

## Lab 14. Cohere and LangChain - a Chatbot for PDF Files

### Objective:

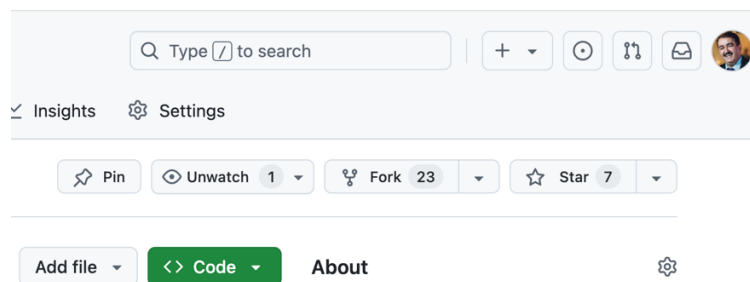
The project uses Chroma Vector Database running as container, NodeJS Typescript as backend and react as frontend for the ChatBot on existing set PDF Files.

**Disclaimer:** The Project is for study only. The code may not work as expected. For example the number of reference may not be exactly the same as the number of PDF Files. Also, there might be missing functionalities. You may use the project to understand the possibility of using LLM and RAG to build a custom chatbot.

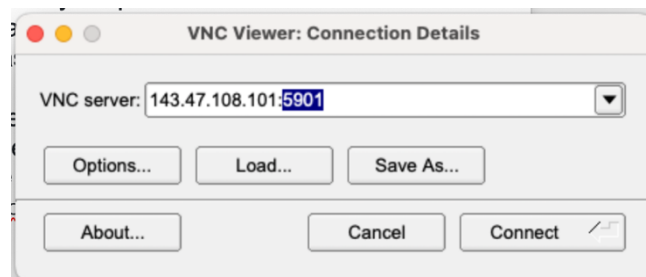
1. Create an account on Github, if not already existing and login.
2. Visit the repository page

<https://github.com/Sangwan70/cohere-chatbot-pdf-chroma.git>

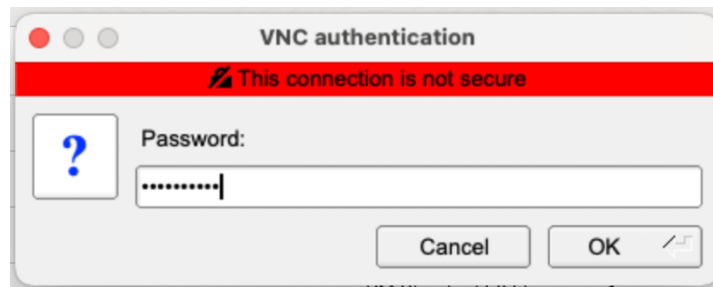
3. Click on **Fork** button to fork the repo to your Github account, so that you can change the code, upload your PDFs and add/finetune features.



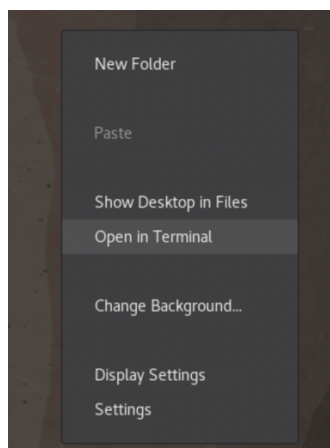
4. Inside **docs** folder (in your Github account), upload your pdf files or folders that contain pdf files. You will be ingesting these files as your knowledge base. Commit the changes. Ensure that the PDF files are not password protected and file names are not space separated.
5. Download and Install (if not already installed) **Tiger VNC Viewer** (for Windows and Mac) or **VNC Viewer** (Only for Windows). Start the Tiger VNC Viewer.
6. Enter the **Public IP** Address of your instance with **port number 5901** as given in screen shot below and click on **Connect**.



- Password for VNC user is "Oracle@123".



7. If the screen is locked, *click anywhere on the screen and press enter key.*
8. You will be logged in as "opc" user.
9. Launch the Terminal by right clicking on the Desktop.



10. Switch to root user with

```
sudo su -
```

11. Now clone the forked repo or download the ZIP to your OCI Instance

```
git clone https://github.com/<Your User ID>/cohere-chatbot-pdf-chroma.git
```

**For Example:**

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

```
File Edit View Search Terminal Help
[opc@server Desktop]$ sudo su -
Last login: Sun Nov  3 08:10:33 GMT 2024 on pts/0
[root@server ~]# git clone https://github.com/Sangwan70/cohere-chatbot-pdf-chroma.git
Cloning into 'cohere-chatbot-pdf-chroma'...
remote: Enumerating objects: 140, done.
remote: Counting objects: 100% (140/140), done.
remote: Compressing objects: 100% (120/120), done.
remote: Total 140 (delta 16), reused 128 (delta 9), pack-reused 0 (from 0)
Receiving objects: 100% (140/140), 29.09 MiB | 72.12 MiB/s, done.
Resolving deltas: 100% (16/16), done.
[root@server ~]#
```

12. Change to cloned repository and Install packages

```
cd cohere-chatbot-pdf-chroma
```

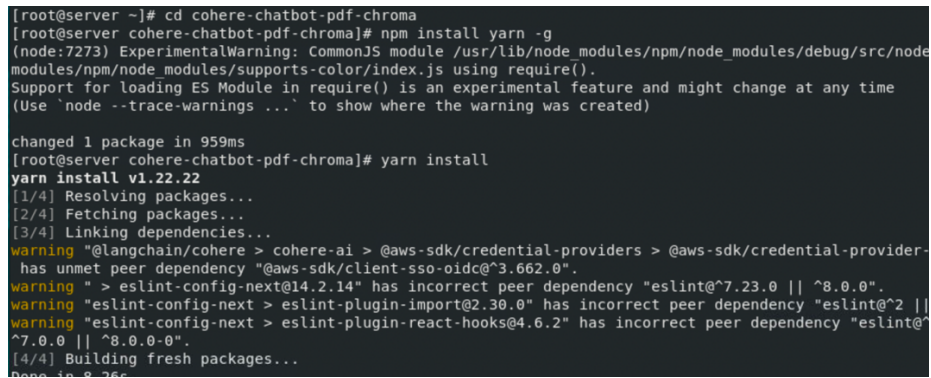
13. Install yarn globally with:

```
npm install yarn -g
```

14. Run the following command to install all packages from `package.json` file:

```
yarn install
```

*After installation, you should see a `node_modules` folder.*



```
[root@server ~]# cd cohere-chatbot-pdf-chroma
[root@server cohere-chatbot-pdf-chroma]# npm install yarn -g
(node:7273) ExperimentalWarning: CommonJS module /usr/lib/node_modules/npm/node_modules/debug/src/node_modules/npm/node_modules/supports-color/index.js using require().
Support for loading ES Module in require() is an experimental feature and might change at any time
(Use 'node --trace-warnings ...' to show where the warning was created)

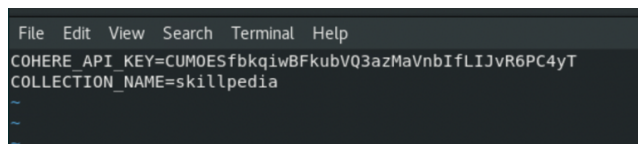
changed 1 package in 959ms
[root@server cohere-chatbot-pdf-chroma]# yarn install
yarn install v1.22.22
[1/4] Resolving packages...
[2/4] Fetching packages...
[3/4] Linking dependencies...
warning "@langchain/cohere > cohere-ai > @aws-sdk/credential-providers > @aws-sdk/credential-provider-
has unmet peer dependency "@aws-sdk/client-sso-oidc@^3.662.0".
warning " > eslint-config-next@14.2.14" has incorrect peer dependency "eslint@^7.23.0 || ^8.0.0".
warning "eslint-config-next > eslint-plugin-import@2.30.0" has incorrect peer dependency "eslint@^2 ||
warning "eslint-config-next > eslint-plugin-react-hooks@4.6.2" has incorrect peer dependency "eslint@
^7.0.0 || ^8.0.0-0".
[4/4] Building fresh packages...
Done in 8.26s
```

15. Create the `.env` file and paste the contents of the `DotEnv` file (shared by instructor) into this file.

```
# vi .env
```

```
# Copy the content from DotEnv File Provided.
```

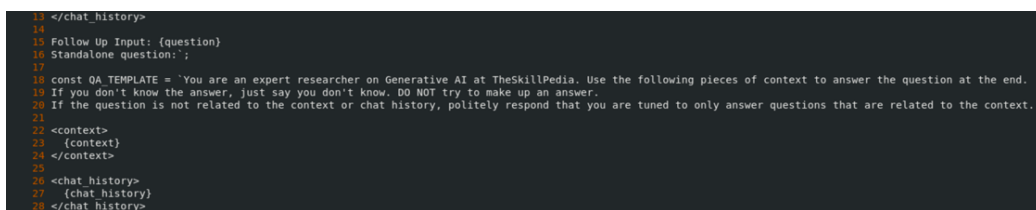
```
COHERE_API_KEY=
COLLECTION_NAME=skillpedia
```



```
File Edit View Search Terminal Help
COHERE_API_KEY=CUM0ESfbkqiWBFkubVQ3azMaVnbIfLIJvR6PC4yT
COLLECTION_NAME=skillpedia
~
~
~
```

16. In `utils/makechain.ts` chain, change the `QA_TEMPLATE` in line no. 18 (**All Details from Line No 5 to Line No 19**) for your own use case. This will be the **"Preamble"** or **"System Message"** and will affect the response from chat bot.

```
# vi utils/makechain.ts
```



```
13 </chat_history>
14
15 Follow Up Input: {question}
16 Standalone question: ''
17
18 const QA_TEMPLATE = `You are an expert researcher on Generative AI at TheSkillPedia. Use the following pieces of context to answer the question at the end.
19 If you don't know the answer, just say you don't know. DO NOT try to make up an answer.
20 If the question is not related to the context or chat history, politely respond that you are tuned to only answer questions that are related to the context.
21
22 <context>
23 {context}
24 </context>
25
26 <chat_history>
27 {chat_history}
28 </chat_history>
```

17. Run Chroma in the container with docker or **podman** (Podman is already installed on the Instance):

```
podman run -d -p 8000:8000 chromadb/chroma:0.4.15
```

```
[root@server cohere-chatbot-pdf-chroma]# podman run -d -p 8000:8000 chromadb/chroma:0.4.15
9d7bea82fce59863978ce6849dc2d7a0a59376e6847d855244c4c3fe7de4c51b
[root@server cohere-chatbot-pdf-chroma]#
```

## Convert your PDF files to embeddings

1. Run the script from **cohere-chatbot-pdf-chroma** directory to 'ingest' and embed your docs.

```
yarn run ingest
```

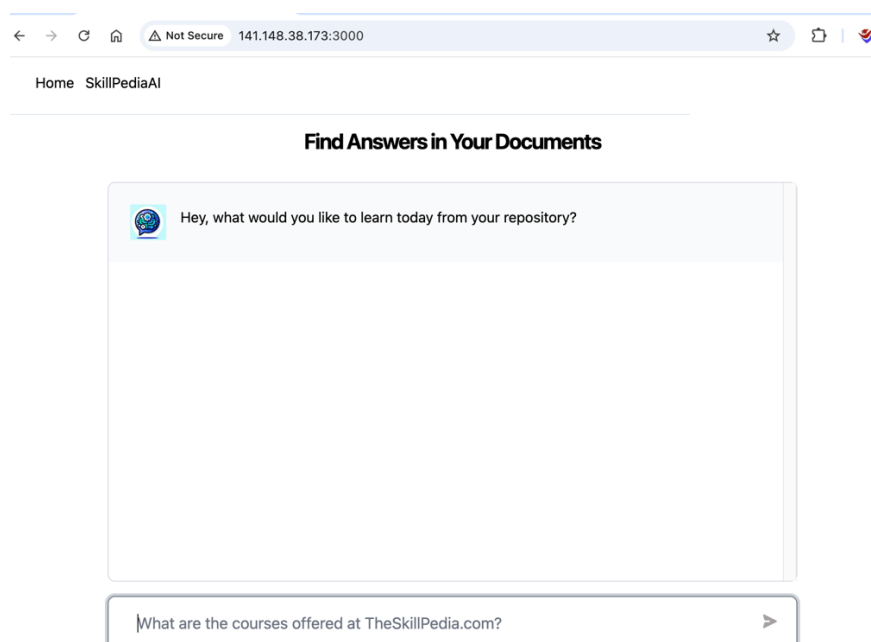
```
}
},
Document {
  pageContent: be added to aggregate to scale the cluster, and in the context of the cloud it also em... +
  ... a high degree of portability since developers are consuming a higher-level API to +
  that is implemented in terms of the specific cloud infrastructure APIs to +
  When your developers build their applications in terms of container images and to +
  deploy them in terms of portable Kubernetes APIs, transferring your application to +
  between environments, or even running in hybrid environments, is simply a matter of to +
  sending the declarative config to a new cluster. Kubernetes has a number of plug ins to +
  that can abstract you from a particular cloud, for example, Kubernetes requires know... +
  how to create load balancers on all major public clouds as well as several different API to +
  state and physical infrastructures. Likewise, Kubernetes PersistentVolume's and to +
  PersistentVolumeClaim's can be used to abstract your applications away from the +
  },
  metadata: {
    source: /home/opc/gpt4-chatbot-pdf-langchain/docs/1403214472773.pdf,
    pdf: [Object],
    loc: [Object]
  }
},
... 1731 more items
}
creating vector store...
ingestion complete
Done in 16.98s.
[opc@genai gpt4-chatbot-pdf-langchain]$
```

