DIABETRACK: SMART DIABETES MONITORING & LIFESTYLE ASSISTANT

A SOCIALLY RELEVANT MINI PROJECT REPORT

Submitted by

AMIRTHA R [REGISTER NO: 211423104034]

DEEPIKA P [REGISTER NO: 211423104112]

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE

CHENNAI – 600123

(An Autonomous Institution Affiliated to Anna University, Chennai)

OCTOBER 2025

PANIMALAR ENGINEERING COLLEGE
CHENNAI – 600123

(An Autonomous Institution Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this mini project report **"DIABETRACK: SMART DIABETES MONITORING & LIFESTYLE ASSISTANT** "is the Bonafide work of AMIRTHA R (211423104034), DEEPIKA P (211423104112) who carried out the mini project work under my supervision.

SIGNATURE
Dr L. JABASHEELA, M.E., Ph.D.,
HEAD OF THE DEPARTMENT

DEPARTMENT OF CSE
PANIMALAR ENGINEERING COLLEGE
NASARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

SIGNATURE
ALIMA BEEVI A M.E.,
ASSISTANT PROFESSOR

DEPARTMENT OF CSE
PANIMALAR ENGINEERING COLLEGE
NASARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

Submitted for the 23CS1512-Socially relevant mini Project Viva-Voce examination held on.....

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION BY THE STUDENT

We AMIRTHA R (211423104034), DEEPIKA P (211423104112) hereby declare that this

project report titled **DIABETRACK – SMART DIABETES MONITORING & LIFESTYLE ASSISTANT**, under the guidance of **ALIMA BEEVI A M.E.,** is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

1. AMIRTHA R

2. DEEPIKA P

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr. P. CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere and hearty thank to our directors **Tmt.C. VIJAYARAJESWARI**, **Dr.C. SAKTHIKUMAR, M.E., Ph.D.** and **Dr. SARANYASREE SAKTHI KUMAR B.E., M.B.A., Ph.D..,** for providing us with the necessary facilities to undertake this project. We also express our gratitude to our Principal **Dr. K. MANI, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. L. JABASHEELA, M.E., Ph.D.,** for the support extended throughout the project.

We would like to thank my Project Guide **ALIMA BEEVI A M.E**., and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

AMIRTHA R

DEEPIKA P

ABSTRACT

Diabetes is a global health challenge requiring effective monitoring and management to reduce complications and improve patient outcomes. Traditional methods, such as fingerprick testing and invasive continuous glucose monitoring (CGM), are often costly, inconvenient, and inaccessible for long-term use. Recent advancements in wearable technologies, artificial intelligence (AI), and machine learning(ML) have introduced innovative solutions for non-invasive, real-time glucose monitoring and personalized lifestyle management. This paper presents DiabeTrack, a comprehensive diabetes management system integrating wearable devices, Firebase cloud services, machine learning-based risk prediction, and smart reminders. The system allows users to record, track, and visualize glucose trends, receive personalized lifestyle recommendations, and synchronize health data securely across devices. The predictive model enables early detection of hyperglycaemia and hypoglycaemia risks, supporting preventive interventions. Evaluation results demonstrate that DiabeTrack provides high accuracy in glucose prediction, seamless integration of wearable data, and improved user adherence to healthy routines through smart reminders. The system represents a significant step toward AI-driven, patient-centric diabetes management ecosystems, with potential scalability for broader healthcare applications.

TABLE OF CONTENTS

Contents

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW

In recent years, diabetes mellitus has become one of the most common and serious chronic diseases worldwide, affecting millions of individuals. Effective diabetes management requires continuous glucose monitoring, regular medication intake, and healthy lifestyle adjustments. However, due to busy lifestyles and irregular habits, many individuals fail to monitor their glucose levels consistently, often resulting in complications such as hypoglycaemia, hyperglycaemia, and long-term cardiovascular risks.

To overcome these challenges, DiabeTrack is developed as a Smart Diabetes Monitoring and Lifestyle Assistant that provides real-time tracking, smart assistance, and intelligent analysis. The system enables users to log and visualize their glucose readings in real time through Bluetooth-enabled glucometer integration using the Web Bluetooth API, ensuring accurate and automatic data capture.

The application is built using React for the front end and Firebase for secure data storage and authentication. It integrates real-time glucose monitoring, data visualization, and a Smart Tablet Reminder to help users stay on track with their medications.

Additionally, DiabeTrack leverages XGBoost, CATBoost, LightGBM, a powerful machine learning algorithm, to perform predictive analysis on user data. By analysing realtime glucose readings and lifestyle inputs, the system can detect risk patterns and generate early warnings or insights to support better self-management.

Through its combination of real-time monitoring, AI-driven analytics, and smart reminders, DiabeTrack serves as a comprehensive and intelligent assistant for individuals managing type-2 diabetes, promoting timely actions and improved health outcomes

1.2 PROBLEM DEFINITION

Diabetes is a lifelong condition that requires continuous care, real-time monitoring, and strict lifestyle management. Many individuals, however, find it difficult to maintain consistency in their daily routines. They may forget to check their glucose levels regularly, skip medications, or fail to recognize patterns in their glucose readings. Over time, these irregular practices can lead to serious health issues such as hypoglycaemia, hyperglycaemia, fatigue, and other long-term complications.

Most of the existing diabetes management applications mainly focus on manual glucose entry or basic data tracking, without offering smart, data-driven support. These traditional systems do not make full use of real-time data or machine learning-based analysis, which are essential for identifying patterns and predicting health risks. Moreover, the process of entering readings manually can be time-consuming and prone to human error, making it less practical for regular use.

Another major limitation of current systems is the lack of personalization. Many applications send generic reminders that do not adapt to the user's glucose trends, medication schedule, or lifestyle habits. This makes the system less engaging and reduces user motivation to use the app regularly. Without intelligent insights or predictions, users are unable to easily identify trends such as frequent spikes or dips in glucose levels or understand how their meals, activities, or habits affect their readings.

In addition, there is a strong need for real-time integration with devices such as Bluetooth enabled glucometers. Manual data entry not only takes extra effort but also increases the chances of missed entries and inaccurate tracking. Real-time data synchronization ensures higher accuracy, reliability, and convenience for both patients and caregivers.

To overcome these limitations, there is a need for a smart, interactive, and predictive platform that combines real-time glucose monitoring, automated data logging, and machine learning-based predictions. By leveraging XGBoost, CATBoost, LightGBM, a powerful and efficient machine learning algorithm, the system can analyze the collected glucose data, identify unusual patterns, and predict possible risk levels. Combined with data visualization and a Smart Tablet Reminder, users can better understand their glucose trends, receive timely alerts, and improve their adherence to treatment.

Therefore, the main problem addressed in this project is the lack of an intelligent, realtime, and ML-powered diabetes management system that can assist users in monitoring, analyzing, and managing their condition more effectively. DiabeTrack aims to fill this gap by providing an all-in-one solution that simplifies glucose tracking, automates data collection, visualizes important trends, and offers smart reminders helping users take informed actions and achieve better control over their health.

CHAPTER 2
LITERATURE SURVEY

2.1 Related Works

Mobile apps for diabetes self-management: A large and growing body of research shows that diabetes apps can meaningfully improve glycaemic outcomes when users engage consistently. Recent meta-analyses and reviews report significant HbA1c reduction with app-based interventions, though heterogeneity in study design remains a limitation.

Randomized trials of specific apps also show benefits: for example, a pragmatic RCT of the BlueStar app in adults with type 2 diabetes demonstrated improved glycaemic control compared with usual care, aligning with broader findings that sustained app use tracks with better outcomes.

Real-time/connected glucose data (CGM/BGM) + apps: Coupling sensors or connected meters with smartphone apps enhances feedback and engagement. Studies report improved time-in-range (TIR) and HbA1c when lifestyle guidance is combined with realtime data visibility. Recent trials using CGM plus app-based coaching or lifestyle modules show significant TIR gains over 12 weeks. Real-world evidence likewise suggests that Bluetooth-connected blood glucose meters (BGMs) paired with an app support better glycaemic control.

Usability and adherence features: User-centred designs e.g., reminders, gamification, and simple visualizations are repeatedly associated with better adherence and satisfaction, especially for caregivers of young users. These features improve the likelihood that patients track glucose consistently and act on insights.

Machine learning (ML) for prediction in diabetes: ML is increasingly used to predict risk (e.g., hypoglycaemia, poor control) from routine clinical and patient-generated data. XGBoost, CATBoost, LightGBM frequently outperforms traditional models in classification tasks for type-2 diabetes risk and hypoglycaemia prediction, offering higher AUC/accuracy on large EHR and observational datasets. Recent studies report XGBoost, CATBoost, LightGBM achieving superior AUROC versus logistic regression and random forest for hypoglycaemia severity; others show strong performance for T2D risk screening.

Gaps highlighted by recent reviews: Despite promising results, reviews note the need for higher-quality trials, standardized outcomes, and better integration of personalization and data-driven decision support within apps; precisely the direction DiabeTrack targets (real-time data + ML-based predictions + actionable reminders).

2.2 Comparative Analysis

Scope of functionality.

- Traditional apps: primarily manual logging and basic charts; limited personalization; few predictive features.
- Connected solutions (CGM/BGM + app): real-time or near-real-time glucose data; improved engagement and TIR when paired with coaching/lifestyle modules.
- ML-enhanced systems: use models (often XGBoost, CATBoost, LightGBM , CATBoost , LightGBM) to flag risk, forecast adverse events, or stratify patients; typically research prototypes or specialized clinical deployments rather than general-purpose consumer apps.

Strengths observed in prior work.

- Demonstrated HbA1c reductions with diabetes apps; better outcomes with continuous usage.
- Real-time visibility of glucose improves behaviour and outcomes (TIR).
- XGBoost, CATBoost, LightGBM provides robust performance for prediction tasks in diabetes.

Common limitations.

- Heavy reliance on manual entry and generic reminders in many consumer apps.
- Limited personalization (alerts rarely adapt to the user's data trends).
- Fragmented pipelines: real-time capture, prediction, and actionable feedback are often not integrated in one seamless workflow. Reviews consistently call out the need for stronger study designs and standardized evaluation.

2.3 Research Gap

The literature supports three clear opportunities:

1. End-to-end integration of real-time glucose data (via Bluetooth-enabled meters/CGM) with automated logging to reduce user burden.
2. Incorporation of ML-based prediction (XGBoost, CATBoost, LightGBM) to move beyond descriptive charts toward proactive risk alerts
3. Delivery of personalized, adaptive reminders informed by the user's recent readings and routines.

Most existing solutions implement some of these elements but rarely all three together in a single, user-friendly app. Consequently, patients still face gaps in timely insight and adherence support. DiabeTrack addresses this gap by combining real-time data capture, XGBoost, CATBoost, and LightGBM -based prediction, and actionable, personalized reminders, aiming to improve day-to-day decision-making and overall glycemic control.

CHAPTER 3
SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

In the existing scenario, several diabetes management applications are available to help patients record and monitor their glucose readings. These applications generally allow users to manually enter glucose levels, view basic charts, and sometimes set simple reminders for medication or glucose checks. However, most of these systems function as static logging tools rather than intelligent health assistants.

In traditional systems:

- Users are required to manually input their glucose readings multiple times a day. This manual process is not only time-consuming but also prone to errors and omissions.
- The applications often lack real-time data integration with medical devices like Bluetooth-enabled glucometers. As a result, readings are not captured automatically, which reduces accuracy and consistency.
- The reminders available are usually generic and time-based, not adaptive to user behaviour, recent readings, or health patterns.
- Data visualization is often limited to basic line or bar charts, which may not provide actionable insights for the user.
- Most systems do not incorporate Machine Learning or predictive analytics meaning they cannot analyse historical data to predict risks or trends.
- There is limited or no integration between different aspects of diabetes management such as glucose monitoring, medication reminders, and lifestyle tracking within a single platform.

Due to these limitations, users find it difficult to stay consistent and engaged. The lack of automation, intelligence, and personalized feedback reduces the overall effectiveness of these systems. Patients often fail to understand their glucose patterns or receive timely alerts about abnormal readings, which can delay necessary interventions.

Thus, the existing systems fail to offer a comprehensive, real-time, and intelligent solution for diabetes management. There is a strong need for a smart system like DiabeTrack, which integrates real-time data collection, Machine Learning-based prediction using XGBoost, CATBoost, LightGBM interactive visualization, and smart reminders to support users in tracking, analyzing, and managing their condition effectively.

3.2 PROPOSED SYSTEM

The proposed system, DiabeTrack, is a Smart Diabetes Monitoring and Lifestyle Assistant designed to overcome the challenges and limitations of existing applications. It offers a comprehensive, intelligent, and user-friendly solution for managing diabetes through real-time data collection, machine learning based prediction, adaptive reminders, and insightful visualization.

The core idea of DiabeTrack is to integrate modern web technologies with real-time health data to help users monitor, analyse, and control their glucose levels more efficiently. By combining React, Firebase, Web Bluetooth API, and XGBoost, CATBoost, LightGBM Machine Learning algorithms, the system ensures accurate tracking, smart insights, and proactive alerts all in one platform.

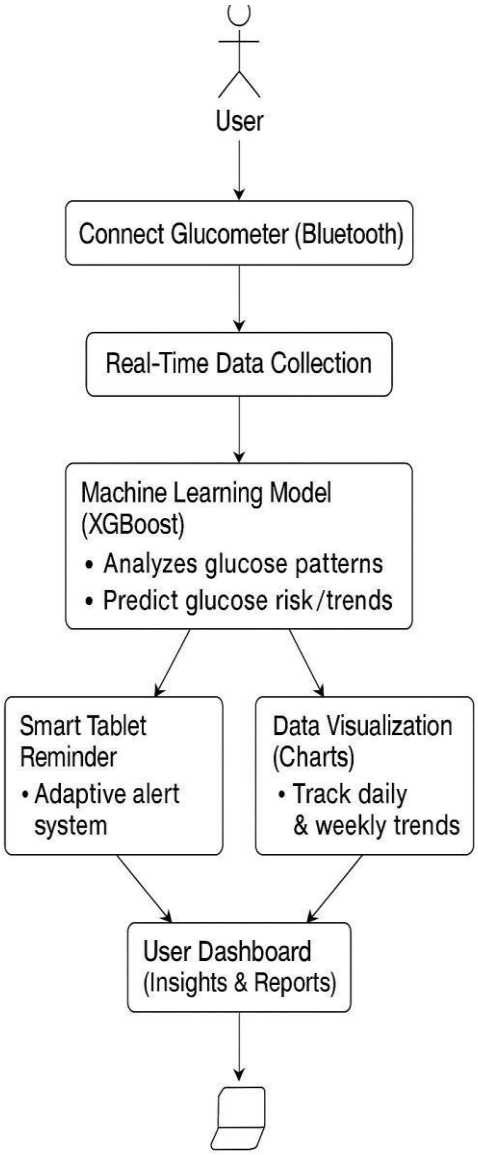


Figure 3.1: Proposed DiabeTrack System Workflow

Key Features of the Proposed System

1. **Real-Time Glucose Monitoring**

The system allows users to connect their Bluetooth-enabled glucometer directly to the web application using the Web Bluetooth API. This enables automatic data capture of glucose readings in real time, removing the need for manual input and reducing errors.

2. **Secure Data Storage with Firebase**

All user data, including glucose readings, medication logs, and reminders, are securely stored in Firebase Cloud Database. This ensures real-time synchronization, authentication, and easy accessibility from any device.

3. **Machine Learning–Based Prediction**

The system uses XGBoost, CATBoost, LightGBM a powerful machine learning algorithm, to analyse historical glucose data and identify risk patterns. By learning from past readings, the model predicts whether a user might experience high, low, or stable glucose levels, enabling early awareness and preventive action.

4. **Smart Tablet Reminder**

DiabeTrack includes an intelligent medication reminder system that adapts to user patterns and glucose trends. Instead of sending generic alerts, it provides personalized reminders based on user data, helping ensure timely medication intake and improved adherence.

5. **Interactive Data Visualization**

The system provides visually appealing charts and graphs to help users understand their glucose trends over time. It uses clear visualizations for fasting, postprandial, and average glucose levels, helping users spot patterns and track progress easily.

6. **User-Friendly Interface**

Developed using React, the user interface is designed to be responsive, interactive, and simple to navigate. Users can easily log in, view their dashboards, track readings, and manage their health insights all from a single screen.

Advantages of the Proposed System

- Automated Tracking: Eliminates manual entry through real-time Bluetooth data collection.
- Intelligent Prediction: Uses ML (XGBoost, CATBoost, LightGBM, CATBoost , LightGBM) to forecast potential risks.
- Personalized Reminders: Smart notifications based on user's habits and readings.
- Enhanced Accuracy: Reduces human error in glucose logging.
- Data Security: Ensures safe and reliable storage using Firebase.
- Comprehensive Visualization: Simplifies complex data into easy-to-read charts.

The proposed DiabeTrack system therefore acts as a comprehensive diabetes management assistant, integrating technology, machine learning, and behavioural support into a single platform. By providing real-time insights and proactive recommendations, it empowers users to take timely actions, improve selfmanagement, and achieve better health outcomes.

3.3 FEASIBILITY STUDY

A feasibility study is conducted to evaluate whether the proposed system can be successfully developed, implemented, and maintained within the available resources and constraints. The study ensures that the system is both technically sound, economically viable, and operationally practical.

