User Manual

DocGenie

Shane Power - 21308533
Razvan Gorea - 21306373
Supervisor - Tomas Ward
Last Updated - 28/4/25

Abstract

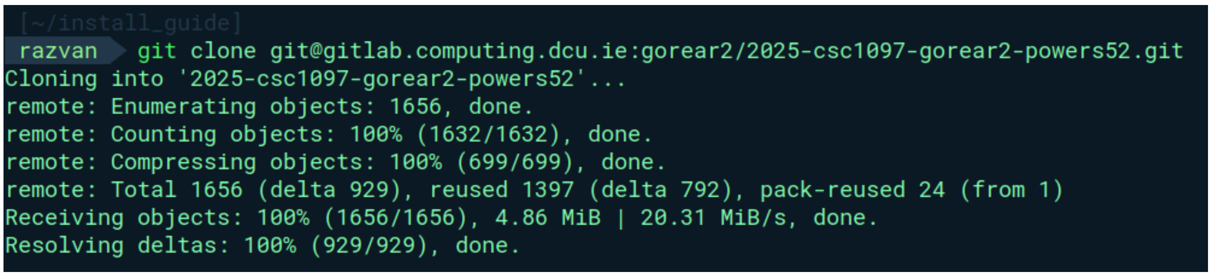
This is the user manual for the DocGenie 4th Year Project. It provides a comprehensive guide to the installation, setup, and usage of the project. All necessary commands and actions required for installation and operation are clearly detailed.

Additionally, a complete user guide covering all available functionality is included.
To assist with each step, relevant screenshots are provided, with the file paths displayed in grey for clarity and to avoid any confusion.

Installation Guide

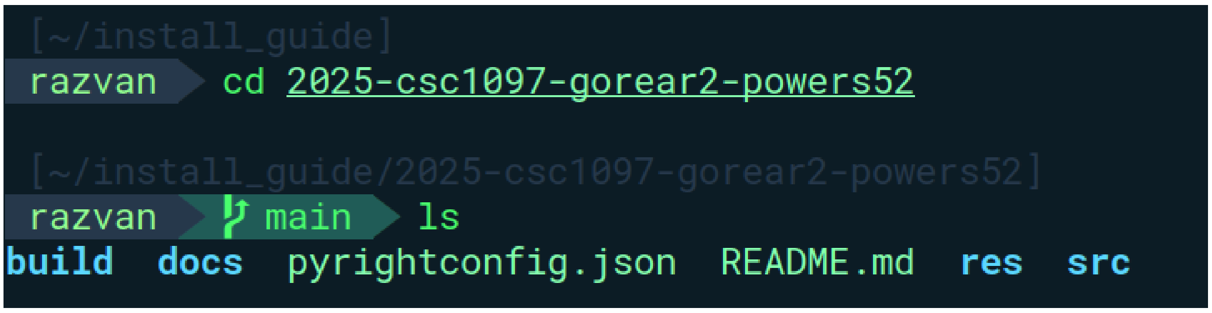
1. First clone the project repository from Gitlab (either through HTTPS or SSH):

   (This is the SSH approach)



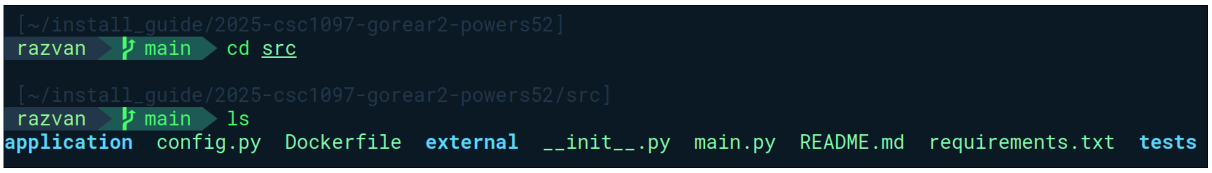
```
[~/install_guide]
razvan  git clone git@gitlab.computing.dcu.ie:gorear2/2025-csc1097-gorear2-powers52.git
Cloning into '2025-csc1097-gorear2-powers52'...
remote: Enumerating objects: 1656, done.
remote: Counting objects: 100% (1632/1632), done.
remote: Compressing objects: 100% (699/699), done.
remote: Total 1656 (delta 929), reused 1397 (delta 792), pack-reused 24 (from 1)
Receiving objects: 100% (1656/1656), 4.86 MiB | 20.31 MiB/s, done.
Resolving deltas: 100% (929/929), done.
```

2. Navigate into the project:



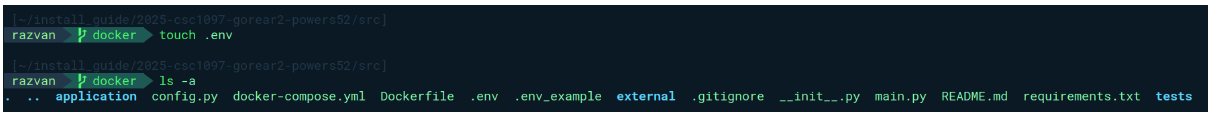
```
[~/install_guide]
razvan  cd 2025-csc1097-gorear2-powers52

[~/install_guide/2025-csc1097-gorear2-powers52]
razvan  main  ls
build  docs  pyrightconfig.json  README.md  res  src
```

