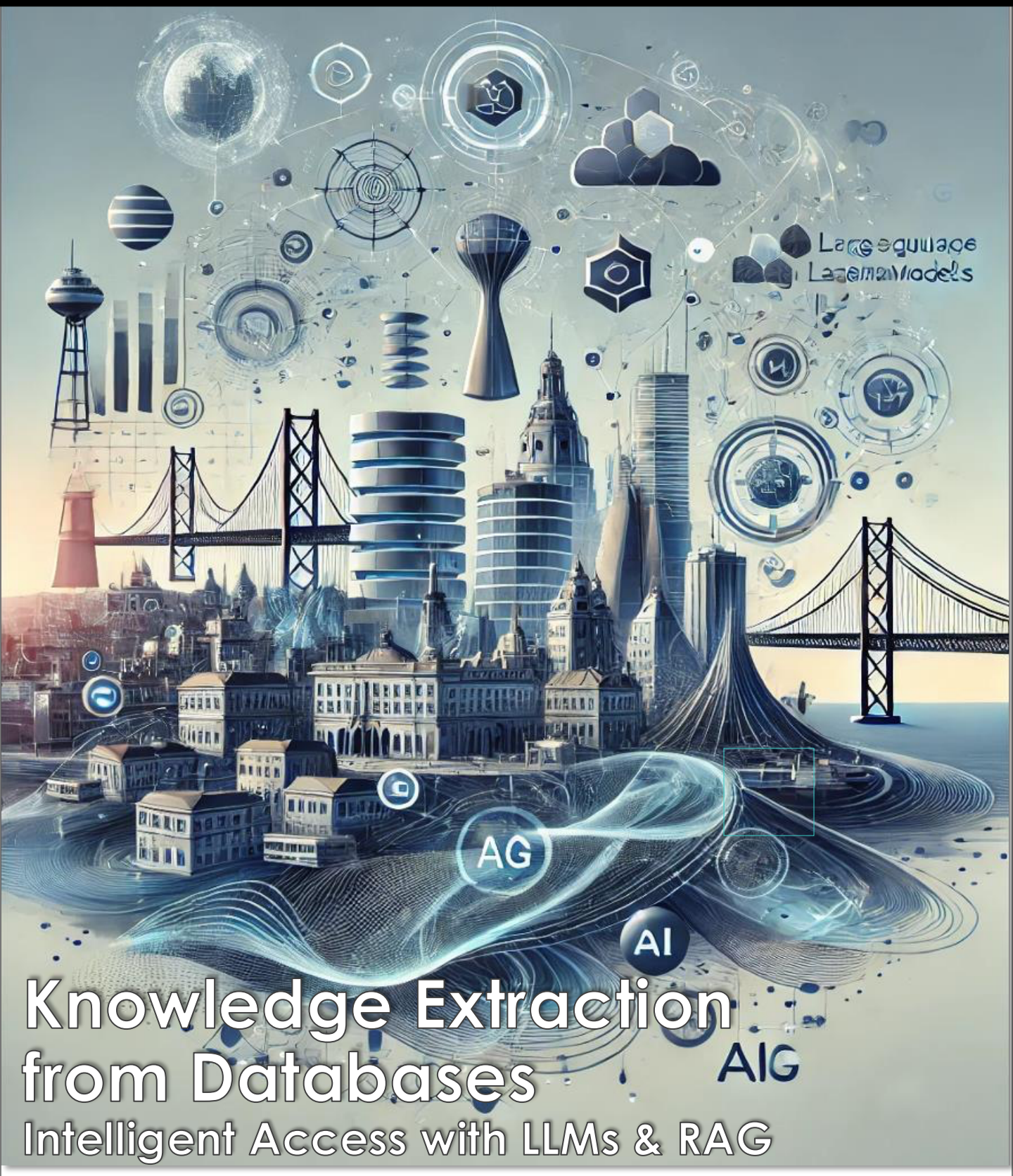
UNIVERSIDADE LUSÓFONA



Knowledge Extraction from Databases

Intelligent Access with LLMs & RAG

Workshop

João Caldeira

March, 2025

Knowledge Extraction from Databases

João Caldeira, Universidade Lusófona, Lisboa, Portugal

25.03.2025

Table of contents

Prefácio **5**
Autor . . . 5
Licença . . . 5

I. Setup **6**

1. Docker **7**
1.1. Download . . . 7
1.2. Instalação . . . 7
1.2.1. Windows . . . 7
1.2.2. macOS . . . 8
1.2.3. Linux . . . 9
1.3. Docker Hub . . . 9
1.3.1. Registo (opcional) . . . 9

2. Azure Data Studio **11**
2.1. Download e Instalação . . . 11
2.2. Configuração . . . 13

II. Workshop **16**

3. Database Server (MS SQL Server) **17**
3.1. Imagem . . . 17
3.2. Contentor . . . 18
3.2.1. Instalação . . . 18
3.3. Windows . . . 18
3.4. macOS . . . 18
3.5. Linux . . . 18
3.6. Autenticação . . . 19

4. Large Language Model (LLM) **20**
4.1. Create a Large Language Model Cluster . . . 20
4.1.1. Create the LLM Engine Container . . . 20
4.1.2. Download your first LLM model . . . 20
4.1.3. Ask your first questions . . . 21

5. User Interface (WebUI) **23**
5.1. Create a Large Language Model Cluster . . . 23
5.1.1. Create the Web UI Container . . . 23
5.1.2. Initial Setup . . . 23
5.1.3. Ask Questions . . . 26

6. Retrieval Augmented Generation (RAG) 30
6.1. Create a Large Language Model Cluster 30
6.1.1. Create the RAG Container 30
6.1.2. Configure RAG 30

List of Figures

1.1. Download de Docker 7
1.2. Configuração Docker Desktop em macOS 8
1.3. Docker Hub 10
1.4. Registo no Docker Hub 10

2.1. Instalação de Azure Data Studio 11
2.2. Iniciar o Azure Data Studio 13
2.3. Configurar ligação ao MS SQL Server 14
2.4. MS SQL Server - Listagem de Objectos 15

3.1. Instalação de SQL Server 17

4.1. Ollama models library 22

5.1. WebUI Console 24
5.2. WebUI Admin User creation 24
5.3. WebUI Admin User creation 25
5.4. WebUI Admin User creation 26
5.5. WebUI Admin User creation 27
5.6. WebUI Admin User creation 27
5.7. WebUI Admin User creation 28
5.8. WebUI Admin User creation 29

6.1. LLM Console 31
6.2. Admin Settings 32
6.3. Connections Settings 32
6.4. Connections Settings 33
6.5. Connections Settings 33
6.6. Connections Settings 34
6.7. Connections Settings 35
6.8. Connections Settings 36
6.9. Pipelines Settings 37
6.10. Pipelines Settings 38
6.11. Pipelines Settings 38
6.12. Pipelines Settings 39
6.13. Pipelines Settings 40
6.14. Docker Environment 41
6.15. RAG Model Selection 42
6.16. RAG Model Selection 43
6.17. Ask Questions 44

List of Tables

Prefácio

Este workshop ensinará a utilizar `containers` para implementar um sistema de `Retrieval Augmented Generation (RAG)`, permitindo consultas a bases de dados sem a necessidade de SQL. Durante a sessão, os participantes aprenderão a configurar e gerir `containers` distintos: um para um `modelo de linguagem (LLM)`, outro para a `interface web` que facilitará a interação do utilizador e um terceiro responsável pelos `pipelines` de processamento e recuperação de informações. O foco será na criação de um fluxo eficiente, onde o modelo de linguagem acede aos dados estruturados por meio de técnicas avançadas de recuperação, transformando a forma como consultas a bases de dados são realizadas.

Este workshop destina-se a profissionais de tecnologia, programadores, engenheiros de dados e entusiastas da inteligência artificial que desejam aprender a utilizar a referida tecnologia em `containers` para implementar um sistema de `Retrieval Augmented Generation (RAG)`.

Alguns exercícios podem começar por listar os objetivos de aprendizagem e os materiais de leitura necessários, com referências adicionais, caso se justifique. A inclusão de notas ajuda a apresentar os métodos e dar alguns exemplos, mas são menos detalhadas do que os materiais de leitura. Algumas notas são mensagens importantes para concluir as tarefas corretamente.

Autor

João Caldeira. Professor Auxiliar na Universidade Lusófona e Investigador Associado na COPELABS em Lisboa, Portugal.

Licença

Este trabalho é licenciado sob as condições descritas em Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

Part I.

Setup

1. Docker

1.1. Download

O Docker pode ser instalado em formato desktop (apenas Windows ou macOS) ou em formato de linha de comando (Windows, macOS e Linux). Aceda à página do **Docker** e faça o **download** da versão correspondente ao seu sistema operativo, tal como demonstrado na Figure 1.1.

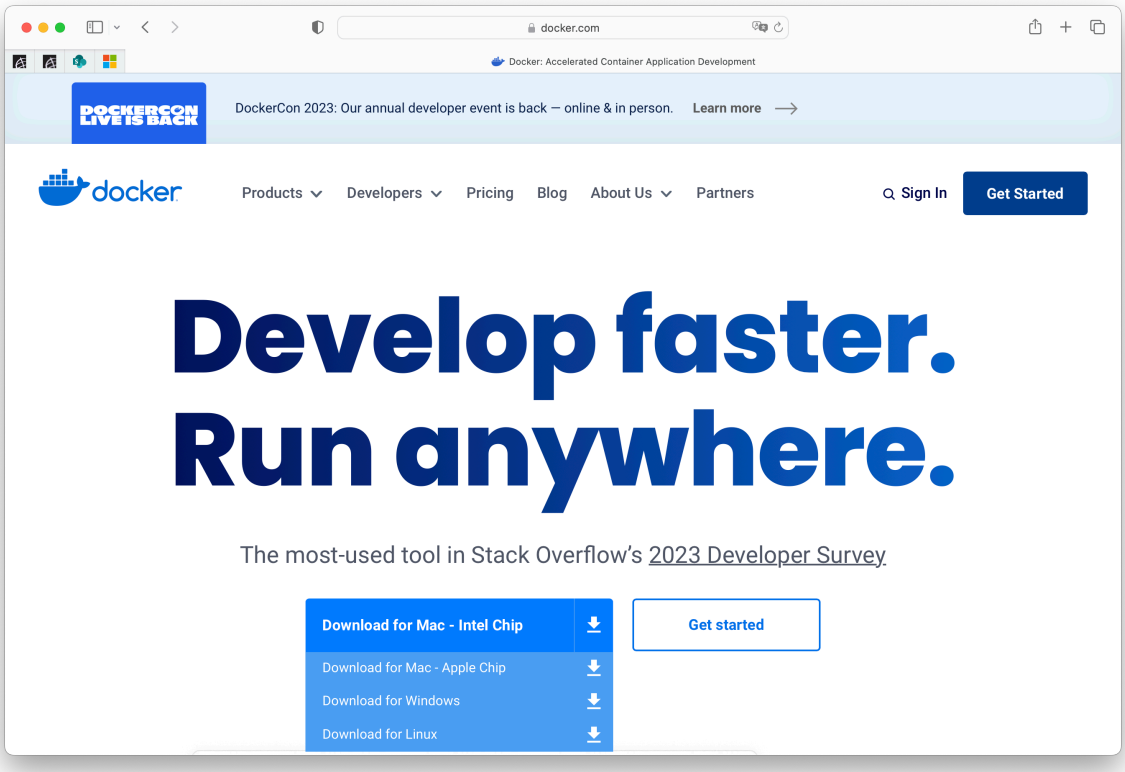


Figure 1.1.: Download de Docker

1.2. Instalação

1.2.1. Windows

> **Nota**
>
> **Instruções válidas apenas para Windows 8 ou posterior**

Obter o instalador da versão atual para Windows do Docker Desktop.

A instalação desta versão irá disponibilizar o cliente de linha de comandos e powershell.

> **Nota**
>
> **A instalação de Docker em ambiente Windows requer a activação da componente Windows System for Linux versão 2 (WSL2).**

Em caso de falha no WSL, consultar o Troubleshooting do Windows Subsystem for Linux | Microsoft Learn

1.2.2. macOS

> **Nota**
>
> **Instruções válidas apenas para MacOs**

Obter o instalador da versão atual para macOS do Docker Desktop.

Em macOS, também se pode instalar directamente em linha de comando utilizando semelhantes procedimentos que se indicam para Linux.

Os utilizadores de Docker em macOS devem ativar a seguinte opção nas configurações do Docker Desktop, tal como apresentado na Figure 1.2:

- Use Rosetta for x86/amd64 emulation on Apple Silicon
- Ativar e fazer Restart

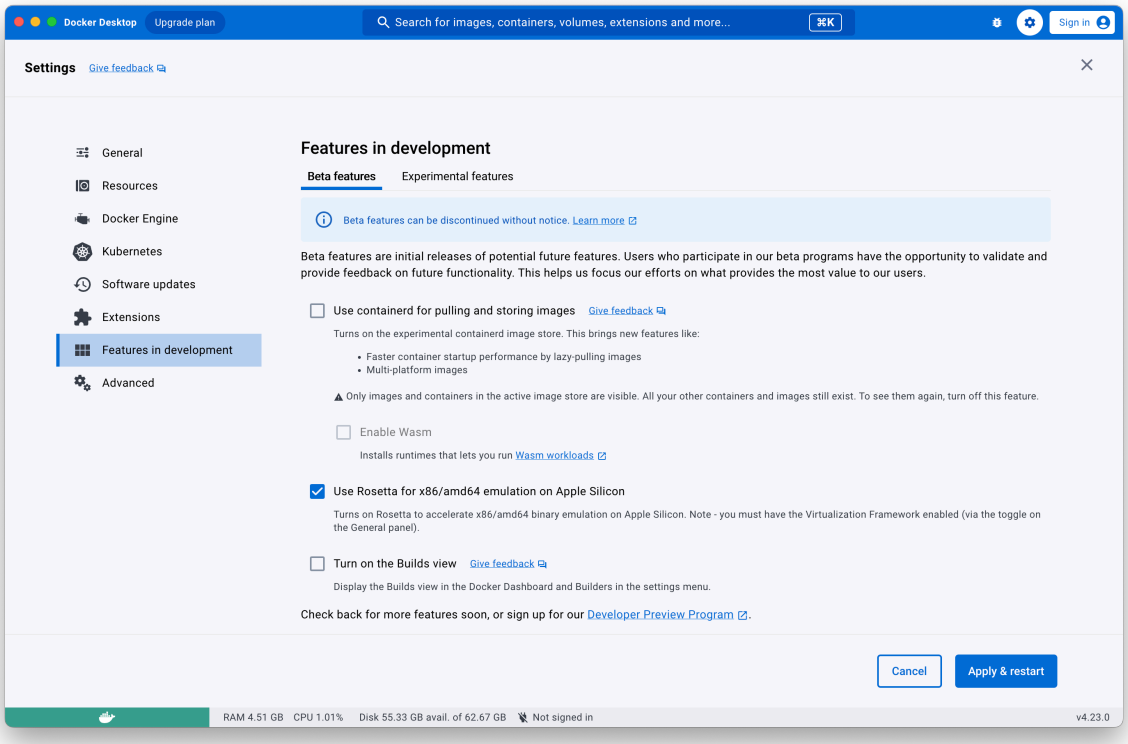


Figure 1.2.: Configuração Docker Desktop em macOS

1.2.3. Linux

> **Nota**
>
> **As indicações referem-se a instalação em Amazon Linux 2 (AWS)**

O processo pode variar em função da implementação Linux utilizada

- Ex.: Ubuntu.
- Actualizar pacotes, garantindo que o sistema operativo está actualizado (opcional)

```
sudo yum update -y
```

- Instalar o Docker Engine

```
sudo amazon-linux-extras install docker
```

- Iniciar o Docker

```
sudo service docker start
```

- Verificar o estado do serviço e versão

```
docker info
```

1.3. Docker Hub

1.3.1. Registo (opcional)

As instalações podem ser feitas de forma anónima ou em formato autenticado, via Docker Hub.

