Moody IoT

A personalized mood feedback voice-app.

Carolina Hernandez Mateo
Department of Computer Science
Binghamton University Binghamton, NY
cherna19@binghamton.edu

ABSTRACT

The motivation for this project was that I couldn't find a recommendation system based on response sentiment analysis. A lot of individuals that consume voice-based apps experiences look for recommenders' systems, but there are not a lot of systems that consider the user response based on how they are feeling at the moment. So that's the reason why I decided to implement this project.

KEYWORDS
Voice-based, Sentiment Analysis, Mood

I. INTRODUCTION

From a human perspective and given the current social distancing situation, more apps are being developed to target different aspects that has been disrupted by COVID-19 virus like entertainments, theaters, concerts, leisure activities and others. This is one of the reasons why developing a personalized mood feedback system for self-mental stability and recommendation of content is a good tool to help in a minor scale with the emotional situation that carries a crisis. For this reason, Moody IoT application was implemented. In the following sections, I will explain the design, tools, implementation of the system.

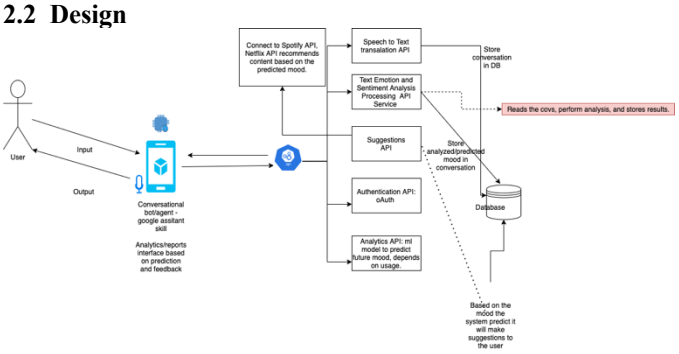
II. SOFTWARE DESIGN AND IMPLEMENTATION

2.1 Hardware component

Voice input

In this project I used an Iphone SE as my hardware component. The IoT component support is the phone microphone as input and the audio output for the conversational application.

2.2 Design



In this design the application flow is as follows:

1. The user talks to "Moody IoT" app.
2. The App connects to the cloud application service in Google cloud to translate the user Query.
3. The application detects the audio intents from the users.
4. It analyzes the user Query Sentiment Score.
5. Returns a recommendation list of Movies or Music. Based on user final decision from the Specified APIs.

2.2 Implementation

2.2.1 Software

Google dialogflow Sdk

This library allows to use language understanding and makes it easier to design and implement a conversational user interface as an app.[1]

Google firebase

This can integrate with dialogflow to save specifics user queries and to enable the webhooks in the app which is the extended logic of the basic conversational experience.[1]

Natural Language API

Analyze in the query in terms of positive, neutral, negative.[2]

Node.js

With its library was possible to implement webhooks and fulfillment for the app.

VsCode

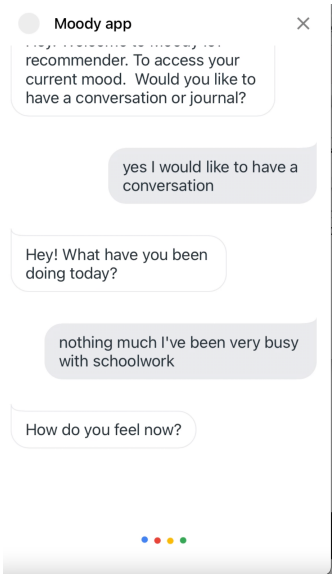
Used as editor in the components that were not edited in google cloud.

2.2.2 Implementation

Photo from VsCode implementation

The application is deployed in google cloud. How this Implementation works from a developer perspective.

The google assistant application is developed with the Action SdK and Dialogflow Sdk. First, i had to develop the Intent of the application from the start of the flow to the end of the conversation flow. The agent would interact With the user response which is the *audioDetectIntect***.** Once the intent is detected the programmed response based On user mood analysis will trigger the next conversational experience. Once the agent has collected the information through the voice-app it will trigger the webhook and reply one movie or song recommendation.



III. Conclusion & Evaluation

The learnings about different approached to implement are very relevant. The need to create better conversational experiences is increasing and understanding how to develop tools that enable to understand user responses better is the goal of a rich conversational application. Applying techniques as Natural language and Cloud computing has giving me an edge on how modern technologies for this type of app works.

References

[1] Google Cloud API's. https://cloud.google.com/apis/docs/overview

[2] https://medium.com/google-cloud/sentiment-analysis-using-google-cloud-machine-learning-552be9b9c39b