Senior Design Project

Moodio

High Level Design Report

Ahmet Akif Uğurtan, Mehmet Taha Çetin, Ulaş İş, Esra Nur Ayaz, Ahmet Ensar

Supervisor: Cevdet Aykanat

Jury Members: H. Altay Güvenir, Hamdi Dibeklioğlu

Innovation Expert: Duygu Gözde Tümer

Dec 31, 2018

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

1. Introduction

1.1 Purpose of the System

Moodio is an mobile application that recommends a music playlist to users according to their overall music taste and current mood. Facial expressions and sound tone will be used to determine the current mood. Users can easily get a recommendation music list by taking a photo or speaking to the mobile phone after connecting their Spotify account to Moodio. A user's music taste will be determined by his/her Spotify music history. Additionally, users will be able to search which mood other users listen most. Thanks to Moodio, a user's music experience will increase.

1.2 Design Goals

1.2.1 Design Trade-offs

1.2.1.1 Memory vs Performance

An electronic device's memory is faster than its disk. Thus, keeping the data of images and songs on memory increases the performance of an application when it wants to reach them but it increases the memory usage too. Moodio will keep the images and songs that user will listen in the memory instead of disk in order to increase performance trading with memory usage. Nowadays mobile phones have a lot memory so more memory usage will not be a problem.

1.2.1.2 Usability vs Functionality

The more functions an application has the more it gets complicated to use. Moodio has only two main functions which are recommending a music list according

to user's mood and music taste and searching which mood other users listened most. Since Moodio will have a few functions, it will focus on usability and it will be easy-to-use.

1.2.1.3 Object-Oriented Programming vs Development Time

Writing code with procedural programming takes less time than object-oriented programming but procedural programming is not easy to read or modify. Thus, Moodio will be coded with Object-Oriented programming by spending more time on the beginning.

1.2.1.4 Quality Assurance vs Time

Quality assurance process takes more time if its criteria are a lot. During the implementation of Moodio, we will spend more time to assure these criteria.

1.2.1.5 Security vs Performance

Moodio will use encryption methods in order to ensure users' private information security. However, applying these methods increase the number of computations the device should do and they decrease the performance of Moodio.

1.2.1.6 AJAX vs Compatibility Issue

Using AJAX for updating data on a application increases the user experience since it only updates necessary components, not the whole screen. However, AJAX sometimes can create compatibility issues in different environments. We will overcome these compatibility issues and use AJAX for Moodio.

1.2.1.7 Data Reliability vs Maintenance Time

Ensuring data reliability costs more maintenance time but data reliability is a must in Moodio. We will ensure data reliability trading with maintenance time.

1.2.2 Design Criteria

1.2.2.1 Memory

Moodio should be used on most of the phones. Nowadays, most of the phones can provide at least 1 GB memory for external applications.Thus, Moodio will not use more than 1 GB memory.

1.2.1.2 Performance

Moodio performance must be enough to ensure best user experience and to satisfy average response time. Server response time and the algorithm for creation of music list must not take so much time.

1.2.1.3 Availability and Dependability

Moodio will depend on having a Spotify Account for user information and Spotify API for reaching user information and recommendation. It will be available on Android and Mac operating systems by using React Native. Everybody who has an android or mac phone will be able to use Moodio.

1.2.1.4 Reliability

Moodio will be developed such that bug occurrence will be as little as possible. Also, data on the database will not be corrupted.

1.2.1.5 Security

Encryption methods will be used to ensure the protection of a user’s private information.

1.2.1.6 Readability

The development part of Moodio such as codes will be readable and easy to understand thanks to Object-oriented programming.

1.2.1.7 Modifiability

Functions of Moodio during the development process will be easy to modify thanks to Object-oriented programming.

1.2.1.8 Usability

Moodio will have a simple yet elegant user interface in order to increase user experience and to satisfy usability.

1.3 Definitions, acronyms, abbreviations

1.3.1 Spotify

Spotify is a music streaming platform developed by Swedish company Spotify AB and it is a freemium service; basic features are free with advertisements or automatic music videos, while additional features, such as improved streaming quality are offered via paid subscriptions. [1]

1.3.2 Ajax

Ajax (short for "Asynchronous JavaScript And XML") is a set of Web development techniques using many web technologies on the client side to create asynchronous Web applications. [2]

1.3.3 NoSQL, Google’s Firebase

A NoSQL database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases. [3]

Firebase is a mobile and web application development platform developed by Firebase, Inc. in 2011, then acquired by Google in 2014. As of October 2018, the Firebase platform has 18 products which are used by 1.5 million apps. [4]

1.3.4 Google App Engine

Google App Engine is a web framework and cloud computing platform for developing and hosting web applications in Google-managed data centers. It provides backend logic and supports many languages including Python and Java. [5]

1.3.4 JSON

In computing, JavaScript Object Notation (JSON) is an open-standard file format that uses human-readable text to transmit data objects consisting of attribute-value pairs and array data types (or any other serializable value). [6]

1.3.5 Android

Android is a mobile operating system developed by Google. It is based on a modified version of the Linux kernel and other open source software and is designed primarily for touchscreen mobile devices such as smartphones and tablets. [7]

1.3.6 iOS

iOS (formerly iPhone OS) is a mobile operating system created and developed by Apple Inc. exclusively for its hardware. It is the operating system that presently powers many of the company's mobile devices, including the iPhone, iPad, and iPod Touch. It is the second most popular mobile operating system globally after Android. [8]

1.3.7 API

In computer programming, an application programming interface (API) is a set of subroutine definitions, communication protocols, and tools for building software. In general terms, it is a set of clearly defined methods of communication among various components. A good API makes it easier to develop a computer program by providing all the building blocks, which are then put together by the programmer. [9]

2. Current Software Architecture

In this chapter, the existing systems which are similar with Moodio will be presented. At the end of the chapter, the features that offered by Moodio different than the other existing systems will be explained.

2.1 Similar apps and platforms

2.1.1 Spotify

Spotify is one of the most popular and successful music application in the field.It is a music streaming platform developed by Swedish company Spotify AB

➢ Available on mobile for Android, Iphone and Windows Phones. Also Spotify can be used on web platform
➢Personalized music list known as playlist. User can choose the category of interest and create playlist from his/her interested music types. Also user can create playlists that contain a combination of local files on his/her computer and tracks from Spotify's own library
➢Spotify recommends the songs according to the music preferences of the each user specifically.
➢ Users can add each other as a friends and people who are friends with each other can see the songs that are listened by the other people
➢The user can select the playlist based on his/her own mood among the categorized playlist preparing according to the certain moods
➢The user can download his/her favorite songs from Spotify.