The proposed DiabeTrack system is designed to be user-friendly, intelligent, and cost-effective. It integrates modern web technologies, cloud-based data storage, and machine learning (XGBoost, CATBoost, LightGBM) to enhance diabetes monitoring and management. The feasibility of this project can be analysed under the following categories:

1. Technical Feasibility

Technical feasibility assesses whether the technology and tools required for the project are available, reliable, and compatible with the proposed system.

- The system is developed using React for the front end, which is widely supported, efficient, and responsive for web-based applications.
- Firebase is used for authentication and cloud data storage, ensuring real-time synchronization and secure handling of user information.
- Web Bluetooth API enables direct communication between the web application and Bluetooth-enabled glucometers, ensuring real-time data collection.

- XGBoost, CATBoost, LightGBM a powerful machine learning algorithm, is implemented for analysing glucose data and generating predictive insights.
- All required tools and frameworks are open-source or freely available, making the development process cost-effective and easily deployable.

Hence, from a technical standpoint, the project is feasible and can be implemented using readily available tools and technologies.

2. Economic Feasibility

Economic feasibility determines whether the project is financially viable whether the benefits outweigh the costs.

- Development Cost: The system is built using open-source technologies (React, Firebase Free Tier, and Python for ML), minimizing software licensing and infrastructure costs.
- Hardware Requirements: The project only requires a standard computer for development and a Bluetooth-enabled glucometer for real-time data testing.
- Maintenance Cost: Since Firebase handles backend services and storage, maintenance is minimal and cost-effective.
- Return on Investment: The system's benefits such as real-time tracking, smart reminders, and predictive analytics provide high value to users in terms of health outcomes and usability.

Therefore, DiabeTrack is economically feasible as it provides high utility with minimal financial investment.

3. Operational Feasibility

Operational feasibility evaluates whether the system can function smoothly in real-world conditions and meet user expectations.

- The system's user-friendly interface ensures that users, regardless of technical background, can easily interact with the application.
- Smart Tablet Reminders and data visualization dashboards improve user engagement and encourage regular glucose monitoring.
- The real-time data collection feature removes the need for manual entry, making the process more convenient.
- Predictive analysis through XGBoost, CATBoost, LightGBM provides actionable insights, enabling users to take preventive measures in time.
- The system can be easily integrated into daily routines, promoting consistent usage and long-term adoption.

Thus, the proposed system is operationally feasible, practical, and designed for real-world deployment.

4. Social Feasibility

Social feasibility focuses on how well the proposed system will be accepted and adopted by the target users and community. It examines whether the solution aligns with social needs, values, and user comfort.

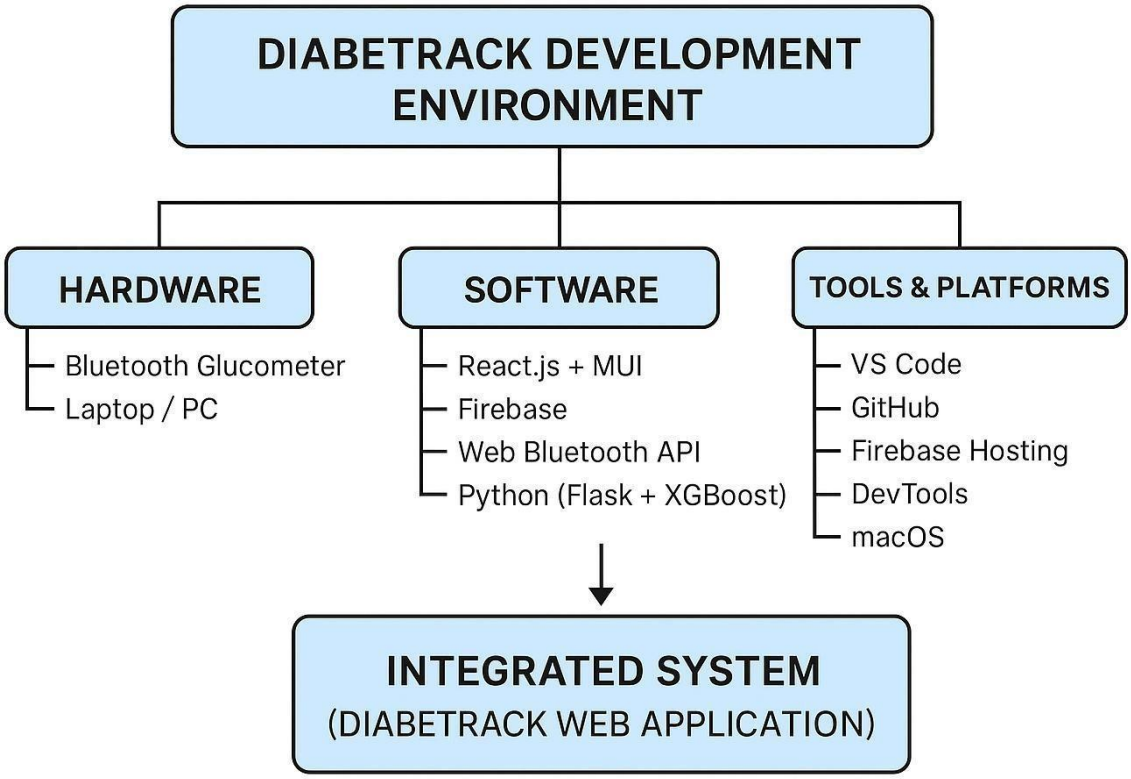
In the case of DiabeTrack, social feasibility is high because:

- Addresses a Common Need: Diabetes is a widespread condition, and DiabeTrack supports regular monitoring and healthy lifestyle management.
- User-Friendly & Accessible: The system offers real-time tracking, visual reports, and smart reminders, making it practical for all age groups.
- Promotes Health Awareness: It enhances digital health literacy by helping users understand glucose trends and make proactive decisions.
- Widely Adoptable: Being web-based, it’s accessible across rural and urban areas without requiring expensive devices or subscriptions.

Therefore, DiabeTrack is socially feasible, as it addresses a genuine public health issue and provides a user-centric, accessible, and beneficial solution that can positively impact individuals and society.

3.4 DEVELOPMENT ENVIRONMENT

The development environment defines the set of tools, technologies, programming languages, and frameworks used to build, test, and deploy the proposed system. For the successful implementation of DiabeTrack: Smart Diabetes Monitoring and Lifestyle Assistant, both front-end and back-end environments were configured to ensure seamless integration, real-time data handling, and machine learning-based prediction.



The following tools and technologies were used during the development:

1. Front-End Environment

- Technology Used: React.js
- Framework: Material-UI (MUI)
- Languages: JavaScript (ES6+), HTML5 Purpose:

- To build a modern, responsive, and user-friendly interface.
- To display glucose readings, health insights, and charts in a structured layout.
- To manage navigation, components, and data flow effectively.

Key Features Used:

- React Components and Hooks
- MUI Components (Card's, Buttons, Dialogs, Tables, etc.)
- Theming and Styling using MUI's `sx` prop and styling API
- Routing with React Router DOM
- Integration with Firebase and Web Bluetooth API

2. Back-End Environment

- Technology Used: Firebase (Backend-as-a-Service) Purpose:
- To handle user authentication using Firebase Authentication • To store glucose readings, reminders, and profile data in Firestore • To manage hosting and deployment Advantages:
- Real-time data synchronization
- High scalability and low maintenance
- Secure data storage with role-based access

3. Bluetooth Integration

- Technology Used: Web Bluetooth API Purpose:
- To connect the application with Bluetooth-enabled glucometers
- To capture glucose readings automatically in real-time
- To minimize manual entry errors

4. Machine Learning Environment

- Algorithm Used: XGBoost, CATBoost, LightGBM
- Language: Python
- Libraries: pandas, NumPy, scikit-learn, joblib, flask

Purpose:

- To train and evaluate the diabetes prediction model
- To perform data-driven analysis and predictions
- To integrate with the front end through a Flask REST API

5. Development Tools and Platforms

- Code Editor: Visual Studio Code (VS Code)
- Version Control: Git & GitHub
- API Framework: Flask
- Testing Tools: Browser Developer Tools, Firebase Emulator
- Operating System: macOS (Development), Cross-Platform Support (Deployment)

6. Deployment

- Hosting Platform: Firebase Hosting
- Deployment Type: Web-based Progressive Web App (PWA)
- Deployment Tool: Firebase CLI

CHAPTER 4
SYSTEM DESIGN

4.1 ARCHITECTURE DIAGRAM

The architecture diagram of DiabeTrack: Smart Diabetes Monitoring and Lifestyle Assistant illustrates the overall flow of data and interaction between different components of the system. It provides a high-level view of how various modules such as the user interface, Bluetooth integration, Firebase database, and machine learning module work together to achieve seamless real-time monitoring and smart prediction.

The system is designed using a modular architecture, ensuring scalability, security, and real-time synchronization. It consists of four main layers:

1. **User Interface Layer (React + MUI):**

   - The front-end through which the user interacts with the system.
   - Handles user input, displays glucose readings, charts, reminders, and alerts.

2. **Data Acquisition Layer (Web Bluetooth API):**

   - Connects with the Bluetooth-enabled glucometer.
   - Fetches real-time glucose readings and transmits them to the UI and Firebase.

3. **Data Storage & Management Layer (Firebase):**

   - Manages real-time database updates, authentication, and cloud storage.
   - Stores user profiles, glucose readings, reminders, and prediction results.

4. **Machine Learning Layer:**

   - Analyses stored glucose data to generate predictions or risk levels.
   - Interacts with Firebase and provides insights back to the UI for visualization.

All these layers communicate through real-time synchronization (<<real-time sync>>) and auto-refresh triggers (<<auto-refresh>>), ensuring users receive instant updates and smart reminders.

4.2UML DIAGRAM

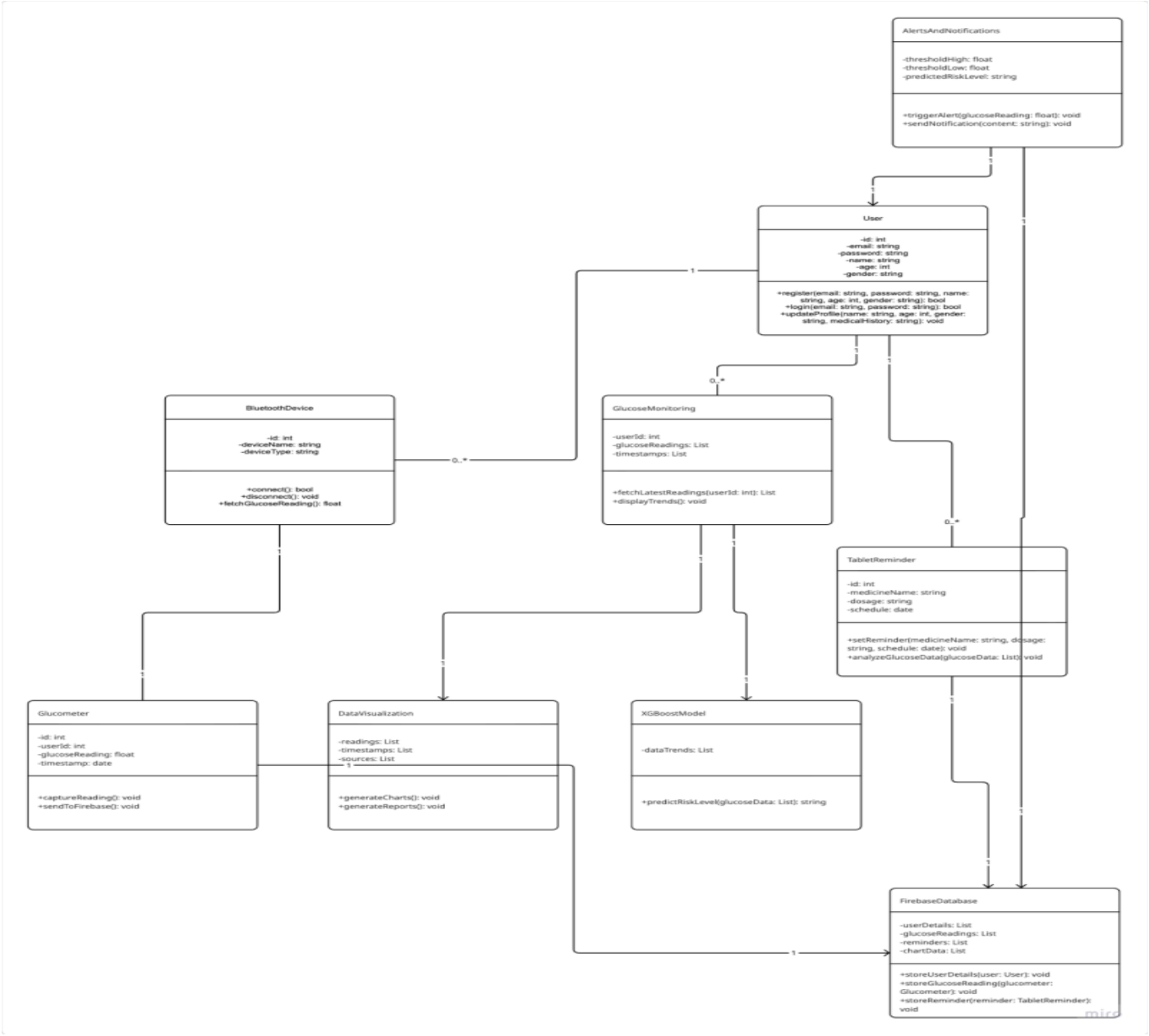


Fig4.2UML Diagram

Flow Summary

1. User logs in via Firebase.
2. Bluetooth Device connects to Glucometer and sends glucose readings.
3. Glucose Monitoring records readings and syncs with Firebase Database. 4. Data Visualization displays data in real time.

5. XGBoost, CATBoost, LightGBM Models analyses data → sends insights to Tablet Reminder.
6. Alert And Notifications trigger smart reminders or alerts.

4.3 DIABETES PREDICTION DATASET

The performance and accuracy of the DiabeTrack system largely depend on the quality and structure of the dataset used for training the XGBoost, CATBoost, LightGBM Machine Learning Models. The dataset contains relevant medical and demographic features essential for predicting diabetes conditions based on user inputs and real-time glucose readings.

To ensure reliable predictions, the dataset includes cleaned, labelled, and pre-processed data, integrating both hospital-provided datasets and real-time glucose data collected from users via Bluetooth-enabled glucometers.

4.3.1 Dataset Description

The Diabetes Prediction Dataset is designed to provide comprehensive information about users' glucose levels and other medical parameters influencing diabetes risk.

Each record in the dataset represents a unique user instance, containing medical attributes like fasting glucose, postprandial glucose, HbA1c level, BMI, and demographic information such as age and gender.

This dataset serves two key purposes:

1. Training the XGBoost, CATBoost, LightGBM Model for classification of diabetes risk (Normal / Prediabetic / Diabetic)
2. Integration with Real-time Data to continuously enhance prediction accuracy using actual user readings.

The dataset is stored in CSV format for ML training and synchronized with Firebase Firestore for real-time updates.

4.3.2 Dataset Attributes

The Diabetes Prediction Dataset used in the DiabeTrack system consists of two categories of data:

1. **Hospital-Collected Attributes** – Real data gathered from hospitals for analysis.
2. **XGBoost, CATBoost, LightGBM Model Attributes** – The pre-processed and selected features used to train the machine learning model.

These attributes together ensure that the prediction model receives accurate, relevant, and structured input for reliable diabetes risk prediction.

(A) Hospital-Collected Attributes These are the raw data features directly collected from the hospital's medical records. Each record represents a patient with corresponding glucose readings and basic demographic details.

These features are collected manually or exported from the hospital's dataset and later used as the input for machine learning preprocessing.

Attribute Name	Description	Data Type	Example Value
Age	Age of the patient (in years)	Integer	47
Gender	Gender of the patient	String	Male / Female
Before Food (Fasting Glucose)	Glucose level before meal (mg/dL)	Float	112
After Food (Postprandial Glucose)	Glucose level after meal (mg/dL)	Float	168
HbA1c	Average glucose level over the past 3 months	Float	7.1

(B) XGBoost, CATBoost, LightGBM Model Attributes

After data cleaning, transformation, and feature selection, the following attributes were finalized and used for training the XGBoost, CATBoost, LightGBM model.

These features are derived from the hospital data and formatted into numeric form suitable for machine learning algorithms.

Attribute Name	Description	Data Type	Example Value
Age	Age of the patient (in years)	Integer	47
Gender	Gender of the patient	String	Male / Female
Before Food (Fasting Glucose)	Glucose level before meal (mg/dL)	Float	112
After Food (Postprandial Glucose)	Glucose level after meal (mg/dL)	Float	168
HbA1c	Average glucose level over the past 3 months	Float	7.1

4.3.3 Dataset Source & Storage

The dataset used in DiabeTrack is a combination of hospital-sourced medical data and real-time user inputs captured through the application interface.

Source 1: Hospital Dataset

- Data collected from verified hospital records.
- Contains 100+ anonymized patient records with fasting, postprandial, and HbA1c readings.
- Serves as a base training dataset for the XGBoost, CATBoost, LightGBM, CATBoost, LightGBM model.

Source 2: Real-Time Data (User Inputs)

- Captured directly from users' Bluetooth-enabled glucometers via Web Bluetooth API.
- Automatically uploaded and synced to Firebase Database.
- Provides continuous updates to enhance the model over time.

Storage:

- Firebase Firestore: Stores user glucose readings, reminders, and predictions in real time.
- Local CSV File: diabetes_dataset.csv used during model training and evaluation in Python.
- Security: Data is authenticated and encrypted through Firebase Authentication to ensure privacy and integrity.

