Patient's Diabetic Foot Wound Analysis through Segmentation and Diabetic Wound Classification

Team Name		
Index Number	Registration Number	Name with Initials
4437	ASP/18/19/013	K. A. P. Dhananjaya
4436	ASP/18/19/012	H. L. A. R. L. Deepasara
4420	ASP/18/19/089	M. A. S. Dimalshan
4446	ASP/18/19/060	G. L. N. Sandarekha
4423	ASP/18/19/095	I. N. Jayasekara

2023/08/17

Abstract

The Deep Learning based Diabetic Foot Wound Classifier and Screening Tool is designed to classify diabetic foot wounds. The purpose is to indicate diabetic patients' severity of foot wounds. For these things, we give a smart solution. Thus we develop the screening tool by integrating the developed deep learning model to assist the patient and clinicians to make the decision regarding the targeting wound. The design approach follows the Agile Scrum model, ensuring adaptability and transparency throughout development.

Acknowledgement

We would like to express our sincere appreciation and gratitude to all those who are contributing to the development of our Deep Learning based Diabetic Foot Wound Classifier and Screening Tool as part of our third-year final project at the Faculty of Applied Sciences, Rajarata University of Sri Lanka. First and foremost, we would like to thank our project supervisor Dr. K. A. S. H. Kulathilake, for his valuable guidance, expertise, and constant support throughout the project. His insights and feedback have been invaluable in shaping our ideas and ensuring the project's success. Our sincere thanks go to all the lecturers and supportive members of the Information and Technology degree program in the Department of Computing at the Faculty of Applied Sciences. Their contributions have greatly enriched our project and enabled us to overcome challenges along the way.

Contents

CHAPTER 1 - INTRODUCTION 7
1.1 Problem to be addressed 7
1.2 Objectives of the project 8
1.3 Project Deliverables 8
1.4 System Design Approach 8
1.4.1 Design Approaches 8
1.4.2 Process Model 9
1.4.3 Model View Controller 9
1.5 Standards to be followed 10
1.5.1 Coding Standards 10
1.6 Organization of the SDS 11
CHAPTER 2 - ARCHITECTURAL DESIGN 13
2.1 System Architecture 13
2.2 Objects and communication 13
2.2.1 Register 15
2.2.2 Manage account 17
2.2.3 Self-screening DFW 19
2.2.4 Create Session 21
2.2.5 Screening DFW 23
2.2.6 View Patient's Details 25
2.2.7 DFW Analytics 27
2.3 State machines 28
2.3.1 Account Registration 28
2.3.2 Video Session 28
2.3.3 Deployment of the system 29
2.3.4 Special algorithms 30
2.4 Integrating and Developing Toolkit 32
CHAPTER 3 - UI DESIGN 33
3.1 PACT Analysis 33
3.1.1 People 35

3.1.2 Activities 36
3.1.3 Context 38
3.1.4 Technologies 39
3.2 UI Design consideration and approaches 41
3.3 Design tools, techniques, templates 42
3.3.1 Design tools 42
3.3.2 Design Techniques 43
3.3.3 Design Template 43
3.4 Input Design Aspect 44
CHAPTER 4 - DATA MANAGEMENT 50
4.1 Data requirement 50
4.2 Design tools, techniques 50
4.2.1 Design tools 50
4.2.2 Design technique 51
4.3 Conceptual database design 52
4.4 Logical database design 53
4.4.1 Map Regular Entities 53
4.4.2 Map Weak Entities 56
4.4.3 Map associated entities 62
4.4.4 Map unary relationships 62
4.4.5 Map ternary(and n-ary) relationships 62
4.4.6 Map super type/ subtype relationships 62
4.4.7 Map aggregation 63
4.5 Schema refinement 64
4.5.1 PERSON table 64
4.5.2 ADMIN table 65
4.5.3 PATIENT table 65
4.5.4 PATIENT_TELE_NO table 66
4.5.5 DOCTOR table 67
4.5.6 WOUND table 68
4.5.7 WOUND_WOUND_IMAGE table 69
4.5.8 MAIL_NOTIFICATION table 69

4.5.9 SESSION table 70
4.5.10 JOIN table 70
4.5.11 Schema refinement tables 71
4.6 PHYSICAL DATABASE DESIGN 72
4.6.1 Data type 72
4.6.2 Query example 77
4.7 SECURITY DESIGN 80
4.7.1 Access control 80
4.7.2 Data encryption 80
CHAPTER 5 - RESEARCH DESIGN 82
5.1 Objective Based Literature Review 82
5.2 Formalizing High-level implementation components 83
5.2.1 To Develop the Deep Learning Model for Segmenting Diabetic Foot Wounds 84
5.2.2 To develop the deep learning model for classifying diabetic foot wound 88
5.2.3 To formalize a mechanism to improve the accuracy of the proposed deep learning model through transfer learning 90
5.3 Data extractions, sample design, test datasets, training data sets 92
5.3.1 Data Extractions 92
5.3.2 Sample Design 93
5.3.3 Test Datasets and Training Datasets 94
5.4 Non- functional Aspects 94
5.5 Validation methods and measurements 95
CHAPTER 6 - REFERENCES 97
CHAPTER 7 - APPROVAL 98

List Of Figures

Figure 1.1 : Graphical Illustration of the Proposed Method 7
Figure 1.2 : Model View Controller 10
Figure 2.1 : System Architecture 13
Figure 2.2 : Class diagram of reusing classes 14
Figure 2.3 : Class diagram for Register 15

Figure 2.4 : Sequence diagram for Register 16
Figure 2.5 : Class diagram for Manage account 17
Figure 2.6 : Sequence diagram for Manage account 18
Figure 2.7 : Class diagram for Self-screening DFW 19
Figure 2.8 : Sequence diagram for Self screening DFW 20
Figure 2.9 : Class diagram for Create session 21
Figure 2.10 : Sequence diagram for Create session 22
Figure 2.11 : Class diagram for Screening DFW 23
Figure 2.12 : Sequence diagram for Screening DFW 24
Figure 2.13 : Class diagram for View Patient's Details 25
Figure 2.14 : Sequence diagram for View Patient's Details 26
Figure 2.15 : Class diagram for DFW Analytics 27
Figure 2.16 : Sequence diagram for DFW Analytics 27
Figure 2.17 : State diagram for User accounts 28
Figure 2.18 : State diagram for Video session 29
Figure 2.19 Component diagram of the system 29
Figure 2.20 : Deployment diagram of the System 30
Figure 3.1 DFW Capture form 44
Figure 3.2 : Patient's Signup form 45
Figure 3.3 : Create Session Form 45
Figure 3.4 : Account Information Report 46
Figure 3.5 : DFW Details Report 47
Figure 3.6 : Wound Analytics Report 47
Figure 3.7 : Video Session Screen 48
Figure 3.8 : Join session screen 48
Figure 3.9 : Home Screen 48
Figure 3.10 : My Session Screen 48
Figure 3.11 : Chat screen 49
Figure 3.12 : Message screen 49
Figure 3.13 : Home screen 49
Figure 4.1 : EER Diagram 52
Figure 4.2 : Person Entity 53
Figure 4.3 Composite attribute 54
Figure 4.4 : Doctor entity 54
Figure 4.5 : Admin Entity 54
Figure 4.6 : Patient entity 55
Figure 4.7 : Composite attribute 55
Figure 4.8 : Multiple attribute 55
Figure 4.9 : Notification entity 56
Figure 4.10 : Doctor and session relationship 56
Figure 4.11 : Doctor and mail notification relationship 61
Figure 4.12 : Admin and Mail notification relationship 61
Figure 4.13 : Person generalization 62

Figure 4.14 : Aggregation relationship 63
Figure 5.1 : System Architecture 83
Figure 5.2 : DFW Segmentation Architecture 84
Figure 5.3 : DFW classification architecture 88
Figure 5.4 : Concept of the ResNet101 transfer learning model 92
Figure 5.5 : Testing Images 93
Figure 5.6 : Training Masks 93
Figure 5.7 : Training images 93
Figure 5.8 : Classification Dataset Samples 94
Figure 5.9 : F1 Table 95

List of Table

Table 1.1: Scrum table 9
Table 2.1 : Integrating and Developing Toolkit table 32
Table 3.1 PACT Analyze Table 34
Table 3.2 : Factors of people 35
Table 4.1 : Person table 64
Table 4.2 : Admin table 65
Table 4.3 Patient table 65
Table 4.4 : Patien_medical_details table 66
Table 4.5 : New patient table 66
Table 4.6 : Patient_tele_no table 66
Table 4.7 : Doctor table 67
Table 4.8 : Doctor_experience table 67
Table 4.9 : New doctor table 67
Table 4.10 : Wound table 68
Table 4.11 : Classification_screen_type table 68
Table 4.12 : Region table 69
Table 4.13 : New wound table 69
Table 4.14 : Wound_wound_image table 69
Table 4.15 : Mail_notification table 69
Table 4.16 : Session table 70
Table 4.17 : Join table 70
Table 4.18 : Person table 77
Table 4.19 : New person table 77
Table 4.20 : New person table 78
Table 4.21 : Classification_screen_type table 78

CHAPTER 1 - INTRODUCTION

Diabetes is a long-term medical disease where the body struggles to control blood sugar (glucose) levels. This causes blood sugar levels to rise, which over time may cause a number of health issues. Diabetic foot ulcer is a chronic wound that can occur in people with diabetes due to poor blood sugar control or circulation. It can lead to hospitalization and expensive healthcare. Minor foot wounds may go unnoticed, leading to serious consequences.

1.1 Problem to be addressed

Past studies have used various models to detect and classify diabetic foot wounds, with Fast R-CNN and YOLO being the most popular. However, these models can be time-consuming and may miss small objects due to their fixed grid system. Additionally, some studies have used models without considering their specific tasks, leading to inaccurate diabetic foot classification due to inaccuracies in wound segmentation and object detection. The goal of our study is to non-invasively detect ulcers without direct clinical intervention. The development of new technologies to improve ulcer diagnoses and treatment strategies has the ability to transform diabetic foot care.

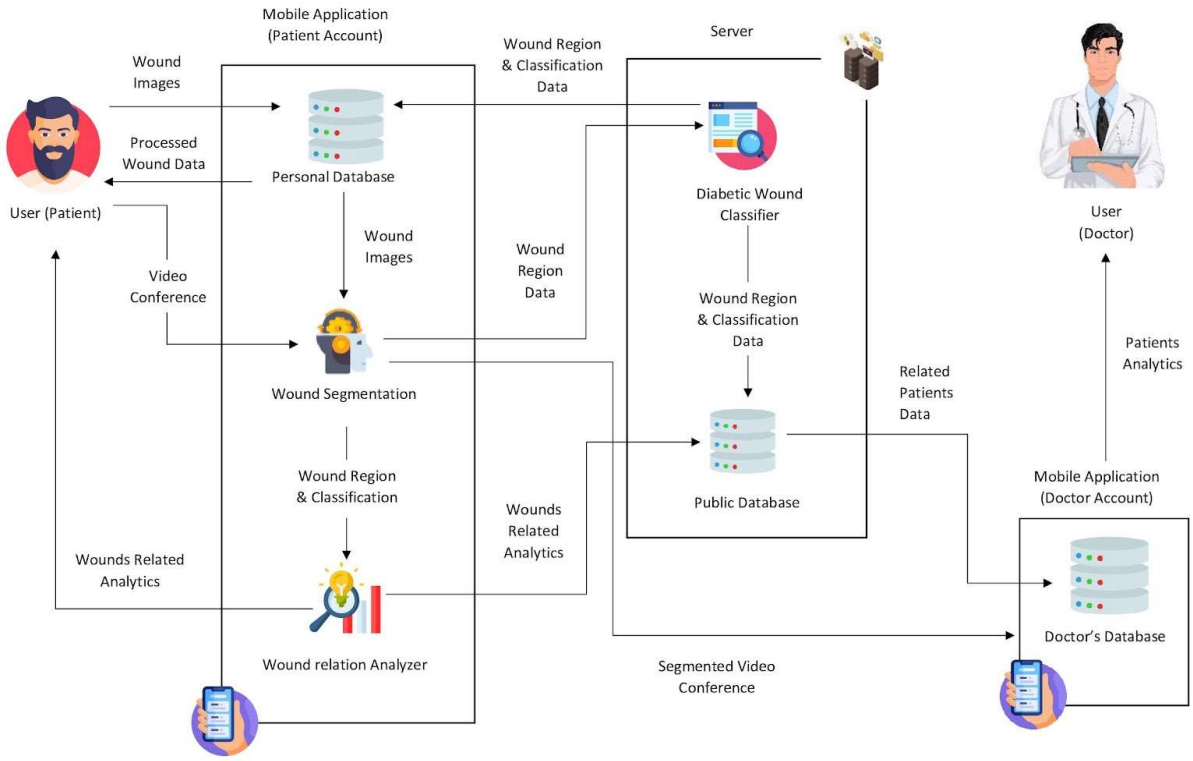


Figure 1.1 : Graphical Illustration of the Proposed Method

1.2 Objectives of the project

The objectives of this proposed research are,

- To develop a segmentation method for extracting the diabetic foot wound and measure the size of the wound.
- To develop the deep learning model for classifying diabetic foot wounds.
- To formalize a mechanism to improve the accuracy of the proposed deep learning model through transfer learning.
- To develop a screening tool by integrating the developed deep learning model to assist the patient and clinicians to make the decision regarding the targeting wound

1.3 Project Deliverables

Project deliverables are the intangible quantifiable goods or services we deliver as the results of the project. The deliverables of our Diabetic Foot Wound Classifier and Screening Tool Features are as follows.

1) The diabetic foot wound segmentation model
2) The diabetic foot wound classification model
3) Technical Report
4) Screening tool for the system

1.4 System Design Approach

1.4.1 Design Approaches

Object Oriented Design

To create our proposed methodology, we employ object-oriented design (OOD). Given that we have numerous functional components, OOD can be useful. Object-oriented design promotes code reusability, modularity, maintainability, and collaboration, making it a widely adopted approach in software development projects. However, the choice of design approach depends on various factors, including project requirements, team expertise, and the nature of the problem domain. Object-oriented design promotes modular code organization.

The system is divided into self-contained objects, each responsible for specific tasks. This modularity makes it easier to understand, modify, and maintain the code. Additionally, objects can be reused in different projects, saving time and effort.

1.4.2 Process Model

Scrum

Firstly, we embarked on our project with great determination, we will pouring our efforts into completing 70% of the work. As we approached the next phase, we were well-aware that the final 30% of the project lay ahead. This stage demanded our utmost attention and dedication, as it held the key to realizing our ultimate objective. We fine-tuned our strategies, fine-tuned our processes, and rallied our team to ensure the successful completion of this critical phase.

Finally, we will accomplish what we set out to do: we try achieved 100% completion of the application integration and evaluation.

Table 1.1: Scrum table

Implementation 01		Implementation 02		Implementation 03	
70%		30%		100%	
Sprint	Duration	Sprint	Duration	Sprint	Duration
Complete UZ/Ux design	03 day	Refine model for improved accuracy	01 week	Integrate the model with the application	01 week
Implement basic UI in flutter	04 day	Implement advance UI features in flutter	01 week	Perform comprehensive testing	01 week
Implement functionality in flutter	01 week	Enhance user experience and usability	01 week	Evaluate and fine-tune the application	01 week
Develop model wire frame	02 week	Perform unit testing and bug fixer	01 week	Dissertation writing and release the application	01 week

1.4.3 Model View Controller

Model :

In our case, it would involve the image classification and segmentation algorithms. The Model component would encapsulate the image processing methods, such as feature extraction, classification models, and segmentation algorithms. It should handle the processing and analysis of the diabetic foot ulcer images.

View :

In our project, the View component can display the user interface for uploading and viewing images, as well as presenting the results of classification and segmentation. The View would provide a way for users to input images and receive visual feedback on the identified ulcers.

Controller :

In our project, the Controller would receive user actions from the View, such as uploading an image for classification or initiating a segmentation process. It would then invoke the relevant methods in the Model to perform the necessary image processing tasks. Once the processing is complete, the Controller would update the View to reflect the results.

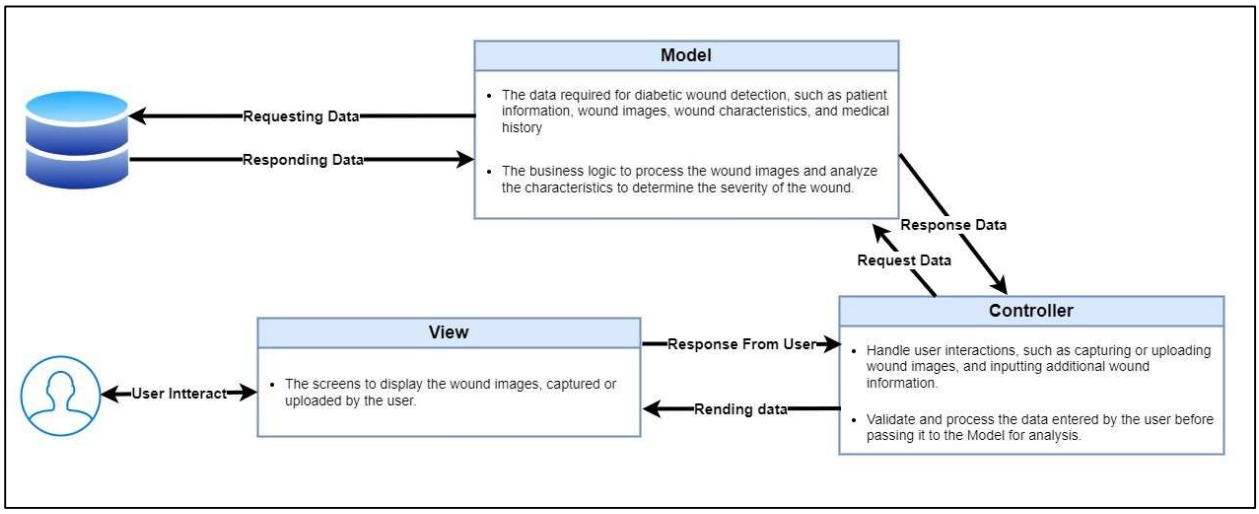


Figure 1.2 : Model View Controller

1.5 Standards to be followed

1.5.1 Coding Standards

It's essential to establish coding standards specific to our development team and project to ensure consistency and maintainability throughout the mobile application development process. Below are the coding standards that we generally follow for mobile application development.

- Comments and Documentation :

  Firstly, we ensure that comments are used sparingly but effectively, providing insights into complex or critical sections of code. Comments should be concise, clear, and focus on the why rather than the how. For documentation, we follow a consistent format and include high-level

overviews, module-level explanations, and detailed function/method descriptions.

- Error Handling

  We start by defining clear guidelines and best practices for error handling. This includes properly documenting error codes, messages, and their corresponding actions. Try-catch blocks are used to encapsulate the error-prone code segments and provide meaningful error messages or log entries for debugging purposes. Implements robust logging mechanisms for reporting and tracking and regularly reviews and updates error handling practices based on feedback and evolving project requirements.

- Version Control and Collaboration

  First we choose a version control system like Git. Then it allows us to track changes, manage branches, and collaborate effectively with others. We need to set up a repository for our project and establish guidelines for branching, committing, and merging code. .Uses branches to work on new features or bug fixes, and merges them back into the master branch after thorough testing and code review. Establishes coding standards and enforces them through code reviews, model guidelines, and automation tools.

- Naming conventions

  In general, we try to use meaningful and descriptive names that accurately represent the purpose and function of the organization. Avoid using vague abbreviations and opt for clarity instead. Additionally, consider using prefixes to indicate the type or purpose of the entity. Regular code reviews can identify any deviations from naming conventions and provide an opportunity for discussion and refinement.

1.6 Organization of the SDS

In the second Chapter, System architecture, objects and communication will be discussed. The processes and the special algorithms that are used in this application will be briefly introduced in the middle section of the second chapter.
Furthermore in the second chapter we will discuss about the state machines, tools, techniques, libraries, third part tools and the implementation environment.

At the beginning of the third chapter, the PACT analysis will be described followed by the UI design consideration and approaches. The design tools, techniques and templates will be discussed in the middle part of the third chapter. Furthermore, in this chapter we will discuss the input, output and dialogue design aspects. Finally, the hosting and installation environment will be conversed at the latter part of the third chapter.

At the beginning of the fourth chapter, the data requirements required to implement this project will be discussed. Then the design tools that we are using throughout the project

will be discussed. Next the EER diagrams. Logical and physical conceptual designs were discussed. Finally, the security mechanism of the database will be discussed.

In this fifth chapter we study the objectives based on the literature review that we have conducted and formalize high level implementation components. Extraction of data, samples design, testing data sets and training data sets are addressed towards the middle of this chapter. Non-functional aspects, the proposed validation methods and measurements are briefly elaborated at the end of this chapter.

CHAPTER 2 - ARCHITECTURAL DESIGN

2.1 System Architecture

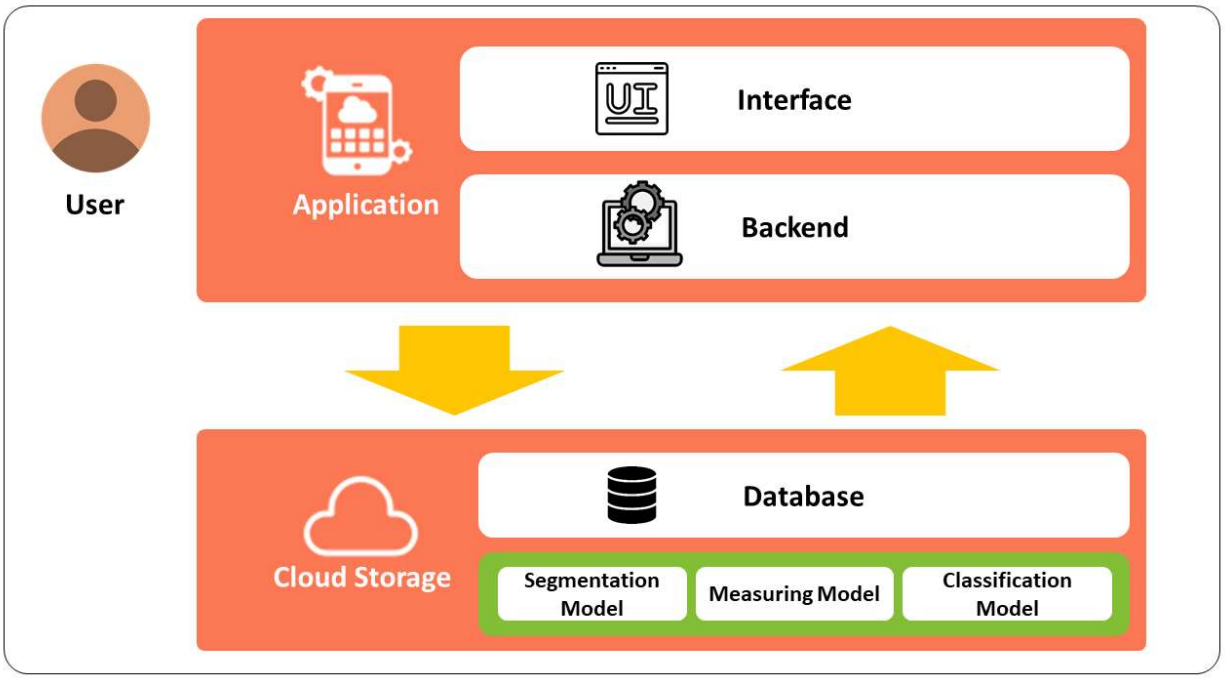


Figure 2.1 : System Architecture

The system architecture can be depicted as consisting of two main components, a mobile application and cloud storage. Within the mobile application, there are two primary elements the User Interface (UI) and the Backend. The UI handles the visual presentation and user interaction, while the Backend manages the underlying functionality and operation of the application.

The cloud storage component includes a Database and three models, the Segmentation model, the Measuring model, and the Classification model. The Database serves as a centralized repository for storing and managing the data required by the system. The three models play distinct roles in executing specific tasks within the system.

To gain a comprehensive understanding of the system, it is essential to analyze the objects and communication involved. This analysis provides valuable insights into the interactions and relationships among the various components and entities.

2.2 Objects and communication

The primary focus is to examine how objects communicate at the class level. To address the complexity of the diagrams, the class diagrams have been divided into use case scenarios to reduce complexity. The following figure illustrates the reuse of classes in each diagram.

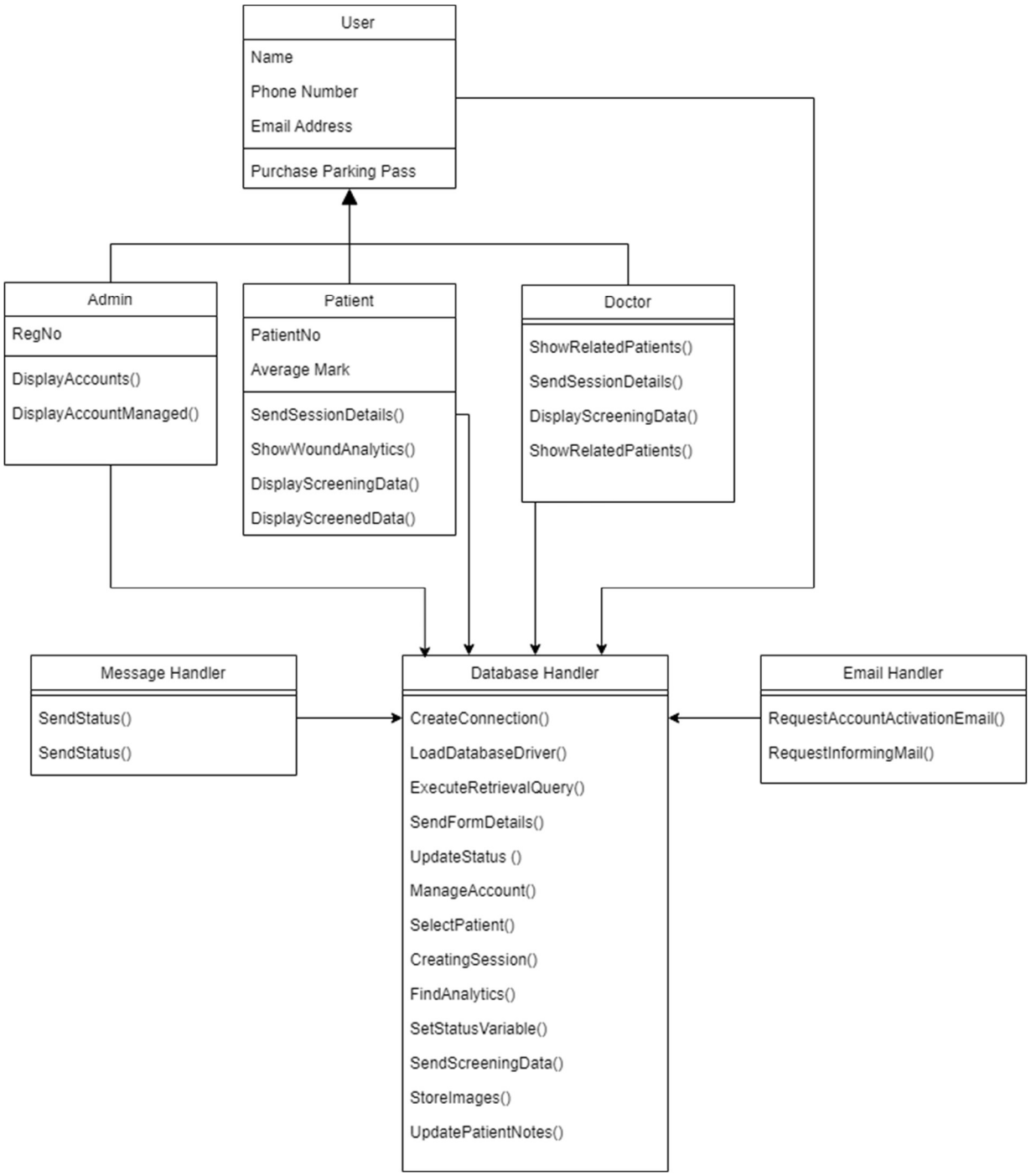


Figure 2.2 : Class diagram of reusing classes

The following subtopics describe the class diagrams based on the main use cases, and they have been designed in collaboration with sequence diagrams. This approach allows for a more detailed understanding of the interactions and behaviors within the system.