3. Navigate into the `**src**` directory (you should see the following files and directories):



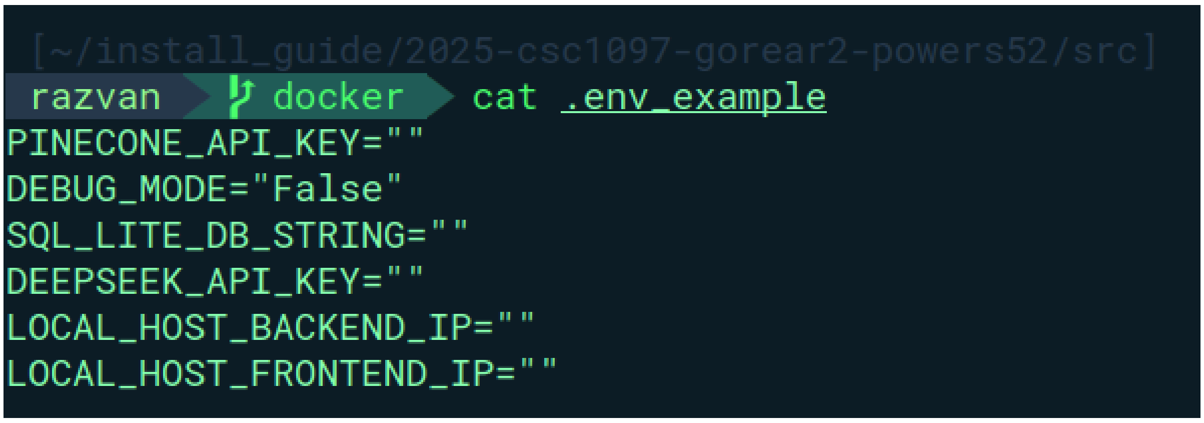
```
[~/install_guide/2025-csc1097-gorear2-powers52]
razvan  main  cd src

[~/install_guide/2025-csc1097-gorear2-powers52/src]
razvan  main  ls
application  config.py  Dockerfile  external  __init__.py  main.py  README.md  requirements.txt  tests
```

4. Create a **.env** file with all the necessary secret variables:



```
[~/install_guide/2025-csc1097-gorear2-powers52/src]
razvan  docker  touch .env

[~/install_guide/2025-csc1097-gorear2-powers52/src]
razvan  docker  ls -a
.  ..  application  config.py  docker-compose.yml  Dockerfile  .env  .env_example  external  .gitignore  __init__.py  main.py  README.md  requirements.txt  tests
```

(The necessary keys needed are provided in the .env_example file)



```
[~/install_guide/2025-csc1097-gorear2-powers52/src]
 razvan   docker  cat .env_example
PINECONE_API_KEY=""
DEBUG_MODE="False"
SQL_LITE_DB_STRING=""
DEEPSEEK_API_KEY=""
LOCAL_HOST_BACKEND_IP=""
LOCAL_HOST_FRONTEND_IP=""
```

5. To setup your Pinecone API Key visit https://www.pinecone.io/

   Once on Pinecone, set up an account, follow their instructions in creating the default `quickstart` index. Once completed, create a Pinecone API key, copy and paste the API key as the value for the `PINECONE_API_KEY` secret variable.

   Your `SQL_LITE_DB_STRING` secret variable should contain the path where your sqlite database will be stored.

   E.g. `SQL_LITE_DB_STRING="sqlite:///application/api/main.db"`

   Your `DEEPSEEK_API_KEY` secret variable is a Mistral API Key. Visit https://mistral.ai/ create an account with them. Create a Mistral API key, copy and paste it as the value for the `DEEPSEEK_API_KEY` secret variable. It's also recommended that you either have some API credit or pay as you go to prevent yourself from getting rate limited.

   Your `LOCAL_HOST_BACKEND_IP` is the local host address that points to the backend service.

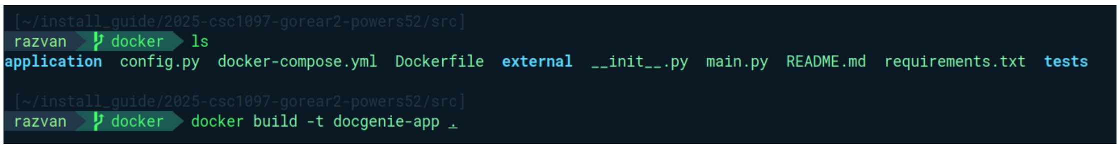
   E.g. `LOCAL_HOST_BACKEND_IP="0.0.0.0"(recommended)`

   Your `LOCAL_HOST_FRONTEND_IP` secret key should contain the value of the local host address where the Next.js application will run on.

   E.g. `LOCAL_HOST_FRONTEND_IP="http://localhost:3000" (recommended)`

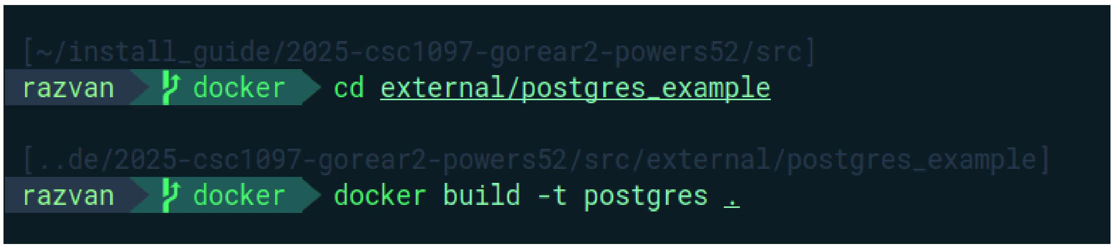
6. Once your .env file is finally created, we will now build all the necessary docker images to run the project:

- First, build the docgenie-app image with the following command:



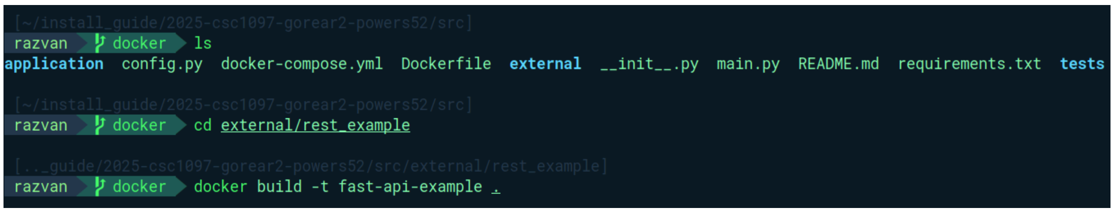
```
[~/install_guide/2025-csc1097-gorear2-powers52/src]
razvan  docker  ls
application  config.py  docker-compose.yml  Dockerfile  external  __init__.py  main.py  README.md  requirements.txt  tests

[~/install_guide/2025-csc1097-gorear2-powers52/src]
razvan  docker  docker build -t docgenie-app .
```

- Next, build the postgres image with the following command (from `**src**`):



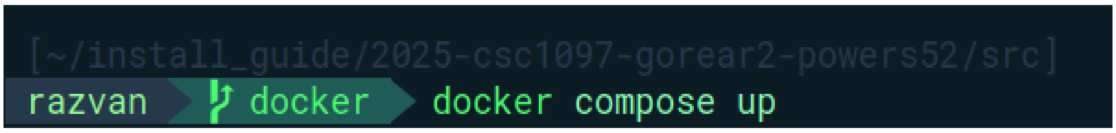
```
[~/install_guide/2025-csc1097-gorear2-powers52/src]
razvan  docker  cd external/postgres_example

[..de/2025-csc1097-gorear2-powers52/src/external/postgres_example]
razvan  docker  docker build -t postgres .
```

- Next, build the fast-api-example image with the following command (from `**src**`):



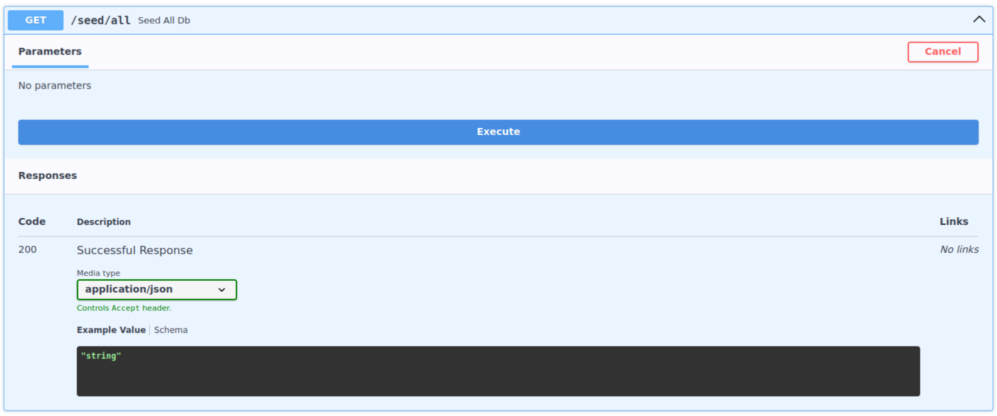
```
[~/install_guide/2025-csc1097-gorear2-powers52/src]
razvan  docker  ls
application  config.py  docker-compose.yml  Dockerfile  external  __init__.py  main.py  README.md  requirements.txt  tests

[~/install_guide/2025-csc1097-gorear2-powers52/src]
razvan  docker  cd external/rest_example

[..._guide/2025-csc1097-gorear2-powers52/src/external/rest_example]
razvan  docker  docker build -t fast-api-example .
```

7. Back in the `**src**` directory run the docker compose file:

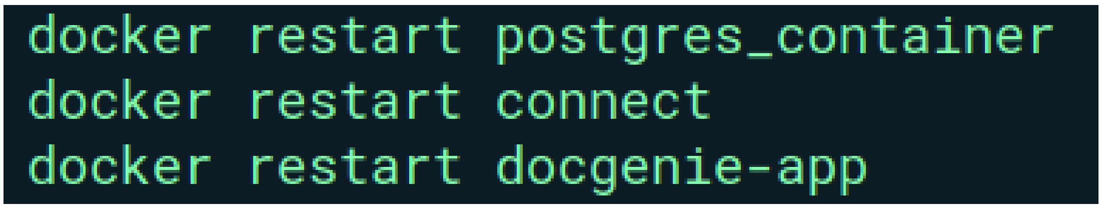


```
[~/install_guide/2025-csc1097-gorear2-powers52/src]
razvan  docker  docker compose up
```

8. Visit http://localhost:8888/docs, navigate to the to the `**/seed/all**` endpoint and trigger it:



9. Restart the `**postgres_container**`, `**connect**`, and `**docgenie-app**` containers **(in that order)** to ensure sure everything is properly initialized and connected to one another:



```
docker restart postgres_container
docker restart connect
docker restart docgenie-app
```

