MedSound AI: A CNN-LSTM Framework for Accurate Respiratory Disease Classification from Lung Sounds

HASEEN ULLAH
01-136221-034
SAMEER ANWAR
01-136221-025

Supervisor: **Ms. Aima Zahoor**

Co-Supervisor: **Dr. Adil Khan**

Bachelor of Science in Artificial Intelligence

Department of Computer Science
Bahria University, Islamabad

December 2025

Certificate

We accept the work contained in the report titled “MedSound AI: A CNN-LSTM Framework for Accurate Respiratory Disease Classification from Lung Sounds”, written by Haseenullah & Sameer Anwar as a confirmation to the required standard for the partial fulfilment of the degree of Bachelor of Science in Artificial Intelligence.

Approved by:

Supervisor: Ms. Aima Zahoor

Co-Supervisor: Dr. Adil Khan

Internal Examiner: Name of the Internal Examiner

External Examiner: Name of the External Examiner

Project Coordinator: Dr. Sohail Akhtar

Head of the Department : Dr. Muhammad Khurram Ehsan

December 28^{th}, 2025

Abstract

Respiratory diseases, including pneumonia and asthma together with bronchitis, create a significant worldwide health issue that particularly affects areas with limited access to trained healthcare professionals. Medical practitioners depend on their clinical skills to assess patients through lung auscultation which results in medical errors and procrastinated diagnosis. The MedSound AI project develops an automated intelligent system that detects respiratory diseases by analyzing lung sound recordings with high accuracy and early detection capabilities.

The system uses a hybrid deep learning architecture which combines Convolutional Neural Networks (CNNs) for audio spectral feature extraction with Long Short-Term Memory (LSTM) networks that analyze respiratory cycle time-based patterns. The model uses publicly accessible lung sound recordings to achieve 94 percent accuracy when identifying different respiratory conditions. The proposed approach reduces the need for manual feature engineering and enhances the robustness of audio-based analysis.

The trained model has been integrated into a user-friendly interface which enables both clinicians and non-specialists to upload lung sound recordings and obtain instant classification results. The framework provides three main deployment options which include accessible and portable and scalable solutions that meet requirements for both clinical settings and resource-limited environments. MedSound AI uses deep learning together with cost-effective digital solutions to help diagnosis occur quickly while bettering patient outcomes and establishing fair healthcare access for all.

Acknowledgments

All praise is for Allah Almighty, who blessed us with the chance to learn, grow, and complete this project with dedication and clarity. We express our deepest gratitude to our parents and teachers because their support and guidance have shaped our academic journey. We express our sincere appreciation to project supervisor Ms. Aima Zahoor and co-supervisor Dr. Adil Khan because they provided us with continuous support and guidance together with their valuable feedback during this project. We acknowledge our friends who supported us during our times of difficulty.

HASEEN ULLAH & SAMEER ANWAR
Bahria University
E-8, Islamabad, Pakistan

December 2025

"We think someone else, someone smarter than us,
someone more capable, someone with more resources will solve that problem.
But there isn't anyone else."

Regina Dugan

Contents

List of Figures

List of Tables

Acronyms and Abbreviations

AI	Artificial Intelligence
ALS	Adventitious Lung Sounds
API	Application Programming Interface
CAS	Continuous Adventitious Sounds
CNN	Convolutional Neural Network
COPD	Chronic Obstructive Pulmonary Disease
CSV	Comma-Separated Values
DAS	Discontinuous Adventitious Sounds
DNN	Deep Neural Network
GPU	Graphics Processing Unit
HTTP	Hypertext Transfer Protocol
ICBHI	International Conference on Biomedical Health Informatics
IoT	Internet of Things
JSON	JavaScript Object Notation
LSTM	Long Short-Term Memory
MFCC	Mel-Frequency Cepstral Coefficients
MoE	Mixture of Experts
OS	Operating System
PDF	Portable Document Format
RAM	Random Access Memory
SDK	Software Development Kit
TCN	Temporal Convolutional Network
URTI	Upper Respiratory Tract Infection
WHO	World Health Organization
XAI	Explainable Artificial Intelligence

Chapter 1

Introduction

1.1 Background

Respiratory diseases which include asthma and pneumonia and bronchitis and chronic obstructive pulmonary disease (COPD) represent significant global health challenges throughout the world. Medical professionals use stethoscope auscultation to identify these diseases which they need to diagnose. The process requires clinician expertise so their skill level determines success. The need for trained respiratory specialists in this area results in diagnostic delays because of its subjective nature.

The current advancements in artificial intelligence (AI) and deep learning enable researchers to conduct automatic lung sound analysis through recorded audio materials. Deep learning methods established today create dependable systems which detect unusual respiratory behaviors and proceed to classify medical conditions. The CNN-LSTM hybrid model achieves superior audio classification performance because it enables systems to acquire both spectral characteristics and time-dependent patterns present in lung sound recordings.

The **MedSound AI** project intends to use these technologies for developing systems which will automatically detect respiratory diseases. The system uses deep learning technology together with an easy to use interface which provides fast and reliable diagnostic solutions that hospitals and telehealth services can use.

1.2 Problem Statement

Many remote areas experience challenges in obtaining reliable and fast respiratory testing services. The accuracy of traditional auscultation methods depends on the skills of trained healthcare professionals which results in various diagnostic outcomes throughout different areas. The MedSound AI system solves this problem through its automated system which identifies respiratory conditions by analyzing recorded lung sounds.

The system analyzes audio signals through its hybrid CNN-LSTM architecture which detects both frequency patterns and temporal patterns of the audio signals. The system can accurately identify four different diseases which include asthma and pneumonia and bronchiectasis and COPD. The system decreases the need for expert analysis because it enables faster diagnosis which results in better healthcare access for patients.

1.3 Objectives

The primary objectives of this project are:

- Develop a deep learning model which can predict respiratory diseases through lung sound recordings.
- The system will provide healthcare professionals with diagnostic tools which deliver accurate results at an early stage.
- The solution will decrease the time needed for diagnosis while enhancing operational efficiency in both clinical settings and remote locations.
- The system offers hospitals clinics and rural healthcare facilities an affordable AI-based solution.
- The application will provide users with a straightforward design which works on multiple platforms.

1.4 Project Scope

The scope of this project includes the following components:

- **Dataset:** The project requires public datasets which include ICBHI 2017 and Chest Wall Lung Sound Database and RespiratoryDatabase@TR and selected Kaggle datasets.
- **Diseases Covered:** The study investigates seven respiratory diseases which include asthma and COPD and pneumonia and bronchiectasis and bronchiolitis and URTI and healthy participants.
- **Model Development:** The project requires designing deep learning models through special deep learning methods which use CNN-LSTM as their main structure and use EfficientNet as their backup structure.
- **Performance Metrics:** The system uses accuracy and precision and recall and F1-score for its evaluation process.
- **Real-Time Deployment:** The system enables instant lung sound analysis through its model which works with both web applications and mobile applications.

1.5 Project Workflow

The deep learning model uses user lung sound recordings which are processed by MedSound AI to determine their matching respiratory condition. The CNN-LSTM architecture extracts audio signal spectral and temporal information which leads to improved disease classification performance. The system operates through its goal of making accurate medical condition predictions which match actual patient health states.

The proposed system operates according to these summary workflow steps:

- The user uploads a lung sound recording.
- The audio is preprocessed and relevant features are extracted.
- The trained CNN-LSTM model analyzes the processed signal.
- The system outputs the predicted respiratory disease along with additional insights.

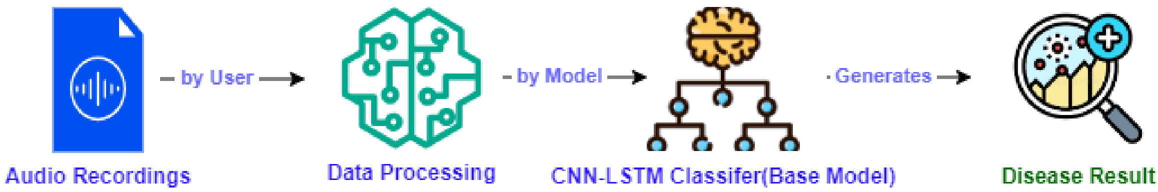

Figure 1.1: Workflow of the MedSound AI Model

1.6 Summary

MedSound AI develops a deep learning program which solves respiratory diseases that pose serious threats to public health. The project required urgent development because doctors in developing countries experience two major issues with traditional auscultation techniques which require excessive skill from medical personnel. The project will use Artificial Intelligence through its hybrid CNN-LSTM framework which will assess and categorize lung sound recordings based on their complex spectral and temporal properties.

This chapter's primary goal demands researchers to create an affordable and dependable diagnostic system which delivers exact medical assistance to doctors in real time. The research defined system boundaries which showed that public datasets could be used to detect seven different respiratory disorders. The documented workflow describes how the system functions starting from lung sound recording input and ending with automatic disease prediction output.

Chapter 2

Literature Review

Asthma and pneumonia together with chronic obstructive pulmonary disease (COPD) represent common health conditions that affect people from every part of the world. The medical conditions need immediate diagnosis and treatment because they can develop into severe or life-threatening health issues. Physicians need quick and accurate diagnosis methods because these methods protect patients from complications while they support better recovery outcomes.

Doctors use auscultation to diagnose lung diseases through their stethoscope based technique which enables them to hear patients breathing sounds. The standard method for respiratory evaluation depends on this technique yet its results depend on the assessment skills of experienced doctors. The two doctors will achieve different results after hearing the same lung sounds because the method allows both false and delayed medical evaluations to occur.

Artificial intelligence development through deep learning technology created new analysis methods which enable automatic lung sound analysis with greater objectivity. Deep learning systems enable the detection of extremely small acoustic patterns that human beings cannot hear. The healthcare industry can use these technologies to help professionals access clearer and more dependable information which supports their decision-making process. AI-based solutions provide cost-effective respiratory evaluation methods which enable people to access superior respiratory function tests in areas that lack trained medical professionals.

2.1 Classification of Lung Sounds

The classification system for respiratory sounds divides them into two groups which include regular lung sounds and adventitious lung sounds (ALSs) normal lung sounds encompass bronchial and vesicular breathing which show proper airflow and normal lung function Adventitious sounds which include wheezes and crackles and rhonchi and stridor present as abnormal sounds that indicate existing medical conditions Each ALS type has specific clinical significance; for instance, wheezes suggest airway obstruction common in asthma or COPD, while crackles may indicate fluid accumulation, pulmonary fibrosis, or early pneumonia. The anatomical source and disease mechanism of an individual case determine the acoustic properties which include pitch and duration and respiratory cycle timing and sound intensity [1]

Figure 2.1 demonstrates different adventitious lung sounds. Stridor originates in the upper airway and is mostly inspiratory. Wheezes and rhonchi arise from narrowed bronchi, with wheezes typically heard during expiration, while crackles occur in smaller airways, predominantly during inspiration [1].

