AI-Powered Medical Chatbot for Disease Diagnosis and Prescription from Chest X-Ray Images

Software Engineering DSN 5003

SRS (Software Requirement Specification)

By-

Aekam Singh Sidhu (23106060)

Arnav Singh (23106054)

Table of Contents

1.0. Project Overview:

The goal of the proposed project is to create and put into use an AI-powered medical chatbot that can look at chest X-ray images and make diagnoses and prescription suggestions. The system's main goal is to find COVID-19 and pneumonia early on, two diseases that have a big effect on world health. The system merges a deep learning model with an intelligent chatbot interface to give speedy, reliable, and accessible diagnostic support.

The system uses ResNet-50 for image classification. This is a sophisticated convolutional neural network architecture that is known for being able to get deep hierarchical information from medical pictures. By processing chest X-rays with ResNet-50, the model can accurately identify between normal, pneumonia, and COVID-19 cases. Once the diagnostic result is ready, the chatbot talks about it in a conversational way, explains the results clearly, and proposes some initial prescription alternatives that are in keeping with conventional clinical procedures.

The main goal is to reduce the time it takes to get a diagnosis, help healthcare workers make decisions, and make it easier for patients to get medical advice, especially in areas where there aren't enough resources or people.

1.1 Purpose

The purpose of this paper is to clearly specify the requirements, objectives, and features of the AI-powered medical chatbot system for disease detection and prescription generation from chest X-ray pictures. This Software Requirements Specification (SRS) acts as a reference for developers, project stakeholders, and end-users, ensuring that all parties share a consistent knowledge of the system’s aims and expected outputs. It provides a systematic framework that guides the design, development, testing, and deployment phases of the project.

Specifically, the document discusses the role of ResNet-50 in processing chest X-ray pictures for appropriate classification into normal, pneumonia, or COVID-19 categories. It also outlines how the chatbot module will communicate with

consumers, evaluate diagnostic data, and make medically aligned prescription suggestions. Additionally, this paper specifies functional requirements, non-functional needs, performance objectives, and system limits. By serving as both a technical blueprint and a communication tool, the document guarantees that the final product fits user needs, complies to quality standards, and supports timely, dependable, and accessible healthcare solutions.

1.2. Scope of Project

The scope of this project comprises the development of an AI-powered medical chatbot capable of evaluating chest X-ray pictures to help disease diagnosis and prescription recommendations, with an emphasis on detecting COVID-19 and pneumonia. The system is intended to provide speedy, reliable, and accessible preliminary diagnostic help to healthcare professionals, patients, and organizations, especially in resource-limited or distant places where direct medical consultation may not be available.

The research includes the installation of ResNet-50, a powerful deep learning architecture, to properly categorize chest X-ray pictures into normal, pneumonia, or COVID-19 categories. Once examined, the chatbot evaluates the results, communicates findings in a user-friendly conversational format, and makes initial prescription suggestions based on accepted medical criteria.

Additionally, the scope comprises user identification, picture upload interfaces, result presentation, and conversational advice to assure usability and accuracy. The technology is designed for integration into healthcare processes, giving scalable, efficient, and secure help while minimizing diagnostic delays and supporting medical decision-making across varied clinical situations.

1.3. Definitions

1.4. References

This Software Requirements Specification is developed in accordance with industry standards, regulatory guidelines, and best practices for medical software development. The following references provide the foundational framework for system requirements, design principles, and compliance criteria:

IEEE Standards:

- IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications - provides the structural framework and content guidelines for this SRS document
- IEEE 12207-2008, Systems and Software Engineering - Software Life Cycle Processes - defines the software development lifecycle standards applicable to medical software
- IEEE 14764-2006, Software Engineering - Software Life Cycle Processes - Maintenance - outlines maintenance requirements for healthcare systems

Healthcare Regulatory Guidelines:

- HIPAA (Health Insurance Portability and Accountability Act) Compliance Guidelines for Healthcare Software - establishes privacy and security requirements for protected health information
- FDA Guidelines for AI/ML-based Medical Devices - provides regulatory framework for artificial intelligence in medical applications
- 21 CFR Part 820, Quality System Regulation - defines quality system requirements for medical device manufacturers
- Medical Device Regulation (MDR) EU 2017/745 - European regulatory framework for medical devices including software

Medical Imaging Standards:

- DICOM (Digital Imaging and Communications in Medicine) Standard - defines protocols for medical image handling, storage, and transmission
- HL7 FHIR (Fast Healthcare Interoperability Resources) - establishes standards for healthcare information exchange

- IHE (Integrating the Healthcare Enterprise) Integration Profiles - provides implementation guidance for healthcare IT integration

Quality and Safety Standards:

- ISO 13485:2016, Medical Devices - Quality Management Systems - establishes quality management requirements for medical device development
- ISO 14971:2019, Medical Devices - Application of Risk Management to Medical Devices - defines risk management processes for medical software
- IEC 62304:2006, Medical Device Software - Software Life Cycle Processes - specifies software development lifecycle requirements for medical devices

1.5. Overview of Document

The next chapter, the Overall Description section, of this document gives an overview of the functionality of the product. It describes the informal requirements and is used to establish a context for the technical requirements specification in the next chapter.
The third chapter, Requirements Specification section, of this document is written primarily for the developers and describes in technical terms the details of the functionality of the product.
Both sections of the document describe the same software product in its entirety, but are intended for different audiences and thus use different language.