2.2.1 Register

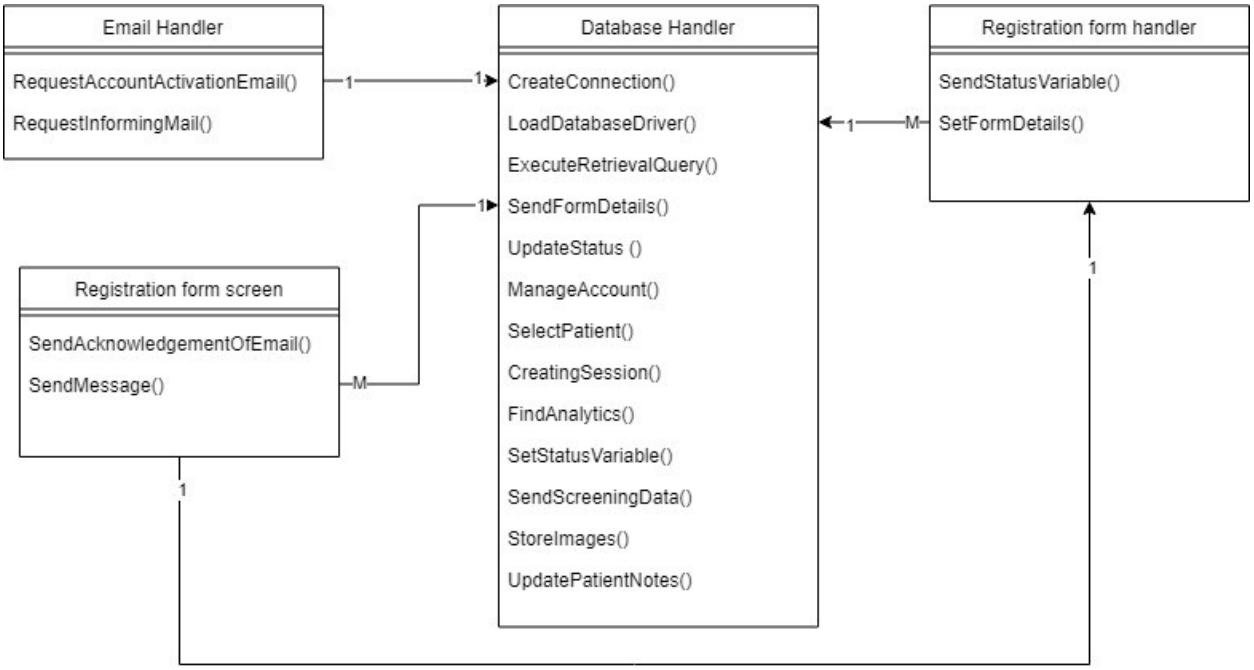


Figure 2.3 : Class diagram for Register

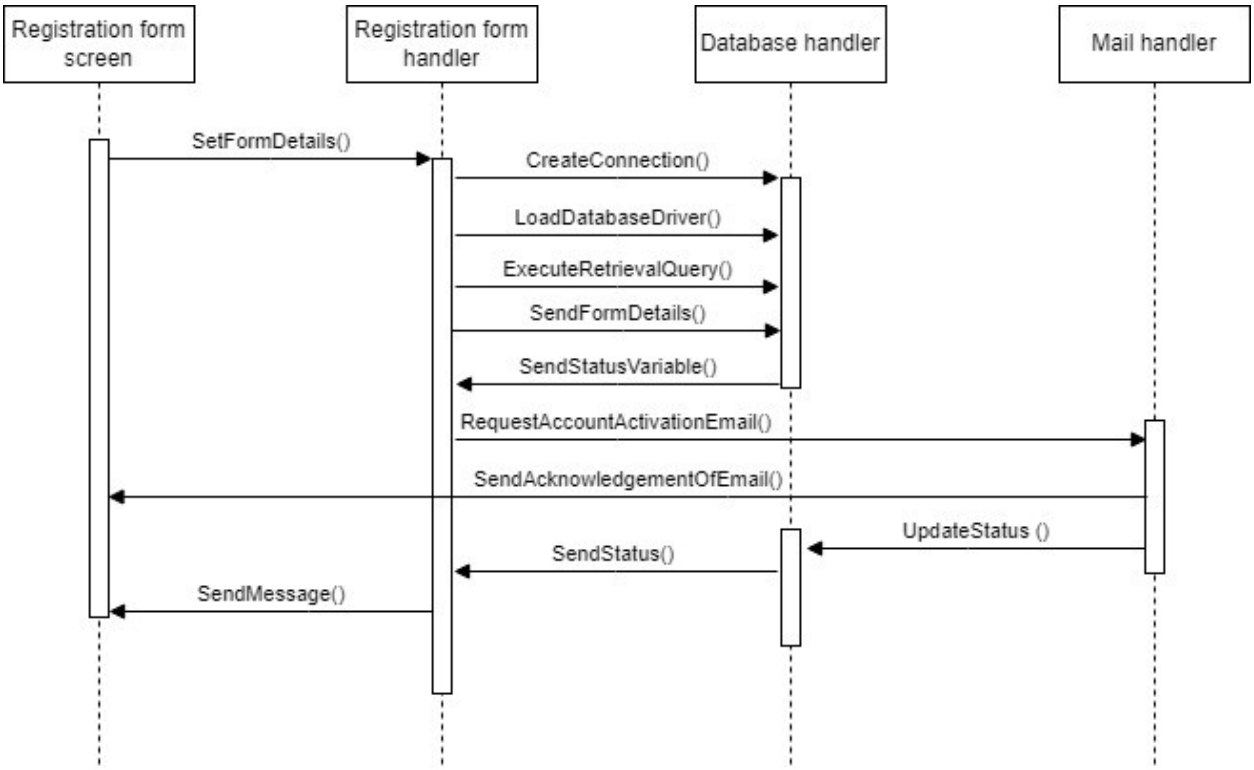


Figure 2.4 : Sequence diagram for Register

In the register use case, there are four key objects involved: The Registration Form Screen, Registration Form Handler, Database Handler, and Mail Handler. These objects work together to facilitate the registration process.

Registration Form Screen is the user interface component where users provide their registration details, such as name, email address, and password.

Once the user submits the registration form, the Registration Form Handler comes into play. It receives the form data and assigns the values to variables within its scope. This handler acts as an intermediary between the user interface and the backend functionality.

The Registration Form Handler performs a check to determine whether a connection to the database exists. If the connection is established, the handler proceeds to send the collected registration data to the Database Handler. The Database Handler is responsible for managing interactions with the database, including storing the user's registration information securely.

After successfully storing the registration data in the database, the Registration Form Handler proceeds to request an account activation email from the Mail Handler. The Mail Handler is designed to handle email-related tasks, such as generating and sending activation emails to users. It leverages appropriate mechanisms to compose and transmit the activation message effectively.

Upon receiving the activation email, the user clicks on the provided activation link, triggering the activation process. As a result, the user's account becomes activated. The Registration Form

Handler is then responsible for updating the status variable associated with the account, indicating its activated state.

2.2.2 Manage account

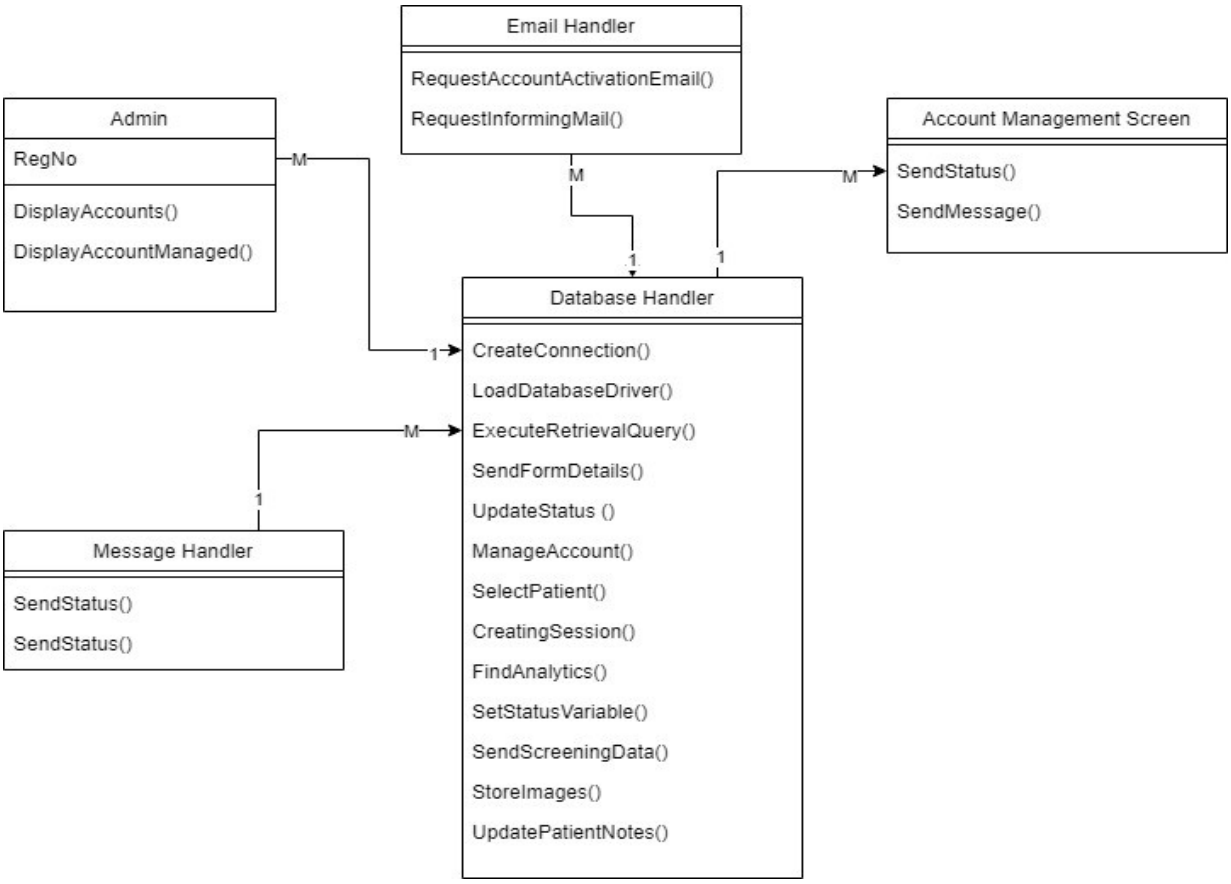


Figure 2.5 : Class diagram for Manage account

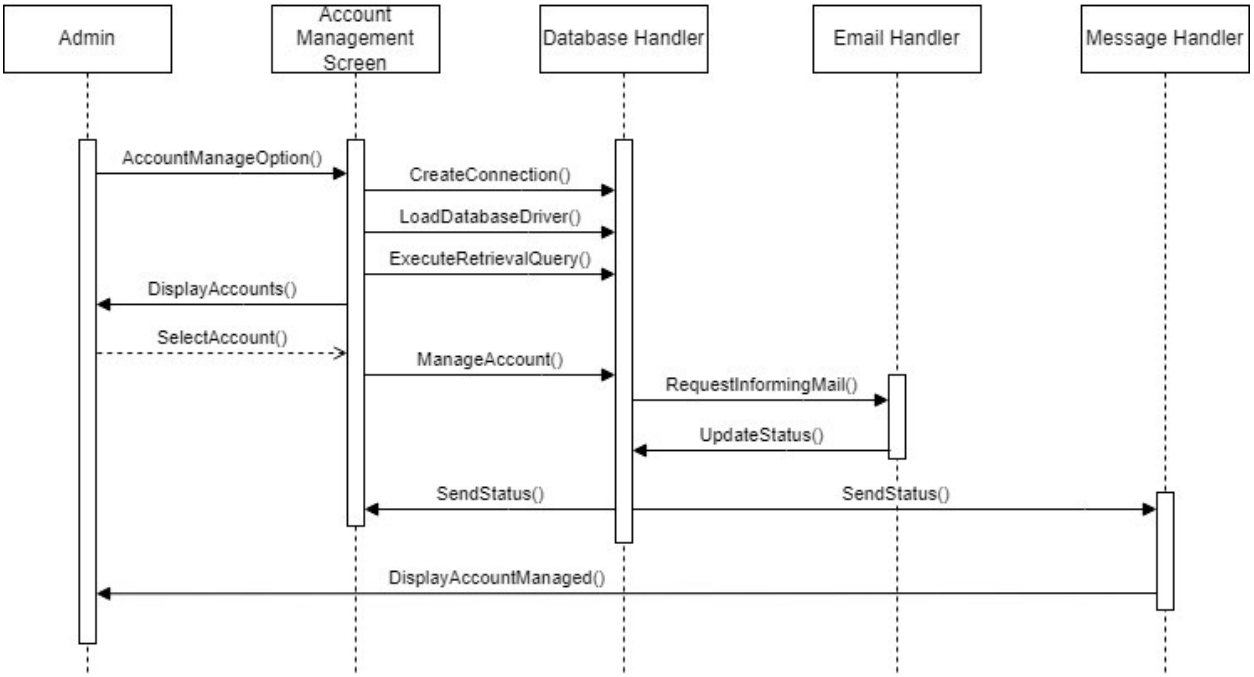


Figure 2.6 : Sequence diagram for Manage account

In this particular use case, there are five key objects involved Admin, Management Screen, Database Handler, Email Handler, and Message Handler. These objects work together to facilitate the account management process.

The Admin represents the administrative user who has access to the system's account management functionalities. They can perform account deactivation and reactivation of user accounts.

The Management Screen is the user interface component that provides the Admin with the necessary options and controls to manage user accounts. It serves as the primary interaction point for the Admin to perform account-related tasks.
Upon selecting the account management option, the Admin contacts the Database Handler to retrieve information about the accounts in the system. The Database Handler interacts with the underlying database, fetching and processing the required account data.

After obtaining the necessary account details, the Database Handler requests an email to be sent from the Email Handler. The Email Handler is responsible for managing email-related tasks. In this case, it generates and dispatches a specific email to the targeted user based on the Admin's actions.

Once the Email Handler completes the email-sending process and updates the status of the corresponding account, the Database Handler sends the updated account status to both the Management Screen and the Message Handler. The Message Handler, in turn, is responsible for acknowledging to the Admin that the account management process has been successfully carried out.

2.2.3 Self-screening DFW

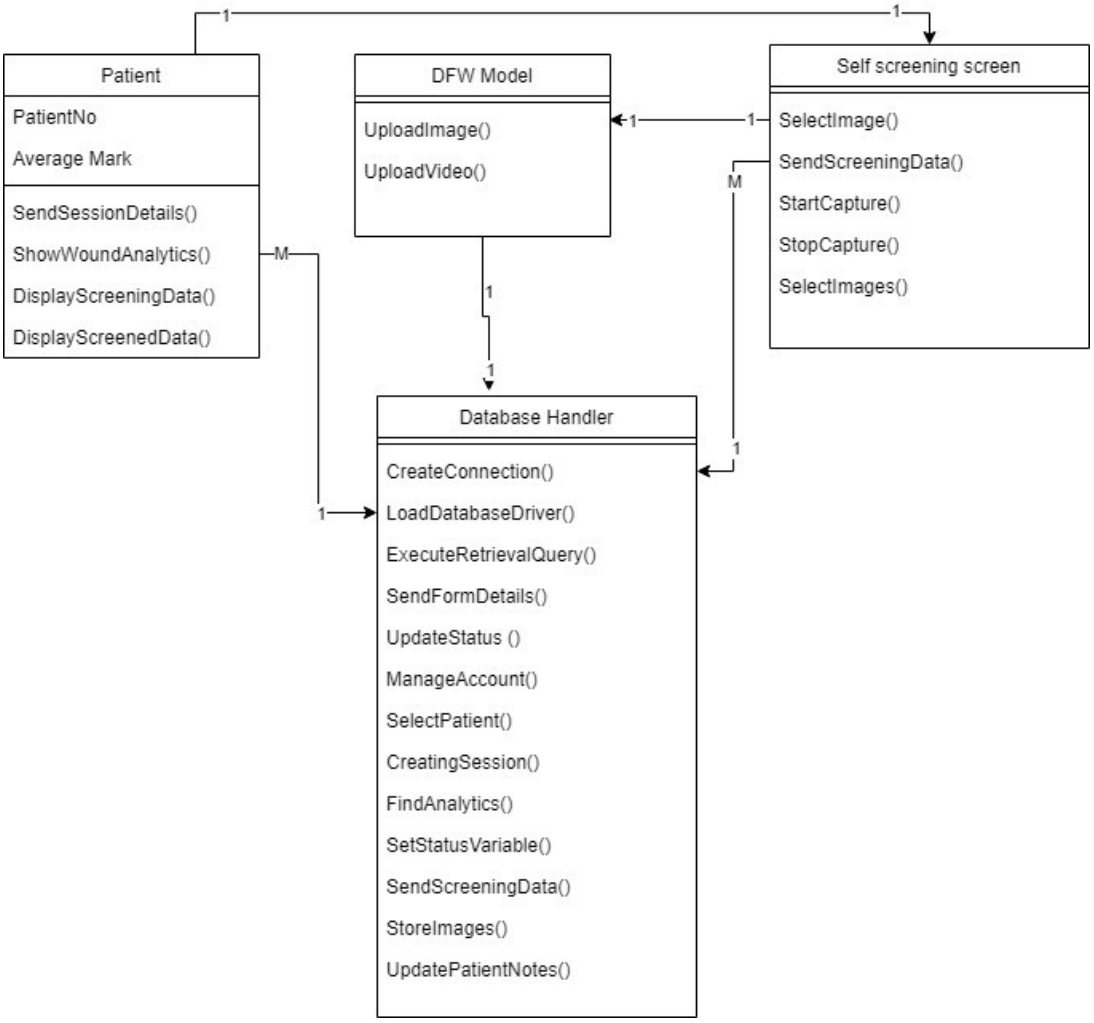


Figure 2.7 : Class diagram for Self-screening DFW

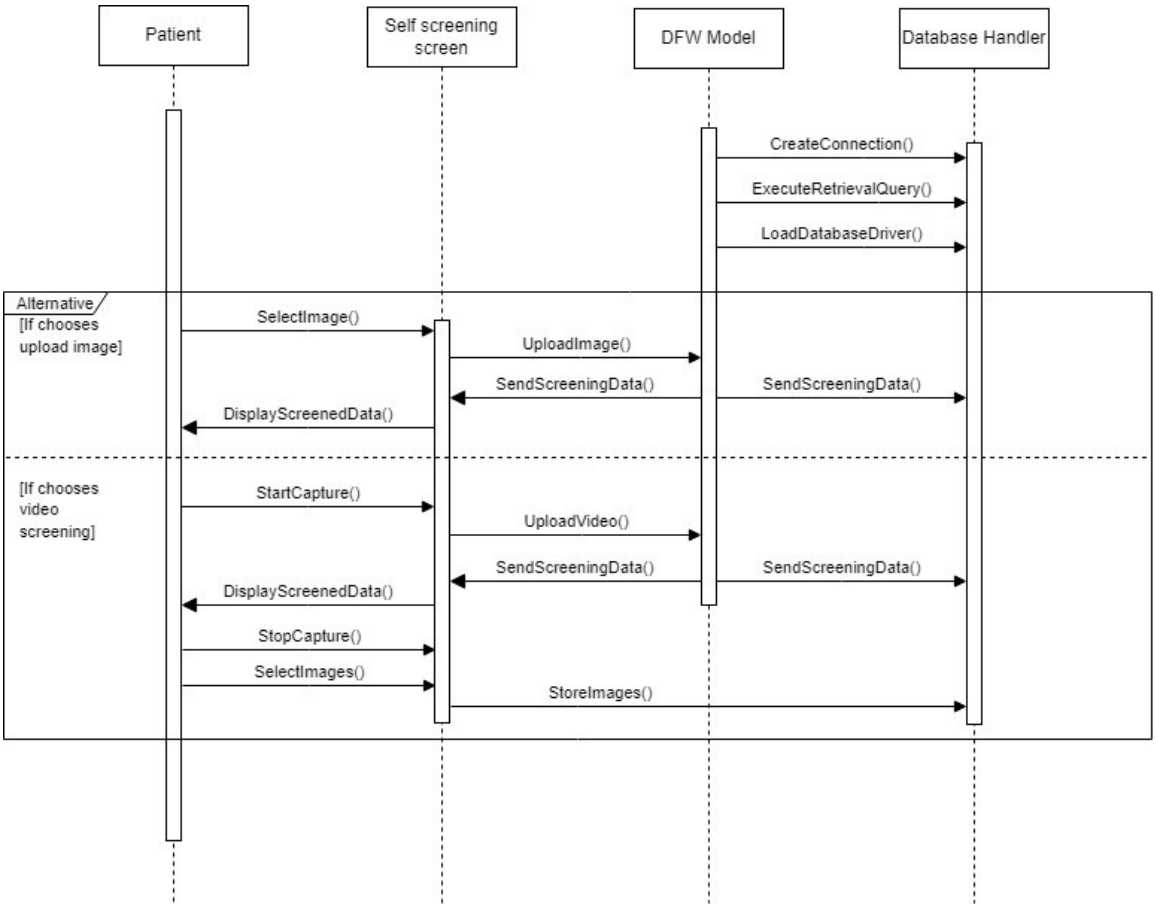


Figure 2.8 : Sequence diagram for Self screening DFW

The self-screening DFW process involves four main objects: The Patient, the Self-Screening Screen, the DFW Model, and the Database Handler.

The DFW Model first checks the connection with the Database Handler.
The Patient then has two options for self-screening: image screening or video screening.
The Patient selects an image from their device. The Self-Screening Screen then sends the image to the DFW Model. The DFW Model analyzes the image and sends the screening data to the Self-Screening Screen. The Self-Screening Screen then displays the screening data to the Patient. The screening data is also sent to the Database Handler for storage.
The Patient starts the video capture. The Self-Screening Screen then uploads the video to the DFW Model. The DFW Model analyzes the video and sends the screening data to the Self-Screening Screen and the Database Handler. The Self-Screening Screen then displays the screening data to the Patient. The Patient can then stop the video capture and select images from the video. The selected images are then stored in the Database Handler.

