A logo of a university

AI-generated content may be incorrect.

AI-Powered Lung Disease Classification System Software Requirements Specification

Team Member :

Abdullah Abu Fodeh

Mohammed Aljammal

Supervisor :

Khalid Alemerien

02/06/2025

# Revision History

| Date | Description | Comments |
| --- | --- | --- |
| 25/2/2025 | Version 1.0 | Initial project setup and dataset collection. |
| 10/4/2025 | Version 1.1 | Implemented custom CNN model and initial training. |
| 20/4/2025 | Version 1.2 | Integrated Flask application with AI model. |
| 25/4/2025 | Version 2.0 | Completed testing and added Grad-CAM functionality. |
| 29/5/2025 | Version 3.0 | Finalized documentation and completed objectives successfully. |

# Document Approval

| Signature | Printed Name | Title | Date |
| --- | --- | --- | --- |
|  | Mohammad Al Jammal | Developer |  |
|  | Abdullah Abu Fodeh | Developer |  |
|  | Khakied Alemerein | Supervisor |  |

**TABLE OF CONTENTS**

Revision History ............................................................................................................................................... 2

Document Approval ....................................................................................................................................... 2

CHAPTER 1: INTRODUCTION ............................................................................................................................

1.1 Purpose ..................................................................................................................................................... 4

1.2 Problem Statement ................................................................................................................................... 5

1.3 Proposed Solution .................................................................................................................................... 5

1.4 Scope ....................................................................................................................................................... 5

1.5 Definitions, Acronyms, and Abbreviations .............................................................................................. 6

1.6 Objectives ................................................................................................................................................ 6

CHAPTER 2: REQUIREMENTS .........................................................................................................................

2.1 Introduction ............................................................................................................................................. 6

2.2 Product Functions .................................................................................................................................... 6

2.3 User Characteristics ................................................................................................................................. 7

2.4 General Constraints .................................................................................................................................. 7

2.5 External Interface Requirements ........................................................................................................... 7

2.5.1 User Interfaces ................................................................................................................................. 7

2.5.2 Hardware Interfaces ......................................................................................................................... 7

2.5.3 Software Interfaces .......................................................................................................................... 8

2.5.4 Communications Interfaces .............................................................................................................. 8

2.6 Functional Requirements ......................................................................................................................... 8

2.6.1 User Authentication ........................................................................................................................ 8

2.6.2 X-ray Upload and Diagnosis ............................................................................................................ 9

2.6.3 Dashboard ....................................................................................................................................... 9

2.6.4 Administrative Functions ................................................................................................................. 9

2.7 Use Cases ................................................................................................................................................. 10

2.8 Classes/Objects ...................................................................................................................................... 10

2.8.1 User .............................................................................................................................................. 10

2.8.2 Diagnosis ....................................................................................................................................... 10

2.8.3 X-ray Image .................................................................................................................................. 10

2.9 Non-Functional Requirements ............................................................................................................... 10

2.9.1 Performance ................................................................................................................................. 11

2.9.2 Reliability ..................................................................................................................................... 11

2.9.3 Availability ................................................................................................................................... 11

2.9.4 Security ......................................................................................................................................... 11

2.9.5 Maintainability .............................................................................................................................. 11

2.9.6 Portability ..................................................................................................................................... 11

2.10 Inverse Requirements ........................................................................................................................... 11

2.11 Other Requirements ............................................................................................................................. 11

CHAPTER 3: ANALYSIS MODELS ...................................................................................................................

3.1 System Architecture ............................................................................................................................... 12

3.2 Use Case Diagram ................................................................................................................................. 13

3.3 Activity Diagram .................................................................................................................................... 14

3.4 Sequence Diagrams ............................................................................................................................... 20

3.5 ER Diagram ........................................................................................................................................... 21

3.6 Class Diagram ....................................................................................................................................... 22

CHAPTER 4: DESIGN AND IMPLEMENTATION ............................................................................................

4.1 Design Overview .................................................................................................................................... 23

4.2 Proposed Model Details ......................................................................................................................... 23

4.3 Snapshots of Main Features ................................................................................................................... 24

4.3.1 Sign In View ................................................................................................................................. 24

4.3.2 Sign Up View ............................................................................................................................... 25

4.3.3 Dashboard View ............................................................................................................................ 25

4.3.4 X-ray Upload View ....................................................................................................................... 26

4.4 Code Samples ........................................................................................................................................ 2

4.4.1 User Signup .................................................................................................................................. 26

4.4.2 X-ray Image Processing ................................................................................................................ 27

4.5 Sample Images of Diseases .................................................................................................................... 27

CHAPTER 5: TESTING ......................................................................................................................................

5.1 Test Cases for Signup ............................................................................................................................ 28

5.2 Testing Units .......................................................................................................................................... 28

CHAPTER 6: CONCLUSION .............................................................................................................................

6.1 Summary ............................................................................................................................................... 28

6.2 Conclusion ............................................................................................................................................. 28

6.3 Challenges .............................................................................................................................................. 28

6.4 Future Work ........................................................................................................................................... 29

REFERENCES ................................................................................................................................................... 29

# 1 Introduction

## Purpose

The purpose of this project is to develop an AI-powered system to assist healthcare professionals in diagnosing lung diseases from chest X-ray images. The system employs a custom Convolutional Neural Network (CNN) to classify images into four categories: COVID-19, Normal, Pneumonia, and Tuberculosis. Integrated with a Flask-based web application, it provides a secure, user-friendly interface for patients, doctors, and administrators, enabling rapid and accurate preliminary diagnoses to enhance patient outcomes and reduce healthcare professionals’ workload.

## 1.2 Problem Statement

Accurate and timely diagnosis of lung diseases via chest X-ray images is critical yet challenging due to human error and time constraints in manual interpretation, particularly in resource-limited settings. Development challenges included curating a high-quality dataset and training a reliable CNN model within three months (February 25 to June 24, 2025), necessitating an automated, efficient solution to support healthcare delivery.

## 1.3 Proposal Solution

The proposed solution is an AI-driven system utilizing a custom CNN model for classifying chest X-ray images, integrated with a Flask web application. The system incorporates Gradient-weighted Class Activation Mapping (Grad-CAM) for visual explanations, enhancing trust in AI diagnostics. It supports a future mobile app to broaden accessibility. Due to time constraints, full mobile deployment is deferred to future phases.

