Speech Emotion Detection

Submitted By

Arisa Asif

Mehveen Zafar

2020-24

Institute of Computer Science

Khwaja Fareed University of Engineering & Information Technology

Rahim Yar Khan

2024

Speech Emotion Recognition

Submitted to
Dr. Muhammad Shadab Alam Hashmi

Institute of Computer Science

In partial fulfilment of the requirements
For the degree of

Bachelor's in Computer Science

By

Arisa Asif

Cosc201101092

Mehveen Zafar

Cosc201101082

2020-24

Khwaja Fareed University of Engineering & Information Technology
Rahim Yar Khan
2024

DECLARATION

I/We hereby declare that this project report is based on my/our original work except for citations and quotations which have been duly acknowledged. I/We also declare that it has not been previously and concurrently submitted for any other degree or award at Khwaja Fareed University of engineering & Information Technology or other institutions.

Reg No	: _____	Reg No	: _____
Name	: _____	Name	: _____
Signature	: _____	Signature	: _____
Date	: _____	Date	: _____

APPROVAL FOR SUBMISSION

I certify that this project report entitled ***Speech Emotion Detection*** was prepared by **Arisa Asif** and **Mehveen Zafar** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor's in Computer Science at Khwaja Fareed University of Engineering & Information Technology.

Approved by:

Signature : _____

Supervisor : Dr. Muhammad Shadab Alam Hashmi

Date : _____

MEETING LOG

ACKNOWLEDGEMENT

I/We would like to thank everyone who had contributed to this project. I/We would like to express my/our gratitude to my/our Project supervisor, *Dr. Muhammad Shadab Alam Hashmi* for his/her invaluable advice, guidance, and his/her enormous patience throughout the development of the project.

In addition, I/we would also like to express my/our gratitude to my/our loving parents and friends who had helped and given me/us encouragement.

ABSTRACT

Speech emotion detection (SED) plays an important role in human-computer interaction, especially in mobile applications. The project focuses on developing a mobile application that uses advanced neural network models to recognize and interpret emotions in speech. The app is built using Flutter, a multifunctional platform for cross-platform integration that ensures compatibility across multiple devices.

The project started with the collection of different documents containing speech patterns of different views. Using neural network architectures such as convolutional neural networks (CNN) and recurrent neural networks (RNN), the system goes through a training and validation phase to learn and extract relevant features from audio data.

The application includes the integration of effective learning methods to integrate neural network models into the Flutter framework and provides real-time analysis of the data message coming from the microphone of the mobile phone. Thanks to this integration, the app is able to accurately analyze and classify emotions and provide users with instant feedback on the emotional content of their messages.

Furthermore, the user interface is designed to be extremely user-friendly and provides users with an intuitive experience. Collect, analyze and visualize desired results. In addition, the application allows for change and expansion, facilitating future development and expansion, such as adding new perspectives or improving accuracy through continuous learning.

Evaluation of the system includes rigorous testing to measure its accuracy, efficiency and immediate effectiveness. The results demonstrate the effectiveness of the SED mobile application and highlight its potential in areas such as mental health care, communication services, and human-computer sociality.

TABLE OF CONTENTS

CHAPTER 1 ..... **1**

INTRODUCTION **1**

1.1. Problem Statement 1

1.2. Objective 1

1.3. Project Scope 2

1.4. Advantages of the system 2

1.5. Relevance to the study program 3

1.6. Chapter Summary 4

CHAPTER 2 ..... **5**

EXISTING SYSTEMS **5**

2.1. Existing Systems 5

2.2. Drawbacks in Existing Systems 5

2.3. Need to Replace Existing Systems 6

2.4. Chapter Summary 7

CHAPTER 3 ..... **8**

REQUIREMENT ENGINEERING **8**

3.1. Proposed System 8

3.2. Understanding the System 8

3.3. Requirement Engineering 10

3.4. Gantt Chart 14

3.5. Hurdles in Optimizing the Current System 16

3.6. Chapter Summary 18

CHAPTER 4 ..... **19**

DESIGN **19**

4.1. Software Process Model 19

4.2. Design 20

4.3. Chapter Summary 25

CHAPTER 5 ..... **26**

DATABASE **26**

5.1. Database Introduction 26

5.2. Selected Database 26

5.3. Firebase Queries 29

5.4. Firebase Tables 29

5.5. Chapter Summary 30

CHAPTER 6 ..... **31**

DEVELOPMENT AND IMPLEMENTATION **31**

6.1. Development of the Computer Program 31

6.2. Implementation Strategy 32

6.3. Tools Selection 33

6.4. Coding 34

6.5. User Interface 34

6.6. Program Deployment 37

6.7. Chapter Summary 37

CHAPTER 7 ..... **39**

TESTING **39**

7.1. Introduction 39

7.2. Testing Methods 39

7.3. Comparison 40

7.4. Software Evaluation 41

7.5. Chapter Summary.....41

REFERENCES.....42

LIST OF FIGURES

FIGURE 3.1 GANTT CHART....16

FIGURE 4.1 METHODOLOGY DIAGRAM....20

FIGURE 4.2 USE CASE DIAGRAM....21

FIGURE 4.3 CLASS DIAGRAM....22

FIGURE 4. 4 ACTIVITY DIAGRAM....23

FIGURE 4. 5 SEQUENCE DIAGRAM....24

FIGURE 4. 6 COMPONENT DIAGRAM....24

FIGURE 6. 1 SIGN UP PAGE....35

FIGURE 6. 2 SIGN IN PAGE....35

FIGURE 6. 3 MULTIPLE USERS....36

FIGURE 6. 4 USER INTERFACE....36

LIST OF TABLES

TABLE 3.1 USER NEEDS OF SYSTEM 10

TABLE 3.2 FUNCTIONAL REQUIREMENT 01 10

TABLE 3.3 FUNCTIONAL REQUIREMENT 02 11

TABLE 3.4 FUNCTIONAL REQUIREMENT 03 11

TABLE 3.5 FUNCTIONAL REQUIREMENT 04 12

TABLE 3.6 FUNCTIONAL REQUIREMENT 05 12

Chapter 1

INTRODUCTION

This final year project focuses on creating a mobile application. It is supposed to use neural networks for speech understanding speech emotion detection. By combining deep learning and mobile technology, the goal is to create a system that is easy to use and quickly recognizes emotions in speech. Flutter, a powerful tool for creating mobile applications, is used here. It helps to seamlessly connect the app between different devices.

The main goal here is to create a system that can capture, sort and understand emotions expressed in speech. It's a step-by-step process that starts with collecting and processing different kinds of emotional speech data.

Using advanced neural network methods such as CNNs and RNNs, the project trains models to better understand complicated speech patterns. This helps the neural network filter out unclear thoughts.

The big thing in this project is mixing these well-trained models into Flutter. This way, the app can capture live voice input through the device's microphone and quickly work out the emotion in the speech.

In addition, the app design is user-friendly, making it easy to access and use, providing a pleasant experience for users who deal with emotional information. The structure of the application is also flexible so that it can be easily improved or modified to meet the needs of users.

1.1. Problem Statement

The main problem here, is to detect emotions through voice. The task is to develop a robust and accurate system for detecting emotions from speech signals. Given an audio input, the system should classify the underlying emotion expressed in the speech into predefined categories such as happiness, sadness, anger, surprise, fear, and neutral.

1.2. Objective

- The primary objective is to achieve high accuracy in classifying emotions from speech signals.

- The system should be robust to variations in emotional expression, including differences in intonation, pitch, rhythm, and duration of speech.
- The system should support real-time processing of speech inputs, enabling timely responses in applications such as human-computer interaction and customer service.
- The system should have an intuitive user interface and be easy to integrate into existing applications and systems. Users should be able to interact with the system effortlessly, enabling seamless integration into various domains and workflows.
- The system should handle sensitive speech data securely, adhering to privacy regulations and best practices for data protection.
- The system should be scalable to handle large volumes of speech data and support concurrent processing of multiple requests. It should efficiently utilize computational resources to accommodate increasing demand and workload.

1.3. Project Scope

- The range of speech emotion detection using a neural network in a mobile application is quite wide.
- Enables real-time analysis and understanding of emotions conveyed by speech.
- This technology can be used in various areas:
  - Mental health monitoring
  - Customer sentiment analysis
  - Virtual assistants
  - Interactive entertainment

Using the mobile app, users can easily capture and process audio input, allowing them to gain insight into emotional states and improve communication and engagement.

1.4. Advantages of the system

Using a mobile application for speech emotion recognition has several advantages:

1. **Improved human-computer interaction:** The application facilitates a more natural and intuitive interaction between users and devices by recognizing and responding to emotional cues in speech, leading to an improved user experience.
2. **Improved communication:** Helps understand the emotional context of spoken language, promoting better communication by interpreting nuances such as tone and sentiment, thereby reducing misunderstandings.

