



Building Application using Gen AI - Case Studies

- **Practical Case Studies using Cohere Models**
- **Build a Chain for Documents with RAG Retriever**
 - Domain Specific ChatBOT
- **Automate AI Workflows with AutoML & LangChain**

Ram N Sangwan



Practical Case Studies - Apps using Generative AI



Case Study 1

Sahayak AI

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



Case Study 1

Sahayak AI

- The existing solutions, such as human translators and generic translation software, have limitations either being too expensive 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 **Summarize** Endpoint and 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.
- 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 with Cohere Command, and some extra scripts that data is processed further for ingestion into the Weaviate Vector Database.



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 over a sea of scattered documentation pages to find the one syntax example.
- Traditional LLM's knowledge pools are limited to their training data, so when they are asked about perhaps newer tech, they may hallucinate and spew nonsense.
- Pylibrarian grant LLM access to complete documentation for Python's most popular libraries using RAG.

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

Case Study 6


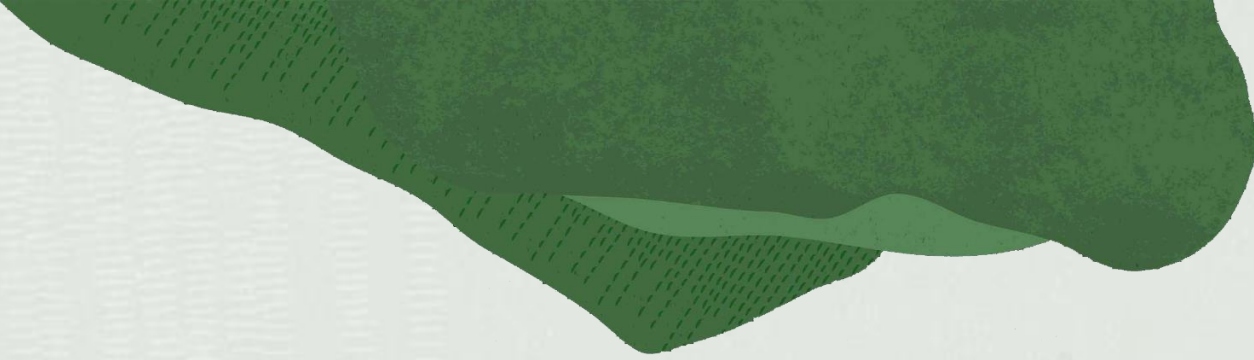
PyLibrarian

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

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

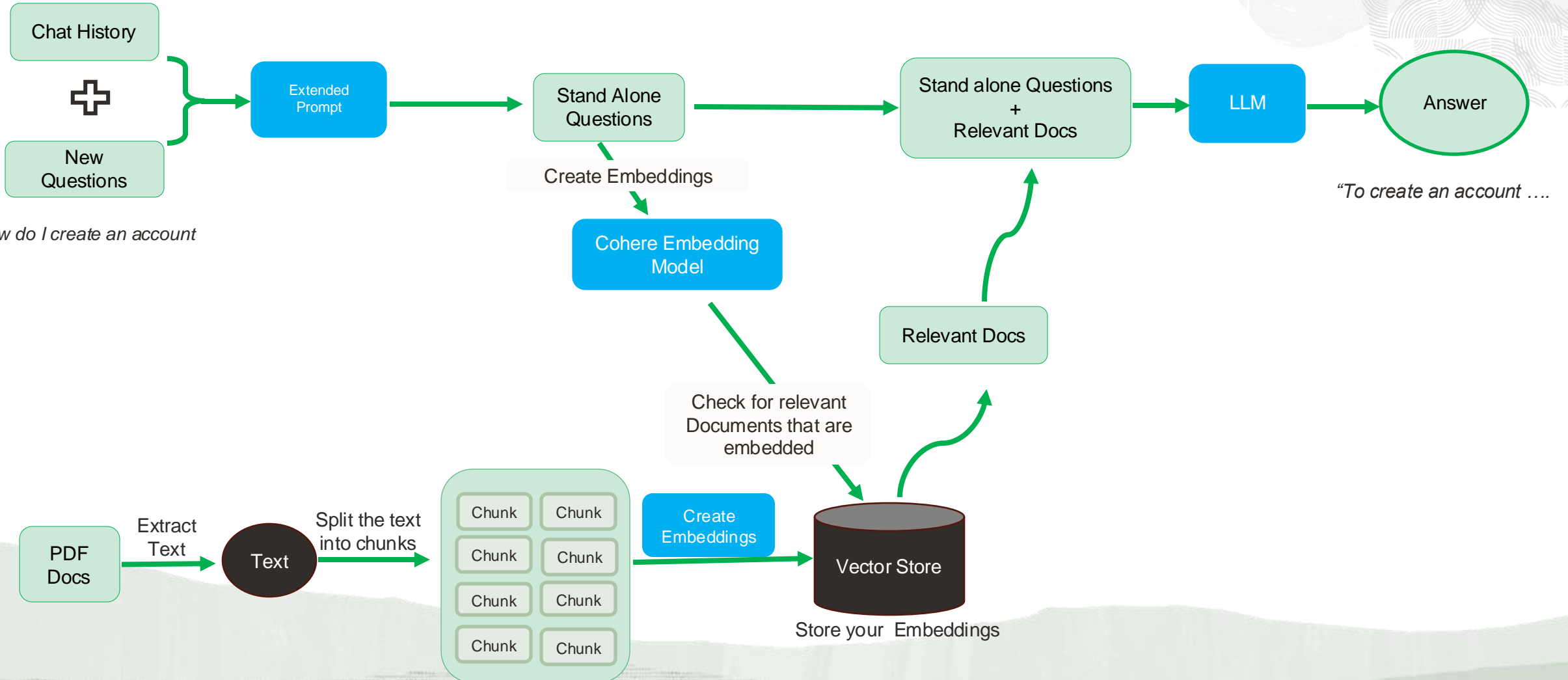
<https://lablab.ai/apps/tech/cohere/cohere>