É recomendado que cada utilizador esteja registado.

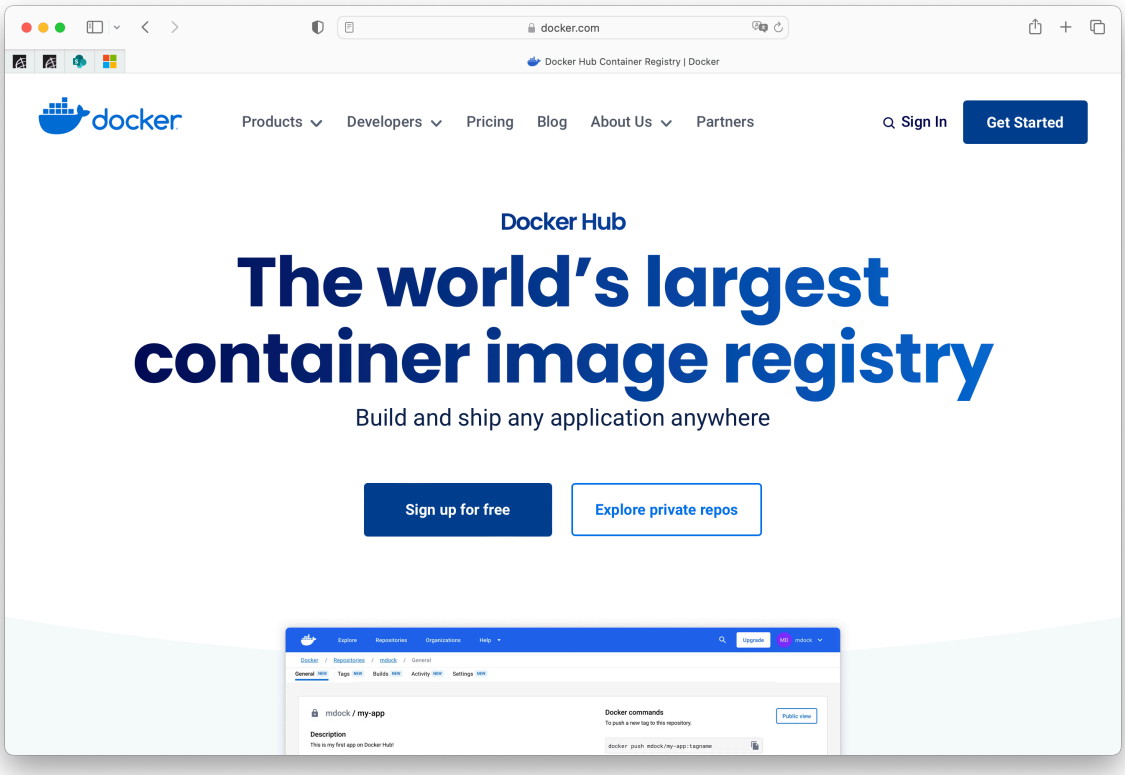


Figure 1.3.: Docker Hub

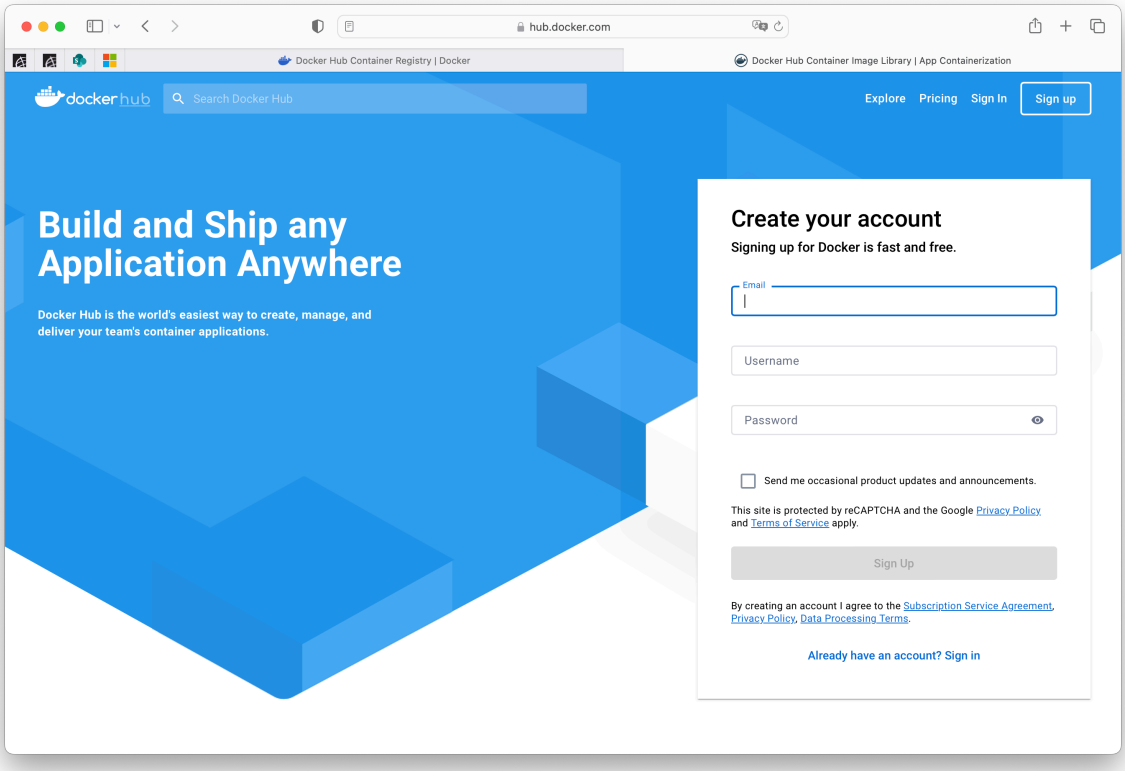


Figure 1.4.: Registo no Docker Hub

2. Azure Data Studio

2.1. Download e Instalação

Procedimentos para instalação e configuração do cliente **Azure Data Studio** para trabalhar com **Microsoft SQL Server** ou outros **Sistemas de Gestão de Bases de Dados Relacional (SGBDR/RDBMS), ex. MySQL/MariaDB/PostGres**.

Faça o download do software da página do Azure Data Studio tal como identificado na Figure 2.1 ou vá direto aqui.

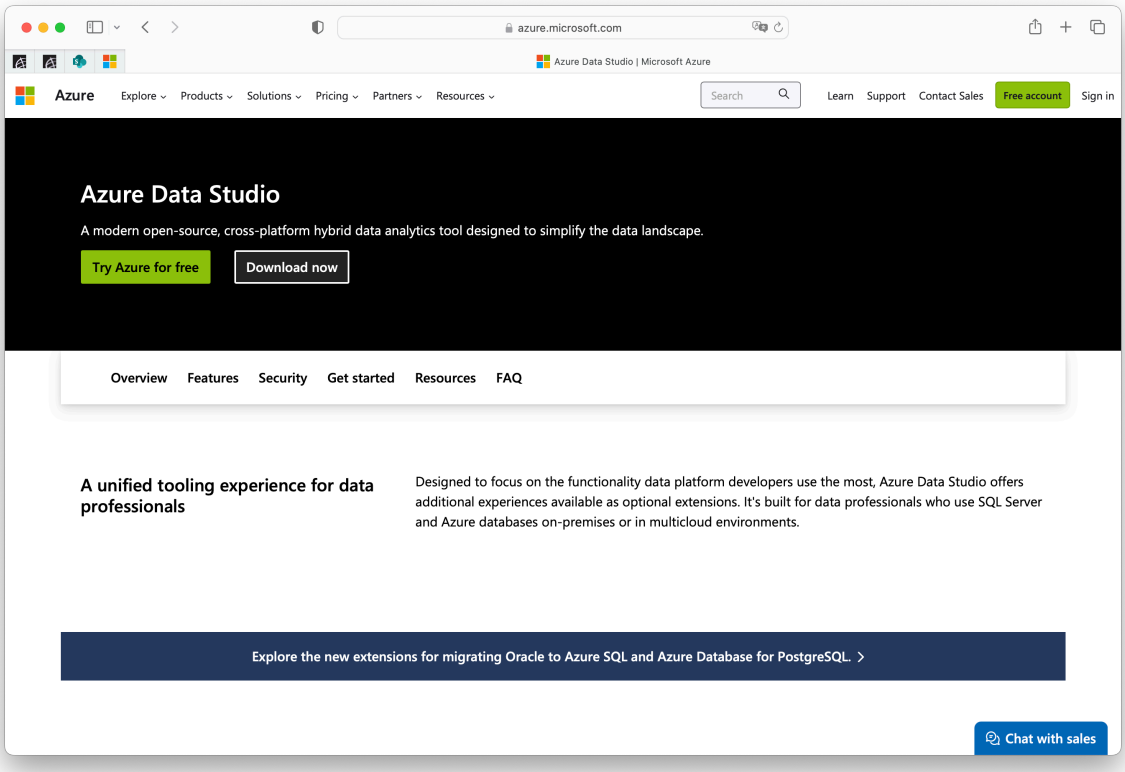
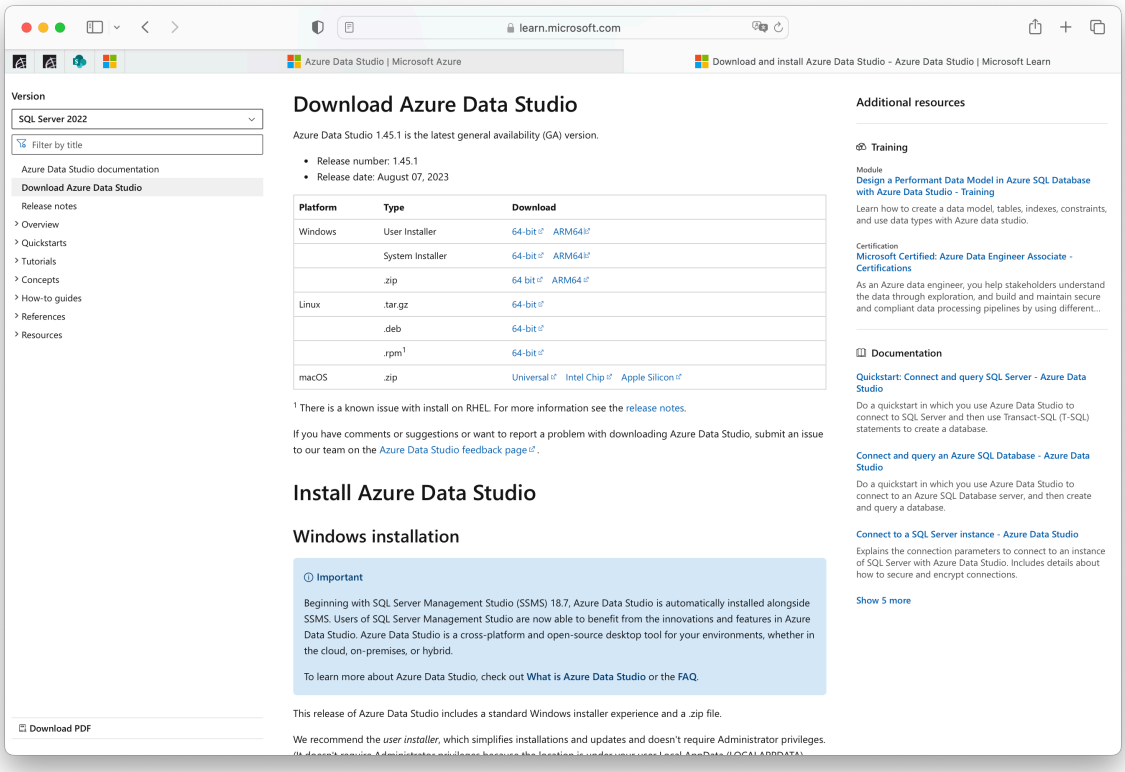


Figure 2.1.: Instalação de Azure Data Studio

Escolha a versão adequada ao seu sistema operativo.



Inicie o **Azure Data Studio** e configure uma ligação à base de dados, ou seja, ao contentor instalado anteriormente.

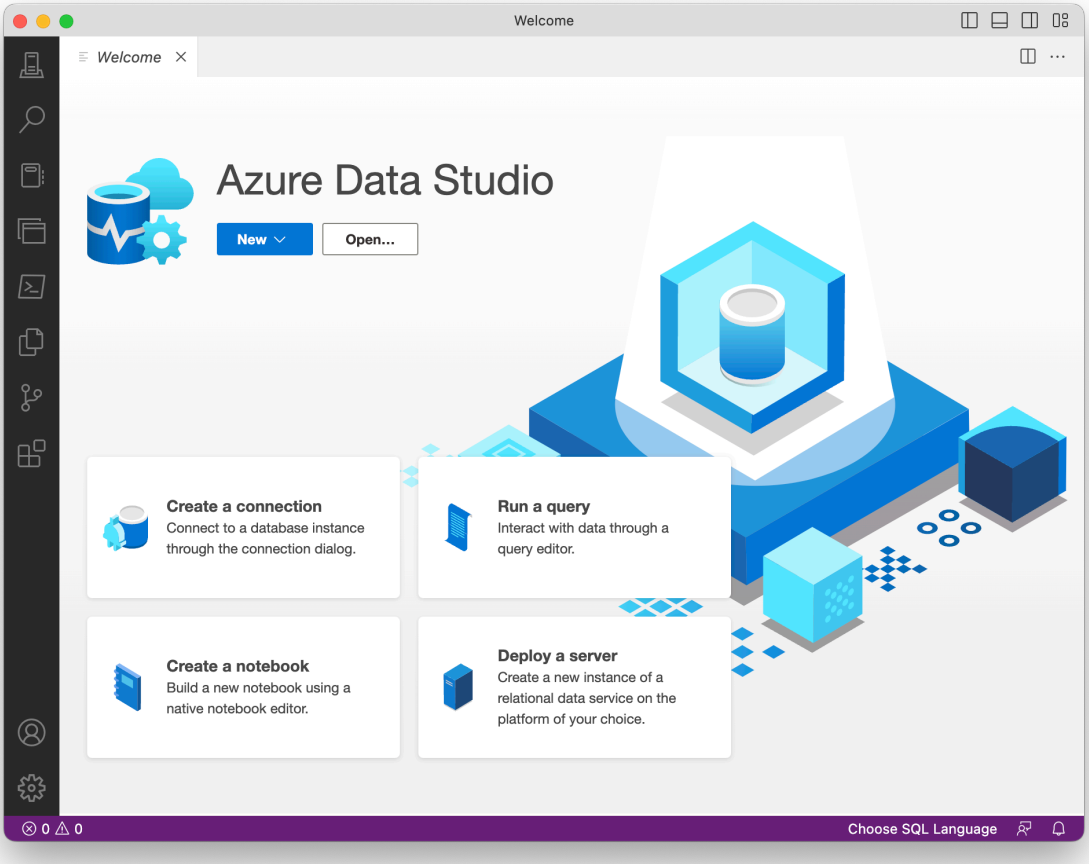


Figure 2.2.: Iniciar o Azure Data Studio

2.2. Configuração

- Cria uma nova conexão, com os seguintes parâmetros
  - **Server:** *localhost (ou endereço IP da máquina onde está o MS SQL Server)*
  - **Authentication Type:** *SQL Login*
  - **User Name:** *sa*
  - **Password:** *Tecweb2025*
  - Os restantes parâmetros devem ser configurados tal como identificado na Figure 2.3.

