Project name: Customize a GPT model to build a chatbot for California Science and Technology University

Lam Dao, Fang Wang, California Science and Technology University

Feb, 24, 2024

Abstract:

This project aims to develop an GenAI-based advanced chatbot for the California Science and Technology University (CSTU) using Retrieval Augmented Generation (RAG) with a GPT model. The goal is to offer a responsive and efficient interface for students, faculty, and staff, simplifying access to information and automating tasks such as course registration and grade inquiries. By leveraging technologies like GPT, RAG, and vector embedding/database, the chatbot seeks to improve user experience and streamline routine processes. The project is divided into three milestones, each progressively defining use cases, implementing the technical approach, and delivering a functional prototype, along with a comprehensive report.

1 USE CASES

1.1 GENERAL USE CASES

Key Users:

- Students: Prospective, current, and graduating students
- Faculty: Professors, researchers
- Administrators: College staff across departments

1.1.1 Student Use Cases

- **General Inquiries**

- Answering frequently asked questions about the college application process, admissions criteria, and required documents.
- Providing details about offered programs, courses, syllabus structure, and options.
- Educating on campus life, including accommodation, societies, and clubs.

- **Admissions Support**

- Guiding students through the college application process.

- Checking application status and providing required next steps.
- Assisting with application documentation.
- Recommending opportunities suited to student interests and strengths.

- **Registration & Scheduling**

- Adding/dropping courses after verifying prerequisites.
- Building optimized course schedules.
- Facilitating easy course registration through a conversational flow.
- Providing waitlist notifications and auto-enrolling if seats open.

1.1.2 Faculty Use Cases

Proving following information and knowledge:

- Provide training programs.
- Provide syllabus and related documents for courses enquiry.
- Provide grade records for each course.
- Provide other academic guidance and recommendations.

1.1.3 Administrator Use Cases

- **Institutional Knowledge Base**

- Maintaining a central data repository on university details.
- Ensuring continuous updates as new data like policies are added.
- Training GenAI model on existing documents and data assets.
- Expanding knowledge via public/licensed datasets.

- **Student Support Automation**

- Handling routine inquiries across departments.
- Integrating with on-premise systems like the registrar database.
- Providing 24/7 availability chatbot agents.

- **Analytics and Reporting**

- Tracking usage metrics like user satisfaction.
- Analyzing trends in student queries and challenges.
- Identifying areas needing additional assistance.
- Monitoring for malicious activities and policy violations.

This session outlines the general use cases for the project, ensuring comprehensive coverage of functionalities for students, faculty, and administrators.

1.2 DETAIL USE CASES / PROBLEM SCOPE

Based on general use cases, we choose and decompose some typical and detailed use cases to define problems to be solved in the next phase of the project.

1.2.1 Building Knowledge Base for CSTU Chatbot

Scenario: This script builds and uploads a knowledge base (KB) for the CSTU chatbot into a Vector database. The KB is extracted from a PDF file ("cstugpt_kb.pdf") and processed into word embeddings using Word2Vec. These embeddings are then inserted into the database along with associated metadata for faster searching and retrieval.

Actors:

- **System:** The Python script using Pinecone and Word2Vec libraries.
- **Knowledge Base File:** The PDF file containing text information for the chatbot's knowledge base.

Processing Steps:

1. **Check and Create Index:**
    - Check if the target exists.
    - If not, create the index with appropriate dimensions and metrics.
2. **Train Word2Vec Model:**
    - Extract text from the KB file and perform text cleaning/preprocessing.
    - Train a Word2Vec model on the extracted text tokens.
    - Save the trained model for future use.
3. **Iterate and Upsert Embeddings:**
    - Divide the KB text into chunks.
    - For each chunk:
        - Generate word embeddings for the chunk text using the trained model.
        - Create metadata containing a unique identifier and the chunk text.
        - Upsert the embedding and metadata to the Pinecone index with a specific namespace.
        - Keep track of the total number of upserted vectors.
4. **Delete Existing Knowledge Base:**
    - Delete all data associated with the namespace in the database index.
5. **Verify Upsert and Index Stats:**
    - Get details about the Pinecone index, including the number of vectors and other stats.

Expected Outcome:

- A successfully built and upserted knowledge base in the database index, accessible by the CSTU chatbot.

- The index should contain word embeddings and associated metadata for each chunk of text from the pdf file.
- Stats confirming the number of upserted vectors and overall index health.

1.2.2 Upload/Update Course Grades:

Scenario: Users want to update their course grades.

Actor: Professor

Processing steps:

  - Request the user's secret code for verification.
  - Call the "update_grades" function with the provided code.
  - Proving grades data.
  - Confirm and feedback.

Expected Outcome:

- Users are requested to provide their secret code for verification.
- If the secret code doesn't match any records, the user is informed politely

1.2.3 Inquire information about CSTU:

Scenario: User wants to learn more about CSTU.

Actor: User

Processing steps:

  - Welcome the user warmly and introduce the chatbot's capabilities.
  - Provide a concise overview of CSTU, including its mission, academic programs, campus life, and achievements.
  - Offer a link to the official CSTU website for more detailed information.
  - If the user asks a specific question not covered in the overview:
    - Try to answer it using the chatbot's knowledge base.
    - If unable, politely direct them to the website for further research.

Expected Outcome:

- The user is warmly welcomed and introduced to the chatbot's capabilities.
- The user receives a concise overview of CSTU, including its mission, academic programs, campus life, and achievements.
- The user is offered a link to the official CSTU website for more detailed information.
- If the user asks a specific question not covered in the overview:

    - The chatbot attempts to answer using its knowledge base.
    - If unable to answer, the user is politely directed to the website for further research.

1.2.4 Register for Courses:

Scenario: User indicates their desire to register for courses.

Actor: User

Processing steps:

    - If applicable, inquire about the user's program, semester, or other relevant factors to assess eligibility.
    - Present a user-friendly interface for browsing and selecting courses from a categorized list or searching based on specific criteria.
    - Clearly summarize chosen courses and request the user's name and email for registration.
    - Upon confirmation:
        - Call the "registration" function with user information and course details.
        - Send a confirmation email to the user (using body parameter).
        - Securely store registration information in "registration_records.csv".
    - Gracefully handle situations where users don't provide required information or make invalid selections.

Expected Outcome:

- If applicable, the user is asked about their program, semester, or other relevant factors to assess eligibility.
- The user is presented with a user-friendly interface to browse and select courses from a categorized list or search based on specific criteria.
- Chosen courses are summarized, and the user is requested to provide their name and email for registration.
- Upon confirmation:
    - The "registration" function is called with user information and course details.
    - A confirmation email is sent to the user.
    - Registration information is securely stored in "registration_records.csv".
    - The system gracefully handles situations where users don't provide required information or make invalid selections.

1.2.5 Get Registration Details:

Scenario: User requests to view their course registration details.

Actor: User

Processing steps:

- Ask for the user's email address to verify their identity.
- Utilize the "get_registration" function with the provided email address.
- Display relevant registration details (courses, dates, etc.) in a clear and organized format.
- If the email address doesn't match any records, inform the user politely.

Expected Outcome:

- User is asked to provide their email address for verification.
- The "get_registration" function is called with the provided email address.
- Relevant registration details (courses, dates, etc.) are displayed in a clear and organized format.
- If the email address doesn't match any records, the user is informed politely.

1.2.6 Get Course Grades:

Scenario: User inquires about their course grades.

Actor: User

Processing steps:

- Request the user's email address for verification.
- Call the "get_grades" function with the provided email address.
- Display retrieved grades information, highlighting key details like course names, marks, and overall GPA (if applicable).
- If the email address doesn't match any records, inform the user politely.

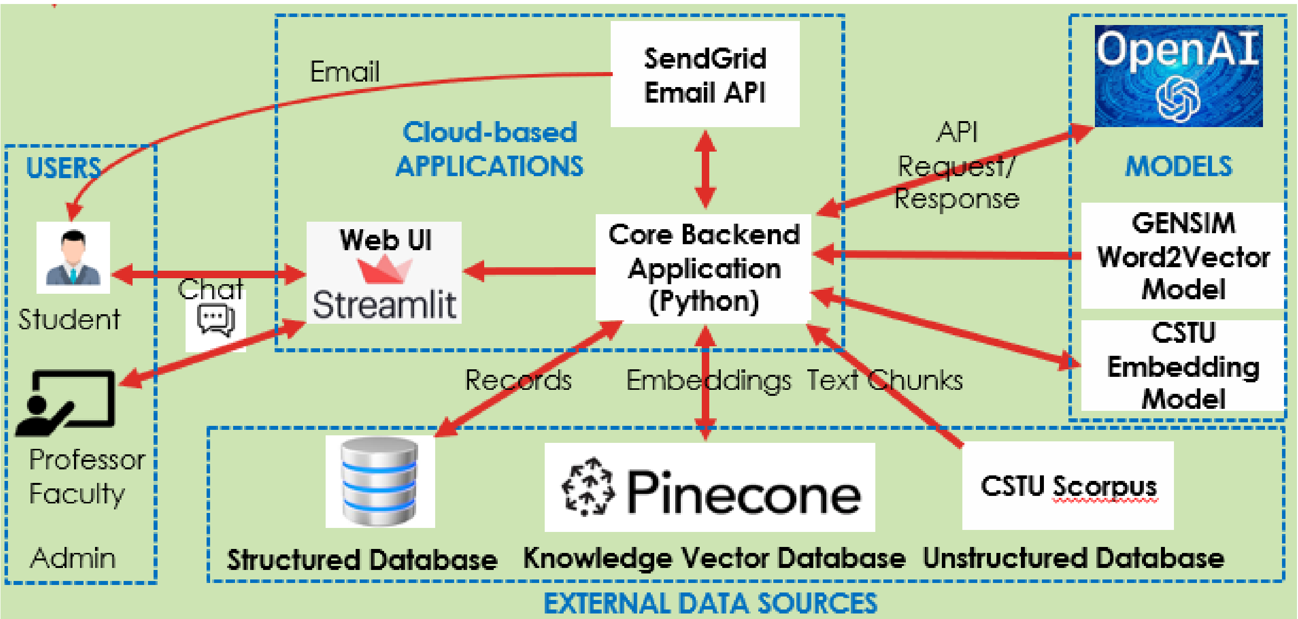
Expected Outcome:

- Users are requested to provide their email address for verification.
- The "get_grades" function is called with the provided email address.
- Retrieved grades information is displayed, highlighting key details like course names, marks, and overall GPA (if applicable).
- If the email address doesn't match any records, the user is informed politely.

2 Implement Technical Approach

Building CSTU knowledge base and implementing the Retrieval-Augmented Generation (RAG) system using a GPT model for generation and implementing a retrieval component tailored to CSTU-specific knowledge sources to deliver accurate, informative, and efficient responses for the chatbot.

2.1 System Architecture



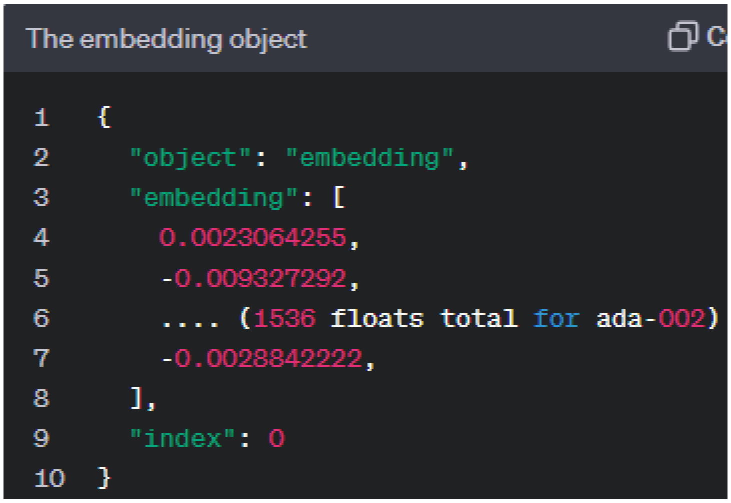
2.2 Large Language Model Selection:

Evaluate and choose a pre-trained model suitable for chatbot development, considering factors such as model size, architecture, and language understanding capabilities.