<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 vector-store for storing embeddings and to retrieve similar docs later.
- 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 install **yarn** globally with

```
npm install yarn -g
```
- Then run to install packages from `package.json`

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

Set up your .env file

Create a .env file with API Keys:

```
COHERE_API_KEY=  
PINECONE_API_KEY=  
PINECONE_ENVIRONMENT=  
PINECONE_INDEX_NAME=
```

```
COHERE_API_KEY=tRGKPmSqy0QKq2f2LbB120INzMd1wYbP9s896Dl3  
  
# Update these with your pinecone details from your dashboard.  
# PINECONE_INDEX_NAME is in the indexes tab under "index name"  
# PINECONE_ENVIRONMENT is in indexes tab under "Environment".  
PINECONE_API_KEY=e89ad6e0-3484-4ea3-b395-8c9a3fa5b184  
PINECONE_ENVIRONMENT=gcp-starter  
PINECONE_INDEX_NAME=skillpedia
```

- Visit www.cohere.com to retrieve API keys.
- Visit [pinecone](https://pinecone.io) 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.
- This namespace will later be used for queries and retrieval.
- In `utils/makechain.ts` chain change the QA_PROMPT for your own use case.

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
```

```
}
},
Document {
  pageContent: 'be added in aggregate to scale the cluster, and in the context of the cloud it also ena- \n' +
    'bles a high degree of portability since developers are consuming a higher-level API \n' +
    'that is implemented in terms of the specific cloud infrastructure APIs. \n' +
    'When your developers build their applications in terms of container images and \n' +
    'deploy them in terms of portable Kubernetes APIs, transferring your application \n' +
    'between environments, or even running in hybrid environments, is simply a matter of \n' +
    'sending the declarative config to a new cluster. Kubernetes has a number of plug-ins \n' +
    'that can abstract you from a particular cloud. For example, Kubernetes services know \n' +
    'how to create load balancers on all major public clouds as well as several different pri- \n' +
    'vate and physical infrastructures. Likewise, Kubernetes PersistentVolume s and \n' +
    'PersistentVolumeClaim s can be used to abstract your applications away from spe-',
  metadata: {
    source: '/home/opc/gpt4-chatbot-pdf-langchain/docs/1653116472775.pdf',
    pdf: [Object],
    loc: [Object]
  }
},
... 1731 more items
]
creating vector store...
ingestion complete
Done in 16.98s.
[opc@genai gpt4-chatbot-pdf-langchain]$
```

- To '**ingest**' and embed your docs.

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)
```

- Launch the browser inside VM and open <http://localhost:3000>
- Or, use the public IP of the instance with port 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?





Automate AI Workflows with AutoGPT & LangChain

Automate AI Workflows with AutoML & LangChain

This can significantly enhance the efficiency and scalability of your ML operations.
Here's a high-level approach

Define Automation Goals:

- And specific processes you want to automate within your AI workflow. Identify the steps that are repetitive or require manual intervention.

Automate AI Workflows with AutoML & LangChain

Integrate with Automation Tools:

- Utilize automation tools compatible with your MLOps stack.
- For instance, consider using Oracle's AutoML and LangChain.
- Explore other tools like Terraform, Ansible, or Oracle Cloud Infrastructure's built-in automation features to manage infrastructure provisioning and configuration.

Automate AI Workflows with AutoML & LangChain

Data Management Automation:

- Automate the ingestion and pre-processing of data using OCI Data Flow.
- Configure these tools to integrate with your data sources, perform data transformations, and load the data into your target database.

Model Training and Tuning:

- Design a workflow that includes your chosen machine learning libraries (e.g., TensorFlow, PyTorch) and integrates with AutoML.
- AutoML can automatically train and tune models based on your dataset and specified hyperparameters.

Automate AI Workflows with AutoML & LangChain

LangChain Integration:

- Configure LangChain to manage these components.
- Automate the deployment of models and pipelines using LangChain's automation features, ensuring seamless integration within your AI workflow.

Orchestrate with Workflow Managers:

- Implement workflow managers like Oracle's Cloud Automation Platform to define and orchestrate your end-to-end AI workflow.



Automate AI Workflows with AutoML & LangChain

Monitoring and Feedback Loop:

- Use tools like Prometheus, Grafana, or Oracle Monitoring to collect and analyse key metrics, logs, and events.

Version Control and Collaboration:

- Implement version control systems like Git to manage versioned datasets, code, and configuration files.

Testing and Validation:

- Ensure thorough testing and validation of your automated workflows.



Thank You