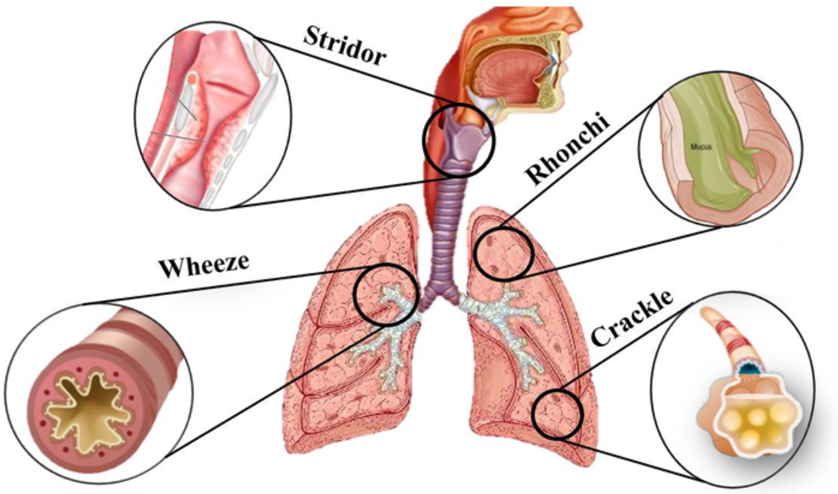

Figure 2.1: Classification of Lung Sounds

2.2 Traditional Approaches to Respiratory Sound Analysis

The first systems for classifying respiratory sounds used statistical methods together with manually developed audio features that included pitch and energy and frequency components. The audio methods needed to solve complex sound problems because their basic design failed to handle intricate sound patterns that changed over time and through different sound frequencies. The study by Altan et al. [2] developed the RespiratoryDatabase@TR which serves as an annotated lung sound dataset that researchers use to study audio-based diagnostic methods. The existing methods still show problems because they become vulnerable to background noise while they fail to work correctly with different types of data.

2.3 Deep Learning in Respiratory Sound Classification

Deep learning has altered respiratory sound analysis by creating automated classification systems which eliminate the need for manually developed sound detection features. The combination of CNNs and LSTMs enables machines to comprehend complex sound patterns which exist in both original audio and audio that has been processed.

The developed models enable lung sound evaluation through automated procedures which do not need operators to manually create measurement parameters.

Shuvo et al. created a lightweight CNN architecture which maintains computational efficiency throughout mobile and embedded systems while achieving high classification accuracy [3].

Pham et al. created a CNN-MoE (Mixture of Experts) system which uses multiple dedicated subnetworks to identify various respiratory disorders. The system exceeded existing performance levels by demonstrating the advantages of using modular design together with expert system components [4].

The research findings demonstrate that deep learning has the ability to improve respiratory disease detection methods. The combination of CNN-based spatial feature extraction and LSTM-based temporal modeling through specialized subnetworks establishes a framework for real-time intelligent systems which can scale to meet the needs of MedSound AI.

2.3.1 Key Studies on Respiratory Sound Classification

Jin et al. (2014) developed advanced spectro-temporal feature extraction techniques which enhanced classification results during tests conducted in noisy conditions. The method used by them combined frequency-domain and time-domain features to create a comprehensive lung sound representation which later impacted audio-based deep learning research according to [5].

Fernando et al. (2022) developed a Temporal Convolutional Network (TCN) system specifically designed to process lung sound recordings. The system succeeded in understanding complete time periods while maintaining full sequence data which resulted in better system comprehensibility and better classification results according to [6].

2.4 Hybrid CNN-LSTM Models

Researchers developed hybrid systems that use both CNN and LSTM networks to process visual data and temporal information. The study of Fernando et al. [6] proved that Temporal Convolution Networks enhance understanding of their output, which serves as a vital requirement for medical use cases. Acharya and Basu [7] showed that personalized adjustments to wearable technology for patients brought two advantages, which improved both actual performance and dependable operation of the devices.

2.5 Public Datasets and Acoustic Techniques

The development of strong models requires high-quality datasets which function as essential training materials. Rocha et al. [8] developed an open-source respiratory sound database which researchers now use as a standard benchmark to test lung sound classification systems. Rao et al. [9] conducted an extensive study on lung sound acoustic analysis methods which they found useful for clinical applications and automated respiratory disease diagnosis. The studies demonstrate that precise data sources and specific sound characteristics lead to models which can accurately predict outcomes in various situations.

2.6 Global Need for AI-Based Solutions

WHO World Health Organization states that more than 10 million people die each year because of respiratory diseases according to research from ICS10. The statistics demonstrate an urgent requirement for diagnostic tools which can be automated to scale at low costs because these tools will enhance healthcare access in developing regions. The limited resources available in these environments create problems for medical professionals because they need equipment which is not accessible to them. Hospitals can use automated systems which analyze lung sounds to deliver quick and affordable screening services that require no invasive procedures at their facilities.

2.7 Summary of Literature Review

The chapter examined how respiratory sound classification developed from its origins with manual methods to its current state which uses deep learning technologies. The study presents two main results which include lightweight CNN models that enable fast sound classification and hybrid CNN-LSTM systems that examine lung sound patterns over time. High-quality datasets together with both MFCCs and spectrograms as feature extraction methods and patient-specific tuning approaches form the foundation of the literature. Researchers have made substantial advances in their work but they still encounter challenges which arise from overlapping frequencies and environmental noise plus their shortage of labeled data. MedSound AI demonstrates the need for intelligent scalable solutions to address these issues.

Table 2.1: Summary of Related Work

Year	Paper Name	Methodology	Limitations / Remarks
2024	LungAdapter [10]	Adapter-augmented AST	State-of-the-art, but transformer-based models are resource-intensive
2023	M2DViT-Fusion Transformer [11]	Vision Transformer fusion	Better feature learning, but expensive to train
2023	BLNet: Bias-Reduced Neural Network [12]	DNN with blocking variable	Improved fairness, but model is complex
2022	Co-tuning and Stochastic Normalization [13]	Pretrained ResNet with co-tuning	Improved ICBHI score, but training is resource-demanding
2022	Hybrid CNN-LSTM with Focal Loss [14]	CNN + LSTM with SpecAugment	High accuracy, but computationally heavy
2022	Robust Temporal Convolution Network [6]	Temporal Convolution Network (TCN)	Interpretable, but requires fine-tuning for real-time
2021	Deep CNN for Crackle, Wheeze Detection [15]	VGG16-based CNN	Good accuracy, dataset limited to clinical recordings
2021	RespireNet [16]	ResNet-34 with augmentation	Achieved high ICBHI score, but requires large data
2021	CNN-MoE Framework for Anomaly Classification [4]	CNN + Mixture of Experts	High accuracy, but complex model training
2020	Deep Neural Network for Wearable Devices [7]	Patient-specific DNN tuning	Personalized, but computationally intensive
2020	Lightweight CNN Model for Detecting Respiratory Diseases [3]	Compact CNN architecture	Efficient, but limited generalization to large datasets

Chapter 3

System Requirements

The system needs and specifications of MedSound AI are outlined in this chapter. The system requires deep learning models to function properly with an easy-to-use and effective deployment system. The lung sound classification system includes three types of requirements which are functional requirements and non-functional requirements and technical requirements together with a feasibility study.

3.1 Existing System

The medical field continues to use conventional diagnostic techniques which include the practice of manual lung auscultation. The methods depend on doctor expertise which creates subjectivity in their results. The inability of resource-limited regions to provide access to trained pulmonologists and advanced diagnostic tools causes both diagnostic delays and incorrect results. AI-based audio classifiers exist in the market but they contain multiple significant limitations:

- The system detects only anomalies which include wheezes and crackles instead of performing disease classification.
- The system needs heavy computational resources which make it impossible to use in real-time systems or mobile devices or web platforms.
- The system requires clinical validation because it cannot handle large-scale operations which need to be expanded.

3.2 Proposed System

MedSound AI addresses these limitations by employing a hybrid CNN-LSTM architecture for the classification of common respiratory diseases, including asthma, pneumonia, and COPD, from lung sound recordings.

Key Features of the Proposed System:

- Accepts audio recordings (.wav or .mp3) of lung sounds.
- Preprocesses data with denoising, MFCC, or spectrogram extraction.
- Utilizes CNN layers for spatial feature extraction.
- Employs LSTM layers to model temporal patterns.
- Outputs disease classification with associated confidence scores.
- Provides a web interface for audio upload and instant predictions.

3.3 Requirement Specifications

Requirement specifications describe what a software system should do and how it should behave. The requirements give developers precise instructions which they need to construct the correct system.

3.3.1 Functional Requirements

The functional requirements define the essential functions that MedSound AI must perform, which include audio uploading, lung sound processing, and diagnostic result generation.

Table 3.1: System Requirements Description

Requirement	**Description**
User Registration/Login	Users can register and securely log in to the system.
Upload Lung Sound File	Users can upload audio recordings (e.g., .wav, .mp3) for analysis.
Audio Preprocessing	The system performs noise reduction, feature extraction (MFCC/spectrogram), and normalization.
Disease Classification	The system analyzes the lung sound using a CNN-LSTM model and classifies it (e.g., Asthma, Pneumonia, COPD).
View Prediction Results	Users can view the disease prediction, along with confidence scores and spectrogram visualizations.
Error Handling	The system displays appropriate error messages for invalid file uploads or prediction failures.

3.3.2 Non-Functional Requirements

Non-functional requirements define the performance, reliability, usability, and security characteristics that MedSound AI must satisfy to operate effectively in real world environments. These requirements ensure that the system is responsive, stable, user friendly, and secure providing consistent and trustworthy results during practical use.

Table 3.2: Non-Functional Requirements of MedSound AI

Requirement	Description
Performance	Classification and result generation should complete within a few seconds per file.
Scalability	The system should support multiple concurrent users uploading and analyzing lung sound files.
Availability	The system should maintain at least 99% uptime during operational hours.
Security	Uploaded audio files and prediction results must be securely stored and transmitted.
Usability	The interface should be clean, responsive, and user-friendly across desktop and mobile devices.
Maintainability	The codebase should be modular and well-documented to allow easy debugging and future upgrades.
Portability	The system should be compatible with modern browsers and deployable on cloud or local servers.
Backup & Recovery	Prediction results and user data should be recoverable in the event of a system crash or failure.