2.2.4 Create Session

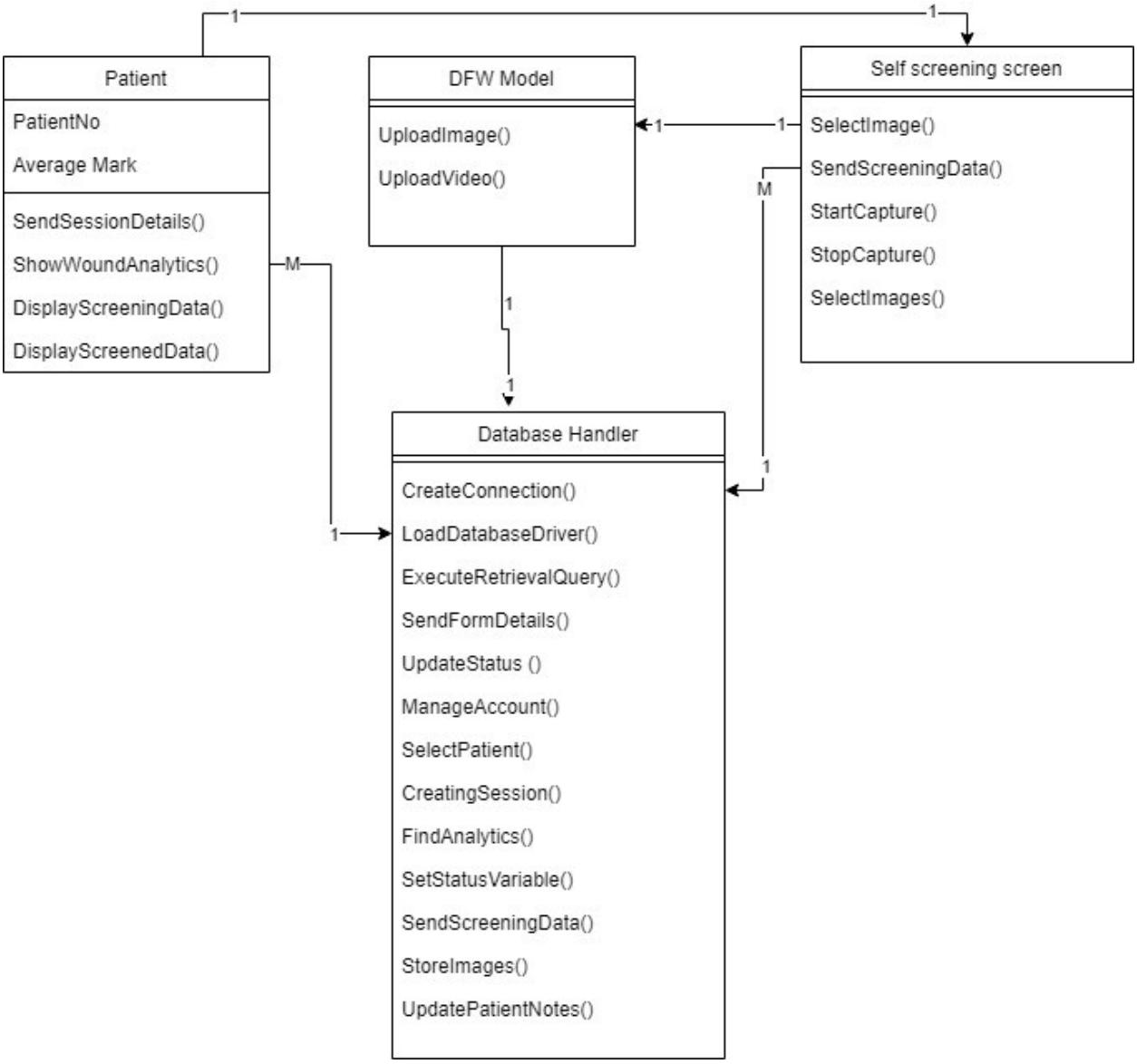


Figure 2.9 : Class diagram for Create session

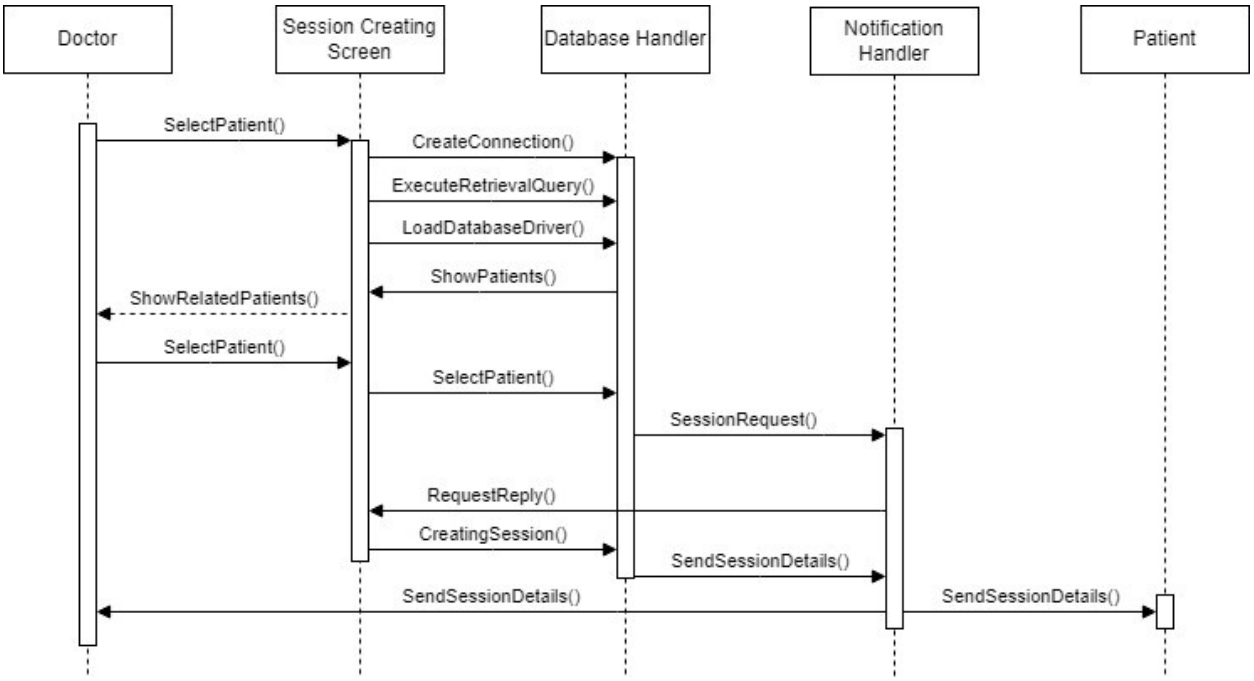


Figure 2.10 : Sequence diagram for Create session

The Create Session Use Case allows a Doctor to schedule a video session with a Patient. The Doctor selects the Patient from the Session Creating Screen. The Session Creating Screen checks the connection with the Database Handler by calling the functions CreateConnection(), ExecuteRetrievalQuery(), LoadDatabaseDriver(). The Database Handler sends the Patient data to the Session Creating Screen. The Session Creating Screen sends the Patient data to the Doctor. The Doctor reviews the Patient data and selects a particular Patient with whom they want to have a video session. The Session Creating Screen sends the selected Patient data to the Database Handler. The Database Handler sends the session request to the Notification Handler. The Notification Handler sends the session request to the Patient. The Patient confirms or denies the session request. If the Patient confirms the session request, the Session Creating Screen creates the session and sends the session data to the Database Handler. The Database Handler stores the session data. The Database Handler sends the session details to the Notification Handler. The Notification Handler sends the session details to the Doctor and the Patient.

2.2.5 Screening DFW

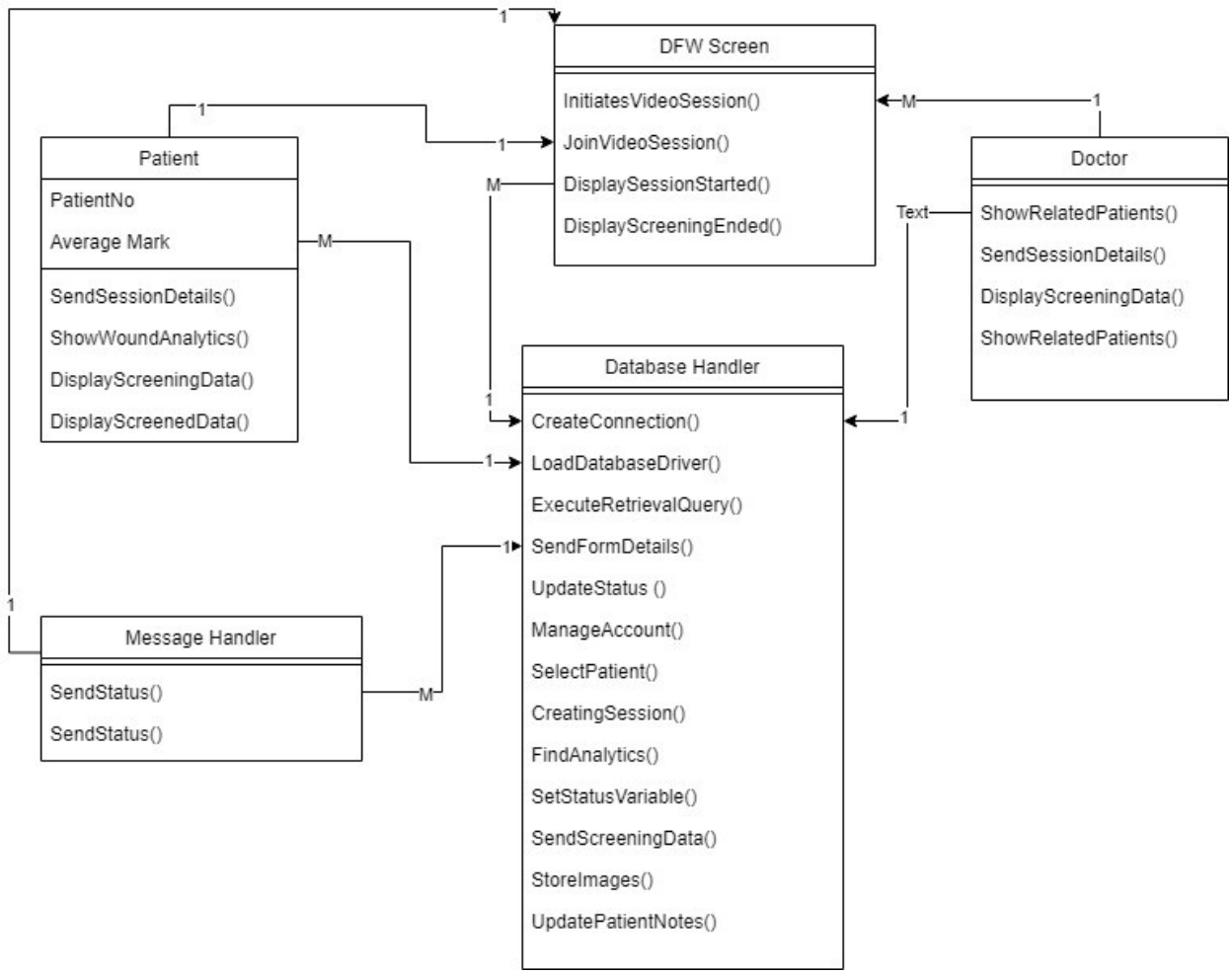


Figure 2.11 : Class diagram for Screening DFW

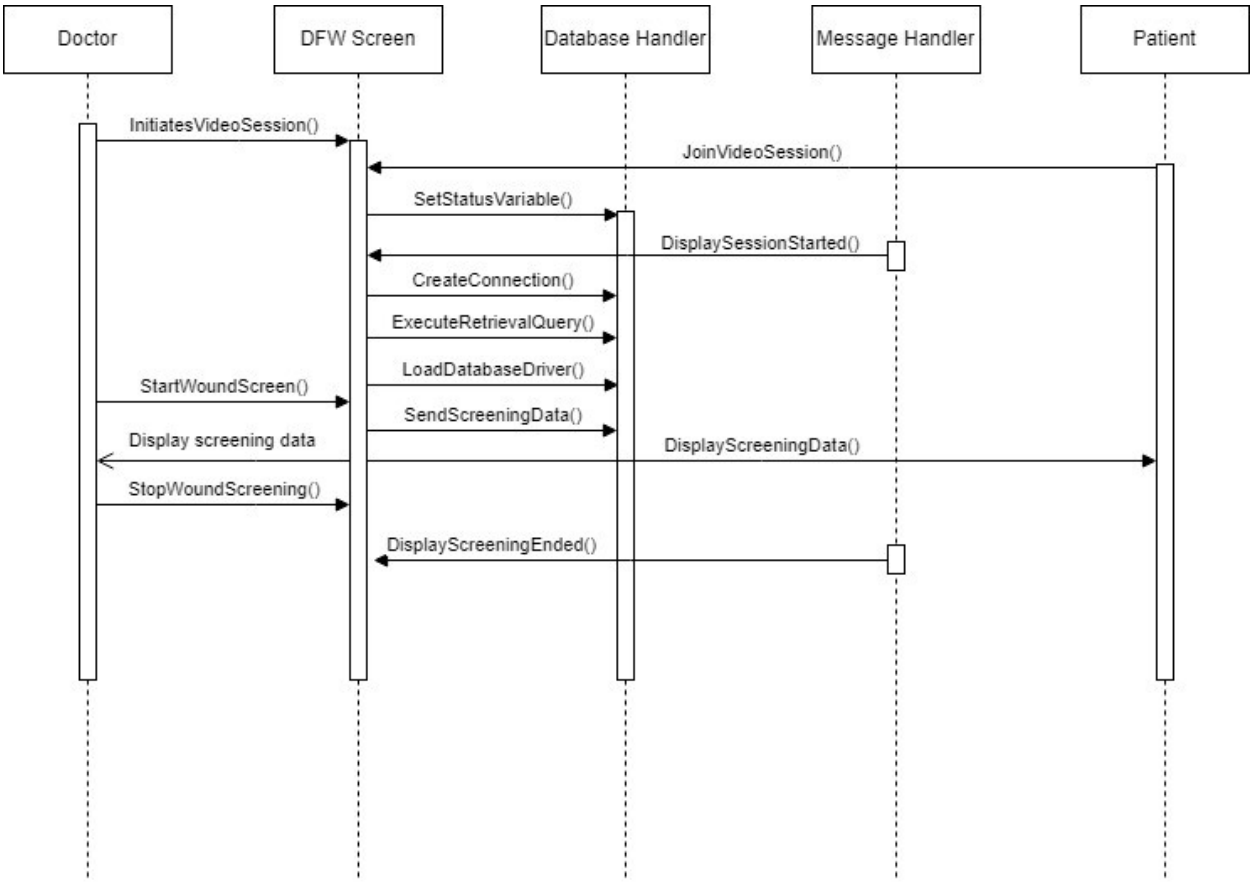


Figure 2.12 : Sequence diagram for Screening DFW

The DFW screening use case involves the following objects: Doctor, DFW Screen, Database Handler, Message Handler, and Patient.

The Doctor initiates the video session, and the Patient joins the video session. Once both parties are connected, a test variable is set. The Message Handler then requests that the session start be displayed. The DFW Screen checks the database connection, creates a connection, executes a query, and loads the database driver functions.

Once the session has started, the Doctor can begin the wound screening. The screening data is then saved to the Database Handler and displayed to the Doctor and Patient. The Doctor can then stop the wound cleaning, and the Message Handler will display that the screening has ended.

2.2.6 View Patient's Details

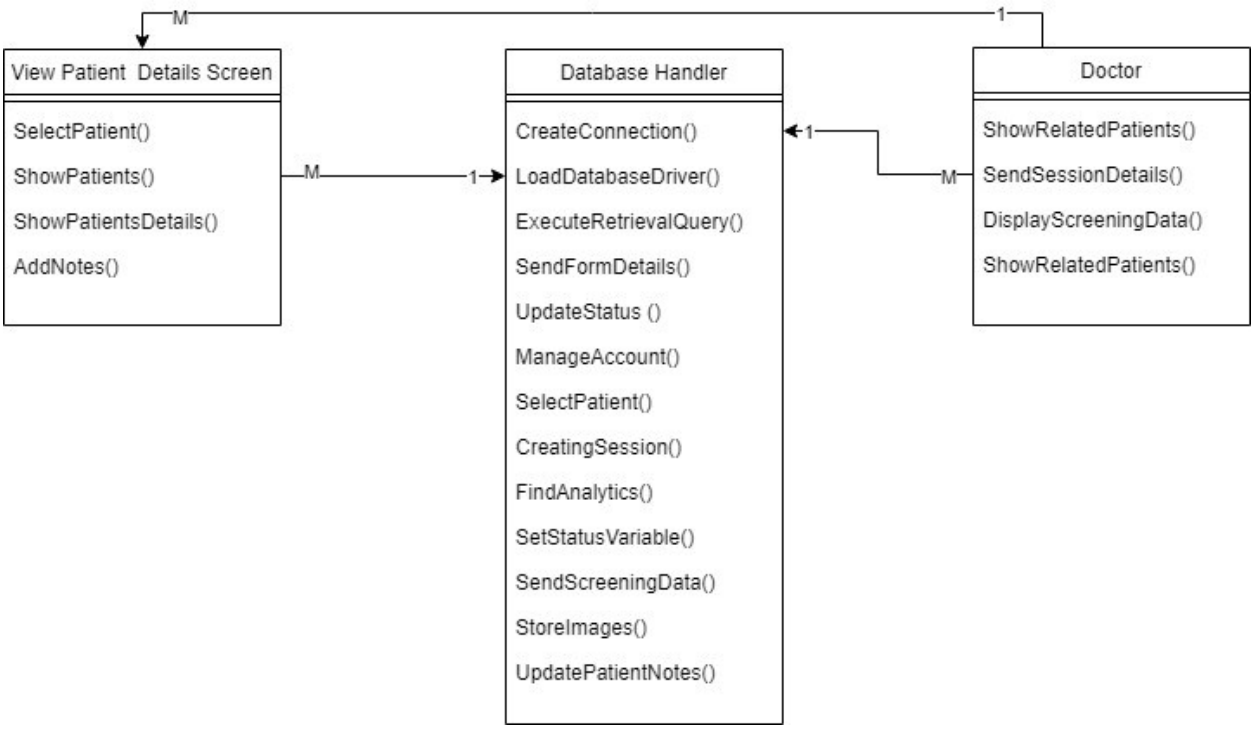


Figure 2.13 : Class diagram for View Patient's Details

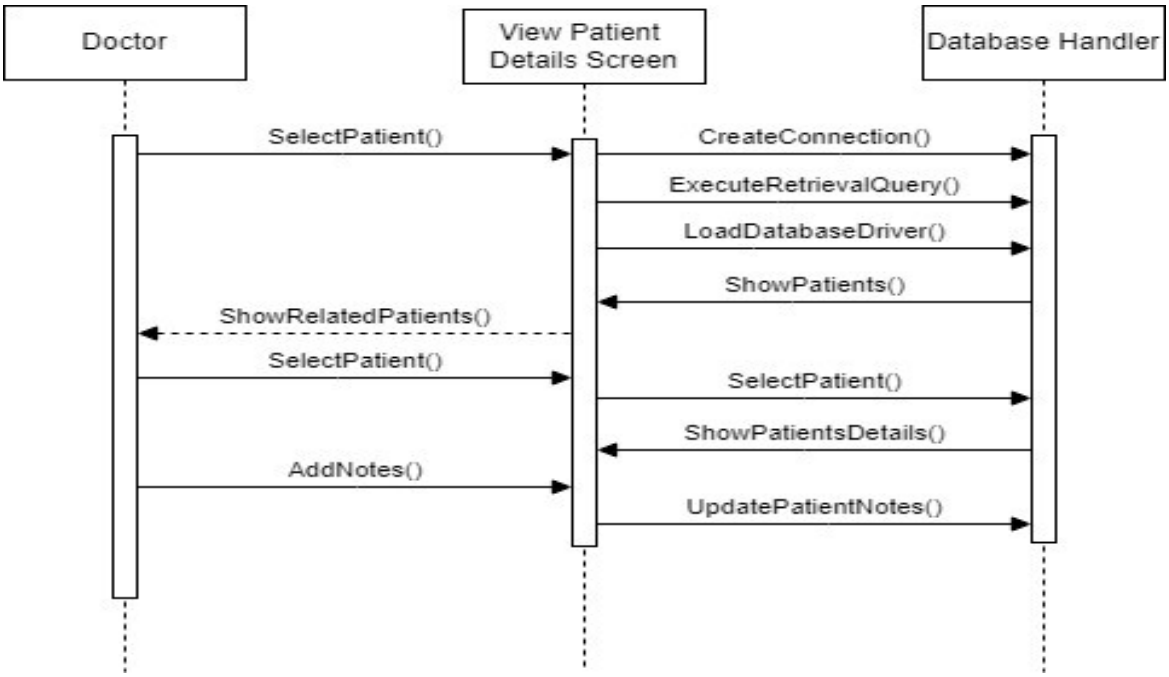


Figure 2.14 : Sequence diagram for View Patient's Details

In the View Patient Details use case, the Doctor has the ability to select a patient and view their details. The View Patient Details screen is responsible for establishing a connection with the Database Handler. Once the connection is established, the Database Handler retrieves the patient data and presents it to the View Patient Details screen. The Doctor can then access and display the relevant patient details. Additionally, the Doctor has the option to select a specific patient and view their details in more depth. The Doctor can also add notes to the patient's record. The View Patient Details screen updates the patient notes in the Database Handler accordingly.

2.2.7 DFW Analytics

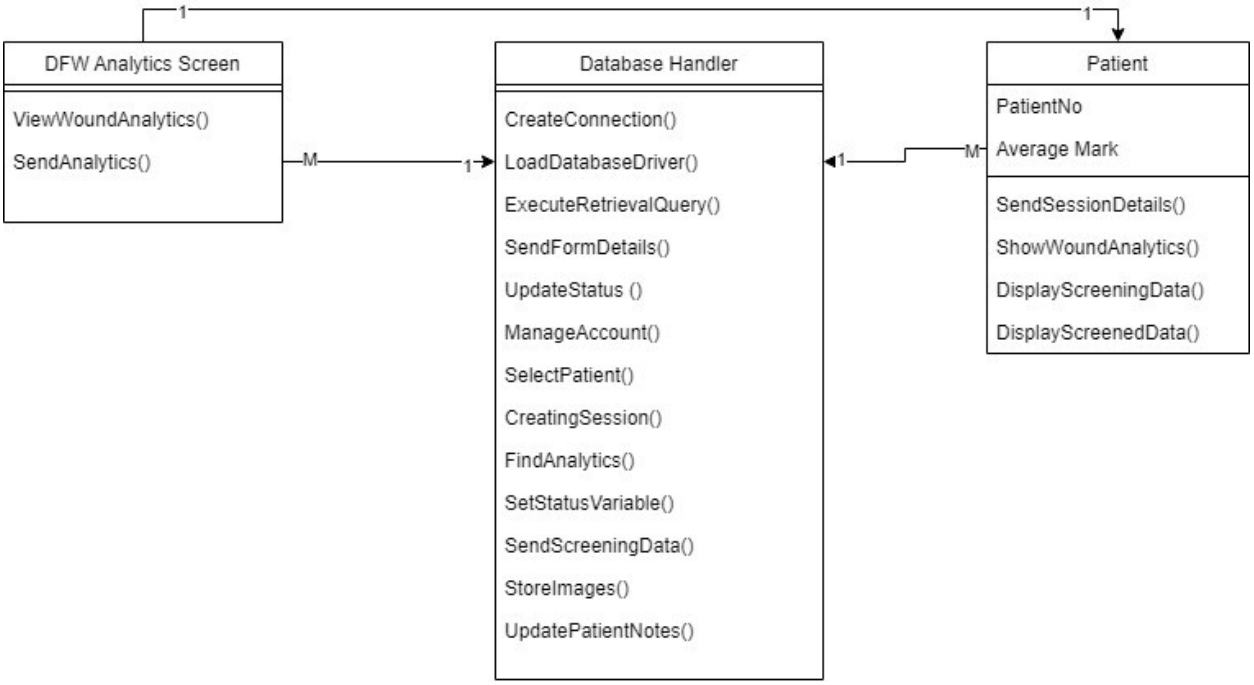


Figure 2.15 : Class diagram for DFW Analytics

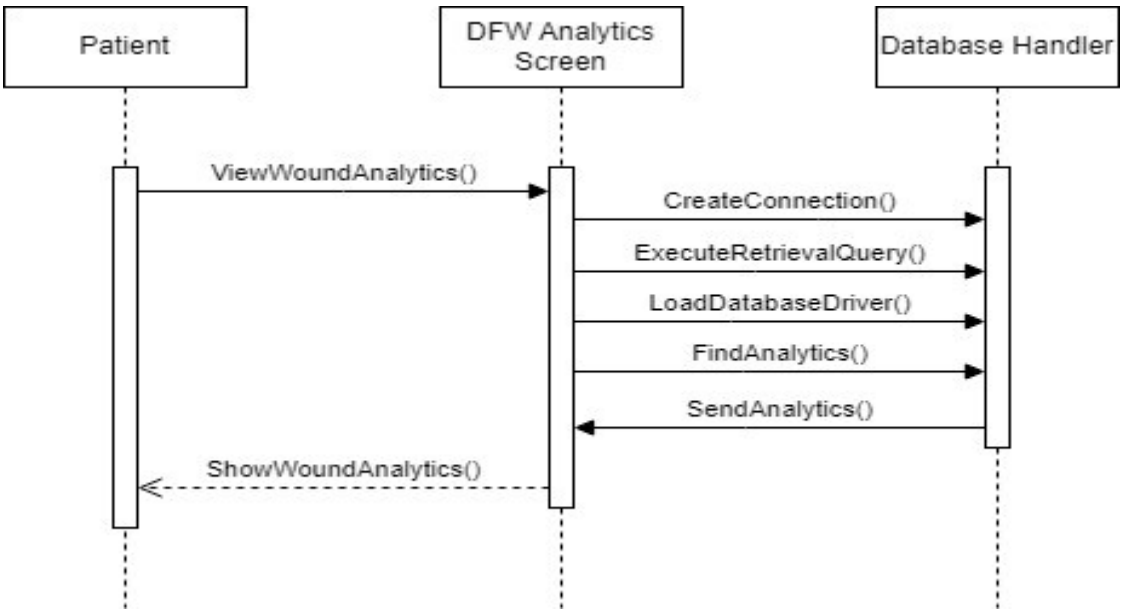


Figure 2.16 : Sequence diagram for DFW Analytics

In the DFW Analytics use case, the Patient can access wound analytics through the View Wound Analytics function. The DFW Analytics screen ensures a connection with the Database Handler

using functions like create connection, execute retrieval query, and load database driver. By utilizing the Database Handler, the DFW Analytics screen retrieves the patient's wound analytics from the database. Subsequently, the DFW Analytics screen presents the wound analytics to the Patient for their viewing.

2.3 State machines

During the object communications mentioned above, two states occur within that communication process. These states reflect the different stages or conditions that objects may go through as they interact with one another.

2.3.1 Account Registration

After the user submits the registration form, the account status is initialized as 0, indicating a pending or inactive state. However, when the user interacts with the activation link, the account status transitions to 1, indicating successful activation and granting access to the account's functionalities.

If the admin chooses to deactivate the account at a later stage, the status is updated to 2, representing a deactivated state where the account's functionalities are temporarily suspended. Nonetheless, the admin maintains the authority to reactivate the account by changing the status back to 1, restoring full access and functionality to the account.

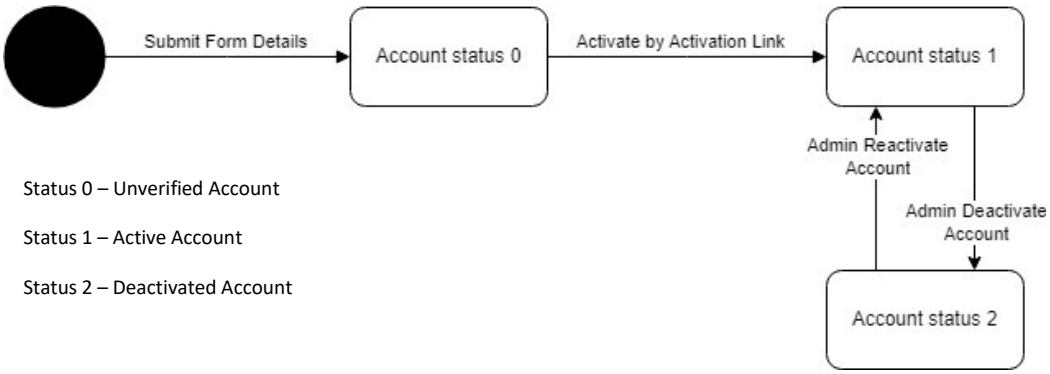
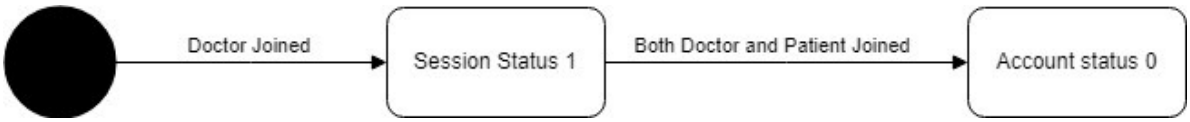


Figure 2.17 : State diagram for User accounts

2.3.2 Video Session

The video session can only be initiated by the doctor, and once the doctor logs in, the status is set to 1. Subsequently, the patient can join the session by logging in, which transitions the status to 0. This workflow ensures that once the patient also logs in, the session becomes active.



Status 0 – Doctor and Patient Only

Status 1 – Doctor Only

Figure 2.18 : State diagram for Video session

2.3.3 Deployment of the system

The following diagrams depict the system architecture in more detail. The Component Diagram illustrates the various components that make up the system, highlighting their interactions and relationships. On the other hand, the Deployment Diagram provides insights into the hardware infrastructure on which the system is deployed. Together, these diagrams provide a comprehensive view of the system's architecture, encompassing both its software components and the hardware resources on which it operates.