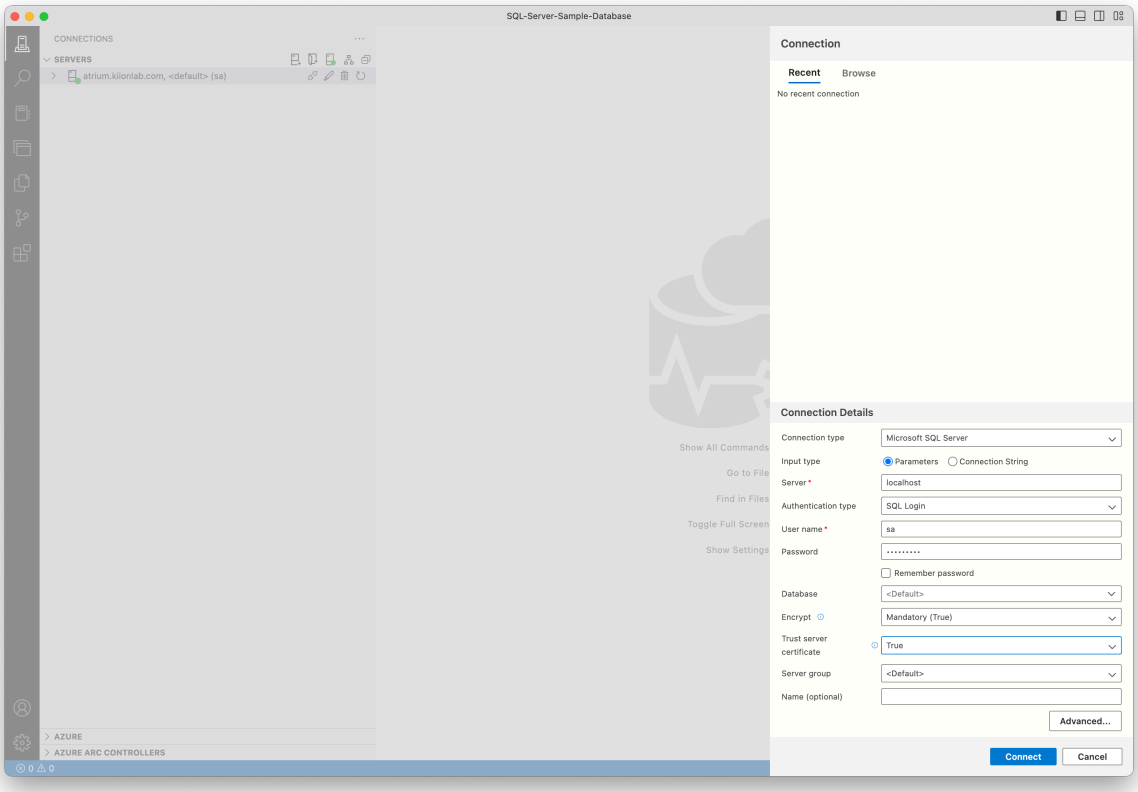


Figure 2.3.: Configurar ligação ao MS SQL Server

No caso de uma ligação bem sucedida à base de dados, a aplicação Azure Data Studio deve estar muito semelhante à apresentada na Figure 2.4.

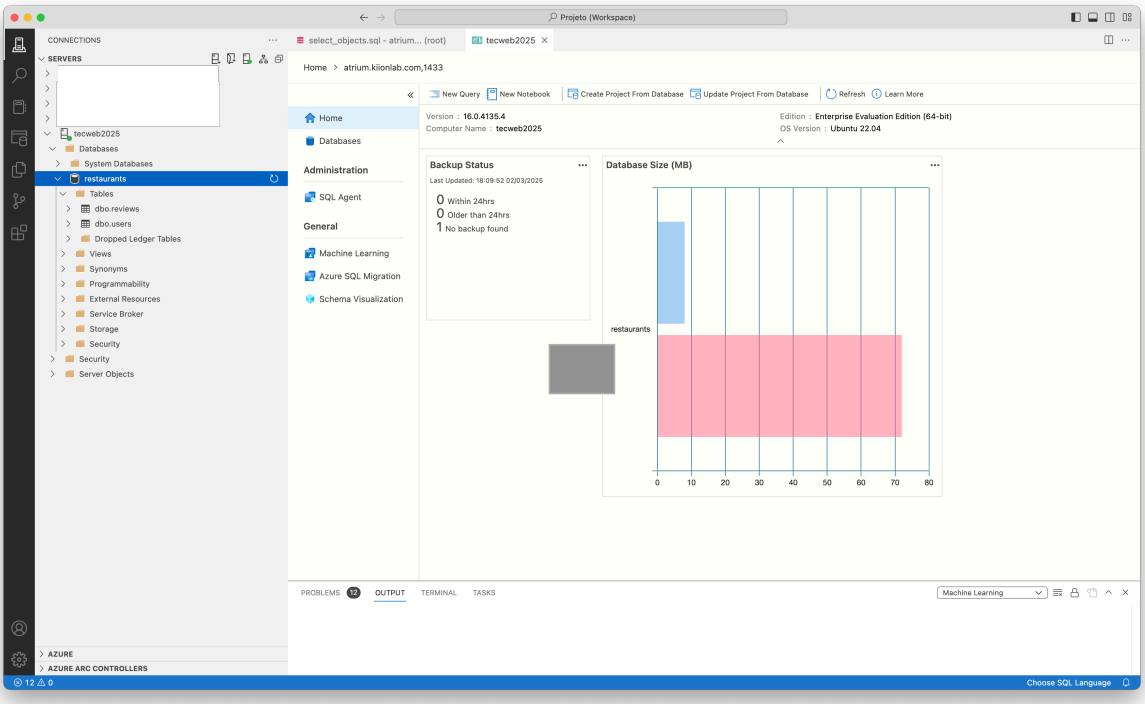


Figure 2.4.: MS SQL Server - Listagem de Objectos

Part II.

Workshop

3. Database Server (MS SQL Server)

3.1. Imagem

A instanciação de um **CONTAINER** requer uma **IMAGEM** e, opcionalmente, uma tag de identificação de versão. - (ex.: mysql:latest instancia a última versão da IMAGEM de MySQL. Se omitida a tag de versão, é utilizada a versão latest)

A versão refere-se à imagem a colocar no container, não ao software base.

Para instanciar um CONTAINER executa-se comando DOCKER RUN, que pode ser acrescido de parâmetros de personalização Os parâmetros dependem da imagem e do ambiente (ex.: -p mapeia uma porta do host para a IMAGEM do CONTAINER).

Neste workshop iremos utilizar um **Sistema de Gestão de Bases de Dados Relacional (SGBDR/RDBMS), o Microsoft SQL Server**.

Este sistema já está pré-instaldo com a imagem identificada na Figure 3.1, de nome **jcppc/tecweb2025** no Docker Hub.

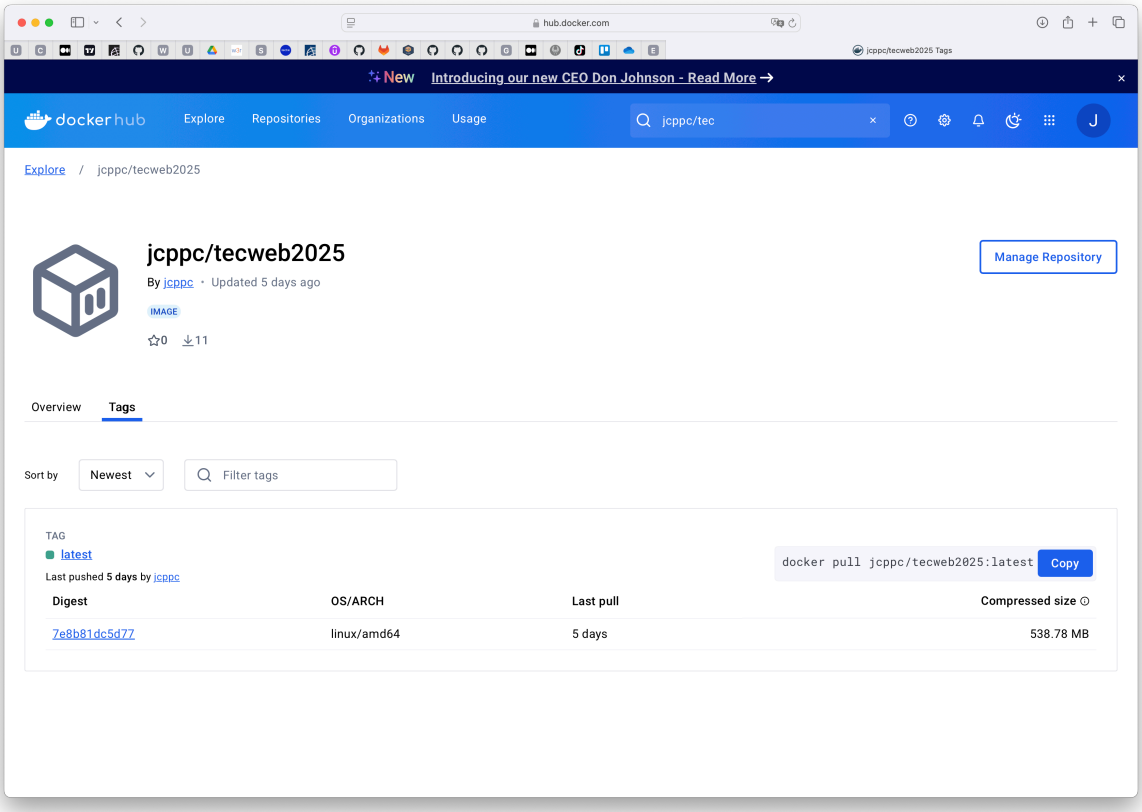


Figure 3.1.: Instalação de SQL Server

> **Nota**
>
> **Consultar o Docker Hub para todas as imagens oficiais para Docker.**

3.2. Contentor

3.2.1. Instalação

- Criar uma rede chamada `workshop_network` executando o seguinte comando.

```
docker network create workshop_network
```

A imagem a utilizar está guardada aqui.

O seguinte comando, conforme o seu sistema operativo, deve ser utilizado para fazer o download dessa imagem e iniciar o contentor.

3.3. Windows

```
docker run -p 1433:1433 --add-host=host.docker.internal:host-gateway --name
 ↪  tecweb2025-db --hostname tecweb2025-db --network workshop_network -d
 ↪  jcppc/tecweb2025:latest
```

3.4. macOS

```
docker run -p 1433:1433 --add-host=host.docker.internal:host-gateway --platform
 ↪  linux/amd64 --name tecweb2025-db --hostname tecweb2025-db --network
 ↪  workshop_network -d jcppc/tecweb2025:latest
```

3.5. Linux

```
docker run -p 1433:1433 --add-host=host.docker.internal:host-gateway --name
 ↪  tecweb2025-db --hostname tecweb2025-db --network workshop_network -d
 ↪  jcppc/tecweb2025:latest
```

3.6. Autenticação

- Para conectar-se à Base de Dados com um software cliente, deve ter em conta os seguintes parâmetros:
  - **Server:** *localhost (ou endereço IP da máquina onde está o MS SQL Server)*
  - **Authentication Type:** *SQL Login*
  - **User Name:** *sa*
  - **Password:** *Tecweb2025*
  - Os restantes parâmetros devem ser configurados de acordo com o identificado na Figure 2.3.

4. Large Language Model (LLM)

4.1. Create a Large Language Model Cluster

4.1.1. Create the LLM Engine Container

- We are going to use the Ollama software. To install without using Docker, it can be found here.
- Run the following `Docker` commands in your machine `command line` prompt.
- Create a volume called `llm-data`

```
docker volume create llm-data
```

- Create the docker container with the `ollama` LLM engine.

```
docker run -d -v llm-data:/root/.ollama -p 11434:11434 --network
 ↪  workshop_network --name llm --hostname llm ollama/ollama
```

4.1.2. Download your first LLM model

- Connect to your `llm` container with this command:

```
docker exec -it llm /bin/bash
```

- Run the following command to download the LLM model `llama3.2`.

```
ollama pull llama3.2
```

> Tip
>
> - You can download and install more/other models by issuing the previous command but for other model names (eg: `mistral`, `phi3.5`, `qwen2.5`, etc).

- Run the following command to confirm the LLM model `llama3.2` was downloaded.

