MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

Software Requirements and Specifications and Design for the Case Study on

Skin Disease Detection Using Deep Learning

SUBMITTED TO

Prof. Srikanth Prabhu
Professor
Department of Computer Science and Engineering

Manipal Institute of Technology

By

Pranamya G Kulal (CSE A - 220905018)

Ranjith G Rao (CSE A - 220905020)

Uday Ojha (CSE A - 220905117)

Jalla Varshith (CSE A - 220905148)

TABLE OF CONTENTS

1) CASE STUDY

Skin Disease Detection Web Application to be developed for automating the process of early detection and diagnosis of skin diseases. The website will utilize a deep learning model to analyze images of skin conditions submitted by users. It will assist dermatologists by providing an initial diagnosis, which helps in efficient patient care and resource management.

A healthcare organization aims to provide a website where users can upload images of skin conditions. These images are analyzed by a pre-trained deep learning model, which identifies possible skin diseases such as eczema, psoriasis, or skin cancer. Based on the model's analysis, the website provides the user with a preliminary diagnosis and confidence level.

Doctors associated with the healthcare organization can review these results and provide further consultation if necessary. The system will allow patients to store their medical history and track their skin conditions over time, facilitating better management and earlier detection of deteriorating conditions.

The system administrator will manage the application's data, such as updating disease databases, ensuring privacy, and scheduling maintenance. The administrator can also modify the AI model's configuration based on new training data, improving the system's efficiency and accuracy.

Software Requirements Specification Document

for

Skin Disease Detection Using Deep Learning

By

Pranamya G Kulal
Ranjith G Rao
Uday Ojha
Jalla Varshith

Table of Contents

1. Introduction

1.1 Purpose

This Software Requirements Specification (SRS) describes the functionality and performance expectations of the "Skin Disease Detection Using Deep Learning" system. The purpose of this project is to develop an automated tool that assists medical professionals and individuals in diagnosing skin diseases through image analysis powered by deep learning algorithms. This system aims to provide accurate, efficient, and accessible solutions for early detection and classification of skin diseases, ultimately improving patient outcomes.

1.2 Scope

This document details the complete functionality of the Skin Disease Detection System. It will cover the system's ability to classify various types of skin diseases from dermoscopic images using convolutional neural networks (CNNs). The SRS includes use-case models, system features, design constraints, assumptions, dependencies, and statistical reporting capabilities. The system will primarily target dermatologists, general physicians, and individuals looking for preliminary insights into potential skin conditions.

1.3 Definitions, acronyms and abbreviations

CNN: Convolutional Neural Network
AI: Artificial Intelligence
ML: Machine Learning

1.4 References

- Research papers on deep learning models for medical image classification.

1.5 Overview

The remainder of this document is divided into several sections. The "Overall Description" provides a high-level perspective on system capabilities and user interactions. "External Interface Requirements" specify the system's hardware, software, and communication needs. The "System Features" section details the functionalities of the detection system. Finally, "Nonfunctional Requirements" define performance and usability criteria essential for the system's effectiveness.

2. Overall Description

2.1 Product Perspective

The **Skin Disease Detection Using Deep Learning** system is designed to assist doctors and patients in diagnosing skin diseases by analyzing images of skin conditions through a deep learning model. The system acts as a diagnostic aid, providing a preliminary assessment to users based on image inputs. This software serves as an auxiliary tool within the healthcare domain, aimed at improving early detection and diagnosis accuracy. It will interact with a secure database to store patient information, images, diagnostic results, and user feedback. Cloud servers may also be used to facilitate high-performance computations and model processing.

2.2 Product Functions

The core functions of the **Skin Disease Detection System** are:

- **Image Upload**: Allows doctors and patients to upload images of skin conditions for analysis.
- **Image Preprocessing**: Automatically resizes, normalizes, and enhances images to ensure they are optimized for model input.
- **Disease Detection**: Uses a deep learning model to classify the skin disease and provides a confidence score based on the analysis.
- **Report Generation**: Generates a diagnostic report that includes the identified disease, confidence level, and recommendations for further action.
- **Feedback Collection**: Allows doctors to provide feedback on the diagnosis accuracy, which can be used to improve the model in future iterations.
- **Data Management**: Stores and manages user data, including images, diagnostic results, and feedback, in a secure database.

2.3 User Characteristics

The primary users of the **Skin Disease Detection System** are:

- **Doctors**: Healthcare professionals who can use the system to aid in diagnosing skin diseases, especially in cases where immediate access to dermatologists is limited.
- **Patients**: Individuals who may upload images of their skin conditions and view preliminary diagnostic results, especially in regions where direct access to dermatology services is not readily available.
- **Researchers**: Medical and data science researchers who can analyze feedback data, review system outputs, and provide input for improving the deep learning model's accuracy and efficacy.

Users are expected to have basic computer literacy. Doctors and researchers should have some understanding of dermatology or deep learning, respectively, to interpret the results effectively.