*Response to Comment: “Future mobile app requires devices with cameras for X-ray capture”* The current system does not support direct X-ray capture via device cameras, as it relies on web-based image uploads. The future mobile app, planned for post-May 2025, will require devices with cameras to enable X-ray capture, but this feature is not implemented in the current iteration.

## 1.4 Scope

The project scope encompasses:

* Collecting and curating a dataset of chest X-ray images from Kaggle, GitHub, and Mendeley Data.
* Developing and training a custom CNN model for classification.
* Implementing a Flask web application with role-based access for patients, doctors, and administrators.
* Integrating Grad-CAM for interpretability.
* Planning a future mobile application.

The project was completed within a three-month period from February 25 to June 24, 2025.

## 1.5 Definitions, Acronyms, and Abbreviations

| Term | Definition |
| --- | --- |
| CNN | Convolutional Neural Network |
| Grad-CAM | Gradient-Weighted Class Activation Mapping |
| SRS | Software Requirements Specification |
| API | Application Programming Interface |
| UI | User Interface |
| UX | User Experience |
| OTP | One-Time Password |
| HTTP | HyperText Transfer Protocol |
| SQLite | A lightweight relational database management system |

### 

## 1.6 Objectives

* Provide rapid, accurate preliminary diagnoses of lung diseases.
* Enhance accessibility via web and planned mobile platforms.
* Build trust in AI diagnostics with visual explanations using Grad-CAM.
* Support healthcare professionals in improving patient outcomes.

# 2 Requirements

## 2.1 Introduction

This section outlines the functional and non-functional requirements for the AI-Powered Lung Disease Classification System, covering the AI model and Flask web application, with provisions for a future mobile extension.

## 2.2 Product Functions

The system supports:

* User signup with OTP verification.
* User signin and session management. • Password reset with OTP verification.
* Profile management with optional image upload.
* X-ray image upload and CNN-based diagnosis processing.
* Diagnosis result viewing with Grad-CAM heatmaps.
* Role-specific dashboards (patient, doctor, admin).
* Administrative functions for user and diagnosis management.
* Contact form for user inquiries.

## 2.3 User Characteristics

* **Patients**: Upload X-rays, view diagnoses, and manage profiles. Require an intuitive, simple interface.
* **Doctors**: Review, edit, and delete diagnoses they create, needing detailed data and management tools.
* **Administrators**: Manage all users and diagnoses, requiring comprehensive access and control.

## 2.4 General Constraints

* High-quality X-ray images (grayscale) for accurate classification.
* Internet connectivity for web app access.
* System runs on standard web servers with Python 3.8+, Flask, and SQLite support.

## 2.5 External Interface Requirements

**2.5.1 User Interfaces** Web pages for:

* Signup, signin, OTP verification, password reset.
* Profile viewing and editing.
* X-ray upload and diagnosis result display.
* Role-specific dashboards (patient, doctor, admin).
* Contact form and admin user management.

**2.5.2 Hardware Interfaces**

* Devices with internet access and browsers (e.g., desktops, laptops).

### 2.5.3 Software Interfaces

* Keras: For CNN model development and inference.
* Flask: Web application framework.
* SQLite: Database management for user and diagnosis data.
* TensorFlow: Backend for model execution.

### 2.5.4 Communications Interfaces

* HTTP for transmission between client and server.
* SMTP (via Gmail) for OTP and contact email delivery.

## 2.6 Functional Requirements

**2.6.1 User Authentication**

### Signup

* **Introduction**: Enables new users to register with OTP verification.
* **Inputs**: First name, last name, username, email, password, phone, birth date, role (patient/doctor), doctor ID (for doctors), profile image (optional).
* **Processing**: Validates inputs, generates OTP, sends verification email, stores hashed password in SQLite.
* **Outputs**: Account created, OTP sent.
* **Error Handling**: Invalid inputs, duplicate username/email, email delivery failure.

### Sign In

* **Introduction**: Authenticates users and initiates sessions.
* **Inputs**: Username, password.
* **Processing**: Verifies credentials against hashed password, creates role-based session.
* **Outputs**: Redirects to dashboard or displays error.
* **Error Handling**: Invalid credentials, unverified account.

### Forgot Password

* **Introduction**: Facilitates password reset with OTP.
* **Inputs**: Email, OTP, new password, confirmation.
* **Processing**: Sends OTP to email, verifies OTP, updates password hash.
* **Outputs**: Password reset confirmation or error.
* **Error Handling**: Invalid email, expired OTP, mismatched passwords.

#### 2.6.2 X-ray Upload and Diagnosis

* **Introduction**: Allows X-ray image uploads for AI diagnosis.
* **Inputs**: X-ray image file (PNG/JPEG/JPG).
* **Processing**: Resizes to 150 × 150 grayscale, normalizes, analyzes with CNN, generates Grad-CAM heatmap, saves to SQLite.
* **Outputs**: Diagnosis (COVID-19, Normal, Pneumonia, Tuberculosis), confidence score, heatmap.
* **Error Handling**: Invalid image format, processing errors.

#### 2.6.3 Dashboard

* **Introduction**: Displays role-specific data and actions.
* **Inputs**: User login session.
* **Processing**: Fetches data from Diagnoses table based on role.
* **Outputs**: Diagnosis list, statistics, management options.
* **Error Handling**: Data retrieval failures.

#### 2.6.4 Administrative Functions

* **Introduction**: Enables admins to manage users and diagnoses.
* **Inputs**: Admin commands (view, delete user/diagnosis).
* **Processing**: Queries/updates Users and Diagnoses tables.
* **Outputs**: Updated system state.
* **Error Handling**: Unauthorized actions, database errors.

## 2.7 Use Cases

* **Patient**: Signs up, signs in, Signs Out, uploads X-ray, views diagnosis, Forgot Password, Edit Profile, View Profile.
* **Doctor**: Signs in, Sign Out, Forgot Password, Sign Up, Edit Profile, View Profile, Upload X-ray, views/creates diagnoses, edits/deletes diagnoses.
* **Admin**: Signs in, Sign Out, Forgot Password, Mange Users, Delete Users, View All Diagnoses, Delete Any Diagnosis Edit Profile, View Profile, Upload X-ray.

## 2.8 Classes / Objects

### 2.8.1 User

* **Attributes**: username, email, password hash, role, fname, lname, phone, birth date, profile image, is verified.
* **Functions**: signup, signin, update profile, verify otp.