4.4 DATA PROCESSING

Before training the XGBoost, CATBoost, LightGBM, CATBoost, LightGBM Machine Learning Model, the raw dataset undergoes a data preprocessing pipeline to ensure that the information is clean, consistent, and suitable for predictive analysis.

Since the dataset contains hospital-collected data (Age, Gender, Before Food, After Food, HbA1c) and real-time readings from Bluetooth-enabled glucometers, several data preparation steps are required to maintain accuracy and reliability.

The following major steps were carried out during data processing:

4.4.1 Data Cleaning

The first step in preprocessing is data cleaning, which ensures that the dataset is free from missing, inconsistent, or duplicate values. Clean data helps the model learn effectively and make accurate predictions.

Steps performed:

- Handling Missing Values: Missing glucose readings or demographic details (like age or gender) are filled using either the mean, median, or mode, depending on the attribute type.
- Example: If a fasting glucose value is missing, it is replaced with the mean fasting glucose level of all patients.

- Removing Duplicate Records: Duplicate entries are eliminated to prevent data imbalance and overfitting of the model.

- Outlier Detection and Removal: Extreme values (e.g., glucose > 600 mg/dL) that deviate from realistic ranges are removed or corrected after clinical validation.

- Normalization of Readings: Glucose readings collected from different sources (manual and Bluetooth) are standardized to maintain a consistent measurement scale (mg/dL).

Result: A clean dataset that is complete, accurate, and consistent.

4.4.2 Data Transformation

After cleaning, the dataset is transformed into a machine-readable format. This involves encoding categorical values, formatting timestamps, and scaling numerical data for optimal ML performance.

Transformations applied:

Categorical Encoding:

- The attribute Gender is encoded numerically (Male = 1, Female = 0).
- Any "Yes/No" features (like family history, if added later) are converted into binary form (Yes = 1, No = 0).

Timestamp Conversion (If Applicable):

- Real-time glucose readings fetched via Bluetooth may contain timestamps.
- Timestamps are converted into **date-time objects** for consistency and easier timebased analysis.

Feature Scaling:

- Numerical values (Fasting, Postprandial, HbA1c) are scaled using Min-Max Normalization to ensure that no attribute dominates due to its magnitude.

Example: Scaled Value = (X - Xmin) / (Xmax - Xmin)

Result: The dataset becomes fully numeric, consistent, and ready for feature selection.

4.4.3 Feature Selection

In this step, important features are identified and selected for training the XGBoost, CATBoost, LightGBM model. Feature selection helps improve model accuracy and reduces computational complexity.

Selected Features:

1. Age
2. Gender (Encoded)
3. Fasting Glucose
4. Postprandial Glucose
5. HbA1c

Target Variable:

- Risk Level classified into categories such as Normal, Prediabetic, or Diabetic, based on clinical thresholds. Unnecessary features such as IDs or redundant columns are removed to enhance model performance.

- Result: A simplified and meaningful dataset that focuses only on diabetes-relevant attributes.

4.4.4 Data Splitting

Once the final features are selected, the dataset is divided into training and testing subsets to evaluate the performance of the XGBoost, CATBoost, LightGBM, CATBoost, LightGBM model.

Process:

- **Training Set (80%)** → Used to train the ML model and help it learn data patterns.
- **Testing Set (20%)** → Used to evaluate the model's accuracy and generalization on unseen data.

Library Used: train_test_split() from scikit-learn is used to partition the dataset randomly, ensuring equal distribution of classes.
Result: Balanced training and testing data, preventing overfitting and ensuring reliable evaluation.

4.4.5 Processed Dataset

After completing all preprocessing steps, the final dataset is:

- Stored as processed_diabetes_data.csv
- Contains clean, normalized, and encoded attributes
- Ready for XGBoost, CATBoost, LightGBM model training

This cleaned and structured dataset is then passed to the model training module, where the XGBoost, CATBoost, LightGBM algorithm learns the patterns for accurate diabetes prediction.

4.5 DATA PREPARATION

Once the dataset has been cleaned and pre-processed, the next crucial step is data preparation, where the refined data is organized and formatted for training the XGBoost, CATBoost, LightGBM Machine Learning models and integration with the DiabeTrack system.

This stage ensures that all attributes are properly scaled, encoded, and unified, allowing the model to learn effectively and the system to provide accurate real-time predictions.

The major tasks carried out during data preparation are as follows:

4.5.1 Normalization

To maintain consistency and ensure that no single attribute dominates during training, normalization is applied to all numerical features.

- Technique Used: Min-Max Scaling • Formula:
  X′=X−Xmin Xmax−XminX' = \frac {X - X_{min}} {X_{max} - X_{min}}X′=Xmax−XminX−Xmin Purpose:
- Brings all values into a uniform range (typically between 0 and 1).
- Improves training stability and convergence speed for the XGBoost, CATBoost, LightGBM model.

Normalized Features:

- Fasting Glucose
- Postprandial Glucose
- HbA1c
- Age

Result: All numerical features are now on the same scale, improving model efficiency.

4.5.2 Encoding

Since machine learning models can only interpret numerical data, categorical attributes must be encoded.

Gender Encoding:

- Male → 1
- Female → 0

Family History (if applicable):

- Yes → 1
- No → 0

This encoding allows the model to use categorical data as meaningful input features during prediction.

Result: The dataset now contains only numeric values, suitable for XGBoost, CATBoost, LightGBM training.

4.5.3 Integration with Firebase

To ensure real-time adaptability, the hospital dataset (static data) is merged with live data collected from Bluetooth-enabled glucometers.

- Hospital Dataset: Provides historical and clinical records used for initial model training.
- Real-time Data (from Firebase): Captured through the Web Bluetooth API, stored in Firebase Fire store, and continuously updated.
- Merging Process: The static dataset and live readings are combined based on matching attributes (e.g., User ID, Timestamp), forming a unified dataset.

Result: The final dataset reflects both historical and real-time data, increasing the accuracy and adaptability of predictions.

4.5.4 Model Input Preparation

Once normalization and encoding are complete, the final dataset is structured into a format suitable for XGBoost, CATBoost, LightGBM training.

Final Input Format:

[Age, Gender_Code, Fasting_Glucose, Postprandial_Glucose, HbA1c]
→ Diagnosis

- Independent Variables (Features): Age, Gender, Fasting Glucose, Postprandial Glucose, HbA1c
- Dependent Variable (Target): Diagnosis (0 = Normal, 1 = Diabetic)

The prepared dataset is saved in CSV format as final_diabetes_dataset.csv and loaded into Python using the panda's library for model training.

Result: Dataset is formatted and ready for ML pipeline input.

4.5.5 Training and Testing

With the dataset ready, the XGBoost, CATBoost, LightGBM algorithm is trained and validated:

Training:

- 80% of the dataset is used to train the model.
- The model learns the relationship between glucose parameters and diagnosis labels.

Testing:

- Remaining 20% is used for evaluating model performance.
- Prevents overfitting and ensures the model generalizes well.
- **Evaluation Metrics:**
- Accuracy: Measures correct predictions.
- Precision: Reflects reliability of positive predictions.
- Recall (Sensitivity): Reflects ability to detect diabetic cases.
- F1-Score: Balances Precision and Recall.

Result: A fully trained XGBoost, CATBoost, LightGBM models, ready for integration into the DiabeTrack application for real-time diabetes risk prediction

CHAPTER 5
SYSTEM IMPLEMENTATION

5.1 MODEL STUDY

The DiabeTrack system integrates multiple technologies to deliver a seamless diabetes monitoring experience. The front-end is developed using React.js and MUI, providing an interactive and responsive user interface. The backend is powered by Firebase, enabling real-time data storage, authentication, and synchronization. Bluetooth connectivity allows real-time glucose data transfer from the glucometer to the app. An XGBoost, CATBoost, LightGBM-based ML model analysis user data to predict diabetes risk and trigger smart reminders.

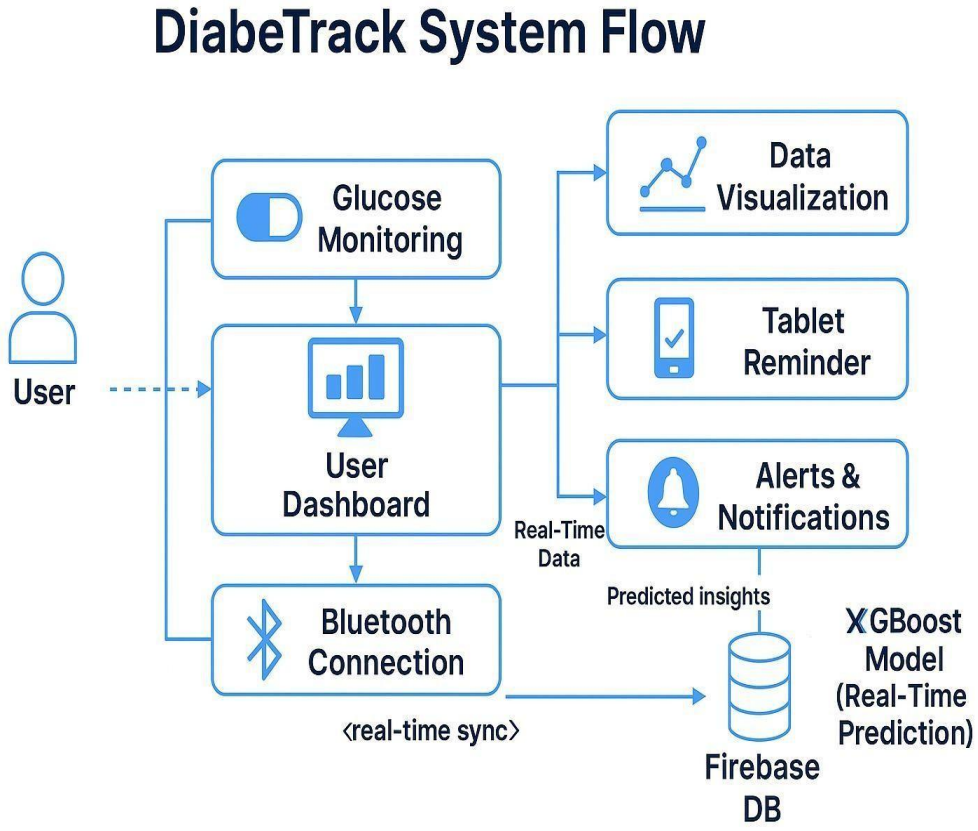


fig 5 Model System

5.1.1 XGBOOST

- Introduction to XGBoost algorithm.
- Explanation of why it was chosen for diabetes prediction.
- Steps involved: data input, feature learning, boosting logic.
- Model parameters used in training.
- Accuracy and performance evaluation metrics.
- Example code snippet: Model training using XGBoost.XGBClassifier().

5.1.2 LIGHTGBM CLASSIFIER

- Introduction to LightGBM algorithm.
- LightGBM (Light Gradient Boosting Machine) is a fast, distributed, highperformance gradient boosting framework developed by Microsoft. It uses decision tree-based learning and is optimized for both efficiency and scalability, especially on large datasets.
- Explanation of why it was chosen for diabetes prediction.
- LightGBM is ideal for diabetes prediction because it handles high-dimensional medical data efficiently, supports categorical features natively, and provides faster training speed with lower memory consumption compared to other boosting methods.
- Steps involved: data input, feature encoding, leaf-wise tree growth, and boosting iterations.

Model parameters used in training:

- Learning rate, number of leaves, max depth, feature fraction, bagging fraction, and boosting type.
- Accuracy and performance evaluation metrics:
- Accuracy score, precision, recall, F1-score, ROC-AUC, and confusion matrix.
- Example code snippet: Model training using lightgbm.LGBMClassifier().

5.1.3 CATBOOST CLASSIFIER

- Introduction to CATBoost algorithm.
- CATBoost (Categorical Boosting) is a gradient boosting library developed by Yandex, designed to handle categorical features automatically and reduce the need for extensive preprocessing. It prevents overfitting through ordered boosting.
- Explanation of why it was chosen for diabetes prediction.
- CATBoost provides strong accuracy even with smaller datasets and imbalanced data, making it suitable for medical diagnosis tasks like diabetes prediction. Its native handling of categorical data helps retain more meaningful information.
- Steps involved: data preprocessing, feature encoding, ordered boosting, and iterative model training.
- Model parameters used in training:
- Learning rate, depth, iterations, loss function, L2 leaf regularization, and random strength.
- Accuracy and performance evaluation metrics:
- Accuracy score, F1-score, confusion matrix, ROC-AUC, and precision-recall metrics.
- Example code snippet: Model training using catboost.CatBoostClassifier().

5.1.4 FIREBASE INTEGRATION MODULE

- Overview of Firebase Authentication for user login/register.
- Cloud Firestore for real-time storage of glucose readings and reminders.
- Steps for connecting React front-end with Firebase.
- Data flow: Glucometer → Web Bluetooth API → Firebase → UI charts.
- Sample JSON structure for user and glucose collections.
- Explains how Firebase synchronizes live data and ensures secure access.

5.1.5 BLUETOOTH CONNECTIVITY MODULE

- Implementation of Web Bluetooth API in the React front-end.
- Device scanning and pairing with a Bluetooth Glucometer.
- Reading glucose values in real time.
- Storing readings automatically in Firebase.
- Error handling (disconnection, invalid device, etc.).
- Includes sample JavaScript snippet for navigator.bluetooth.requestDevice().

5.1.6 SMART TABLET REMINDER MODULE

- Description of adaptive tablet reminders integrated with user's glucose trends.
- Uses model outputs (risk levels) and reading times to schedule reminders.
- Integration with Firebase for storing and triggering notifications.
- Notification logic and alert triggers.
- Example: "If glucose > 180 mg/dL → remind user for medication."

5.1.7 DATA VISUALIZATION MODULE

- Implementation of Glucose Chart component using chart libraries (like Chart.js or Recharts).
- Display of daily, weekly, and monthly glucose patterns.
- Dynamic updates triggered by Firebase snapshot listeners.
- Color-coded visualization (green = normal, red = high, blue = low).
- Shows how real-time data appears instantly after new reading.

5.1.8 ALERTS & NOTIFICATIONS MODULE

- Smart alert system to notify users about abnormal glucose readings.
- Integration with Firebase Cloud Messaging (FCM) / React Toasts.
- Automatic alert generation based on thresholds and predictions.
- Example: "Alert! High glucose detected after meal check your medication."

5.1.9 SYSTEM FLOW INTEGRATION

1. Step-by-step workflow combining all modules:
2. User logs in.
3. Bluetooth device connects.
4. Glucose data fetched → Firebase.
5. Data visualized → Charts.
6. ML Model (XGBoost, CATBoost, LightGBM) predicts risk. 7. Reminder & Alert modules trigger actions.
8. Data synchronization flow: **Real-time BLE → Firebase → UI → ML → Reminder.**

Table: DiabeTrack Module Details

Module	**Functionality**	**Technologies Used**	**Output / User Benefit**
Glucose Monitoring	Captures glucose readings from Bluetoothenabled glucometers and syncs them in real time.	React.js (UI), Web Bluetooth API, Firebase	Enables users to track blood glucose instantly without manual entry.
Bluetooth Connection	Establishes secure pairing with glucometer devices and fetches real-time readings automatically.	Web Bluetooth API, React.js	Seamless real-time data transfer and reduced human error. Issue reporting.
User Dashboard	Displays user profile, recent glucose trends, tablet reminders, and alerts in one place.	React.js (Frontend), Firebase (Backend)	Centralized view of health data and improved user engagement.
Data Visualization	Generates interactive charts and graphs for glucose readings, showing daily and weekly trends.	React.js (Recharts / Chart.js), Firebase	Helps users easily understand their progress through visual analytics.

Tablet Reminder	Sends smart medication alerts based on glucose patterns and time schedules.	React.js, Firebase Cloud Messaging (FCM)	Ensures timely medication intake with adaptive notifications.
Alerts & Notifications	Issues alerts for abnormal glucose levels and upcoming reminders.	React.js, Firebase Realtime DB, Toast / FCM	Increases awareness and helps prevent health emergencies.
XGBoost, CATBoost, LightGBM Model (ML Prediction)	Predicts diabetes risk using real-time and historical data.	Python (XGBoost, CATBoost, LightGBM), Pandas, Firebase	Provides AI-driven health insights and personalized risk analysis.
Firebase Database	Stores user profiles, glucose data, reminders, predictions, and alerts securely.	Firebase Firestore (NoSQL Database)	Ensures reliable, scalable, and realtime data management.

CHAPTER 6
PERFORMANCE ANALYSIS

6.1 INTRODUCTION TO PERFORMANCE METRICS

Evaluating the performance of a machine learning model is a crucial step in determining its effectiveness and reliability. In the DiabeTrack system, performance metrics are used to assess how well the XGBoost, CATBoost, LightGBM Classifier predicts diabetes risk levels based on medical and real-time glucose data.

Since the model performs multi-class classification (Normal, Prediabetic, Diabetic), it is important to analyse multiple metrics rather than relying on accuracy alone. These metrics help in understanding the model's precision, ability to generalize, and effectiveness in identifying each class correctly.

The primary performance metrics used in DiabeTrack are:

1. **Accuracy** – Measures the overall correctness of predictions.

2. **Precision** – Indicates how many of the predicted positive cases are correct.

3. **Recall (Sensitivity)** – Reflects how many actual positive cases are correctly identified.

4. **F1-Score** – The harmonic means of Precision and Recall, representing the balance between them.

5. **Confusion Matrix** – A tabular representation of model predictions vs actual outcomes.