Desired Components of the Proposed System

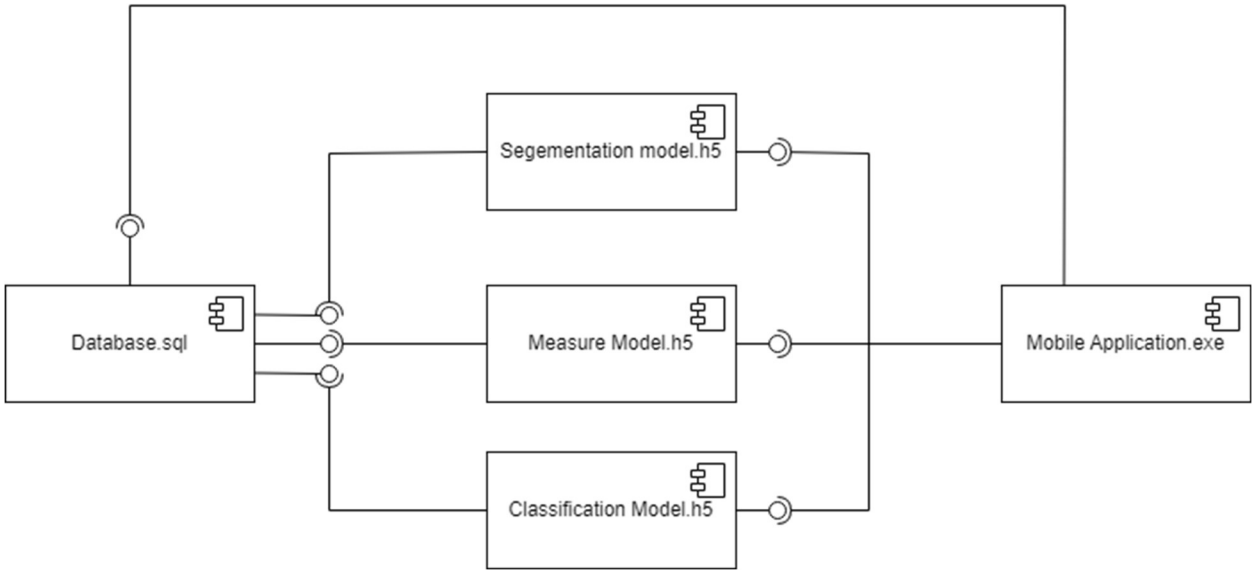


Figure 2.19 Component diagram of the system

Deployment View of the System

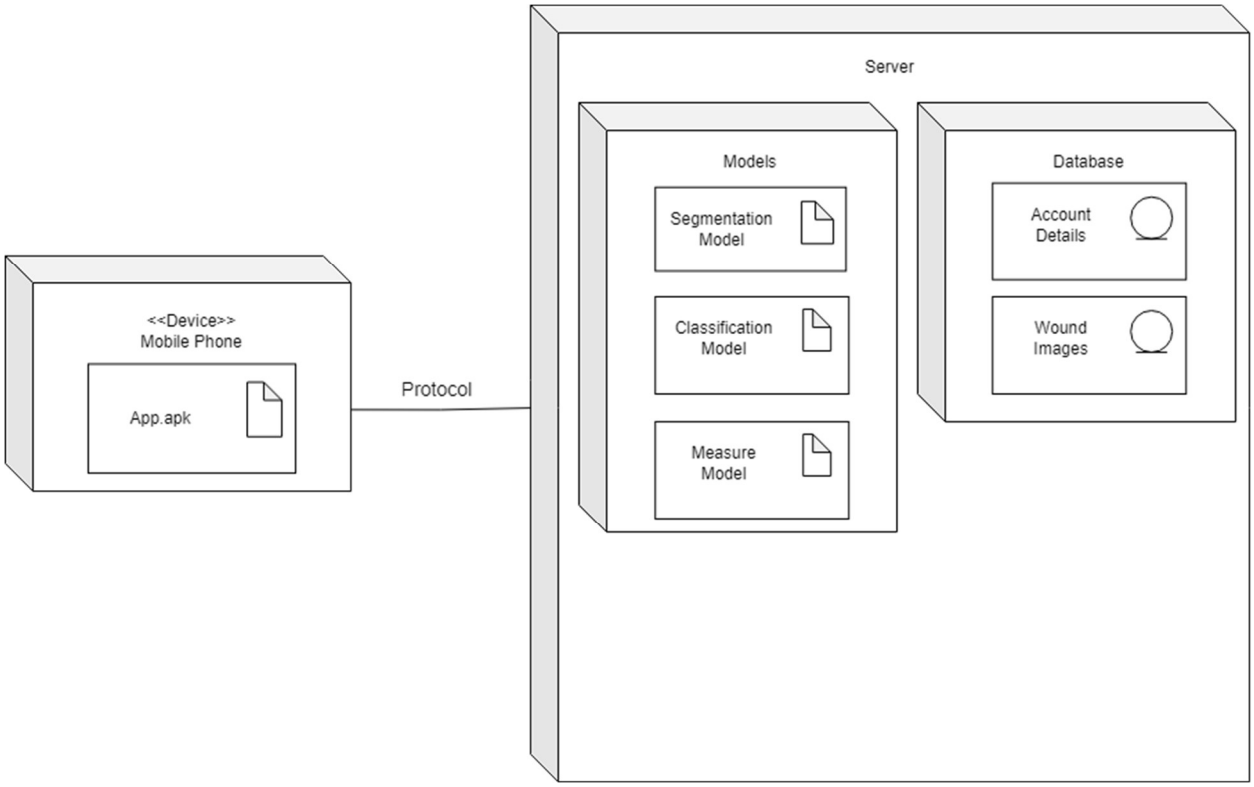


Figure 2.20 : Deployment diagram of the System

2.3.4 Special algorithms

We employ a specific algorithm to train our models, which involves the following steps.

1. Forward propagation
2. Error calculation
3. backward propagation

Forward propagation involves passing input data through the neural network in a forward direction, from the input layer to the output layer. Neurons in each layer perform calculations on their inputs, apply activation functions, and pass the results to the next layer. This process continues until the final output is generated, allowing the network to make predictions based on the input data.

Error calculation quantifies the difference between the predicted output of the network and the actual output (ground truth) for a given input. Various error or loss functions, such as mean squared error (MSE) or cross-entropy loss, are used to measure this discrepancy. The error calculated during forward propagation provides valuable information about the network's performance and guides the subsequent steps.

Backward Propagation (Backpropagation):

Backward propagation involves calculating gradients of the error with respect to the network's weights and biases. Starting from the output layer and moving backward, the gradients are computed using the chain rule of calculus, allowing us to determine the impact of each parameter on the overall error. These gradients are then used to update the network's weights and biases, typically using optimization algorithms like gradient descent. The goal is to minimize the error by adjusting the network's parameters, improving the model's ability to make accurate predictions.

2.4 Integrating and Developing Toolkit

Table 2.1 : Integrating and Developing Toolkit table

Tool/Technique/Library/ language/Framework		Purpose
Tools, & Techniques	Draw.io	We use Draw.io for draw Flowcharts, UML diagrams, organizational charts, and network diagrams.
	MySQL	We use MySQL to manage our data in the since because it is a popular and reliable database management system that offers robust features for storing, managing, and retrieving data.
Implementation Environment	Visual Studio Code	Visual Studio Code (VS Code) is a versatile code editor with features like syntax highlighting, debugging, and an integrated terminal. Its extensive extension ecosystem enables customization. We've planned to use Visual Studio as a text editor to implement the backend of our mobile application.
	Google Colab	Google Colab is utilized for coding and running Python notebooks, specifically for our image-related tasks. Its pre-configured environment provides access to high-performance computing resources, making it optimal for training machine learning models on our dataset. We can take advantage of Colab's seamless integration with TensorFlow and OpenCV, enabling smooth usage of these libraries. Additionally, Colab facilitates real-time collaboration with team members and easy sharing of notebooks, enhancing our productivity and project management.
	Firebase	Firebase is chosen for our mobile application development due to its integrated and scalable backend infrastructure. It simplifies the development process and offers seamless integration with frontend frameworks or mobile SDKs, ensuring efficient and effective creation of our application.
Libraries	Tensor Flow	TensorFlow is a popular open-source library for machine learning and deep learning. We chose TensorFlow for our system because it provides a flexible and efficient framework for building and training neural network models, particularly for image recognition tasks. Its extensive support and capabilities in this area make it a suitable choice for developing the models required for our application.
	Keras	We use Keras for creating deep models which can be productized on smartphones.
	OpenCV	We use OpenCV with the combination of python to implement our image processing techniques.

CHAPTER 3 - UI DESIGN

UI design, or User Interface design, is the discipline of designing the visual and interactive elements of a digital product or application. It involves creating the overall look and feel of the interface, including layout, typography, color schemes, and visual elements such as icons and images. UI designers consider principles of usability, accessibility, and user experience (UX) to ensure that the interface is intuitive and user-friendly. The design process typically involves wireframing, prototyping, and iterative testing to refine the interface and optimize the user's interaction with the product [1]. Effective UI design is crucial in enhancing user satisfaction, engagement, and overall usability of digital products.

3.1 PACT Analysis

PACT analysis is a framework used to understand and analyze the various factors that influence the design and implementation of a technology solution. PACT stands for People, Activities, Context, and Technologies. Each component represents a key aspect that needs to be considered during the analysis phase. By conducting a PACT analysis, designers and stakeholders gain a comprehensive understanding of the users, their activities, the context in which the technology will be used, and the available technologies. This analysis helps inform the design process and ensures that the technology solution is tailored to meet the needs, preferences, and constraints of the intended users and their specific context [2].

Table 3.1 shows the PACT Analyze of diabetic Foot Wound application for each component with factors.

Table 3.1 PACT Analyze Table

Screen	People				Activities								Context		Technology						
Screen	Demographics	Skill Level	Language (English)	Interaction Level (Simple)	Fill Out The Form	Select The Facts	Navigation	Readability(Easy)	Camera the wound	Sending Message	Receiving Message	Analyze	Physical Environment	Social Integration	Input Method	Output Method	Scrolling	Navigation	Responsive	Colors	Typography
Login Screen	Y	Y	Y	Y	Y	-	-	Y	-	-	-	-	Y	-	Y	-	Y	Y	Y	Y	Y
User Type Screen	-	-	Y	Y	-	Y	-	Y	-	-	-	-	-	-	Y	-	-	-	Y	Y	Y
Register Screen	Y	Y	Y	Y	Y	Y	-	Y	-	-	-	-	Y	-	Y	-	Y	Y	Y	Y	Y
Account Activation Screen	-	Y	Y	Y	-	-	-	Y	-	-	-	-	-	-	Y	-	Y	Y	Y	Y	Y
Forget Password Screen	Y	Y	Y	Y	Y	-	-	Y	-	-	-	-	Y	-	Y	-	Y	Y	Y	Y	Y
Verify Password Screen	-	Y	Y	Y	Y	-	-	Y	-	-	-	-	-	-	Y	-	Y	Y	Y	Y	Y
New Password Screen	-	Y	Y	Y	Y	-	-	Y	-	-	-	-	Y	-	Y	-	Y	Y	Y	Y	Y
Home Screen	Y	Y	Y	Y	-	Y	Y	Y	-	-	-	-	Y	-	Y	-	Y	Y	Y	Y	Y
Account Deactivation Screen	-	Y	Y	-	Y	Y	Y	Y	-	-	-	Y	-	-	Y	Y	Y	Y	Y	Y	Y
Account Reactive Screen	-	Y	Y	-	Y	Y	Y	Y	-	-	-	Y	-	-	Y	Y	Y	Y	Y	Y	Y
All Accounts Screen	-	Y	Y	-	Y	Y	Y	Y	-	-	-	Y	-	-	Y	Y	Y	Y	Y	Y	Y
DFW Capture Screen	Y	Y	Y	Y	Y	Y	Y	Y	Y	-	-	-	Y	-	Y	-	Y	Y	Y	Y	Y
DFW Details Screen	Y	Y	Y	Y	-	Y	Y	Y	-	-	-	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
DFW session Screen	Y	Y	Y	Y	Y	Y	Y	Y	Y	-	-	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Create Session screen	-	Y	Y	Y	Y	Y	Y	Y	-	-	-	-	-	-	Y	-	Y	Y	Y	Y	Y
Join Session Screen	Y	Y	Y	Y	Y	Y	Y	Y	-	-	-	-	Y	-	Y	-	Y	Y	Y	Y	Y
My Sessions Screen	Y	Y	Y	Y	-	Y	Y	Y	-	-	-	-	Y	-	Y	Y	Y	Y	Y	Y	Y
Session Details Screen	Y	Y	Y	Y	-	Y	Y	Y	-	-	-	-	-	-	Y	Y	Y	Y	Y	Y	Y
Select Patient Screen	-	Y	Y	Y	Y	Y	Y	Y	-	-	-	-	-	-	Y	-	Y	Y	Y	Y	Y
Patient's Report Screen	-	Y	Y	Y	Y	Y	Y	Y	-	-	-	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
DFW Analytics Screen	Y	Y	Y	Y	Y	Y	Y	Y	-	-	-	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Message Screen	Y	Y	Y	Y	-	Y	Y	Y	-	Y	Y	-	-	Y	Y	Y	Y	Y	Y	Y	Y
Chat Screen	Y	Y	Y	Y	Y	Y	Y	Y	-	Y	Y	-	-	Y	Y	Y	Y	Y	Y	Y	Y

3.1.1 People

Understanding the characteristics, behaviors, and needs of the users is a key aspect of the PACT analysis. By considering factors such as demographics, user goals, preferences, and skill levels, designers can tailor the interface to meet the specific requirements and expectations of the target audience. This user-centered approach ensures that the interface is intuitive, usable, and provides a positive experience for the users. By gaining insights into the users' characteristics and needs, designers can make informed decisions about the layout, functionality, and overall design of the interface, ultimately leading to a more successful and user-friendly product.

Table 3.2 : Factors of people

Actor	Demographics			Language	Skill Level	Interaction level
	Age	Gender	Education			
Patient	Between 20-79	Both	Normal	English	Good	Simple
Doctor	Every Doctors	Both	Intermediate	English	Better	
Admin	20 +	Both	Intermediate	English	Better	

- **Age**

  Selecting a patient age range between 20-79 is based on the fact that diabetes can affect individuals across various age groups. According to the IDF Diabetes Atlas (9th edition, 2019), the age-specific prevalence of diabetes is commonly reported for adults aged 20-79 years old. This age range allows for addressing the majority of diabetic patients who are likely to seek medical attention and use the app for managing their condition.

- **Gender**

  The inclusion of both genders (both) in the table recognizes that diabetes can affect both men and women. Design considerations should ensure that the interface is inclusive and accessible to users of all genders.

- **Language**

  The language factor specifies that the interface will be in English. While it's important to consider localization for different languages, English is often a widely used language in healthcare settings and can cater to a broad user base.

- **Skill Level**

Patients are described as having a good familiarity with technology, indicating that they are comfortable using digital tools. This suggests that the interface can incorporate more advanced features and interactions while still being user-friendly.

- **Education**

  The education level factor is not explicitly mentioned in the table, but it is generally understood that patients may have varying levels of education. Designing the interface with clear instructions, simple language, and appropriate information hierarchy can accommodate users with different education backgrounds.

- **Interaction level**

  The patient's interaction level is described as normal, indicating a regular level of engagement with the interface. This suggests that the interface should be straightforward and efficient for patients to input wound measurements, track progress, and access relevant information.

For doctors and administrators, the familiarity with technology is better, indicating a higher level of technological proficiency. Their interaction level is described as intermediate, suggesting that they may require more advanced features and capabilities for their specific tasks such as reviewing data, making treatment decisions, and managing the system.

3.1.2 Activities

Activities refer to the specific tasks, actions, or workflows that users perform within the interface. When designing an interface, it is crucial to analyze and understand the sequence of actions and goals that users aim to achieve. By examining the context in which these activities occur, designers can create interfaces that support users in efficiently and effectively completing their tasks. There are mainly consider seven Activity factors: Fill Out The Form, Select The Facts, Navigation, Readability(Easy), Camera the wound, Sending Message, Receiving Message, and Analyze.

- **Fill Out The Form**

  This activity factor represents the need for users to fill out forms with relevant information. In your case, users (patients, doctors, and admins) might need to provide details about the diabetic foot wounds, such as patient information, wound characteristics, and severity indicators. So, this factor would be relevant in screens where users input or update information related to the wounds.

- **Select The Facts**

  This activity factor refers to the selection or choice of specific information or options. In this app, users might need to select relevant facts or parameters during the classification or analysis process. For example, an unregistered user needs to select his correct user type that is a Patient Or Doctor, to register in this application. This factor would be applicable in screens where users make selections or choices.

- **Navigation**

  Navigation involves moving between different screens or sections of the app. Users, including patients, doctors, and admins, would navigate through various screens to access different features, such as viewing wound details, patient reports, analytics, and managing accounts. Therefore, navigation is an essential factor for all screens in your app.

- **Readability**

  Readability focuses on the ease of understanding and comprehending the information presented in the UI. Since your app deals with medical information and wound data, it is crucial to ensure that the text, labels, and instructions are clear and easy to read for users. This factor should be considered for all screens to enhance usability and accessibility.

- **Camera the Wound**

  This activity factor specifically relates to capturing images of the foot wounds using a camera. In your app, users might need to capture images of the diabetic foot wounds for analysis and documentation purposes. This factor would be applicable in screens where users can access the camera functionality to capture and upload images.

- **Sending Message**

  Sending messages involves the ability to communicate or exchange information within the app. In your case, this factor may be relevant for screens where users can send messages to doctors or admins, such as asking questions or providing additional information about the wounds.

- **Receiving Message**

  Receiving messages refers to the ability to receive and view incoming messages from other users. In your app, screens where users can view and respond to messages received from patients, doctors, or admins would include this activity factor.

- **Analyze**

  This factor represents the analysis or processing of data within the app. Since this app aims to classify diabetic foot wounds and segment the wound region mask with measurements, the analysis factor would be relevant in screens where algorithms or calculations are applied to the captured wound data to determine severity levels or generate analytics.

3.1.3 Context

The physical environment, social context, cultural context, and organizational context are crucial aspects to consider when designing and implementing technology.

- **Physical Environment**

  In the case of the diabetic foot wound classification and segmentation app, the physical environment refers to the settings where the technology will be used. This can include hospitals, clinics, or even the patient's home. Factors to consider include lighting conditions. Ensuring the app's interface is visually clear, legible, and adaptable to different lighting conditions is important.

- **Social Integration**

  Social integration can be achieved through various communication channels that enable interaction and collaboration between patients, doctors, and admin.

  - **Chatting with Doctors**

    By incorporating a chat feature, patients can directly communicate with their assigned doctors. This allows for real-time communication, enabling patients to ask questions, discuss concerns, seek medical advice, or provide updates on their condition. The chat feature promotes a more personalized and efficient way for patients to connect with their doctors, enhancing the overall patient-doctor relationship.

  - **Sharing Reports with Doctors**

    Patients can use the app to securely share their medical reports, including wound images, test results, or any other relevant documentation, with their doctors. This facilitates efficient information exchange and enables doctors to review and analyze the reports remotely. By allowing patients to easily share reports, the app promotes seamless communication and collaboration between patients and doctors.

- **Communication with Admins**

  In certain situations, patients and doctors may need to communicate with admins for administrative or logistical purposes. This could include requesting assistance, scheduling appointments, resolving issues, or seeking additional resources. By integrating communication channels, such as messaging or email functionalities, patients and doctors can interact with admins, ensuring effective and timely support and assistance.

3.1.4 Technologies

Technologies are essential to optimize the user interface (UI) for the specific devices and platforms that will be used.

- Input Methods

  Considering the input methods of the target devices is important for designing intuitive and user-friendly interfaces. For touch-based devices like smartphones and tablets, gestures and touch interactions can be incorporated into the UI design. These are technologies used in user interfaces for input.
  - Text Box
  - Drop-down Menus
  - Radio Buttons
  - Checkboxes
  - Buttons

- Output Method

  Output methods refer to the ways in which information or feedback is presented to the users of the diabetic foot wound app.

  - Push Notifications

    Push notifications are a method of output that involves sending messages or alerts to the user's device outside of the app. These notifications can be used to provide timely updates, reminders, or important information related to the app, such as scheduling wound assessments or providing access to new features or resources.

  - Wound Analytics

    Wound analytics involves analyzing the data and measurements obtained from the wound assessment and classification process. The app can utilize algorithms and data analysis techniques to generate meaningful insights and statistics about the

wounds. This can include metrics such as wound size, depth and healing progress. The analytics can be presented visually through charts, graphs, or numerical summaries to provide users, particularly doctors, with a comprehensive understanding of the wound characteristics and trends over time.

- Reports

  Reports provide a structured summary of the wound assessment and classification results. The app can generate detailed reports that include all the relevant information, such as wound images, measurements, severity classification, and any additional notes or comments. These reports can be customized to suit the needs of different users, such as doctors or administrators. The reports can be generated in various formats, including PDF or HTML, and can be easily shared or printed for documentation or further analysis.

- Typography

  Selecting appropriate font sizes is crucial for ensuring readability and accessibility of the app's content. Considerations should be made to accommodate users of varying age groups and visual abilities. Using a legible font with an appropriate size helps users easily read instructions, labels, and any textual information within the app. It's important to strike a balance between font size and available screen space, ensuring that the text remains readable without overcrowding the interface.

- Colors

  Color choices in the UI design impact the app's aesthetics, readability, and the conveyance of important information. For the diabetic foot wound app, it's important to select colors that promote visual clarity and ease of interpretation. For example, using contrasting colors between text and background ensures readability. Additionally, colors can be employed to highlight important elements, such as buttons or notifications, and to convey different states or categories within the app. Considering any color preferences or cultural associations related to medical contexts can further enhance the user experience.

- Scrolling and Navigation

  Smartphone apps often require vertical scrolling due to limited screen space. Designers should optimize the scrolling experience, ensuring that important information is presented in a logical and organized manner. Navigation menus, tabs, or collapsible sections can be implemented to provide easy access to different app features and content without overwhelming the screen with too many options at once.

- Responsive Design

  Given the wide range of smartphone screen sizes available, it's crucial to design the app to be responsive. Responsive design ensures that the app's user interface adapts and adjusts seamlessly to different screen sizes. This allows users to have a consistent and optimal experience, regardless of whether they are using a smaller screen on a compact smartphone or a larger screen on a phablet or tablet.

3.2 UI Design consideration and approaches

When designing the user interface (UI) for the diabetic foot wound classification and segmentation app, there are several considerations and approaches that can enhance the user experience. Here are some key UI design considerations and approaches for this case:

- **User-Centric Design**

  Prioritize user needs, goals, and preferences throughout the design process. Conduct user research and gather feedback to understand their expectations, workflows, and pain points. Design the UI to be intuitive, user-friendly, and focused on fulfilling their requirements effectively.

- **Clear and Consistent Layout**

  Create a clean and organized layout that presents information in a logical and structured manner. Use a consistent visual hierarchy, grouping related elements together, and employing grid systems to ensure a cohesive and balanced interface. Clear and concise labels and instructions help users understand the app's functionalities.

- **Intuitive Navigation**

  Design an intuitive navigation system that allows users to easily navigate through different screens and features of the app. Use familiar navigation patterns such as bottom navigation bars, tabs, or a hamburger menu for accessing different sections of the app. Ensure that navigation elements are easily recognizable and accessible.

- **Visual Design and Branding**

  Create a visually appealing and consistent design that reflects the app's purpose and brand identity. Choose appropriate colors, typography, and graphical elements that align with the medical context while considering accessibility guidelines. Visual elements should not only enhance the aesthetics but also improve readability and usability.

- **Contextual Help and Guidance**

  Provide contextual help and guidance to assist users in understanding the app's features and functionalities. Include tooltips, onboarding screens, or contextual pop-ups to guide users through complex processes or provide additional information when needed. Help documentation or a dedicated FAQ section can also be included to address common user questions.

- **Responsive Design**

  Ensure the UI is responsive and adaptable to different screen sizes and orientations. Optimize the layout, font sizes, and touch targets to accommodate various Android devices, including smartphones and tablets.

- **Usability Testing and Iteration**

  Conduct usability testing with representative users to identify areas of improvement and validate the effectiveness of the UI design. Collect feedback and iterate on the design based on user insights to refine the app's interface.

3.3 Design tools, techniques, templates

In the world of UI design, utilizing the right tools, techniques, and templates is crucial for creating effective and visually appealing interfaces.

3.3.1 Design tools

- **Figma**

  Figma's cloud-based collaboration, cross-platform compatibility, robust design and prototyping features, component-based design approach, design system support, developer handoff capabilities, and integration options make it a preferred choice for many UI designers. Its user-friendly interface and continuous updates based on user feedback further contribute to its reputation as one of the best design tools available.

- **Adobe Photoshop**

  Adobe Photoshop is primarily used for editing and manipulating digital images. It offers a wide range of features and tools for tasks such as retouching photos, adjusting colors and lighting, cropping and resizing images, and creating complex compositions. Photoshop is known for its powerful image editing capabilities, including layers, masks, filters, and various selection tools. It is widely used by photographers, digital artists, and graphic

designers for tasks ranging from simple photo enhancements to advanced digital artwork creation.

- **Adobe Illustrator**

  Adobe Illustrator, on the other hand, is a vector graphics editor. It is specifically designed for creating and editing scalable vector-based artwork. Unlike raster-based images (such as those created in Photoshop), vector graphics are composed of mathematical equations that define shapes, lines, and colors. This allows vector graphics to be scaled infinitely without losing image quality. Illustrator provides a range of tools for creating and manipulating vector-based objects, such as drawing paths, applying gradients, and working with typography. It is widely used for creating logos, illustrations, icons, and other graphics that require precise and scalable designs.

3.3.2 Design Techniques

- **Wireframing**
  Creating low-fidelity wireframes to outline the basic layout and structure of the interface. This helps in visualizing the placement of key elements and content without focusing on visual details.

- **Prototyping**
  Building interactive prototypes to simulate the app's functionality and user interactions. Prototypes allow for user testing and validation of the design before development.

3.3.3 Design Template

- **Material Design**
  Google's design system that provides guidelines, components, and templates for creating visually consistent and intuitive Android interfaces. Material Design offers pre-designed UI components and layout templates for different screen sizes and types of app screens.

