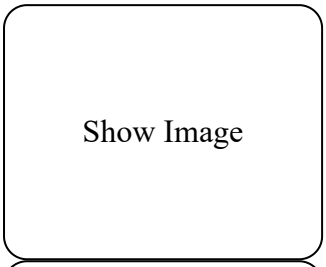
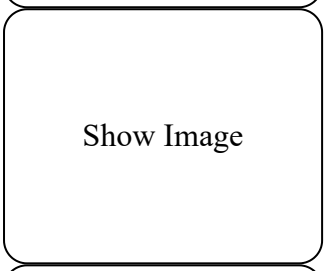
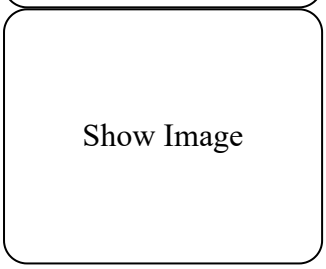
🐳 AI-Powered SQL Chat Application - Docker Setup Guide

A conversational AI application that allows users to interact with MySQL databases using natural language. This guide shows how to run the application using Docker.







📋 Prerequisites

Before running this application, ensure you have:

1. **Docker Desktop** installed on your computer
   - Windows: Download Docker Desktop for Windows
   - Mac: Download Docker Desktop for Mac
   - Linux: Install Docker Engine
2. **Docker Desktop must be running** (check for the whale icon in your system tray)
3. **Groq API Key** (Free - instructions below)

🔑 Step 1: Get Your Free Groq API Key

1. Visit console.groq.com
2. Sign up for a free account (no credit card required)
3. Navigate to **API Keys** section

4. Click **Create API Key**
5. Copy the key (starts with `gsk_...`)
6. Keep it handy for the next step

📁 Step 2: Verify Project Files

Your project folder should contain these files:

```
project-folder/
├── app.py                # Main application
├── .env                  # API keys (you'll create this)
├── BankingDB.sql         # Banking database
├── Chinook.sql           # Music store database
├── requirements.txt      # Python dependencies
├── Dockerfile            # Docker configuration
├── docker-compose.yml    # Multi-container setup
├── .dockerignore         # Docker ignore rules
└── README.md             # This file
```

⚙️ Step 3: Configure Environment Variables

1. Open the project folder in a text editor or terminal
2. Create a file named `.env` (exactly, with the dot at the start)
3. Add the following line:

```
env

GROQ_API_KEY=your_actual_groq_api_key_here
```

Replace `your_actual_groq_api_key_here` with the API key you got from Step 1.

Example:

```
env

GROQ_API_KEY=gsk_abc123xyz456def789ghi012jkl345mno678pqr901stu234
```

Important Notes:

- The file must be named `.env` exactly (not `.env.txt`)
- No spaces around the `=` sign

- Keep this file private - don't share your API key

🚀 Step 4: Run the Application

For Windows:

1. Open **Command Prompt** or **PowerShell**
2. Navigate to the project folder:

```
cmd

  cd path\to\project-folder
```

3. **IMPORTANT**: Stop local MySQL if running:

```
cmd

  net stop MySQL80
```

(This prevents port conflicts)

4. Build and start the application:

```
cmd

  docker-compose up --build
```

For Mac/Linux:

1. Open **Terminal**
2. Navigate to the project folder:

```
bash

  cd /path/to/project-folder
```

3. **IMPORTANT**: Stop local MySQL if running:

```
bash

  sudo service mysql stop
```

4. Build and start the application:

```
bash
```

```
docker-compose up --build
```

⏳ Step 5: Wait for Initialization

You'll see logs in the terminal. Wait for these messages:

```
✓ Container banking_mysql         Started
✓ Container banking_streamlit_app Started

You can now view your Streamlit app in your browser.
Local URL: http://localhost:8501
```

This takes 30-60 seconds as MySQL initializes both databases.

🌐 Step 6: Open the Application

Open your web browser and go to:

```
http://localhost:8501
```

You should see the **Chat with MySQL** interface!

🗄️ Step 7: Connect to a Database

In the application sidebar, enter these details:

For Banking Database:

```
Host: mysql
Port: 3306
User: root
Password: admin
Database: BankingDB
```

For Chinook Database:

```
Host: mysql
Port: 3306
User: root
```

Password: admin
Database: Chinook

Click **Connect** and wait for the success message.

💬 Step 8: Start Asking Questions!

