GlucoSense: Smart Non-Invasive Glucose Monitoring System

Session: 2021 – 2025

Submitted by:

Aiman Malik 2021-CE-23

Ayesha Ahmed 2021-CE-18

Samreen Razzaq 2021-CE-13

Urwah Imran 2021-CE-15

Supervised by:

Dr. Yasir Saleem

Co-Supervised by:

Engr. Afeef Obaid

Department of Computer Engineering

University of Engineering and Technology

Lahore, Pakistan

GlucoSense: Smart Non-Invasive Glucose Monitoring System

(Session 2021 Computer Engineering)

The thesis is to be submitted to the Department of Computer Engineering, University of Engineering and Technology, Lahore for the partial fulfillment of the requirement for the Bachelor's degree in Computer Engineering.

Internal Examiner	**External Examiner**
Signature:	Signature:
Name:	Name:
Designation:	Designation:
Chairman	**Dean**
Signature:	Signature:
Dr. Ali Hammad Akbar	Prof.Dr. Muhammad Shoaib

Department of Computer Engineering

University of Engineering and Technology

Lahore, Pakistan

Declaration

We declare that the work contained in this thesis is my own, except where explicitly stated otherwise. In addition, this work has not been submitted to obtain another degree or professional qualification.

Aiman Malik
Signed: _____
Date: **29/04/25**

Ayesha Ahmed
Signed: _____
Date: **29/04/25**

Samreen Razzaq
Signed: _____
Date: **29/04/25**

Urwah Imran
Signed: _____
Date: **29/04/25**

Acknowledgments

We express our profound gratitude to Allah Almighty for the strength and guidance provided throughout this work. Immense appreciation is extended to our supervisor, Dr. Yasir Saleem, and Engr. Afeef Obaid, whose invaluable guidance, support, and mentorship have been instrumental in our journey. Their insights and encouragement have not only inspired us but also played a pivotal role in the completion of this dissertation. Special thanks are also extended to Dr. Warrad Ahmad, from General Hospital as well as Madiha Muzaffar, a Nursing Supervisor from Mayo Hospital, for generously providing us with access to the dataset, facilitating our research endeavors which were critical to our project's success. Their contributions have significantly enriched our work and furthered our understanding of the subject matter. Additionally, we owe a debt of gratitude to our parents whose unconditional love, encouragement, and support have been our constant source of strength. It is with great honour that we submit this dissertation, a testament to the collective guidance and support we have been fortunate to receive.

Dedicated to our Families

Contents

List of Figures

List of Tables

Abbreviations

IR	**I**nfra**R**ed
NIR	**N**ear **I**nfra**R**ed
ML	**M**achine **L**earning
AI	**A**rtificial **I**ntelligence
NN	**N**eural **N**etwork
ANN	**A**rtificial **N**eural **N**etwork
ULS	**U**ltra**S**onic **S**ensor
ADC	**A**nalog to **D**igital **C**onverter
LPF	**L**ow **P**ass **F**ilter
GUI	**G**raphical **U**ser **I**nterface
UML	**U**nified **M**odeling **L**anguage
IoT	**I**nternet **o**f **T**hings
BGL	**B**lood **G**lucose **L**evel
GSM	**G**lobal **S**ystem for **M**obile Communications

Abstract

The GlucoSense: Smart Non-Invasive Glucose Monitoring System is an innovative health monitoring tool designed to estimate blood glucose levels without the need for pricking. Leveraging infrared (IR) sensors, signal amplification, filtering techniques, and machine learning models, the system provides fast and non-invasive glucose readings. It incorporates an Arduino Nano for sensor data processing and an ultrasonic sensor to measure finger thickness, enhancing accuracy through personalized calibration. A neural network model is trained to predict glucose levels based on collected sensor data. The results are displayed via a user-friendly mobile app, which also maintains weekly trends and stores historical data. The system aims to offer a convenient, pain-free alternative to traditional glucose monitoring methods. Future improvements include refining prediction accuracy and expanding features for clinical integration.

Keywords: GlucoSense, non-invasive glucose monitoring, IR sensor, ultrasonic sensor, machine learning, neural network, Arduino Nano, mobile app, health monitoring system, blood glucose prediction

Chapter 1

Introduction

1.1 Overview

The GlucoSense: Smart Non-Invasive Glucose Monitoring System is an innovative healthcare solution designed to provide a painless and convenient alternative to traditional finger-prick blood glucose tests. It utilizes infrared (IR) sensors to collect real-time data from the user's finger without skin pricking. To ensure accurate and interference-free measurements, the sensor components are housed within a light-isolated enclosure that effectively prevents ambient light from affecting the readings, which is essential for reliable non-invasive data acquisition.

The analog signals from the IR sensors are amplified and filtered through dedicated electronic circuits and processed using an Arduino Nano microcontroller. An ultrasonic sensor is also employed to measure finger thickness, which is factored into a calibration formula to enhance prediction precision. The processed data is fed into a neural network-based machine learning model that estimates blood glucose levels.

The system integrates with a user-friendly mobile application that displays real-time readings, maintains a history of previous values, and generates weekly trends for better glucose management. With its non-invasive approach, refined signal handling, and mobile connectivity, GlucoSense offers an accessible and user-centric solution for continuous glucose monitoring. Future developments will focus on improving prediction accuracy and expanding integration into clinical healthcare settings.

1.2 Motivation

The increasing demand for accessible, painless, and real-time glucose monitoring solutions has driven the development of the GlucoSense: Smart Non-Invasive Glucose Monitoring System. Traditional glucose testing methods rely on invasive finger-prick techniques, which can be uncomfortable, discouraging regular monitoring, especially for diabetic patients. Additionally, frequent pricking poses risks of skin irritation, infection, and user non-compliance.

To address these limitations, GlucoSense offers a non-invasive approach that combines infrared sensing, signal processing, and machine learning to estimate blood glucose levels without the need for blood samples. The system aims to enhance the user experience by minimizing discomfort while still maintaining reliable results. Moreover, the integration of mobile technology ensures easy access to readings, historical data, and trends for both users and healthcare providers.

This project is motivated by the goal of empowering individuals with a smart, user-friendly, and non-invasive alternative for glucose monitoring, promoting proactive health management and improving the overall quality of care, particularly for chronic disease patients.

1.3 Problem Statement

Effective blood glucose monitoring is essential for managing diabetes and ensuring timely medical intervention. However, current glucose monitoring techniques predominantly rely on invasive finger-prick methods, which can be painful, inconvenient, and discouraging for regular use. These conventional approaches not only pose discomfort but also limit the frequency of monitoring, leading to delayed detection of abnormal glucose levels and potential health complications.

Moreover, access to advanced and user-friendly monitoring devices remains limited in remote or resource-constrained areas, further widening the gap in effective diabetes management. Existing solutions often lack personalization, are cost-intensive, and do not offer real-time insights or integration with smart healthcare platforms.

The GlucoSense system addresses these challenges by providing a non-invasive, affordable, and smart solution for real-time glucose monitoring. By leveraging IR sensing, ultrasonic-based finger thickness estimation, signal processing, and neural network-based predictions, GlucoSense aims to overcome the limitations of traditional methods. This research seeks to bridge the gap between the need

for continuous, painless glucose tracking and the constraints of current diagnostic tools, thereby improving patient care, encouraging frequent monitoring, and supporting early intervention strategies.

1.4 FYP Objectives

The following are the goals of the Smart non-invasive Glucose Monitoring System project:

1. Signal Processing Enhancement: Improve data reliability by integrating amplification circuits and low-pass filters to clean and strengthen sensor signals before analysis.

2. Light Interference Reduction: Design a controlled enclosure to block ambient light interference, ensuring accurate sensor readings in varying environments.

3. Finger Thickness Compensation: Use an ultrasonic sensor to measure finger distance and estimate thickness, incorporating this data into glucose level calibration for improved accuracy.

4. ML-Based Prediction: Implement and train a neural network model to process the collected data and predict glucose levels accurately based on sensor input and calibration parameters.

5. Mobile Application Integration: Develop a user-friendly mobile application to display real-time glucose readings, maintain history, and generate weekly tracking charts.

6. Accessibility and Affordability: Ensure the solution is low-cost and portable, making it suitable for widespread use, especially in resource-limited or remote regions.

7. Validation and Testing: Conduct rigorous testing by comparing system readings with conventional glucometers to evaluate performance, consistency, and reliability.

8. Contribution to Digital Healthcare: Promote innovation in healthcare technology by offering a modern, patient-friendly solution that supports proactive diabetes management and enhances overall care.

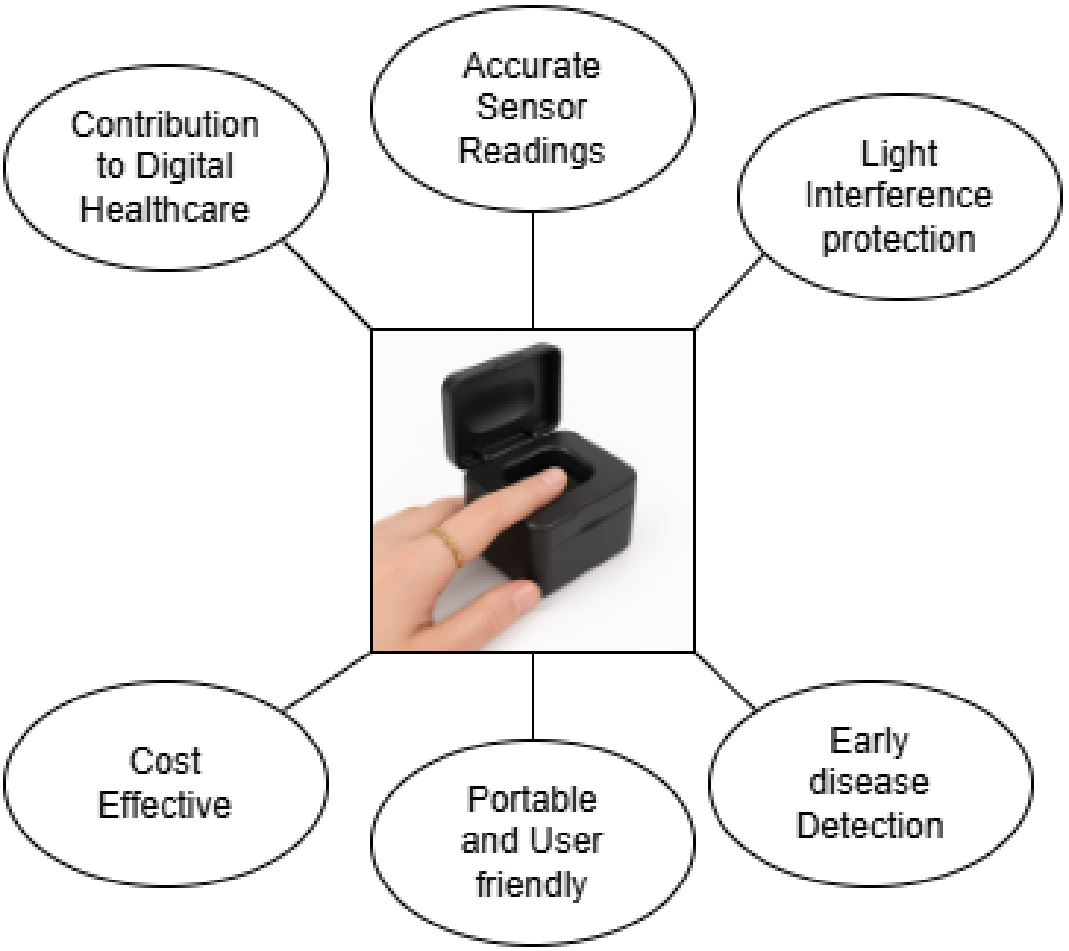

FIGURE 1.1: FYP Objectives

1.5 Socio Economic Benefits

The suggested approach has the potential to bring about a number of socioeconomic advantages, such as:

- Non-invasive glucose monitoring promotes patient comfort and encourages regular tracking, leading to better disease management and early intervention.
- The system is designed to be cost-effective, making it accessible to a wider population, including individuals in low-income or remote areas.
- Reducing reliance on consumables like test strips and needles lowers long-term healthcare costs for both patients and medical facilities.
- Integration with mobile applications enhances user engagement and self-care, minimizing the need for frequent clinic visits.

- Early detection of abnormal glucose levels helps prevent severe complications, thereby reducing the socio-economic burden of chronic diabetes care.

Chapter 2

Literature Review

2.1 Background

Over the years, significant developments have been made in the field of healthcare diagnostics, driven by technological innovations that have transformed how healthcare professionals monitor and manage chronic diseases. One critical area of focus has been blood glucose monitoring, which plays a vital role in the management of diabetes — a widespread and growing health concern worldwide.

Traditionally, blood glucose monitoring relies on invasive methods such as finger-prick tests, where a drop of blood is analyzed using a glucometer. While effective, these methods are painful, inconvenient, and often lead to poor compliance among patients requiring frequent monitoring.

Recognizing the need for a more patient-friendly solution, non-invasive glucose monitoring technologies have emerged. These technologies aim to provide accurate blood glucose readings without the need for skin penetration, enhancing patient comfort and promoting regular monitoring.

The GlucoSense: Smart Non-Invasive Glucose Monitoring System addresses this need by using infrared (IR) sensors to capture data from a user's finger without drawing blood. These are all measured as part of Glucose Monitoring System [1]. As shown in Figure 2.1, the system employs an advanced signal acquisition setup within a light-isolated enclosure to prevent ambient light interference, ensuring reliable data collection.

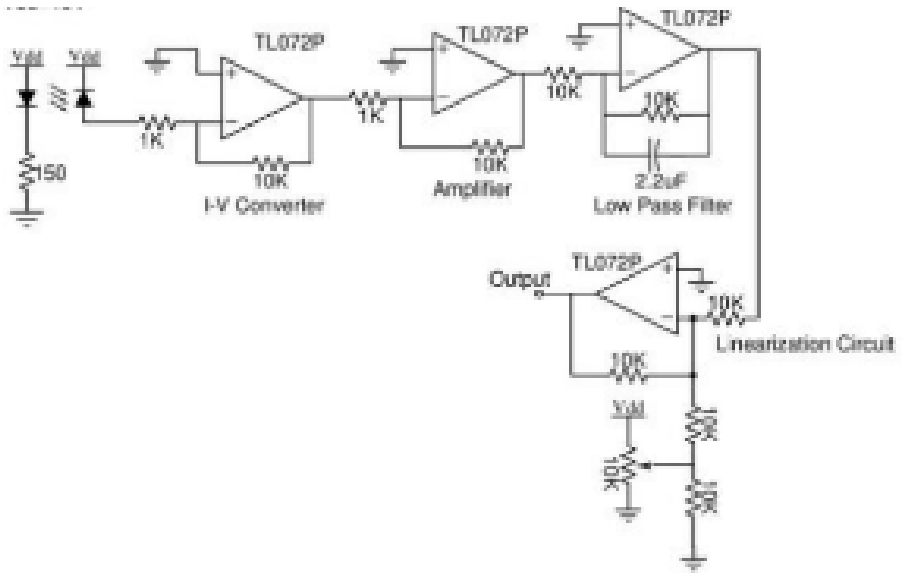

FIGURE 2.1: Working of IR Sensor

The collected IR signals are processed through amplification and filtering circuits to remove noise and enhance signal quality. An ultrasonic sensor is also used to measure finger thickness, an important factor considered during calibration to increase the accuracy of glucose level predictions.

The cleaned and preprocessed signals are fed into a machine learning model powered by a neural network, which is trained to estimate blood glucose levels based on the unique patterns captured from the user's finger. The processed results are then displayed through a user-friendly mobile application, which not only provides real-time readings but also maintains historical data and trends for better diabetes management.

The traditional manual method for blood glucose testing, as shown in Figure 2.2, though widely used, poses challenges regarding pain, risk of infection, and patient compliance, especially for frequent testing needs.

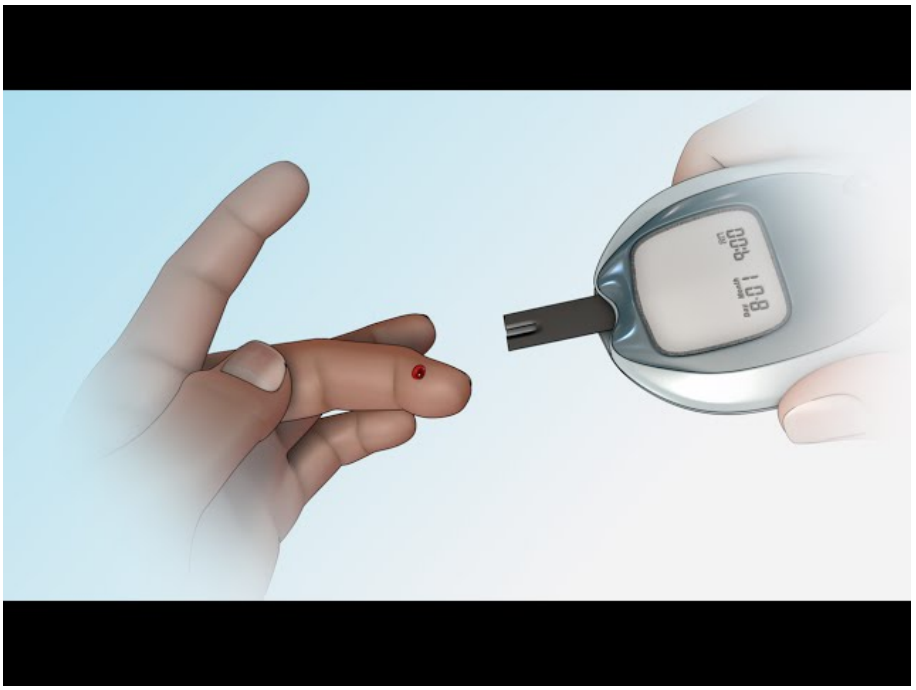

FIGURE 2.2: Traditional Finger-Prick Blood Glucose Testing Method

The GlucoSense project offers a ground-breaking non-invasive alternative, aiming to:

- Eliminate the discomfort and inconvenience associated with frequent finger-pricking.
- Provide reliable and real-time blood glucose monitoring using IR-based data acquisition.
- Utilize advanced signal processing and machine learning techniques for high-accuracy readings.
- Ensure portability, affordability, and accessibility for users across diverse healthcare settings, including remote and underserved regions.