- **Bootstrap**
  A popular front-end framework that provides pre-designed UI components and responsive grid layouts. Bootstrap offers templates and themes that can be customized to create a consistent and responsive design across different devices

3.4 Input Design Aspect

Figure (3.1 , 3.2 , 3.3) displays the UI interfaces of the Input design aspect which are the wound capture screen, the patient's sign-up screen, and the create session screen

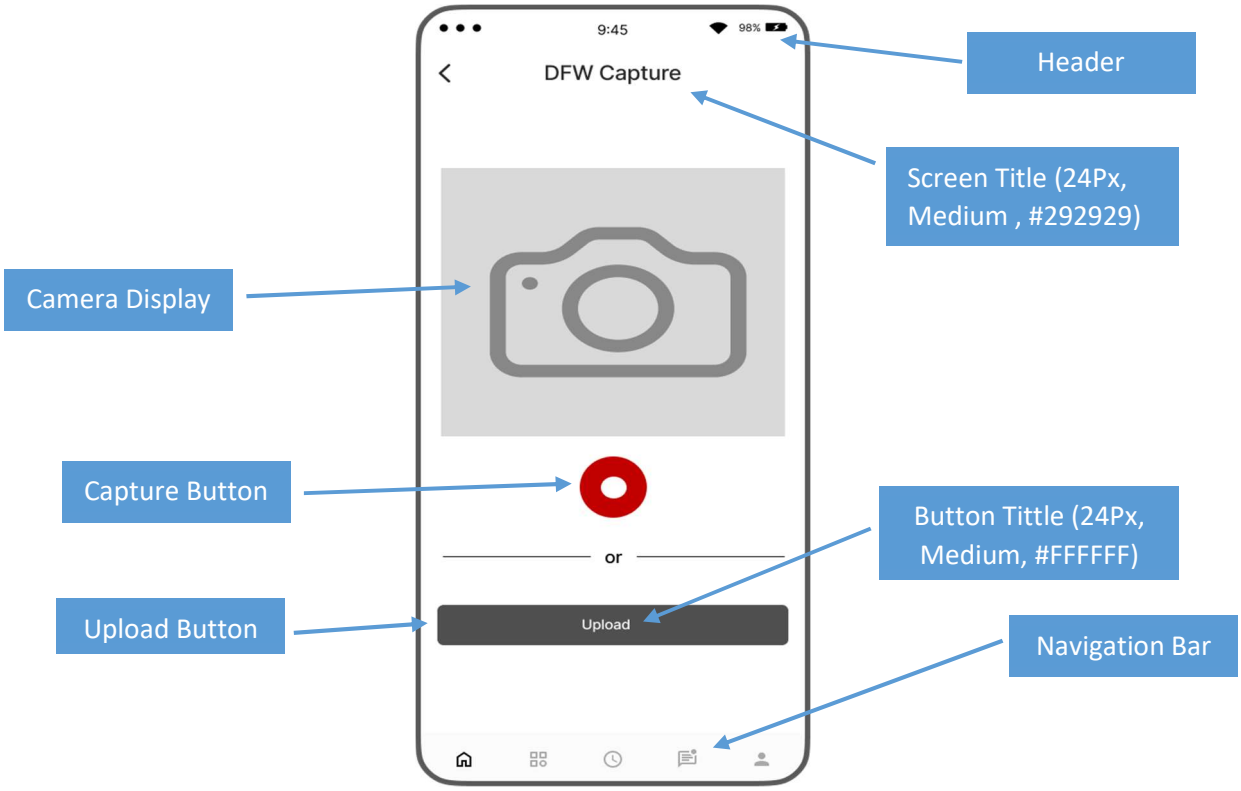


Figure 3.1 DFW Capture form

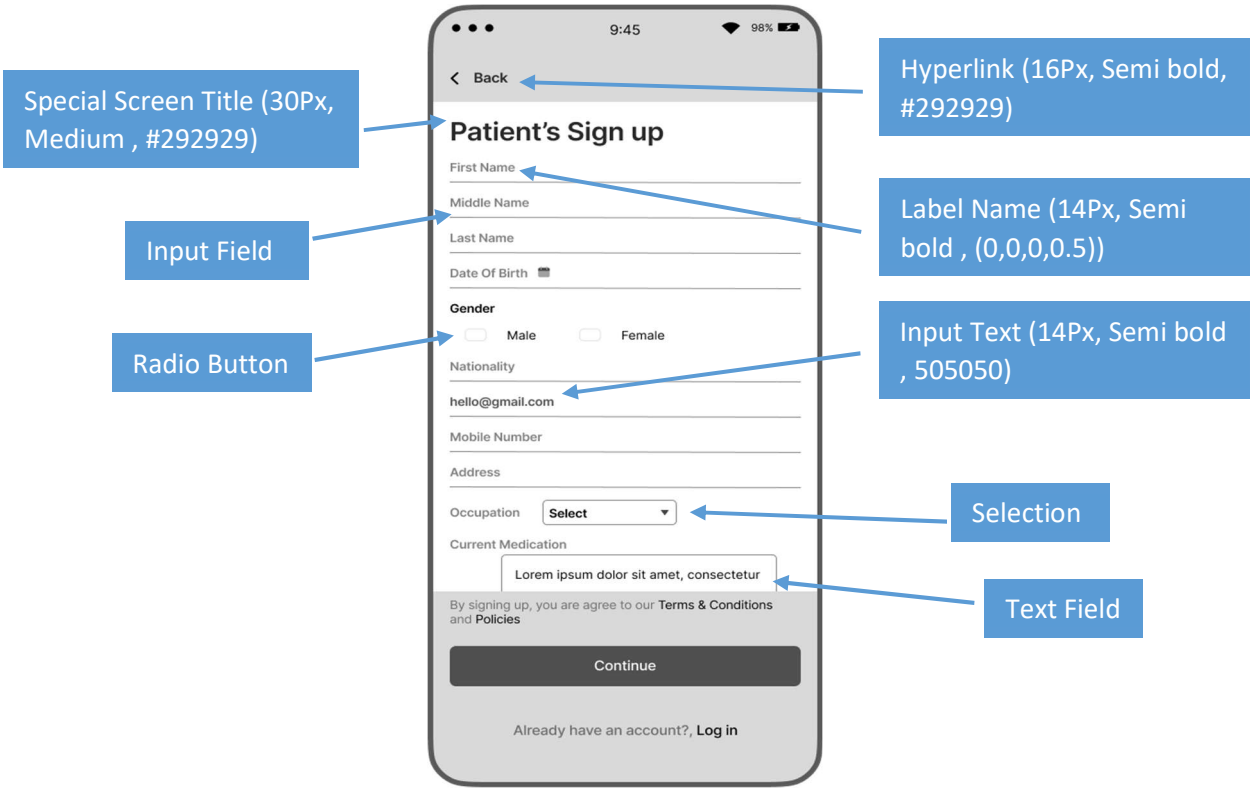


Figure 3.2 : Patient's Signup form

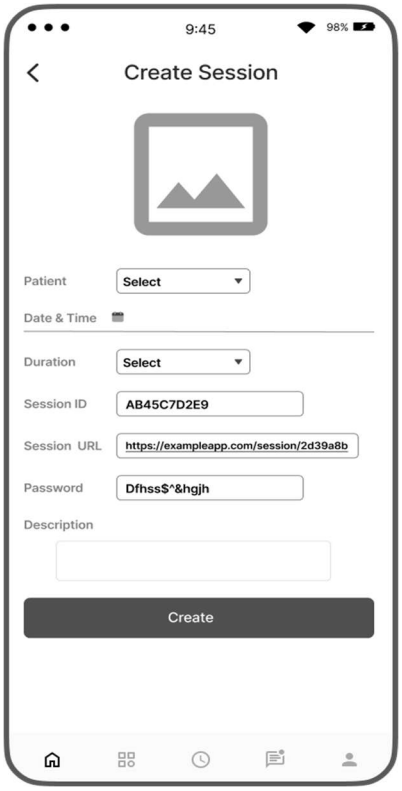


Figure 3.3 : Create Session Form

3.5 Output Design Aspect

Figure (3.4, 3.5, 3.6) displays the UI interfaces of the Output design aspect which are the account information screen, the DFW details screen and the DFW analytics screen

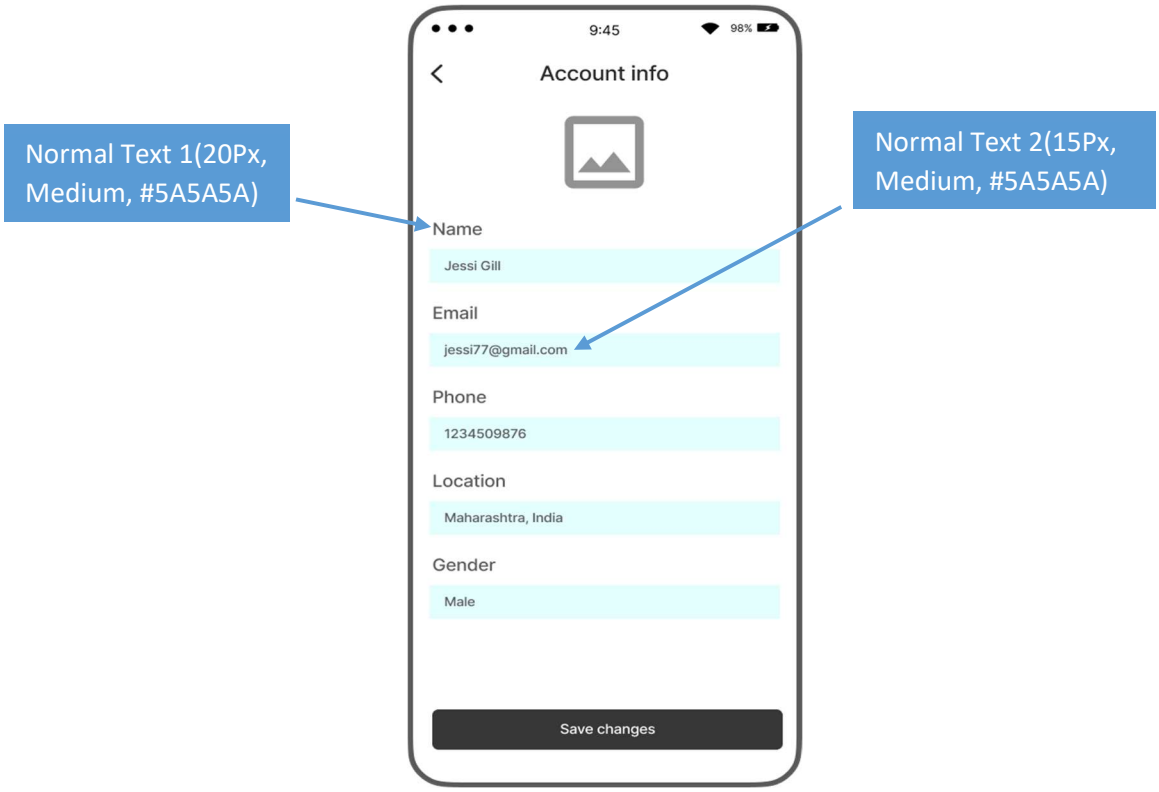


Figure 3.4 : Account Information Report

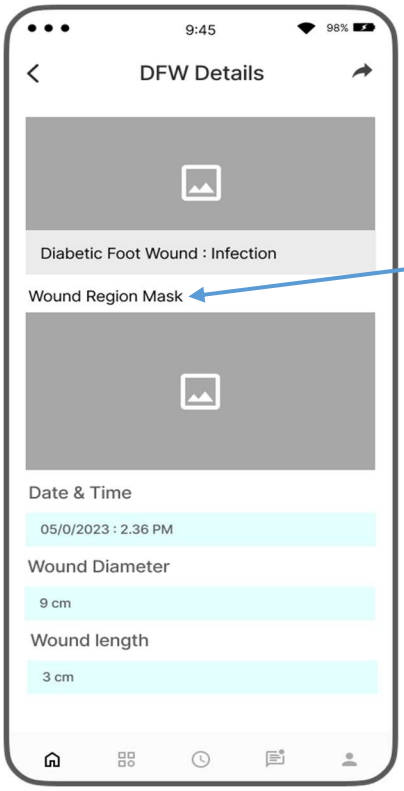


Figure 3.5 : DFW Details Report

Sub Title (18Px, Regular, #000000)

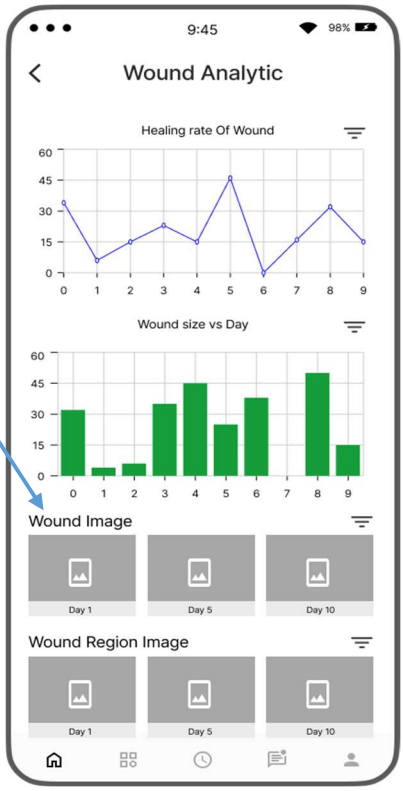


Figure 3.6 : Wound Analytics Report

3.6 Dialogue Design Aspect

This section displays the dialogue design aspect of the application. Figure (3.9-3.10) display the dialogue of the DFW video session and figure (3.11-3.13) displays the dialogue of chat screen

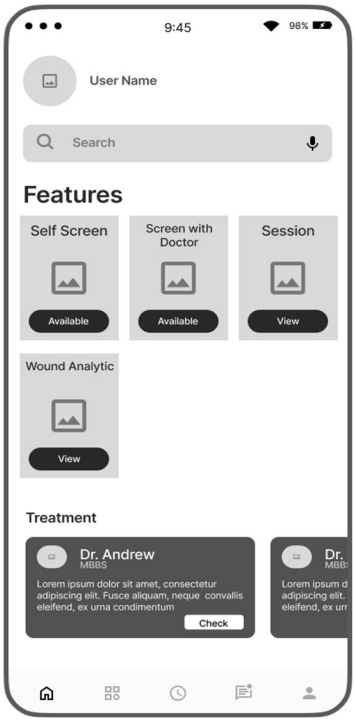


Figure 3.9 : Home Screen

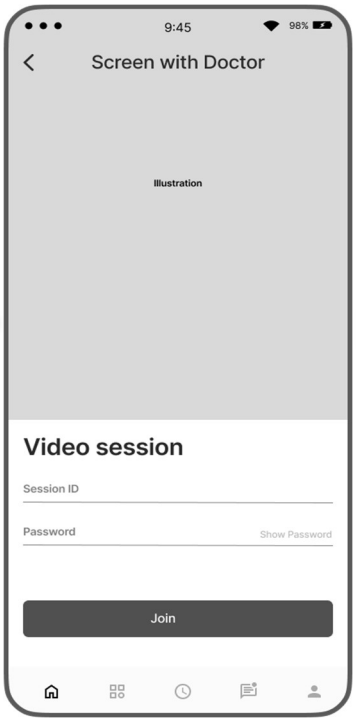


Figure 3.8 : Join session screen

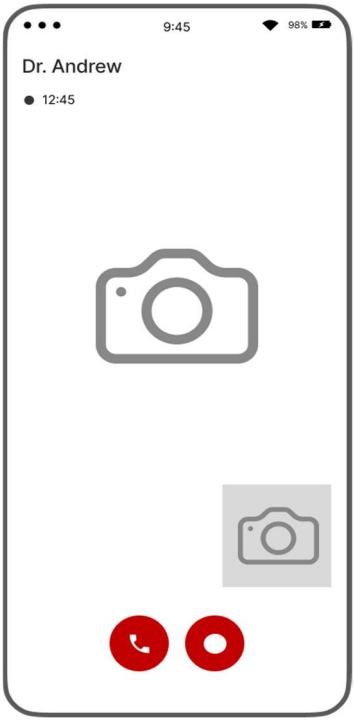


Figure 3.7 : Video Session Screen

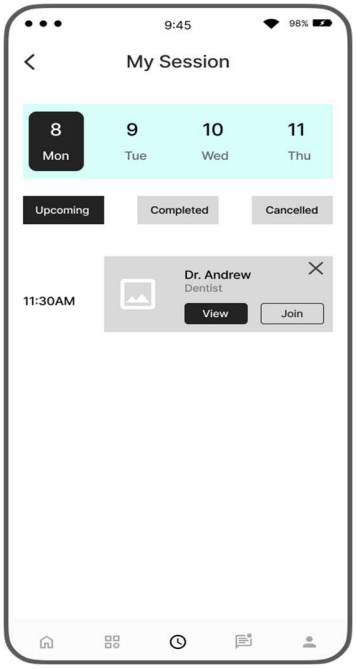


Figure 3.10 : My Session Screen

Figure 3.13 : Home screen

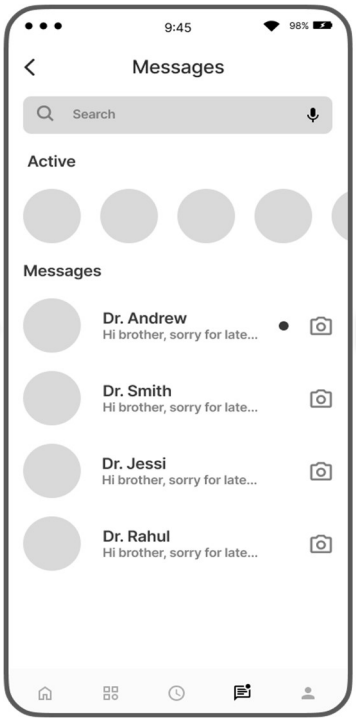


Figure 3.12 : Message screen

Figure 3.11 : Chat screen

3.7 Hosting/ installation environment

The diabetic foot wound app, which is based on deep learning, requires a web server hosting environment that consists of a SQL database, classification model, and segmentation model. This hosting environment ensures the storage, retrieval, and management of patient data, as well as the deployment of the trained models for classification and segmentation tasks.

The app is designed to be compatible with any Android device running version 5 or above. To ensure optimal performance, the device should have a minimum camera resolution of 12 megapixels. Additionally, a minimum of 3GB RAM is recommended to ensure smooth and responsive operation of the application without any noticeable delays or performance issues.

By considering these specifications for the hosting and installation environment, the app can provide efficient data processing, accurate classification results, and seamless user experience on Android devices meeting the specified requirements.

CHAPTER 4 - DATA MANAGEMENT

Our project's data management is a key component. We will keep the essential information, such as photos of diabetic foot wounds datasets. Since we are managing our project in accordance with these specifics, we must take into account a substantial amount of data. Our datasets must be kept in a strong database system. To access, store, and retrieve data from the database more efficiently and precisely, several strong data management approaches were needed. The data management of our system utilizing EER diagrams and the database design will be covered in this chapter.

4.1 Data requirement

The main purpose of the data management section is to gather information on wound classification type and segments of diabetic foot wounds and store it in the database. The images that we get as inputs will be handled to image processing techniques, and the results will be compared to the data kept in the database. It is necessary to thoroughly fulfill the data requirements in order to acquire these accurate outcomes. Below is a list of the data required for the project.

- Store the insert, delete, and update data of doctors and patients details.
- Store the details of the mail notification and their attributes.
- Store the details whether it is self-screening or video-screening.
- Store the details of the wounds (wound classification & wound segmentation).
- Store the details of the video session to be recommended for each Diabetic Foot Wound (DFW).

4.2 Design tools, techniques

4.2.1 Design tools

- **MySQL**

  Since its goal is to effectively store, organize, and manipulate structured data, **MySQL** is used for data management. It offers performance and scalability for managing massive data volumes and supporting applications with significant traffic. MySQL protects sensitive data with strong security measures. It is well-liked among developers because of its simplicity, interoperability, and integration possibilities. Since MySQL is an open-source database, it is inexpensive and does not require the high licensing fees associated with commercial databases.

- **Visual paradigm**

  Visual paradigms in DBMS are used for database design (conceptual database design,logical database design & physical database design), query building. Using visual paradigms in database design and management helps improve understanding, communication, maintenance, and documentation of the database structure and relationships. It enhances collaboration and facilitates effective decision-making when working with complex data systems.

4.2.2 Design technique

- **Normalization**

  Normalization is a technique used in database design to eliminate redundancy and improve data integrity. Normalization reduces data duplication, ensures consistency, and simplifies data maintenance and updates.

4.3 Conceptual database design

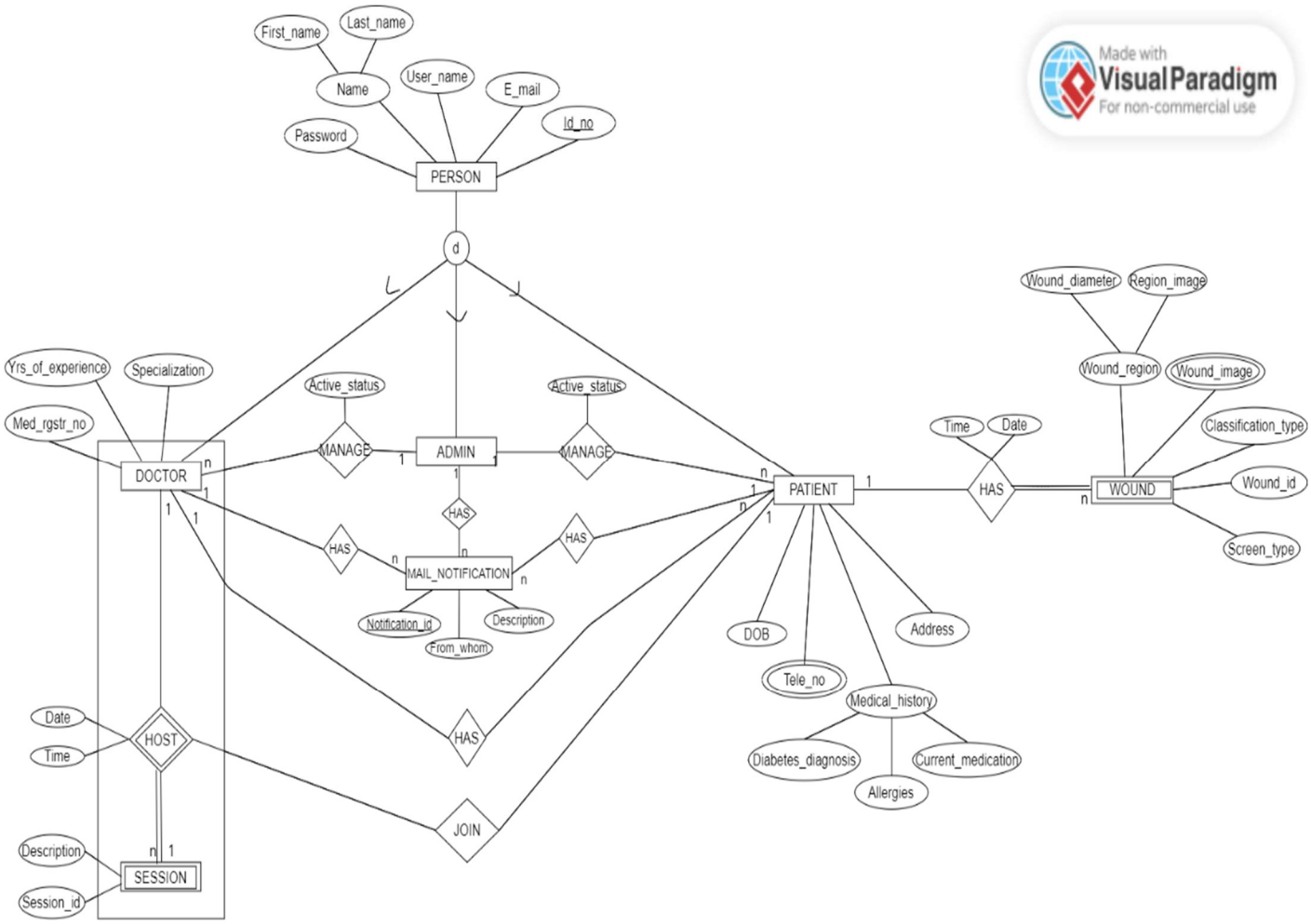


Figure 4.1 : EER Diagram

4.4 Logical database design

There are 8 standard steps in logical database design.

Map regular entities
Map weak entities
Map binary relationships
Map associated entities
Map unary relationships
Map ternary(and n-ary) relationships
Map super type/sub types
Map aggregation

But we are using 5 steps in our data management.

Below, we put doc_id, patient_id, admin_id as the primary key of doctor, patient, admin, which means (it will be displayed), and the one put for person is id_no.

4.4.1 Map Regular Entities

- Consider PERSON strong entity,

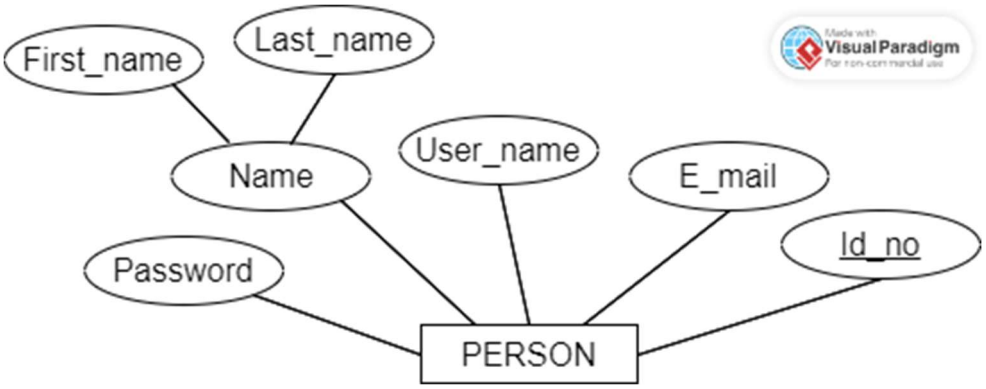


Figure 4.2 person entity

PERSON (Id_no, E_mail,Password,First_name,Last_name,User_name)
Primary key: Id_no

Composite attribute

PERSON(Id_no,
First_name,Last_name)

Primary key: Id_no

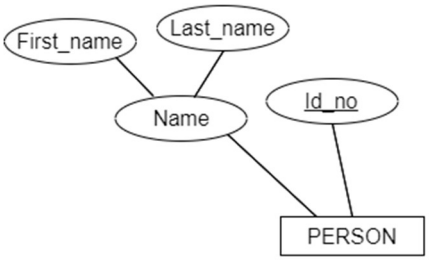


Figure 4.3 : Composite attribute

- Consider DOCTOR strong entity,

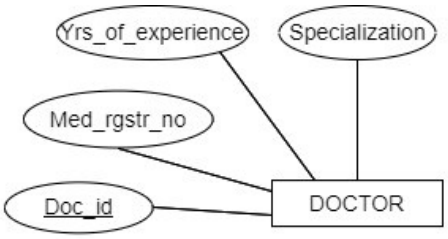


Figure 4.4 : Doctor entity

DOCTOR (Doc_id,Med_rgstr_no,Yrs_of_experience,Specialization)
Primary key: Doc_id

- Consider ADMIN strong entity,

Figure 4.5 : Admin Entity

ADMIN (Admin_id)
Primary key: Admin_id

- Consider PATIENT strong entity,

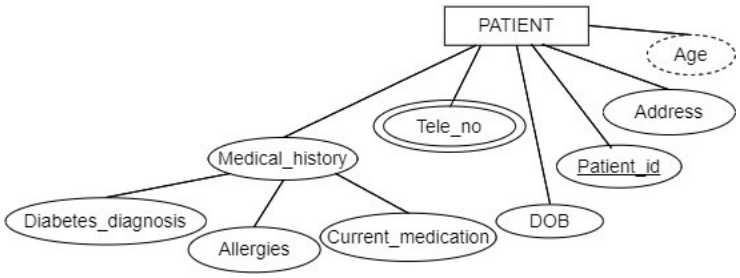


Figure 4.6 : Patient entity

PATIENT(Patient_id,Address,DOB,Diabetic_diagnosis,Current_medication,Allegies)
Primary key: Patient_id

Composite attribute

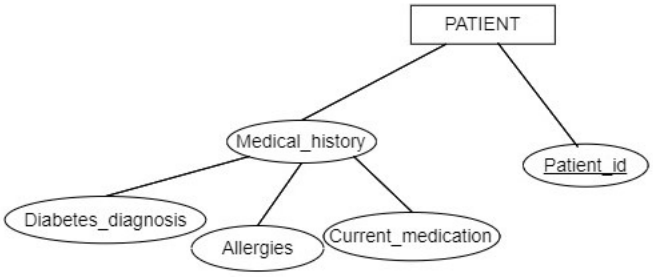


Figure 4.7 : Composite attribute

PATIENT(Patient_id,Diabetic_diagnosis,Current_medication,Allegies)
Primary key: Patient_id

Telephone number is the multiple attributes.Therefore It use to create new table.