3. **Real-time emotional feedback:** Users receive immediate feedback on their emotional expressions, promoting self-awareness and enabling adjustments in communication based on recognized emotions.
4. **Versatile deployment:** Using Flutter allows the application to be deployed on multiple platforms, ensuring compatibility with various devices and operating systems and expanding its availability.
5. **Potential mental health application:** The application could serve as a mental health support tool, assisting in emotional monitoring and providing insights into one's emotional state, potentially assisting therapists or individuals in managing emotions.
6. **Tailored User Experiences:** With emotion recognition, an app can tailor its responses or suggestions, customize experiences based on detected emotional context, and create personalized interactions.
7. **Efficiency in emotional analysis:** The capabilities of neural networks enable fast and accurate analysis of complex emotional patterns in speech and ensure a high level of accuracy in recognizing different emotions.
8. **Adaptability and learning:** The application can evolve over time by continuously learning from user interactions, allowing for improvements in the accuracy of emotion detection and system response.
9. **Empowering Different Areas:** In addition to personal use, applications of this technology cover a range of areas such as customer service, education, entertainment and therapy, offering tailored emotional assistance or insights.
10. **Technological advances:** By merging neural networks with Flutter, the project contributes to the advancement of emotion recognition technology in mobile applications and paves the way for further innovation.
11. **Potential for research and insights:** The data collected and analysed could contribute to research in psychology, linguistics, and emotional intelligence and provide valuable insights into human emotions and behaviour.

1.5. Relevance to the study program

In this project, we would explore the fascinating field of analysing and understanding the emotions of speech. We would delve into the world of neural networks, which are powerful algorithms inspired by the human brain, to develop models that can effectively recognize and classify different emotions based on speech patterns.

Using Flutter, a popular mobile app development framework, we can apply our knowledge to create a mobile app that can analyse and interpret emotions from speech input. This involves the design and implementation of algorithms that can extract relevant features from speech signals, such as pitch, intensity and spectral characteristics. These features are then fed into a neural network model that is trained to identify and classify different emotions such as happiness, sadness, anger or surprise and others.

By combining the power of neural networks with the versatility of Flutter, we can create a mobile app that can accurately recognize and interpret emotion from spoken word. This technology has various potential applications, such as improving human-computer interaction, enhancing virtual assistants, or even helping to monitor mental health.

Overall, the study of speech emotion recognition using neural networks and Flutter would involve exploring the theory, algorithms and practical implementation of this exciting field to develop innovative and compelling applications that can understand and respond to human emotions expressed in speech.

1.6. Chapter Summary

This chapter discusses a mobile application that uses neural networks to accurately detect and classify emotions from spoken speech. The app allows users to record their speech and receive real-time emotion analysis. The app aims to improve understanding and awareness of emotions through innovative technology. The scope of the project includes the design of an intuitive and user-friendly interface, the implementation of functions to capture and process audio input from the user. The benefits of using a speech emotion recognition mobile app include accurate emotion recognition and multi-language support. The app can be deployed on both IOS and Android platforms, making it portable.

Chapter 2

EXISTING SYSTEMS

2.1. Existing Systems

1. EmoVoice(Platform: iOS)
2. Moodies Emotion Analytics(Platform: iOS, Android)

2.2. Drawbacks in Existing Systems

The disadvantages of EmoVoice and Moodies Emotion Analytics are:

1. **Accuracy limitations:**

- Dependence on data quality: The accuracy of these systems strongly depends on the quality and variety of training data. Biases or limitations in the data set may affect performance and generalizability across different demographic groups.
- Inaccuracy in complex emotions: Distinguishing subtle emotional nuances from vocal cues alone can be challenging, leading to potential misinterpretation or inaccurate emotion recognition.

2. **Interpretation challenges:**

- Contextual understanding: Emotions are context-dependent, and analyzing emotions based solely on vocal cues may lack contextual understanding, leading to misinterpretations.
- Cultural variation: These systems may struggle to recognize emotions in different cultural environments due to biases in the training data or cultural differences in vocal expressions.

3. **Privacy and Ethical Concerns:**

- Privacy: Speech emotion analysis involves the handling of sensitive user data, raising privacy concerns and potential misuse or mishandling of personal data.
- Consent and Transparency: Apps may not explicitly address user consent or provide transparency about how data is collected, used, or shared for sentiment analysis.

4. **Technical limitations:**

- Real-time processing: Depending on available computing resources, real-time emotion recognition may face processing speed or response time issues, impacting the user experience.
- Resource intensity: Deep learning models used for emotion recognition can require significant computing resources, which affects their performance on mobile devices with limited capabilities.

5. **User engagement and satisfaction:**

- User Interface: The user interface may not be as intuitive or user-friendly, impacting user engagement and satisfaction.
- Limited feedback loop: Users may not receive adequate feedback or explanations about how emotions are interpreted, reducing overall app usefulness and user trust.

6. **Regulatory Compliance:**

1. Compliance: Compliance with data protection laws and regulations regarding the handling of personal data may not be explicitly addressed or emphasized in these systems.

2.3. Need to Replace Existing Systems

1. **Enhanced Functionality:** Our Project will offer improved functionality, accuracy, or other features that existing systems lack.
2. **Technological Advances:** Advances in technology, particularly in machine learning and mobile app development, could enable more sophisticated and efficient systems.
3. **Improving user experience:** Our Project aims to provide a better user experience, ease of use or a more intuitive interface compared to existing systems.
4. **Better performance:** Improvements to existing systems can mean increasing the speed, accuracy, or overall performance of emotion recognition or analysis.
5. **Wider compatibility:** Developing a system that works seamlessly across multiple platforms (IOS, Android, etc.) could be an incentive if existing systems are limited to specific platforms.
6. **Customization and adaptability:** The creation of a new system may focus on meeting specific needs or specific requirements that are not adequately addressed by existing systems.

2.4. Chapter Summary

This chapter outlines the limitations of existing systems like EmoVoice and Moodies Emotion Analytics, emphasizing the need for a new system that overcomes these limitations while highlighting the key motivations for developing a replacement system.

Chapter 3

REQUIREMENT ENGINEERING

3.1. Proposed System

This project encapsulates the essence of the project by emphasizing the development of a mobile application that uses the capabilities of neural networks to recognize and interpret emotions from speech in real time. Flutter integration ensures cross-platform compatibility and an intuitive user interface, contributing to better interaction between users and devices.

It focuses on using advanced technologies (neural networks) and a versatile framework (Flutter) to create a practical solution for understanding and responding to speech-borne emotions, with the goal of improving user experiences and communication paradigms in human-computer interactions.

3.2. Understanding the System

The purpose of this project is to develop a mobile application capable of recognizing and interpreting emotions expressed in speech. This application aims to improve human-computer interaction by enabling devices to understand and respond to emotional cues in spoken language.

3.2.1. User Involvement

We understand the needs of users. Engage potential users, which may include individuals looking for better communication tools or those interested in understanding the emotional cues in conversations. Conduct surveys to understand their requirements and expectations from an emotion recognition application. We get user feedback on user interface design.

We will allow users to test the functionality of the application at various stages of development.

After the app is released, encourage users to provide ongoing feedback and suggestions for improvements. Create user feedback channels and iterate the app based on user suggestions for continuous improvements.

3.2.2. Stakeholders

2. **End Users:**

A person who interact with the application to recognize emotions in spoken language. These users may include individuals seeking better communication tools or those interested in understanding emotions in conversations.

3. **Developers:**

Now the team creates, develops and manages the application. This includes software developers, technical experts, and Flutter developers.

3.2.3. Domain

1. **Human_Computer Interaction (HCI):**

This project focuses on human interaction with computers and technology. The SER mobile application aims to improve communication through the recognition of emotions in speech and participate in the research and development of human-computer relations.

2. **Artificial Intelligence (AI) and Machine Learning:**

This project focuses on artificial intelligence and machine learning, especially speech and emotion recognition. It involves training neural networks to understand and interpret emotional content in verbal data.

3. **Mobile App Development:**

In mobile app development, this project uses Flutter as a framework to create cross-platform mobile applications.It involves understanding mobile UI/UX design, optimization, and mobile-specific considerations.

4. **Psychology and Emotional Intelligence:**

Relates to psychology, particularly the understanding and expression of emotions. This application aims to contribute to the field by detecting emotional nuances in speech.

5. **Ethical considerations and privacy:**

In the field of ethics and privacy, this project addresses the role of cognitive awareness and reflective ethics in relation to user privacy and data processing.

3.2.4. Needs of System

Table 3.1User Needs of System

SR #	Needs	Need ID
1	Audio Data Collection	ID1
2	Data Preprocessing	ID2
3	Real-time Speech Analysis	ID3
4	Neural Network Integration	ID4
5	Emotion Recognition and Classification	ID5
6	User Interaction and Feedback	ID6
7	Cross-Platform Compatibility	ID7
8	Performance Optimization	ID8
9	Data Privacy and Security	ID9

3.3. Requirement Engineering

3.3.1. Functional Requirements

Table 3.2 Functional Requirement 01

Functional Requirement ID	FR-01
Name	Real-time Speech Analysis
Description	The application must analyze spoken language in real-time, capturing and processing audio input from the device's microphone.
Input	Audio Stream
Output	Processed Audio, Emotion Labels
Precondition	Microphone Access, Stable Audio
Postcondition	Real-time Analysis, Accuracy and Reliability

Table 3.3 Functional Requirement 02

Functional Requirement ID	FR-02
Name	Neural Network Integration
Description	The application should have functionality to accurately recognize and classify various emotions conveyed in speech, such as happiness, sadness, anger, etc.
Input	Model Architecture and Parameters
Output	Predictions
Precondition	Trained Model Availability
Postcondition	Scalability

Table 3.4 Functional Requirement 03

Functional Requirement ID	FR-03
Name	Emotion Recognition and Classification
Description	The application will incorporate neural network models for emotion recognition, ensuring efficient integration within the application's architecture.
Input	Features Extraction Parameters
Output	Emotion Labels
Precondition	Model selection and training
Postcondition	Generalization

Table 3.5 Functional Requirement 04

Functional Requirement ID	FR-04
Name	User Interaction and Feedback
Description	The application will provide user-friendly interfaces displaying detected emotions, either through visual indicators or text-based feedback.
Input	User Actions
Output	User Interface Updates
Precondition	Appropriate Permissions
Postcondition	User Satisfaction

Table 3.6 Functional Requirement 05

Functional Requirement ID	FR-05
Name	Cross-Platform Compatibility
Description	The application should be developed using Flutter to ensure compatibility across multiple mobile platforms (Android and iOS) from a single codebase.
Input	Application Codebase, Development Tools and SDKs
Output	Functional Application
Precondition	Cross-Platform Framework
Postcondition	Successful Deployment

3.3.2. Non-Functional Requirements

1. **Accuracy and Precision:**

The application should achieve a higher accuracy in emotion recognition, aiming for a specified precision level to ensure reliable results.