3.4 System Use Case Modeling

3.4.1 Use Case Diagram

Use case diagrams help identify interactions between system users (actors) and the system, clarifying functional requirements.

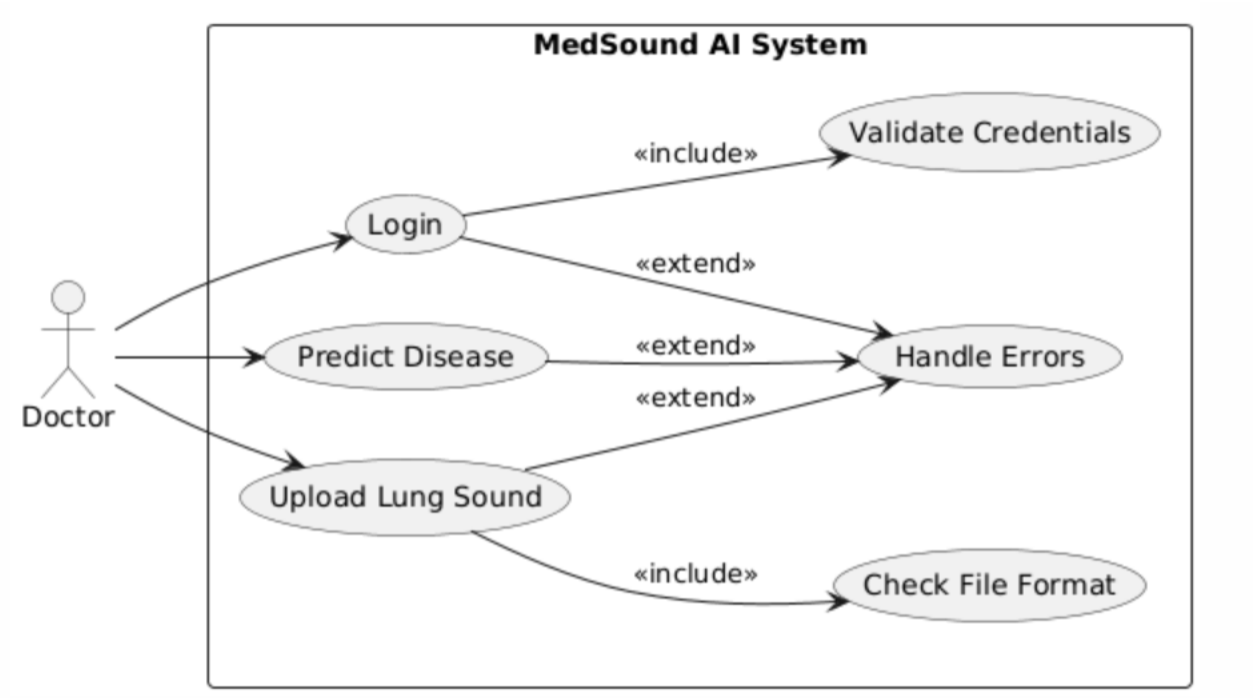

Figure 3.1: Use Case Diagram for MedSound AI

3.4.2 Use Case Description

Use case descriptions provide detailed explanations of how actors (e.g., doctors) interact with the system. They define the actor, their goals, main workflow, preconditions, postconditions, and alternative flows. Use cases for MedSound AI include:

- Uploading lung sound files
- Automatic preprocessing and classification using CNN-LSTM
- Viewing and downloading prediction results
- Accessing historical predictions for follow-up or review

3.4.3 Use Case 01: Login

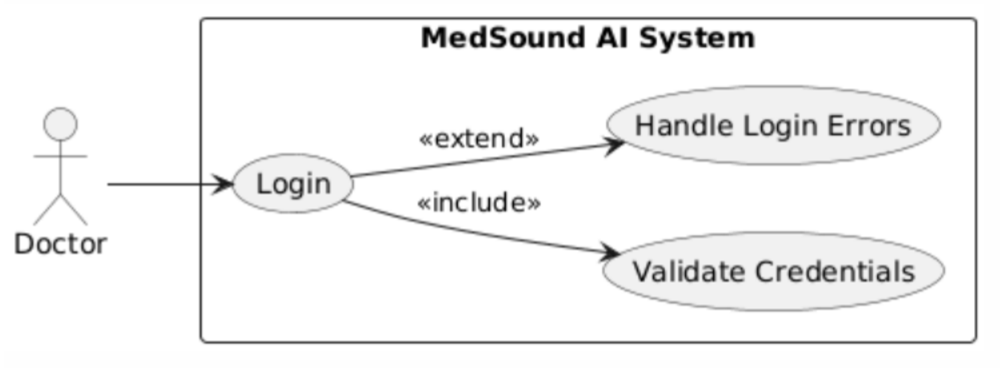

Figure 3.2: Login and Registration Use Case Diagram

Table 3.3: Use Case Specification: Login

Field	Description
Use Case Name	Login
Use Case ID	UC-0001
Actor	Doctor (User)
Description	Doctor logs into the MedSound AI system using valid credentials.
Pre-Conditions	1. Doctor must be registered in the system. 2. Doctor must have a valid email and password. 3. System must be connected to the database.
Trigger	Doctor clicks the “Login” button.
Normal Flow	1. Doctor enters email and password. 2. System validates the credentials. 3. System grants access. 4. Doctor is redirected to the dashboard.
Alternative Flow	1. Doctor enters invalid credentials. 2. System displays an error message.
Post-Conditions	1. Doctor is successfully logged into the system. 2. User session is created. 3. Dashboard is displayed to the doctor.

3.4.4 Use Case 02: Upload Lung Sound

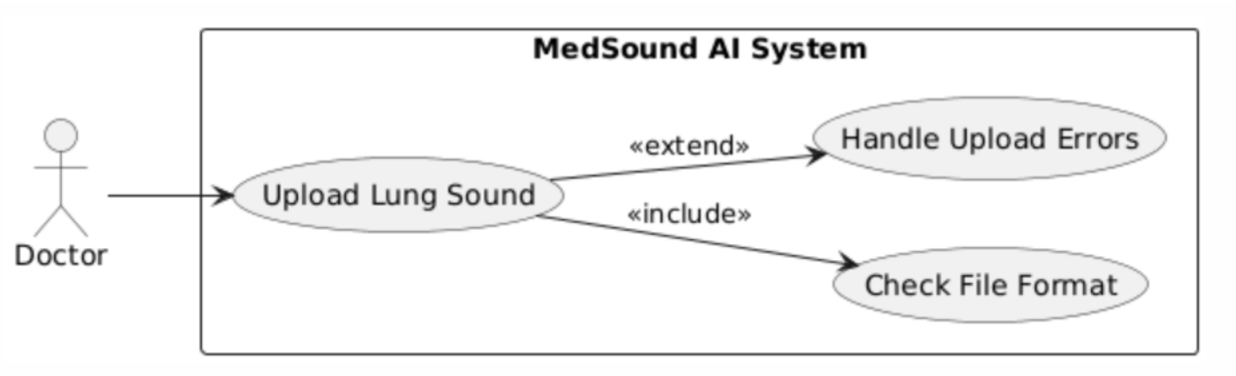

Figure 3.3: Uploading Lung Sound Use Case Diagram

Table 3.4: Use Case Specification: Upload Lung Sound

Field	**Description**
Use Case Name	Upload Lung Sound
Use Case ID	UC-0002
Actor	Doctor (User)
Description	Doctor uploads a lung sound file (e.g., .wav) for analysis in the MedSound AI system.
Pre-Conditions	1. Doctor must be logged into the system. 2. Doctor must have access rights to upload lung sound files. 3. Audio file must exist and be in a supported format (e.g., .wav).
Trigger	Doctor selects the “Upload Audio” option.
Normal Flow	1. Doctor clicks on upload. 2. Selects the audio file from the device. 3. System processes and stores the file. 4. System confirms successful upload to the doctor.
Alternative Flow	1. File format is invalid or audio is corrupted. 2. System displays an error message and aborts upload.
Post-Conditions	1. Audio file is successfully stored in the system database. 2. File is ready for analysis. 3. Doctor receives confirmation of successful upload. 4. If upload fails, no file is stored and an error is displayed.

3.4.5 Use Case 03: Predict Disease

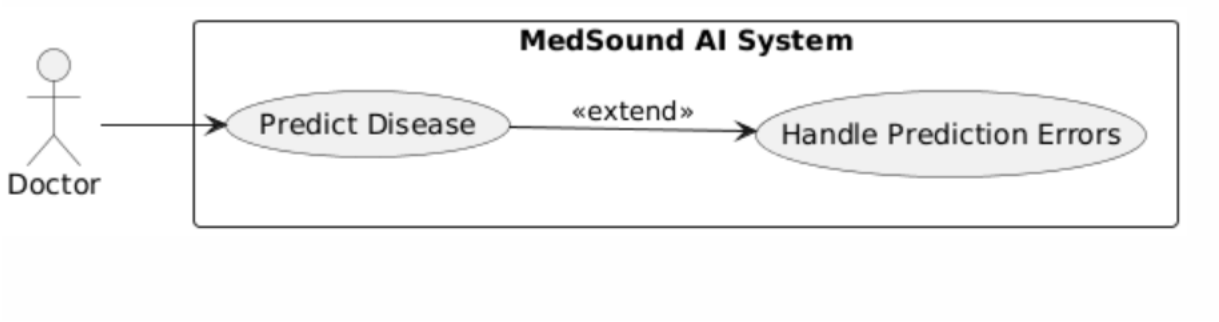

Figure 3.4: Predicting Disease Use Case Diagram

Table 3.5: Use Case Specification: Predict Disease

Field	**Description**
Use Case Name	Predict Disease
Use Case ID	UC-0003
Actor	Doctor (User)
Description	Doctor receives a classification result for an uploaded lung sound using the CNN-LSTM model.
Pre-Conditions	1. Doctor must be logged into the system. 2. Lung sound file must have been successfully uploaded. 3. System's CNN-LSTM model must be ready for prediction.
Trigger	Doctor clicks the "Predict" button after uploading a lung sound.
Normal Flow	1. System processes the uploaded lung sound using the CNN-LSTM model. 2. System returns the predicted respiratory condition. 3. Doctor views the prediction result.
Alternative Flow	1. Uploaded file is invalid or corrupted. 2. CNN-LSTM model encounters an error. 3. System displays an error message and aborts prediction.
Post-Conditions	1. Prediction result is displayed to the doctor. 2. System logs the prediction activity. 3. If prediction fails, no result is stored and an error is shown.

3.5 System Sequence Diagram

A system sequence diagram shows how system components interact with each other during different time periods. The system shows how users interact with the user interface and preprocessing modules and the classification system to finish their diagnostic work.

