Building Application using LLMs - Case Studies

- **Practical Case Studies using Cohere**
- **Build a Chain for Documents with RAG Retriever**
  - Domain Specific ChatBOT

Ram N Sangwan

Practical Case Studies - Apps using Generative AI

Case Study 1

Sahayak AI

- SahayakAI is an AI-driven software tool designed to transform the way legal documents are translated and understood in India.
- Legal documents in India are predominantly in English, which presents a formidable challenge in legal understanding and literacy for the vast majority of the population.

Case Study 1

Sahayak AI

- The existing solutions, such as human translators and generic translation software, have notable limitations ranging from being too expensive to or lack of understanding of legal terminologies.
- To solve this, this AI solution translates English 'legalese' into Hindi.
- It uses Cohere's multilingual AI model Aya to achieve this.

Case Study 2

AI-Powered **Charting**

- Healthcare professionals spend 16 hours per week on paperwork and administrative tasks.
- One out of every five days is dedicated to post-consultation paperwork and referencing HCPCS codes for insurance companies.
- In the US, an inefficient administration and overhead in healthcare squander $600 billion annually, accounting for 30% of the total healthcare expenditure.
- Leveraging Cohere's Summarize API and the Coral Chat API, along with OpenAI's Whisper model, Charting enables physicians and patients to record conversations during clinical visits or video call sessions.

Case Study 2

AI-Powered **Charting**

- The Whisper model generates transcriptions by accessing the device's microphone, and then
- Cohere's Summarize API produces insightful bullet-point summaries of the full transcriptions. Another key feature of Charting is its Search function.
- Using Cohere's Chat API, physicians and patients can search the internet and obtain ad-free answers.
- To build trustworthiness, the top 5 sources of these answers, along with their original URLs, are displayed for further research

Case Study 3

Chatbot for Knowledge Management at Morgan Stanley Wealth Management

- With the help of LLM, Morgan Stanley is changing how its wealth management personnel locate relevant information.
- The LLM powers an internal-facing chatbot that performs a comprehensive search of wealth management content and effectively unlocks the cumulative knowledge of Morgan Stanley Wealth Management.
- LLM's extraordinary capability to access, process, and synthesize content almost instantaneously is leveraged by Morgan Stanley's internal-facing chatbot.
- The chatbot is trained on the company's vast content repository, which covers insights on capital markets, asset classes, industry analysis, and economic regions around the globe.

Case Study 4

RAG Fusion with Cohere and Weaviate

- RAG Fusion generates variations of the user's question under the hood.
- It retrieves matching documents for each variation and performs a fusion between them with re-ranking.
- A variation may match better into a small DB than the original question.
- First a data enrichment technique is used: After QnA for the Cohere Command fine tuning, with some extra scripts that data is processed further for ingestion into the Weaviate platform.

Check More Use cases at https://lablab.ai/apps/tech/cohere/cohere

Case Study 5

Market AI

- Managing a hedge fund portfolio demands swift and informed trading decisions, a challenge compounded by the vast data influencing stock prices.
- Market AI leverage Cohere, Langchain, and Weaviate.
- This solution is tailored for investors seeking to efficiently conduct research on publicly traded companies.
- It offers seamless access to critical information drawn from company filings and provides interactive capabilities to monitor current portfolio status.

Case Study 6

PyLibrarian

- Working with a new library, SDK, or API, software developers often waste hours hopelessly poring over a sea of scattered documentation pages to find the one syntax examples.
- Traditional LLM's knowledge pools are limited to their training data, so when they are asked about perhaps newer tech, they may be rendered useless, or even worse, hallucinate and spew nonsense.
- Pylibrarian grant LLM access to complete documentation for Python's most popular libraries using RAG architecture.

https://lablab.ai/event/cohere-coral-hackathon/new-grads/pylibrarian

Case Study 6

PyLibrarian

- Pylibrarian was built by processing, embedding (using cohere.embed), and storing documentation pages into Weaviate's vector database.
- Upon a user query, it can search for the most relevant pages of documentation to that query.
- Using Cohere's chat endpoint's document mode, the chatbot synthesizes a response citing the documents, leading to far more consistent, grounded responses.