2.0. Overall Description :

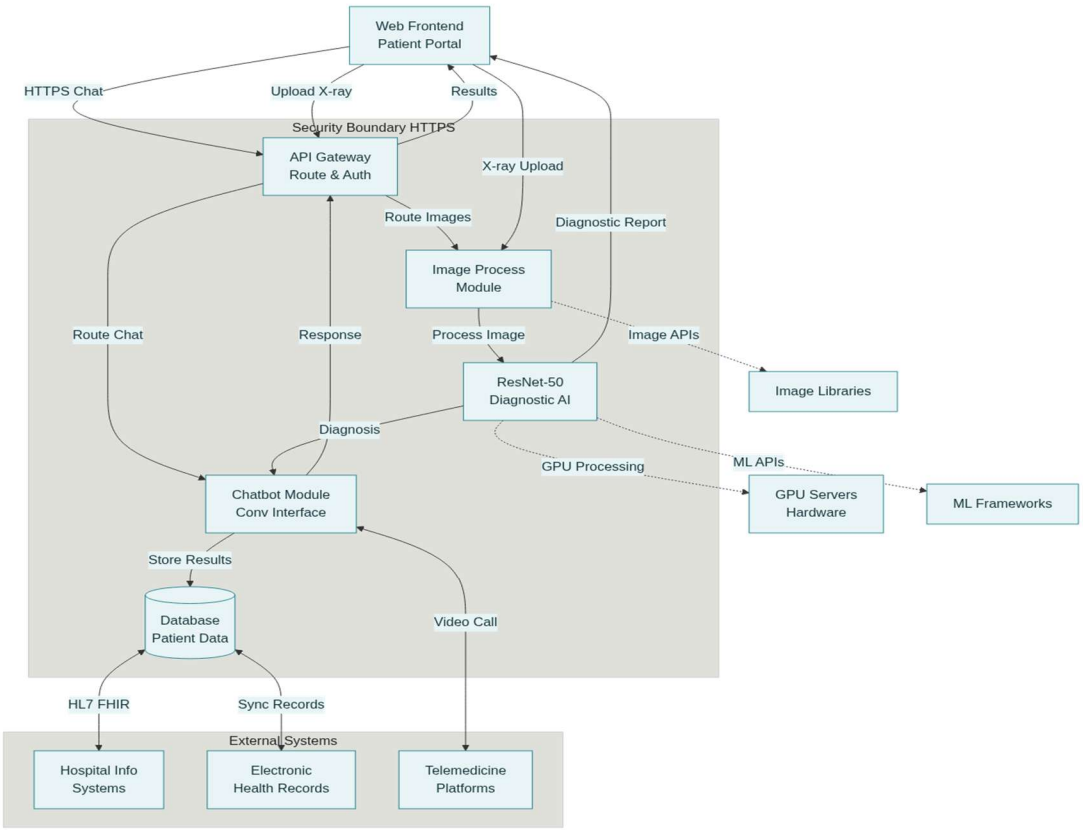
2.1. Product Perspective

The proposed AI-powered medical chatbot is an unique and self-contained software system that mixes deep learning–based image processing with an interactive conversational interface. While it can work as a standalone diagnostic support tool, it may also be included into larger healthcare management systems such as hospital information systems (HIS) or telemedicine platforms. In such

circumstances, the program will exchange data through defined APIs, enabling smooth integration with patient record systems and electronic health records (EHR).

The system largely relies on ResNet-50 for chest X-ray image categorization, which operates as the fundamental analytical module. The chatbot module is put on top of this diagnostic engine to communicate results effectively to consumers. The system supports the following interfaces and constraints:

- System Interfaces: APIs for integration with machine learning models.
- User Interfaces: Web front-end for patients and healthcare providers.
- Hardware Interfaces: Requires GPU-enabled servers for effective training and inference.
- Software Interfaces: Interfaces with image processing libraries, deep learning frameworks, and databases.
- Communication Interfaces: Secure data sharing via HTTPS protocols.
- Memory and Operations: Optimized to process huge medical pictures with minimal delay.
- Site Adaptation: Configurable for deployment in hospitals, diagnostic centers, or cloud platforms.



2.1.1 System Interfaces

The AI-powered medical chatbot system integrates with numerous external and internal systems to offer accurate disease diagnosis and prescription recommendations. The key system interfaces are outlined below:

- Kaggle Dataset Repository: The system extracts chest X-ray pictures from publicly available Kaggle datasets. This interface allows the software to retrieve labeled images for model training, validation, and testing. Data is obtained in standard formats (JPEG, PNG) and preprocessed for compliance with ResNet-50.

- Cloud Storage Services: Large datasets and processed results may be stored and accessible through cloud platforms to ensure scalability and efficient resource consumption.
- Notification Systems: Interfaces with email or SMS gateways to alert healthcare providers or users of diagnostic outcomes.

2.1.2 User Interfaces

The AI-powered medical chatbot provides an intuitive interface for both patients and healthcare providers. The primary user interface is web- or mobile-based, including a sleek dashboard for image submission, result visualization, and chatbot interaction. Users can contribute chest X-ray images in standard formats (JPEG, PNG) acquired from Kaggle or local files. Once uploaded, the system evaluates the photos using ResNet-50 and delivers diagnostic data in a conversational way.

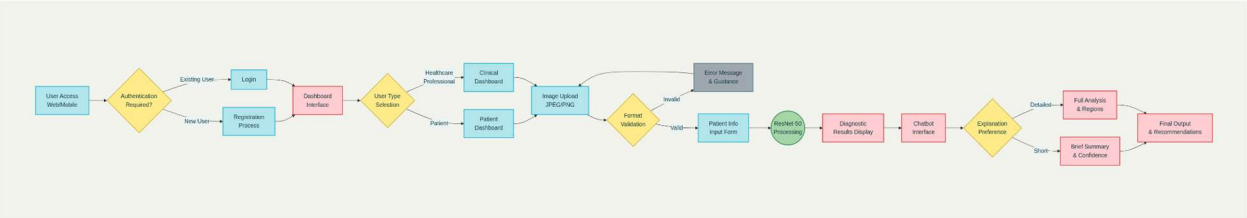
1. **Logical Characteristics:**

- Structured pages with well designated parts for image submission, patient information, and diagnostic output.
- Chat interface with dynamic discussion flow, choices, and buttons to guide users through the diagnostic procedure.
- Display of diagnostic results with confidence scores, graphical highlighting of impacted regions, and suggested tentative prescriptions.
- Option for quick or thorough explanations of outcomes, providing customisation based on user expertise.

2. **Optimization Guidelines:**

- Error notifications should be clear and actionable; e.g., “Invalid image format. Please provide JPG/PNG.”
- System should allow users with minimal training to conduct an X-ray upload and obtain findings within five minutes.

- Consistent layout and understandable font sizes to promote usability for clinicians and patients.



2.1.3 Hardware Interfaces

The AI-powered medical chatbot system interacts with numerous hardware components to enable quick processing, accurate picture interpretation, and smooth user engagement. The system is meant to run on servers or workstations equipped with GPU acceleration for deep learning inference and training. Specifically, NVIDIA GPUs with CUDA capability are advised to maximize ResNet-50 performance. CPU requirements include multi-core processors with support for modern instruction sets (e.g., AVX, SSE) to handle auxiliary activities like as data preparation, database searches, and user interface rendering.

Supported Devices and Interfaces:

- User Terminals: Desktop computers, laptops, tablets, or cellphones capable of accessing the web interface. These terminals interface with the system via normal web protocols (HTTPS).
- Image Acquisition Devices: Chest X-ray pictures originate from local uploads; the system supports common file formats (JPEG, PNG, DICOM).
- Storage Devices: Local SSDs or cloud storage for picture and model data.
- Network Interfaces: Ethernet or Wi-Fi connections with TCP/IP protocol for data transfer between the server and user terminals.

The system offers full-screen, responsive web interfaces on numerous device types and ensures optimal data transfer, low latency processing, and secure connections across all hardware components.

2.1.4 Software Interfaces

The AI-powered medical chatbot relies on numerous software components to perform successfully. The primary software interfaces are as follows:

Operating System

- Name: Windows
- Mnemonic: Win11
- Specification Number: OS-W11
- Version Number: 11 Pro
- Source: https://www.microsoft.com/windows

Windows 11 provides the platform for operating all backend services, including deep learning frameworks, databases, and the web server. It also enables GPU acceleration for ResNet-50 inference and training.

Deep Learning Framework

- Name: TensorFlow
- Mnemonic: TF
- Specification Number: TF-2.12
- Version Number: 2.12
- Source: https://www.tensorflow.org

TensorFlow is used to develop and train the ResNet-50 model for chest X-ray classification. Input images in JPEG, PNG, or DICOM format are processed, generating diagnostic results and confidence scores.

Database Management System

- Name: MySQL
- Mnemonic: MySQL
- Specification Number: DBMS-8.0
- Version Number: 8.0
- Source: https://www.mysql.com

MySQL contains user data, X-ray metadata, diagnostic results, and pharmaceutical recommendations. The interface allows SQL queries and RESTful API access for secure data management.

Front-End Web Server

- Name: React
- Mnemonic: REACT
- Specification Number: FE-18
- Version Number: 18.x
- Source: https://reactjs.org

React acts as the web front-end framework, delivering a responsive and interactive interface. It interfaces with backend APIs to upload photos, display diagnostic data, and facilitate chatbot interactions in real time.

These software interfaces together provide smooth integration, secure data management, and effective functioning of the chatbot system on Windows 11 with a React-based front end.

5.2.1.5 Communication Interfaces

The AI-powered medical chatbot system relies on numerous communication interfaces to enable secure, reliable, and efficient data transfer between the front-end, backend, and external systems.

- Front-End to Backend Communication: The React-based web interface talks with the backend via RESTful APIs over HTTPS to ensure encrypted data transmission. JSON is utilized as the standard messaging format for providing X-ray picture data, user information, and receiving diagnostic results and confidence scores.
- Notification Interfaces: Diagnostic findings or alarms can be communicated via email or SMS gateways. Standard protocols such as SMTP or REST-based messaging APIs are utilized to assure reliability.

These communication interfaces guarantee secure, efficient, and robust interactions among all system components, providing real-time diagnostics and user notifications.

2.1.6 Memory Constraints

The AI-powered medical chatbot system has special memory needs due to the processing of high-resolution chest X-ray pictures and the use of the ResNet-50 deep learning model.

Primary Memory (RAM):

The system requires at least 16 GB of RAM for standard operation for faster inference and simultaneous multi-user access.
RAM is utilized to load the ResNet-50 model, process picture data, execute convolution operations, and temporarily store intermediate results during inference.

Secondary Memory (Storage):

Local storage must have a minimum of 500 GB SSD, mostly for storing datasets, trained model weights, logs, and temporary user uploads.

For cloud deployment, storage must scale dynamically to support growing datasets and model checkpoints.

Database storage (MySQL) will manage structured data such as patient information, image metadata, diagnostic results, and prescription recommendations.

Memory management algorithms will ensure that huge image files and model computations do not exceed available resources, maintaining system stability and efficient processing.

2.1.7 Operations

The AI-powered medical chatbot supports numerous normal and special procedures to facilitate disease diagnosis and prescription suggestions from chest X-ray images.

- Modes of Operation: The system allows user-initiated operations, including image upload, chatbot interaction, and result retrieval. Healthcare professionals or patients can submit chest X-ray images, request analysis, and examine diagnostic outputs. The technology also offers automated processes such as batch processing of several photos for research or hospital reporting reasons.

- Periods of Operation: The system is designed for continuous availability. Interactive operations occur when consumers actively upload photographs or communicate with the chatbot. Unattended processes include model training, data preprocessing, database maintenance, and routine system monitoring, which can be planned during off-peak hours to enhance performance.

- Data Processing Support Functions: These include picture preprocessing (resizing, normalization), model inference using ResNet-50, creation of diagnostic results, and updating the database with metadata and recommendations.

- Backup and Recovery Operations: Automated backups of user data, X-ray pictures, and model weights are performed frequently. Recovery methods

are in place to restore system operation in case of hardware failures, data corruption, or unexpected downtime, ensuring reliability and data integrity.

2.1.8 Site Adaptation Requirements

The AI-powered medical chatbot system is designed to be adaptive for deployment across varied environments, including hospitals, diagnostic centers, clinics, and cloud-based platforms.

- Hardware Adaptation: The system must be interoperable with ordinary desktop or laptop PCs, GPU-enabled servers, and tablets or smartphones for user access. GPU parameters must allow ResNet-50 model inference, while RAM and storage needs should handle image processing and model storage as defined in the memory restrictions.

- Software Adaptation: The system is built to function on Windows 11 with a React-based web interface and TensorFlow backend. Any site deploying the system must assure installation of needed libraries, database systems (MySQL), and secure network setups.

- Network Adaptation: Sites must provide consistent internet or intranet access for communication between the frontend, backend, and cloud storage (if needed). HTTPS protocols and authentication techniques are essential for safe data transfer.

- Environmental Adaptation: The system must be setup to support variable X-ray image formats, including DICOM, JPEG, and PNG. It should also handle multi-user access while retaining performance and dependability.

These site adaptation requirements ensure that the system performs best across diverse deployment circumstances while fulfilling healthcare standards for dependability, security, and usability.

2.2 Product Functions

The AI-powered medical chatbot system must perform numerous essential activities to enable chest X-ray–based diagnosis and prescription recommendations, especially targeting COVID-19 and pneumonia detection.

- User Authentication and Management: The system must allow users to register, log in, and maintain profiles securely. Access control guarantees that only authorized workers or patients can submit photographs or access diagnostic results.

- Image Upload and Preprocessing: Users should be able to upload chest X-ray images in standard formats (JPEG, PNG, DICOM). The system will automatically preprocess photos, including resizing, normalization, and enhancement, to prepare them for model analysis.

- Diagnostic Analysis: The basic feature employs ResNet-50 to identify uploaded photos into categories: normal, pneumonia, or COVID-19. The technology will generate diagnostic results with confidence scores and identify afflicted locations on the X-ray where applicable.

- Chatbot Interaction: The system provides a conversational interface to explain diagnostic results, suggest preliminary prescriptions, and answer user queries based on standard clinical guidelines.

- Data Storage and Retrieval: All uploaded photos, diagnostic results, and medication advice are securely saved in a MySQL database, facilitating retrieval for future reference.

- Reporting and Notifications: The system can generate reports of diagnostic outcomes and alert users or healthcare providers via email or SMS when results are ready.

These functional requirements ensure that the system is accurate, user-friendly, secure, and fully capable of assisting medical decision-making.

2.3 User Characteristics

The intended users of the AI-powered medical chatbot system include healthcare professionals, such as doctors, radiologists, and clinical staff, as well as individuals seeking preliminary diagnostic support.

- Healthcare Professionals: These users often have advanced education in medicine, clinical training, and familiarity with evaluating medical images. They are familiar utilizing computer-based systems but require an intuitive interface for efficient interaction. Their experience supports demand for features like confidence scores, thorough image annotations, and advanced reporting functions.

- Patients: Patient users may have varied levels of educational background and technical ability. Most will have basic computer or smartphone literacy but may not understand intricate medical terminology or image interpretations. This informs the design of user-friendly interfaces, conversational chatbot explanations, simpler menus, and clear error messages.

- General Characteristics: All users are anticipated to interact with the system in numerous contexts, including hospital settings, home surroundings, or remote telemedicine sessions. Users may access the system intermittently or continuously, needing reliability, fast reaction times, and minimum training. The system must handle these changes to enable accurate, accessible, and safe use.

The development of the AI-powered medical chatbot system is subject to many constraints that limit design and implementation choices.

- Regulatory Policies: The system must comply with healthcare standards such as HIPAA or GDPR to ensure patient data privacy, secure storage, and authorized processing.

- Hardware Limitations: Deployment requires GPU-enabled servers for ResNet-50 model inference. Memory, storage, and processing speed must fulfill minimal standards to handle high-resolution chest X-ray images efficiently.

- Software and Application Interfaces: The system must interact with MySQL databases, React-based front-end, and optional HIS/EHR systems, placing limits on communication protocols and data formats.

- Parallel Operations: The system should handle simultaneous multi-user access without loss in performance, necessitating careful resource management.

- Audit and Control Functions: All user interactions, image uploads, and diagnostic results must be logged for traceability and auditing.

- dependability and Criticality: As a healthcare application, strong dependability is crucial; system outage or errors may effect medical decisions.

- Safety and Security Considerations: Encrypted data transfer, secure authentication, and role-based access are required to protect sensitive patient data and maintain operational safety.

2.5 Assumptions and Dependencies

The AI-powered medical chatbot system runs under numerous assumptions and dependencies that influence the requirements mentioned in this SRS.

- Operating System: It is presumed that the target deployment environment will have Windows 11 installed. Any change in the operating system may involve revisions to software interfaces, library dependencies, and performance optimization tactics.

- Dataset Availability: The system depends on publicly available chest X-ray datasets from Kaggle for training, validation, and testing of the ResNet-50 model. Changes in dataset format, labeling standards, or availability could alter preprocessing techniques, model correctness, and training procedures.

- Hardware Resources: The system assumes access to GPU-enabled servers for efficient model inference. Limited or unavailable GPU resources may require revisions in performance expectations or deployment tactics.

- Internet Connectivity: For cloud storage, updates, or notification services, dependable internet access is assumed. Interruptions may impede real-time diagnostic delivery and remote communication.

- Software Dependencies: The system depends on frameworks such as TensorFlow, React, and MySQL. Updates or incompatibility in these libraries may need modifications in software setup or integration methodologies.

These assumptions and dependencies are crucial for ensuring the system performs as planned and for planning any required adjustments.

2.6 Apportioning of Requirements

Certain requirements of the AI-powered medical chatbot system may be deferred for implementation in future versions due to resource restrictions, priority concerns, or technological dependencies.

- Advanced Diagnostic Features: While the present version concentrates on detecting COVID-19 and pneumonia from chest X-ray pictures, support for additional lung illnesses or multi-modal imaging (e.g., CT scans) may be introduced in forthcoming versions.

- Integration with External Systems: Full integration with multiple hospital information systems (HIS), electronic health records (EHR), and

telemedicine platforms may be incrementally established. Initially, the system may run as a standalone solution with optional API support, with deeper integration anticipated for future iterations.

- Enhanced User Personalization: Future versions may provide more complex chatbot customization options, such as multi-language support, voice interactions, and adaptive conversational responses based on user expertise.

- Performance Optimizations: Advanced GPU/TPU acceleration, distributed training, and real-time batch processing may be incorporated in upcoming versions to improve inference speed and scalability for large-scale deployments.

Deferring these needs guarantees the core system stays functional and dependable, while future versions can incrementally expand functionality, usability, and interoperability.

3. Specific Requirements

This section outlines the comprehensive functional and non-functional requirements of the AI-powered medical chatbot system, detailing what the system must accomplish and the quality standards it must achieve. These requirements give a clear plan for design, development, testing, and deployment.

3.0 Functional Requirements

User Registration and Authentication:
Users must be able to establish accounts, log in securely, and manage profiles. Role-based access control must discriminate between patients, healthcare workers, and administrators.

Chest X-ray Image Upload:
Users can upload photos in JPEG, PNG, or DICOM formats.
The system will evaluate image quality, format, and resolution before processing.

Image Preprocessing:
Automated preprocessing comprising scaling, normalization, and contrast enhancement.
Preprocessing assures compatibility with ResNet-50 for proper analysis.

Diagnostic Analysis:
ResNet-50 model must categorize pictures into normal, pneumonia, or COVID-19.
Confidence scores and region highlights must be presented with results.

Chatbot Interaction:
Conversational interface interprets diagnostic results and explains outcomes.
Chatbot delivers preliminary prescription suggestions based on medical criteria.

Data Storage and Retrieval:
MySQL database stores user data, image metadata, diagnostic results, and recommendations.
Users can access prior results securely.

Reporting & Notifications:
Automatic creation of diagnostic reports.
Notifications by email or SMS for result availability.

Administrative Functions:
System administrators can manage user accounts, monitor system performance, and modify settings.

3.1 External Interfaces

This subsection specifies all inputs to and outputs from the AI-powered medical chatbot system, complementing preceding interface definitions while adding content, format, and operational specifics.

Chest X-ray Image Input

- Purpose: To give the system with patient imaging data for diagnostic analysis.
- Source: Uploaded by user via online interface or downloaded from PACS (if integrated).
- Valid Range/Accuracy: High-resolution photos (minimum 1024×1024 pixels). JPEG, PNG, or DICOM formats only.
- Units: Pixel intensity values.
- Timing: On-demand by user or batch processing.
- Relationships: Preprocessed before passing to ResNet-50 model.
- Screen Format: Upload window with progress indicator and confirmation.
- Data Format: File-based (binary).
- End Message: “Upload Successful” or error messages describing format or size concerns.

Diagnostic Output

- Purpose: Provides classification of X-ray as normal, pneumonia, or COVID-19.
- Destination: Displayed on user interface; optionally kept in database or emailed/SMS sent.
- Accuracy: Includes confidence score (0–100%).
- Timing: Results generated shortly after model inference (usually <1 minute per image).
- Screen Format: Text and graphical highlights of affected regions.
- Data Format: JSON for API communication, visual overlay pictures for UI.
- End Message: “Diagnostic Complete” with opportunity to download report.

Chatbot Interaction

- Purpose: Explains results, provides prescriptions, and answers queries.
- Source/Destination: Web interface conversational module.
- Format: Text-based with menu buttons for navigation.

These external interfaces provide proper data interchange, fast feedback, and user-friendly interaction while retaining accuracy and reliability of diagnostic operations.

3.2 Functions

The AI-powered medical chatbot system shall implement the following core functions to process inputs and generate outputs accurately and reliably:

Input Validation:
The system shall check the format, resolution, and integrity of uploaded chest X-ray images (JPEG, PNG, DICOM).
The system shall reject photos that do not meet minimum resolution or contain faulty data and produce an error message identifying the issue.

Preprocessing:
The system shall resize, normalize, and enhance images to prepare them for ResNet-50 analysis.
The system shall log preprocessing settings for reproducibility.

Diagnostic Analysis:
The system shall classify images into normal, pneumonia, or COVID-19 categories using ResNet-50.
The system shall calculate confidence scores and identify affected locations on the X-ray.

Chatbot Interaction:
The system shall deliver diagnostic results using a conversational interface.
The system shall react to user queries regarding results, preliminary prescriptions, and next steps.

Error Handling and Recovery:

The system shall manage communication failures, storage faults, or computational overflows by notifying the user and logging the occurrence.
The system shall allow recovery and reprocessing of photos without data loss.

Output Generation:
The system shall generate JSON-formatted diagnostic results, graphical overlays, and downloadable reports.
Outputs shall correspond directly to approved inputs, preserving correctness and traceability.

3.3 Performance Requirements

The AI-powered medical chatbot system shall meet the following performance standards to ensure reliable and timely operation:

Static Requirements (Capacity):
The system shall handle at least 50 simultaneous users engaging with the chatbot and uploading chest X-ray images.
The system shall accommodate up to 10,000 stored X-ray pictures in the database, with scalability to increase storage dynamically in future expansions.
Each terminal (laptop, tablet, or smartphone) shall provide full-featured access to the web interface, including image upload, seeing results, and chatbot interactions.

Dynamic Requirements:
The system shall process at least 95% of single-image diagnostic requests within 60 seconds under typical load conditions.
The system shall handle batch processing of up to 100 pictures simultaneously, completing inference and result creation within 10 minutes.
The system shall maintain a response time of less than 10 second for chatbot query responses.
The system shall log all transactions, including uploads and outcomes, with no data loss.

These measurable performance parameters ensure that both real-time and batch processing operations are handled efficiently, maintaining system responsiveness and reliability for users.

3.4 Logical Database Requirements

The AI-powered medical chatbot system shall leverage a MySQL relational database to store and manage information required for disease diagnosis and prescription recommendations.

- Types of Information: The database shall include user profiles, patient demographics, uploaded chest X-ray images and metadata, diagnostic results including confidence scores, preliminary medication suggestions, system logs, and chatbot conversation histories.

- Frequency of Use: User login and image uploads occur often, but retrieval of prior diagnostic results and reports occurs intermittently. Chatbot conversations may occur numerous times per session for each user.

- Accessing Capabilities: Authorized users shall be able to query their own information, while healthcare professionals and administrators can access aggregated data, with role-based access control protecting data protection.

- Data Entities and Relationships: The database shall model entities including Users, Images, Diagnostics, Prescriptions, and Interactions, with defined relationships such as Users-to-Images (one-to-many) and Diagnostics-to-Prescriptions (one-to-one).

- Integrity Constraints: Each image and diagnostic result must be paired with a valid user ID. Constraints ensure uniqueness, proper formats, and referential integrity across tables.

- Data Retention: All patient-related data and diagnostic results will be preserved for at least 5 years, following applicable healthcare legislation, with secure deletion procedures as appropriate.

3.5 Design Constraints

The AI-powered medical chatbot system is subject to many design limitations that influence architecture, technology selection, and implementation approaches.

- Hardware Constraints: The system must run on GPU-enabled servers to efficiently conduct ResNet-50 inference and image preprocessing. Limited RAM, storage, or GPU availability can restrict the number of concurrent users or the size of photos that can be processed.

- Software Constraints: The system is required to operate on Windows 11 with a React front-end, TensorFlow backend, and MySQL database. All libraries and frameworks must be compatible with these software platforms, limiting the choice of programming languages, APIs, and third-party tools.

- Standards and Regulatory Constraints: The system must comply with healthcare data privacy and security regulations, such as HIPAA and GDPR. This imposes limits on encryption, data access control, audit logging, and data retention regulations.

- Interface Constraints: Integration with external systems, such as PACS or EHR, must adhere to DICOM and HL7 standards, constraining data formats and communication protocols.

- Operational Constraints: The system must support responsive real-time diagnostics and multi-user interactions without exceeding defined latency standards.

These design restrictions assure system stability, regulatory compliance, and compatibility with current hardware and software environments.

3.5.1 Standards Compliance

The AI-powered medical chatbot system shall conform with existing healthcare, software, and data management standards to assure interoperability, security, and regulatory adherence.

- Report Format: Diagnostic reports created by the system shall follow conventional medical report forms, including patient identities, image metadata, diagnostic results, confidence scores, and preliminary prescriptions. Reports shall be exportable in PDF and JSON formats for consistency across institutions.

- Data name: All database entities, picture files, and log entries shall utilize consistent name rules to ensure traceability and prevent conflicts. User IDs, image IDs, and session identifiers must be unique and follow alphanumeric coding requirements.

- Audit Tracing: The system shall retain full audit logs of all user operations, including image uploads, diagnostic analyses, chat exchanges, and administrative modifications. Each log entry shall record timestamp, user ID, action done, and before-and-after values when applicable. This provides responsibility, promotes regulatory compliance (e.g., HIPAA, GDPR), and facilitates troubleshooting.

- Accounting and Access Procedures: Access to sensitive patient data and system operations shall be role-based, with all access events logged. Any adjustments to user accounts or diagnostic results must be recorded to preserve an immutable trail for auditing reasons.

These standards compliance criteria guarantee regulatory conformance, secure data processing, and consistent reporting across healthcare contexts.

3.6 Non Functional Requirements

The AI-powered medical chatbot system will possess certain critical software attributes to assure performance, reliability, usability, and maintainability.

3.6.1 Reliability: The system shall offer consistent and accurate diagnostic results with 99% uptime for all core services. Errors in image processing or model inference shall be automatically logged and highlighted for review.
3.6.2 Availability: The system shall be accessible 24/7 for authorized users, with minimal downtime during maintenance or updates. Automated failover techniques shall be implemented to preserve service continuity.

3.6.3 Security: The system shall feature role-based access control, encrypted data transmission via HTTPS, and secure storage of photos and user data. All sensitive operations shall be logged for auditing purposes.

3.6.4 Maintainability: The system will be modular, with clearly defined frontend, backend, and database components to ease upgrades, bug fixes, and scalability. Code and configuration management shall follow version control best practices.

3.6.5 Portability: The system shall be deployable on Windows 11 with GPU support and adaptable for future migration to cloud-based platforms. Dependencies on specific hardware or libraries shall be avoided to ensure flexibility.

These software qualities jointly ensure the system's stability, safety, and usefulness in clinical and research situations.

3.7 Organizing the Specific Requirements

The exact requirements for the AI-powered medical chatbot system are vast and cover various functional and non-functional elements, including image processing, model inference, database management, user interaction, and security compliance. To enhance clarity, understanding, and traceability, the precise requirements in Section 3 are grouped into discrete categories and subcategories.

- Functional Requirements: These are classified according to system functionalities, including user authentication, image upload and preprocessing, diagnostic analysis, chatbot interaction, data storage and retrieval, reporting, and administrative functions. Each functional need is written as a "shall" phrase to ensure verifiability and traceability.
- Non-Functional Requirements: These include performance measurements, dependability, availability, security, maintainability, and portability. Non-functional criteria are classified based on measurable attributes such as response time, throughput, uptime, data quality, and auditability.

- External Interfaces: Requirements for user, hardware, software, and communication interfaces are grouped separately to clarify interactions with external systems.

- Database Requirements: Logical database requirements are stated independently to specify data entities, relationships, constraints, and retention policies.

This organized arrangement allows developers, testers, and stakeholders to rapidly discover, understand, and verify individual requirements, assuring proper implementation and system validation.

3.7.1 System Mode

The AI-powered medical chatbot system operates in numerous modes to support varied processes and user interactions. Organizing specific requirements by system

mode ensures that behavior, performance, and interfaces are explicitly defined for each operating scenario.

- Normal Mode: In this mode, the system performs standard diagnostic operations. Users can upload chest X-ray images, communicate with the chatbot, examine diagnostic results, and receive reports. The system is anticipated to handle concurrent user requests with regular processing times, maintain database integrity, and enable secure access to authorized users.

- Training Mode: This mode is intended for model development and improvement. Developers and data scientists can upload new datasets, retrain the ResNet-50 model, and evaluate model performance using test and validation sets. Processing in this mode may involve considerable computing load and extended inference periods.

- Emergency Mode: In emergency or high-priority circumstances, the system shall prioritize urgent diagnostic requests, minimizing processing latency for critical cases. Notifications to healthcare providers are hastened, and system tracking is enhanced to assure traceability.

Organizing requirements by system mode allows stakeholders to easily understand the system's behavior, performance objectives, and interface interactions under varied operational conditions.

3.7.2 User Class

The AI-powered medical chatbot system delivers separate functionality based on the class of user, ensuring that access, capabilities, and information display are appropriate for each position. Organizing requirements by user class facilitates unambiguous specification of permissions, operations, and system interactions.

- Patient Users: Patients can register, log in, and upload chest X-ray photos for analysis. They can check diagnostic results, download reports, and interact

with the chatbot for explanations and preliminary assistance. Patients have no access to administrative services or historical data of other users.

- Healthcare Professionals: This user class comprises doctors, radiologists, and clinical staff. They can do all patient activities and can also provide feedback to increase model accuracy and manage patient interactions via the chatbot.

Organizing the needs by user class ensures role-specific access control, functional clarity, and adherence to security and privacy rules.

3.7.3 Objects

The AI-powered medical chatbot system models real-world elements as objects within the software, allowing systematic representation of data and associated operations. Organizing requirements by object clarifies attributes, services, and interrelationships, boosting modularity and traceability.

- Patient Object: Represents individual patients.
- Attributes: Name, age, gender, contact data, medical history, uploaded X-ray photos.
- Functions: Upload photos, examine diagnostic results, obtain reports, interact with chatbot for explanations and guidance.
- X-ray Image Object: Represents submitted diagnostic images.
- Attributes: File format, resolution, timestamp, related patient ID.
- Functions: Preprocess for model input, save in database, link with diagnostic outcomes.
- Diagnostic Result Object: Represents analysis outputs.
- Attributes: Classification (Normal, Pneumonia, COVID-19), confidence score, annotations.
- Functions: Display to users, save in database, generate reports, trigger notifications.

Grouping entities as objects allows unambiguous description of data, activities, and interactions, facilitating maintainability and extension of the system.

3.7.4 Feature

The AI-powered medical chatbot system delivers numerous externally sought characteristics that deliver value to users through predetermined sequences of inputs and associated system answers. Organizing requirements by feature ensures clarity of functionality and user expectations.

Feature 1: Chest X-ray Image Upload

- Stimulus: User picks and uploads a chest X-ray image via the online interface.

- Response: System validates image format and resolution, preprocesses it, and stores it in the database. A confirmation notice ("Upload Successful") is displayed.

Feature 2: Diagnostic Analysis

- Stimulus: Preprocessed image is submitted for examination.

- Response: ResNet-50 model classifies the image as normal, pneumonia, or COVID-19, calculates confidence ratings, and indicates afflicted regions. Results are kept and displayed to the user.

Feature 3: Chatbot Interaction

- Stimulus: User inputs queries concerning diagnostic results or preliminary prescriptions.

- Response: Chatbot understands the inquiry, collects relevant information, and offers a response in a conversational fashion.

Feature 4: Report Generation and Notifications

- Stimulus: User wants a report or automated notifications are triggered.

- Response: System generates a PDF/JSON report, sends email/SMS notifications, and logs the transaction.

Organizing requirements by feature ensures each required service is thoroughly stated in terms of input stimuli, processing, and output responses, boosting usability and traceability

3.7.5 Stimulus

The AI-powered medical chatbot system can be arranged by stimuli, specifying its functions in response to certain events or inputs from users or external sources. This method ensures clarity in how the system operates under diverse operational conditions.

Stimulus 1: Image Upload Event

- Trigger: User submits a chest X-ray image.

- System Response: The system validates the file format and resolution, preprocesses the image, and stores it in the database. Errors such as unsupported formats or corrupted files prompt a warning with a descriptive error message.

Stimulus 2: Diagnostic Request Event

- Trigger: User sends a preprocessed image for examination.

- System Response: ResNet-50 performs classification, creates confidence ratings, highlights affected regions, stores results, and shows output to the user.

Stimulus 3: Chatbot Query Event

- Trigger: User sends a question regarding results or prescriptions.

- System Response: Chatbot understands the inquiry, gets relevant information, and offers a text-based response in a conversational fashion.

Stimulus 4: Notification Trigger

- Trigger: Diagnostic results become available.

- System Response: System automatically generates reports, sends email or SMS notifications, and logs the transaction for auditing.

Organizing functions by stimulus ensures the system reliably responds to specific inputs, maintains correctness, and increases user experience.

3.7.6 Response

The AI-powered medical chatbot system can be organized by answers, outlining all functions that assist generating a certain output or system reaction. This technique clarifies how inputs are handled to produce outputs.

Response 1: Diagnostic Result Generation

- Function: After a chest X-ray is uploaded and preprocessed, the system shall categorize the image using ResNet-50, calculate confidence ratings, and highlight afflicted regions.

- Supporting Functions: Image validation, preprocessing, model inference, result storage, and logging.

Response 2: Chatbot Reply Generation

- Function: When a user presents a query regarding diagnostic results or preliminary prescriptions, the system creates a conversational reply.

- Supporting Functions: Query interpretation, data retrieval from diagnostic results, response formatting, and distribution through the web interface.

Response 3: Report Generation

- Function: Upon completion of analysis, the system generates a downloadable report in PDF or JSON format.

- Supporting Functions: Compilation of diagnostic results, integration of patient metadata, formatting, and storage.

Response 4: Notifications

- Function: Notify users or healthcare professionals via email or SMS when results are ready.

- Supporting Functions: Trigger detection, message formatting, delivery scheduling, and logging.

Organizing by response guarantees that each output is well-supported, traceable, and consistently produced to fulfill user and system needs.

3.7.7 Functional Hierarchy

The AI-powered medical chatbot system's total functionality can be arranged into a functional hierarchy, grouping functions by common inputs, outputs, or internal

data access. This structure promotes clarity, traceability, and understanding of the system's activities.

Level 1: User Interaction Functions

- Input: User registration, login credentials, and X-ray image uploads.
- Output: Acknowledgments, error warnings, and chatbot responses.
- Supporting Subfunctions: Input validation, session management, and user authentication.

Level 2: Diagnostic Processing Functions

- Input: Preprocessed chest X-ray pictures.
- Output: Classification findings (normal, pneumonia, COVID-19), confidence ratings, and annotated photos.
- Supporting Subfunctions: Image preprocessing, ResNet-50 inference, result logging, and error handling.

Level 3: Data Management Functions

- Input: User data, photographs, diagnostic results, and system logs.
- Output: Data retrieval for reports and analytics.
- Supporting Subfunctions: Database access, data integrity checks, backup, and retention.

Level 4: Reporting and Notification Functions

- Input: Diagnostic results and user preferences.
- Output: PDF/JSON reports, email/SMS notifications.
- Supporting Subfunctions: Report formatting, message scheduling, and delivery tracking.