The user starts the audio file processing by uploading an audio file to the interface which starts the preprocessing sequence that includes three steps: noise reduction and normalization and MFCC extraction. The CNN–LSTM model receives the processed audio features to detect lung sounds through both spatial and temporal analysis. The classifier receives the model prediction which sends the final output result back to the user interface for display purposes.

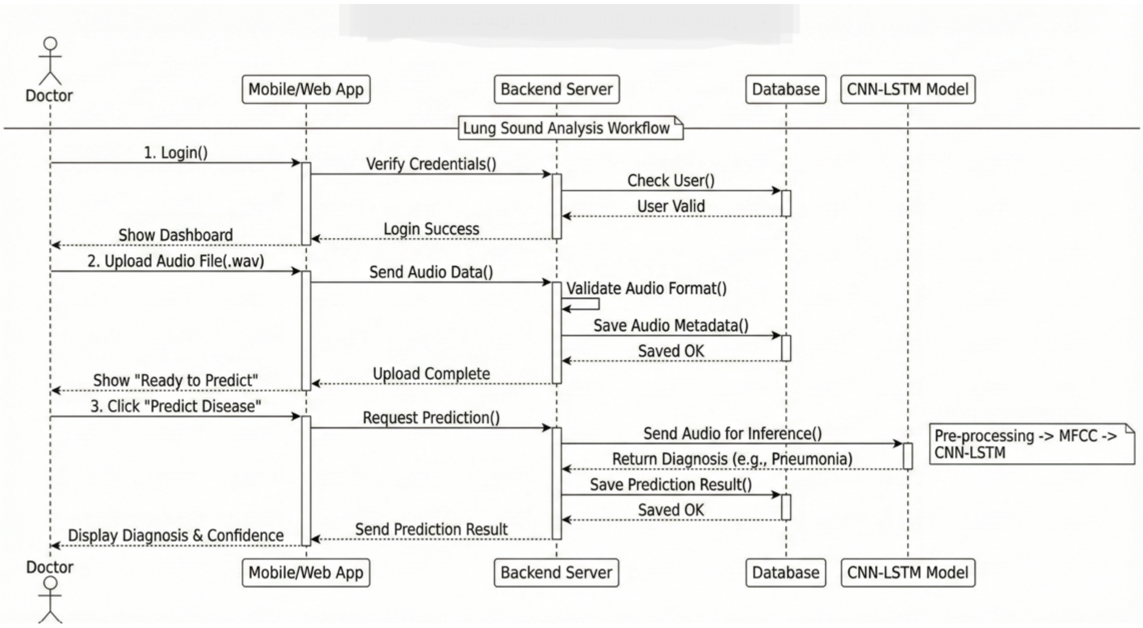

Figure 3.5: System Sequence Diagram

Login SD

A doctor can access the MedSound AI system through the Login use case by using their valid login information. The sequence diagram (Figure 3.6) shows how the doctor interacts with the system. The doctor enters their email and password, and the system validates the credentials. The system grants access to the doctor and redirects them to the dashboard when their credentials are valid, but when their credentials are invalid, the system shows an error message.

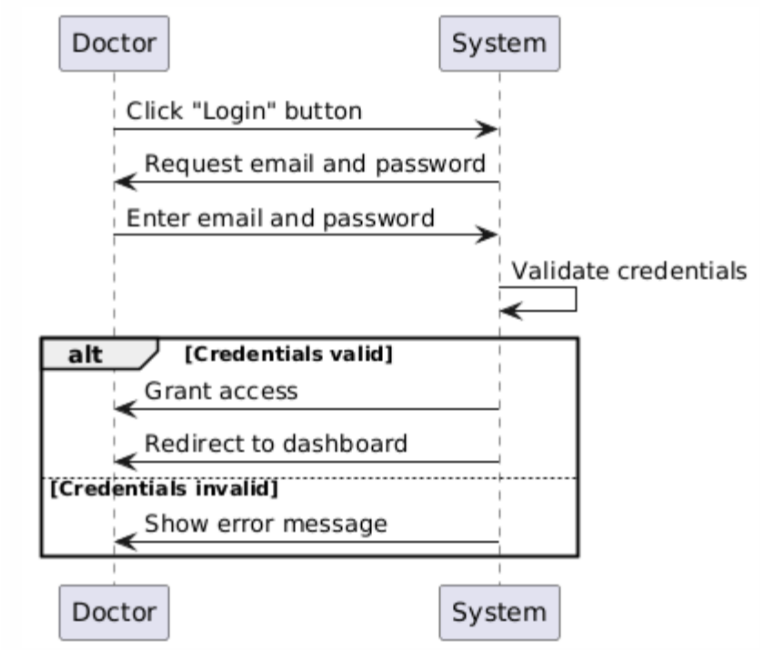

Figure 3.6: Sequence Diagram for Login Use Case

Upload Lung Sound SD

The Upload Lung Sound use case enables the doctor to upload a lung sound file for analysis. The doctor begins the process by selecting the "Upload Audio" option which lets them pick a file from their device according to the sequence diagram in Figure 3.7. The system processes and stores the file. The system displays a confirmation when the file format is valid but shows an error message when the file format is invalid.

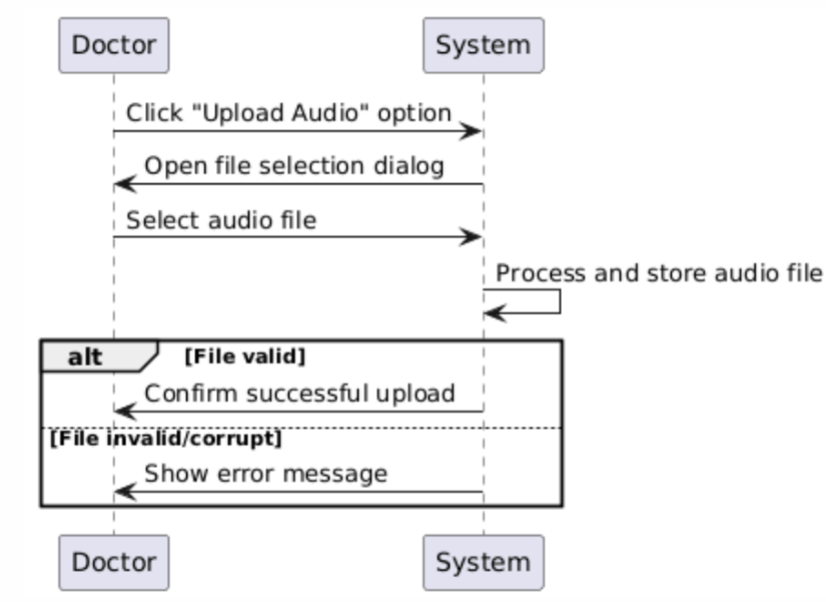

Figure 3.7: Sequence Diagram for Upload Lung Sound Use Case

Predict Disease SD

The Predict Disease use case enables doctors to obtain classification results after they submit lung sound recordings for analysis. The sequence diagram (Figure 3.8) shows that the doctor activates the "Predict" button which leads the system to analyze the audio through its CNN-LSTM model and display the expected respiratory condition. The system displays the correct error message when it encounters any errors during operation.

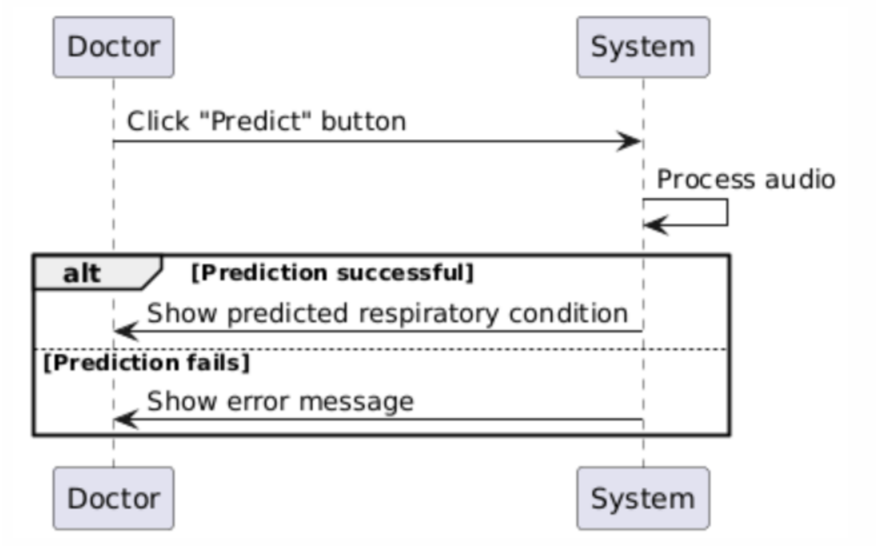

Figure 3.8: Sequence Diagram for Predict Disease Use Case

3.6 Summary

The MedSound AI system required detailed analysis through its entire functional analysis and modeling process. The key components covered include the identification of system actors, use case specifications, and interaction sequences. The three main operational scenarios which were described included *Login*, *Upload Lung Sound*, and *Predict Disease*.The system behavior expectations were clearly defined through the pre-conditions and triggers and normal and alternative flows and post-conditions which described each use case.

The system required particular sequence diagrams for each operational scenario because these diagrams displayed how doctors interacted with the system through each operational step. The diagrams provide insights into how operations function together with their error handling methods and system response mechanisms. Chapter 3 builds the system's operational structure because it connects requirement evaluation with subsequent system development.

Chapter 4

Design

The design of **MedSound AI** develops a system that can grow to handle different demands while providing high efficiency for its task of identifying respiratory illnesses through the analysis of lung sound recordings. The chapter provides an explanation of the system's operational framework which includes its architectural design and subsystem components together with its design requirements and operational diagrams.

4.1 System Architecture

The system architecture diagram shows all software module interactions through its system architecture diagram. The system displays user interfaces and processing modules and storage components and external systems to show how data moves through the system.

