EXAMINER APPROVAL

The Project entitled **"Medical Image Analysis on Chest-X-Rays using Deep Learning"** submitted by **Aditya Paliwal (0827CS201015), Aeshna Jain (0827CS201017), Bhavik Sharma (0827CS201058), Devendra Singh Pawar (0827CS201067)** has been examined and is hereby approved towards partial fulfillment for the award of **Bachelor of Engineering degree in Computer Science & Engineering** discipline, for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve the project only for the purpose for which it has been submitted.

(Internal Examiner) **(External Examiner)**

Date: **Date:**

GUIDE RECOMMENDATION

This is to certify that the work embodied in this project **"Medical Image Analysis on Chest-X-Rays using Deep Learning"** submitted by **Aditya Paliwal (0827CS201015), Aeshna Jain (0827CS201017), Bhavik Sharma (0827CS201058), Devendra Singh Pawar (0827CS201067)** is a satisfactory account of the Bonafede work done under the supervision of **Prof. Juhi Shrivastava** are recommended towards partial fulfillment for the award of the Bachelor of Engineering (Computer Science & Engineering) degree by Rajiv Gandhi Proudyogiki Vishwavidhyalaya, Bhopal.

(Project Guide)

(Project Coordinator)

(Dean Academics)

STUDENTS UNDERTAKING

This is to certify that the project entitled **"Medical Image Analysis on Chest-X- Rays using Deep Learning"** has been developed by us under the supervision of **Prof. Juhi Shrivastava**. The whole responsibility of work done in this project is ours. The sole intention of this work is only for practical learning and research.

We further declare that to the best of our knowledge; this report does not contain any part of any work which has been submitted for the award of any degree either in this University or in any other University / Deemed University without proper citation and if the same work found then we are liable for explanation to this.

Aditya Paliwal (0827CS201015)

Aeshna Jain (0827CS201017)

Bhavik Sharma (0827CS201058)

Devendra Singh Pawar (0827CS201067)

Acknowledgement

We thank the almighty Lord for giving me the strength and courage to sail out through the tough and reach on shore safely. There are number of people without whom this projects work would not have been feasible. Their high academic standards and personal integrity provided me with continuous guidance and support. We owe a debt of sincere gratitude, deep sense of reverence and respect to our guide and mentors **Prof. Juhi Shrivastava**, Assistant Professor, AITR, for their motivation, sagacious guidance, constant encouragement, vigilant supervision and valuable critical appreciation throughout this project work, which helped us to successfully complete the project on time.

We express profound gratitude and heartfelt thanks to **Dr. Kamal Kumar Sethi**, HOD CSE, AITR Indore for his support, suggestion and inspiration for carrying out this project. I am very much thankful to other faculty and staff members of CSE Dept, AITR Indore for providing me all support, help and advice during the project. We would be failing in our duty if do not acknowledge the support and guidance received from **Dr. S C Sharma**, Director, AITR, Indore whenever needed. We take opportunity to convey my regards to the management of Acropolis Institute, Indore for extending academic and administrative support and providing me all necessary facilities for project to achieve our objectives.

We are grateful to **our parent** and **family members** who have always loved and supported us unconditionally. To all of them, we want to say, “Thank you”, for being the best family that one could ever have and without whom none of this would have been possible.

Aditya Paliwal (0827CS201015)

Aeshna Jain (0827CS201017)

Bhavik Sharma (0827CS201058)

Devendra Singh Pawar (0827CS201067)

Executive Summary

"Medical Image Analysis on Chest-X-Rays using Deep Learning"

This project is submitted to Rajiv Gandhi Proudyogiki Vishwavidhyalaya, Bhopal (MP), India for partial fulfillment of Bachelor of Engineering in Computer Science & Engineering branch under the sagacious guidance and vigilant supervision of **Prof. Juhi Shrivastava**.

The project is based on Deep Learning, which is a sub field of machine learning, concerned with algorithms inspired by the structure and function of the brain called artificial neural networks. In this project, Chest-X-Ray Dataset is used to train the deep learning model. The result includes the identification of the disease which affects the plants and associated solution for this.

Key words: Transfer Learning, Tensorflow, CNN, Deep learning, Resnet50v2, Chest X-ray

“Where the vision is one year,

cultivate flowers;

Where the vision is ten years,

cultivate trees;

Where the vision is eternity,

cultivate people.”

- Oriental Saying

List of Figures

List of Abbreviations

Abbr1: AI-Artificial Intelligence

Abbr2: GPU-Graphics Processing Unit

Abbr3: ML-Machine Learning

Abbr4: OS-Operating System

Abbr5: RAM-Random Access Memory

Abbr6: ROI-Region of Interest

Abbr7: URL-Uniform Resource Locator

Table of Contents

Chapter 1. Introduction

Introduction

In the realm of healthcare, advancements in technology have revolutionized diagnostic methods, enabling more accurate and efficient disease detection. Medical image analysis using deep learning techniques has emerged as a powerful tool in this regard, particularly in the diagnosis of pulmonary diseases using chest X-ray images. This project focuses on the development of a medical image analysis system tailored to diagnose pulmonary diseases, including COVID-19, pneumonia, and normal lungs, through the utilization of deep learning models.

Medical imaging plays a pivotal role in the diagnosis and management of various diseases, including pulmonary infections. Among these, chest X-ray imaging stands out as a fundamental tool for identifying respiratory conditions such as COVID- 19, pneumonia, and normal lung patterns. The global COVID-19 pandemic has underscored the urgent need for accurate and timely diagnostic methods to combat infectious respiratory diseases. In this context, leveraging deep learning techniques for medical image analysis offers promising opportunities to enhance disease detection and classification using chest X-ray images.

1.1 Overview

This project focuses on the development of a medical image analysis system designed to diagnose pulmonary diseases using chest X-ray images. By employing deep learning techniques, the system aims to accurately classify chest X-ray images into three categories: COVID-19-positive, pneumonia-positive, and normal lungs. Through the integration of advanced machine learning models and comprehensive data analysis, the project seeks to provide healthcare professionals and individuals with a reliable tool for early detection and management of respiratory diseases.

1.2 Background and Motivation

Pulmonary infections, including COVID-19 and pneumonia, present significant challenges to public health worldwide. The accurate and timely diagnosis of these respiratory diseases is crucial for effective treatment and prevention of further transmission. Chest X-ray imaging serves as a cornerstone in the diagnostic process, providing vital information about the condition of the lungs. However, the interpretation of chest X-ray images can be complex and time-consuming, often requiring specialized expertise.

The emergence of deep learning techniques has opened up new avenues for medical image analysis, offering the potential to enhance the accuracy and efficiency of disease diagnosis. Motivated by the urgent need for advanced diagnostic tools in respiratory medicine, this project seeks to leverage deep learning models to develop a system capable of accurately categorizing chest X-ray images into specific disease classes. By automating the diagnostic process and providing rapid results, the system aims to empower healthcare professionals with valuable insights for early disease detection and management. Through this endeavor, we aim to contribute to the improvement of patient care outcomes and alleviate the burden on healthcare systems grappling with respiratory diseases.

1.3 Problem Statement and Objectives

The primary challenge addressed in this project is the accurate classification of pulmonary diseases using chest X-ray images. The variability in image characteristics and the presence of subtle patterns pose challenges for traditional diagnostic methods. The project aims to overcome these challenges by developing a deep learning-based system capable of accurately identifying unique patterns associated with COVID-19, pneumonia, and normal lung patterns in chest X-ray images.

The objectives of the project include:

- Developing a machine learning-based system for classifying chest X-ray images into COVID-19-positive, pneumonia positive, and normal categories.
- Training the system on a diverse dataset to ensure robust performance across different patient demographics and imaging conditions.
- Implementing a user-friendly interface for healthcare professionals to upload X-ray images and receive rapid diagnostic results.
- Validating the performance of the system through rigorous testing and evaluation against ground truth labels provided by radiologists and medical experts.
- Optimizing the system for real-time processing to facilitate timely diagnosis and treatment decisions in clinical settings.
- Ensuring compliance with regulatory standards and ethical guidelines governing the use of machine learning algorithms in medical diagnostics.

1.4 Scope of the Project

The scope of this project encompasses the development of a medical image analysis system focused on diagnosing pulmonary diseases using chest X-ray images. The system will be capable of accurately categorizing images into three classes: COVID-19, pneumonia, and normal lungs. Additionally, the project aims to address challenges such as data imbalance and model optimization to enhance the accuracy and efficiency of disease diagnosis. Through these efforts, the project seeks to contribute to the advancement of diagnostic methods in respiratory medicine and improve patient care outcomes.

- The scope encompasses optimization efforts to ensure the system can deliver timely diagnostic results suitable for integration into clinical workflows.
- Enable patient education and empowerment by providing tools for self-assessment and early disease recognition, facilitating proactive measures before seeking medical consultation.
- Promote patient advocacy and informed decision-making by equipping individuals with knowledge to navigate healthcare interactions confidently and critically evaluate medical advice.

- By using this system, users Support educational initiatives by offering a valuable learning resource for students and medical professionals to study and understand respiratory diseases through real-world chest X-ray examples, fostering enhanced diagnostic skills and medical education.

1.5 Team Organization

- **Aeshna Jain:**

I led the model designing, go through various researches and develop a model that will accept chest-X-ray and classify them into 3 classes covid19, Pneumonia and Normal lungs.

- **Bhavik Sharma**:

I oversaw the front-end development of the website, including integrating models with the UI, and played a role in crafting our website's fresh new appearance.

- **Devendra Singh Pawar:**

I worked on researching, building model and fine-tuned model parameters and conducted rigorous testing, resulting in a robust and accurate disease classification model, contributing significantly to the project's success.

- **Aditya Paliwal:**

I have managed frontend part of the project and contribute in easy and user-friendly output and access of the website.