### 2.8.2 Diagnosis

* **Attributes**: id, patient id, diagnosis, confidence, description, treatment, medicines, heatmap, xray image, created by.
* **Functions**: create diagnosis, view diagnosis, update diagnosis, delete diagnosis.

### 2.8.3 X-ray Image

* **Attributes**: image path, upload date.
* **Functions**: upload image, process image.

## 2.9 Non-Functional Requirements

### 2.9.1 Performance

* CNN model accuracy: ≥ 93.54%.
* Web app response time: *<* 2 seconds for most operations.
* Image processing time: *<* 5 seconds per X-ray.

### 2.9.2 Reliability

* Uptime: 99*.*5%.
* Comprehensive error handling and logging.

**2.9.3 Availability**

* Accessible 24/7 with minimal planned downtime.

### 2.9.4 Security

* Passwords hashed using Werkzeug’s generate password hash.
* OTP verification for signup and password reset.
* Role-based access control via Flask sessions.

### 2.9.5 Maintainability

* Modular code structure in routes.py.
* Detailed logging for debugging.

### 2.9.6 Portability

* Web app compatible with major browsers (Chrome, Firefox, Edge, Brave).
* Future mobile app planned for iOS and Android.

## 2.10 Inverse Requirements

* Prevent unauthorized access to patient data and system functions.

## 2.11 Other Requirements

* User training materials for healthcare professionals.
* Future support for multiple languages.

# 3 Analysis Models

## 3.1 System Architecture

The system architecture integrates a Flask-based web interface, a custom CNN model for classification, a SQLite database for data storage, and an SMTP email service for notifications. The web interface handles user interactions, the CNN model processes Xray images, the database stores user and diagnosis data, and the email service delivers OTPs and notifications. Figure 1 illustrates this architecture, ensuring secure and efficient processing.

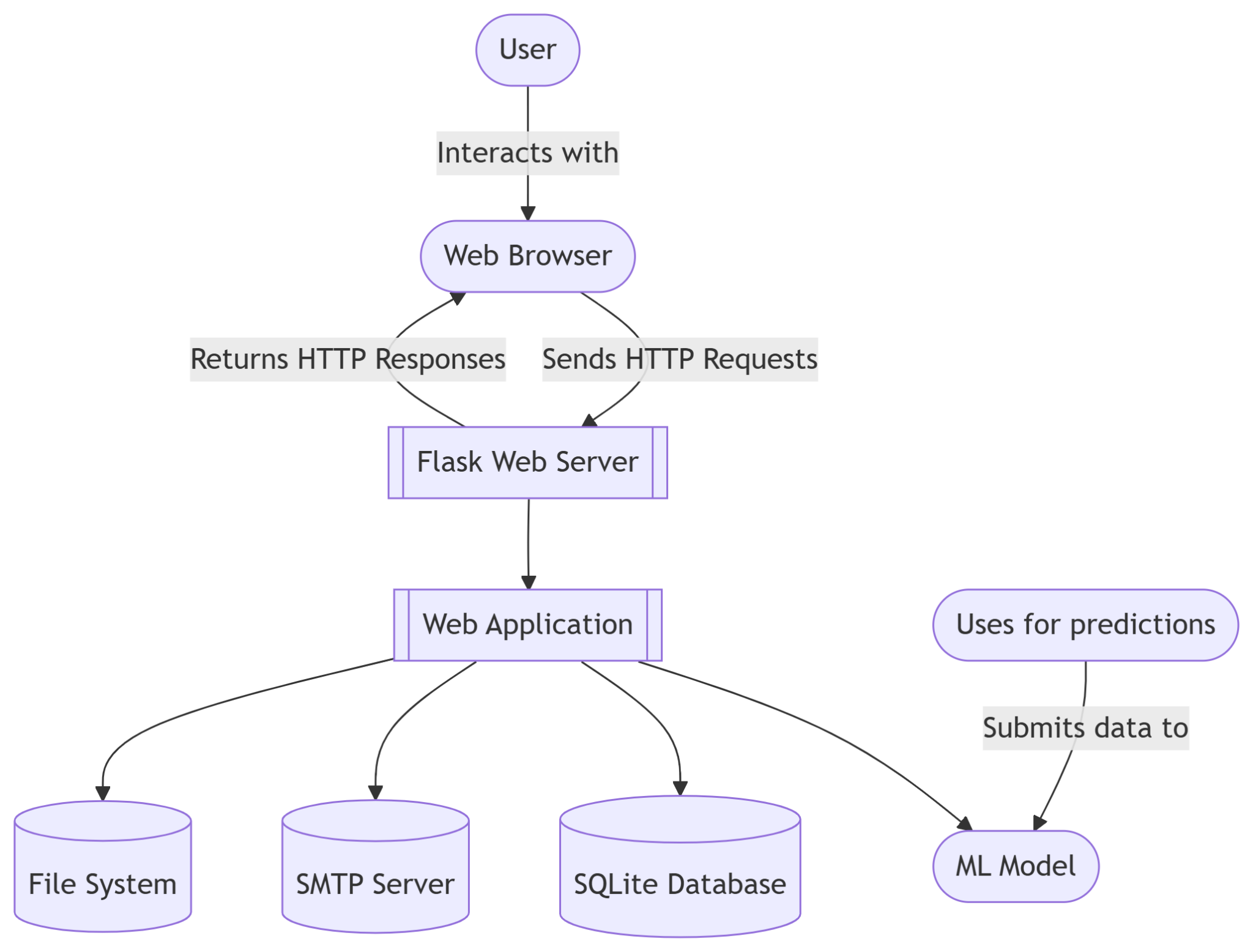


Figure 1: System Architecture Diagram: Interaction between the web interface, AI model, database, and email service for secure and efficient X-ray processing.

## 3.2 Use Case Diagram

The use case diagram outlines the primary interactions of patients, doctors, and administrators with the system, including signup, X-ray upload, diagnosis management, and administrative functions. Figure 2 provides a high-level overview of these interactions, as detailed in Section 2.7.