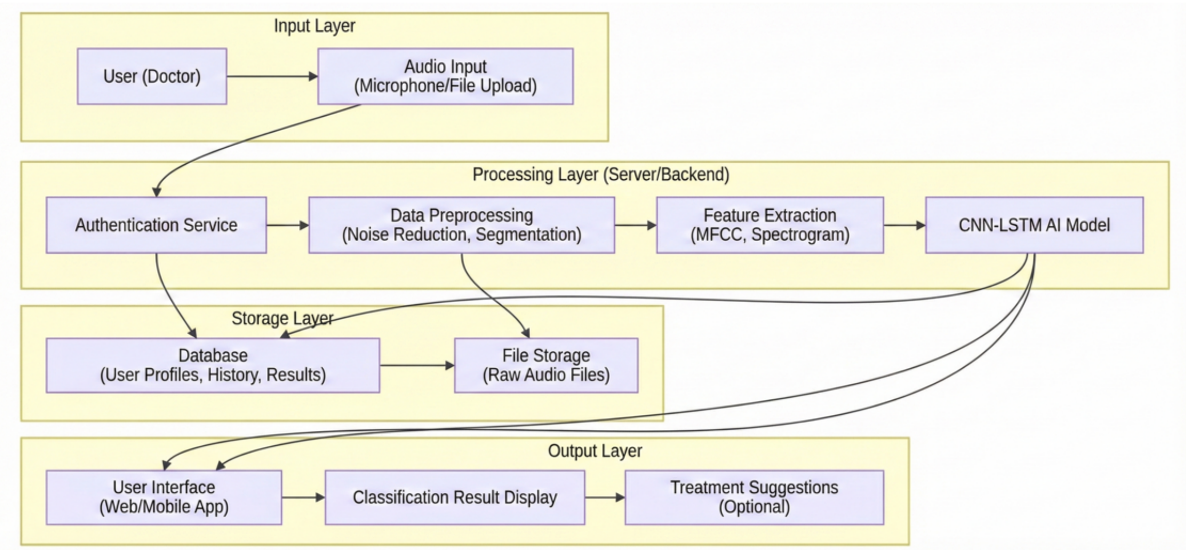

Figure 4.1: System Architecture Diagram

Figure 4.1 presents the complete architectural workflow of MedSound AI. The system performs three main tasks after a user uploads an audio file: it checks the audio format, it saves the recording in a temporary location, and it sends the audio file to the preprocessing module. The system extracts MFCC features which it sends to the CNN–LSTM model for processing. The model displays respiratory condition results to the user after it makes a prediction. The system will offer medication recommendations which depend on the diagnosis when this feature becomes active. The system architecture provides efficient input-to-output operations which ensure system reliability and user accessibility.

4.1.1 Context Diagram

The Context Diagram functions as a visual tool which depicts a system through its single process while showing its connections to external users and systems. The system boundary is defined through a diagram which displays information flow between the system and all its surrounding entities.

The system boundary which Figure 4.2 shows demonstrates the methods through which external users and internal components exchange information. The diagram provides a high-level view of the information flow entering and leaving MedSound AI.

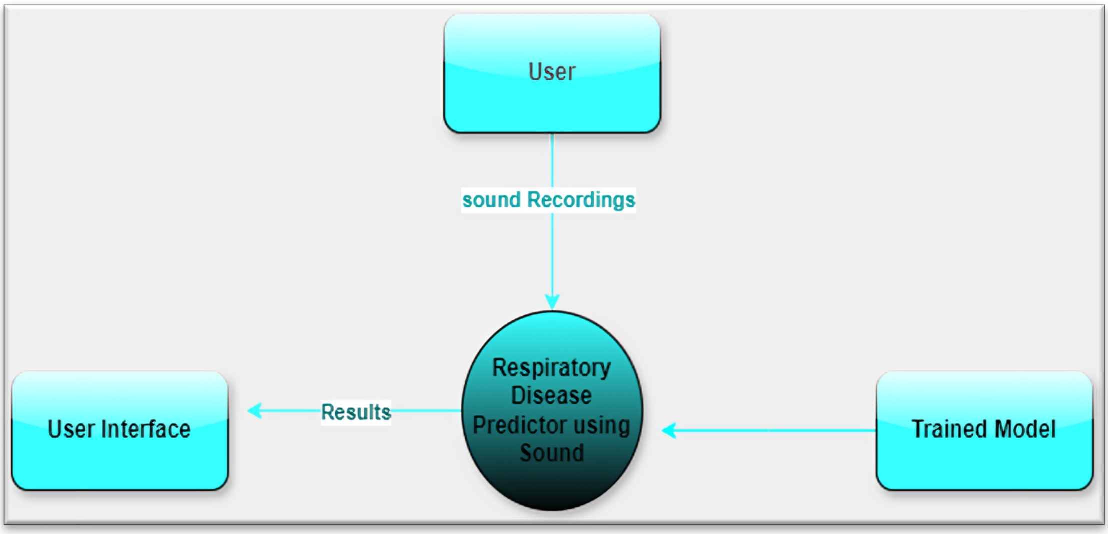

Figure 4.2: Context Diagram

4.2 Design Constraints

Design constraints in Software Engineering are the fixed limitations, rules, or conditions that a system must follow during its development. The constraints direct system design and implementation methods. The final product must fulfill technical requirements and organizational requirements and operational requirements.

4.2.1 Risk Analysis

- **Data Quality:** Limited availability of large, labeled lung sound datasets (e.g., ICBHI).
- **Model Bias:** Imbalanced datasets may reduce generalization across populations.
- **Audio Noise:** Environmental sounds may interfere with classification accuracy.
- **Hardware Constraints:** Real-time processing requires optimized model deployment.
- **Medical Acceptance:** Clinical integration requires regulatory validation and explainable outputs.

4.2.2 Software Requirements

Table 4.1: Software Requirements table

Component	Description
Language	Python 3.8+
Libraries	PyTorch/TensorFlow, Keras, Librosa, NumPy, Pandas, Scikit-learn
Modeling Tools	Google Colab, Jupyter Notebook, PyCharm
Web Framework	FastAPI for backend
Frontend	HTML, CSS, JavaScript

4.2.3 Hardware Requirements

Table 4.2: System Resource Requirements

Resource	Minimum	Recommended
RAM	8 GB	16–32 GB
Processor	Quad-core CPU or higher	Intel i7 9th Gen or higher
GPU	Not required (for training only)	NVIDIA GPU (Colab/Cloud or local CUDA)
Storage	10 GB	10+ GB for datasets and logs
OS	Windows/Linux/Mac	Any with Python 3 support

4.3 Design Methodology

- **Problem Identification:** The project is launched by an understanding about the requirement for the affordable, fast, and accurate solution for diagnosis of respiratory diseases using lung sounds.
- **Data Collection and Preprocessing:** Researchers collected lung sound datasets from two sources which include ICBHI and Kaggle. The recordings underwent cleaning, denoising, and feature extraction (MFCCs, spectrograms) to prepare them for model training.
- **Model Architecture (CNN–LSTM):** The extracted features, which convolutional layers process, enable the layers to detect nearby patterns. The LSTM layers of the system extract time-based features which they use to understand respiratory cycle patterns. The final layer produces the predicted disease class.
- **System Development:** A web interface that is user-friendly was developed using Flask or FastAPI to import audio files and get immediate feedback from the model.
- **Training and Evaluation:** The dataset was split into 70% training, 15% validation, and 15% testing to make sure a fair evaluation is achieved and overfitting is prevented.
- **Deployment:** The trained model can be executed on any resource depending on the computer power: on a local machine or on a cloud server, or, in certain cases, even on a mobile phone.

4.3.1 Design Workflow

MedSound AI system development uses an Agile and Iterative development model for its design workflow. The selected workflow enables the system design and development process to maintain ongoing testing and adaptable development while achieving incremental system enhancements. The project required multiple model iterations because it used deep learning models and processed audio data to enhance classification performance through model architecture improvements.

Development processes in this workflow separate into multiple short development cycles. The cycle starts with gathering requirements and creating a design schedule for development work. The system development process continues with system component design and implementation of the CNN-LSTM model and backend and frontend module system integration. The testing and evaluation process occurs after implementation to assess model accuracy and loss and prediction outcomes. The next iteration begins after evaluation outcomes lead to feedback incorporation into the current system.

The Agile design workflow enabled the team to make multiple updates to the model structure and hyperparameters and preprocessing methods. The system allowed the team to find problems that occurred during the development process which included overfitting and data imbalance before they became problematic in the next development stage. The iterative process enabled both AI model development and complete system design to progress toward project goals in an efficient manner.

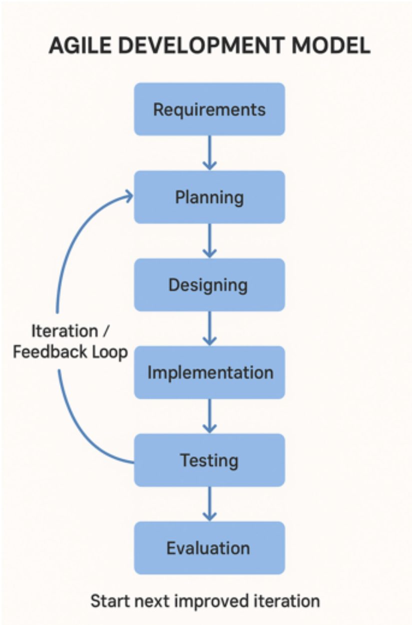

Figure 4.3: Agile Development Model Diagram

4.4 High Level Design

High-Level Design shows system structure through its description of how major components of the system work together. The system provides an overview of its architecture which does not include specific implementation details.

4.4.1 Conceptual Logical Design

The system uses Conceptual Logical Design to create an abstract model which displays its operational functions without revealing its actual implementation methods. The system displays its logical structure through its relationships and operational processes and its established rules.

4.4.1.1 Component Diagram

A Component Diagram shows the primary elements of a system and their interconnections through a basic diagram that depicts their operational relationships. The diagram provides a visual representation of the system's structure by displaying its components without needing to show technical specifications.

Figure 4.4 displays the system main components which work together to complete the diagnostic process. The diagram functions as a development guide which helps maintain clear and consistent component relationships throughout the project.