Example Questions for BankingDB:

- "Show me all customers"
- "What is the total balance across all accounts?"
- "Which customers have the highest loan amounts?"
- "List all transactions from November 2024"
- "Show me all active credit cards"

Example Questions for Chinook:

- "Which artists have the most albums?"
- "Show me the top 10 best-selling tracks"
- "List all customers from Brazil"
- "What are the most popular music genres?"
- "Show me total sales by country"

🛑 Stopping the Application

To stop the containers:

Press Ctrl + C in the terminal where docker-compose is running

Then run:

```bash
docker-compose down
```

To start again later (without rebuilding):

```bash
docker-compose up
```

To completely remove everything (including data):

```bash
```

```
docker-compose down -v
```

🔧 Troubleshooting

Issue 1: "Port 3306 already in use"

Problem: Local MySQL is running **Solution:**

```
bash

# Windows
net stop MySQL80

# Mac/Linux
sudo service mysql stop
```

Issue 2: "Empty compose file"

Problem: `docker-compose.yml` not found or incorrectly named **Solution:**

- Verify the file exists in the project folder
- Ensure it's named exactly `docker-compose.yml`
- Check it's not saved as `docker-compose.yml.txt`

Issue 3: Blank page in browser

Problem: Browser cache or CORS issue **Solution:**

1. Clear browser cache (Ctrl + Shift + Delete)
2. Try incognito mode (Ctrl + Shift + N)
3. Try a different browser
4. Access using: `http://127.0.0.1:8501`

Issue 4: "GROQ_API_KEY not set"

Problem: `.env` file missing or incorrect **Solution:**

1. Verify `.env` file exists in project root
2. Check the API key is correct
3. Restart containers: `docker-compose restart`

Issue 5: Containers keep restarting

Problem: Check logs for errors **Solution:**

```bash
docker-compose logs streamlit
docker-compose logs mysql
```

Share the error messages for further help.

Issue 6: Database connection failed

Problem: MySQL not fully initialized **Solution:**

- Wait 30 seconds after starting containers
- Verify database name is correct (case-sensitive)
- Check credentials: `root` / `admin`

📊 Checking Container Status

View running containers:

```bash
docker ps
```

You should see:

- `banking_mysql`
- `banking_streamlit_app`

View container logs:

```bash
```

```
# All logs
docker-compose logs

# Streamlit logs only
docker-compose logs streamlit

# MySQL logs only
docker-compose logs mysql

# Follow logs in real-time
docker-compose logs -f
```

Verify databases are created:

```
bash

docker exec -it banking_mysql mysql -uroot -padmin -e "SHOW DATABASES;"
```

You should see:

- BankingDB
- Chinook

🎯 What's Inside the Containers?