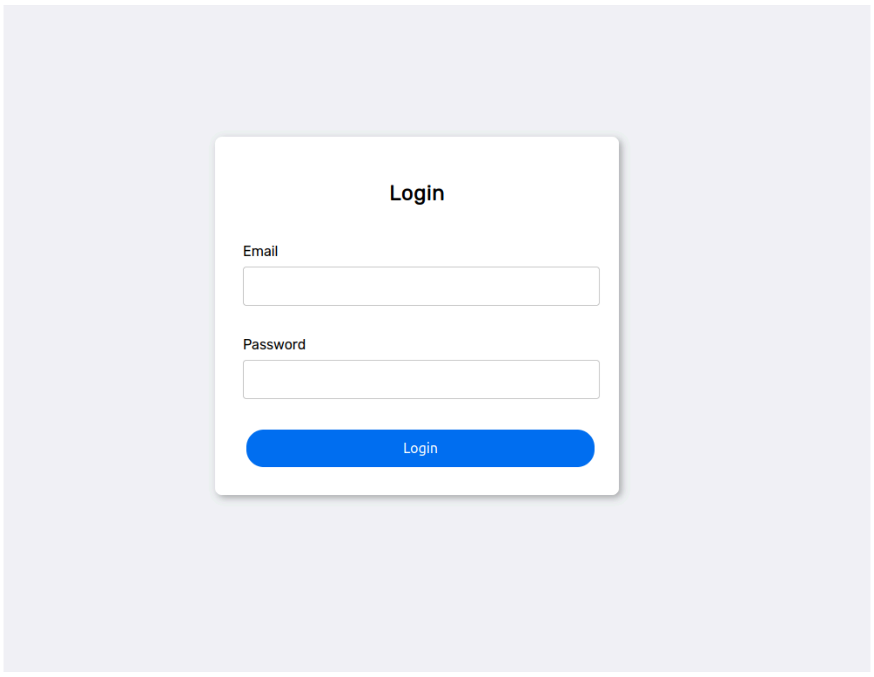
10. If the postges event listening isn't being picked up by docgenie, just restart the docgenie-app container.

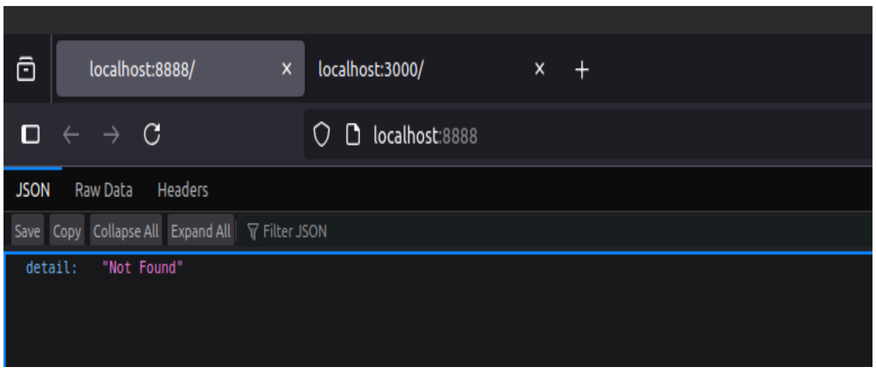
11. You can verify the services are running through visiting the following urls on your browser:

    http://localhost:8888/ and http://localhost:3000/ respectively

    (This depends on your **.env** secrets as earlier established)

    (You should see the following web pages)





```
detail:   "Not Found"
```

User Guide

Welcome to the User Guide!

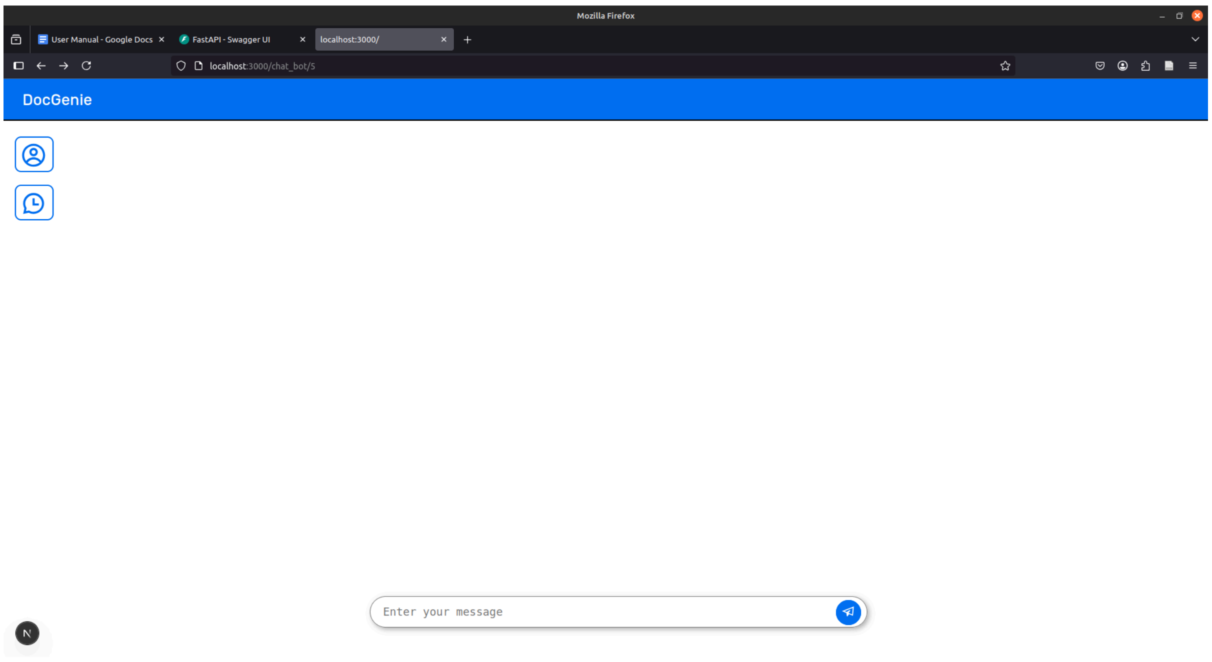
To start interacting with the system itself navigate to http://localhost:3000/.