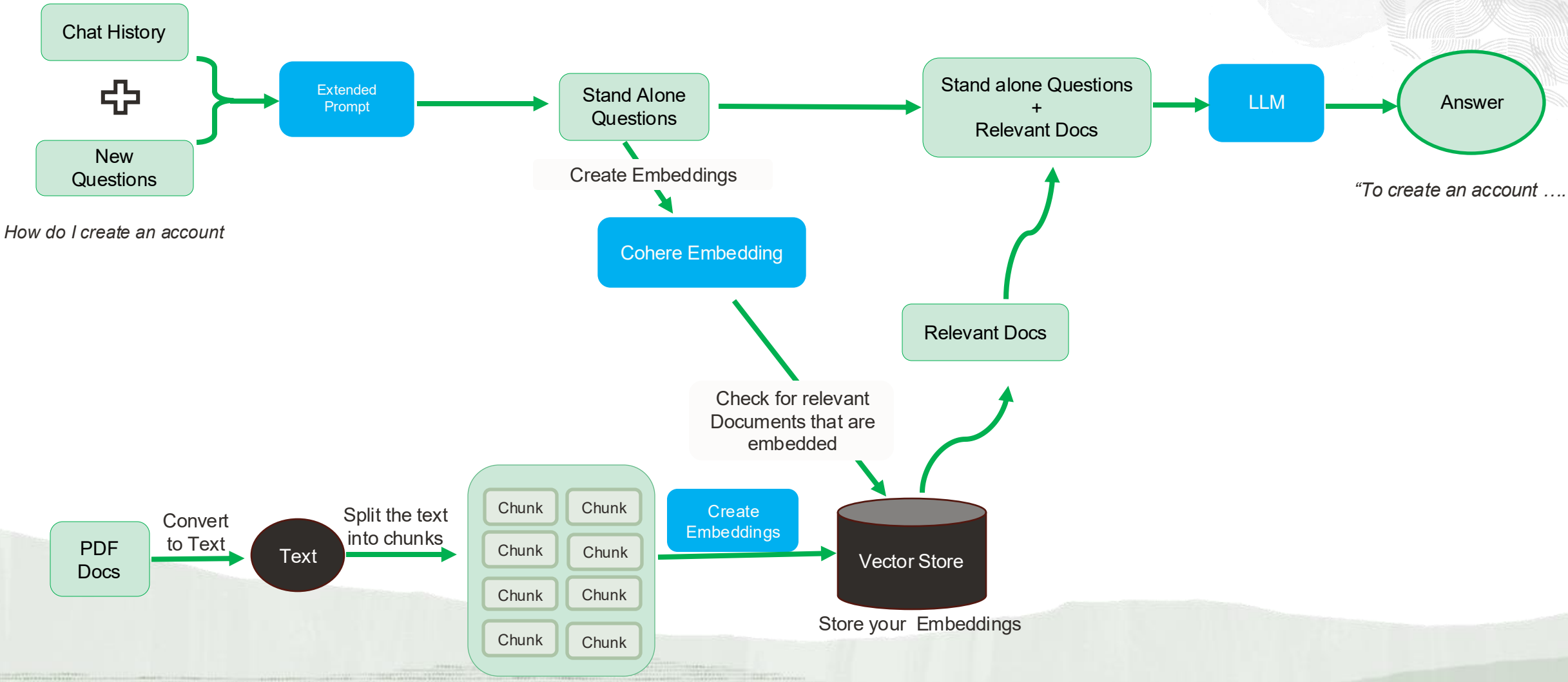
https://github.com/bert-luo/docbot

https://lablab.ai/apps/tech/cohere/cohere

https://github.com/NityaSG/Stock-AI

Building ChatBot for a set PDF documents.

Our Use Case - The Reference Project For Study



Creating a Chatbot for PDF Files

- LangChain is a framework that makes it easier to build scalable AI/LLM apps and chatbots.
- Pinecone is a vectorstore for storing embeddings and your PDF in text to later retrieve similar docs.
- Study a Project that creates a ChatBot based on existing set PDF Files.
- Fork the repo `https://github.com/Sangwan70/cohere-chatbot-pdf-langchain.git`
- Now clone or download the ZIP in the VM provided

```
git clone https://github.com/Sangwan70/cohere-chatbot-pdf-langchain.git
```

```
info@MacBook-Pro Day 5 % git clone https://github.com/Sangwan70/cohere-chatbot-pdf-langchain.git

Cloning into 'cohere-chatbot-pdf-langchain'...
remote: Enumerating objects: 52, done.
remote: Counting objects: 100% (52/52), done.
remote: Compressing objects: 100% (43/43), done.
remote: Total 52 (delta 3), reused 52 (delta 3), pack-reused 0
Receiving objects: 100% (52/52), 1.59 MiB | 4.04 MiB/s, done.
Resolving deltas: 100% (3/3), done.
info@MacBook-Pro Day 5 %
```

Create a Chatbot for PDF Files

- Install packages. First run

  ```
  npm install yarn -g
  ```

- To install yarn globally (if you haven't already). Then run:

  ```
  yarn install
  ```

- After installation, you should now see a node_modules folder.

