# 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.

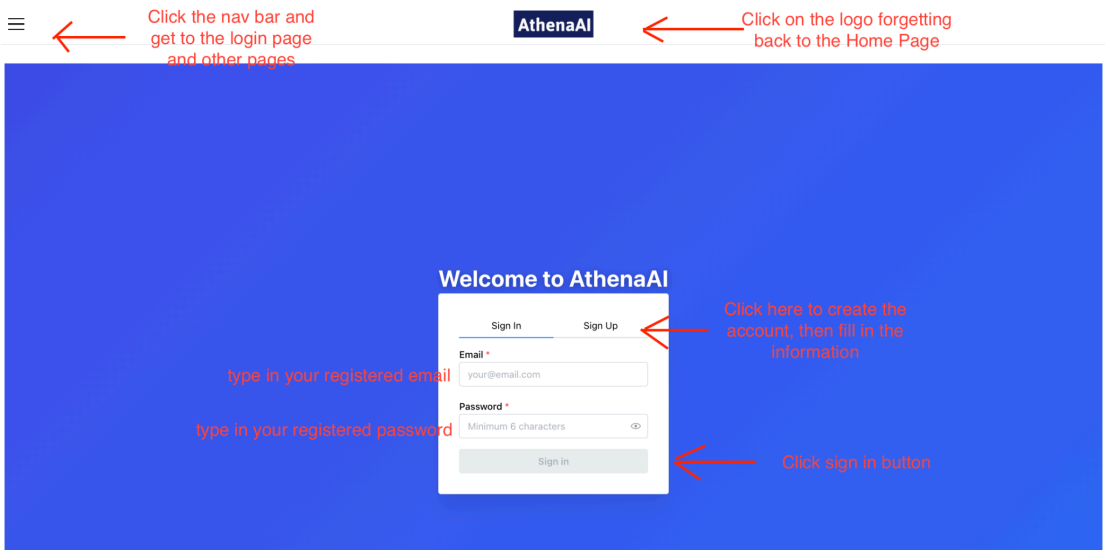
---

## Part 1: User Manual

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

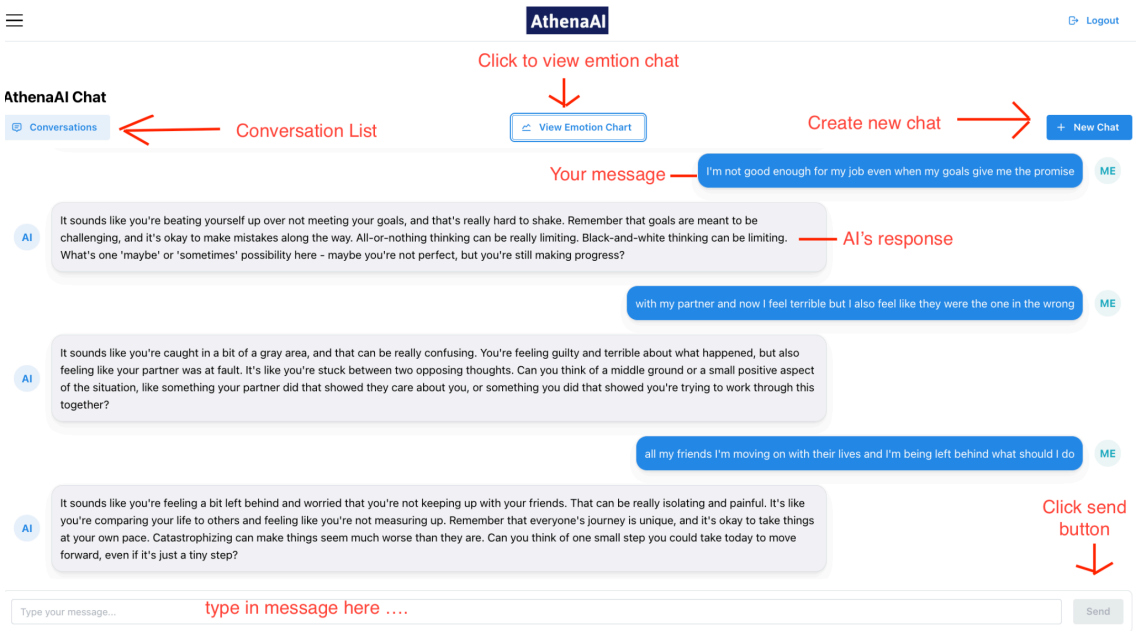
### 1.1. Getting Started

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

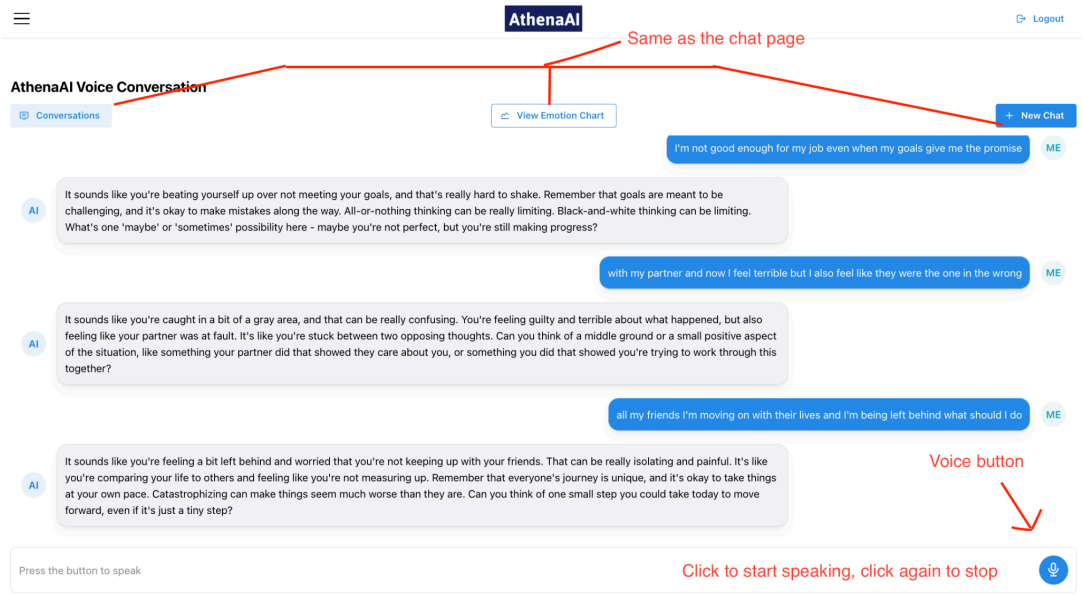


### 1.2. Main Features and Pages

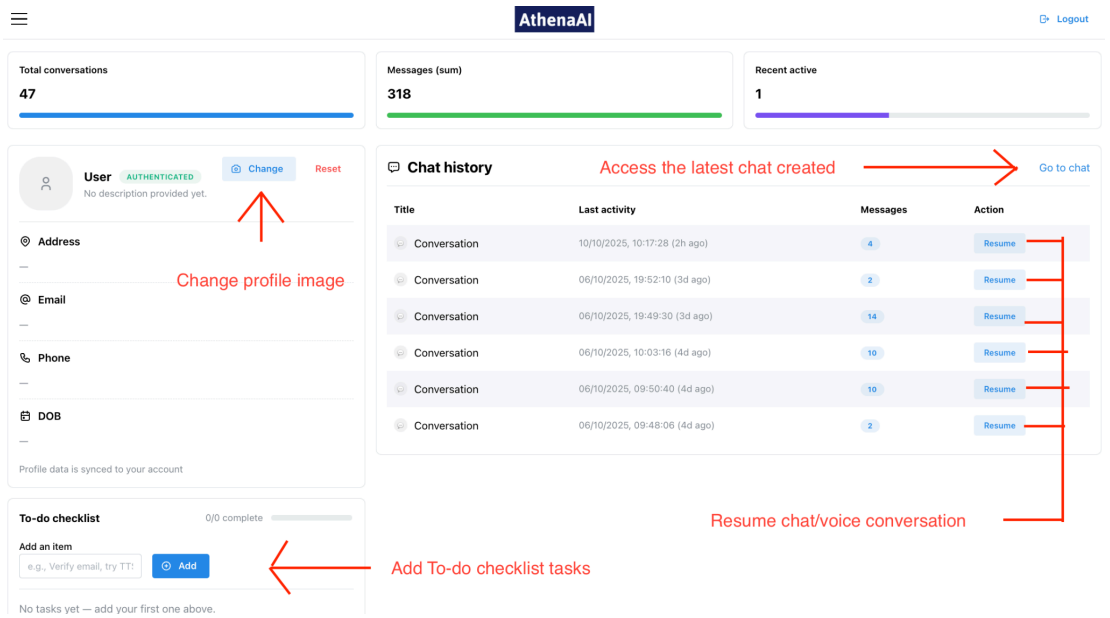
- **Informational Pages:**
  - A brief overview of the site's static pages: Home, Blog, Service, and Contact, which provide product and project information.
- **Chat Page:**
  - The primary feature for interacting with the virtual psychologist via text.
  - How to send a message and receive a CBT-based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



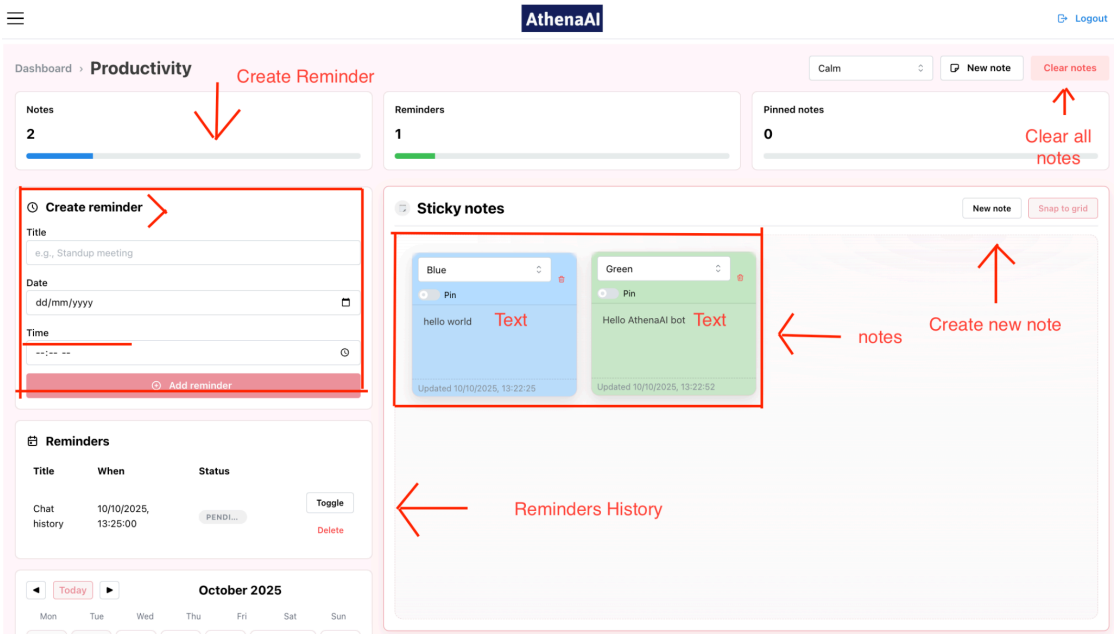

- **Voice Page:**
  - The main feature for voice-based interaction.
  - How to press the button to speak and receive an empathetic, spoken response from the bot.
  - Includes the "View Emotion Chart" feature for conversation analysis.



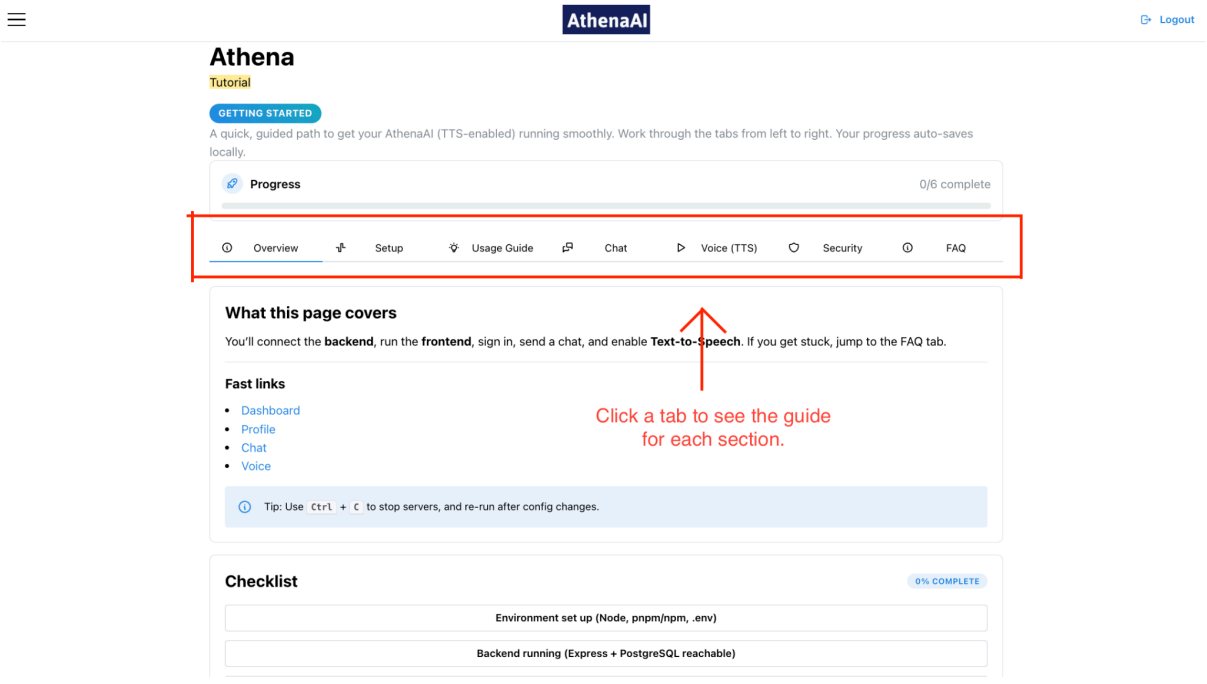

- **Profile Page:**
  - How to view your account information.
  - How to access and review your entire conversation history.



- **Productivity Page:**
    - A dedicated space for users to write and manage notes and comments.




- **Tutorial Page (base on your request):**
    - An in-depth manual embedded within the application, providing detailed instructions for all features.



---

## 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)**

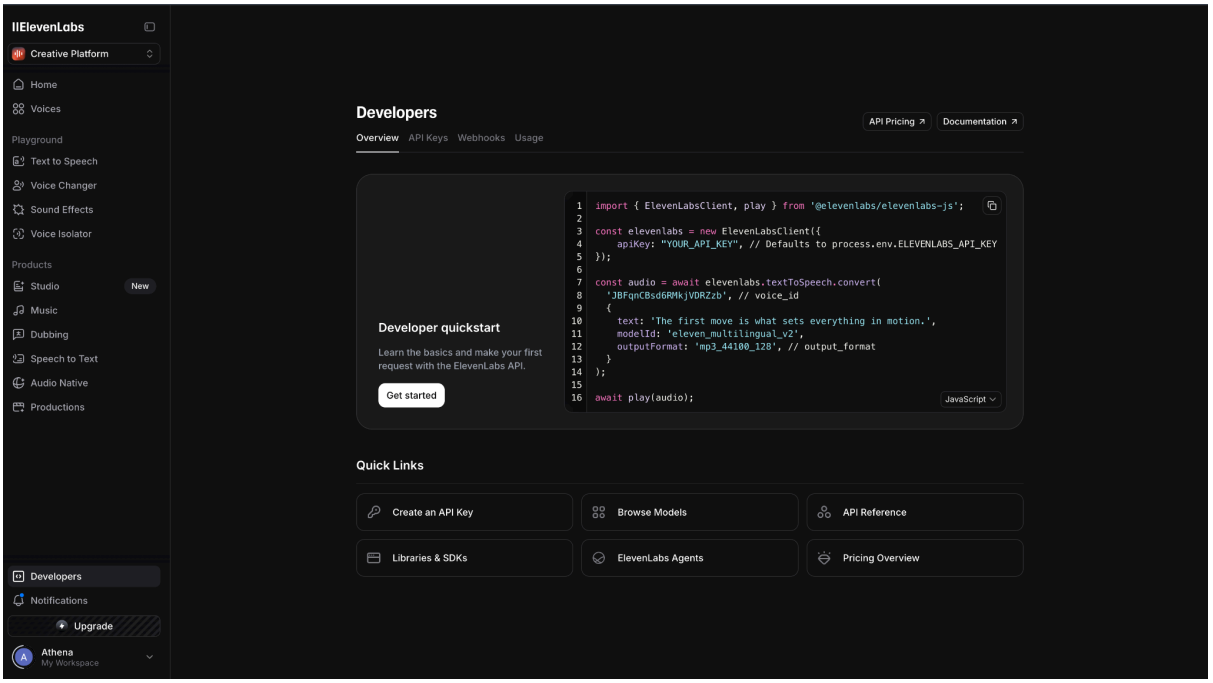
  You can choose one of the following two methods to get the project source code:

  - **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.

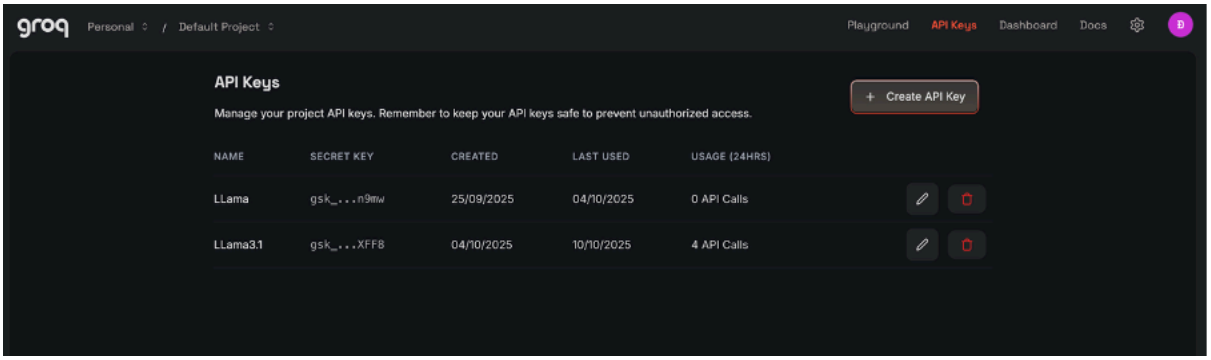
- **ElevenLabs API Key (for Text-to-Speech):** This service requires an API key from ElevenLabs for voice and Text-to-Speech functionality.
    1. Go to the ElevenLabs website (https://elevenlabs.io/) and Sign up or Log in.
    2. In the bottom-left corner of the dashboard, click on the **Developers** 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`




  - **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`):**

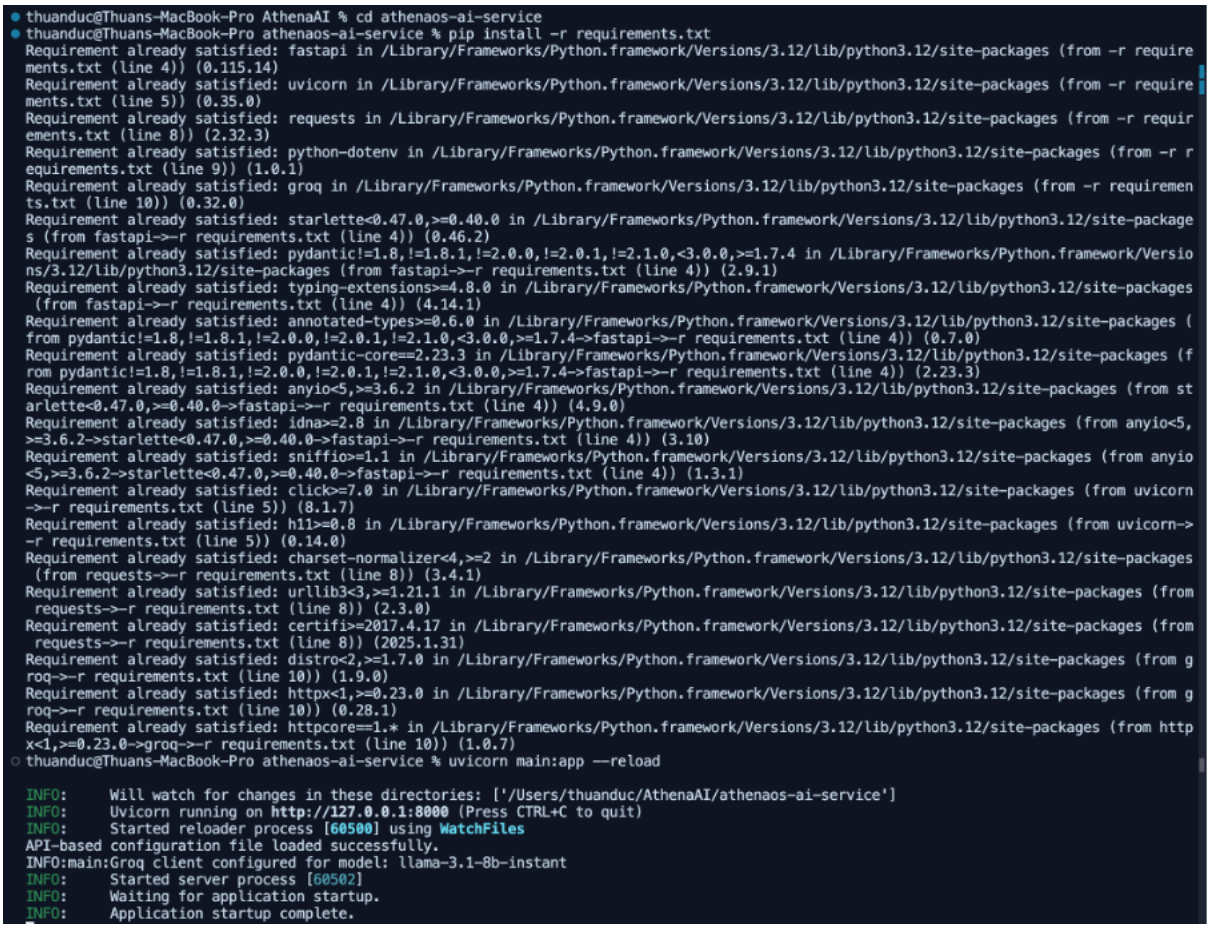
        - In the .env file, ensure the VITE_API_BASE_URL variable is set to the correct address of your running backend service (e.g., http://localhost:8888).
- **Step 3: Run the Backend Service**
    - Open terminal 1, navigate to athenaos-backend, install dependencies (npm install), and run the start command (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).

```
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`).




- **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