

AthenaAI: User Manual & System Installation Guide

Introduction

- **Project Name:** Virtual Psychologist
- **Overview:** This project aims to build a personalized and accessible mental health support system using Generative Artificial Intelligence (GenAI). The system is designed to provide therapeutic interactive sessions based on Cognitive Behavioral Therapy (CBT).

The project's architecture consists of the following main components:

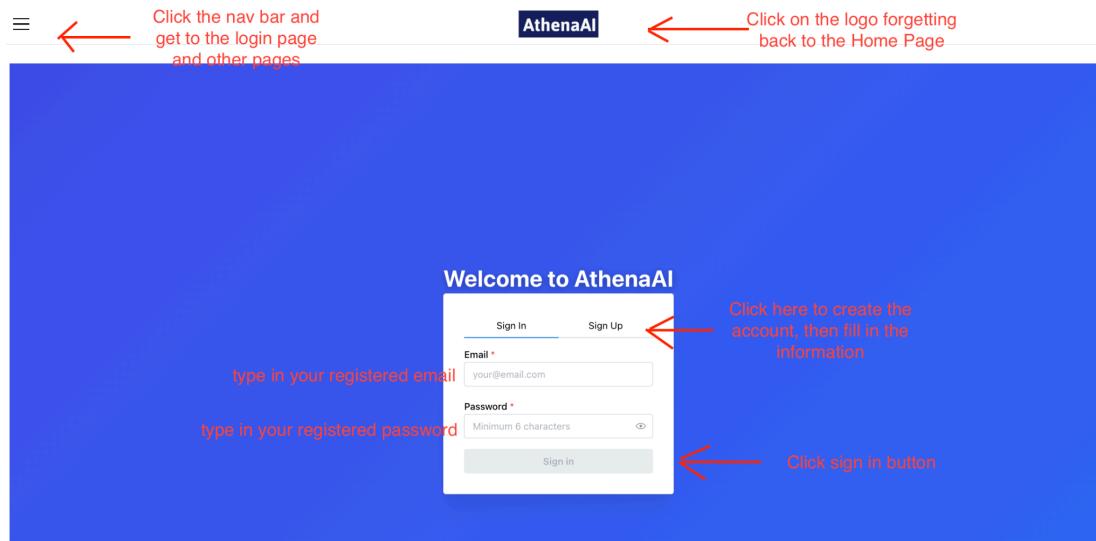
- **Frontend:** The user interface is built with **React** and **TypeScript** to ensure a safe, efficient, and interactive user experience.
- **Backend:** The server is developed with **Node.js**, responsible for handling business logic, managing user data, and acting as a communication intermediary.
- **AI Service:** A dedicated AI service built with **Python**, tasked with integrating and orchestrating multiple API keys from various Large Language Models (LLMs) to generate intelligent therapeutic responses.

Part 1: User Manual

- **Audience:** This guide is intended for end-users and does not require technical knowledge.