2.4 Constraints

The **Skin Disease Detection System** operates under several constraints:

- **Data Privacy Regulations**: Must comply with healthcare data protection laws, such as GDPR or HIPAA, to ensure that all patient data is securely stored and handled.
- **Image Quality**: Requires high-quality, high-resolution images for accurate disease detection. Low-quality images may affect the model’s accuracy.
- **Model Accuracy**: The deep learning model has a certain error margin and may not always achieve 100% accuracy, requiring validation by medical professionals.
- **Device Compatibility**: The system must be compatible across various devices, such as desktops, tablets, and smartphones, as users may access the system from different platforms.
- **Processing Power**: Complex deep learning computations may require high processing power, which could lead to longer processing times or necessitate cloud-based computational resources.

2.5 Assumptions and Dependencies

- **Image Format**: It is assumed that users will upload images in a supported format (e.g., JPEG, PNG).
- **Internet Connection**: The system relies on a stable internet connection for image upload, deep learning processing (especially if cloud-based), and report generation.
- **Regular Model Updates**: Based on new data and medical feedback, the model will require periodic updates to improve accuracy.
- **User Cooperation**: Users are expected to provide accurate information and feedback, especially from doctors, to help improve the system's functionality.
- **Training Data Quality**: The system’s performance heavily depends on the quality and diversity of the dermatology dataset used for training the deep learning model.

3. Specific Requirements

3.1 External Interfaces

The **Skin Disease Detection System** will interface with the following external components:

- **User Interface (UI)**: A web or mobile interface for doctors, patients, and researchers to interact with the system. The UI will allow users to upload images, view diagnostic results, and provide feedback.

    - **Input Requirements**: Upload buttons for image input, feedback forms, and login credentials.
    - **Output Requirements**: Display diagnostic reports, confidence scores, and feedback confirmation.
- **Database Interface**: Interface for storing and retrieving images, diagnostic results, user feedback, and model performance data.
    - **Input Requirements**: APIs for storing images, user data, and diagnostic results.
    - **Output Requirements**: APIs for retrieving historical data, user profiles, and reports.
- **Deep Learning Model Interface**: Interface for the image processing model that handles image analysis and returns diagnostic predictions.
    - **Input Requirements**: Preprocessed image data.
    - **Output Requirements**: Classification result with a confidence score.

3.2 Functions

The system must perform the following functions:

- **User Authentication**: Allow users to log in securely using unique credentials to access the platform.
- **Image Upload and Validation**: Enable users to upload images. Validate images for quality and format before processing.
- **Image Preprocessing**: Automatically adjust images (resizing, normalization, etc.) to ensure compatibility with the model.
- **Disease Detection and Prediction**: Use a deep learning model to analyze images and classify skin diseases, providing a confidence score with each result.
- **Report Generation**: Generate a diagnostic report that includes the detected disease, confidence level, and suggested next steps.
- **Feedback Collection**: Allow doctors to submit feedback on the accuracy of diagnoses, which will be stored for future improvements.
- **Data Management**: Securely manage data, including user profiles, images, diagnostic results, and feedback in a database.

3.3 Performance Requirements

The system's performance should meet the following requirements:

- **Response Time**: The system should preprocess images and provide diagnostic results within 10 seconds for local processing or 30 seconds for cloud-based processing.
- **Throughput**: The system should handle simultaneous requests from multiple users, supporting at least 50 concurrent users without performance degradation.
- **Availability**: The system should be available 24/7 with a minimum uptime of 99.5% to support continuous access.

- **Accuracy**: The model should have a classification accuracy of at least 85%, with continuous improvement expected as feedback data is incorporated.
- **Scalability**: The system should be able to scale to handle increased load, both in terms of user requests and data storage, as more users and data are added.

3.4 Logical Database Requirements

The system's database will store and manage the following information:

- **User Data**: User profiles, roles (doctor, patient, researcher), login credentials, and usage history.
- **Image Data**: Uploaded images along with metadata, such as upload timestamp, quality indicators, and linked user ID.
- **Diagnostic Results**: The classification results, confidence scores, and timestamps associated with each image processed.
- **Feedback Data**: Feedback from doctors about the accuracy of the model's diagnosis.
- **System Logs**: Logs of system activities, errors, and performance metrics for monitoring and troubleshooting.

3.5 Design Constraints

The system's design is subject to the following constraints:

- **Data Privacy and Security**: Must comply with data privacy laws such as HIPAA and GDPR, ensuring secure storage, processing, and handling of patient data.
- **Image Quality Standards**: The system must enforce minimum image resolution and quality standards to ensure model accuracy.
- **Platform Compatibility**: The system should be accessible on both web and mobile platforms, supporting a wide range of devices and browsers.
- **Resource Constraints**: Depending on the processing power, local device processing may be limited, and some tasks may need to be offloaded to a cloud-based server.

3.6 Software System Quality Attributes

The system will prioritize the following quality attributes:

- **Reliability**: The system should provide consistent performance with minimal downtime.
- **Usability**: A user-friendly interface for easy image upload, report viewing, and feedback submission.
- **Security**: Strong data protection mechanisms including encryption, secure authentication, and access control.
- **Maintainability**: Modular design to allow easy updates to the deep learning model, UI, and database structure.