Set up your .env file

Copy .env.example into .env Your .env file should look like this:

```
COHERE_API_KEY=

PINECONE_API_KEY=

PINECONE_ENVIRONMENT=

PINECONE_INDEX_NAME=
```

- Visit Cohere to retrieve API keys and insert into your .env file.
- Visit pinecone to create and retrieve your API keys, and also retrieve your environment and index name from the dashboard.

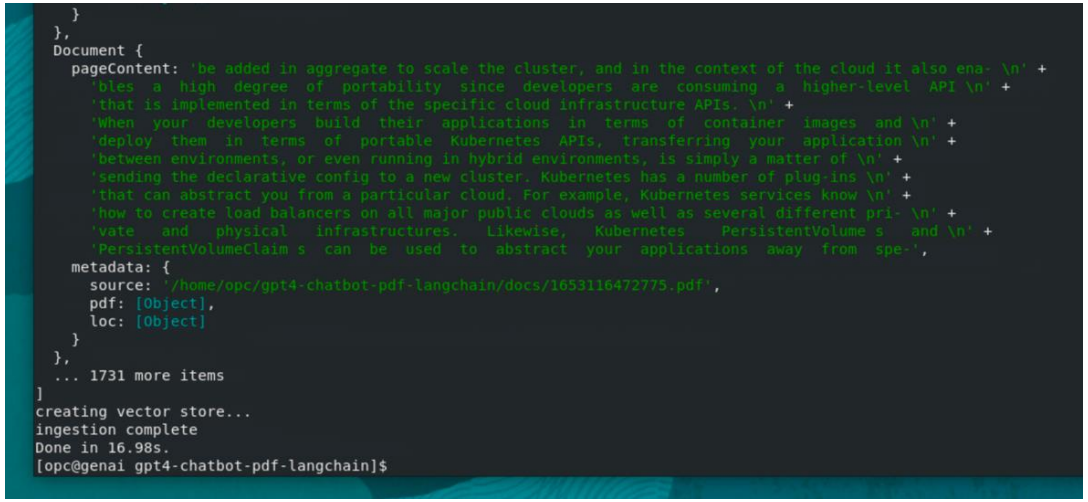
Create a Chatbot for PDF Files

- In the config folder, replace the *PINECONE_NAME_SPACE* with a namespace where you'd like to store your embeddings on Pinecone when you run `npm run ingest`.
- This namespace will later be used for queries and retrieval.
- In `utils/makechain.ts` chain change the QA_PROMPT for your own use case.
- Please verify outside this repo that you have access to **command** LLM, otherwise the application will not work.

Convert your PDF files to embeddings

- Inside docs folder (Create the Folder if needed), add your pdf files or folders that contain pdf files.

```
yarn run ingest
```



- To 'ingest' and embed your docs.
- Check Pinecone dashboard to verify your namespace and vectors have been added.

Run the app

- Once you've verified that the embeddings and content have been successfully added to your Pinecone, you can run the app with

