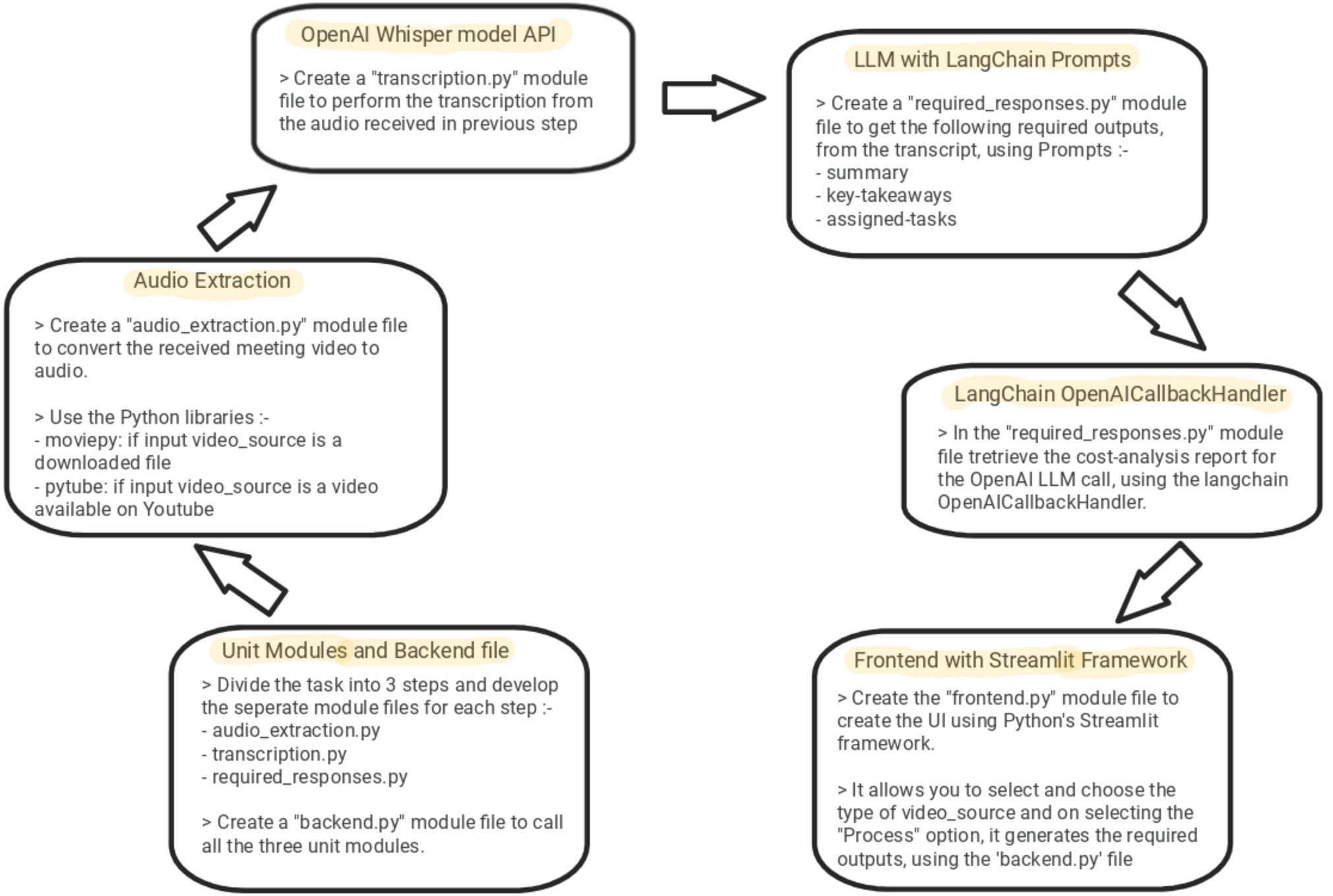
Unit Modules and Backend file
> Divide the task into 3 steps and develop the seperate module files for each step :-
- audio_extraction.py
- transcription.py
- required_responses.py
> Create a "backend.py" module file to call all the three unit modules.
Audio Extraction
> Create a "audio_extraction.py" module file to convert the received meeting video to audio.
> Use the Python libraries :-
- moviepy: if input video_source is a downloaded file
- pytube: if input video_source is a video available on Youtube
OpenAI Whisper model API
> Create a "transcription.py" module file to perform the transcription from the audio received in previous step
LLM with LangChain Prompts
> Create a "required_responses.py" module file to get the following required outputs, from the transcript, using Prompts :-
- summary
- key-takeaways
- assigned-tasks
LangChain OpenAICallbackHandler
> In the "required_responses.py" module file tretrieve the cost-analysis report for the OpenAI LLM call, using the langchain OpenAICallbackHandler.
Frontend with Streamlit Framework
> Create the "frontend.py" module file to create the UI using Python's Streamlit framework.
> It allows you to select and choose the type of video_source and on selecting the "Process" option, it generates the required outputs, using the 'backend.py' file