- **Scalability**: Ability to scale vertically and horizontally to handle growing data volumes and user demands.
- **Accuracy and Precision**: High classification accuracy for reliable diagnostics, with confidence scores to assist users in decision-making.

3.7 Object-Oriented Models

The object-oriented design models for the **Skin Disease Detection System** include:

- **Class Diagram**: Defines the main classes such as `User`, `Image`, `DiagnosticReport`, `Feedback`, and `DeepLearningModel`.
  - **User Class**: Includes attributes like `userID`, `role`, `name`, and `email`, with methods for login, upload image, and view report.
  - **Image Class**: Includes attributes like `imageID`, `resolution`, `format`, and `timestamp`, with methods for validation and preprocessing.
  - **DiagnosticReport Class**: Includes attributes like `reportID`, `diseaseType`, `confidenceScore`, and `timestamp`, with methods to generate reports.
  - **Feedback Class**: Includes attributes like `feedbackID`, `doctorID`, `diagnosticResult`, and `comment`, with methods to record and view feedback.
  - **DeepLearningModel Class**: Contains methods for `classifyImage` and `trainModel`, storing model parameters and accuracy metrics.
- **Sequence Diagrams**: Demonstrates workflows such as image upload, preprocessing, diagnosis, and report generation (as previously outlined).
- **Use Case Diagrams**: Depict use cases such as “Upload Image”, “View Diagnosis”, “Submit Feedback”, and “Generate Report” for various actors (Doctor, Patient, System).

4. Functional Requirements

R1: Authenticate User

- **Input**: Username and password entered by the user.
- **Process**: System verifies the credentials and checks the user role (e.g., patient, doctor).
- **Output**: Access to system features based on the user role, or an error message if authentication fails.

R2: Upload and Validate Image

- **Input**: Skin condition image uploaded by the user (accepted formats: JPEG, PNG).
- **Process**: System validates image quality, size, and resolution to ensure it meets the requirements for analysis.
- **Output**: Confirmation of a successful upload, or an error message if the image does not meet standards.

R3: Preprocess Image

- **Input**: Uploaded image that meets the required quality standards.
- **Process**: Image is processed (e.g., resized, normalized) to standardize it for the deep learning model.
- **Output**: Preprocessed image ready for disease detection analysis.

R4: Detect Disease

- **Input**: Preprocessed image.
- **Process**: Deep learning model analyzes the image, runs classification algorithms, and generates a prediction with a confidence score.
- **Output**: Predicted skin disease with a confidence level (e.g., 90% confidence in diagnosis).

5. Non Functional Requirements-

1. Performance

- The deep learning model should process uploaded images and generate a preliminary diagnosis within 10 seconds.
- The system should handle up to 1,000 concurrent users without performance degradation.

2. Scalability

- The system must be scalable to accommodate future growth, supporting up to 10,000 concurrent users.
- It should be capable of integrating with new deep learning models and additional features without requiring significant re-engineering.

3. Security

- All patient data, including images, medical history, and diagnoses, must be encrypted both at rest and in transit (using AES-256 encryption for data storage and TLS 1.3 for data transmission).

4.Usability

- The app interface should be intuitive and simple, designed to be used by non-technical users without extensive guidance.

5. Maintainability

- The system should be modular, allowing for easy updates and maintenance of individual components (e.g., the AI model, user interface).

- The codebase should follow standard coding conventions and include comprehensive documentation for ease of maintenance.
- Regular software updates should be scheduled quarterly to ensure the app is up-to-date with the latest technology.

6. Reliability

- The system should provide consistent diagnosis accuracy, with a confidence level of at least 85% based on the AI model's performance.
- In case of a system error or failure, the application should have automated recovery procedures to restore normal operations within 5 minutes.

7. Compatibility

- The application must be compatible with iOS (13.0 and above) and Android (9.0 and above).
- The app should function smoothly across different devices with varying screen sizes and resolutions.

8.Data Privacy

- Users should have full control over their personal data and the ability to delete their information from the system at any time.
- The app must follow data minimization principles, collecting only necessary information for diagnosis and care.

9. Legal Compliance

- The application must comply with relevant data protection regulations like GDPR for users in Europe and HIPAA for U.S. healthcare services.

10. Availability

- The system should ensure high availability with a cloud-based infrastructure, allowing seamless 24/7 access for both users and healthcare professionals.

These non-functional requirements help ensure that the SDDMA will not only function properly but also deliver a secure, reliable, and scalable experience for users and healthcare professionals.

3) Use Case Description:

Use Case 1: Upload Image

- **Use Case ID**: UC1
- **Actor**: User
- **Description**: User uploads an image of the skin area to be analyzed.
- **Preconditions**: User is logged into the system (if required).
- **Postconditions**: The image is successfully stored and ready for processing.
- **Main Flow**:
    1. User selects the image upload option.
    2. User browses their files and selects an image to upload.
    3. System accepts the image file and stores it.
    4. System notifies the user that the image has been successfully uploaded.