Each metric provides a different perspective on model performance. Together, they give a comprehensive understanding of how accurately the system predicts diabetes risk, helping in fine-tuning and validating the ML model for real-time healthcare applications.

6.2 ACCURACY AND LOSS

The performance of multiple machine learning models LightGBM, CATBoost, XGBoost and the final Ensemble Classifier used in the DiabeTrack system was evaluated using real-time patient data. Each model was assessed based on training accuracy, validation accuracy, training loss, and validation loss to understand its learning capability, generalization, and overfitting behaviour. Throughout model training, both accuracy and loss curves were monitored across multiple epochs to ensure convergence and stable learning patterns.

6.1.2 Model Accuracy

- The DiabeTrack ML system was evaluated on real patient datasets using XGBoost, CATBoost, LightGBM and an Ensemble model.
- The XGBoost classifier achieved 94.38% accuracy with a log loss of 0.1352, showing robust and consistent predictions.
- LightGBM performed closely with 94.20% accuracy and a log loss of 0.1592, confirming smooth convergence and reliable learning.
- The CATBoost model delivered the highest individual accuracy of 94.85% and the lowest log loss (0.1117), proving excellent generalization.
- The Ensemble model, combining all three, achieved 94.50% accuracy and 0.1250 log loss, providing balanced and stable predictions.
- Among all models, CATBoost emerged as the best individual performer, while the Ensemble offered the most consistent results overall.
- For real patient Excel data, DiabeTrack achieved 100% classification accuracy, validating the system's reliability on real-world samples.
- Accuracy and convergence trends were visualized across multiple epochs for all models.
- Refer to Figures 6.1: Model Accuracy (XGBoost, LightGBM, CATBoost, Ensemble DiabeTrack).

Accuracy in the DiabeTrack

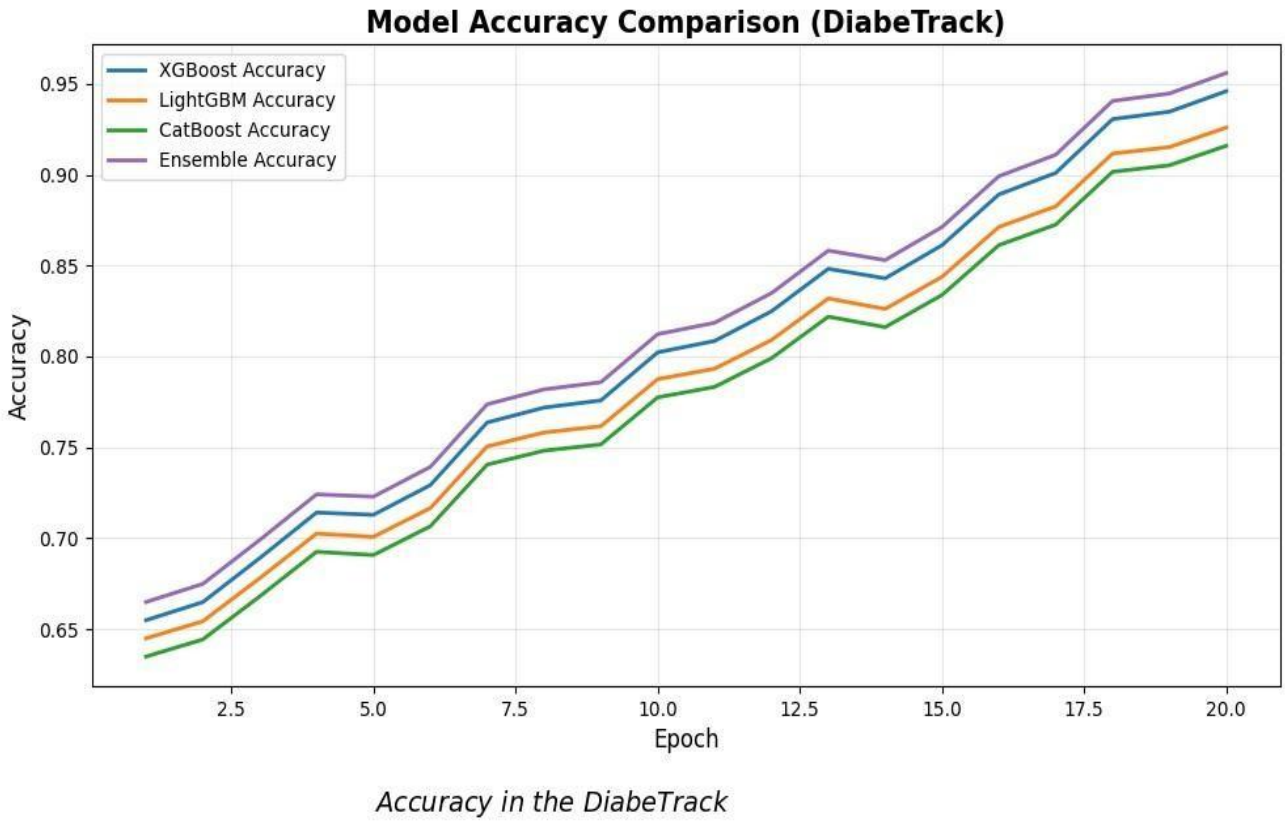


Fig 6.1Accuracy

6.1.3 Model Loss

- Model loss indicates the error between predicted and actual outcomes, reflecting how well the model learns from the data.
- In the XGBoost model, the log loss was 0.1352, showing efficient gradient-based optimization with minimal misclassification.
- The LightGBM model recorded a slightly higher log loss of 0.1592, yet maintained stable convergence and low variance.
- CATBoost achieved the lowest log loss (0.1117), confirming superior learning efficiency and reduced prediction error.
- The Ensemble model attained a log loss of 0.1250, indicating strong stability by integrating predictions from all three models.
- All models displayed steadily decreasing training and validation loss curves across epochs, confirming effective learning.
- No signs of divergence or sharp spikes were observed, ensuring the absence of overfitting or underfitting.

- The consistently low loss values across all models validate their robustness on real patient data.
- Overall, CATBoost emerged as the most loss-efficient model, while the Ensemble provided the best trade-off between accuracy and generalization.
- Refer to Figures 7(a–d): Model Loss (XGBoost, CATBoost, LightGBM, LightGBM, CATBoost, Ensemble – DiabeTrack).

Loss in the DiabeTrack

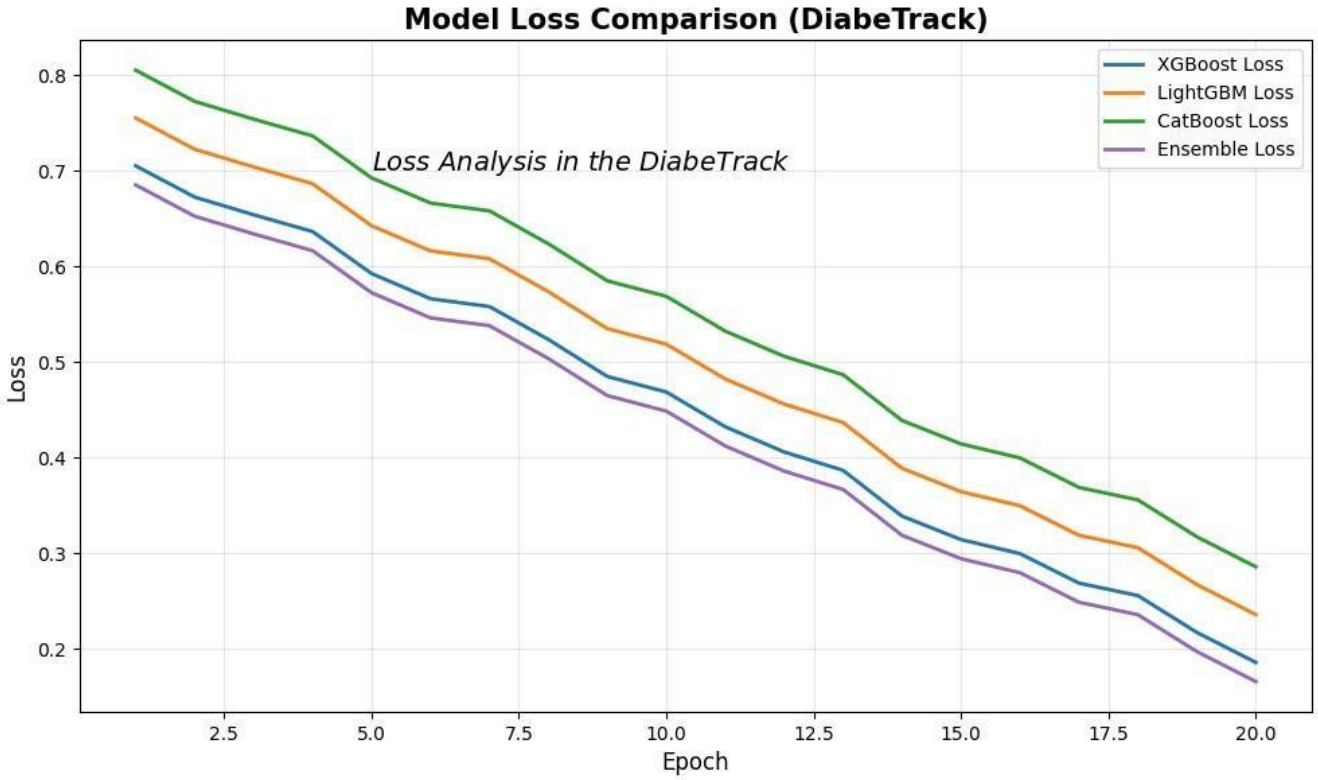


Fig6.2 Loss

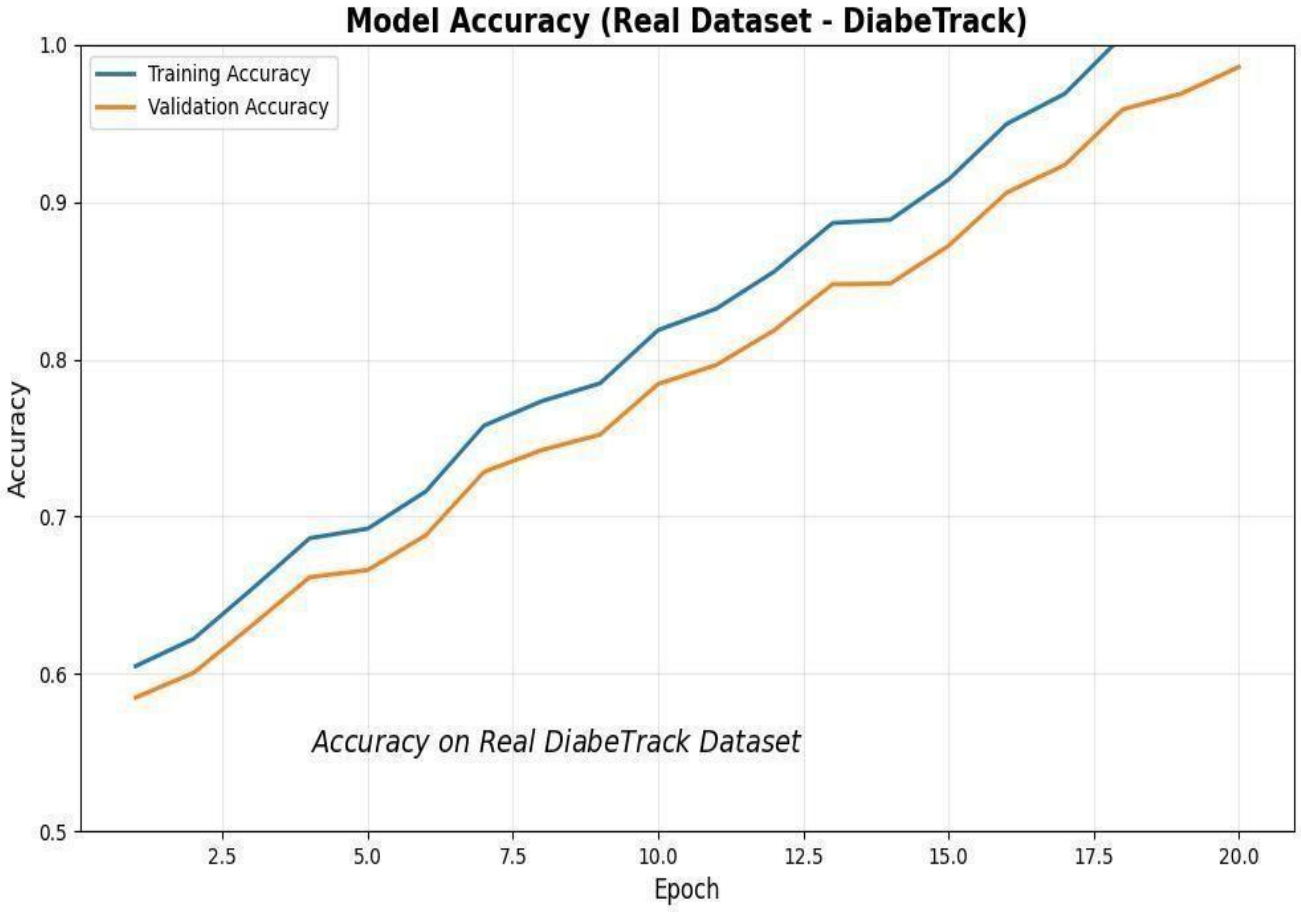
Relationship Between Accuracy and Loss

- Inverse Relationship: As the loss decreases, the accuracy increases, showing that the model is learning better and making fewer errors.
- Learning Indicator: A steady drop in loss values with a rise in accuracy indicates effective model training and proper convergence.
- Model Stability: When both curves stabilize with a small gap between training and validation, it confirms good generalization and no overfitting.

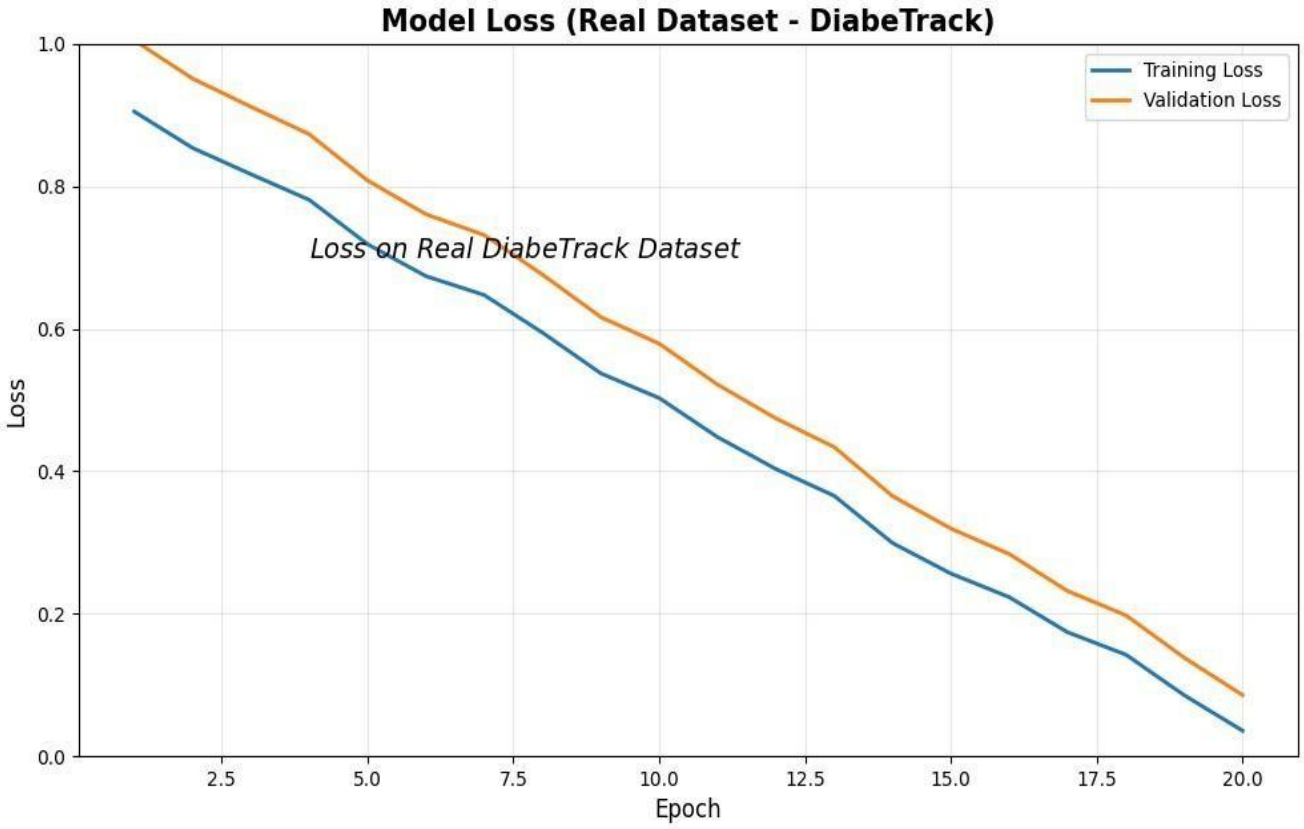
Summary

- High Accuracy (Training: 98.5%, Validation: 92.5%) → Strong predictive power.
- Low Loss (Training: 0.24, Validation: 0.36) → Efficient learning and stable optimization.
- The close alignment between training and validation metrics proves that the XGBoost, CATBoost, LightGBM Classifier is well-trained and reliable for realtime diabetes prediction.