Enter the following credentials:

Email: jane.smith@example.com
Password: hello

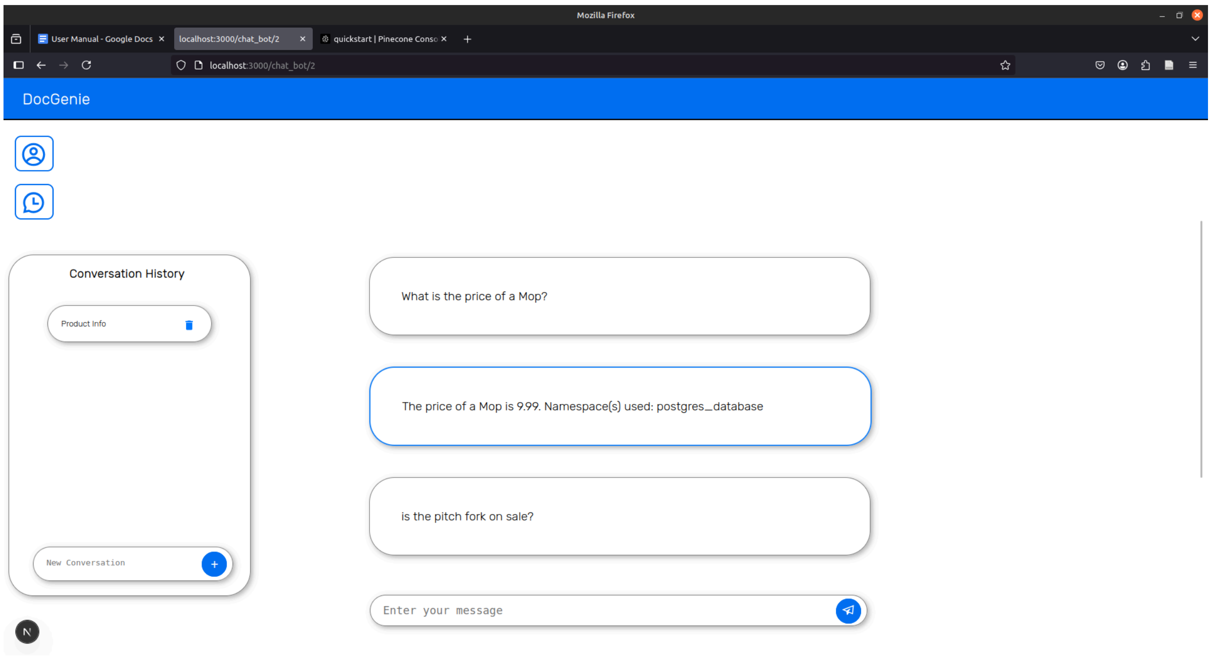
Then click the Login Button

You should now be on the main page of the user interface:



The purpose of this user interface is to allow you to interact with your datastores (e.g PostgreSQL database, web hook etc.) through natural language. This is possible through pinecone namespaces and Artificial Intelligence. To put it in simple terms, you form a query in natural language, in relation to your data in a given namespace and then the Artificial Intelligence system forms a response. The response is based on your user query and the data present in the namespace(s).

Here is an example:


DocGenie

Conversation History

Product Info

New Conversation

What is the price of a Mop?

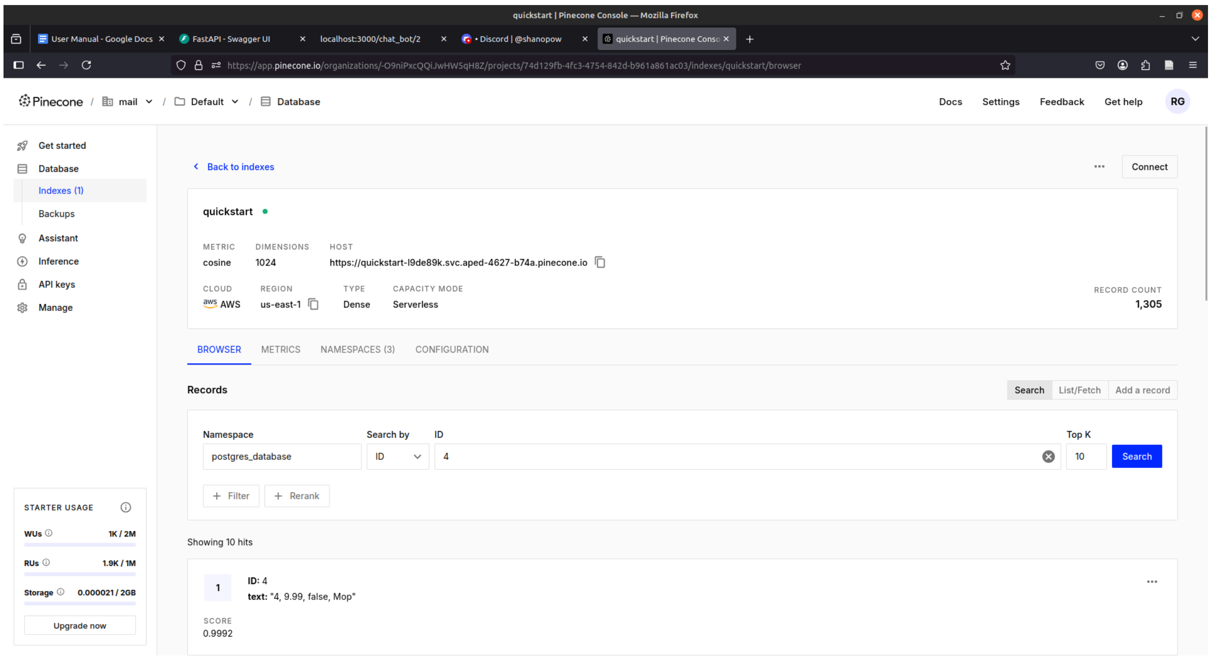
The price of a Mop is 9.99. Namespace(s) used: postgres_database

is the pitch fork on sale?