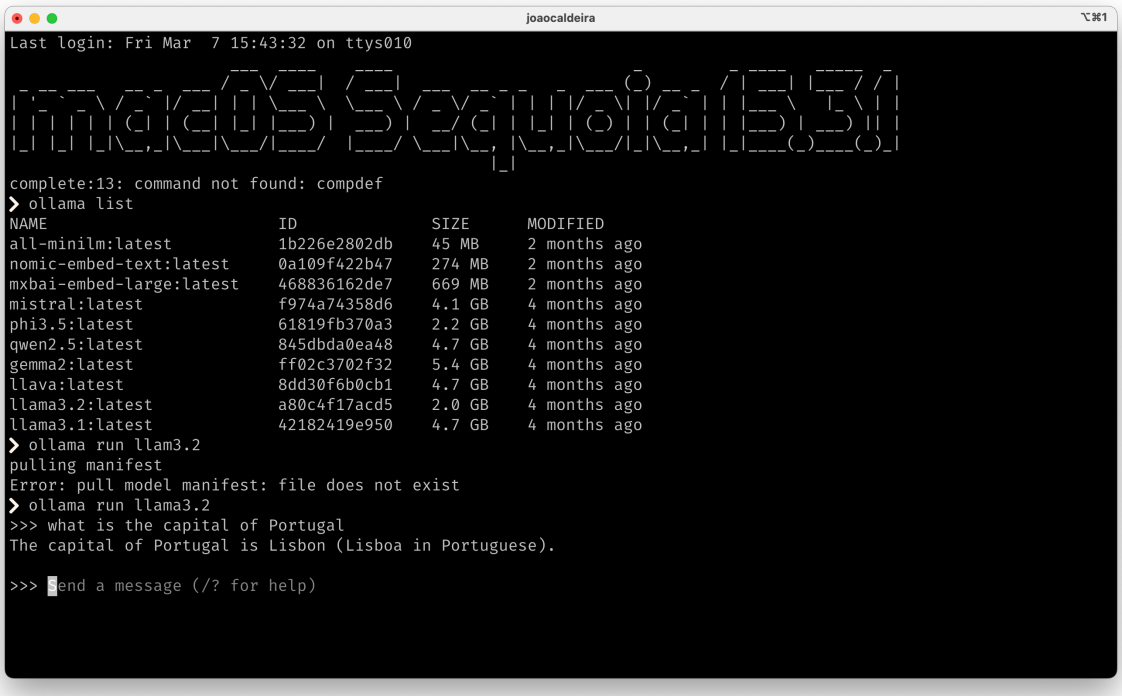
```
ollama list
```

4.1.3. Ask your first questions

- Run the following command to run the LLM model `llama3.2`.

```
ollama run llama3.2
```

- Once you have the prompt, you can start to ask questions to the LLM model `llama3.2`.



- You can download and use other models. See the full list compatible with the `ollama` LLM engine here.

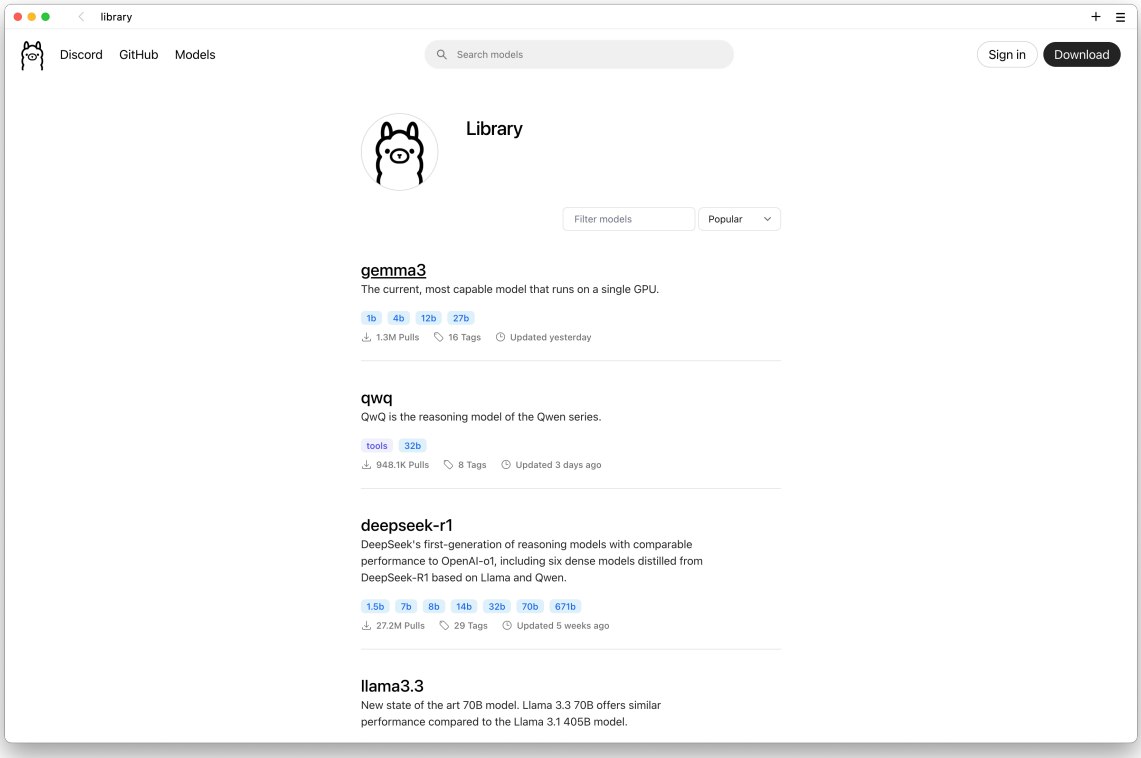


Figure 4.1.: Ollama models library

5. User Interface (WebUI)

5.1. Create a Large Language Model Cluster

5.1.1. Create the Web UI Container

- Create a volume called `webui-data`

```
docker volume create webui-data
```

- Create the docker container with the `Webui` web graphical user interface by running the following command.

```
docker run -d -p 3000:8080 --add-host=host.docker.internal:host-gateway -v
 ↪  webui-data:/app/backend/data --network workshop_network --name webui
 ↪  --hostname webui --restart always ghcr.io/open-webui/open-webui:main
```

5.1.2. Initial Setup

- You can now access your web interface in http://localhost:3000.
- Press `Get Started`.



Figure 5.1.: WebUI Console

- Create the `Admin` user.

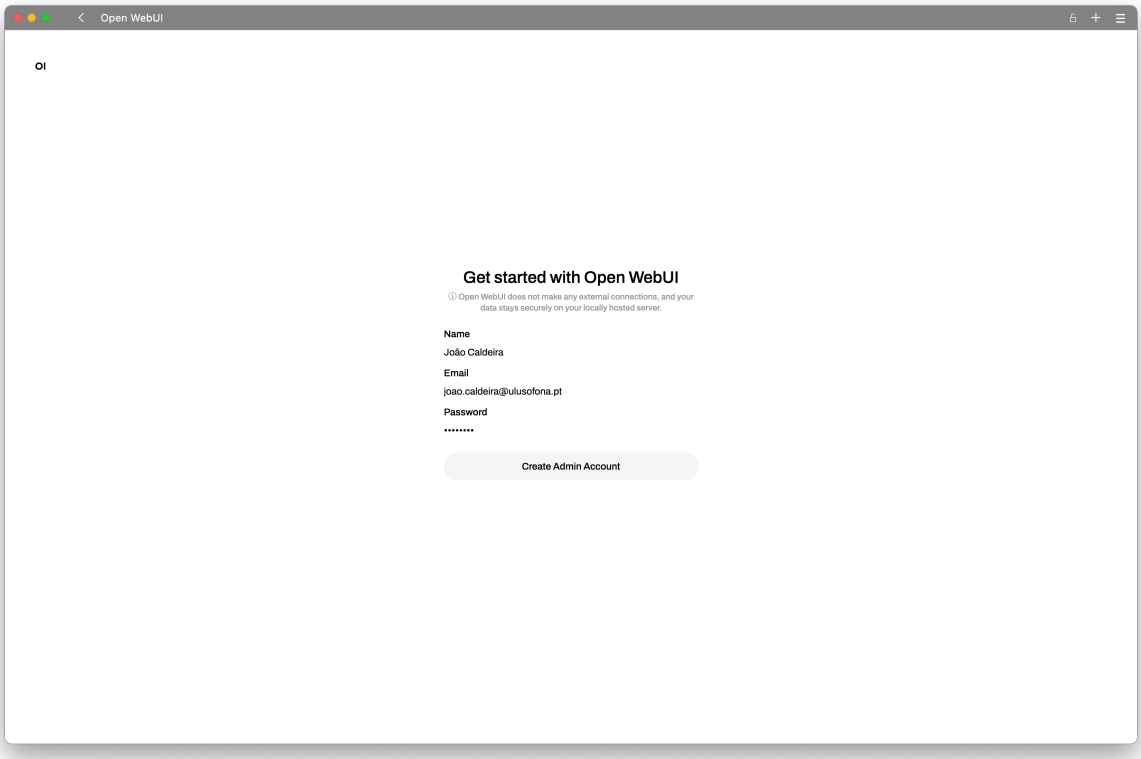


Figure 5.2.: WebUI Admin User creation

- Accept the `Release Notes`.

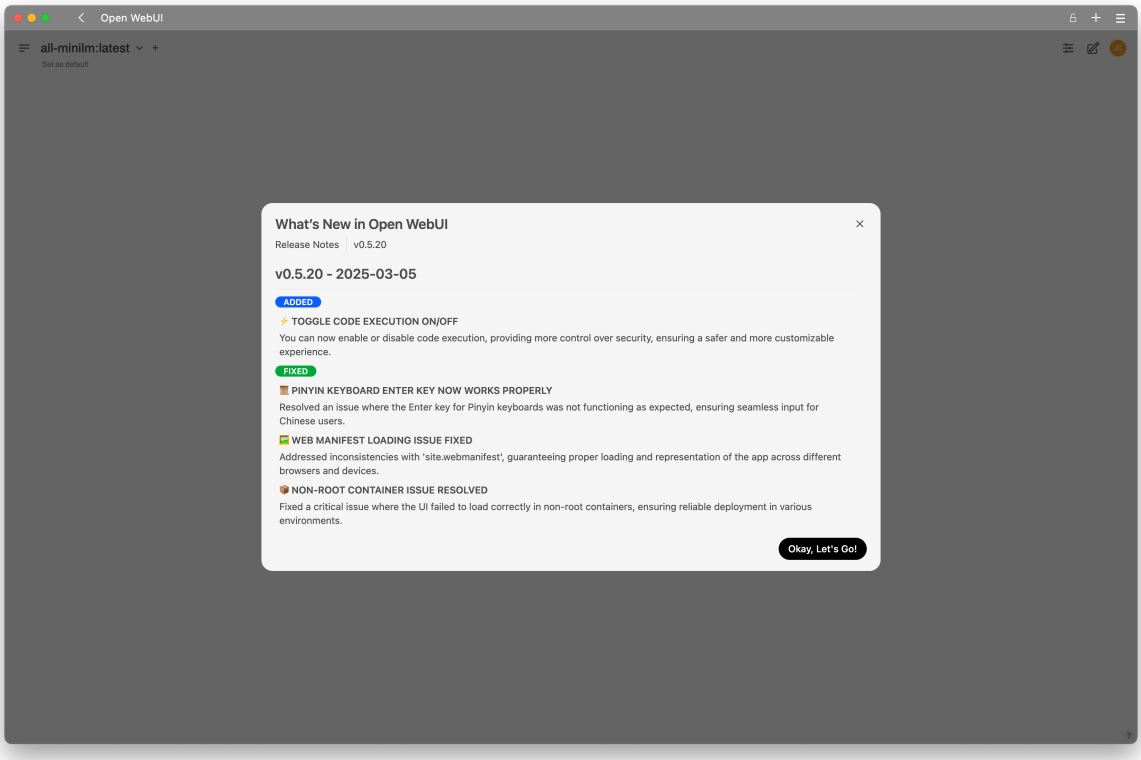


Figure 5.3.: WebUI Admin User creation

- View the existing models already downloaded in the `LLM engine`. In this exercise, you should have `llama3.2` already.

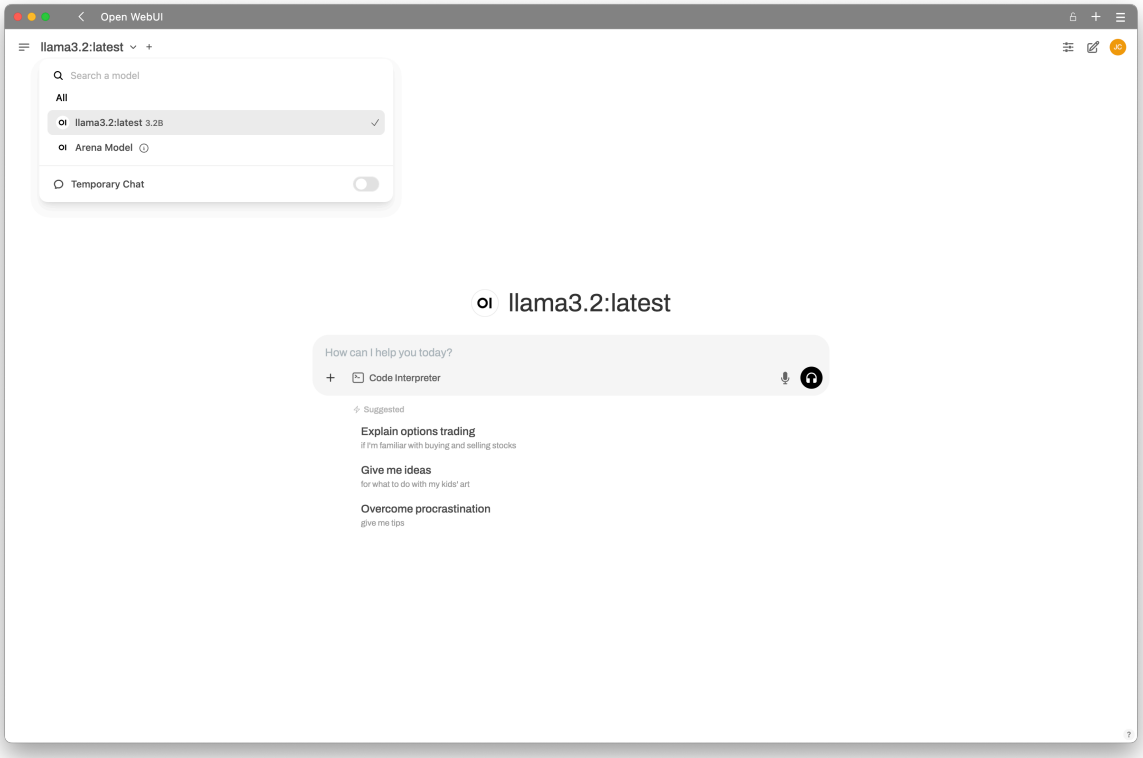


Figure 5.4.: WebUI Admin User creation

5.1.3. Ask Questions

- Pick your model and you are able to ask your first questions.



Figure 5.5.: WebUI Admin User creation

- View the answers.

