The Ultimate AI Voice Pipeline

Lesson 4: Giving Ava a voice

MIGUEL OTERO PEDRIDO
FEB 26, 2025

If you open **WhatsApp** and scroll through your recent chats, I bet you'll see more just text messages - memes, GIFs, videos … and the star of today's article: **voice notes!** 🗣️

When Jesús Copado and I began the **Ava** project, we knew one thing for sure - Av needed a voice. **Not just any voice, but a unique one**. Don't believe me? Take a listen:

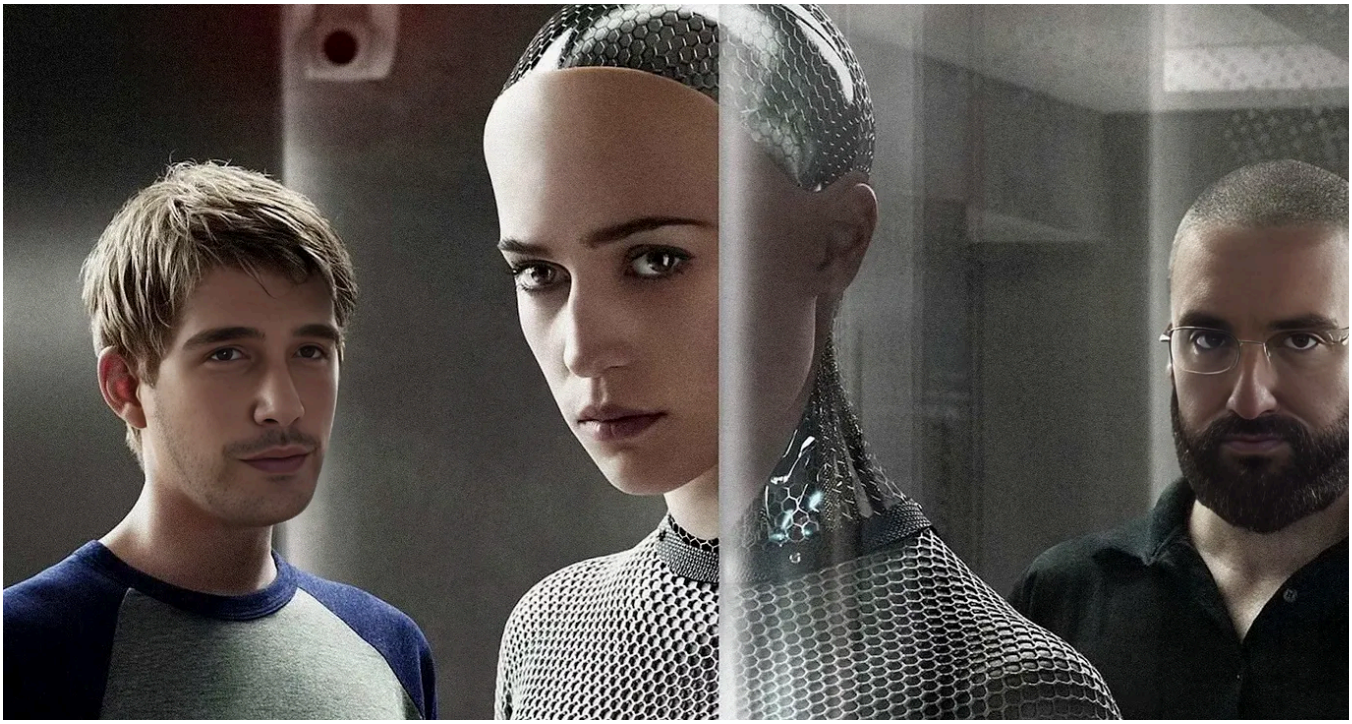
1× 0:00 -0:03

To make that happen, we needed two key systems: **STT (speech-to-text)** to turn incoming voice notes into text and **TTS (text-to-speech)** to convert Ava's replies audio.

That brings us to our focus today: **Ava's Voice Pipeline.**

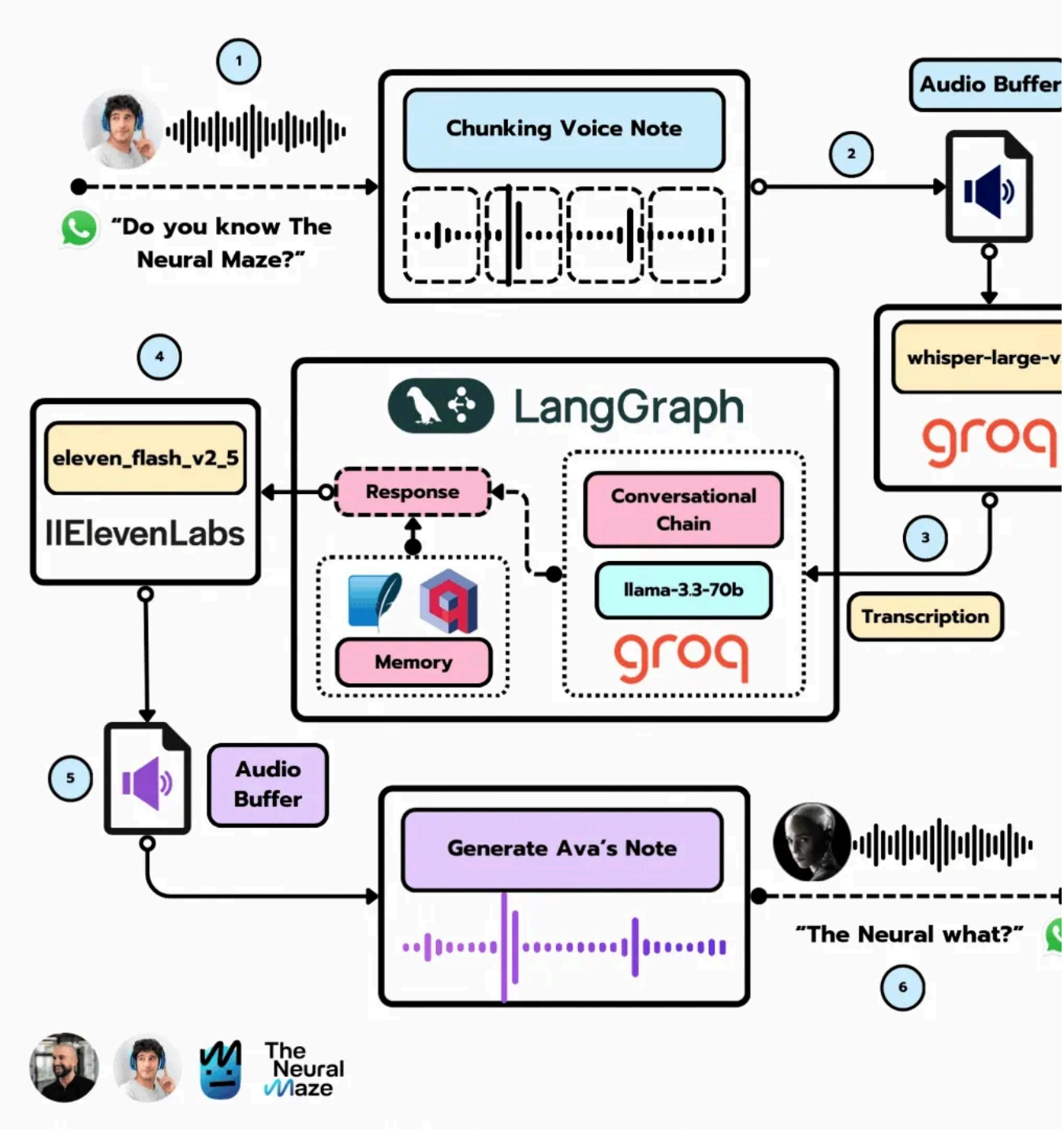
Ready? Let's begin! 👇

> [Check the code here!](#) 👨‍💻



> This is the **fourth lesson** of **“Ava: The Whatsapp Agent”** course. This lesson builds on the theory and code covered in the previous ones, so be sure to chec them out if you haven’t already!
>
> - Lesson One: Project Overview
> - Lesson Two: Dissecting Ava’s brain
> - Lesson Three: Unlocking Ava’s memories

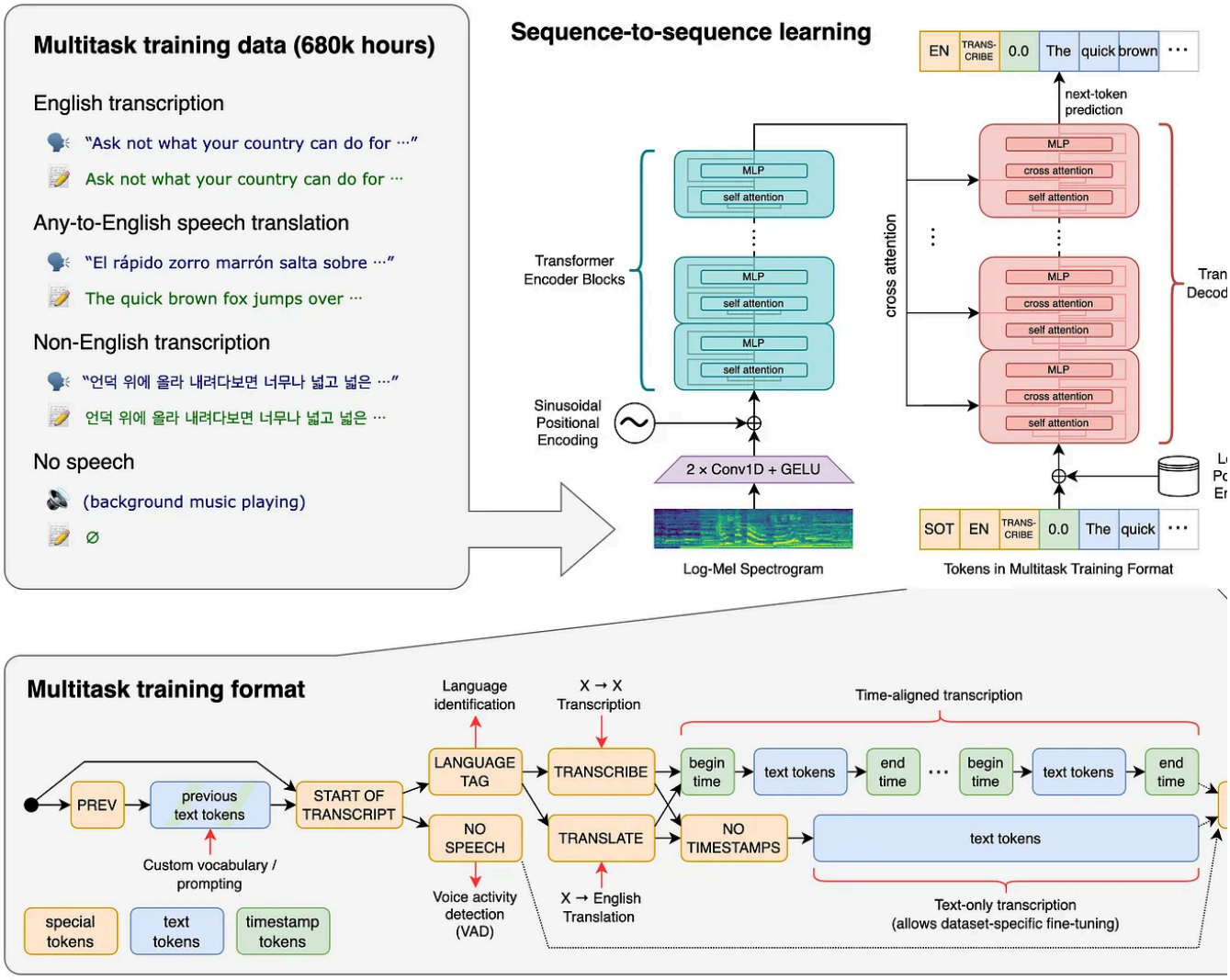
Ava’s Voice Pipeline



Ava's Voice Pipeline Overview

Like we mentioned before, this pipeline has **two main parts** - the **TTS module** an the **STT module** - which can work together or on their own. Let's go over each on the next sections.

STT Module - Whisper 👂



Whisper architecture (source: https://cdn.openai.com/papers/whisper.pdf)

Whisper is a key part of Ava's **STT (speech-to-text) module** because it helps Av accurately transcribe voice messages. If you're not familiar with "Whisper", it's an advanced model from OpenAI that can handle multiple languages, different accen and even background noise - perfect for WhatsApp voice messages! 📱

In the Voice Pipeline diagram, you'll see we're using Groq's Whisper - sorry, we ar hosting Whisper ourselves 😅 - , which you can check out here.

To use this model in our code, we've created a `SpeechToText` class inside Ava's modules. This class handles all the audio transcription logic - you'll find the magic the `transcribe` method in the snippet below! 👀

```python
class SpeechToText:
    """A class to handle speech-to-text conversion using Groq's Whisper model."""

    def __init__(self):
        """Initialize the SpeechToText class"""
        self._client: Optional[Groq] = None

    @property
    def client(self) -> Groq:
        """Get or create Groq client instance using singleton pattern."""
        if self._client is None:
            self._client = Groq(api_key=settings.GROQ_API_KEY)
        return self._client

    async def transcribe(self, audio_data: bytes) -> str:
        """Convert speech to text using Groq's Whisper model.

        Args:
            audio_data: Binary audio data

        Returns:
            str: Transcribed text

        Raises:
            ValueError: If the audio file is empty or invalid
            RuntimeError: If the transcription fails
        """
        if not audio_data:
            raise ValueError("Audio data cannot be empty")

        try:
            # Create a temporary file with .wav extension
            with tempfile.NamedTemporaryFile(suffix=".wav", delete=False) as temp_file
                temp_file.write(audio_data)
                temp_file_path = temp_file.name

                try:
                    # Open the temporary file for the API request
                    with open(temp_file_path, "rb") as audio_file:
                        transcription = self.client.audio.transcriptions.create(
                            file=audio_file,
                            model="whisper-large-v3-turbo",
                            language="en",
                            response_format="text",
                        )

                    if not transcription:
                        raise SpeechToTextError("Transcription result is empty")

                    return transcription

                finally:
                    # Clean up the temporary file
```

```
            os.unlink(temp_file_path)

        except Exception as e:
            raise SpeechToTextError(
                f"Speech-to-text conversion failed: {str(e)}"
            ) from e
```

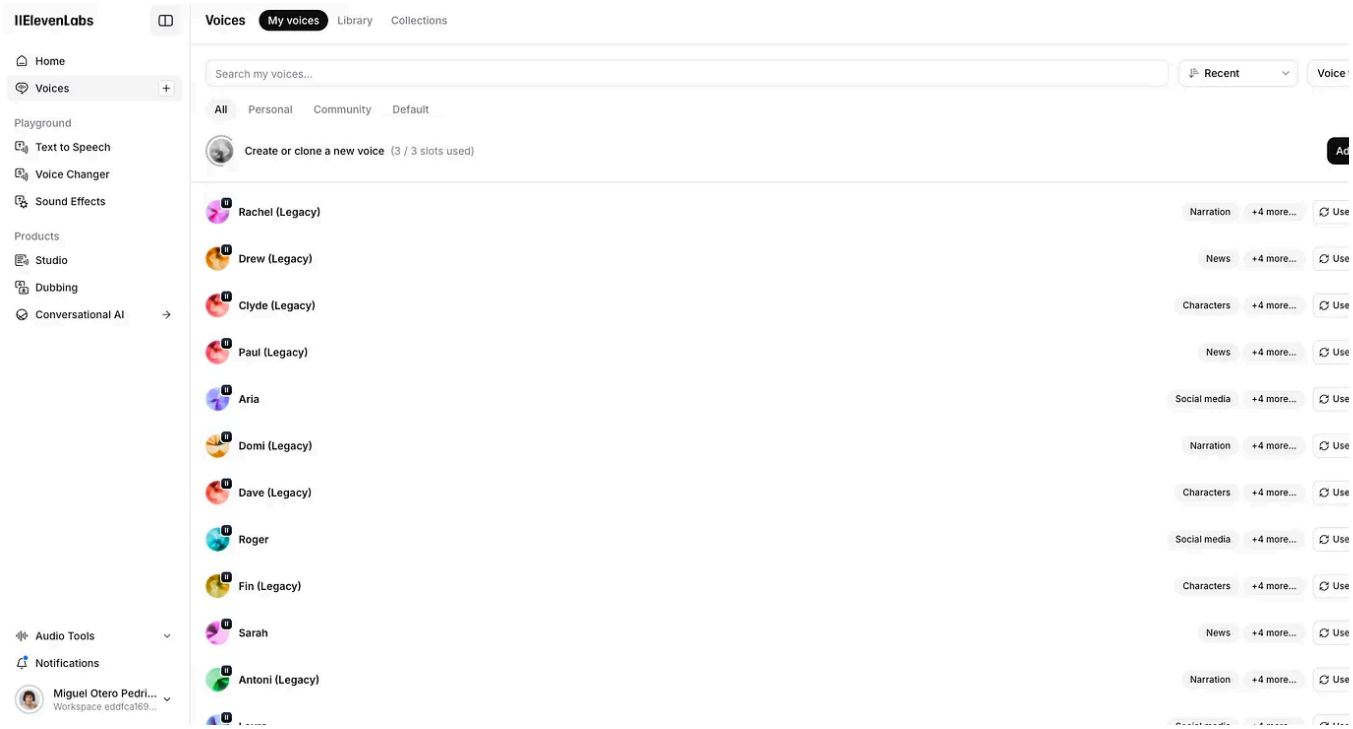
The SpeechToText class takes care of all the transcription logic

TTS Module - ElevenLabs 🗣️

Once the audio messsage is transcribed, it moves to the LangGraph workflow (Av brain, remember? 😉), which takes care of generating a response - using the short/long-term memories, Ava's activities, etc.

But this response is just text! We need a voice, and ... guess who has amazing voi ready to go?

You got it - ElevenLabs is in the house 😎



Creating custom voices in ElevenLabs

Like I mentioned at the start, we didn't want Ava to have just any voice - we wante something unique. That's why we're using **ElevenLabs' custom voice features**.

In the end, all we need is an `ELEVENLABS_VOICE_ID`, which uniquely defines Ava's voice.

So, to add the voice generation logic to the code, we followed the same approach as the STT module: we created a `TextToSpeech` class inside Ava's modules.

Check the `synthesize` method in the snippet below! 👀

```python
class TextToSpeech:
    """A class to handle text-to-speech conversion using ElevenLabs."""

    def __init__(self):
        """Initialize the TextToSpeech class"""
        self._client: Optional[ElevenLabs] = None

    @property
    def client(self) -> ElevenLabs:
        """Get or create ElevenLabs client instance using singleton pattern."""
        if self._client is None:
            self._client = ElevenLabs(api_key=settings.ELEVENLABS_API_KEY)
        return self._client

    async def synthesize(self, text: str) -> bytes:
        """Convert text to speech using ElevenLabs.

        Args:
            text: Text to convert to speech

        Returns:
            bytes: Audio data

        Raises:
            ValueError: If the input text is empty or too long
            TextToSpeechError: If the text-to-speech conversion fails
        """
        if not text.strip():
            raise ValueError("Input text cannot be empty")

        if len(text) > 5000:  # ElevenLabs typical limit
            raise ValueError("Input text exceeds maximum length of 5000 characters"

        try:
            audio_generator = self.client.generate(
                text=text,
                voice=Voice(
                    voice_id=settings.ELEVENLABS_VOICE_ID,
                    settings=VoiceSettings(stability=0.5, similarity_boost=0.5),
                ),
                model=settings.TTS_MODEL_NAME,
            )

            # Convert generator to bytes
            audio_bytes = b"".join(audio_generator)
            if not audio_bytes:
                raise TextToSpeechError("Generated audio is empty")

            return audio_bytes

        except Exception as e:
```

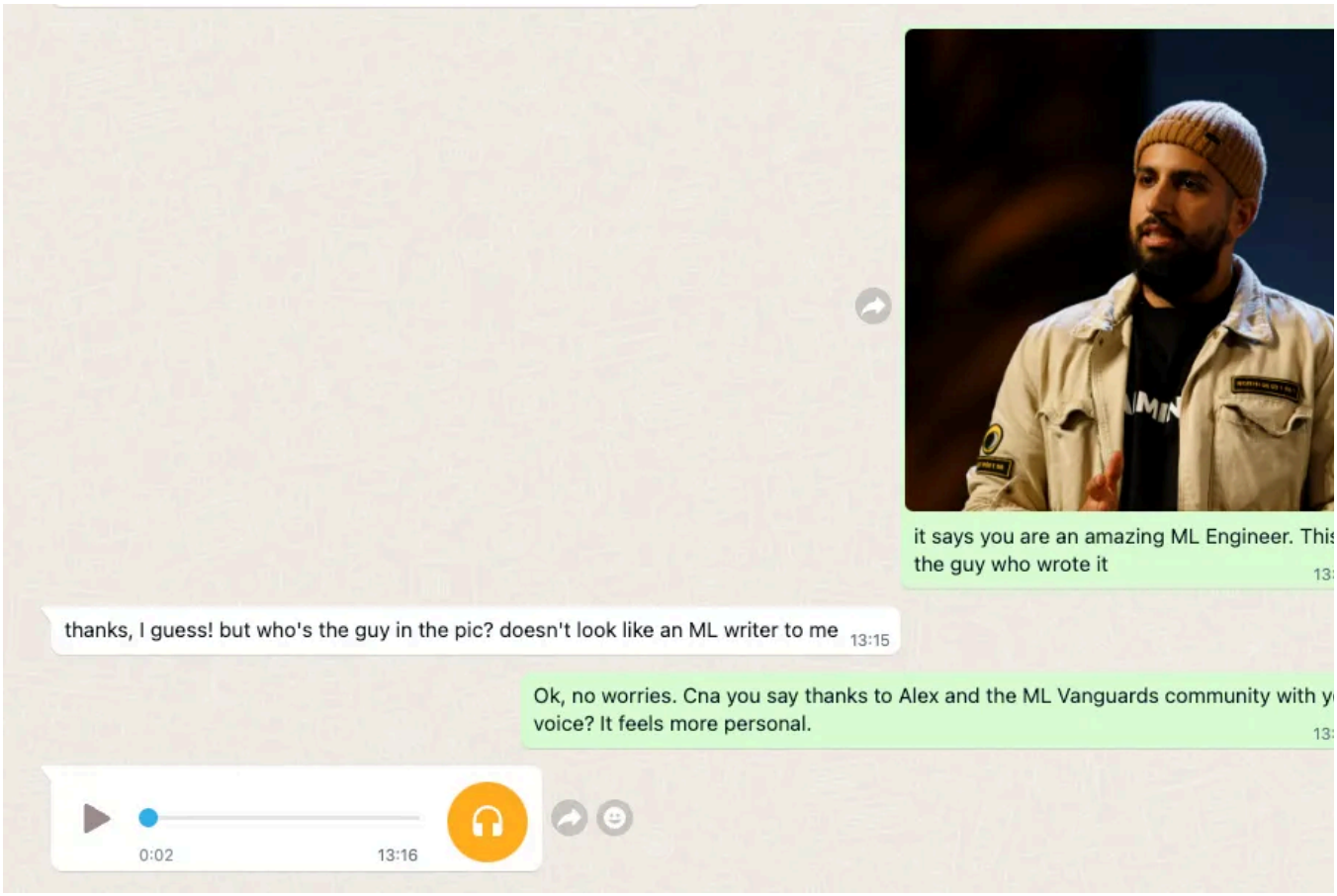
```
            raise TextToSpeechError(
                f"Text-to-speech conversion failed: {str(e)}"
            ) from e
```

This class gets called from the **audio_node**, as shown below, generating an **audio_buffer** that gets stored in [LangGraph's state](#).

```
async def audio_node(state: AICompanionState, config: RunnableConfig):
    current_activity = ScheduleContextGenerator.get_current_activity()
    memory_context = state.get("memory_context", "")

    chain = get_character_response_chain(state.get("summary", ""))
    text_to_speech_module = get_text_to_speech_module()

    response = await chain.ainvoke(
        {
            "messages": state["messages"],
            "current_activity": current_activity,
            "memory_context": memory_context,
        },
        config,
    )
    output_audio = await text_to_speech_module.synthesize(response)

    return {"messages": response, "audio_buffer": output_audio}
```

LangGraph's state will be picked up by the **WhatsApp webhook endpoint** (more that in Lesson 6), turning it into a voice message you'll get from Ava!

> Check out Ava in action - roasting Alexandru Vesa both through text and voice messages! 🤣🤣🤣



Alex getting roasted by Ava 🔥

And that's all for today! 🙌

Just a quick reminder - **Lesson 5** will be available next **Wednesday, March 5th.** F
don't forget there's a **complementary video lesson** on [Jesús Copado's YouTub
channel](#)**.**

> We **strongly recommend** checking out **both resources** (**written** lessons and **video** lessons) to **maximize** your **learning experience**! 🙂

Thanks for reading The Neural Maze! Subscribe for free to receive new posts and support my work.

18 Likes ∙ 5 Restacks

← Previous Next

Discussion about this post

Comments Restacks

Write a comment...

Vishnu 9 Mar *Edited*

I tried running your repo, but I hit the error. sqlite3.OperationalError: unable to open databas

Something to do with creation of app/data folder.

This is after I launch chainlit and type my first message.

LIKE REPLY