PATIENT_TELE_NO (Patient_id,Tele_no)
Primary key: Patient_id,Tele_no

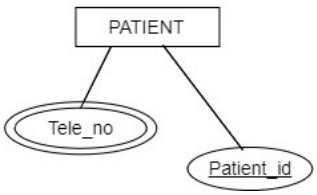


Figure 4.8 : Multiple attribute

- Consider MAIL_NOTIFICATION strong entity,

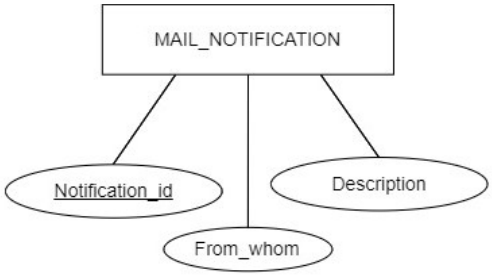


Figure 4.9 : Notification entity

MAIL_NOTIFICATION (Notification_id,From_whom_Descriptopn)
Primary key: Notificaton_id

4.4.2 Map Weak Entities

- Consider SESSION weak entity,

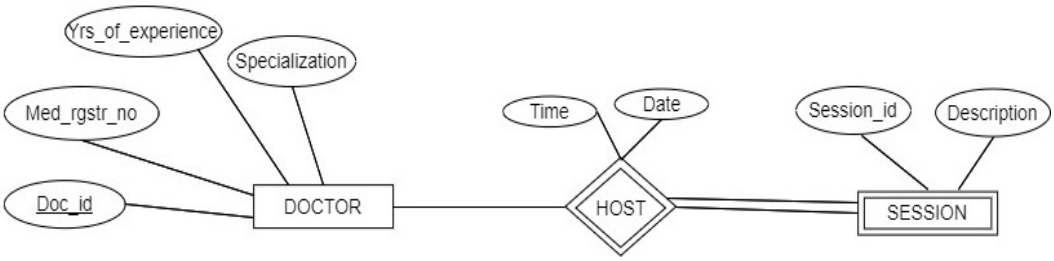


Figure 4.10 : Doctor and session relationship

SESSION is a weak entity.

SESSION(Doc_id,Session_id,Description,)

Primary key: Doc_id,Session_id

- Consider WOUND weak entity,

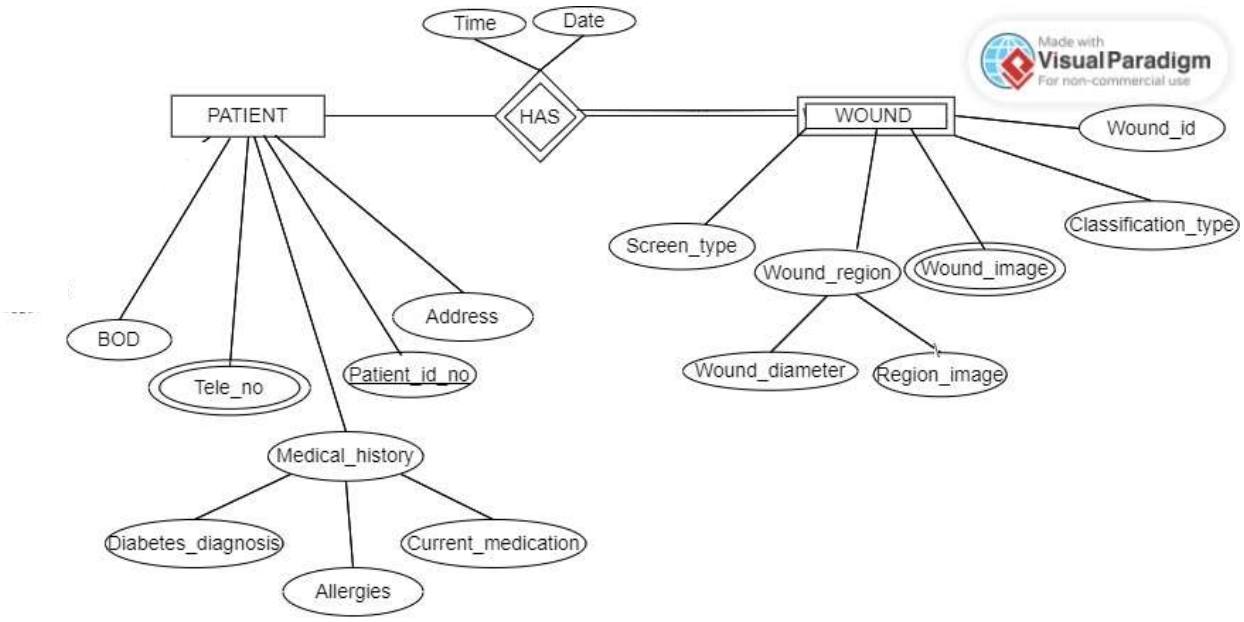


Figure 4.11 : Patient and wound relationship

WOUND is a weak entity,

WOUND(Patient_id,Wound_id,Screen_type,Classification_type,Wound_diameter,Region_image)
Primary key: Patient_id,Wound_id

Wound_image is a multiple attribute.Therefore we have to create another table.

WOUND_WOUND_IMAGE(Patient_id,Wound_id,Wound_image)
Primary key: Patient_id,Wound_id

4.4.1 Map Binary Relationships

Our EER diagram has lot of one to many(1:N) relationships. But it includes one to one relationships(1:1) and doesn't include many to many(M:N) relationships.

- **Map binary one to many binary relationships**

  - Consider only DOCTOR & ADMIN entities,

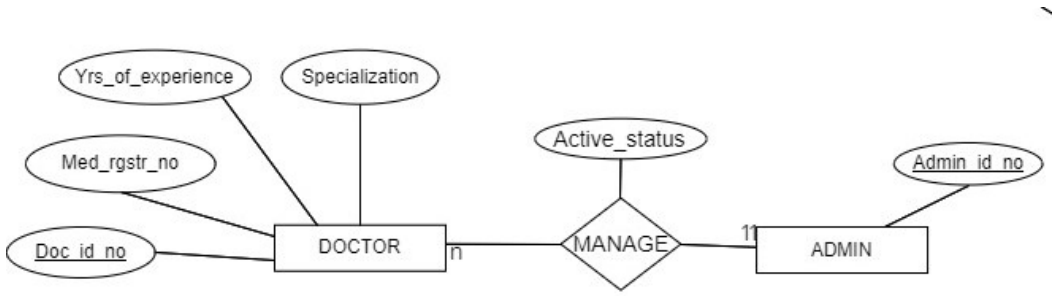
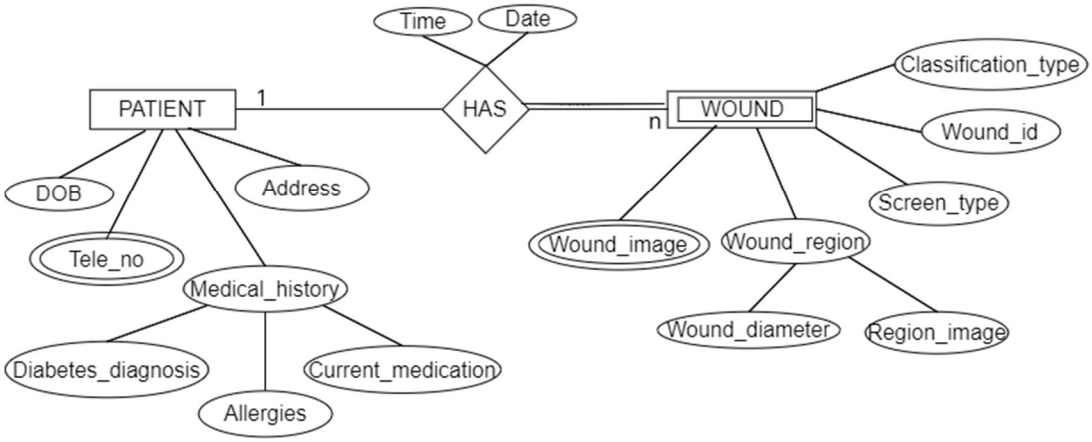


Figure 4.13: Doctor and admin relationship

DOCTOR (Doc_id,Med_rgstr_no,Yrs_of_experience,Specialization, Active _status,Admin_id_no)
Primary key: Doc_id

ADMIN (Admin_id)
Primary key: Admin_id

- Consider only PATIENT & WOUND entities,



WOUND(Patient_id,Wound_id,Screen_type,Classification_type,Wound_diameter,Region_image,Date,Time)

Figure 4.14 :Patient and wound relationship

(Here wound entity is a weak entity. Because primary key is Patient_id,Wound_id (in the above figure 4.3))

PATIENT (Patient_id,Address,DOB,Diabetic_diagnosis,Current_medication,Allegies)
Primary key: Patient_id

PATIENT_TELE_NO (Patient_id,Tele_no)
Primary key: Patient_id_no,Tele_no

- Consider only PATIENT & MAIN_NOTIFICATION entities,

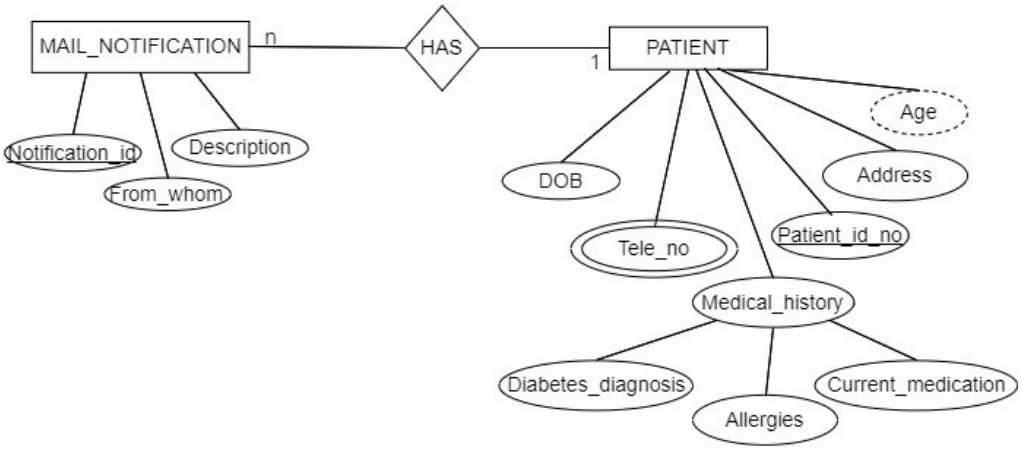


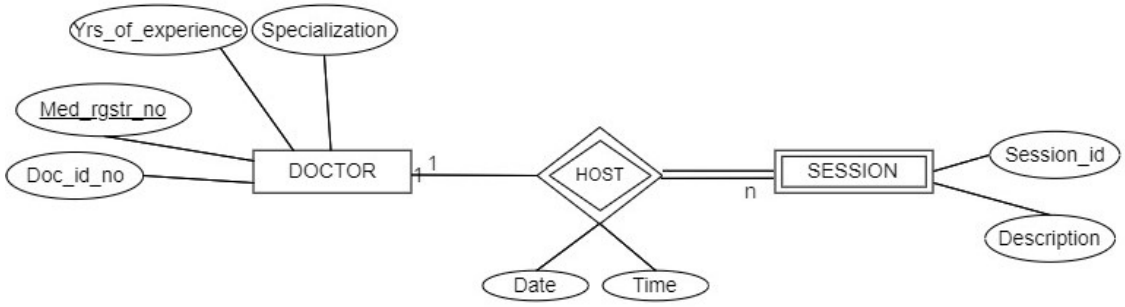
Figure 4.15: Mail notification and patient relationship

PATIENT (Patient_id,Address,DOB,Diabetic_diagnosis,Current_medication,Allegies)
Primary key: Patient_id

MAIL_NOTIFICATION (Notificaton_id,From_whom_Descriptopn,Patient_id)
Primary key: Notificaton_id

PATIENT_TELE_NO (Patient_id_no,Tele_no)
Primary key: Patient_id_no,Tele_no

- Consider only DOCTOR & SESSION entities,



- *Figure 4.16: Doctor and session relationship*

DOCTOR (Doc_id,Med_rgdtr_no,Yrs_of_experience,Specialization,)
Primary key: Doc_id

SESSION(Doc_id,Session_id,Description Date,Time)
(Here SESSION entity is a weak entity. Because primary key is Doc_id,Session_id (in the above figure 4.3))

- Consider only DOCTOR & MAIL_NOTIFICATION entities,

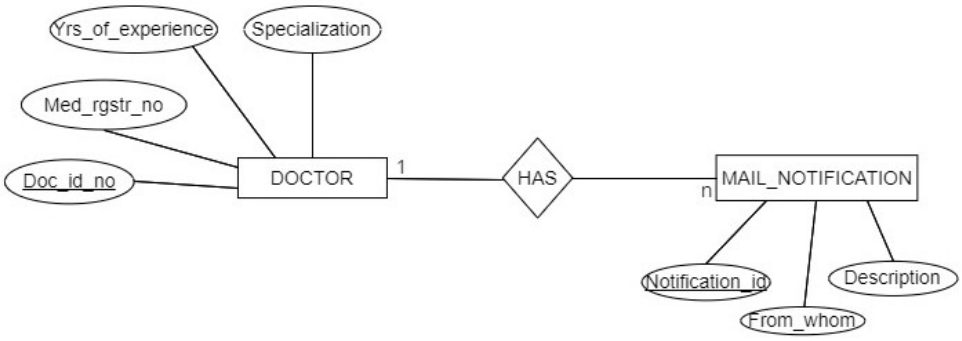


Figure 4.11 : Doctor and mail notification relationship

DOCTOR (Doc_id,Med_rgstr_no,Yrs_of_experience,Specialization)
Primary key: Doc_id

MAIL_NOTIFICATION (Notificaton_id,From_whom_Descriptopn,Doc_id)
Primary key: Notificaton_id

- Consider only ADMIN & MAIL_NOTIFICATION entities,

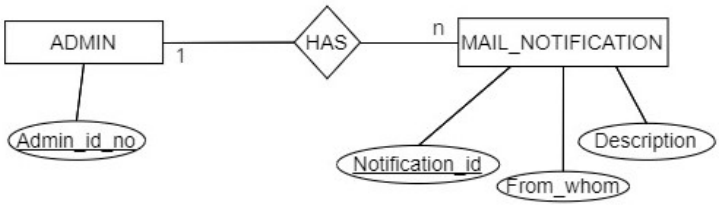


Figure 4.12 : Admin and Mail notification relationship

MAIL_NOTIFICATION(Notificaton_id,From_whom_Descriptopn,Admin_id)
Primary key: Notificaton_id

ADMIN(Admin_id)
Primary key: Admin_id

4.4.3 Map associated entities

There are no any map associated relationships in EER diagram. Therefore skip this step.

4.4.4 Map unary relationships

There are no any map unary relationships in EER diagram. Therefore skip this step.

4.4.5 Map ternary(and n-ary) relationships

There are no any map ternary(and n-ary) relationships in EER diagram. Therefore skip this step.

4.4.6 Map super type/ subtype relationships

This is the diajoint relationship. We can called it "generalization". Here super class is a PERSON and Sub classes are DOCTOR ,PATIENT, ADMIN

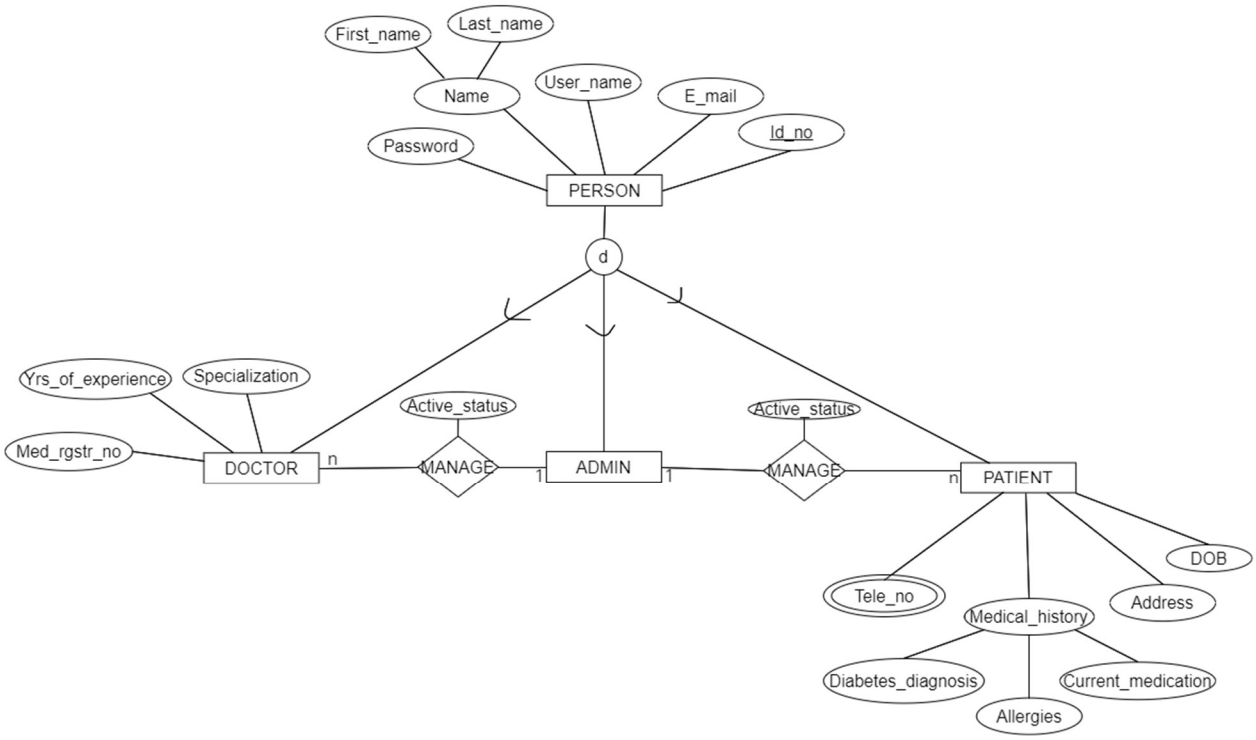


Figure 4.13 : Person generalization

PERSON (Id_no,E_mail,Password,First_name,Last_name,User_name)
Primary key: Id_no

PATIENT(Patient_id,Address,DOB,Diabetic_diagnosis,Current_medication,Allegies,)
Primary key: Patient_id

ADMIN (Admin_id)
Primary key: Admin_id

DOCTOR (,Doc_id,Med_rgstr_no,Yrs_of_experience,Specialization,)
Primary key: Doc_id

4.4.7 Map aggregation

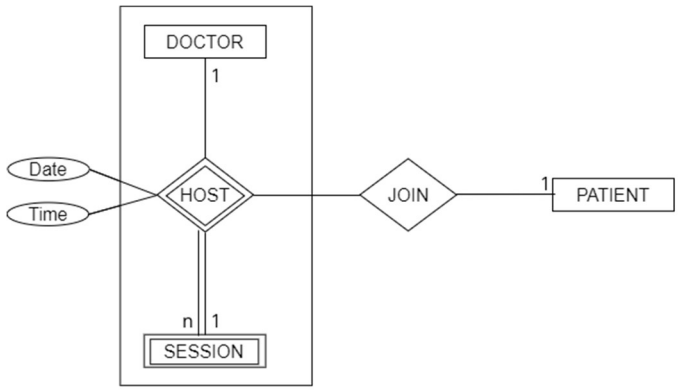


Figure 4.14 : Aggregation relationship

According to all the attributes here,

PATIENT(Patient_id,Address,DOB,Diabetic_diagnosis,Current_medication,Allegies)
Primary key: Patient_id

DOCTOR(Doc_id,Med_rgstr_no,Yrs_of_experience,Specialization,Active_status,)
Primary key: Doc_id

SESSION(Doc_id,Session_id,Description,Patient_id,Date,Time)
Primary key: Doc_id,Session_id
Foreign key: Doc_id

JOIN(Doc_id,Session_id, P,Patient_id)
Primary key: Doc_id,Session_id, P,Patient_id

4.5 Schema refinement

First, let's identify the functional dependencies for each tables. After that, data is normalized in 3 steps.

- 1st Normal Form (1NF):The provided table already satisfies 1NF as it contains a primary key (P_code) and each column holds atomic values. There are no repeating groups.Insertion, deletion, and updating anomalies do not apply to 1NF.

- 2nd Normal Form (2NF):Since there is only one non-key attribute set in the PERSON table, there are no partial dependencies. Therefore, it automatically satisfies 2NF.

- 3rd Normal Form (3NF):The PERSON table also satisfies 3NF because there are no transitive dependencies present. All non-key attributes are directly dependent on the primary key (P_code).

If since the provided tables already satisfies 1NF, 2NF, and 3NF, there is no need for further schema refinement.After that we can get all refine tables.

4.5.1 PERSON table

Table 4.1 : Person table

Id_no	E_mail	User_name	Password	First_name	Last_name
A001	john@exmple.com	Nirmal	=23asb	Namal	Perera
A002	jane@example.com	abc456	Jane#53	Smith	Janesmith
D003	michael@example.com	xyz789	M4%4el	Johnson	mjohnson

- **Functional Dependency**
  Id_no-> E_mail, User_name,Password, First_name, Last_name

- **Normalization**
  Table already satisfies 1NF, 2NF, and 3NF, there is no need for further schema refinement.

4.5.2 ADMIN table

Table 4.2 : Admin table

Admin_id
A001
A002
A003

- **Normalization**
  Table already satisfies 1NF, 2NF, and 3NF, there is no need for further schema refinement.

4.5.3 PATIENT table

Patient_id	Address	DOB	Diabetic_diagnosis	Current_medication	Allergies	Admin_id	Active_status	Doc_id
P001	123 Main St	1980-01-01	Yes	Insulin	Penicillin	A001	Active	D001
P002	456 Oak Ave	1975-05-15	No	Metformin	None	A002	Active	D002
P003	789 Elm Rd	1992-09-30	Yes	None	Latex	A003	Inactive	D003

Table 4.3 Patient table

- **Functional Dependency**
  Patient_id -> Address,DOB, Diabetic_diagnosis, Current_medication, Allergies,Admin_id,Active_status,Doc_id

  We can create,
  Med_details_no,Patient_id-> Diabetic_diagnosis, Current_medication, Allergies

- **Normalization**
  The attributes Diabetic_diagnosis, Current_medication, Allergies depend on the Med_details_no.To eliminate the transitive dependency, we can create a new table called PATIENT_MEDICAL_DETAILS with the attributes

Patient_id,Med_details_no,Diabetic_diagnosis, Current_medication, Allergies.Now we can get new refine table,

PATIENT_MEDICAL_DETAILS table

Table 4.4 : Patien_medical_details table

Patient_id	Med_details_no	Diabetic_diagnosis	Current_medication	Allergies
P001	MED001	Yes	Insulin	Penicillin
P002	MED002	No	Metformin	None
P003	MED003	Yes	None	Latex

PATIENT table

Table 4.5 : New patient table

Patient_id	Address	DOB	Admin_id	Active_status	Med_details_no	Doc_id
P001	123 Main St	1980-01-01	A001	Active	MED001	D001
P002	456 Oak Ave	1975-05-15	A002	Active	MED002	D002
P003	789 Elm Rd	1992-09-30	A003	Inactive	MED003	D003

4.5.4 PATIENT_TELE_NO table

Table 4.6 : Patient_tele_no table

Patient_id	Tele_no
P001	0751234567
P001	0779876543
P003	0785555555

- **Functional Dependency**
  Patient_id -> Tele_no

- **Normalization**
  The table does not have any repeating groups or multivalued dependencies, so it is already in 1NF.Since there is only one non-key attribute in the table, there are no partial dependencies. Therefore, the table is already in 2NF.There are no transitive dependencies in the table, so it is already in 3NF.No transitive dependencies exist in this table.

4.5.5 DOCTOR table

Table 4.7 : Doctor table

Doc_id	Med_rgstr_no	Yrs_of_experience	Specialization	Admin_id	Active_status
D001	M001	10	Cardiology	A001	Active
D002	M002	5	Dermatology	A002	Active
D003	M003	8	Pediatrics	A003	Inactive

- **Functional Dependency**
  Doc_id -> P-code, Med_rgstr_no,Yrs_of_experience,Specialization, Admin_no,Active_status

  Med_rgstr_no -> Yrs_of_experience

- **Normalization**
  The attributes Yrs_of_experience depend on the Med_rgstr_no.To eliminate the transitive dependency, we can create a new table called DOCTOR_EXPERIENCE with the attributes Med_rgstr_no,Yrs_of_experience.

Now we can get new refine tables,

DOCTOR_EXPERIENCE table

Table 4.8 : Doctor_experience table

Med_rgstr_no	Yrs_of_experience
M001	10
M002	5
M003	8

DOCTOR table

Table 4.9 : New doctor table

Doc_id	Specialization	Admin_id	Active_status	Med_rgstr_no
D001	Cardiology	A001	Active	M001
D002	Dermatology	A002	Active	M002
D003	Pediatrics	A003	Inactive	M003

4.5.6 WOUND table

Table 4.10 : Wound table

Patient_id_no	Wound_id	Screen_type	Classification_type	Wound_diameter	Region_image	Date	Time	Doc_id
P001	W0001	video	Schema	5	image1.jpg	2023-06-01	10:00 AM	D001
P002	W0002	self	Inspection	7	image2.jpg	2023-06-02	11:30 AM	D002
P003	W0003	self	Schema	6	image3.jpg	2023-06-03	02:15 PM	D003

- **Functional Dependency**
  Patient_id, Wound_id -> Screen_type, Classification_type, Wound_diameter, Region_image, Date,Time

  We can create,
  Scrn_id_ classi_id -> Screen_type, Classification_type

  Reion_id -> Wound_diameter,Region_image

- **Normalization**
  The attributes Screen_type, Classification_type depend on the Scrn_id_ classi_id. To eliminate the transitive dependency, we can create a new table called CLASSIFICATION_SCREEN_TYPE with the attributes Scrn_id_ classi_id ,Screen_id_ classi_id.

