EE250 Final Project: ShazamPI

Team Members: Jeremy Pogue, Jaya Travis

ShazamPi Final Project:

At a high level, our IoT system uses the Shazam API to recognize any song being played, then display that song on the Raspberry Pi's LCD. We had a few primary goals that presented obstacles in implementation. Firstly, we wanted to avoid manually recording the audio and saving the file in the correct directory. We were able to automate this process using the PyAudio Python library. This library allows us to automatically take audio samples using a MacBook during runtime, combine them into a stream, and save the result to a file. We also did not want the user to provide the audio file to be analyzed manually. Therefore, we feed the same output file obtained from PyAudio into the Shazam API directly.

The project implementation accomplishes the above goals, as demonstrated in the demo video. It does so via communication between two primary nodes: a laptop and a Raspberry Pi. On startup (i.e., once the server is running on the Raspberry Pi and the client code has begun on the computer), the user is asked to press ENTER to start sampling audio. Then, using PyAudio, audio samples are taken from the computer microphone at specific intervals for 10 seconds, and the resulting stream is saved to a file. To avoid cluttering the main folder, this file is stored as a .wav (audio) file in the "data" folder. Then, the Shazam API takes this audio file as input and returns a JSON containing information about the audio's closest song match (title, lyrics, music video link, etc.). We then send this JSON to the Raspberry Pi server using HTTP protocols and the Flask app. The server parses this JSON for the title and artist of the song and displays this information on the LCD using the GrovePi shield. Below is the block diagram of the ShazamPi:

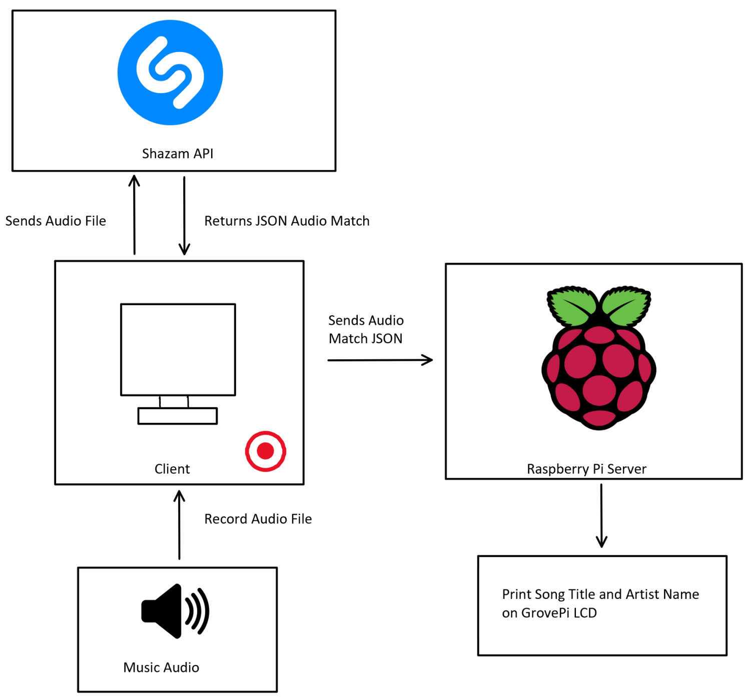

Figure 1: ShazamPi Block Diagram

We used many libraries to implement our project, which presented many challenges. Frequently, libraries would have prerequisites that were not immediately obvious (for example, the PyAudio library required the portaudio library, which could not be installed with pip3). Additionally, pulling from the Shazam API presented many challenges, particularly in getting a certificate to make API requests. However, upon getting the libraries/API to function, they were relatively intuitive. The use of the GrovePi also made the embedded design significantly easier.

Regarding future steps, it would be exciting for the song to be played by a speaker on the Raspberry Pi. However, that poses significant challenges with synching the audio with the music being played (it would require some kind of microphone) and playing multiple frequencies with a cheap buzzer.