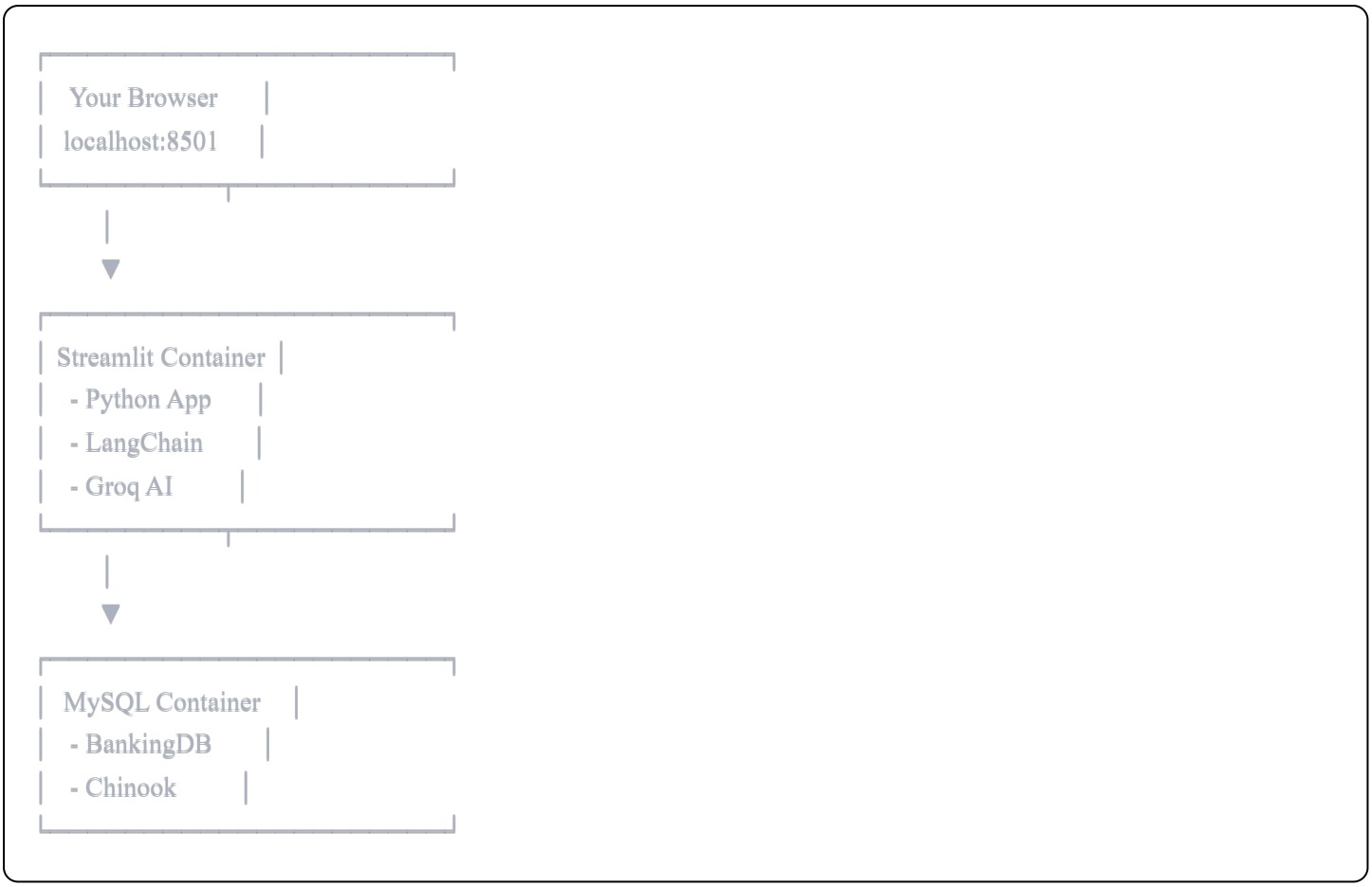
MySQL Container:

- MySQL 8.0 server
- Two pre-loaded databases:
  - **BankingDB**: Banking system with customers, accounts, loans, transactions
  - **Chinook**: Music store with artists, albums, tracks, customers, sales
- Persistent data storage

Streamlit Container:

- Python 3.12
- Streamlit web application
- LangChain AI framework
- Groq AI integration
- All required dependencies

🏗️ Architecture



🔄 Rebuilding After Changes

If you modify the code:

```bash
# Stop containers
docker-compose down

# Rebuild and start
docker-compose up --build
```

📦 What Docker Does

Docker handles everything automatically:

- ✅ Installs Python 3.12
- ✅ Installs all Python dependencies
- ✅ Sets up MySQL 8.0 server
- ✅ Creates both databases
- ✅ Loads sample data

✅ Configures networking between containers
✅ Exposes the app on port 8501

No manual installation needed!

🎓 For Mentors/Reviewers

Quick Start (3 commands):

```bash
# 1. Navigate to project folder
cd path/to/project-folder

# 2. Add your Groq API key to .env file
echo GROQ_API_KEY=your_key_here > .env

# 3. Run everything
docker-compose up --build
```

Then open: **http://localhost:8501**

💡 Tips

- First startup takes 1-2 minutes (building images + database initialization)
- Subsequent starts are much faster
- Data persists between restarts
- Use `docker-compose down -v` only if you want to reset everything
- Keep Docker Desktop running while using the app

✅ Success Checklist

Before presenting, verify:

- [ ] Docker Desktop is running
- [ ] `.env` file has valid Groq API key
- [ ] All project files are present
- [ ] Ran `docker-compose up --build` successfully
- [ ] Can access `http://localhost:8501` in browser
- [ ] Can connect to both BankingDB and Chinook
- [ ] Can ask questions and get AI responses

📞 Quick Reference

Component	URL/Command
Application	http://localhost:8501
Health Check	http://localhost:8501/_stcore/health
Start Containers	`docker-compose up`
Stop Containers	`docker-compose down`
View Logs	`docker-compose logs -f`
Rebuild	`docker-compose up --build`

🌟 Technologies Used

- **Docker & Docker Compose** - Containerization
- **Python 3.12** - Programming language
- **Streamlit 1.29.0** - Web framework
- **MySQL 8.0** - Database
- **LangChain** - AI orchestration
- **Groq (LLama 3.3 70B)** - AI model
- **SQLAlchemy** - Database ORM

Ready to demonstrate! 🚀

For any issues during setup, check the Troubleshooting section above.