2.1.2 Deezer

Deezer is also famous music application in this field. Deezer is an Internet-based music streaming service. It allows users to listen to music content from record labels including Sony Music, Universal Music Group, and Warner Music Group on various devices online or offline. Created in Paris, France, Deezer currently has 53 million licensed tracks in its library, with over 30,000 radio channels, 14 million monthly active users, and 6 million paid subscribers.

➢Available on mobile for Android, Iphone and Windows Phones. Also Deezer can be used on web platform
➢The user can discover a playlist or create his / her own, and discover the collections prepared by genre
➢The user does not need an internet connection to listen to his favorite music. Once downloaded, the user can listen at any time.
➢User-specific personal recommendations allow users to discover newcomers, sports, podcasts and much more
➢The user can create his own music archive from playlist, album, mix and much more, and if the user already has an archive he prepared, he can transfer his favorite MP3 files and playlists to Deezer
➢The user can follow the lyrics of the song he is listening to using Deezer

2.1.3 Pandora

Pandora is a music streaming and automated music recommendation internet radioservice powered by the Music Genome Project. The service plays songs that have similar musical traits. Basically, Pandora is an Online Radio station that lets you listen to and stream similar songs to you as you write a favorite song. Pandora has been classified into categories according to thousands of songs, tones, genres, etc. from the database of the site. After selecting a song, the characteristics

of the song in the database are read and other songs close to these features are randomly presented and presented to the user.

➢Available on mobile for Android, Iphone and Windows Phones. Also Pandora can be used on web platform
➢The user can tune into established genre stations, other users' stations or create their own stations based on their musical interests.
➢The user then provides positive or negative feedback (as thumbs up or thumbs down) for songs chosen by the service, and the feedback is taken into account in the subsequent selection of other songs to play.
➢Each track played can be responded to with favorable (thumbs up) or unfavorable (thumbs down) buttons, which determine if it and similar songs should be played in the station

2.1.4 Shazam

The application can identify music, movies, advertising, and television shows, based on a short sample played and using the microphone on the device. This application brings out the music that is playing in your current environment, which songs are singing by who, and the video of the song on youtube.

➢Available on mobile for Android, Iphone and Windows Phones.
➢The user can find the song that listen at that moment using the microphone of his/her device
➢The user can see a list of all songs that are found so far.
➢The user can share the music that he/she found in social media environments
➢The user can see the lyrics of songs that he/she found.

2.2 Moodio

The application that is proposed aims to gather important features of services that are mentioned above. The main difference between Moodio and the other music applications is that Moodio will help users in the stage of music playlist selection. Although the other music applications suggest the music playlist according to the music preferences of users, Moodio will offer the music playlist according to the mood of users specifically determined from their facial analysis and voice expression. Moodio aims to increase the music experience of user according to the his/her music taste based on the facial analysis and voice expression of user.

3. Proposed Software Architecture

3.1 Overview

In this section, the software architecture and the subsystem decomposition of the system are explained in detail. First, the partition of the system to different layers is explained by using diagrams and classes. Then, hardware/software mapping of the system is given with a component diagram, which shows how different parts of our system will work in different hardware components. After that, persistent data management section is given in order to explain our database systems and objects. Access control and security part will explain the boundaries and the security measures that we will take in our system. Global software control section explains general flow and how the software is controlled. In the end, boundary conditions section discusses how initialization, terminalization and failure conditions of our system.

3.2 Subsystem Decomposition

Moodio will be based on client-server architecture. The server side is responsible for analyzing the mood of the user according to their photo and voice and suggesting a playlist according to that. Client-side is the mobile application. Client side will use and depend on the server side. Handling user inputs, interacting with the user and getting the photo and voice is client side's responsibility.

Client side will be decomposed in two subsystems, i.e. there will be two layers in the client, these are Presentation Tier and Control Tier. Presentation Tier is basically view and it handles the user-interface functionalities. The Control Tier is controller, client actions are handled such as logging-in to the system, searching, taking a photo and recording the voice, viewing homepage or playlists.

Server-side is also decomposed into two subsystems, which are Data Tier and Logic Tier. In Logic Tier, the core logic of the application lies. In this subsystem, the client's photo and voice are processed and analyzed to find the mood of the user and recommend a playlist from Spotify according to the emotion and the previous data of that user. In Data Tier, there will be the information on user, the voice and photo records and analysis of these records, recommended playlists and songs for each user will be kept in a database server. Briefly, we will have a four-layer architecture in a client-server model.