2.2.1 Embedding model

There are two options:

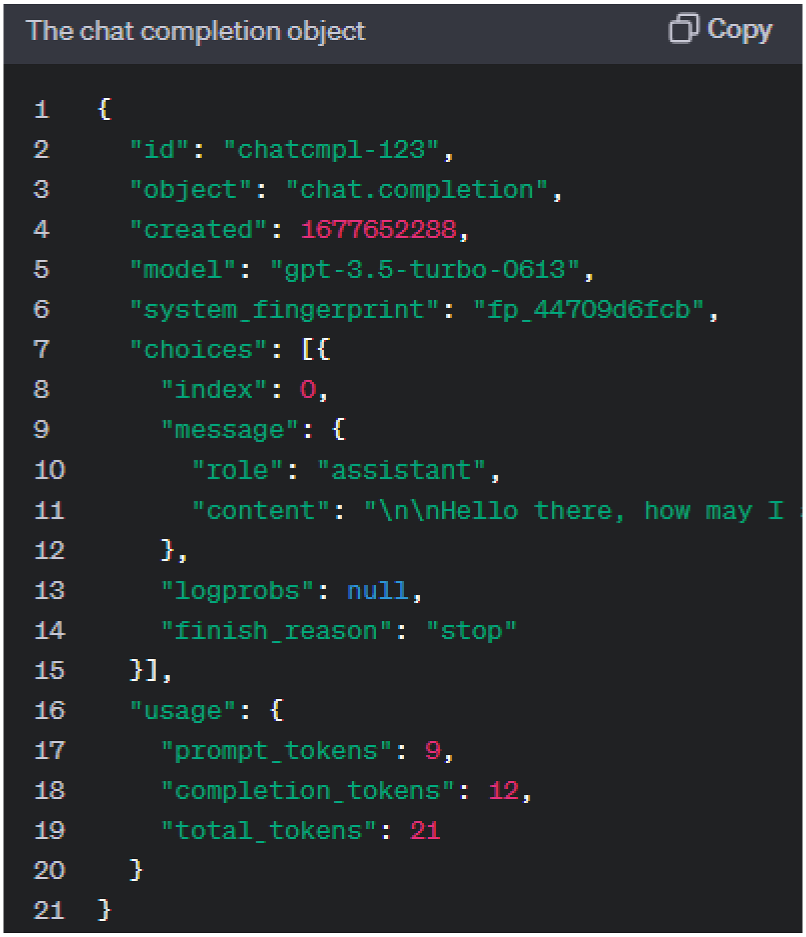
- Building CSTU-embedding-model based on training vectors from CSTU text corpus by Gensim Word2Vector model.
- Using pre-trained OpenAI Text-embedding-ada-002 with maximum 8,192 input tokens and output dimension of 1,536.


The embedding object

```
{
  "object": "embedding",
  "embedding": [
    0.0023064255,
    -0.009327292,
    .... (1536 floats total for ada-002)
    -0.0028842222,
  ],
  "index": 0
}
```

2.2.2 Chat completion model

OpenAI gpt-3.5-turbo-0613 can understand and generate natural language, code, function call and have been optimized for chat using the Chat Completions API.



```
{
  "id": "chatcmpl-123",
  "object": "chat.completion",
  "created": 1677652288,
  "model": "gpt-3.5-turbo-0613",
  "system_fingerprint": "fp_44709d6fcb",
  "choices": [{
    "index": 0,
    "message": {
      "role": "assistant",
      "content": "\n\nHello there, how may I
    },
    "logprobs": null,
    "finish_reason": "stop"
  }],
  "usage": {
    "prompt_tokens": 9,
    "completion_tokens": 12,
    "total_tokens": 21
  }
}
```

2.3 Data Preparation:

Steps:

- Gather a diverse set of training data, including webpages, catalogues, questions related to CSTU, user interactions, and potential scenarios.
- Clean and preprocess the collected data to ensure consistency and relevance.
- Format data with a compatible format.

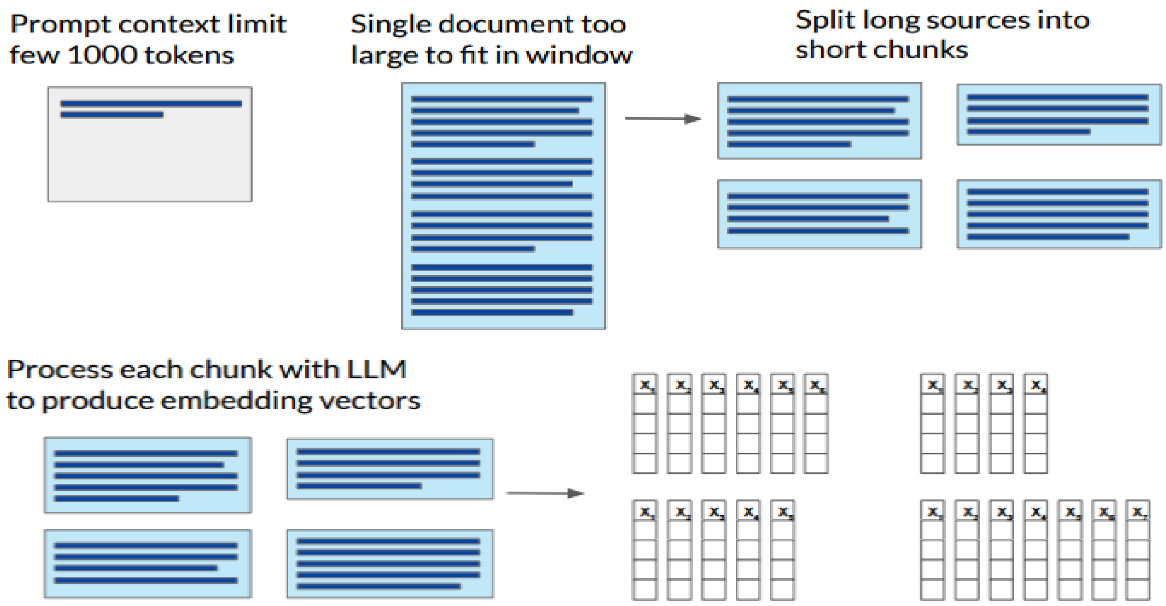
Requirements:

- Data must fit inside the context window.

  The OpenAI **gpt-3.5-turbo-0613** model has a context window of **16,000 tokens**. It can consider up to 16,000 words at once during its processing.

  The OpenAI Text-embedding-ada-002 model can process a maximum of 8,192 input tokens and output dimension of 1,536.

- Data must be in format that allows its relevance to be assessed at inference time: Embedding


Prompt context limit few 1000 tokens
Single document too large to fit in window
Split long sources into short chunks
Process each chunk with LLM to produce embedding vectors

2.4 Building knowledge base:

Organize the KB effectively for efficient retrieval by the RAG system using metadata tagging, hierarchical information architecture etc.

Pinecone is used as a cloud-based vector database to store, index, query, retrieve variety of multi-dimensional vector data including text, image, audio and code with API Integration and different real-time vector search methods:

- Creating index and storing numerical vectors generated from embedded document into knowledge base.
- Enriching knowledge base: Pinecone's indexing capabilities enable us to continuously enrich its knowledge base by adding new vector data.
- Performing a comparison and similarity search in the knowledge base for generated embedding of user input.
- Retrieving most relevant information by extracting the corresponding metadata from these vectors, including the content of the knowledge base entries

Its benefits: Scalability, Reliability, Simplicity, Performance

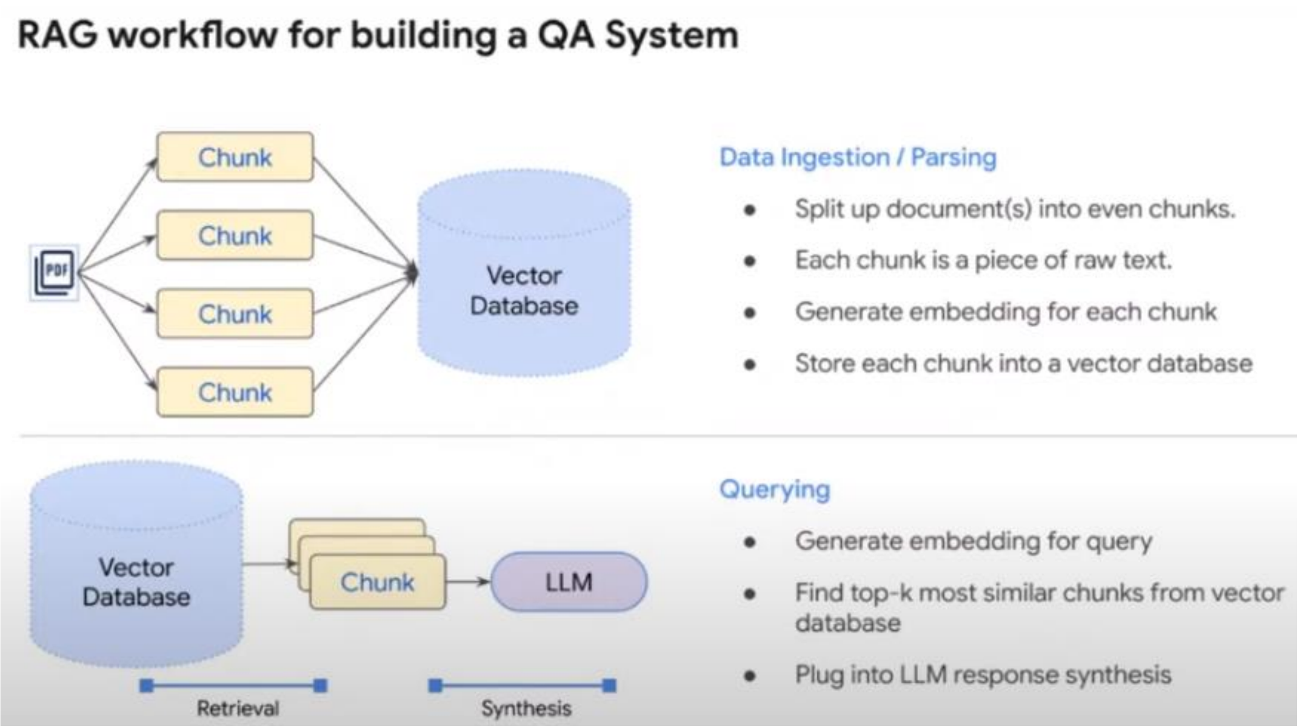
Workflow for Knowledge Base:

1. Training an embedding model
2. Establish Connection to Pinecone API
3. Create a database Index

4. Chunking text data and embedding
5. Upserting embedding vectors into knowledge vector database
6. Validating knowledge vector database status
7. Schedule updating knowledge vector database

2.5 Deploying RAG architecture:

RAG is a popular architecture pattern of implementing grounding with LLM with text chat UI. The idea is to have the LLM text chat UI as a front end for the document retrieval with vector search and summarization of the result. The selected model is integrated with CSTU domain-specific Knowledge Base and continually enhance knowledge base integration for optimal context-aware responses.