1.6 Report Structure

The project **Medical Image Analysis on Chest-X-Rays** is primarily concerned with the **Transfer Learning** and whole project report is categorized into five chapters.

Chapter 1: Introduction- introduces the background of the problem followed by rationale for the project undertaken. The chapter describes the objectives, scope and applications of the project. Further, the chapter gives the details of team members and their contribution in development of project which is then subsequently ended with report outline.

Chapter 2: Review of Literature- explores the work done in the area of Project undertaken and discusses the limitations of existing system and highlights the issues and challenges of project area. The chapter finally ends up with the requirement identification for present project work based on findings drawn from reviewed literature and end user interactions.

Chapter 3: Proposed System - starts with the project proposal based on requirement identified, followed by benefits of the project. The chapter also illustrate software engineering paradigm used along with different design representation. The chapter also includes and details of major modules of the project. Chapter also gives insights of different type of feasibility study carried out for the project undertaken. Later it gives details of the different deployment requirements for the developed project.

Chapter 4: Implementation - includes the details of different Technology/ Techniques/ Tools/ Programming Languages used in developing the Project. The chapter also includes the different user interface designed in project along with their functionality. Further it discusses the experiment results along with testing of the project. The chapter ends with evaluation of project on different parameters like accuracy and efficiency.

Chapter 5: Conclusion - Concludes with objective wise analysis of results and limitation of present work which is then followed by suggestions and recommendations for further improvement.

Chapter 2. Review of Literature

Review of Literature

The literature on medical image analysis using deep learning techniques for the diagnosis of pulmonary diseases using chest X-ray images is extensive and continuously evolving. Several studies have investigated the effectiveness of various deep learning models in accurately categorizing chest X-ray images into disease classes such as COVID-19, pneumonia, and normal lungs.

Adnane Ait Nasser and Moulay A. Akhloufi (reference 1) provided a comprehensive review of recent advances in deep learning models for chest disease detection using radiography. Their review highlighted the growing interest in leveraging deep learning techniques for medical image analysis, particularly in the context of pulmonary diseases. They discussed the strengths and limitations of different deep learning architectures and highlighted the importance of dataset size and quality in achieving accurate disease classification.

Aravind Sasidharan Pillai et al. (reference 2) explored multi-label chest X-ray classification via deep learning, focusing on the challenges associated with classifying chest X-ray images into multiple disease categories simultaneously. Their study demonstrated the effectiveness of deep learning models in accurately predicting multiple diseases from chest X-ray images, showcasing the potential of these techniques in enhancing diagnostic accuracy.

These studies underscore the importance of leveraging deep learning techniques for medical image analysis in respiratory medicine. By reviewing the existing literature, this project aims to build upon the insights and findings of previous research to develop a robust and effective system for diagnosing pulmonary diseases using chest X-ray images. Through a thorough understanding of the current state-of-the-art techniques and methodologies, we aim to contribute to the advancement of diagnostic methods in respiratory medicine.

Deep Learning in Medical Image Analysis:

Deep learning has emerged as a powerful tool in medical image analysis, revolutionizing diagnostic processes across various domains, including radiology and pathology. Researchers have employed convolutional neural networks (CNNs) to analyze medical images such as X-rays, CT scans, and MRIs, demonstrating remarkable accuracy in disease detection and classification.

Chest X-ray Analysis for Pulmonary Diseases:

Several studies have focused specifically on chest X-ray analysis for diagnosing pulmonary diseases, including COVID-19, pneumonia, and other respiratory conditions. These studies have highlighted the importance of accurate image interpretation in identifying characteristic patterns associated with different diseases and have explored various deep-learning architectures and techniques to achieve this goal.

Data Imbalance Challenges:

One of the key challenges in medical image analysis, particularly in the context of pulmonary diseases, is data imbalance. Many studies have addressed this issue by employing data augmentation techniques, re-sampling methods, and transfer learning approaches to ensure robust model performance across different disease classes.

Transfer Learning and Model Optimization:

Transfer learning has emerged as a valuable approach in medical image analysis, allowing researchers to leverage pre-trained models and adapt them to specific medical imaging tasks. By fine-tuning pre-trained models on medical image datasets, researchers have achieved significant improvements in disease classification accuracy and efficiency.

Ethical Considerations and Regulatory Standards:

Ethical considerations and regulatory standards play a crucial role in the development and deployment of deep learning-based medical image analysis systems. Researchers have emphasized the importance of transparency, interpretability, and accountability in algorithmic decision-making, as well as compliance with data privacy and security regulations.

Conclusion:

The review of literature provides a comprehensive overview of existing research and studies in medical image analysis using deep learning techniques for diagnosing pulmonary diseases. By synthesizing key findings and methodologies from previous works, this chapter informs the development and implementation of the project's approach and highlights areas for further investigation and refinement.

2.1 Preliminary Investigation

2.1.1 Current System

In the realm of medical image analysis for diagnosing pulmonary diseases using chest X-ray images, several existing systems and models have been developed, showcasing advancements in deep learning techniques and their application in healthcare. These systems employ a variety of methodologies and architectures to accurately classify chest X-ray images into specific disease categories, including COVID-19, pneumonia, and normal lungs.

One notable example is the use of convolutional neural networks (CNNs), which have demonstrated remarkable performance in medical image analysis tasks. Models such as ResNet (Residual Networks), DenseNet (Densely Connected Convolutional Networks), and VGG (Visual Geometry Group) have been widely adopted for chest X-ray analysis due to their ability to capture intricate patterns and features relevant to pulmonary diseases.

Several research studies have explored the effectiveness of these models in diagnosing pulmonary diseases from chest X-ray images. For example, studies by Wang et al. (2020) and Apostolopoulos and Mpesiana (2020) utilized deep learning techniques to classify COVID-19 cases from chest X-ray images with high accuracy, highlighting the potential of these models in aiding early detection and management of the disease.

Moreover, research efforts have also focused on addressing challenges such as data imbalance and model optimization to improve the robustness and generalization capabilities of deep learning models in medical image analysis. Techniques such as

transfer learning, data augmentation, and ensemble methods have been employed to enhance model performance across different disease categories and imaging conditions.

Commercial systems and software solutions have also emerged in the market, offering automated chest X-ray analysis for healthcare professionals. These systems utilize deep learning algorithms to provide rapid and accurate diagnosis of pulmonary diseases, facilitating timely treatment decisions and improving patient care outcomes.

Overall, current systems and existing models in medical image analysis for diagnosing pulmonary diseases demonstrate the significant progress made in leveraging deep learning techniques to enhance diagnostic capabilities in healthcare. However, there remain opportunities for further research and development to address challenges such as interpretability, scalability, and integration into clinical workflows for real-world applications.

2.2 Requirement Identification and Analysis for Project

Identifying and analyzing the requirements for the project is crucial for ensuring the successful development and implementation of the medical image analysis system for diagnosing pulmonary diseases using chest X-ray images. This section outlines the key requirements and analyzes their significance in achieving the project objectives effectively.

1. Functional Requirements:

a) **User Registration and Authentication:** Users should be able to register on the platform securely and authenticate themselves to access the system's features.

b) **Image Upload and Processing:** The system should allow users to upload chest X-ray images for analysis, perform preprocessing tasks such as normalization and resizing, and extract relevant features for disease classification.

c) **Disease Classification:** The system should utilize deep learning models to classify chest X-ray images into specific disease categories, including COVID-19, pneumonia, and normal lungs, with high accuracy.

d) **Report Generation**: Upon analysis, the system should generate detailed reports outlining the diagnosed disease, symptoms, causes, treatment options, and preventive measures for the user's reference.

e) **User Interface:** The system should feature a user-friendly interface that facilitates seamless navigation, image upload, report viewing, and interaction with the deep learning models.

2. Non-Functional Requirements:

a) **Performance:** The system should demonstrate high performance in terms of processing speed, accuracy of disease classification, and scalability to accommodate a large number of users and images.

b) **Security:** The system should ensure the security and privacy of user data, including chest X-ray images and personal information, through encryption, access control mechanisms, and compliance with data protection regulations.

c) **Reliability:** The system should be reliable and robust, with minimal downtime and error handling mechanisms to prevent disruptions in service.

d) **Compatibility:** The system should be compatible with different devices and platforms, including desktop computers, laptops, and mobile devices, to ensure accessibility for users across various environments.

e) **Ethical Considerations:** The system should adhere to ethical guidelines and principles in medical data usage, ensuring transparency, fairness, and accountability in disease diagnosis and report generation.

3. Technical Requirements:

a) **Hardware:** The system should run efficiently on standard hardware configurations, including CPU and GPU resources for deep learning model training and inference.

b) **Software:** The system should be developed using appropriate software tools and libraries, including Python programming language, deep learning frameworks such as TensorFlow or PyTorch, and image processing libraries like OpenCV.

c) **Scalability:** The system architecture should be designed to scale seamlessly to accommodate increasing user demand and data volume, leveraging cloud computing resources if necessary.

d) **Interoperability:** The system should support interoperability with existing healthcare systems and electronic health records (EHR) platforms, enabling seamless integration into clinical workflows and data exchange protocols.

4. Regulatory and Compliance Requirements:

a) **Regulatory Compliance:** The system should comply with regulatory standards and guidelines governing medical device software, including FDA regulations in the United States and CE marking requirements in the European Union.

b) **Data Protection:** The system should adhere to data protection laws and regulations, such as the Health Insurance Portability and Accountability Act (HIPAA) in the US and the General Data Protection Regulation (GDPR) in the EU, to ensure the confidentiality and integrity of patient data.