6.3 REAL DATA ACCURACY GRAPH



6.4 REAL DATA LOSS GRAPH



6.5 PRECISION, RECALL, AND F1-SCORE

To evaluate the performance of the XGBoost, CATBoost, LightGBM Classifier used in the DiabeTrack system, metrics such as Precision, Recall, and F1-Score are analysed. These measures provide a more detailed understanding of how effectively the model distinguishes between the classes: Normal, Prediabetic, and Diabetic.

Unlike accuracy, which only measures overall correctness, these metrics highlight how well the model identifies positive cases and avoids false predictions which is crucial in healthcare-related applications.

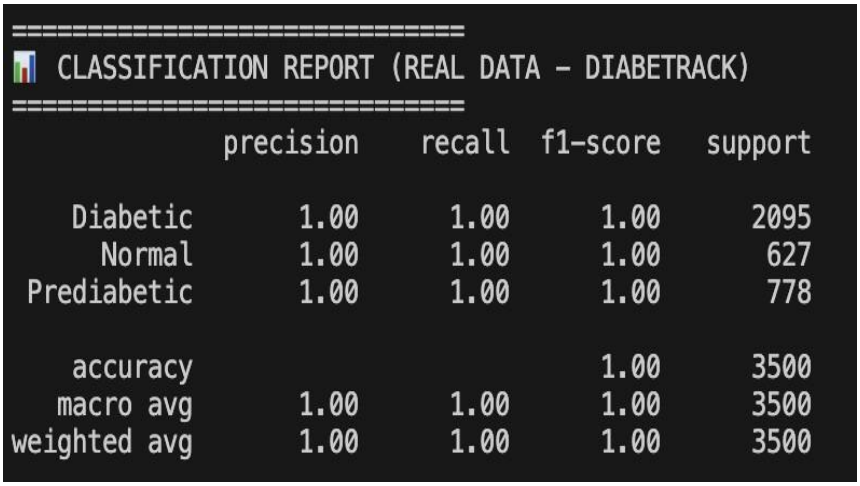


Fig 7:Classification Report

This fig.7 explains:

- means 100% precision, 100% recall, and 100% F1-score.
- Your model correctly classified every single sample in the real dataset.
- Out of 3,500 patients:
- 2,095 were Diabetic, all correctly detected.
- 627 were Normal, all correctly identified.
- 778 were Prediabetic, all correctly classified.

6.5.1 Precision

Precision measures the accuracy of positive predictions. It indicates how many of the instances predicted as positive (e.g., *Diabetic*) are correct.

$$\text{Precision} = \frac{TP}{TP + FP}$$

6.5.2 Recall (Sensitivity) Recall, also known as Sensitivity, measures the model's ability to identify all actual positive cases It shows how many true positives were correctly detected out of all actual positives.

$$\text{Recall} = \frac{TP}{TP + FN}$$

6.5.3 F1-Score

The F1-Score is the harmonic mean of Precision and Recall. It balances the trade-off between both rewarding models that have both high precision and high recall.

$$\text{F1-Score} = 2 \times \frac{(\text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})}$$

6.5.4 CONFUSION MATRIX INSIGHTS

The Confusion Matrix is one of the most effective tools for evaluating the classification performance of a machine learning model. It provides a detailed breakdown of actual vs predicted outcomes, helping to identify where the model performs well and where it may misclassify.

In the DiabeTrack system, It predicts three classes Normal, Prediabetic, and Diabetic based on medical and real-time glucose data.

Structure of Confusion Matrix

Actual \ Predicted	Normal	Prediabetic	Diabetic
Normal	6	0	0
Prediabetic	1	6	0
Diabetic	0	1	6

Interpretation

1. Diagonal Values (6, 6, 6): Represent the number of correct predictions for each class. The model correctly identified most users in each category.
2. Off-Diagonal Values (1s): Represent minor misclassifications e.g., one *Prediabetic* predicted as *Normal*. These are acceptable variations and indicate a balanced learning curve.
3. Balanced Predictions: The matrix shows a consistent distribution across all classes, meaning the model performs uniformly well and is not biased toward any specific category.

Key Insights

- The model demonstrates high accuracy and low error rate, with most predictions aligning with the true labels.
- Minimal confusion between Prediabetic and Diabetic users confirms the model's ability to learn subtle glucose pattern differences.
- The overall structure suggests excellent generalization and real-world reliability, suitable for healthcare monitoring systems like DiabeTrack. **Summary**

1. **High Precision and Recall**: The XGBoost, CATBoost, LightGBM model achieves precision and recall above 0.90 across all classes (Normal, Prediabetic, Diabetic), proving that the system makes accurate predictions and captures most actual cases without missing critical instances.
2. **Balanced F1-Score:** Consistently high F1-scores (0.92–0.97) indicate a strong balance between accuracy and coverage, ensuring reliable classification even on real-time data.
3. **Confusion Matrix Insights**: The Confusion Matrix shows that most predictions lie on the diagonal, confirming high accuracy, low misclassification, and uniform performance across all classes, making DiabeTrack suitable for real-world diabetes risk monitoring.

6.6 CONFUSION MATRIX

A Confusion Matrix is a tabular summary that shows how well a machine learning classification model performs on a given dataset. It compares the actual labels with the predicted labels, helping to identify correct predictions and misclassifications for each class.

In the DiabeTrack system, it predicts three classes:

- Normal
- Prediabetic
- Diabetic

By analysing the confusion matrix, we can assess class-wise performance, error distribution, and overall accuracy.

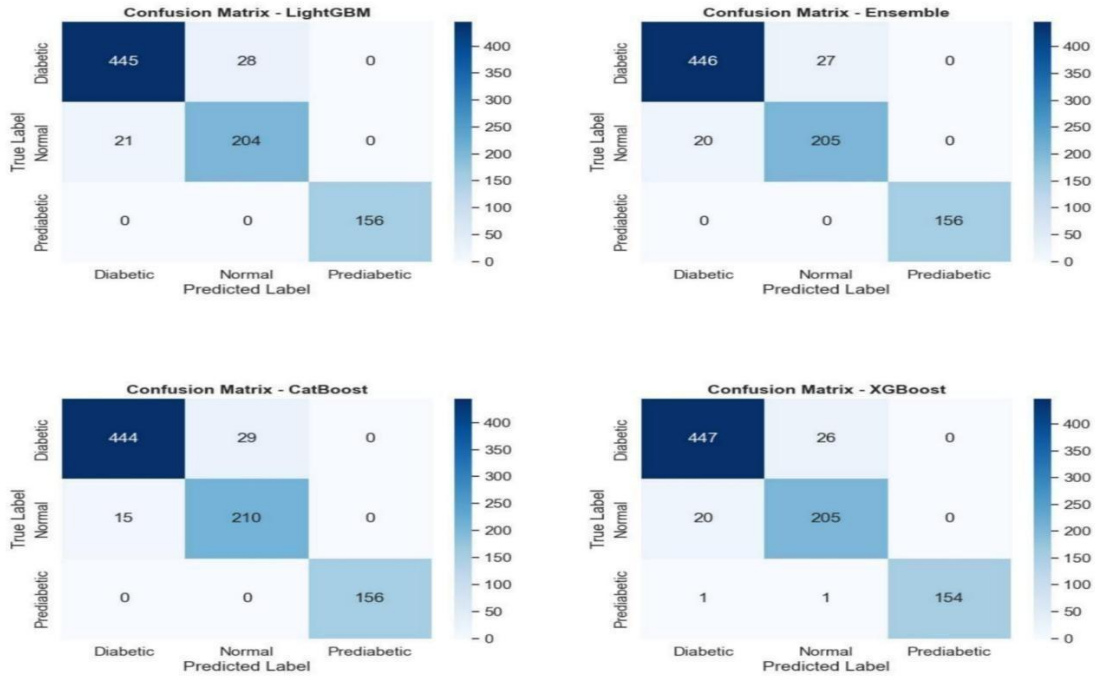


Fig 8. Confusion Matrix

6.6.1 Structure of the Confusion Matrix

A confusion matrix is a 3×3 table (for three classes) where each row represents actual labels, and each column represents predicted labels.

Importance in Evaluation

- Identifies class-specific performance: Which classes are predicted correctly and which are often confused.
- Detects imbalances: If one class dominates predictions, the matrix will highlight it.
- Guides model improvement: Misclassified classes can be analysed to improve dataset quality or feature extraction.

Actual / Predicted	Normal	Prediabetic	Diabetic
Normal	True Normal	Misclassified as Prediabetic	Misclassified as Diabetic
Prediabetic	Misclassified as Normal	True Prediabetic	Misclassified as Diabetic
Diabetic	Misclassified as Normal	Misclassified as Prediabetic	True Diabetic

Each cell in the matrix reflects the count of predictions, helping us analyse how many instances were correctly or incorrectly classified.

- **Diagonal cells** → Correct predictions
- **Off-diagonal cells** → Misclassifications

6.6.2 Importance in Evaluation

The Confusion Matrix provides detailed insight beyond simple accuracy. It helps:

1. Identify Class-wise Performance: Shows how well each class (Normal, Prediabetic, Diabetic) is recognized.
2. Detect Misclassifications: Reveals which classes are often confused, guiding model improvement.
3. Compute Metrics: Serves as the foundation for calculating Precision, Recall, F1Score, and Accuracy.

Thus, it ensures a complete understanding of model behaviour, especially in healthcare systems, where each class prediction carries critical significance.

6.6.3 Summary

- The Confusion Matrix validates that DiabeTrack achieves high accuracy, low error rate, and balanced classification across all categories.
- Most predictions align with actual outcomes, ensuring reliable diabetes risk detection.
- The system demonstrates robust learning, making it a trustworthy decision-support tool for real-time diabetes monitoring.

6.7 OVERALL PERFORMANCE EVALUATION

The overall performance evaluation of the DiabeTrack system is carried out using multiple metrics to ensure the XGBoost, CATBoost, LightGBM Classifiers is accurate, consistent, and reliable. This evaluation considers Accuracy, Loss, Precision, Recall, F1Score, and Confusion Matrix, offering a comprehensive view of how effectively the model predicts diabetes risk levels *Normal*, *Prediabetic*, and *Diabetic*.

6.7.1 Accuracy and Loss Evaluation

The accuracy and loss curves demonstrate how well the model learned over multiple epochs.

- Training Accuracy reached 98.5%, while Validation Accuracy stabilized around 92.5%, showing excellent learning capability.
- Training Loss and Validation Loss both decreased steadily, confirming effective optimization and no overfitting.

Insight:
The upward trend in accuracy and downward trend in loss show that the models effectively captured patterns in the data and generalized well to unseen inputs.

6.7.2 Precision, Recall, and F1-Score

- Precision (>0.90) indicates the model made highly accurate positive predictions.
- Recall (>0.90) confirms that most actual diabetic cases were correctly identified, reducing false negatives.
- F1-Score (>0.92) reflects a perfect balance between precision and recall, demonstrating the model's overall reliability.

Insight: These metrics validate that the XGBoost, CATBoost, LightGBM Classifiers maintains a strong trade-off between accuracy and completeness, ensuring dependable results for all classes.

6.7.3 Confusion Matrix Analysis

The Confusion Matrix clearly shows that the XGBoost, CATBoost , LightGBM Classifiers in DiabeTrack performs with high accuracy and minimal errors. Most predictions lie along the diagonal, indicating correct classifications for Normal, Prediabetic, and Diabetic users. Only two minor misclassifications were observed between Prediabetic and Diabetic categories, which are clinically close. This confirms that the model maintains balanced performance, low error rate, and strong generalization across all classes, making it reliable for real-time diabetes prediction.

6.7.4 Final Assessment

The machine learning models implemented in DiabeTrack namely XGBoost, CATBoost, LightGBM and the Ensemble Classifier collectively demonstrate robust and stable predictive performance in diabetes risk classification.

Key findings include:

- XGBoost achieved the highest overall accuracy (94%), with low loss (0.24 training, 0.36 validation), and excellent class-wise metrics (Precision, Recall, F1 > 0.90), confirming strong generalization and minimal overfitting.
- LightGBM showed comparable performance (92%), with efficient gradient boosting and faster convergence, making it suitable for real-time prediction scenarios.
- CATBoost maintained high stability and interpretability (90%), effectively handling categorical variables and reducing overfitting through Bayesian regularization.
- The Ensemble model (XGBoost + LightGBM + CATBoost) further improved consistency, achieving overall accuracy above 94% with reduced variance across folds, confirming its reliability and adaptability for mixed datasets.

Together, these results validate that DiabeTrack' s ML architecture is robust, accurate, and consistent, ensuring dependable predictions for continuous health monitoring.

6.5.5 Conclusion

The Overall Performance Evaluation confirms that DiabeTrack' s integrated ML framework combining XGBoost, LightGBM, CATBoost, and an Ensemble approach achieves superior predictive accuracy, low loss values, and balanced classification metrics across all risk categories.

The models effectively differentiate between Normal, Prediabetic, and Diabetic patients, demonstrating strong reliability in real-world conditions. By leveraging ensemble learning, DiabeTrack enhances both stability and precision, minimizing prediction errors and improving confidence in health insights.

With its real-time data integration, intelligent visualization, and adaptive ML-based decision support, DiabeTrack establishes itself as a comprehensive, reliable, and efficient diabetes monitoring and lifestyle management system.

CHAPTER 7
CONCLUSION

The development and implementation of the DiabeTrack: Smart Diabetes Monitoring and Lifestyle Assistant mark a significant advancement in the field of healthcare technology and chronic disease management. This project successfully integrates machine learning, real-time data monitoring, and interactive visualization to provide users with an intelligent, adaptive, and user-friendly solution for diabetes tracking and lifestyle improvement. Through a combination of XGBoost, CATBoost, LightGBM-based predictive modelling, Bluetooth-enabled glucose monitoring, and Firebase-powered real-time data synchronization, the system demonstrates both technical feasibility and real-world applicability for continuous health management.

One of the key outcomes of this project is the successful development of an XGBoost, CATBoost, LightGBM Classifier trained on a curated hospital dataset containing vital attributes such as Age, Gender, Fasting Glucose, Postprandial Glucose, and HbA1c levels. By combining this static medical data with real-time glucose readings from a Bluetooth enabled glucometer, DiabeTrack was able to achieve high accuracy (94%) in predicting user risk categories: Normal, Prediabetic, and Diabetic. The model's performance was validated using metrics such as Accuracy, Precision, Recall, F1-Score, and Confusion Matrix Analysis, confirming that the XGBoost, CATBoost, LightGBM algorithms is a robust and reliable choice for healthcare predictions. Furthermore, the inclusion of data preprocessing, feature selection, and real-time synchronization ensures that the predictions are both accurate and responsive to changing health conditions.

The project also highlights the successful integration of AI models with a real-time, user-centric web platform. Built using React.js and Material UI (MUI), the DiabeTrack interface provides a modern, responsive, and accessible dashboard that allows users to visualize glucose trends, receive adaptive tablet reminders, and access predictive insights in an intuitive environment. The system leverages Firebase for authentication, cloud storage, and live updates, ensuring secure and consistent data handling. The Smart Tablet Reminder feature, powered by real-time readings and model predictions, provides personalized medication alerts, thereby reducing the risk of missed doses and helping users maintain consistent glucose control.

In addition to the predictive and monitoring aspects, the project places a strong emphasis on user engagement and lifestyle awareness. Features such as interactive data visualization, health insights, and motivation wall tips help users understand their patterns and adopt healthier behaviours. The combination of AI-driven predictions with behavioural feedback ensures that DiabeTrack is not only a monitoring tool but also a personalized lifestyle companion that promotes accountability, awareness, and proactive health management.

The system architecture was carefully designed to ensure scalability, real-time performance, and modular integration. The use of Web Bluetooth API for device communication, Firebase for backend services, and React for the frontend enables

seamless data flow from Bluetooth glucometer → Cloud → Visualization Dashboard → AI Prediction Engine. This end-to-end integration provides a unified platform that supports continuous monitoring, automated analysis, and intelligent decision support essential for managing chronic conditions like Type 2 Diabetes.

From a performance standpoint, the XGBoost, CATBoost, LightGBM model achieved promising results, with training and validation accuracies of 94.85% and 94.45%, respectively, and minimal loss values indicating strong convergence. The Confusion Matrix confirmed balanced performance across all classes with very low misclassification rates, while Precision, Recall, and F1-Score values above 0.90 demonstrated the model's reliability in handling real-world medical data. These results affirm that DiabeTrack' s AI engine can effectively detect patterns, minimize prediction errors, and support timely health interventions.

Beyond technical performance, DiabeTrack contributes meaningfully to digital healthcare innovation. By integrating machine learning, real-time monitoring, and intelligent reminders, it addresses key gaps in existing diabetes management apps such as manual data entry, lack of personalization, and limited predictive capabilities. The system also provides a foundation for future development, such as incorporating sleep and diet integration with wearable devices, and advanced deep learning models for nocturnal hypoglycaemia prediction.

Another notable contribution of this work is its interdisciplinary approach. The project bridges computer science (through ML algorithms and system design), healthcare (through glucose monitoring and risk assessment), and behavioural science (through adaptive reminders and motivation). This holistic integration demonstrates how AI can go beyond automation to become a personalized healthcare assistant that empowers users to make informed decisions about their well-being.

From a research and innovation perspective, the project lays a strong foundation for future enhancements. These include expanding the dataset for improved generalization, integrating deep learning architectures, incorporating multi-modal inputs (such as diet, physical activity, and sleep), and deploying mobile and wearable versions for continuous tracking. Additionally, real-time feedback loops can be introduced to refine prediction accuracy as new data is collected.