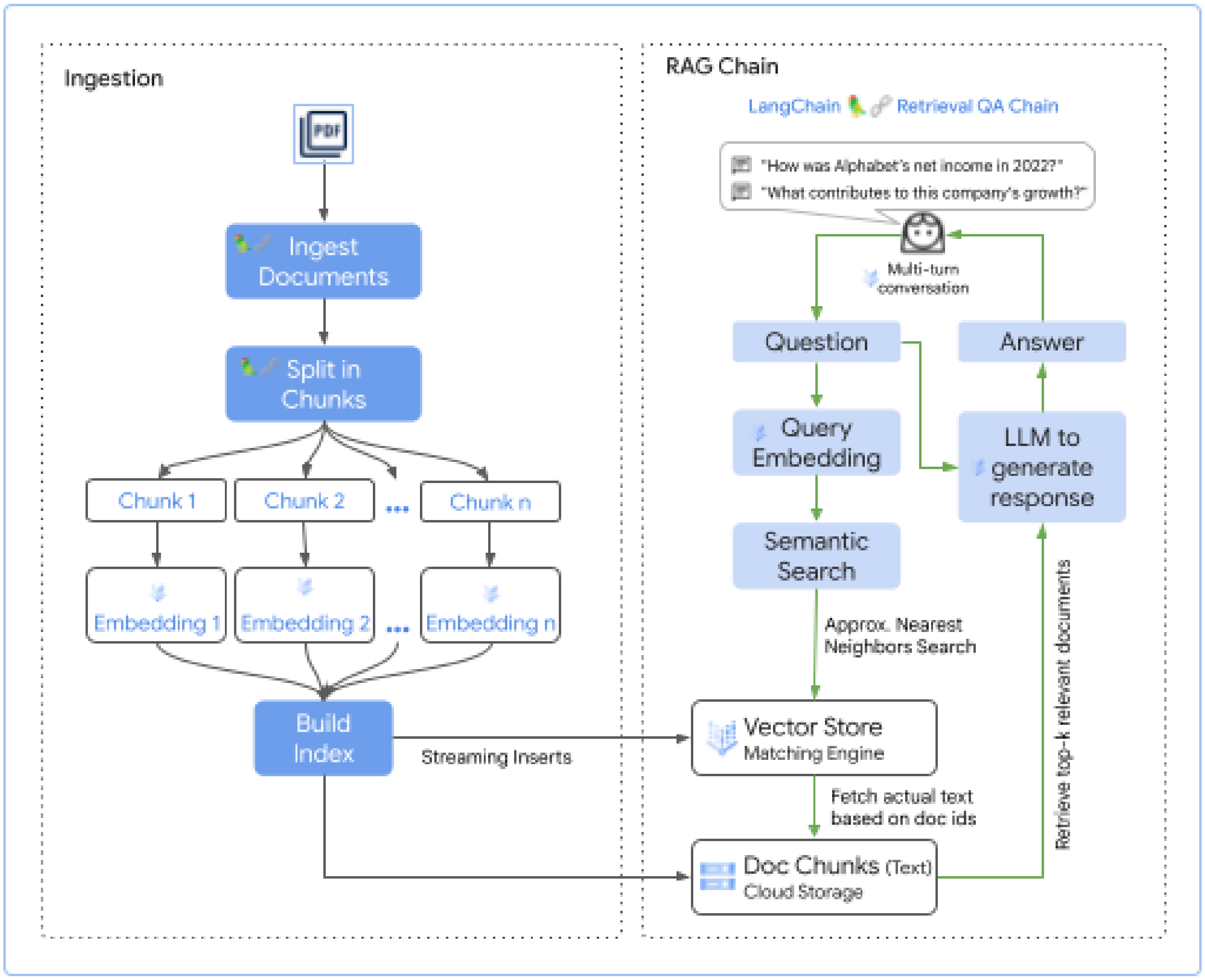
RAG workflow for building a QA System

Data Ingestion / Parsing

- Split up document(s) into even chunks.
- Each chunk is a piece of raw text.
- Generate embedding for each chunk
- Store each chunk into a vector database

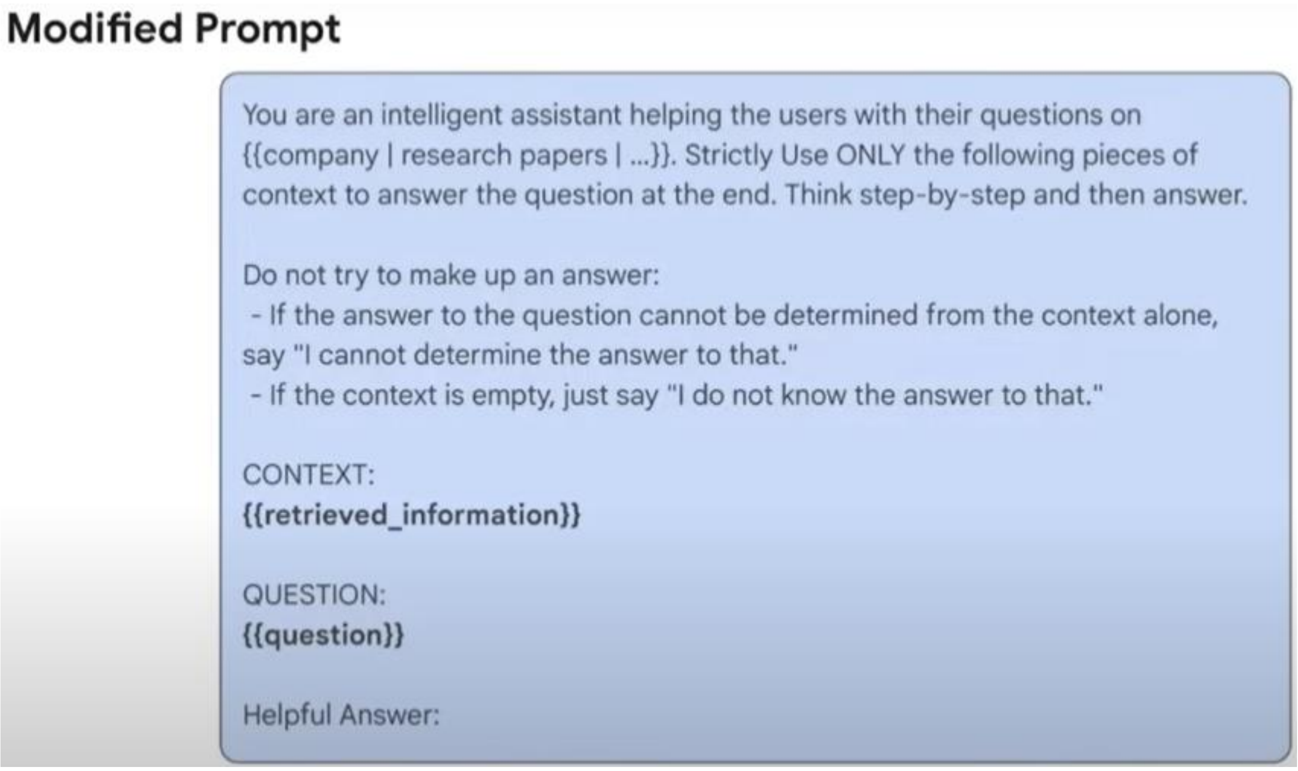
Querying

- Generate embedding for query
- Find top-k most similar chunks from vector database
- Plug into LLM response synthesis



Workflow for chatbot:

1. **User Input:** could be a question, an inquiry of information, or a request for a task.
2. **Embedding vector search**: generating an embedding for the user's input and performing a similarity search for generated embedding of user input in the knowledge base
3. **Retrieval of Relevant Information:** extracts the corresponding information (metadata) from these vectors, including the content of the knowledge base entries
4. **Response Generation:** Generating a response to the user's query based on the retrieved relevant information, initial system guide and context history.
5. **Display Response:** Displaying answers to the user through the web user interface



Modified Prompt

You are an intelligent assistant helping the users with their questions on {{company | research papers | ...}}. Strictly Use ONLY the following pieces of context to answer the question at the end. Think step-by-step and then answer.

Do not try to make up an answer:
- If the answer to the question cannot be determined from the context alone, say "I cannot determine the answer to that."
- If the context is empty, just say "I do not know the answer to that."

CONTEXT:
{{retrieved_information}}

QUESTION:
{{question}}

Helpful Answer:

2.6 Coding tasks

2.6.1 Installation, import, setup libraries and environment variables

- Libraries: openai, Word2Vec, tokenizers, python-dotenv, pinecone-client, nltk, numpy, pandas, sendgrid, streamlit, PyPDF2 , etc.
- Environment variables: OPENAI_API_KEY, PINECONE_API_KEY, SENDGRID_API_KEY
- Setting Up and Initializing Session State for OpenAI, Pinecone, Sendgrid

2.6.2 Building knowledge base

- Training and saving the embedding model
- Creating Pinecone index
- Chunking and embedding CSTU documents
- Upserting embedding vectors into database

2.6.3 Building chatbot

- Defining initial system guide/context
- Defining chat completion API
- Defining OpenAI API function call
- Loading embedding model
- Initializing Streamlit Session State
- Controlling user and system interactions
- Embedding user chat input into an input vector

- Searching and retrieving metadata from knowledge base
- Composing prompts to query OpenAI GPT model
- Getting and analyzing response from OpenAI GPT
- Handling internal functions and responding to user.
- Updating chat history context

3 Working prototype and Demo

3.1 Prototype

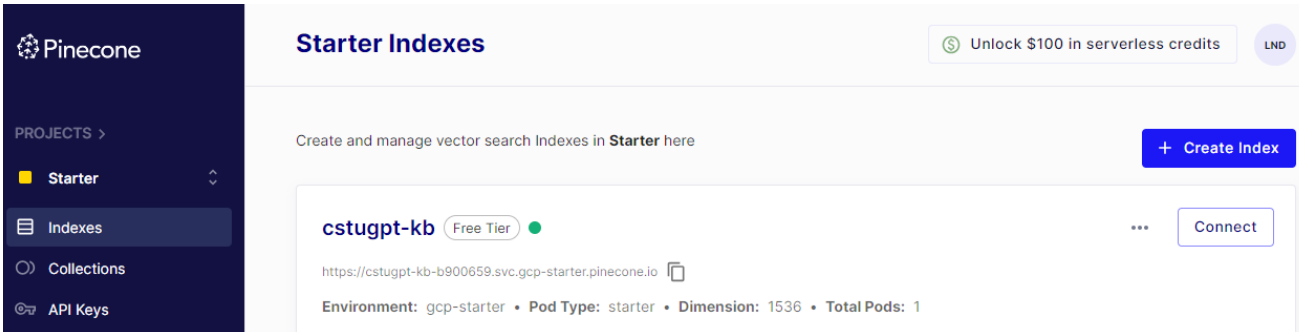
The prototype is deployed for testing, and user feedback is collected for continuous optimization.

- CSTU embedding model
- CSTU knowledge base
- Chat bot

Source code: github.com/dnlamvhit/CSTU-ChatBot/blob/main/chatbot.py

Chatbot: https://cstu-chatbot-v1.streamlit.app

3.2 Demo



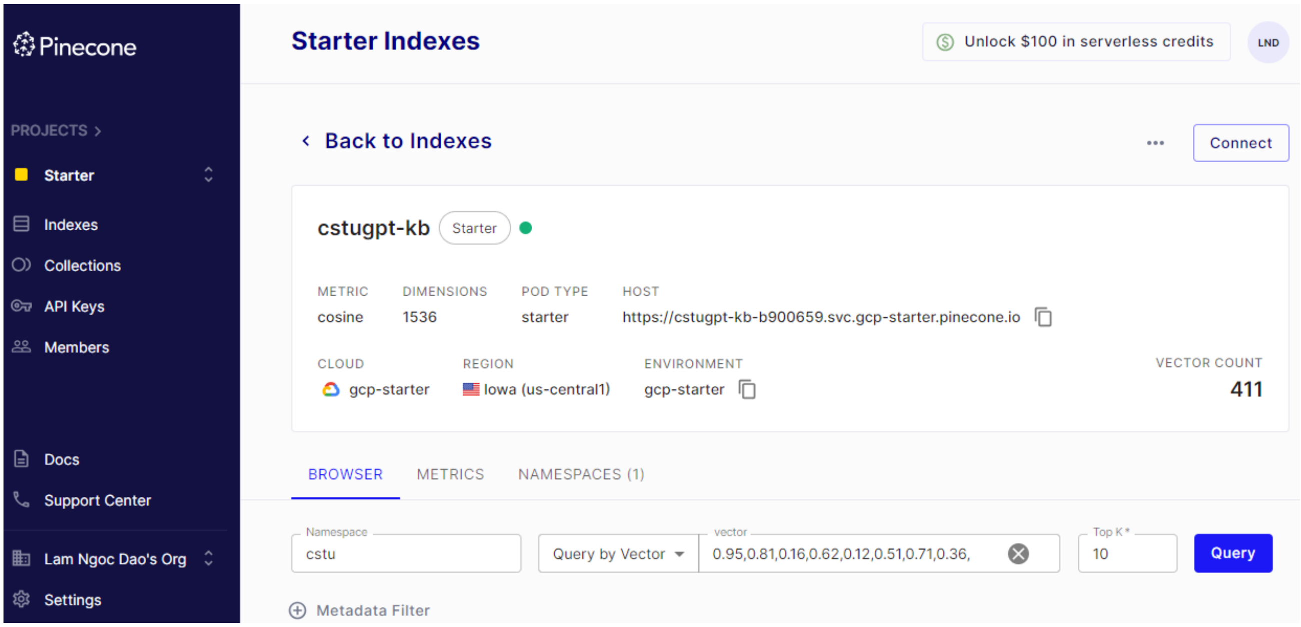
Starter Indexes

Create and manage vector search Indexes in **Starter** here

cstugpt-kb

https://cstugpt-kb-b900659.svc.gcp-starter.pinecone.io

Environment: gcp-starter • **Pod Type:** starter • **Dimension:** 1536 • **Total Pods:** 1