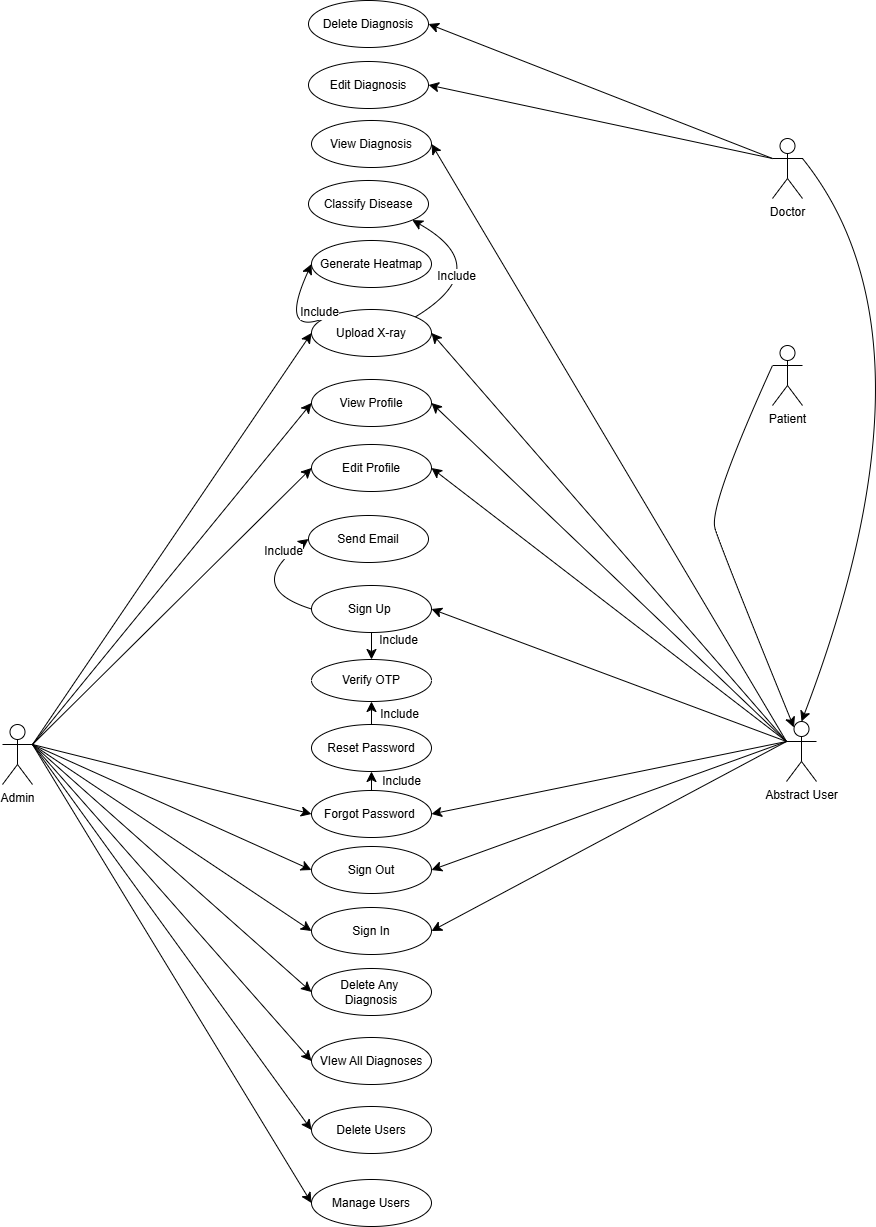


Figure 2: Use Case Diagram: Primary interactions for patients, doctors, and administrators.

## 3.3 Activity Diagram

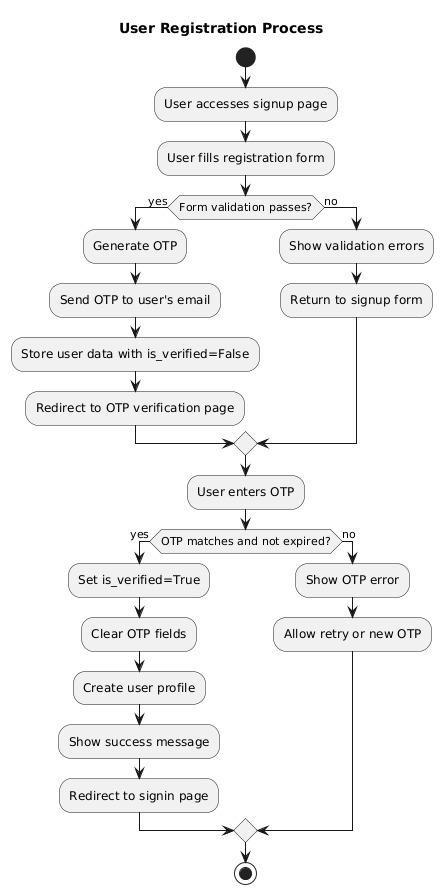


Figure 3: User Registration Process Flowchart (Section 3.8)