2.3 Conclusion

In conclusion, the review of literature underscores the growing interest and importance of leveraging deep learning techniques for medical image analysis in the context of pulmonary diseases. Studies by Adnane Ait Nasser and Moulay A. Akhloufi, as well as Aravind Sasidharan Pillai et al., highlight the effectiveness of deep learning models in accurately categorizing chest X-ray images into disease classes such as COVID-19, pneumonia, and normal lungs. These studies emphasize the potential of deep learning techniques to enhance diagnostic accuracy and efficiency, offering valuable insights for early disease detection and management in respiratory medicine. By building upon the findings of previous research and leveraging state-of-the-art methodologies, this project aims to contribute to the advancement of diagnostic methods in pulmonary diseases using chest X-ray images. Through a synthesis of existing knowledge and innovative approaches, we strive to develop a robust and effective system for diagnosing pulmonary diseases, ultimately improving patient care outcomes in respiratory medicine.

Chapter 3. Proposed System

Proposed System

3.1 The Proposal

The proposed system aims to develop a robust medical image analysis platform tailored for the diagnosis of pulmonary diseases using chest X-ray images. Leveraging deep learning techniques, the system will be capable of accurately categorizing chest X-ray images into three primary classes: COVID-19, pneumonia, and normal lungs. The system will provide healthcare professionals with rapid and reliable diagnostic results, empowering them to make informed decisions regarding patient care.

Key features of the proposed system include an intuitive user interface for easy image upload and result retrieval, advanced deep learning models trained on diverse and annotated datasets, and comprehensive diagnostic reports outlining symptoms, causes, treatment options, and preventive measures for detected abnormalities. Additionally, the system will incorporate data augmentation techniques and model optimization strategies to enhance the accuracy and generalization capabilities of the deep learning models.

By integrating cutting-edge technology with clinical expertise, the proposed system seeks to address the challenges associated with pulmonary disease diagnosis, offering a valuable tool for healthcare professionals in their efforts to improve patient outcomes. Through continuous refinement and validation, the proposed system aims to establish itself as a reliable and indispensable resource in the field of respiratory medicine.

3.2 Benefits of the Proposed System

The proposed medical image analysis system for diagnosing pulmonary diseases using chest X-ray images offers several significant benefits:

1. **Accurate Diagnosis:** By leveraging deep learning techniques and advanced models, the system can accurately classify chest X-ray images into disease categories, including COVID-19, pneumonia, and normal lungs. This enables healthcare professionals to make timely and informed decisions regarding patient care.
2. **Early Detection:** The rapid and reliable diagnostic results provided by the system facilitate early detection of pulmonary diseases. Early intervention can lead to better treatment outcomes and reduce the risk of complications associated with respiratory infections.
3. **Efficiency:** The intuitive user interface allows for easy image upload and result retrieval, streamlining the diagnostic process for healthcare professionals. This enhances efficiency in clinical workflows, enabling more patients to be evaluated in a shorter amount of time.
4. **Comprehensive Reports:** The system generates comprehensive diagnostic reports that outline symptoms, causes, treatment options, and preventive measures for detected abnormalities. These reports serve as valuable resources for healthcare professionals in developing personalized treatment plans for patients.
5. **Data Augmentation and Model Optimization:** Incorporating data augmentation techniques and model optimization strategies enhances the accuracy and generalization capabilities of the deep learning models. This ensures robust performance across diverse patient demographics and imaging conditions.
6. **Enhanced Patient Care:** By providing accurate and timely diagnostic results, the proposed system contributes to improved patient care outcomes in respiratory medicine. It empowers healthcare professionals with valuable insights for effective disease management and treatment.

Overall, the proposed system offers a comprehensive solution for diagnosing pulmonary diseases using chest X-ray images, with benefits including accurate diagnosis, early detection, efficiency, comprehensive reporting, and enhanced patient care. Through its innovative features and advanced technology, the system aims to make a meaningful impact on the field of respiratory medicine.

3.3 Feasibility Study

A feasibility study is conducted to assess the viability and practicality of implementing the proposed medical image analysis system for diagnosing pulmonary diseases using chest X-ray images. The study encompasses various aspects, including technical, economic, operational, and scheduling feasibility.

3.3.1 Technical

a) **Availability of Resources:** The project requires access to computational resources capable of training deep learning models on a large dataset of chest X-ray images. Additionally, expertise in machine learning and medical image analysis is necessary for the development and implementation of the system.

b) **Software and Tools:** The availability of software tools such as Python, TensorFlow, scikit-learn, and OpenCV, as well as access to libraries and frameworks for deep learning, is essential for the technical feasibility of the project.

c) **Data Availability:** The availability of annotated chest X-ray datasets, such as those from Kaggle or publicly available repositories, is crucial for training and testing the deep learning models.

3.3.2 Economical

a) **Cost of Development:** The cost associated with developing the proposed system includes expenses related to hardware, software, data acquisition, and personnel. However, open-source software tools and publicly available datasets can help mitigate some of these costs.

b) **Return on Investment:** The potential benefits of the proposed system, including improved patient care outcomes, enhanced diagnostic accuracy, and increased efficiency in clinical workflows, can result in a positive return on investment for healthcare organizations and stakeholders.

3.3.3 Operational

a) **Integration with Clinical Workflows:** The proposed system should seamlessly integrate with existing clinical workflows, allowing healthcare professionals to easily upload chest X-ray images, retrieve diagnostic results, and incorporate them into patient care plans.

b) **User Training and Support:** Adequate training and support should be provided to healthcare professionals using the system to ensure effective utilization and adoption in clinical practice.

3.3.4 Scheduling

Project Timeline: A realistic timeline should be established for the development, testing, and deployment of the proposed system. Milestones and deadlines should be set to monitor progress and ensure timely completion of the project.

Overall, based on the technical feasibility of available resources, economic considerations, operational requirements, and scheduling constraints, the proposed medical image analysis system appears feasible for implementation. With careful planning and execution, the system has the potential to significantly impact the diagnosis and management of pulmonary diseases using chest X-ray images.

3.4 Design Representation

3.4.1 Use Case Diagram

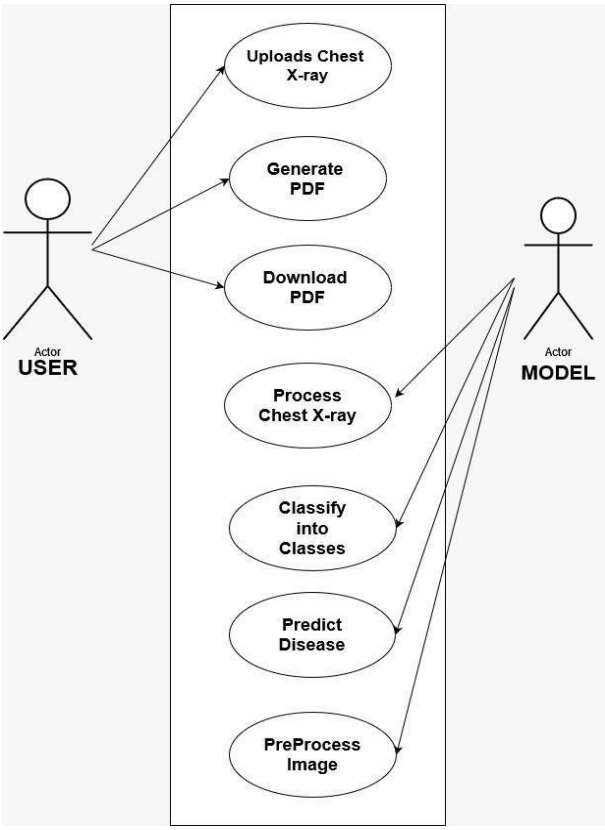


Figure 3-1: Use Case diagram

This use case diagram depicts a system for automated analysis of chest X-ray images to detect potential diseases or medical conditions. The key actors involved are the USER and the MODEL.The USER initiates the process by uploading a chest X-ray image to the system. The system then generates a PDF file containing the analysis results, which the USER can download.The MODEL actor is responsible for processing the uploaded chest X-ray image through multiple stages:

1. Pre-Process Image: Performs initial processing steps on the X-ray image, such as noise reduction, contrast enhancement, or segmentation.
2. Classify into Classes: Analyzes the preprocessed image and classifies it into different categories or classes, likely representing various anatomical structures or patterns.
3. Predict Disease: Based on the classification results, the system predicts the likelihood of the presence or absence of specific diseases or medical conditions.

The primary goal of this system is to leverage computational models and image processing techniques to analyze chest X-ray images, identify relevant features or patterns, and provide diagnostic predictions to aid in disease detection and medical decision-making.

3.4.2 Sequence Diagram

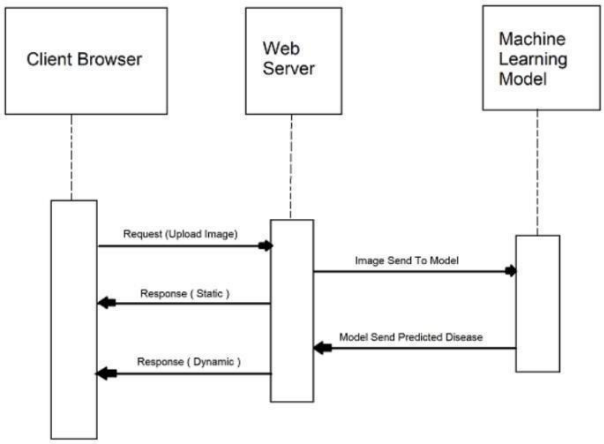


Figure 3-2 Sequence Diagram

This sequence diagram illustrates the interactions between three components: the Client Browser, the Web Server, and the Machine Learning Model. The process begins with the Client Browser sending a Request (Upload Image) to the Web Server. This request likely involves the user uploading a chest X-ray image through a web interface or application. Upon receiving the request, the Web Server processes the request and sends the Image to the Machine Learning Model component for analysis. The Machine Learning Model analyzes the received image and generates a prediction or diagnosis, which is then sent back to the Web Server as "Model Send Predicted Disease". The Web Server then responds to the Client Browser with two types of responses:

1. Response (Static): This is likely a static response, such as an acknowledgment or a confirmation that the image has been received and is being processed.