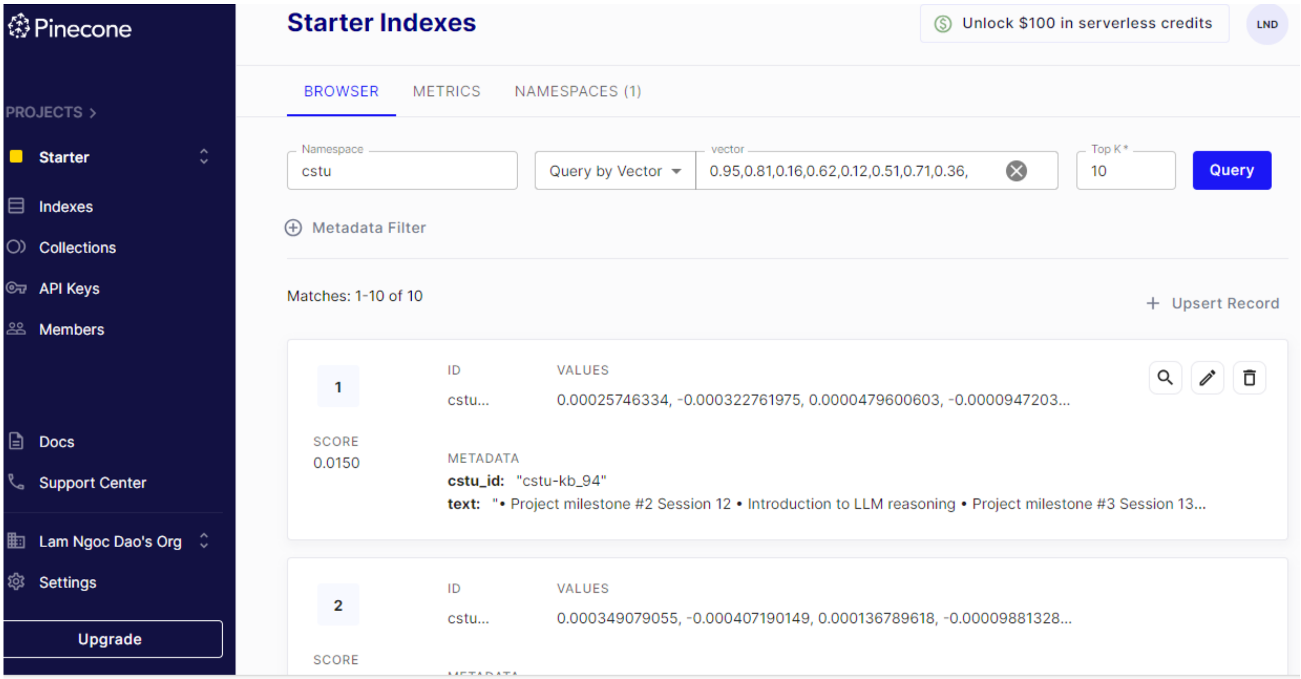


Starter Indexes

cstugpt-kb Starter

METRIC: cosine
DIMENSIONS: 1536
POD TYPE: starter
HOST: https://cstugpt-kb-b900659.svc.gcp-starter.pinecone.io
CLOUD: gcp-starter
REGION: Iowa (us-central1)
ENVIRONMENT: gcp-starter
VECTOR COUNT: 411

Namespace: cstu
Query by Vector
vector: 0.95,0.81,0.16,0.62,0.12,0.51,0.71,0.36,
Top K *: 10



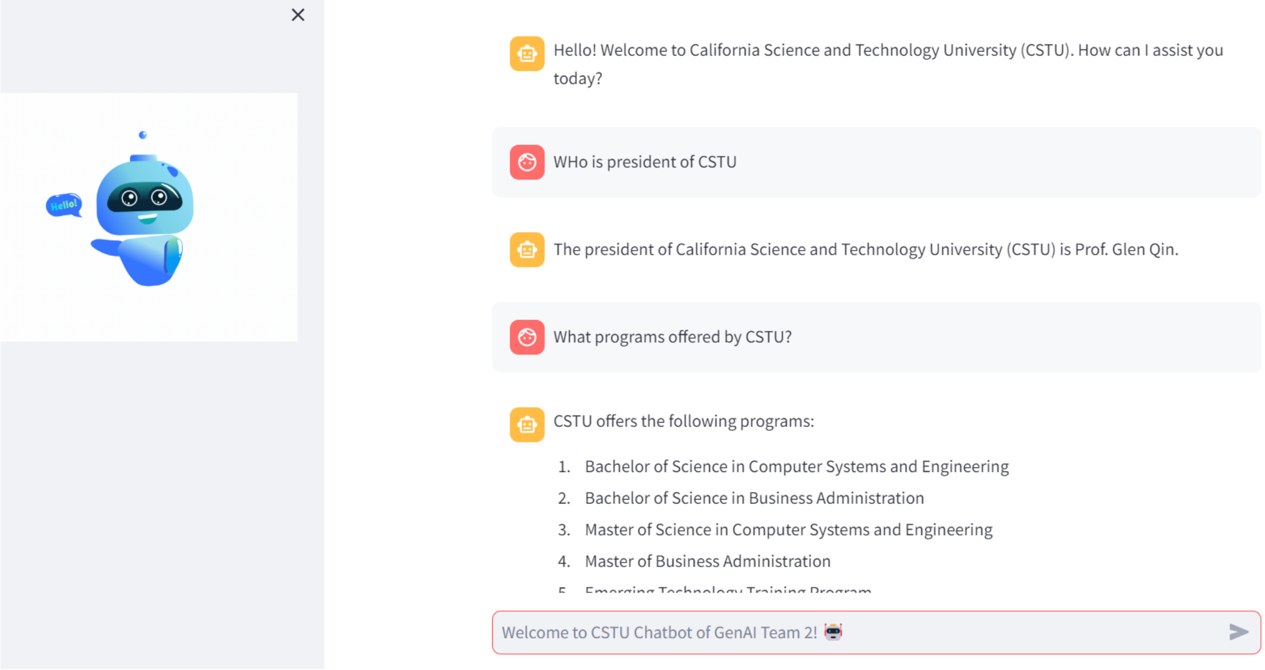
Starter Indexes

Namespace: cstu
Query by Vector
vector: 0.95,0.81,0.16,0.62,0.12,0.51,0.71,0.36,
Top K *: 10

Matches: 1-10 of 10

1
ID: cstu...
VALUES: 0.00025746334, -0.000322761975, 0.0000479600603, -0.0000947203...
SCORE: 0.0150
METADATA
cstu_id: "cstu-kb_94"
text: "• Project milestone #2 Session 12 • Introduction to LLM reasoning • Project milestone #3 Session 13...

2
ID: cstu...
VALUES: 0.000349079055, -0.000407190149, 0.000136789618, -0.00009881328...
SCORE



Hello! Welcome to California Science and Technology University (CSTU). How can I assist you today?

WHo is president of CSTU

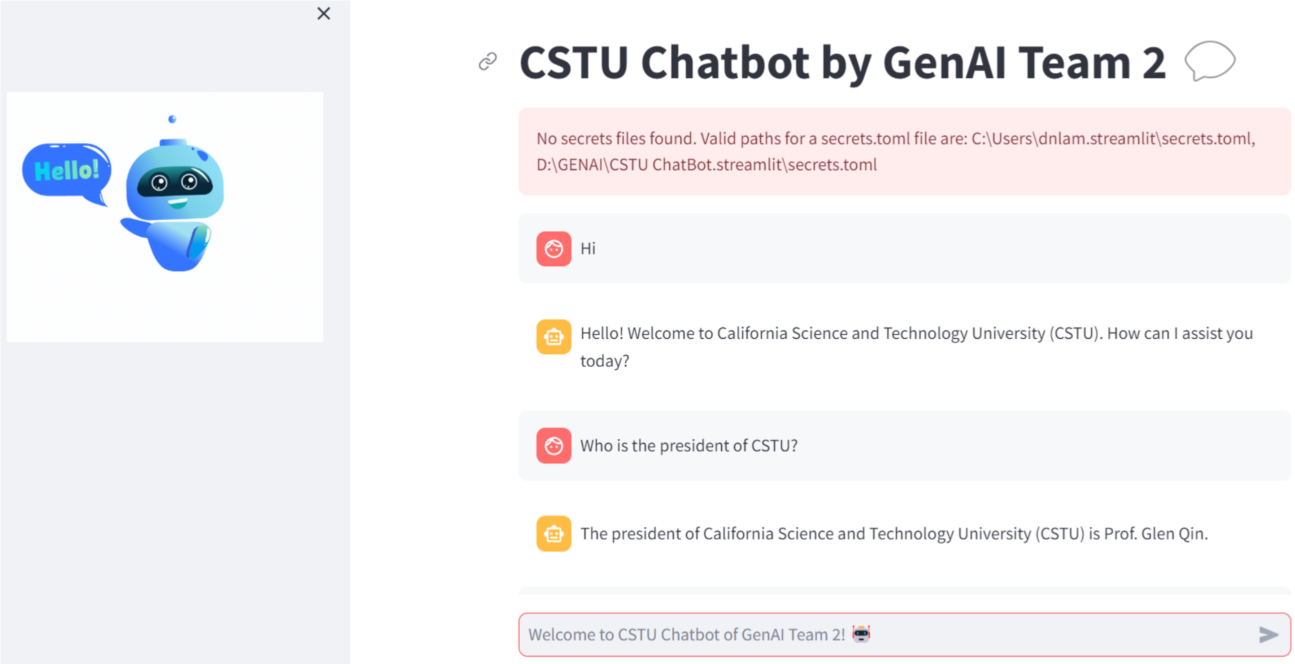
The president of California Science and Technology University (CSTU) is Prof. Glen Qin.

What programs offered by CSTU?

CSTU offers the following programs:

1. Bachelor of Science in Computer Systems and Engineering
2. Bachelor of Science in Business Administration
3. Master of Science in Computer Systems and Engineering
4. Master of Business Administration
5. Emerging Technology Training Program

Welcome to CSTU Chatbot of GenAI Team 2!



CSTU Chatbot by GenAI Team 2

No secrets files found. Valid paths for a secrets.toml file are: C:\Users\dnlam.streamlit\secrets.toml, D:\GENAI\CSTU ChatBot.streamlit\secrets.toml

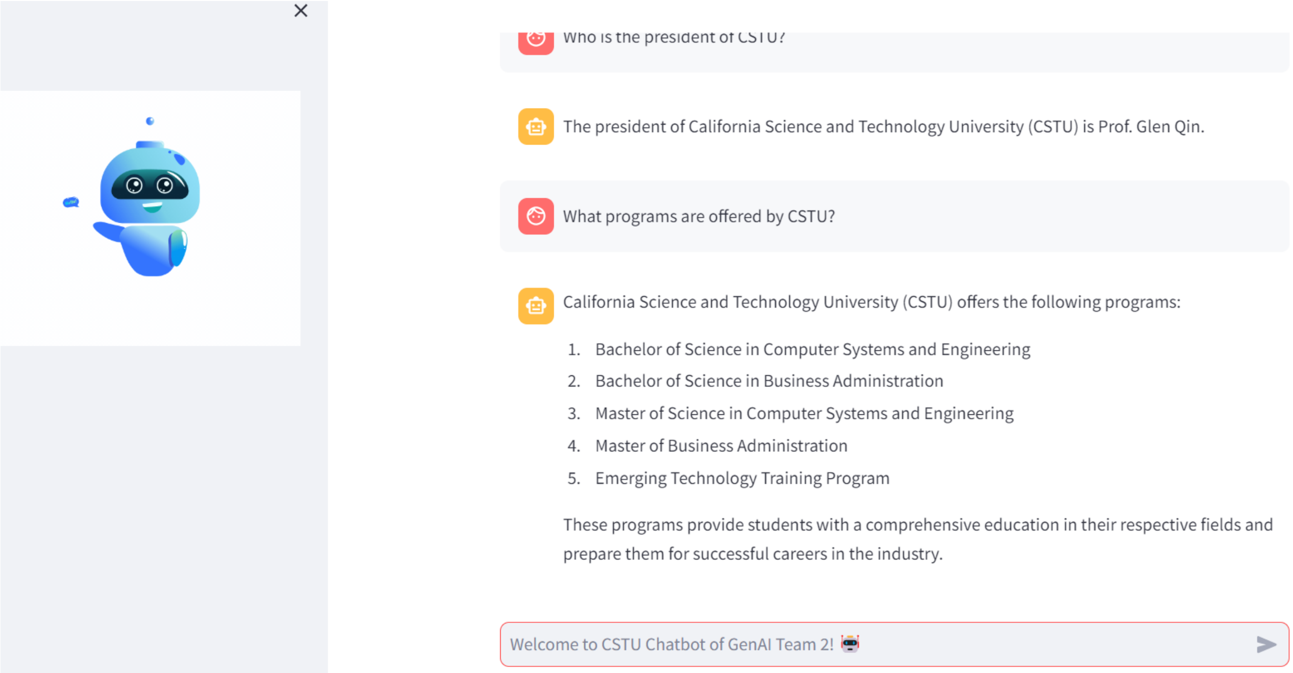
Hi

Hello! Welcome to California Science and Technology University (CSTU). How can I assist you today?