The creation, evaluation, and implementation of the GlucoSense system are detailed in this thesis, showcasing its potential to revolutionize diabetes management by improving patient adherence, enhancing early intervention opportunities, and ultimately contributing to better healthcare outcomes.

2.2 What others have done

In recent years, significant progress has been made in leveraging technology, particularly machine learning and sensor-based systems, [2] for the non-invasive monitoring of glucose levels. Researchers and healthcare professionals have developed

various systems capable of estimating blood glucose levels without the need for finger-prick tests, utilizing innovative sensors and machine learning models.

One notable advancement is the use of infrared (IR) sensors, [3] which have been employed in several non-invasive glucose monitoring systems. These sensors are capable of detecting glucose levels by analyzing the interaction of infrared light with the skin. Advances in signal processing and calibration techniques have helped improve the accuracy and reliability of these readings.

Additionally, the integration of machine learning algorithms has played a crucial role in enhancing prediction precision. Neural network models, [4] trained on large datasets, can effectively estimate blood glucose levels by learning from various physiological features such as finger thickness, age, family history, and sensor readings.

Furthermore, researchers have explored the use of mobile applications to make glucose monitoring more accessible and convenient for users. These apps enable real-time tracking, history management, and trend generation, helping users manage their glucose levels more effectively.These non-invasive systems offer a user-friendly and less painful alternative to traditional glucose testing.

Moreover, advancements in signal amplification, filtering, and data processing have led to improved system accuracy, minimizing the impact of external factors such as ambient light and interference. These technologies are paving the way for future clinical integration and wider adoption in diabetes management.

Overall, the advancements made in non-invasive glucose monitoring have the potential to transform diabetes management, offering a more comfortable, accurate, and cost-effective solution for continuous glucose monitoring. Continued research and development in this field hold promise for better patient outcomes and more efficient healthcare delivery.

2.3 Literature Survey

This research [5] highlights the challenges associated with traditional glucose monitoring methods, which rely on invasive skin pricking. Regular finger-prick tests often cause discomfort, leading to poor adherence among patients. The study emphasizes the need for non-invasive solutions to improve patient comfort and promote consistent glucose level monitoring.

In this work, [6] NIR-based glucometers are investigated for non-invasive glucose

monitoring by utilizing near and mid-infrared light; however, they encounter challenges related to skin thickness variations and system complexity. Similarly, microwave sensing systems and metabolic heat confirmation techniques are explored, relying on body parameter measurements, but they demand high stability and accurate calibration. Furthermore, Raman spectroscopy and acousto-optic systems, combining light and ultrasound, are also considered, though they are significantly affected by motion artifacts and laser stability issues.

In this work, [7] multifactorial skin analysis techniques are explored for non-invasive glucose monitoring by examining various skin parameters such as color, texture, and temperature. Previous studies utilized these features to assess glucose levels, showing promising results. By adopting a combined parameter evaluation approach, researchers have demonstrated that integrating multiple skin characteristics can enhance the accuracy and reliability of non-invasive glucose measurements.

In this study, [8] machine learning algorithms are integrated into non-invasive glucose monitoring systems to enhance the accuracy and reliability of glucose measurements. Various models are trained to analyze multifactorial data such as skin texture, temperature, and optical responses. The effectiveness of these methods strongly depends on the quality of data collection and preprocessing. Continuous improvements in device calibration and algorithm optimization aim to ensure more precise and consistent glucose level predictions.

In this review work [9], a detailed analysis of non-invasive blood glucose monitoring technologies is presented, focusing on addressing the limitations of conventional invasive methods. The study explores various techniques including optical, microwave, and electrochemical methods, evaluating their principles, benefits, and challenges such as measurement accuracy and physiological interferences. It highlights the potential of these non-invasive solutions to enhance patient comfort, ensure continuous monitoring, and improve diabetes management, while also proposing future directions for expanding the scope of biomarker detection beyond glucose.

The study, [10] discusses the design and implementation of a non-invasive blood glucose meter that measures glucose levels using saliva instead of blood. The goal is to facilitate remote monitoring of diabetes through an Android application. The proposed glucometer uses electrochemical detection with platinum nanoparticles and the glucose oxidase enzyme. This non-invasive glucometer, equipped with Bluetooth connectivity, promises to enhance diabetes management. The device

aims to make glucose monitoring more convenient and painless, thereby improving patient compliance and health outcomes.

2.4 Scope of FYP

With the development and implementation of an innovative, non-invasive glucose monitoring system, the GlucoSense project aims to revolutionize healthcare diagnostics, particularly for diabetic patients seeking painless and real-time monitoring. By integrating advanced infrared sensing, signal processing circuits, and a machine learning-based prediction model into a compact and user-friendly device, the system ensures continuous and accurate glucose tracking without the need for blood samples.

The project's tasks include building a reliable hardware prototype, refining the prediction model for enhanced accuracy, designing a mobile application for real-time data visualization, and ensuring robust calibration mechanisms. Prioritizing accessibility, sustainability, and future scalability, GlucoSense seeks to make non-invasive glucose monitoring widely available, ultimately improving diabetes management and overall public health outcomes.

2.5 SDG Mapping

Several Sustainable Development Goals (SDGs) are in line with the Smart Non-Invasive Glucose Monitoring System, including:

- Good Health and Well-being (SDG 3) [11]: The technology promotes better health outcomes by offering a painless, non-invasive, and more accessible method for continuous blood glucose monitoring.
- Industry, Innovation, and Infrastructure (SDG 9) [12]: The development of GlucoSense showcases technological innovation by combining infrared sensing, signal processing, and machine learning for advanced healthcare solutions.

Chapter 3

System Design and its requirements

3.1 Overview of the System Design

The GlucoSense: Smart Non-Invasive Glucose Monitoring System is designed to provide a needle-free method for estimating blood glucose levels using infrared (IR) sensors and physiological measurements like finger thickness. It integrates hardware (IR sensors, amplifiers, Arduino Nano, ultrasonic sensor) with a machine learning model and mobile app to offer real-time predictions, historical tracking, and user-friendly data visualization for health monitoring.

3.2 Hardware Components and Setup

To achieve accurate and real-time non-invasive glucose estimation, the following hardware components were used and integrated in the GlucoSense system.

3.2.1 Infrared (IR) Sensor

To achieve reliable non-invasive glucose readings, infrared (IR) sensors are utilized to detect variations in light absorption when a finger is placed between the emitter and receiver. These sensors measure how much IR light [13] is absorbed or scattered by tissues, which correlates with blood glucose levels.

To ensure the accuracy of measurements, the IR setup is housed within a light-isolated enclosure that blocks ambient light interference and provides a controlled sensing environment.

- Key Features of IR Sensor Module:

  - Near-infrared operation for deeper tissue interaction.
  - Opposed emitter-receiver configuration for consistent sensing.
  - Enclosed setup to minimize external light effects.

3.2.2 TLP072 Operational Amplifier and Low-Pass Filter

The analog signals generated by the IR sensor are inherently weak; therefore, they are amplified using a TLP072 dual operational amplifier [14]. After amplification, signals are passed through a low-pass filter to remove unwanted high-frequency noise and retain only the meaningful physiological signal.

The amplification and filtering improve the quality and stability of the readings, which are crucial for reliable machine learning predictions.

- Key Features of Amplification and Filtering Circuit:
  - High-gain, low-noise amplification using TLP072 op-amps.
  - Cut-off frequency tuned to remove high-frequency noise.
  - Simple and efficient circuitry to ensure clean signal output.

3.2.3 Arduino Nano Microcontroller

An Arduino Nano microcontroller [13] acts as the brain of the GlucoSense system, interfacing with both the IR signal circuit and the ultrasonic sensor. It is responsible for reading analog values from the sensors, executing preprocessing calculations, and sending this data to the machine learning model.

Its compact form factor and reliable I/O capabilities make it ideal for embedding in portable healthcare applications. The Nano is also responsible for serial communication with external devices like mobile apps or data loggers.

- Specifications of Arduino Nano:
  - ATmega328P microcontroller.
  - 5V operating voltage.
  - Multiple analog input pins to handle IR and ultrasonic sensors.
  - Lightweight and compact for easy integration [15].

3.2.4 Ultrasonic Sensor (HC-SR04)

To personalize glucose predictions based on finger characteristics, an ultrasonic sensor (HC-SR04) [16] is used to measure the thickness of the user's finger. The sensor determines the distance between the finger and the sensor, which is converted into a radius, then used to calculate the circumference—a proxy for finger thickness.

This value becomes an additional input to the neural network model, improving its accuracy by adding anatomical context to IR readings.

- Key Features of Ultrasonic Sensor:
  - Accurate distance measurement from 2 cm to 400 cm.
  - Operating frequency of 40 kHz [17].
  - Simple integration with microcontrollers.

3.2.5 LCD Display Module (16x2 without I2C)

For real-time feedback during testing and calibration phases, a 16x2 LCD module is connected to the Arduino Nano. It displays the current sensor readings and system status, ensuring ease of monitoring without needing external devices.

The module uses an 8-bit parallel connection through D0 to D7 pins [18] , making it straightforward to interface with the microcontroller.

- Key Features of LCD Module:
  - Displays 16 characters per line across 2 lines.
  - Backlight for clear visibility.
  - Direct wired connection without the use of an I2C interface.

3.2.6 Power Supply (9V Battery)

The entire system is powered by a 9V battery [19] , making it portable and independent of external power sources. A voltage regulation circuit ensures stable 5V operation for the Arduino Nano and sensor modules.

Using a battery allows the GlucoSense device to be easily transported and used in various environments, enhancing its usability in field applications.

- Power Supply Considerations:

  - 9V battery source for portability.
  - Voltage regulator ensures stable 5V output.
  - Easy replacement and maintenance.

3.3 Software

3.3.1 System Requirements

To run the GlucoSense system effectively, the following system requirements must be met:

- **Operating System:** The system is compatible with widely used operating systems such as Windows and macOS.
- **Processor:** A multi-core processor is recommended to efficiently handle sensor data processing, machine learning inference, and mobile application integration.
- **Memory (RAM):** A minimum of 8GB RAM is suggested for smooth operation; however, 16GB or more is beneficial for faster processing and handling large datasets.
- **Storage:** Adequate storage is required to maintain user profiles, glucose prediction logs, and machine learning models. SSD storage is preferred to enhance data retrieval and processing speeds.
- **Network Connectivity:** Reliable internet connectivity is essential to support real-time communication between the Arduino hardware and the GlucoSense mobile application through cloud services.
- **Microcontroller Platform:** An Arduino Nano is used as the primary controller for collecting sensor data and transmitting readings to the processing system.
- **Mobile Platform:** The Flutter-based GlucoSense app ensures seamless access and visualization of predicted glucose levels on smartphones. It is compatible with both Android and iOS devices.
- **Database Management System:** A lightweight cloud database system is integrated with the mobile application to store user-specific data, including glucose predictions and historical trends.

By ensuring that the system meets these requirements, users can experience real-time glucose monitoring, secure data storage, and smooth integration between hardware and mobile platforms through GlucoSense.

3.3.2 Development and Analysis Tools

The software development involved multiple environments and libraries for hardware interfacing, data processing, and model deployment. The breakdown is given below:

Arduino IDE and Embedded C++ (Firmware Development)

- **Analog Sensor Reading:**
  - The Arduino continuously measured voltage outputs from the IR sensor after amplification and low-pass filtering stages.
  - The ADC (Analog-to-Digital Converter) of the Arduino Nano read analog voltages corresponding to the non-invasive glucose signals.
- **Signal Filtering:**
  - To minimize noise and fluctuations, a hardware low-pass filter was designed.
  - Software-side averaging techniques were also implemented to smooth out the readings.
- **Serial Communication:**
  - Sensor data is transmitted via serial communication to a connected system (such as a PC or mobile device) for subsequent processing.
- **Purpose:**
  - Arduino firmware provides a lightweight mechanism to accurately acquire non-invasive glucose signals and pre-process them before feeding into the prediction model.

Python Libraries (Neural Network Model Development)

- **Data Handling and Preprocessing:**
  - `numpy`: For managing arrays and mathematical operations.
  - `pandas`: For structuring, cleaning, and manipulating the dataset containing non-invasive features and corresponding glucose levels.

- **Model Building and Training:**
  - `tensorflow` and `keras`: Used to define and train a sequential neural network model consisting of:
    * Two hidden layers with 64 and 32 neurons respectively, using ReLU activation.
    * An output layer predicting a continuous glucose value.
    * Compilation: The model is compiled with the Adam optimizer and mean squared error (MSE) loss.
- **Model Evaluation:**
  - `scikit-learn`: Employed for train-test splitting, feature standardization using StandardScaler, and calculating evaluation metrics such as MAE, RMSE, and R^2 score.
- **Visualization:**
  - `matplotlib` and `seaborn`: Used to plot correlations, feature distributions, and model performance graphs during experimentation and result analysis.
- **Purpose:**
  - Python-based machine learning pipelines process the non-invasive sensor data along with patient meta-data (e.g., age, meal status) to predict invasive glucose values with high accuracy.

Google Colab (Cloud-Based Model Development)

- **Google Drive Integration:**
  - `drive.mount`: Enables accessing datasets stored on Google Drive for model training.
- **Cloud-Based Notebooks:**
  - Utilized for leveraging free GPUs, faster model training, collaborative development, and resource-efficient experimentation.
- **Purpose:**
  - Google Colab offers an accessible cloud environment for rapid prototyping, training, and evaluation of the neural network without requiring expensive local hardware.

3.3.3 Flutter (Mobile Application Development)

To provide a user-friendly and accessible experience, a dedicated Android application was developed using Flutter and integrated with backend services. The key aspects of the mobile application development are as follows:

- **Platform:** The mobile application is developed using Flutter, allowing for a responsive, attractive, and cross-platform interface optimized for Android devices.
- **Programming Language:** The app is primarily coded in Dart, ensuring fast compilation and smooth runtime behavior.
- **User Interface (UI):**
  - Designed using Flutter widgets to offer an intuitive and seamless navigation experience.
  - Integrated real-time glucose value displays, historical tracking charts, and user profile management features.
  - Followed the GlucoSense color theme: green (#48872B), white, and black for a consistent visual identity.
- **Backend Connectivity:**
  - The Flutter application communicates with a Flask backend server via HTTP requests.
  - Data related to glucose levels and user profiles is fetched, displayed, and updated through secured API endpoints.
- **Database Management:**
  - The backend server is connected to a MySQL database managed via PHP scripts, which ensures structured storage and retrieval of user data, glucose readings, and historical records.
- **Data Visualization:**
  - Real-time glucose trends and historical reports are visualized within the app using dynamic charts and tables, allowing users to monitor their health effortlessly.
- **Authentication and Session Management:**

- The app securely manages user sessions and personal data, ensuring privacy and providing a customized experience for each user.

By developing the mobile application with Flutter and integrating it with robust backend services, GlucoSense ensures real-time, accurate, and personalized glucose monitoring for users. The seamless interaction between hardware, machine learning models, and the mobile application offers a complete end-to-end non-invasive glucose tracking solution.

3.3.4 Deployment Diagram of System

The process and structural elements of our GlucoSense Smart Non-Invasive Glucose Monitoring System are depicted in detail in the deployment diagram shown in Figure 3.1. This figure illustrates the important phases, from sensor data acquisition using IR and ultrasonic sensors, to signal amplification, followed by detection and glucose level prediction through machine learning models. It also shows how report generation is handled and how the results are integrated into the Flutter-based mobile application, allowing users to view real-time glucose readings and historical trends. The diagram highlights the seamless flow of data between hardware components, processing modules, backend services, and the user interface, ensuring a comprehensive and user-friendly experience.

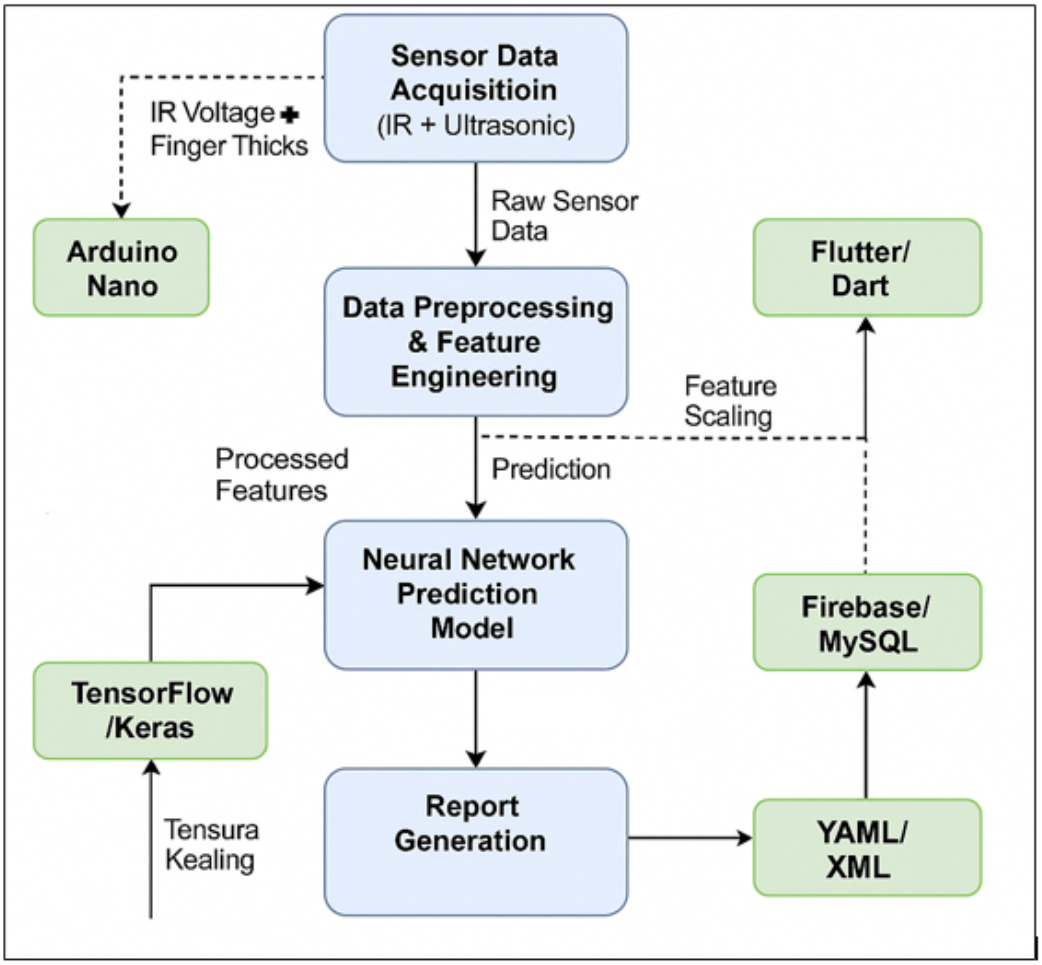

FIGURE 3.1: Deployment Diagram

3.4 Proposed Project Plan

3.4.1 Preparing Data

The project will commence with extensive efforts to collect non-invasive sensor data, including infrared signals, ultrasonic sensor outputs, and manually recorded invasive glucose levels for validation purposes. Data will be gathered under controlled conditions, ensuring proper categorization based on physiological states such as fasting, after meal, and random glucose measurements. Ethical guidelines, informed consent, and data privacy protocols will be strictly followed throughout the data acquisition process to maintain confidentiality and integrity.