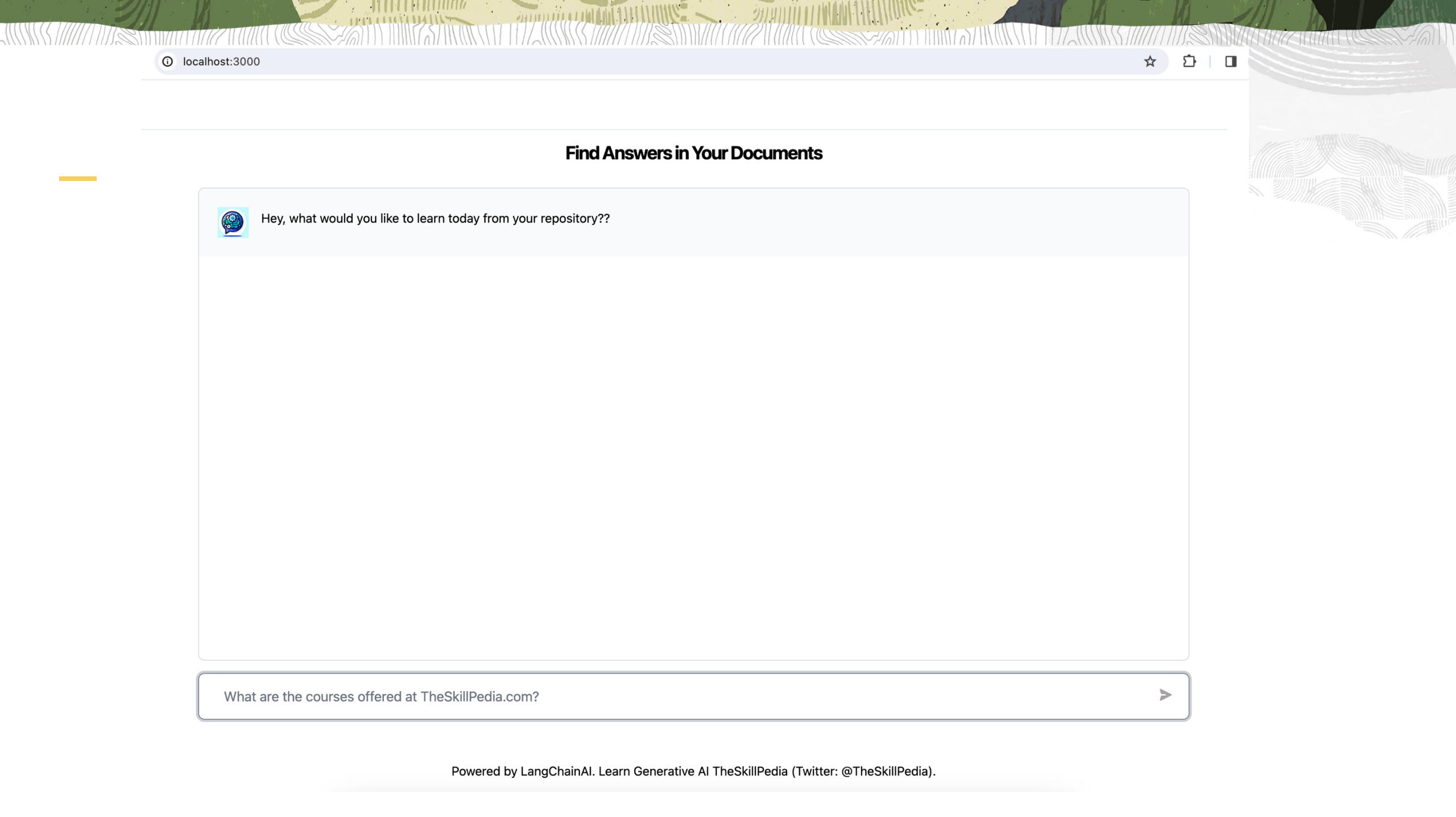
```
npm run dev
```

```
info@MacBook-Pro cohere-chatbot-pdf-langchain % npm run dev

> gpt4-langchain-pdf-chatbot@0.1.0 dev
> next dev

(node:9116) [DEP0040] DeprecationWarning: The `punycode` module is deprecated. Plea
(Use `node --trace-deprecation ...` to show where the warning was created)
(node:9117) [DEP0040] DeprecationWarning: The `punycode` module is deprecated. Plea
(Use `node --trace-deprecation ...` to show where the warning was created)
ready - started server on 0.0.0.0:3000, url: http://localhost:3000
info  - Loaded env from /Users/info/GenAI/Day 5/cohere-chatbot-pdf-langchain/.env
event - compiled client and server successfully in 1256 ms (173 modules)
wait  - compiling...
event - compiled successfully in 36 ms (139 modules)
wait  - compiling / (client and server)...
event - compiled client and server successfully in 818 ms (1323 modules)
```

- To launch the local dev environment, and then type a question in the chat interface.
- Launch the browser inside VM and open http://localhost:3000
- Or, use the public IP of the instance with port 3000.


localhost:3000
Find Answers in Your Documents
Hey, what would you like to learn today from your repository??
What are the courses offered at TheSkillPedia.com?
Powered by LangChainAI. Learn Generative AI TheSkillPedia (Twitter: @TheSkillPedia).

Thank You