Who is the president of CSTU?

The president of California Science and Technology University (CSTU) is Prof. Glen Qin.

Welcome to CSTU Chatbot of GenAI Team 2!


Who is the president of CSTU?

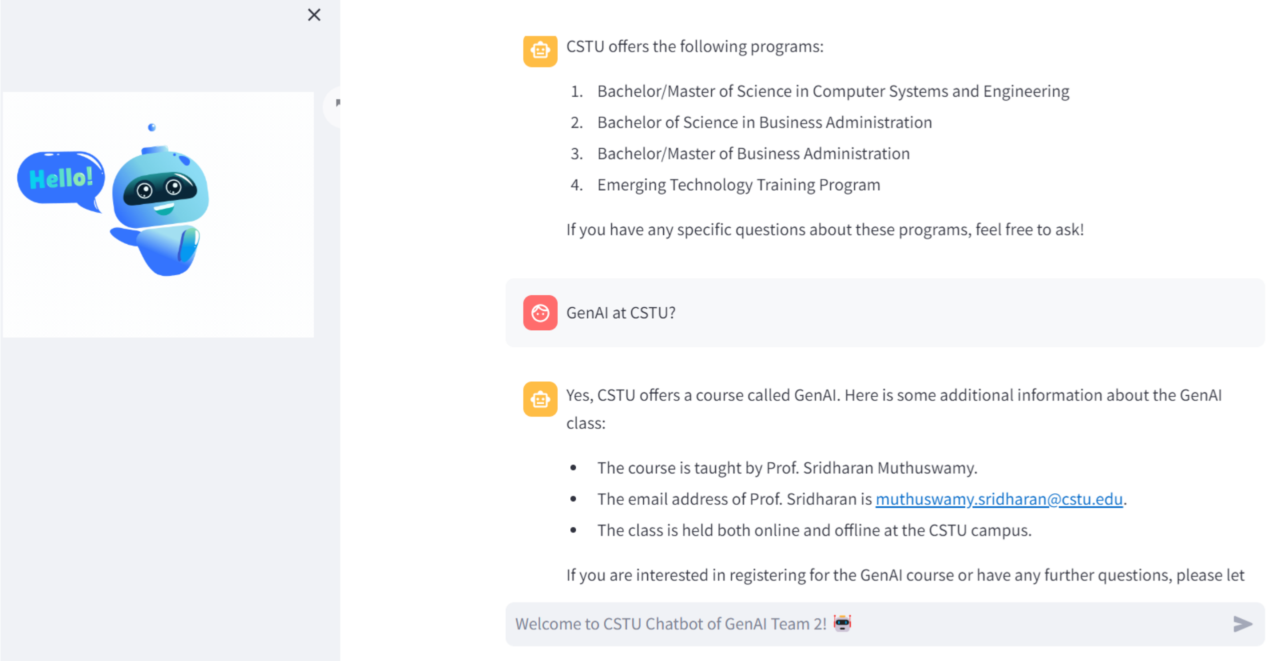
The president of California Science and Technology University (CSTU) is Prof. Glen Qin.

What programs are offered by CSTU?

California Science and Technology University (CSTU) offers the following programs:

1. Bachelor of Science in Computer Systems and Engineering
2. Bachelor of Science in Business Administration
3. Master of Science in Computer Systems and Engineering
4. Master of Business Administration
5. Emerging Technology Training Program

These programs provide students with a comprehensive education in their respective fields and prepare them for successful careers in the industry.



CSTU offers the following programs:

1. Bachelor/Master of Science in Computer Systems and Engineering
2. Bachelor of Science in Business Administration
3. Bachelor/Master of Business Administration
4. Emerging Technology Training Program

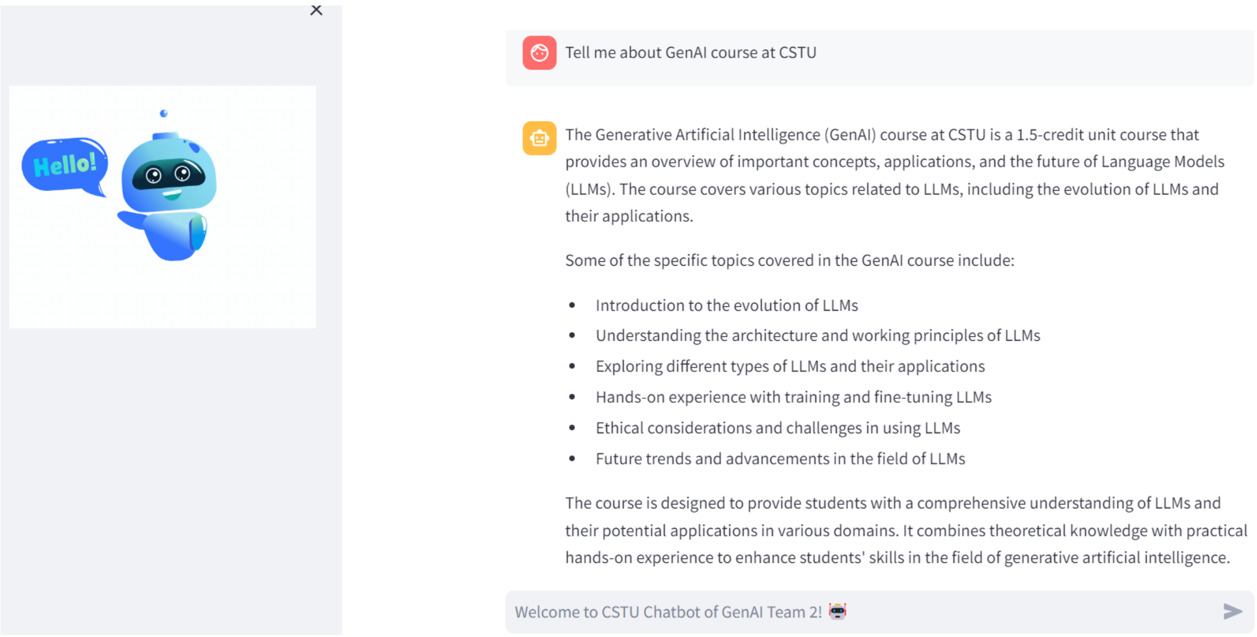
If you have any specific questions about these programs, feel free to ask!

GenAI at CSTU?

Yes, CSTU offers a course called GenAI. Here is some additional information about the GenAI class:

- The course is taught by Prof. Sridharan Muthuswamy.
- The email address of Prof. Sridharan is muthuswamy.sridharan@cstu.edu.
- The class is held both online and offline at the CSTU campus.

If you are interested in registering for the GenAI course or have any further questions, please let



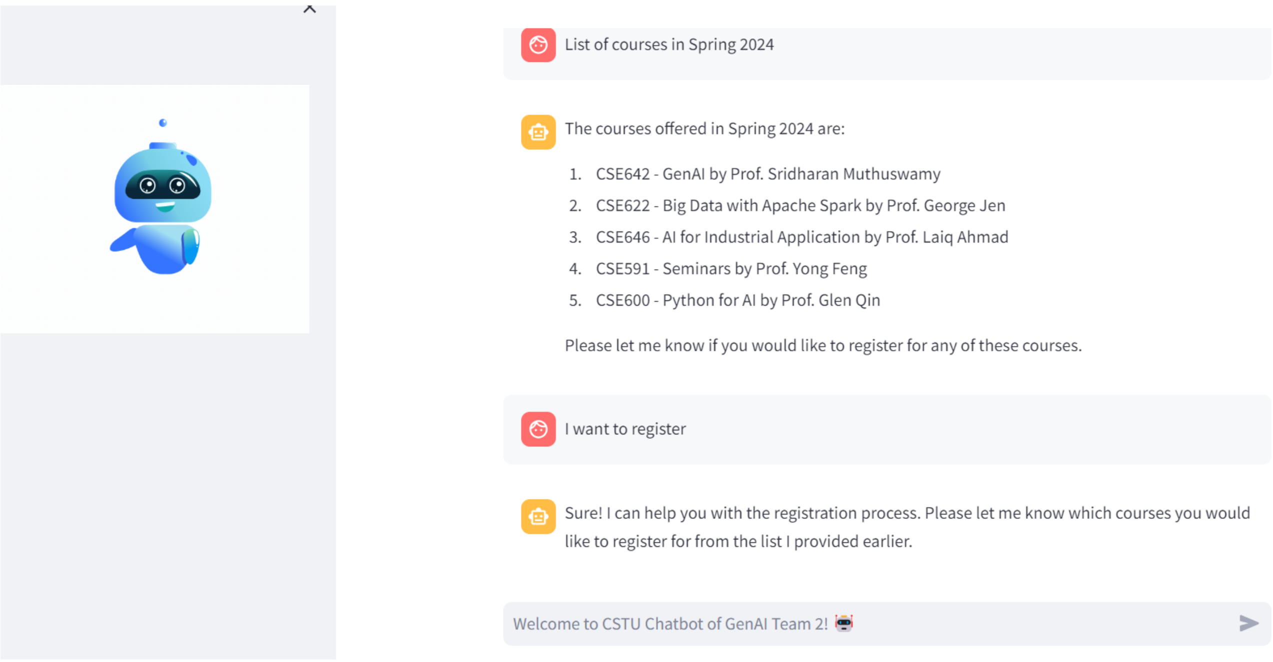
Tell me about GenAI course at CSTU

The Generative Artificial Intelligence (GenAI) course at CSTU is a 1.5-credit unit course that provides an overview of important concepts, applications, and the future of Language Models (LLMs). The course covers various topics related to LLMs, including the evolution of LLMs and their applications.

Some of the specific topics covered in the GenAI course include:

- Introduction to the evolution of LLMs
- Understanding the architecture and working principles of LLMs
- Exploring different types of LLMs and their applications
- Hands-on experience with training and fine-tuning LLMs
- Ethical considerations and challenges in using LLMs
- Future trends and advancements in the field of LLMs

The course is designed to provide students with a comprehensive understanding of LLMs and their potential applications in various domains. It combines theoretical knowledge with practical hands-on experience to enhance students' skills in the field of generative artificial intelligence.



List of courses in Spring 2024

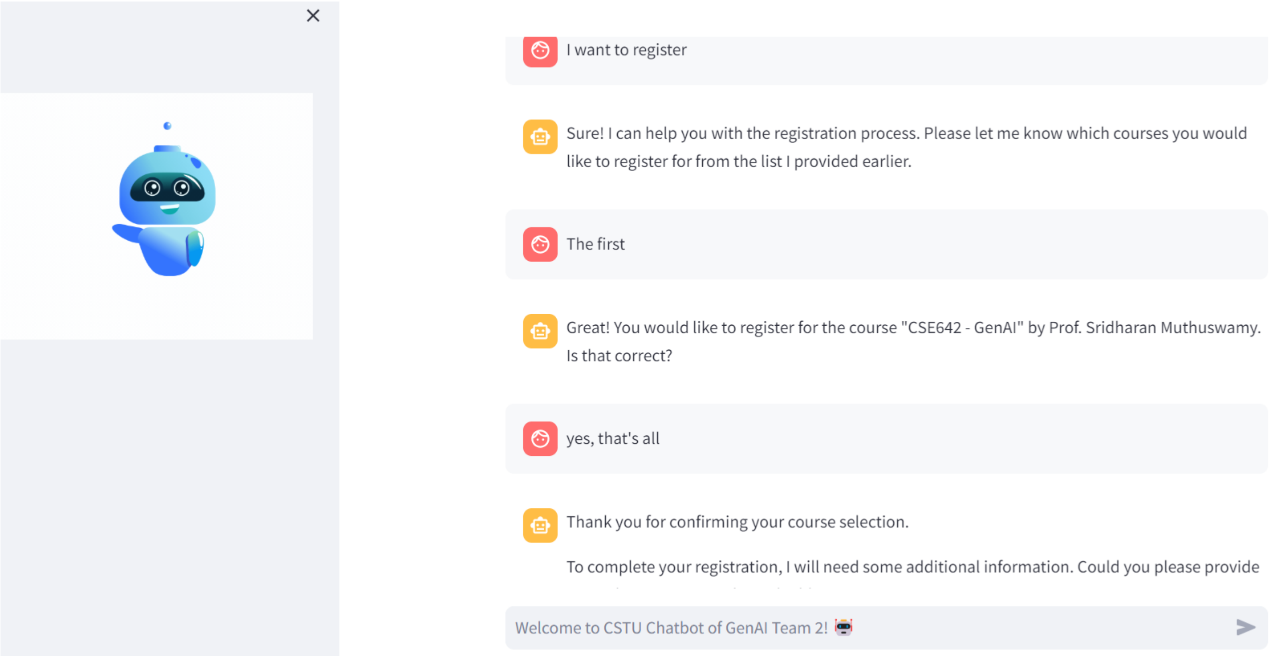
The courses offered in Spring 2024 are:

1. CSE642 - GenAI by Prof. Sridharan Muthuswamy
2. CSE622 - Big Data with Apache Spark by Prof. George Jen
3. CSE646 - AI for Industrial Application by Prof. Laiq Ahmad
4. CSE591 - Seminars by Prof. Yong Feng
5. CSE600 - Python for AI by Prof. Glen Qin

Please let me know if you would like to register for any of these courses.