Use Case 2: Process Image

- **Use Case ID**: UC2
- **Actor**: System
- **Description**: The system processes the uploaded image for disease detection.
- **Preconditions**: Image has been uploaded by the user.
- **Postconditions**: Image is processed and prepared for analysis.
- **Main Flow**:
    1. System retrieves the uploaded image.
    2. System performs pre-processing steps on the image (e.g., resizing, normalization).
    3. System stores the processed image data, making it ready for prediction.

Use Case 3: Predict Disease

- **Use Case ID**: UC3
- **Actor**: System
- **Description**: The system uses a trained deep learning model to predict the type of skin disease based on the processed image.
- **Preconditions**: Image has been processed and is ready for analysis.
- **Postconditions**: The system generates a disease prediction.

- **Main Flow**:
  1. System inputs the processed image into the trained model.
  2. Model analyzes the image and outputs a predicted disease classification.
  3. System saves the prediction results for the user to view.

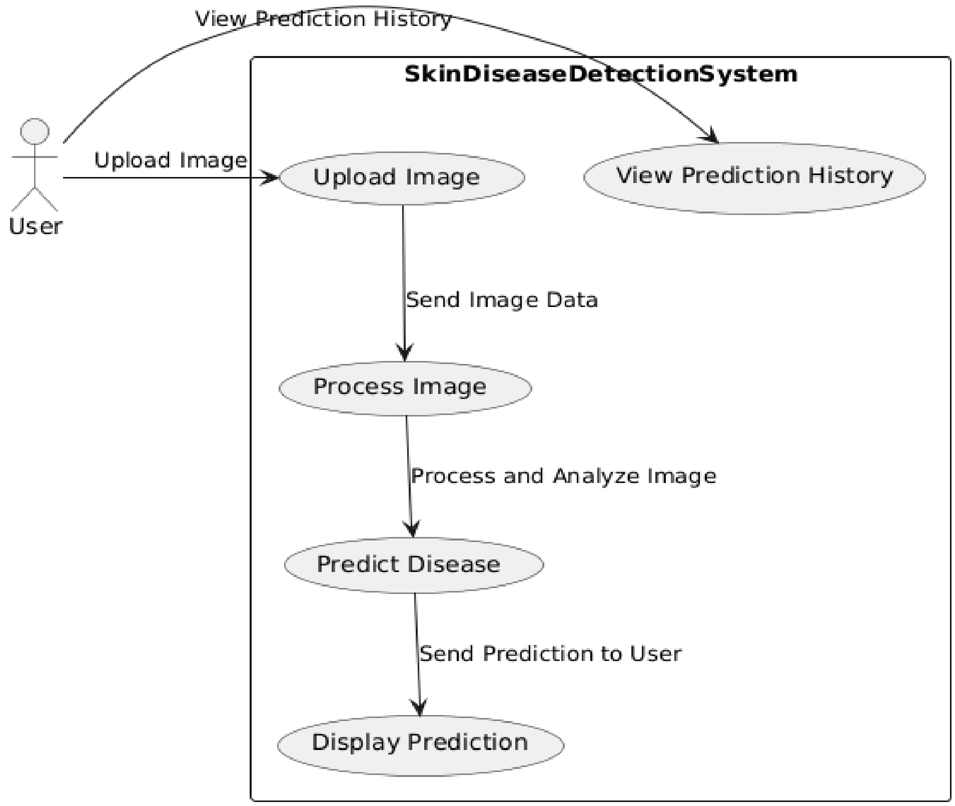
Use Case 4: Display Prediction

- **Use Case ID**: UC4
- **Actor**: User, System
- **Description**: System displays the prediction results to the user.
- **Preconditions**: Prediction has been generated.
- **Postconditions**: Prediction results are visible to the user.
- **Main Flow**:
  1. System retrieves the prediction results.
  2. System displays the disease type or probability score on the user’s screen.
  3. User views the prediction.