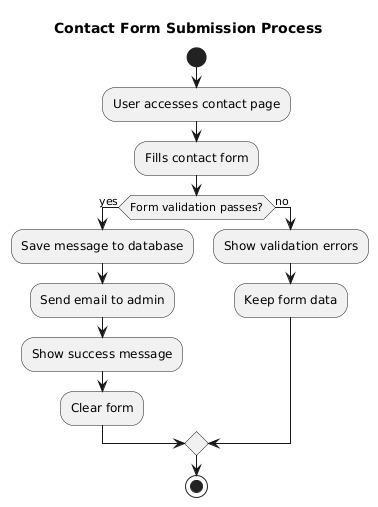


Figure 4: Contact activity flowchart

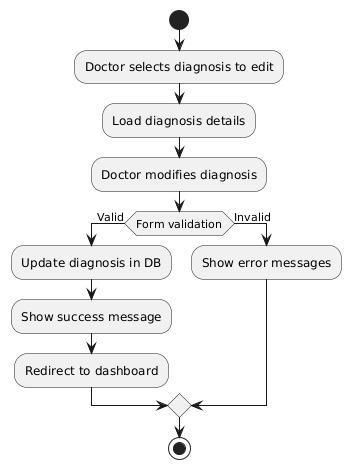


Figure 5: Doctor editable diagnosis flowchart

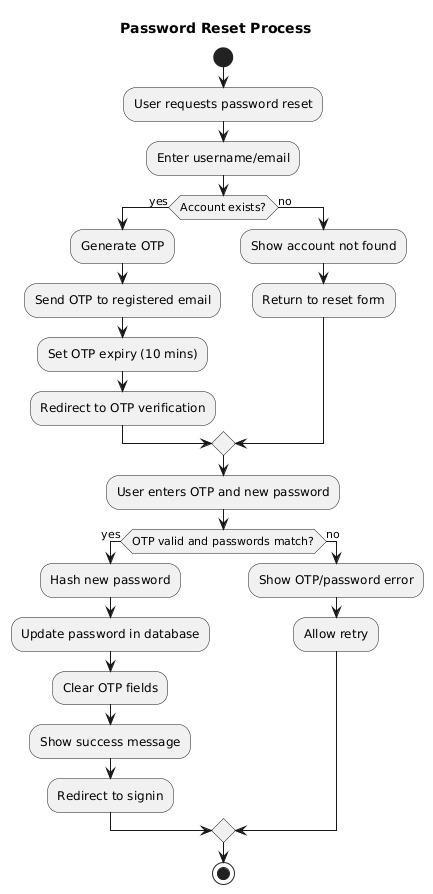


Figure 6: Password Reset flowchart

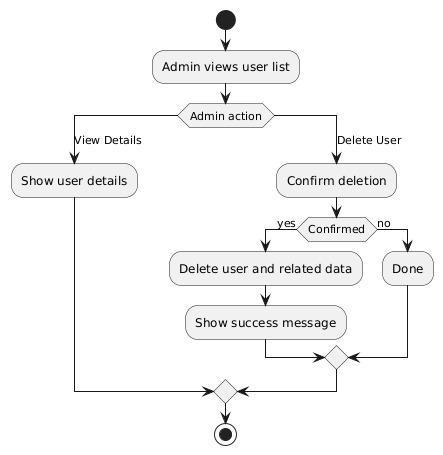


Figure 8: Admin managing ability flowchart

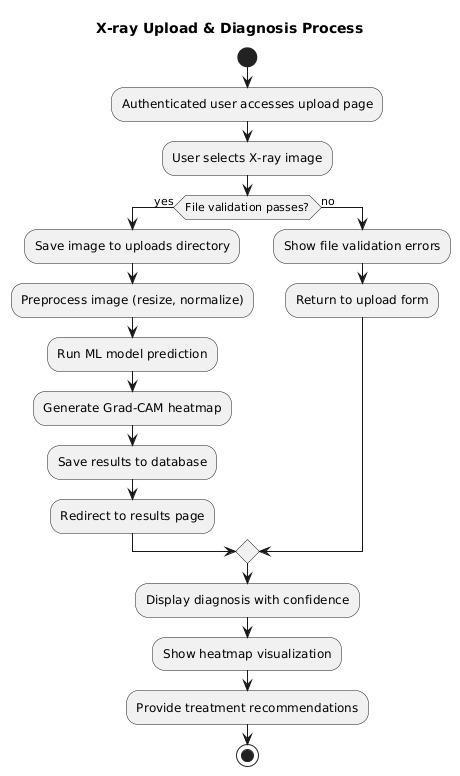
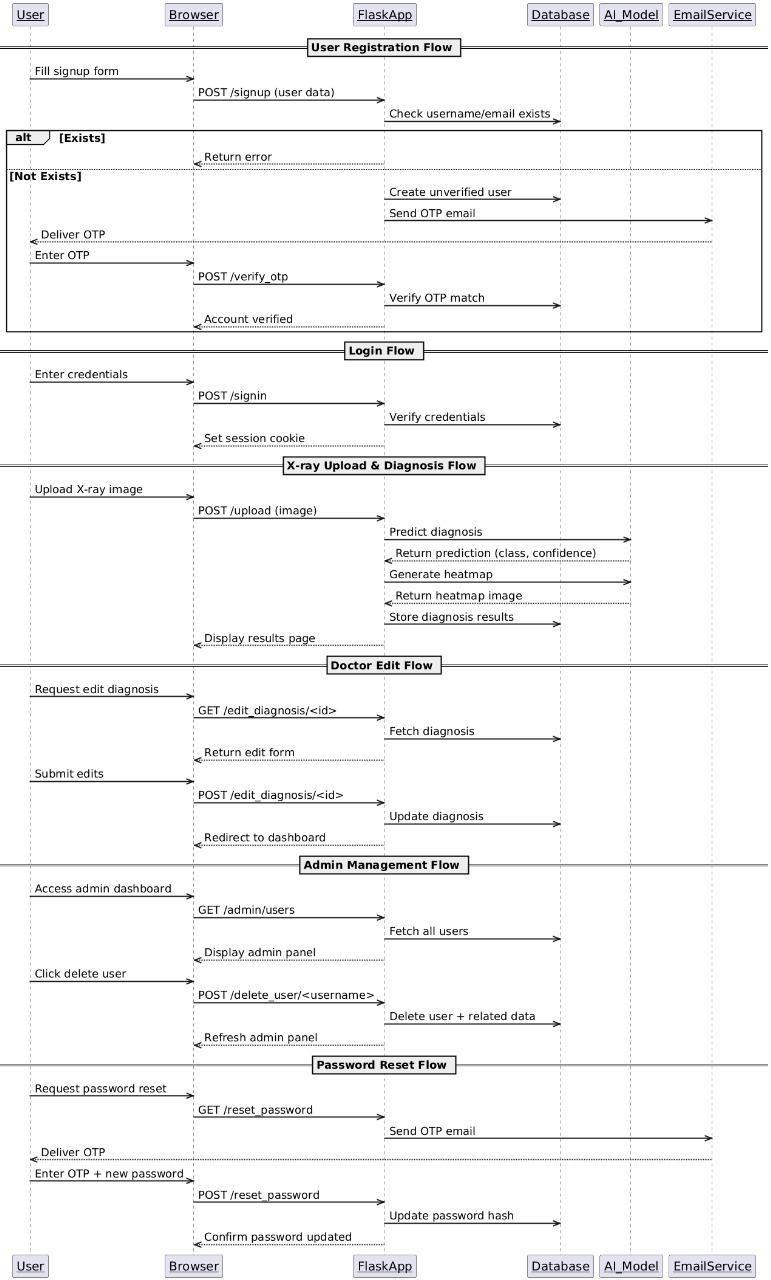


Figure 9: X-ray upload flowchart

The diagnosis process involves several steps: uploading an X-ray image, preprocessing (resizing to 150×150 grayscale and normalization), CNN classification, Grad-CAM heatmap generation, and storing/displaying results. This workflow ensures accurate and interpretable outcomes, as shown in Figures 9.