2. **Real-time Processing:**

The application should ensure minimal latency in emotion recognition processing, providing real-time feedback on detected emotions.

3. **Usability and User Experience:**

We should create an intuitive and user-friendly interface for ease of interaction, ensuring a pleasant user experience.

We should optimize the application's usability for individuals with varying technological proficiencies.

4. **Compatibility and Portability:**

We must ensure compatibility across multiple mobile platforms (Android and iOS) using Flutter, maintaining consistent performance and functionality.

5. **Data Privacy and Security:**

We will implement robust security measures to protect user data, ensuring compliance with privacy regulations and encryption of sensitive information.

6. **Reliability and Availability:**

Maintain high reliability, minimizing system downtime or disruptions in emotion recognition functionalities.

Ensure the application is available and accessible to users whenever required.

7. **Performance Optimization:**

We will ensure efficient memory and battery usage on mobile devices.

We will enhance application performance to handle various speech patterns and languages.

3.3.3. Need to Feature Mapping

Feature mapping in speech emotion recognition (SER) refers to the process of identifying and extracting relevant features from speech signals that help recognize emotions. In the context of building a SER mobile application using neural networks and Flutter, the need for feature mapping is significant for several reasons:

1. **Understanding Emotion Representation:** Feature mapping helps to understand how emotions manifest in speech signals by extracting relevant acoustic properties (such as pitch, intensity, spectral characteristics) associated with different emotions.
2. **Preparation of input data for neural networks**: Neural networks require appropriate input representations. The mapping function transforms raw speech signals into a format that neural networks can efficiently process for emotion recognition.
3. **Selection of features for model training:** Identifying the most discriminating features from speech signals increases the neural network's ability to accurately classify emotions. Feature mapping helps in selecting the most relevant attributes.
4. **Improving model performance:** Information feature extraction leads to better performance of the model in identifying emotions from speech, thereby increasing the accuracy and reliability of the SER system.
5. **Dimensionality and complexity reduction:** Feature mapping can include techniques such as dimensionality reduction, which helps reduce the complexity of the input data while preserving the essential information for emotion recognition.
6. **Adapting to different speech patterns:** Effective feature mapping allows the model to generalize well across different speech patterns, accents, and languages, making the SER application more adaptive and inclusive.
7. **Real-time processing and efficiency:** Efficient extraction of relevant features enables real-time or near-real-time processing of speech signals, which is crucial for a responsive mobile SER application.
8. **Integration with Flutter:** Mapping extracted features to UI elements in Flutter enables visual representation or feedback of recognized emotions, improving user experience and interaction.

3.4. Gantt Chart

Task	Duration	Dependencies
Phase 1: Planning and Research	4 weeks	----
Conduct market research	1 week	----

Define project scope and objectives	1 week	Market research
Identify stakeholders and their needs	1 week	Scope and objectives defined
Create requirements document	1 week	Stakeholder needs identified
Phase 2: Development	12 weeks	Planning completed
Develop UI/UX wireframes and design	3 weeks	Requirements finalized
Implement audio capture and processing	2 weeks	UI/UX design completed
Integrate neural network for emotion recognition	4 weeks	Audio processing implemented
Implement cross-platform functionality	2 weeks	Neural network integration
Phase 3: Testing and Optimization	6 weeks	Development completed
Conduct unit testing and debugging	2 weeks	Development phase completed
Perform system integration testing	2 weeks	Unit testing and debugging
Optimize performance and accuracy	2 weeks	System integration testing
Phase 4: Deployment and Launch	2 weeks	Testing and optimization completed
Prepare for app store submission	1 weeks	Testing and optimization completed

Deploy app on Android and iOS platforms	1 weeks	App store submission preparation
Phase 5: Post-launch Evaluation	2 weeks	Deployment and launch completed
Gather user feedback and conduct evaluation	2 weeks	App deployment completed

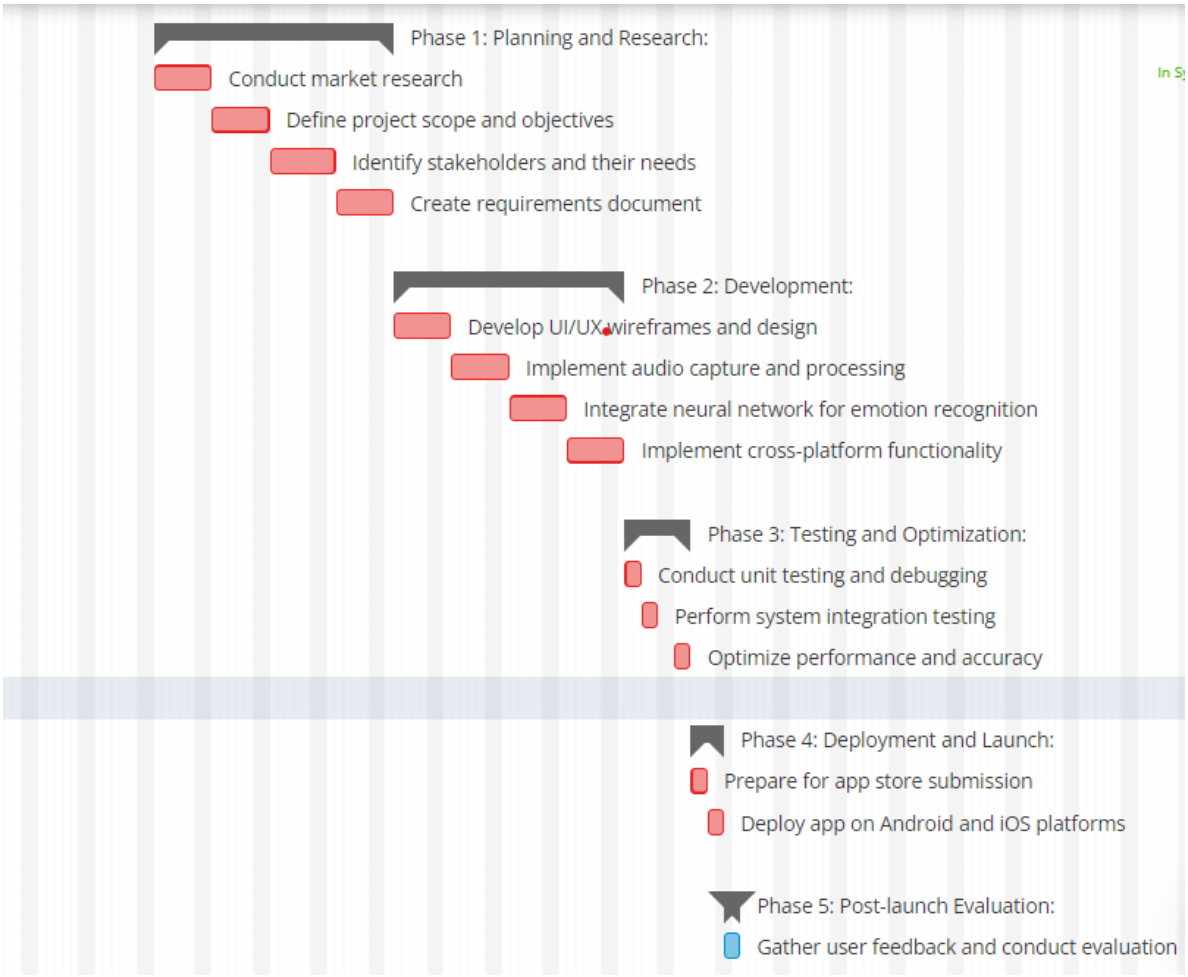

Figure 3.1 Gantt Chart

3.5. Hurdles in Optimizing the Current System

1. **Optimization of the algorithm:** Neural networks used for emotion recognition require extensive fine-tuning and optimization to balance accuracy and processing speed.

Achieving high accuracy without compromising real-time performance can be challenging.

2. **Limitation of computing resources:** Mobile devices have limited computing power compared to desktops or servers. Optimizing a neural network to run efficiently on mobile hardware while maintaining accuracy is a hurdle.
3. **Privacy Concerns**: Emotion recognition involves manipulating sensitive user data (speech input). Balancing the need for privacy and the effectiveness of the recognition model can present implementation challenges.
4. **Consistency across multiple platforms:** Ensuring consistent performance and user experience across mobile platforms (Android and iOS) with Flutter may require additional effort due to differences in device capabilities and OS behavior.
5. **Variability in speech patterns:** Emotions expressed in speech can vary greatly depending on accent, languages, intonation and cultural nuances. Building a model that comprehensively recognizes different emotional stimuli is a challenge.
6. **Real Time Processing:** Achieving real-time processing without significant latency is critical to the user experience. Optimizing an application for instant processing of speech data can be challenging, especially given changing network conditions.
7. **Model size and deployment:** Embedding a neural network model in a mobile application requires a balance between model size and accuracy. Large models can affect the size and performance of the application.
8. **Testing and Verification:** Rigorous testing across different speech patterns, accents, and environmental conditions is necessary to ensure the generalizability and robustness of the model. This testing can be time consuming and resource intensive.
9. **User interface optimization:** Creating an intuitive and responsive user interface that effectively communicates recognized emotions can be challenging, especially while maintaining the simplicity and clarity of the application.
10. **Continuous Improvement:** As new speech patterns or emotions emerge, constantly updating and improving the model to accommodate these changes is a constant challenge.