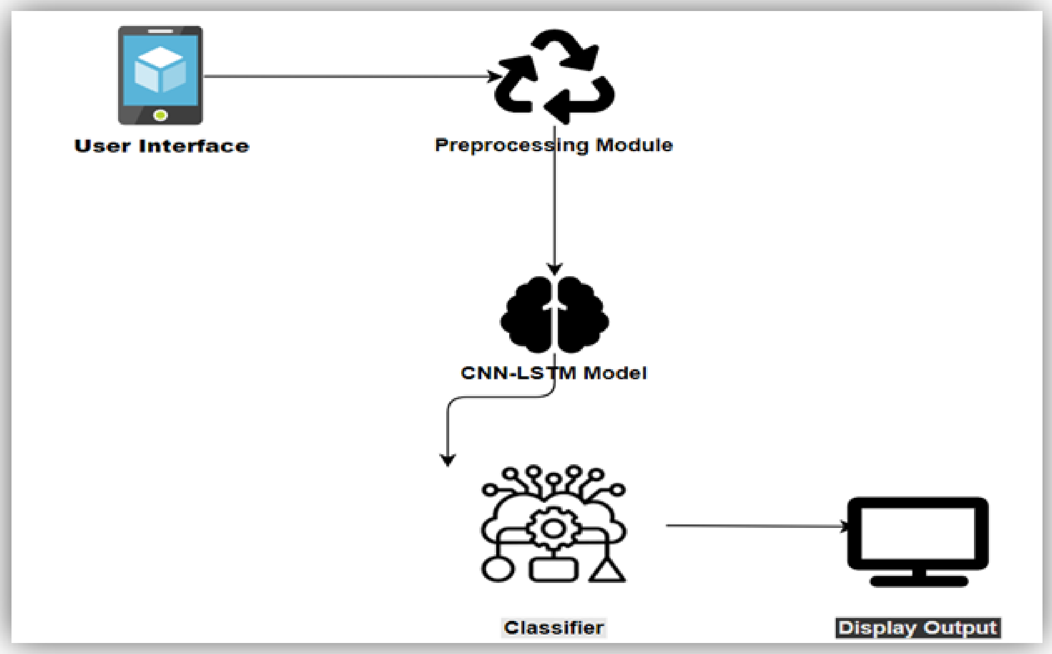

Figure 4.4: Component Diagram

4.4.2 Process

A process refers to the sequence of steps or actions that define how tasks are carried out and how data flows through a system. The system operates through its organized procedures which convert incoming materials into valuable results.

4.4.2.1 Process Interaction Diagram

A Process Interaction Diagram shows all the ways different processes or system components interact with each other. The system shows how information flows through its various processes together with the coordination needed to complete each action required for system operation.

The user starts the system by uploading a lung sound recording which is demonstrated in Figure 4.5. The system processes data through its internal functions which operate in a sequential manner from preprocessing to feature extraction and classification until it determines the correct disease category.

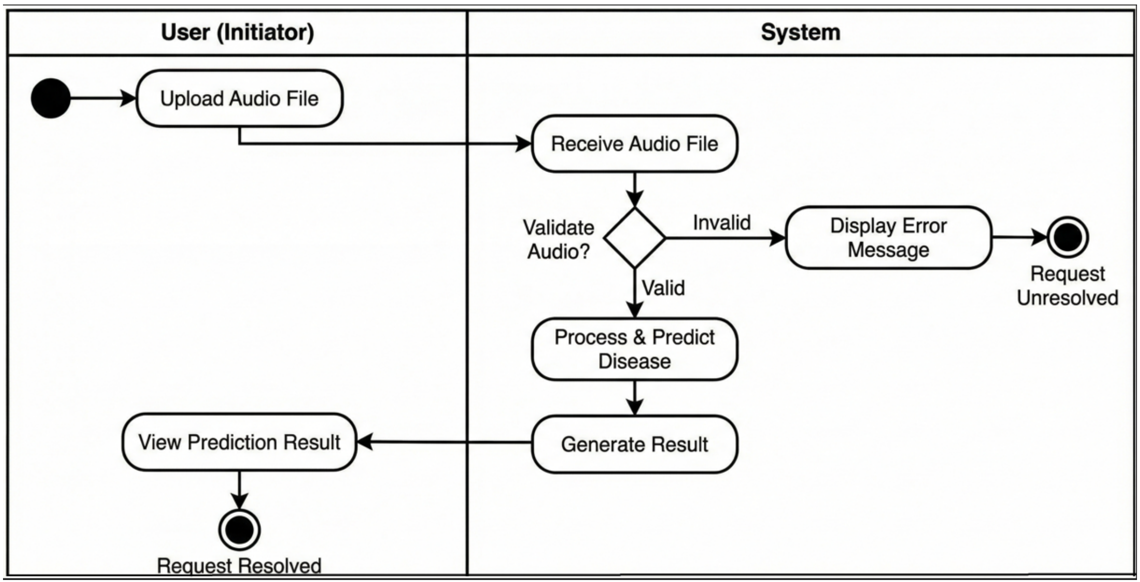

Figure 4.5: Process Interaction Diagram

4.5 Low Level Design

Low-Level Design (LLD) defines the complete system implementation details of all system components. The system's internal operation is described through the definition of algorithms, data structures, control logic, and module interaction data. Low-Level Design provides a clear blueprint for developers who need to implement each module through a process that guarantees proper module implementation and successful system integration with other system parts. The system assists organizations in discovering problems during testing while simplifying testing and debugging processes and all future maintenance work.

4.5.1 Activity Diagram

The activity diagram displays the system's dynamic behavior through its representation of user actions that lead to the completion of the diagnosis process. The system processes user input to create its final output through a series of activities that the user performs. The activity diagram functions like a flowchart because it shows points where users make choices and times when two tasks run at once and points where the system transitions into new operational states.

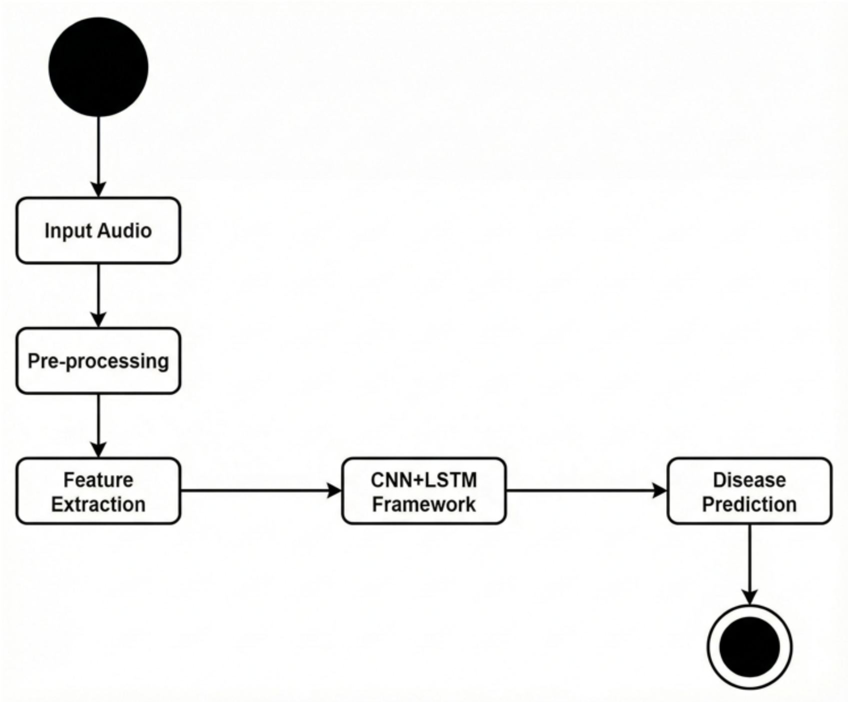

Figure 4.6: Activity Diagram

4.6 Summary

The MedSound AI system design includes system architecture and design constraints and Agile design workflow. The system design includes both high-level and low-level design elements which show its complete operational structure and internal system functions. The design allows the CNN-LSTM model to function seamlessly with both user interface components and backend system elements. The chapter establishes a solid basis which guides both the implementation process and system deployment for the MedSound AI system.

Chapter 5

System Implementation

The step known as system implementation represents the time when engineers build a complete operational system based on their initial design work. The system development process requires developers to write software code while they establish the system environment and connect all system components and test each system part to confirm its operational capabilities. The project team needs to complete three tasks which include software debugging and performance enhancement and module communication testing. The system enters production after developers complete its creation and the system undergoes ongoing surveillance to detect and fix problems. The system requires routine maintenance together with system updates and security improvements to remain operational and protect its performance. The system implementation process requires complete documentation and user training programs and feedback collection systems to ensure that users receive the necessary support and the system meets organizational targets. The system undergoes ongoing evaluation and improvement processes to meet changing business requirements and new technological advancements. The implementation process establishes a system which fulfills its operational requirements while providing value and dependable performance and user-friendly functionality throughout its operational lifespan.

5.1 Integrated Development Environment (IDE)

The development team for MedSound AI accomplished their work by utilizing a set of advanced tools which enabled them to conduct experiments and develop their models with high efficiency.

- **Frontend and Backend IDE:** The entire application, including the Flask backend and the HTML/CSS-based user interface, was implemented using **Visual Studio Code (VS Code)**. Its powerful debugging tools, Python extensions, and integrated terminal made development and testing significantly easier.

- **Model Training Environment: Google Colab** was selected for training the CNN-LSTM model. Its free GPU support and seamless compatibility with TensorFlow and Keras enabled efficient training of deep learning models without requiring high-end local hardware.

5.2 Tools and Technologies

Table 5.1: Tools and Technologies Used in MedSound AI

Technology	**Purpose**
Python [17]	Core programming language for model and backend development
Flask [18]	Web framework for handling routes and integrating the AI model
TensorFlow / Keras [19]	Building and training the CNN-LSTM model
Librosa [20]	Audio feature extraction (MFCCs, Mel spectrograms)
NumPy / Pandas [21]	Data preprocessing and handling
Matplotlib [22]	Visualization of spectrograms and training performance
HTML, CSS, JavaScript [23]	Frontend design and interactivity
Google Colab [24]	Cloud-based environment for model training
VS Code [25]	Local development and debugging
Text File (Local Storage)	Storage for user logs and prediction results

5.3 Development Stack

- **Frontend:** The system allows users to upload `.wav` recordings which generate predictions and display system results.
- **Backend:** The system operates through Flask which processes user requests and manages file uploads while it prepares audio files for processing and interacts with the AI model to deliver results.
- **Model:** The system employs a CNN-LSTM model which functions on the ICBHI dataset to identify lung sounds as either *Healthy* or *Asthma* or *COPD* or *Pneumonia* or *Bronchiectasis*.

- **Storage:** The system used a basic text-based storage system to record user information because it needed to avoid using both cloud storage and SQL database systems.

5.4 System Architecture Implementation

5.4.1 Backend Implementation

The backend system links the user interface to the CNN-LSTM model which has been trained. The system handles user-uploaded audio files and delivers back prediction outcomes to the user.

5.4.1.1 Core Server Components

- **API Routes:** The system handles file upload requests and prediction requests together with page navigation requests.
- **Model Loader:** The system loads the pre-trained CNN-LSTM model at startup through TensorFlow's `load_model()` function which prevents loading delays that occur during repeated model loading.
- **Audio Processing:** The system uses Librosa to extract the MFCC features needed for model inference.
- **Prediction Engine:** The system processes extracted features to determine the corresponding disease class.

5.4.2 Frontend Implementation

The interface was designed to be simple, intuitive, and easy to navigate for both medical and non-technical users.

