_cases/web_scrapin

    ```
    "pip install -q langchain-openai lar
    ```

  - Iterate through CTU data-abundant websites and list those websites to scrape using LangChain's dependencies
  - Store data in a folder for input into a LangFlow directory

- Documents

- Utilize Langflow's directory component to input required PDF or text files.

Text Splitter

- Link LangFlow's Text-Splitter component to inputted directories to split data into usable chunks for the embedding model.

Ollama

Install Ollama model for embedding and generation

1. Download and Install Ollama from (https://ollama.com/download)
2. Start Ollama
   a. From executable
   b. Or from the command line with "ollama start"
3. Initial Ollama version 3.2 with "ollama run llama3.2"

AstraDB / Faiss

Set up AstraDB or FAISS Vector Store for data storage and management

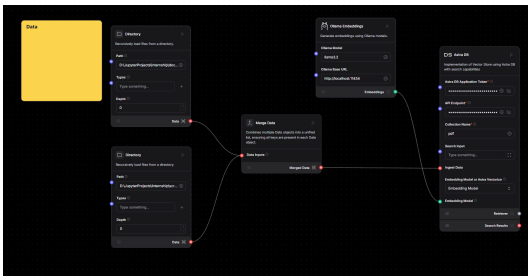
1. Create an AstraDB Account:
   a. Signup at (https://www.datastax.com/)
2. Create a New Database
   a. Store generated endpoint
   b. Store generated application token

1. Add FAISS component
2. Create and link an available directory path to store vectors

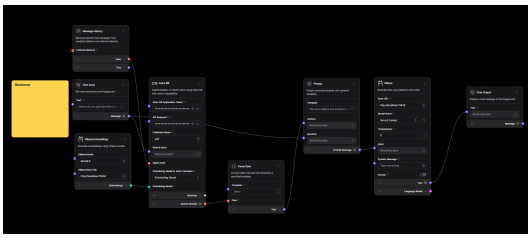
Langflow

Setup Langflow for easier visual workflow

1. Create a Virtual Environment to make python library implementation simpler
    a. In cmd:
        i. mkdir “Name”
        ii. cd “Name”
        iii. “python -m venv venv”
        iv. venv\Scripts\activate
2. Install LangFlow
    a. Install LangFlow’s Python package in IDE or command line: “pip install langflow”
3. Start LangFlow
    a. “langflow run”
    b. or “python -m langflow run”
4. Access Langflow
    a. Open the browser and visit (http://localhost:7860) for UI access
5. Import given workflow as a JSON file
    a. If not available:
        i. Load data by connecting the following nodes:
            1. Data Workflow
                a. Input personal AstraDB information and file Paths



2. Retriever and Generation Workflow



Setup for Users

1. Open chatbot interface.
2. Enter desired query, such as “What does CTU specialize in?”
3. Receive an answer relevant to the CTU data fed to the model.