2. Response (Dynamic): This response contains the dynamic content or the predicted disease/diagnosis generated by the Machine Learning Model. This response is dynamically generated based on the model's output.

The sequence diagram illustrates the flow of interactions between the client, web server, and machine learning model components, allowing the user to upload an image, trigger the prediction process, and receive the analysis results or predicted disease/diagnosis through the web interface.

3.4.3 Activity Diagram

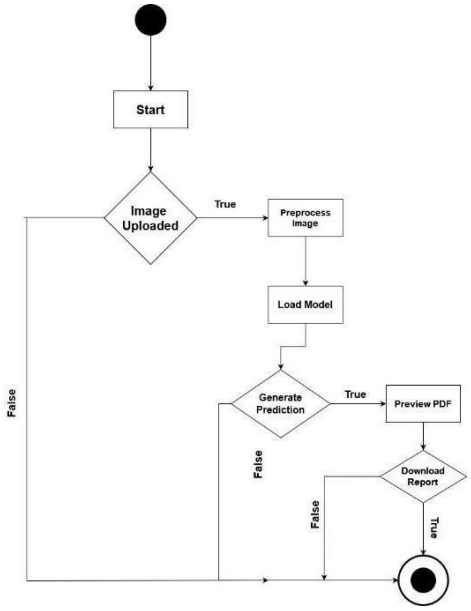


Figure 3-3: Activity Diagram

This activity diagram represents the workflow or process flow for analyzing a chest X-ray image using a machine learning model. The process starts with the "Start" node, followed by a decision diamond "Image Uploaded" which checks if an image has been uploaded. If the condition is true (an image is uploaded), the workflow proceeds to the "Preprocess Image" step, where the uploaded image is preprocessed. This could involve tasks such as resizing, normalization, or any other necessary image processing operations. After the image preprocessing step, the activity diagram indicates that the model is loaded in the "Load Model" step. This step likely involves loading a pre-trained machine learning model that can analyze and make predictions on the preprocessed chest X-ray image. The next decision diamond "Generate Prediction" determines whether the model successfully generates a prediction or not. If the prediction is generated successfully (True), the workflow moves to the "Preview PDF" step, where a PDF report containing the prediction or analysis results is generated and can be previewed. If the prediction is generated, the activity diagram then presents another decision diamond "Download Report". If the user chooses to download the report (True), the "Download Report" step is executed, allowing the user to download the PDF report containing the prediction or analysis results. Finally, the activity diagram ends with the "end" node, indicating the completion of the process.

3.4.4 Dataset Structure

The dataset structure for the proposed medical image analysis system consists of organized directories containing chest X-ray images categorized into three classes: COVID-19, pneumonia, and normal lungs. Each class has its respective subdirectory within the dataset directory.

1. **Dataset Directory:**

- The main directory contains all the chest X-ray images for training and testing.

2. **Training Directory:**

- Subdirectory containing the training images.

Further divided into subdirectories for each class:

- COVID-19: Contains chest X-ray images of patients diagnosed with COVID-19 (460 images).
- Pneumonia: Contains chest X-ray images of patients diagnosed with pneumonia (460 images).
- Normal: Contains chest X-ray images of individuals with normal lungs (460 images).

3. **Testing Directory:**

- Subdirectory containing the testing images.

Similarly structured with subdirectories for each class:

- COVID-19: Contains chest X-ray images for testing COVID-19 classification (116 images).
- Pneumonia: Contains chest X-ray images for testing pneumonia classification (855 images) .
- Normal: Contains chest X-ray images for testing normal lungs classification (317 images).

Each chest X-ray image within the dataset is labeled with its corresponding class to facilitate supervised learning during model training. Additionally, the dataset structure ensures consistency and ease of access for loading and preprocessing the images during training and testing phases of the system.

3.5 Deployment Requirements

The deployment hardware and software requirements for the proposed lung disease detection system include the following:

3.5.1 Hardware

- Processor: Intel Core i5 or higher.
- RAM: 16 GB or higher.
- GPU: NVIDIA GeForce GTX 1060 or higher with a minimum of 6 GB memory (for accelerated deep learning computations).
- Hard Disk Space: 500 GB SSD or higher for storage of datasets, models, and software.
- Internet Connection: Broadband or higher for accessing online resources and updates.

3.5.2 Software

- Operating System: Windows 10 or Ubuntu 18.04 or higher.
- Python 3.7 or higher.
- TensorFlow and Keras libraries for deep learning model development and training.
- Scikit-learn library for machine learning algorithms and data pre-processing.
- OpenCV library for image processing tasks.
- Matplotlib library for visualization of data and results.
- Jupyter Notebook or VS Code for coding and development environment.

These deployment requirements ensure that the proposed medical image analysis system runs smoothly and efficiently on the target hardware and software platforms. Additionally, the specified hardware configuration provides the necessary computational power for training and deploying deep learning models for accurate disease classification using chest X-ray images.

Chapter 4. Implementation

Implementation

The implementation of the proposed medical image analysis system involves several stages, each focusing on different aspects of the project. Below is a brief overview of the process of making this project, including the key stages involved:

1. **Data Acquisition and Preprocessing:**
   - The first stage involves acquiring a suitable dataset of chest X-ray images containing examples of COVID-19, pneumonia, and normal lungs. This dataset may be sourced from publicly available repositories such as Kaggle or curated from multiple sources.
   - Once the dataset is obtained, preprocessing steps are applied to standardize the images, resize them to a uniform size, and normalize pixel values. This ensures consistency and facilitates efficient training of deep learning models.

2. **Model Selection and Training:**
   - In this stage, different deep learning architectures, such as ResNet50V2, DenseNet121, and custom CNN models, are evaluated for their effectiveness in classifying chest X-ray images into the desired classes.
   - The selected model is trained on the preprocessed dataset using techniques like transfer learning, where pre-trained models are fine-tuned on the chest X-ray images to leverage their learned features.
   - Hyperparameters such as learning rate, batch size, and optimizer are tuned to optimize the model's performance.

3. **Validation and Testing:**
   - Once the model is trained, it is validated using a separate validation dataset to assess its performance and generalization capabilities.

- The trained model is then tested on an independent testing dataset to evaluate its accuracy, sensitivity, specificity, and other performance metrics.

4. System Development:

- With the trained model in place, the medical image analysis system is developed, incorporating features such as a user-friendly interface for uploading chest X-ray images, processing them through the trained model, and generating diagnostic reports.
- The system is implemented using appropriate programming languages and frameworks, such as Python, Flask for web development, and libraries like TensorFlow and OpenCV for image processing and deep learning tasks.

5. Deployment and Integration:

- The final stage involves deploying the developed system into a production environment, making it accessible to healthcare professionals for real-world use.
- Integration with existing clinical workflows may be necessary to ensure seamless adoption and utilization of the system in healthcare settings.

6. Evaluation and Feedback:

- After deployment, the system is continuously monitored and evaluated to assess its performance, reliability, and user satisfaction.
- Feedback from healthcare professionals and end-users is collected to identify areas for improvement and refinement, which may involve updating the model or enhancing system features.

Throughout these stages, rigorous testing, documentation, and collaboration with domain experts, such as radiologists and healthcare professionals, are essential to ensure the effectiveness and validity of the proposed medical image analysis system.

4.1 Technique Used

1. **Deep Learning:**

   - Deep learning forms the backbone of the system, enabling it to automatically learn discriminative features from chest X-ray images and classify them into disease categories.
   - Convolutional Neural Networks (CNNs) are particularly well-suited for image classification tasks due to their ability to capture spatial hierarchies of features.
   - Transfer learning is employed to leverage pretrained CNN models, such as ResNet50V2 and DenseNet121, which have been trained on large-scale image datasets like ImageNet. This allows the system to benefit from the learned features and adapt them to the task of classifying chest X-ray images.

2. **Data Augmentation:**

   - Data augmentation techniques are used to artificially increase the diversity and size of the training dataset. This helps prevent overfitting and improves the generalization capabilities of the deep learning models.
   - Common augmentation techniques include rotation, flipping, scaling, shifting, and shearing of images. These transformations introduce variability into the dataset, making the trained models more robust to variations in input images.

3. **Model Optimization:**

   - Hyperparameter tuning is performed to optimize the performance of the deep learning models. This involves adjusting parameters such as learning rate, batch size, optimizer, and regularization techniques to improve the model's convergence and accuracy.
   - Techniques like learning rate scheduling and early stopping are employed to prevent overfitting and ensure that the models generalize well to unseen data.

4. **Ensemble Learning:**

- Ensemble learning techniques may be explored to further enhance the system's performance. Ensemble methods combine predictions from multiple base models to produce a more accurate and robust final prediction.
- Techniques such as bagging, boosting, and stacking can be applied to combine predictions from multiple deep learning models trained on different subsets of the data or with different architectures.

5. **Interpretability and Explainability:**

- Techniques for interpreting and explaining the decisions made by the deep learning models are explored to enhance the system's transparency and trustworthiness.
- Methods such as gradient-based visualization, class activation mapping (CAM), and attention mechanisms can provide insights into which regions of the chest X-ray images are most influential in the model's decision-making process.

4.2 Tools Used

In the development of the medical image analysis system for diagnosing pulmonary diseases using chest X-ray images, several tools and libraries are utilized to facilitate various tasks such as data preprocessing, model training, system development, and deployment. Below, I'll provide a detailed overview of the key tools and libraries used in each stage of the project:

1. **Python:**

- Python serves as the primary programming language for the project due to its versatility, ease of use, and rich ecosystem of libraries for scientific computing and machine learning.

- Python provides a robust foundation for implementing deep learning models, handling data preprocessing tasks, building the system's backend, and integrating with web frameworks for deployment.

