Senior Design Project

Moodio

Low 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

Feb 18, 2019

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.

Table of Contents

1. Introduction

1.1. Object 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. Interface documentation guidelines

In this report, all class names are typed singular. They are all named ‘ClassName’. Attributes (variables) and methods follow the same convention: ‘variableName’, ‘methodName()’. Class descriptions are formed in such order that class name comes first, then the attributes, finally the methods are listed. The detailed outline is provided below:

ClassName

• Description of class

Attributes

• Attribute name

• Type of attribute

Methods

- Method name
- Parameters
- Return value

1.3. Engineering standards

As in the previous reports, UML (Unified Modeling Language) design principles are used in the description of class interfaces, diagrams, scenarios and use cases, subsystem compositions, and hardware-software components depiction. The reason we chose UML is its simplicity, wide-use. IEEE engineering standards are used for design of the project.

1.4. Definitions, acronyms, and abbreviations

UI: User Interface

Client:The part of the system the users interact with

Server:The part of the system responsible from logical operations, scheduling, and data management

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]

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]

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). [3]

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. [4]

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. [5]

2. Packages

Moodio is has 4 packages: Presentation, Controller, Logic, and Data pack- ages. Presentation and Controller packages are in Client subsystem while the Logic and Data packages are in Server subsystem.

2.1 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.

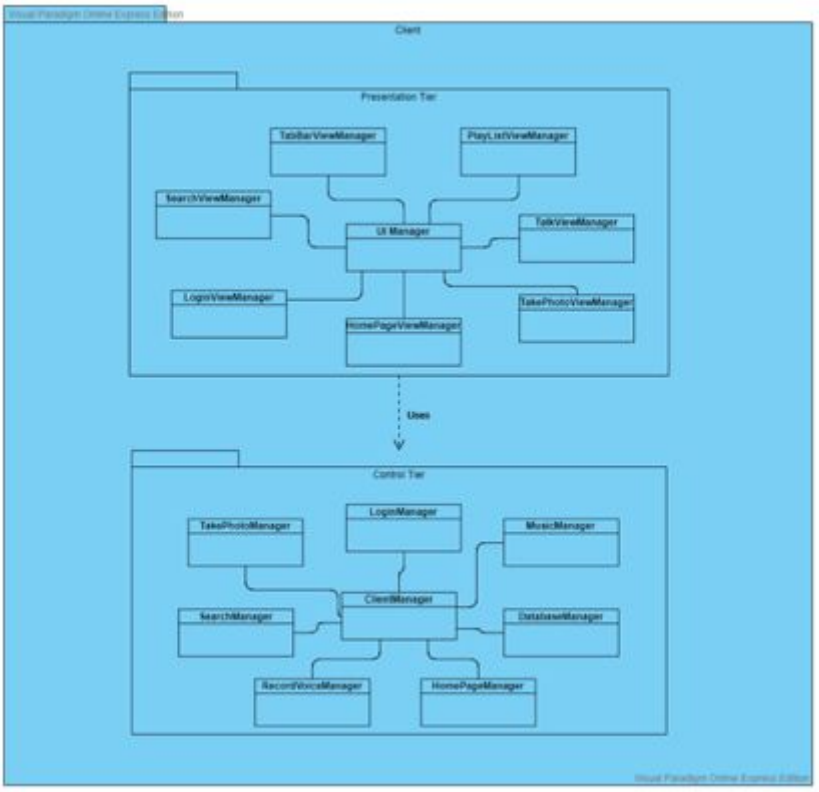

Figure 7: Subsystem Decomposition for Client

2.1.1 Presentation

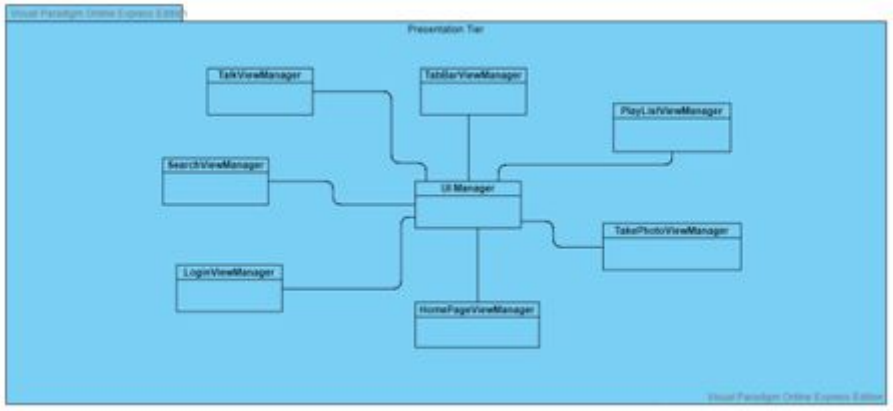

Figure 8: Subsystem Decomposition for Presentation Package

➢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.