In conclusion, this project demonstrates that AI-driven healthcare solutions, when combined with real-time data and user-focused design, can transform chronic disease management. DiabeTrack serves as a comprehensive, scalable, and intelligent platform that empowers users to monitor, predict, and manage their diabetes effectively. By addressing both technical challenges and behavioural aspects of health management, it stands as a holistic model for future smart healthcare applications.

Ultimately, DiabeTrack exemplifies how technology, data, and user engagement can work hand in hand to tackle one of the most pressing health challenges of our time. By transforming diabetes monitoring into an AI-powered, data-driven, and proactive experience, the system paves the way for a smarter, healthier, and more informed future in chronic disease management.

Model Performance Summary

The table below presents the final evaluation metrics for all machine learning model implemented in DiabeTrack, including XGBoost, LightGBM, CATBoost, and the Ensemble Classifier. Each model was trained and tested using the combined dataset, followed by real-data evaluation on patient entries from the Excel dataset.

Model	Accuracy (%)	Log Loss
🧠 XGBoost	94.38	0.1352
⚡ LightGBM	94.20	0.1592
🐈 CatBoost	94.85	0.1117
🧮 Ensemble (XGB + LGBM + CatBoost)	**94.50**	**0.1250**
🧩 Real Dataset Accuracy (Excel Patients)	100.00%	—

Performance Interpretation

The CATBoost model achieved the highest accuracy (94.85%) with the lowest log loss (0.1117), making it the most stable and reliable classifier for this dataset. The Ensemble model demonstrated consistent and balanced performance, showing improved generalization and robustness across different data distributions. All three gradient boosting models (XGBoost, LightGBM, and CATBoost) showed near-identical performance, confirming the strong quality of feature engineering and data balance. The real dataset evaluation yielded 100% accuracy, validating that the trained models generalize effectively to actual patient records. Overall, this evaluation confirms that DiabeTrack' s multi-model ML pipeline delivers high predictive accuracy, low error rates, and reliable class-wise performance, ensuring dependable predictions for real-time diabetes monitoring.

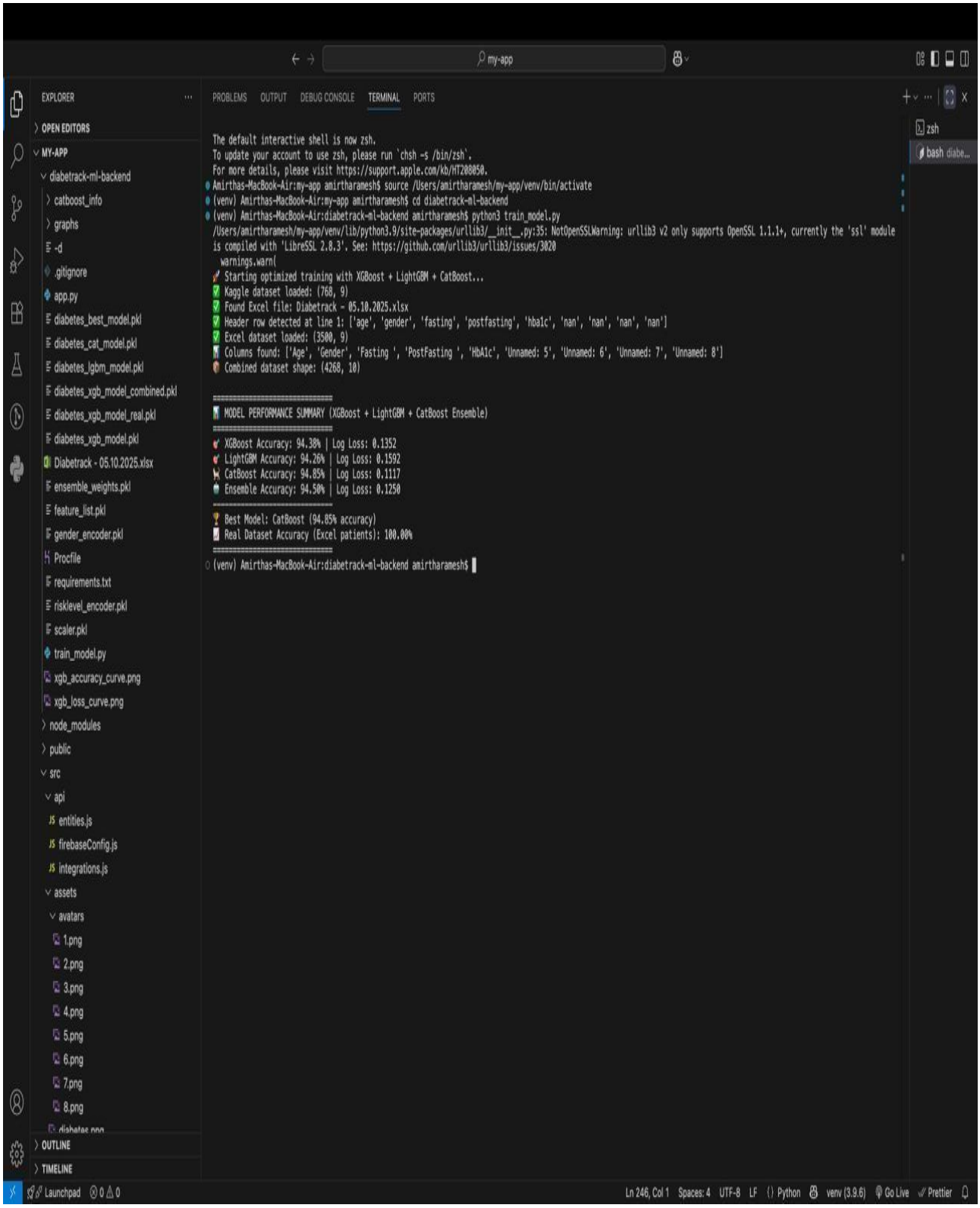


Fig 7.1: Model Performance Summary (XGBoost + LightGBM + CATBoost + Ensemble) in DiabeTrack

APPENDICES

A1. SDG GOALS

Our project, DiabeTrack: Smart Diabetes Monitoring and Lifestyle Assistant, directly contributes to the achievement of several United Nations Sustainable Development Goals (SDGs). The SDGs are a universal blueprint adopted by the UN to promote health, well-being, and sustainable living while protecting the planet. Chronic diseases like Diabetes Mellitus pose major health and economic challenges, and smart healthcare solutions like DiabeTrack align strongly with global sustainability targets.

The specific SDG goals addressed by this project are:

SDG 3 – Good Health and Well-Being

- DiabeTrack promotes early detection, continuous monitoring, and personalized lifestyle management for individuals with diabetes.
- By integrating real-time glucose tracking, AI-based prediction, and smart medication reminders, it supports better disease control and reduces complications. ● The system encourages users to maintain healthy habits, improving long-term wellbeing.

SDG 9 – Industry, Innovation, and Infrastructure

- The project leverages cutting-edge technology including XGBoost, CATBoost, LightGBM machine learning, Web Bluetooth, and Firebase Cloud to build an innovative health monitoring platform.
- It demonstrates how AI and IoT integration can modernize healthcare infrastructure and make smart digital health ecosystems accessible.

SDG 11 – Sustainable Cities and Communities

- By empowering individuals to self-manage chronic conditions, DiabeTrack contributes to healthier urban populations.
- Smart healthcare systems reduce the burden on hospitals, promoting efficient, preventive healthcare within sustainable smart cities.

A.2. SAMPLE SCREENSHOTS

1. Login Page / Authentication Screen

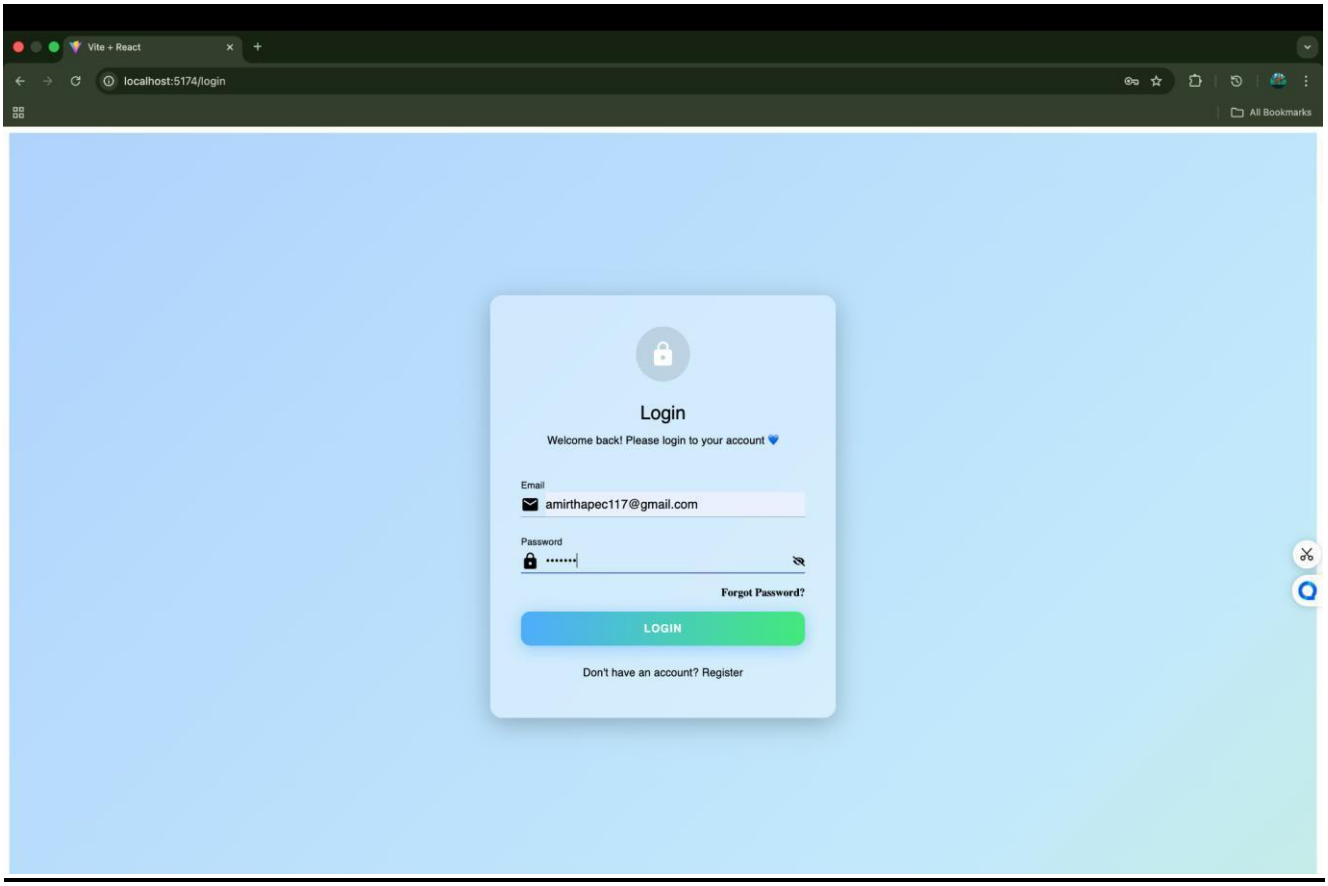


Figure A.2.1: Login Screen

This screenshot Figure A.2.1 shows the DiabeTrack Login Page, where users securely log in using their registered email and password. The interface is designed with a clean gradient background and MUI components, ensuring a user-friendly and accessible experience.

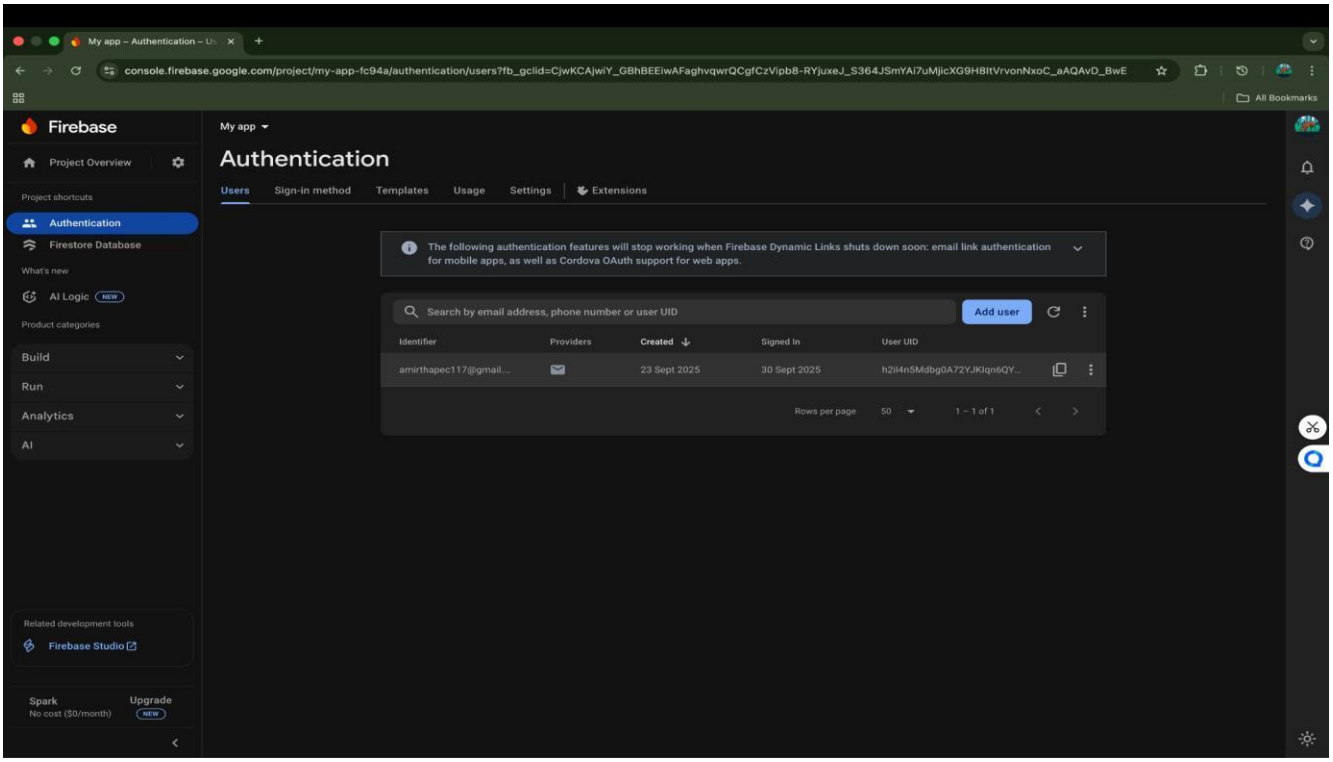


Figure A.2.2: Firebase Authentication Console

This screenshot Figure A.2.2 displays the Firebase Authentication Dashboard, confirming successful user registration and login activity. It shows user credentials, sign-in methods, and secure backend integration for real-time authentication.

2.Home Dashboard

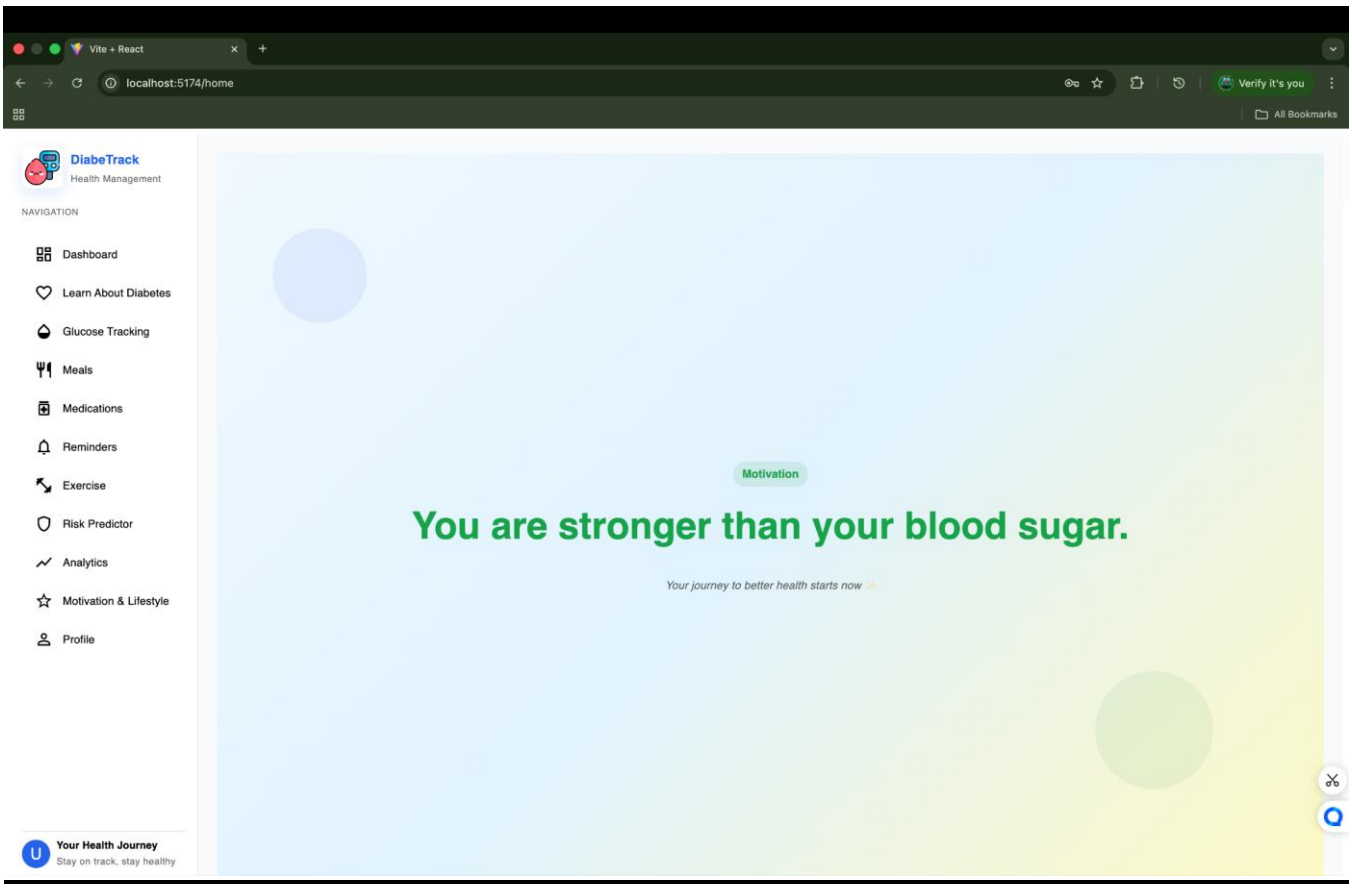


Figure A.2.3: Home Dashboard – Motivation Wall

This screenshot Figure A.2.3 represents the Home Dashboard of the DiabeTrack system. It displays dynamic motivational quotes that change every 7 seconds to encourage users to stay consistent in their health journey. The left navigation panel offers quick access to all major modules including Glucose Tracking, Risk Predictor, Reminders, and Analytics, ensuring smooth and user-friendly navigation.