Enter your message

Explanation:

There is a namespace called “postgres_database”. In that namespace there is a product record about “Mop”:


quickstart

METRIC	DIMENSIONS	HOST
cosine	1024	https://quickstart-l9de89k.svc.aped-4627-b74a.pinecone.io

CLOUD	REGION	TYPE	CAPACITY MODE	RECORD COUNT
AWS	us-east-1	Dense	Serverless	1,305

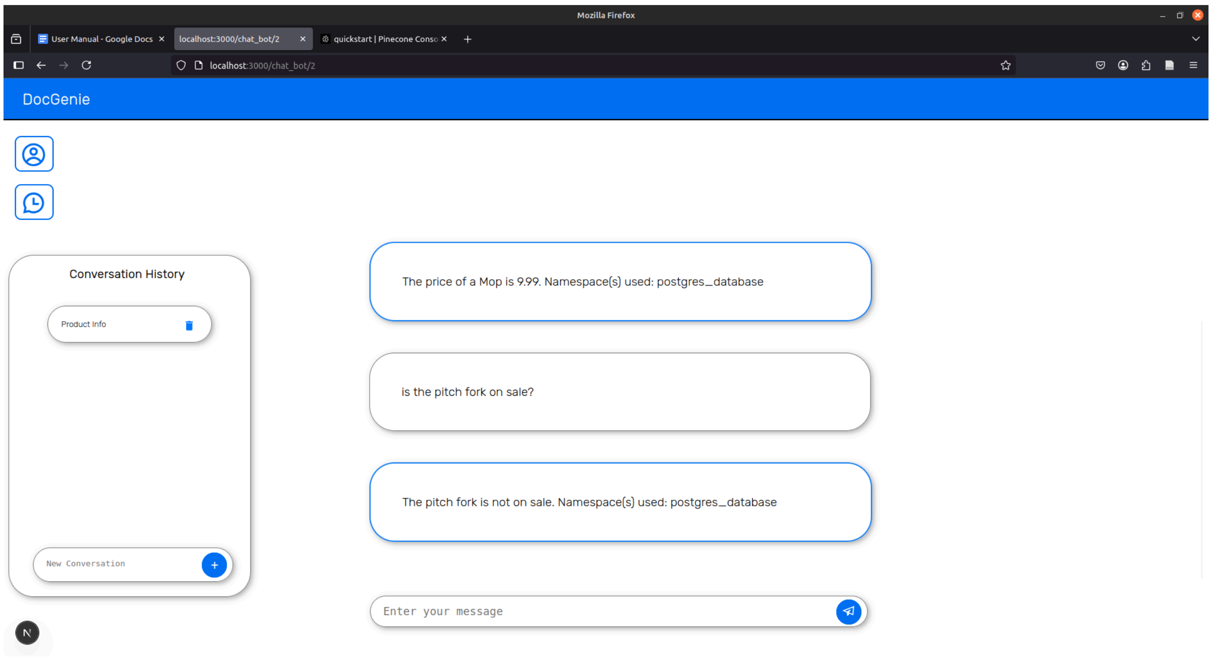
Records

Namespace: postgres_database
Search by: ID
ID: 4
Top K: 10

Showing 10 hits

1
ID: 4
text: "4, 9.99, false, Mop"
SCORE
0.9992

By asking, “What is the price of a Mop?” the system pulls in context for “postgres_database” namespace to answer the question. The question is then displayed on the user interface followed by the generated response. We can see in the response that the price of a Mop is “9.99”. Another example is asking if “is the fork on sale?”. We can see that in the response that the fork isn’t on sale:

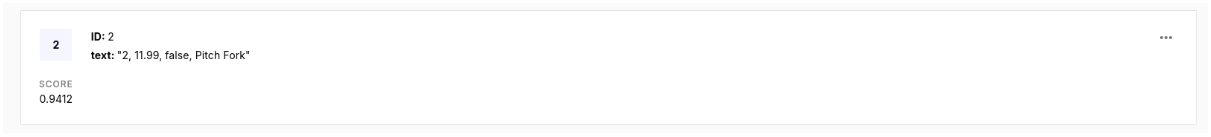

Mozilla Firefox
User Manual - Google Docs
localhost:3000/chat_bot/2
quickstart | Pinecone Conso
localhost:3000/chat_bot/2
DocGenie
Conversation History
Product Info
New Conversation
Enter your message

The price of a Mop is 9.99. Namespace(s) used: postgres_database

is the pitch fork on sale?