I want to register

Sure! I can help you with the registration process. Please let me know which courses you would like to register for from the list I provided earlier.



I want to register

Sure! I can help you with the registration process. Please let me know which courses you would like to register for from the list I provided earlier.

The first

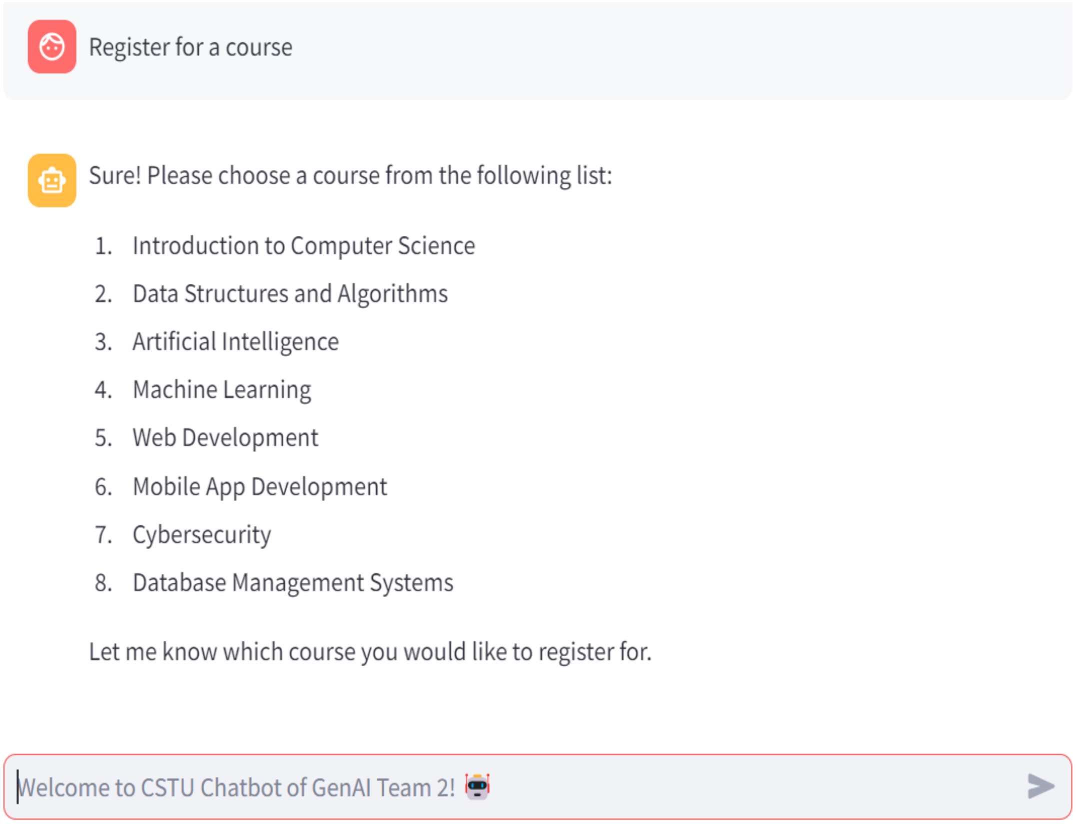
Great! You would like to register for the course "CSE642 - GenAI" by Prof. Sridharan Muthuswamy. Is that correct?

yes, that's all

Thank you for confirming your course selection.

To complete your registration, I will need some additional information. Could you please provide

Welcome to CSTU Chatbot of GenAI Team 2! 🤖



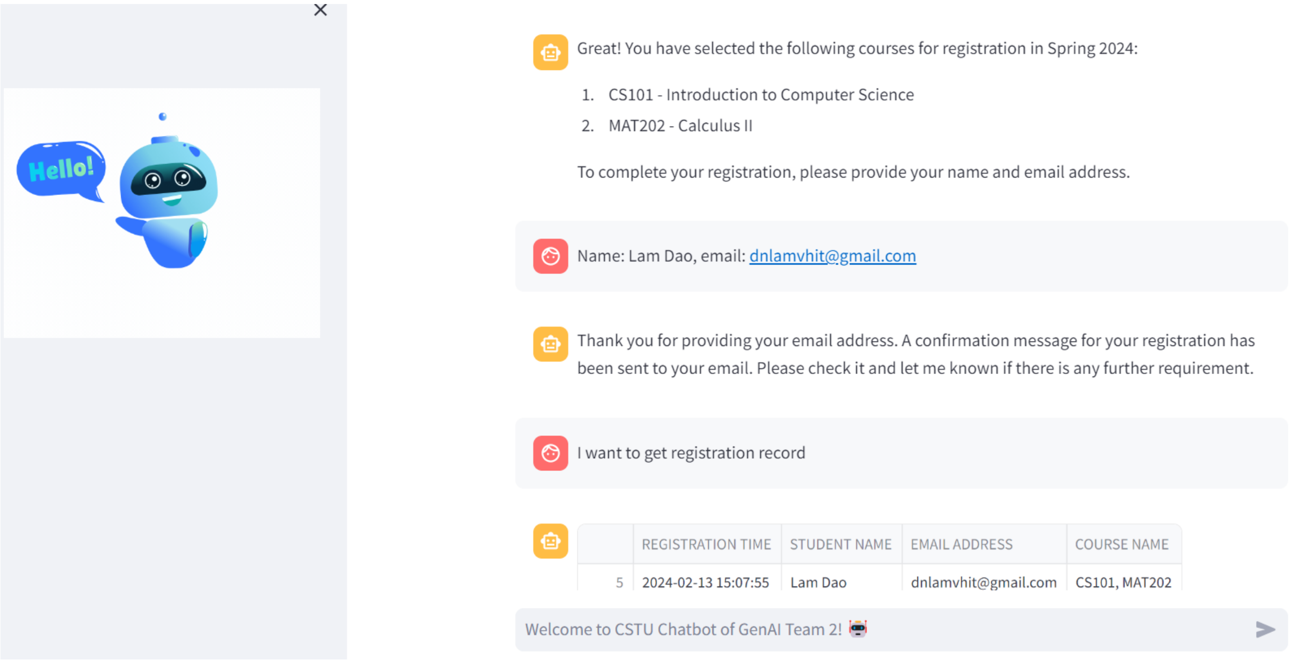
Register for a course

Sure! Please choose a course from the following list:

1. Introduction to Computer Science
2. Data Structures and Algorithms
3. Artificial Intelligence
4. Machine Learning
5. Web Development
6. Mobile App Development
7. Cybersecurity
8. Database Management Systems

Let me know which course you would like to register for.

Welcome to CSTU Chatbot of GenAI Team 2! 🤖



Great! You have selected the following courses for registration in Spring 2024:

1. CS101 - Introduction to Computer Science
2. MAT202 - Calculus II

To complete your registration, please provide your name and email address.

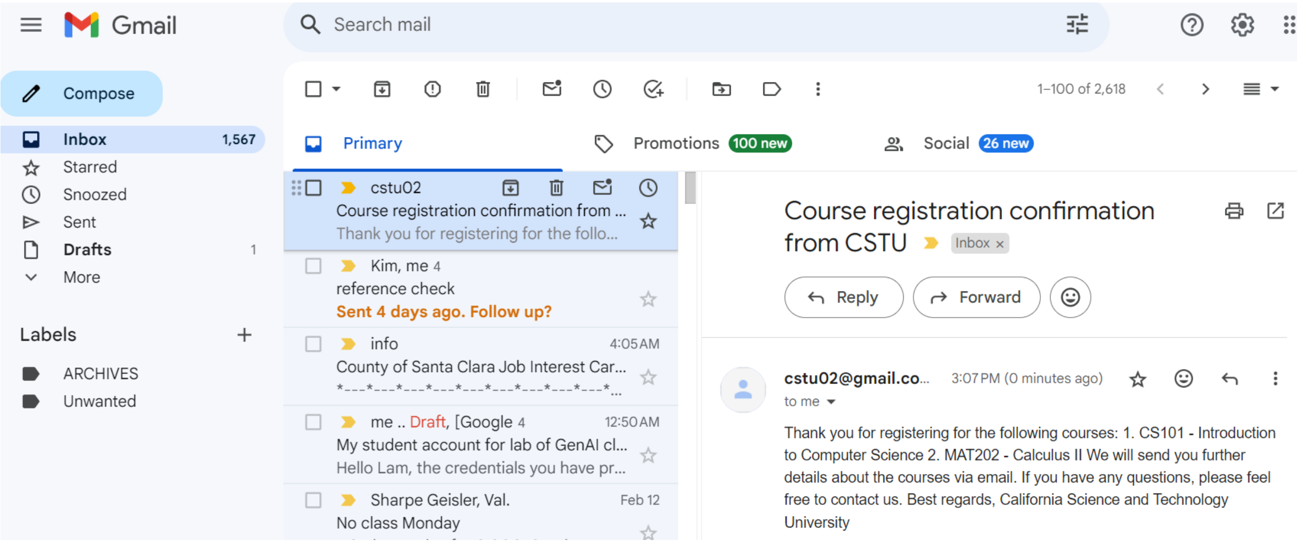
Name: Lam Dao, email: dnlamvhit@gmail.com

Thank you for providing your email address. A confirmation message for your registration has been sent to your email. Please check it and let me known if there is any further requirement.

I want to get registration record

	REGISTRATION TIME	STUDENT NAME	EMAIL ADDRESS	COURSE NAME
5	2024-02-13 15:07:55	Lam Dao	dnlamvhit@gmail.com	CS101, MAT202

Welcome to CSTU Chatbot of GenAI Team 2!



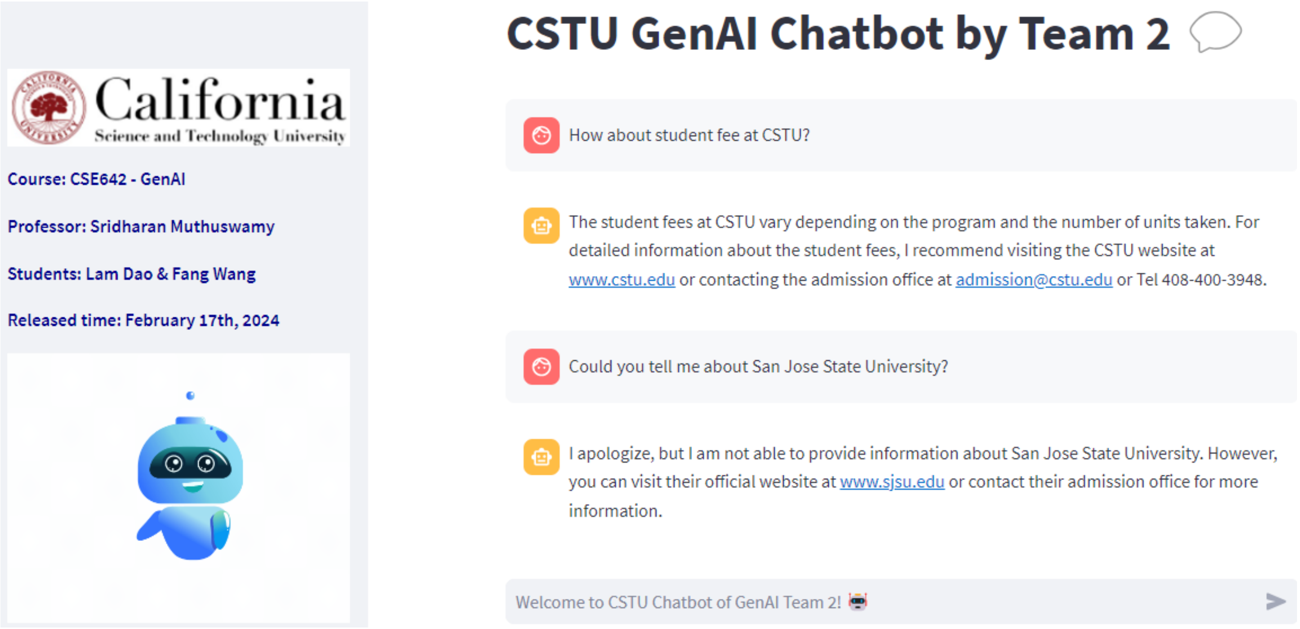
cstu02	Course registration confirmation from ... Thank you for registering for the follo...	
Kim, me 4	reference check Sent 4 days ago. Follow up?	
info	County of Santa Clara Job Interest Car... *---*---*---*---*---*---*---*---*...	4:05 AM
me .. Draft, [Google 4	My student account for lab of GenAI cl... Hello Lam, the credentials you have pr...	12:50 AM
Sharpe Geisler, Val.	No class Monday	Feb 12