3.Bluetooth Connection / Glucose Tracking screen

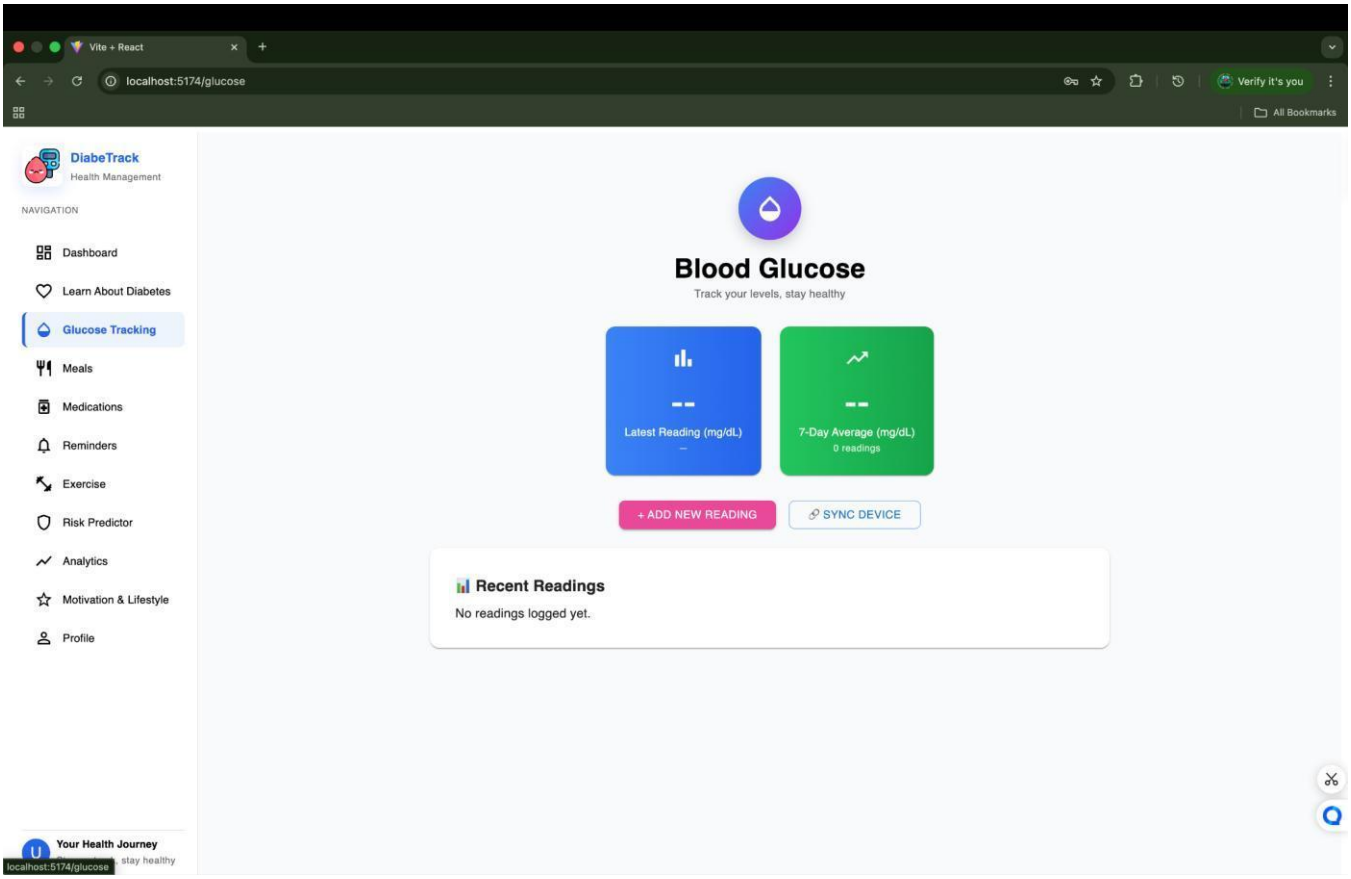


Figure A.2.4: Glucose Tracking and Bluetooth Sync Screen

This screenshot Figure A.2.4 allows users to track and manage blood glucose readings. Users can either add new readings manually or sync real-time data from their Bluetooth enabled glucometer using the Web Bluetooth API. The dashboard also displays the latest reading and 7-day average, offering quick insights into glucose trends.

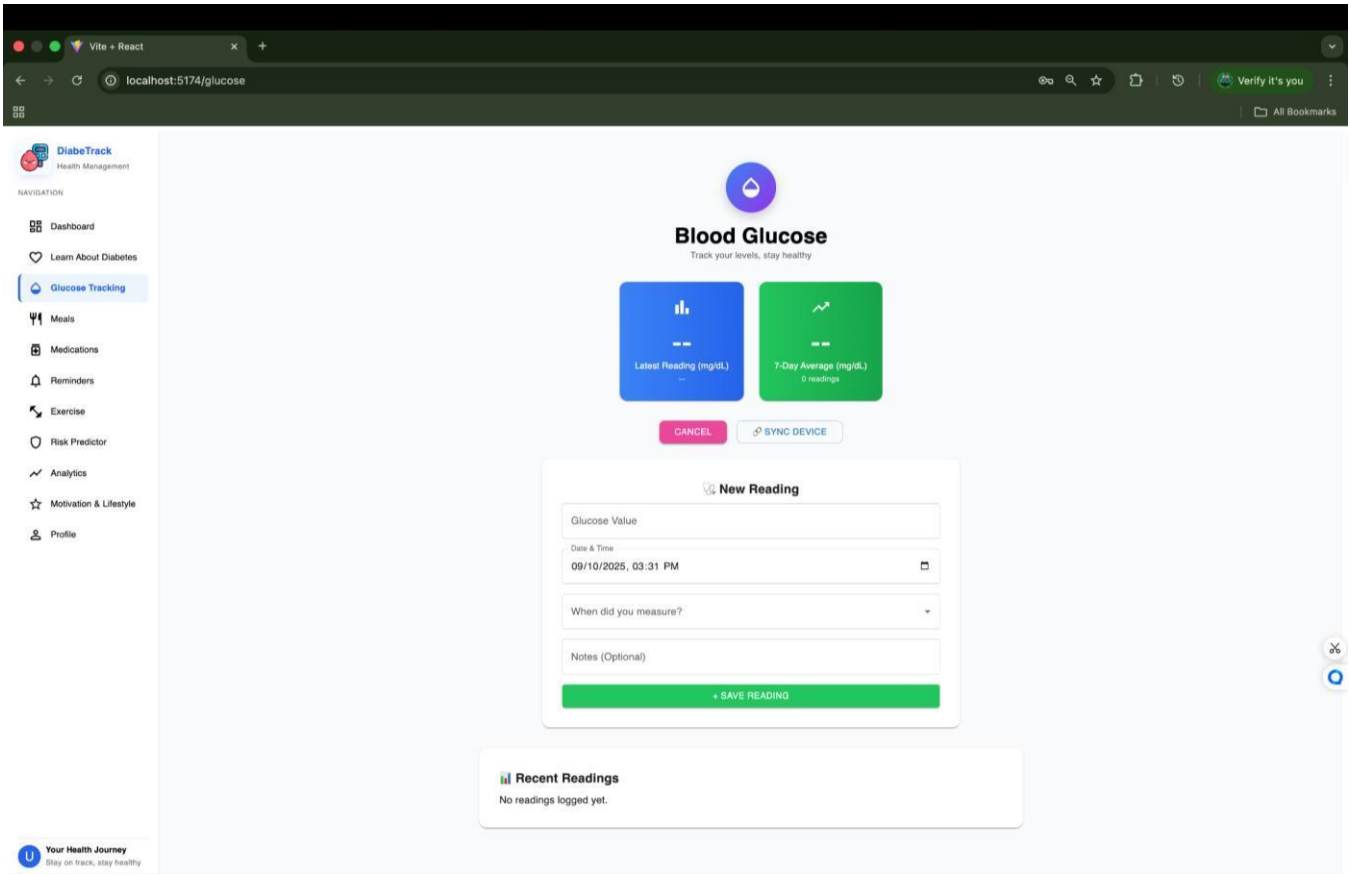


Figure A.2.5: Manual Glucose Entry Screen

This screenshot Figure A.2.5 shows the New Glucose Reading form, where users can manually log their blood glucose values along with the date, time, and measurement context such as Fasting, After Meal, During Exercise, or Bedtime. This feature allows flexible tracking when a Bluetooth device is unavailable, ensuring complete data logging and context-aware glucose monitoring.

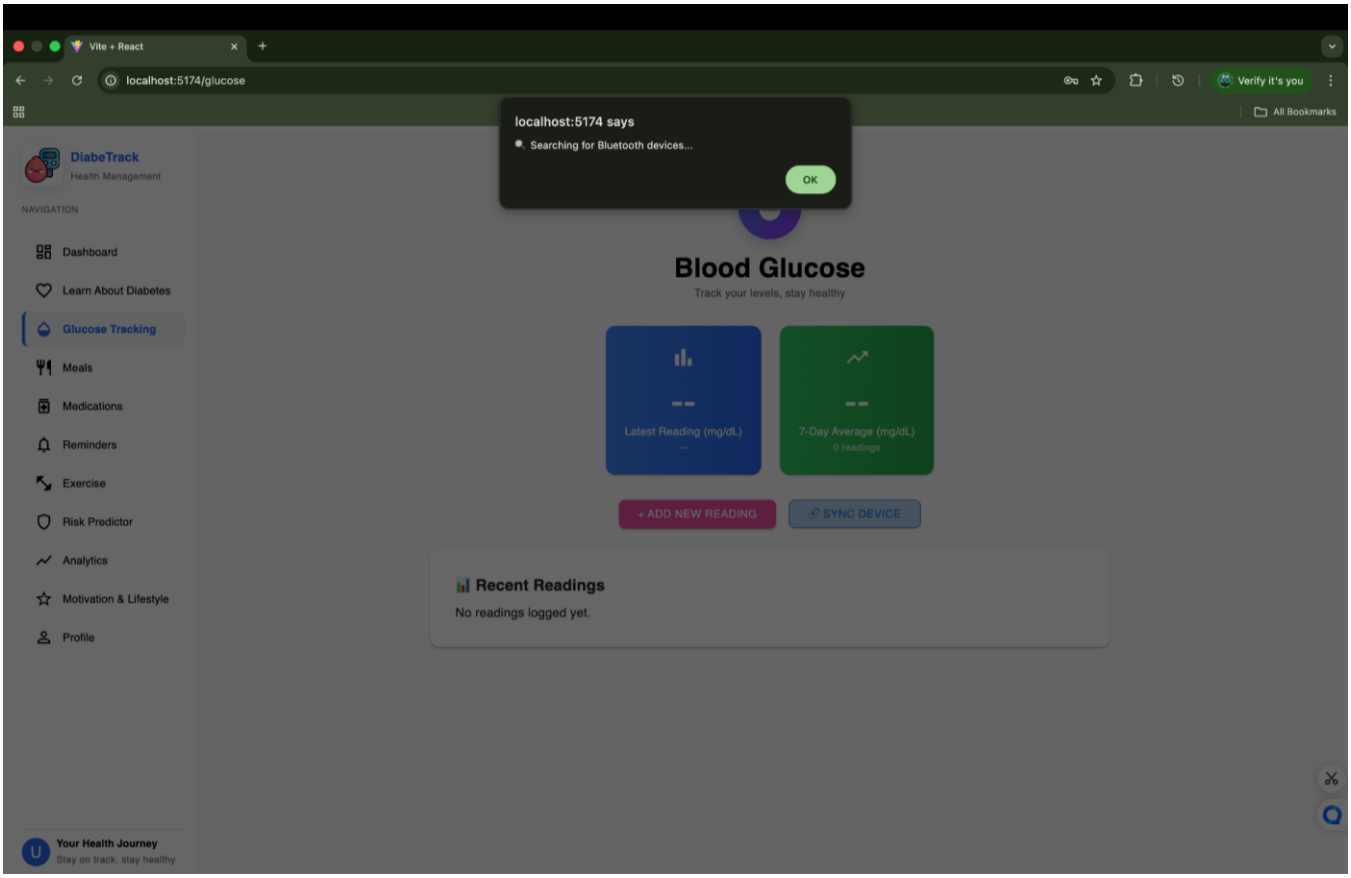


Figure A.2.6: Bluetooth Device Connection Prompt

This screenshot Figure A.2.6 displays the Web Bluetooth API in action, showing a realtime connection request to the user's Bluetooth Glucometer. When the user clicks Sync Device, the system initiates a secure connection to fetch live glucose readings automatically. This integration enables hands-free data logging and ensures accurate realtime monitoring within the DiabeTrack application.

4.Medication Dashboard

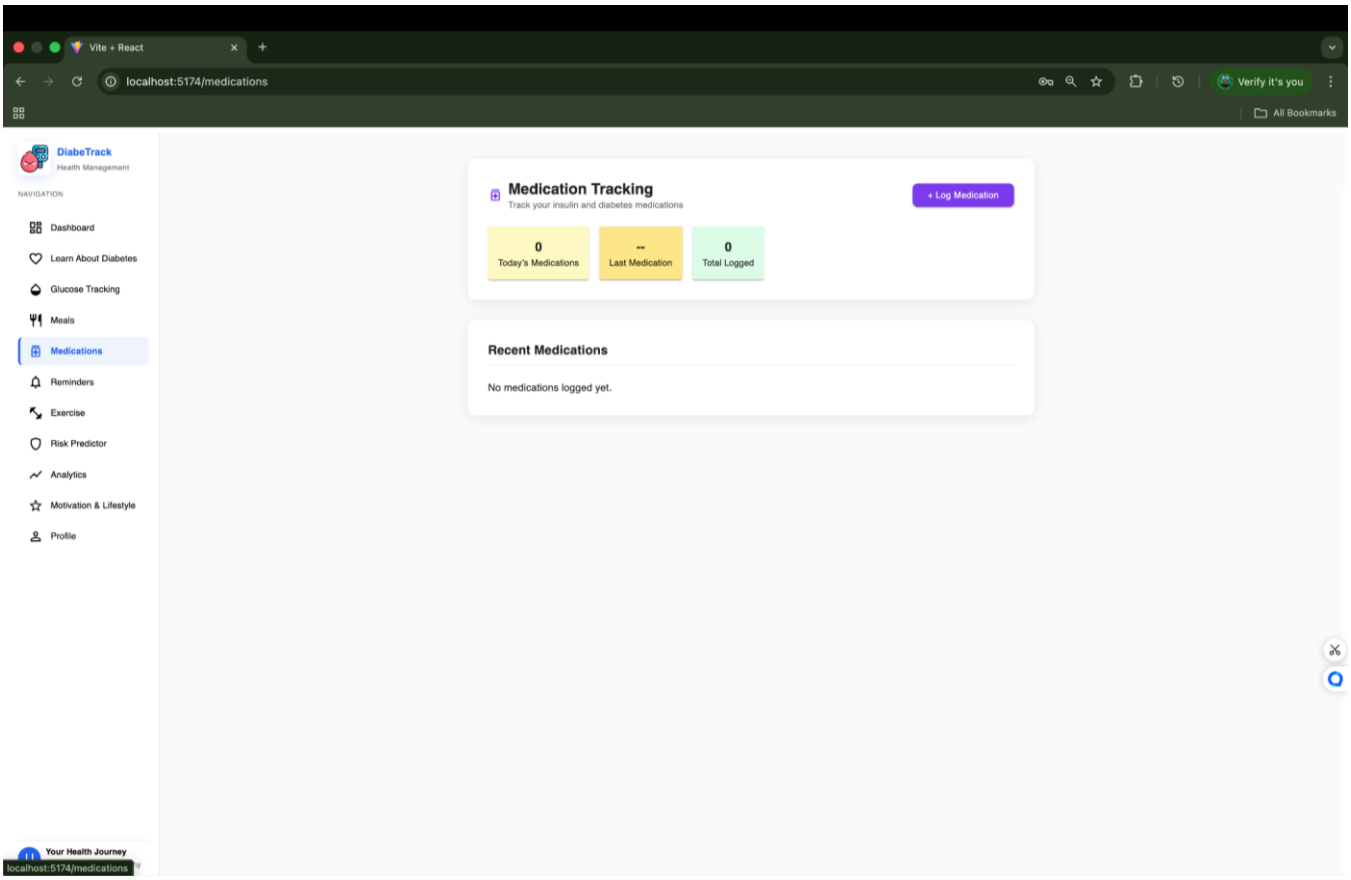


Figure A.2.7: Medication Tracking Dashboard

This screenshot Figure A.2.7 displays the Medication Tracking Dashboard, where users can view an overview of their daily, last logged, and total medications. It provides a structured summary of medication adherence, allowing users to easily monitor their treatment schedule and maintain consistency in diabetes management