2.1.2 Controller

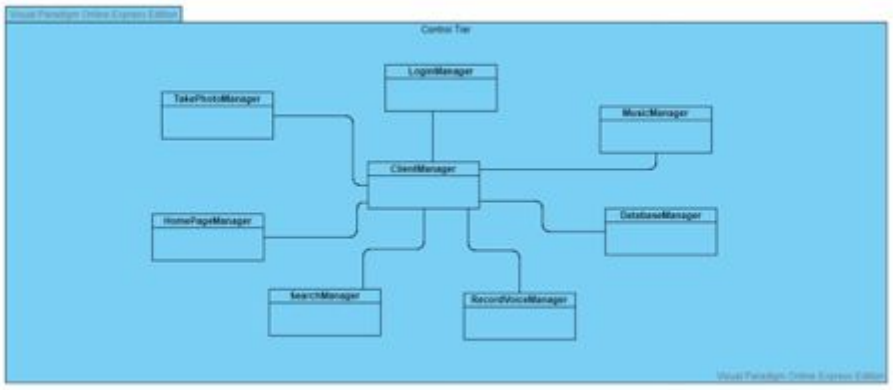

Figure 9: Subsystem Decomposition for Control Package

➢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

2.2 Server

2.2.1 Logic

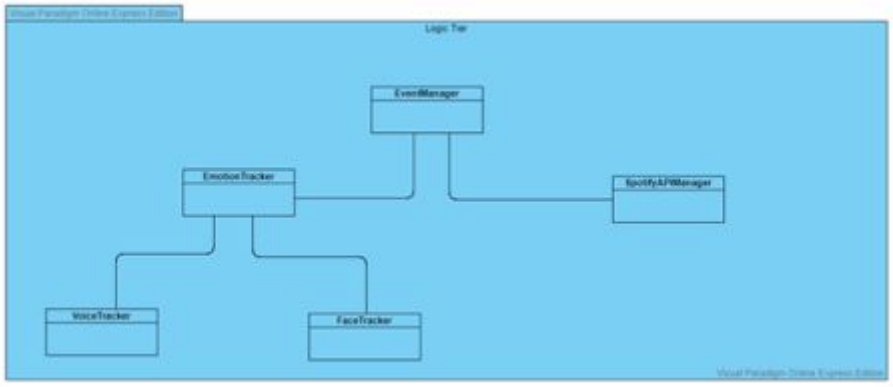

Figure 10: Subsystem Decomposition for Logic Package

➢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.

2.2.2 Data

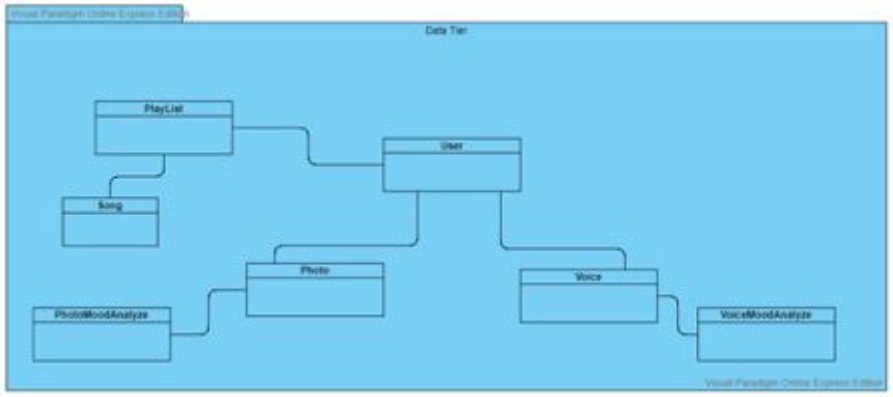

Figure 11: Subsystem Decomposition for Data Package

➢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

3. Class Interfaces

3.1 Client

3.1.1 Presentation Tier Classes

class LoginViewManager
This class is responsible for the UI of the login screen and the operations on it.
Attributes
private String email private String password private String username
Methods
public boolean successfulLogin() : This method is used when the user enters information correctly and redirected to the home page. public boolean failedLogin() : This method is used when the user enter information incorrect and shown the login view again.
getters and setters

