A report submitted in partial fulfilment of the requirements for the degree of Bachelor of Science (BSc) in

COMPUTER SCIENCE

UNIVERSITY OF THE WEST OF ENGLAND

FACIAL EMOTION RECOGNITION FOR MUSIC RECOMMENDATION SYSTEM

By

YIE NIAN CHU

Supervisor: Craig Duffy

School of Computing
and Creative Technologies

UNIVERSITY OF THE WEST OF ENGLAND

Date of submission: April 16, 2024
Word count: WORD COUNT

DECLARATION

I, Yie Nian Chu confirm that the work presented in this report is my own. Where information has been derived from other sources, I confirm that this has been indicated in the report.

_____

Yie Nian Chu

ABSTRACT

ACKNOWLEDGEMENTS

ACRONYMS

ACRONYMS

AI Artificial Intelligence

CNNs Convolutional Neural Networks

Conv2D Convolutional Layer

ERD Entity-Relationship Diagram

FER Facial Emotion Recognition

FK Foreign Key

GD Gradient Descent

JS JavaScript

k-NN K-Nearest Neighbors

LR Linear Regression

ML Machine Learning

MLP Multi-Layer Perceptron

MSE Mean Squared Error

PK Primary Key

RF Random Forest

SVM Support Vector Machine

UI User Interface

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1. BACKGROUND

2. LITERATURE REVIEW

2.0.1. Introduction of Music

The development of human communication and social cohesiveness is closely linked to the origins of music. Protohumans probably created the first musical sounds through rhythmic awareness and vocalizations with variable pitch before musical instruments were invented. Primitive humans showed the ability to compose music even before they learned to speak. The development of early human civilizations may have benefited from this musical talent by fostering social cohesion and collaboration.

The comparison of early speech or the pitched sounds of animals to historical music demonstrates how primitive music was as a medium of expression before sophisticated language. A tight relationship exists between the growth of rhythmic awareness and the capacity to produce percussion sounds and the formation of music in the evolutionary timeline. The ability to perceive and produce music was probably important for social interactions when Homo neanderthalensis and Homo sapiens evolved. One major explanation for the evolution of music is the innate human desire to collaborate and form social contexts for doing so. Early human society appears to have relied heavily on music, judging from its role in fostering social cohesiveness, entertaining, enabling dance, and supporting ritualistic behaviors.

Essentially, the origins of music can be found in the basic human urge for collaboration, socialization, and the development of cultural contexts that support these endeavors. Even in its most primitive incarnations, music influenced human communities and may have helped Homo sapiens outperform other hominid species in terms of cultural development.

(a) Cuneiform Alphabet
Image from (Finkel and Taylor, 2021)

(b) World's Earliest Music Composition
Image from (Porter and Molana-Allen, 2018)

2.0.2. Evolution and Diversity of Music

The history of music is extensive and ancient, and it is an essential part of human culture. In Syria, a cuneiform "alphabet" (a) containing the earliest known written composition of music was found (b) (Porter and Molana-Allen, 2018). It is thought to have been composed some 3400 years ago. The oldest musical instruments, according to archaeological evidence, date back approximately 40,000 years, but the history of music is far older still (Killin, 2018). Despite not yet being documented in the archaeological record, these instruments offer a window into far older musical endeavors. Proto-musical evolution most likely started about 400,000 years ago, based on the social brain hypothesis (Dunbar, 1998). Musical traditions developed further as people left Africa and spread over the world, reaching a turning point during the Holocene. This historical voyage demonstrates the persistent and varied impact that music has played in human history. Moving forward, there are so many types of music, each representing unique expression of culture, emotion and creativity:

- **Classical Music**: a diverse and evolving tradition, extending beyond the commonly associated period of 1750 to 1820 and encompassing composers from Bach to contemporary artists. It serves as a living and influential force in the world of music, shaping compositions for orchestras, chamber ensembles, solo performers, and even finding unexpected expressions in various genres, from video-game scores to popular music. (Gabler, 2013)

- **Jazz**: originating in early 20th-century New Orleans, is characterized by complex harmony, syncopated rhythms, and a focus on improvisation. Emerging from a rich tradition of ragtime and blues, jazz evolved into a versatile genre that expanded

its influence globally, encompassing popular music standards, modal music, and even avant-garde compositions.(Beek, 2021)

- **Blues**: both a musical form and genre, initially associated with melancholy themes, has evolved to encompass a broader range of subjects and emotions, aiming to uplift through music.(Chaudhuri, 2022) Characterized by specific chord progressions, a walking bass, call and response, and unique features like microtonality and flattened ’blue’ notes, blues is known for its distinct sound and expressive style. (BBC, 2023)

- **Electronic**: utilizes diverse sound sources, ranging from recorded sounds captured by microphones to those generated by electronic oscillators and complex computer installations. Typically played back through loudspeakers, electronic music can be created using various technologies, with the exception of "live electronic music," which involves real-time performance. (Hiller, 2019)

- **Country**: originating in rural Southern and Western areas in the early 20th century, was initially labeled "hillbilly music" before adopting the official term "country and western music" in 1949. Rooted in the ballads, folk songs, and popular tunes of English, Scots, and Irish settlers, country music gained commercial recognition in the early 1920s, marked by its realistic portrayal of rural life contrasting with the sentimental tone prevalent in popular music of that era. (Britannica, 2019)

- **Hip-Hop**: a cultural movement that gained widespread popularity in the 1980s and ’90s, serves as the foundational music for rap—a style featuring rhythmic and/or rhyming speech, which became the movement’s enduring and influential art form. Hip-hop, beyond its musical dimension, encompasses diverse elements such as graffiti art, breakdancing, and social activism, reflecting a multifaceted expression of urban culture and creativity. (Tate and Light, 2019)

- **Rock**: a form of popular music that emerged in the 1950s and by the end of the 20th century became the dominant global music genre, influencing the recording industry, international retail, and radio and television playlists. While dictionary definitions often focus on its strong beat and instrumentation, the cultural significance of rock lies in its social and ideological distinctions from other music

genres, particularly its development as a term to distinguish certain attitudes and practices from those associated with pop music.(Frith, 2018)

2.0.3. Music and Emotion

Since ancient times, people have been fascinated by the paradoxical connection that exists between music and emotion. Even though music is an abstract art form that appears to be removed from everyday life, it has a profound potential to evoke strong emotional responses. This ability of music to trigger strong emotions is also demonstrated in other social circumstances, such as advertising. This encounter is further enhanced by the relationship that exists between music and our individual life experiences. Emotions, influenced by these encounters, provide our perception and thought processes a personalized meaning that connects the abstract quality of music to the concrete events of our everyday life. This complex tapestry that highlights the profound influence of music on our emotional environment is created by the blending of music, emotion, and human experience (Juslin and Sloboda, 2013).

Numerous musical elements that have been thoroughly explored are combined to convey emotions through music. Pace, mode, harmony, interval, rhythm, sound level, timbre, timing, articulation, accents, tone attacks and decays, and vibrato are some of these characteristics. Emotion in music is expressed through compositional elements as well as performance elements. Still, it’s not an easy task to express feelings through music. Certain musical elements can be employed to convey a range of moods, showing that certain elements are not always reliable predictors of a certain emotion. The Lens Model (Juslin and Sloboda, 2013), which characterizes emotional expression in music as involving probabilistic and partially redundant auditory cues, sheds more light on this complexity. Listeners combine several cues for successful emotion recognition, and the redundancy of cues allows for a high level of emotion recognition through different combinations, offering room for creativity and personal expression (Pereira et al., 2011).

2.1. ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

2.1.1. Artificial Intelligence

Artificial Intelligence (AI) is a broad field that includes using technology to build machines and computers that can replicate cognitive abilities connected to human

intellect. These abilities include making recommendations, language understanding, data processing, and visual perception. AI should be viewed as a group of technologies incorporated into systems rather than as a stand-alone system that can understand, learn, and respond to complex problems.

2.1.2. Machine Learning

Machine Learning (ML) is a branch of AI focused on enabling machines and systems to learn and enhance their performance through experience. Instead of relying on explicit programming, machine learning employs algorithms to analyze vast datasets, derive insights, and subsequently make informed decisions. These algorithms continually improve their performance as they are exposed to more data. The outcomes of this learning process are the machine learning models, which become more proficient with increased exposure to data.

2.1.3. Machine Learning Algorithms

2.1.3.1. Linear Regression

Linear Regression (LR) is a supervised learning algorithm used to model the relationship between a dependent variable (target) and one or more independent variables (features). The fundamental assumption of LR is that there exists a linear relationship between the input variables and output. (IBM, 2022)

$$y = mx + b \tag{2.1}$$

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n + \epsilon \tag{2.2}$$

A LR model can be represented by the equation 2.2 where Y represented the dependent variable. β_0 is the y-intercept, $\beta_1, \beta_2, \cdots, \beta_3$ are the coefficients and the ϵ is an error term, representing the unobserved factors that affect Y but are not accounted for by the model. The logic of it is same as Linear Equation (2.1) where using the gradient of the line (m), the value of x and the y-intercept (b) to get the value of y, which is what we are trying to predict.

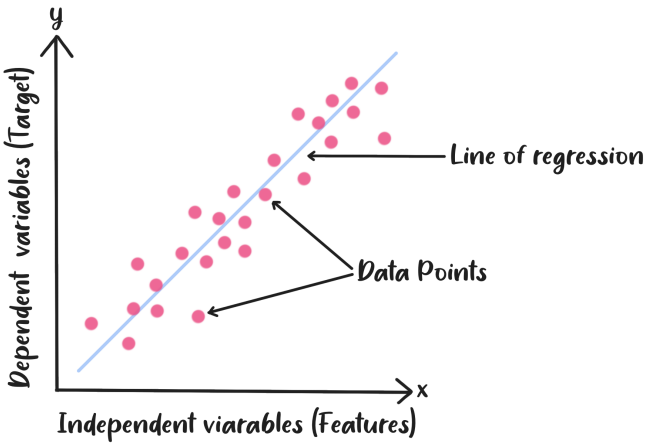


Figure 2.2: Linear Regression

While using LR, the main objective is to find the values of $\beta_0, \beta_1, ..., \beta_n$ that minimize the error between the predicted value ($\hat{Y}$) and the actual value (Y) in the training data.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{2.3}$$

Therefore, Mean Squared Error (MSE), a cost function to measures the average squared difference between $\hat{Y}$ and Y, is introduced to quantify the goodness of fit of the model to the training data. If the current result is not optimized, Gradient Descent (GD), an optimization algorithm, will be used to adjust the coefficients, $\beta_0, \beta_1, ..., \beta_2$, towards the direction that minimizes the MSE until it reach the convergence (Figure 2.3).

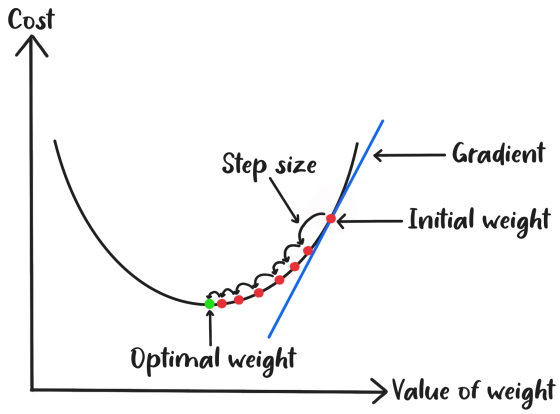


Figure 2.3: Gradient Descent

2.1.3.2. Support Vector Machine

Support Vector Machine (SVM) is a supervised ML algorithm where we used for classification, regression and outliers detection. The main goal of it is to find the hyperplane that best separates the data into different classes based on statistical approaches. (Géron, 2019)

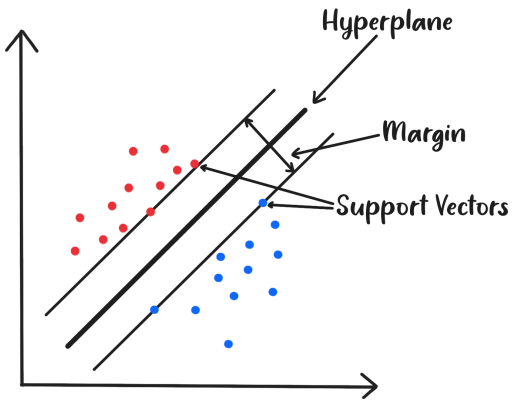


Figure 2.4: Support Vector Machine

While using SVM, the greater the margin, the better the result would be as it has better generalization to new or unseen data. There are two types of SVM for classification, which are Linear SVM and Non-linear SVM. A linear SVM finds the optimal hyperplane that maximizes the margin between classes for linearly separable data as Figure 2.4.

$$f(x) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b) \tag{2.4}$$

The decision function of linear SVM (Equation 2.4) is used to defines the ability to classify data points into different classes. When the result is greater than or equal to zero, the prediction would be positive. If $f(x)$ is less than zero, the decision function predicts the negative class.

For non-linearly separable data, SVM uses kernel functions such as polynomial, sigmoid and radial basis function (RBF), to map the data into a higher-dimensional space where hyperplane can separate the classes. The equation 2.5 is the decision

$$f(x) = \text{sign}(\sum_{i=1}^{n} \alpha_i y_i K(x, x_i) + b) \quad (2.5)$$

$$K(x, x_i) = (x \cdot x_i + c)^d \quad (2.6)$$

$$K(x, x_i) = \exp(-\frac{\| x - x_i \|^2}{2\sigma^2}) \quad (2.7)$$

$$K(x, x_i) = \tanh(\alpha x \cdot x_i + c) \quad (2.8)$$

function of non-linear SVM. The kernel function, denoted by $K(x, x_i)$ in the equation, would be replaced by equation 2.6 if a polynomial kernel were used. Similar with the other kernels, if the RBF kernel is employed, it would be exchanged with equation 2.7, and for sigmoid kernel, it would be replaced with equation 2.8.

2.1.3.3. Random Forest

Random Forest (RF) is an ensemble learning algorithm that belongs to the family of decision tree-based methods. A group of decision trees that have been trained on various dataset subsets make up the forest of RF, and the final result is derived from averaging the predictions of each individual tree. (IBM, 2023*b*)

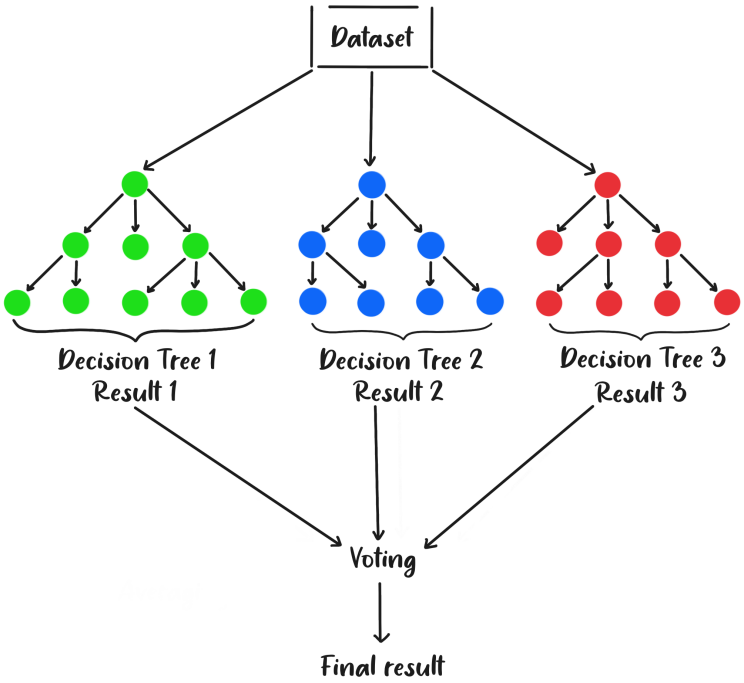


Figure 2.5: Random Forest

As Figure 2.5 shown, RF builds multiple Decision Trees and combines them to get a

more accurate and stable prediction than any individual model. This process is called bagging which is one of a type of ensemble learning. In the process, the entire dataset is separated into subsets and each decision tree is trained individually on a subset that is selected at random. This adds variety and unpredictability to the trees. The training process will then generate results for each model, and the final output is determined by the "votes" for a class from each tree. The class with the majority of votes is chosen as the final prediction.

2.1.3.4. K-Nearest Neighbors

K-Nearest Neighbors (k-NN) is a intuitive supervised machine learning method used for both classification and regression tasks. The main idea of k-NN is to predict the label of new data point based on its k-nearest data points in the feature space. (IBM, n.d.*b*)

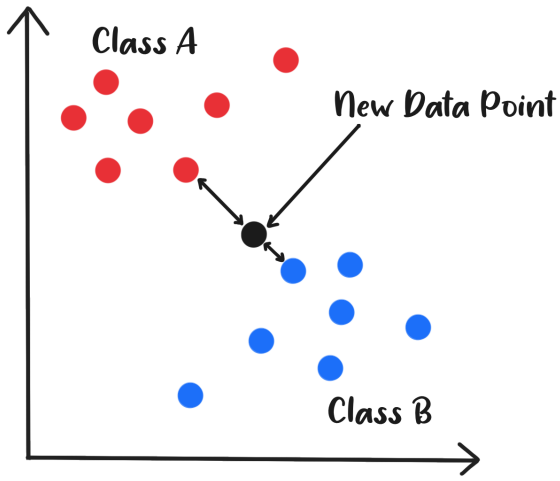


Figure 2.6: K-Nearest Neighbors

k-NN uses distance metric (Equation 2.9) to calculate how similar two data points are to one another. k-NN finds the k training data points that, according to the selected distance metric, are closest to a given data point. In classification tasks, the majority class among a new data point's k-nearest neighbors predicts the class that the data point will fall into.

$$d(P,Q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{2.9}$$

$$\hat{Y} = \text{argmax}_y \left(\sum_{i=1}^{k} I(y_i = y) \right) \tag{2.10}$$

The key hyperparameter of k-NN is the value of k (Equation 2.10), representing the number of nearest neighbors to consider. The choice of k can significantly impact the performance of the algorithm, and it is often selected through cross-validation.