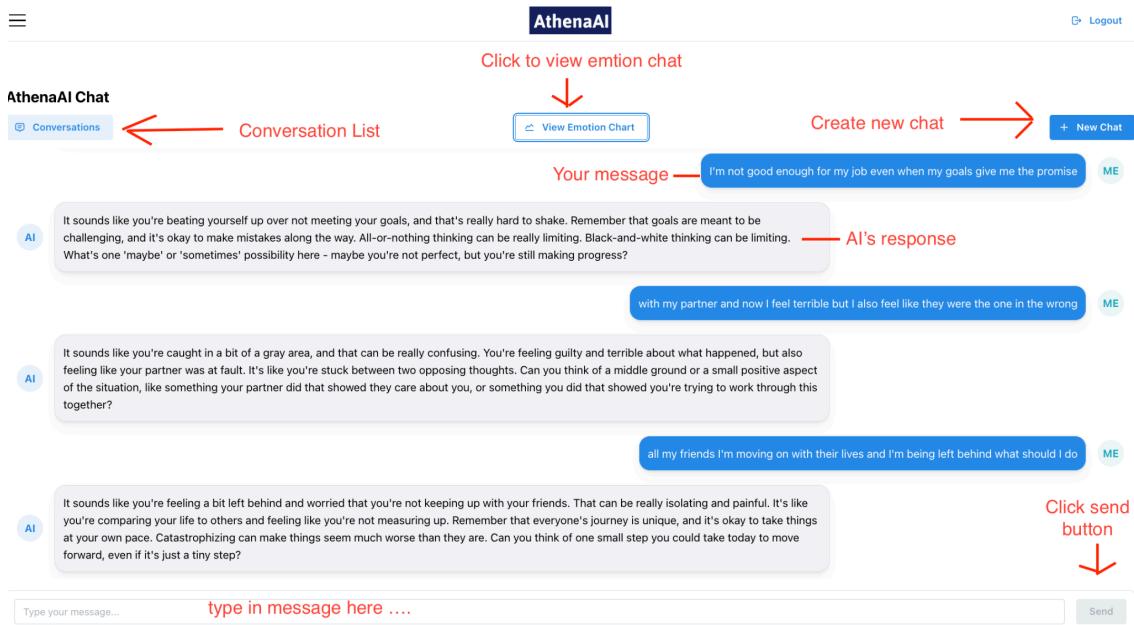
1.1. Getting Started

- **Creating an Account & Logging In:** Step-by-step instructions for registering a new account and logging into the platform.

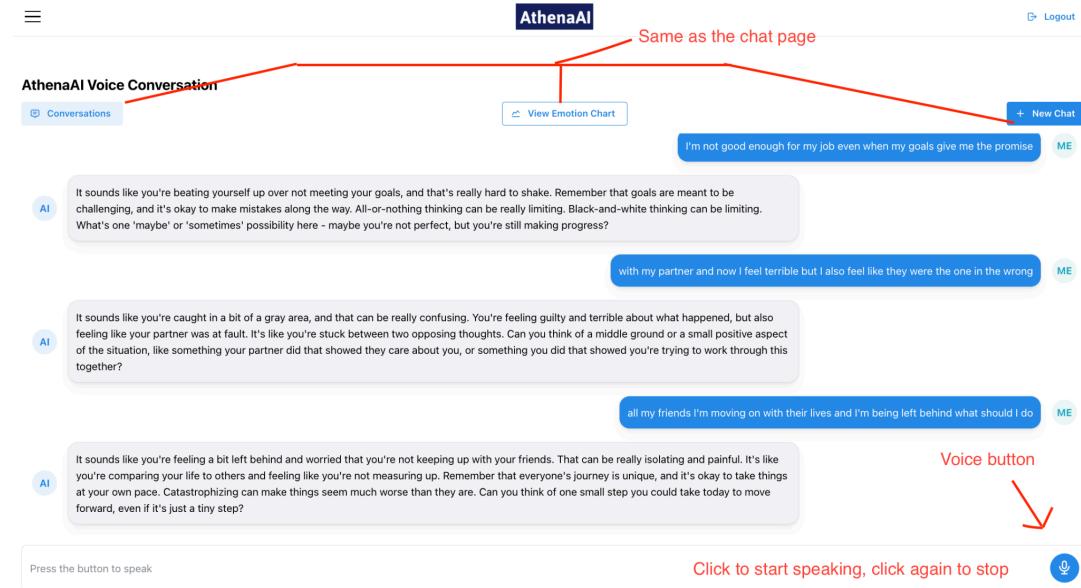


1.2. Main Features and Pages

- **Informational Pages:**
 - A brief overview of the site's static pages: Home, Blog, Service, and Contact, which provide product and project information.
- **Chat Page:**
 - The primary feature for interacting with the virtual psychologist via text.
 - How to send a message and receive a CBT-based response.
 - How to use the "View Emotion Chart" feature to see an emotional analysis of the conversation.



- **Voice Page:**
 - The main feature for voice-based interaction.
 - How to press the button to speak and receive an empathetic, spoken response from the bot.
 - Includes the "View Emotion Chart" feature for conversation analysis.



- **Profile Page:**
 - How to view your account information.
 - How to access and review your entire conversation history.

AthenaAI

Logout

Total conversations
47

Messages (sum)
318

Recent active
1

User AUTHENTICATED

No description provided yet.

