Documentation

Nexacura

Table of Contents

[Overview of Nexacura 3](#_Toc163389128)

[Installation and Setup Guide for Nexacura 4](#_Toc163389129)

[Getting Started 4](#_Toc163389130)

[Step 1: Download the Project Files 4](#_Toc163389131)

[Backend Setup 4](#_Toc163389132)

[Frontend Setup 5](#_Toc163389133)

[Backend Setup Troubleshooting 6](#_Toc163389134)

[Problem Description: 6](#_Toc163389135)

[Solution: 6](#_Toc163389136)

[Features and Functionality of Nexacura 7](#_Toc163389137)

[User Login 7](#_Toc163389138)

[Description: 7](#_Toc163389139)

[How to Use: 7](#_Toc163389140)

[Connect to AI Avatar 7](#_Toc163389141)

[Description 7](#_Toc163389142)

[How to Use: 7](#_Toc163389143)

[Voice Interaction with AI Avatar 7](#_Toc163389144)

[Description 7](#_Toc163389145)

[How to Use: 7](#_Toc163389146)

[Receiving AI Avatar's Response 7](#_Toc163389147)

[Description: 7](#_Toc163389148)

[How to Use: 7](#_Toc163389149)

[Disconnecting from AI Avatar 7](#_Toc163389150)

[Description: 7](#_Toc163389151)

[How to Use: 7](#_Toc163389152)

[Backend Code Functionality in Nexacura 8](#_Toc163389153)

[AudioConverter Class 8](#_Toc163389154)

[Purpose: 8](#_Toc163389155)

[Functionality: 8](#_Toc163389156)

[Error Handling: 8](#_Toc163389157)

[Usage: 8](#_Toc163389158)

[ConversationHistoryManager Class 9](#_Toc163389159)

[Purpose: 9](#_Toc163389160)

[Functionality: 9](#_Toc163389161)

[Usage: 9](#_Toc163389162)

[Example: 9](#_Toc163389163)

[OpenAiPsychologist Class 10](#_Toc163389164)

[Purpose: 10](#_Toc163389165)

[Functionality: 10](#_Toc163389166)

[Usage: 10](#_Toc163389167)

[WhisperTranscriber Class 11](#_Toc163389168)

[Purpose: 11](#_Toc163389169)

[Functionality: 11](#_Toc163389170)

[Conclusion 12](#_Toc163389171)

[Team members 12](#_Toc163389172)

# Overview of Nexacura

Nexacura is a prototype website that integrates OpenAI's GPT-4 and Whisper APIs with Microsoft Azure's 3D Real-Time Avatar API to provide a platform where users can interact with an AI avatar in real-time. The website aims to explore the possibilities of combining advanced conversational AI with visual representation, making it possible for users to engage in meaningful conversations with a digital persona. It's designed for users interested in AI technology, looking for interactive experiences or needing assistance through AI-driven conversations.

A screenshot of a website

Description automatically generated

A person in a black shirt

Description automatically generated

# Installation and Setup Guide for Nexacura

## Getting Started

Ensure you have Node.js and npm installed on your system to proceed with setting up Nexacura. This guide will walk you through configuring both the backend and frontend parts of the project.

### Step 1: Download the Project Files

Clone or download the Nexacura project files from the GitHub repository to your local machine.

### Backend Setup

Navigate to the nexacura-backend folder in your terminal.

Install dependencies by running:

A black and grey rectangular object

Description automatically generated

Create a .env file in the nexacura-backend folder and add the following lines, replacing the placeholders with your actual data:

A screen shot of a computer

Description automatically generated

### Frontend Setup

Navigate to the nexacura-frontend folder.

Install dependencies by executing:

A black and grey rectangular object

Description automatically generated

Configure the avatar application by locating a file named avatarAppConfig.js in your project. Update this file with the specific configuration for your application:

A computer screen with text on it

Description automatically generated

# Backend Setup Troubleshooting

After completing the initial setup of the Nexacura backend, if you encounter any issues, particularly related to bcrypt, follow these steps to resolve them:

Issue: Backend Crashes or Fails to Start

## Problem Description:

Sometimes, the backend may fail to start or crash after setup, often due to issues with native dependencies like bcrypt.

## Solution:

If you suspect bcrypt is causing the startup issues, a simple uninstall and reinstall might fix the problem. This process will rebuild bcrypt for your current environment, which can resolve compatibility issues.

Uninstall bcrypt: Navigate to the nexacura-backend folder in your terminal and run:

A black and grey rectangular object

Description automatically generated

Reinstall bcrypt: After successfully uninstalling, reinstall bcrypt by running:

A black and grey rectangular object

Description automatically generated

Reattempt starting the backend server after reinstalling bcrypt. This should resolve any issues caused by bcrypt dependency.

# Features and Functionality of Nexacura

## User Login

Description: Secure user authentication to access personalized features and interact with the AI avatar.

How to Use: Enter your credentials on the login page and click the 'Login' button to access your account.

## Connect to AI Avatar

Description: Establish a real-time connection with the Azure server-hosted AI avatar for interactive communication.

How to Use: After logging in, click the 'Connect' button to initiate the connection process with the AI avatar.

## Voice Interaction with AI Avatar

Description: Speak to the AI avatar using real-time voice recognition powered by OpenAI's Whisper API, which transcribes audio to text for the AI to process.