2.1.3.5. Neural Networks

2.1.3.5.1 Multi-Layer Perceptron

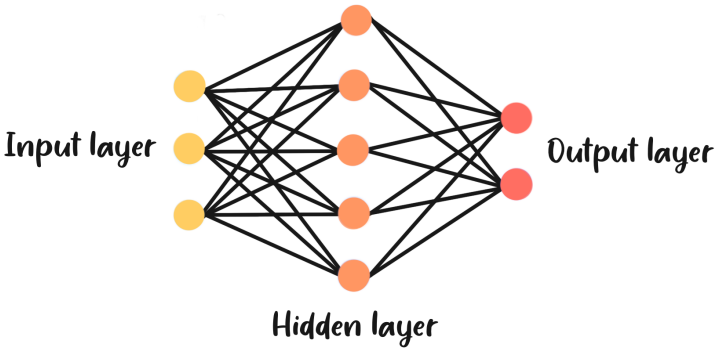


Figure 2.7: Multi-Layer Perceptron

An input layer, one or more hidden layers, and an output layer are the minimum number of nodes that make up a Multi-Layer Perceptron (MLP) feedforward artificial neural network type (Figure 2.7). All nodes in these layers—aside from those in the input layer—are linked using a specific weight and employ a nonlinear activation function. Because of its nonlinearity, the network may learn and carry out more complicated tasks as well as represent intricate connections between the input and output. A MLP model is trained by comparing its output to the expected output and propagating errors back through the network to alter the weights. This process is known as backpropagation. (Haykin et al., 2014)

2.1.3.5.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a family of deep learning algorithms that are mostly utilized for processing input that has a grid structure, like images. (Yamashita et al., 2018)

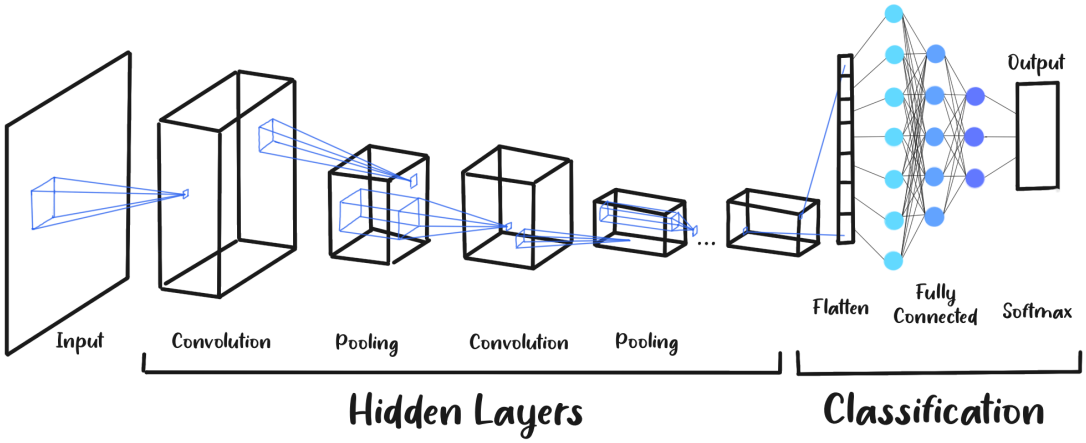


Figure 2.8: Convolutional Neural Networks

CNNs are designed to adaptively learn spatial hierarchies of features from the data. This learning process includes convolution layers, pooling layers, flatten layer, and fully connected layers. By applying filters, also known as kernels, to the input, these layers carry out the convolution process and produce feature maps. Local elements like textures and edges are captured throughout this procedure. Pooling layers, which come after convolutional layers, help to reduce the number of parameters and computation in the network by reducing the spatial dimensions (width and height) of the input volume.

The features from the input image are retrieved by the convolutional and pooling layers, and the next stage is to categorize the features which is done in flatten layer. The feature maps are converted into a one-dimensional vector in the flatten layer, which is necessary for fully connected layers. The flattened vector is then fed into the fully connected layers (which resemble the standard neural network layers with fully connected nodes) for the classification task. These fully connected layers divide the image into discrete groups based on the high-level characteristics found in the preceding levels.

The last layer of layers in the network architecture play the crucial job of generating the output. The output layer typically uses a softmax activation function in multi-class classification settings to translate the network's raw output into probabilities given to each class. The output node with the highest probability is then chosen to determine the anticipated class.

2.2. FACIAL EMOTION RECOGNITION

Human emotions can be inferred from facial expressions. Deciphering these signs of emotion has become a popular research topic in the fields of Human Computer Interaction and Psychology. (Vemou et al., 2021) The development of Facial Emotion Recognition (FER) technology has been significantly aided by technological advancements, particularly with the introduction of ML and Pattern Recognition.

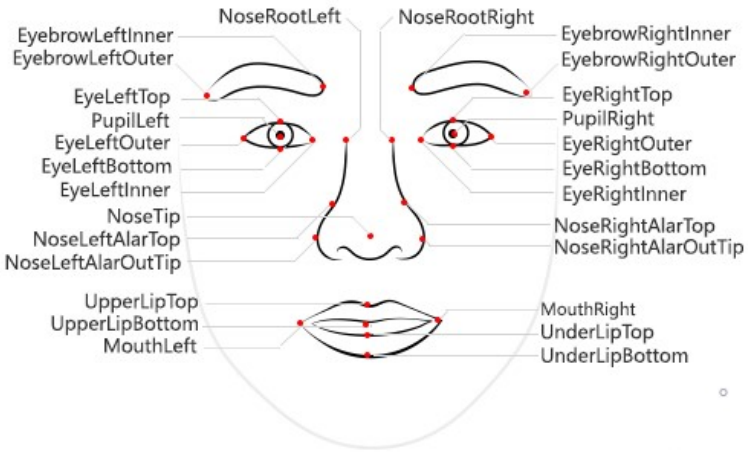


Figure 2.9: Facial Landmarks
Image from (Farley et al., 2023)

FER is a broad field that intersects with Computer Science, AI, Psychology and other fields. It involves analyzing a person's facial expressions in still images and videos in order to determine their emotional state. A three-step approach is used in the methodology: face detection, facial expression identification, and categorization of the expression into a certain emotional state. Facial landmark (Figure 2.9) detection and analysis of changes in their positions are key components of this complex process. FER attempts to offer insights into people's emotional experiences by identifying muscular contractions linked to various emotions from visual clues found in facial expressions.

As stated by Tarnowski et al. (2017), the creative feature extraction from facial expressions using coefficients that detailed aspects of emotional states is what makes the research successful. They distinguished between seven different emotional states: happiness, sorrow, surprise, wrath, fear, and contempt, as well as the more subdued

Emotions	neutral	joy	surprise	anger	sadness	fear	disgust
neutral	881	125	2	155	340	48	101
joy	106	922	7	238	115	1	154
surprise	13	1	1135	8	8	390	13
anger	130	151	6	862	81	2	120
sadness	229	130	33	101	823	104	88
fear	41	0	220	5	88	871	4
disgust	76	147	73	107	21	60	996

(a) Confusion Matrix for k-NN Classifier

Emotions	neutral	joy	surprise	anger	sadness	fear	disgust
neutral	1160	75	9	29	644	61	41
joy	81	1178	0	57	141	0	129
surprise	3	0	1153	4	2	426	10
anger	58	137	0	1346	44	0	88
sadness	122	13	5	0	561	75	2
fear	8	1	308	1	74	910	2
disgust	44	72	1	39	10	4	1204

(b) Consfusion Matrix for MLP Classifier

Tables from (Tarnowski et al., 2017)

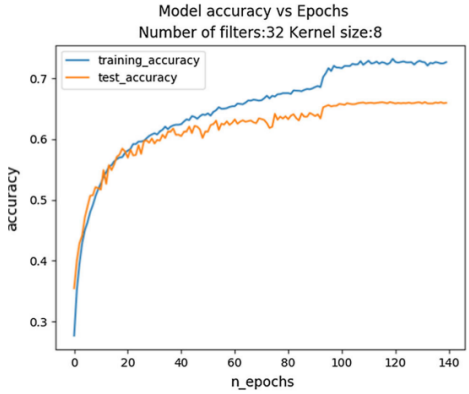
displays of neutrality. For featre computation, they employed a three-dimensional facial model as an alternative to conventional two-dimensional approaches. This enables them to collect more detailed and subtle data, which may improve the accuracy of identifying emotions. On top of that, they used the MLP neural network and the k-NN classifier to classify each emotional state. According to their research and comparative analysis, the MLP is more accurate than the k-NN in classifying emotional states, with a 73% classification accuracy compare to a 63% accuracy for k-NN (Tarnowski et al., 2017).

Mellouk et al. (2020) showed in their study that FER may be achieved with great accuracy and effectiveness by utilizing deep learning techniques. They provided a thorough analysis of multiple FER databases, emphasizing their diversity in terms of picture, video content, lightning circumstances, and demographic variances—all of which are important determinants of FER performance—in order to guarantee the credibly of the results.

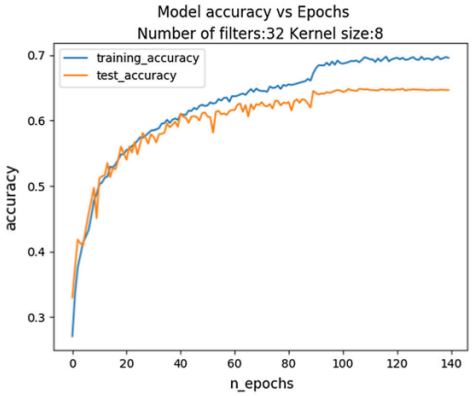
Traditional facial recognition techniques included manually defining and extracting features from facial photos, a procedure that was frequently less flexible and efficient. Examples of these techniques include Local Binary Patterns (LBP), Facial Action Coding System (FACS), Local Directional Patterns (LDA), and Gabor wavelet. CNNs and LSTMs can automatically extract and learn complex patterns from facial data, according to Mellouk et al., which will improve the reliability and accuracy of emotion recognition. Furthermore, the difficulties that traditional approaches faced—such as variances in facial characteristics due to diverse demographics, occlusions, and data diversity—are resolved with the use of deep learning, increasing their versatility and effectiveness. The study also examines preprocessing methods including image scaling, cropping, normalization, and data augmentation that are crucial for improving

the accuracy of these deep learning models.

With the help of deep learning and the preprocessing techniques they found, the outcome demonstrates proficiency in accurately classifying the fundamental emotions, with some models reaching over 90% accuracy under specific circumstances (Mellouk and Handouzi, 2020). It indicates that as machines improve at deciphering human emotions, interactions between humans and machines may become more intuitive and natural.



(a) Accuracy variation: Model1



(b) Accuracy variation: Model2

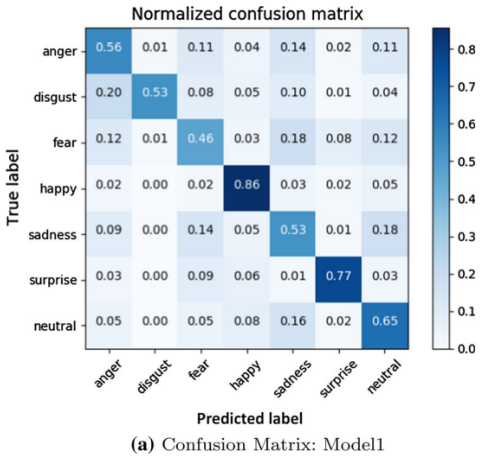
(a) Accuracy variation with 32 Filters and 8 kernel size for Model 1

(b) Accuracy variation with 32 Filters and 8 kernel size for Model 2

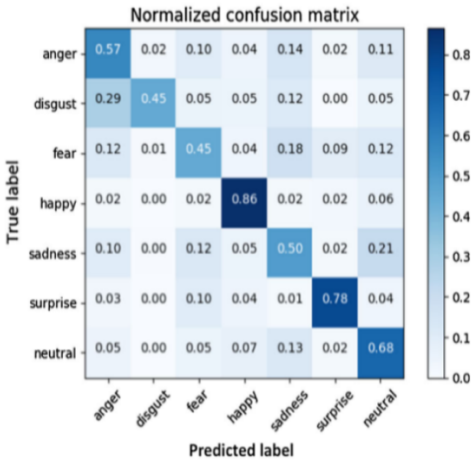
Graphs from (Agrawal and Mittal, 2019)

Based on the findings of Agrawal et al.'s work, the kernel size and the number of filters significantly impact CNNs accuracy. Using the FER-2013 dataset as their primary emphasis, two CNN architectures is put forth after a thorough analysis of various kernel sizes and filter counts. To find the optimal set of parameters that could yield the best convergence and accuracy, Agrawal et al. ran tests with the combination of 6 different kernel sizes (2, 4, 8, ..., 64) and 8 different number of filters (2, 4, 8, ..., 256). They discovered that when network depth increased, a network with 32 filters and an 8 kernel size demonstrated a discernible gain in accuracy (Figure 2.11a, Figure 2.11b).

Even while Model 2 is simpler due to its constant kernel size, lack of dropout layers, and fully connected layers, it was nevertheless able to achieve an accuracy of 65% on the FER-2013 dataset, which is comparable to human performance (Agrawal and Mittal, 2019). Furthermore, in comparison with other emotions, the proposed models were



(a) Confusion Matrix for Model 1



(b) Confusion Matrix for Model 2

Graphs from (Agrawal and Mittal, 2019)

able to categorize happiness and surprise with a higher degree of accuracy, which is consistent with people's challenges in picking out distinct emotions (Figure 2.12a, Figure 2.12b).

Gestures	Recognition Accuracy
Neutral	99.00%
Smile	98.50%
Anger	98.50%
Scream	99.50%
Overall	98.88%

Figure 2.13: Linear Regression Classification Accuracies Table
Table from (Naseem et al., 2010)

A pivotal study by Naseem et al. (2010) incorporates the analysis of facial expressions, recognizing them as crucial variations in appearance induced by internal emotions or social communications. In order to evaluate their LR Classification approach, they therefore took into account occlusion modes, brightness changes, and expressions such as scream, smile, rage, and neutral Notably, the LR Classification algorithm showed an excellent recognition accuracy for all facial expressions tested, averaging 98.88% in a 100D feature space (Figure 2.13). For the screaming expression, the algorithm outperformed other accuracies by achieving an accuracy of 99.5% (Figure 2.13). This great accuracy demonstrates the reliability and efficacy of the LRC approach in handling a wide range of facial emotions.

	Facial Expression				Total
	Anger	*Happiness*	*Sadness*	*Surprise*	
Training (70%)	31	48	19	58	156
Testing (30%)	14	21	9	25	69
Total	45	69	28	83	225
Success	79%	95%	89%	96%	90%

Figure 2.14: Random Forest Classification Accuracy
Table from (Munasinghe, 2018)

According to Munasinghe (2018), RF Classifier are capable of handling facial expression variability well and without overfitting. Also, the researcher asserts that facial landmarks (Figure 2.9) provide an accurate feature extraction capability that capture subtle changes in facial emotions. A facial feature vector obtained from these landmarks and normalized to reduce variance in face size is used to discern emotions with a RF Classifier. With the aid of feature vector, the RF Classifier achieved an average success rate of 90% in classifying four different emotions: anger, happiness, sadness, and surprise (Figure 2.14).

	happy	surprise	fear	angry	sad	disguise
h	144	7	5	10	11	23
s	21	147	6	10	1	15
f	1	5	143	8	19	24
a	11	2	21	132	26	8
s	6	14	18	27	119	16
d	12	13	24	21	6	124

Figure 2.15: Support Vector Machine Accuracy
Table from (Xia, 2014)

Li Xia (2014) presents a unique method of facial emotion detection that employs multi-classification SVM. The study proposes a two-on-two classification method which is an innovative approach to overcome the limitations of traditional classification methods like one-against-one (classifier is trained for each pair of classes) and one-against-the-rest (classifier is trained against all other classes combined). With this novel method, the classification process is faster with fewer sub-classifiers and reduced classification errors. The results of this study were impressive, showing the classifier in this investigation demonstrated a high average recognition rate of 92.7% when six distinct emotions were considered, including happiness, surprise, anger, fear, disgust,

and sadness.