The pitch fork is not on sale. Namespace(s) used: postgres_database

This response is correct as if we check the record on the pinecone namespace we can verify if the fork is on sale or not:



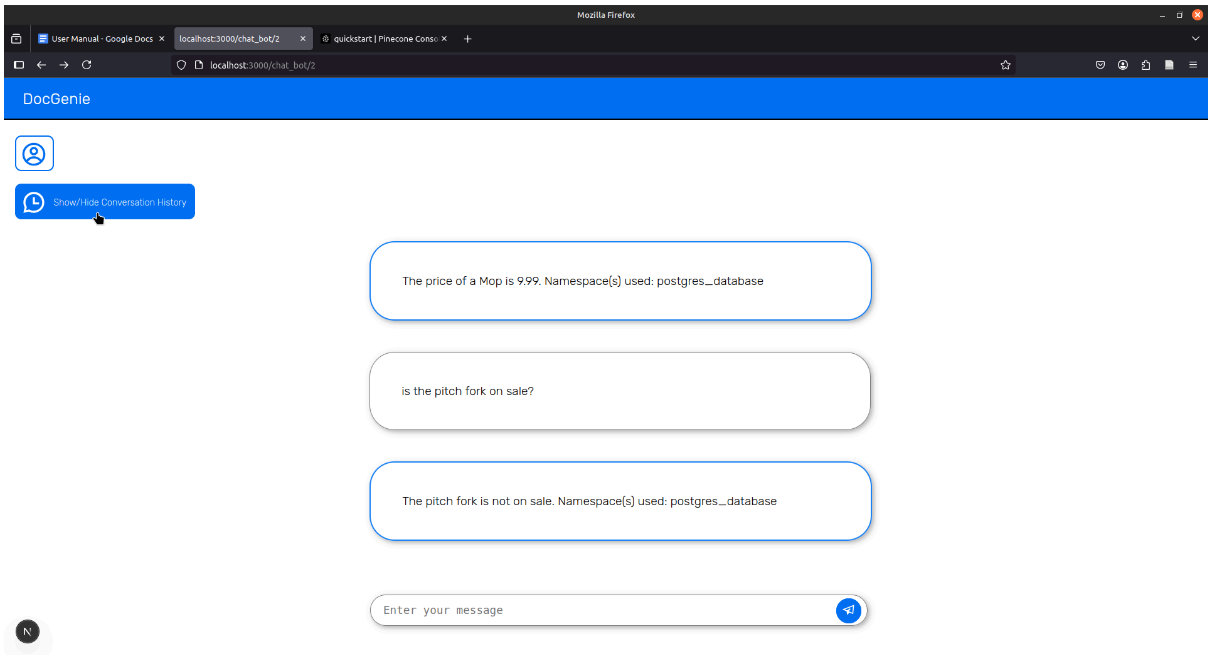
2

ID: 2
text: "2, 11.99, false, Pitch Fork"

SCORE
0.9412

That concludes the main functionality of the user interface. Here are some additional features.

You can create new conversations using the “Conversation History” menu on the left. To toggle the menu in and out just click the “Show/Hide Conversation History” at the top left of the screen:

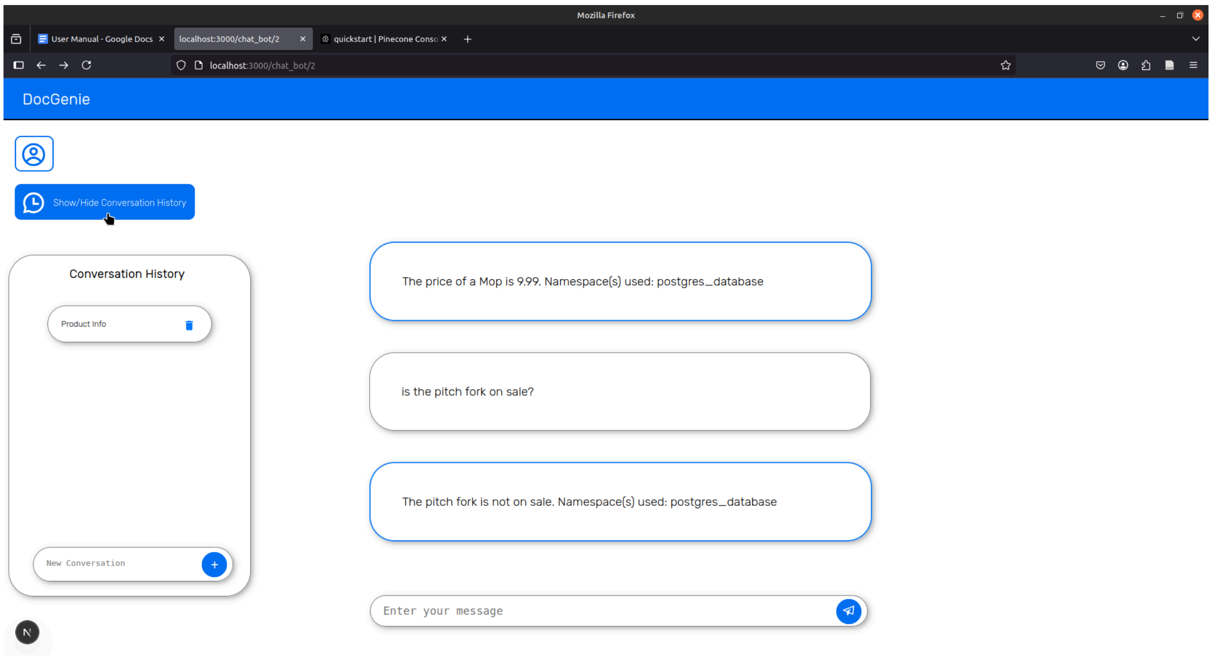

Mozilla Firefox
User Manual - Google Docs
localhost:3000/chat_bot/2
quickstart | Pinecone Conso
localhost:3000/chat_bot/2
DocGenie
Show/Hide Conversation History
Enter your message

The price of a Mop is 9.99. Namespace(s) used: postgres_database

is the pitch fork on sale?