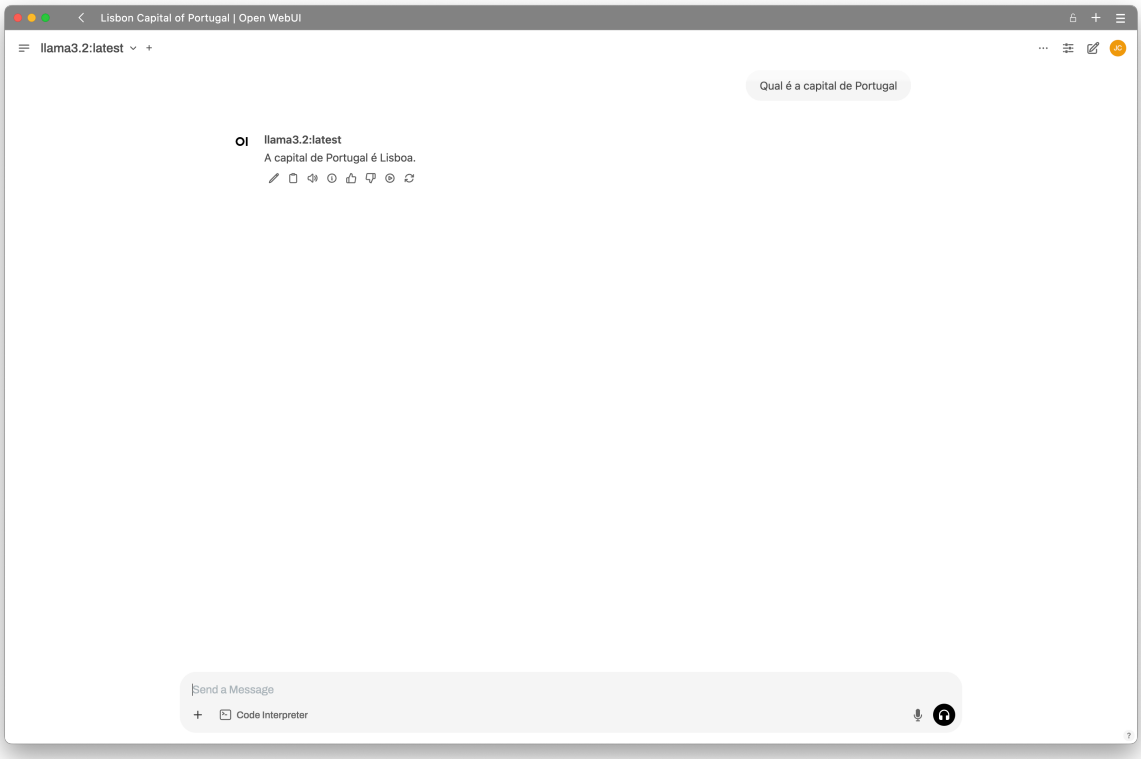


Figure 5.6.: WebUI Admin User creation

5.1.3.1. Connect WebUI with ChatGPT

- You can also connect to ChatGPT API and use it's models.
- Navigate to the `Admin Panel > Settings > Connections` section.
- Add your `ChatGPT Key`, similar to this one, in the last field of the form.

> **!** Important
>
> This ChatGpt Key is no longer valid. It's used here only as an example.

```
sk-proj-XlXRiWlfetudLBrRrP6B4C_RRjDpJu03LJ9smXeDz3p-gE7PLg-a1Td6qFCWoydnIL2pxhJL6dT3BlbkFJ5E
```

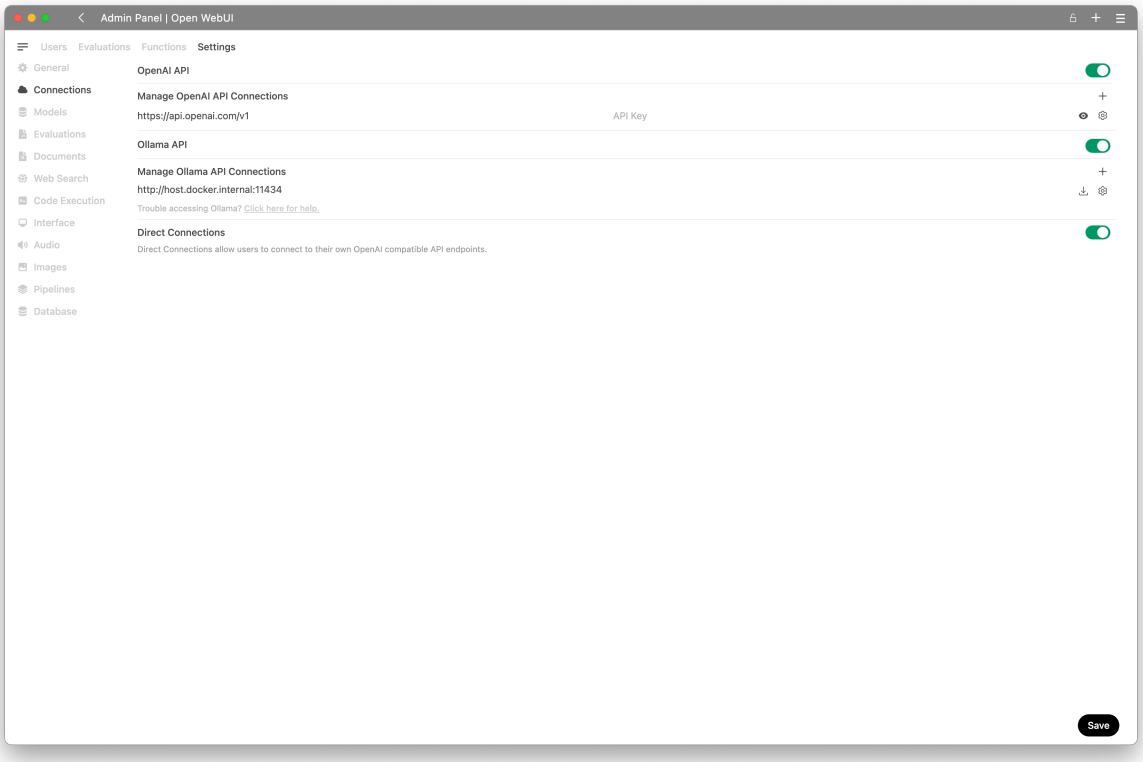


Figure 5.7.: WebUI Admin User creation

- Return to Workspace and you can now view that ChatGPT models are also listed.

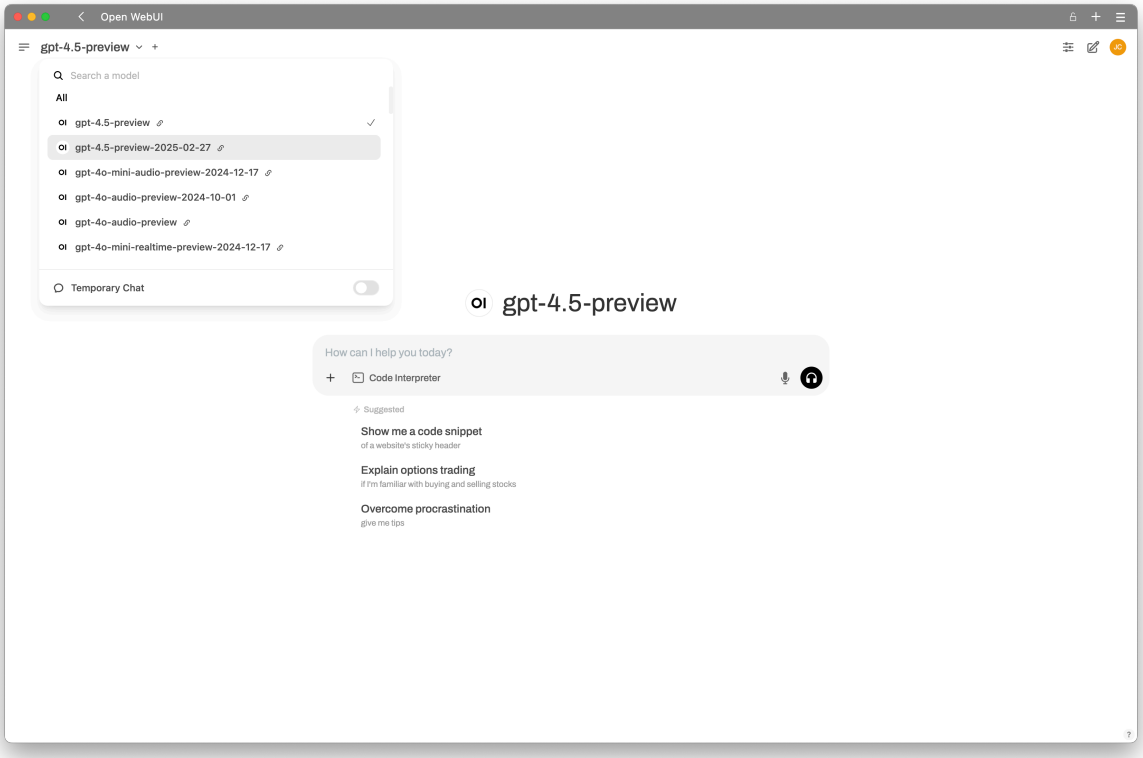


Figure 5.8.: WebUI Admin User creation

6. Retrieval Augmented Generation (RAG)

6.1. Create a Large Language Model Cluster

6.1.1. Create the RAG Container

- Create a volume called `rag-data`.

```
docker volume create rag-data
```

- Create the container with the `rag` engine in the folliwng way. The software is documented here.

```
docker run -d -p 9099:9099 --add-host=host.docker.internal:host-gateway
 ↪  --platform linux/amd64 -v rag-data:/app/pipelines --network
 ↪  workshop_network --name rag --hostname rag --restart always
 ↪  jcppc/tecweb2025-rag:latest
```

6.1.2. Configure RAG

- Access the WebUI and configure the LLM to use RAG http://localhost:3000.
- Follow these steps:
  - You should have the following screen or very similar.

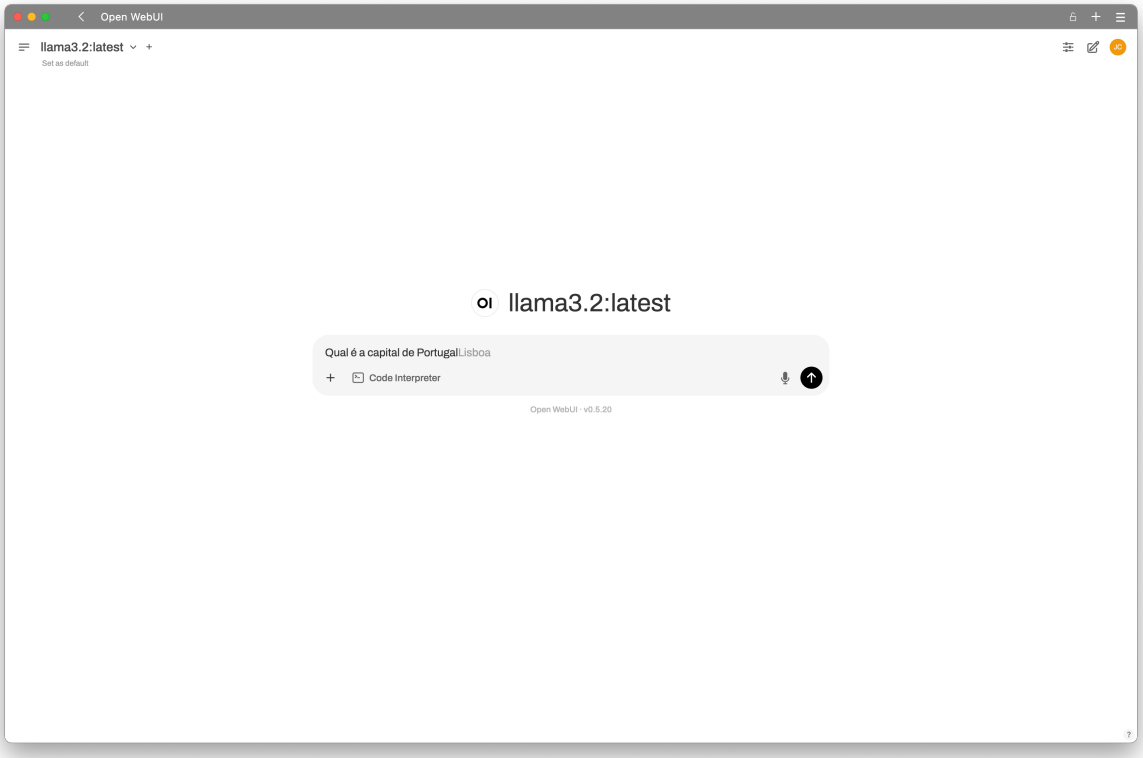


Figure 6.1.: LLM Console

- Navigate to the `Admin Panel` > `Settings` > `Connections` section in the top right side of Open WebUI.