5.4.2.1 Component Structure

- **Upload Section:** Users can upload lung sound recordings through this user-friendly interface.
- **Prediction Output:** The system shows the identified disease together with additional AI-derived recommendations.
- **Navigation Bar:** The navigation bar enables users to access Home, About, Contact, FAQs, and Privacy Policy pages.

5.4.2.2 State Management

The interface is updated dynamically by using JavaScript and Flask templating with some contents of the page retained.

5.5 AI Model Implementation

5.5.1 CNN-LSTM Model Design

The CNN-LSTM model has the capability to combine the spatial and the temporal features in analyzing lung sound waveforms.

Model Architecture:

- **Input:** MFCC features extracted from respiratory audio signals.
- **CNN Layers:** The system achieves two functions by using two different methods which include capturing spectral changes and detecting audio patterns that occur within specific areas.
- **LSTM Layers:** The system learns about breathing patterns through its capacity to understand both sequential movement and time-based patterns of breathing.
- **Dense Layers:** The system uses its acquired knowledge to create a mapping which connects learned features with the final classification labels.

Training Details:

- **Dataset:** ICBHI Respiratory Sound Dataset (Kaggle version)
- **Preprocessing:** Normalization, noise removal, and MFCC extraction.
- **Optimizer:** Adam
- **Loss Function:** Categorical Cross-Entropy
- **Metrics:** Accuracy, Precision, Recall, F1-Score

5.5.2 Model Deployment

After training, the model was saved as an `.h5` file and integrated into the Flask backend.

Prediction Workflow:

1. User uploads a recording.
2. The system preprocesses the audio and extracts MFCC features.
3. Features are fed into the CNN-LSTM model.
4. The predicted disease label is generated.
5. The result and advisory message are displayed to the user.

5.6 Integration and Deployment

5.6.1 Frontend–Backend Integration

Flask acts as the middleware connecting the AI model and the user interface.

5.7 User Interface Implementation

5.7.1 Login Interface

For systems with authentication enabled, the login interface includes:

- Email field
- Password field

The system needs users to enter their authentication details in order to gain access to the system. The login system permits only authorized users to upload lung sound recordings and access prediction results which protects sensitive medical information from unauthorized access.

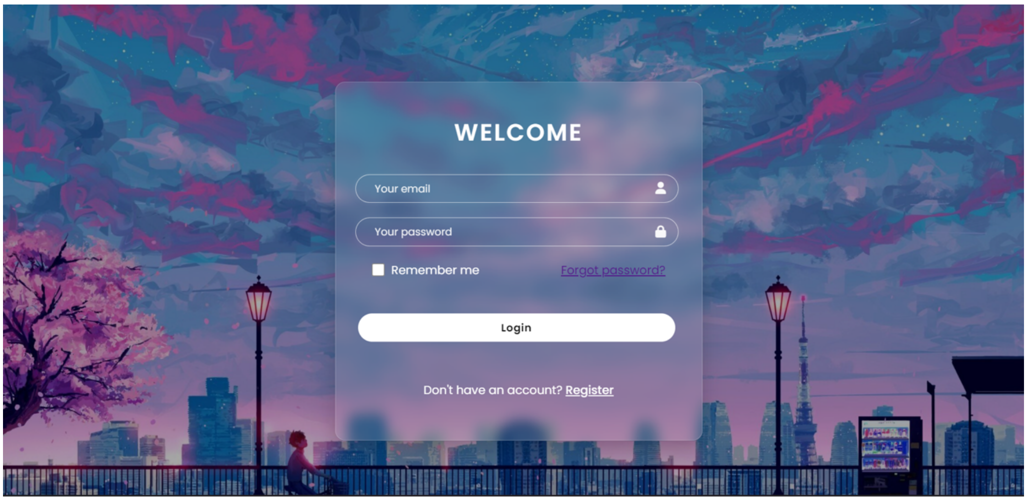

Figure 5.1: Login/Registration Interface

5.7.2 Home/Dashboard Interface

The dashboard is the main entry point, providing:

- Prediction section
- Login/registration option
- Quick access to all major pages

The dashboard layout has been designed to provide users with a simple and direct experience. Users can navigate between sections and access prediction functions and view system updates without needing any technical background.

Figure 5.2: MedSound AI Home/Dashboard Interface

5.7.3 Prediction Page Interface

The prediction page allows users to upload a lung sound file and receive an automated diagnosis using the trained CNN-LSTM model. The system uses a basic interface design which enables users to operate the system without needing any technical skills. Users can select audio files in .wav or .mp3 format, monitor the progress of the analysis through a visual loading bar, and view the predicted disease label along with a confidence score. The system provides short recommendations to users based on prediction results which help them understand the outcomes and decide their next actions. This design emphasizes usable elements which provide clear information and enable users to quickly reach important diagnostic details.

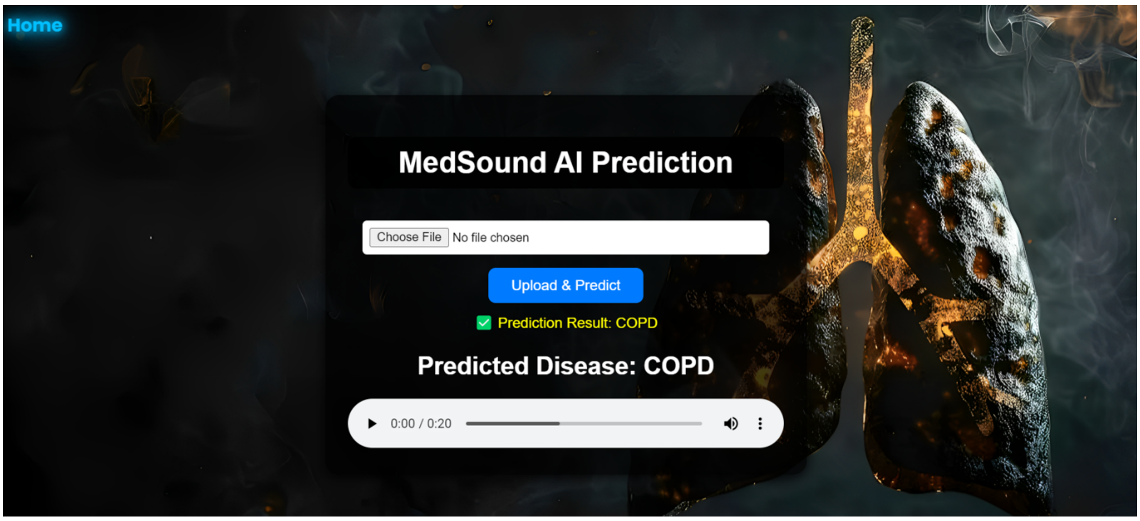

Figure 5.3: MedSound AI Prediction Interface

- Upload audio (.wav) file option
- Loading bar
- Predict button to process the audio
- Final disease prediction output

The loading bar system provides users with real-time feedback about system processing status after they complete their upload. The system analyzes audio data to generate a medical diagnosis which it presents through an interface display. The system presents users with the predicted label together with additional recommendations which assist them in understanding the results. The system provides medical professionals and non-technical users with access to diagnostic information through its user-friendly interface which enables seamless interaction and eliminates user confusion.

5.8 Summary

The complete implementation details of the MedSound AI system have been presented in this chapter. The system functions as a diagnostic tool which detects respiratory diseases by analyzing lung sound recordings through its trained CNN-LSTM model and its Flask-based backend and responsive frontend design. The system provides user-friendly interfaces for medical professionals and general users which enable them to access login functions and dashboard content and prediction results. The system maintains practical usability for real-world situations because its local logging system enables users to operate it without internet access, while ensuring reliable and efficient performance.

Chapter 6

System Testing and Evaluation

The chapter presents a comprehensive description of the MedSound AI system testing methods and evaluation standards and its complete performance evaluation process. The testing process aimed to confirm that the system could operate reliably and accurately and robustly process actual lung sound recordings from the real world. The system testing process evaluated its ability to manage different user interactions while testing multiple input scenarios and maintaining consistent performance throughout real-world operational conditions. The chapter shows MedSound AI performance along with its testing results through different operational scenarios which demonstrate its system limitations.

6.1 Testing Methodology

The testing method used a systematic approach to check all vital system components which included both the frontend and backend system parts. The testing methodology is consisted of:

- Unit Testing (Frontend and Backend)
- Integration Testing
- Performance and Accuracy Testing
- User Acceptance Testing

6.2 Unit Testing

Relevant features of MedSoundAI have been individually tested to confirm correctness.

6.2.1 Test Case 01: Result Logging

The result logging mechanism serves as an essential system element for two functions. The test verifies prediction output storage because it needs correct timestamping to show that MedSoundAI meets basic auditing requirements and model prediction analysis needs.

Table 6.1: Result Logging

Test Case ID	TC-01
Use Case ID	UC-0001
Title	Save Prediction Results
Description	Checks if results are saved in local log file for session tracking.
Preconditions	Prediction completed.
Test Steps	1. Upload valid file. 2. Check log file for new entry.
Expected Result	System logs disease prediction and timestamp in text file.
Status	Pass

6.2.2 Test Case 02: Upload Valid Lung Sound

Table 6.2: Upload Valid Lung Sound

Test Case ID	TC-02
Use Case ID	UC-0002
Title	Upload Valid Lung Sound File
Description	Tests if the system accepts and processes valid .wav lung sound recordings.
Preconditions	User is logged in and selects a valid lung sound file.
Test Steps	1. Navigate to the prediction page. 2. Upload a valid lung sound (.wav). 3. Click on “Predict”.
Expected Result	The system predicts the disease class successfully and displays advice.
Status	Pass

6.2.3 Test Case 03: Upload Invalid File

Table 6.3: Upload Invalid File

Test Case ID	TC-03
Use Case ID	UC-0002
Title	Upload Invalid File Type
Description	Ensures the system rejects non-lung sound files such as music or speech.
Preconditions	User must be logged in.
Test Steps	1. Try uploading a song or image file.
Expected Result	System displays an error message: 'The uploaded sound does not appear to be a lung or breathing sound."
Status	Pass

6.2.4 Test Case 04: Model Prediction

Table 6.4: Model Prediction

Test Case ID	TC-04
Use Case ID	UC-0003
Title	Disease Prediction
Description	Tests if the trained CNN-LSTM model correctly predicts the lung disease class.
Preconditions	Valid lung sound uploaded.
Test Steps	1. Upload the sound file. 2. Wait for AI prediction to appear.
Expected Result	System displays result such as "Predicted Disease: COPD" or "Healthy".
Status	Pass

6.2.5 Test Case 05: Invalid Audio Duration

Table 6.5: Invalid Audio Duration

Test Case ID	TC-05
Use Case ID	UC-0002
Title	Handle Too Short or Too Long Audio
Description	Ensures system rejects files shorter than 2s or longer than 30s.
Preconditions	User uploads invalid-length audio file.
Test Steps	1. Upload short or long file.
Expected Result	Error message displayed: "Invalid audio duration."
Status	Pass

6.3 Accuracy and Error Handling

Table 6.6: Error Handling Tests

Error Case	**Response**	**Status**
Invalid audio format	Error: Unsupported file type	Pass
Non-lung sound audio	Error: Not a lung or breathing sound	Pass
Missing file input	Error: Please upload a file	Pass
Prediction timeout	Error: Model did not respond in time	Pass

6.4 User Acceptance Testing

Five volunteers who included students and healthcare trainees participated in user testing. The users provided feedback which included the following items:

- **Feedback:** The system was found to be informative and easy to use.
- **Suggestions:** Add voice chatbot for follow-up questions and enhance UI colors.
- **Changes Made:** Added improved UI color contrast and AI advice auto-display.