  The attributes Wound_diameter,Region_imagedepend on the Region_id. To eliminate the transitive dependency, we can create a new table called REGION with the attributes Region_id,Wound_diameter,Region_image.

  Now we can get new refine table,

Table 4.11 : Classification_screen_type table

Scrn_id_ classi_id	Screen_type	Classification_type
SC001	Video	Inspection
SC002	Video	Inspection
SC003	Self	Schena
SC004	Self	Schema

REGION table

Table 4.12 : Region table

Region_id	Wound_diameter	Region_image
R0001	5	image1.jpg
R0002	7	image2.jpg
R0003	6	image3.jpg

WOUND table

Table 4.13 : New wound table

Patient _id	Wound _id	Date	Time	Doc_id	Scrn_id_ classi_id	Region_id
P001	W0001	2023-06-01	10:00 AM	D001	SC001	R0001
P002	W0002	2023-06-02	11:30 AM	D002	SC002	R0002
P003	W0003	2023-06-03	02:15 PM	D003	SC003	R0003

4.5.7 WOUND_WOUND_IMAGE table

Table 4.14 : Wound_wound_image table

Patient_id	Wound_id	Wound_image
P001	W0001	image1.jpg
P002	W0002	image2.jpg

- **Functional Dependency**
  Patient_id_no, Wound_id -> Wound_image

- **Normalization**
  Table already satisfies 1NF, 2NF, and 3NF, there is no need for further schema refinement.

4.5.8 MAIL_NOTIFICATION table

Table 4.15 : Mail_notification table

Notification_id	From_whom	Description	Patient_id	Doc_id	Admin_id
N0001	John	Doctor activation	P001	D001	A001
N0002	Jane	Session start	P002	D002	A002

N0003	Michael	Patient deactivation	P003	D003	A003

- **Functional Dependency**
  Notification_id -> From_whom_Description, Patient_id, Doc_id, Admin_id

- **Normalization**
  Table already satisfies 1NF, 2NF, and 3NF, there is no need for further schema refinement.

4.5.9 SESSION table

Table 4.16 : Session table

Doc_id	Session_id	Description	Date	Time
D001	S0001	Cardiology Consultation	2023-06-01	10:00 AM
D002	S0002	Dermatology Consultation	2023-06-02	11:30 AM
D003	S0003	Pediatrics Consultation	2023-06-03	02:15 PM

- **Functional Dependency**
  Doc_id, Session_id -> Description, Date,Time

- **Normalization**
  Table already satisfies 1NF, 2NF, and 3NF, there is no need for further schema refinement.

4.5.10 JOIN table

Table 4.17 : Join table

Session_id	Patient_id
S0001	P001
S0002	P002
S0003	P003

- **Normalization**
  Table already satisfies 1NF, 2NF, and 3NF, there is no need for further schema refinement.

4.5.11 Schema refinement tables

PERSON

Id_no	E_mail	User_name	Password	First_name	Last_name

ADMIN

Admin_id

PATIENT

Patient_id	Address	DOB	Admin_id	Active_status	Med_details_no	Doc_id

PATIENT_MEDICAL_DETAILS

Patient_id	Med_details_no	Diabetic_diagnosis	Current_medication	Allergies

PATIENT_TELE_NO:

Patient_id	Tele_no

DOCTOR_EXPERIENCE

Med_rgstr_no	Yrs_of_experience

DOCTOR

Doc_id	Specialization	Admin_id	Active_status	Med_rgstr_no

CLASSIFICATION_SCREEN_TYPE

Scrn_id_classi_id	Screen_type	Classification_type

REGION

Region_id	Wound_diameter	Region_image

WOUND

Patient_id	Wound_id	Date	Time	Doc_id	Session_id	Scrn_id_classi_id	Region_id

WOUND_WOUND_IMAGE

Patient_id	Wound_id	Wound_image

MAIL_NOTIFICATION

Notification_id	From_whom	Description	Patient_id	Doc_id	Admin_id

SESSION

Doc_id	Session_id	Description	Date	Time

JOIN

Session_id	Patient_id

4.6 PHYSICAL DATABASE DESIGN

4.6.1 Data type

Using refine tables we can create SQL statement for tables.From there we can get a clear idea about the data type.

- **PERSON table**

Id_no	E_mail	User_name	Password	First_name	Last_name

```
CREATE TABLE PERSON (
  Id_no VARCHAR (4) NOT NULL,
  E_mail VARCHAR(100) UNIQUE,
  User_name VARCHAR(100)
  Password VARCHAR(6),
  First_name VARCHAR(30),
  Last_name VARCHAR(30),
  PRIMARY KEY( Id_no),
);
```

- **ADMIN table**

Admin_id

```
CREATE TABLE ADMIN (
 Admin_id VARCHAR(4) NOT NULL,
  PRIMARY KEY(Admin_id)
  );
```

- **PATIENT table**

Patient_id	Address	DOB	Admin_id	Active_status	Med_details_no	Doc_id

```
CREATE TABLE PATIENT (
  Patient_id VARCHAR(4)  NOT NULL,
  Address VARCHAR(150),
  DOB DATE,
  Admin_id VARCHAR(4) NOT NULL,
  Med_details_no VARCHAR(6) NOT NULL,
  Doc_id VARCHAR(4) NOT NULL,
  Active_status VARCHAR(8),
  PRIMARY KEY(Patient_id),
  FOREIGN KEY (Doc_id) REFERENCES DOCTOR(Doc_id),
  FOREIGN KEY (Admin_id) REFERENCES ADMIN(Admin_id),
 FOREIGN KEY (Med_details_no) REFERENCES PATIENT_MEDICAL_DETAILS
(Med_details_no),
);
```

- **PATIENT_MEDICAL_DETAILS table**

Med_details_no	Patient_id	Diabetic_diagnosis	Current_medication	Allergies

```
CREATE TABLE PATIENT_MEDICAL_DETAILS (
  Med_details_no VARCHAR(6) NOT NULL,
  Patient_id VARCHAR(4) NOT NULL,
  Diabetic_diagnosis VARCHAR(120),
  Current_medication VARCHAR(120),
  Allergies VARCHAR(120),
  PRIMARY KEY(Med_details_no,Patient_id),
  FOREIGN KEY (Patient_id) REFERENCES PATIENT(Patient_id)
);
```

- **PATIENT_TELE_NO table**

Patient_id	Tele_no

```
  CREATE TABLE PATIENT_TELE_NO (
  Patient_id VARCHAR(255) NOT NULL,
  Tele_no VARCHAR(11) NOT NULL,
  PRIMARY KEY (Patient_id, Tele_no),
  FOREIGN KEY (Patient_id) REFERENCES PATIENT(Patient_id)
```

);

- **DOCTOR_EXPERIENCE table**

Med_rgstr_no	Yrs_of_experience

```
CREATE TABLE DOCTOR_EXPERIENCE (
Med_rgstr_no VARCHAR(4) NOT NULL,
Yrs_of_experience float(50),
PRIMARY KEY(Med_rgstr_no)
FOREIGN KEY (Med_rgstr_no) REFERENCES DOCTOR (Med_rgstr_no)
);
```

- **DOCTOR table**

Doc_id	Specialization	Admin_id	Active_status	Med_rgstr_no

```
CREATE TABLE DOCTOR (
Doc_id VARCHAR(4) NOT NULL,
Specialization VARCHAR(120),
Admin_id VARCHAR(4) NOT NULL,
Active_status VARCHAR(8),
Med_rgstr_no VARCHAR(4) NOT NULL,
PRIMARY KEY(Doc_id),
FOREIGN KEY (Admin_id) REFERENCES ADMIN (Admin_id),
);
```

- **CLASSIFICATION_SCREEN_TYPE table**

Scrn_id_classi_id	Screen_type	Classification_type

```
CREATE TABLE CLASSIFICATION_SCREEN_TYPE (
Scrn_id_classi_id VARCHAR(5) NOT NULL,
Screen_type VARCHAR(10),
Classification_type VARCHAR(10),
PRIMARY KEY (Scrn_id_classi_id)
);
```

- **REGION table**

Region_id	Wound_diameter	Region_image

```
CREATE TABLE REGION (
Region_id VARCHAR(5) NOT NULL,
Wound_diameter FLOAT(5),
Region_image NVARCHAR(2000),
PRIMARY KEY(Region_id)
);
```

- **WOUND table**

Patient_id	Wound_id	Date	Time	Doc_id	Scrn_id_classi_id	Region_id

```
CREATE TABLE WOUND (
  Patient_id VARCHAR(4) NOT NULL,
  Wound_id VARCHAR(5) NOT NULL,
  Date DATE NOT NULL,
  Time TIME,
  Doc_id_VARCHAR(4) NOT NULL,
  Scrn_id_classi_id VARCHAR(5) NOT NULL,
  Region_id VARCHAR(5) NOT NULL,
  PRIMARY KEY(Patient_id, Wound_id),
  FOREIGN KEY (Doc_id) REFERENCES DOCTOR (Doc_id),
  FOREIGN KEY (Region_id) REFERENCES REGION (Region_id),
  FOREIGN KEY (Scrn_id_classi_id) REFERENCES CLASSIFICATION_SCREEN_TYPE
(Scrn_id_classi_id),
);
```

- **WOUND_WOUND_IMAGE table**

Patient_id	Wound_id	Wound_image

```
CREATE TABLE WOUND_WOUND_IMAGE (
  Patient_id VARCHAR(4) NOT NULL,
  Wound_id VARCHAR(5) NOT NULL,
  Wound_image NVACHAR(2000),
  PRIMARY KEY (Patient_id, Wound_id),
  FOREIGN KEY (Patient_id) REFERENCES PATIENT(Patient_id),
  FOREIGN KEY (Wound_id) REFERENCES WOUND(Wound_id,Patient_id)
```

- **MAIL_NOTIFICATION table**

Notification_id	From_whom	Description	Patient_id	Doc_id	Admin_id

```
CREATE TABLE MAIL_NOTIFICATION (
  Notification_id VARCHAR(5) NOT NULL,
  From_whom VARCHAR(50),
  Description VARCHAR(100),
  Patient_id VARCHAR(4) NOT NULL,
  Doc_id_VARCHAR(4) NOT NULL,
  Admin_id_VARCHAR(4) NOT NULL,
  PRIMARY KEY(Notification_id),
  FOREIGN KEY (Patient_id) REFERENCES PATIENT(Patient_id),
  FOREIGN KEY (Doc_id) REFERENCES DOCTOR(Doc_id),
  FOREIGN KEY (Admin_id) REFERENCES ADMIN(Admin_id)
);
```

- **SESSION table**

Doc_id	Session_id	Description	Date	Time

```
CREATE TABLE SESSION (
  Doc_id VARCHAR(4) NOT NULL ,
  Session_id VARCHAR(5) NOT NULL,
  Description VARCHAR(100) NOT NULL,
  Date DATE NOT NULL,
  Time TIME,
  PRIMARY KEY (Doc_id, Session_id),
  FOREIGN KEY (Doc_id) REFERENCES DOCTOR(Doc_id),);
```

- **JOIN table**

Session_id	Patient_id

```
CREATE TABLE SESSION (
  Session_id VARCHAR(5) NOT NULL,
  Patient_id VARCHAR(4) NOT NULL,
  PRIMARY KEY (Patient_id, Session_id),
  FOREIGN KEY (Session_id) REFERENCES SESSION(Session_id),
  FOREIGN KEY (Patient_id) REFERENCES PATIENT(Patient_id)
);
```

4.6.2 Query example

- Consider the below PERSON table,

Table 4.18 : Person table

Id_no	E_mail	User_name	Password	First_name	Last_name
P001	john@example.com	Nirmal	=23asb	Namal	Perera
D002	jane@example.com	abc456	Jane#53	Smith	Janesmith
P003	michael@example.com	xyz789	M4%4el	Johnson	mjohnson

- **To get Id_no, First_name, Last_name from person table.**

```
SELECT PERSON Id_no, First_name, Last_name
FROM PERSON
```

Table 4.19 : New person table

Id_no	First_name	Last_name
P001	Namal	Perera
D002	Smith	Janesmith
P003	Johnson	mjohnson

- **Insert to this data for PERSON table**

(P004,dfhfgxbc@example.com,shamy,4%44el,Johnson,mjohnson)

INSERT INTO PERSON(Id_no,E_mail,User_name,Password,First_name,Last_name)
VALUES(P004,dfhfgxbc@exam.com,shamy,4%44el,Gihan,misaon)

Table 4.20 : New person table

Id_no	E_mail	User_name	Password	First_name	Last_name
P001	john@example.com	Nirmal	=23asb	Namal	Perera
D002	jane@example.com	abc456	Jane#53	Smith	Janesmith
P003	michael@example.com	xyz789	M4%4el	Johnson	mjohnson
P004	dfhfgxbc@exam.com	shamy	4%44el	Gihan	misaon

- Consider the below CLASSIFICATION_SCREEN_TYPE table,

Table 4.21 : Classification_screen_type table

Scrn_id_ classi_id	Screen_type	Classification_type
SC001	Video	Inspection
SC002	Video	Schema
SC003	self	Inspection
SC004	self	Schema

- **To get only the data of those whose screen_type is video in CLASSIFICATION_SCREEN_TYPE table**

SELECT* FROM CLASSIFICATION_SCREEN_TYPE where Screen_type = 'video'

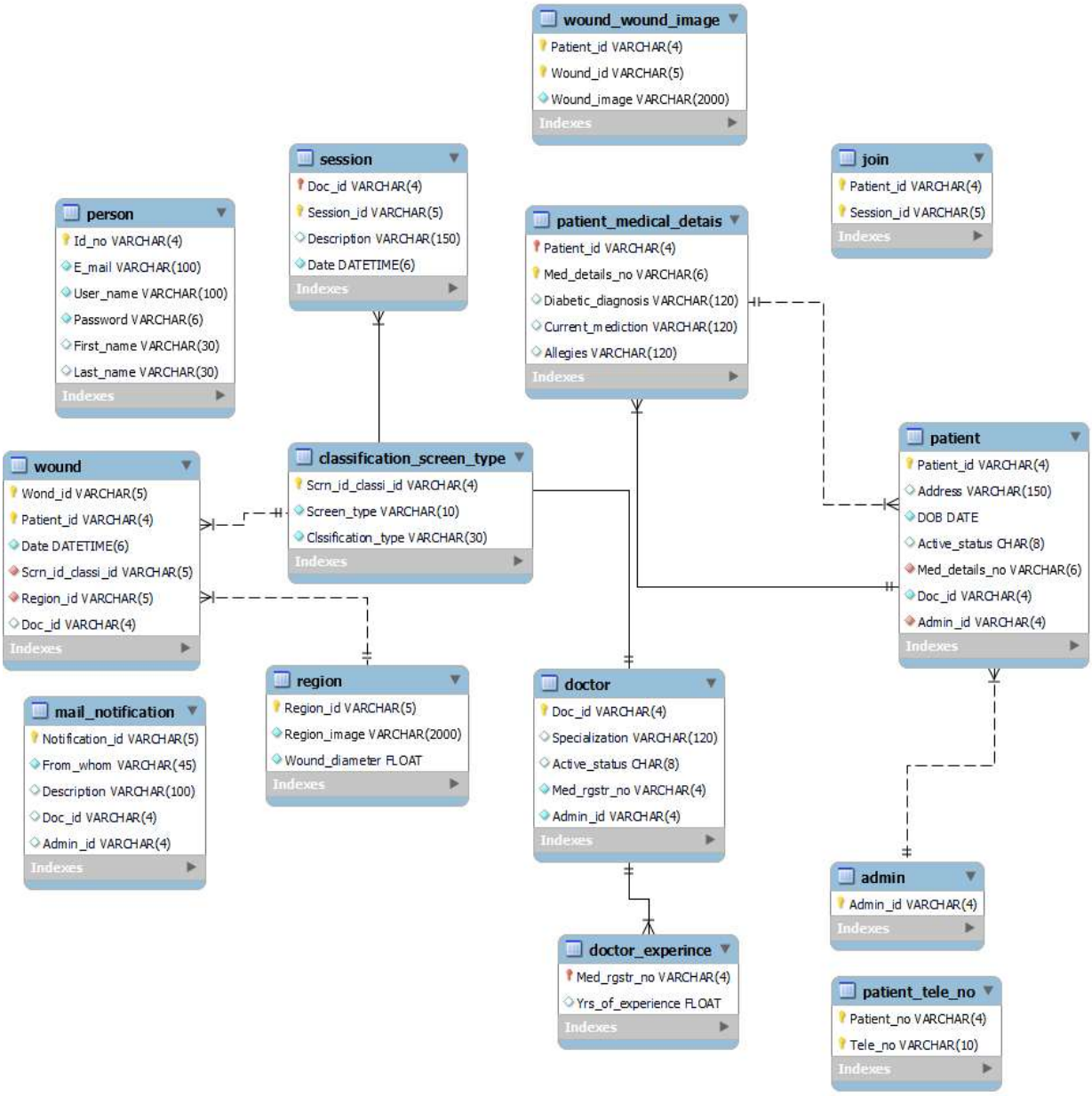
Figure 4.42 :New classification_screen_type table

Scrn_id_ classi_id	Screen_type	Classification_type
SC001	Video	Inspection
SC002	Video	Inspection

- **To calculate the amount of Scrn_id_ classi_id data in the CLASSIFICATION_SCREEN_TYPE table ?**

SELECT count(Scrn_id_ classi_id) FROM CLASSIFICATION_SCREEN_TYPE

Count(Scrn_id_ classi_id)
2



4.7 SECURITY DESIGN

4.7.1 Access control

In login the database system, It helps prevent unauthorized access to the app's resources and database, ensuring compliance with data protection regulations. For that we use **Role-Based Access Control (RBAC)**. It is suitable for using role-based access control (RBAC) in the following areas:

- **User Registration and Authentication:**
  To ensure appropriate access and permissions, assign roles (such as Admin, Doctor, and Patient) throughout the registration process.

- **User Management:**
  Manage user roles within the system through managing user access, which enables administrators to assign, change, or revoke responsibilities as necessary.

- **Functionality and Feature Access:**
  Control access to different app functionalities and features based on roles. For example, doctors may have access to patient records and treatment management features, while patients may have access to their own health information and appointment scheduling.

- **Data Access and Privacy:**
  Use RBAC to limit access to private patient information and make sure that only authorized roles (like doctors) have access to it.

4.7.2 Data encryption

We can use data encryption securely store passwords in DBMS using a **hashing algorithm in Flutter**.

Using a hashing algorithm to securely store passwords in a DBMS is crucial for protecting user data. Hashing algorithms are one-way functions that convert passwords into fixed-size strings called hashes.

The purpose of using a hashing algorithm is to ensure password protection, verification, and defense against attacks. By storing passwords as hashes, even if the database is compromised, the actual passwords remain hidden.

During the verification process, the entered password is hashed and compared with the stored hash for authentication. This allows for secure password verification without exposing the actual passwords. By choosing a strong hashing algorithm and incorporating additional security measures like salting, the security of stored passwords can be significantly enhanced.

Overall, using a hashing algorithm is essential for maintaining the confidentiality of passwords and protecting user data from unauthorized access.

CHAPTER 5 - RESEARCH DESIGN

The literature review survey conducted on the proposed system makes up the majority of the research design chapter. The technology and methodology used in earlier studies related to the project idea are summarized in the literature review section. This chapter also provides a quick overview of the project's high-level implementation components as well as its non-functional components.

5.1 Objective Based Literature Review

Several studies have explored the application of various machine learning models, including CNN, SVM, and RNN, for the classification of diabetic foot wounds. These models have shown promising results in accurately categorizing diabetic foot wounds based on their severity and characteristics.

CNN, in particular, has emerged as one of the most effective deep learning models for image classification tasks. Its ability to automatically learn hierarchical features from input images makes it well-suited for analyzing complex and diverse wound patterns. CNN models consist of multiple layers of convolutional and pooling operations, allowing them to capture intricate details and spatial relationships within the wound images.

In recent studies, researchers have utilized the DFUC 2021 dataset, a widely recognized standard dataset, along with the expertise of validated specialists, to classify diabetic foot wounds. Yap et al. employed efficientNetB0, a powerful deep learning architecture, to accurately categorize these wounds, focusing primarily on infection and ischemia. Similarly, Chandrabalan and Hoon utilized another advanced model called InceptionResNetV2 to effectively classify diabetic foot wounds. These efforts contribute to the ongoing advancements in the field of diabetic foot wound classification, providing valuable insights for the development of improved diagnosis and treatment strategies[3, 4].

Due to their complexity and large number of layers and parameters, these deep learning models can be time- and resource-intensive to develop and implement. Their deployment on devices with limited resources or in real-time applications with constrained processing power may be hindered by the significant computational resources they may need, such as high-end GPUs and large memory capacities. Popular convolutional neural network (CNN) designs used for image segmentation tasks include LinkNet and U-Net. For semantic segmentation, which entails categorizing each pixel in an image into one or more classes, LinkNet and U-Net were created. Limited contextual understanding could come from this, particularly for complex scenes or objects with significant contextual context. Furthermore, lack of scalability and lack of interpretability are other limitations of these models.

Several models are prominent in wound segmentation research. U-Net, designed for image segmentation, employs an encoder-decoder architecture with skip connections for precise object

boundary localization. Mask R-CNN combines object detection and pixel-level segmentation, excelling in accurate object delineation. FCN, a fully convolutional network, preserves spatial information by replacing fully connected layers. FCN variants, including FCN-8s and FCN-ResNet, capture context and spatial details effectively. U-Net, Mask R-CNN, and FCN have proven their efficacy in diverse applications, owing to their unique architectures and advanced techniques, making them valuable tools for complex segmentation tasks.

When observing research papers we can identify that most of the researchers used U-Net for wound segmentations of their researches. In[5] they have showed that U-Net is a more effective one for the wound segmentation. And in[6] they have also show that the U-Net is a effective one for the image segmentation they used hybrid method in there research.

5.2 Formalizing High-level implementation components

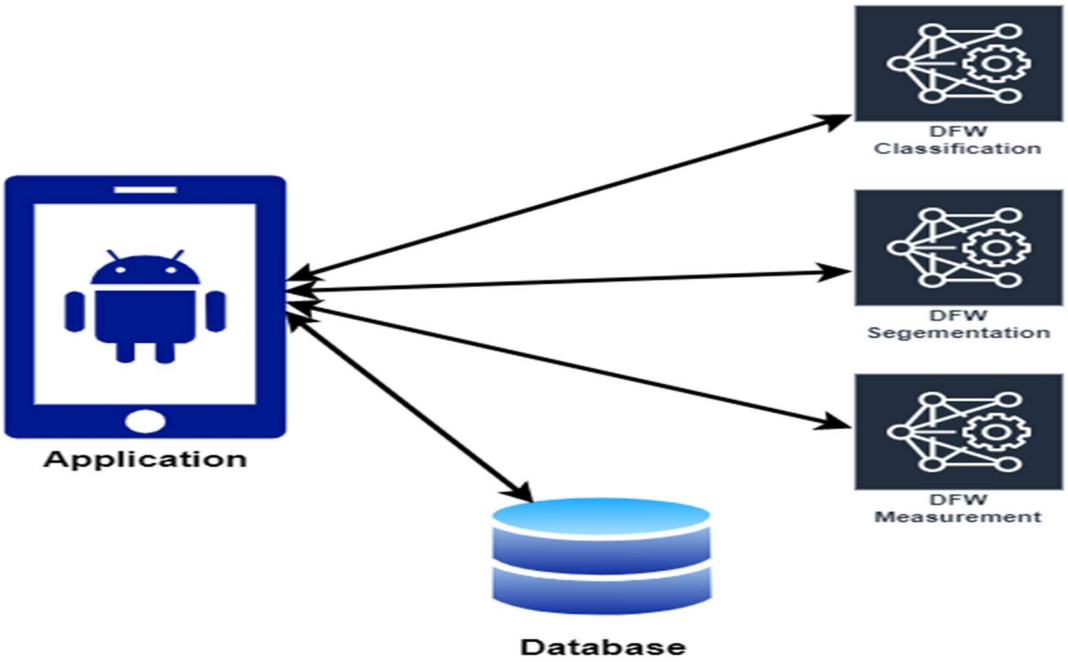


Figure 5.1 : System Architecture

The application connects to a MySQL database deployed in the cloud, enabling seamless access to relevant information. The database serves as the central repository for three modules: diabetic foot wound classification, diabetic foot wound segmentation, and wound measurement. Each module interacts with the database to store and retrieve data related to their respective functionalities. The diabetic foot wound classification module stores classification results, providing insights into the severity or type of diabetic foot wounds. The diabetic foot wound segmentation module stores segmented wound region masks, allowing for precise identification of

the wound area. Lastly, the measurement module stores measured wound size data, capturing important metrics such as diameter or length. By connecting these modules to the database, the application ensures efficient data storage, retrieval, and management, enabling healthcare professionals to access and analyze wound-related information effectively.

5.2.1 To Develop the Deep Learning Model for Segmenting Diabetic Foot Wounds

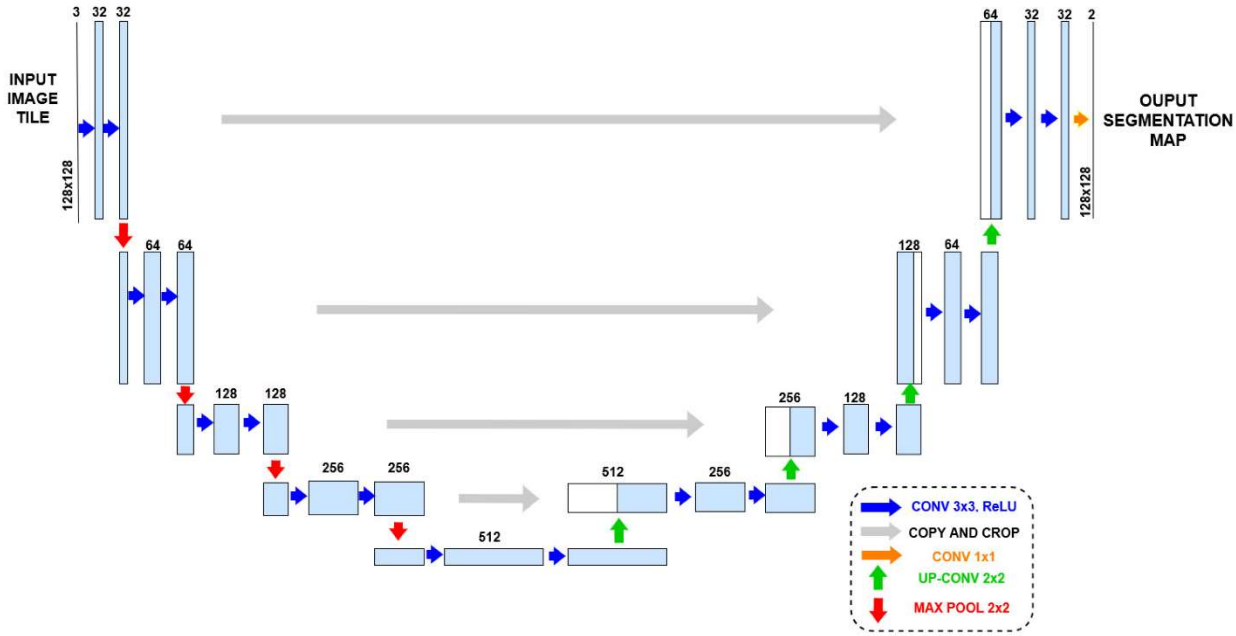


Figure 5.2 : DFW Segmentation Architecture

- **Data Preparation**

We have discovered a dataset of diabetic foot wound segmented images that can be used to train a model for identifying wound regions. This dataset has been meticulously annotated by clinicians, providing accurate delineation of the wound boundaries.