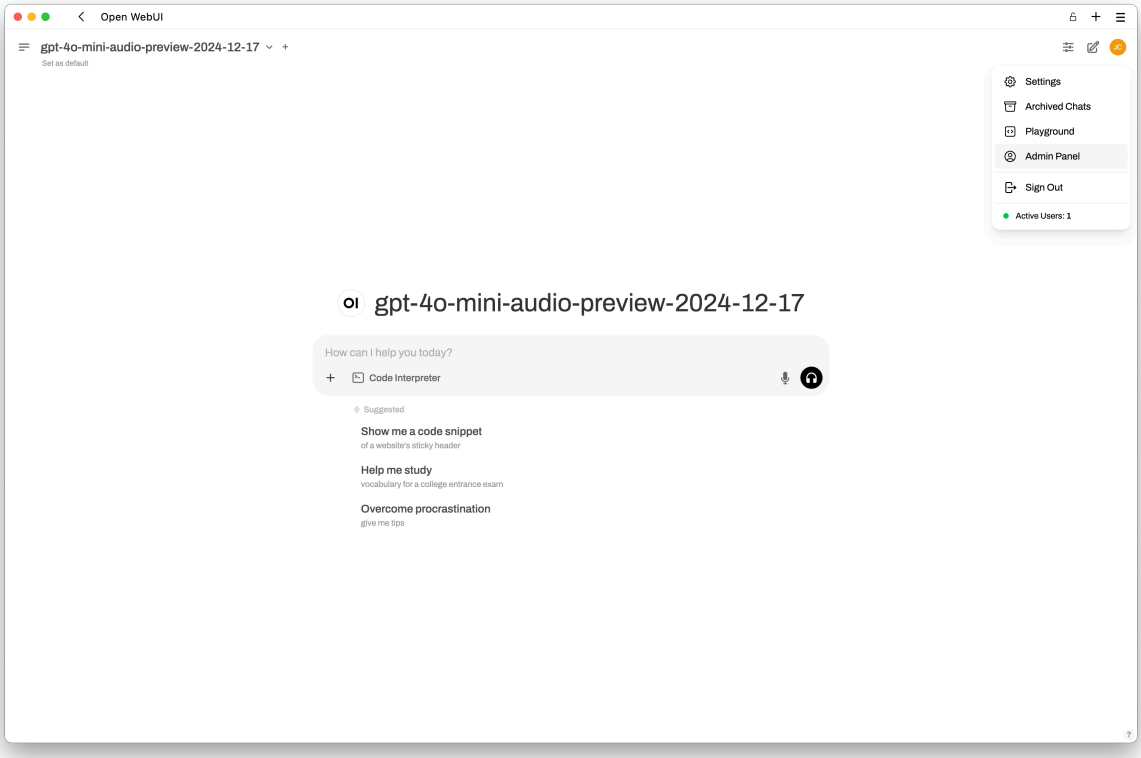


Figure 6.2.: Admin Settings

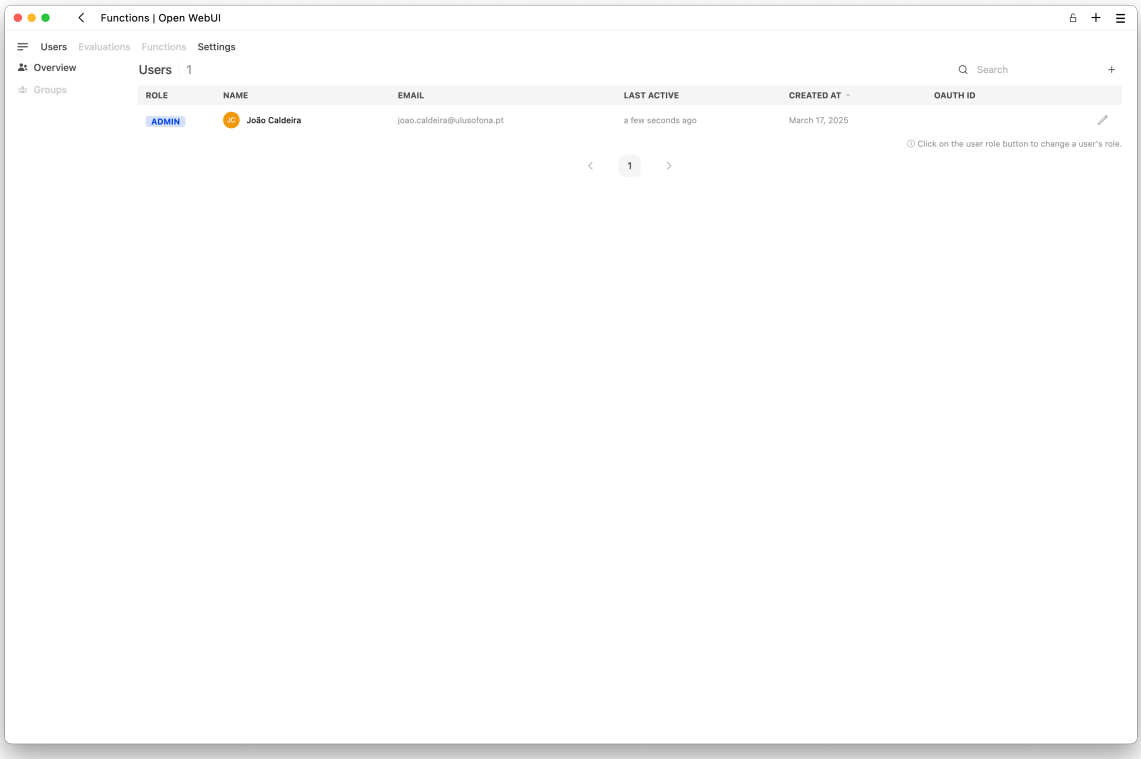


Figure 6.3.: Connections Settings

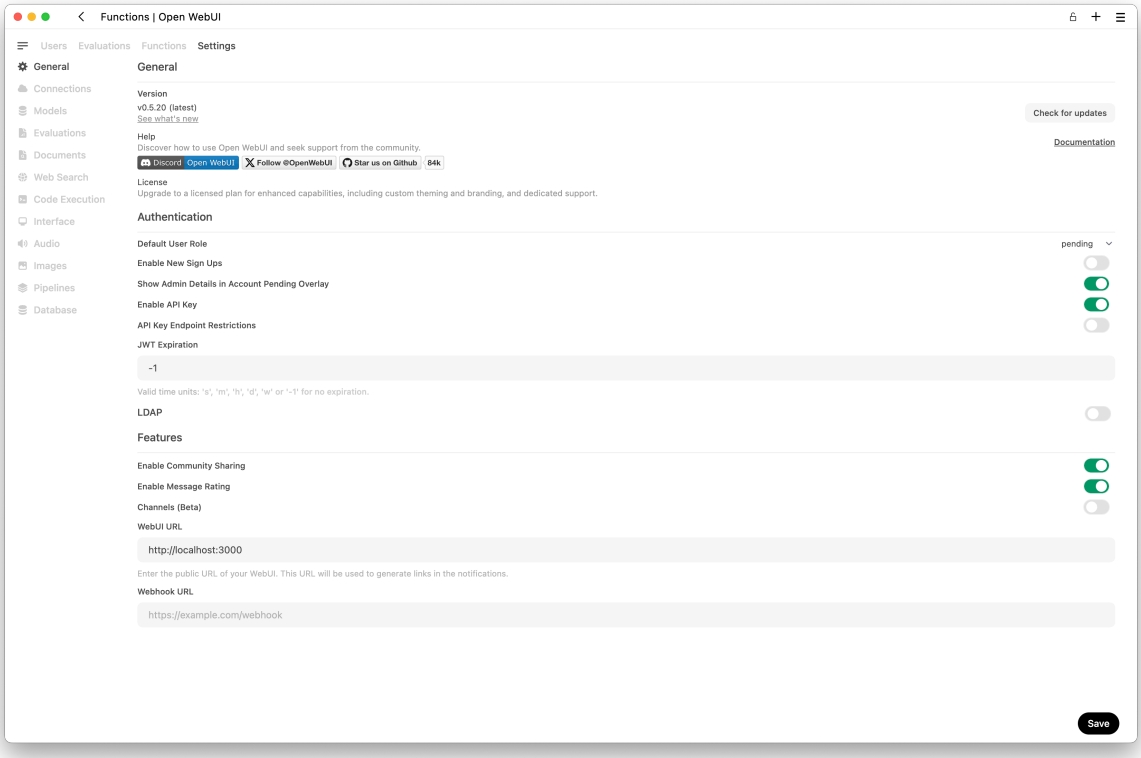


Figure 6.4.: Connections Settings

- When you're on this page, you can press the + button to add another connection.

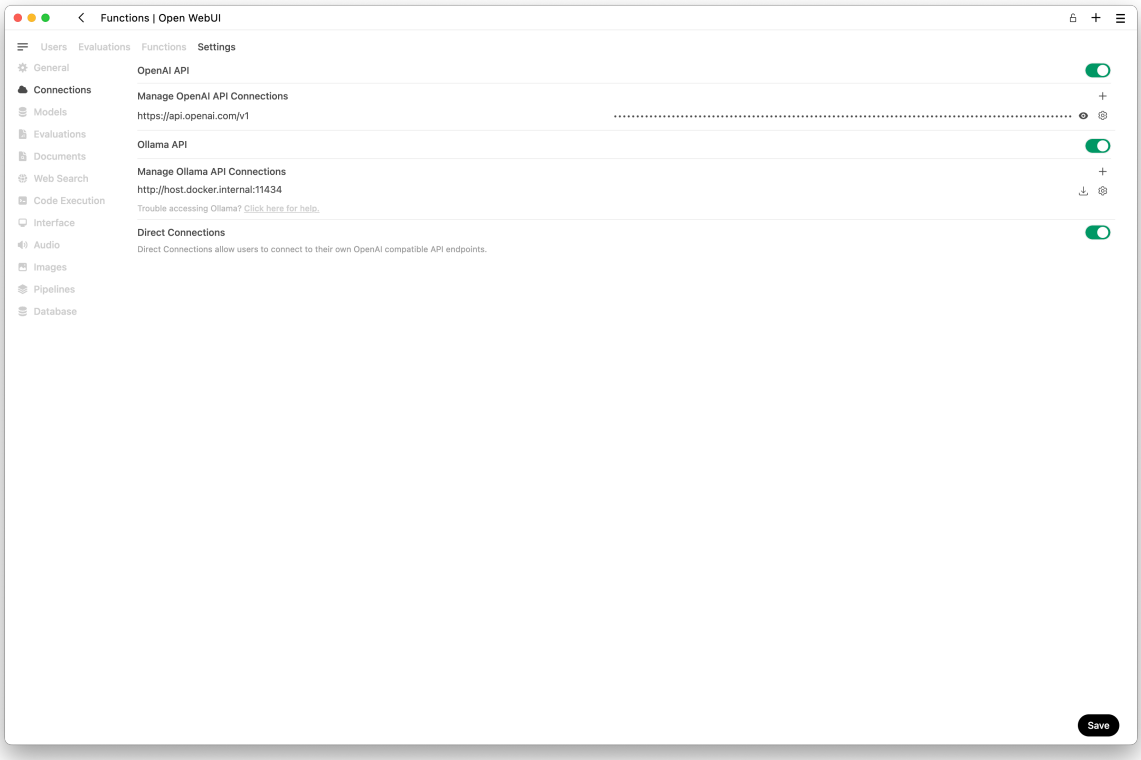


Figure 6.5.: Connections Settings

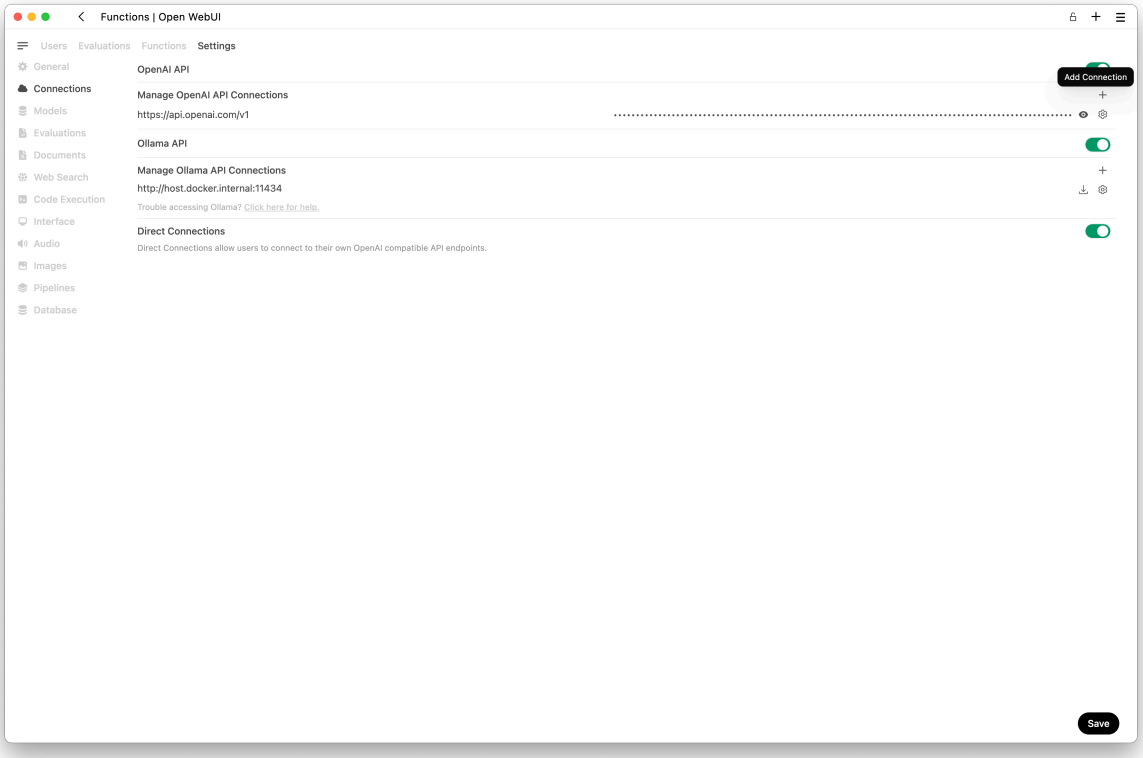


Figure 6.6.: Connections Settings

- Set the API URL to `http://host.docker.internal:9099` and the API key to `0p3n-w3bu!` .
- Verify your connection.

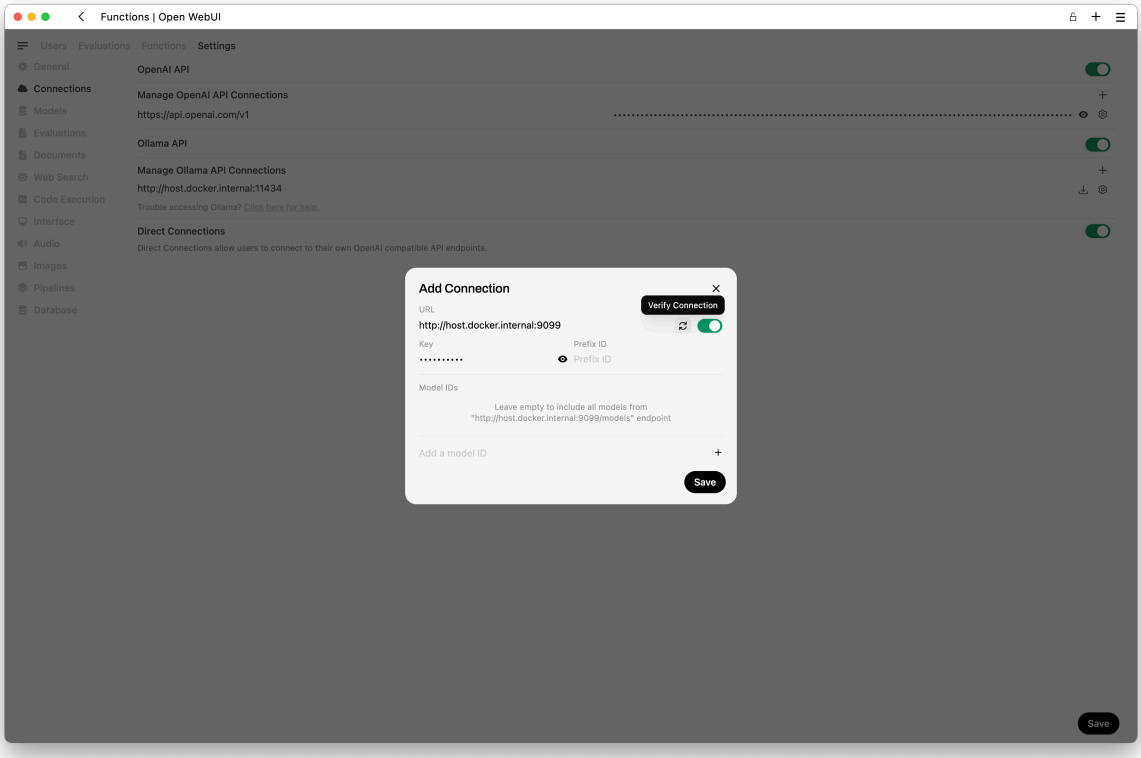


Figure 6.7.: Connections Settings

Once you've added your pipelines connection and verified it, you will see a screen similar to this one.

