

# MONTCLAIR STATE UNIVERSITY

## Emotion Detection and Music Player System Software Project Documentation

by  
*Team SKYN*

### **Team Members:**

Addo, Kwame

Hassan, Yewande

Mahmood, Nusaiba

Sapp, Shantia

### **Prepared for:**

CSIT515\_02SP24

Software Engineering

Dr. Hubert A. Johnson

07 May 2024

## Thesis statement

This document is intended for use by all members of **Team SKYN** (listed above), the client and all other project stakeholders. The client wants a software system that uses facial recognition to detect emotions and automatically play a song based on the detected emotion. The human face plays an important role in the relationship between an individual's behavior and emotional state. Music also affects an individual's emotional state and can influence various emotions. Team SKYN aims to utilize this relationship between music and emotions to create a software product that will use facial recognition to detect the user's emotions, and then automatically select and play a song that aligns with this detected emotion. Team SKYN's objectives consist of analysis of the client's requirements and specifications, design of the software product with a clear and well-planned design approach, development of the software product, testing and evaluation of the product, and finally delivery of the good quality software product to the client, to achieve this project's goal.

Project Time Frame: 3.5 Months

Start Date: Jan 16th 2024, End Date: May 1, 2024

# Table of Contents

<b>Table of Contents.....</b>	<b>1</b>
<b>SOFTWARE SPECIFICATION DOCUMENT.....</b>	<b>4</b>
<b>1. Introduction.....</b>	<b>5</b>
1.1 Purpose.....	5
1.2 Intended Audience.....	5
1.3 Feasibility.....	5
• 1.3.1 Time Feasibility.....	5
• 1.3.2 Technical Feasibility.....	5
• 1.3.3 Financial Feasibility.....	6
1.4 Cost Estimate.....	6
1.5 Product Scope.....	6
<b>2. Overall Description.....</b>	<b>7</b>
2.1 Product Perspective.....	7
2.2 User Class and Characteristics.....	7
2.3 Product Features.....	7
2.4 Assumptions, Dependencies, and Constraints.....	8
<b>3. System Features and Requirements.....</b>	<b>8</b>
3.1 Functional Requirements.....	8
3.2 External Interface Requirements.....	9
• 3.2.1 Hardware Interfaces.....	9
• 3.2.2 Software Interfaces.....	10
• 3.2.3 User Interfaces.....	10
• 3.2.4 Communication Interfaces.....	10
3.3 Nonfunctional Requirements.....	11
<b>4. Specification Summary and Conclusion.....</b>	<b>11</b>
4.1 Summary.....	11
<b>SOFTWARE DESIGN DOCUMENT.....</b>	<b>12</b>
<b>1. Introduction.....</b>	<b>13</b>
1.1 Abstract.....	13
1.2 Intended Audience.....	13
<b>2. Design Scope.....</b>	<b>13</b>
2.1 Design Scope.....	13
• 2.1.1 Project Description.....	13
• 2.1.2 Project Objectives.....	13
2.2 Product Features.....	14
• 2.2.1 Sign In/Sign Out of the application.....	14
• 2.2.2 Emotion Detection.....	14
• 2.2.3 Add and Play Music.....	14
<b>3. Architectural Overview.....</b>	<b>14</b>
3.1 Architectural Overview.....	14

3.2 Key Components of Design and Architecture.....	15
3.3 Use Case Diagram.....	16
3.4 Class Diagram.....	17
3.5 Sequence Diagram.....	18
<b>4. System Requirements.....</b>	<b>18</b>
4.1 System Requirements.....	18
4.2 Device System Memory.....	18
4.3 Device Operating System Requirements.....	19
4.4 Python 3.11.8.....	19
<b>5. Design Considerations.....</b>	<b>19</b>
5.1 Design Considerations.....	19
5.2 Programming Languages.....	19
5.3 Database.....	20
5.4 Graphical User Interface.....	21
5.5 Facial Recognition and Emotion Detection.....	21
5.6 Music Player.....	21
5.7 Music and Emotions Data.....	21
<b>6. User Interface Design.....</b>	<b>22</b>
6.1 User Interface Flow.....	22
• 6.1.1 Screen 1: Sign In.....	22
• 6.1.2 Screen 2: Sign Up.....	23
• 6.1.3 Screen 3: Welcome <User>!.....	26
• 6.1.4 Screen 4: Add Music.....	27
• 6.1.5 Screen 5: Emotion Detection.....	29
• 6.1.6 Screen 6: Music Player.....	30
6.2 External Interface Requirements.....	31
<b>7. Security Design.....</b>	<b>31</b>
7.1 Security Design.....	31
7.2 User Authentication.....	31
7.3 User Authorization.....	32
7.4 Password Hashing.....	32
<b>8. Deployment and Operations.....</b>	<b>32</b>
8.1 Deployment and Operations.....	32
8.2 Version control system.....	32
8.3 Continuous Integration and Continuous Deployment (CI/CD).....	33
8.4 Implementation Document.....	33
<b>9. Design Summary and Conclusion.....</b>	<b>33</b>
9.1 Summary.....	33
<b>SOFTWARE USER MANUAL.....</b>	<b>34</b>
1. Introduction.....	35
1.1 Introduction.....	35
2. Quick Start Guide.....	35
2.1 Install Python.....	35

2.2 Download Application (GitHub version).....	36
2.3 Install Application (Physical copy version).....	36
2.4 Launch Application.....	36
3. User Interface.....	38
3.1 User Interface Flow.....	38
• 3.1.1 Screen 1: Sign In.....	38
• 3.1.2 Screen 2: Sign Up.....	40
• 3.1.3 Screen 3: Welcome <User>!.....	42
• 3.1.4 Screen 4: Add Music.....	43
• 3.1.5 Screen 5: Emotion Detection.....	45
• 3.1.6 Screen 6: Music Player.....	46
3.2 External Interface Requirements.....	48
4. Contact Team SKYN.....	48
4.1 Feedback, Suggestions, and Support.....	48
<b>SOFTWARE TEST DOCUMENT.....</b>	<b>49</b>
<b>1. Introduction.....</b>	<b>50</b>
1.1 Introduction.....	50
1.2 Abstract.....	50
1.3 Intended Type of Test to Perform.....	50
• 1.3.1 Unit Testing.....	50
• 1.3.2 Integration Testing.....	50
• 1.3.3 Function Testing.....	50
1.4 Environment and Pretest Background.....	50
<b>2. Test Plan.....</b>	<b>51</b>
2.1 Test Plan.....	51
2.2 Functional Requirements.....	51
• 2.2.1 Ability to sign in and sign out of the system.....	51
• 2.2.2 Ability to detect facial emotions within the scope of the system.....	51
• 2.2.3 Ability to play music based on the emotion detected by the system.....	51
• 2.2.4 Ability to quit the application and switch between pages on the application..	52
2.3 Testing (id location).....	52
2.4 Schedule.....	52
2.5 Resource Requirements.....	52
2.6 Testing Approaches.....	52
<b>3. Results.....</b>	<b>53</b>
3.1 Test Cases.....	53
3.2 Accuracy and Limitations of the Data Set.....	55
<b>4. Future Scope.....</b>	<b>55</b>
4.1 Future Scope.....	55
<b>References.....</b>	<b>56</b>

# **SOFTWARE SPECIFICATION DOCUMENT**

# 1. Introduction

## 1.1 Purpose

The purpose of this document is to discuss the requirements and specifications that were initially defined for the Emotion Detection and Music Player software system that has been developed by Team SKYN.

This software system consists of facial recognition and emotion detection functionalities and a music player. The software system uses its facial recognition and emotion detection functionalities to detect the user's current emotion. The software system then plays a suitable song, from an existing list of songs, that aligns with the user's detected emotion. The song is played via the software system's music player.

The primary focus of this project is to deliver the software system, as described above, to the client.

## 1.2 Intended Audience

This document is intended for use by all the members of the development team (Team SKYN), the client, and all other project stakeholders.

This Software Specification document is organized into four main sections as follows:

1. **Introduction**
2. **Overall Description**
3. **System Features and Requirements**
4. **Summary and Conclusion.**

These four main sections are further divided into subsections for additional details.

## 1.3 Feasibility

This section will discuss the feasibility of this project, including its ***Time Feasibility***, ***Technical Feasibility***, and ***Financial Feasibility***.

- **1.3.1 Time Feasibility**

The project will be completed in one academic semester (three and a half months).

- **1.3.2 Technical Feasibility**

Through Team SKYN's collective research, it was concluded that this project is technically feasible and offers the possibility for future improvements based on the client's wants and needs.

- **Input:** The system will be able to recognise human faces from the user's live camera feed.

- **Process:** The system will be able to use the input of facial expressions from the live camera feed to determine the user's portrayed emotions. The system will then match the detected emotion to a suitable song from an existing list of songs within the system itself.
  - **Data Store:** Default music files will be included with the final software product. The user will also be able to add their own music files to the system from the device they are using to access it.
  - **Output:** The system will play a suitable song, from the system's existing list of songs, based on the emotion detected from the user's input as described above. The user will also be able to play, pause, skip, and return to songs, and perform other basic music player functionalities which are described in further detail in section **3.1 Functional Requirements**.
- **1.3.3 Financial Feasibility**
    - **Return on Investment (ROI):** Team SKYN believes that this system will be a beneficial quality of life service to its users. It will remove the user's burden of deciding what song to listen to by allowing them to depend on the system's emotion detection functionalities to make the decision for them. In return, this will be highly beneficial to Team SKYN as the team members will be able to include the project in their personal portfolios. The time invested into successfully delivering this product will indirectly generate revenue for the individual team members as they move forward with their software engineering careers.

## 1.4 Cost Estimate

This section will briefly describe and provide the total cost estimate for developing this system. These resources will be needed to complete and deliver the project successfully.

The **Estimated Total Cost** to complete the project, within the time span of **three and a half months**, is between the range of **\$200,000 ~ \$300,000**. This will be the estimated total cost associated with the project, including initial investment, operational costs, and potential returns.

## 1.5 Product Scope

**Problem:** It is tedious and time-consuming to manually look through a large list of songs and decide on a song to listen to for any given emotion.

**Solution:** Develop a software system which will recognize human faces, and will be able to detect the emotions of an individual based on their facial expressions. This system will then select and automatically play a song from a list of songs based on the detected emotion of the user. For the scope of this project, the emotions that this project's software system can detect will be limited to: Neutral, Happiness, Surprise, Sadness, Anger, Disgust, and Fear.

**Deliverables:** This software system will use a pre-trained machine learning model to recognize different emotions based on facial expressions. The system will use the data set from the



trained model to detect and determine emotions from the user's input (live camera feed) and select a suitable song to play. A suitable song is a song that matches the emotion detected. For the scope of this project, songs will be selected from an existing list of songs within the system. The user will be able to add their own music files to this existing list of songs from within the system.

**Expectations:** A software system will be developed. This system will be expected to recognize if the user input is a human face or not, and be able to decipher emotions from facial expressions found in the user input. The system will then be expected to match the detected emotion to a song and play that song for the user via the system's music player.

## 2. Overall Description

### 2.1 Product Perspective

The developers' perspective for this product is to clearly specify what the goal for the system will be and the methods for accomplishing that goal. The final project product will consist of a complete software system that will function as an emotion detection system, which will be able to play a song from a preset list based on emotions detected by the facial recognition function. The development team will first map out the system's requirements and then find the relationship between specifications needed for the system to function efficiently. All project progress and findings will be recorded and documented to ensure that all actors associated with this software are informed on important modifications and design changes.

### 2.2 User Class and Characteristics

Music plays a key role in enhancing an individual's life. It is an important medium of entertainment for music lovers, listeners, and may also serve as a therapeutic catalyst for certain individuals.

With this in mind, the intended user and target demographic for this product will mainly be university students, faculty, and staff members of Montclair State University.

Primary access and consideration will be given to the students and educators in the CSIT 515 Software Engineering course of Spring 2024.

### 2.3 Product Features

Users of this system will have access to the following functions:

- **Sign In:** The user will be able to sign in and access the application using a valid username and password combination.
- **Sign Up:** The user will be able to create a valid username and password combination to use as credentials to sign in and access the application.
- **Welcome <User>!** The user will be able to navigate to different parts of the application from this screen. These will include **Add Music**, **Emotion Detection**, and also the ability to sign out and return to the **Sign In** screen.

- **Add Music:** The user will be able to add their own music files to the existing list of songs in the system.
- **Emotion Detection:** The user will be able to engage with the system's facial recognition and emotion detection functions, which will detect their current emotion in real-time via a live camera feed.
- **Music Player:** The user will be able to play and listen to music, based on the system detected emotion, from the system's existing list of songs. This feature is described in further detail in section **3.1 Functional Requirements**.
- **Back / Sign Out / Quit:** The users will be able to navigate across the different software application screens. **Back** will allow the user to return to the previous screen. **Sign Out** will return the user to the **Sign In** Screen. **Quit** will allow the user to exit the application.

## 2.4 Assumptions, Dependencies, and Constraints

For this project, the development team will assume the following to be true:

- Specific facial expressions can be associated with specific emotions.
- Emotions shown on an individual's face have direct correlations with specific music types.
- Emotion recognition and detection can only be carried out accurately on a human face.

The possible setbacks and constraints of this **system** will include:

- **Emotion detection efficacy:** Accuracy of emotion detections will affect the selection of suitable songs.
- **Hardware/Software malfunction:** The system will be unable to fulfill its main functionalities if any of the required hardware and/or software interfaces (as described in section **3.2 External Interface Requirements**) malfunction. These will need to be addressed and fixed first before running the system again.

The possible setbacks and constraint of this **project** will include:

- **Time constraints:** Project deliverables will be completed within the specified time frame of one academic semester (three and a half months).
- **Financial constraints:** The project and its deliverables must adhere to and not exceed the cost estimations (as described in **1.4 Cost Estimate**).

# 3. System Features and Requirements

## 3.1 Functional Requirements

- **Facial Recognition:** The software will detect human faces in real time. The system will detect and recognize the user's face by taking into consideration the eye color, nose shape, mouth shape, face shape (oval, square, circular), and the different distances between these facial features. With these considerations, the system will be able to verify that the user's face is a human face, and detect different facial expressions for the **Emotion Detection** function as described below.

- **Emotion Detection:** The system will detect emotions through facial expressions. Different facial expressions will be determined using the **Facial Recognition** function as described above. The system will be able to detect the current emotion of the individual from their facial expressions and translate the facial recognition data into emotions recognizable by the system. The emotions that will be supported by this system are: neutral, happiness, surprise, sadness, anger, disgust, and fear. **Emotion Detection** will be based on the input received directly from the user's live camera feed. The current detected emotion will be displayed on the user interface in real time.
- **Emotions Data:** The seven detectable emotions supported by this system will be stored in an appropriate data structure for use within the system.
- **Music Player:** The system will be equipped with a music player that will be used to play a suitable song after the **Facial Recognition** and **Emotion Detection** is done. This music player will have the functions to play and pause the current song, and skip to the previous or next song in the list.
- **Music Data:** The system will have different music directories for each of the seven detectable emotions. These directories will contain songs that align with the respective emotion for that directory. There will be default music files included in each directory with the final software product. Any additional music files will be provided by the user, from the device they are using to access the system. The users will be able to add their own music files to these directories from within the software application system. The system will select and play music from these directories based on the **Emotion Detection** result.
- **Sign-up and Sign-in:** Users will be able to sign up and sign in with a valid username and password. Valid usernames and passwords will be stored in a database. During sign in and sign up, the entered username and entered password validities will be queried against the system's database to allow or deny access to the rest of the application.
- **Updates, Fixes and User Communication:** If the need arises, the software system will receive updates and fixes by the development team for improvements and new features. The application will be available for initial download and future updates in a GitHub repository. Changes will be communicated clearly and transparently via release notes posted within this repository. Users will be responsible for viewing these communications and downloading any available updates.

## 3.2 External Interface Requirements

### ● 3.2.1 Hardware Interfaces

- **Device:** A computer will be required to run the system's software as it will be designed as a desktop application. Additionally, it will be necessary for the **Device** to have sufficient storage space to accommodate the system's software application and any additional music files provided by the user.
- **Internet Connectivity:** The system's software application will be able to run locally on the user's **Device** but an internet connection will be required to initially download the

application and any of its updates (Certain users may not require an internet connection if they received a physical copy of the software product.) An internet connection will also be required if any required peripherals are connected to the **Device** via WiFi.

- **Audio Output Peripheral:** An audio output peripheral will be required to listen to the audio output from the system. This will be either a built-in speaker, external speaker, or headphones connected to the **Device**.
- **Image Capturing Peripheral:** The system will require an image capturing peripheral to capture the user's face for **Facial Recognition** and **Emotion Detection** (as described in section 3.1 *Functional Requirements*). This will either be a built-in webcam or smartphone camera, or a digital camera connected to the **Device** via USB, WiFi or Bluetooth.

- **3.2.2 Software Interfaces**

- **Operating System:** The device will require a compatible operating system to be able run the emotion detection and music player system software. Compatible operating systems include Windows 10 or higher.
- **Web Browser:** The device may require an updated modern chromium browser to be installed, in order to access the GitHub repository for the initial download of this software system. (Certain users may not require this if they received a physical copy of the software product.)

- **3.2.3 User Interfaces**

- This software system will be a desktop application that will prioritize ease of use and user friendliness. The development team will consider the responsiveness of the system and ensure that the main user interface will look universally appealing on all desktop devices.

- **3.2.4 Communication Interfaces**

- **GitHub Repository:** Users will be able to download a copy of the system's software application from its dedicated GitHub repository to their own device. Both the downloadable package and this repository will contain a **README** file which will communicate important information about the system to the user. This file will also contain information to assist users with the initial setup. Communications regarding future updates and fixes will also be available in the GitHub repository.

### 3.3 Nonfunctional Requirements

- **Privacy and Security:** The system will not collect or share the user's data outside of its intended use within the system's functionalities, thereby ensuring that user data is not compromised. User data includes but is not limited to sign in credentials, live camera feed, emotions detected, and music files added.
- **Real-Time Processing:** The system will be able to optimize functionalities to ensure timely emotion detection and music playback without noticeable delays.
- **Portability:** The system will implement responsive design principles to ensure the application adapts to different screen sizes and resolution, and provides an optimal user experience across various desktop devices. Users will be able to access this application from different desktop devices such as laptops and PCs.

## 4. Specification Summary and Conclusion

### 4.1 Summary

This Software Specification Document is intended for use by all members of **Team SKYN**, the client and all other project stakeholders. The purpose of this document is to discuss the requirements and specifications that were initially defined for the Emotion Detection and Music Player software system that has been developed by Team SKYN. The system mainly consists of facial recognition and emotion detection functionalities. The software system uses its facial recognition and emotion detection functionalities to detect the user's current emotion, and then plays a song that aligns with the detected emotion. Songs are chosen from existing directories within the system and are played via the system's music player.

The primary focus of this project will be on delivering a software system as described above, that will accurately detect the user's current emotion, and select suitable songs to align with the detected emotion, to playback to the user. This software will assume that emotions shown on an individual's face have a direct correlation with music and songs. The system will make use of its implemented functionalities and requirements to ensure that emotions are detected accurately matched to the right songs, within a user-friendly, desktop application interface. Some of the possible setbacks and constraints of this system and the overall project will include: emotion detection efficacy, hardware/software malfunctions, time constraints, and financial constraints.

The intended user and target demographic for this product will mainly be the university students, faculty, and staff members of Montclair State University. First access and consideration will be given to the students and educators in the CSIT 515 Software Engineering course of Spring 2024.

# **SOFTWARE DESIGN DOCUMENT**

# 1. Introduction

## 1.1 Abstract

The purpose of this document is to provide a top-down description of the design and architecture for an Emotion Detection and Music Player system that has been developed by Team SKYN.

This system is a desktop application that lets users detect their current emotion through facial recognition, and then play music on the application's music player, based on the detected emotion. The system supports the detection of the following seven emotions: neutral, happiness, surprise, sadness, anger, disgust, and fear.

## 1.2 Intended Audience

This design document is intended for use by all stakeholders of this project.

The intended user and target demographic for the overall product will mainly be the university students, faculty, and staff members of Montclair State University.

First access and consideration will be given to the students and educators in the CSIT 515 Software Engineering course of Spring 2024.

Although the primary, intended audience has been described above, the final software system's usage will not be limited to this demographic as it will be publicly available on GitHub to all stakeholders (including those that do not fit this demographic).

# 2. Design Scope

## 2.1 Design Scope

This section discusses the boundaries and objectives of the software system.

- **2.1.1 Project Description**

The emotion detection software aims to provide music to CSIT 515 students based on the emotion detected via the system's camera feed. The software application will feature a music player to play music based on the following seven detectable emotions: neutral, happiness, surprise, sadness, anger, disgust, fear.

The overall objective is to implement all functionalities as specified in the **Software Specification Document (Pg 4 - 11)**

- **2.1.2 Project Objectives**

- Develop a system that is a user-friendly desktop application.
- Implement functionalities that allow the users to add their own music to the system, and store them in the respective music-emotion directories within the application.

- Implement facial recognition and emotion detection functionalities that recognize a human face, and detect emotions via the user's camera feed in real time.
- Implement music player functionalities that automatically play a suitable song when a valid human face and emotion is detected via the user's camera feed. A suitable song is defined as a song that aligns with the detected emotion.
- Include sign up and sign in functionalities that allow users to authenticate before gaining full access to the system application.
- Include database functionalities to store the user's sign up/sign in credentials for use within the application.

## 2.2 Product Features

Users of this system will have access to the following functions which are described in further detail in **Section 5 Design Considerations** and **Section 6 User Interface Design**.

### • 2.2.1 Sign In/Sign Out of the application

This process will require the users to enter a valid User Name and valid Password combination.

A valid Username has a length between 4-10 characters, starts with a letter, and contains only numbers and letters (no spaces, no symbols, and no special characters).

A valid Password has a length of at least 8 characters and contains no spaces.

### • 2.2.2 Emotion Detection

This feature allows the user to capture and input their camera feed into the system for facial recognition and emotion detection. Through this feature the system is able to recognize human faces and detect one of the following emotions on the recognized faces: neutral, happiness, surprise, sadness, anger, disgust, fear

### • 2.2.3 Add and Play Music

This feature enables users to add their own music files (.mp3 file type only) to the system's music-emotion directories. Any music that is added to the system's music-emotion directories, can be played via music player once the user has passed through the Emotion Detection feature.

## 3. Architectural Overview

### 3.1 Architectural Overview

This section will describe and illustrate the software system's architecture and structure, by providing an overview of the system's components, subsystems, and their interactions.

The development team's design approach consists of an object-oriented design that identifies classes of objects and their interrelationships. The design will also implement modular



decomposition and focus on assigning functions to different modules. It will be partitioned into subsystems by functionality and built upon pre-existing functionality where necessary.

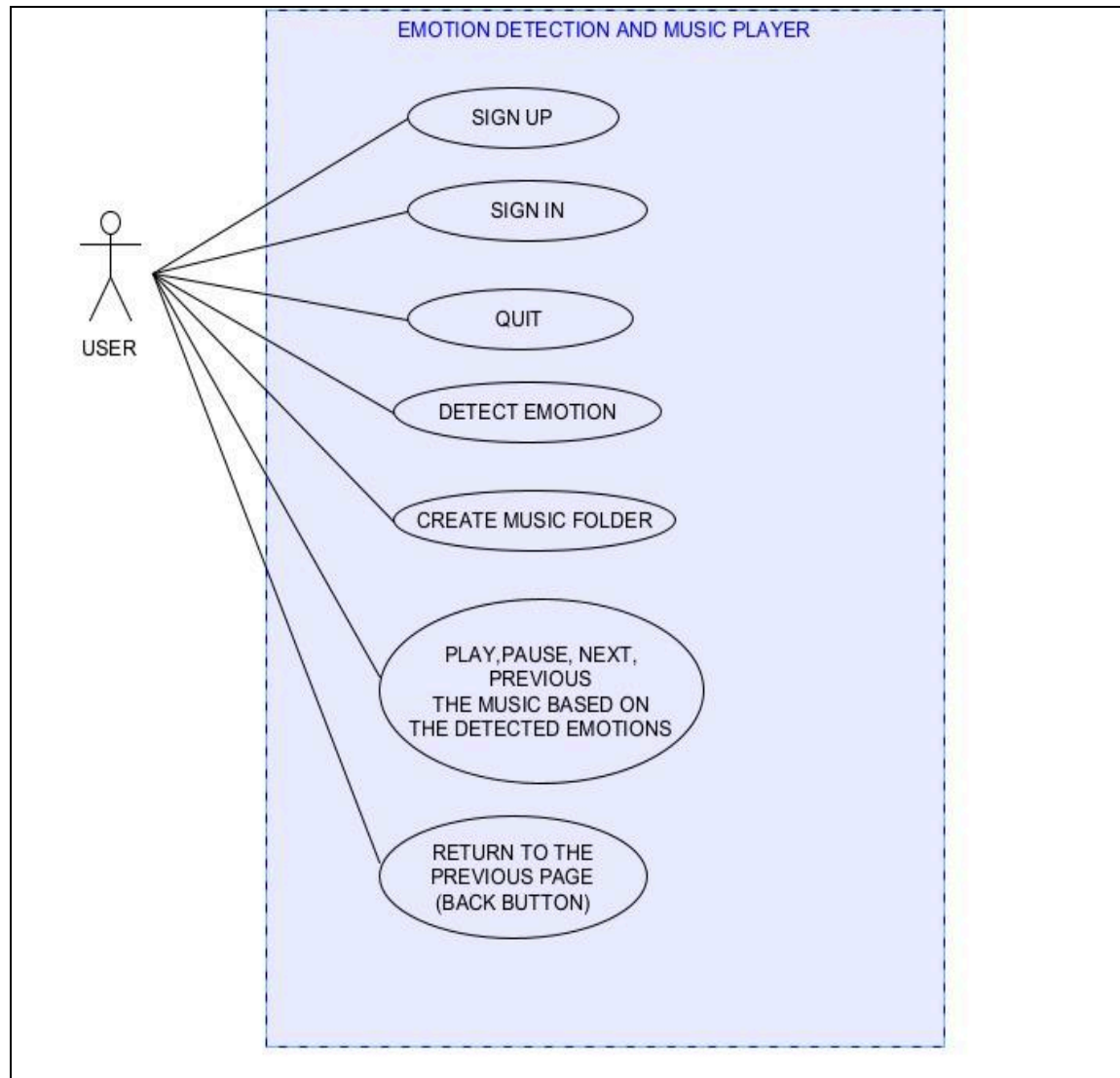
## 3.2 Key Components of Design and Architecture

This section describes the system components and each one's purpose. To learn more about the specific tools, technologies, frameworks and libraries that were used to implement these components, please refer to **Section 5 Design Considerations**.

- **User Interface:** This component allows the user to directly interact with the application product through a user-friendly GUI (Graphical User Interface) that was built specifically for this software application system.
- **Sign In/SignOut Module:** This allows the user to sign in using a valid username and password combination, and gain access to the rest of the application's user screens.
- **Facial Recognition and Emotion Detection:** This handles the user input received from their live camera feed. The input is then compared and interpreted by the system by referencing a pre-trained machine learning model with an existing data set. This data set allows the system to recognize human faces and detect emotions from the user's input.
- **Music Player:** The system has a built in music player - no external music player is used. This allows the user to play music files within the system, based on their emotion detection results. Music is played from predefined music directories within the system. These directories are named and categorized by the seven detectable emotions within this system's scope (neutral, happiness, surprise, sadness, anger, disgust, and fear.)
- **Database:** This component stores the user's sign in credentials locally on the user's device and handles queries related to user authentication. The software application accesses this database whenever the user attempts to sign up or sign in to the application. This is needed to ensure that the user is entering a valid username and valid password, and a valid combination of the two.

### 3.3 Use Case Diagram

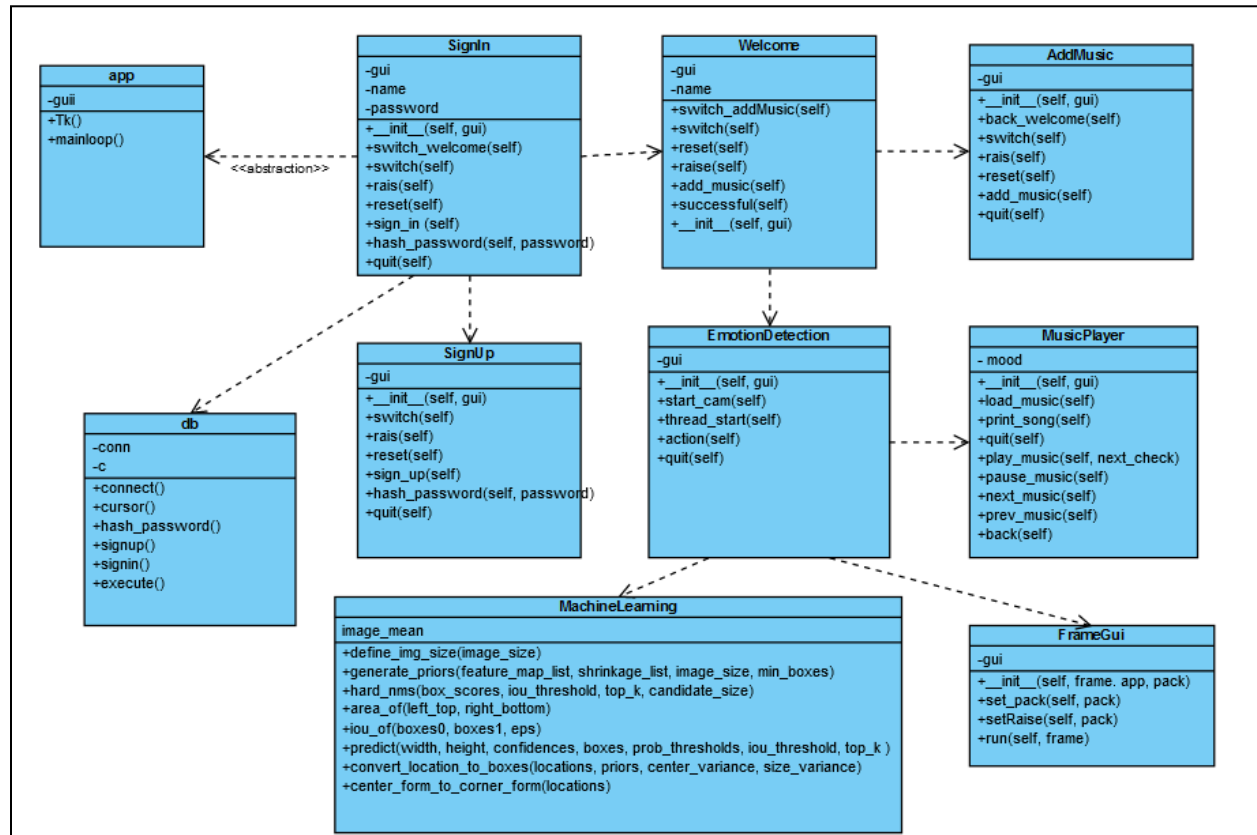
The functionalities performed by various users of the system are illustrated in the Use Case diagram in **Figure 3.3.1** below. The users make use of the components described in **Section 3.2 Key Components of Design and Architecture** to perform these functionalities.



**Figure 3.3.1** Use Case diagram shows the functionalities to be performed by various users of this system.

### 3.4 Class Diagram

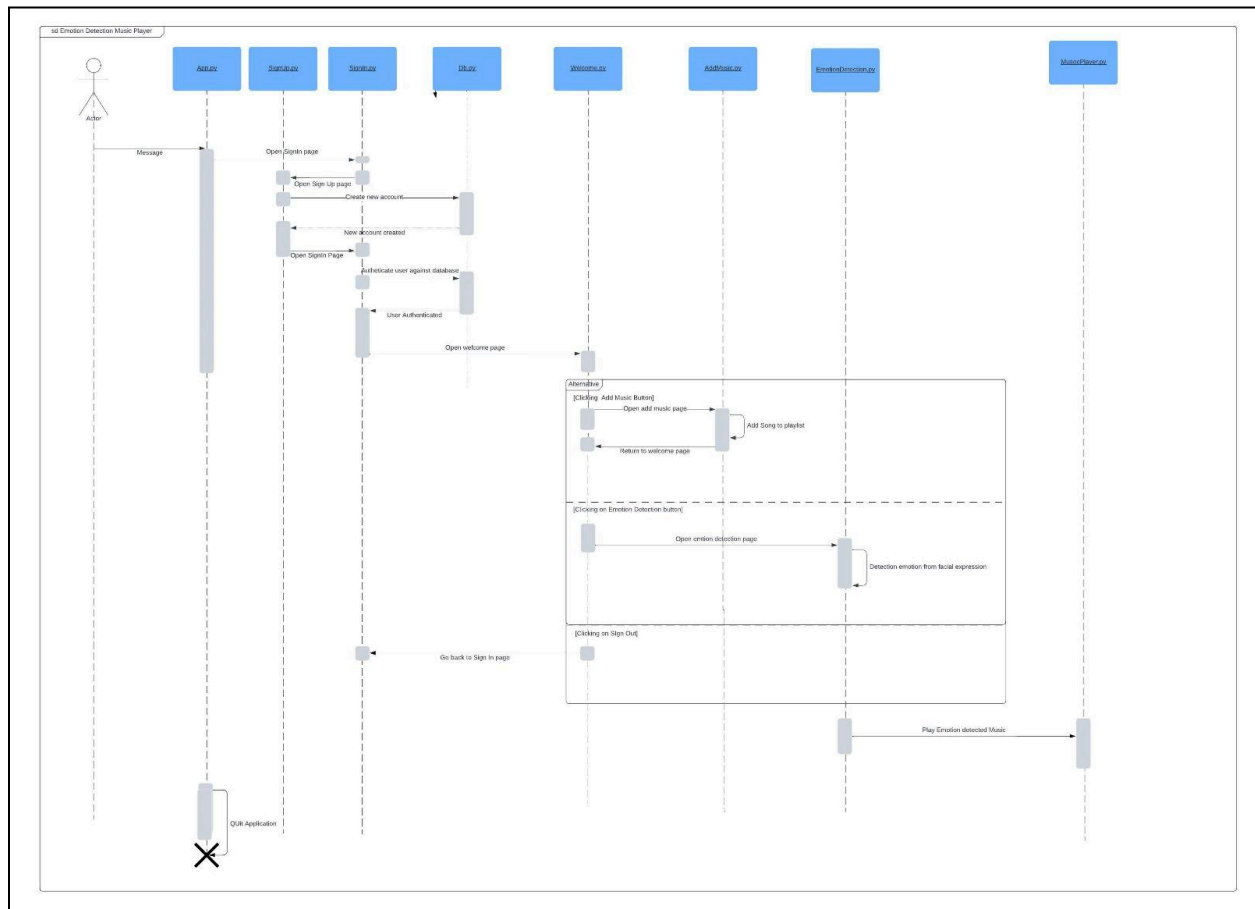
The class diagram in **Figure 3.4.1** shows the relationship among the different classes in the system.



**Figure 3.4.1** Reflects the different modules present in the system

## 3.5 Sequence Diagram

The sequence diagram in **Figure 3.5.1** shows the sequence of operations in the system.



**Figure 3.5.1** Shows the sequence of operations in the system.

## 4. System Requirements

### 4.1 System Requirements

This section briefly discusses the minimum capabilities that a device must satisfy in order to smoothly run this system software application. These requirements were defined and described in detail as the functional and nonfunctional requirements in the **Software Specification Document (Pg 4)**.

To learn more about how and why the following baselines were selected, please refer to the **Software Test Document (Pg 36)** where each requirement is explained as a result of testing the system.

### 4.2 Device System Memory

The user's device will require sufficient memory (16 GB RAM or higher) in order to meet the system performance needs. This is required to process the emotion detection in real time and ensure that it runs smoothly within the user interface. Without sufficient memory, users may

experience frozen frames and flickering screens when running the emotion detection features of the application.

### 4.3 Device Operating System Requirements

The user's device will require an appropriate Operating System (Windows 10 or higher) in order to view the original graphical user interface. Personalization and Theme Customization features in other operating systems (MacOS, Linux) can conflict with or override colors and fonts within the original GUI.

### 4.4 Python 3.11.8

The user's device will require Python 3.11.8 to be installed in order to run the application files.

This can be obtained from the following source:

<https://www.python.org/downloads/release/python-3118/>

## 5. Design Considerations

### 5.1 Design Considerations

This section will discuss the design decisions and trade offs made during the software design process for this application system.

This section will also describe the specific tools, technologies, frameworks, libraries, data structures, and database schemas that have been used to implement the components described in **Section 3.2 Key Components of Design and Architecture**.

### 5.2 Programming Languages

This application has been developed using the **Python 3.11.8** programming language and version.

This programming language and version was selected because it offered the simplest implementation of the Emotion Detection functionality using the pre-existing data set from the Machine Learning Model (described in **Section 5.5 Facial Recognition and Emotion Detection**.)

Additionally, Python has several libraries that allow for quick and easy implementation of the remaining functionalities.

The python libraries that this system utilizes are as follows:

- The **Tkinter** library is used to implement the user-friendly **Graphical User Interface**.
- The **sqlite3** library is used to implement the database for the **Sign In/Sign Up** functionality.
- The **OpenCV** library is used to implement the live camera feed that is needed for **Facial Recognition and Emotion Detection**.

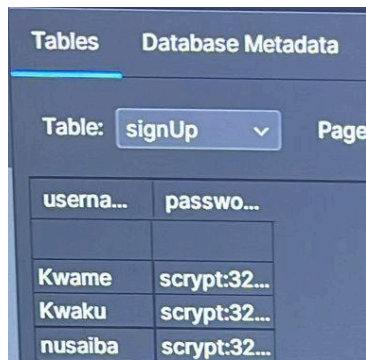
- The **Pygame** library is used to implement the **Music Player** functionalities.

### 5.3 Database

This application uses the **sqlite3** library to implement an embedded database within the system. The database stores the user's sign in credentials and handles queries related to user authentication which is required for the **Sign In** and **Sign Up** functionalities. The database is not visible to the user but it is utilized in the backend by the software application system.

**SQLite** is a database engine that was originally written in the C programming language. But this system does not make use of the C language version. The functionalities of the original SQLite are implemented into the software application product using the **sqlite3** python module version that was written by Gerhard Häring. This module provides an sql interface which is compliant with python's database-api specification.

The system's database consists of valid usernames and hashed passwords. This information is stored locally on the user's device within the *mydatabase.db* file as a flat file database with two columns (See **Figure 5.3.1**). This .db file is automatically created by the software application when the user attempts to sign up or sign in for the first time through the application GUI.



username	password
Kwame	script:32...
Kwaku	script:32...
nusaiba	script:32...

**Figure 5.3.1 Flat File Database Schema sample shows Usernames and Hashed Passwords as they are stored in the *mydatabase.db* file.**

The software application accesses this embedded database, via its backend interface, whenever the user attempts to sign up or sign in to the application from the frontend GUI.

The database query functions are then handled through **sqlite3**. Database queries are run to handle the following cases:

- Usernames must be between 4-10 characters, begin with a letter, and contain only numbers and letters (no spaces).
- Passwords must be at least 8 characters and contain no spaces.
- Usernames must be unique. Users cannot sign up with a username that already exists in the database.
- Username and password must match a valid combination within the database to allow the user to sign in to the application. The user cannot sign in with an invalid username and password combination.

## 5.4 Graphical User Interface

The Graphical User Interface has been designed and implemented using the **Tkinter** library. **Tkinter** is a standard GUI library that is provided with the Python programming language.

For further details on the GUI Design screens and User Interface Flow, please refer to **Section 6 User Interface Design**.

## 5.5 Facial Recognition and Emotion Detection

The application system is able to process live camera feed input to recognize human faces and determine the emotions portrayed. This is done using two things.

First, the system utilizes the **OpenCV (Open Source Computer Vision)** python library to recognize visual input in the form of human faces. **OpenCV** is an open source computer vision and machine learning software library. The system receives visual input directly from the user's live camera feed using the **OpenCV** library's functions.

Second, the application uses the dataset from Microsoft's **FER+** machine learning model to identify emotions. This is a trained machine learning model with an existing dataset.

The application uses the **OpenCV** library and **FER+** model to then interpret a detected emotion to be used to decide what music to play in the music player.

## 5.6 Music Player

### Music Player

After processing the visual input as described in **Section 5.5 Facial Recognition and Emotion Detection**, and finding a suitable song, the system will then play this song from the application's music player screen which is described in further detail in **Section 6 User Interface Design**. The user can then choose to pause, play, or skip the current song.

This built-in music player has been designed and implemented using the **Pygame** python library. It allows the user to play music files (.mp3 file format only) within the system, based on their emotion detection results.

Music is played from predefined music directories within the system. These directories are named and categorized by the seven detectable emotions within this system's scope (neutral, happiness, surprise, sadness, anger, disgust, and fear.)

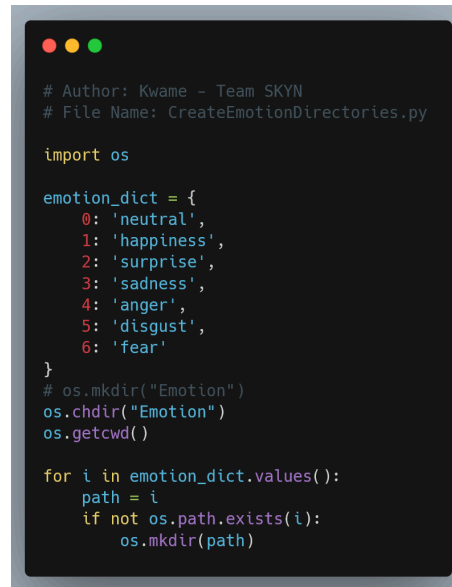
## 5.7 Music and Emotions Data

### Music files (.mp3)

Only .mp3 files are supported by the system's music player. Music files are stored within the application's Emotion directory. Each emotion has its own directory. Users can add their own music files to these directories through the application interface described in **Section 6 User Interface Design**. Music files must be in the .mp3 file format in order to be played via the system's music player.

## Emotions Dictionary

The detectable emotions that will be supported by this system, are stored in a python dictionary data structure within the program's code (See **Figure 5.7.1**). Emotions are listed as key-value pairs with a key ranging from 0-6 for each emotion.



```
# Author: Kwame - Team SKYN
# File Name: CreateEmotionDirectories.py

import os

emotion_dict = {
    0: 'neutral',
    1: 'happiness',
    2: 'surprise',
    3: 'sadness',
    4: 'anger',
    5: 'disgust',
    6: 'fear'
}

# os.mkdir("Emotion")
os.chdir("Emotion")
os.getcwd()

for i in emotion_dict.values():
    path = i
    if not os.path.exists(i):
        os.mkdir(path)
```

*Figure 5.7.1 Emotions listed in a Python dictionary*

## 6. User Interface Design

### 6.1 User Interface Flow

This section describes the user interface flow and presents the different user screens that have been implemented in the final application product.

The application's Graphical User Interface was designed using Python's tkinter library which is described in **Section 5 Design Considerations**. The GUI screens are described below.

#### • 6.1.1 Screen 1: Sign In

This will be the first screen that will be visible to the user when the application is launched. The application will launch in full screen mode by default.

From this screen the user will be able to:

1. Sign in to the application with an existing and valid username and password combination.
2. Navigate to **Screen 2: Sign Up (Figure 6.1.2a)** to sign up for the system.
3. Exit the application.

Sign In button ⇒ validate username and password ⇒ **Screen 3: Welcome <User>!**

Sign Up button ⇒ **Screen 2: SignUp Screen**

Quit button ⇒ Exit the application





**Figure 6.1.1a Screen 1: Sign In**



**Figure 6.1.1b Screen 1: Sign In - Invalid username or password error shown**

- **6.1.2 Screen 2: Sign Up**

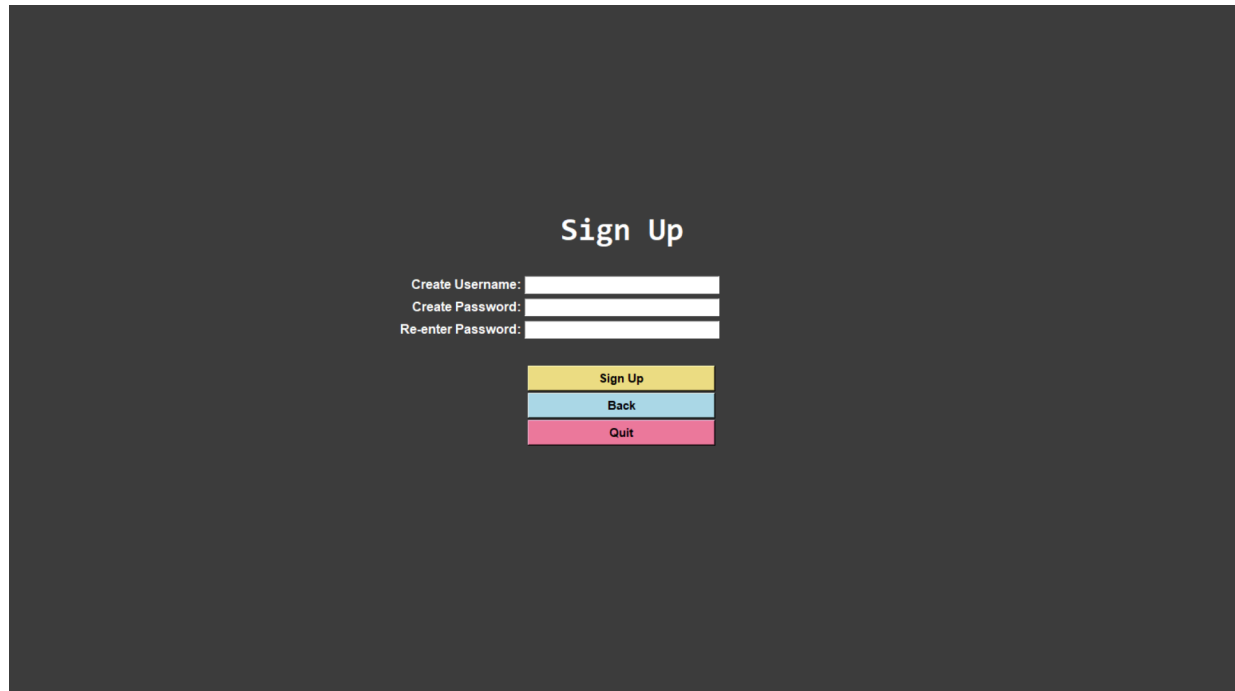
From this screen the user can

1. Sign up for the system using a valid username and password combination.
2. Navigate back **Screen 1: Sign In** (**Figure 6.1.1a**) after successful sign up
3. Exit the application.

Sign Up button ⇒ returns to **Screen 1: Sign In** on successful Sign Up

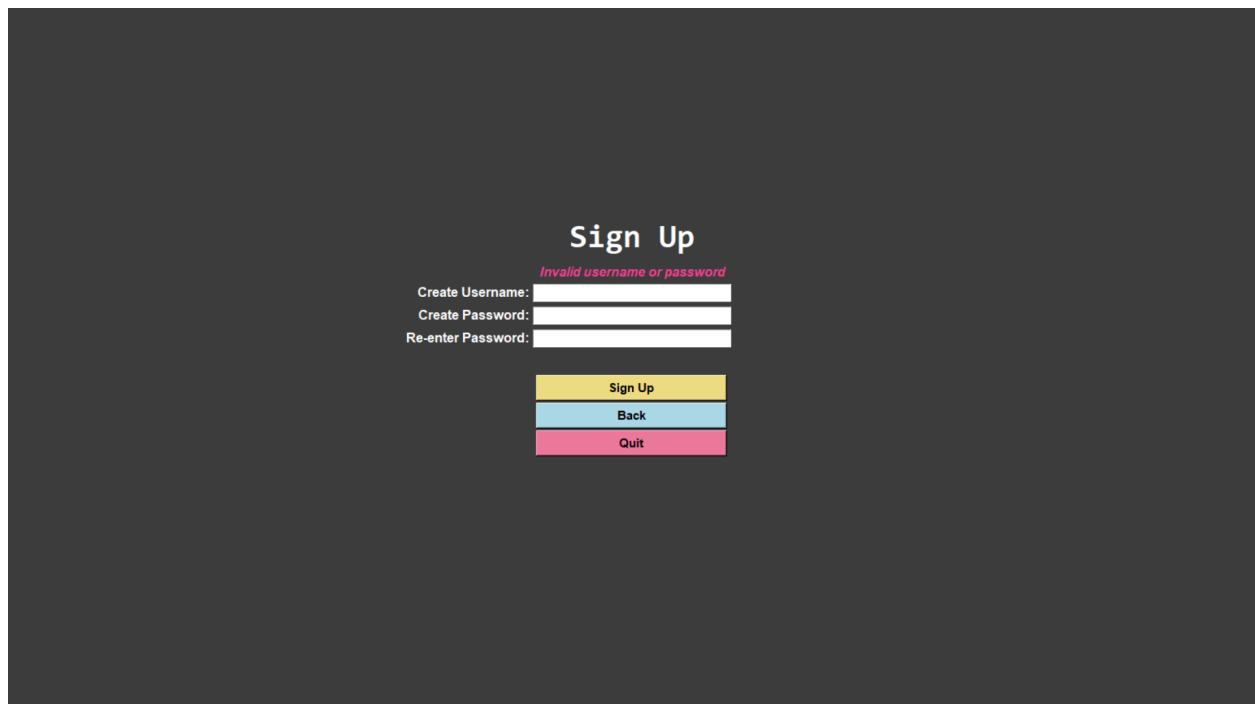
Back button ⇒ returns to **Screen 1: Sign In** without signing up

Quit button ⇒ Exit the application



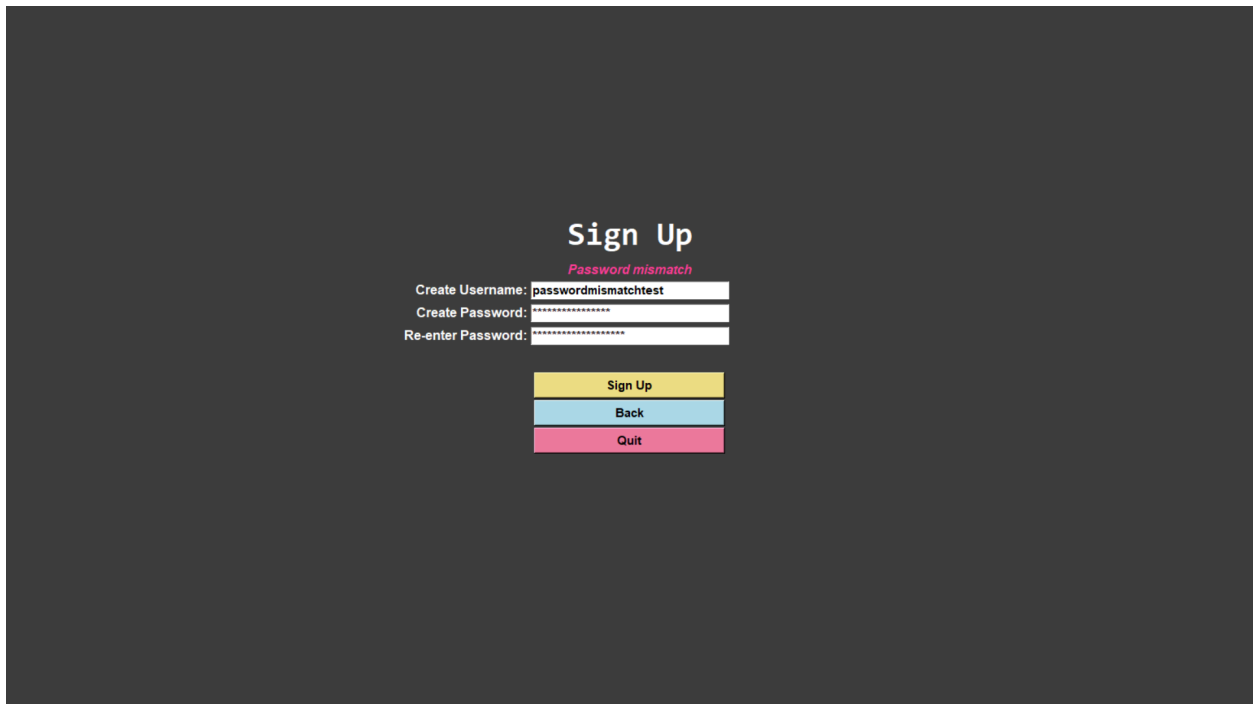
The screenshot shows a dark gray background with the title "Sign Up" in white. Below the title are three white input fields with labels: "Create Username:", "Create Password:", and "Re-enter Password:". Below the input fields are three buttons: a yellow "Sign Up" button, a light blue "Back" button, and a pink "Quit" button.

**Figure 6.1.2a Screen 2: Sign Up**

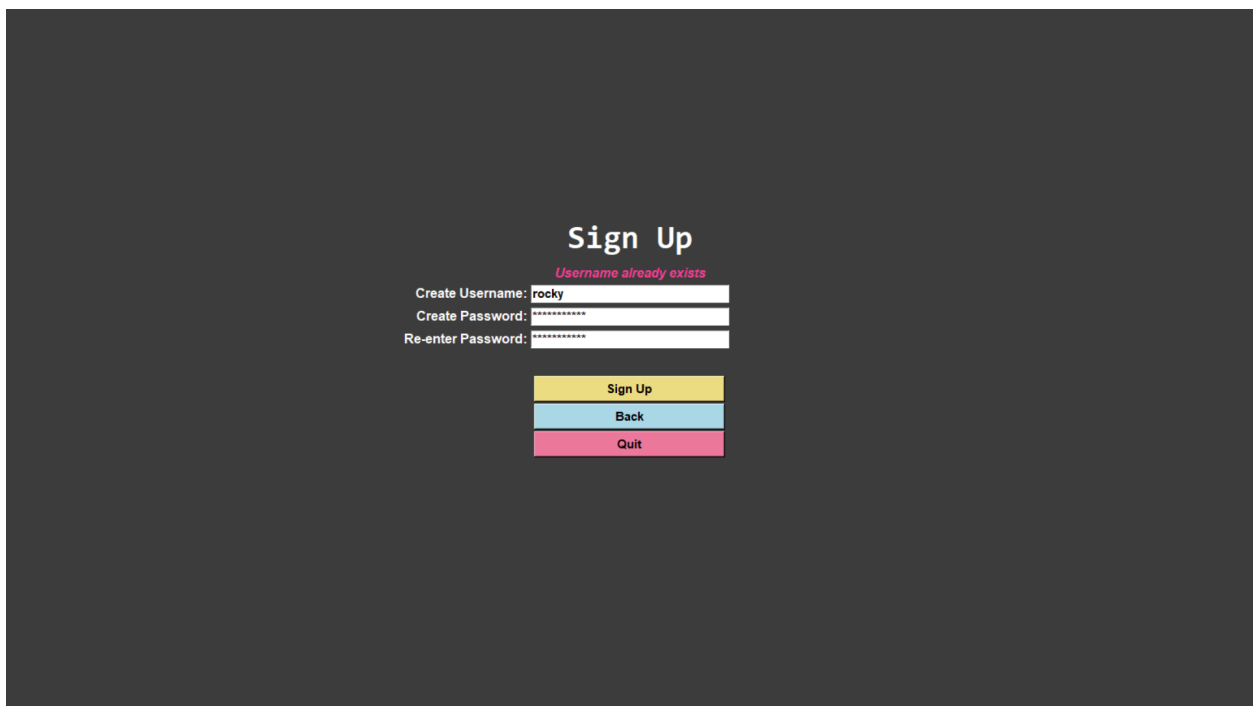


The screenshot shows the same "Sign Up" screen as Figure 6.1.2a, but with an error message "Invalid username or password" in pink text above the "Create Username:" input field. The input fields and buttons remain the same.

**Figure 6.1.2b Screen 2: Sign Up - Invalid username or password error shown**



**Figure 6.1.2c Screen 2: Sign Up - Password mismatch error shown**



**Figure 6.1.2d Screen 2: Sign Up - Username already exists error shown**

- **6.1.3 Screen 3: Welcome <User>!**

The user can only access this screen after successfully signing into the application using a valid username and password combination.

This welcome screen displays the current signed-in user's username as shown in **Figure 6.1.3** with the example of Rocky (the name of Montclair State University's mascot).

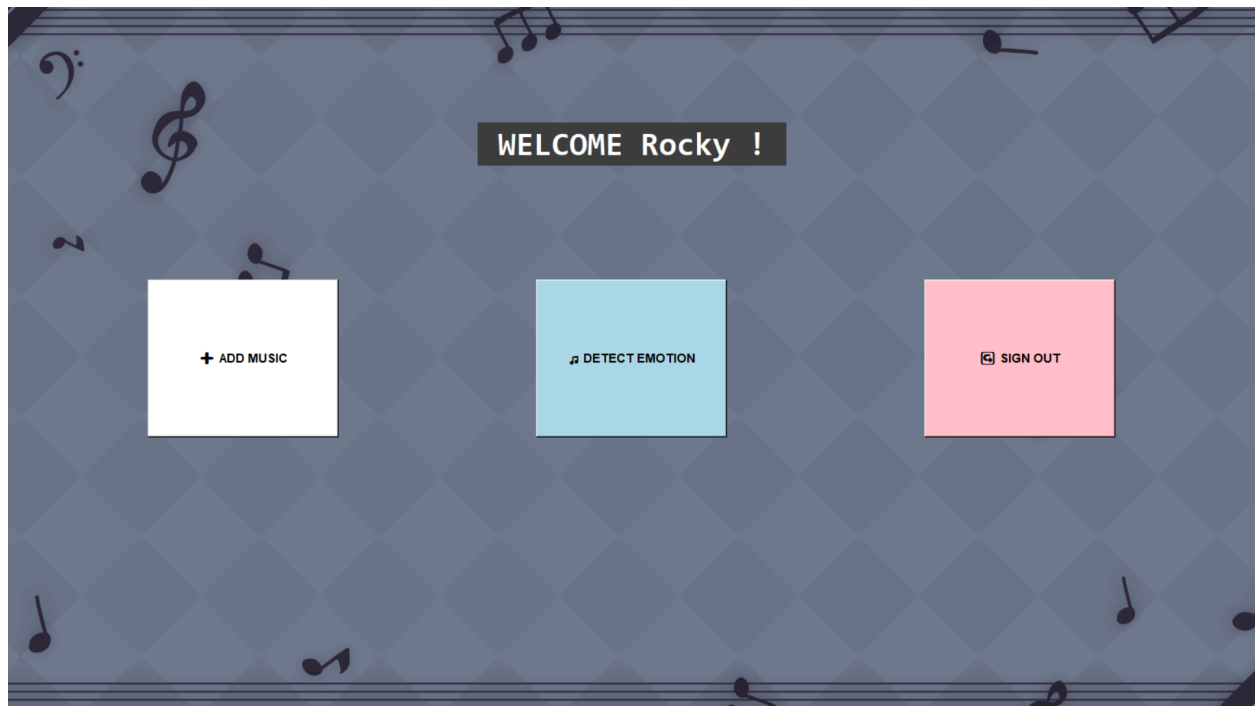
From this screen the user will be able to:

1. Navigate to **Screen 4: Add Music** (**Figure 6.1.4a**) to add their own music files for the application to play.
2. Run the emotion detection music player (navigate to **Screen 5: Emotion Detection** as shown in **Figure 6.1.5**) to detect their current emotion and play an appropriate song based on the detected emotion.
3. Sign out and navigate back to **Screen 1: Sign In** (**Figure 6.1.1a**) screen.

Add Music button ⇒ **Screen 4: Add Music**

Detect Emotion button ⇒ **Screen 5: Emotion Detection**

Sign Out button ⇒ **Screen 1: Sign In**



**Figure 6.1.3 Screen 3: Welcome <User>!**

- **6.1.4 Screen 4: Add Music**

From this screen the user will be able to:

1. Add their own music files to the application's music directories which are organized by emotion. The user must select which emotion's music directory they want to add their own files to first.
2. Return to **Screen 3: Welcome <User> !** (*Figure 6.1.3*).
3. Sign Out and navigate back to **Screen 1: Sign In** (*Figure 6.1.1a*).

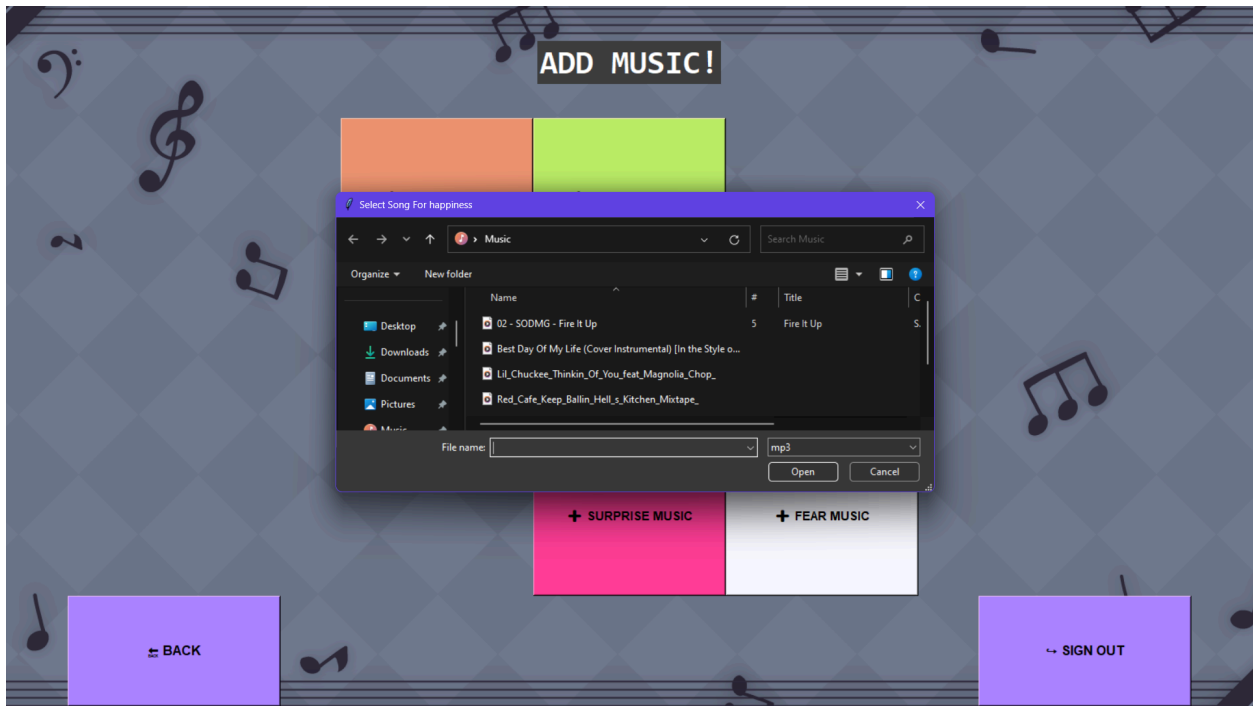
Emotion button ⇒ Opens dialog box to select music to add to Emotion directory (*Figure 6.1.4b*)

Back button ⇒ **Screen 3: Welcome <User>!**

Sign Out button ⇒ **Screen 1: Sign In**



*Figure 6.1.4a Screen 4: Add Music*



**Figure 6.1.4b Screen 4: Add Music - Dialog Box pop up to Select Song for happiness**



**Figure 6.1.4c Screen 4: Add Music - Dialog Box pop up for successfully adding happiness music to the happiness music directory**

- **6.1.5 Screen 5: Emotion Detection**

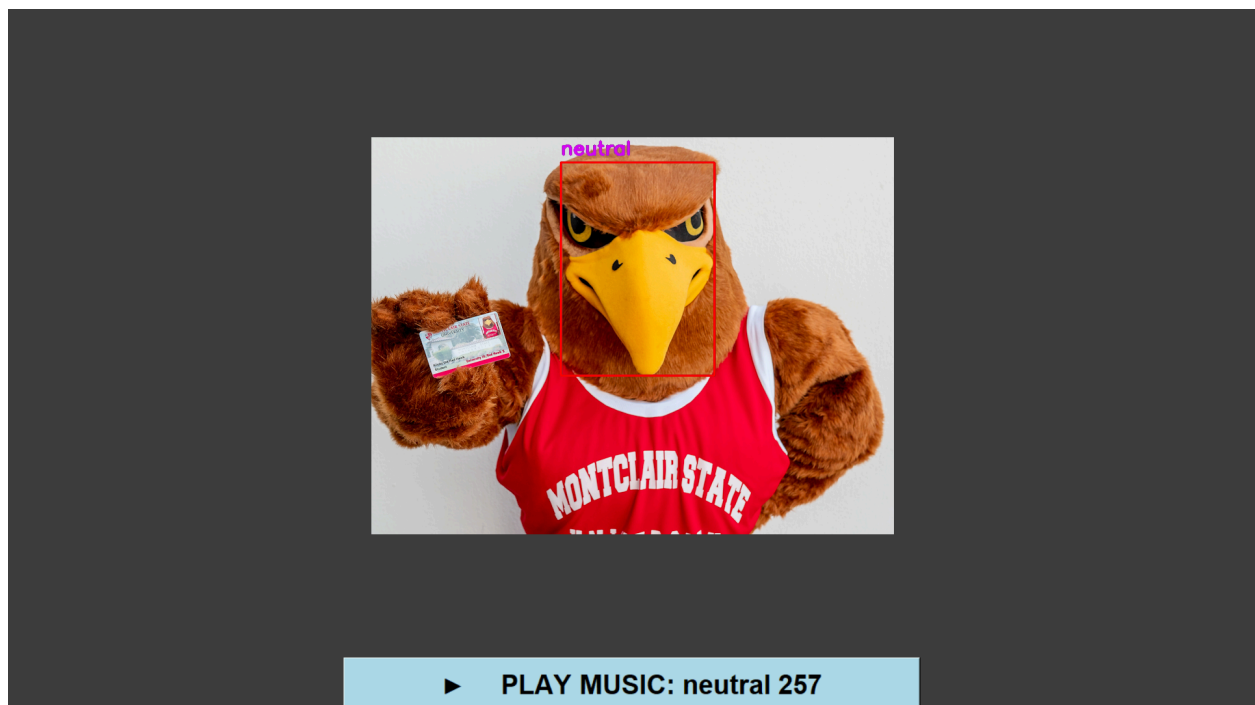
This screen shows a live camera feed with real time facial recognition and emotion detection. Human faces that have been successfully recognized by the system are shown in the live camera feed with a red bounding box over them. The current detected emotion is shown with magenta colored text, above the top left corner of the red bounding box.

A play music button at the bottom of the screen displays the current detected emotion in real time, along with a total number count of how many times the displayed emotion has been detected so far within the current emotion detection session. When the user clicks this play music button, it ends the current emotion detection session. Then the emotion detection count is used to decide which emotion music directory should be selected to play music from in **Screen 6: Music Player (Figure 6.1.6)** based on the emotion detection.

**Figure 6.1.5** illustrates an example of this screen using Montclair State University's mascot Rocky. Rocky is considered to have a human face for the purpose of illustrating this example.

From this screen the user will be able to:

1. Use the emotion detection feature to detect their current emotion
2. Navigate to **Screen 6: Music Player (Figure 6.1.6)** to play music based on their current detected emotion.



**Figure 6.1.5 Screen 5: Emotion Detection**

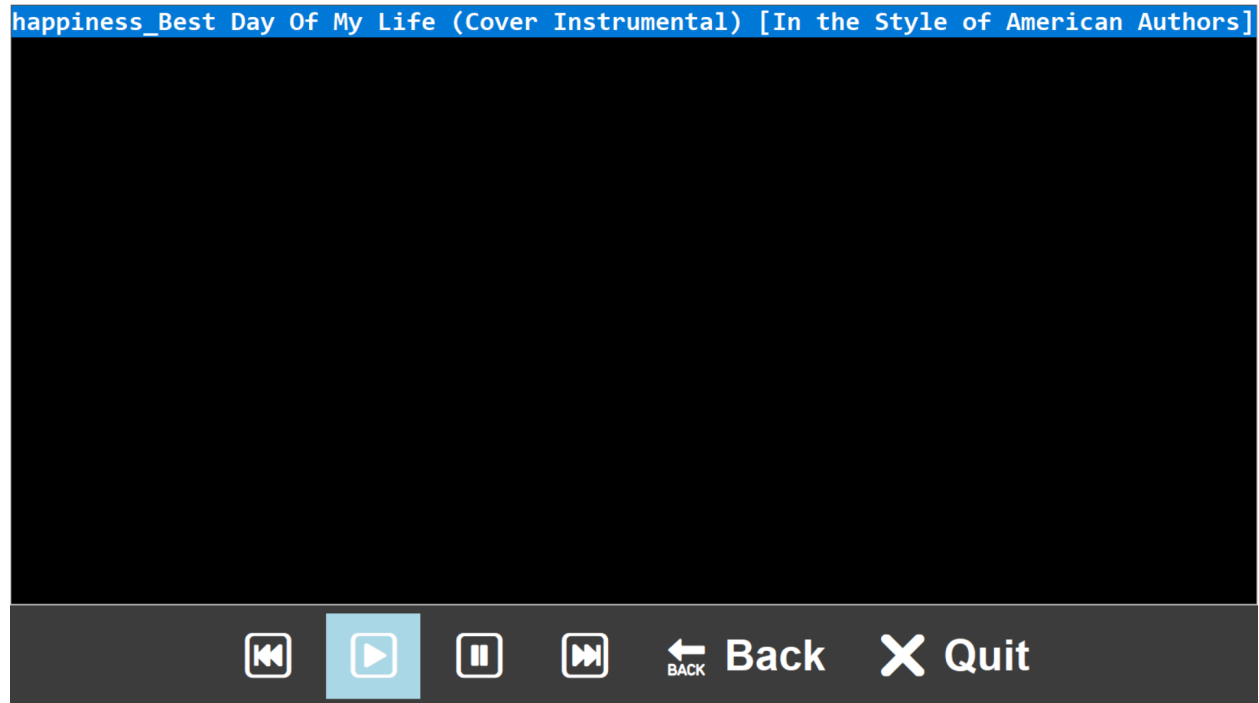
- **6.1.6 Screen 6: Music Player**

This screen is the system's built-in music player which can only be accessed after Emotion Detection is completed in **Screen 5: Emotion Detection** (*Figure 6.1.5*).

Based on the emotion detected, the music player launches with the respective emotion music directory and automatically plays the first song in that directory. The user can interact with the music through the music player by clicking on the play, pause, next song, previous song buttons.

Back button ⇒ **Screen 5: Emotion Detection**

Quit button ⇒ Exit the application



*Figure 6.1.6 Screen 6: Music Player*



## 6.2 External Interface Requirements

### Software Interface

The application is compatible with Windows 10 or higher operating systems. The base application will require a minimum disk space of five hundred megabytes (500 MB) and a RAM capacity of at least 16 GB or higher. Default music files will be provided in the base application's music directory, but users will also be able to add their own music files via the Add Music screen as described in **Section 6.1.4**.

### Hardware Interface

The application's emotion detection feature will require a webcam or external camera to be connected and available on the user's system.

### Communication Interface

The application's emotion detection feature will use Microsoft's FER+ machine learning model to identify emotions. This trained machine learning model with an existing dataset. The user will be able to communicate their current emotions to the application by following the gestures and facial expressions shown in **Figure 6.2.1** below.



**Figure 6.2.1 FER+ Dataset Examples ([github.com/Microsoft/FERPlus](https://github.com/Microsoft/FERPlus))**

## 7. Security Design

### 7.1 Security Design

This section describes the security measures and mechanisms implemented in the software system to protect against potential threats and vulnerabilities. It describes a secure and controlled user interaction within the system. The measures taken are described in the sections below.

### 7.2 User Authentication

The system will verify the identity of its users attempting to access the software using the sign in and sign up pages. This identity verification is done by ensuring that every user signs up with a unique username and strong password. A valid username and password combination is then required to sign in to the system. Users cannot access the emotion detection application without signing in. Implementing strong authentication mechanisms will help ensure that only authorized users can access the system.

## 7.3 User Authorization

Only authenticated users, as in users that have entered a valid username and password combination at the time of sign in, will be authorized to access the functionalities within the application.

## 7.4 Password Hashing

When users sign-up within the application system, the username and password combinations are stored locally on their device within an embedded database in the system. The passwords are stored in this database as hashed passwords. This adds an extra layer of protection against vulnerabilities that could expose sensitive password information to unauthorized parties.

# 8. Deployment and Operations

## 8.1 Deployment and Operations

Deploying and maintaining the software system is a very essential phase in the process of software development. It involves several architectural and operational considerations to ensure efficient management.

Here is a description of the deployment architecture and operational considerations for deploying and maintaining the application system:

## 8.2 Version control system

GitHub is used to store and manage the code for the software product. A repository has been created. All code contributors pull from the main branch and git checkout -b to a new branch, where they can make their changes.

Two main branches have been created: the main branch and the development branch. The main branch will consist of the final code that will go live (production). The development branch will be for in-house, to see if any improvement can be made before it goes live.

During development here are the processes the developers will go through to reduce merge conflict in the code:

- “git pull dev” the code from development at all time to get the most update code
- “git checkout -b” to branch out of the dev
- Make the changes
- “git add .” to add
- “git commit -m “ To write the commit message so other developers will understand what update is being added to the code
- “git push origin the current branch” To push the branch to the repository
- Then a pull request will be done on the github repository to merge the developers changes to dev-branch.

## 8.3 Continuous Integration and Continuous Deployment (CI/CD)

- The development team aims to implement CI/CD pipelines to automate the build, test it, and deploy processes.
- GitHub actions will be used to trigger tests automatically whenever code is pushed to the repository.
- Automated deployment to production will be utilized after successful testing to ensure rapid releases.

## 8.4 Implementation Document

All instructions are written in the README.md file in the code base. Additionally, all functional code has been updated with comments where necessary, so that the code files are easily readable.

# 9. Design Summary and Conclusion

## 9.1 Summary

This Software Design Document is intended for use by all stakeholders of this project. The purpose of this document is to provide a top down description of the design and architecture for an Emotion Detection and Music Player system that has been developed by Team SKYN.

This system is a desktop application that lets users detect their current emotion through facial recognition, and then play music on the application's music player, based on the detected emotion. The system supports the detection of the following seven emotions: neutral, happiness, surprise, sadness, anger, disgust, and fear.

The intended user and target demographic for the overall product will mainly be the university students, faculty, and staff members of Montclair State University. First access and consideration will be given to the students and educators in the CSIT 515 Software Engineering course of Spring 2024. However, the final software system's usage will not be limited to this demographic as it will be publicly available on GitHub to all stakeholders of the system, including stakeholders that do not fit into the originally intended demographic.

The application contains six main screens which are the SignUp/SignIn, Welcome, Add Music, Emotion Detection, and Music Player screens. Music files will be stored within the application's file directories with each of the seven supported emotions having its own directory. Users will be able to add their own music files to these directories through the application interface.

# **SOFTWARE USER MANUAL**

# 1. Introduction

## 1.1 Introduction

This document will provide a user guide on how to install and use the **Emotion Detection and Music Player** software system that has been developed by Team SKYN.

This software system consists of facial recognition and emotion detection functionalities and a built-in music player.

This system is a desktop application that uses its facial recognition and emotion detection functionalities to detect the user's current emotion. The application will automatically play music via its built-in music player, based on the user's detected emotion. This music will match the user's detected emotion.

The system supports the detection of the following seven emotions: neutral, happiness, surprise, sadness, anger, disgust, and fear.

## 2. Quick Start Guide

### 2.1 Install Python

The user's device will require Python 3.11.8 to be installed in order to run the application files.

This can be obtained from the following source:

**<https://www.python.org/downloads/release/python-3118/>**

After navigating to the URL above via a compatible web browser, scroll down to the bottom of the web page under the **Files** section as shown in **Figure 2.1.1a**.

Files						
Version	Operating System	Description	MD5 Sum	File Size	GPG	Sigstore
<a href="#">Gzipped source tarball</a>	Source release		7fb0bfaa2f6aae4aadcd51abe957825	25.3 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">XZ compressed source tarball</a>	Source release		b353b8433e560e1af2b130f56dfbd973	19.1 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">macOS 64-bit universal2 installer</a>	macOS	for macOS 10.9 and later	0903e86fd2c61ef761c94cb226e9e72e	42.7 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">Windows installer (64-bit)</a>	Windows	Recommended	77d17044fd0de05e6f2cf4f90e87a0a2	24.9 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">Windows installer (ARM64)</a>	Windows	Experimental	ae1b38fa57409d9a0088a031f59ba625	24.2 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">Windows embeddable package (64-bit)</a>	Windows		9199879fbad4884ed93ddf77e8764920	10.7 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">Windows embeddable package (32-bit)</a>	Windows		104bf63ef10c06298024a61676a11754	9.6 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">Windows embeddable package (ARM64)</a>	Windows		6b989558c662f877e2e707d640673877	10.0 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>
<a href="#">Windows installer (32-bit)</a>	Windows		45d4b29f26ca02b1ccf13451ea136654	23.7 MB	<a href="#">SIG</a>	<a href="#">.sigstore</a>

**Figure 2.1.1a Files for Python 3.11.8 download**

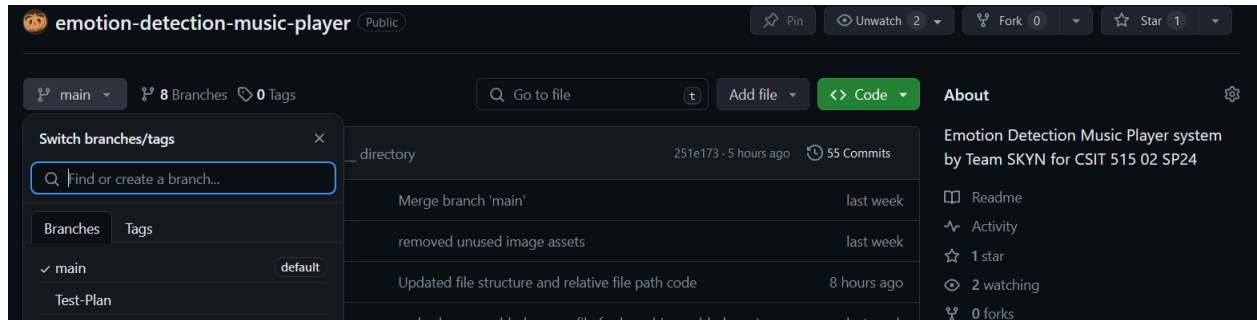
Select the **Windows Installer (64-bit)** file or select whichever file (from the list in **Figure 2.1.1a**) is compatible with the Operating System being used in the desktop device. Follow the instructions from the Python Installer to successfully install Python.

## 2.2 Download Application (GitHub version)

The Emotion Detection and Music Player System, developed by Team SKYN, can be downloaded from the following GitHub repository:

**<https://github.com/smeraldoflower/emotion-detection-music-player>**

After navigating to the URL above via a compatible web browser, first ensure that the **main branch** is selected as shown in **Figure 2.2.1** Then scroll down the web page and follow the instructions in the README file to download the application. Once the application is downloaded follow instructions in **Section 2.4 Launch Application**.



**Figure 2.2.1 Select the main branch on the GitHub repository**

## 2.3 Install Application (Physical copy version)

Some users have received a physical copy of the Emotion Detection and Music Player System, developed by Team SKYN. The physical copy version is stored in a USB flash drive.

To install this version of the application, first connect the USB flash drive via a USB port on the desktop device where the application is to be installed.

If the device does not have any matching USB ports, please follow the steps to downloading and installing the GitHub version described in **Section 2.2 Download Application (GitHub Version)** above.

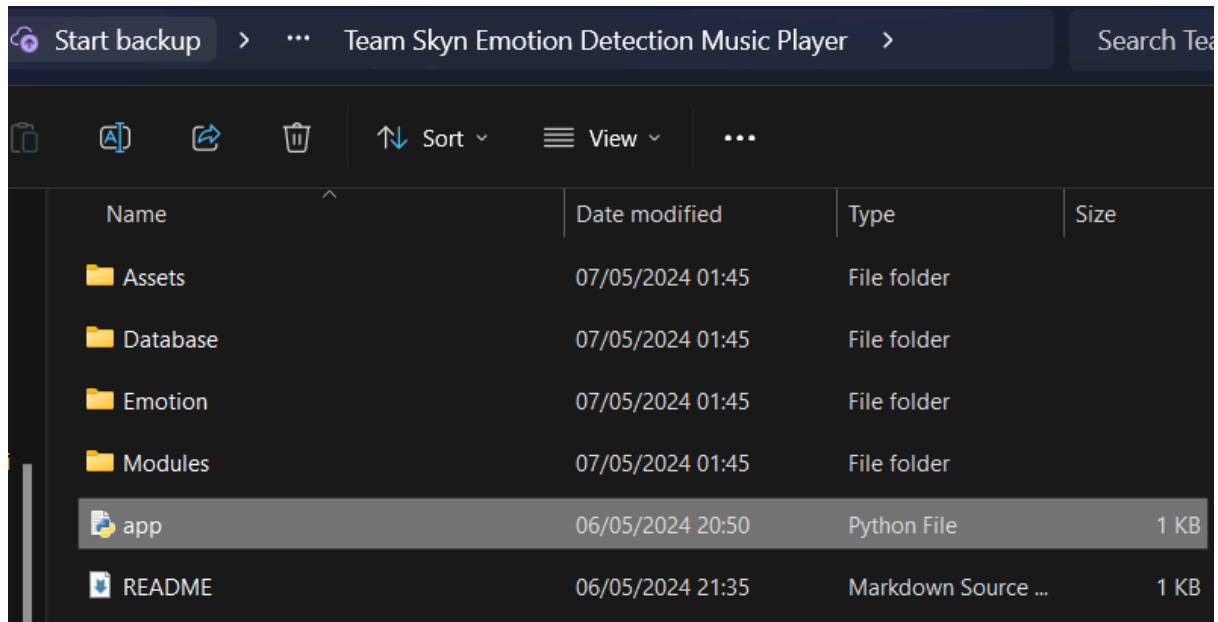
Once the USB flash drive is successfully connected to the device. Copy the folder named **“Team SKYN Emotion Detection Music Player”** to the desired installation location on the device.

Disconnect the USB flash drive from the desktop device once this folder has been copied successfully.

## 2.4 Launch Application

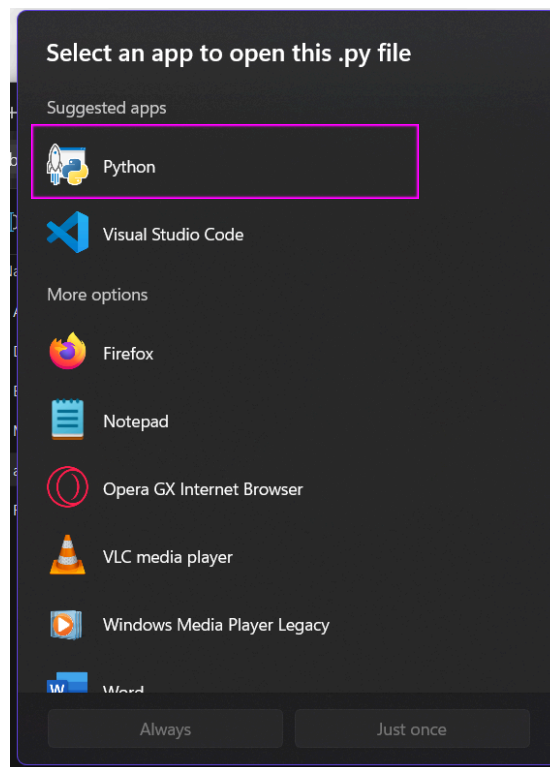
After the application is successfully installed, in the desired location, on the desktop device, open the folder named **“Team SKYN Emotion Detection Music Player”** from the location where it was installed.

Then select the file named **“app” (app.py)** within this folder and double click it as shown in **Figure 2.4.1a**



**Figure 2.4.1a Select the main branch on the GitHub repository**

When the application is launched for the first time, the system may prompt the user to select an appropriate launching application for this program as shown in **Figure 2.4.1b**. In this case the user should select Python and click the “Always” button on the dialogue box.



**Figure 2.4.1b Select an Python to open the app.py file**

The application has launched successfully if the Sign In screen can be seen, as described in **Section 3 User Interface**.

## 3. User Interface

### 3.1 User Interface Flow

This application consists of six user screens as follows:

- **Screen 1: Sign In**
- **Screen 2: Sign Up**
- **Screen 3: Welcome <User> !**
- **Screen 4: Add Music**
- **Screen 5: Emotion Detection**
- **Screen 6: Music Player**

- **3.1.1 Screen 1: Sign In**

This will be the first screen that will be visible to the user when the application is launched. The application will launch in full screen mode by default.

From this screen the user will be able to:

1. Sign in to the application with an existing and valid username and password combination.
2. Navigate to **Screen 2: Sign Up (Figure 3.1.1a)** to sign up for the system.
3. Exit the application.

Existing users can use this screen to sign in to the main application.

Below are the steps to follow to successfully sign in:

**Step 1: Username and Password:** Existing users should provide their user credentials in the username and password field and click on the **Sign in** button to get authenticated.

**Step 2: Sign Up Button:** New Users should click this button to navigate to the **Sign Up** page for account creation. Follow the steps for **Sign Up** in **Section 3.1.2 Screen 2: Sign Up**

**Step 3: Quit Button:** Clicking this button terminates the application.

Sign In button ⇒ validate username and password ⇒ **Screen 3: Welcome <User>!**

Sign Up button ⇒ **Screen 2: SignUp Screen**

Quit button ⇒ Exit the application





***Figure 3.1.1a Screen 1: Sign In***



***Figure 3.1.1b Screen 1: Sign In - Invalid username or password error shown***

- **3.1.2 Screen 2: Sign Up**

From this screen (*Figure 3.1.2a*) the user can

1. Sign up for the system using a valid username and password combination.
2. Navigate back **Screen 1: Sign In** (*Figure 3.1.1a*) after successful sign up
3. Exit the application.

An account creation is required before this software's main functionalities can be used. Below are the steps to follow for successful sign up:

**Step 1: Create Username:** Enter a username between 4 to 10 alphanumeric characters only in the username field. The username has to be unique and must begin with a letter.

**Step 2: Create Password:** Enter a password in the password field. The password has a requirement of eight characters or above and should contain no spaces.

**Step 3: Re-enter Password:** Re-Enter the same password as in the create password field to confirm password for verification.

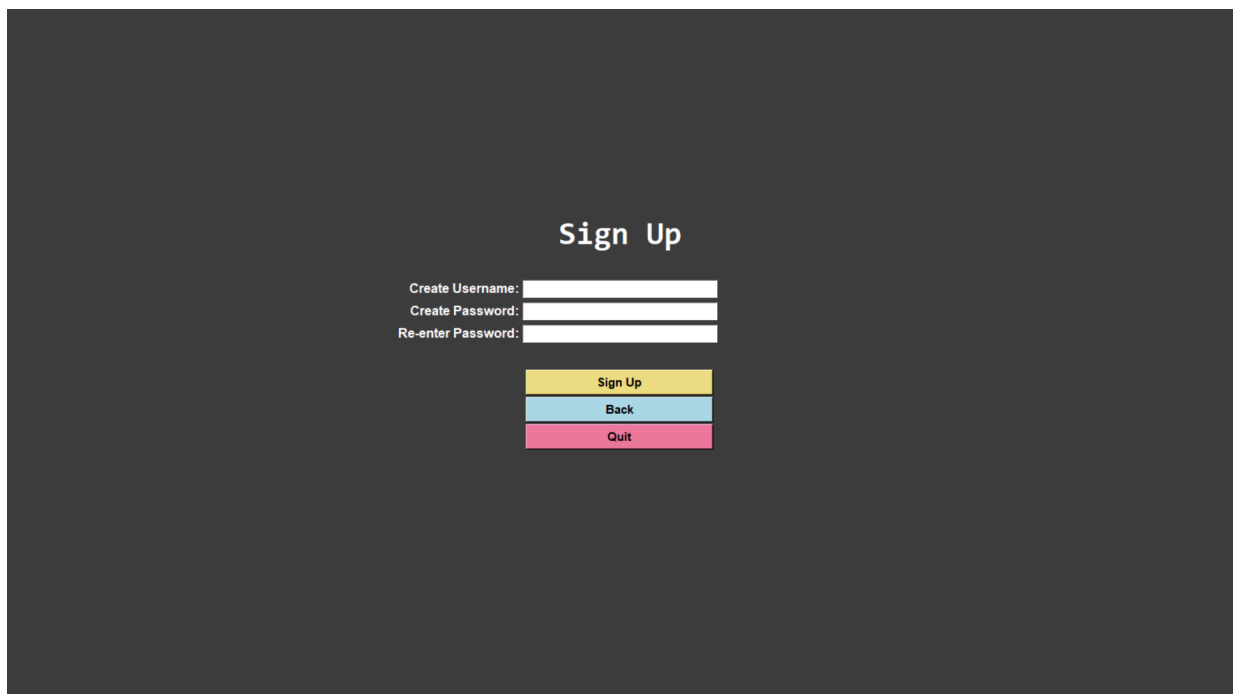
**Step 4: Sign Up Button:** On click, there username and password requirements are checked and if all requirements are met, the user will be routed to **Screen 1: Sign In** page.

**Step 4.1: Unsuccessful Sign Up:** If requirements are not met follow the instructions in the error messages as shown in *Figures 3.1.2b, 3.1.2c, 3.1.2d*

Sign Up button ⇒ returns to **Screen 1: Sign In** on successful Sign Up

Back button ⇒ returns to **Screen 1: Sign In** without signing up

Quit button ⇒ Exit the application

The image shows a 'Sign Up' screen with a dark gray background. At the top center, the title 'Sign Up' is displayed in white. Below the title, there are three white input fields with labels to their left: 'Create Username:', 'Create Password:', and 'Re-enter Password:'. Each label is followed by a white rectangular input field. Below these fields, there are three buttons stacked vertically. The top button is yellow and labeled 'Sign Up'. The middle button is light blue and labeled 'Back'. The bottom button is pink and labeled 'Quit'.

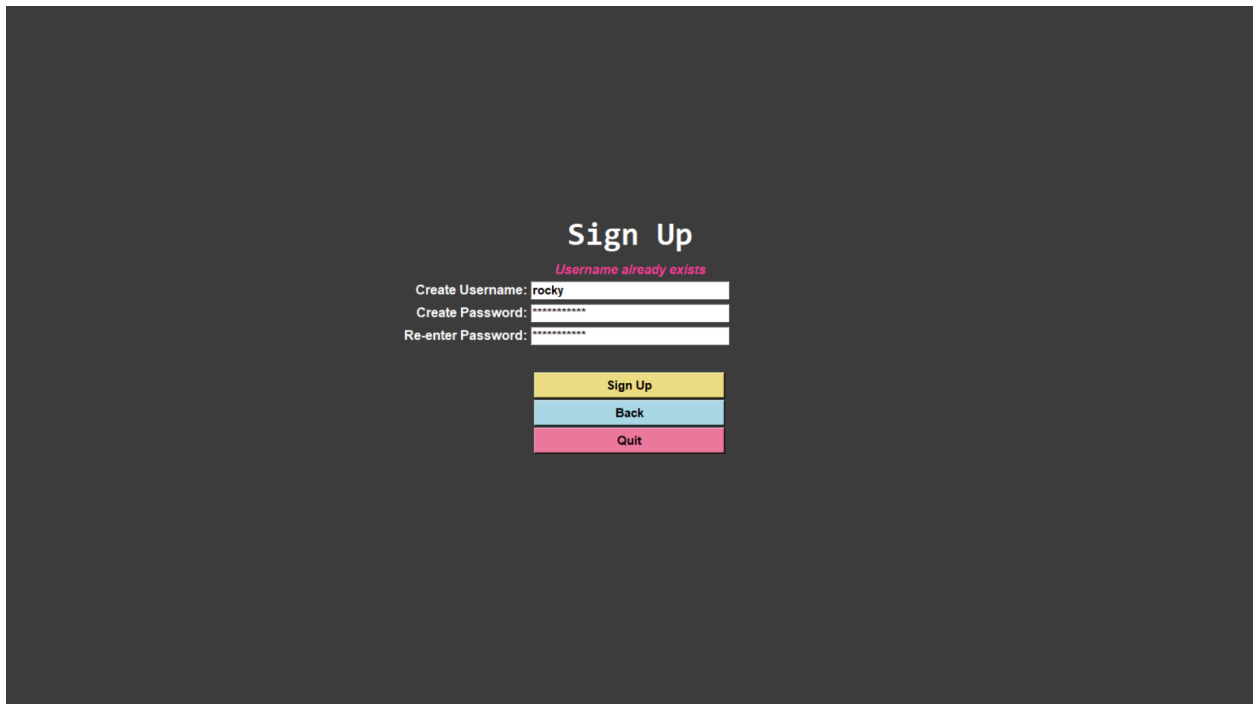
**Figure 3.1.2a Screen 2: Sign Up**

The image shows a 'Sign Up' form on a dark grey background. The form has three input fields: 'Create Username:', 'Create Password:', and 'Re-enter Password:'. Above the 'Create Password:' field, the text 'Invalid username or password' is displayed in red. Below the input fields are three buttons: 'Sign Up' (yellow), 'Back' (light blue), and 'Quit' (pink).

**Figure 3.1.2b Screen 2: Sign Up - Invalid username or password error shown - Enter a valid username and valid password.**

The image shows the same 'Sign Up' form as in Figure 3.1.2b. In this version, the 'Create Username:' field contains the text 'passwordmismatchtest'. Above the 'Create Password:' field, the text 'Password mismatch' is displayed in red. The 'Create Password:' and 'Re-enter Password:' fields are filled with asterisks. Below the input fields are the same three buttons: 'Sign Up' (yellow), 'Back' (light blue), and 'Quit' (pink).

**Figure 3.1.2c Screen 2: Sign Up - Password mismatch error shown - Enter matching passwords**



**Figure 3.1.2d Screen 2: Sign Up - Username already exists error shown - Select and Enter a different username.**

- **3.1.3 Screen 3: Welcome <User>!**

The user can only access this screen after successfully signing into the application using a valid username and password combination.

This welcome screen displays the current signed-in user's username as shown in **Figure 3.1.3** with the example of Rocky (the name of Montclair State University's mascot).

From this screen the user will be able to:

1. Navigate to **Screen 4: Add Music (Figure 3.1.4a)** to add their own music files for the application to play.
2. Run the emotion detection music player (navigate to **Screen 5: Emotion Detection** as shown in **Figure 3.1.5**) to detect their current emotion and play an appropriate song based on the detected emotion.
3. Sign out and navigate back to **Screen 1: Sign In (Figure 3.1.1a)** screen.

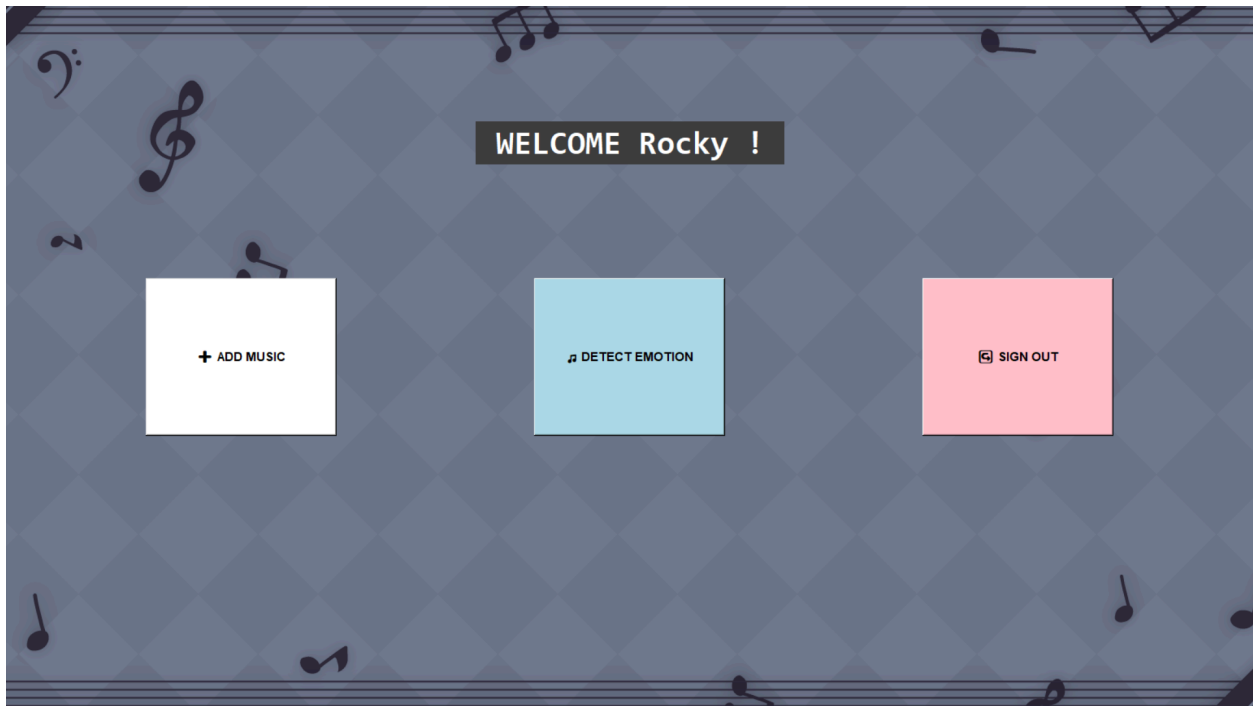
This screen has three buttons:

1. **Add Music Button:** On click, the user is routed to the "Add Music" screen
2. **Emotion Detection button:** On click, the user is routed to the Emotion Detection screen which launches their camera for facial recognition and emotion detection.
3. **Sign Out button:** Clicking on this button signs the user out of the application and routes the user back to the **Sign In** page.

Add Music button ⇒ **Screen 4: Add Music**

Detect Emotion button ⇒ **Screen 5: Emotion Detection**

Sign Out button ⇒ **Screen 1: Sign In**



**Figure 3.1.3 Screen 3: Welcome <User>!**

#### • 3.1.4 Screen 4: Add Music

From this screen the user will be able to:

1. Add their own music files to the application's music directories which are organized by emotion. The user must select which emotion's music directory they want to add their own files to first.
2. Return to **Screen 3: Welcome <User> ! (Figure 3.1.3)**.
3. Sign Out and navigate back to **Screen 1: Sign In (Figure 3.1.1a)**.

This screen allows the user to add music into a specific music emotion playlist matching one of the seven moods in the scope of the system.

Follow the instructions below to successfully add music to an emotion playlist:

**Add Emotion Music Buttons:** Clicking on these buttons will open a file dialog which will enable the user to go to any directory to select a song they want to add the specific playlist. The name of the buttons correspond to one of the emotions this system supports. (See **Figure 3.1.4a, 3.1.4b, 3.1.4c**)

**Back Button:** On click, the user is routed to the **Welcome** page.

**Quit Button:** On click, application is terminated.

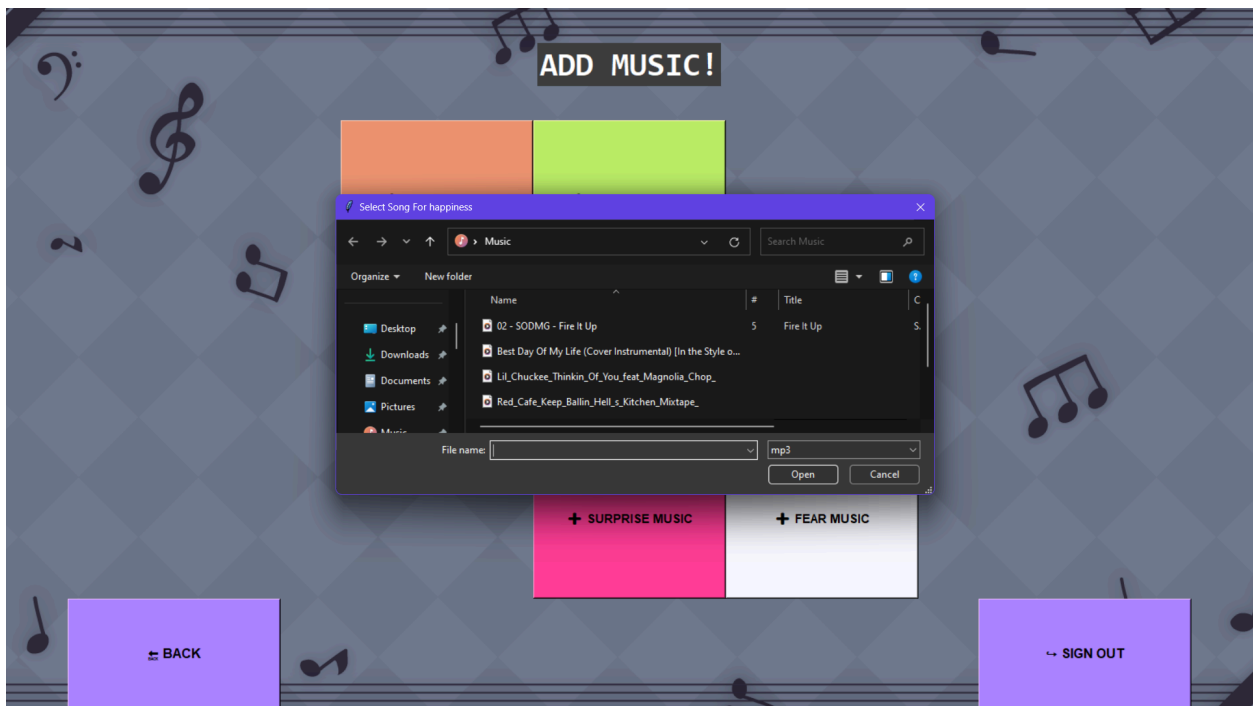
Emotion button ⇒ Opens dialog box to select music to add to Emotion directory (**Figure 3.1.4b**)

Back button ⇒ **Screen 3: Welcome <User>!**

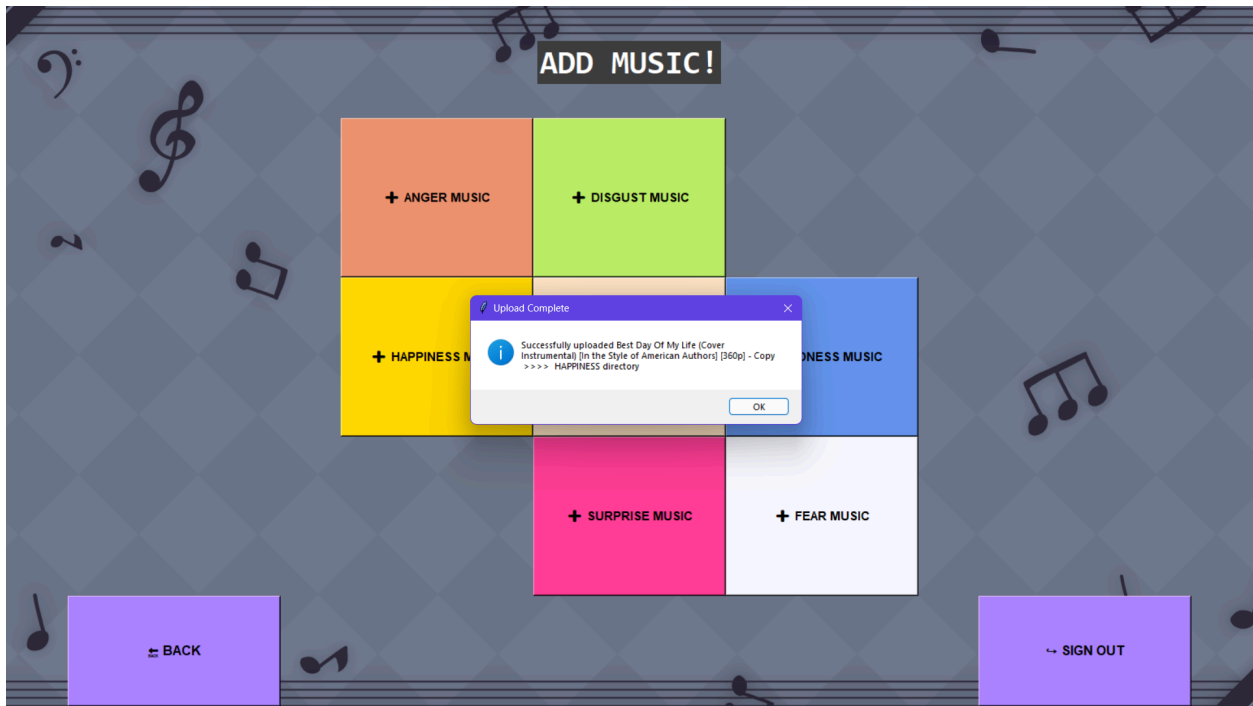
Sign Out button ⇒ **Screen 1: Sign In**



**Figure 3.1.4a Screen 4: Add Music**



**Figure 3.1.4b Screen 4: Add Music - Dialog Box pop up to Select Song for happiness**



**Figure 3.1.4c Screen 4: Add Music - Dialog Box pop up for successfully adding happiness music to the happiness music directory**

### • 3.1.5 Screen 5: Emotion Detection

This screen shows a live camera feed with real time facial recognition and emotion detection. Human faces that have been successfully recognized by the system are shown in the live camera feed with a red bounding box over them. The current detected emotion is shown with magenta colored text, above the top left corner of the red bounding box.

A play music button at the bottom of the screen displays the current detected emotion in real time, along with a total number count of how many times the displayed emotion has been detected so far within the current emotion detection session. When the user clicks this play music button, it ends the current emotion detection session. Then the emotion detection count is used to decide which emotion music directory should be selected to play music from in **Screen 6: Music Player (Figure 3.1.6)** based on the emotion detection.

**Figure 3.1.5** illustrates an example of this screen using Montclair State University's mascot Rocky. Rocky is considered to have a human face for the purpose of illustrating this example.

From this screen the user will be able to:

1. Use the emotion detection feature to detect their current emotion
2. Navigate to **Screen 6: Music Player (Figure 3.1.6)** to play music based on their current detected emotion.

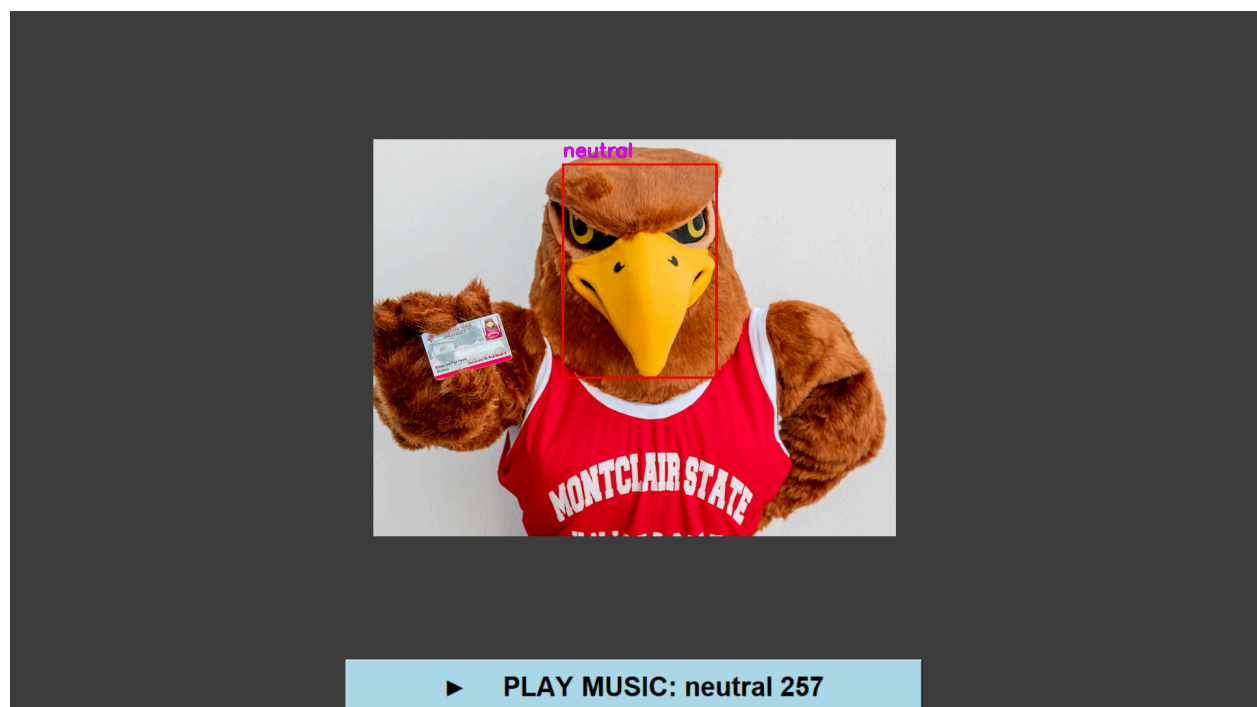
Follow these steps for successful Emotion Detection and Music Playing:

**Step 1:** Ensure camera is enabled

**Step 2:** Emotions will be automatically detected if there is a human face in the camera's view. There will be a red bounding box drawn on the detected human face of the user, with the user's current emotion shown above this box in magenta colored font.

**Step 3:** Imitate the gestures and facial expressions shown in **Figure 3.2.1** for better and desired emotion detection results. Hold the expression for at least 30s to allow the emotion to become the dominant emotion in the current emotion detection session.

**Step 4: Play Music Button:** On click, the most dominant emotion detected from the facial expression will be chosen and a music player will open with a playlist of music matching the detected emotion.



**Figure 3.1.5 Screen 5: Emotion Detection**







- **3.1.6 Screen 6: Music Player**

This screen (**Figure 3.1.6**) is the system's built-in music player which can only be accessed after Emotion Detection is completed in **Screen 5: Emotion Detection (Figure 3.1.5)**.

Based on the emotion detected, the music player launches with the respective emotion music directory and automatically plays the first song in that directory. The user can interact with the music through the music player by clicking on the play, pause, next song, previous song buttons.

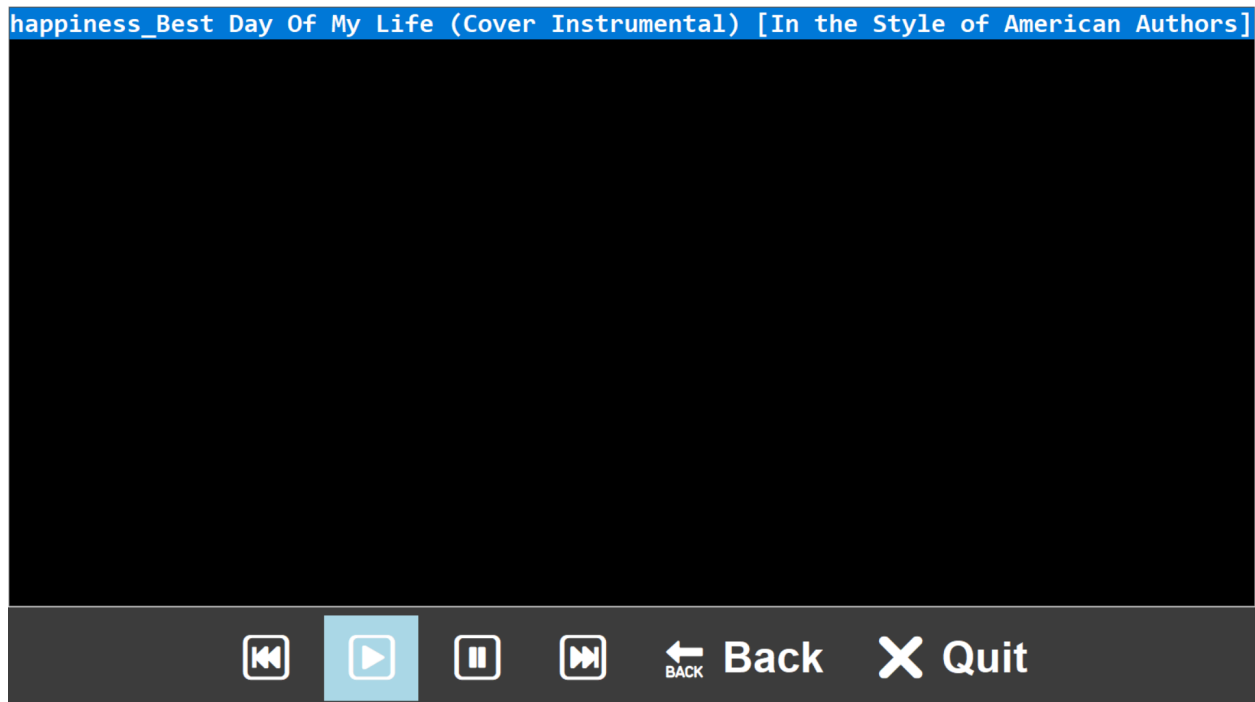
The following steps will guide the user through this process:



1. Music from the playlist corresponding to the detected emotion will automatically play when this page opens.
2. **Control Buttons:**
  - a.  **Pause Button** will pause the music.
  - b.  **Play Button** will play the selected music from the playlist;
  - c.  **Next Button** will skip the current song to the next song.
  - d.  **Previous Button** will go to the previous song.
  - e.  **Back Button:** On click, the user is routed back to the **Emotion Detection** screen.
  - f.  **Quit Button:** This button terminates the application.

Back button ⇒ **Screen 5: Emotion Detection**

Quit button ⇒ Exit the application



*Figure 3.1.6 Screen 6: Music Player*

## 3.2 External Interface Requirements

### Software Interface

The application is compatible with Windows 10 or higher operating systems. The base application will require a minimum disk space of five hundred megabytes (500 MB) and a RAM capacity of at least 16 GB or higher. Python 3.11.8 is required to run this application.

Default music files will be provided in the base application's music directory, but users will also be able to add their own music files via the Add Music screen as described in **Section 3.1.4**.

### Hardware Interface

The application's emotion detection feature will require a webcam or external camera to be connected and available on the user's system.

### Communication Interface

The application's emotion detection feature will use Microsoft's FER+ machine learning model to identify emotions. This trained machine learning model with an existing dataset. The user will be able to communicate their current emotions to the application by following the gestures and facial expressions shown in **Figure 3.2.1** below.



*Figure 3.2.1 FER+ Dataset Examples ([github.com/Microsoft /FERPlus](https://github.com/Microsoft/FERPlus))*

## 4. Contact Team SKYN

### 4.1 Feedback, Suggestions, and Support

The user can contact the members of Team SKYN directly via GitHub to leave feedback and suggestions or seek technical support related to this software product.

At least one member of Team SKYN will respond within three business days.

**Addo, Kwame:** [github.com/addo12kwame](https://github.com/addo12kwame)

**Hassan, Yewande:** [github.com/yewande-hassan](https://github.com/yewande-hassan)

**Mahmood, Nusaiba:** [github.com/smeraldoflower](https://github.com/smeraldoflower)

**Sapp, Shantia:** [github.com/ET-ElegantThug](https://github.com/ET-ElegantThug)

# **SOFTWARE TEST DOCUMENT**

# 1. Introduction

## 1.1 Introduction

This document is intended for testing the Emotion Detection and Music Player software system that has been developed by Team SKYN.

The purpose of testing is to check the performance of the system and reduce project risks. Testing ensures that the final software product is robust, reliable, efficient and meets the requirements defined in the **Software Specification Document (Pg 4)**.

Emotion-detection and music player system is designed to detect seven major facial emotions and music would be played based on the emotion detected. The system will be tested for the sign-in and sign-out functionalities, emotion recognition and the music player.

## 1.2 Abstract

The test document serves as the basis of our system testing strategies and test implementations. The test plan will verify the accuracy of facial recognition software, reliability of emotion detection algorithms, and quality of our user interface, and the systems overall ability to perform.

## 1.3 Intended Type of Test to Perform

- **1.3.1 Unit Testing**

Testers will use this type of testing to test the different modules(sign in, signup, welcome page, music player, emotion detection) that makes up the software application.

- **1.3.2 Integration Testing**

Tester will start integration testing after the application is bundled and package the app.

- **1.3.3 Function Testing**

Tester will compare the functionality expected in the requirement specification to ensure that every test case passes the check.

## 1.4 Environment and Pretest Background

**Hardware Environment:** Computer(MacOS/Windows), Sufficient CPU/memory, Camera for facial/emotion detection

**Software Environment :** User interface, Facial recognition, Emotion detection, Music player

## 2. Test Plan

### 2.1 Test Plan

This testing is integration testing, that is software modules are tested in a group, it is also smoke testing, and will be conducted in a **"Black box"** type, that is without access to the source code.

During testing, it is planned to check the implementation of the main functions of the application, conduct positive and negative tests, as well as check non-functional requirements. The main functions include the following items:

- Ability to sign in and sign out of the system
- Ability to detect facial emotions within the scope of the system
- Ability to play music based on the emotion detected by the system
- Ability to quit the application and switch between pages on the application

### 2.2 Functional Requirements

The following functional requirements are to be tested.

- **2.2.1 Ability to sign in and sign out of the system**

This use case should be tested on:

1. The application should check the validation of each users that wants access to the system, thereby ensuring that they have the right sign in details which is a username not less than 10 characters and a password
2. Users can sign up if they are new users, after which they will be directed to the sign in page
3. Prospective users without a valid sign in details should not be authenticated

- **2.2.2 Ability to detect facial emotions within the scope of the system**

This use case should be tested on:

1. The application should only detect a face, hence check if it is a face and not an object
2. The application should only detect a face at a time not two faces
3. The application should be able to detect all seven emotions within the scope of the application

- **2.2.3 Ability to play music based on the emotion detected by the system**

This use case should be tested on:

1. The application should play a music from the specific emotion detected music directory
2. Users should be able to pause, switch to the next or previous audio music in the emotion directory in question

- **2.2.4 Ability to quit the application and switch between pages on the application**

This use case should be tested on:

1. User should be able to to quit the application from any page on the application
2. Users should be able to switch between the pages on the application, there should be a functional back and next button on the application

## **2.3 Testing (id location)**

Testing is to be done within the following locations:

**Virtual environment:** Visual Studio Code IDE

**Physical environment:** Montclair State University on-campus, off-campus with remote access.

## **2.4 Schedule**

Continuous testing will take three weeks to detect sufficient bugs in the system and ensure that after testing the system is robust.

## **2.5 Resource Requirements**

Successful testing for this system will require several resources within the following resource categories:

- **Hardware/Software:** to run the software application system and tests
- **Personnel:** Team SKYN, and Beta Testing User volunteers
- **Testing Materials**
- **Test Training**

## **2.6 Testing Approaches**

The following testing approaches have been adopted for this project:

The conducted testing can be outstripped as:

- By the degree of automation: manual.
- According to the knowledge of the system: black box
- By the degree of isolation of components: integration

## 3. Results

### 3.1 Test Cases

When the test effort is complete, document results, identify any discrepancies that were handled.

Script	Action	Expected Result	Actual Result	Degree of Completeness
Different users sign up with the same username	Filling the username input	An error message "Username already exists"	An error message "Username already exists"	Pass
Sign up with username < 10 characters	Filling the username input with <10	An error message "Invalid username"	An error message "Invalid username and password"	Fail
Password not matching confirm password during sign up	Mismatching password and confirm password	An error message "Password mismatch"	An error message "Password mismatch"	Pass
Sign up with password < 8 characters	Filling the username input with < 8	An error message "Invalid password"	An error message "Invalid username and password"	Fail
Camera detection of objects	Camera detecting an object instead of a face	It should not detect any object, the camera should only detect a face	Camera only detects a face, if an object is placed in front of the camera it doesn't detect it	Pass

**Table 3.2.1a Test Cases**

<b>Script</b>	<b>Action</b>	<b>Expected Result</b>	<b>Actual Result</b>	<b>Degree of Completeness</b>
Camera detecting facial emotions	Camera detecting the seven different emotions within the scope of the system	It should detect all seven emotions based on the facial expression and correctly tell the user what emotion was detected	The camera detects the emotion if the users facial expression matches any of the seven emotions the system understands	Pass
Playback Music	Plays a song that aligns with the user's detected emotion.	The application should playback a music from the corresponding to the directory of the emotion detected	The application playback a music from the corresponding to the directory of the emotion detected	Pass
Functional Buttons	Switch between pages on the application	The users should be able to navigate between pages on the application, thereby using the next, back buttons	The users are able to navigate between pages on the application, thereby using the next, back buttons	Pass
Music Player	Functional Play, pause, next, previous buttons in the music player	Users should be able to pause, switch to the next or previous audio music in the emotion directory	The users are able to pause, switch to the next or previous audio music in the emotion directory	Pass

***Table 3.2.1b Test Cases Continued***



## 3.2 Accuracy and Limitations of the Data Set

The accuracy of the machine learning model's data set has been tested during its pre-training phase. **Figure 3.2.1** shows a confusion matrix for the probability scheme with the seven detectable emotions.

	Neutral	Happiness	Surprise	Sadness	Anger	Disgust	Fear
Neutral	90.27%	1.91%	1.48%	4.95%	1.13%	0.00%	0.26%
Happiness	2.32%	94.47%	1.22%	1.22%	0.77%	0.00%	0.00%
Surprise	6.64%	3.08%	86.97%	0.71%	1.18%	0.00%	1.42%
Sadness	23.21%	1.67%	0.72%	67.94%	3.59%	0.48%	2.39%
Anger	10.16%	3.28%	0.66%	2.30%	82.30%	0.66%	0.66%
Disgust	10.53%	0.00%	5.26%	0.00%	57.89%	26.32%	0.00%
Fear	4.35%	0.00%	29.35%	8.70%	5.43%	0.00%	52.17%

**Figure 3.2.1** Confusion Matrix for the probability scheme (“Training Deep Networks for Facial Expression Recognition with Crowd-Sourced Label Distribution” (Barsoum et. al))

From the matrix above, it can be observed that **Neutral** and **Happiness** are the emotions that are most frequently and most accurately detected, while **Fear** and **Disgust** are the emotions that are least frequently and least accurately detected.

## 4. Future Scope

### 4.1 Future Scope

Possible enhancements and updates to this system are as follows:

- Support the detection of more than seven emotions
- Support a larger music catalog
  - Access to external music platforms such as YouTube, Spotify, Apple Music
- User Profiles and Analytics of Emotion Detections in a dashboard gui
- Forgot Password? functionality
- Mobile and Smartphone supported application version
- Automated playlist generation based on Emotion Detection Analytics

# References

- [1] "Activity Diagrams | Unified Modeling Language (UML)." GeeksForGeeks, January 15. 2024, <https://www.geeksforgeeks.org/unified-modeling-language-uml-activity-diagrams/>
- [2] Barsoum, Emad, (et. al), "*Training Deep Networks for Facial Expression Recognition with Crowd-Sourced Label Distribution*", September 2016, [arxiv.org/abs/1608.01041](https://arxiv.org/abs/1608.01041)
- [3] "Free UML Tool for Fast UML Diagrams." UMLet, UMLet 15.0 stand-alone, February 19. 2024, <https://www.umlet.com/>
- [4] Isaac, "*Using SQLITE3 in Python*", November 7. 2021, [isaacstechblog.com/blog/sqlite3-in-python/](https://isaacstechblog.com/blog/sqlite3-in-python/)
- [5] Johnson, Hubert. *(0.5) Sec 02 Due Dates Spring 2024*. Montclair State University, 2024.
- [6] Lawrence, P. S. (1998). *Software engineering: theory and practice*. Pearson Education India.
- [7] LucidChart, "*How to Create Software Design Documents*", [www.lucidchart.com/blog/how-to-create-software-design-documents](https://www.lucidchart.com/blog/how-to-create-software-design-documents)
- [8] Microsoft, "*FER+*", June 2023, [github.com/Microsoft/FERPlus](https://github.com/Microsoft/FERPlus)
- [9] OpenCV, "*About - OpenCV*", [opencv.org/about/](https://opencv.org/about/)
- [10] Python.org, "*Python 3.11.8*", [www.python.org/downloads/release/python-3118/](https://www.python.org/downloads/release/python-3118/)
- [11] Python.org, "*sqlite3-DB-API 2.0 interface for SQLite database*", [docs.python.org/3/library/sqlite3.html](https://docs.python.org/3/library/sqlite3.html)
- [12] Satya Mallick, *LearnOpenCV - Facial Emotion Recognition*, [github.com/spmallick/learnopencv/tree/master/Facial-Emotion-Recognition](https://github.com/spmallick/learnopencv/tree/master/Facial-Emotion-Recognition)
- [13] Visual Paradigm, *Visual Paradigm Online*, [online.visual-paradigm.com/app/diagrams/](https://online.visual-paradigm.com/app/diagrams/)