The pitch fork is not on sale. Namespace(s) used: postgres_database

(Before)


Mozilla Firefox
User Manual - Google Docs
localhost:3000/chat_bot/2
quickstart | Pinecone Conso
localhost:3000/chat_bot/2
DocGenie
Show/Hide Conversation History
Conversation History
Product Info
New Conversation
Enter your message

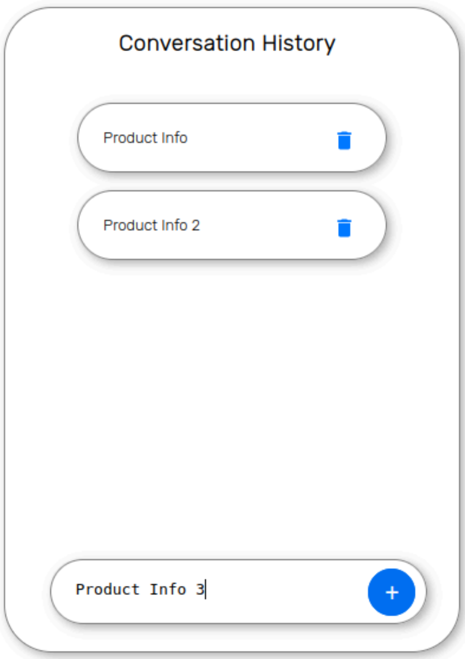
The price of a Mop is 9.99. Namespace(s) used: postgres_database

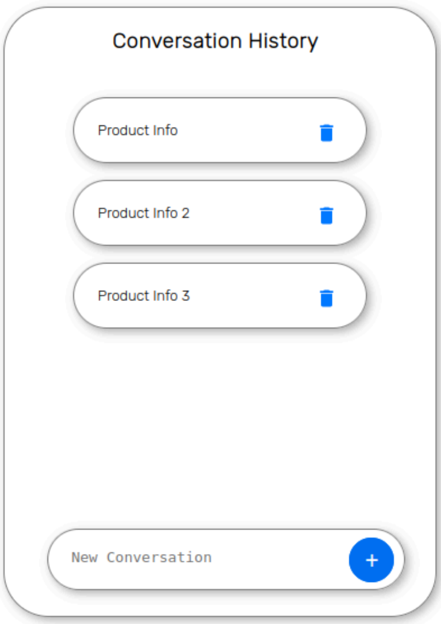
is the pitch fork on sale?

The pitch fork is not on sale. Namespace(s) used: postgres_database

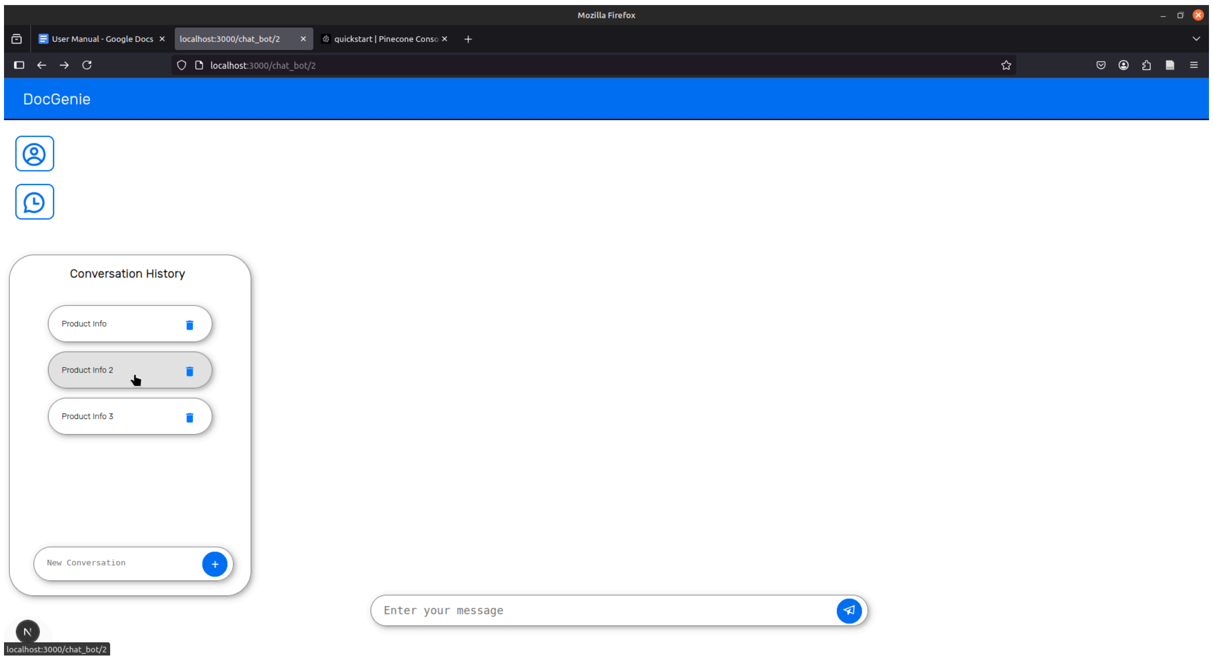
(After)

You then give your new conversation a name in the search bar within the Conversation History menu, then create it either through pressing your Enter key or the **"+"** button:





You can then swap between different conversations by clicking on them:


Mozilla Firefox
User Manual - Google Docs
localhost:3000/chat_bot/2
quickstart | Pinecone Conso
localhost:3000/chat_bot/2
DocGenie
Conversation History
Product Info
Product Info 2
Product Info 3
New Conversation
Enter your message
localhost:3000/chat_bot/2

You can see that since “Product Info 2” and “Product Info 3” are brand new conversations, they don’t have any chats! We can still swap back to the “Product Info” conversation and see all the relevant chats associated with that conversation. Chats will always belong to a conversation and persist even if the system gets rebooted. As well as adding new conversations you can also delete them by clicking the trash can button at the right side of each conversation element.