Change profile image

Address

Email

Phone

DOB

Profile data is synced to your account

To-do checklist
0/0 complete

Add an item
e.g., Verify email, try TT!

Add

Chat history

Access the latest chat created

Go to chat

Title	Last activity	Messages	Action
Conversation	10/10/2025, 10:17:28 (2h ago)	4	Resume
Conversation	06/10/2025, 19:52:10 (3d ago)	2	Resume
Conversation	06/10/2025, 19:49:30 (3d ago)	14	Resume
Conversation	06/10/2025, 10:03:16 (4d ago)	10	Resume
Conversation	06/10/2025, 09:50:40 (4d ago)	10	Resume
Conversation	06/10/2025, 09:48:06 (4d ago)	2	Resume

Resume chat/voice conversation

Add To-do checklist tasks

- **Productivity Page:**

- A dedicated space for users to write and manage notes and comments.

AthenaAI

Dashboard > Productivity

Create Reminder

Notes: 2

Reminders: 1

Pinned notes: 0

Clear all notes

Sticky notes

Blue: Text
hello world

Green: Text
Hello AthenaAI bot

Updated 10/10/2025, 13:22:25

Updated 10/10/2025, 13:22:52

New note Snap to grid

notes Create new note

Reminders History

- **Tutorial Page (base on your request):**

- An in-depth manual embedded within the application, providing detailed instructions for all features.

Athena

Tutorial

GETTING STARTED

Progress: 0/6 complete

Overview Setup Usage Guide Chat Voice (TTS) Security FAQ

What this page covers

You'll connect the **backend**, run the **frontend**, sign in, send a chat, and enable **Text-to-speech**. If you get stuck, jump to the **FAQ** tab.

Fast links

- Dashboard
- Profile
- Chat
- Voice

Tip: Use `Ctrl + C` to stop servers, and re-run after config changes.

Click a tab to see the guide for each section.

Checklist

0% COMPLETE

Environment set up (Node, pnpm/npm, .env)

Backend running (Express + PostgreSQL reachable)

Part 2: System Installation & Deployment Manual

- **Audience:** This guide is intended for developers who wish to run the project in a local environment.

2.1. Architecture Overview

- A brief description of the project's 3-service architecture:
 - `athenaos-frontend`: Role, technologies (React, TypeScript).
 - `athenaos-backend`: Role, technologies (SQLite).
 - `athenaos-ai-service`: Role, technologies (Python, Fast API/Flask).

2.2. Prerequisites

- **Software:**
 - Node.js: v20.14.0
 - Python: 3.13.5
 - pnpm (or npm/yarn).
 - Git.
- **Database:** SQLite

2.3. Local Installation Guide

- **Step 1: Get the Source Code (Download or Clone)**

You can choose one of the following two methods to get the project source code:

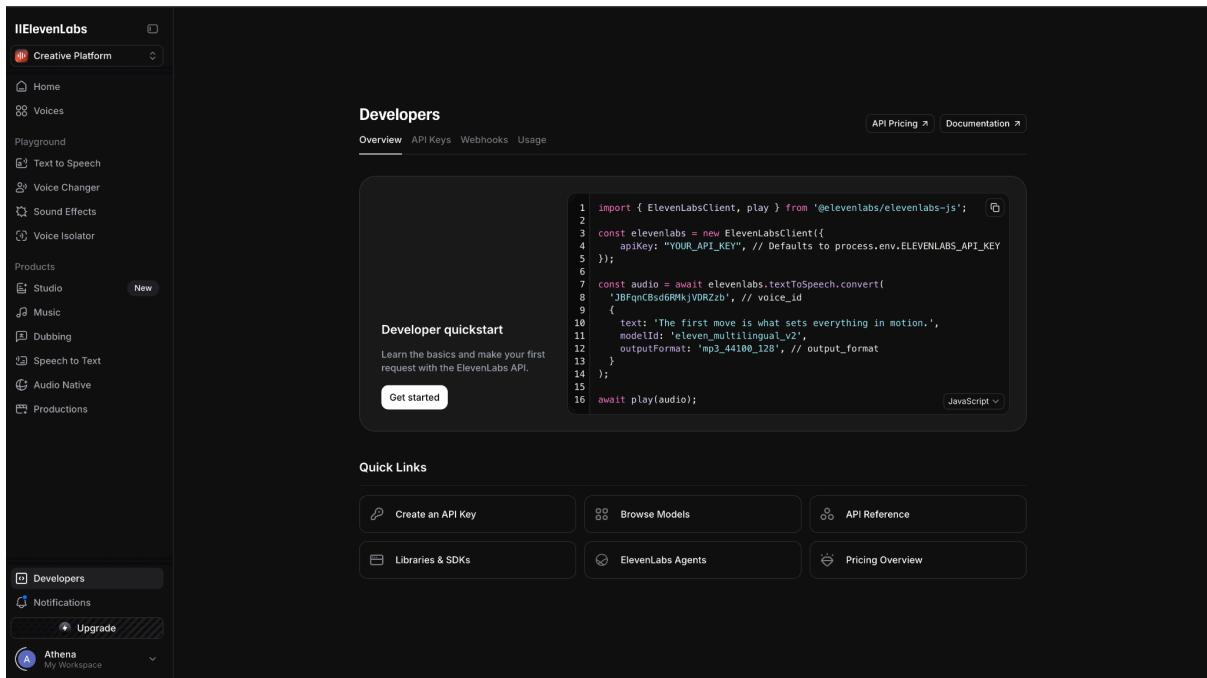
- **Option 1 (Recommended): Clone from GitHub Repository** Open your terminal and run the following command to clone the project from GitHub:
`git clone https://github.com/AthenaAI-Virtual24/AthenaAI-Local.git`
- This will create the `AthenaAI-Local` folder containing all three service directories.
- **Option 2: Download ZIP file from Google Drive**
 1. Access the Google Drive link:
https://drive.google.com/drive/folders/19HZZ7a830DNNLUD0_pfEvjK3-7ogddUU?usp=drive_link
 2. Download the project's .zip file.
 3. Unzip the file to a location of your choice on your local machine. This will create the main project folder containing the three service directories
- **Step 2: Configure Environment Variables**
 - For each service (frontend, backend, ai-service), locate the `.env` file and follow the instructions below to fill in the required values.
 - **For the Backend (`athenaos-backend`):**
 - **Standard Variables:** Based on the provided `.env` file, ensure the following variables are set. For security, it is highly recommended to change the `JWT_SECRET` to your own unique, complex string.

- **ElevenLabs API Key (for Text-to-Speech):** This service requires an API key from ElevenLabs for voice and Text-to-Speech functionality.

1. Go to the ElevenLabs website (<https://elevenlabs.io/>) and Sign up or Log in.
2. In the bottom-left corner of the dashboard, click on the **Developers** option (located just above "Notifications" and "Upgrade").
3. On the "User API keys" page, click the **+ Create Key** button.
4. Give your key a name (e.g., "Athena").
5. **Important:** Copy the newly generated API key (it will only be shown once upon creation) and save it immediately.
6. After the key is created, find it in the list and click the **Edit** icon (the pencil symbol) next to it.
7. The "Edit API Key" menu will appear on the right. Ensure the following permissions are granted by toggling them to **"Access"**:
 - Text to Speech

"Read":

- User
 - History
 - Models
8. Click "Save Changes" at the bottom of the menu.
 9. In the `.env` file for `athenaos-backend`, paste the **copied key (from step 5)** into the `ELEVENLABS_API_KEY` variable: `ELEVENLABS_API_KEY=your_copied_api_key_here`



- **For the AI Service (athenaos-ai-service):**
 - The .env file for this service requires two API keys.
 - **Groq API Key (for Llama 3.1 Model):**
 1. Go to the **Groq Console** (<https://console.groq.com/>) and log in.
 2. Navigate to the **API Keys** tab and click **Create API Key**.
 3. Name your key and copy the generated key string (it will only be shown once).
 4. In the .env file for **athenaos-ai-service**, paste the **copied key (from step 3)** into the **GROQ_API_KEY** variable:
`GROQ_API_KEY=your_copied_api_key_here`