2. Run the app with

```
npm run dev
```

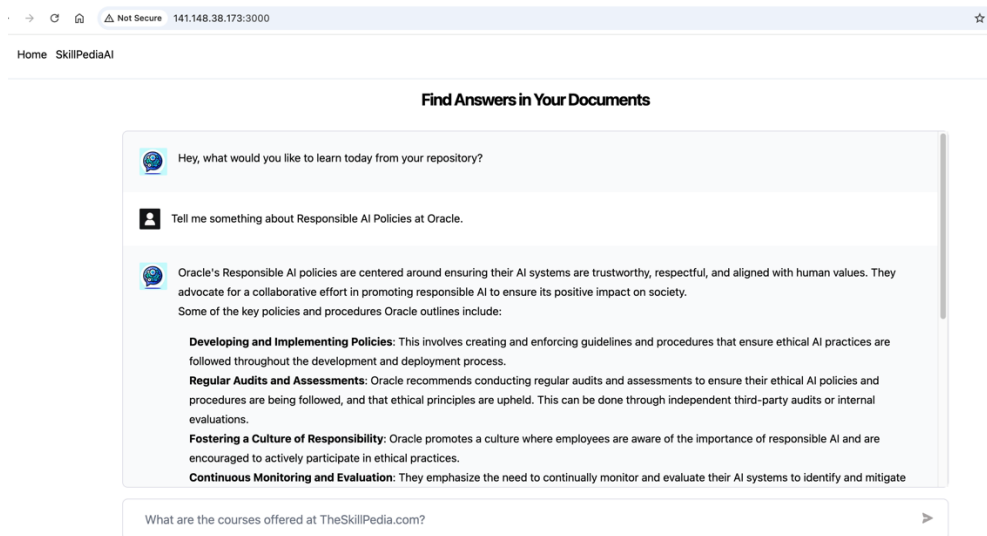
To launch the local dev environment, and then type a question in the chat interface.

Launch the browser and open <http://localhost:3000>, Or visit your browser with <http://<Public IP>:3000>.



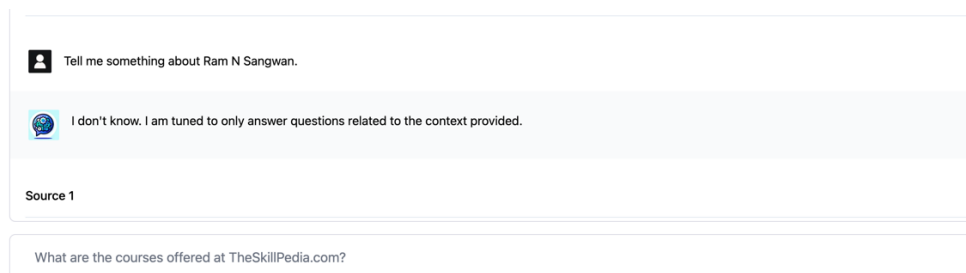
3. Ask a Question. Type a question about Responsible AI at Oracle. For example

"Tell me something about Responsible AI Policies at Oracle."



4. Try a question not available in PDF Documents:

"Tell me something about Ram N Sangwan."



**The Chatbot has politely refused to answer since we have configured the system prompt to respond only if the question is related to the context provided.**