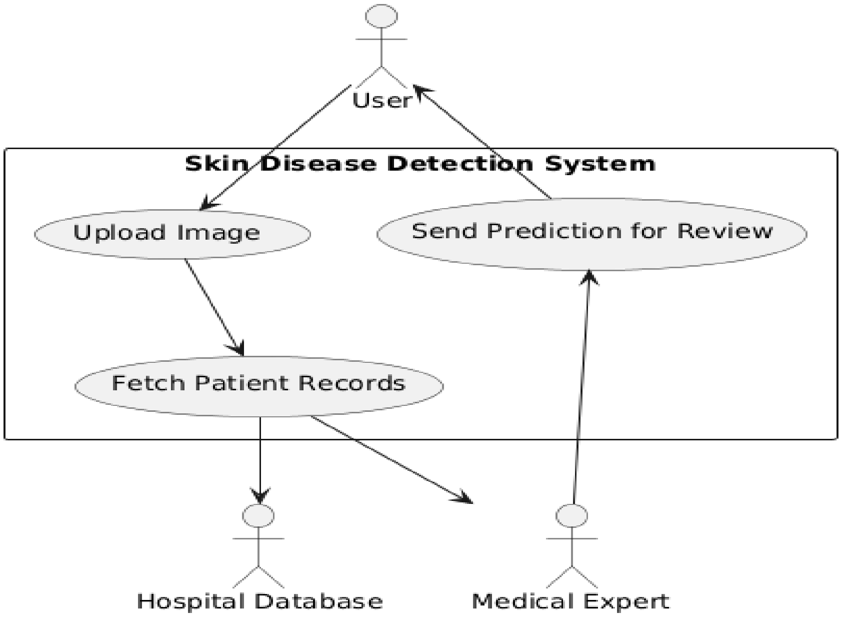
Use Case 5: View Prediction History

- **Use Case ID**: UC5
- **Actor**: User
- **Description**: User can view a history of previous predictions they have requested.
- **Preconditions**: User has accessed the prediction system in the past.
- **Postconditions**: Past predictions are displayed.
- **Main Flow**:
  1. User selects the option to view prediction history.
  2. System retrieves and displays the list of past predictions associated with the user.
  3. User selects an entry to view details of a previous prediction (optional).

4) Use Case Diagram:

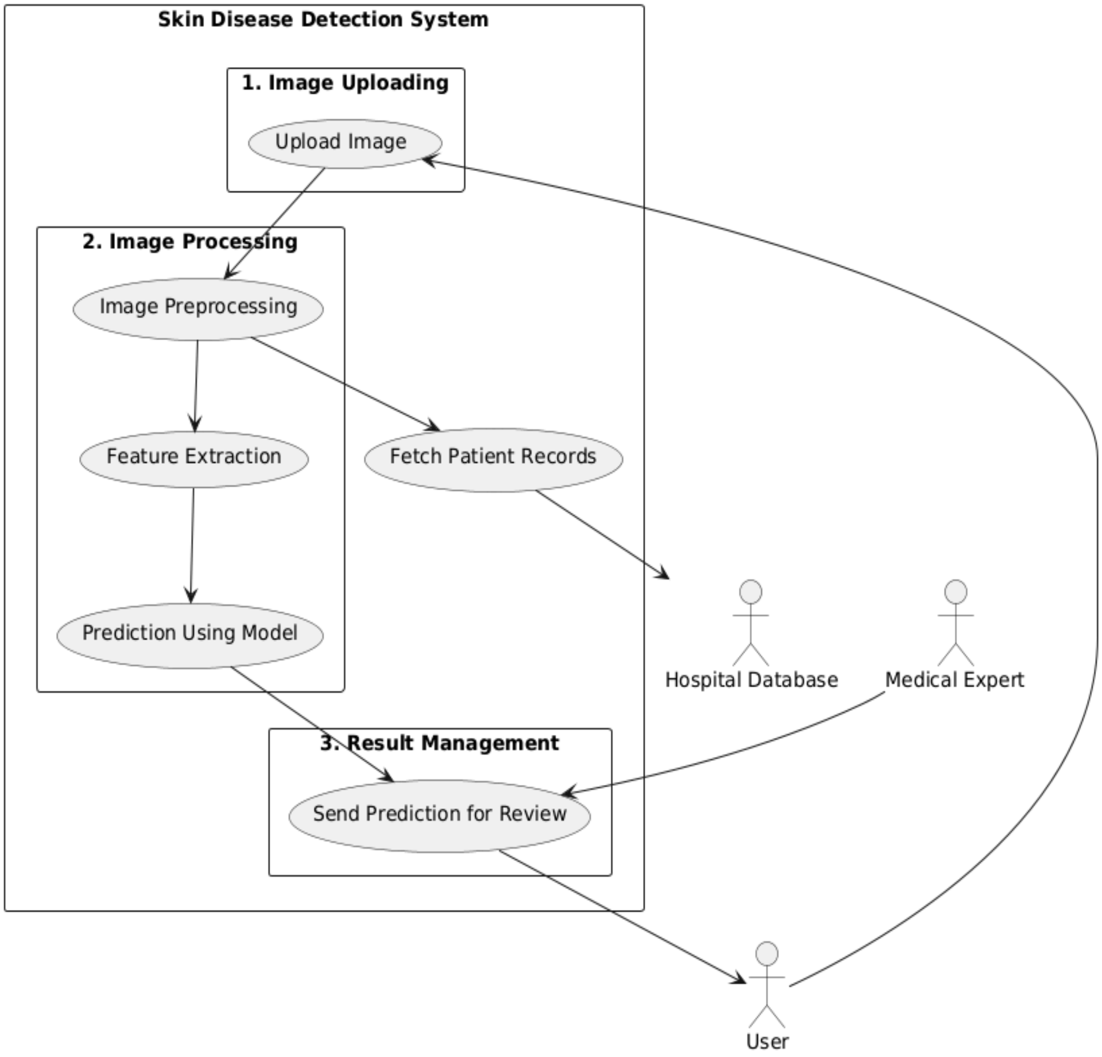


5) Context Diagram:

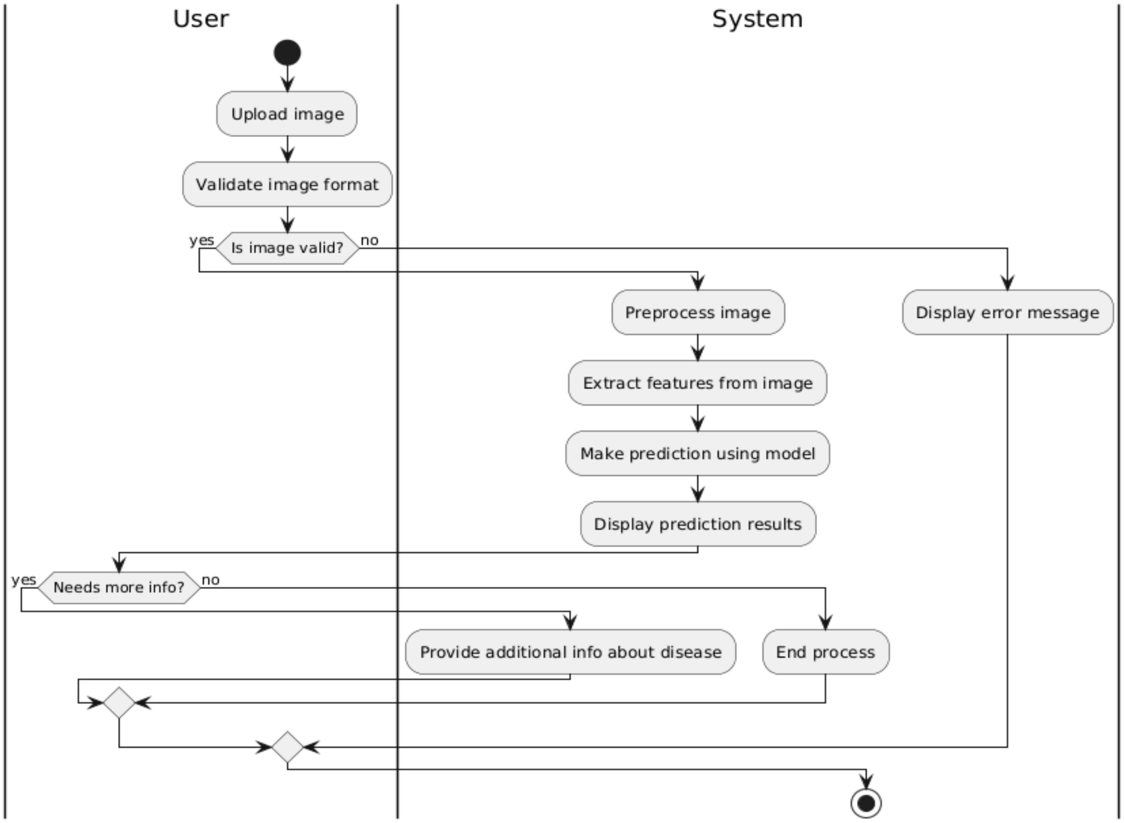


6) Data Flow Diagram:

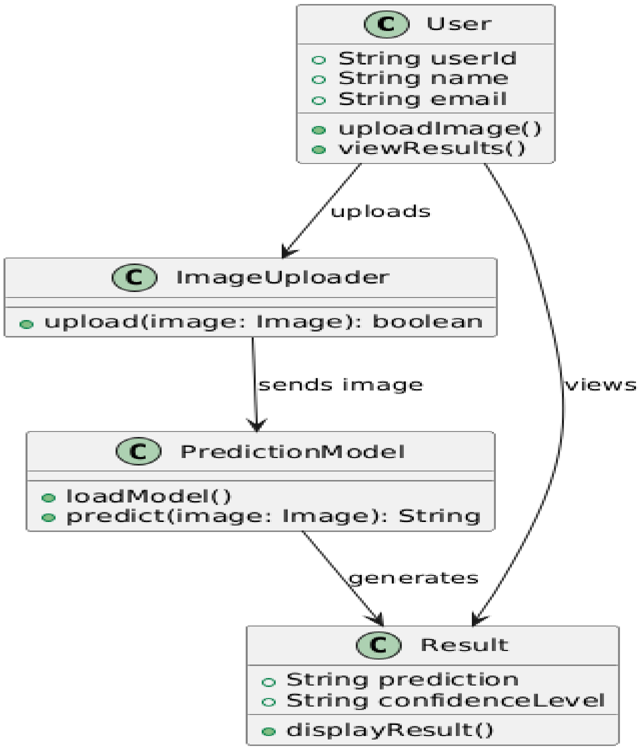
DFD Level 1:



7) Activity Diagram:

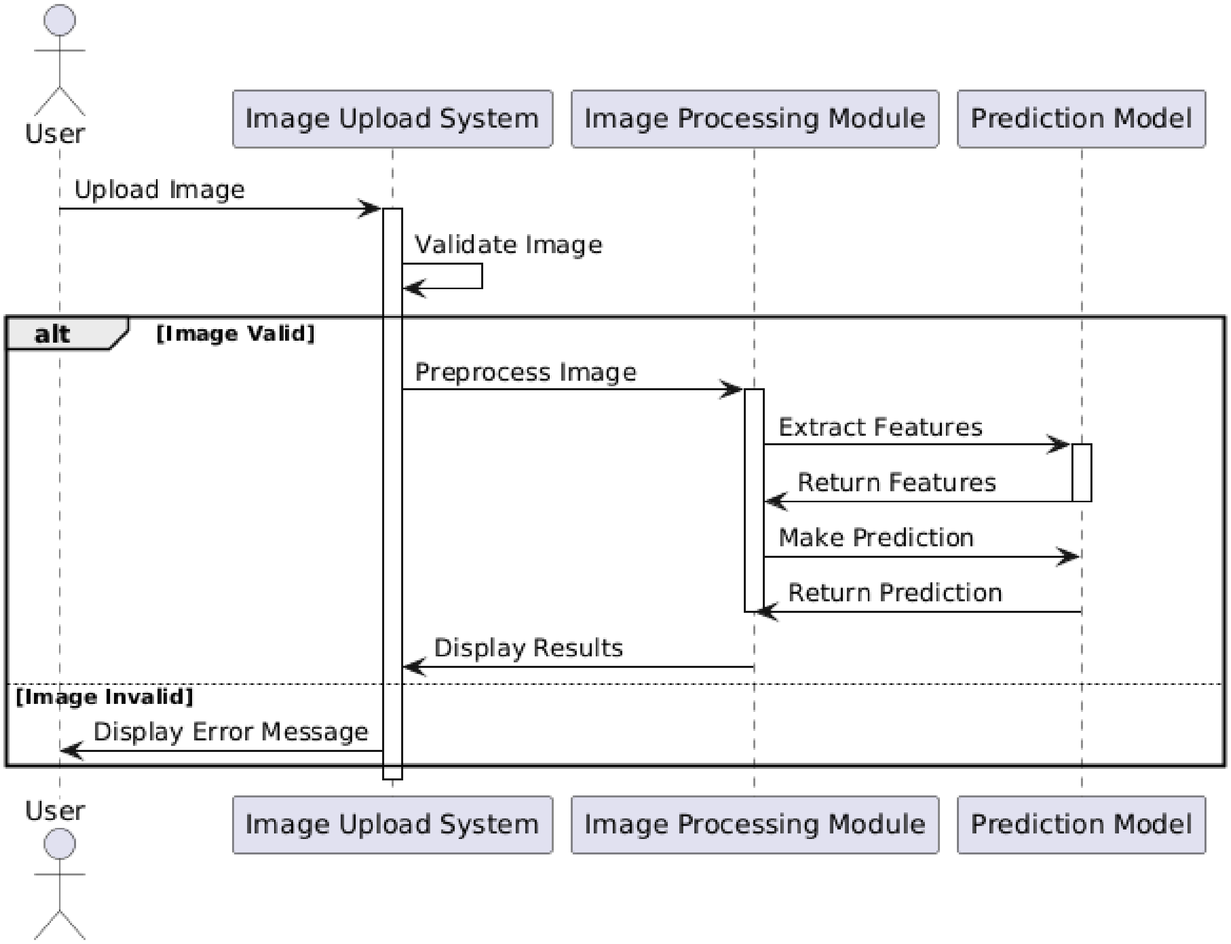


8) Class Diagram:

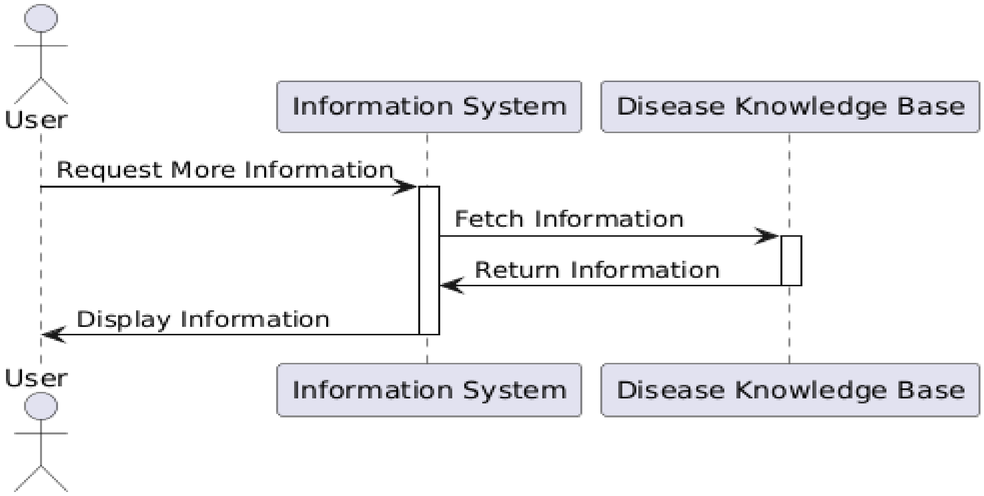


9) Sequence Diagram:

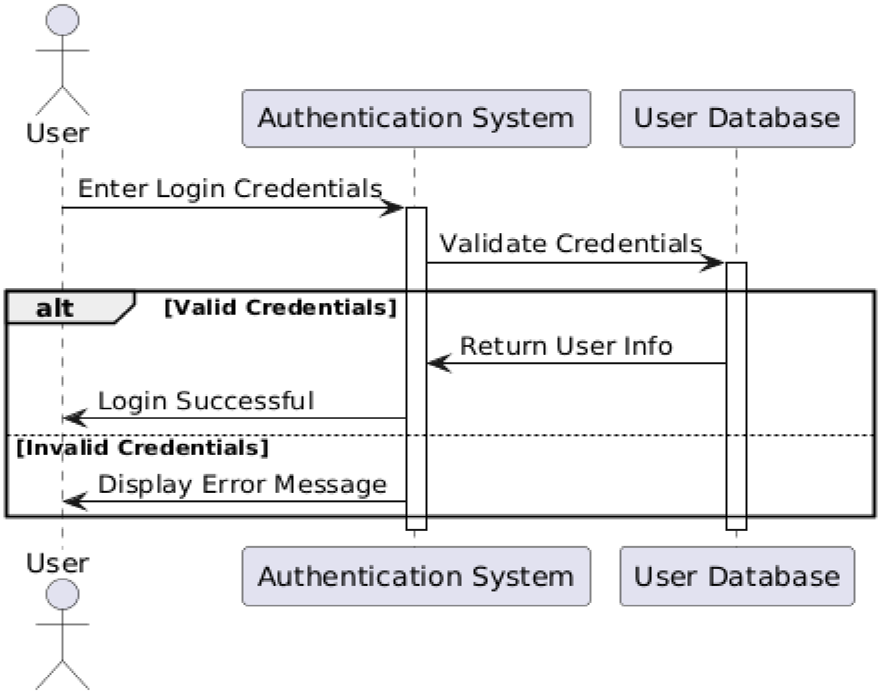
1. Image Upload and Prediction Sequence Diagram:



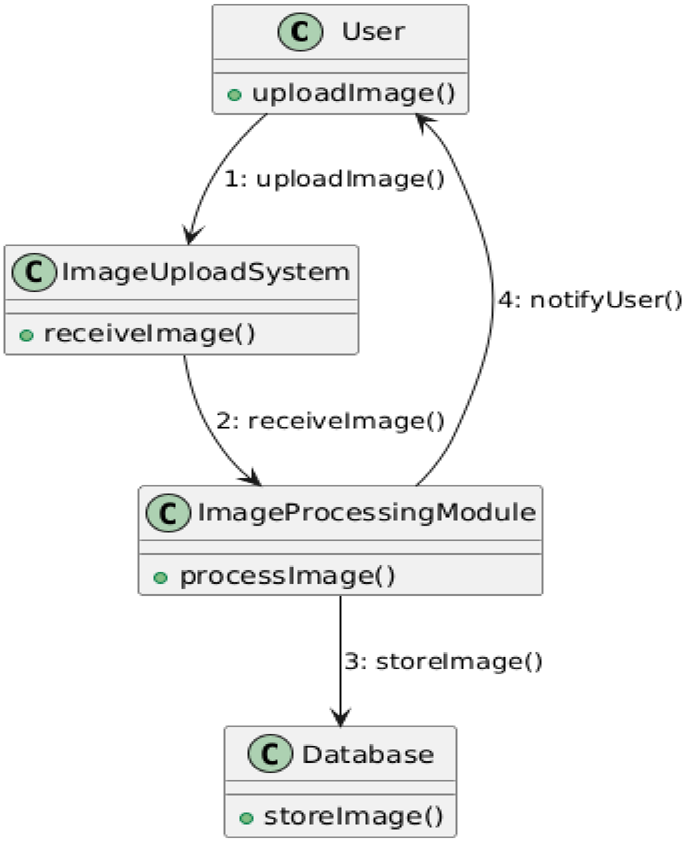
2. User Information Request Sequence Diagram:



3. User Login Sequence Diagram:



10) Collaboration Diagram;



11) Future Enhancements:

Only few diseases and few images in a dataset are used to identify the disease. In future many skin diseases and not only disease based on skin every type of disease and wound can be implemented in the system. This process can be extended to make this model a standard procedure for preliminary skin disease diagnosis method as it will reduce the treatment and diagnosis time.

12) References:

1. Fundamentals of Software Engineering by Rajib Mall
2. Convolutional Neural Networks (CNNs) for Skin Lesion Classification
3. Skin Disease Detection and Classification Using Deep Learning