3.6. Chapter Summary

In this chapter, we discussed about the SER application. It includes all Functional Requirements as well as NonFunctional Requirements. It explains hurdles which we w ill faceduring development process.

Chapter 4

DESIGN

4.1. Software Process Model

The **Iterative process model** involves cyclic iterations of planning, development, testing, and refining. It gathers initial requirements for the SER application, focusing on its core functionalities, user interactions, and platform compatibility. It plans the development process and establish the project's goals and objectives. There will be First Iteration in which core development starts. Then, it will test and evaluate initial iteration. This testing and evaluation provides feedback. Based on feedback, refine the neural network model to enhance emotion recognition accuracy and add support for more nuanced emotions. In the same way, iterations continued. In the final, it combines the refined components from multiple iterations into the final SER application version. It performs comprehensive testing, ensuring cross-platform compatibility, accuracy, and user satisfaction. At last, it prepares the application for deployment on app stores (IOS and Android).

4.1.1. Benefits of Selected Model

Continuous Improvement: The Iterative model enables continuous improvements based on user feedback, resulting in a more refined and user-friendly application.

Flexibility and adaptability: It can accommodate evolving requirements and technological advances in neural networks and UI frameworks like Flutter.

Early Deliverables: The model delivers functional features in each iteration, allowing early testing and validation of specific features.

Risk Mitigation: It identifies and resolves issues early in the development cycle, reducing potential risks and improving the overall quality of the application.

4.1.2. Limitations of Selected Model

While the iterative model offers several benefits, it also presents certain limitations.

1. **Resource intensive:** Iterative development often requires significant resources, including time, skilled personnel, and continuous user engagement for feedback. Developing and refining neural network models requires significant computing resources and expertise.

2. **Increased costs:** Continuous iteration and refinement can lead to increased development costs, especially if extensive changes or rework are required in neural network training or complex UI/UX redesigns.
3. **User Engagement Challenges:** Collecting and incorporating user feedback on an ongoing basis can be challenging, especially if availability or user interest declines over several iterations.
4. **Testing and Verification:** Ensuring comprehensive testing and validation across multiple iterations can be time-consuming. Testing the accuracy and efficiency of emotion recognition models requires large datasets and validation procedures.

4.2. Design

4.2.1. Methodology of the Proposed System

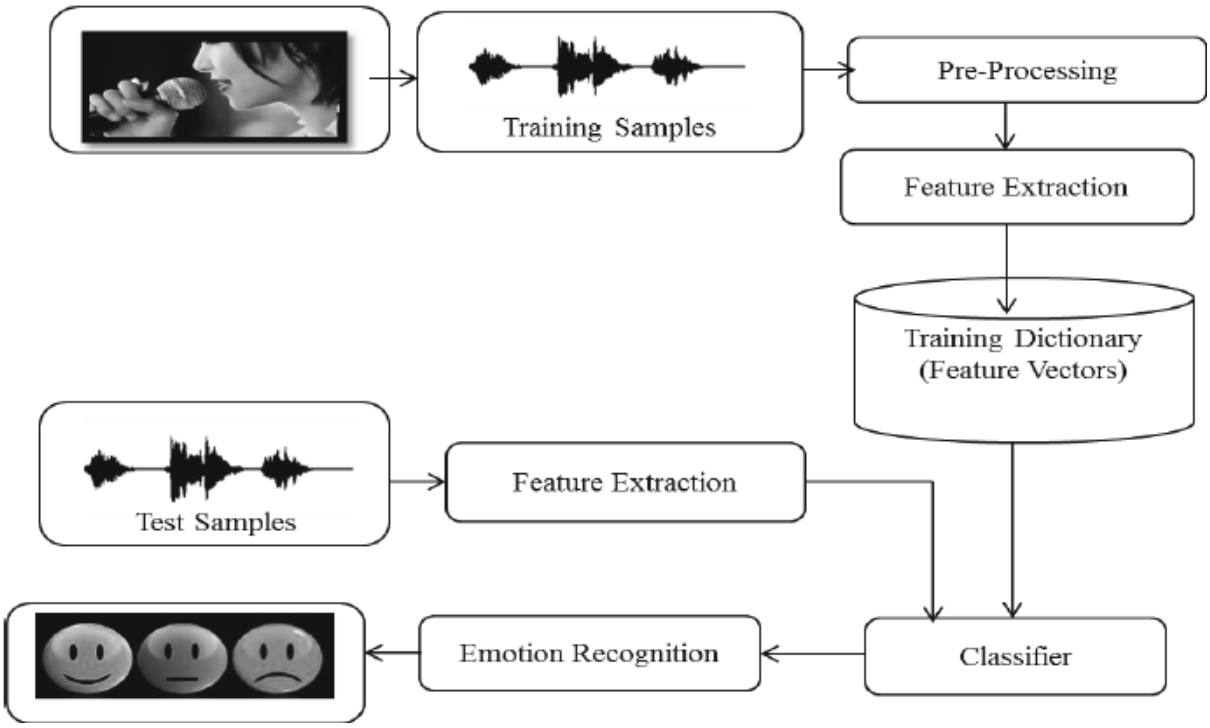

Figure 4.1 Methodology Diagram

4.2.2. UML Diagrams

4.2.2.1. *Use Case Diagram of the System*

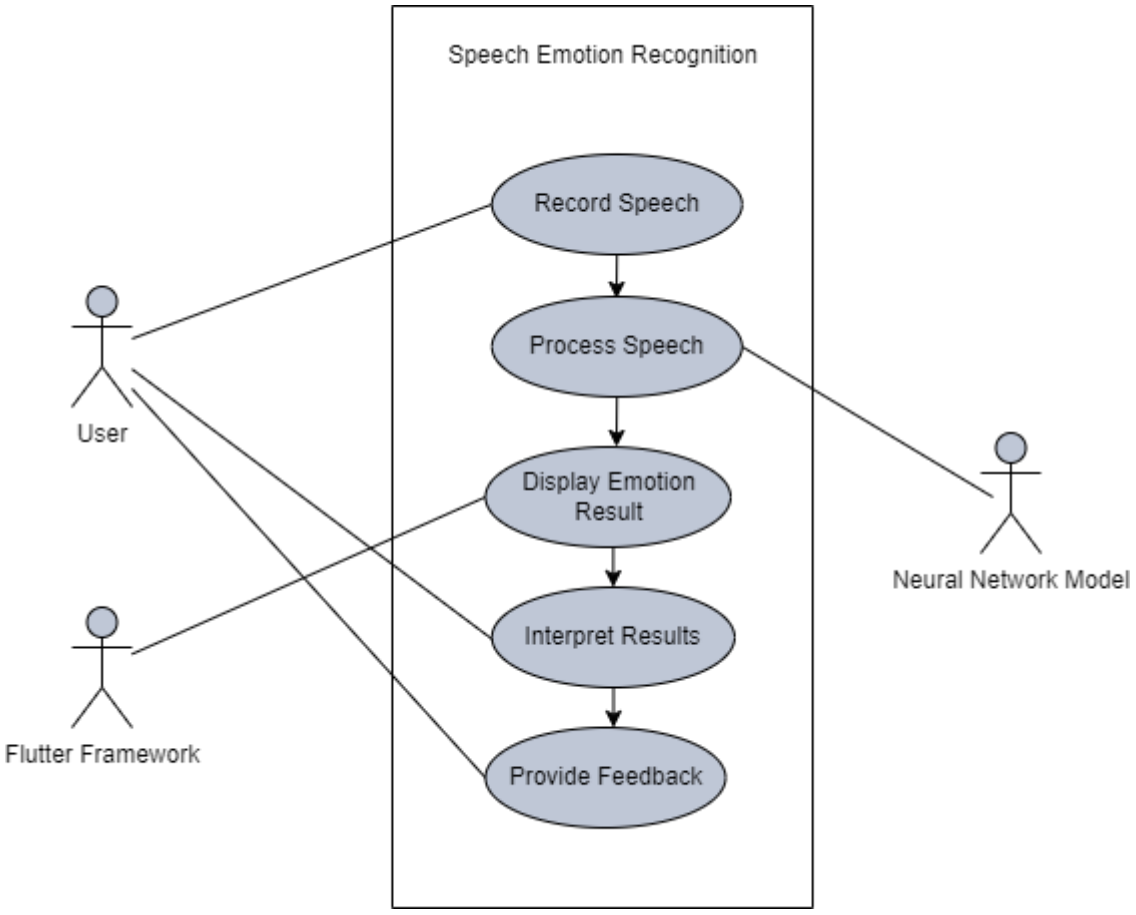

Figure 4.2 Use Case Diagram

4.2.2.2. *Class Diagram of the System*

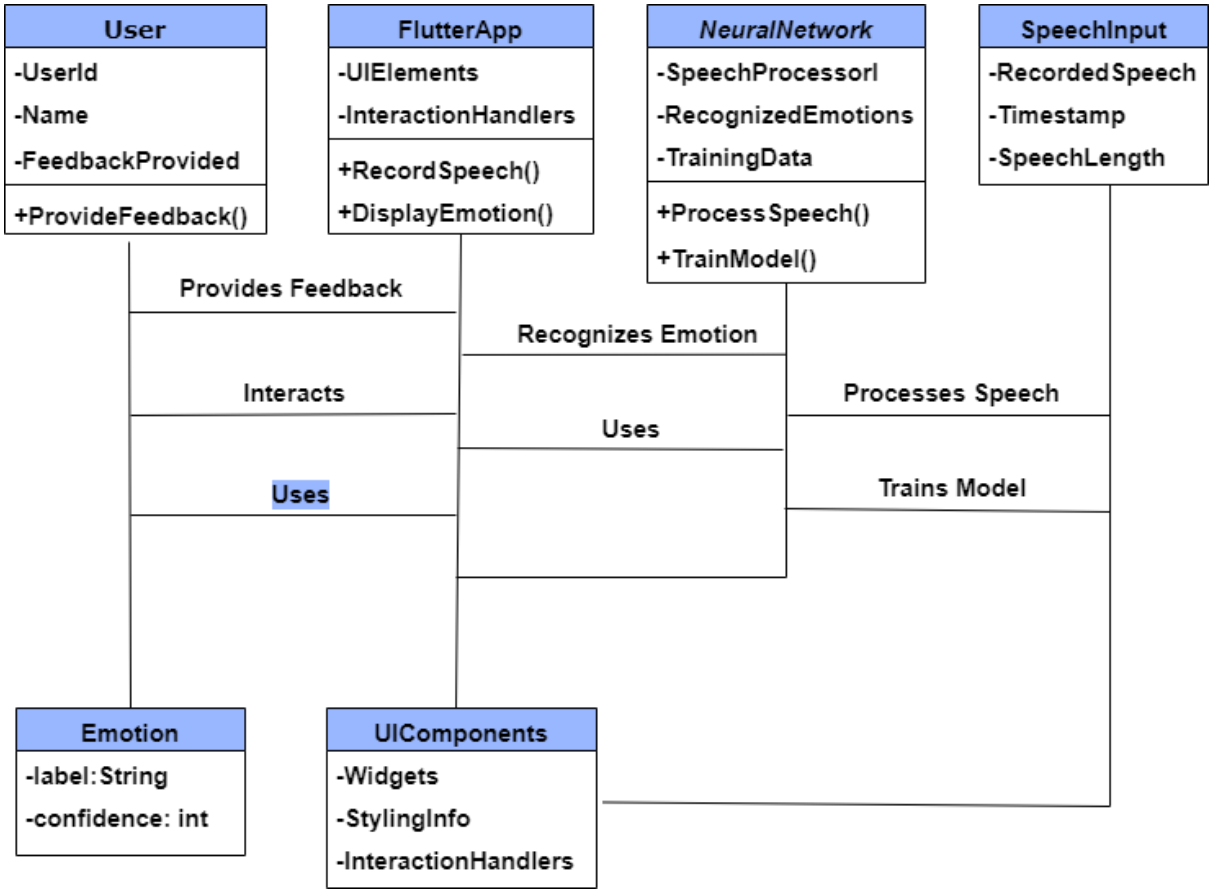

Figure 4.3 Class Diagram

4.2.2.3. *Activity Diagram of the System*

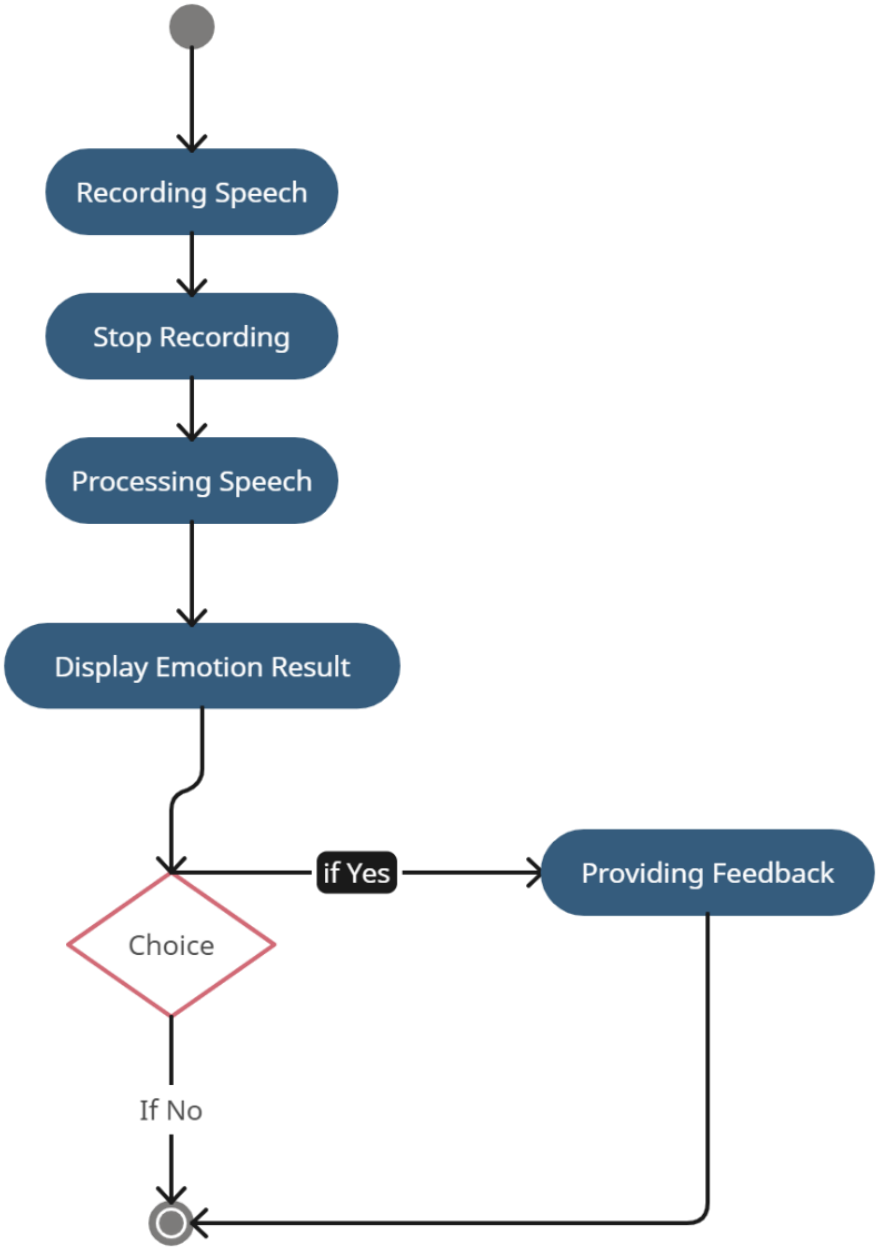

Figure 4.4 Activity Diagram

4.2.2.4. *Sequence Diagram of the System*

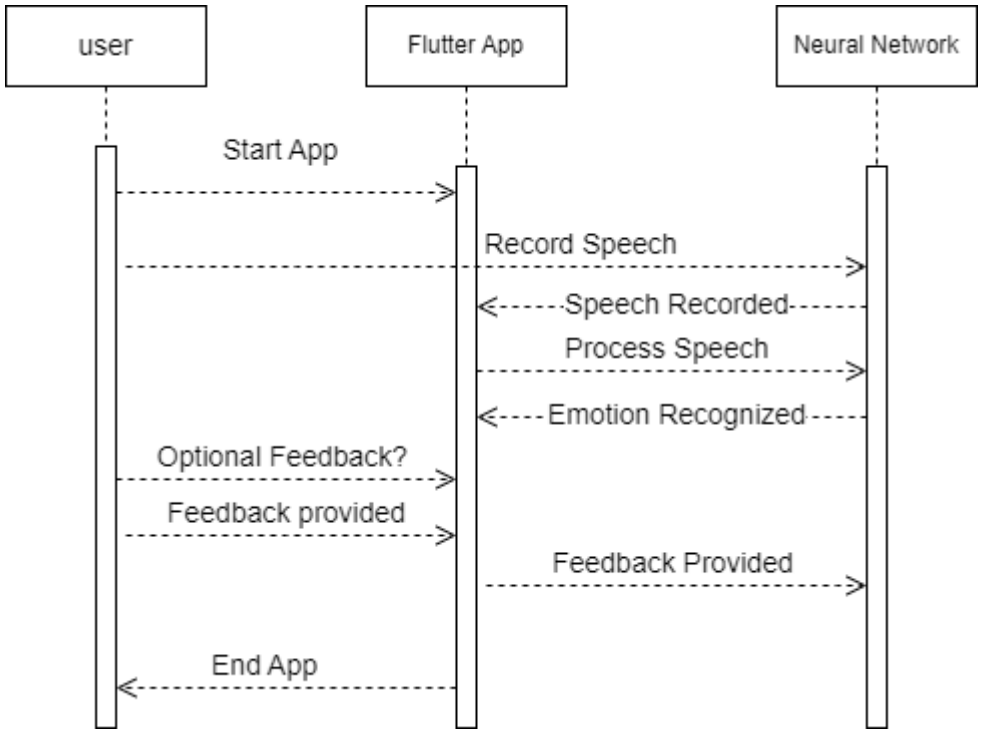

Figure 4.5 Sequence Diagram

4.2.2.5. *Component Diagram of the System*

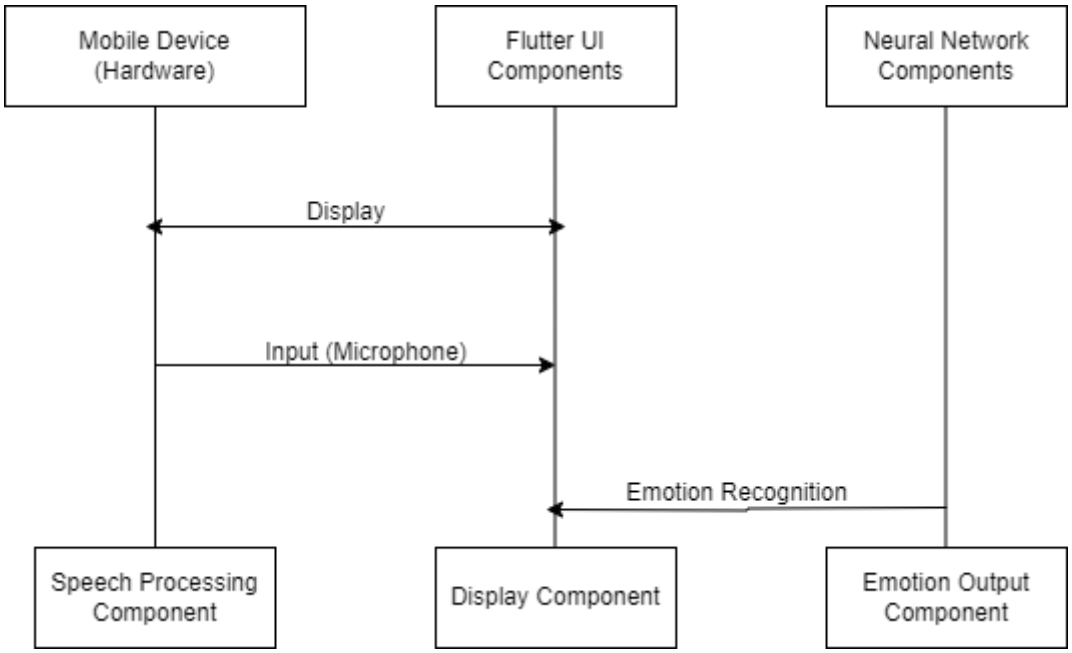

Figure 4.6 Component Diagram

4.3. Chapter Summary

This chapter describes the software process model .It discuss the benefits of the model as well as its limitations .Moreover, it consists of the methodology diagram and UML diagrams of the proposed system.

Chapter 5

DATABASE

5.1. Database Introduction

A database is a collection of data or information stored, usually electronically, on a computer. Databases are usually managed by a database management system (DBMS). Database and DBMS together with their respective applications are called database system and are often referred to as a single database. Model to improve performance and data query efficiency. Data can be easily accessed, managed, updated, modified, managed and organized. Most databases use Structured Query Language (SQL) to collect and query data.

5.2. Selected Database

We are using Firebase for our proposed system. Firebase is Google's mobile and web development platform. It provides a variety of tools and services to help developers create great applications quickly and efficiently.

Firebase provides a real-time database, which is, at its core, a cloud-hosted NoSQL database that allows you to store and sync user data over time. This means that changes made by one user are instantly reflected across all devices, providing a great customer experience.

5.2.1. Reasons for Selection of the Database

1. **Real-time Data Syncing:** Firebase offers real-time data synchronization, meaning any changes made to your data are immediately propagated to all connected clients. This feature is crucial for applications requiring live updates, such as messaging apps, collaborative tools, or live sports scores.
2. **Scalability:** Firebase scales effortlessly to handle both small-scale and large-scale applications. Whether you're just starting out with a few users or have millions of active users, Firebase can accommodate your needs without requiring manual sharding or complex scaling configurations.
3. **Offline Support:** Firebase's real-time database includes robust offline support. This means your app can continue to function even when the device loses its internet connection. Once the connection is restored, Firebase automatically syncs any pending changes, ensuring a seamless user experience.

4. **Cross-platform Compatibility:** Firebase supports multiple platforms, including iOS, Android, and web. This allows developers to use a single backend solution for all their client applications, reducing development time and maintenance overhead.
5. **Authentication and Security:** Firebase provides built-in authentication services, allowing you to easily integrate user authentication into your app using email/password, social login (Google, Facebook, etc.), or custom authentication systems. Additionally, Firebase offers fine-grained security rules to control access to your data, ensuring only authorized users can read or write data.
6. **Serverless Architecture:** Firebase is a serverless platform, meaning developers can focus on building their applications without worrying about managing servers or infrastructure. Firebase handles server maintenance, scaling, and security updates automatically, allowing developers to focus on writing code and delivering features.
7. **Built-in Analytics:** Firebase includes built-in analytics capabilities, providing insights into user behavior, app usage, and engagement metrics. This data helps developers make informed decisions about app optimization, user retention, and feature prioritization.
8. **Integrated Ecosystem:** Firebase integrates seamlessly with other Google services, such as Google Cloud Platform, Google Analytics, and Google AdMob. This allows developers to leverage additional tools and services to enhance their apps without significant integration effort.

5.2.2. Benefits of the Selected Database

1. **Cloud Storage:** Firebase offers cloud storage solutions for storing and serving user-generated content, such as images, videos, and documents. This feature allows developers to offload storage management to Firebase, reducing the complexity of their applications.
2. **Analytics:** Firebase includes built-in analytics capabilities that provide insights into user behavior, app usage, and engagement metrics. These analytics help developers make informed decisions about app optimization, user retention, and feature prioritization.
3. **Performance Monitoring:** Firebase offers performance monitoring tools that allow developers to track app performance metrics, such as app startup time, screen

rendering time, and network latency. This helps identify performance bottlenecks and optimize app performance for a better user experience.

4. **Crash Reporting:** Firebase provides crash reporting tools that help developers identify and fix issues quickly. Developers receive real-time crash reports, stack traces, and device information, allowing them to diagnose and resolve problems efficiently.
5. **Easy Integration:** Firebase integrates seamlessly with other Google services, such as Google Cloud Platform, Google Analytics, and Google AdMob. This allows developers to leverage additional tools and services to enhance their apps without significant integration effort.

5.2.3. Limitations of the Selected Database

1. **Limited Query Capabilities:** Firebase's real-time database has limited querying capabilities compared to traditional SQL databases. Complex queries involving multiple filters or aggregations may be challenging to implement efficiently.

2. **Data Structure Complexity:** Firebase's NoSQL database requires denormalizing data to optimize for read operations, which can lead to increased data redundancy and complexity. Maintaining data consistency and integrity in a denormalized schema may require careful planning and management.
3. **Security Rules Complexity:** Firebase's security rules system can be complex to configure, especially for applications with intricate data access requirements. Writing and maintaining security rules to enforce fine-grained access control may require a deep understanding of Firebase's security model.
4. **Vendor Lock-in:** Using Firebase ties your application to Google's infrastructure and ecosystem. While Firebase provides a comprehensive suite of services, migrating away from Firebase to another platform can be challenging and time-consuming.
5. **Cost Considerations:** While Firebase offers a generous free tier, scaling your application may incur additional costs, especially for bandwidth, storage, and database operations. Understanding Firebase's pricing model and monitoring usage is essential to avoid unexpected expenses as your application grows.

6. **Vendor Reliability and Support:** Relying on a third-party service like Firebase means trusting the vendor's reliability and support. While Google generally provides robust infrastructure and support, occasional outages or service disruptions may impact your application's availability and performance.
7. **Customization Constraints:** Firebase's pre-built services offer convenience but may lack customization options for specific use cases. Developers requiring highly customized or specialized features may find Firebase's offerings insufficient and need to implement custom solutions or integrate with other services.
8. **Regulatory Compliance:** Firebase's data storage and processing may be subject to regulatory requirements, such as data residency or privacy laws. Ensuring compliance with applicable regulations, especially in regulated industries or regions, is crucial when using Firebase for sensitive data.

5.3. Firebase Queries

Following are the firebase queries used:

- firebase login
- firebase projects:list
- firebase login:ci
- firebase init
- flutterfire Configure

5.4. Firebase Tables

In Firebase's real-time database, data is organized into a JSON tree structure rather than traditional tables. Here's how you can conceptualize tables in Firebase:

Root Node: The root of your Firebase database serves as the entry point for your data. It's like the root directory of a file system.

Nodes: Nodes are like tables in a relational database. Each node represents a collection of related data. For example, you might have a "users" node to store user profiles or a "posts" node to store blog posts.

Keys: Keys are unique identifiers for each node. They are similar to primary keys in relational databases and are used to access individual nodes. Keys are typically generated by Firebase when data is added but can also be manually specified.

Fields: Fields are like columns in a table. They represent individual pieces of data stored within each node. For example, a user node might have fields like "name," "email," and "age."

5.5. Chapter Summary

This chapter covers the benefits as well as the limitations of firebase database .Further ,it covers the basic queries and tables of the database.

Chapter 6

DEVELOPMENT AND IMPLEMENTATION

6.1. Development of the Computer Program

Speech Emotion Detection (SED) is a branch of affective computing that aims to recognize human emotions from speech signals. It involves the analysis of various acoustic features extracted from speech samples to classify the emotional state of the speaker. The following paragraphs outline the basic components of a Speech Emotion Detection Program.

Components:

1. **Data Collection:**
- The program begins by collecting a dataset of speech samples labeled with corresponding emotions (e.g., happy, sad, angry).
- Diverse datasets with variability in speakers, languages, and emotional expressions are essential for robust model training.
2. **Preprocessing:**
- Raw audio data undergo preprocessing to extract relevant features.
- Common preprocessing steps include noise removal, normalization, and feature extraction (e.g., Mel-frequency cepstral coefficients (MFCCs), pitch, energy).
3. **Feature Extraction:**
- Features extracted from the preprocessed audio serve as input for emotion classification algorithms.
- Features may include prosodic features (e.g., pitch, duration), spectral features (e.g., MFCCs), and higher-level features derived from linguistic analysis.
4. **Emotion Classification:**
- Machine learning algorithms such as Support Vector Machines (SVM), Convolutional Neural Networks (CNN), or Recurrent Neural Networks (RNN) are trained on the extracted features.
- During training, the model learns to map input features to corresponding emotional labels.
- Evaluation metrics such as accuracy, precision, recall, and F1-score are used to assess model performance.

5. **Model Deployment:**

- Once trained, the SED model can be deployed in real-time or batch processing environments.
- In real-time scenarios, the model processes incoming audio streams and predicts the emotional state of the speaker.
- In batch processing, the model analyzes pre-recorded audio files in batches.

6. **Performance Evaluation:**

- The performance of the SED program is evaluated using validation datasets and cross-validation techniques.
- Continuous monitoring and model retraining may be necessary to maintain optimal performance, especially in dynamic environments.

6.2. Implementation Strategy

1. **Define project scope and objectives:**

- Clearly outline the goals and objectives of the speech emotion detection project.
- Determine the target emotions to be detected, the intended application, and any specific requirements or constraints.

2. **Model Selection and Training:**

- Choose appropriate machine learning algorithms or deep learning architectures for emotion classification.
- Experiment with various models such as Support Vector Machines (SVM), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), or hybrid models to determine the best-performing approach.
- Split the dataset into training, validation, and testing sets for model evaluation.
- Train the selected model using the training data and optimize hyperparameters to improve performance.

3. **Model Evaluation and Validation:**

    - Evaluate the trained model using appropriate evaluation metrics (e.g., accuracy, precision, recall, F1-score) on the validation set.
    - Perform cross-validation techniques to ensure robustness and generalization of the model.

- Fine-tune the model based on validation results and iterate if necessary to improve performance.

4. **Deployment and Integration:**
   - Deploy the trained model in the desired environment, whether it's real-time processing or batch processing.
   - Integrate the model into the target application or system, ensuring compatibility and scalability.
   - Implement mechanisms for monitoring model performance and handling edge cases or exceptions.
5. **Continuous Improvement and Maintenance:**
   - Continuously monitor the performance of the deployed model in real-world scenarios.
   - Collect feedback from users and stakeholders to identify areas for improvement.
   - Incorporate new data and retrain the model periodically to adapt to evolving speech patterns and user preferences.
   - Stay updated with advancements in speech processing and machine learning techniques to enhance the system's capabilities over time.
6. **Ethical Considerations:**
   - Consider ethical implications such as data privacy, bias in model predictions, and potential misuse of emotion detection technology.
   - Implement measures to ensure fairness, transparency, and accountability throughout the development and deployment process.

6.3. Tools Selection

6.3.1. Hardware

The app’s hardware requirements were determined based on the computational needs of the machine learning model and the target platform. The minimum hardware specifications for end-users’ devices were identified to ensure optimal performance.

6.3.2. Software

The software tools selected for the app’s development included:

- **Programming languages:** Python was used for training and evaluating the machine learning model, while Dart was used for developing the app's frontend using the Flutter framework.
- **Machine learning libraries:** Popular machine learning libraries such as TensorFlow or PyTorch were employed for model development, training, and evaluation.
- **Development frameworks:** Flutter, a cross-platform framework, was chosen for its ability to build native-like apps for both iOS and Android platforms.
- **Integrated Development Environment (IDE):** IDEs like PyCharm or Jupyter Notebook were utilized for machine learning model development, while Android Studio or Visual Studio Code were used for app development.

6.4. Coding

The coding process involved implementing the machine learning model, integrating it with the Flutter framework, and developing the app's features and functionalities. The best coding practices, including code modularity, code reviews, and version control, were followed to ensure maintainability and collaboration.

6.5. User Interface

6.5.1. Description

Main Interface:

The main interface of the app provides a clean and intuitive layout for users to interact with.The main interface includes options for users to perform various actions such as recording audio, analyzing pre-recorded audio files, accessing settings, and viewing results.

Audio Recording Section:

This section allows users to record their voice directly through the app for real-time emotion detection.It includes a prominent "Record" button to initiate the recording process.Visual indicators such as waveform representation or a timer display the progress of the recording.Users can stop the recording once they finish speaking.

Analysis Results:

Once the emotion detection process is complete, the app displays the results in a visually appealing format.Emotions detected from the input audio are listed along with confidence scores or probabilities.Users can easily interpret the results and take appropriate actions based on the detected emotions.

Games and food:

User can also play games and perform meditation by the section.This section includes yoga sessions and online games to help users with their stressfull emotions.

6.5.2. Interface Screenshots

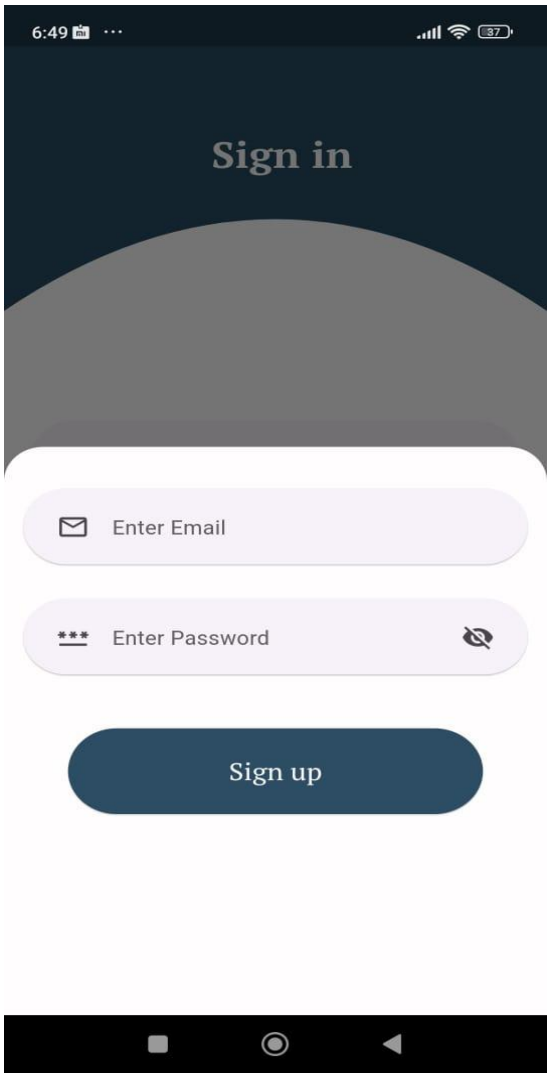

Figure 6.1 Sign Up page

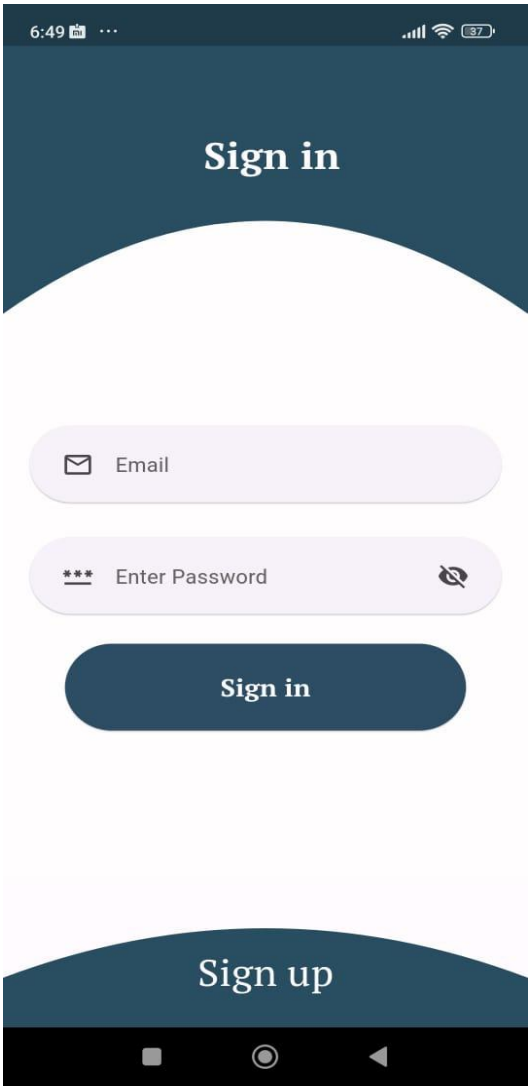

Figure 6.2 Sign In page

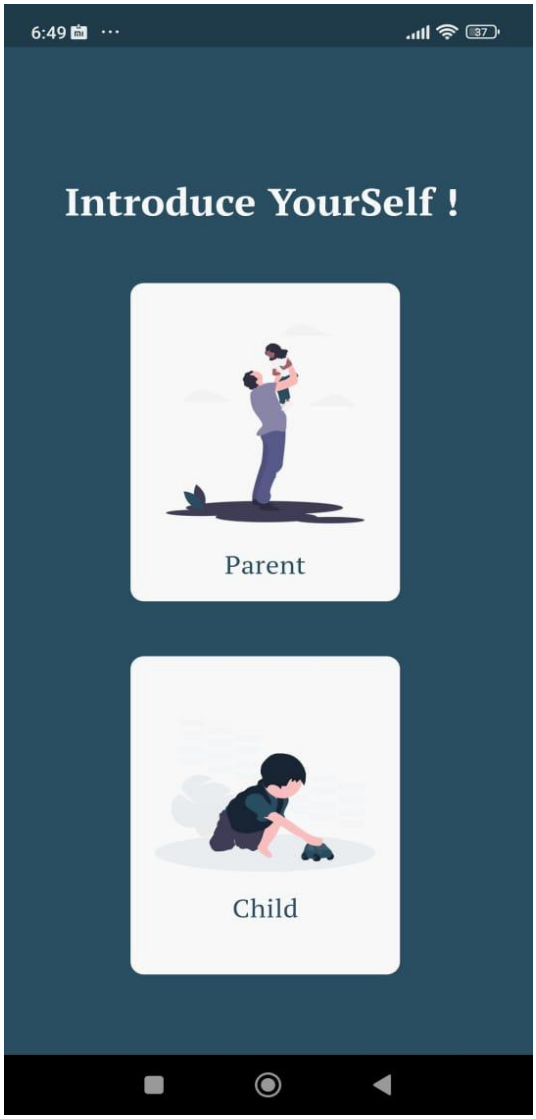

Figure 6.3 Multiple Users

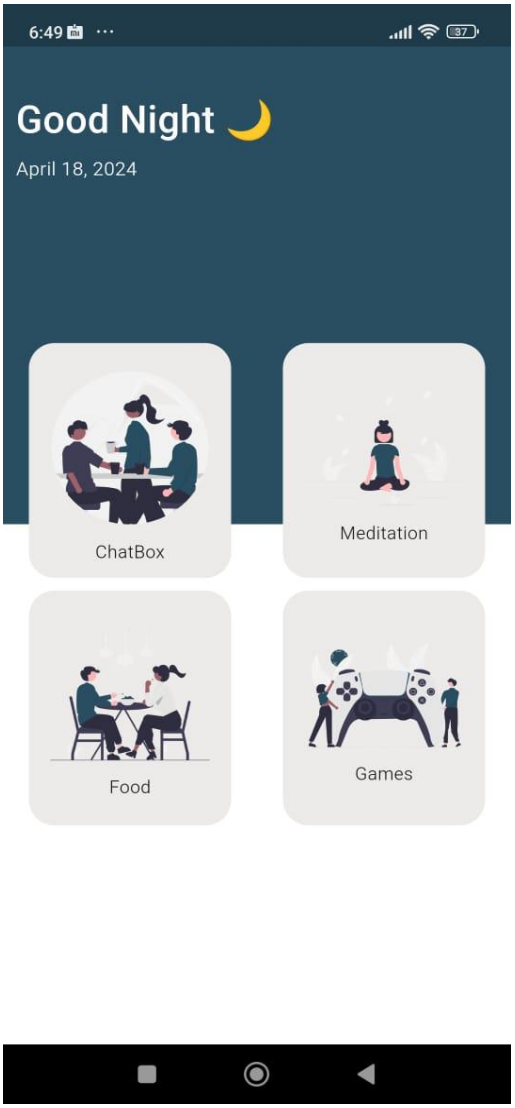

Figure 6.4 User Interface

6.6. Program Deployment

Deploying a speech emotion detection app involves making the application accessible to users on various platforms, ensuring smooth functionality, and maintaining reliability. Following are steps for deploying a speech emotion detection app:

Select Deployment Platforms:

Determine the platforms on which the app will be deployed, such as mobile devices (iOS, Android), desktop computers (Windows, macOS, Linux), or web browsers.Consider the target audience and their preferred devices to prioritize platform selection.

Build Platform-Specific Versions:

Develop platform-specific versions of the app using suitable frameworks and technologies.For mobile apps, use platforms like Android Studio (for Android) or Xcode (for iOS) to develop native applications.

Testing and Quality Assurance:

Conduct thorough testing of the deployed app to ensure functionality, performance, and reliability.Perform unit testing, integration testing, and end-to-end testing to identify and fix any issues or bugs.

Security Measures:

Implement security measures to protect user data and ensure secure communication between the app and backend servers.

App Store Submission (for Mobile Apps):

Prepare the app for submission to relevant app stores (e.g., Google Play Store, Apple App Store) by adhering to their guidelines and policies.Create app listings with descriptive metadata, screenshots, and promotional materials to attract users.

6.7. Chapter Summary

This chapter covers the development of computer program and hardware and software requirements.It also includes information about the user interface and their needs for convenient use of this application.

Chapter 7

TESTING

7.1. Introduction

Testing a speech emotion detection app involves verifying that it accurately recognizes and classifies emotions based on spoken input. The testing process for such an application includes

- Functional Testing
- Performance Testing
- Accuracy Testing
- Robustness Testing
- Usability Testing
- Security and Privacy Testing
- Compatibility Testing

7.2. Testing Methods

Testing a speech emotion detection app requires a combination of manual and automated methods to ensure accuracy, reliability, and usability. Here are some common methods used for testing this application:

1. Manual Testing:

- **User Testing:** Have testers use the application and provide feedback on its accuracy and usability.
- **Expert Evaluation:** Have domain experts assess the app's performance and provide insights into its effectiveness.
- **Real-world Scenarios:** Test the app in real-world environments to evaluate its performance in practical situations.

2. Automated Testing:

- **Unit Testing:** Test individual components or functions of the app to ensure they work correctly in isolation.
- **Integration Testing:** Verify that different modules or components of the app work together seamlessly.

- **Functional Testing:** Test the app's functionality against predefined requirements to ensure it performs as expected.
- **Performance Testing:** Evaluate the app's response time, resource usage, and scalability under various conditions.
- **Security Testing:** Test the app for vulnerabilities such as data leakage or unauthorized access.
- **Usability Testing:** Assess the app's user interface and user experience to ensure it's intuitive and easy to use.
- **Compatibility Testing:** Test the app across different devices, platforms, and environments to ensure compatibility.

3. Data-driven Testing:

- **Data Quality Testing:** Evaluate the quality and diversity of the training data used for emotion detection.
- **Dataset Augmentation:** Enhance the training dataset with additional samples to improve the app's performance.
- **Cross-validation:** Use techniques like k-fold cross-validation to assess the app's performance across multiple datasets.

4. Error Analysis:

- **Confusion Matrix:** Analyze the app's performance by examining the confusion matrix to identify misclassifications and errors.
- **Error Patterns:** Identify common error patterns or misinterpretations made by the app and address them accordingly.

5. Continuous Monitoring:

- Implement monitoring tools to continuously monitor the app's performance in production environments and detect any issues or degradation in performance over time.

7.3. Comparison

The app’s performance and accuracy in speech emotion detection were compared with existing similar apps and tools available in the market. This comparison aimed to assess the app’s effectiveness and identify any areas for improvement.

7.4. Software Evaluation

7.4.1. Testing Strategy

A testing strategy was developed to guide the testing process. This strategy outlined the objectives, scope, and approach to be followed during testing. It also defined the resources, timelines, and responsibilities for each testing phase.

7.4.2. Test Plans

Detailed test plans were created, specifying the test scenarios, test data, and expected outcomes for each test case. These test plans provided a systematic approach to cover different aspects of the app's functionality and performance.

7.4.3. Test Cases

Test cases were designed to validate specific features, functionalities, and scenarios of the app. Each test case included a description, preconditions, inputs, expected results, and actual results. The test cases covered a wide range of emotion detection via speech to ensure the app's robustness in accurately detecting the emotions.

7.4.4. Test Report

After conducting the tests, a comprehensive test report was generated. This report summarized the test results, including any issues or bugs encountered during testing. It also provided recommendations for improvements or enhancements to enhance the app's performance and accuracy.

7.5. Chapter Summary

This chapter discussed the testing process of the Ocular Disease Recognition app. It covered the testing methods employed, software evaluation strategies, and the creation of test plans and test cases. The comparison with existing standards and the generation of a test report are also discussed.

REFERENCES

[1] MDPI. , “Speech Emotion Recognition with Deep Learning: A Review” Sensors, 2021.

[2] Liu C., Wang Z., & Zhang L. , “Speech emotion recognition: Features and classification models” International Journal of Data Science and Analysis, 2020.

[3] S. P. Maity, M. H. Kolekar, and S. R. Mahadevappa ,“Deep Convolutional Neural Network Based Regression Approach for Robust Emotion Recognition from Speech” Journal of Applied Soft Computing,2020.

[4] M. Soleymani, M. Pantic, and T. Pun, “A Survey on Speech Emotion Recognition: Features, Classification Schemes, and Databases,” *Jouranl of Speech language,* 2012.

[5] I. Mollahosseini, M. H. Mahoor, and R. G. Ward, “Speech Emotion Recognition Using Deep Neural Network and Extreme Learning Machine,” Journal of Signal Processing Systems, 2016.

APPENDIX (Optional)