- **Data Preprocessing**

To prepare wound images for training the U-Net model, a series of preprocessing steps can be followed. Firstly, a dataset of wound images and their corresponding ground truth masks needs to be collected. The dataset should be carefully cleaned to remove any irrelevant or noisy images that may negatively impact the training process.

Next, data augmentation techniques can be applied to increase the diversity of the dataset. This involves performing transformations such as rotation, scaling, flipping, and adjusting brightness and contrast on the images. By generating augmented versions of the images, the model can learn to generalize better and handle different variations in wound appearances.

After augmentation, the images should be resized to a consistent resolution while maintaining the aspect ratio. This ensures uniformity in input size for the U-Net model. Additionally, it is important to normalize the pixel values of the images to a common range, such as [0, 1] or [-1, 1]. Normalization helps in stabilizing the training process and improving convergence.

Simultaneously, the corresponding ground truth masks should undergo similar transformations and resizing as the images. It is crucial to ensure that the masks are binary, with pixels representing either background or wound regions.

To evaluate the performance of the trained model, the preprocessed dataset should be split into training, validation, and testing sets. The training set is used to train the U-Net model, while the validation set helps in tuning hyperparameters and monitoring performance. Finally, the testing set is used to assess the model's performance on unseen data.

By following these preprocessing steps, the wound images can be properly prepared for training the U-Net model. This systematic approach ensures that the model receives high-quality data, leading to more accurate and robust segmentation results.

- **Model Architecture**

The U-Net model architecture is characterized by its encoder-decoder structure with skip connections, which enables accurate segmentation of object boundaries. It consists of two main parts: the contracting path (encoder) and the expansive path (decoder).

The contracting path consists of multiple convolutional layers followed by max-pooling operations. This part of the network gradually reduces the spatial dimensions of the input image while capturing high-level semantic information. Each convolutional layer is typically followed by a rectified linear unit (ReLU) activation function, which introduces non-linearity.

The expansive path, also known as the decoder, is responsible for upsampling the feature maps and recovering the spatial resolution. It involves a series of upsampling layers followed by convolutional layers. The skip connections are introduced by concatenating feature maps from the contracting path to the corresponding layers in the expansive path. These skip connections help in preserving fine-grained details and improving localization accuracy.

The final layer of the U-Net architecture typically consists of a 1x1 convolutional layer followed by a sigmoid activation function. This sigmoid activation produces a probability map for each pixel, indicating the likelihood of belonging to the target class (wound or background).

The U-Net model architecture can be customized by adjusting the number of layers, the filter sizes, and the skip connections to suit the specific requirements of the segmentation task. Additionally, techniques like batch normalization and dropout can be incorporated to improve generalization and reduce overfitting.

Overall, the U-Net architecture's unique design allows it to effectively capture both local and global contextual information, enabling accurate and detailed segmentation of wounds[7].

- **Model Compilation**

The compilation of the U-Net model involves configuring the loss function, optimizer, and evaluation metrics. For wound segmentation, a common choice is binary cross-entropy as the loss function, Adam or RMSprop as the optimizer, and evaluation metrics such as accuracy, IoU, or dice coefficient. By specifying these components, the model is prepared for training using the preprocessed wound image dataset. During training, the model's weights are updated using techniques like mini-batch gradient descent and backpropagation to optimize its performance.

- **Model Training**

During the training phase, the U-Net model is fed with preprocessed wound images along with their corresponding ground truth segmentation masks. The model is trained to learn the mapping between input images and their corresponding segmentation masks through an iterative optimization process. The training data is divided into mini-batches, and for each batch, the model computes the predicted segmentation masks. A loss function, such as binary cross-entropy, is then used to measure the dissimilarity between the predicted masks and the ground truth masks. The model's weights are updated using gradient descent optimization techniques, such as backpropagation, to minimize the loss. This process is repeated for multiple epochs until the model converges and achieves satisfactory segmentation performance. The training process aims to optimize the model's parameters to accurately segment wound regions based on the provided images.

- **Model Evaluation**

In wound segmentation model evaluation, the dataset is split into training and evaluation sets. Metrics like Intersection over Union (IoU), precision, recall, and F1 score are commonly used. IoU measures the overlap between predicted and ground truth masks. Precision and recall evaluate the model's ability to identify wounds accurately and avoid false positives and false negatives. The F1 score provides a balanced measure of precision and recall. Visual inspection and expert assessment of segmented results are also important for evaluating fine details and clinical suitability. These evaluations help gauge the model's performance, generalization, and applicability in clinical settings.

- **Model Optimization**

Model optimization involves fine-tuning the hyperparameters and architecture to improve the model's performance. Techniques such as learning rate scheduling, weight regularization, and dropout can be employed. Data augmentation methods like rotation, scaling, and flipping can help increase the model's ability to generalize. Optimization algorithms like stochastic gradient descent with momentum or Adam can be used to update model parameters. Additionally, model compression techniques, such as quantization or pruning, can reduce model size and inference

time. By carefully optimizing the model, its accuracy, speed, and memory efficiency can be enhanced.

5.2.1.1 To Develop the Deep Learning Model for Measure Diabetic Foot Wounds

To create a wound measurement model using images captured by a mobile phone camera, these steps will be followed.

- **Image Preprocessing**

Apply preprocessing techniques to enhance the captured images. This may involve tasks such as resizing, noise reduction, contrast adjustment, and color normalization.

- **Wound Segmentation**

Wound segmentation process is explained in the previous chapter.

- **Measurement Calculation**

Utilize the extracted features to calculate the desired wound measurements. For example, calculating the wound area by counting the number of pixels within the segmented region or estimating the length and width using geometric approximations.

- **Calibration**

Incorporate a calibration step to ensure accurate measurements. This involves calibrating the mobile camera's properties, such as focal length and sensor size, to obtain measurements in real-world units (e.g., centimeters or inches). Calibration can be achieved by capturing reference objects of known dimensions within the image and using them as a reference for scaling the measurements. Calibration process included several steps.

Capture Calibration Images: Start by capturing calibration images that contain reference objects with known dimensions. These reference objects should be placed within the wound image and should be easily identifiable. For example, including a ruler, a coin, or any other object with known dimensions.

Measure Reference Object Dimensions: Manually measure the dimensions of the reference objects in the calibration images using a reliable measuring tool, such as a physical ruler. Note down the actual measurements of the reference objects in real-world units (e.g., centimetres or inches).

Extract Reference Object Regions: Use image processing techniques to identify and extract the regions of the reference objects within the calibration images. This can be done by thresholding, edge detection, or any other suitable segmentation algorithm.

Calculate Scale Factors: Calculate the scale factors by dividing the known dimensions of the reference objects by their corresponding pixel measurements in the image. For example, if a reference object with a known length of 5 centimeters corresponds to 100 pixels in the image, the scale factor would be 5 cm / 100 pixels = 0.05 cm/pixel.

Apply Scale Factors: Apply the scale factors obtained from the calibration process to convert the pixel measurements of the wound region to real-world measurements. Multiply the pixel count by the appropriate scale factor to obtain the measurement in the desired units.

- **Validation**

Validate the accuracy of the measurements obtained from the mobile application. Compare the results with ground truth measurements taken manually by medical professionals or through other reliable measurement techniques. This validation process helps assess the accuracy and reliability of your mobile wound measurement model.

5.2.2 To develop the deep learning model for classifying diabetic foot wound

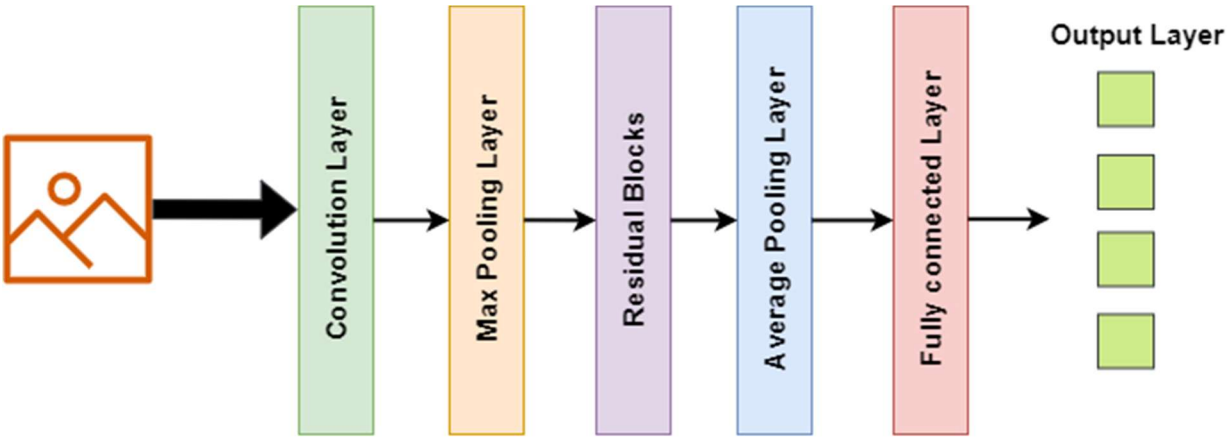


Figure 5.3 : DFW classification architecture

- **Data Preparation**
  Collect a dataset of diabetic foot wound images labeled with their corresponding classes (infection, ischemia, both)[8]. Split the dataset into training and testing sets.

- **Data Preprocessing**
  Resize the images to a consistent size, normalize the pixel values, and perform any additional preprocessing steps like data augmentation (flip horizontal and vertical, rotate -

180 degree, add Gaussian noise, crop, shear, scale and adjust contrast by gamma 2.0) to increase the diversity of training data.

- **Model Architecture:**
  - **Input Layer**: Accepts the image input with the specified dimensions.
  - **Convolutional Layers**: The initial stage consists of a single convolutional layer followed by a max pooling layer, which downsamples the input image. This stage is responsible for extracting low-level features from the input image
  - **Residual Blocks**: The core component of the ResNet-101 architecture is a series of residual blocks. Each residual block contains multiple convolutional layers and introduces skip connections (or residual connections) that bypass certain layers. This allows the network to learn residual mappings instead of directly fitting the desired underlying mappings.
    - **Bottleneck Structure**: Within each residual block, there is a bottleneck structure that helps reduce the computational cost. It involves a combination of 1x1, 3x3, and 1x1 convolutional layers to transform the input dimensions while keeping the number of parameters manageable.
    - **Identity Mapping**: The skip connections in ResNet-101 enable the direct propagation of activations from earlier layers to later layers. This identity mapping allows the network to avoid degradation in performance caused by the vanishing gradient problem.
  - **Global Average Pooling**: After the series of residual blocks, a global average pooling layer is applied to aggregate spatial information by taking the average of each feature map. This reduces the spatial dimensions to a single value per feature map.
  - **Fully Connected Layer**: Finally, a fully connected layer is added to perform the actual classification. This layer maps the aggregated features to the desired number of output classes.

- **Output Layer**: The final fully connected layer with softmax activation outputs the predicted probabilities for each class.

- **Model Compilation**
  Define the loss function (categorical cross-entropy) and optimizer (Adam) for training the model. Additionally, you can specify evaluation metrics such as accuracy.

- **Model Training**
  Train the CNN model on the training dataset using the compiled settings. Adjust the hyperparameters[4]to optimize the model's performance. Monitor the training process by tracking the loss and accuracy on both the training and validation sets.

- **Model Evaluation**
  Evaluate the trained model on the testing dataset to assess its performance. Calculate metrics such as accuracy, precision, recall, and F1-score to measure its classification performance.

- **Model Optimization**
  Fine-tune the model by experimenting with different architectural modifications, hyperparameter tuning, and regularization techniques (dropout, batch normalization) to improve its accuracy and generalization.

5.2.3 To formalize a mechanism to improve the accuracy of the proposed deep learning model through transfer learning

Many researchers have demonstrated that the use of TL in medical image classification tasks is effective and efficient.

We can apply difference transfer learning to our diabetic foot wound classification model, namely VGG19,MobileNetV2, InceptionResNetV2, and Resnet101[9].

Furthermore, ResNet can make the structure reach 1,000 layers through residual function, and at the same time make the classification effect of the structure reach an excellent effect. The core idea of ResNet [10] is to introduce an identity shortcut connection structure, directly skip one or more layers.

It's important to note that the choice of the specific architecture and its suitability for our task also depends on factors such as the size of our dataset, computational resources, and specific requirements. It's recommended to experiment with different models and architectures,

considering factors like model complexity, training time, and performance on our specific task, to determine the most suitable option for our image segmentation and classification needs.

The pre-processing stage was carried out for testing purposes on the Google Collab platform. The experimental results showed that the ResNet101 model achieved the best accuracy value of 0.993 (99%) with a precision of 1.00[11]. According we use to REsNEt as a transfer learning models to our project. As well as we have a large dataset and need to capture highly intricate features. So ResNet101 may be a good choice.

To develop a classification model using ResNet-101 and transfer learning from ImageNet

we can follow these steps:

- **Dataset Preparation**

Collect or acquire a dataset of images that you want to classify. Ensure that the dataset is properly labeled with the corresponding class/category for each image.

- **Load Pretrained ResNet-101**

Download the pretrained ResNet-101 model that has been trained on the ImageNet dataset. There are various deep learning frameworks like TensorFlow and Keras that provide pre-trained ResNet-101 models.

- **Model Architecture**

Import the ResNet-101 model and remove the top layers, which are responsible for the original ImageNet classification. Retain the convolutional layers as they contain learned features. We can freeze these layers to prevent them from being updated during training.

- **Add Custom Classifier**

Add a new classifier on top of the ResNet-101 convolutional layers. This classifier consists of fully connected layers and an output layer with the number of units equal to the number of classes in our dataset. This new classifier will be trained to classify your specific image classes.

- **Data Preprocessing**

Preprocess your dataset by resizing the images to match the input size required by the ResNet-101 model. Additionally, you may normalize the pixel values to a common range, such as [0, 1], or apply other preprocessing techniques like data augmentation to enhance the model's performance.

- **Training**

Split our dataset into training and validation sets. Use the training set to train the model. During training, you'll pass the images through the ResNet-101 layers, compute the output, and update the weights of the custom classifier using backpropagation. Experiment with different hyperparameters such as learning rate, batch size, and number of epochs to optimize the model's performance.

- **Evaluation**

Evaluate the trained model using the validation set to assess its classification accuracy. We can calculate metrics such as accuracy, precision, recall, or F1-score to measure the model's performance.

- **Fine-tuning (Optional)**

If the model's performance is not satisfactory, you can consider fine-tuning the pretrained ResNet-101 model. This involves unfreezing some of the convolutional layers and allowing them to be updated during training. Fine-tuning can help the model adapt to our specific dataset and potentially improve its classification accuracy.

- **Prediction**

Once the model is trained and evaluated, we can use it to make predictions on new, unseen images. Pass the images through the model and obtain the predicted class probabilities or labels.

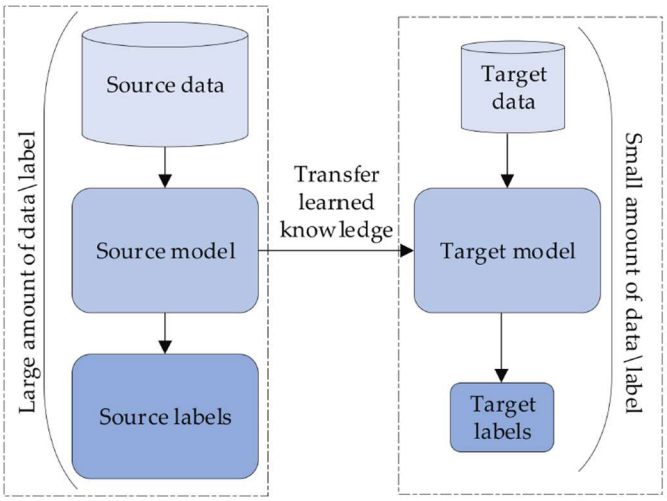


Figure 5.4 : Concept of the ResNet101 transfer learning model

5.3 Data extractions, sample design, test datasets, training data sets

5.3.1 Data Extractions

The dataset is gathered from DFUC2022 dataset. We received approval from the UK National Health Service Research Ethics Committee (reference number is 15/NW/0539) to use DFU

images. The DFUs were capture at room lighting, in full foot view, around 30-40cm away with the DFU centered. There cameras were used, i.e., Kodak DX4530, Nikon D3300 and Nikon COOLPIX P100. All images were taken by experienced podiatrist and physician in foot clinic. Images were then down sampled to 640 × 480 and stored in JPG format.

5.3.2 Sample Design

We have a segmentation dataset and a classification dataset including test and training image samples. Some sample images are given below.

Segmentation Dataset Samples (Test)

Figure 5.5 : Testing Images

Segmentation Dataset Samples (Training)

Figure 5.7 : Training images

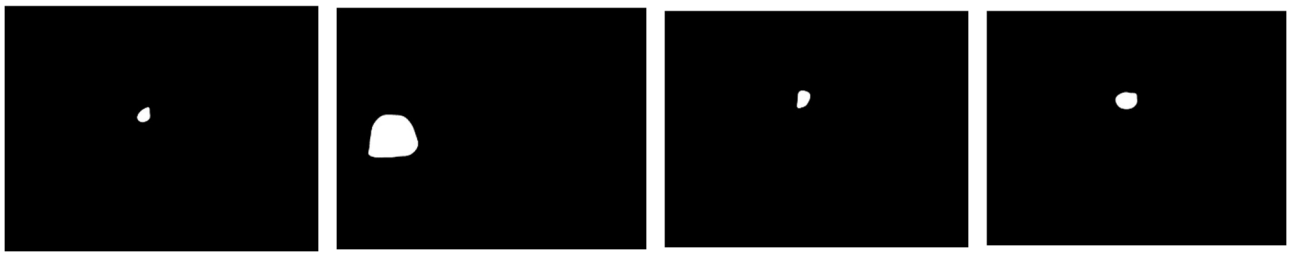
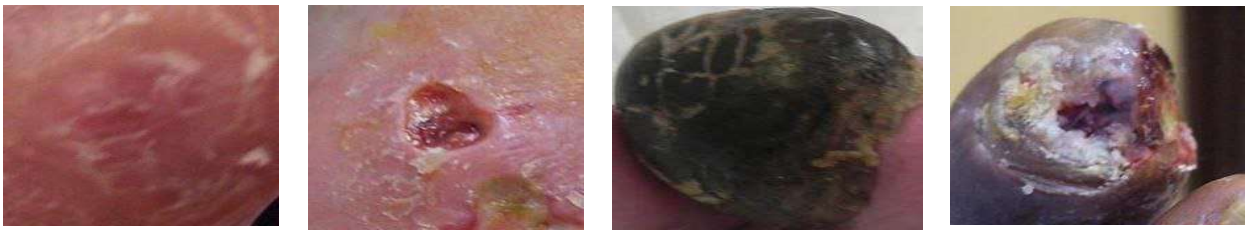


Figure 5.6 : Training Masks

Classification Dataset Samples



Normal Infection Ischemia Infection & Ischemia

Figure 5.8 : Classification Dataset Samples

5.3.3 Test Datasets and Training Datasets

Segmentation Dataset

[9]Introduced the Foot Ulcer Segmentation Dataset. This work focused on the development of segmentation CNNs using 1210 foot photographs exhibiting DFU which were collected over a 2 year period from 889 patients. They provided ground truth masks provided by wound care experts. Compressed before being converted to PNG. The provided ground truth files also appeared to be a mix of human and machine-generated masks. The images were 512 × 512 with 2000 for training and 2000 for test. The capture equipment was a Canon SX 620 hs and an iPad Pro.

Classification Dataset

For DFU classification, we followed a 50-50 split for training and testing set (case exclusion). From the training set, we further split 20% from training set, which we used as a validation set. The total images for DFUC2021 is 15,683, with 5,955 images for training set (2,555 infection only, 227 ischaemia only, 621 both infection and ischaemia, and 2,552 without ischaemia and infection), 3,994 unlabeled images and 5,734 images for testing set[4].

5.4 Non- functional Aspects

Confusion matrix is used to evaluate the classification and segmentation performance of our model. From the confusion matrix, calculate metrics such as accuracy, precision, recall, F1 score, and specificity are calculated.

- Classification Accuracy

  The F1 Score can be used to measure the accuracy of classification. It can calculate F1 score by measuring precision and recall[10].

- Segmentation Accuracy

  The Jaccard Similarity Coefficient (JSC) can be used to segmentation accuracy. For the image segmentation task, the similarity between the segmented mask image and the segmented image is evaluated using Jaccard Similarity Coefficient (JSC). The closer its calculated value is to 1, the better the segmentation is[11].

5.5 Validation methods and measurements

Validation is a process used in scientific projects to obtain the information needed to determine the capability of a given technology or the desired outcome. Validation can be refered as Dyanmic Testing of the product, There are several methods to validate our project activities.

Performance Evaluation Metrics:

Evaluate the performance of the model using the validation set. Calculate metrics such as accuracy, precision, recall, F1 score (for classification), and dice coefficient (for segmentation).

F1 Score

		Positive	Negative	
Predicted Label		True Positive (TP)	False Positive (FP)	Positive
		False Negative (FN)	True Negative (TN)	Negative
		True Label		

Figure 5.9 : F1 Table

$$F1\ Score = 2\frac{Precision \times Recall}{Precision + Recall}$$

$$F1\ Score = \frac{2TP}{2TP + FP + FN}$$

Jaccard Similarity Coefficient (JSC)

- First, the predicted and actual binary wound images are converted into sets or arrays. In these binary images, set the pixel values to 1 for the segmented region and 0 for the background.
- The union and intersection must be calculated by performing an element-wise logical "AND" operation between the predicted and ground truth arrays and performing an "OR" operation.
- Jaccard Similarity Coefficient (JSC) can be calculated using the following formula by counting the pixels that have a value of 1 in the intersection and union arrays.

$$\text{Jaccard Similarity Coefficient (JSC)} = \frac{\text{The count of pixels in the intersection images}}{\textit{The count of pixels in the union images}}$$

By calculating the Jaccard Similarity Coefficient (JSC) for multiple wound images and averaging the results, we can obtain overall assessment of the segmentation accuracy of our system.

Model Training:

Develop and train deep learning model on the training set. This may involve designing a convolutional neural network (CNN) architecture such as ResNet and optimizing the model using a suitable loss function and optimization algorithm.

Integration Testing:

Identify the components such as the deep learning models for classification and segmentation, the user interface, data preprocessing modules, and any external dependencies.

By performing integration testing, we can validate that the different components of diabetic foot wound classifier and screening tool work together seamlessly and produce the expected results. It helps identify any issues or bugs that may arise due to the integration of the various modules and ensures the overall functionality and reliability of this project.

CHAPTER 6 - REFERENCES

[1] W. Lidwell, K. Holden, and J. Butler, *Universal principles of design, revised and updated: 125 ways to enhance usability, influence perception, increase appeal, make better design decisions, and teach through design*. Rockport Pub, 2010.

[2] J. Preece, Y. Rogers, H. Sharp, D. Benyon, S. Holland, and T. Carey, *Human-computer interaction*. Addison-Wesley Longman Ltd., 1994.

[3] I. C. Dipto *et al.*, "Quantifying the Effect of Image Similarity on Diabetic Foot Ulcer Classification," in *Diabetic Foot Ulcers Grand Challenge*: Springer, 2022, pp. 1-18.

[4] M. H. Yap, B. Cassidy, J. M. Pappachan, C. O'Shea, D. Gillespie, and N. D. Reeves, "Analysis towards classification of infection and ischaemia of diabetic foot ulcers," in *2021 IEEE EMBS International Conference on Biomedical and Health Informatics (BHI)*, 2021: IEEE, pp. 1-4.

[5] T. Lan, Z. Li, and J. Chen, "FusionSegNet: Fusing global foot features and local wound features to diagnose diabetic foot," *Computers in Biology and Medicine,* p. 106456, 2022.

[6] A. Brodzicki and J. Jaworek-Korjakowska, "DFU-Ens: End-to-End Diabetic Foot Ulcer Segmentation Framework with Vision Transformer Based Detection," *Diabetic Foot Ulcers Grand Challenge: Third Challenge, DFUC 2022, Held in Conjunction with MICCAI 2022, Singapore, September 22, 2022, Proceedings,* vol. 13797, p. 101, 2023.

[7] C. Kendrick *et al.*, "Translating clinical delineation of diabetic foot ulcers into machine interpretable segmentation," *arXiv preprint arXiv:2204.11618,* 2022.

[8] M. Goyal, N. D. Reeves, S. Rajbhandari, N. Ahmad, C. Wang, and M. H. Yap, "Recognition of ischaemia and infection in diabetic foot ulcers: Dataset and techniques," *Computers in biology and medicine,* vol. 117, p. 103616, 2020.

[9] C. Wang *et al.*, "FUSeg: The foot ulcer segmentation challenge," *arXiv preprint arXiv:2201.00414,* 2022.

[10] M. Goyal, N. D. Reeves, S. Rajbhandari, and M. H. Yap, "Robust methods for real-time diabetic foot ulcer detection and localization on mobile devices," *IEEE journal of biomedical and health informatics,* vol. 23, no. 4, pp. 1730-1741, 2018.

[11] J. M. J. Valanarasu, V. A. Sindagi, I. Hacihaliloglu, and V. M. Patel, "Kiu-net: Towards accurate segmentation of biomedical images using over-complete representations," in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2020: 23rd International Conference, Lima, Peru, October 4–8, 2020, Proceedings, Part IV 23*, 2020: Springer, pp. 363-373.

CHAPTER 7 - APPROVAL

Registration Number	Index Number	Name	Signature
ASP/18/19/013	4437	K.A.P.Dhananjaya	
ASP/18/19/012	4436	H.L.A.R.L.Deepasara	
ASP/18/19/089	4420	M.A.S.Dimalshan	
ASP/18/19/060	4446	G.L.N.Sandarekha	
ASP/18/19/095	4423	I.N.Jayasekara	

Date: 2023/08/17

Approval of the supervisor(s)

I agree / ~~disagree~~ with the Design aspects stipulated in this Software Design Specification.

Name: Dr. K.A.S. H. Kulathilake

Department/ Organization: Computing

Signature: