🎙️ **Task 11: Transcribe and Summarize Audio**

In this task we’ll build a lightweight tool that transcribes audio, summarizes the content, and extracts meaningful insights—all powered by **OpenAI's Whisper and GPT models**.

**📚 Theory**

**Whisper** is an automatic speech recognition (ASR) model developed by OpenAI. It’s trained on 680,000 hours of multilingual and multitask supervised data collected from the web—making it highly robust in real-world environments.

**🔑 Key Features of Whisper:**

* **Multilingual transcription**: Can transcribe speech in multiple languages.
* **Translation**: Can translate non-English speech into English.
* **Punctuation and casing**: Automatically adds punctuation and capitalization for readability.
* **Robustness**: Handles accents, background noise, and technical jargon better than many traditional ASR systems.
* **No need for fine-tuning**: Works well out-of-the-box for most tasks.

Under the hood, Whisper uses an **encoder-decoder Transformer architecture**, the same kind that powers large language models like GPT. It treats audio transcription as a **sequence-to-sequence language generation** problem.

For developers, this means you can upload an audio file and get a ready-to-use transcript in seconds—perfect for rapid prototyping of audio-based AI tools.

**🧠**[**Whisper Speech-to-Text**](https://platform.openai.com/docs/guides/speech-to-text)

AI models like Whisper and GPT enable developers to turn speech into **readable, structured, and insightful outputs** without the need for traditional NLP pipelines. In this task, you'll combine speech-to-text, text summarization, and lightweight analytics using OpenAI APIs.

* **Whisper API** allows you to transcribe audio with high accuracy, automatically adding punctuation, formatting, and even detecting the language.
* **GPT** can then take this transcription and:
  + Generate a summary capturing key ideas
  + Extract custom analytics, like word count, speaking speed, or recurring topics

Together, these tools streamline how we derive **structured, actionable insights** from raw audio recordings—useful in interviews, podcasts, meetings, and more.

**🧩 AI Technique: This challenge combines several AI techniques**

**1. 🎧 Speech-to-Text Integration (ASR)**

You'll learn to:

* Upload audio to an AI model
* Parse structured responses from the Whisper API
* Handle edge cases like unclear speech or non-English content

*Real-world use case: Meeting minutes generation, podcast transcription, or compliance recordings.*

**2. ✍️ Text Summarization**

Using a language model like GPT-4, you’ll:

* Convert long transcripts into concise summaries
* Focus on preserving core intent and main takeaways
* Write effective prompts that guide the summarizer

*Real-world use case: Executive summaries from call transcripts, TL;DRs for recorded lectures, or summarizing interview content.*

**3. 📊 Analytical Prompt Engineering**

You’ll guide the model to:

* Calculate custom metrics (e.g., word count, speaking speed)
* Extract structured entities (like frequently mentioned topics)
* Format results as valid JSON objects

This reinforces your understanding of **structured prompting**, where the AI acts more like a data processor than a writer.

*Real-world use case: Speech analytics platforms, call center insight dashboards, or content strategy tools.*

Combining **multimodal inputs** (audio) and **multi-step prompting workflows** (summarize + analyze) is a key AI engineering skill. This task helps you practice:

* When to call which model
* How to split tasks across APIs
* How to structure inputs and validate outputs

**🛠️ Task**

Your task is to create a **console application** that:

* Accepts a spoken audio file
* Transcribes it using OpenAI's Whisper API
* Summarizes the transcription using GPT model
* Extracts custom statistics from the transcript, such as:
  + Total word count
  + Speaking speed (in words per minute)
  + Frequently mentioned topics and how many times each is mentioned
* Saves transcription result in a separate file (each new transcription should create a new separate file)
* Returns **summary and analytics** to the user in console

**💬 Example of analytics:**

{    
  
"word\_count": 1280,    
"speaking\_speed\_wpm": 132,    
"frequently\_mentioned\_topics": [      
{ "topic": "Customer Onboarding", "mentions": 6 },      
{ "topic": "Q4 Roadmap", "mentions": 4 },      
{ "topic": "AI Integration", "mentions": 3 }

]

}

🎧 You are given an [**AUDIO FILE**](https://ventionteamsinc-my.sharepoint.com/:u:/g/personal/mikita_sauko_ventionteams_com/EQ-2UN6KK7VFt-naH8IsjiMBRWI-XuRbIzYh5lwO7gwM2A?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJPbmVEcml2ZUZvckJ1c2luZXNzIiwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IldlYiIsInJlZmVycmFsTW9kZSI6InZpZXciLCJyZWZlcnJhbFZpZXciOiJNeUZpbGVzTGlua0NvcHkifX0&e=fYK0bc) to work with.

**📌 Requirements**

* The app must include calls to **OpenAI API**
* The app must use **OpenAI whisper-1 model**
* The app must be able to accept **any audio file**, not only the provided sample
* README.md must contain **clear and detailed instructions** on how to run the application
* Analytics must include:
  + Word count
  + Speaking speed (WPM)
  + Top 3+ frequently mentioned topics
* Output must be clear, properly formatted, and align with all the requirements stated in task description