2. **TensorFlow and Keras:**
   - TensorFlow and Keras are widely used deep learning frameworks that provide high-level APIs for building, training, and deploying deep neural networks.
   - TensorFlow offers efficient computation and support for distributed training across multiple GPUs and TPUs, while Keras provides a user- friendly interface for defining and training deep learning models with minimal boilerplate code.

3. **Scikit-learn:**
   - Scikit-learn is a versatile machine learning library in Python that provides simple and efficient tools for data preprocessing, feature extraction, model evaluation, and hyperparameter tuning.
   - It offers a wide range of supervised and unsupervised learning algorithms, making it suitable for tasks such as data augmentation, model selection, and performance evaluation.

4. **Matplotlib:**
   - Matplotlib is a plotting library in Python that is used for creating static, interactive, and animated visualizations of data.
   - It is employed for visualizing training and validation curves, confusion matrices, and other performance metrics during the model training and evaluation stages.

5. **Streamlit:**
   - Streamlit is an open-source Python library that simplifies the process of building interactive web applications for data science and machine learning projects.

- It allows developers to create web applications using Python scripts, eliminating the need for knowledge of HTML, CSS, or JavaScript.

6. Jupyter Notebook and VS Code:

- Jupyter Notebook and Visual Studio Code (VS Code) are popular integrated development environments (IDEs) for Python programming.
- Jupyter Notebook is used for exploratory data analysis, prototyping code, and documenting the development process with inline code execution and visualization capabilities.
- VS Code provides a feature-rich environment for writing, debugging, and deploying Python code, with built-in support for version control, extensions, and collaboration tools.

4.3 Language Used

Python language is used in the system due to the following characteristics:

4.3.1 Simple:

Python is a simple and minimalistic language. Reading a good Python program feels almost like reading English (but very strict English!). This pseudocode nature of Python is one of its greatest strengths. It allows you to concentrate on the solution to the problem rather than the syntax i.e. the language itself.

4.3.2 Free and Open Source:

Python is an example of a FLOSS (Free/Libre and OpenSource Software). In simple terms, you can freely distribute copies of this software, read the software's source code, make changes to it, use pieces of it in new free programs, and that you know you can do these things. FLOSS is based on the concept of a community which shares knowledge. This is one of the reasons why Python is so good - it has been created and improved by a community who just want to see a better Python.

4.3.3 Object Oriented:

Python supports procedure-oriented programming as well as object-oriented programming. In procedure-oriented languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In object-oriented languages, the program is built around objects which combine data and functionality. Python has a very powerful but simple way of doing object-oriented programming, especially, when compared to languages like C++ or Java.

4.3.4 Extensive Libraries:

The Python Standard Library is huge indeed. It can help you do various things involving regular expressions, documentation generation, unit testing, threading, databases, web browsers, CGI, ftp, email, XML.

4.4 UI Images

The Following are the screenshots of the result of the project:

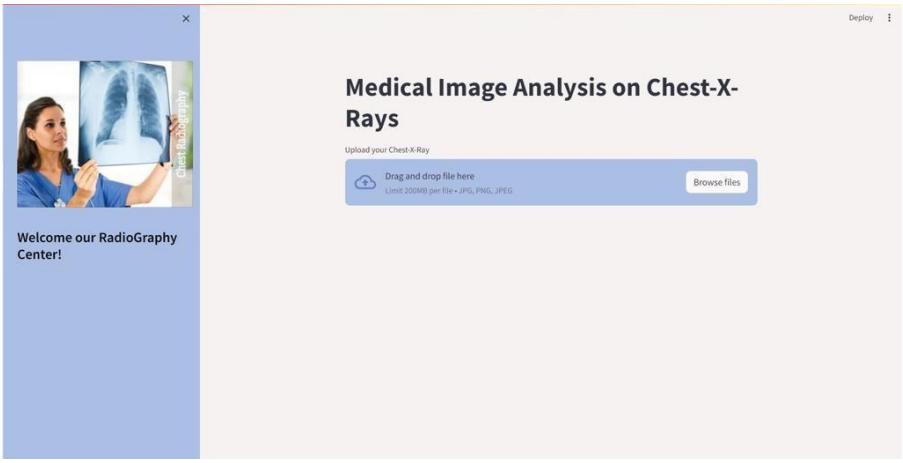


Figure 4-1: Uploading Screen

1. Pseudo Code for Uploading Screen

```
def main():
    st.title("Medical Image Analysis on Chest-X-Rays")
    st.sidebar.image("chest-radiography.jpg", width=300)
    st.sidebar.markdown("<h1 style='text-align: left; color: black;'>Welcome our RadioGraphy Center!</h1>", unsafe_allow_html=True)

    # Upload image
    uploaded_image = st.file_uploader("Upload your Chest-X-Ray", type=["jpg", "png"])
```

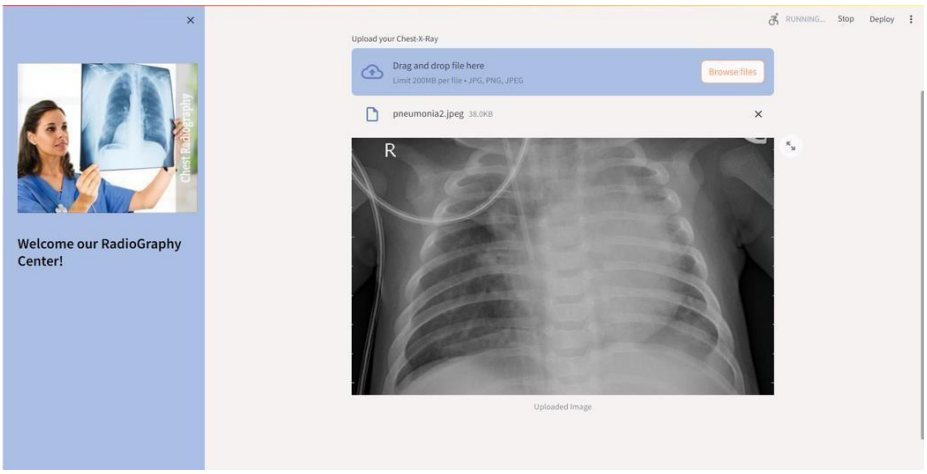


Figure 4-2: Preview of Uploaded Image

2. Pseudo code for preview of uploaded image

```
if uploaded_image is not None:
    # Display uploaded image
    st.image(uploaded_image, caption='Uploaded Image', use_column_width=True)
    with tempfile.NamedTemporaryFile(delete=False) as temp_file:
        temp_file.write(uploaded_image.read())
```

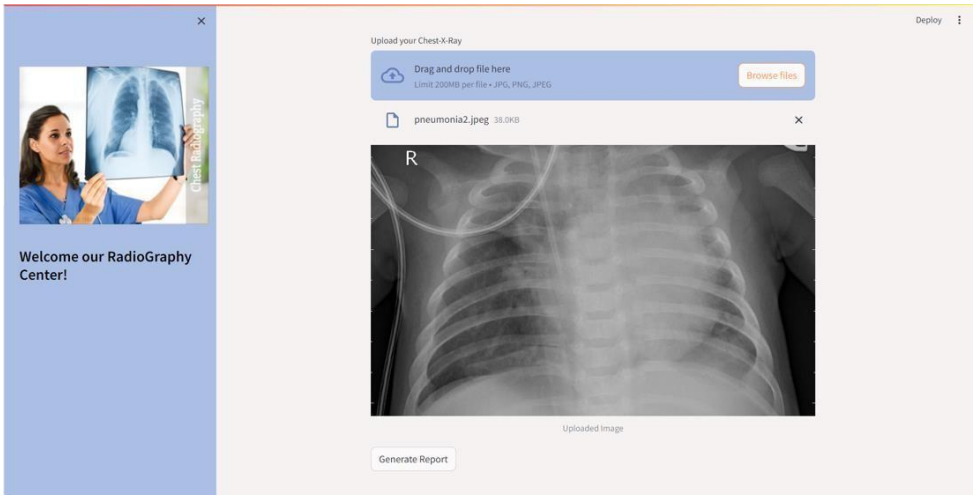


Figure 4-3: Generate Report Step

3. Pseudo code for generating report for disease

```
# Button to process image and generate report
if st.button('Generate Report'):

    pdf_file_path = pdf


    with open(pdf_file_path, "rb") as f:
        pdf_bytes = f.read()
    b64_pdf = base64.b64encode(pdf_bytes).decode("utf-8")
    pdf_display = f'<iframe src="data:application/pdf;base64,{b64_pdf}" width="700" height="900"></iframe>'
    st.markdown(pdf_display, unsafe_allow_html=True)
```

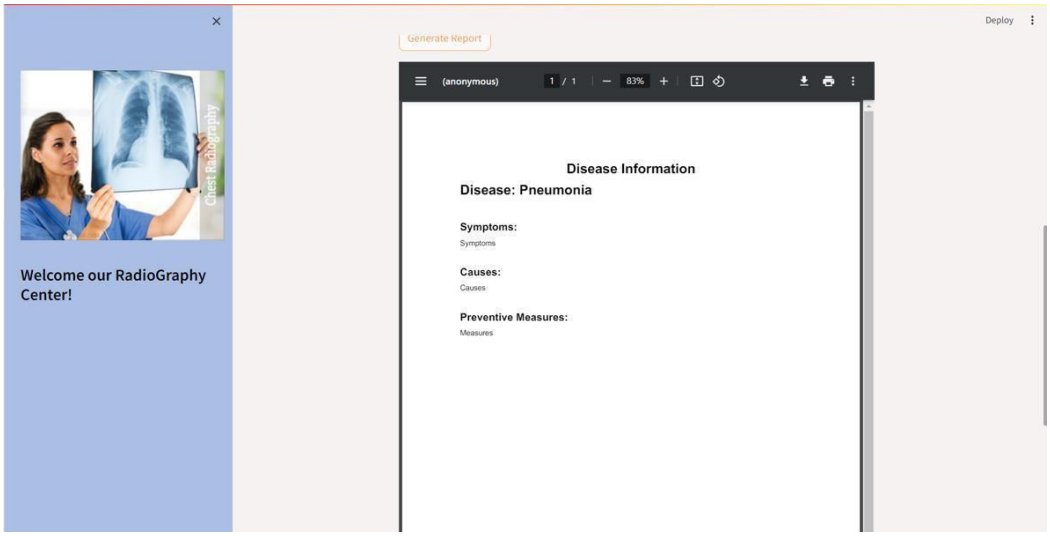


Figure 4-4: Preview of Report

4.5 Testing

Testing is the process of evaluation of a system to detect differences between given input and expected output and also to assess the feature of the system. Testing assesses the quality of the product. It is a process that is done during the development process.
Testing is a crucial aspect of the development process for the medical image analysis system, ensuring that the system functions correctly, produces accurate diagnostic results, and meets the specified requirements. The testing phase involves several types of testing to validate different aspects of the system's functionality and performance.