3.4.2 Data Pre-processing

Upon successful data collection, the pre-processing phase will begin to enhance the quality and consistency of the gathered dataset. This will involve noise reduction techniques to eliminate fluctuations and artifacts from the sensor readings, normalization of signal values for uniformity, and feature scaling to prepare the data for model training. Any irrelevant or redundant features will be identified and removed to ensure the dataset remains focused, clean, and suitable for efficient machine learning model development.

3.4.3 Applying Different Techniques

With a well-prepared dataset, multiple machine learning algorithms will be applied to analyze and predict glucose levels accurately. A neural network model [20] , trained on the extracted sensor features, will be utilized to establish non-invasive glucose prediction capabilities. Feature engineering techniques will be employed to derive meaningful attributes from the sensor signals. The trained model will then be integrated with a backend system for real-time prediction and further deployed onto the mobile application for user accessibility.

3.4.4 Analyzing Results

After model deployment, a comprehensive evaluation will be conducted to assess the accuracy, reliability, and usability of the GlucoSense system. Performance metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared values will be calculated to quantify prediction performance. Additionally, user feedback on the mobile application's interface, report clarity, and overall experience will be collected. Any challenges encountered during testing phases will be documented, and iterative improvements will be made to refine system accuracy and user satisfaction.

3.5 Visualization and Process Representation

3.5.1 Gantt Chart

The project timeline is organized through a Gantt chart [21] that sequences all development phases from hardware prototyping to clinical validation. As given in Figure: 3.2, this visualization tool enables efficient resource allocation and milestone tracking, particularly for coordinating parallel workstreams like sensor calibration and mobile app development.

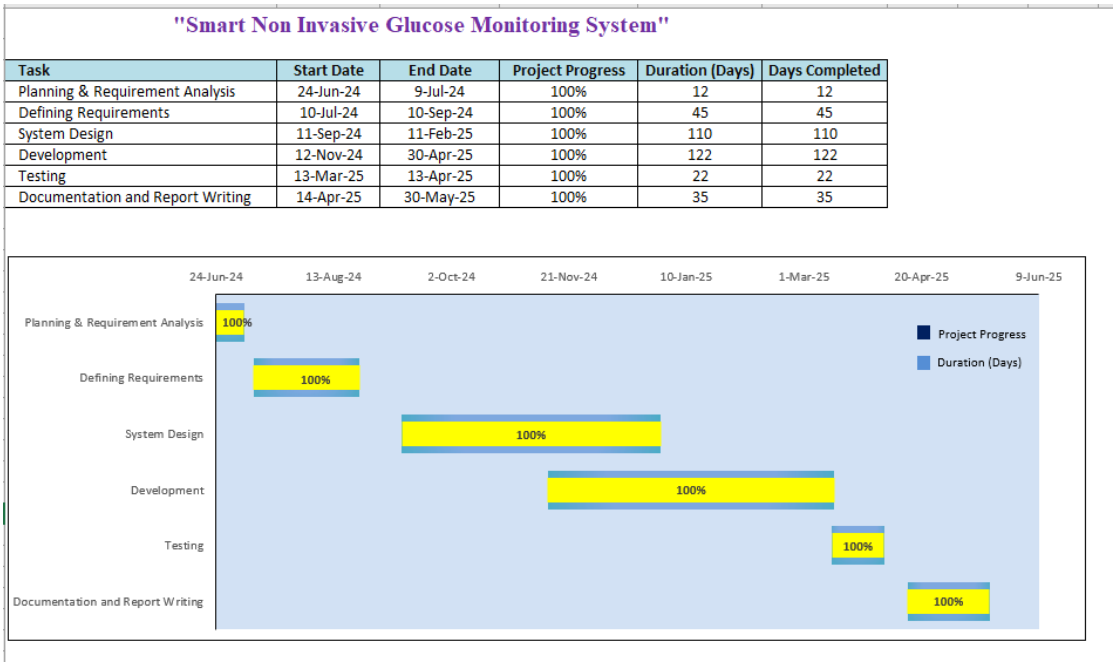

FIGURE 3.2: Gantt Chart

3.5.2 Data Flow Diagrams

The data flow diagrams [22] are presented to illustrate the flow of data involved in the Smart Non Invasive Glucose Monitoring System. This graphical representation shown in figures 3.3, 3.4, and 3.5 visually depicts the workflow of the system, outlining the sequence of operations from sensor data acquisition to glucose level prediction and report generation.

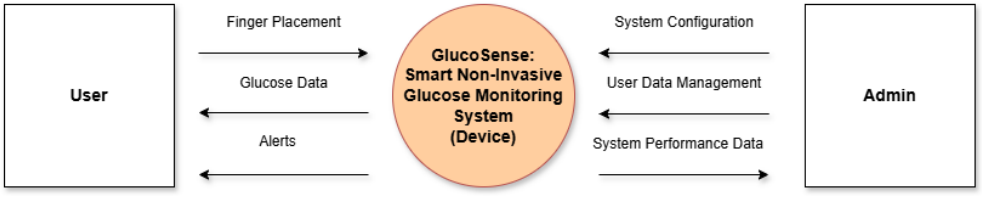

FIGURE 3.3: Level-0 DFD

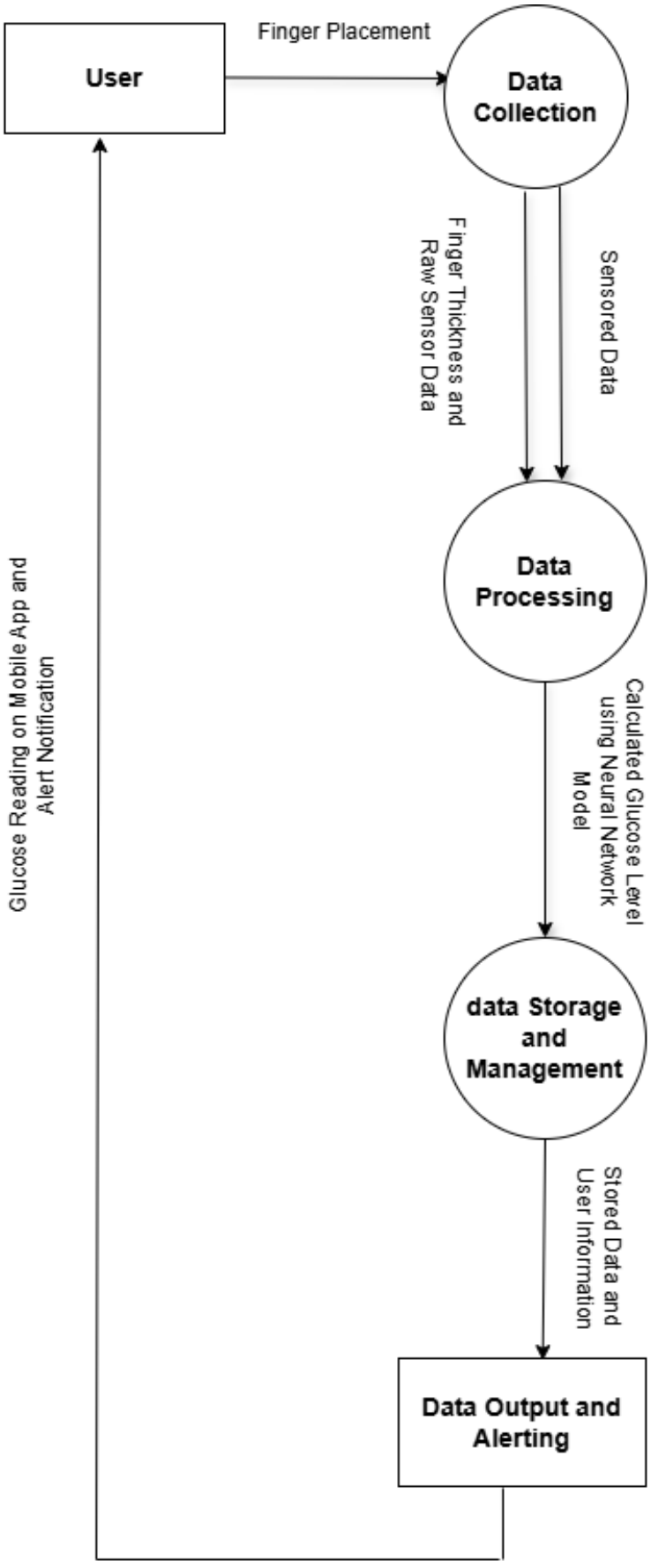

FIGURE 3.4: Level-1 DFD

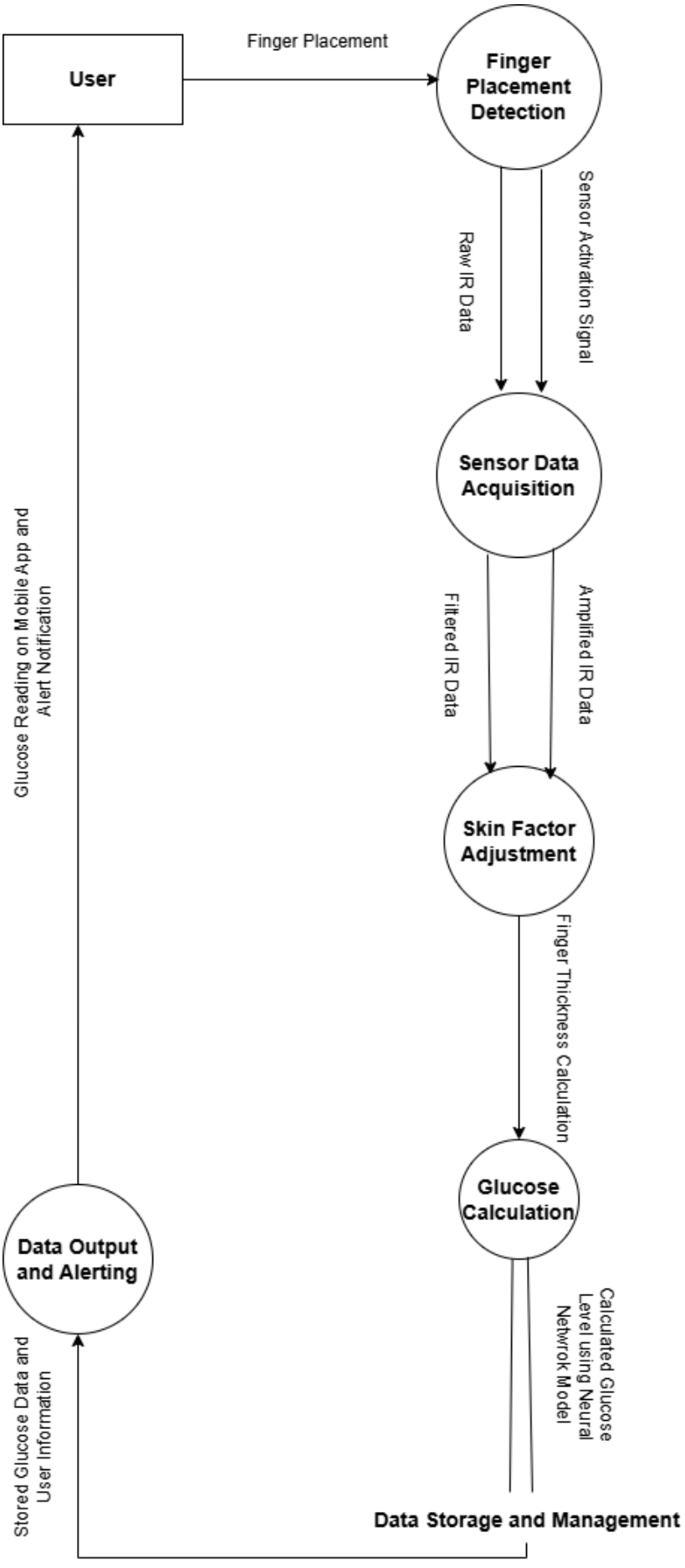

FIGURE 3.5: Level-2 DFD

3.5.3 State Transition Diagram

The state transition diagram [23] for the GlucoSense system visually maps how the system transitions between different states such as idle, data collection, data processing and alert generation as given in figure: 3.6. It assists in identifying potential issues, optimizing the workflow, and ensuring reliable performance for real-time glucose monitoring and timely user notifications.

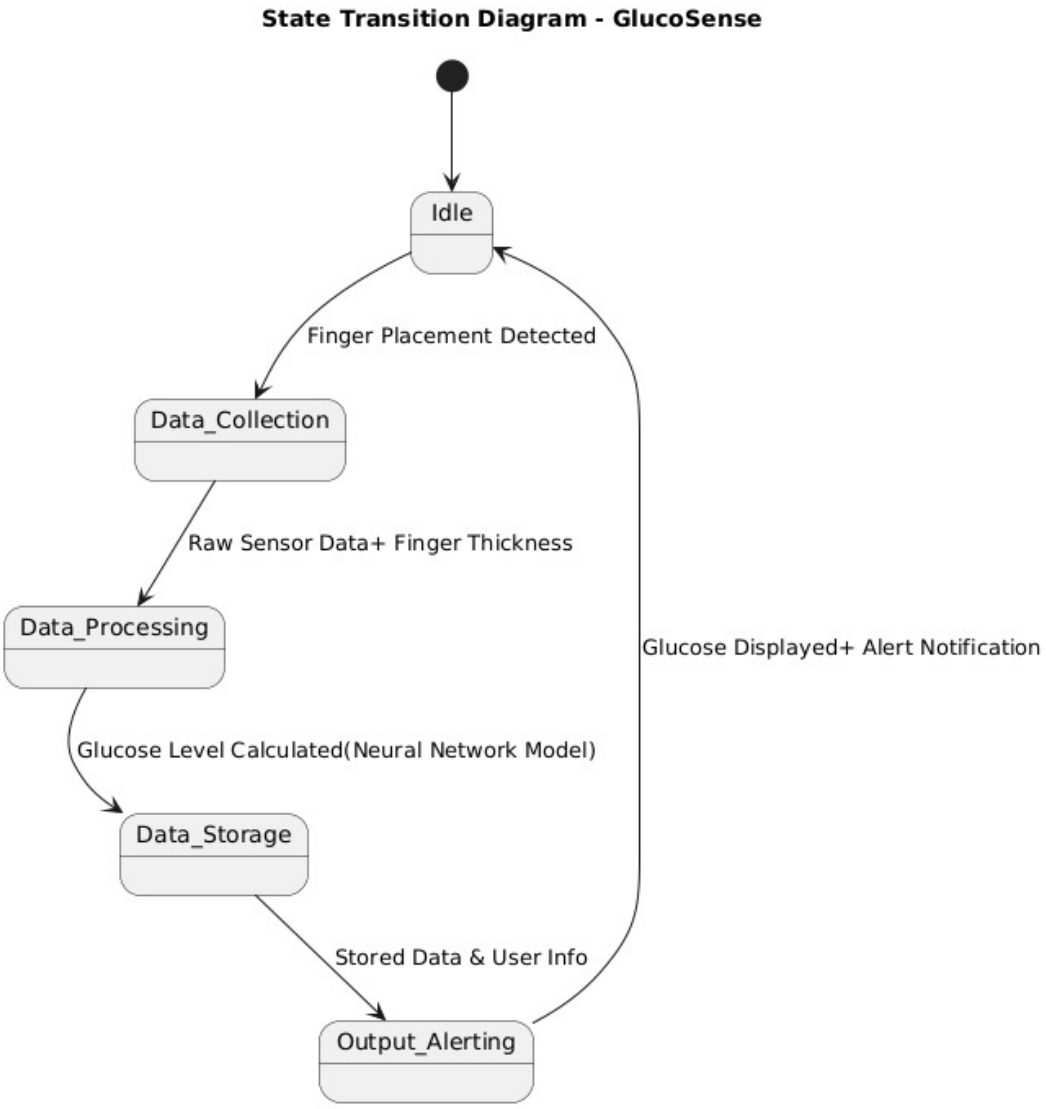

FIGURE 3.6: State Transition Diagram

3.5.4 Sequence Diagram

In UML (Unified Modeling Language), a sequence diagram [24] is a kind of interaction diagram that shows how different objects or system components interact with one another over time. It displays the sequence of messages sent between objects to execute a certain job or situation, providing a thorough perspective of

the runtime behavior of the system. Sequence diagrams are very helpful in helping developers understand the order of events and communication patterns during system execution by showing the flow of data and control among various system components.

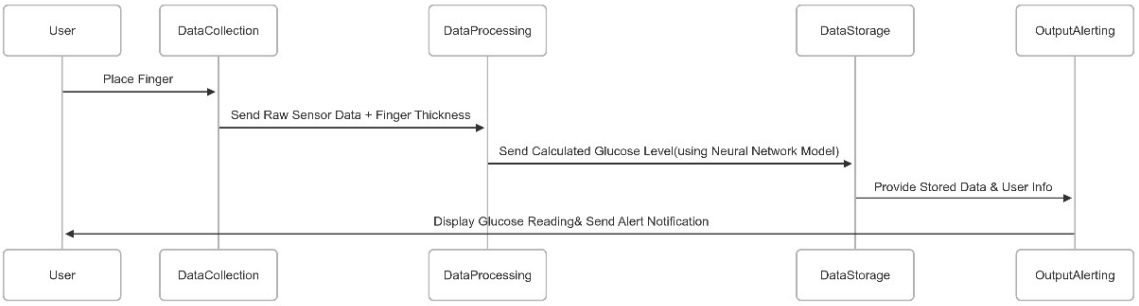

FIGURE 3.7: Sequence Diagram

The sequence diagram given in figure: 3.7 outlines the sequence of actions from sensing glucose levels, transmitting the data to the microcontroller, storing information in the database, and updating the mobile app. The process begins with the user logging into the GlucoSense mobile application. Once authenticated, the microcontroller collects real-time data from the IR glucose sensor and the ultrasonic sensor. This data is then transmitted to the Phpadmin database for storage. The mobile application requests the latest glucose readings from the database and displays the results to the user. If abnormal glucose levels are detected, the system generates a timely alert through the mobile app. The database also saves session information for continuous monitoring and future reference.