## 3.4 Sequence Diagrams



The sequence diagram for X-ray upload and diagnosis details the interactions from image upload to diagnosis generation, including preprocessing, AI model inference, and result storage. Figure 7 depicts this process, as described in Section 2.6.2.

## 

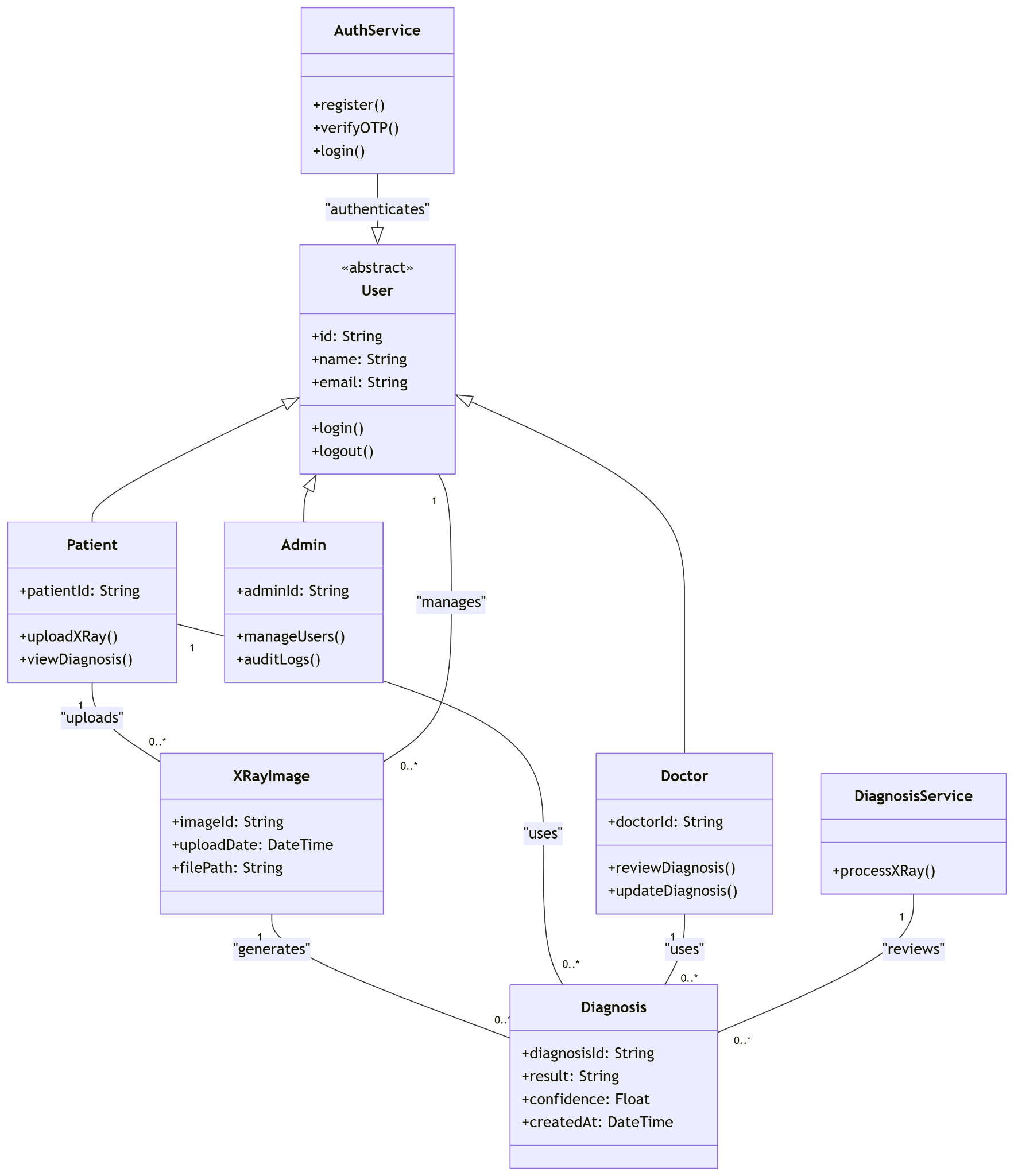
## 3.5 ER Diagram

A diagram of a diagram

AI-generated content may be incorrect.

Figure 5: ER Diagram: Database schema showing relationships between Users, Diagnoses, and X-ray Images.

## 3.6 Class Diagram

****

**Figure 9: This diagram depicts the relationships and methods between key system entities, including User, Patient, Doctor, Admin, XRayImage, Diagnosis, DiagnosisService, and AuthService, as outlined in Section 2.8.**

# 4 Design and Implementation

## 4.1 Design Overview

The system uses Flask as the backend framework, defined in routes.py, with SQLite for data storage. The CNN model, loaded from best model.keras, is integrated via Keras/TensorFlow. The frontend employs HTML, CSS, and JavaScript for a responsive UI.

## 4.2 Proposed Model Details

The proposed AI model is a custom Convolutional Neural Network (CNN) designed to classify chest X-ray images into four categories: COVID-19, Normal, Pneumonia, and Tuberculosis. Key specifications include:

* **Input**: Grayscale X-ray images resized to 150 × 150 pixels.
* **Architecture**:
  + Convolutional Layers: Multiple layers with ReLU activation for feature extraction (e.g., 32 filters of 3 × 3 in the first layer).
  + Max-Pooling Layers: Reduce spatial dimensions (e.g., 2 × 2 pooling).
  + Dropout Layers: Rate of 0.5 to prevent overfitting.
  + Fully Connected Layers: Dense layers with softmax activation for outputting class probabilities.
* **Training**:
  + Dataset: Over 30,000 X-ray images from sources listed in Section 7.
  + Techniques: Data augmentation (rescale, horizontal\_flip) to enhance generalization.
  + Optimizer: Adam with a learning rate of 0.001.
  + Epochs: 50, with early stopping based on validation loss.
* **Accuracy**: Achieved 93.54% on the test set.
* **Interpretability**: Grad-CAM generates heatmaps to highlight influential image regions, stored alongside diagnoses.
* **Integration**: Loaded into Flask via load model from Keras, processing uploads in real-time.

This model balances accuracy and efficiency, making it suitable for deployment within the web application.

## 4.3 Snapshots of Main Features

### 4.3.1 Sign In View

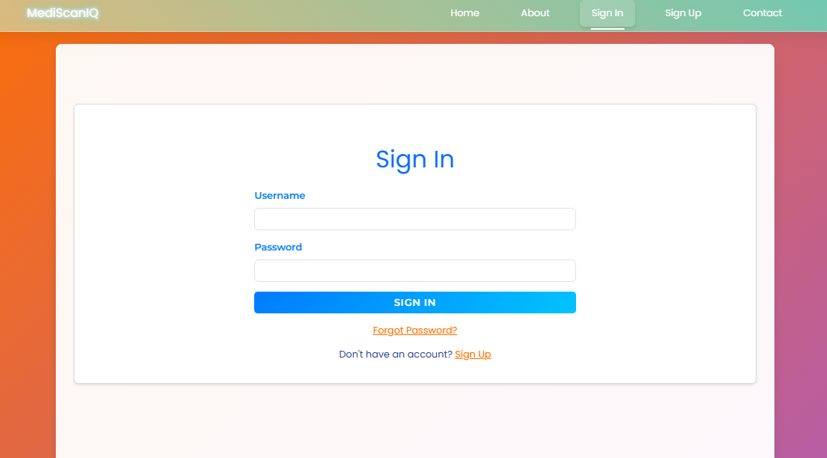


Figure 7: Sign In Page: Secure interface with username and password fields.

### 4.3.2 Sign Up View

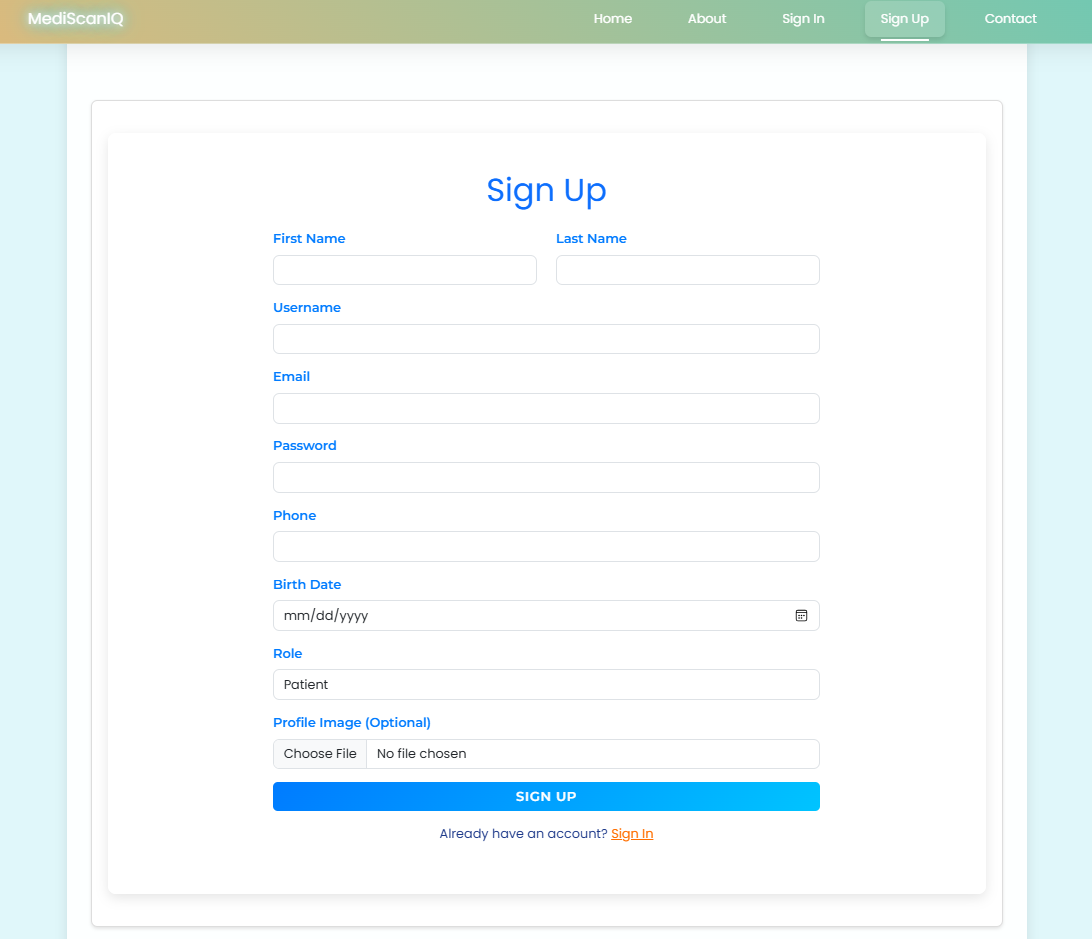


Figure 8: Sign Up Page: Registration interface with OTP verification fields.

### 4.3.3 Dashboard View

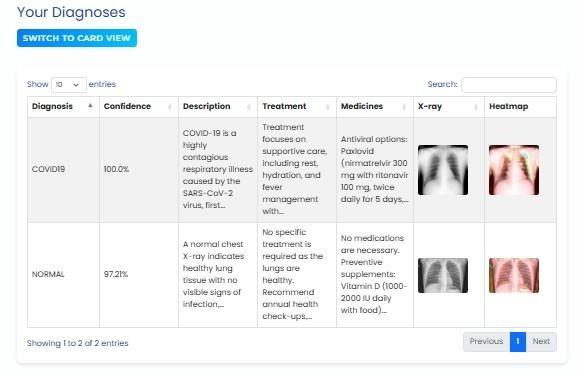


Figure 9: Patient Dashboard: Displays personal diagnoses.