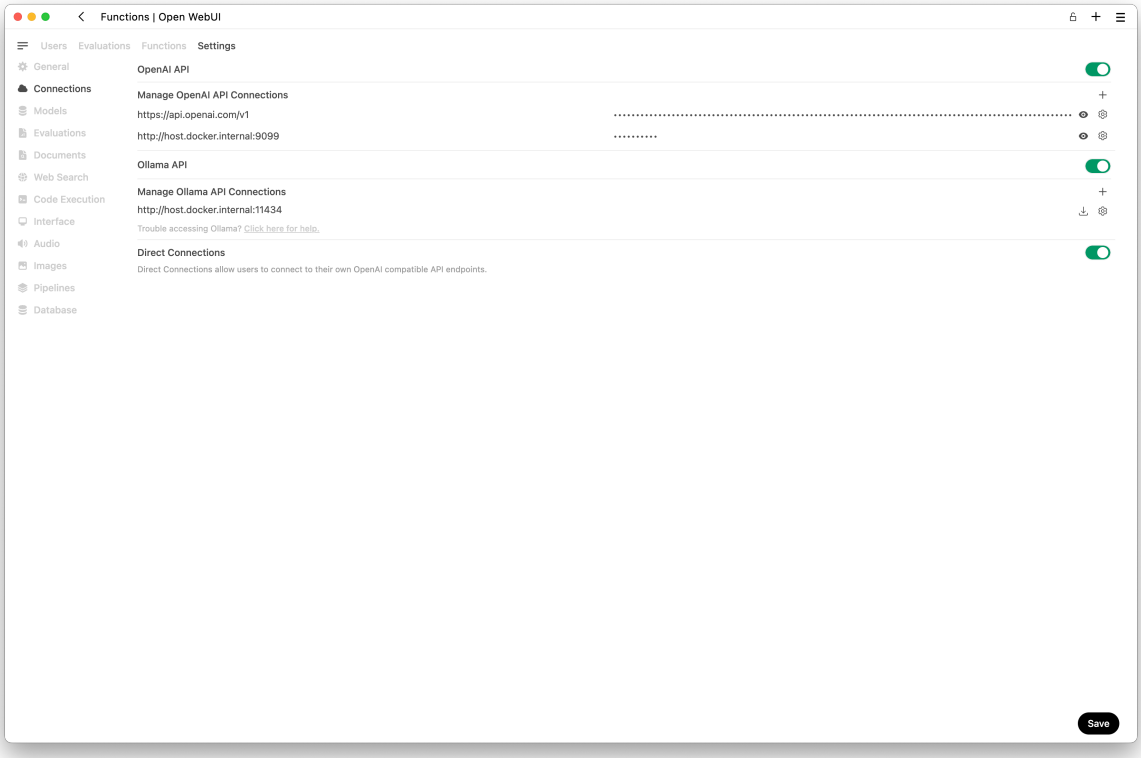


Figure 6.8.: Connections Settings

- Navigate to the `Admin Panel` > `Settings` > `Pipelines` section in Open WebUI.
- You should have a screen similar to this one.

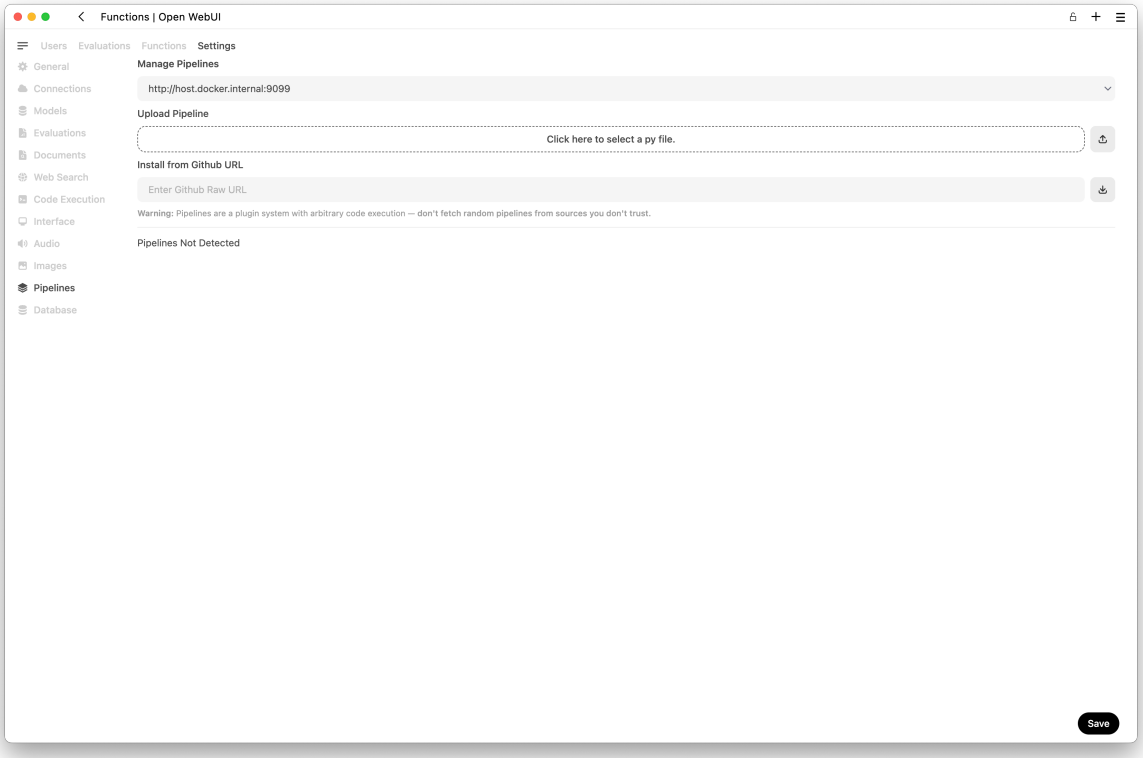


Figure 6.9.: Pipelines Settings

- Download the RAG pipeline file from here. The pipeline file it is already prepared with the code for this workshop.
- Uncompress the file and select it to the Open WebUI interface.
- Press the `upload` button on the right side of the panel.

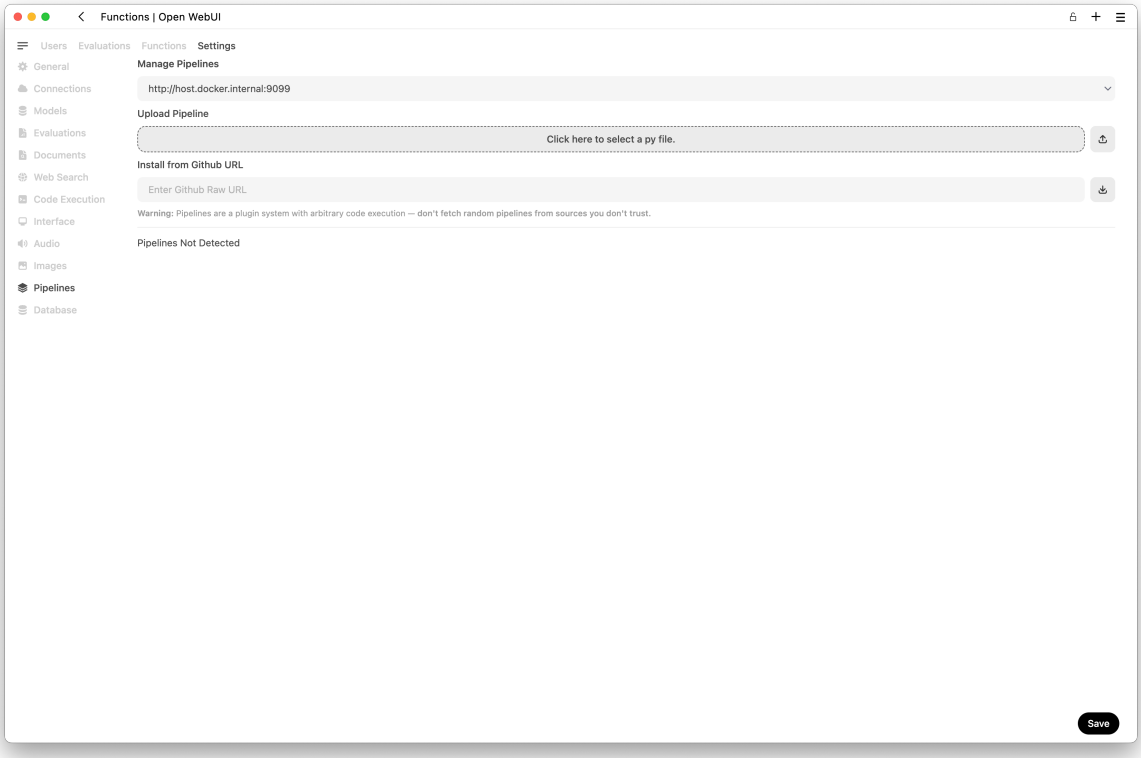


Figure 6.10.: Pipelines Settings

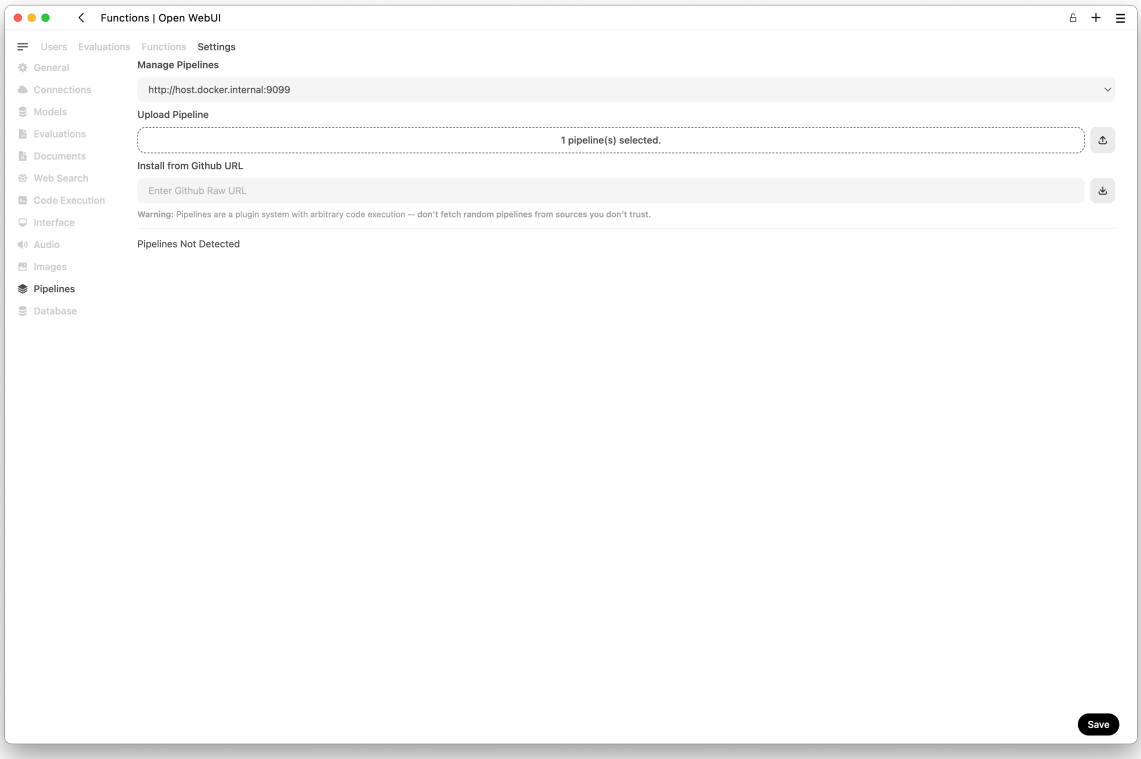


Figure 6.11.: Pipelines Settings

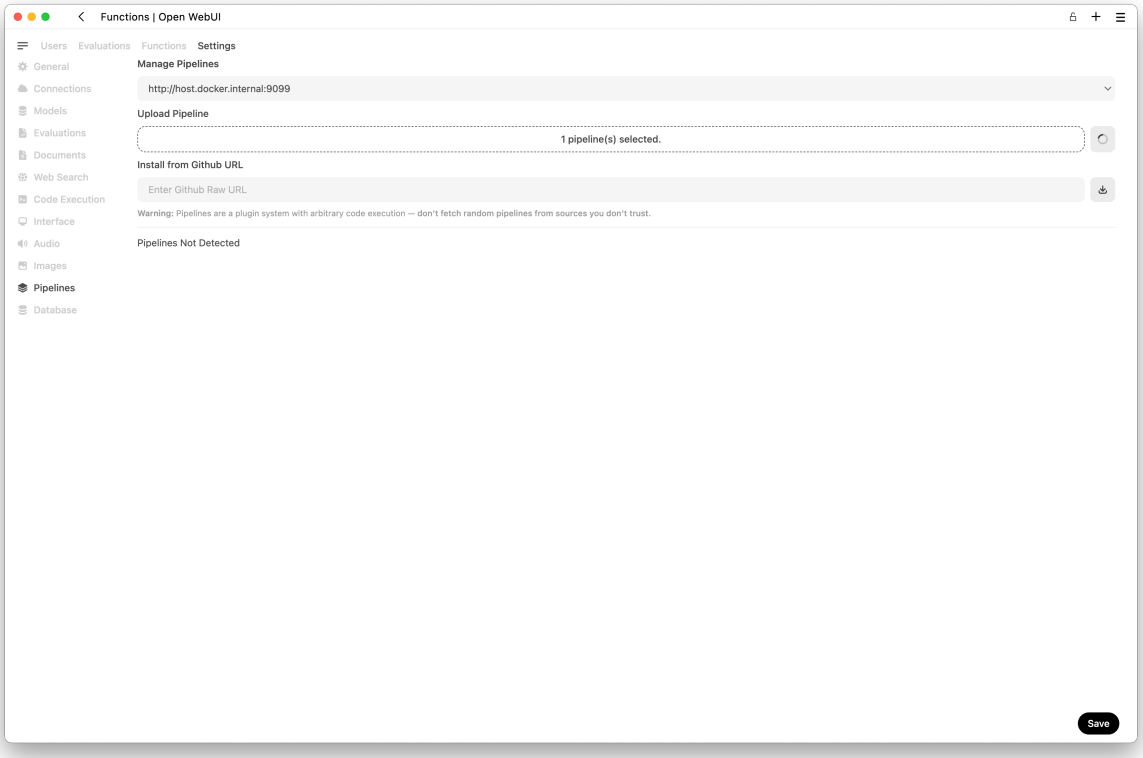


Figure 6.12.: Pipelines Settings

- You should see a screen like the following.

> **!** Important
>
> If you don't see a screen like the following, run the following commands:
> `docker exec -it rag /bin/bash`
> `pip install -r requirements.txt`
>
> - Exit and restart the `rag` container and go back to the same steps.
>   - Uncompress the file and select it to the Open WebUI interface.
>   - Press the `upload` button on the right side of the panel.