3.5.5 Use Case Diagram

An illustration of the interactions between different actors or users and the functionality of the system is provided by a use case diagram [25]. It offers a high-level summary of how the system behaves as seen from the eyes of various user roles. Use case diagrams typically consist of use cases, which show the particular functions or actions that users can carry out within the system, and actors, which represent the various user types engaging with the system. Use case diagrams aid in the understanding of the system's scope and intended functionality by showing the relationships between users and functionalities. In the early phases of system development, they are useful tools for communicating and capturing user requirements and system behavior.

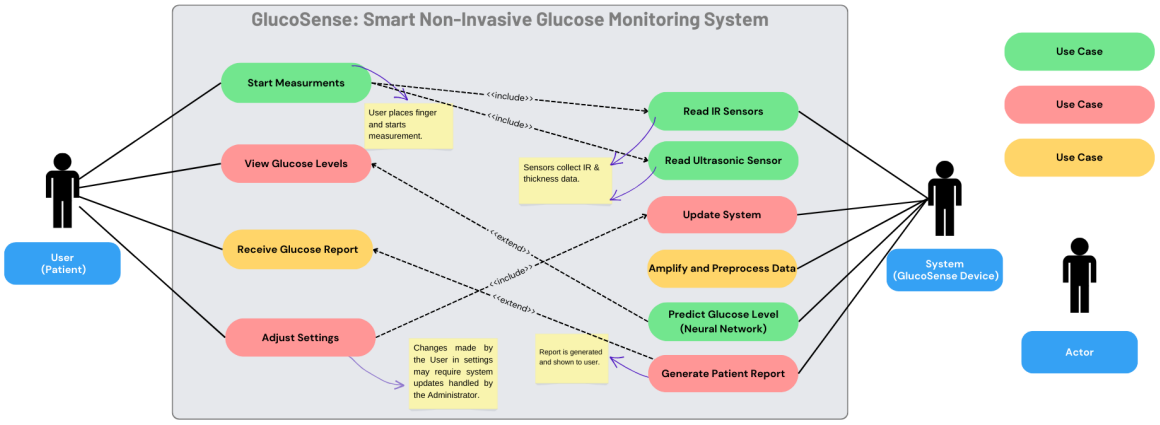

FIGURE 3.8: Use Case Diagram

The figure: 3.8 showcases the engagement between users, such as the Admin and Simple User, and the system's functionalities. In this context:

Actors involved encompass the User, responsible for starting glucose measurements, viewing glucose levels, receiving alerts, viewing history, adjusting settings, and reporting issues, and the Administrator, who manages the system's backend operations. Use cases outlined include:

- Administrator privileges encompass managing the sensors connected to the system, updating the system software, monitoring data transmission to ensure accuracy, and troubleshooting any reported issues to maintain smooth

system functionality.

- User capabilities entail initiating glucose measurements, monitoring their glucose levels in real time, receiving alerts for abnormal readings, accessing historical data, adjusting system settings for personalization, and reporting any issues faced during usage.

3.5.6 Interaction Flow Chart

This provides a visual representation [26] of the various pathways and interactions between users and the system interface. It outlines the sequence of user actions, system processes, and feedback mechanisms, highlighting the flow of information and control within the system. By delineating the steps involved in user-system interactions, the interaction flowchart shown in figure: 3.9 begins with user authentication, followed by sensor initialization and data collection. The collected sensor data is processed by the microcontroller and forwarded to the Phpadmin database. The mobile application fetches this processed data to display glucose readings and alerts. This diagram simplifies the understanding of the user journey and ensures smooth and efficient interaction between the user and the system components.

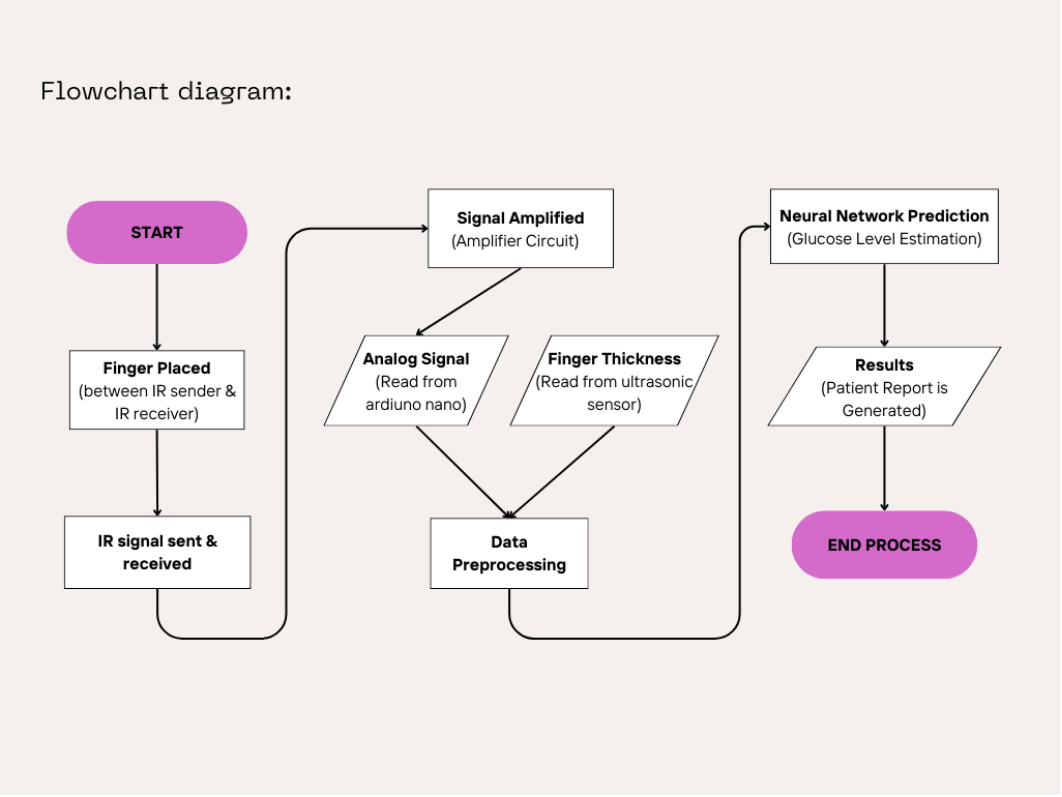

FIGURE 3.9: Interaction FlowChart

3.6 General Proposed Model

GlucoSense: Smart Non-Invasive Glucose Monitoring System is designed as a complete, integrated framework that combines advanced sensing technology, real-time data processing, machine learning prediction models, and mobile application interfaces to provide a non-invasive, accurate method for glucose monitoring. The model connects multiple hardware and software components to capture biological signals, process them intelligently, and deliver meaningful glucose level information to users, offering an accessible and efficient solution for diabetes management.

1. **Overall Architecture**: The overall architecture of the proposed model starts with the acquisition of raw biological signals using an infrared (IR) sensor and an ultrasonic sensor. The IR sensor captures optical responses related to blood glucose concentrations, while the ultrasonic sensor measures finger thickness to enhance the accuracy of the glucose prediction. These analog signals are then passed through a hardware-based amplification and low-pass filtering stage using TLP072 operational amplifiers to reduce noise and preserve signal quality. The processed signals are read by the Arduino Nano microcontroller, which formats and transmits the data wirelessly to a server. A trained neural network model hosted on the server analyzes the incoming sensor data to predict blood glucose levels. The predicted results are stored in a phpMyAdmin (MySQL) database and simultaneously sent to a Flutter-based mobile application, where users can view their glucose readings, track their history, and receive health alerts in real time.

2. **Components**: The system comprises several key components working together harmoniously. The IR sensing unit detects glucose-related optical signals, and the ultrasonic sensor measures finger dimensions, offering additional contextual data. The amplification and low-pass filter circuit conditions the sensor signals for accurate analog-to-digital conversion by the Arduino Nano. The microcontroller plays a crucial role in aggregating sensor data and forwarding it to the backend server. On the server side, a neural network model processes the data to output glucose level predictions. The phpMyAdmin database securely manages user profiles, historical data, and prediction results, while the Flutter mobile app provides an interactive platform for users to start measurements, view glucose trends, receive alerts, and manage system settings.

3. **Technology Integration**:The GlucoSense system integrates a range of technologies to achieve its functionality. It combines optical and ultrasonic sensing methods with analog signal conditioning techniques to ensure precise signal acquisition. The Arduino Nano facilitates real-time data handling and transmission. The backend employs a machine learning model, specifically a neural network trained on relevant features extracted from sensor data, to perform non-invasive glucose prediction. Data storage and management are efficiently handled through phpMyAdmin, offering structured and secure database operations. Finally, a Flutter-based cross-platform mobile application ensures users can easily access and interact with their data, receive timely notifications, and adjust system settings as needed, providing a seamless end-to-end user experience.

4. **Flexibility and Scalability**: The proposed model is designed with flexibility and scalability in mind to adapt to different user needs and future technological advancements. It can easily incorporate additional features such as new types of sensors, more sophisticated machine learning models, or integration with broader healthcare management systems. The system architecture allows for modular updates, where individual components like the prediction model or the mobile application can be upgraded independently. This ensures that GlucoSense can evolve with the growing demands of the healthcare industry and remain compatible with newer standards for medical devices and patient data security.

5. **Anticipated Results**: The GlucoSense system is expected to deliver accurate and timely glucose level predictions without the need for invasive blood sampling. Users will receive easy-to-understand reports through the mobile application, including trend graphs, historical data analysis, and real-time alerts when glucose levels are out of the normal range. The system aims to empower users with actionable insights into their health, improve daily diabetes management, and potentially reduce the number of invasive blood tests required.

6. **Validation and Assessment**: Extensive validation and assessment processes are planned to ensure the system's accuracy, reliability, and usability. Initial validation will involve testing the system's predictions against standard invasive glucose measurements to establish clinical reliability. The system will also undergo independent dataset testing, real-world pilot studies with users, and iterative improvements based on user feedback and observed

system performance. Continuous monitoring will be maintained to ensure the machine learning model remains effective across a diverse range of users and usage conditions.

7. **Timeline and Milestones**: The development of GlucoSense will follow a well-structured timeline, beginning with sensor data collection and initial circuit design, followed by machine learning model training and integration with the microcontroller and server database. Testing phases will cover both hardware performance and software prediction accuracy. Mobile application development will run parallel to backend development to ensure smooth integration. Regular evaluations and refinements will be conducted at each stage to ensure timely progress towards the final goal of delivering a reliable, efficient, and user-friendly non-invasive glucose monitoring solution.

3.7 Proposed System

The proposed complete architecture of the system is given in figure: 3.10:

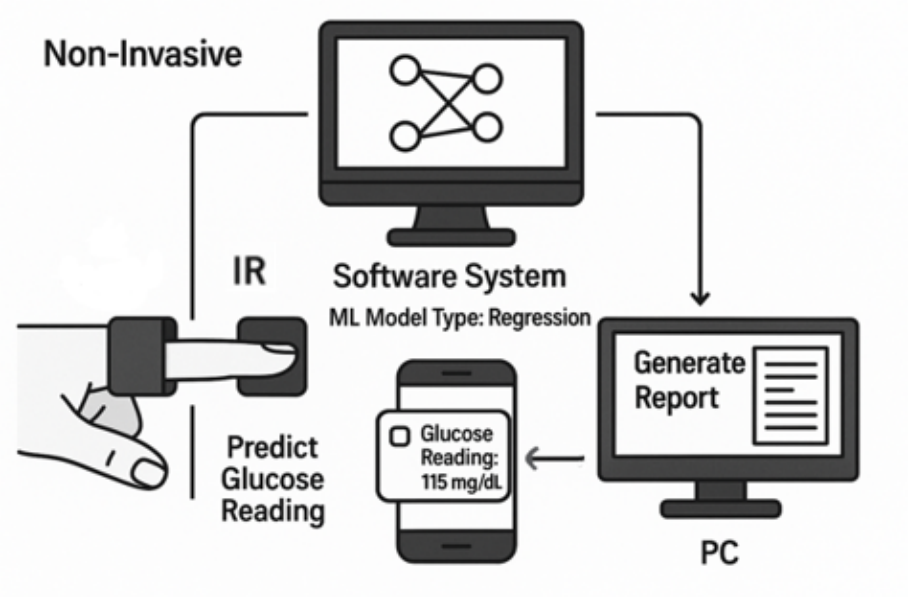

FIGURE 3.10: Proposed System

Chapter 4

Implementation and Experiments

In order to successfully accomplish its objectives, the GlucoSense Smart Non-Invasive Glucose Monitoring System required a systematic strategy that integrated hardware development, software modeling, and machine learning techniques. The following sections describe the complete implementation and experimental procedures of the system:

4.1 Collecting Raw Dataset

Careful preparation and strict adherence to ethical guidelines were necessary during the raw data collection procedure for the GlucoSense system to guarantee the accuracy and applicability of the gathered information.

4.1.1 Sources and Acquisition Methods

- The sources of the dataset included real-world data collection at **Mayo Hospital Lahore** and **General Hospital Lahore**, focusing specifically on diabetic and non-diabetic patients.
- We collected anonymized patient samples through direct collaboration with hospital staff. Each patient underwent both traditional invasive glucose measurement (finger prick blood glucose test) and our non-invasive GlucoSense device reading simultaneously.
- Non-invasive sensor readings were collected using the GlucoSense prototype, which included an IR sensor (amplified and filtered using TLP072 op-amps), an ultrasonic sensor for finger thickness estimation, and an Arduino Nano for data acquisition.

4.1.2 Data Types

- The collected raw data encompassed invasive blood glucose readings (measured using standard medical glucometers) and non-invasive sensor readings obtained from the GlucoSense device.
- Additional features such as age, weight, gender, medical history, meal status (fasting or post-meal), and finger thickness were also recorded for each patient to enhance model prediction accuracy.

4.1.3 Criteria for Dataset Selection

- Patients of different age groups, genders, and medical histories were included to ensure the dataset represents diverse demographics and physiological conditions.
- Emphasis was placed on collecting data both in fasting and postprandial (after meal) conditions to train a robust model capable of handling varying glucose level states.
- Data integrity was ensured by synchronous collection of invasive and non-invasive readings under the same environmental conditions (room temperature, lighting, etc.).

4.1.4 Ethical Considerations and Compliance Requirements

- Ethical considerations were paramount throughout the data collection process, with strict adherence to patient privacy regulations. No personally identifiable information (PII) was recorded.
- Data collection procedures followed hospital policies and obtained verbal consent from all participants.
- The gathered data was used exclusively for research purposes and stored securely to protect patient confidentiality.

4.2 Dataset Acquisition and Description

We used a primary source for data collection by building our own dataset through real-time data acquisition in collaboration with healthcare institutions. This ensured that the dataset closely reflects real-world clinical scenarios and diverse patient demographics relevant for non-invasive glucose monitoring.

4.2.1 Data Source

- **Primary Data Source:**

  The primary data was collected manually from patients in Mayo Hospital Lahore and General Hospital Lahore. Each patient's non-invasive sensor readings were taken using the GlucoSense prototype, immediately followed by a standard invasive blood glucose test using medically approved glucometers.

- **GlucoSense Device Measurements:** Non-invasive readings using our GlucoSense device,utilized an IR sensor (with amplification and filtering through TLP072 op-amps) to capture reflected light intensity from the finger tissue, and an ultrasonic sensor to estimate finger thickness, which influences glucose prediction accuracy.

- **Invasive Readings:** Blood glucose levels were measured invasively to serve as the ground truth label for model training and validation.

- **Compiled Dataset:** A comprehensive `GlucoSense_dataset.csv` [27] was created from the collected information, containing both input features (sensor readings, demographic details) and corresponding invasive glucose reference values (targets for machine learning model training).

4.2.2 Data Collection Process

- Patients were requested to place their finger between the IR sensor and receiver assembly of the GlucoSense device. Non-invasive sensor readings were captured after amplification through TLP072 op-amps and noise filtration via low-pass filters.

- At the same time, our project team manually performed the standard invasive blood glucose measurements using hospital-approved glucometers and finger-prick techniques to ensure accurate reference readings.

- Additional contextual information such as fasting/post-meal status, age, weight, and gender was also manually recorded during the data collection sessions.

- All collected data was organized systematically in structured Excel sheets, ensuring each patient's non-invasive readings, demographic data, and invasive glucose measurements were linked accurately. The structured data was

then exported into a final CSV format, named `GlucoSense_dataset.csv`, for model training and evaluation.

- Strict hygiene protocols, such as sterilizing the GlucoSense device and using disposable lancets for invasive tests, were followed diligently during the entire data collection phase to ensure patient safety and data reliability.

- The dataset was acquired through direct collaboration with Mayo Hospital Lahore and General Hospital Lahore, where patient volunteers were enrolled under ethical compliance procedures. Working closely with hospital staff ensured authentic data collection while maintaining strict patient privacy and data protection standards.

4.2.3 Data Description

The `GlucoSense_dataset.csv` contains 151 samples collected from real patients at Mayo Hospital and General Hospital, Lahore. The dataset includes both non-invasive sensor readings and corresponding invasive blood glucose measurements.

Non-invasive Sensor Features

- IR Sensor Output (after amplification and filtering)
- Ultrasonic Sensor Measurement (finger thickness)

Patient Features

- Age
- Gender
- Weight
- History
- Medical Condition (Diabetic / Non-Diabetic)
- Meal Status (Fasting / Post-Meal)

Label

- Invasive Blood Glucose Level (mg/dL)

Dataset Statistics

- Total Samples: 151 (as per the latest updated file)
- Age Range: From teenagers to elderly patients (diverse representation)
- Glucose Levels: Covering normal, prediabetic, and diabetic ranges
- Balanced Distribution: Data collected from both male and female patients across different meal conditions

The collected dataset underwent a comprehensive and systematic acquisition process, with active collaboration from medical professionals to obtain patient samples, invasive glucose readings, and corresponding non-invasive sensor data. Throughout the process, strict ethical standards and patient confidentiality protocols were maintained to ensure data integrity and security.

4.3 Data Analysis and Visualisation

Preparing the raw dataset for analysis and model training involved several essential steps to ensure its readiness for machine learning tasks and to improve the robustness of the non-invasive glucose monitoring system. For a better understanding of the dataset characteristics, data analysis was performed using different visualization techniques.

For better understanding we did data analysis. For this, we used different data visualisation techniques.

4.3.1 GlucoSense Dataset

The fdisplays multiple visualizations generated from the GlucoSense dataset to better understand the distribution, trends, and relationships among the collected features.

Here's a description of each plots:

- The first plot as shown in figure 4.1 is a pie chart showing the proportion of diabetic and non-diabetic participants. The chart is divided into two segments labeled "Diabetic" and "Non-Diabetic," illustrating the balance between the two classes in the dataset. This provides a quick overview of the dataset's classification distribution relevant for training and evaluating predictive models.

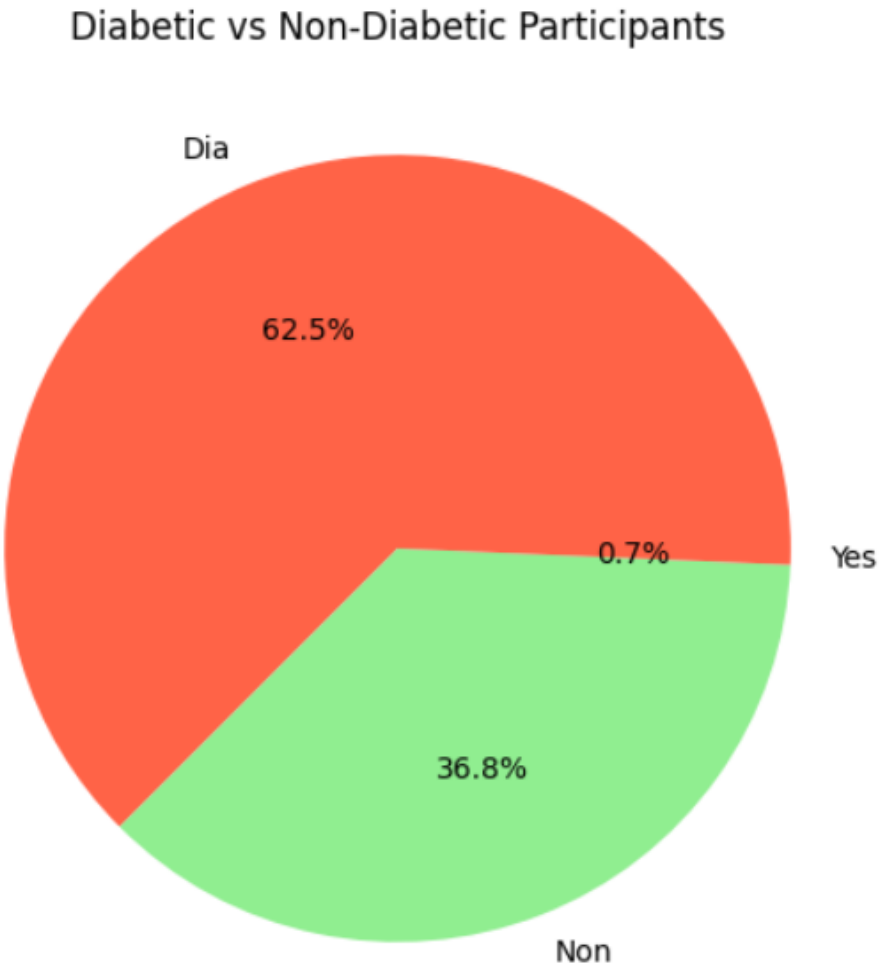

FIGURE 4.1: Diabetic vs Non-Diabetic Participants

- The second plot is a line chart as shown in figure 4.2 comparing "Invasive" and "Non-Invasive" glucose readings for each participant. Two separate lines are plotted: one for the invasive glucose measurements and another for the non-invasive measurements. The x-axis represents participant numbers, while the y-axis shows the glucose levels. This chart highlights the trend and differences between the two methods across all participants.

- The third plot is a heatmap as shown in figure 4.3 displaying the correlation among numerical features such as Age, Weight, Thickness (cm), Invasive glucose level, and Non-Invasive glucose level. The colors range from light to dark, indicating weaker to stronger correlations. This visualization helps identify relationships between variables, which can guide feature selection and modeling decisions in the glucose prediction system.

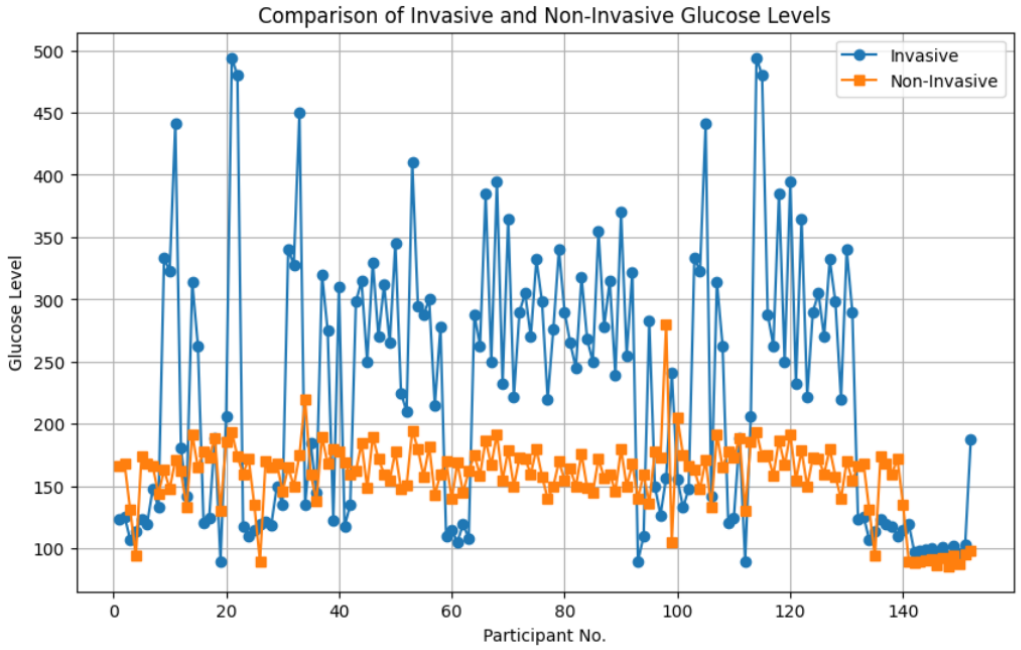

FIGURE 4.2: Line Chart of Invasive and Non-Invasive Glucose Levels

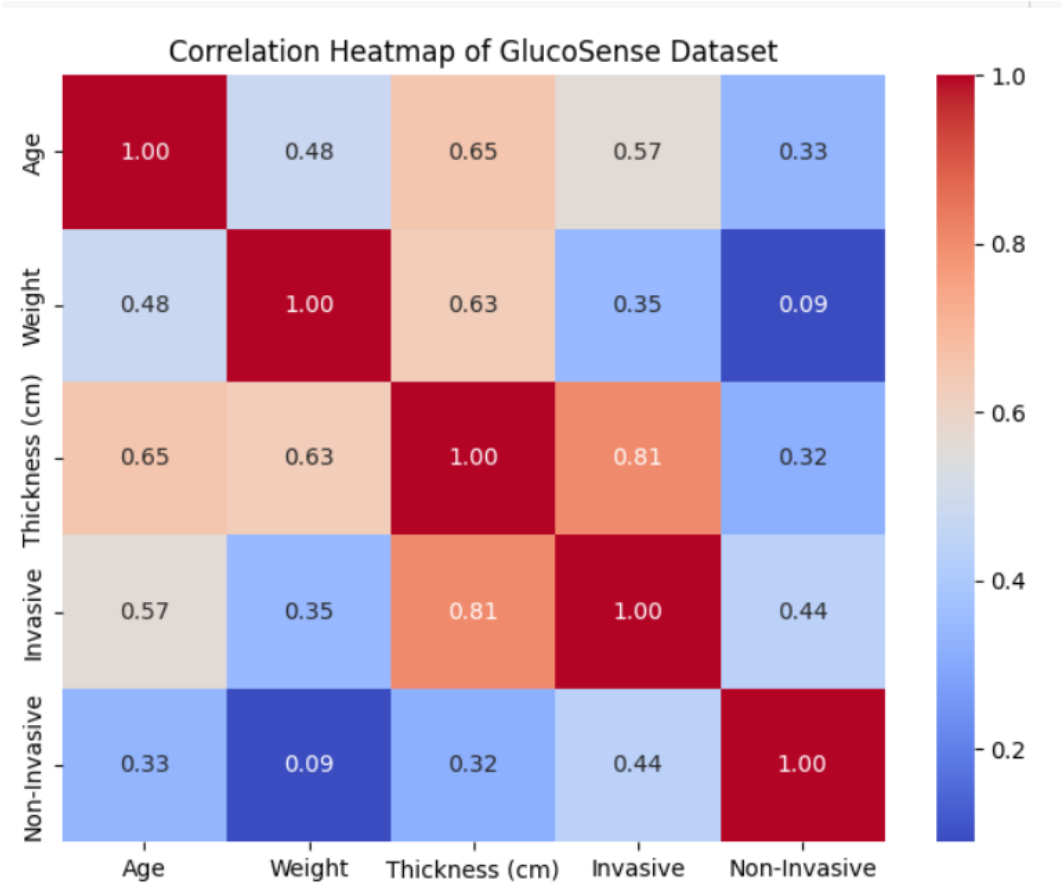

FIGURE 4.3: Correlation Heatmap of GlucoSense Dataset

4.4 Data Cleaning and Preprocessing

The preprocessing steps applied to the dataset were essential for extracting meaningful features and standardizing the data for analysis, particularly for the GlucoSense: Smart Non-Invasive Glucose Monitoring System. Since the project relies heavily on sensor readings captured non-invasively (via IR sensor, ultrasonic sensor, and processed through neural networks), proper cleaning and preprocessing of the single dataset was crucial for ensuring accurate predictions.

4.4.1 Sensor Data Preprocessing Techniques

- Analog Signal Conditioning: Before digitization, the analog signals from the IR receiver and ultrasonic sensor were passed through TLP072-based low-pass filter circuits to suppress high-frequency noise. This hardware filtering step helped stabilize the raw signal and remove minor fluctuations caused by ambient light or interference.

- Voltage Scaling: The Arduino Nano handles a 10-bit ADC with a 0–5V range, appropriate voltage dividers and gain adjustments were applied using the amplifier circuit to ensure that the received signal stayed within measurable bounds, maximizing ADC resolution.

- Averaging for Stability: To reduce noise and improve stability, multiple analog readings were taken over a short time window and averaged. This software-based moving average helped reduce random fluctuations and provided smoother data to the prediction model.

4.4.2 Feature Extraction Methods

In the GlucoSense system, feature extraction is performed from patient information and sensor measurements. After collecting raw data through IR sensors and other inputs, specific important features are selected and extracted to train the neural network model. The extracted features include gender (encoded as Male = 1, Female = 2), age (in years), weight (in kilograms), history of diabetes (encoded as Yes = 1, No = 0), and meal/fast status (encoded numerically as Fasting = 0, Two hours after lunch = 1, Random = 2, or a custom value like 1.5). Additionally, diabetic status (based on user declaration: Diabetic = 1, Non-Diabetic = 0), finger thickness (measured using an ultrasonic sensor in centimeters), and a non-invasive value (calculated by dividing the IR sensor voltage reading by 6) are also extracted. These features are carefully chosen because they show significant correlations with glucose levels according to medical literature and experimental

analysis. A correlation heatmap was also generated to visualize the relationships among features, confirming that these factors influence blood glucose levels.

4.4.3 Addressing Preprocessing Challenges

- Computational Complexity: To optimize computational efficiency, preprocessing tasks such as data cleaning, encoding, and scaling were streamlined using optimized algorithms within the Google Colab environment. Efficient execution ensured smooth handling of dataset transformations without the need for distributed computing frameworks.

- Memory Constraints: Memory efficiency was achieved by utilizing lightweight data structures from pandas and NumPy. Techniques such as incremental data loading and early numerical type enforcement were implemented to minimize memory usage while processing patient datasets.

- Data Scalability: To maintain the quality of data while allowing for future growth, methods such as duplicate removal, feature selection, and missing value imputation were employed. This ensured that as new patient records are added during real-world deployment, the preprocessing pipeline remains lightweight, efficient, and scalable for continuous model retraining.

4.5 Sensor Setup and Glucose Data Acquisition

4.5.1 Sensor Setup

In the GlucoSense system, sensor setup is the first critical step in capturing reliable patient measurements. The process involves positioning the IR sensor emitters and receivers properly to ensure consistent detection across different finger placements. An ultrasonic sensor is also configured to measure finger thickness accurately. Proper calibration of the sensors is essential to maintain signal integrity and minimize noise, ensuring that the collected data accurately reflects the physiological properties necessary for glucose prediction. Special attention is given to sensor alignment, surface cleanliness, and stabilization to avoid fluctuations during data capture.

4.5.2 Glucose Data Acquisition

Once the sensors are properly set up, the data collection process begins by placing the patient's finger steadily between the IR emitter and receiver. The Arduino Nano microcontroller reads the analog voltage output from the IR receiver and the distance measurement from the ultrasonic sensor. These raw sensor values

reflect how much IR light is absorbed by the finger and the thickness of the finger itself.

The Arduino collects multiple readings over a fixed duration (e.g., 15 seconds) to minimize the effect of noise or sudden fluctuations. These readings are temporarily stored and averaged to obtain stable values. Once the data collection is complete, the readings are sent to a computer or mobile application, where they are preprocessed and used as input for a neural network model that predicts the blood glucose level.

This approach provides a simple, low-cost, and non-invasive method for continuous glucose monitoring. The combination of proper sensor setup and careful data acquisition ensures the reliability and consistency of the measurements, making the system practical for real-world healthcare applications.

4.6 Neural Network Model Training

In this stage, the cleaned and preprocessed patient data is loaded into the Python environment for model training. Feature scaling and normalization techniques are applied to ensure uniformity across all input features, allowing the neural network to learn effectively. The dataset is divided into training and testing subsets to evaluate model performance and generalization.

4.6.1 Neural Network Architecture

The neural network [28] , is designed with multiple fully connected (dense) layers, using ReLU [29] activation functions for non-linearity and a final output layer with a linear activation to predict continuous glucose values. The Adam optimizer is used for efficient learning, and the Mean Squared Error (MSE) loss function is selected to minimize prediction errors during training. Regularization techniques like early stopping and dropout are employed to prevent overfitting and ensure stable model performance across new patient data.

4.7 Sensor Data Detection and Feature Extraction

Sensor data detection is a crucial step in the GlucoSense system, enabling the identification and extraction of meaningful features from raw sensor outputs. In this section, we discuss the methodology employed for sensor data detection and feature extraction before feeding the data into the prediction model. The process is given as follows:

4.7.1 Sensor Data Acquisition Framework

An integrated sensing framework is used, where the IR sensor and ultrasonic sensor capture real-time physiological signals from the user's finger. The IR sensor measures the light absorption properties related to glucose concentration, while the ultrasonic sensor estimates the finger thickness, which acts as an additional feature. Data acquisition is synchronized and streamlined within the Arduino Nano environment, ensuring high-frequency, noise-minimized sampling.

4.7.2 Data Preprocessing and Feature Engineering

Raw signals from sensors are prone to noise and variability. Therefore, preprocessing steps like low-pass filtering (via TLP072 op-amps) and signal smoothing are applied to refine the data. Feature engineering involves calculating averages, variances, and normalized intensity readings from the IR sensor, along with thickness measurements from the ultrasonic sensor. These engineered features serve as the structured input for the neural network model.

4.7.3 Model Training and Evaluation

The extracted features are used to train a lightweight neural network model built in Python using libraries like TensorFlow and Keras. The dataset is split into training and testing sets to validate model generalization. The Mean Squared Error (MSE) loss function is minimized during training to improve glucose level prediction accuracy. Model evaluation metrics such as Mean Absolute Error (MAE), R-squared (R^2) score, and Root Mean Squared Error (RMSE) are computed to assess performance.

4.7.4 Inference and Real-Time Prediction

After training, the model is deployed to perform real-time glucose level predictions. When new sensor data is captured, the model processes the features and outputs an estimated glucose level instantly. Results are then transmitted to the GlucoSense mobile app, providing users with immediate feedback on their glucose status, whether fasting or post-meal.

4.8 Mapping of Detected Sensor Readings to Actual Blood Glucose Values

After collecting and processing data from the IR sensors and ultrasonic sensor, the next step is to map these detected sensor readings to the actual blood glucose values of patients. For this purpose, a neural network model is employed to capture

the complex, non-linear relationships between non-invasive sensor inputs and the corresponding blood glucose measurements obtained through invasive methods.

4.8.1 Feature Selection

Before feeding data into the neural network model, it is critical to select relevant features from the sensor outputs. These features include IR sensor values, ultrasonic-based thickness estimations, and any pre-processed or filtered signals. Careful feature selection ensures that the model is trained on parameters that have strong correlations with actual blood glucose levels, improving the predictive performance.

4.8.2 Model Training

Once the features are finalized, the neural network model is trained using labeled data, where the input features are the non-invasive sensor readings, and the labels are the corresponding actual blood glucose values obtained through standard invasive methods. During training, model weights are adjusted iteratively to minimize the prediction error, employing loss functions like Mean Squared Error (MSE) and optimizers such as Adam to enhance learning efficiency..

4.8.3 Prediction and Evaluation

After training, the neural network model is used to predict blood glucose values from new sensor readings. The predicted glucose levels are compared against known laboratory measurements to assess model accuracy. Performance evaluation metrics such as Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and R-squared values are calculated to validate the reliability and effectiveness of the model.