2.3. MUSIC RECOMMENDATION BASED ON FER

From Chakrapani et al.'s approach, music recommendation system with deep learning algorithm could enhance the listening experience by accurately detect and interpret the user's emotions. This is achieved by using CNNs to analyze the user's age, gender, and facial emotion. Based on these data, the system would cater to the user's preferences and present mood. To ascertain the user's emotional state, they used the webcam to take pictures of the user and then processed the image using the CNNs models. The system then provided tailored music recommendations based on the predictions generated by the CNNs models. This approach offers a creative and user-centric alternative for music selection based on emotional cues while streamlining playlist construction and management.(S et al., 2023)

Additionally, Athavle et al. discovered that using CNNs model helps a music recommendation system to accurately detect emotions and subsequently recommend music that aligns with the user's mood. They train a CNNs model for emotion detection in their work. While maintaining great precision, this method lowers total system costs and computing time. The system uses real-time emotion detection to work, and then sends the data to the CNNs model to classify the user's emotions. An appropriate playlist will be recommended as soon as the technology determines the user's current feeling, making the user experience engaging and responsive. In order to guarantee optimal classification accuracy and efficacy, they employed categorial cross-entropy as a loss function to manage missing and anomalous values inside the FER2013 dataset. Despite their result being less accurate than Chakrapani et al.'s work (71%), it nevertheless shows that the model is effective and trustworthy in identifying emotions from facial expressions. (Athavle, 2021)

3. REQUIREMENTS

3.1. FUNCTIONAL REQUIREMENTS AND NON-FUNCTIONAL REQUIREMENTS

3.1.1. Introduction

To ensure that the project is built effectively, a comprehensive framework of functional and non-functional requirements that are carefully crafted to meet the objectives and user expectations should be created. Functional requirements provide the application's foundational architecture, dictating essential tasks such as playlist generation and user registration to ensure the application performs reliably and intuitively. Non-functional requirements, on the other hand, are focused on performance quality and include things like system scalability, security, and efficiency.

3.1.2. Functional Requirements

Req. No.	Categories	Requirements	Priority
FR1	User Registration and Account Management	The system must allow user to register by providing a unique username, user's actual name, date of birth, email, and password.	High
FR2		The system must verify user accounts through an email verification process.	High

Req. No.	Categories	Requirements	Priority
FR3		Users must be able to login with their email or username and password. A "Remember Me" option should allow users to stay logged in for 7 days.	High
FR4		Users can access a settings page to update their name, date of birth, email, password, and profile picture. Usernames cannot be changed.	Medium
FR5		Users must be able to reset their passwords through a password reset feature on the login page.	Medium
FR6	Facial Emotion Recognition	The application integrates a machine learning model to recognize user's facial emotions via their device's camera.	High
FR7	Spotify Web Playback Integration	The system integrates with Spotify Web Playback SDK to play music within the web application.	High
FR8		The application must allow users to connect their Spotify account before accessing music playback services. This integration should facilitate authentication and authorization seamlessly within the web application.	High
FR9	Music Recommendation System	The application must generate playlists based on the user's recognized emotion using an algorithm.	High

Req. No.	Categories	Requirements	Priority
FR10	User Interface and Experience	The web application supports a toggle between light and dark themes, automatically detecting and applying the user's device theme upon first use.	Medium
FR11		The application supports multiple languages: English, Japanese, Chinese, Korean, and Malay.	Low

Table 3.1: Functional Requirements

3.1.3. Non-Functional Requirements

Req. No.	Categories	Requirements	Priority
NFR1	Performance and Scalability	The application shall load within 3 seconds for 95% of its users under standard network conditions.	High
NFR2		The system must be scalable to support up to 100 concurrent users without significant degradation in performance.	High
NFR3	Compliance and Security	All user data, including passwords and personal information, must be encrypted.	High
NFR4		The application must implement secure authentication mechanisms to prevent unauthorized access.	High
NFR5		User data must be stored in a secure database with access strictly limited ot the backend server. The database shall not be directly accessible from any public network (0.0.0.0/0).	High
NFR6		User passwords must be encrypted using a secure hashing algorithm (e.g., bcrypt) to ensure their safety even in the event of a data breach.	High
NFR7		All forms of data transmission involved in user authentication and registration must be over HTTPS, and sensitive information shall not appear in URLs or any part of the HTTP request visible to the client side.	High

Req. No.	Categories	Requirements	Priority
NFR8		The application must comply with relevant data protection and privacy regulations, including GDPR where applicable, ensuring user's rights to privacy and data security are upheld.	High
NFR9	Usability	The application shall be designed with a user-friendly interface, ensuring ease of navigation and accessibility.	Medium
NFR10		User input fields should provide immediate feedback to correct errors or invalid data.	Medium
NFR11	Compatibility and Interoperability	The web application must be compatible with the latest versions of Chrome, Firefox, Safari and Edge browsers.	High
NFR12		The system must ensure seamless integration with the Spotify API and maintain compatibility with Spotify's update.	High
NFR13	Localization and Internationalization	The application must support multi-language interfaces, allowing users to switch languages easily.	Medium
NFR14		Date and time formats should adapt to the user's selected language and region preferences.	Low
NFR15	Maintenance and Support	The system should be designed to allow easy updates and maintenance without significant downtime.	Medium

Req. No.	Categories	Requirements	Priority
NFR16		Documentation must be provided for end-users and developers, detailing usage, integration features, and troubleshooting steps.	Medium
NFR17	Application Performance	The facial emotion recognition feature must provide a response within 5 seconds from the time of user's request under standard network conditions.	High
NFR18		The system should ensure a Spotify playback start time of less than 3 seconds after user selection or playlist generation.	High
NFR19		The web application's overall time to interactive (TTI) should not exceed 5 seconds for 90% of its users under standard network conditions.	High
NFR20	User Interface Design	The application must adhere to WCAG 2.1 AA standards for color contrast, navigability, and text size to ensure accessibility for users with disabilities.	High
NFR21		All user interface components (buttons, links, form elements) must be navigable using a keyboard in a logical order to support users with mobility or visual impairments.	High

Req. No.	Categories	Requirements	Priority
NFR22	Data Handling and Authentication	Implement OAuth 2.0 for secure authentication with Spotify, ensuring that user credentials are handled safety and in line with best security practices.	High
NFR23		Apply secure session management practices, including the generation of unique session tokens for users during login and their secure storage on the client side.	High

Table 3.2: Non-Functional Requirements

4. METHODOLOGY

4.1. INTRODUCTION

To effectively navigate the intricacies of software development and guarantee the project's success, choosing an appropriate approach is essential. The concepts, procedures, and practices that guide a project's development, implementation, and completion are collectively referred to as its methodology. The variety of alternative methods, each with specific advantages and applicability to various project types, means that selecting a methodology should be done with careful consideration. This section explores the reasoning behind the choice of an Agile-based methodology, with a particular emphasis on the Kanban methodology, made by the use of Notion for project management. This decision was driven by the project's requirement for flexibility, continuous improvement, and a visual workflow management system. The following sections will outline the comparative comparison between various approaches, along with the reasoning behind choosing Agile due to its alignment with the project's objectives and task specifications.

4.2. RESEARCH METHODOLOGY

4.2.1. Waterfall methodology

Waterfall methodology, with its linear and sequential approach, stands as a traditional yet relevant framework for software development projects that require a clear, phased progression. Requirements, design, implementation, testing, development and maintenance are the steps in this process. They guarantee that one phase must be finished before moving on to the next, which makes them especially appropriate for projects with stable, well-defined requirements that are unlikely to change. Crespo-Santiago & de la Cruz Dávila-Cosme (Crespo-Santiago and Dávila-Cosme,

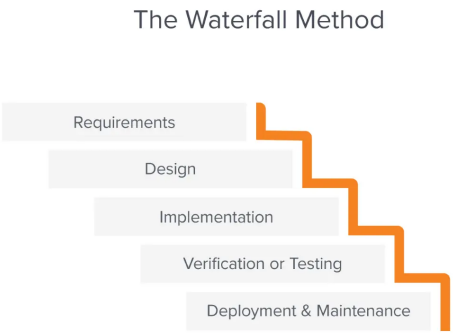


Figure 4.1: Waterfall Methodology (Communitcation Team, 2022)

2022) highlight the Waterfall methodology helps maintain the scope of their library project within the requirements, estabilishing cost and time control, and documenting evidence of project governance.

4.2.2. Spiral methodology

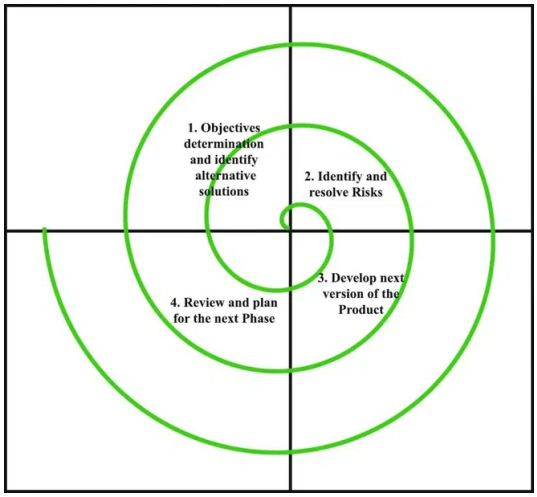


Figure 4.2: Spiral Methodology (Kumar Pal, 2018)

Spiral methodology, an evolutionary software development process introduced by Barry Boehm in 1986, is a model for process flexibility and risk control in software development. (Boehm, 1986) The methodology is distinguished by its four-phase cycle approach, which includes planning, risk analysis, implementation, evaluation. This allows for ongoing iterations that involve setting project goals, identifying potential risks, carrying out development, and incorporating stakeholder feedback. Its iterative design ensures a flexible and adaptive development process by permitting incremental product

improvements based on changing needs and stakeholder input. The Spiral methodology provides a methodical approach to managing the complexities and uncertainties inherent in software development. It shines in contexts where project needs are ambiguous or subject to change because of its heavy emphasis on risk management.

4.2.3. Agile methodology

Figure 4.3: Agile Methodology (Laoyan, 2022)

Agile methodology, an approach originated from the Agile Manifesto, published in 2001 by a group of software developers, prioritizes adaptability, collaboration, customer satisfaction and timely delivery of high-quality software. While implementing Agile methodology, project is breaks into small, manageable pieces, known as iterations or sprints. Each sprint involves cross-functional teams working on various aspects like planning, design, coding, and testing, with a working iteration of the product delivered at the end of each cycle. This methodology works effectively for projects whose requirements are changing or unclear since it allows ongoing feedback and adjustment.

4.3. COMPARISON AND SELECTION

Software development can be approached differently using the Agile, Waterfall, and Spiral techniques, each with its own set of benefits and difficulties. From Table 4.1, Agile is highly flexible and adaptable, ideal for projects with evolving requirements, but may lead to unpredictable costs. Waterfall is straightforward and orderly, perfect for projects with well-defined requirements, but inflexible to changes. Spiral combines iterative development with focusing on risk management, but it could be costly. The project's scale and the clarity of its needs determine which technique is best: Waterfall for its

Table 4.1: Comparison of Methodologies

Aspect	Agile	Waterfall	Spiral
Pros			
	• High flexibility and adaptability to changes. • Frequent releases and feedback. • Enhanced customer satisfaction. • Reduced time to market.	• Simple and easy to understand and use. • Clear project milestones and deliverables. • Well-suited for projects with defined requirements.	• Focus on risk management. • Flexibility in design and development. • Suitable for large, complex projects with uncertain risks.
Cons			
	• Less predictable budget and timeline. • Requires close collaboration and customer involvement. • Not ideal for low-change projects or those with fixed requirements.	• Difficult to incorporate changes once the project has started. • Potential for late discovery of problems or errors. • Not suitable for projects where requirements may evolve.	• Can be complex and costly to implement. • Requires significant risk assessment expertise. • May lead to prolonged project duration due to iterative nature.

structure, Agile for its flexibility, or Spiral for its risk emphasis.

4.4. JUSTIFICATION FOR CHOOSING AGILE

The Agile methodology, particularly the Kanban variant, was chosen in order to satisfy the demand for an adaptable, graphical, and iterative developments process. Given the dynamic nature of the project and its ever-changing objectives, Kanban, which places a strong focus on continuous delivery and workflow efficiency, is an excellent fit.

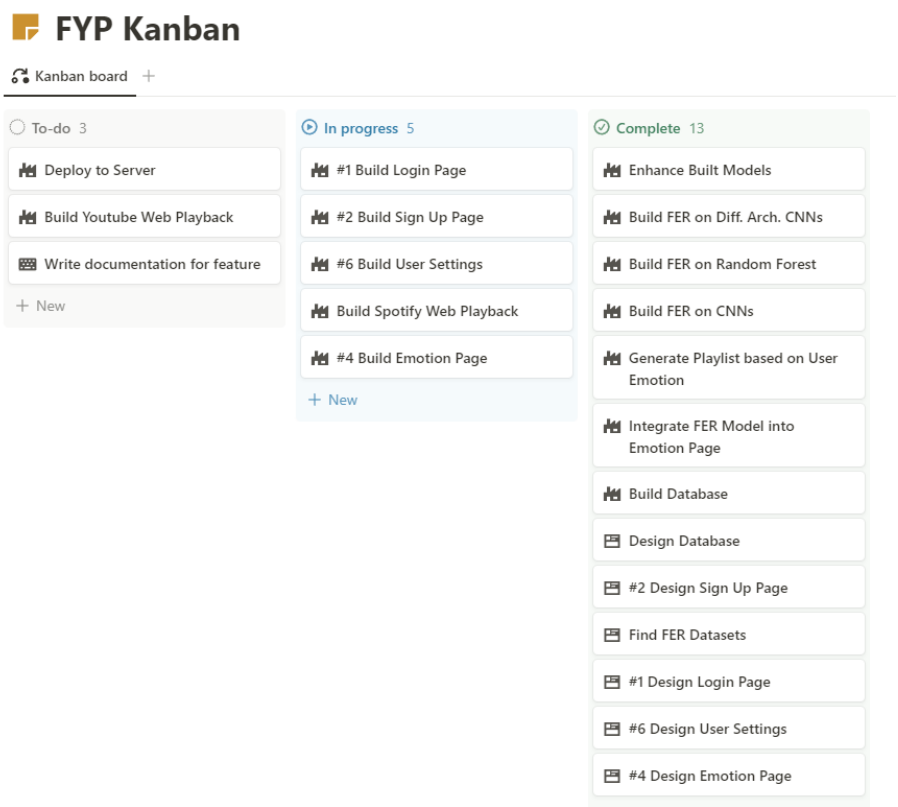


Figure 4.4: Kanban from Notion

In this project, Kanban is implementing using Notion. It gives tasks a visual representation, making it easier to organize and keep track of them as they go through various stages of development. Also, with the adaptability of Notion's platform, project plans could be updated easily which is crucial for keeping the plan responsive to changing project dynamics.

Kanban method's inherent simplicity and its focus on delivering work just-in-time are particularly beneficial for projects demanding flexibility and time efficiency. Kanban with Notion provides a clear picture of the project's progress, enabling developer to identify

and resolve bottlenecks quickly, and efficiently manage task prioritization. Therefore, the combination of Notion’s features with Agile Kanban creates a strong foundation for project management by fusing Kanban’s visual clarity and streamlined efficiency with Agile’s flexibility.

5. DESIGN

5.1. INTRODUCTION

The design phase of the web application, focused on using emotion recognition to generate music therapy playlists, represents a bridge from theoretical concepts to a operational system. The varied design approaches that were used to develop an application with a solid technical framework and user-friendly interface are covered in this section.

The core aspect of the application is how it uses captured frame analysis to determine a user's emotional state by utilizing FER technology. This sense of emotional then guided the creation of customized music playlists, fusing the advanced machine learning techniques with the restorative properties of music to provide a unique advantageous user experience.

This project uses diagrams such as block diagram, use case diagram, sequence diagram and etc., where each focusing on a different aspect of the architecture and functionality of the system. Futhermore, this section delves into the visual and functional components of the application's user interface as illustrated by the Logo Design and Interface Design. The logo, as the visual cornerstone of the application's brand identify, have been thoughtfully designed to improve application usability and user engagement.

Additionally, the ML model architecture design, which describes the underlying algorithms and data processing methods used in FER, is also discussed in this section. This discussion covers the model's validation, training, and integration into the broader application ecosystem to ensure accurate emotional analysis.

All of these components provide a thorough explanation of the system's design, addressing structural, behavioral, and aesthetic factors from the abstract to the concrete.

5.2. WEB APPLICATION

5.2.1. UML Diagrams

5.2.1.1. Block Diagram

Block diagram present a high-level overview of the system's architecture, highlighting the major components and their interactions. (Freeman, n.d.) This offers insights into the overall system design and its flow.

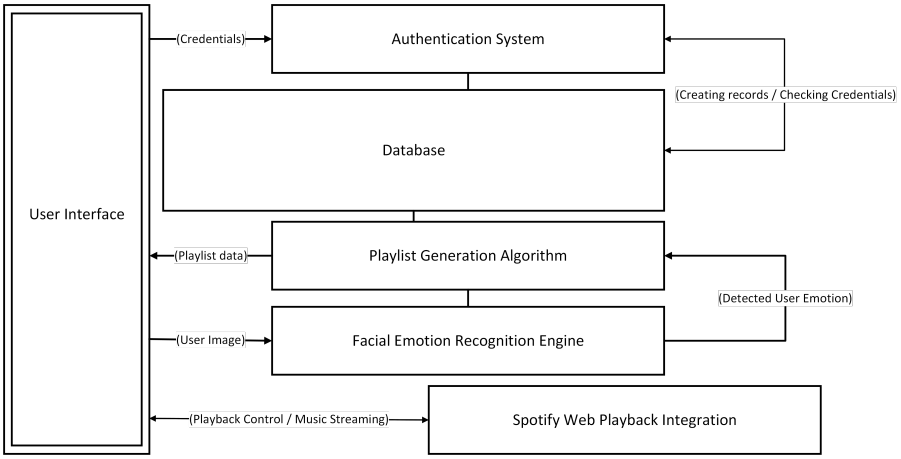


Figure 5.1: Block Diagram

From Figure 5.1, there are different components and each component has its own responsibilities to enable FER and playlist generation for music therapy. User Interface (UI) is the gateway through which users interact with the application. It captures user inputs such as credentials for the authentication process, frames for emotion recognition, and user actions for music playback controls.