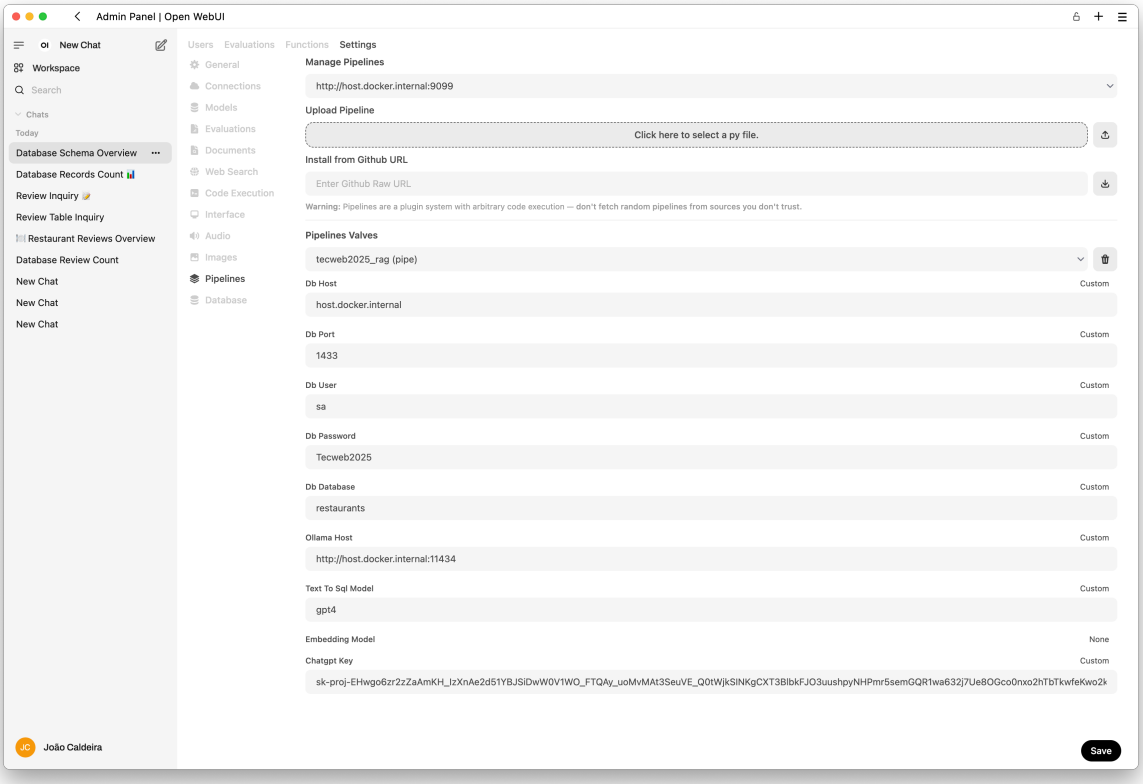


Figure 6.13.: Pipelines Settings

- If you want to use Ollama models, change ChatGPT Key field from `Custom` to `None` and enter an existing Ollama model name (should be download first in the LLMs container) in the field `Text to SQL Model`.
- If you want to use ChatGPT integration, keep the field as `Custom` and enter a ChatGPT Key and a OpenAI model in the field `Text to SQL Model`.
- Add a `ChatGPT Key`, similar to this one, in the last field of the form.

> **!** Important
>
> This ChatGpt Key is no longer valid. It's used here only as an example.

```
sk-proj-XlXRiWlfetudLBrRrP6B4C_RRjDpJu03LJ9smXeDz3p-gE7PLg-a1Td6qFCWoydnIL2pxhJL6dT3BlbkFJ5E
```

- Press the button `Save` at the bottom of the screen.
- Restart the `rag` (pipelines) container in `Docker`.
- Connect to your `rag` container with this command:

```
docker exec -it rag /bin/bash
```

- Once inside the `rag` container run the following command to install missing libraries.

```
pip install -r requirements.txt
```

- Exit and restart the `rag` container.
- At the end of installation, your `Docker` should look like this.

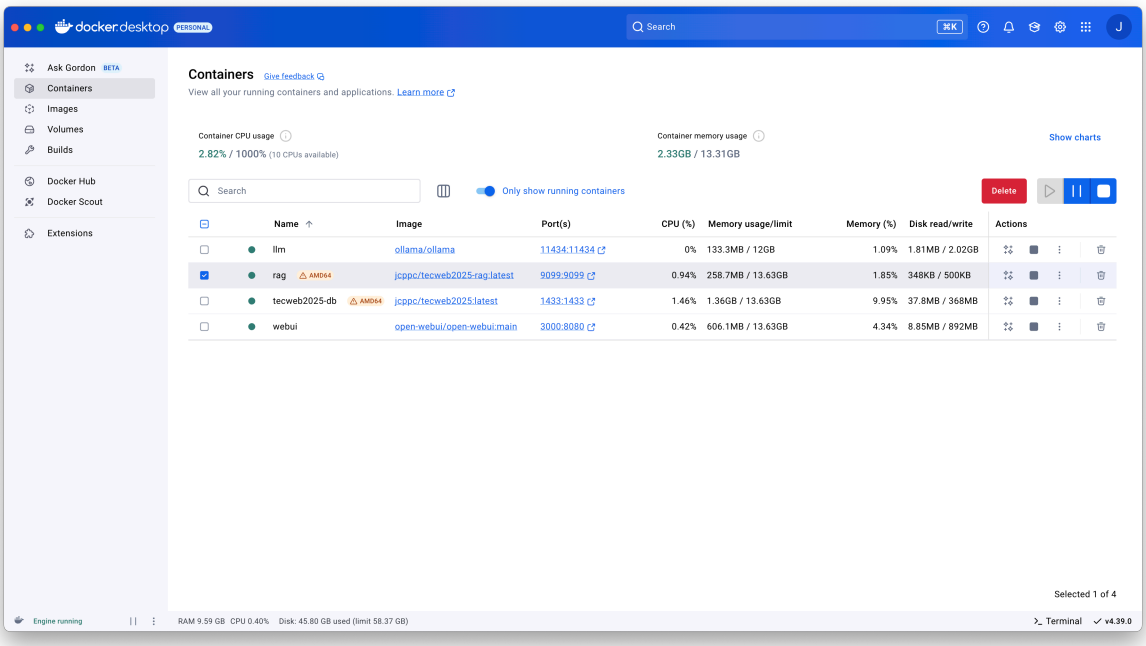


Figure 6.14.: Docker Environment

- Access again your main WebUI console and select the `Model` > `Tecweb2025` .

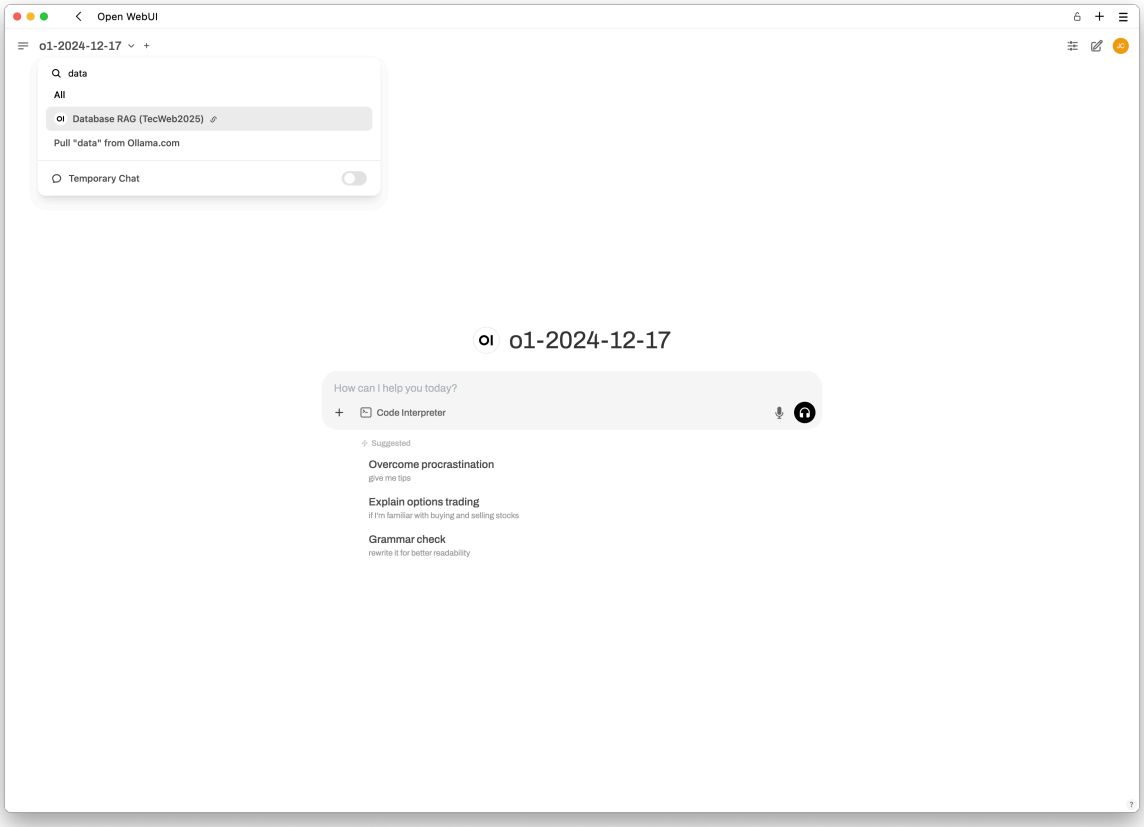


Figure 6.15.: RAG Model Selection

- You can set this as your `default model` if you wish.

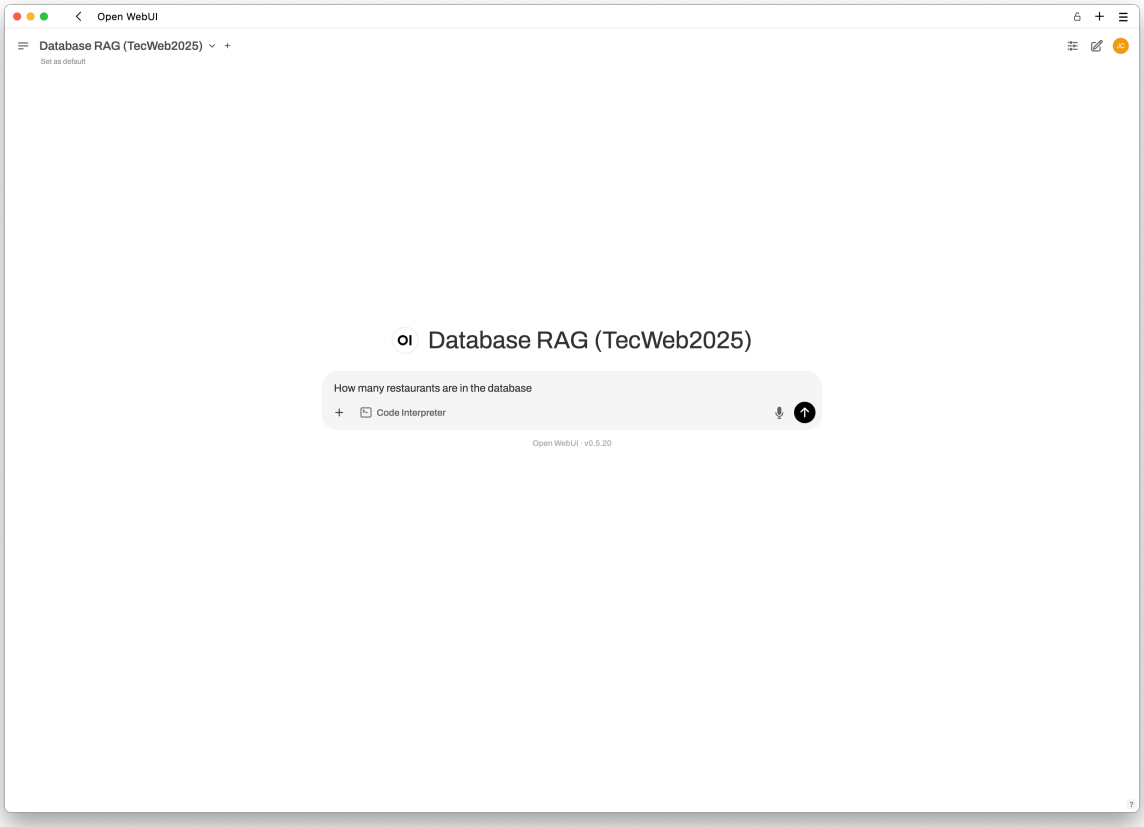


Figure 6.16.: RAG Model Selection

- Start to ask questions.

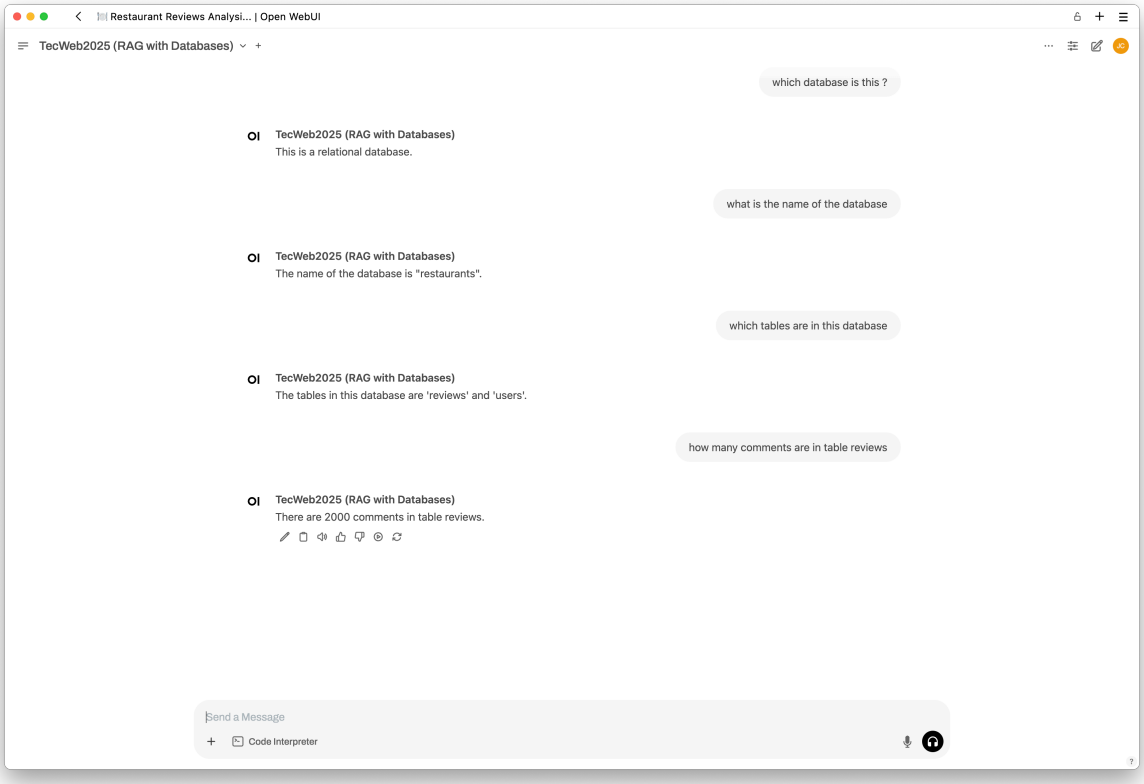


Figure 6.17.: Ask Questions

Congratulations

You are now using LLMs & RAG to run queries on databases.

UNIVERSIDADE
LUSÓFONA