### 4.3.4 X-ray Upload View

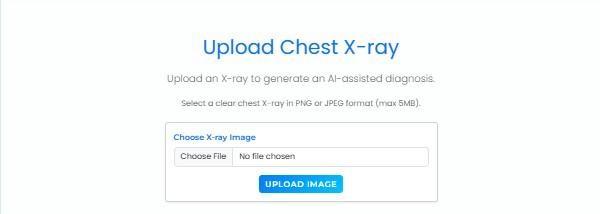
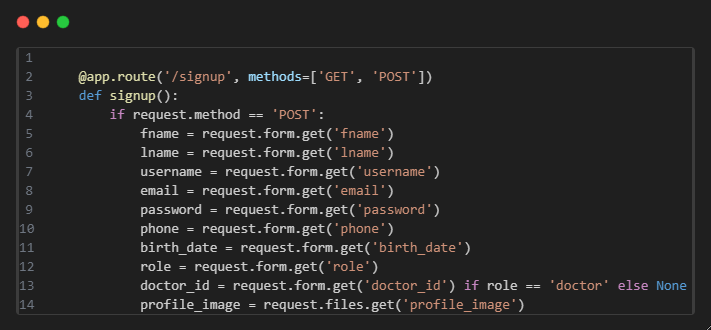


Figure 10: X-ray Upload Page: Interface with diagnosis and heatmap feedback.

## 4.4 Code Samples

**4.4.1 User Signup **

**4.4.2 X-ray Image Processing**

## 4.5 Sample Images of Diseases

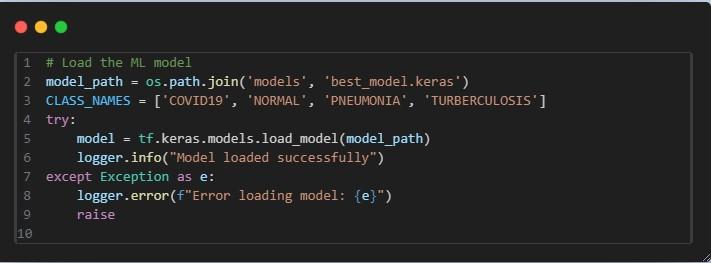
Sample X-ray images for each disease category (COVID-19, Normal, Pneumonia, Tuberculosis) are provided to illustrate typical inputs processed by the model. These images, sourced from the datasets listed in Section 7, are shown in Figure 13.



Figure 11: Sample Disease Images: X-ray images for COVID-19, Normal, Pneumonia, and Tuberculosis.

# 5 Testing

## 5.1 Test Cases for Signup

### Test ID Description Input Expected Notes

**Output**

TEST-01 Valid signup Valid details True Account created.

TEST-02 Invalid password Short password False Requires 8+ chars.

TEST-03 Duplicate username Existing username False Must be unique.

TEST-04 Invalid email Email w/o @ False Must be valid format

## 5.2 Testing Units

* **Unit Tests**: Authentication, image processing.
* **Integration Tests**: Full workflow from upload to diagnosis.
* **Performance Tests**: CNN accuracy (≥ 93.54%), processing time (*<* 5 seconds).

# 6 Conclusion

## 6.1 Summary

The system integrates a custom CNN with a Flask web application, achieving over 93.54% accuracy in lung disease classification and providing role-specific access.

## 6.2 Conclusion

The system meets its objectives, offering a reliable tool for healthcare professionals, validated through testing.

## 6.3 Challenges

* Limited availability of diverse datasets.
* Tight training schedule.
* Mobile app conversion issues.
* Device capability limitations.
* Limited medical/biological expertise.

## 6.4 Future Work

* Enhance model accuracy.
* Develop a mobile app.
* Integrate with hospital systems.
* Classify disease severity.
* Publish a research paper.
* Release the dataset on Kaggle.

# 7 References

1. Chest X-Ray Images (Pneumonia). [https://www.kaggle.com/datasets/paultimot](https://www.kaggle.com/datasets/paultimothymoney/chest-xray-pneumonia)hymoney/ [chestxray-pneumonia.](https://www.kaggle.com/datasets/paultimothymoney/chest-xray-pneumonia) Field: AI & Healthcare, No. of images: 5,856.
2. COVID-19 Chest X-Ray Dataset. [https://www.kaggle.com/datasets/scipygaura](https://www.kaggle.com/datasets/scipygaurav/covid19-chest-xray-processed-dataset)v/ [covid19-chestxray-processed-dataset.](https://www.kaggle.com/datasets/scipygaurav/covid19-chest-xray-processed-dataset) Field: AI & Healthcare, No. of images: 4,551.
3. COVID-Pneumonia-Normal Chest X-ray. [https://www.kaggle.com/datasets/ engadhamibrahim/covid-pneumonia-normal-chest-xra.](https://www.kaggle.com/datasets/engadhamibrahim/covid-pneumonia-normal-chest-xra) Field: AI & Healthcare, No. of images: 16,000.
4. Tuberculosis (TB) Chest X-ray Cleaned Database. [https://www.kaggle.com/ datasets/scipygaurav/tuberculosis-tb-chest-x-ray-cleaned.](https://www.kaggle.com/datasets/scipygaurav/tuberculosis-tb-chest-x-ray-cleaned) Field: AI & Healthcare, No. of images: 69,100.
5. Dataset of Tuberculosis Chest X-rays Images. [https://www.kaggle.com/datasets/ tawsifurrahman/tuberculosis-tb-chest-xray-datase.](https://www.kaggle.com/datasets/tawsifurrahman/tuberculosis-tb-chest-xray-datase) Field: AI & Healthcare, No. of images: 4,000.
6. Tuberculosis Chest X-rays. [https://gts.ai/dataset-download/pneumonia-tuberculosis-wi F](https://gts.ai/dataset-download/pneumonia-tuberculosis-with-normal-non-x-ray/)ield: AI & Healthcare, No. of images: 14,000.
7. Lungs Disease Dataset (4 types). [https://www.kaggle.com/datasets/omkarmanoha](https://www.kaggle.com/datasets/omkarmanohardalvi/lungs-disease-dataset-4-types/data)rdalvi/ [lungsdisease-dataset-4-types/data.](https://www.kaggle.com/datasets/omkarmanohardalvi/lungs-disease-dataset-4-types/data) Field: AI & Healthcare, No. of images: 10,000.
8. COVID-19 Radiography Database. [https://www.kaggle.com/datasets/tawsifurr](https://www.kaggle.com/datasets/tawsifurrahman/covid19-radiography-database)ahman/ [covid19radiography-database.](https://www.kaggle.com/datasets/tawsifurrahman/covid19-radiography-database) Field: AI & Healthcare, No. of images: 30,000.
9. Dataset of Tuberculosis Chest X-rays Images. [https://data.mendeley.com/dataset](https://data.mendeley.com/datasets/8j2g3csprk/1)s/ [8j2g3csprk/1.](https://data.mendeley.com/datasets/8j2g3csprk/1) Field: AI & Healthcare, No. of images: 3,000.