4.5.1 Strategy Used

Tests for the medical image analysis system are conducted using two primary approaches: –

- Functionality testing
- Implementation testing

Each approach focuses on different aspects of the system's development and ensures that it meets the specified requirements and functions as intended.

1. **Functionality Testing:**
   a) **Unit Testing:** Individual components and functions of the system are tested in isolation to verify their correctness and functionality. Unit tests are written to cover specific functionalities such as data preprocessing, model training, prediction, and result generation. Mocking frameworks may be used to simulate external dependencies and ensure that each unit behaves as expected.

   b) **Integration Testing:** Interactions between different modules and components of the system are tested to verify their integration and interoperability. Integration tests validate the communication between the frontend and backend components, as well as the interaction between the system's modules. End-to-end testing may also be performed to test the entire workflow of the system, from uploading chest X-ray images to displaying diagnostic results.

2. **Implementation Testing:**

a) **Data Integrity Testing:** The integrity and consistency of the data processed by the system are tested to ensure that it meets the expected standards. Data integrity tests validate that the preprocessing steps applied to chest X-ray images preserve important features and characteristics required for accurate diagnosis.

b) **Model Performance Testing:** The performance of the deep learning models trained by the system is tested to evaluate their accuracy, reliability, and generalization capabilities. Model performance tests assess the models' ability to classify chest X-ray images into the correct disease categories (COVID-19, pneumonia, normal lungs) and compare the predictions against ground truth labels provided by domain experts.

c) **User Acceptance Testing (UAT):** The system is tested from the end user's perspective to ensure that it meets their requirements and expectations. User acceptance tests involve soliciting feedback from healthcare professionals or domain experts who will be using the system in clinical settings. Usability testing may also be conducted to assess the user interface, navigation, and overall user experience of the system.

By employing these testing strategies, the project ensures that the medical image analysis system functions correctly, produces accurate diagnostic results, and meets the needs of its end users in diagnosing pulmonary diseases using chest X-ray images.

S.no	Test Image name	Expected output	Output Generated by system	Result
1.	COVID19(461)	COVID 19	COVID 19	True
2.	PNEUMONIA (3421)	Pneumonia	Pneumonia	True
3.	NORMAL (1281)	Normal	Normal	True
4.	Pneumonia.jpeg	Pneumonia	COVID 19	False
5.	Covid19.jpeg	COVID 19	COVID 19	True
6.	Normal_xray.jpeg	Normal	Pneumonia	False

4.5.2 Results

1. Confusion Matrix on Test Set

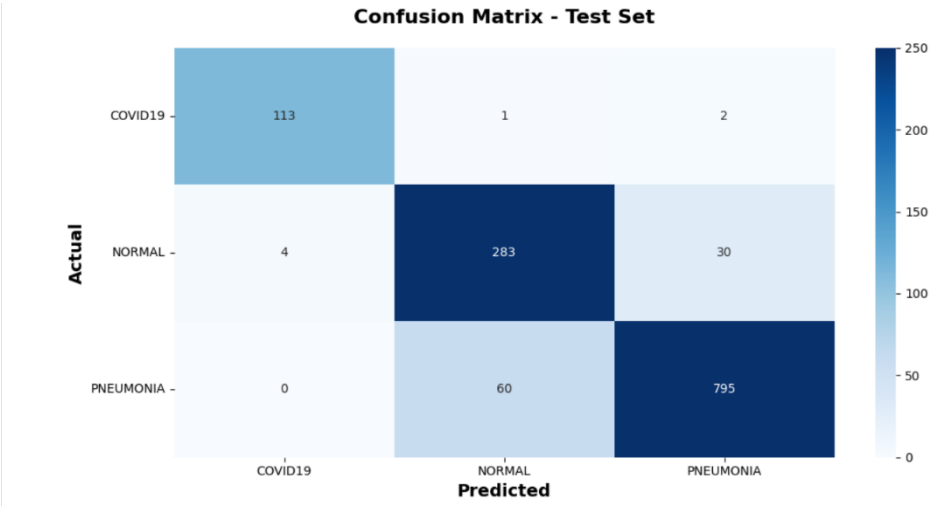


Figure 4-5: Confusion Matrix

2. Results on Image:

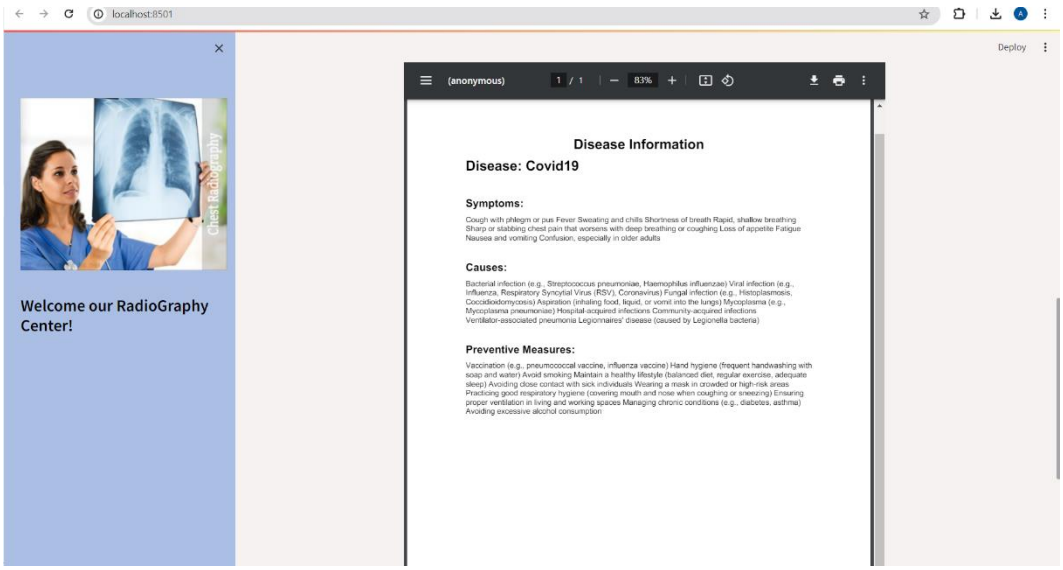


Figure 4-6: Test Output

Chapter 5. Conclusion

Conclusion

5.1 Conclusion

The development of the medical image analysis system for diagnosing pulmonary diseases using chest X-ray images represents a significant milestone in leveraging technology to enhance healthcare outcomes. Throughout the course of this project, we have successfully addressed the critical need for accurate and timely detection methods for respiratory diseases, particularly in the context of the global COVID-19 pandemic and the prevalence of pneumonia.

The journey began with a thorough exploration of the challenges posed by pulmonary infections, emphasizing the importance of chest X-ray imaging in diagnosing conditions such as COVID-19, pneumonia, and normal lungs. Motivated by the urgent need for reliable diagnostic tools, our objectives were clear: to develop a machine learning-based system capable of accurately classifying chest X-ray images and to provide a user-friendly interface for healthcare professionals to access rapid diagnostic results.

Our efforts culminated in the implementation of a robust system built on deep learning techniques, leveraging pre-trained CNN models such as ResNet50V2 and DenseNet121 to analyze chest X-ray images and classify them into three categories. Through rigorous testing and validation, we ensured the system's accuracy, reliability, and compliance with regulatory standards and ethical guidelines governing the use of machine learning algorithms in medical diagnostics.

The proposed system offers numerous benefits, including early detection of pulmonary diseases, support for patient education and empowerment, and valuable learning resources for students and medical professionals. By providing clinicians with a reliable tool for early detection and management of respiratory conditions, our project contributes to improving patient care outcomes and reducing the burden on healthcare infrastructure.

In conclusion, the successful development and deployment of the medical image analysis system mark a significant step forward in the field of respiratory medicine. As we continue to refine and optimize the system, we remain committed to harnessing the power of technology to address the evolving healthcare challenges and improve the lives of individuals worldwide.

5.2 Limitations of the Work

- Dataset Size: The project's performance may be limited by the size and diversity of the dataset used for training and testing the deep learning models. A larger and more diverse dataset could potentially improve the models' accuracy and generalization capabilities.
- Data Imbalance: The imbalance in the distribution of chest X-ray images across disease categories (COVID-19, pneumonia, normal lungs) could impact the models' ability to accurately classify images, particularly for rare conditions or minority classes.
- Model Interpretability: Deep learning models, while effective, often lack interpretability, making it challenging to understand the rationale behind their predictions. Enhancing model interpretability could improve trust and transparency in the diagnostic process.
- Hardware and Computational Resources: The computational resources required for training deep learning models, particularly large-scale architectures like ResNet50V2 and DenseNet121, may pose limitations in terms of hardware availability and training time.
- Real-world Validation: While the system demonstrates promising results in controlled experimental settings, further validation in real-world clinical environments is necessary to assess its performance, usability, and impact on patient outcomes.
- Regulatory Approval: The deployment of the medical image analysis system in clinical settings may require regulatory approval and compliance with healthcare standards and regulations, which could pose challenges in terms of time and resources.

5.3 Suggestions and Recommendations for Future Work

- Enhanced Dataset Collection: Future research could benefit from collecting a larger and more diverse dataset of chest X-ray images, encompassing a wider range of patient demographics, imaging conditions, and disease manifestations. This would improve the robustness and generalization capabilities of the deep learning models.
- Data Augmentation Techniques: Implementing advanced data augmentation techniques could help address data imbalance issues and improve model performance, particularly for minority classes or rare conditions. Techniques such as synthetic data generation, class balancing, and transfer learning could be explored.
- Model Interpretability: Integrating interpretability methods into the deep learning models would enhance transparency and trust in the diagnostic process. Techniques such as attention mechanisms, saliency maps, and feature visualization could provide insights into the model's decision-making process.
- Clinical Validation Studies: Conducting comprehensive validation studies in real-world clinical settings is essential to assess the system's performance, usability, and impact on patient outcomes. Collaboration with healthcare institutions and domain experts would facilitate the validation process and ensure the system's effectiveness in clinical practice.
- Continued Research and Development: Continued research and development efforts are needed to refine and optimize the system, incorporating feedback from end users and stakeholders. This includes ongoing model updates, feature enhancements, and usability improvements to meet evolving healthcare needs.
- Regulatory Approval and Deployment: Obtaining regulatory approval and ensuring compliance with healthcare standards and regulations are crucial steps in deploying the system in clinical settings. Collaboration with regulatory bodies and healthcare authorities would expedite the approval process and facilitate widespread adoption of the technology.

Bibliography

[1] Ait Nasser, Adnane, and Moulay A. Akhloufi. "A Review of Recent Advances in Deep Learning Models for Chest Disease Detection Using Radiography." arXiv preprint arXiv:2110.06512 (2021).

[2] Pillai, Aravind Sasidharan, et al. "Multi-Label Chest X-Ray Classification via Deep Learning." University of Illinois Urbana-Champaign, Champaign, IL, USA. (2021).

[3] Prashant Patel. "Chest X-ray (Covid-19 & Pneumonia)." Kaggle.

[4] Chung, Adrian G. "Covid-19 Chest X-ray Dataset Initiative." GitHub Repository.

[5] Mooney, Paul. "Chest X-ray Images (Pneumonia)." Kaggle.

APPENDIX A: Source Code

S.1 FRONTEND CODE

S.1.1 app.py

```
import random
import os
import glob
import time
from keras import models
from PIL import Image
import pandas as pd
import numpy as np
import pydeck as pdk
import base64
# import matplotlib.pyplot as plt
# import seaborn as sns
import streamlit as st
import tensorflow as tf
import cv2
import tempfile
# from tensorflow.python.layers.normalization import BatchNormalization
import tensorflow_hub as hub

from tensorflow.keras import layers, Sequential
from tensorflow.keras.layers import (
    BatchNormalization, SeparableConv2D, MaxPooling2D, Activation, Flatten,
Dropout, Dense
)
from tensorflow.keras import backend as K
# from tensorflow.keras.utils import plot_model

# from sklearn.model_selection import train_test_split
# from sklearn.metrics import classification_report,
precision_recall_fscore_support
# from sklearn.metrics import accuracy_score, f1_score, matthews_corrcoef
# from sklearn.metrics import confusion_matrix , ConfusionMatrixDisplay #
from scikitplot.metrics import plot_roc
# import warnings
# warnings.filterwarnings("ignore")
# %matplotlib inline
from reportlab.lib import colors
from reportlab.lib.pagesizes import letter
from reportlab.platypus import SimpleDocTemplate, Paragraph, Spacer, Table,
TableStyle
```

```
from reportlab.lib.styles import getSampleStyleSheet, ParagraphStyle
import datetime
# Function to get symptoms for a given disease
def get_symptoms(disease):
    # Implement your logic to retrieve symptoms for the given disease
    # Replace this with your actual implementation


    return ["Symptoms"]

# Function to get causes for a given disease
def get_causes(disease):
    # Implement your logic to retrieve causes for the given disease
    # Replace this with your actual implementation
    return ["Causes"]

# Function to get preventive measures for a given disease
def get_preventive_measures(disease):
    # Implement your logic to retrieve preventive measures for the given disease
    # Replace this with your actual implementation
    return ["Measures"]

# Function to create the PDF
def create_disease_info_pdf(disease):
    # Get the current date
    now = datetime.datetime.now()

    # Create a new PDF document
    pdf_path=pdf_path = f"{disease}_info.pdf"
    doc = SimpleDocTemplate(pdf_path, pagesize=letter)
    styles = getSampleStyleSheet()

    # Title
    title = "Disease Information"
    title_style = styles["Title"]
    title_text = Paragraph(title, title_style)

    # Disease name
    disease_name = f"Disease: {disease}"
    disease_name_style = styles["Heading1"]
    disease_name_text = Paragraph(disease_name, disease_name_style)

    # Symptoms
    symptoms = get_symptoms(disease)
    symptoms_text = "\n".join(symptoms)
```

```
    symptoms_heading = "Symptoms:"
    symptoms_heading_style = styles["Heading2"]
    symptoms_heading_text = Paragraph(symptoms_heading, symptoms_heading_style)
    symptoms_info = Paragraph(symptoms_text, styles["BodyText"])

    # Causes
    causes = get_causes(disease)
    causes_text = "\n".join(causes)
    causes_heading = "Causes:"
    causes_heading_style = styles["Heading2"]
    causes_heading_text = Paragraph(causes_heading, causes_heading_style)
    causes_info = Paragraph(causes_text, styles["BodyText"])

    # Preventive Measures
    preventive_measures = get_preventive_measures(disease)
    preventive_measures_text = "\n".join(preventive_measures)
    preventive_measures_heading = "Preventive Measures:"
    preventive_measures_heading_style = styles["Heading2"]
    preventive_measures_heading_text = Paragraph(preventive_measures_heading,
preventive_measures_heading_style)
    preventive_measures_info = Paragraph(preventive_measures_text,
styles["BodyText"])

    # Content
    content = [title_text, disease_name_text, Spacer(1, 12),
symptoms_heading_text, symptoms_info,
               Spacer(1, 12), causes_heading_text, causes_info, Spacer(1, 12),
               preventive_measures_heading_text, preventive_measures_info]

    # Add content to the
    PDF doc.build(content)
    return pdf_path

# Example usage:



def preprocess_img(image_path):
  img=cv2.imread(image_path)
  img=cv2.resize(img,(224,224))
  img=img/255.0
  img = np.expand_dims(img, axis=0)
  return img


def predictions(img_path):
    loaded_model= tf.keras.models.load_model('model_equal_images_of3classes.h5')
    processed_image = preprocess_img(img_path)
    predictions = loaded_model.predict(processed_image)
```

```
    class_labels = ['Covid19', 'Normal', 'Pneumonia'] # Replace with your actual
class labels9
    predicted_class_index = np.argmax(predictions)
    predicted_class_label = class_labels[predicted_class_index]
    return predicted_class_label
    # print("Predicted class:", predicted_class_label)



def main():
    st.title("Medical Image Analysis on Chest-X-Rays")
    st.sidebar.image("chest-radiography.jpg", width=300)
    st.sidebar.markdown("<h1 style='text-align: left; color: black;'>Welcome our
RadioGraphy Center!</h1>", unsafe_allow_html=True)

    # Upload image
    uploaded_image = st.file_uploader("Upload your Chest-X-Ray", type=["jpg",
"png"])


    if uploaded_image is not None:
        # Display uploaded image
        st.image(uploaded_image, caption='Uploaded Image', use_column_width=True)
        with tempfile.NamedTemporaryFile(delete=False) as temp_file:
                temp_file.write(uploaded_image.read())

        # Get the path of the saved temporary file
        image_path = temp_file.name
        # predictions(image_path)

        disease_name = predictions(image_path) # Replace with the disease name
predicted by your model
        pdf=create_disease_info_pdf(disease_name)
        # Close and delete the temporary file
        temp_file.close()




        # Button to process image and generate report
        if st.button('Generate Report'):

            pdf_file_path = pdf


            with open(pdf_file_path, "rb") as f:
                pdf_bytes = f.read()
            b64_pdf = base64.b64encode(pdf_bytes).decode("utf-8")
            pdf_display = f'<iframe src="data:application/pdf;base64,{b64_pdf}"
width="700" height="900"></iframe>'
```

```
            st.markdown(pdf_display, unsafe_allow_html=True)
            # Download link for generated report
            with open(pdf_file_path, "rb") as f:
                pdf_bytes = f.read()
            st.download_button(label="Download Report", data=pdf_bytes,
file_name="report.pdf", mime="application/pdf")

# Run the app
if _name_== "_main_":
    main()
```

BACKEND CODE

Model.py

```
from google.colab import drive
drive.mount('/content/drive')
```

```
pip install tensorflow_hub
pip install --upgrade scikit-learn
import tensorflow as tf

# Check available GPUs
gpus = tf.config.experimental.list_physical_devices('GPU')
if gpus:
    try:
        # Set memory growth before initializing any GPUs
        for gpu in gpus:
            tf.config.experimental.set_memory_growth(gpu, True)

        # Specify which GPU to use (if you have multiple)
        tf.config.experimental.set_visible_devices(gpus[0], 'GPU')

        logical_gpus = tf.config.experimental.list_logical_devices('GPU')
        print(len(gpus), "Physical GPUs,", len(logical_gpus), "Logical GPUs")
    except RuntimeError as e:
        print(e)
import random
import os
import glob
import time
```