class HomePageViewManager
This class handles the view of the home page and the operations users can do on it, such as taking a photo or recording voice for discovering mood.
Attributes
User user
Methods
public boolean viewProfile(User user) : This method is used when user selects profile to be displayed. public boolean viewSearch() : This method is used when user chooses search. public boolean viewPlaylists(User user) : This method is executed when user wants to see the playlists generated before.

public boolean viewVoiceAndPhoto(User user) : This method is used when the user wants to generate a playlist according to their mood and to take photo and record voice.
getters and setters

class SearchViewManager
This class is responsible from the search screen, it contains search bar and the results.
Attributes
String content Mood mood
Methods
public Playlist searchPlaylist(String content) : This method searches the given string in playlists created before and returns them
public List<Playlist> searchPlaylistByMood(Mood mood) : This method returns playlists with given mood.
getters and setters

class TabBarViewManager
This class is responsible from the tab bar, which contains a menu for the functionality of the application.
Attributes
User user
Methods
public boolean ViewHomePage() : This method is used when we direct the user to home page from tab bar.
getters and setters

class PlayListViewManager
This manager class handles the view of the playlist that is generated according to the detected mood of the user.
Attributes

Playlist pl Song song User user
Methods
public boolean play(Song song) : This method plays the selected song in the playlist from the Spotify.
public boolean ratePlaylist(Playlist pl, int rating) : This method is used for rating the playlist that was recommended to the user.
getters and setters

class TalkViewManager
This class is responsible from the UI of the talk view, in which users will record their voice which will be used in detecting the mood of the user.
Attributes
User user
Methods
public Sound recordSound(User user) : This method is used to direct the user to record screen for recording the voice of the user to be used for detecting the mood of the user.
getters and setters

class TakePhotoViewManager
This class handles UI for the take photo view, in which users will take their photos to be used in determining the mood of the user.
Attributes
User user
Methods
public Image takePhoto(User user) : This method is used to direct the user to the take photo screen to get the photo of the user to be used for detecting the mood of the user.
getters and setters

class UIManager
This class is responsible from managing the UI and displaying all contents.
Attributes
public String state
Methods
public void viewHomePage() : This method is used when the home page will be displayed. public void viewTabBar() : This method is used when the tab bar screen will be shown in the application. public void viewSearch() : This method is used when the user selects search and the search screen will be shown. public void viewPlaylist() : This method is used when the user wants to view a certain playlist. public void viewTakePhoto() : This method is used when the user will be shown the take photo screen. public void viewTalk() : This method is used when the user records their voice.
getters and setters

3.1.2 Control Tier Classes

class LoginManager
This class represent the activity of logging in the user to the system
Attributes
private String email private String password private String username private String reEnteredPass
Methods
public boolean checkLoginInfo(String email, String password): Gets email and password. According to the verification of these, updates the database with current e-mail and password.
public int createAccount(String username, String password, String email): This method gets username, email and password. It creates an account with given parameters in database for each user

getters and setters

class SearchManager
This class responsible for the making search according to given key by the user
Attributes
User user
Methods
public List <SearchResult> getRecentSearches(User user): Gets user object and returns a list of results for recent searches done by each user
public List <SearchResult> getSearches(String keyword): This method returns a list of search results that is related with keyword which is entered by user
public List <SearchResult> getPopularSearches(): This method returns a list of most popular searches in the Moodio
getters and setters

class ServerConnector
This class basically keeps track of the voice that recorded by user
Attributes
-
Methods
public void sendUpdateInfo(parameters): According to the request parameters changes and updates in server
public boolean getInfo(parameters) : This method gets information from the database server
getters and setters

class TakePhotoManager
This class is basically keeps track of the picture that is taken by the user.
Attributes
User user
Methods
public Image takeSelfie(User user) : Gets user object and returns an image of face picture that is taken by user
public Image resizeSelfie(Image img): Gets the image of the picture that is taken by user and resize it then returns new format of picture
getters and setters

class RecordVoiceManager
This class basically keeps track of the voice that recorded by user
Attributes
User user
Methods
public void recordVoice (User user) : This method record the voice of each users, gets user object and returns the voice of user
getters and setters

class MusicManager
Music Manager class keeps track of each users recommended playlist based on the moods using either the picture or voice of the user.
Attributes
List <PlayList> playlist User user

Methods
public <PlayList> getList (User user,String pictureMood,String voiceMood) : This method gets the user object and moods that are found using both voice and image, and returns the appropriate playlist for each user public <Playlist> getListImg (User user, String pictureMood) : This method gets the user object and the mood of user founded from image, returns the suitable playlist to user based on only the mood that is detected by image public <Playlist> getListVoice(User user,String voiceMood): This method gets the user object and the mood of user founded from voice, returns the suitable playlist to user based on only the mood that is detected by voice
getters and setters

3.2 Server

3.2.1 Logic Tier Classes

class AuthorizationService
This class checks the token and authorizes the correct user accordingly.
Attributes
private String token: The token received from Spotify after the user accepts to share his/her Spotify information with Moodio.
Methods
public boolean loginWithToken(String token): Gets token and updates the database with the current token.
public boolean signup(String token): Saves the token of a new user on the database.
getters and setters

class SpotifyAPIService
This class sends requests to the Spotify API.
Attributes

private String token: The token received from Spotify after the user accepts to share his/her Spotify information with Moodio.
Methods
public String[] getUserInformation(String token): Get Spotify profile information of a user.
public String[] getPlaylists(String token): Get Spotify playlists of a user.
public String[] getAPlaylist(int playlistId, String token): Get tracklist of a playlist of a user.
public String[] getFollowingArtists(String token): Get artists who the user follows.
public String[] getUsersSavedTracks(String token): Get the saved tracklist of a user.
public String[] search(int searchKey, String token): Get artists, albums or playlists according to the search key.
public String[] getMoodPlaylist(String mood, String token): Get a playlist according to the user's mood.
public boolean savePlaylist(int userId, String[] Playlist, String token): Save a playlist to user's Spotify account.
public String[] getGenres(String token): Get genre categories of Spotify.
getters and setters

class EmotionService
This class detects the emotion.
Attributes
private String token: The token received from Spotify after the user accepts to share his/her Spotify information with Moodio.
Methods
public String detectWithPhoto(Image selfie): Detect the mood according to the user's face in selfie photo.
public String detectWithAudio(Clip audio): Detect the mood according to the user's speech.
getters and setters

3.2.2 Data Tier Classes

class User
This class represents the users
Attributes
private int id
private String email
private String username
private String password
private String token: The token received from Spotify after the user accepts to share his/her Spotify information with Moodio.
private <List>Playlist playlists
private <List>String photos: Names of the photo files
private <List>String voices: Names of the voice files
Methods
getters and setters

class Playlist
This class keeps information about a generated playlist
Attributes
private int id
private String spotifyId: The unique id generated by Spotify
private String name
private String description
private String photo
private String voice
private DateTime dateCreated

private <List>Song songs
Methods
getters and setters

class Song
This class keeps information about a song in a playlist
Attributes
private int id
private String spotifyId: The unique id generated by Spotify
private String name
private String singer
private String length
private String album
private double valence: This value is used for determining mood of the song
private double arousal: This value is used for determining mood of the song
Methods
getters and setters

class PhotoMoodAnalyze
This class keeps information about the mood based on the facial analyze of the user
Attributes
private int id
private double valence: This value is used for determining mood of the photo
private double arousal: This value is used for determining mood of the photo
Methods
getters and setters

class VoiceMoodAnalyze
This class keeps information about the mood based on the voice analyze of the user
Attributes
private int id
private double valence: This value is used for determining mood of the voice
private double arousal: This value is used for determining mood of the voice
Methods
getters and setters

4. Glossary

API: Application Programming Interface

UI : User Interface

Ajax: Asynchronous JavaScript And XML

JSON: JavaScript Object Notation

5. References

[1] https://en.wikipedia.org/wiki/Spotify
[2] https://en.wikipedia.org/wiki/Ajax_(programming)
[3] https://en.wikipedia.org/wiki/JSON
[4] https://en.wikipedia.org/wiki/Android_(operating_system)
[5] https://en.wikipedia.org/wiki/Application_programming_interface [10]
https://firebase.google.com/docs/database/