How to Use: Click 'Start Recording' to capture your voice, which will be transcribed and forwarded to the AI for a response.

## Receiving AI Avatar's Response

Description: Hear the AI avatar's spoken responses after processing your input through OpenAI's GPT-4 AI.

How to Use: To listen to the avatar's response after it has processed your query, click the 'Speak' button. The avatar will start talking, providing you with the requested information or continuing the conversation.

## Disconnecting from AI Avatar

Description: Safely end your interactive session with the AI avatar.

How to Use: Click the 'Disconnect' button to terminate the connection with the AI avatar when you have finished the interaction.

# Backend Code Functionality in Nexacura

## AudioConverter Class

### Purpose:

Converts WebM audio files to MP3, ensuring compatibility with the Whisper API.

### Functionality:

Uses fluent-ffmpeg to handle conversion.

Provides convertWebMToMP3 method to perform the conversion and handle it asynchronously.

### Error Handling:

Logs conversion errors and success messages to the console.

### Usage:

Invoke convertWebMToMP3 and manage the result with Promises.

Best Practices:

This class is key for audio processing within the Nexacura backend, allowing voice inputs to be transcribed accurately.

## ConversationHistoryManager Class

### Purpose:

Manages conversation history for a user session by handling the storage, retrieval, and deletion of messages in a text file. This is essential for maintaining context in ongoing conversations with the OpenAI API, which does not retain history between API calls.

### Functionality:

Initializes with a directory path and an optional filename, defaulting to "conversationHistory.txt".

Ensures that the history file exists or creates a new one if it doesn’t.

Appends new messages to the history file, reads the current history, and clears the history when needed.

### Usage:

Instantiate the class with the path where you want to store the history.

Call appendToHistory(message) to add a user’s message or the AI’s response to the file.

Use readConversationHistory() to get the entire conversation when making a new request to the OpenAI API to maintain context.

Use clearHistory() to reset the conversation, which can be useful when starting a new session or if the user requests to delete their data.

### Example:

A screen shot of a computer program

Description automatically generated

The ConversationHistoryManager is a crucial component that ensures Nexacura's conversations are coherent by providing the AI with previous exchanges, allowing for meaningful and contextually aware interactions.

## OpenAiPsychologist Class

### Purpose:

Interfaces with the OpenAI API to generate conversational responses and manages the conversation history for contextual continuity.

### Functionality:

Uses axios for HTTP requests to communicate with OpenAI's API.

The answer method takes user input as userText, generates the appropriate payload, and posts it to the OpenAI API endpoint.

Utilizes DynamicFolderCreator to organize conversation history in a dedicated directory.

Employs ConversationHistoryManager to read and append to the conversation history, ensuring context is maintained in the conversation.

### Usage:

Instantiate OpenAiPsychologist with the user's request object.

Call the answer method with the user's input text. This method handles the creation of dynamic folders, reads existing conversation history, and appends new messages.

The method returns a promise, which resolves to the AI's response.

The OpenAiPsychologist class is central to Nexacura's backend, bridging user input with AI-generated responses and providing a smooth, context-aware conversational experience.

## WhisperTranscriber Class

### Purpose:

Handles the transcription of MP3 audio files into text using OpenAI's Whisper API, which follows after audio files are converted from WebM to MP3 format by the AudioConverter class.

### Functionality:

Sends audio files to the Whisper API endpoint for transcription.

The transcribeAudio method manages the upload of the MP3 file and retrieval of the transcribed text.

Usage:

Instantiate WhisperTranscriber.

Use transcribeAudio with the path to the MP3 file to start transcription.

The method returns a promise with the transcription result from the Whisper API.

This class makes it possible for Nexacura to include audio inputs in user interactions by converting speech to text, thus allowing the AI to generate relevant responses based on the spoken words.

# Conclusion

So, Nexacura took a bit of a twist. Turned out, the costs for using Microsoft Azure’s avatars were way higher than expected. That was a bit of a surprise, honestly, and it kind of threw a wrench into launching the project as originally planned. The whole experience was a bit of an eye-opener on the importance of digging deep into the costs and details of external services before diving headfirst into building something around them.

I was even thinking about adding a payment feature to Nexacura, but with the Azure costs being a bit of a shocker, hitting pause on that didn’t seem like such a bad idea. It’s all good, though. Every project has its ups and downs, and this one’s no different. The cool part is all the stuff I’ve learned along the way, and none of the work is going to waste. I’ve got some ideas on how to pivot and use what we’ve built for something new and equally exciting.

While Nexacura's journey took an unexpected detour because of the Azure costs, it's worth highlighting the tech stack we rallied behind. It was robust, versatile, and totally geared up for the task:

Frontend, React: Kept our user interface sleek and interactive.

Backend, Node.js and Express: These two made sure our server-side logic was on point.

Database, MongoDB: Our go-to for a flexible, no-fuss database solution.

Chatting , OpenAI: GPT-4 for smart conversations and Whisper for turning spoken words into text, these APIs were crucial.

Microsoft Azure: Even though we hit a snag with the cost, Azure's 3D Avatar service was a bold choice for bringing our AI interactions to life visually.

## Team members

Peter Kaszap-Nagy, Balint Kaszap-Nagy