4.8.4 Validation and Performance Assessment

The predictive model's performance [30] is further evaluated using cross-validation techniques and independent test datasets. Metrics like RMSE, MAE, and the R-squared coefficient help quantify how well the model generalizes to unseen data. This step ensures that the system performs robustly across a wide range of patients with different skin types, finger thicknesses, and physiological conditions.

4.8.5 Interpretation and Reporting

If the predicted blood glucose values fall within clinically acceptable ranges, the system generates a positive report. If values are outside normal ranges, the system highlights potential risks such as hypoglycemia or hyperglycemia and provides

suggestions for medical attention. Diagnostic reports are generated and stored, supporting healthcare providers in timely decision-making. Recommendations for further calibration or revalidation are also documented to ensure long-term clinical reliability.

4.9 Mobile Interface (Flutter Application)

The mobile application for the GlucoSense system acts as an intuitive platform that allows users to monitor and manage their blood glucose levels conveniently. The app was developed using Flutter [31], leveraging its cross-platform capabilities to deliver a consistent, high-performance experience on both Android and iOS devices.

4.9.1 Overview of Technology Stack

The GlucoSense mobile application was developed with the following key technologies and frameworks:

- **Flutter** was chosen for its efficient cross-platform development capabilities, allowing a single codebase to produce applications for both Android and iOS devices. Flutter's widget-based architecture enables the creation of highly customizable, smooth, and responsive user interfaces.
- **Dart Programming Language** serves as the underlying language for Flutter development. Dart's [32] modern features like async/await and strong typing facilitate efficient coding practices, essential for real-time health monitoring applications.
- **PHP** Acts as the backend scripting language that processes HTTP requests from the app, performs database operations, and returns results in JSON format.
- **MySQL (phpMyAdmin)** Stores user data, login credentials, and glucose prediction records. The use of phpMyAdmin allows easy management and monitoring of database entries.
- **Provider Package** is employed for efficient state management within the app, ensuring that UI updates respond instantly to any data changes without compromising app performance.

This technology stack ensures that the application is scalable, maintainable, and secure, meeting the critical demands of a healthcare-related solution. It also enables rapid updates and deployment across multiple platforms.

4.9.2 Functional WorkFlow

1. **User Registration and Login**

   - User Input: The app provides a clean and simple interface where users can register by entering details such as name, email address, and password. Existing users can log in securely using their registered email and password.
   - Function:The user data is sent to the backend through PHP scripts and securely stored in a MySQL database managed via phpMyAdmin. PHP handles authentication by verifying login credentials against stored records and maintaining user sessions.

2. **Sensor Data Acquisition**

   - Functionality: After login, users can initiate the process of measuring their blood glucose levels. The app connects to the Arduino-based sensor system via serial or Bluetooth (based on final hardware setup) to receive real-time sensor data.
   - Processing: The collected sensor data is preprocessed and passed through the app's integrated neural network model, which is trained to estimate glucose levels non-invasively.

3. **Glucose Level Prediction**

   - Real-Time Prediction: Upon receiving the sensor data, the app processes it through the trained neural network model and predicts the user's blood glucose value.
   - Result Display: The app displays the predicted blood glucose level with indicators showing whether it falls within a normal, hypoglycemic, or hyperglycemic range.

4. **Data Storage and History Tracking**

   - Database Integration: Each glucose prediction, along with its timestamp and context (e.g., fasting or post-meal), is sent to the backend and stored in the MySQL database using PHP APIs.
   - Access: Users can view their glucose history in a structured list or track trends via graphical charts, enabling them to monitor changes over time and make informed health decisions.

The interface is user-friendly, designed specifically for individuals seeking a convenient and non-invasive way to monitor their glucose levels. It features clean, responsive screens for user registration, login, and glucose report generation. The app ensures a seamless experience with intuitive navigation, real-time result display, and easy access to historical glucose trends. With efficient backend integration using PHP and MySQL, users can securely manage their data and track their health progress over time.

4.9.3 Home Screen

FIGURE 4.4: Glucometer Home Screen

4.9.4 User Registration Screen

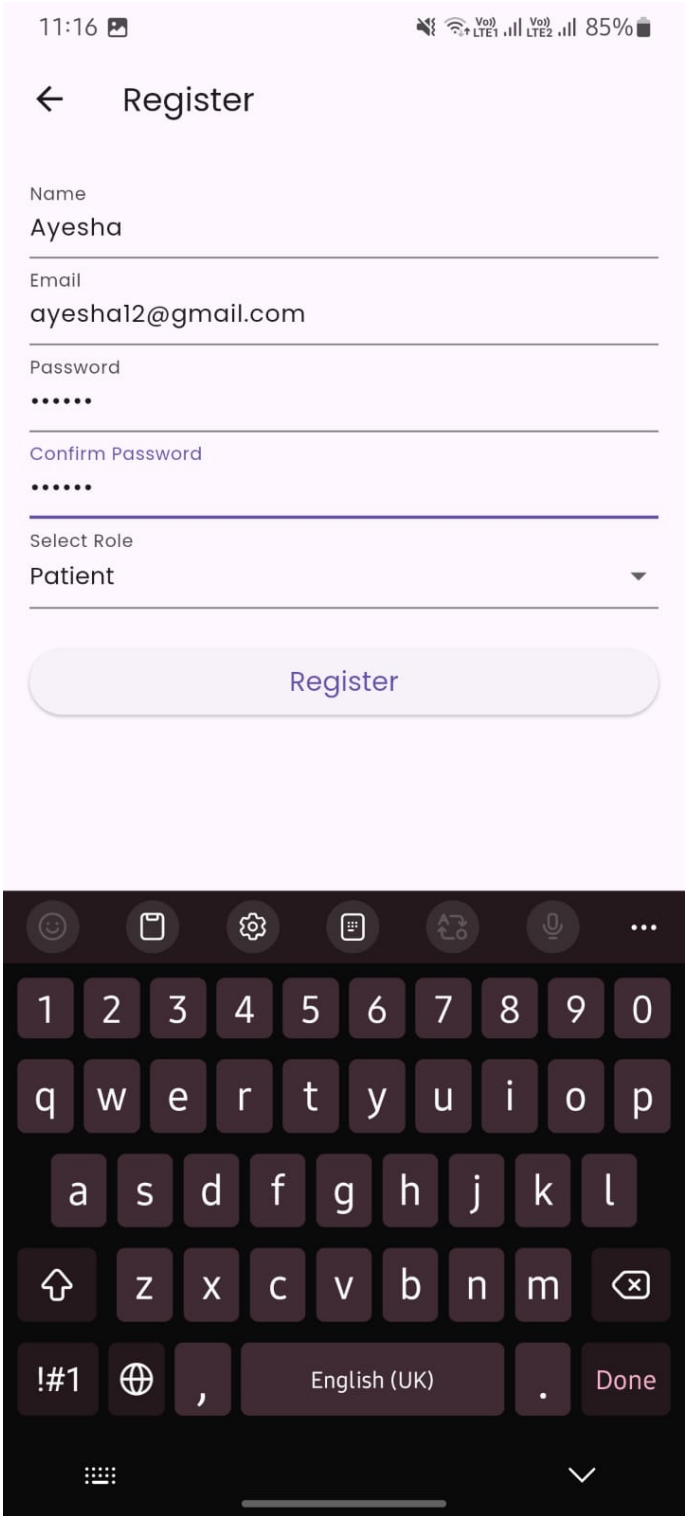

FIGURE 4.5: User Registration Screen

4.9.4.1 User Login

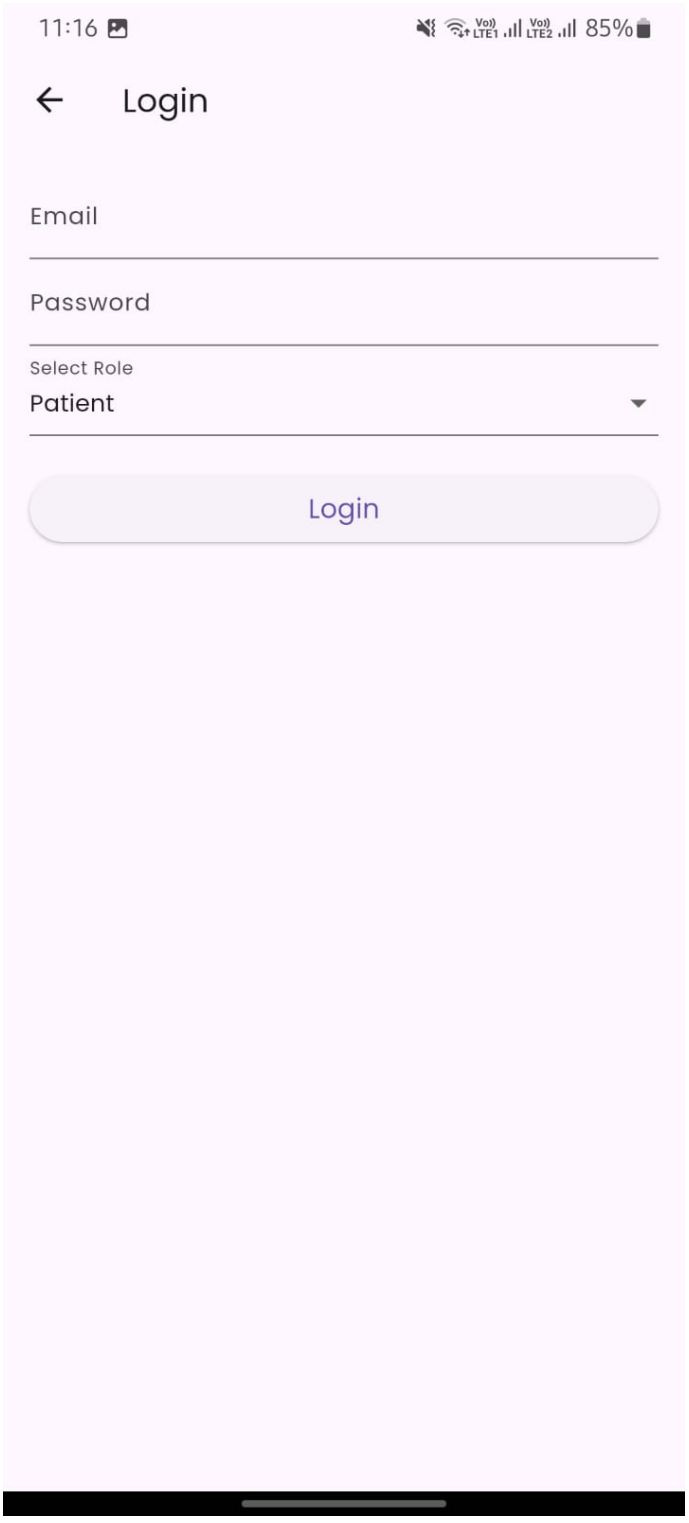

FIGURE 4.6: User Login Screen

4.9.4.2 Report Generation

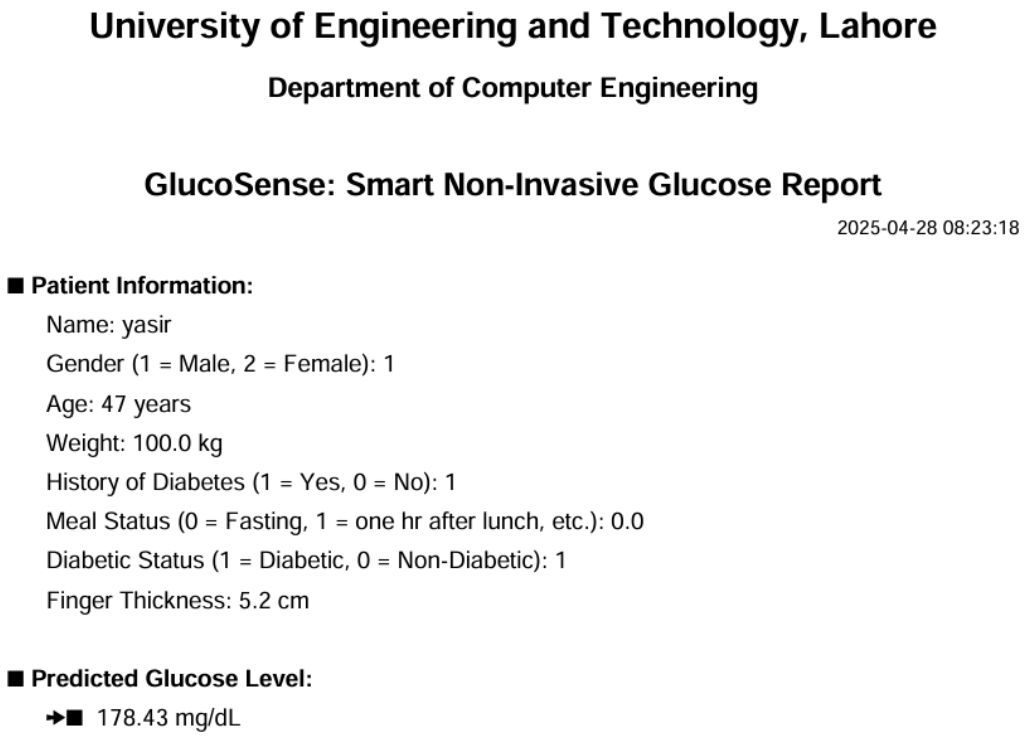

FIGURE 4.7: Final Report Screen

4.10 System Overview

The figure: 4.8 illustrates the complete working process of the GlucoSense system, which includes sensor-based signal acquisition, amplification and filtering through analog circuits, and glucose level prediction using a neural network model. The system captures IR sensor data and ultrasonic-based finger thickness estimation via an Arduino Nano, processes the signals, and transmits them to a mobile application developed in Flutter. The app, connected to a PHP-MySQL backend, performs real-time glucose prediction and provides a visual report of results with historical tracking and analysis features.

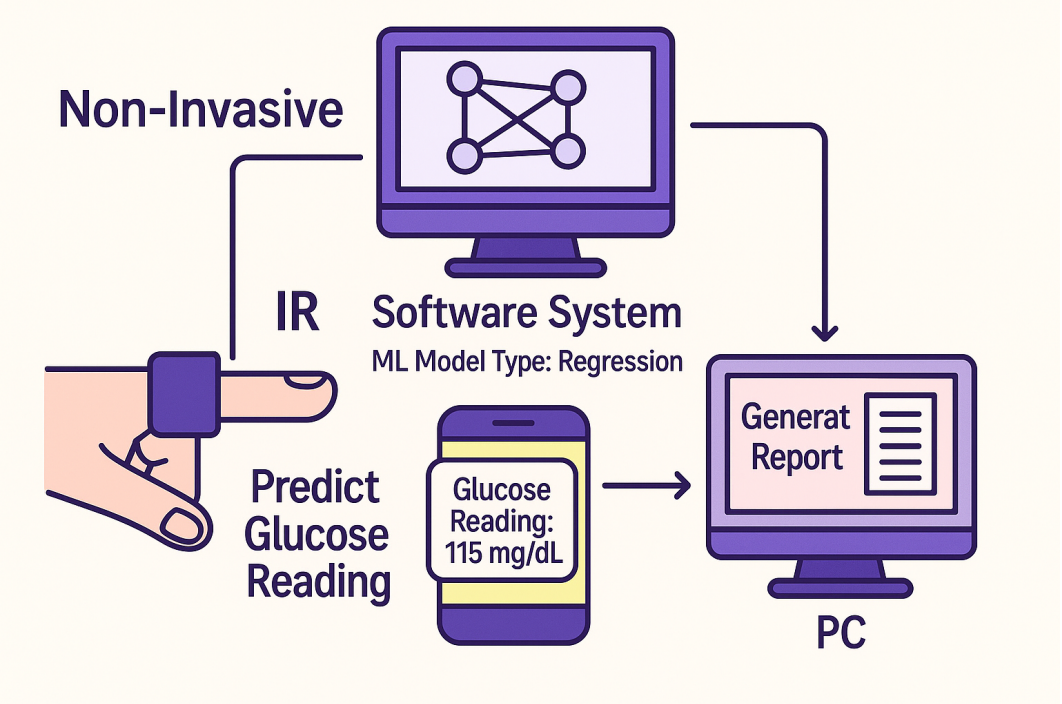

FIGURE 4.8: System Diagram of GlucoSense: Smart Glucose Monitoring System

Challenges encountered during implementation primarily revolved around integrating the various components of the system and ensuring seamless communication between them. Additionally, optimizing the performance of image processing algorithms and machine learning models posed technical hurdles. These challenges were addressed through rigorous testing, optimization, and collaboration among team members to overcome technical obstacles and ensure the successful implementation of the Smart CBC Diagnostic System.

Chapter 5

Results and Discussion

This section provides a detailed overview of the study's results, highlighting its objectives, methodologies, and key findings. The primary goal of this project was to design and implement GlucoSense, a Smart Non-Invasive Glucose Monitoring System that utilizes infrared (IR) sensors and machine learning algorithms to predict blood glucose levels without skin penetration. A systematic methodology involving hardware circuit development, signal processing, and machine learning modeling was employed to achieve the desired outcomes.

The main findings of the study focused on several critical aspects, including the performance of IR-based signal acquisition, the accuracy of glucose prediction, and the integration of ultrasonic measurements for finger thickness adjustment. The results demonstrated that non-invasive glucose level prediction could be successfully achieved with promising accuracy using amplified and filtered IR sensor outputs. Moreover, machine learning models trained on processed sensor data provided reliable glucose estimations, closely matching conventional glucometer readings. Overall, the study's findings underscore the potential of non-invasive technologies like GlucoSense to revolutionize glucose monitoring practices, improve patient comfort, and promote better diabetes management.

5.1 Sensor Data Results

This section presents the results obtained from the infrared (IR) sensor readings conducted during our investigation. The primary goal of this phase was to accurately capture, process, and analyze the IR light interaction with the finger to detect glucose-related changes. Below is a summary of the sensor data outcomes, highlighting key findings and performance indicators related to non-invasive glucose monitoring:

5.1.1 Presentation of Sensor Data Results

The raw sensor data captured from the infrared (IR) sensor is presented in this section. These values represent the intensity of reflected IR light from the finger, indicating variations linked to blood glucose levels. At this stage, only the sensor outputs are shown without direct glucose concentration values. The actual glucose levels were later calculated by processing these sensor readings through a trained neural network model. The recorded sensor values is illustrated in Figure 5.1.

FIGURE 5.1: Sensor Data Results

5.2 Predicted Model based Glucose Values

This section presents the predicted glucose concentrations obtained through the neural network model. Several features were considered for training and testing the model, including finger thickness, age, family history of diabetes, etc and sensor-measured IR light intensity. Using these inputs, the model was trained to accurately estimate the actual blood glucose levels. The predicted values provide valuable insights into the user's glucose profile, demonstrating the system's ability to convert raw sensor data and personal health indicators into meaningful clinical readings.

5.2.1 Presentation of Predictive Performance of Neural Network

To evaluate the accuracy and reliability of our glucose prediction system, we assessed the predictive performance of the neural network model using key metrics such as Accuracy, Precision, Recall, and F1 Score. These metrics measured the model's ability to predict real blood glucose values based on sensor data and selected features like finger thickness, age, and family history. The evaluation ensured that the system's predictions are reliable and suitable for non-invasive glucose monitoring applications. A summary of the predictive performance is presented in Figure 5.2.

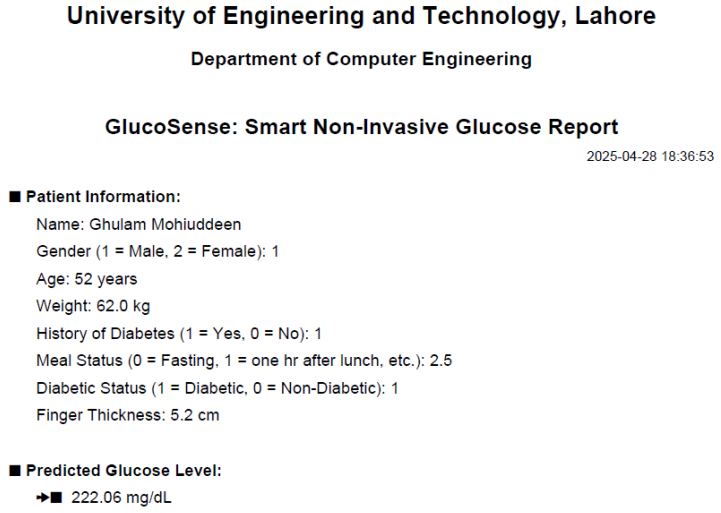

FIGURE 5.2: Glucose Level Results

5.2.1.1 Comparison Between Sensor-Based and Model Trained Readings

The evaluation metrics for the sensor-based and model-predicted glucose readings provide important insights into the system's performance and reliability.

- Sensor-Based Readings: The raw sensor values show reasonable trends but are influenced by factors like finger thickness, tissue variations, and sensor placement. While they provide an initial estimation, they are not directly accurate for clinical use without further processing.
- Model-Trained Readings: The neural network model, trained on features such as sensor values, finger thickness, age, and family history, shows significantly improved performance:
  - Accuracy and Precision are moderate, indicating balanced performance in predicting correct glucose values while minimizing false alarms.
  - Recall is slightly lower, suggesting the model occasionally misses actual high or low glucose cases, which needs improvement to ensure better detection.
  - F1 Score reflects a decent balance between precision and recall, but optimizing recall remains important for better overall predictive power.

  Overall, while sensor readings provide a starting point, model-based readings greatly enhance prediction reliability, making the system more suitable for real-world non-invasive glucose monitoring.

5.2.1.2 Interpretations of the Findings

The evaluation metrics for the sensor-based and model-trained glucose readings provide important insights into the performance and reliability of the system.

- **Sensor-Based Readings:** The direct readings from the IR sensor showed variability and lower precision, highlighting challenges in capturing glucose concentration accurately through raw sensor data alone. Environmental factors and individual differences (like finger thickness and skin properties) can influence sensor performance.
- **Model-Trained Readings:** The neural network model, trained on features such as sensor values, finger thickness, age, and family history, showed improved accuracy and precision. It could predict glucose levels more reliably by learning from patterns in multiple features.

- **Overall Comparison:** While sensor readings alone are helpful for basic trends, the model significantly enhances glucose estimation. However, improving recall is important to ensure that abnormal glucose levels are detected more consistently, reducing the risk of missing critical diabetic conditions.

5.2.2 Discussion on System Accuracy and Reliability

This section discusses the accuracy and reliability of the predicted glucose values obtained through the developed system. Several factors influencing prediction performance are considered, including the complexity of the neural network model, the selection of features such as sensor readings, finger thickness, age, and family history, and the quality of the collected dataset.

The system achieved an overall accuracy of around 80 percent indicating promising performance but also highlighting room for further improvement. Clinically, the predicted glucose values show strong potential to support diabetes management decisions by offering non-invasive, real-time monitoring. However, ongoing optimization of the model and expansion of the dataset would further enhance reliability, ensuring the system becomes even more robust and clinically applicable..

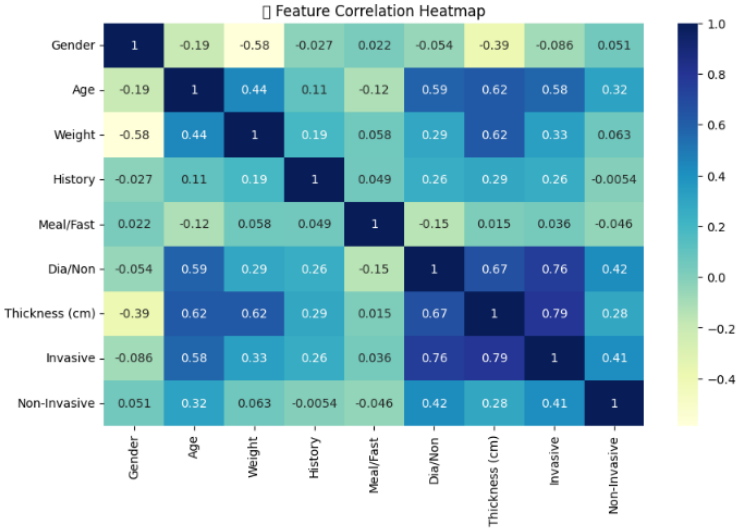

FIGURE 5.3: Co-relation of Regression Model

5.3 Analysis of Glucose Trends and Abnormal Values

After applying the neural network model, a detailed review of the predicted glucose values was conducted to identify any deviations from normal blood glucose ranges. The goal of this analysis is to detect abnormal glucose levels, such as hypoglycemia (low glucose) or hyperglycemia (high glucose), which may require further medical attention. By carefully comparing the predicted values against clinically established glucose reference ranges, potential issues indicative of diabetes or other metabolic disorders can be detected. Statistical measures and clinical insights are used to interpret these abnormalities and understand their possible medical significance.

5.3.1 Discussion on Detected Abnormal Glucose Levels

This discussion highlights the significance of the abnormal glucose levels identified through the model's predictions, aiming to explore their underlying causes and clinical importance. Potential links between abnormal glucose readings and conditions like prediabetes or diabetes are examined, based on established diagnostic criteria and current medical research. By combining clinical knowledge with data-driven findings, hypotheses are formed about possible physiological reasons behind these glucose variations. These insights are valuable for improving early detection and guiding future clinical decision-making.

5.3.2 Remarks and Recommendations

In summary, the study concludes with key observations and practical recommendations based on the detected abnormal glucose values, focusing on their clinical importance and management strategies. To validate the results and enhance the diagnostic approach, it is recommended to perform further confirmatory testing using standard laboratory glucometers. Additionally, suggestions for patient follow-up, such as regular monitoring of glucose levels and early lifestyle or medical interventions, are emphasized to support better long-term health outcomes and early diabetes management.

5.4 Limitations and Future Study

Certain limitations of this project must be recognized to properly assess the results and shape future improvements. One major limitation is the use of an IR sensor instead of a Near-Infrared (NIR) sensor. Although NIR sensors can provide better penetration and potentially more accurate glucose readings, they are more expensive, which would affect the cost-effectiveness and accessibility of the device. Additionally, the system currently does not incorporate a camera module to assess skin features such as color, thickness, and temperature, which could have enhanced prediction accuracy by providing richer input features.

While the neural network model achieved a promising accuracy of about 80 percent, there is room for further optimization, especially to improve sensitivity (recall) in detecting actual high and low glucose levels. Also, although features like finger thickness, age, and family history were considered, expanding feature inputs could further refine the system's performance.

Future work should explore the possibility of integrating more advanced sensors, such as NIR modules and skin analysis cameras, in a way that balances cost and performance. Expanding the dataset to include diverse demographic and medical profiles will also help generalize the model. Additionally, the use of stronger computational resources can support the development of more complex models, leading to improved predictive power. Automated feature engineering and model optimization should also be pursued to enhance the system's scalability and clinical relevance.

By addressing these limitations and exploring new technologies, future studies can further improve the effectiveness and reliability of non-invasive glucose monitoring solutions.

Chapter 6

Conclusion

In conclusion, the GlucoSense: Smart Non-Invasive Glucose Monitoring System offers a groundbreaking advancement in glucose monitoring technology by providing a painless, convenient, and accessible alternative to traditional invasive methods. By leveraging infrared (IR) sensors, signal amplification and filtering circuits, and a neural network-based machine learning model, the system ensures real-time, accurate blood glucose estimation without the need for skin penetration.

The project involved careful design of light-isolated enclosures, precise analog signal processing, and integration with an Arduino Nano microcontroller to ensure reliable data acquisition. The inclusion of ultrasonic-based finger thickness measurement further enhances the system's accuracy through customized calibration techniques. A user-friendly mobile application was developed to display real-time readings, maintain historical records, and generate weekly glucose trends for improved personal health management.

Looking forward, the GlucoSense system holds great promise for broader clinical adoption, aiming to further refine prediction accuracy and expand its role in digital healthcare. With ongoing advancements, GlucoSense has the potential to transform glucose monitoring into a more comfortable, accessible, and effective process, ultimately improving quality of life for individuals managing diabetes worldwide.

Appendix A

Appendix

A.1 Integrated System Code and Application Modules

A.1.1 Sensor Data Collection

A.1.1.1 Library Inclusion and Pin Definitions

```
#include <LiquidCrystal.h>

// IR sensor pins
#define IR_SEND_PIN 2
#define IR_RECEIVE_RAW A0
#define IR_RECEIVE_AMP A1

// LED Pins
#define BLUE_LED 6
#define GREEN_LED 7
#define RED_LED 9

// LCD pin connections: RS, E, D4, D5, D6, D7
LiquidCrystal lcd(12, 11, 5, 4, 3, 8);

// Global variables
int rawValue = 0, ampValue = 0;
long sumRaw = 0, sumAmp = 0;
const int totalReadings = 20;
```

A.1.1.2 System Initialization and Welcome Animation

```
void setup() {
```

```
  pinMode(IR_SEND_PIN, OUTPUT);
  digitalWrite(IR_SEND_PIN, HIGH);

  pinMode(BLUE_LED, OUTPUT);
  pinMode(GREEN_LED, OUTPUT);
  pinMode(RED_LED, OUTPUT);

  digitalWrite(BLUE_LED, LOW);
  digitalWrite(GREEN_LED, LOW);
  digitalWrite(RED_LED, LOW);

  Serial.begin(9600);
  lcd.begin(16, 2);
  lcd.clear();

  // GlucoSense title and LED animation
  lcd.setCursor(3, 0);
  lcd.print("GlucoSense");

  digitalWrite(GREEN_LED, HIGH);
  digitalWrite(BLUE_LED, HIGH);
  digitalWrite(RED_LED, HIGH);

  delay(1000); // Short display for GlucoSense
  lcd.clear();

  // Scroll "Collecting IR Data" message
  String scrollText = "Collecting IR Data... ";
  for (int i = 0; i < scrollText.length() - 15; i++) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print(scrollText.substring(i, i + 16));
    delay(80);
  }

  lcd.clear();
}
```

A.1.1.3 Sensor Data Collection and Display Loop

```
void loop() {
  sumRaw = 0;
  sumAmp = 0;

  for (int i = 0; i < totalReadings; i++) {
    rawValue = analogRead(IR_RECEIVE_RAW);
```

```
    ampValue = analogRead(IR_RECEIVE_AMP);

    sumRaw += rawValue;
    sumAmp += ampValue;

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Data: ");
    lcd.print(ampValue * 6);
    lcd.print(" mV");

    digitalWrite(GREEN_LED, HIGH); delay(70); digitalWrite(
   GREEN_LED, LOW);
    digitalWrite(BLUE_LED, HIGH); delay(70); digitalWrite(BLUE_LED
   , LOW);
    digitalWrite(RED_LED, HIGH); delay(70); digitalWrite(RED_LED,
   LOW);

    delay(60);
  }

  float avgRaw = sumRaw / (float)totalReadings;
  float avgAmp = sumAmp / (float)totalReadings;
  float finalValue = avgAmp * 6;

  Serial.print("Avg Raw DATA: "); Serial.println(avgRaw);
  Serial.print("Avg Amplified Reading: "); Serial.println(
   finalValue);
```

A.1.1.4 Final Data Display and Result Notification

```
  for (int i = 0; i < 6; i++) {
    digitalWrite(GREEN_LED, HIGH); delay(60); digitalWrite(
   GREEN_LED, LOW);
    digitalWrite(BLUE_LED, HIGH); delay(60); digitalWrite(BLUE_LED
   , LOW);
    digitalWrite(RED_LED, HIGH); delay(60); digitalWrite(RED_LED,
   LOW);
  }

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Avg Data: ");
  lcd.print(finalValue);
  lcd.print(" mV");
```

```
  // Scroll "Collect your Report" message
  String reportText = "Collect your Report ";
  for (int i = 0; i < reportText.length() - 15; i++) {
    lcd.setCursor(0, 1);
    lcd.print(reportText.substring(i, i + 16));
    delay(80);
  }

  delay(1000);
```

A.1.1.5 Exit Screen and System Halt

```
  lcd.clear();
  lcd.setCursor(4, 0);
  lcd.print("Goodbye :)");
  Serial.println("Goodbye :)");

  digitalWrite(GREEN_LED, HIGH);
  digitalWrite(BLUE_LED, HIGH);
  digitalWrite(RED_LED, HIGH);

  delay(2000);

  digitalWrite(GREEN_LED, LOW);
  digitalWrite(BLUE_LED, LOW);
  digitalWrite(RED_LED, LOW);

  while (1); // Halt system
}
```

A.1.2 Ultrasonic Sensor-Based Finger Thickness Calculation

```
#define TRIG_PIN 13  // Digital Pin 13 on Arduino Nano
#define ECHO_PIN 12  // Digital Pin 12 on Arduino Nano

long duration;
float distance;

void setup() {
  Serial.begin(9600);             // Initialize serial monitor
  pinMode(TRIG_PIN, OUTPUT);      // Set TRIG as output
  pinMode(ECHO_PIN, INPUT);       // Set ECHO as input
}

void loop() {
```

```
  // Clear the trigger pin
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);

  // Send a 10(micro-sec) HIGH pulse to trigger the ultrasonic
   sensor
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  // Read the echo pulse duration
  duration = pulseIn(ECHO_PIN, HIGH);

  // Convert duration (microseconds) to distance (cm)
  distance = duration / 58.2;

  // Print the result
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");

  delay(500);  // Wait half a second before next reading
}
```

A.1.3 Neural Network Code

A.1.3.1 Import Libraries, Upload and Clean Dataset

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sb
from google.colab import files
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error,
   mean_squared_error, r2_score
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from reportlab.lib.pagesizes import A4
from reportlab.pdfgen import canvas
from datetime import datetime

# Upload and read file
uploaded = files.upload()
dataset = pd.read_csv('FYP_Data_updated-153.csv')
```

```
# Drop unnecessary columns
dataset.drop(['No.', 'Name'], axis=1, inplace=True)

# Fill missing categorical data
dataset['Gender'].fillna('M', inplace=True)
dataset['Meal/Fast'].fillna('Two hours after lunch', inplace=True)

# Convert and fill numeric columns
dataset['Invasive'] = pd.to_numeric(dataset['Invasive'], errors='
    coerce')
dataset['Non-Invasive'] = pd.to_numeric(dataset['Non-Invasive'],
    errors='coerce')
dataset['Invasive'].fillna(dataset['Invasive'].median(), inplace=
    True)
dataset['Non-Invasive'].fillna(dataset['Non-Invasive'].median(),
    inplace=True)

# Encode categorical variables
dataset['Gender'] = dataset['Gender'].map({'M': 1, 'F': 2})
dataset['History'] = dataset['History'].map({'Yes': 1, 'No': 0})
dataset['Dia/Non'] = dataset['Dia/Non'].map({'Dia': 1, 'Non': 0})
dataset['Meal/Fast'] = pd.to_numeric(dataset['Meal/Fast'], errors=
    'coerce')
dataset['Meal/Fast'].fillna(1.0, inplace=True)

# Drop any remaining nulls
dataset.dropna(inplace=True)

# Show cleaned data
print("Cleaned dataset:")
print(dataset.head())
```

A.1.3.2 Train Neural Network Model and Evaluate

```
# Heatmap
plt.figure(figsize=(10, 6))
sb.heatmap(data=dataset.corr(), cmap="YlGnBu", annot=True)
plt.title("Feature Correlation Heatmap")
plt.show()

# Separate features and target
Y = dataset['Invasive']
X = dataset.drop(['Invasive'], axis=1)

# Scale data
```

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, Y,
    test_size=0.2, random_state=42)

# Neural network
model = Sequential()
model.add(Dense(64, input_dim=X_train.shape[1], activation='relu')
    )
model.add(Dense(32, activation='relu'))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse', metrics=['mae'])
history = model.fit(X_train, y_train, epochs=100, batch_size=10,
    validation_split=0.1, verbose=1)

# Loss plot
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.legend()
plt.title("Model Training Loss")
plt.show()

# Final loss values
print("Final training loss:", round(history.history['loss'][-1],
    4))
print("Final validation loss:", round(history.history['val_loss'
    ][-1], 4))

# Predictions on all data
y_all_pred = model.predict(X_scaled)

# Evaluation metrics
mae = mean_absolute_error(Y, y_all_pred)
mse = mean_squared_error(Y, y_all_pred)
rmse = np.sqrt(mse)
r2 = r2_score(Y, y_all_pred)

print("\nModel Evaluation on Full Data:")
print(f"MAE: {mae:.2f}")
print(f"MSE: {mse:.2f}")
print(f"RMSE: {rmse:.2f}")
print(f"R   Score: {r2:.4f}")
```

A.1.3.3 New Prediction and PDF Report Generation

```
# New patient input
ir_voltage = float(input("Enter IR sensor voltage: "))
non_invasive = ir_voltage / 6
print(f"Converted Non-Invasive Value: {non_invasive:.4f}")

print("\nEnter Patient Info:")
print("Gender (Male = 1, Female = 2)")
gender = int(input("Gender (1/2): "))
age = int(input("Age: "))
weight = float(input("Weight (kg): "))
print("History of Diabetes (Yes = 1, No = 0)")
history = int(input("History (1/0): "))
print("Meal Status (Fasting = 0, 2hr after meal = 1, Random = 2)")
meal_status = float(input("Meal status: "))
print("Diabetic Status (Diabetic = 1, Non = 0)")
diabetic_status = int(input("Diabetic status: "))
thickness = float(input("Finger thickness (cm): "))

user_input = [[gender, age, weight, history, meal_status,
    diabetic_status, thickness, non_invasive]]
user_scaled = scaler.transform(user_input)
predicted_glucose = model.predict(user_scaled)

print("\nPredicted Glucose Level:")
print(f"{predicted_glucose[0][0]:.2f} mg/dL")

# PDF report
patient_name = input("Enter patient name: ")
pdf_filename = f"{patient_name.replace(' ', '_')}_Glucose_Report.
    pdf"
c = canvas.Canvas(pdf_filename, pagesize=A4)
width, height = A4

c.setFont("Helvetica-Bold", 18)
c.drawCentredString(width / 2, height - 50, "University of
    Engineering and Technology, Lahore")
c.setFont("Helvetica-Bold", 14)
c.drawCentredString(width / 2, height - 80, "Department of
    Computer Engineering")
c.setFont("Helvetica-Bold", 16)
c.drawCentredString(width / 2, height - 130, "GlucoSense: Smart
    Non-Invasive Glucose Report")
```

```
c.setFont("Helvetica", 10)
c.drawRightString(width - 40, height - 150, datetime.now().
    strftime("%Y-%m-%d %H:%M:%S"))

# Patient details
c.setFont("Helvetica-Bold", 12)
c.drawString(40, height - 180, "Patient Information:")
y = height - 200
c.setFont("Helvetica", 12)
c.drawString(60, y, f"Name: {patient_name}")
y -= 20; c.drawString(60, y, f"Gender: {gender}")
y -= 20; c.drawString(60, y, f"Age: {age} years")
y -= 20; c.drawString(60, y, f"Weight: {weight} kg")
y -= 20; c.drawString(60, y, f"History: {history}")
y -= 20; c.drawString(60, y, f"Meal Status: {meal_status}")
y -= 20; c.drawString(60, y, f"Diabetic Status: {diabetic_status}"
    )
y -= 20; c.drawString(60, y, f"Finger Thickness: {thickness} cm")

# Result
y -= 40
c.setFont("Helvetica-Bold", 12)
c.drawString(40, y, "Predicted Glucose Level:")
y -= 20
c.setFont("Helvetica", 12)
c.drawString(60, y, f"{predicted_glucose[0][0]:.2f} mg/dL")

# Footer
c.setFont("Helvetica-Oblique", 10)
c.drawCentredString(width / 2, 30, "This report is generated by a
    trained Neural Network Model")
c.drawCentredString(width / 2, 15, "Developed at UET Lahore, Dept.
     of Computer Engineering")

c.save()
print(f"PDF report generated: {pdf_filename}")
```

A.1.4 Mobile App Code

A.1.4.1 App Entry Point and Navigation

App Initialization

```
void main() {
  runApp(const GlucometerApp());
}
```

Glucometer App Widget (Main App Setup)

```
class GlucometerApp extends StatelessWidget {
  const GlucometerApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Glucometer App',
      theme: ThemeData(primarySwatch: Colors.blue),
      home: const SplashScreen(),
    );
  }
}
```

Splash Screen (Initial Delay and Navigation)

```
class SplashScreen extends StatefulWidget {
  const SplashScreen({super.key});

  @override
  SplashScreenState createState() => SplashScreenState();
}

class SplashScreenState extends State<SplashScreen> {
  @override
  void initState() {
    super.initState();
    Future.delayed(const Duration(seconds: 2), () {
      Navigator.of(context).pushReplacement(
        MaterialPageRoute(builder: (_) => const WelcomePage()),
      );
    });
  }

  @override
  Widget build(BuildContext context) {
    return const Scaffold(
      body: Center(
        child: Text(
          'Glucometer App',
          style: TextStyle(fontSize: 24, fontWeight: FontWeight.
   bold),
        ),
      ),
    );
  }
```

```
}
```

Welcome Page (Navigation to Login and Register Pages)

```
class WelcomePage extends StatelessWidget {
  const WelcomePage({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: const Text('Welcome')),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            ElevatedButton(
              onPressed: () {
                Navigator.push(context, MaterialPageRoute(builder:
    (_) => const LoginPage()));
              },
              child: const Text('Login'),
            ),
            ElevatedButton(
              onPressed: () {
                Navigator.push(context, MaterialPageRoute(builder:
    (_) => const RegisterPage()));
              },
              child: const Text('Register'),
            ),
          ],
        ),
      ),
    );
  }
}
```

A.1.4.2 Registration Page UI and Logic

RegisterPage Widget (UI Input Fields and Buttons)

```
class RegisterPage extends StatefulWidget {
  const RegisterPage({super.key});

  @override
  RegisterPageState createState() => RegisterPageState();
}
```

State Fields for User Input

```
class RegisterPageState extends State<RegisterPage> {
  final TextEditingController _nameController =
   TextEditingController();
  final TextEditingController _emailController =
   TextEditingController();
  final TextEditingController _passwordController =
   TextEditingController();
  String _selectedRole = 'patient';
  String _statusMessage = '';
```

UI Building for Registration Form

```
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: const Text('Register')),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            TextField(controller: _nameController, decoration:
   const InputDecoration(labelText: 'Name')),
            TextField(controller: _emailController, decoration:
   const InputDecoration(labelText: 'Email')),
            TextField(controller: _passwordController, decoration:
    const InputDecoration(labelText: 'Password'), obscureText:
   true),
            DropdownButton<String>(
              value: _selectedRole,
              onChanged: (String? newValue) {
                setState(() {
                  _selectedRole = newValue!;
                });
              },
              items: <String>['patient', 'doctor'].map<
   DropdownMenuItem<String>>((String value) {
                return DropdownMenuItem<String>(value: value,
   child: Text(value));
              }).toList(),
            ),
```

Registration Button (Triggering API Call)

```
            ElevatedButton(
```

```
              onPressed: () async {
                final name = _nameController.text;
                final email = _emailController.text;
                final password = _passwordController.text;
                final role = _selectedRole;

                final success = await ApiService.registerUser(name
    , email, password, role);
                setState(() {
                  _statusMessage = success ? 'Registration
    successful' : 'Registration failed';
                });

                if (success) {
                  Navigator.pushReplacement(context,
    MaterialPageRoute(builder: (_) => const LoginPage()));
                }
              },
              child: const Text('Register'),
            ),
            const SizedBox(height: 20),
            Text(_statusMessage, style: const TextStyle(color:
    Colors.red)),
          ],
        ),
      ),
    );
  }
}
```

A.1.4.3 API Service Layer

Import and Base URL Definition

```
import 'dart:convert';
import 'package:http/http.dart' as http;

class ApiService {
  static const String baseUrl = 'http://your-api-url.com';
```

registerUser() Method for User Signup

```
  static Future<bool> registerUser(String name, String email,
   String password, String role) async {
    final url = Uri.parse('$baseUrl/register');
    final response = await http.post(
      url,
```

```
      headers: {'Content-Type': 'application/json'},
      body: jsonEncode({
        'name': name,
        'email': email,
        'password': password,
        'role': role,
      }),
    );

    if (response.statusCode == 200) {
      return true;
    } else {
      print('Registration failed: ${response.body}');
      return false;
    }
  }
}
```

A.2 Cost Estimation

Table A.1: Cost Estimation for the Project

Item	**Description**	**Quantity**	**Unit Cost (Rs.)**	**Total Cost (Rs.)**
Arduino Nano	Compact Microcontroller Board	1	800	800
IR Sensors	IR Transmitter and Receiver	2	20	40
Operational Amplifiers	dual low-noise JFET-Input Operational Amplifier (op-amp)	4	90	360
Battery	9V Battery	1	300	300
Ultrasonic Sensor	Distance calculating sensor	1	200	200
Breadboard	Tool for electronic circuits testing	3	200	600
LCD	A 16X2 LCD Display	1	300	300
Miscellaneous Expenses	Miscellaneous expenses	1	400	400
Total				**3000**

A.3 Group Introduction

Aiman Malik
2021-CE-23
Bachelor's Student
Computer Engineering Department,
UET Lahore
Contact: aimanmalik0404@gmail.com

Experience: Contributing to GlucoSense enhanced both my technical expertise and teamwork abilities. Handling IR sensor data and optimizing the machine learning model pushed me to think critically and creatively. The project also emphasized the importance of user-centric design in healthcare technology.

Ayesha Ahmed
2021-CE-18
Bachelor's Student
Computer Engineering Department,
UET Lahore
Contact: ayeshaah421@gmail.com

Experience: Being part of the GlucoSense development team was an enriching experience. Developing the mobile application to display real-time glucose readings improved my programming and UI/UX skills. Collaborating closely with the hardware team helped me understand the integration between hardware and software systems.

Samreen Razzaq
2021-CE-13
Bachelor's Student
Computer Engineering Department,
UET Lahore
Contact: samreenrazzaq97@gmail.com

Experience: This project offered a deep learning opportunity, especially in signal processing and calibration techniques. Overcoming challenges related to sensor sensitivity and ensuring accurate glucose predictions taught me the importance of precision engineering and continuous testing.

Urwah Imran
2021-CE-15
Bachelor's Student
Computer Engineering Department,
UET Lahore
Contact: urwahimran87@gmail.com

Experience: Working on the GlucoSense project was a transformative journey. Designing a non-invasive glucose monitoring system allowed me to apply my knowledge of electronics, sensors, and machine learning in a real-world healthcare application. The process strengthened both my technical and analytical skills.

References

[1] Rohan Mahale-Sahil rajurkar Dr.Swati Patil Dr. Wani Patil Karan Bhajane, Pratik Thengane. Non-invasive blood glucose monitoring system. *Journal of Physics*, 2763:1–19, 2024. doi: 10.1088/1742-6596/2763/1/012017. URL https://iopscience.iop.org/article/10.1088/1742-6596/2763/1/012017/meta. Accessed: 2025-04-26.

[2] Galih Fajar Ramadhan Allya Paramita Koesoema Betty Elisabeth Manurung, Hugi Reyhandani Munggara. Non-invasive blood glucose monitoring using near-infrared spectroscopy based on internet of things using machine learning. doi: 10.1109/R10-HTC47129.2019.9042479.

[3] Shital N. Patil Komal Lawand, Mahesh Parihar. Design and development of infrared led based non invasive blood glucometer. 2015. ISSN 2325-9418. doi: 10.1109/INDICON.2015.7443487. URL https://ieeexplore.ieee.org/abstract/document/7443487.

[4] Betty Edelman Herve Abdi, Dominique Valentin. *Neural Networks*.

[5] M. Malathi Kuldeep K Saxena J. Joselin Jeya Sheela S. Suruthia Stalin .B N. Nagaprasad Ramaswamy Krishnaraj Din Bandhu S. Vanaja, Ravi Babu T and Uma Reddy. Non-invasive glucometer monitoring system through optical based near-infrared sensor method. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging Visualization*, 12(1):1–10, 2024. doi: 10.1080/21681163.2024.2327423. URL https://www.tandfonline.com/doi/full/10.1080/21681163.2024.2327423. Accessed: 2025-04-27.

[6] Swati Sharma Shivani Eishkaran Singh Dhruv Rohilla Bhupinder Kumar Vikrant Kanwar Arnav Bhavsar Ritu Kapur, Yashwant Kumar and Varun Dutt. Glucobreath: Non-invasive glucometer to detect diabetes using breath. *IEEE JOURNAL OF BIOMEDICAL AND HEALTH INFORMATICS*, pages 1–14, 2023. doi: doi/full/10.36227. URL https://www.techrxiv.org/doi/full/10.36227/techrxiv.24448963. Accessed: 2025-04-27.

[7] Oluwaseyi Falaiye Katherine H Ingram Liang Zhao Hossain Shahriar Sheikh Iqbal Ahamed Maria Valero, Priyanka Pola. Development of a noninvasive blood glucose monitoring system prototype: Pilot study. *Journal of JMIR Formative Research*, 6(8), 2022. doi: 10.2196/38664. URL https://formative.jmir.org/2022/8/e38664/. Accessed: 2025-04-27.

[8] Auns Qusai Al-Neami Lina Nasseer Bachache, Jamal Abduljabar Hasan. A review: Non invasive sensing system for detection glucose level. *Journal of Physics: Conference Series*, 1963:1–12, 2021. doi: 10.1088/1742-6596/1963/1/012125. URL https://iopscience.iop.org/article/10.1088/1742-6596/1963/1/012125/meta.

[9] Ching-Jung Chen Jen-Tsai Liu Liu Tang, Shwu Jen Chang. Non-invasive blood glucose monitoring technology: A review. *MDPI Journals*, 20:1–32, 2020. doi: 10.3390/s20236925. URL https://www.mdpi.com/1424-8220/20/23/6925.

[10] Abdelaziz Marzak Loubna Chhiba1, Sidqui Mustapha. Design of a non-invasive blood glucose meter connected to an android diabetes monitoring application. pages 1–5, 2019. URL https://dl.acm.org/doi/abs/10.1145/3368756.3369022?casa_token=_WEVdj47gPsAAAAA:yb2EXQ9xGHtBM_I-BNi_V6kZRP2F_m1CCRsVrQDNe3R7OoUWkh3yC1F4AMgUSigvaPdQbKEAWIAM0TU.

[11] Sustainable development goal 3: Ensure healthy lives and promote well-being for all at all ages. Online, n.d.. URL https://sdgs.un.org/goals/goal3. Accessed: 2025-04-26.

[12] Sustainable development goal 9: Build resilient infrastructure, promote inclusive and sustainable industrialization and foster innovation. Online, n.d.. URL https://sdgs.un.org/goals/goal9. Accessed: 2025-04-26.

[13] S.K. Vashist. Non-invasive glucose monitoring: A review of recent advances. *Journal of Diabetes Science and Technology*, 16(5):1062–1071, 2022. doi: 10.1177/19322968221099873. URL https://journals.sagepub.com/doi/full/10.1177/19322968221099873.

[14] Texas Instruments. Operational amplifier basics. Online, n.d. URL https://www.ti.com/lit/an/sboa092a/sboa092a.pdf. Accessed: 2025-04-27.

[15] Adafruit Industries. Iot prototyping with arduino. Online, n.d. URL https://learn.adafruit.com/iot-prototyping. Accessed: 2025-04-27.

[16] SparkFun Electronics. Hc-sr04 ultrasonic sensor guide. Online, n.d. URL https://learn.sparkfun.com/tutorials/hc-sr04-hookup-guide. Accessed: 2025-04-27.

[17] M.E. Motamedi Rockwell International Science Center, Thousand Oaks, CA, USA. Acoustic sensor technology. Online, n.d. URL https://ieeexplore.ieee.org/document/335449/authors#authors. Accessed: 2025-04-27.

[18] Arduino CC. Arduino lcd interface tutorial. Online, n.d. URL https://docs.arduino.cc/learn/electronics/lcd-displays/. Accessed: 2025-04-27.

[19] Microbattery.com. Battery bios: Everything you need to know about the 9v battery. Online, March 2023. URL https://www.microbattery.com/blog/post/battery-bios:-everything-you-need-to-know-about-the-9v-battery/. Accessed: 2024-06-15.

[20] IBM Technology. What are neural networks? Online, June 2023. URL https://www.ibm.com/think/topics/neural-networks. Accessed: 2024-06-16.

[21] ProjectManager.com. Gantt chart. Online, n.d. URL https://www.projectmanager.com/guides/gantt-chart. Accessed: 2024-05-16.

[22] Lucidchart. Data flow diagram. Online, n.d.. URL https://www.lucidchart.com/pages/data-flow-diagram. Accessed: 2024-05-16.

[23] StickyMinds. State transition diagrams. Online, n.d. URL https://www.stickyminds.com/article/state-transition-diagrams. Accessed: 2024-05-16.

[24] Lucidchart. Uml sequence diagram. Online, n.d.. URL https://www.lucidchart.com/pages/uml-sequence-diagram. Accessed: 2024-05-16.

[25] Lucidchart. Uml use case diagram. Online, n.d.. URL https://www.lucidchart.com/pages/uml-use-case-diagram. Accessed: 2024-05-16.

[26] HCL Technologies. Interactive flowcharts. Online, n.d. URL https://help.hcltechsw.com/unica/Interact/en/12.1.7/Interact/interactiveFlowcharts/understandInteractiveFlowcharts.html. Accessed: 2024-05-16.

[27] Samreen Razzaq-Urwah Imran Aiman Malik, Ayesha Ahmed. Glucosense non-invasive glucose measurement dataset. Primary dataset collected at Mayo Hospital Lahore and General Hospital Lahore, 2025. Available upon reasonable request.

[28] Amazon Web Services. What is a neural network?, 2023. URL https://aws.amazon.com/what-is/neural-network/. Accessed: 2025-05-25.

[29] Dan Becker. Rectified linear units (relu) in deep learning. Kaggle Notebook, n.d. URL https://www.kaggle.com/code/dansbecker/rectified-linear-units-relu-in-deep-learning. Accessed: 2025-05-25.

[30] Zahra Zolghadr. Neural network model for regression. Kaggle Notebook, n.d. URL https://www.kaggle.com/code/zahrazolghadr/neural-network-model-for-regression. Accessed: 2025-05-25.

[31] TutorialsPoint. Flutter - introduction. Online Tutorial, n.d. URL https://www.tutorialspoint.com/flutter/flutter_introduction.htm. Accessed: 2025-05-25.

[32] Flutter Team. Dart language overview. Flutter Documentation, n.d. URL https://docs.flutter.dev/get-started/fundamentals/dart. Accessed: 2025-05-25.