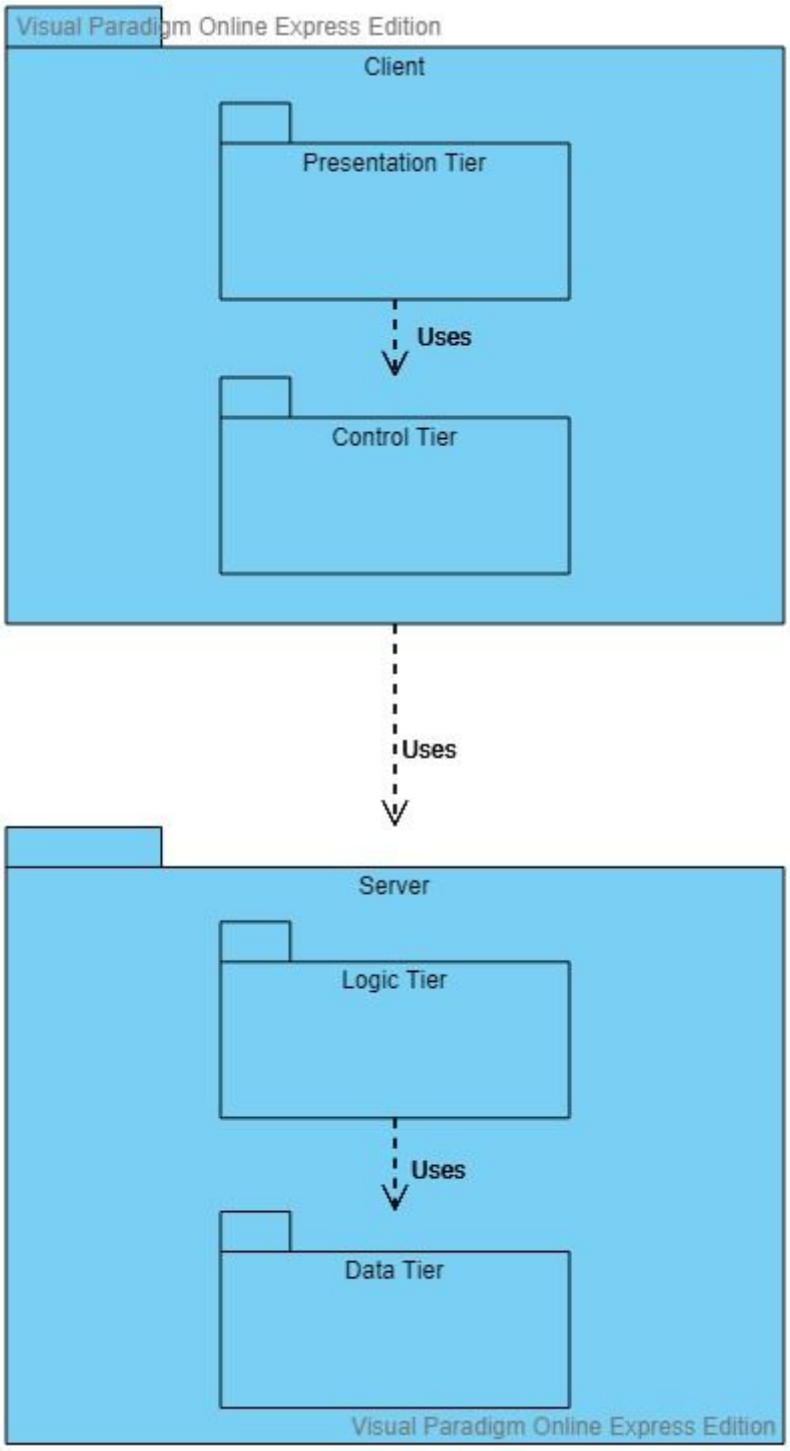

Figure 1: Subsystem Decomposition

Client

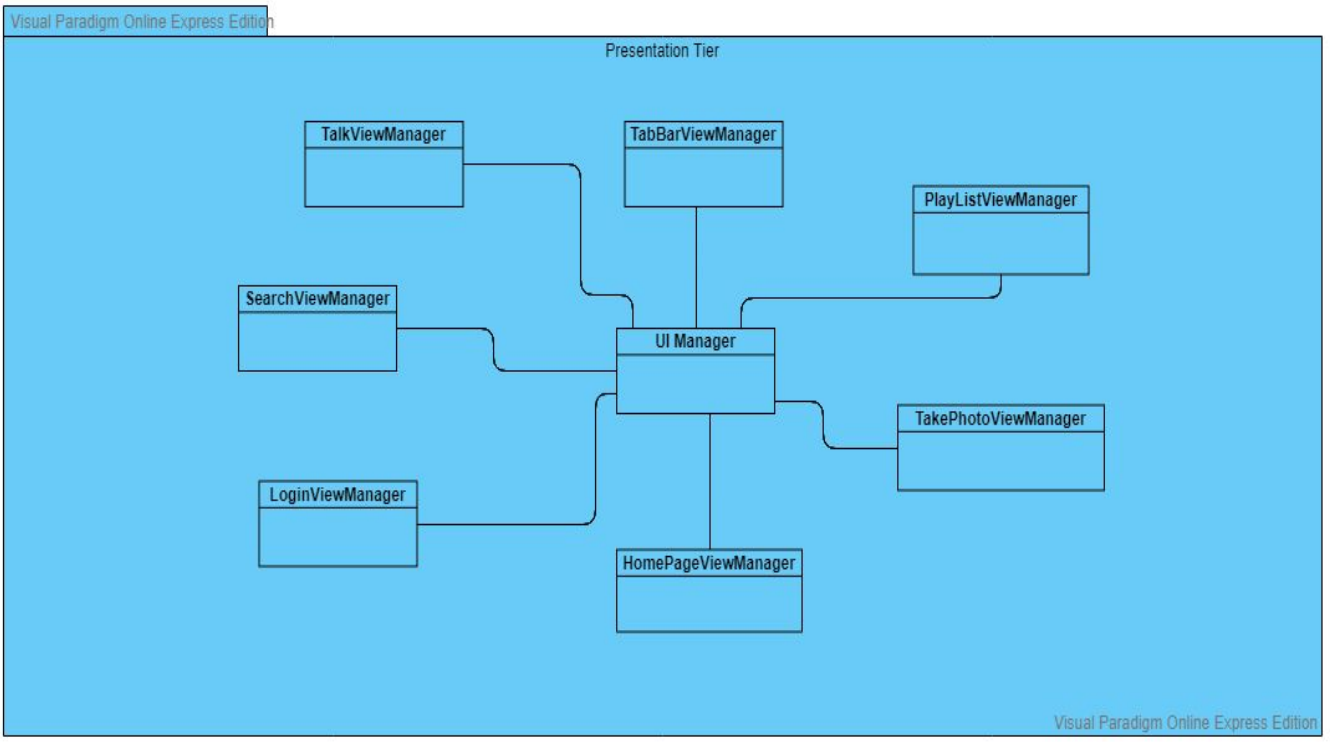

Figure 2: Subsystem Decomposition for Presentation Tier

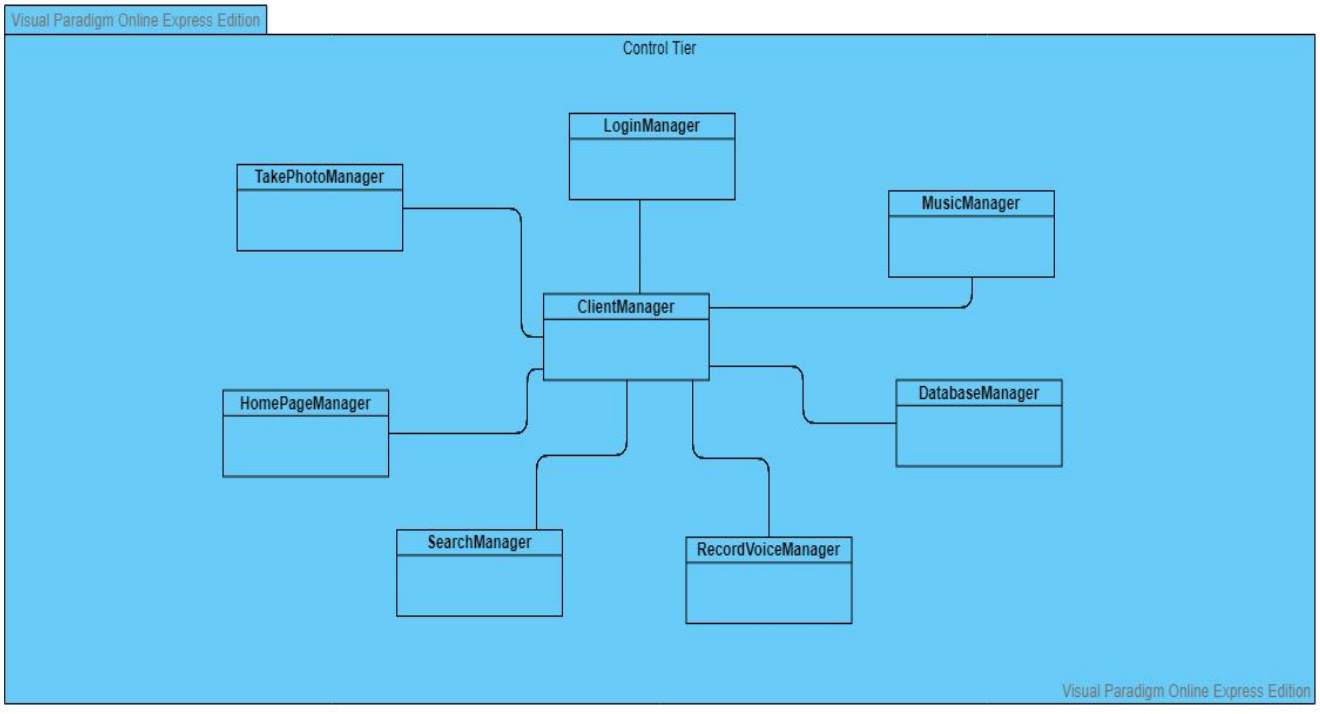

Figure 3: Subsystem Decomposition for Control Tier

Server

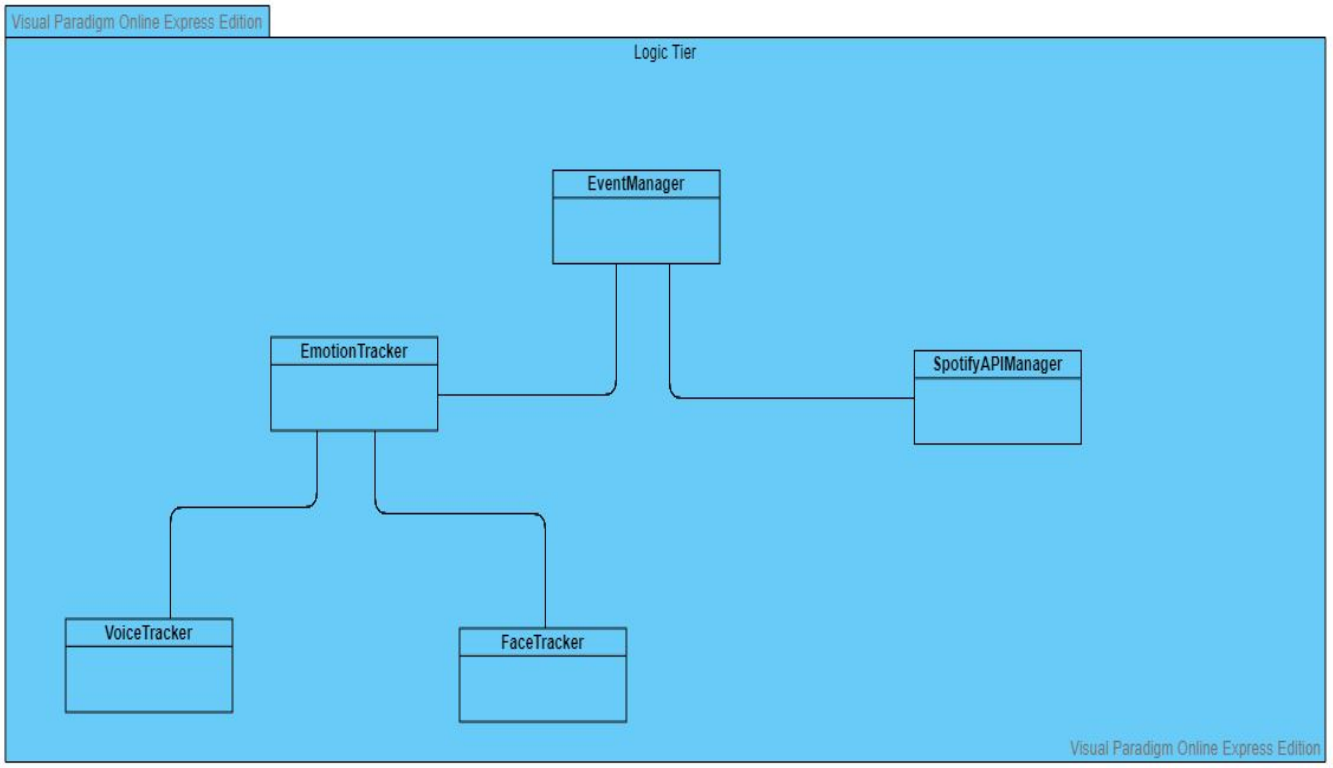

Figure 4: Subsystem Decomposition for Logic Tier

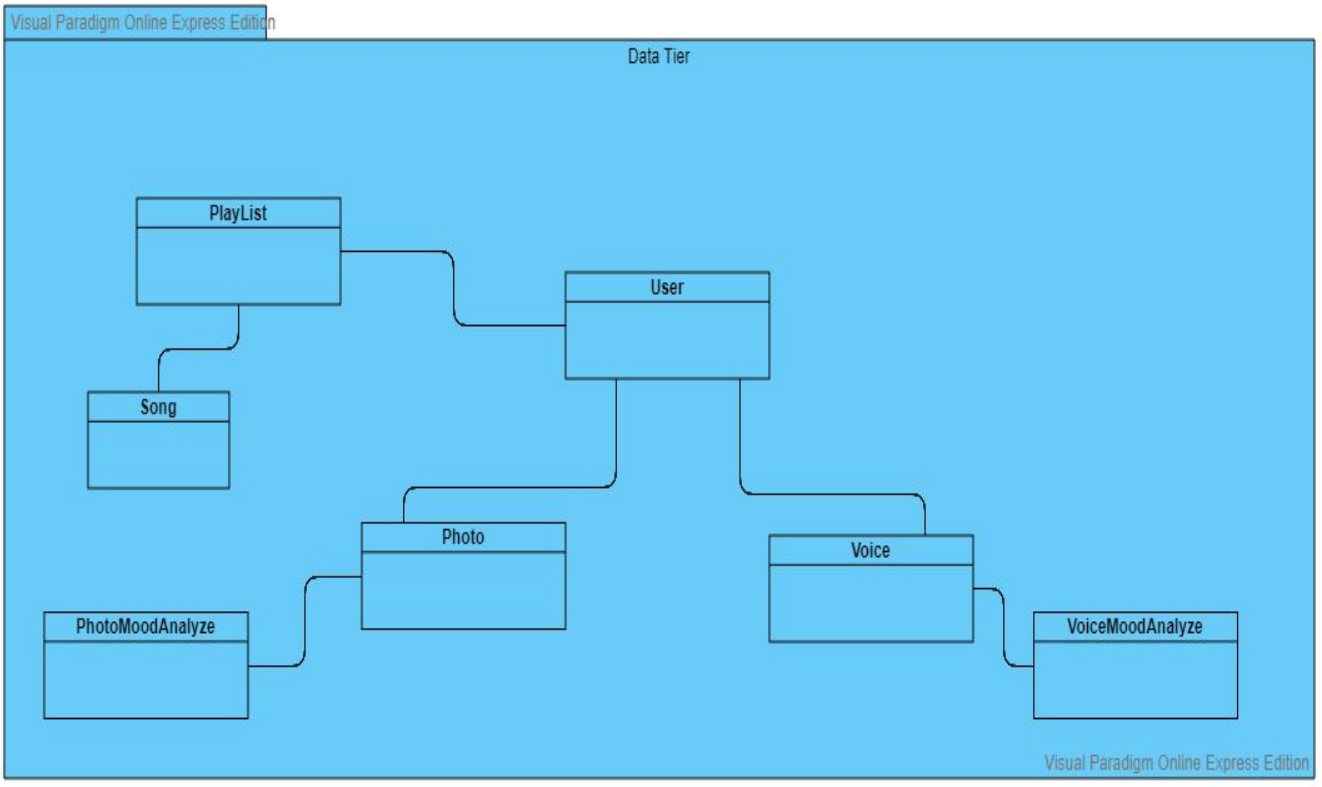

Figure 5: Subsystem Decomposition for Data Tier

3.3 Hardware/software mapping

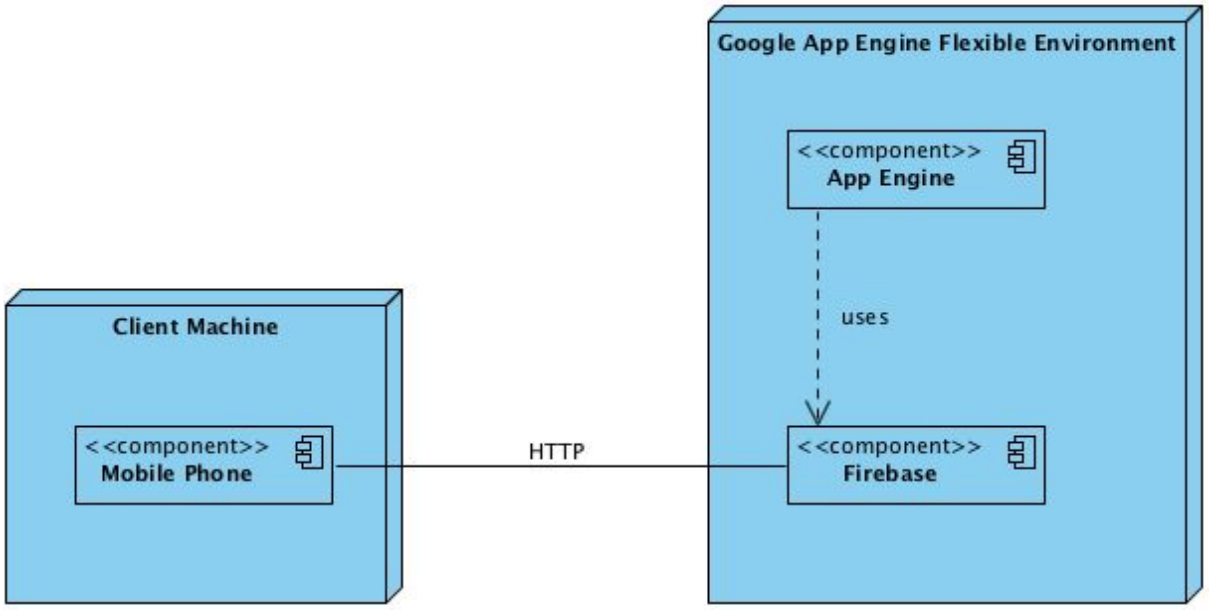

Figure 6: Component diagram

Moodio will be developed for Android and iOS with the help of React Native. Moodio will use phone's resources such as camera, microphone, memory and so on. We will use camera take photo and microphone to record the voice and we will deploy to Firebase to be processed in Heroku.