The screenshot shows the Groq API Keys management interface. At the top, there's a navigation bar with 'Playground', 'API Keys' (which is highlighted in red), 'Dashboard', 'Docs', and other icons. Below the navigation is a section titled 'API Keys' with a sub-instruction: 'Manage your project API keys. Remember to keep your API keys safe to prevent unauthorized access.' A 'Create API Key' button is visible. The main area displays a table of API keys:

NAME	SECRET KEY	CREATED	LAST USED	USAGE (24HRS)	Actions
Llama	gsk_...n9mw	25/09/2025	04/10/2025	0 API Calls	
Llama3.1	gsk_...XFFB	04/10/2025	10/10/2025	4 API Calls	

- **Hugging Face Access Token (for supporting models):**
 1. Go to **Hugging Face** (<https://huggingface.co/>) and log in.
 2. Navigate to your profile **Settings > Access Tokens**.
 3. Click **New token**, give it a name, and select the **Read** role.
 4. Copy the generated token (it will only be shown once).
 5. In the .env file for **athenaos-ai-service**, paste the **copied key (from step 4)** into the **HUGGINGFACE_API_KEY** variable:
`HUGGINGFACE_API_KEY=your_copied_api_key_here`

- **For the Frontend (athenaos-frontend):**

- In the `.env` file, ensure the `VITE_API_BASE_URL` variable is set to the correct address of your running backend service (e.g., `http://localhost:8888`).

- **Step 3: Run the Backend Service**

- Open terminal 1, navigate to `athenaos-backend`, install dependencies (`npm install`), and run the start command (`npm start`).

```
● thuanduc@Thuans-MacBook-Pro athenaos-backend % npm install
  up to date, audited 286 packages in 711ms

  37 packages are looking for funding
    run `npm fund` for details

  found 0 vulnerabilities
○ thuanduc@Thuans-MacBook-Pro athenaos-backend % npm start

  > start
  > node server.js

  Running in development mode, using SQLite.
  [dotenv@17.2.2] injecting env (3) from .env — tip: 🗝 encrypt with Dotenvx: https://dotenvx.com
  Executing (default): SELECT 1+1 AS result
  Database connection has been established successfully.
  Executing (default): SELECT name FROM sqlite_master WHERE type='table' AND name='Users';
  Executing (default): PRAGMA INDEX_LIST('Users')
  Executing (default): PRAGMA INDEX_INFO('sqlite_autoindex_Users_1')
  Executing (default): PRAGMA INDEX_INFO('sqlite_autoindex_Users_2')
  Executing (default): SELECT name FROM sqlite_master WHERE type='table' AND name='Conversations';
  Executing (default): PRAGMA INDEX_LIST('Conversations')
  Executing (default): SELECT name FROM sqlite_master WHERE type='table' AND name='Messages';
  Executing (default): PRAGMA INDEX_LIST('Messages')
  All models were synchronized successfully.
  Server is running on port 8888
```

- **Step 4: Run the Frontend Service**

- Open terminal 2, navigate to `athenaos-frontend`, install dependencies (`npm install`), and run the start command (`npm run dev`).

```
pyenv shell integration not enabled. Run `pyenv init` for instructions.
● thuanduc@Thuans-MacBook-Pro AthenaAI % cd athenaos-frontend
● thuanduc@Thuans-MacBook-Pro athenaos-frontend % npm install
  npm warn EBADENGINE Unsupported engine {
  npm warn EBADENGINE   package: 'vite@7.1.5',
  npm warn EBADENGINE   required: { node: '^20.19.0 || >=22.12.0' },
  npm warn EBADENGINE   current: { node: 'v20.14.0', npm: '10.7.0' }
  npm warn EBADENGINE }

  up to date, audited 264 packages in 612ms

  59 packages are looking for funding
    run `npm fund` for details

  found 0 vulnerabilities
○ thuanduc@Thuans-MacBook-Pro athenaos-frontend % npm run dev

  > athenaos-frontend@0.0.0 dev
  > vite

  You are using Node.js 20.14.0. Vite requires Node.js version 20.19+ or 22.12+. Please upgrade your Node.js version.

  VITE v7.1.5 ready in 173 ms
  + Local: http://localhost:5173/
  + Network: use --host to expose
  + press h + enter to show help
```