Course registration confirmation from CSTU

cstu02@gmail.co... 3:07 PM (0 minutes ago)

to me

Thank you for registering for the following courses: 1. CS101 - Introduction to Computer Science 2. MAT202 - Calculus II We will send you further details about the courses via email. If you have any questions, please feel free to contact us. Best regards, California Science and Technology University



Course: CSE642 - GenAI

Professor: Sridharan Muthuswamy

Students: Lam Dao & Fang Wang

Released time: February 17th, 2024

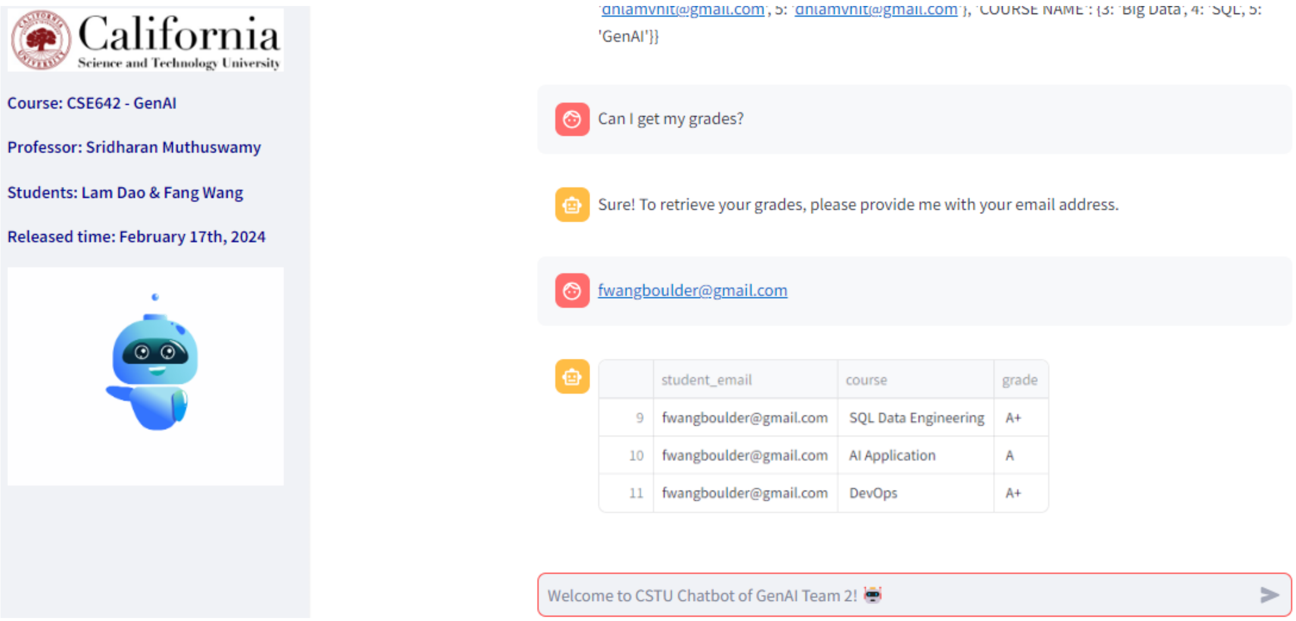
CSTU GenAI Chatbot by Team 2

How about student fee at CSTU?

The student fees at CSTU vary depending on the program and the number of units taken. For detailed information about the student fees, I recommend visiting the CSTU website at www.cstu.edu or contacting the admission office at admission@cstu.edu or Tel 408-400-3948.

Could you tell me about San Jose State University?

I apologize, but I am not able to provide information about San Jose State University. However, you can visit their official website at www.sjsu.edu or contact their admission office for more information.



Course: CSE642 - GenAI

Professor: Sridharan Muthuswamy

Students: Lam Dao & Fang Wang

Released time: February 17th, 2024

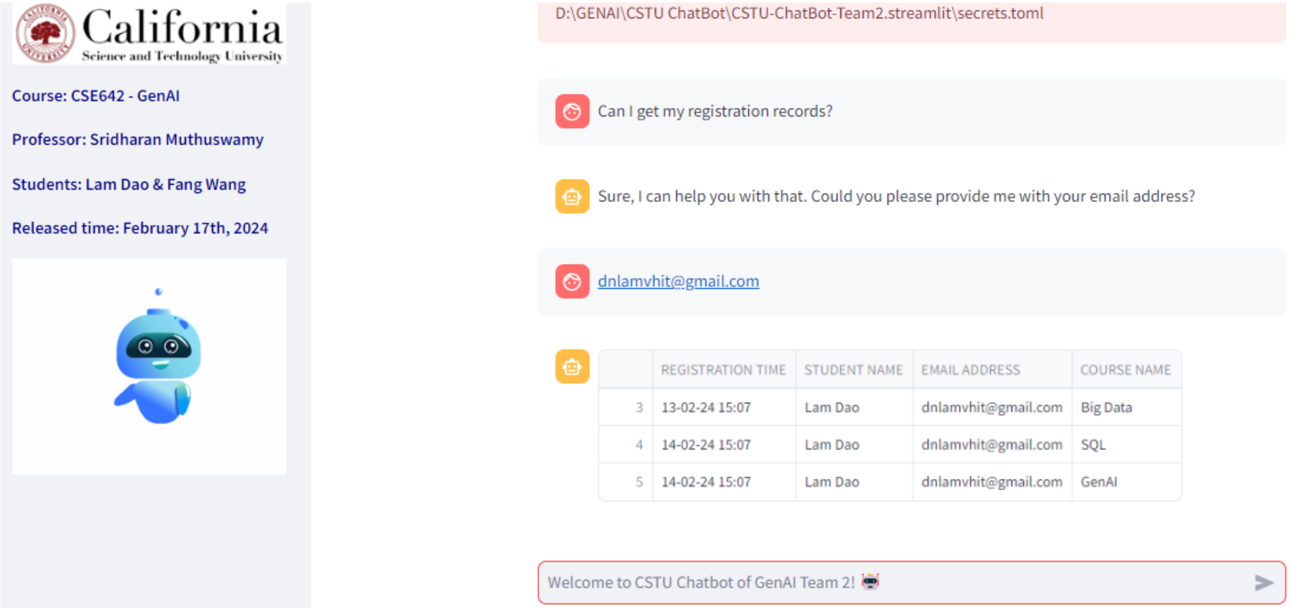
'dhlamvhit@gmail.com', 5: 'dhlamvhit@gmail.com'}, 'COURSE NAME': {3: 'Big Data', 4: 'SQL', 5: 'GenAI'}}

Can I get my grades?

Sure! To retrieve your grades, please provide me with your email address.

fwangboulder@gmail.com

	student_email	course	grade
9	fwangboulder@gmail.com	SQL Data Engineering	A+
10	fwangboulder@gmail.com	AI Application	A
11	fwangboulder@gmail.com	DevOps	A+



California Science and Technology University

Course: CSE642 - GenAI

Professor: Sridharan Muthuswamy

Students: Lam Dao & Fang Wang

Released time: February 17th, 2024

D:\GENAI\CSTU ChatBot\CSTU-ChatBot-Team2.streamlit\secrets.toml

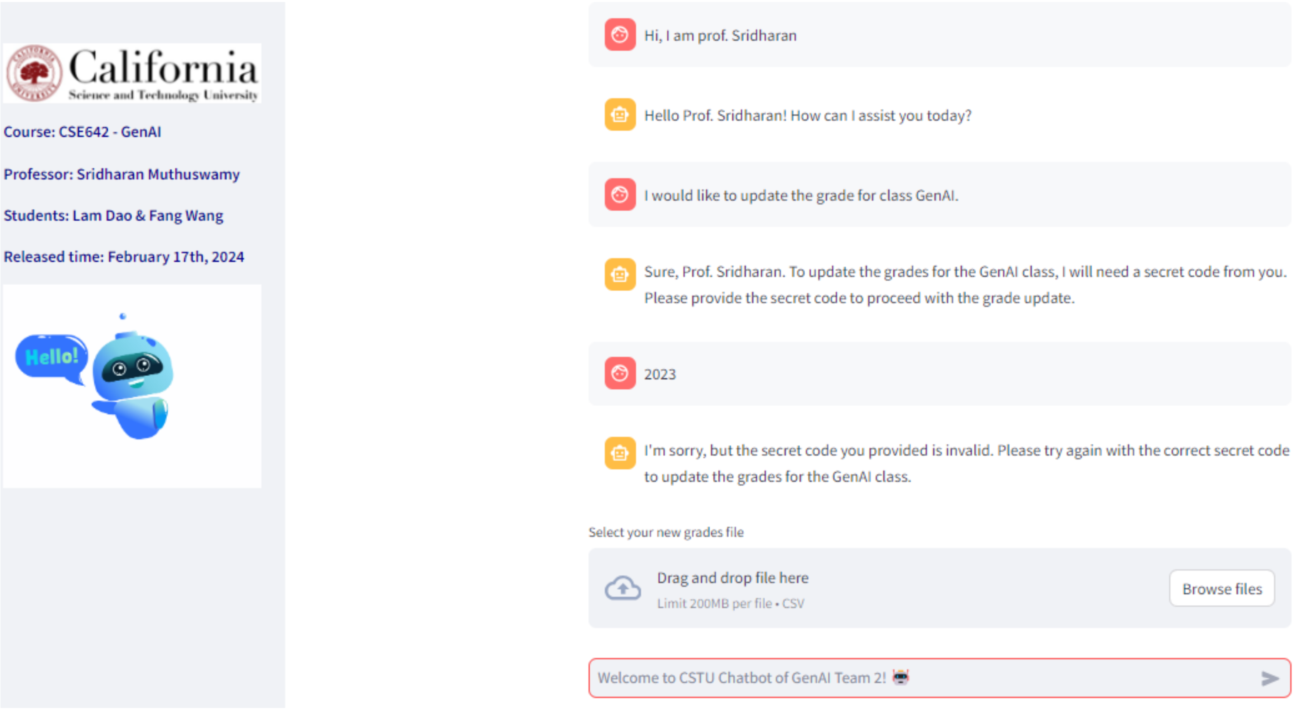
Can I get my registration records?

Sure, I can help you with that. Could you please provide me with your email address?

dnlamvhit@gmail.com

	REGISTRATION TIME	STUDENT NAME	EMAIL ADDRESS	COURSE NAME
3	13-02-24 15:07	Lam Dao	dnlamvhit@gmail.com	Big Data
4	14-02-24 15:07	Lam Dao	dnlamvhit@gmail.com	SQL
5	14-02-24 15:07	Lam Dao	dnlamvhit@gmail.com	GenAI

Welcome to CSTU Chatbot of GenAI Team 2! 🤖



California Science and Technology University

Course: CSE642 - GenAI

Professor: Sridharan Muthuswamy

Students: Lam Dao & Fang Wang

Released time: February 17th, 2024

Hi, I am prof. Sridharan

Hello Prof. Sridharan! How can I assist you today?

I would like to update the grade for class GenAI.

Sure, Prof. Sridharan. To update the grades for the GenAI class, I will need a secret code from you. Please provide the secret code to proceed with the grade update.