Authentication System manages user identity verification and access control. When user provides credentials, either username or email, and password to this component, it will validates them against stored data in the database. Upon successful validation, users can access the full functionality of the services. This component also handles account creation, where new user details are stored securely in the database.

Database stores and manages all persistent data, including user credentials, profile information, and any data pertinent to playlist generation processes. It ensures data integrity and provides efficient access for other components. FER Engine utilizes advanced algorithms and machine learning models. This engine analyses captured

frame, which contains user's facial expression, to detect emotional states. The result of this analysis is then used to tailor the music playlist to the user's current emotional needs.

Playlist Generation Algorithm takes the detected emotion from the FER Engine and constructs a playlist that suits the identified mood and therapeutic requirements. It queries a database of songs, which stored internally, to select appropriate tracks. Then, Spotify Web Playback Integration acts as the music service interface. This component is responsible for fetching the actual music tracks from Spotify and controlling the playback within the application, such as playing, pausing, and etc. based on user input through the UI.

This diagram simplifies the conceptual understanding of the system and forms the architectural backbone of the application. It also outlines how user actions translate into system responses and result in a music therapy experience.

5.2.1.2. Use Case Diagram

Use Case diagram is a visual representation of the system's functionalities and the interactions that different types of users have with these functionalities. (IBM, 2021)

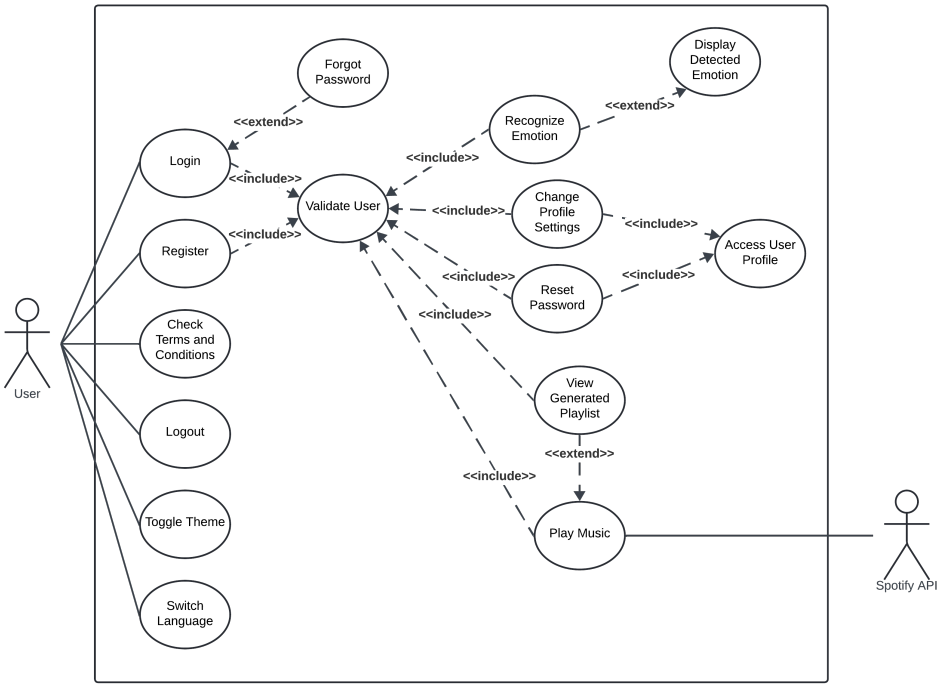


Figure 5.2: Use Case Diagram

In Figure 5.2, there are two actors, ‘User’ and the ‘Spotify API’. The ‘User’ represents any individual who interacts with the application for its services. The ‘Spotify API’ acts as an external system that the application communicates with to stream music.

The use case diagram delineates the extent of the application’s capability from the ‘User’ point of view by illustrating the user interactions. This helps to understand what the system does, but not how it does it, thus separating the ‘what’ from the ‘how’ in system functionalities. When a use case incorporates another’s behavior or expands the behavior under some circumstances, it is represented by dashed lines with arrows labelled ‘includes’ and ‘extends.’ For example, several use cases in the application such as ‘Recognize Emotion’, ‘Change Profile Settings’, and ‘Play Music’, require the ‘User’ to be authenticated. This precondition is captured in the ‘Validate User’ use case, which is included in these use cases to ensure that the ‘User’ is logged in before granting access to the services.

5.2.1.3. Sequence Diagrams

Sequence diagrams provide an illustrative view of how various components of a system work together ot accomplish a process, acting as blueprints of interaction across time.(Bell, 2004) They are especially helpful in understanding the flow of messages and actions between objects in response to software system events.

Figure 5.4 presents the series of interactions that commence with capturing a user’s facial expression and terminates when the system identifies the user’s emotional state. The sequence is outlined from the user interface to the backend services, where the input is processed by the ‘Emotion Recognition Engine’ to identify emotion. The detected emotion will then influence the music playlist generation.

Figure 5.5 shows how the service uses the detected emotion from Figure 5.4 to choose and assemble a series of songs into a coherent playlist. From retrieving suitable songs based on the emotional analysis to returning the playlist to the user for playback through Spotify Web Playback service, it demonstrates the cooperation between the system’s components.

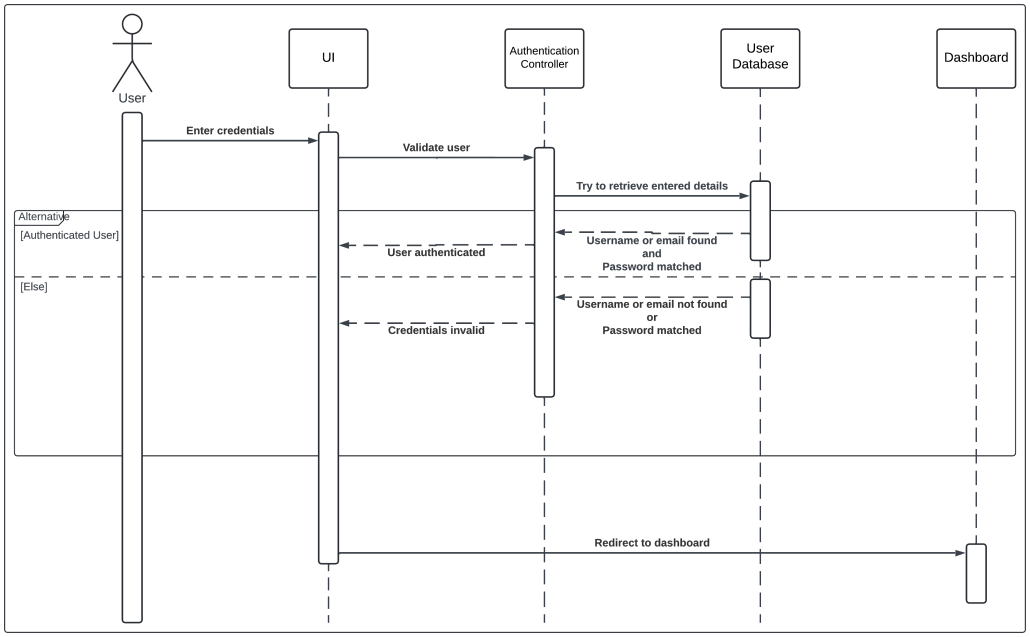


Figure 5.3: Login Sequence Diagram

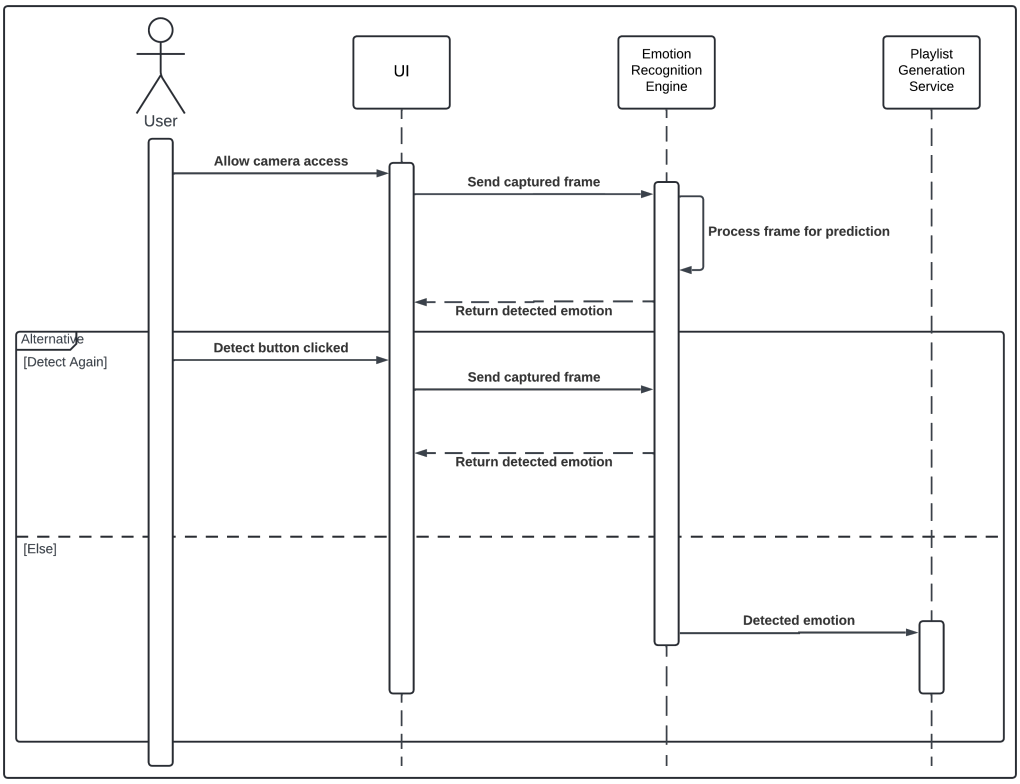


Figure 5.4: Emotion Recognition Sequence Diagram

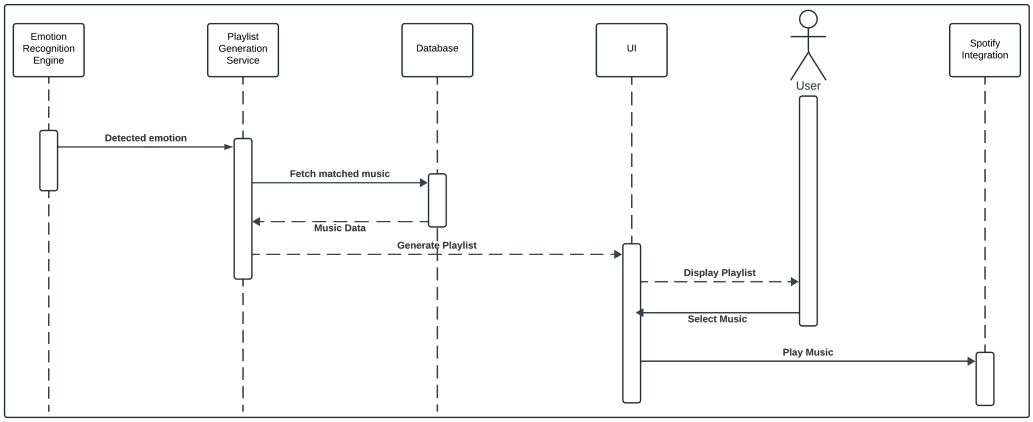


Figure 5.5: Playlist Generation Sequence Diagram

5.2.1.4. Flowchart

Flowchart represent the process of a system, the decisions that need to be made, and the flow of control from one step to the next. Troubleshooting and system design are made considerably easier with the aid of flowchart as they make the sequential phase and logic paths in complicated processes visible. (Lucidchart, 2019)

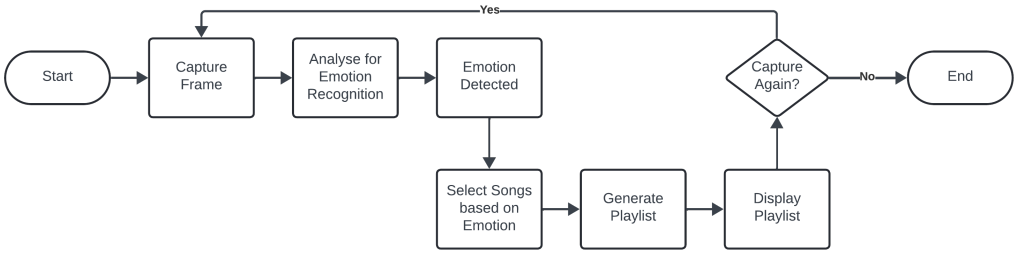


Figure 5.6: Flowchart For Emotion Recognition and Playlist Generation

Figure 5.6 shows the steps taken from the moment a user engages with the feature to capture their emotional state to the point where the system recognizes and outputs the detected emotion. Following the FER phase, the identified emotion is then used to select suitable music, creating playlist and presenting it to the user.

5.2.1.5. Entity-relationship Diagram

Entity-Relationship Diagram (ERD) is a structured representation of the data entities within the web application and the interconnections between them. This illustration outlines the database schema, which is foundational for storing, retrieving, and

managing data that the application operates upon.

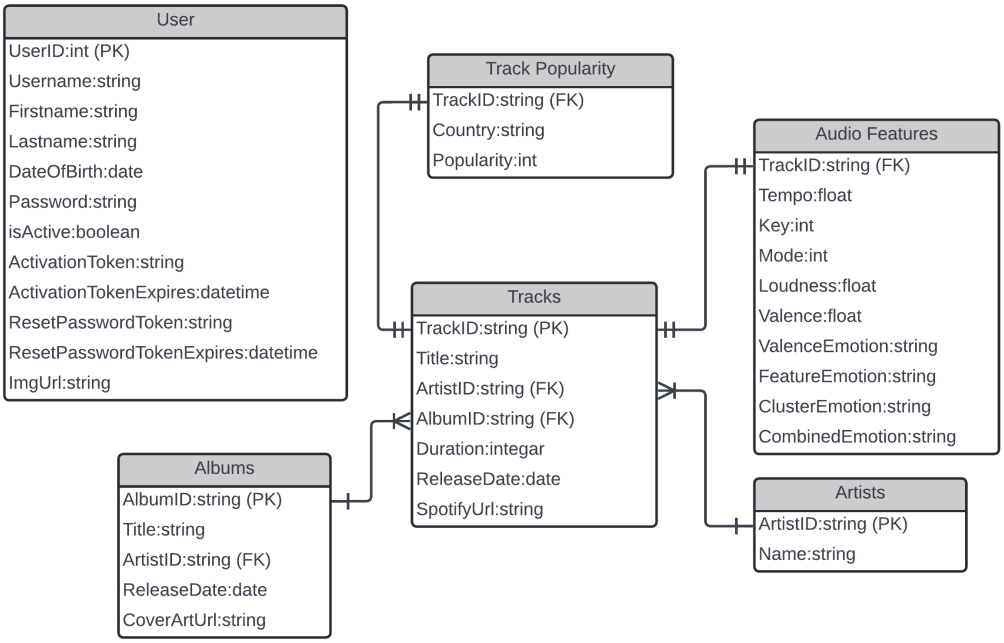


Figure 5.7: Entity-relationship Diagram

Figure 5.7 presents the ERD for the application. ‘User’ representing the application’s users, containing ‘UserID’, ‘Username’ and other personal details which could be modified in ‘User Settings Page’ (Figure 5.15). ‘Albums’, ‘Tracks’, ‘Audio Features’, ‘Track Popularity’, and ‘Artists’ are entities that stored music data from song title to information such as its tempo, its loudness and other audio features.

The relationship between entities are defined by Primary Key (PK) and Foreign Key (FK), enforcing referential integrity within the database. For example, a one-to-many relationship from ‘Artists’ to ‘Tracks’ indicates that one artist can have multiple tracks, whereas a one-to-one relationship between ‘Tracks’ and ‘Audio Features’ indicates that each track has its own unique features.

5.2.2. Logo Design

Figure 5.8a and Figure 5.9a is made up of stylized waves that reference the soothing rhythms of ocean waves while also visualizing the representation of an audio signal. This duality shows the fundamental purpose of the application, using the ability of music to evoke and influence emotions, providing a calming and therapeutic user

experience.

(a) Logo **(b)** Banner

Figure 5.8: Light Theme Logo

(a) Logo **(b)** Banner

Figure 5.9: Dark Theme Logo

Accompanying the logo, Figure 5.8b and Figure 5.9b incorporates the application's name, 'Sentirhy', which itself is a portmanteau derived from 'Sentiment' and 'Rhythm'. This naming convention shows the application's focus on sentiment analysis and rhythm.

5.2.3. Interface Design

UIs are created through the process of interface design. The arrangement of panels, as well as the style of components the user will interact with such as buttons, text fields and others are all included.

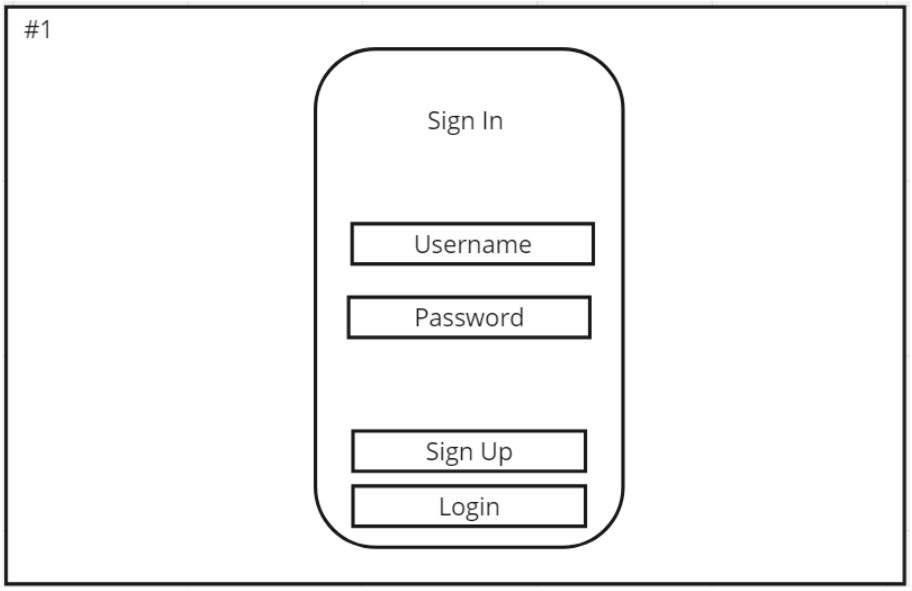


Figure 5.10: Login Page

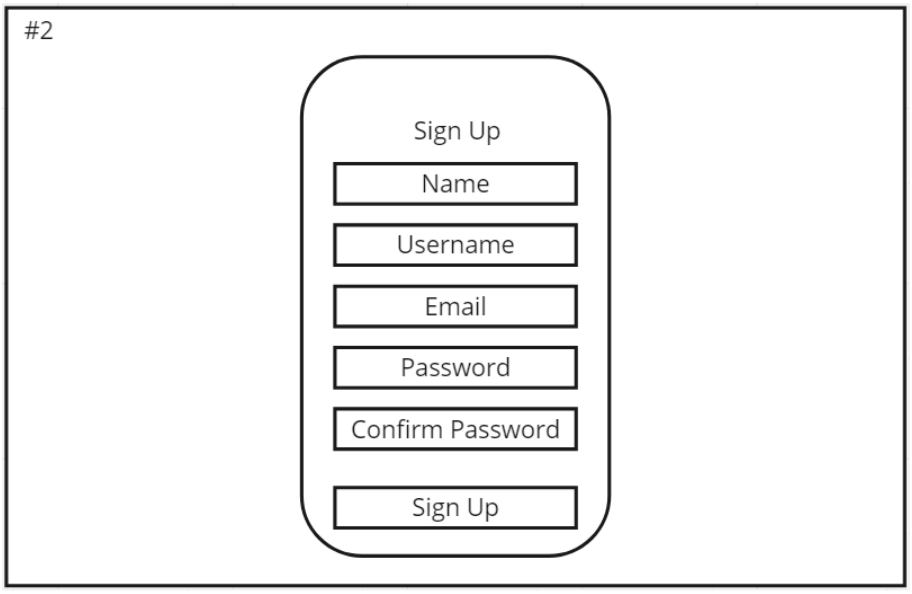


Figure 5.11: Sign Up Page

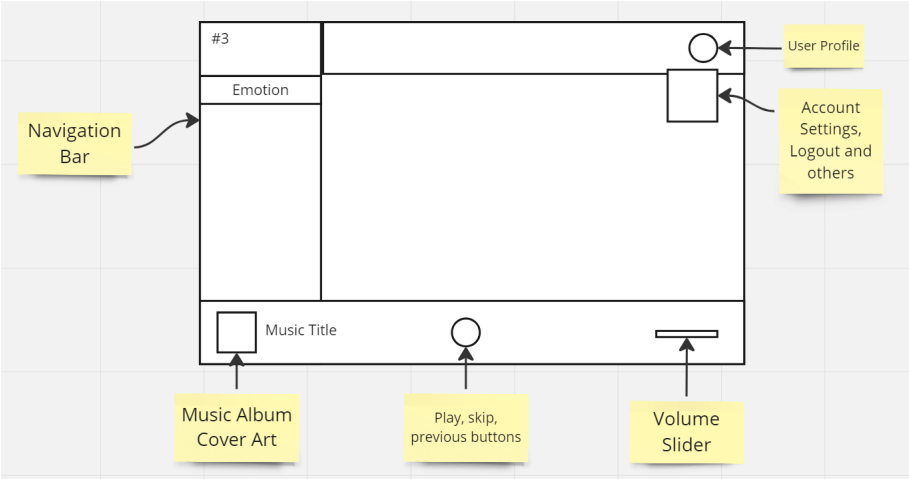


Figure 5.12: Dashboard Page

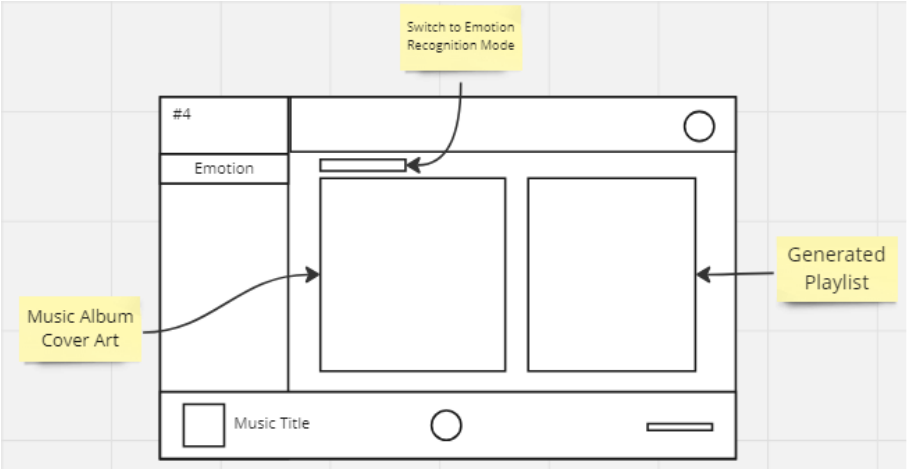


Figure 5.13: Emotion Music Page

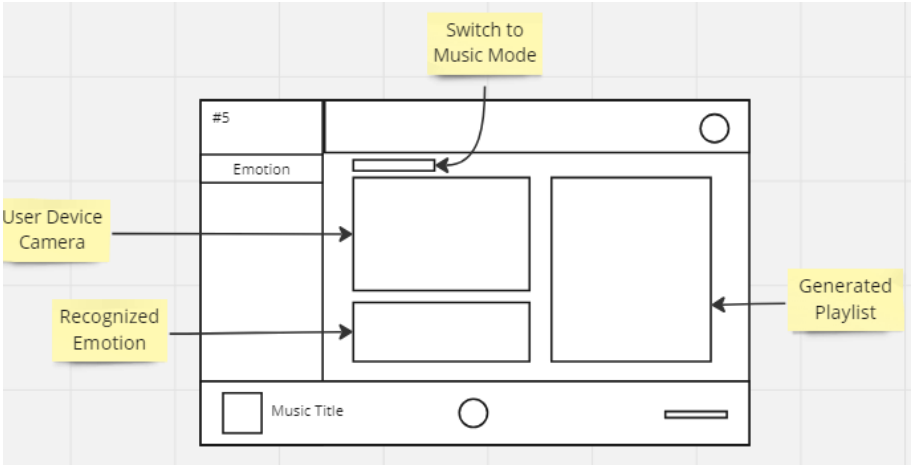


Figure 5.14: Emotion Recognition Page

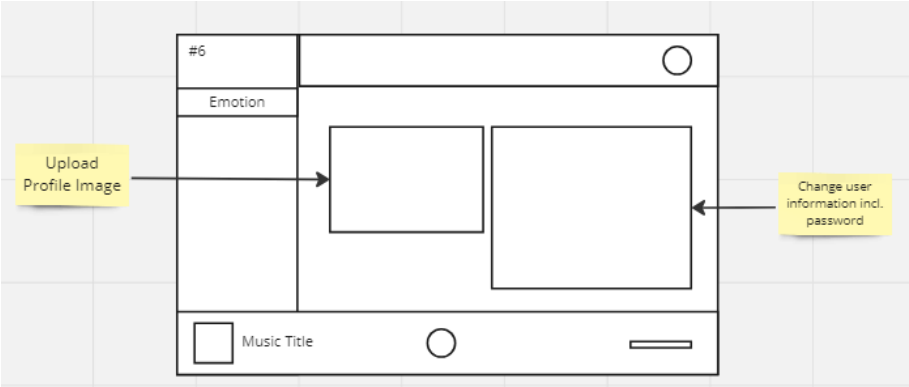


Figure 5.15: User Settings Page

5.3. ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

5.3.1. Models Architecture

In computer vision, a model's architecture is the basis that backs up its capacity for learning and generalization. The design decisions made when structuring a model determine not only its performance, but also its efficiency in extracting and understanding the complex patterns found in the data. A FER model must be able to manage the complexity of human facial emotions with ease. Therefore, the architectures explored and selected are designed with the goal of capturing a complex hierarchy of features.

Layer (Type)	Output Shape	Parameters
Input	(None, 48, 48, 1)	0
Conv2D (3x3x32)	(None, 48, 48, 32)	320
Conv2D (3x3x64)	(None, 48, 48, 64)	18,496
BatchNormalization	(None, 48, 48, 64)	256
MaxPooling2D (2x2)	(None, 24, 24, 64)	0
Dropout	(None, 24, 24, 64)	0
Conv2D (3x3x128)	(None, 24, 24, 128)	73,856
Conv2D (3x3x256)	(None, 22, 22, 256)	295,168
BatchNormalization	(None, 22, 22, 256)	1,024
MaxPooling2D (2x2)	(None, 11, 11, 256)	0
Dropout	(None, 11, 11, 256)	0
Flatten	(None, 30976)	0
Dense	(None, 1024)	31,720,448
Dropout	(None, 1024)	0
Dense (Softmax)	(None, 4)	4,100

Table 5.1: Detailed Architecture of the CNNs Model 1

Model 1 (Table 5.1) is the first attempt at tackling the challenging issue of FER, based

on the well-proven architectures similar to VGG-16 (Simonyan and Zisserman, 2015), which are intended to identify nad leverage the strengths of facial feature differentiation. The model started with a conservative number of filters in the Convolutional Layer (Conv2D), from 32 and incrementally increasing to 64, 128, and 256, to ensure the model is able to capture a spectrum of features from basic to complex. Those interspersed 'Dropout' layers are used to prevent over-fitting, which then giving the model greater generalization capabilities.

In order to reduce dimensionality, pooling layers are implemented throughout the architecture. This will compress the spatial volume of features before they are flattened into a format suitable for dense network processing. The learning process ends in the 'Dense' layers that, through a 'Softmax' activation, control the emotional categorization, converting the complex network of extracted features into perceptible emotional states.

Model 2 (Table 5.2) is a refined version of the FER model. With the addition of L2 regularization and a varied number of filter sizes, this model differs from the first model. This version provides theoretical performance improvements, such as higher generalization in unseen data. Additionally, this model version is more similar to the VGG-16 model; but, because of resource constraints, the original VGG-16 model, which in theory should yield better results, is not implemented in order to use fewer computer resources.

Layer (Type)	Output Shape	Parameters
Input	(None, 48, 48, 1)	0
Conv2D (3x3x64)	(None, 48, 48, 64)	640
BatchNormalization	(None, 48, 48, 64)	256
ReLU Activation	(None, 48, 48, 64)	0
Conv2D (3x3x64)	(None, 48, 48, 64)	36,928
BatchNormalization	(None, 48, 48, 64)	256
ReLU Activation	(None, 48, 48, 64)	0
MaxPooling2D (2x2, stride 2)	(None, 24, 24, 64)	0
Conv2D (3x3x128)	(None, 24, 24, 128)	73,856
BatchNormalization	(None, 24, 24, 128)	512
ReLU Activation	(None, 24, 24, 128)	0
Conv2D (3x3x128)	(None, 24, 24, 128)	147,584
BatchNormalization	(None, 24, 24, 128)	512
ReLU Activation	(None, 24, 24, 128)	0
MaxPooling2D (2x2, stride 2)	(None, 12, 12, 128)	0
Conv2D (3x3x256)	(None, 12, 12, 256)	295,168
BatchNormalization	(None, 12, 12, 256)	1,024
ReLU Activation	(None, 12, 12, 256)	0
Conv2D (3x3x256)	(None, 12, 12, 256)	590,080
BatchNormalization	(None, 12, 12, 256)	1,024
ReLU Activation	(None, 12, 12, 256)	0
Conv2D (3x3x256)	(None, 12, 12, 256)	590,080
BatchNormalization	(None, 12, 12, 256)	1,024
ReLU Activation	(None, 12, 12, 256)	0
MaxPooling2D (2x2, stride 2)	(None, 6, 6, 256)	0
Flatten	(None, 9216)	0
Dense	(None, 4096)	37,752,832
BatchNormalization	(None, 4096)	16,384
ReLU Activation	(None, 4096)	0
Dense	(None, 4096)	16,781,312
BatchNormalization	(None, 4096)	16,384
ReLU Activation	(None, 4096)	0
Dense (Softmax)	(None, 4)	16,388

Table 5.2: Detailed Architecture of the CNNs Model 2

6. IMPLEMENTATION

6.1. INTRODUCTION

This section covers how the research was put into practice, with a particular emphasis on how machine learning models were put into practice and how they were integrated into a working web application for FER.

6.2. ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

The preparation of datasets, the training environment, and the computational resources employed will be discussed in the following section. Providing specific details about the measures taken to guarantee that the data was suitable for learning, the augmentation methods used to improve model performance, and the training approaches used to maximize model performance. The evaluation metrics and the outcome of testing the models on hypothetical data are also included in this ML section.

6.2.1. Setup and Preparation

6.2.1.1. Environment Setup

The models are developed with a robust software environment designed for advanced ML tasks. The preferred programming language was Python (Python, 2019), which is well-known for its adaptability and power in data manipulation and machine learning. Then, the research and model training were carried out with Jupyter Notebook (Jupyter, 2019), which is an interactive computing environment that enables real-time code execution, analysis and visualization. The libraries used for model training are listed below.

- **Keras**: Chosen for its user-friendly API which operates on top of TensorFlow, it

enabled to build and train neural network models with relative ease. (Team, n.d.)

- **NumPy**: It is used for handling of numerical operations on array, forming the foundation of data structures in ML tasks. (Numpy, 2009)
- **Pandas**: This library provided robust data manipulation and cleaning capabilities which makes organizing tabular data and dataset transformation more easier. (Pandas, 2018)
- **Scikit-learn (sklearn)**: A broad range of machine learning tools, including cross-validation methods, model assessment, and preprocessing, are available in this library and aid in the improvement of learned models. (Scikit-learn, 2019)
- **OpenCV (cv2)**: This library provided extensive image processing capabilities, which were crucial in manipulating and preparing facial images for training. (OpenCV, 2019)
- **Matplotlib and Seaborn**: These libraries were used for data visualization. They allow findings to be converted into charts, which provide further insight into the functioning of the models. (Matplotlib, 2012) (seaborn, 2012)

6.2.1.2. Data Preparation

6.2.1.2.1 Data Collection

Figure 6.1: FER-2013 Dataset

A dataset capable of precisely capturing a wide range of human emotions through facial expression was needed for training the model. Therefore, the FER-2013 (Dumitru

et al., 2013) dataset which is a well-known benchmark in the field of FER from the Kaggle competition platform was chosen. A wide range of facial expressions are included in this dataset, and each of these grayscale images of faces are labelled with one of seven emotions. This makes it suited for training FER models.

Figure 6.2: CK+ Dataset

To increase the diversity of the data, CK+ (Lucey et al., 2010) dataset is included to the FER-2013. This dataset contains labelled facial expressions from varied populations and light scenarios. This addition was to provide the models a more comprehensive learning environment by exposing them to a greater variety of face emotions and characteristics.

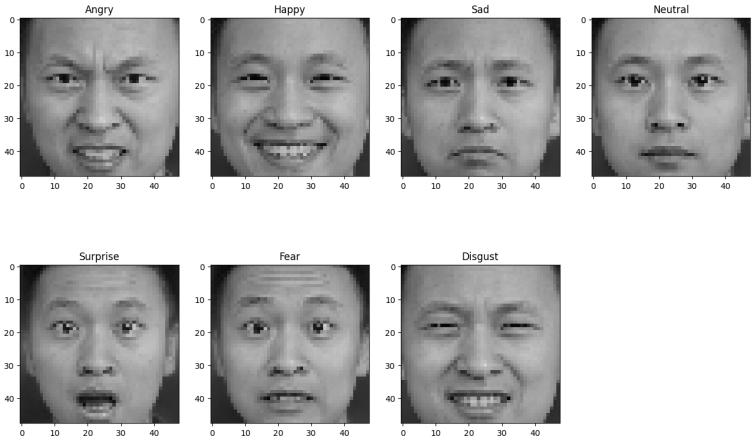


Figure 6.3: SZU-EmoDage Dataset

Moreover, SZU-EmoDage (Who, 2022) dataset was also included to the collection to ensure the models are capable of identifying facial expressions on Asian faces. With these additions, each dataset was integrated with a different set of challenges and

viewpoints, which then improve the model's accuracy and practicality in real-world situation.

6.2.1.2.2 Preprocessing Steps

Those collected data went through a series of preprocessing stages to standardize input features and optimize them for efficient ML before training. Even though the data were already in grayscale, all images were first converted to grayscale using the `cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)` function to ensure that all data were in grayscale in order to simplify the input data and reduce computational cost. After conversion, all images were shrunk to 48x48 pixels. It is because neural networks require consistent picture sizes in order to handle batch data efficiently and handle all inputs equally during the learning process.

Following resizing, images were flattened from a format of 2D arrays to a 1D array with 2304 elements (48x48). This transformation is necessary because machine learning algorithms generally require a flat array of features for each sample to be handle and analyse. The flattened image data was then compiled into a pandas DataFrame, which made data handling and analysis in Python simpler. Before training models, pixel values from each image were standardized to lie within the range of [0,1]. This normalization make sure that all input characteristics contribute equally to the model's learning because having different scale will influence the results.

In order to simplify the model and better match the project with its use in music recommendation, the number of emotion categories was cut down from seven to four. The four categories were kept were 'Neutral', 'Happy', 'Sad' and 'Angry'. These emotions are more easily applied and useful for determining musical mood.

6.2.1.2.3 Augmentation

Keras' function '`ImageDataGenerator()`' is used to construct a strategic data augmentation process that improved the model's robustness and allowed it to generalize across a variety of face emotions and situations. The augmentation techniques that were used included random rotation of images up to 10°. This mimics the natural tilts that occur in expressive moments. Additionally, width and height shifts, which translate images by up to 10% of their size in both directions. This helps with

situations when faces are not in the center of the frame.

To help the model learn to identify emotions from faces at different camera distances, random zooming was also applied to some of the images. Moreover, the images were arbitrarily rotated horizontally so that the model could be trained on mirror copies of faces, therefore double the range of face orientations the model saw in the training.

6.2.1.2.4 Dataset Splitting

A key phase in preparing for training and evaluation of ML models is splitting the dataset into training, validation, and testing sets. The training set was composed from the ‘Training’ data from FER-2013 and the complete CK+ dataset. This is to increase the model’s capacity to generalize across various demographic groups and emotional states by exposing it too a greater variety of facial expressions and environmental factors.

For validation, the ‘PublicTest’ subset of the FER-2013 was used. This collection is essential for adjusting the hyper-parameters of the model and for providing an unbiased evaluation of the model’s fit on the training dataset. The ‘PrivateTest’ subset of the FER-2013 was used to do the final evaluation of the model’s performance. In addition to the training, validation, and testing data, SZU-EmoDage dataset is used specifically for transfer learning purpose.

6.2.2. Training Process

In the training process, a common training framework was introduced for both models (Table 5.1 and Table 5.2) to ensure consistency in the methodological approach. But some specific adjustments to address the unique characteristics of each model is allowed. The training process started with a thorough grid search to optimize hyper-parameters including dropout rates, kernel sizes, convolutional filter counts, and dense layer unit counts. Both models shared the same parameter grid as Figure 6.4. This approach made it easier to explore the configuration space in an organized way, ensuring that each model was tuned to perform optimally.

After searching through the parameter grid (Appendix A), the process focuses on configuring each model’s architecture to best capture the nuances of facial expression for FER. Model 1 (Table 5.1) has a relatively simple architecture, which require careful

```
param_grid = {
    'conv_1_filters': [32, 64],
    'conv_2_filters': [64, 96, 128],
    'conv_3_filters': [128, 192, 224],
    'conv_4_filters': [128, 160, 224, 256],
    'conv_1_kernel': [3, 6, 8],
    'conv_2_kernel': [3, 5, 8],
    'conv_3_kernel': [3, 5, 8],
    'conv_4_kernel': [3, 5, 8],
    'dropout_1': [0.1, 0.2, 0.4],
    'dropout_2': [0.0, 0.2, 0.3],
    'dropout_3': [0.0, 0.2, 0.3, 0.4],
    'dense_units': [512, 768, 1024],
    'l1_reg': [0.01, 0.001, 0.0001],
    'optimizer': ['adam', 'sgd'],
    'learning_rate': [1e-2, 1e-3, 1e-4],
    'epochs': [10, 50, 100]
}
```

Figure 6.4: Parameter Grid

tuning if dropout rates and filter sizes to balance feature learning against the risk of over-fitting. Model 2 (Table 5.2) has more additional layers to capture a richer set of features before the final classification layer . But this led to a more thorough analysis required for the network's depth in relation to its performance on the validation set, which assisted in identifying when to add dropout and how to adjust the pooling layers.

```
es = EarlyStopping(monitor='val_accuracy', patience=5, restore_best_weights=True)
```

Figure 6.5: Early Stopping

To prevent over-fitting, early stopping mechanism, which configured as 'EarlyStopping' callback (Figure 6.5), is integrated into the training to monitor validation accuracy and automatically halt training if no improvement was detected for five consecutive epochs. Most important of all, the callback (Figure 6.5) was set to restore the weights form the epoch with the best validation accuracy, ensuring that the model maintained any progress made prior to the plateau. This approach allows to train the model as long as beneficial, in the meantime, avoiding over-fitting and maintaining the optimal state achieved during training.

```
steps_per_epoch=len(train_X) / batch_size
```

Figure 6.6: Steps per epoch

Furthermore, the '`steps_per_epoch`' (Figure 6.6) parameter was calculated to ensure that each epoch processed the entire dataset. With these techniques, early stopping and calculated epoch steps, the models were able to achieve a balance between computational prudence and strong learning. Consequently, the models were well-positioned for dependable performance in FER, having learned from training data efficiently and been prevented from the risk of over-fitting.

6.2.3. Model Evaluation

6.2.4. Model Comparison and Selection

Property	Model 1	Model 2
Input shape	48 x 48 x 1	48 x 48 x 1
Weight layers	6	10
Conv layers	4	7
Kernel size	3x3	3x3
Training params	32,113,028	56,303,556
Model size	-	-
Accuracy (%)	72.36	-

Table 6.1: Comparison of Model 1 with Model 2

6.3. WEB APPLICATION

The web application uses the PERN stack (Figure 6.7), which is an acronym for PostgreSQL (Group, 2019), Express (Foundation, 2017), React (Source, 2024) and Node.js (Node.js, 2023). This set of technologies provides a comprehensive end-to-end foundation for creating dynamic web application.

The frontend was built with React.js, a widely-used JavaScript (JS) UI framework,

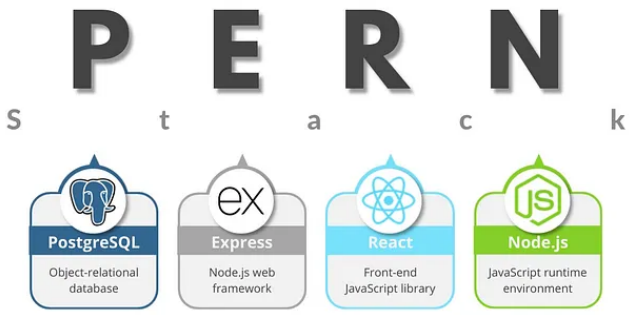


Figure 6.7: PERN Stack (Alves, 2023)

which was chosen for its declarative and effective method of creating UI. For the backend, Node.js was chosen for its non-blocking, event-driven architecture. This architecture is well-suited for managing workloads that involve asynchronous operations, real-time applications and I/O-bound tasks. The backend API is then constructed using Express.js, a simple and adaptable Node.js web application framework. PostgreSQL is the chosen database system as it is well-known for its advanced features, data integrity and robustness.

6.3.1. Login and Registration

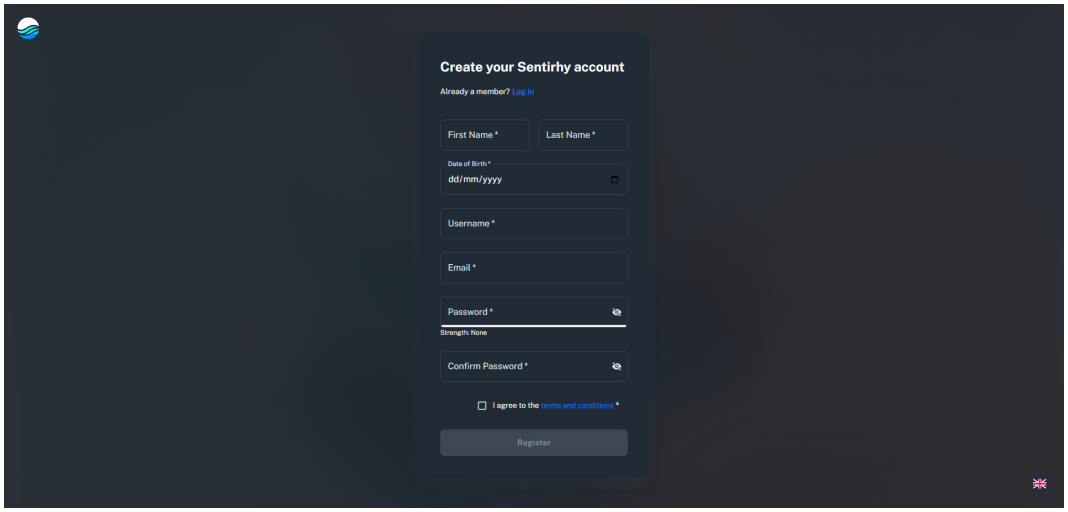


Figure 6.8: Register Page - UI

Users first provide their personal information, which consists of their date of birth, fist and last names, preferred username, and email address, to begin the registration process. (Figure 6.8) The entered data is then validated by cross-referencing it with existing entries in the ‘sentirhy.user’ table within the PostgreSQL database. If there is

duplicated username or email, the user will be notified that the credentials have already been taken and the process will stopped. When there is no duplicate entry, the user's password is securely hashed with 'bcrypt', and the crypto module sends an unique activation token to the user's email for account verification. The generated token with an expiration date is also stored in the database with the user credentials for security purposes.

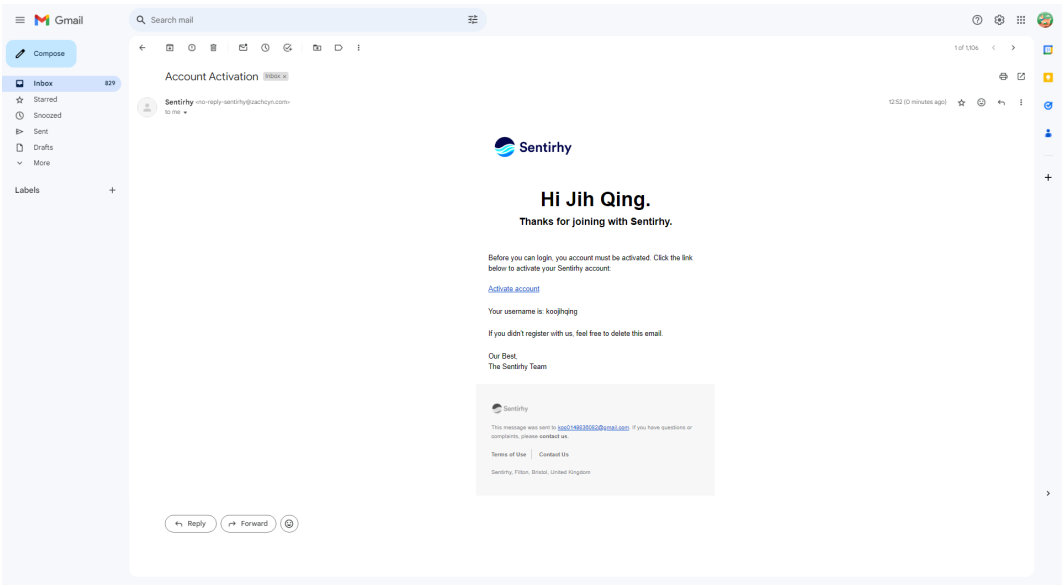


Figure 6.9: Account Activation Email - UI

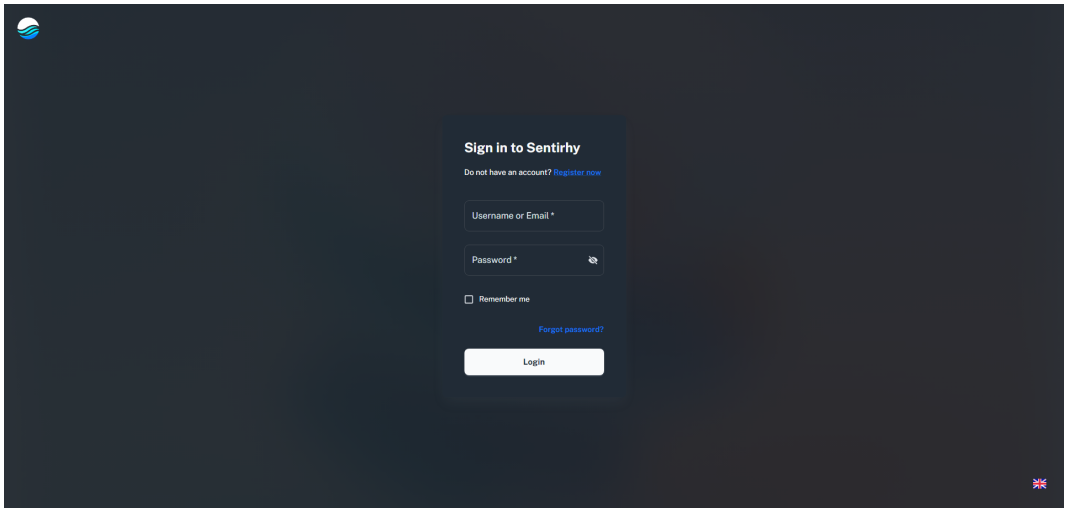


Figure 6.10: Login Page - UI

An activation email (Figure 6.9) is sent out using the 'nodemailer' after the new user's database entry. For the login process (Figure 6.10), the username and password

are required. After user submitted the credentials, the system compares those credentials to those kept in the database. Then, access to the application is granted with a successful match and an activated account.

6.3.2. Integration with Music Services

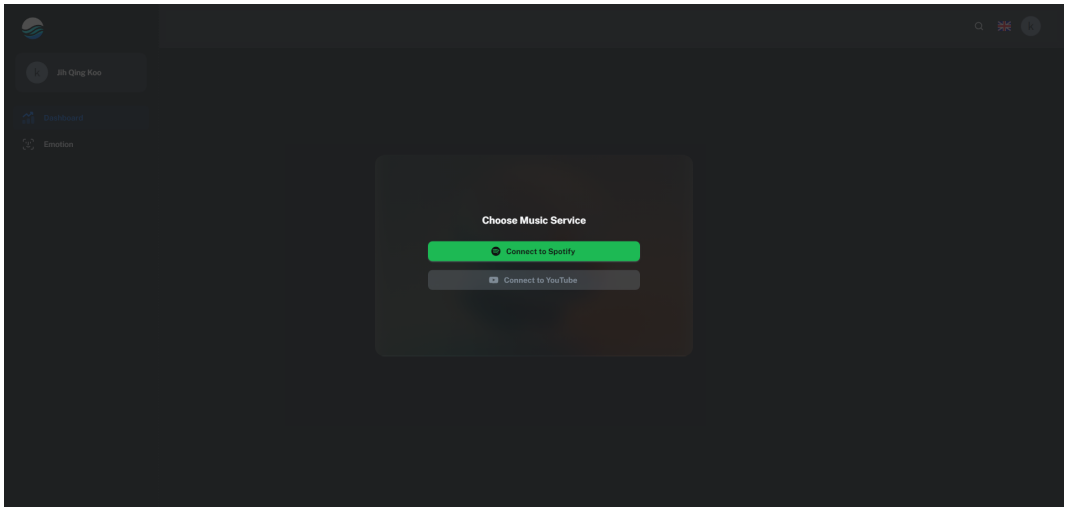


Figure 6.11: Music Services Connection - UI

As users log into the application, they are greeted with a dashboard. There will be a modal component (Figure 6.11) showing on the dashboard, allowing users to connect to different music services, which is required to the app's function to play music. Users are redirected to Spotify's authorization interface through an OAuth 2.0 authorization flow in order to grant permissions when they choose to link their accounts. Once user's permission is acquired, an access token from Spotify's token endpoint gives the application the authority to request services from the Spotify API on the user's behalf, like playing music. Unfortunately, as of right now, the only integration that is operational is Spotify; Youtube connectivity is indicated as a planned addition.

6.3.3. Emotion Detector

When user enters the Emotion Detection page (Figure 6.12), they encounter an interactive interface that requires access to the camera to proceed. With the user's permission, the application launches the Haarcascade frontal face identification algorithm

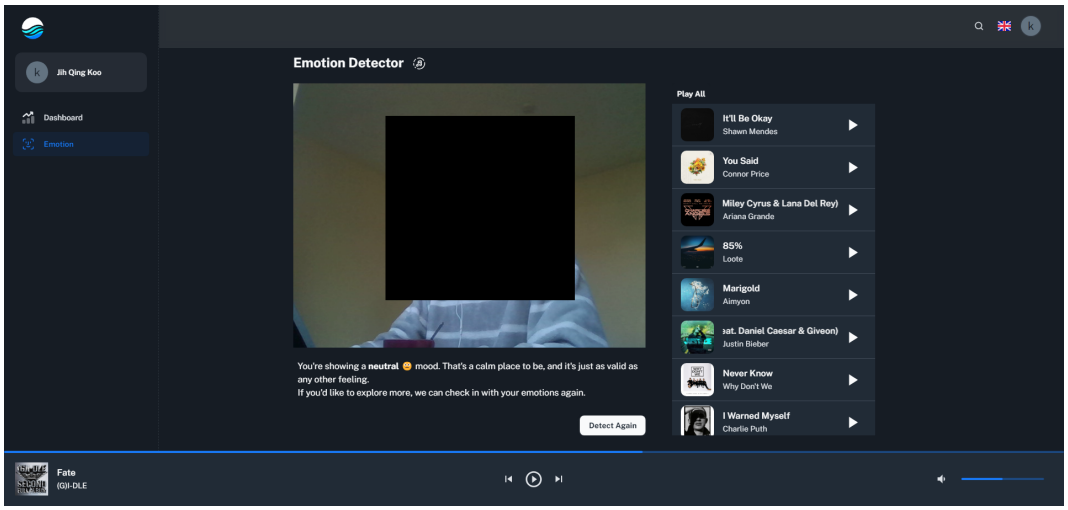


Figure 6.12: Emotion Detector - UI

and opencv.js, the JS library of OpenCV, enabling real-time face detection through the live video feed.

The system is designed to identify faces quickly and captures the frame. The

```
async function predictEmotion(imageFilename) {
    const imagePath = path.join(__dirname, '/user_emotion/', imageFilename);
    const model = await loadModel()
    const imageBuffer = await fs.promises.readFile(imagePath);
    const tensor = tf.node.decodeImage(imageBuffer).resizeBilinear([48,
    ↪  48]).mean(2).expandDims(-1).expandDims(0).toFloat().div(tf.scalar(255));
    const prediction = model.predict(tensor);
    const emotionIndex = prediction.argMax(1).dataSync()[0];

    return emotions[emotionIndex];
}
```

Figure 6.13: Load Model and Preprocess data

captured frame is then sent to the backend in a binary format appropriate for transmission over the network. The backend, which has been configured with TensorFlow.js, awaits the binary format data to perform its function. It loads the trained model, then resize the image to 48x48 pixels, converted to grayscale, normalized to scale the pixel values, and batched for the model's evaluation in order to meet the model's needs. (Figure 6.13)

When the image is in an acceptable format, the model starts to recognize the user's current emotional state. Once the emotion is deciphered, the system responds by creating playlist that corresponds with the feeling. (Figure 6.14) The generated playlist

```
function generateProgressivePlaylist(user_emotion_tracks, neutral_tracks,
↪ happy_tracks, playlistLength) {

    const allTracks = user_emotion_tracks.concat(neutral_tracks, happy_tracks);
    allTracks.sort((a, b) => a.valence - b.valence);
    const step = Math.floor(allTracks.length / playlistLength);

    const selectedTracks = [];
    for (let i = 0; i < playlistLength && (i * step) < allTracks.length; i++) {
        selectedTracks.push(allTracks[i * step]);
    }

    return selectedTracks;
}
```

Figure 6.14: Generate Playlist

is then dynamically displayed on the UI.

7. PROJECT EVALUATION

BIBLIOGRAPHY

, G. (2019), ‘Descending into ml: Training and loss | machine learning crash course’, https://developers.google.com/machine-learning/crash-course/descending-into-ml/training-and-loss.

, M. (2021), ‘Country music guide: History and sounds of country music’, https://www.masterclass.com/articles/country-music-guide.

, s.-l. (n.d.), ‘1.4. support vector machines’, https://scikit-learn.org/stable/modules/svm.html#:~:text=Support%20vector%20machines%20(SVMs)%20are.

Agrawal, A. and Mittal, N. (2019), ‘Using cnn for facial expression recognition: a study of the effects of kernel size and number of filters on accuracy’, *The Visual Computer* **36**, 405–412.

Alves, R. (2023), ‘Get started with the pern stack: an introduction and implementation guide’.
URL: *https://medium.com/@ritapalves/get-started-with-the-pern-stack-an-introduction-and-implementation-guide-e33c55d09994*

Athavle, M. (2021), ‘Music recommendation based on face emotion recognition’, *Journal of Informatics Electrical and Electronics Engineering (JIEEE)* **2**, 1–11.

BBC (2023), ‘History of the blues’, https://www.bbc.co.uk/bitesize/articles/zkbh2v4.

Beek, M. (2021), ‘What is jazz music?’, https://www.classical-music.com/articles/jazz-music-what-it-is-and-how-it-evolved.

Bell, D. (2004), ‘Explore the uml sequence diagram’.
URL: *https://developer.ibm.com/articles/the-sequence-diagram/*

Berwick, R. (n.d.), ‘An idiot’s guide to support vector machines (svms) r. berwick, village idiot svms: A new generation of learning algorithms’, https://web.mit.edu/6.034/wwwbob/svm-notes-long-08.pdf.

Boehm, B. (1986), ‘A spiral model of software development and enhancement’, *ACM SIGSOFT Software Engineering Notes* **11**, 22–42.

Britannica (2019), *country music | Definition, Artists, History, & Facts*.

Chaudhuri, L. (2022), ‘What is blues music?’, https://www.classical-music.com/articles/blues-music.

Communitcation Team, A. (2022), ‘Waterfall methodology a complete guide’, https://business.adobe.com/blog/basics/waterfall.

Crespo-Santiago, C. A. and Dávila-Cosme, S. d. l. C. (2022), ‘Waterfall method: A necessary tool for implementing library projects’, *HETS Online Journal* **1**(2), 81–92.

Dumitru, Goodfellow, I., Cukierski, W. and Bengio, Y. (2013), ‘Challenges in representation learning: Facial expression recognition challenge’.
URL: *https://kaggle.com/competitions/challenges-in-representation-learning-facial-expression-recognition-challenge*

Dunbar, R. I. M. (1998), ‘The social brain hypothesis’, *Evolutionary Anthropology: Issues, News, and Reviews* **6**(5), 178–190.

E R, S. (2021), ‘Random forest | introduction to random forest algorithm’, https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/.

Farley, P., Urban, E. and Mehrotra, N. (2023), ‘Face detection, attributes, and input data - face - azure ai services’, https://learn.microsoft.com/en-us/azure/ai-services/computer-vision/concept-face-detection.

Finkel, I. and Taylor, J. (2021), ‘How to write cuneiform’, https://www.britishmuseum.org/blog/how-write-cuneiform.

Foundation, O. (2017), ‘Express - node.js web application framework’.
URL: *https://expressjs.com/*

Freeman, J. (n.d.), ‘Block diagram | complete guide with examples’.
URL: *https://www.edrawsoft.com/block-diagram.html*

Frith, S. (2018), *Rock | music*.

Gabler, J. (2013), ‘What is classical music?’, https://www.yourclassical.org/story/2013/10/15/what-is-classical-music.

Gandhi, R. (2018), ‘Introduction to machine learning algorithms: Linear regression’, https://towardsdatascience.com/introduction-to-machine-learning-algorithms-linear-regression-14c4e325882a.

GeeksforGeeks (2018), ‘K-nearest neighbours - geeksforgeeks’, https://www.geeksforgeeks.org/k-nearest-neighbours/.

Group, T. P. G. D. (2019), ‘Postgresql: The world’s most advanced open source database’.
URL: *https://www.postgresql.org/*

Géron, A. (2019), *Hands-on machine learning with Scikit-Learn and TensorFlow concepts, tools, and techniques to build intelligent systems*, 2 edn, O’Reilly Media, Inc.

Harrison, O. (2018), ‘Machine learning basics with the k-nearest neighbors algorithm’, https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761.

Haykin, S., York, N., San, B., London, F., Sydney, T., Singapore, T., Mexico, M. and Munich, C. (2014), ‘Neural networks and learning machines third edition’, https://cours.etsmtl.ca/sys843/REFS/Books/ebook_Haykin09.pdf.

Hiller, L. (2019), *electronic music | Definition, History, & Facts*.

IBM (2021), ‘Use-case diagrams’.
URL: *https://www.ibm.com/docs/en/rational-soft-arch/9.6.1?topic=diagrams-use-case*

IBM (2022), ‘About linear regression | ibm’, https://www.ibm.com/topics/linear-regression#:~:text=Resources-.

IBM (2023*a*), 'What are convolutional neural networks? | ibm', https://www.ibm.com/topics/convolutional-neural-networks.

IBM (2023*b*), 'What is random forest? | ibm', https://www.ibm.com/topics/random-forest#:~:text=Random%20forest%20is%20a%20commonly.

IBM (n.d.*a*), 'What is gradient descent? | ibm', https://www.ibm.com/topics/gradient-descent.

IBM (n.d.*b*), 'What is the k-nearest neighbors algorithm? | ibm', https://www.ibm.com/topics/knn#:~:text=Next%20steps-.

Jupyter (2019), 'Project jupyter'.
URL: *https://jupyter.org/*

Juslin, P. N. and Sloboda, J. A. (2013), 'Music and emotion', *The Psychology of Music* pp. 583–645.

Killin, A. (2018), 'The origins of music', *Music & Science* **1**, 205920431775197.

Kumar Pal, S. (2018), 'Software engineering | spiral model', https://www.geeksforgeeks.org/software-engineering-spiral-model/.

Kwiatkowski, R. (2021), 'Gradient descent algorithm — a deep dive', https://towardsdatascience.com/gradient-descent-algorithm-a-deep-dive-cf04e8115f21.

Laoyan, S. (2022), 'What is agile methodology? (a beginner's guide)', https://asana.com/resources/agile-methodology. Accessed: 2023-04-09.

Lucey, P., Cohn, J. F., Kanade, T., Saragih, J., Ambadar, Z. and Matthews, I. (2010), 'The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression', pp. 94–101.

Lucidchart (2019), 'What is a flowchart'.
URL: *https://www.lucidchart.com/pages/what-is-a-flowchart-tutorial*

Mali, K. (2021), 'Linear regression | everything you need to know about linear regression', https://www.analyticsvidhya.com/blog/2021/10/everything-you-need-to-know-about-linear-regression/.

Mandal, M. (2021), ‘Cnn for deep learning | convolutional neural networks (cnn)’, https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/.

Matplotlib (2012), ‘Matplotlib: Python plotting - matplotlib 3.1.1 documentation’.
URL: *https://matplotlib.org/*

McCollum, S. (2019), ‘Hip-hop: A culture of vision and voice’, https://www.kennedy-center.org/education/resources-for-educators/classroom-resources/media-and-interactives/media/hip-hop/hip-hop-a-culture-of-vision-and-voice/.

Mellouk, W. and Handouzi, W. (2020), ‘Facial emotion recognition using deep learning: review and insights’, *Procedia Computer Science* **175**, 689–694.

Mishra, M. (2020), ‘Convolutional neural networks, explained’, https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939.

Munasinghe, M. I. N. P. (2018), ‘Facial expression recognition using facial landmarks and random forest classifier’, *2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)* .

Naseem, I., Togneri, R. and Bennamoun, M. (2010), ‘Linear regression for face recognition’, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **32**, 2106–2112.

Node.js (2023), ‘Node.js’.
URL: *https://nodejs.org/en*

Numpy (2009), ‘Numpy’.
URL: *https://numpy.org/*

OpenCV (2019), ‘Opencv library’.
URL: *https://opencv.org/*

Pandas (2018), ‘Python data analysis library’.
URL: *https://pandas.pydata.org/*

Pentreath, R. (2022), ‘Why do we call classical music ’classical music’?’, https://www.classicfm.com/discover-music/music-theory/why-do-we-call-it-classical-music/.

Pereira, C. S., Teixeira, J., Figueiredo, P., Xavier, J., Castro, S. L. and Brattico, E. (2011), ‘Music and emotions in the brain: Familiarity matters’, *PLoS ONE* **6**, e27241.

Porter, L. and Molana-Allen, L. (2018), ‘Did syria create the world’s first song?’, https://www.bbc.com/travel/article/20180424-did-syria-create-the-worlds-first-song.

Python (2019), ‘Python’.
URL: *https://www.python.org/*

S, M. C. D., Iram, S., Bhat, R., Supritha, L. and Leelavathi, S. (2023), ‘Music recommendation based on facial emotion recognition’, *nternational journal of advanced research in computer and communication engineering* **12**.

Saha, S. (2018), ‘A comprehensive guide to convolutional neural networks - the eli5 way’, https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

Saini, A. (2021), ‘Support vector machine(svm): A complete guide for beginners’, https://www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/.

Scikit-learn (2018), ‘sklearn.ensemble.randomforestclassifier — scikit-learn 0.20.3 documentation’, https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.

Scikit-learn (2019), ‘scikit-learn: machine learning in python’.
URL: *https://scikit-learn.org/stable/*

seaborn (2012), ‘seaborn: Statistical data visualization - seaborn 0.9.0 documentation’.
URL: *https://seaborn.pydata.org/*

Simonyan, K. and Zisserman, A. (2015), ‘Very deep convolutional networks for large-scale image recognition’.

Source, M. O. (2024), ‘React’.
URL: *https://react.dev/*

Srivastava, T. (2019), ‘Introduction to knn, k-nearest neighbors : Simplified’, https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/.

Tarnowski, P., Kołodziej, M., Majkowski, A. and Rak, R. J. (2017), ‘Emotion recognition using facial expressions’, *Procedia Computer Science* **108**, 1175–1184.

Tate, G. and Light, A. (2019), *hip-hop | Definition, History, Culture, & Facts*.

Team, K. (n.d.), ‘Keras documentation: Keras api reference’.
URL: *https://keras.io/api/*

Vemou, K., Horvath, A. and Zerdick, T. (2021), ‘Facial emotion recognition’, https://edps.europa.eu/system/files/2021-05/21-05-26_techdispatch-facial-emotion-recognition_ref_en.pdf.

Who (2022), ‘A chinese face dataset with dynamic expressions and diverse ages synthesized by deep learning’, *osf.io* .
URL: *https://osf.io/7a5fs/*

Xia, L. (2014), ‘Facial expression recognition based on svm’, *2014 7th International Conference on Intelligent Computation Technology and Automation* .

Yamashita, R., Nishio, M., Do, R. K. G. and Togashi, K. (2018), ‘Convolutional neural networks: an overview and application in radiology’, *Insights into Imaging* **9**, 611–629.

Yiu, T. (2019), ‘Understanding random forest’, https://towardsdatascience.com/understanding-random-forest-58381e0602d2.

APPENDICES

Appendix A.

Appendix B.

```
2023-12-27 23:22:26.546868:
(32, 64, 128, 128, 3, 3, 3, 3, 0.1, 0.0, 0.0, 512, 'adam', 32, 10) CHECKED
Best Score: 0.5725693106651306
Best Params: {'conv_1_filters': 32, 'conv_2_filters': 64, 'conv_3_filters': 128, 'conv_4_filters': 128, 'conv_1_kernel': 3, 'conv_2_kernel': 3, 'conv_3_kernel': 3, 'conv_4_kernel': 3, 'dropout_1': 0.1, 'dropout_2': 0.0, 'dropout_3': 0.0, 'dense_units': 512, 'optimizer': 'adam'}
(32, 64, 128, 128, 3, 3, 3, 3, 0.1, 0.0, 0.0, 768, 'adam', 32, 10) CHECKED
Best Score: 0.5725693106651306
Best Params: {'conv_1_filters': 32, 'conv_2_filters': 64, 'conv_3_filters': 128, 'conv_4_filters': 128, 'conv_1_kernel': 3, 'conv_2_kernel': 3, 'conv_3_kernel': 3, 'conv_4_kernel': 3, 'dropout_1': 0.1, 'dropout_2': 0.0, 'dropout_3': 0.0, 'dense_units': 512, 'optimizer': 'adam'}
2023-12-27 23:24:26.831029:
(32, 64, 128, 128, 3, 3, 3, 3, 0.1, 0.0, 0.0, 1024, 'adam', 32, 10) CHECKED
Best Score: 0.5765619874000549
Best Params: {'conv_1_filters': 32, 'conv_2_filters': 64, 'conv_3_filters': 128, 'conv_4_filters': 128, 'conv_1_kernel': 3, 'conv_2_kernel': 3, 'conv_3_kernel': 3, 'conv_4_kernel': 3, 'dropout_1': 0.1, 'dropout_2': 0.0, 'dropout_3': 0.0, 'dense_units': 1024, 'optimizer': 'adam'}
2023-12-27 23:25:37.281540:
(32, 64, 128, 128, 3, 3, 3, 3, 0.1, 0.0, 0.2, 512, 'adam', 32, 10) CHECKED
Best Score: 0.5906981825828552
Best Params: {'conv_1_filters': 32, 'conv_2_filters': 64, 'conv_3_filters': 128, 'conv_4_filters': 128, 'conv_1_kernel': 3, 'conv_2_kernel': 3, 'conv_3_kernel': 3, 'conv_4_kernel': 3, 'dropout_1': 0.1, 'dropout_2': 0.0, 'dropout_3': 0.2, 'dense_units': 512, 'optimizer': 'adam'}
2023-12-27 23:26:47.249360:
(32, 64, 128, 128, 3, 3, 3, 3, 0.1, 0.0, 0.2, 768, 'adam', 32, 10) CHECKED
Best Score: 0.5906981825828552
Best Params: {'conv_1_filters': 32, 'conv_2_filters': 64, 'conv_3_filters': 128, 'conv_4_filters': 128, 'conv_1_kernel': 3, 'conv_2_kernel': 3, 'conv_3_kernel': 3, 'conv_4_kernel': 3, 'dropout_1': 0.1, 'dropout_2': 0.0, 'dropout_3': 0.2, 'dense_units': 512, 'optimizer': 'adam'}
2023-12-27 23:27:57.635618:
(32, 64, 128, 128, 3, 3, 3, 3, 0.1, 0.0, 0.2, 1024, 'adam', 32, 10) CHECKED
Best Score: 0.5906981825828552
Best Params: {'conv_1_filters': 32, 'conv_2_filters': 64, 'conv_3_filters': 128, 'conv_4_filters': 128, 'conv_1_kernel': 3, 'conv_2_kernel': 3, 'conv_3_kernel': 3, 'conv_4_kernel': 3, 'dropout_1': 0.1, 'dropout_2': 0.0, 'dropout_3': 0.2, 'dense_units': 512, 'optimizer': 'adam'}
2023-12-27 23:29:09.047680:
(32, 64, 128, 128, 3, 3, 3, 3, 0.1, 0.0, 0.3, 512, 'adam', 32, 10) CHECKED
Best Score: 0.5906981825828552
```

```
Best Params: {'conv_1_filters': 32, 'conv_2_filters': 64, 'conv_3_filters': 128, 'conv_4_filters': 128, 'conv_1_kernel': 3, 'conv_2_kernel': 3, 'conv_3_kernel': 3, 'conv_4_kernel': 3, 'dropout_1': 0.1, 'dropout_2': 0.0, 'dropout_3': 0.2, 'dense_units': 512, 'optimizer': 'adam'}
2023-12-27 23:30:18.471026:
(32, 64, 128, 128, 3, 3, 3, 3, 0.1, 0.0, 0.3, 768, 'adam', 32, 10) CHECKED
Best Score: 0.5906981825828552
Best Params: {'conv_1_filters': 32, 'conv_2_filters': 64, 'conv_3_filters': 128, 'conv_4_filters': 128, 'conv_1_kernel': 3, 'conv_2_kernel': 3, 'conv_3_kernel': 3, 'conv_4_kernel': 3, 'dropout_1': 0.1, 'dropout_2': 0.0, 'dropout_3': 0.2, 'dense_units': 512, 'optimizer': 'adam'}
```

Appendix C.

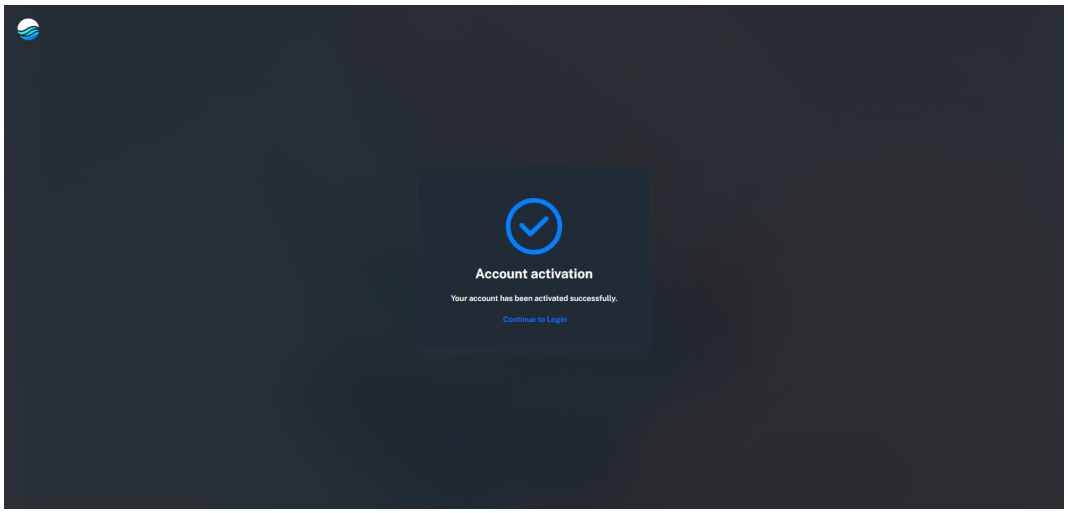


Figure .1: Account Activated - UI

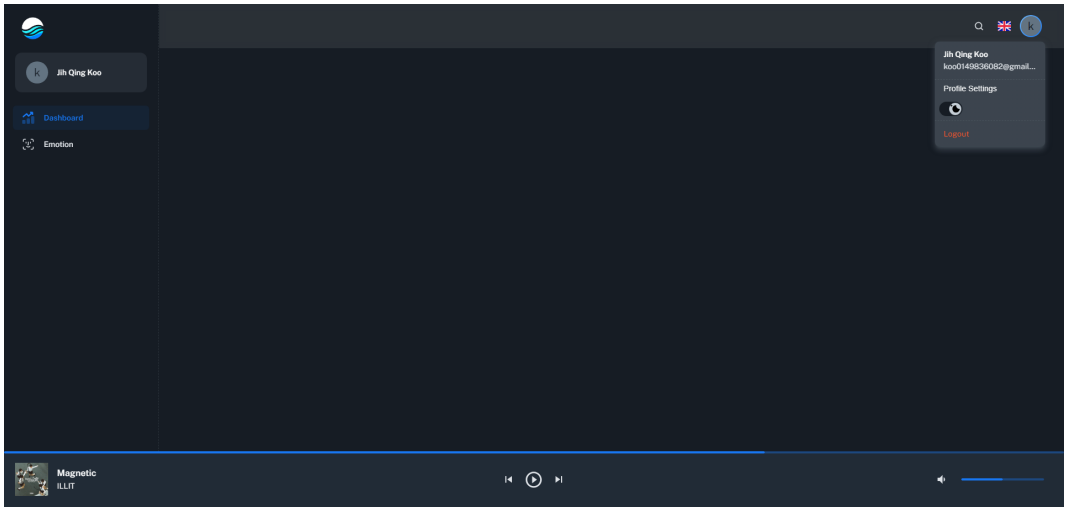


Figure .2: Dashboard - UI

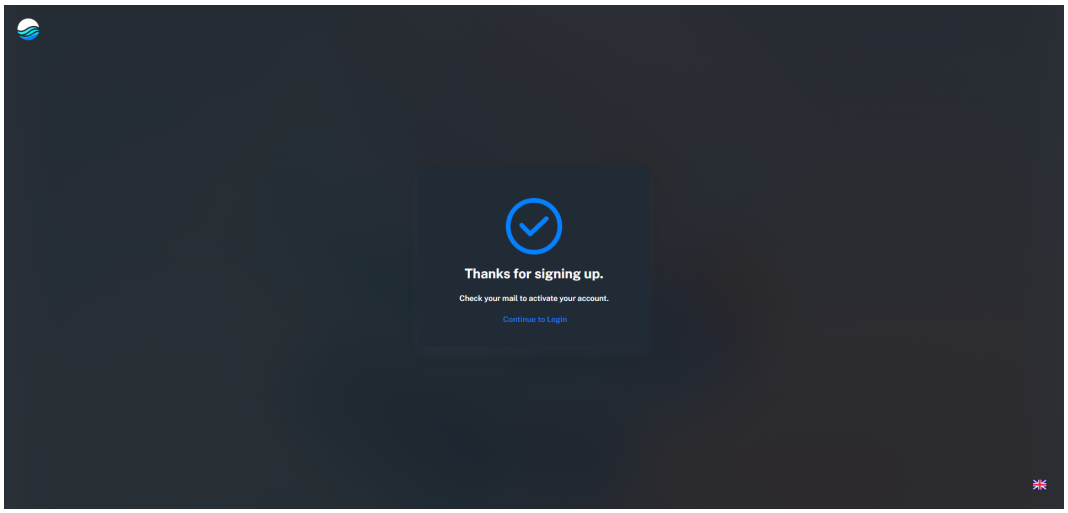


Figure .3: Register Successful - UI

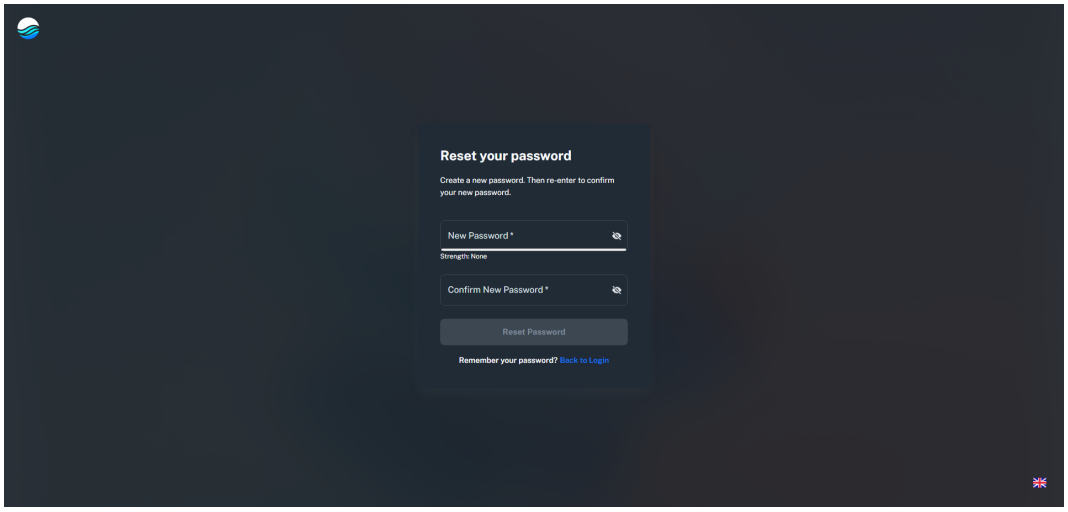


Figure .4: Reset Password - UI

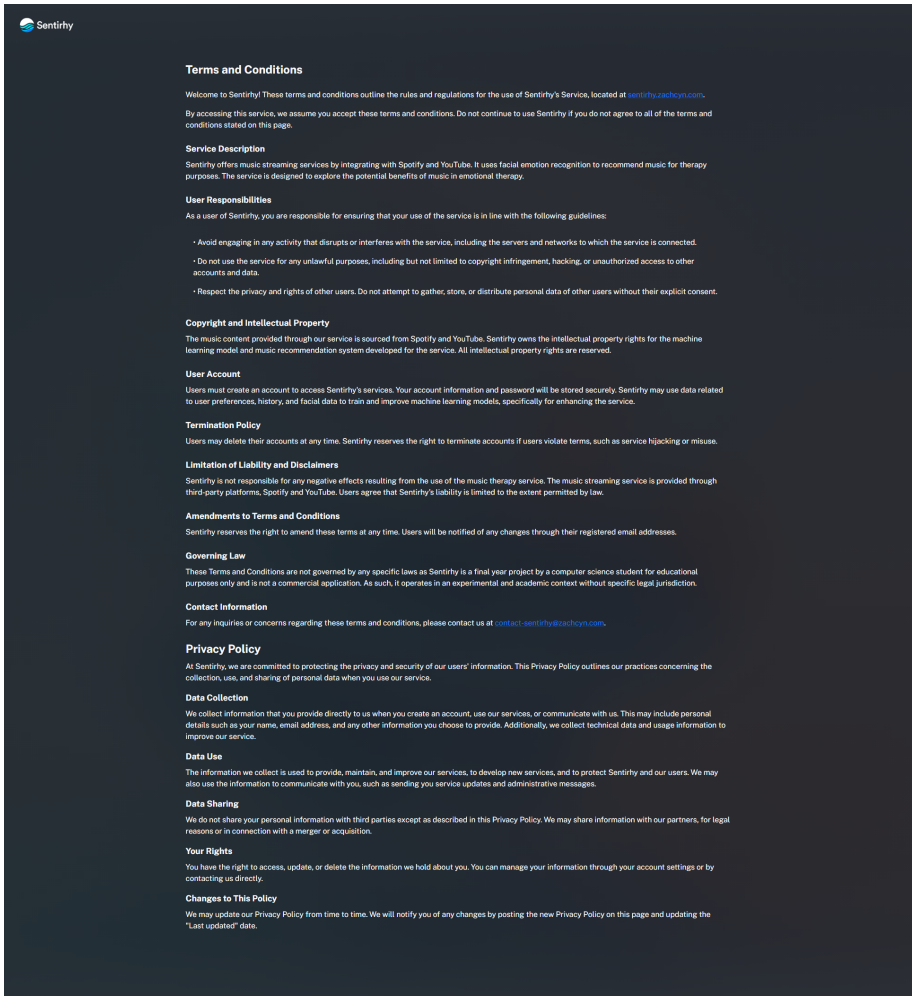


Figure .5: Terms and Conditions - UI

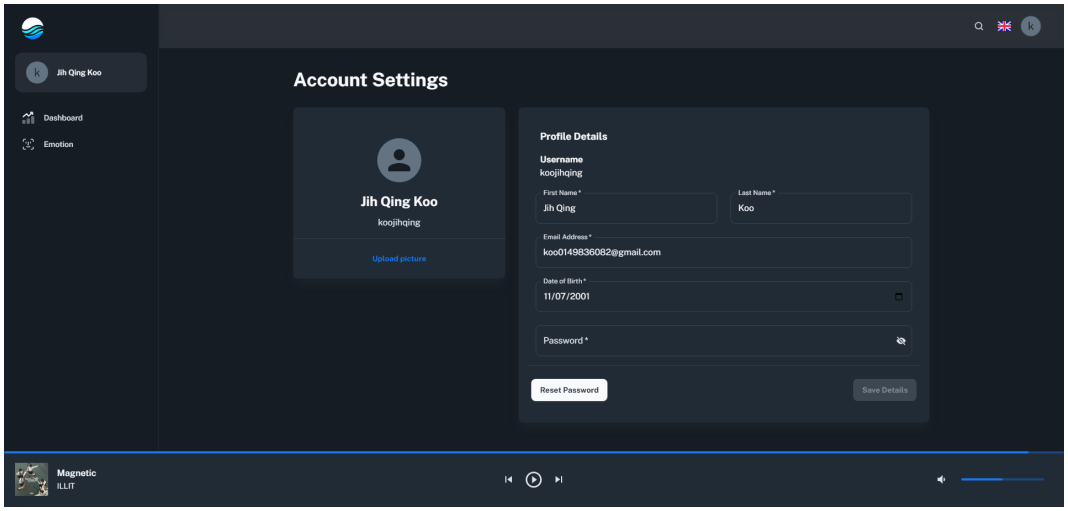


Figure .6: User Settings - UI

Appendix D.

Test No.	Cats.	Intention	Expected Result	Actual Result	Pass/Fail
1	UI	Registration	User registered	User registered and information added to database	Pass
2	UI	Login	User logged in	User logged in successfully	Pass
3	UI	Incorrect credentials upon login	User login failed and error message display	Error message displayed and user unable to login	Pass
4	UI	Account Activation	User received email and activate account from link	User activate account successfully from link	Pass
5	UI	Reset Password without Authentication	Reset link sent to user, and password reset successfully	Link sent, password reset successfully	Pass
6	UI	Connect Spotify API	Redirect user to Spotify Authentication Page, and redirect back to ‘Dashboard’	Granted permission from Spotify and brought user back to ‘Dashboard’	Pass
7	UI	Connect Youtube API	Redirect user to Youtube Authentication Page, and redirect back to ‘Dashboard’	Youtube API connection not implemented	Fail

Test No.	Cats.	Intention	Expected Result	Actual Result	Pass/Fail
8	UI	Play Music	Music play through Web Playback	Music played	Pass
9	UI	Skip Music	Skip music through Web Playback	Music skipped	Pass
10	UI	Previous Music	Play previous music through Web Playback	Previous Music played	Pass
11	UI	Adjust Volume	Volume higher or lower	Volume does adjusted through the slider	Pass
12	UI	Mute	Mute Web Playback	Music does muted	Pass
13	UI	Toggle light/dark theme	Theme switched when user pressed the button	Theme switched	Pass
14	UI	Detect user's system theme	Theme detected and switched to user's system theme	Theme switched accordingly	Pass
15	UI	Switch language	Web application language switched to user's preferred language	Language switched	Pass
16	UI	Change personal details in Settings	Update entered details to database	User's information is updated	Pass

Test No.	Cats.	Intention	Expected Result	Actual Result	Pass/Fail
17	UI	Change profile picture in Settings	Saved image to server and update the file path to database	Image saved and database updated	Pass
18	UI	Reset Password in Settings	Update new password to database	New password saved in database with encrypted	Pass
19	UI	Capture user's frame when face detected	Frame captured	Frame Captured with face showed clearly	Pass
20	ML	Emotion Detection	Emotion detected on the captured frame	Emotion detected	Pass
21	UI	Display recognized emotion on UI	Detected emotion is showed on user's screen	Detected emotion showing on user's screen	Pass
22	Server	Generate playlist	Generate playlist based on user's current emotion	Playlist generated based on user's current emotion	Pass
23	UI	Logout	User logged out, authentication token removed	Token removed and user logged out	Pass

Table 1: Test Table

Appendix E.

Date	Time	Location	Title	Attendees
16-10-2023	1505-1700	X Block	Project Idea and Aim Discussion	Yie Nian Chu, Ali Suhail, Martin Serpell
17-10-2023	1500-1515	2Q17	Project Idea and Aim Discussion	Yie Nian Chu, Craig Duffy
23-10-2023	1505-1605	X Block	Project Idea and Aim Disucssion	Yie Nian Chu, Ali Suhail, Martin Serpell
25-10-2023	1400-1430	Microsoft Teams	Final Decision on Project Idea	Yie Nian Chu, Craig Duffy
30-10-2023	1505-1520	X Block	Project Discussion	Yie Nian Chu, Ali Suhail, Martin Serpell
22-11-2023	1000-1030	Microsoft Teams	Progress Follow Up (Lit. Review)	Yie Nian Chu, Craig Duffy
06-12-2023	1000-1020	Microsoft Teams	Progress Follow Up (Lit. Review)	Yie Nian Chu, Craig Duffy
12-12-2023	1330-1400	2Q17	Development Discussion	Yie Nian Chu, Craig Duffy
16-01-2024	1500-1530	Microsoft Teams	Project Poster Discussion	Yie Nian Chu, Joseph Cauvy-Foster, Craig Duffy
06-02-2024	1600-1630	2Q17	Progress Follow Up (Dev)	Yie Nian Chu, Craig Duffy
05-03-2024	1200-1215	2Q17	Progress Follow Up (Dev)	Yie Nian Chu, Craig Duffy

Table 2: Meeting Log