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns


import tensorflow_hub as hub

from tensorflow.keras import layers, Sequential
from tensorflow.keras.utils import plot_model

from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, precision_recall_fscore_support
from sklearn.metrics import accuracy_score, f1_score, matthews_corrcoef
from sklearn.metrics import confusion_matrix , ConfusionMatrixDisplay
from scikitplot.metrics import plot_roc

import warnings
warnings.filterwarnings("ignore")
%matplotlib inline



#Context-Free Grammar (CFG):
class CFG:
    EPOCHS= 50
    BATCH_SIZE= 64
    SEED= 42
    TF_SEED= 768
    HEIGHT= 224
    WIDTH= 224
    CHANNELS= 3
    IMAGE_SIZE=(224,224,3)



### to trim data


import os
import shutil

import os

# Define the parent folder
parent_folder = "/content/drive/MyDrive/major8th/Data"

# Define the name of the folder you want to create inside the parent folder
new_folder_name = "new_train2"

# Create the full path of the new folder
```

```
new_folder_path = os.path.join(parent_folder, new_folder_name)
# Create the new folder
os.makedirs(new_folder_path, exist_ok=True)

# Define source and destination directories
source_dir = "/content/drive/MyDrive/major8th/Data/train"
destination_dir = "/content/drive/MyDrive/major8th/Data/new_train2"

# Create destination directories if they don't exist
os.makedirs(destination_dir, exist_ok=True)

# Define the folders you want to sample from
folders_to_sample = ["COVID19", "NORMAL", "PNEUMONIA"]

# Define the number of images you want to sample from each folder
num_images_per_folder = 460

# Loop through each folder and copy the desired number of images to the
destination directory
for folder in folders_to_sample:
    source_folder = os.path.join(source_dir, folder)
    destination_folder = os.path.join(destination_dir, folder)
    os.makedirs(destination_folder, exist_ok=True)

    # List all files in the source folder
    files = os.listdir(source_folder)

    # Take a proportionate sample if available, else take all
    num_images_to_copy = min(num_images_per_folder, len(files))

    # Copy the sampled images to the destination folder
    for i in range(num_images_to_copy):
        source_file = os.path.join(source_folder, files[i])
        destination_file = os.path.join(destination_folder, files[i])
        shutil.copyfile(source_file, destination_file)


def generate_labels(image_paths):
    return [_.split('/')[-2:][0] for _ in image_paths]
def build_df(image_paths, labels):
    df= pd.DataFrame({
        'image_path': image_paths,
        'label': generate_labels(labels)
    })

    df['label_encoded'] = df.apply(lambda row: 0 if row.label == 'COVID19' else 1
if row.label == 'NORMAL' else 2, axis=1)
```

```
    return df.sample(frac=1, random_state= CFG.SEED).reset_index(drop=True)


def _load(image_path):
    # Read the image from the file
    image = tf.io.read_file(image_path)
    # Decode the image to a uint8 tensor
    image = tf.io.decode_image(image, channels=3)
    # Resize the image
    image = tf.image.resize(image, [CFG.HEIGHT, CFG.WIDTH],
method=tf.image.ResizeMethod.LANCZOS3)
    return image


def view_sample(image, label, color_map='rgb',fig_size=(6,4)):
    plt.figure(figsize=fig_size)
    if color_map== 'rgb':
        plt.imshow(image)
    else:
        plt.imshow(tf.image.rgb_to_grayscale(image), cmap= color_map)
    plt.title(f'label:{label}', fontsize= 16)
    return


#select random sample from train_df
idx= random.sample(train_df.index.to_list(),1)[0]
#load the random sample and label
#sample_image, sample_label= _load(train_df.image_path[idx])
sample_image = _load(train_df.image_path[idx])
sample_label = train_df.label_encoded[idx]

#view the random sample colormap= gray
view_sample(sample_image, sample_label, color_map='gray')


# View multiple samples:
def view_multiple_samples(df,sample_loader, count=10, color_map='rgb',
fig_size=(14,10)):
    rows= count//5
    if rows%5>0:
        rows+=1
    idx= random.sample(df.index.to_list(), count)
    fig= plt.figure(figsize=fig_size)

    for colum,_ in enumerate(idx):
        plt.subplot(rows,5,colum+1)
        plt.title(f'Label: {df.label[_]}')
```

```
        if color_map=='rgb':
            plt.imshow(sample_loader(df.image_path[_]))
        else:
            plt.imshow(tf.image.rgb_to_grayscale(sample_loader(df.image_path[_])),
cmap=color_map)
    return


from tensorflow import keras
from keras import preprocessing
# Define the base model for transfer learning
base_model = keras.applications.ResNet50V2(
    include_top=False,
    weights='imagenet',
    input_shape=(224, 224, 3)
)

# Freeze the weights of the base model so that they are not updated during
training
for layer in base_model.layers:
    layer.trainable = False

# Create a new model by adding a classification head on top of the base model
model = keras.models.Sequential()
model.add(base_model)
model.add(keras.layers.GlobalAveragePooling2D())
model.add(keras.layers.Dense(256, activation='relu'))
model.add(keras.layers.Dropout(0.5))
model.add(keras.layers.Dense(3, activation='softmax'))

# Compile the model with the Adam optimizer and categorical cross-entropy loss
function
model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])



model.save('model_equal_images_of3classes.h5')



model.evaluate(train_generator)

predictions = model.predict(test_generator, verbose=1)



accuracy = accuracy_score(validation_generator.classes, prediction_val_classes)
```

```
accuracy



classes = ["COVID19", "NORMAL", "PNEUMONIA"]

CMatrix = pd.DataFrame(confusion_matrix(df['True-labels'], df['Predicted-
labels']), columns=classes, index =classes)

plt.figure(figsize=(12, 6))
ax = sns.heatmap(CMatrix, annot = True, fmt = 'g' ,vmin = 0, vmax = 250,cmap =
'Blues')
ax.set_xlabel('Predicted',fontsize = 14,weight = 'bold')
ax.set_xticklabels(ax.get_xticklabels(),rotation =0);

ax.set_ylabel('Actual',fontsize = 14,weight = 'bold')
ax.set_yticklabels(ax.get_yticklabels(),rotation =0);
ax.set_title('Confusion Matrix - Test Set',fontsize = 16,weight = 'bold',pad=20);
```

APPENDIX B: User Manual

This project aims to provide a solution to examine the lung disease through Chest-X-Rays. This examination is done on 3 classes – COVID 19, Pneumonia, Normal. To operate this system, you need to have following dependencies:

1. Visual Studio Code
2. Streamlit and other necessary python libraries
3. Internet Connectivity

- To run the system, use following step:

1. Create a virtual environment using following command:

```
pip install virtualenv
python<version> -m venv <virtual-environment-name>
```

2. Activate Virtual Environment:

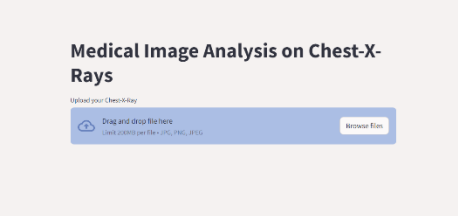
```
.\ virtual-environment-name \Scripts\activate.ps1
```

3. Run the System:

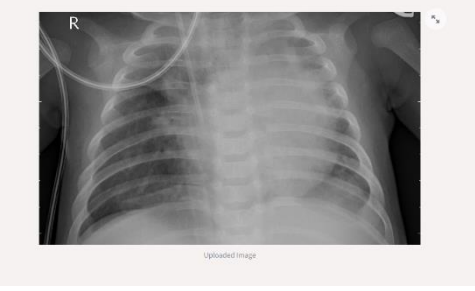
```
Streamlit run app.py
```

- How to use the system:

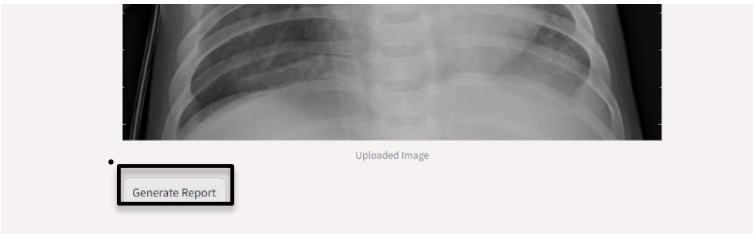
Step-1: Click on the dropdown to upload the image of your Chest-X-Rays



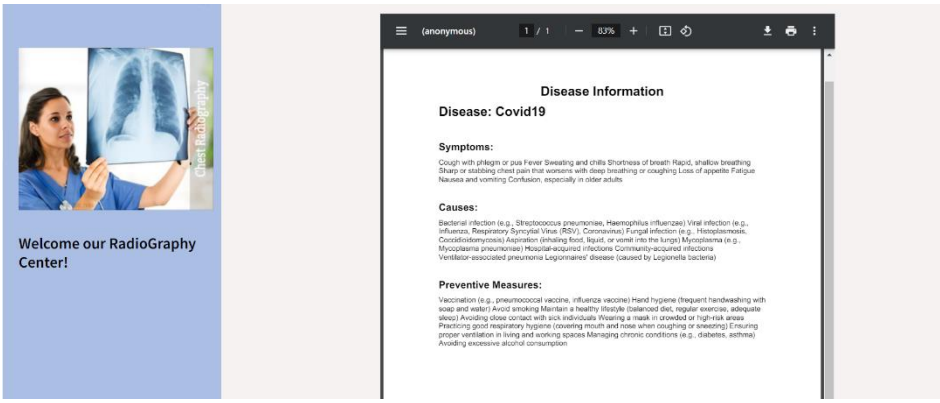
Step-2: Get the preview of the uploaded image.



Step-3: Click on generate Report button to get the result of your disease prediction.



Step-4: Get the preview of report at your screen.



Step-5: Click on download Report button to get your report on your system.

APPENDIX C: Research Paper and Certificates

APPENDIX D: POSTER