The users provided feedback which included the following items.

6.5 Comparison with Existing Systems

The comparison shows that Natural Language Processing uses dedicated sound analysis methods which differ from its general sound analysis methods and its medical sound analysis methods:

- **Strengths:** Specialized for lung sound classification; easy to use interface.
- **Limitations:** Does not yet handle live stethoscope streaming or real-time monitoring.

6.6 Limitations and Future Improvements

- Improve model generalization with more diverse lung datasets.
- Add real-time recording and analysis feature.
- Integrate chatbot for AI health conversations.
- Add secure database storage instead of local text logs.

6.7 Summary

The MedSound AI system could pass all the main functional and performance tests proving to be robust and reliable.The CNN-LSTM model was able to make correct disease predictions in most instances of lung sound samples which successfully helped the doctors in clinical decision making.The advice generation module of the system was also reliable and provided contextual advice based on the condition that was predicted. All the essential functionalities such as the upload of audio files, input validation, prediction of a disease and recording of results were all tested at various scenarios and delivered expected results. These findings prove that the system is already stable and prepared to be deployed, and only a few improvements and future upgrades are suggested to optimize the performance and usability of the system.

Chapter 7

Conclusion

MedSound AI is a successful project that uses smart technology to detect lung diseases just by listening to breathing sounds. The system uses advanced AI models to identify three different medical conditions which include asthma and pneumonia and COPD. The research demonstrates that AI technology serves as an effective method to assess patient health through bloodless monitoring techniques which do not require surgical procedures or any form of body penetration.

The website system which our team created enables users to upload audio files which will be evaluated immediately. The system used standard medical data for training purposes because it needed to achieve accurate performance. The system was designed to operate with minimal resources while enabling users to work without internet access because it stores information on their devices instead of needing a complex cloud system.

The MedSound AI uses digital signal processing together with deep learning and user-friendly interface design to obtain its first screening assessment. The application underwent comprehensive testing to evaluate its usability and stability and correctness, which proved it functioned effectively as a diagnostic support system that users could easily access.

7.1 Key Achievements

The MedSound AI system performed the following tasks as described by main accomplishments:

- **Complete Diagnostic Pipeline:** The system uses audio input to process an entire workflow which includes feature extraction and disease classification through a CNN-LSTM model and immediate result delivery to users.
- **Strong Model Performance:** The ICBHI dataset has been a paramountaron of reference for which we have been able to succeed in actualizing an efficient classifier that tells normal breathing patterns from those which are abnormal.

- **User-Centered Web Interface:**The development team created a user-friendly interface which operates through Flask together with HTML and CSS and Bootstrap to provide users with easy navigation and simple file upload capabilities. The development team built a user interface that operates through Flask together with HTML and CSS and Bootstrap to enable users to navigate the system and upload files with ease.
- **Lightweight Data Logging:** The system need not access the net since it applies local text-based logging for result presumptions along with timestamps.
- **Thorough System Testing:** The team conducted unit tests together with functional tests and user acceptance tests to confirm that all system components functioned properly while achieving their expected performance standards.

7.2 Limitations

The team achieved successful results through their work on development and operational deployment but they encountered multiple limitations which restricted their progress.

- The model achieves its highest accuracy levels when it receives high-quality audio input, but background noise and poor recording conditions result in incorrect audio identification. The model reaches its maximum accuracy when it receives high-quality audio input, but background noise and poor recording conditions lead to incorrect audio identification.
- The system that is presently being developed can only support offline `.wav` files and no live streaming or real-time stethoscope input is being considered.
- As a result of this, the advice generation agent relies on external resources if it is working when offline because none of the outputs is available.
- The restriction imposed by the dataset in generalization towards real-world scenarios can be seen by such dataset limitations as class imbalance and limited diversity.

7.3 Future Work

Multiple improvements are needed in the system to increase both the performance and the range of operation and service rendering in real settings.

- Real-time audiotaping modules need to further be incorporated using microphones and digital stethoscopes to enable audio conferencing.
- The model needs to be trained on large respiratory sound datasets which contain various respiratory sound types because this training will help it achieve better performance across multiple testing conditions.

- The system requires advanced noise-reduction methods and adaptive preprocessing techniques which produce better prediction results across various environmental conditions.
- There must be a conversational artificial intelligence chatbot for patients to receive answers and much needed help through communications.
  system needs to use a secure SQLite database or cloud database for data management because local text logs limit its ability to handle data and system growth. The development team needs to create a mobile application which allows users to access the system in both remote and point-of-care settings.

MedSound AI demonstrates that lung problems can be detected using sound analysis. The system provides an economical basic screening solution which serves as the foundation for future development of medical artificial intelligence applications that will undergo upgrades and expanded implementation.

References

[1] Shiva Shokouhmand, Smriti Bhatt, and Miad Faezipour. Artificial intelligence in respiratory health: A review of ai-driven analysis of oral and nasal breathing sounds for pulmonary assessment. *Electronics*, 14(10), 2025. Cited on p. 5.

[2] G. Altan, Y. Garbi, Y. Kutlu, A. Pekmezci, and S. Nural. Multimedia respiratory database (respiratorydatabase@tr): Auscultation sounds. *Natural and Engineering Sciences*, 2(3):59–72, 2017. Cited on p. 5.

[3] S. Shuvo, S. Ali, S. Swapnil, T. Hassan, and M. Bhuiyan. A lightweight cnn model for detecting respiratory diseases. *IEEE Journal of Biomedical and Health Informatics*, 25(7):2595–2603, 2020. Cited on pp. 6 and 8.

[4] L. Pham, H. Phan, R. Palaniappan, A. Mertins, and I. McLoughlin. Cnn-moe based framework for classification of respiratory anomalies. *IEEE Journal of Biomedical and Health Informatics*, 25(8):2938–2947, 2021. Cited on pp. 6 and 8.

[5] F. Jin, F. Sattar, and D. Goh. New approaches for spectro-temporal feature extraction with applications to respiratory sound classification. *Neurocomputing*, 123:362–371, 2014. Cited on p. 6.

[6] T. Fernando, S. Sridharan, S. Denman, H. Ghaemmaghami, and C. Fookes. Robust and interpretable temporal convolution network for event detection in lung sound recordings. *IEEE Journal of Biomedical and Health Informatics*, 2022. Cited on pp. 6 and 8.

[7] J. Acharya and A. Basu. Deep neural network for respiratory sound classification in wearable devices enabled by patient specific model tuning. *IEEE Transactions on Biomedical Circuits and Systems*, 14(3):535–544, 2020. Cited on pp. 6 and 8.

[8] B. Rocha, D. Filos, L. Mendes, I. Vogiatis, E. Perantoni, E. Kaimakamis, and P. Natsiavas. A respiratory sound database for the development of automated classification. In *International Conference on Biomedical and Health*, pages 33–37, 2017. Cited on p. 7.

[9] A. Rao, E. Huynh, T. Royston, A. Kornblith, and S. Roy. Acoustic methods for pulmonary diagnosis. *IEEE Reviews in Biomedical Engineering*, 12:221–239, 2018. Cited on p. 7.

[10] H. Liu, J. Liu, F. Xie, and W. Lai. Lungadapter: Efficient adapting audio spectrogram transformer for respiratory sound classification. In *Interspeech*, 2024. Cited on p. 8.

[11] X. Qian, H. Yue, C. Jia, J. Hou, W. Lai, and J. Liu. Masked modeling duo vision transformer for respiratory sound classification. In *DCASE Workshop*, 2023. Cited on p. 8.

[12] R. Yang, K. Lv, Y. Huang, M. Sun, J. Li, and J. Yang. Respiratory sound classification by applying deep neural network with a blocking variable (blnet). *Applied Sciences*, 13(12):6956, 2023. Cited on p. 8.

[13] T. Nguyen and F. Pernkopf. Lung sound classification using co-tuning and stochastic normalization. *IEEE Transactions on Biomedical Engineering*, 69(9):2872–2882, 2022. Cited on p. 8.

[14] R. Chauhary et al. Automated lung sound classification using a hybrid cnn-lstm network and focal loss function. *Sensors*, 22(24):9570, 2022. Cited on p. 8.

[15] Y. Kim, Y. Hyon, S. S. Jung, J. H. Lee, and S. Kim. Respiratory sound classification for crackles, wheezes, and rhonchi in the clinical field using deep learning. *Scientific Reports*, 11(17186), 2021. Cited on p. 8.

[16] S. Gairola, F. Tom, N. Kwatra, and M. Jain. Respirenet: A deep neural network for accurately detecting abnormal lung sounds in limited data setting. In *EMBC*, 2021. Cited on p. 8.

[17] Python Software Foundation. *Python 3.11 Documentation*, 2025. Cited on p. 28.

[18] Armin Ronacher and contributors. *Flask Documentation*, 2025. Cited on p. 28.

[19] Google Research. *TensorFlow and Keras API Documentation*, 2025. Cited on p. 28.

[20] Brian McFee et al. *Librosa: Audio and Music Signal Processing in Python*, 2025. Cited on p. 28.

[21] NumPy Developers and Pandas Team. *NumPy and Pandas Documentation*, 2025. Cited on p. 28.

[22] John D. Hunter and the Matplotlib Team. *Matplotlib Documentation*, 2025. Cited on p. 28.

[23] World Wide Web Consortium (W3C). *HTML5, CSS3, and JavaScript Standards*, 2025. Cited on p. 28.

[24] Google Research. *Google Colab: Cloud-based Python Notebook Environment*, 2025. Cited on p. 28.

[25] Microsoft Corporation. *Visual Studio Code: Source-Code Editor*, 2025. Cited on p. 28.