- **Step 5: Run the AI Service**

- Open terminal 3, navigate to athenaos-ai-service, install dependencies (pip install -r requirements.txt), and run the start command (uvicorn main:app --reload).

```
● thuanduc@Thuans-MacBook-Pro AthenaAI % cd athenaos-ai-service
● thuanduc@Thuans-MacBook-Pro athenaos-ai-service % pip install -r requirements.txt
Requirement already satisfied: fastapi in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from -r requirements.txt (line 4)) (0.115.14)
Requirement already satisfied: uvicorn in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from -r requirements.txt (line 5)) (0.35.0)
Requirement already satisfied: requests in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from -r requirements.txt (line 8)) (2.32.3)
Requirement already satisfied: python-dotenv in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from -r requirements.txt (line 9)) (0.1.0)
Requirement already satisfied: grot in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from -r requirements.txt (line 10)) (0.32.0)
Requirement already satisfied: starlette<0.47.0,>=0.40.0 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from fastapi->-r requirements.txt (line 4)) (0.46.2)
Requirement already satisfied: pydantic!=1.8,!=1.8.1,!=2.0.0,!=2.0.1,<3.0.0,>=1.7.4 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from fastapi->-r requirements.txt (line 4)) (2.9.1)
Requirement already satisfied: typing-extensions>=4.8.0 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from fastapi->-r requirements.txt (line 4)) (4.14.1)
Requirement already satisfied: annotated-types>=0.6.0 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from pydantic!=1.8,!=1.8.1,!=2.0.0,!=2.0.1,<3.0.0,>=1.7.4->fastapi->-r requirements.txt (line 4)) (0.7.0)
Requirement already satisfied: pydantic-core==2.23.3 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from pydantic!=1.8,!=1.8.1,!=2.0.0,!=2.0.1,<3.0.0,>=1.7.4->fastapi->-r requirements.txt (line 4)) (2.23.3)
Requirement already satisfied: aiohttp<5.0.0,>=0.40.0->fastapi->-r requirements.txt (line 4) (4.9.0)
Requirement already satisfied: idna>=2.8 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from anyio<5,>=3.6.2->starlette<0.47.0,>=0.40.0->fastapi->-r requirements.txt (line 4)) (3.10)
Requirement already satisfied: sniffio<1.1 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from anyio<5,>=3.6.2->starlette<0.47.0,>=0.40.0->fastapi->-r requirements.txt (line 4)) (1.3.1)
Requirement already satisfied: click>=7.0 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from uvicorn->-r requirements.txt (line 5)) (8.1.7)
Requirement already satisfied: h11>=0.8 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from uvicorn->-r requirements.txt (line 5)) (0.14.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from requests->-r requirements.txt (line 8)) (3.4.1)
Requirement already satisfied: urllib3<3,>=1.21.1 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from requests->-r requirements.txt (line 8)) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from requests->-r requirements.txt (line 8)) (2025.1.31)
Requirement already satisfied: distro<2,>=1.7.0 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from grotq->-r requirements.txt (line 10)) (1.9.0)
Requirement already satisfied: httpx<1,>=0.23.0 in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from grotq->-r requirements.txt (line 10)) (0.28.1)
Requirement already satisfied: httpcore=>1.* in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (from httpx<1,>=0.23.0->grotq->-r requirements.txt (line 10)) (1.0.7)
○ thuanduc@Thuans-MacBook-Pro athenaos-ai-service % uvicorn main:app --reload
INFO:     Will watch for changes in these directories: ['~/Users/thuanduc/AthenaAI/athenaos-ai-service']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [6050] using WatchFiles
API-based configuration file loaded successfully.
INFO:main:Grot client configured for model: llama-3.1-8b-instant
INFO:     Started server process [60502]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
```

- **Step 6: Verification**

- Open a web browser and navigate to the frontend's address (e.g., <http://localhost:5173>) to confirm that everything is working.

≡

AthenaAI

Logout