2023

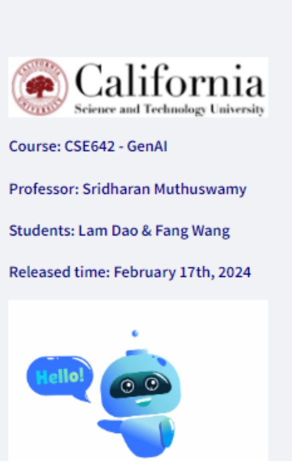
I'm sorry, but the secret code you provided is invalid. Please try again with the correct secret code to update the grades for the GenAI class.

Select your new grades file

Drag and drop file here

Limit 200MB per file • CSV

Welcome to CSTU Chatbot of GenAI Team 2! 🤖

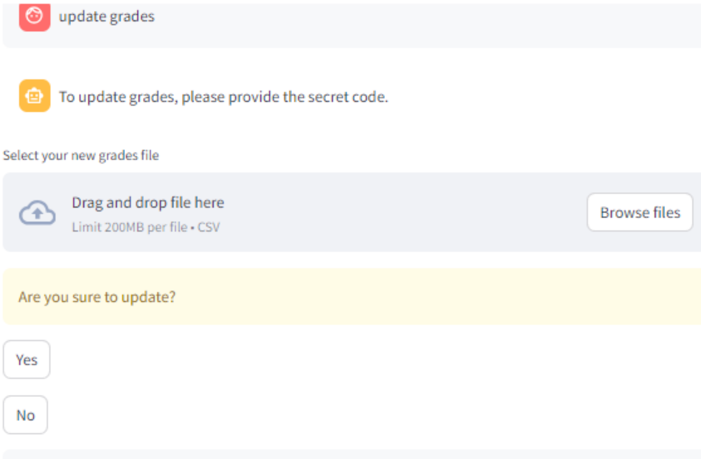


Course: CSE642 - GenAI

Professor: Sridharan Muthuswamy

Students: Lam Dao & Fang Wang

Released time: February 17th, 2024


update grades

To update grades, please provide the secret code.

Select your new grades file

Drag and drop file here
Limit 200MB per file • CSV

Are you sure to update?

Scripts for demonstration:

- Could you tell me about San Jose State University?
- Who is president of CSTU?
- Tell me about the CSTU
- What programs are offered by CSTU?
- Give me the link for reference about MSCSE at CSTU.
- List of courses in Spring 2024 at CSTU.
- Tell me about GenAI course at CSTU?
- Who is the professor of this course and what is his email?
- How about time schedule and project presentation of this class?
- Could you help me to register?
- I select the top 3 courses in above list.
- How about the student fee for these courses?
- Can I get my registration records?
- Can I get my course grades?
- Hi, I am professor Sridharan.
- I would like to update grades for GenAI course.
-

3.3 Performance Evaluation:

3.3.1 Evaluation method criteria:

Conduct comprehensive testing to evaluate accuracy, response time, task completion rate and make refinements based on results (Semantic similarity, text search, sentence similarity tasks).

By considering these evaluation, we can gain a comprehensive understanding of the RAG-based GPT model's performance and identify areas for further refinement and enhancement. Involving stakeholders, subject matter experts, and end-users in the evaluation process will ensure the chatbot meets the specific needs and expectations of the California Science and Technology University.

Key evaluation criteria:

1. Chatbot Functionality:
    - Accuracy of Responses: Evaluate how accurately the chatbot responds to user queries related to CSTU. Test it with a diverse set of questions covering different domains like courses, policies, resources, etc.
    - Handling of Ambiguity: Assess how well the chatbot handles ambiguous or unclear queries. It should be able to seek clarification or provide relevant suggestions when faced with ambiguity.
    - Ensure the chatbot functions as intended by testing various user queries and scenarios to verify accurate responses. Check if the chatbot handles different types of questions, prompts, and user inputs appropriately.
2. User Experience/feedback:
    - Conversational Flow: Evaluate the naturalness and coherence of the chatbot's responses. The conversation should flow smoothly and logically, with responses that are contextually relevant and easy to understand.
    - Response Time: Measure the response time of the chatbot. Responses should be generated promptly to enhance user engagement and satisfaction.
    - User Interface Design: Assess the design and usability of the chatbot interface, if applicable. Ensure it is user-friendly, accessible, and provides a seamless experience for CSTU's students, faculty, and staff.
3. RAG Model Performance:
    - Effectiveness of Retrieval: Evaluate how well the RAG model retrieves relevant information from the knowledge base. This is crucial for ensuring that the chatbot provides accurate and contextually appropriate responses.
    - Generation Quality: Assess the quality of the generated responses. Check for coherence, relevance, clarity, and factual consistency in the language used.
4. Knowledge Base Updates:
    - Frequency of Updates: Evaluate how frequently the knowledge base is refreshed with new information. A regularly updated knowledge base is essential to ensure the chatbot remains current and accurate.
5. Security and Privacy:
    - Data Security: Ensure that user data is handled securely, and the chatbot complies with privacy regulations, such as FERPA (Family Educational Rights and Privacy Act).
    - Authentication: If the chatbot involves sensitive information, evaluate the effectiveness of any authentication mechanisms implemented.

6. Scalability:
   - Performance under Load: Test how well the chatbot performs under varying levels of user load. Ensure it remains responsive and efficient during peak usage periods.
7. Error Handling:
   - Graceful Error Handling: Assess how the chatbot handles errors or situations where it cannot provide a satisfactory response. It should gracefully communicate limitations or seek alternative ways to assist the user.
8. Accuracy and Relevance:
   - Evaluate the accuracy of responses by comparing chatbot answers with expected outcomes. Assess whether the chatbot provides relevant information by using test cases and sample conversations to validate correctness.
9. Additional Considerations:
   - Alignment with project goals: Ensure the evaluation metrics and criteria are aligned with the specific objectives defined for the chatbot project.
   - Baselining and comparison: Compare the performance of your customized GPT model with a baseline (e.g., pre-trained GPT model without RAG) or other chatbots serving similar purposes.
   - Long-term evaluation: Continuously monitor and evaluate the chatbot's performance over time to track its effectiveness and identify areas for improvement.

3.3.2 Evaluation of embedding models:

	CSTU-Embedding-Model based on Word2Vector	**OpenAI Text-embedding-ada**
Functionality	-Limited purpose	-General-purpose,
Quality	-Depend on the size and diversity of the training corpus, not specifically excel on text search and sentence similarity.	-Outperform on semantic search and sentence similarity. - High-quality text embeddings.
Dimensionality	-Vector size: 300 dimensions by default, customizable	-Vector size: 1536 dimensions The higher dimensionality can potentially capture more nuanced semantic relationships.
Processing rate	-Faster	-Slower
Dependancy	-Integrated, offline use without the need for external services	-Reliance on API calls
Context	- Capture semantic meaning for specific context, not be able to produce a vector if a word was not	-Good understanding of context, semantics, and syntax. It can capture complex

	present in the training data, not account for sub-word information -Struggle with capturing complex contextual nuances. It tends to represent words based on co-occurrence patterns.	relationships between words and phrases. -Longer context length up to 8192 tokens making it more suitable for working with long documents
Domain Flexibility	-Open/trainable or fine-tunable on own domain-specific data, more suitable for certain specialized tasks	-Closed/not customizable or fine-tunable for specific domains
Scaling/Adaptability	-Scaling or knowledge changes requires creating new embedding	
Resource	-Lightweight, requiring less storage and computational resources, suitable for resource-constrained environments.	It's a larger model, requiring more resources.
Cost	-Free, cost-effective	-Paid tokens

3.3.3 Benifits

- Contextual Relevance: introducing contextually rich data into the generative process, offering more nuanced and situationally appropriate responses than a generalized LLM might provide.
- Access to Fresh Data: can access the most recent information, which may be fresher than the data on which the LLM was trained, keeping the model's outputs current.
- Cost-Effective Data Updates: The data can be updated continually and cost-effectively, avoiding the need for expensive and time-consuming retraining of the LLM.
- Quick and Affordable Implementation: Implementing is relatively cheaper and faster than fine-tuning LLMs, making it an efficient solution for enhancing model performance.
- Governance and Control: allowing for governance over the LLM's responses by implementing access controls and entitlements, ensuring appropriate use based on the user's identity.
- Improved Factual Accuracy: By tapping into external knowledge sources, it mitigates factual inaccuracies and hallucinations that standard LLMs might produce.
- Domain-Specific Tailoring: It can integrate domain-specific information, allowing it to tailor outputs to specialized fields or user requirements.
- Increased Explainability: The ability to retrieve source information for generated text enhances the transparency of the AI's decision-making process and aids in understanding the basis for its outputs.

- Scalability: smaller datasets yet still leverage the extensive knowledge available in external sources, making them potentially more scalable than models reliant solely on limited training corpora.

3.4 Challenges and improvement solutions

1. Complexity of Combining Retriever and Generator:

- Challenge: Integrating a retriever and a generator into a unified model increases complexity, demanding significant computational resources.
- Solution: Train the retriever and generator components separately to simplify the development process and reduce the computational burden. This approach allows for more focused optimization of each component before their integration.

2.Context Retention in Extended Conversations:

- Challenge: The system may struggle to maintain context or remember user inputs during longer interactions, leading to less coherent responses.
- Solution: Implement mechanisms to retain and prioritize key information throughout the conversation, ensuring the model can reference important context when generating responses.

3.Limitations Due to API Restrictions:

- Challenge: Free API accounts often come with limitations on tokens and functionalities, restricting the model's capabilities.
- Solution: Utilize open-source tools and datasets to fine-tune your model, allowing for customization and overcoming API limitations without incurring additional costs.

4. Resource Constraints:

- Challenge: Limited computational and storage resources can hinder the performance and scalability of RAG implementations.
- Solution: Leverage cloud technologies, which offer scalable computing resources and storage solutions, to accommodate the demands of RAG models efficiently.

5. Accuracy and Appropriateness of Responses:

- Challenge: The chatbot may generate responses that are inaccurate or inappropriate due to gaps in the knowledge base.
- Solution: Continuously expand and update the knowledge base with high-quality, diverse information to improve the accuracy and appropriateness of the chatbot's responses.

6. API Service Failures:

- Challenge: Occasional failures of API services can disrupt the functionality of RAG systems.
- Solution: Implement graceful error handling, fallback mechanisms, and caching strategies to maintain service continuity during API downtimes.

7. Inconsistencies Between Retrieved and Generated Content:

- Challenge: Merging content from two models can result in inconsistencies, affecting the coherence of the responses.
- Solution: Establish consistency checks and coherence mechanisms to ensure that the retrieved information aligns with the generated content, maintaining a seamless integration of both components.

8. Computational Expense of Running Models:

- Challenge: Operating both retrieval and generative models simultaneously can be computationally demanding.
- Solution: Optimize the models through algorithmic improvements, utilize hardware acceleration (e.g., GPUs), and explore model distillation techniques to reduce the computational load without sacrificing performance.

By addressing these challenges with targeted solutions, the effectiveness and efficiency of Retrieval-Augmented Generation systems can be significantly enhanced, leading to more reliable, coherent, and user-friendly conversational agents.

Summarization

Lessons learned

- **Starting with Word2Vec:**
    - Word2Vec is a good starting point due to its simplicity and flexibility.
    - It allows straightforward implementation and can handle various text data types.
- **Consider OpenAI embedding model:**
    - For complex tasks, consider using OpenAI Text-embedding-ada-002.
    - This model is well-suited for large datasets and sophisticated text understanding.
    - It’s ideal for tasks requiring advanced natural language processing capabilities.
- **Secure User Information Handling:**
    - Ensure secure storage and handling of user information.
    - Comply with privacy regulations to protect users’ data and privacy.
- **Regular Knowledge Base Updates:**

  - Keep the knowledge base up-to-date with the latest information.
  - This ensures accurate and current responses to user queries.
- **Personalization Opportunities:**
  - Explore opportunities to personalize interactions based on user preferences and history.
  - Enhance user engagement and satisfaction through tailored responses.

Future development

- **Natural Language-Based Login and Security:**
  - Implement a secure login system using natural language processing for user authentication and access control.
- **Chat-Based Career Advisor:**
  - Develop a chatbot feature that pulls information from student records.
  - Provide personalized career advice and guidance to students.
- **Marketing Engine:**
  - Create a marketing engine that collects data from user interactions.
  - Generate personalized emails or messages for referrals and promotions.
- **Knowledge Base Update Automation:**
  - Implement a system to automatically update the chatbot’s knowledge base.
  - Gather new information from user interactions and external sources.
- **Chatbot-Based Course and Student Management:**
  - Develop functionality for managing course schedules, student records, and other administrative tasks.
  - Streamline administrative processes for students and staff through chatbot interactions.