We are using React Native to use mobile phone's hardware resources. For the communication we will use Firebase SDK. The photo and the recorded audio will be uploaded to Firebase. App Engine watches Firebase for new data. After the upload, App Engine will produce the playlist and store in Firebase. Firebase will return the processed playlist to mobile phone.

3.4 Persistent data management

In Moodio, we need to store objects such as User, Photo, Voice, Analysis Result in order to provide the user with more accurate recommendations and satisfying experience. Hence, we need to store the data in our database and manage it properly because data needs to be persistent. We are planning to store

our data in a cloud-hosted NoSQL database. We plan to use Google's Firebase. Firebase Real-time Database is a cloud-hosted database. The data is stored in JSON and synchronized real time to every client. In our mobile application, each client will share one Real-time Database and receive updates automatically. [10]

3.5 Access control and security

In our system, there is only one type of user, which is a regular user and all users have the same access and permissions in the application. The users can take their photos and record their voices through the application, get the results and see the playlists recommended to them, search for playlists and view homepage and menu bar and choose options from there. Users will not have access to any kind of activities that may harm other users.

To ensure security, every user will login to the application through their Spotify accounts, their Spotify usernames and passwords. We will also their get their data from Spotify API from their accounts, therefore, users need to give a permit for that. Moreover, the users won't be able to get recommendations unless they connect their Spotify accounts since we display the music in Spotify and use their Spotify data in our application. The users will not have access to other users' account information or profiles, their username or passwords.

The application assures the security of the personal information of the users. The data of the user is not shared with third-party applications unless the user gives permission to do so.

3.6 Global software control

Moodio has a centralized, event-driven software control. In server side, we have EventManager that controls the communication between Spotify API Manager, and tracking emotions through the photo and the voice of user.

When user logs in to the system through their Spotify accounts, Spotify API checks the user's Spotify username and password. If they are correct, the user is given access to the application.

When user searches for a playlist, the information is gathered from the SearchViewManager and sent to UIManager. Then, ClientManager is notified about the search and sends the information to SearchManager. The manager searches the key from the Playlist object in the database. It returns the playlists searched and they are displayed to the user.

When a user takes photo or records voice, TakePhotoViewManager and TalkViewManager informs the UIManager and ClientManager gets informed about this request. TakePhotoManager and RecordVoiceManager handles getting the inputs and then sends them to the VoiceTracker and FaceTracker in the server side. Then, EmotionTracker is used to find the user's current mood and this information is sent to the EventManager for recommending the playlist to the user by using the Data Layer in server side. Then, the recommended playlist is displayed to the user to be displayed from Spotify. The user will also rate the playlist that they were recommended and it affects the next recommendations to the user.

3.7 Boundary conditions

3.7.1 Initialization

To use this application, the user needs to have a phone that has Android or iOS. The user can download and install the application to their Android or iOS device and start using it. The user needs to have a Spotify account to use the application. Users will login to the application through their Spotify accounts. Since the application will retrieve the data realtime to function and users will take photos and record voices and these will be processed with the user's data, internet connection is required to use the application.

3.7.2 Termination

Users can logout from the application by clicking logout button and they will logout from their Spotify accounts in the application. Unless they logout while they close the application, users will be automatically login when they open the application again. Moreover, the application will run in the background unless they close it.

3.7.3 Failure

If the internet connection is lost, program will not run. Internet connection is needed for retrieving realtime data, therefore if there is no data, the application will not work. If the connection is lost while user takes a photo or records voice, the recording process will be disrupted and the user will be notified about the loss of connection. If the application cannot get the user's photo or record their voice for some reason, user will be notified and required to do the same process again

4 Subsystem Services

4.1 Client

Since all the functionalities that will be implemented for the mobile applications, the client means mobile applications basically. The client layer is the presentation tier of our project. The user can login in the system using through client layer. The request will be made from client to server in order to login. This service is responsible from the managing user operations such as creating playlist according to the mood, and introducing data from the server. This subsystem contains Control Tier and Presentation Tier. Presentation Tier is responsible for the all visual interactions in the system. Control Tier is used by the Presentation Tier in order to provide communication between server and client.

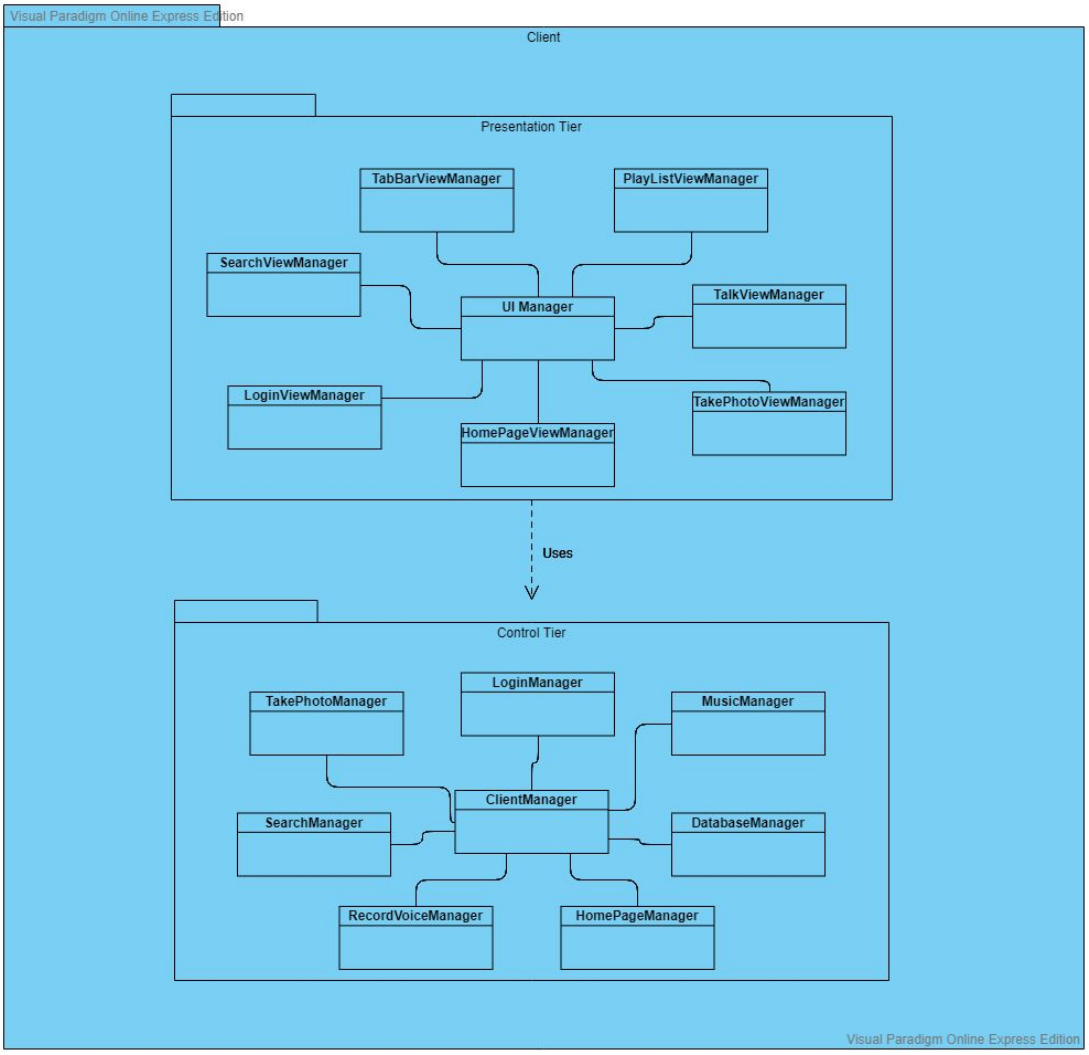

Figure 7: Subsystem Decomposition for Client

4.1.1 Presentation Tier

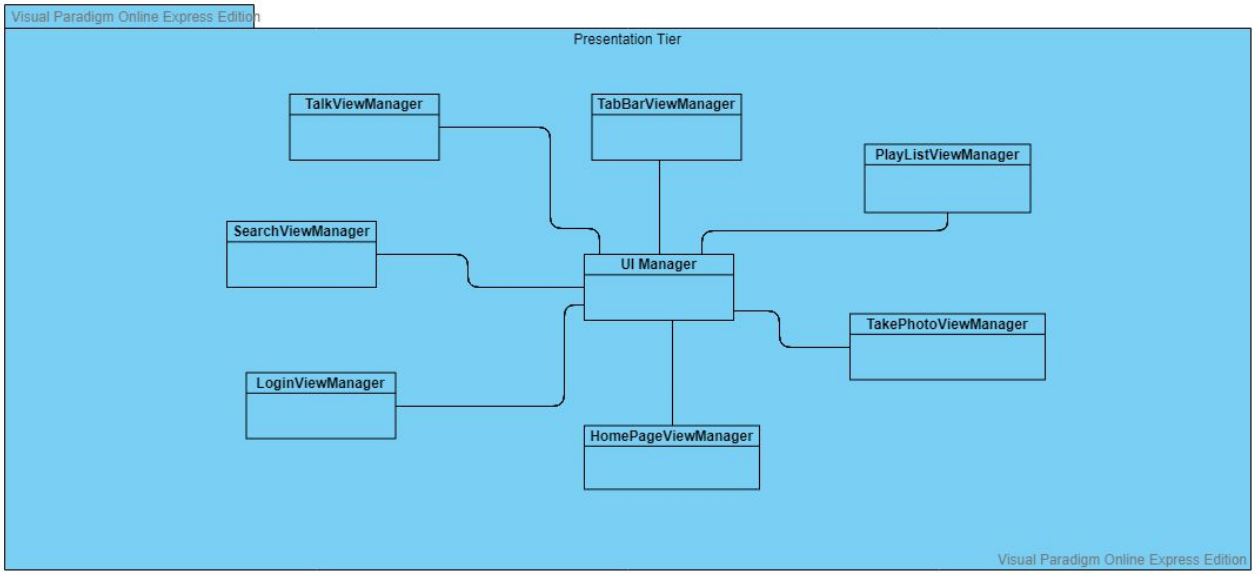

Figure 8: Subsystem Decomposition for Presentation Tier

Presentation subsystem contains the visual operations which are related with UI

➢**UI Manager**: The controller of the Presentation Tier. It operates the task distribution between the other components of Presentation Tier

➢**PlayListViewManager**: This class handles the user interface for the playlist that is created according to the mood of user.

➢**TakePhotoViewManager**: This class handles the user interface for the take photo view which is responsible for the discovering mood from the facial analysis of the user.

➢**HomePageViewManager**: User initially directed to the homepage of the system by the UI manager. In homepage, user either can take her/his own photo to discover

his/her own mood or he/she can record his/her own voice in order to discover his/her own mood.

➢**LoginViewManager**: This class is responsible for arranging all operations related UI of the login screen

➢**SearchViewManager**: This class handles search user interfaces, containing results and searchbar

➢**TalkViewManager**: This class handles the user interface for the record voice view which is responsible for the discovering mood from the voice analysis of the user.

4.1.1 Control Tier

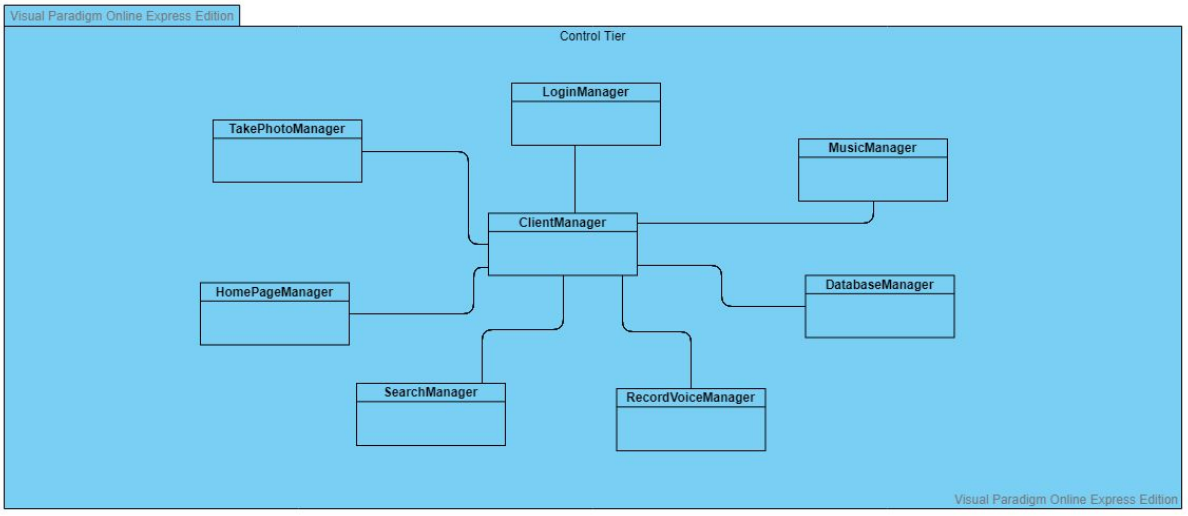

Figure 9: Subsystem Decomposition for Control Tier

Control subsystem is responsible for arranging the operations of client. Also this subsystem provides the communication between client and server.

➢**CilentManager**: The controller of the Control Tier. It arranges the task distribution in the Control Tier.

➢**LoginManager**: This class handles the login operations of the user

➢**MusicManager**: Music Manager class keeps track of each users recommended playlist based on the moods using either the picture or voice of the user. Also it updates the server with the playlist that is determined by the system according to the mood of user.

➢**DatabaseManager**: This class manages the connections between Amazon Web Services and database server.

➢**HomePageManager**: This class keeps track of the list determined by either facial expressions or sound expressions of user. Also provides user to play the recommended music list

➢**SearchManager**: This class handles the categories that is determined by the Spotify and responsible for the making search according to given key by the user

➢**TakePhotoManager**: This class is basically keeps track of the picture that is taken by the user.

➢**ServerConnector**: This class handles the communication between client and server

➢**RecordVoiceManager**: This class basically keeps track of the voice that recorded by user

4.2 Server

4.2.1 Logic Tier

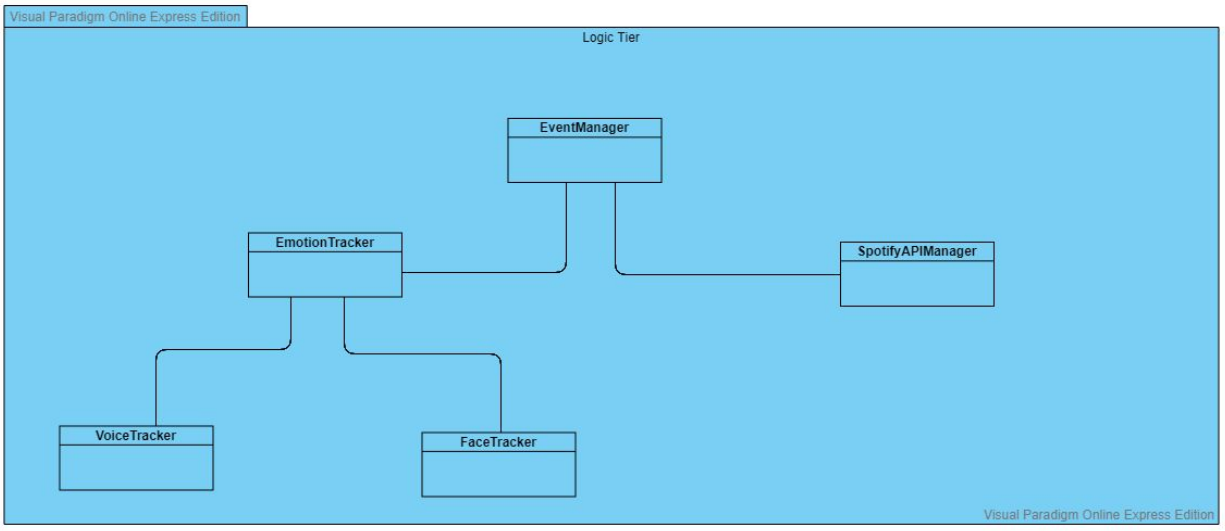

Figure 10: Subsystem Decomposition for Logic Tier

Logic Tier is responsible from the core logic of the application. In this tier, user data is processed and analyzed to recommend a playlist.

➢**EventManager**: This class handles the process of playlist recommendation by gathering the mood information of the user and the data obtained from Spotify.

➢**SpotifyAPIManager**: This class is responsible from getting our data and information from Spotify to classify the songs and recommend a playlist in Spotify.
➢**EmotionTracker**: This class determines the current mood of the user by using the photo taken and the voice recorded by the user.
➢**VoiceTracker**: This class is for getting the voice record of the user and analyze and process it for tracking emotion step.
➢**FaceTracker**: This class gets the photo of the user and analyzes the face of the user for emotion detection.

4.2.2 Data Tier

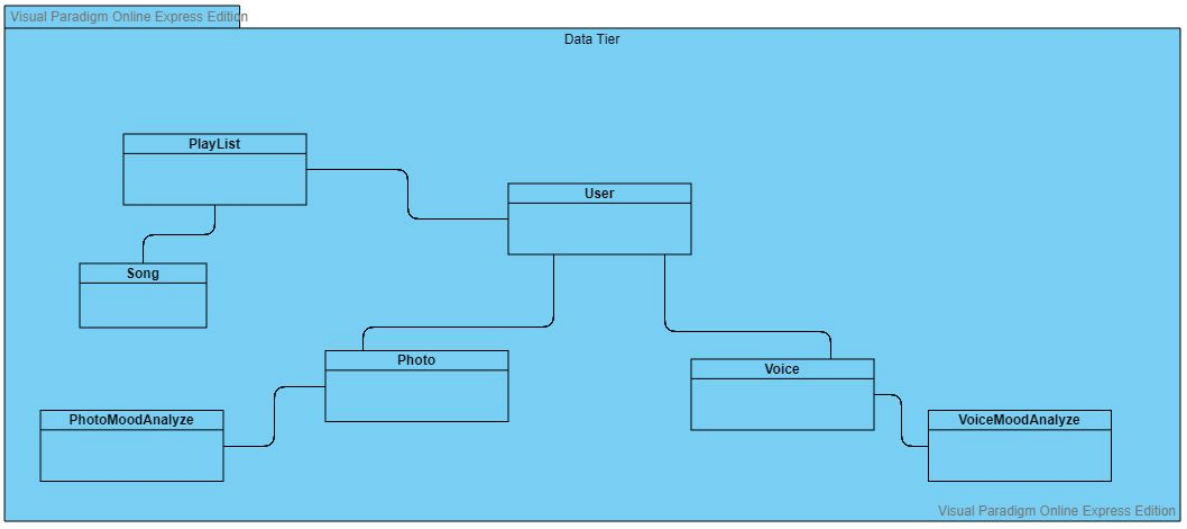

Figure 11: Subsystem Decomposition for Data Tier

Data Tier is the lowest layer of our system. It provides persistent storage of the data. This application layer stores our data in a well-structured way. Also Data Tier sends data that is stored in this layer to Logic Tier. Basically controls the data flow between user and Moodio

➢**User**: This data class represents users of Moodio. User class contains all the data which is related with the user.
➢**Playlist**: This data class keeps the information about unique playlist of each users.

➢**Song**: This data class keeps the seed information about category, singer, name, length and album name of each song

➢**PhotoMoodAnalyze**: This data class keeps the seed information about the mood based on the facial analyze of each user

➢**VoiceMoodAnalyze**:This data class keeps the seed information about the mood based on the voice analyze of each user

5 References

[1] https://en.wikipedia.org/wiki/Spotify

[2] https://en.wikipedia.org/wiki/Ajax_(programming)

[3] https://en.wikipedia.org/wiki/NoSQL

[4] https://en.wikipedia.org/wiki/Firebase

[5] https://en.wikipedia.org/wiki/Google_App_Engine

[6] https://en.wikipedia.org/wiki/JSON

[7] https://en.wikipedia.org/wiki/Android_(operating_system)

[8] https://en.wikipedia.org/wiki/IOS

[9] https://en.wikipedia.org/wiki/Application_programming_interface

[10] https://firebase.google.com/docs/database/

[11] https://en.wikipedia.org/wiki/Deezer

[12] https://www.dummies.com/social-media/spotify/spotify-features/

[13] https://www.deezer.com

[14] https://en.wikipedia.org/wiki/Pandora_Radio