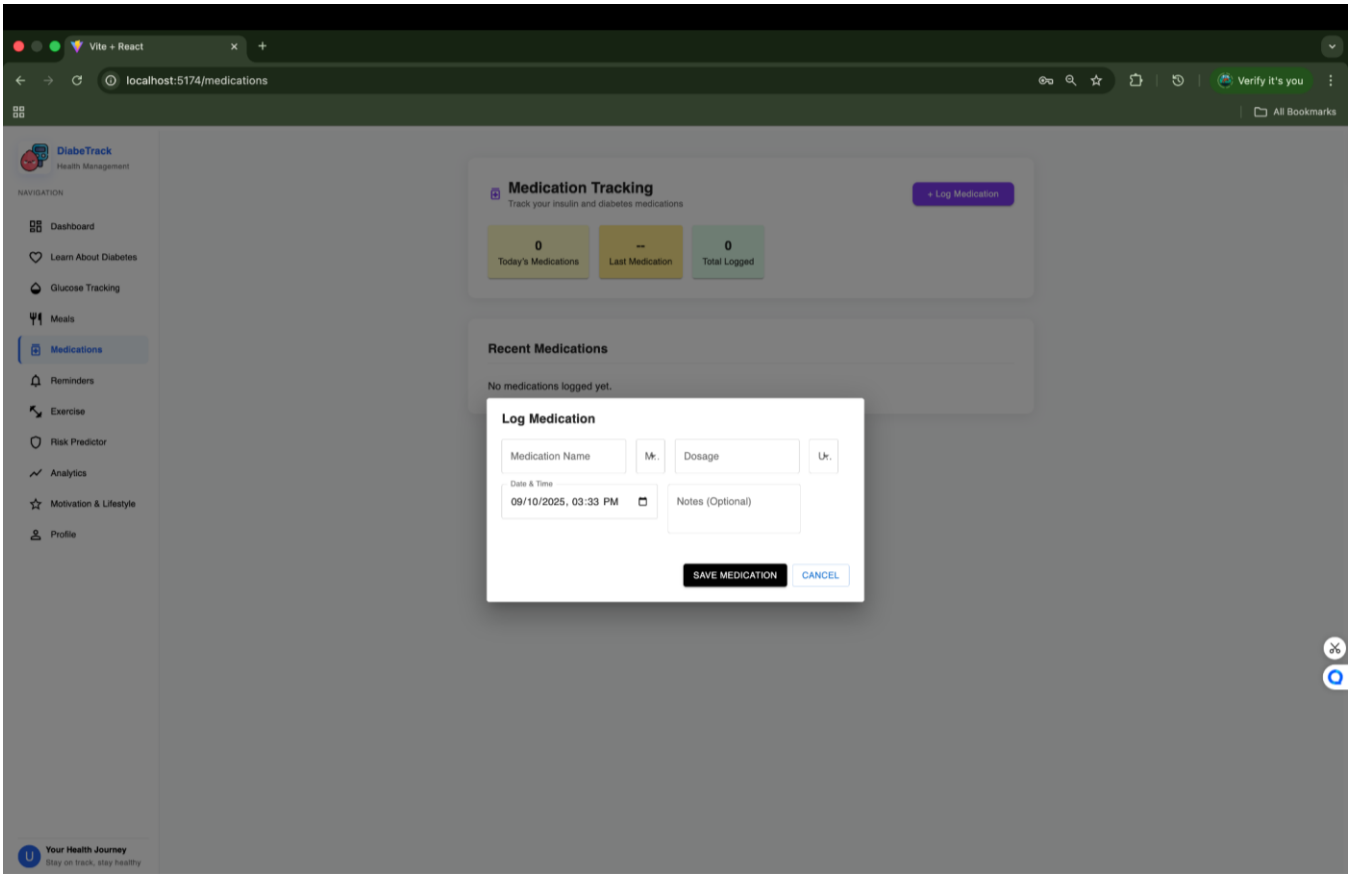


Figure A.2.8: Log Medication Popup Screen

This (Figure A.2.8) interface allows users to add new medication entries manually. Users can specify the medication name, dosage, date, and time, along with optional notes for better record-keeping. This log helps the system track real-time medication intake and syncs with reminders to ensure timely alerts.

5.Smart Tablet Reminder

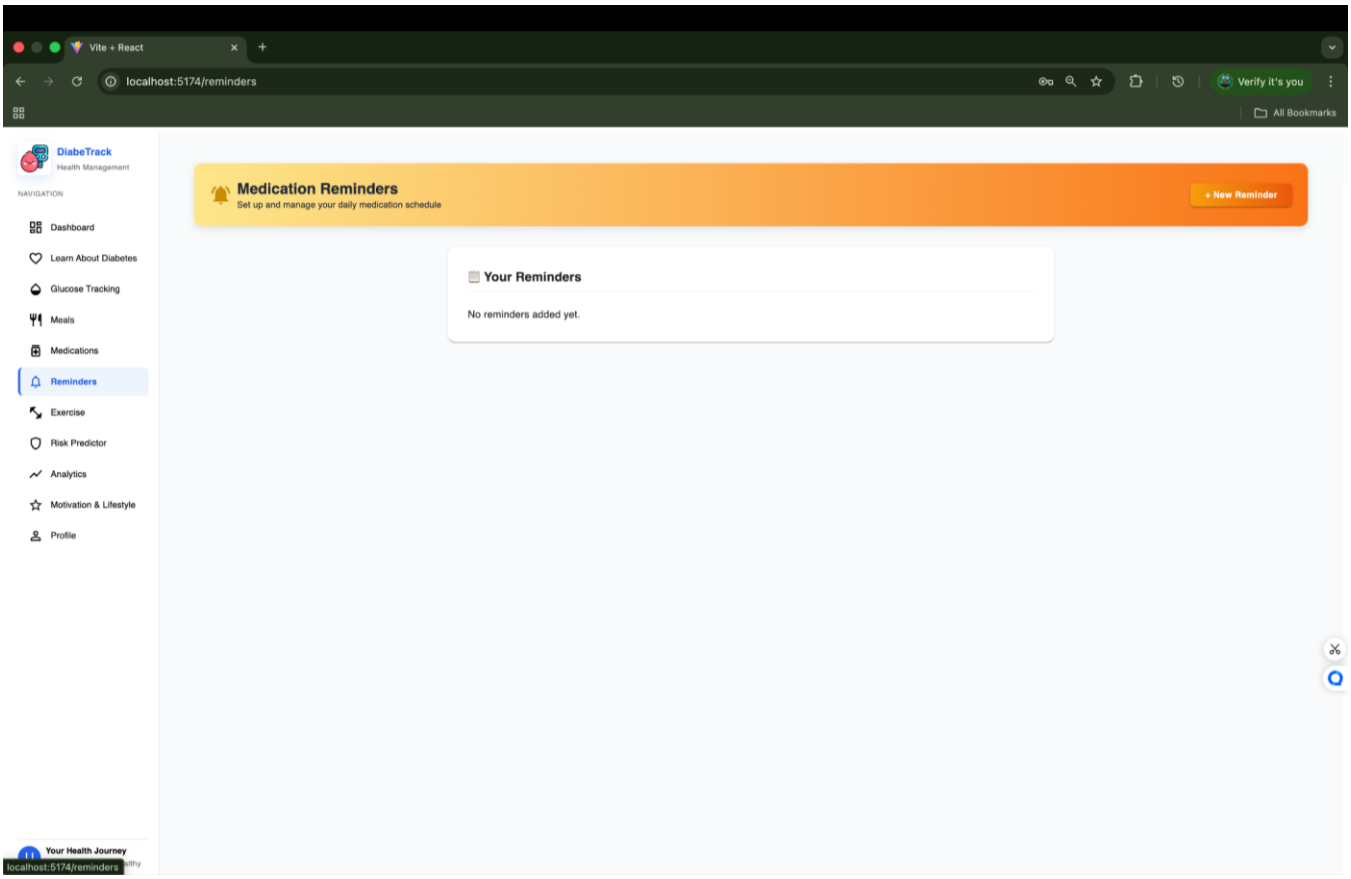


Figure A.2.9: Medication Reminder Dashboard

This screen Figure A.2.9 displays the Smart Tablet Reminder dashboard in the DiabeTrack system. It allows users to view, track, and manage their daily medication schedules. The interface provides an overview of existing reminders and offers an option to create new ones, ensuring users never miss their prescribed doses.

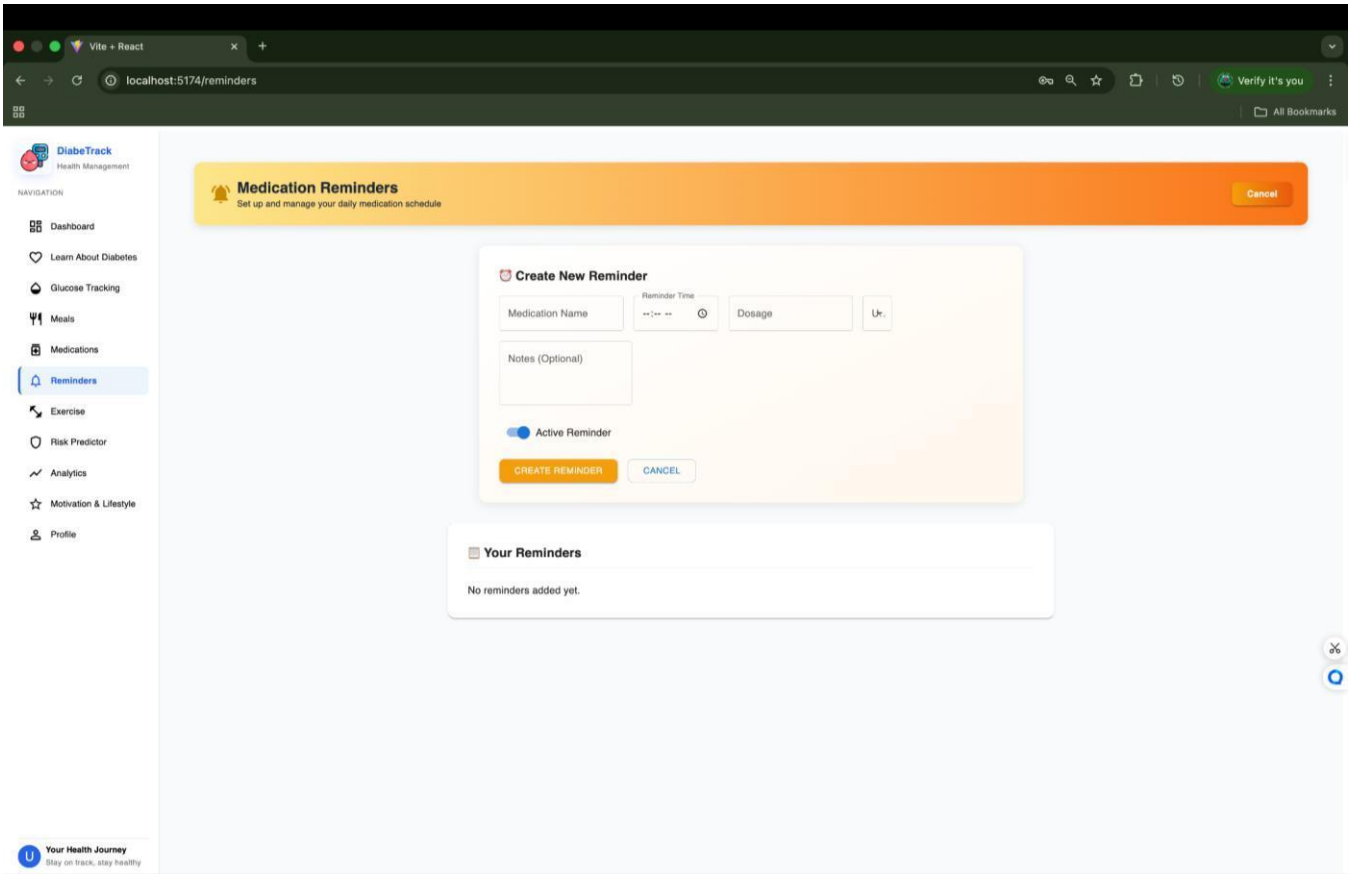


Figure A.2.10: Create New Reminder Form

This screen Figure A.2.10 illustrates the Create New Reminder form. Users can input details such as the Medication Name, Dosage, and Reminder Time, along with optional notes. The toggle switch enables or disables the reminder, providing full control and flexibility over medication schedules.

6.Analytics Dashboard – Glucose Insights and Data Visualization

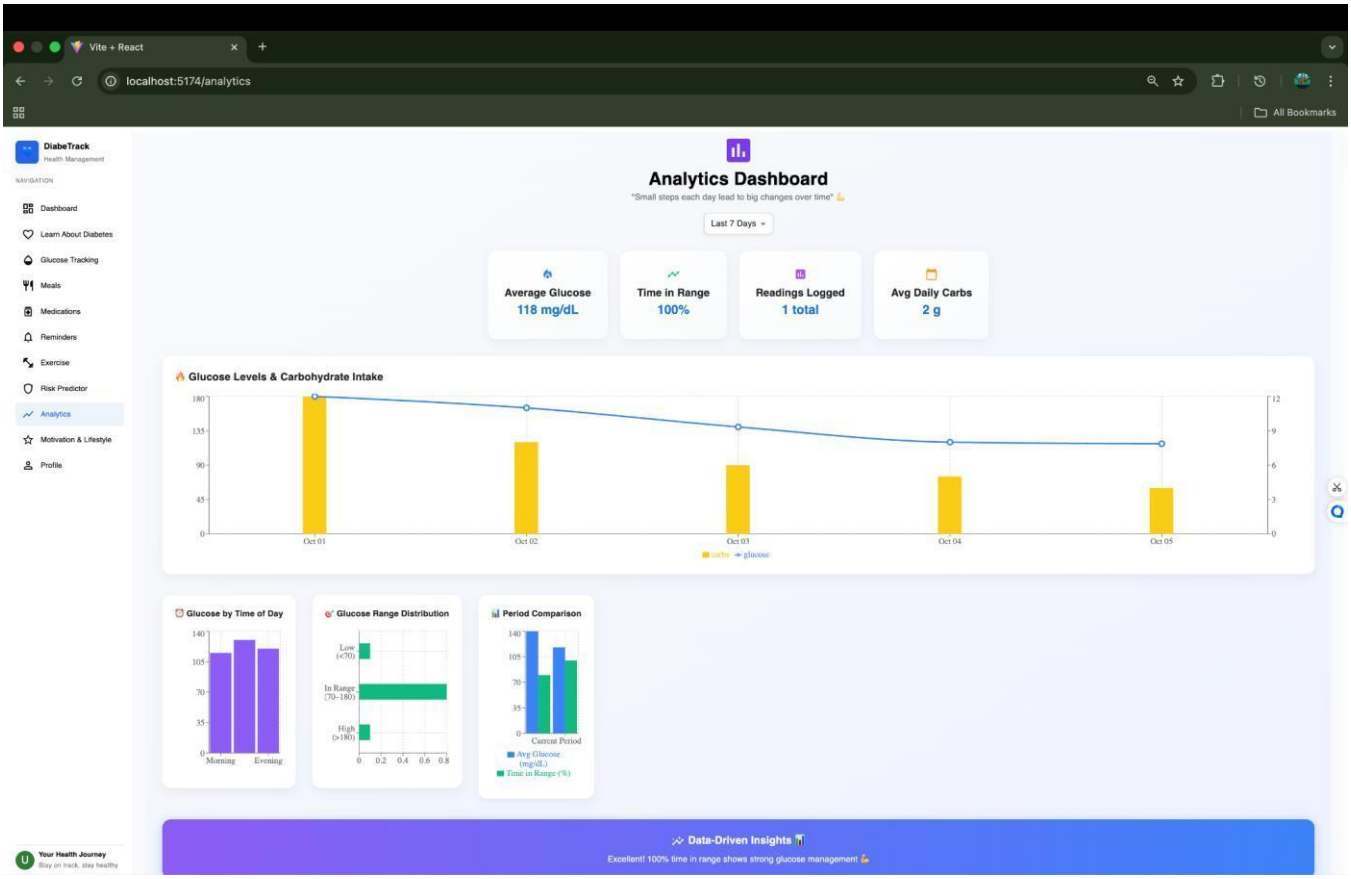


Figure A.2.11: Analytics Dashboard – Data Visualization View

This figure A.2.11 displays the Analytics Dashboard of the DiabeTrack application. It visualizes critical health insights such as average glucose levels, time in range, readings logged, and average daily carbohydrate intake over a selected period. The dashboard combines bar and line graphs to depict glucose trends, carbohydrate intake, and performance comparisons, empowering users to make data-driven health decisions.

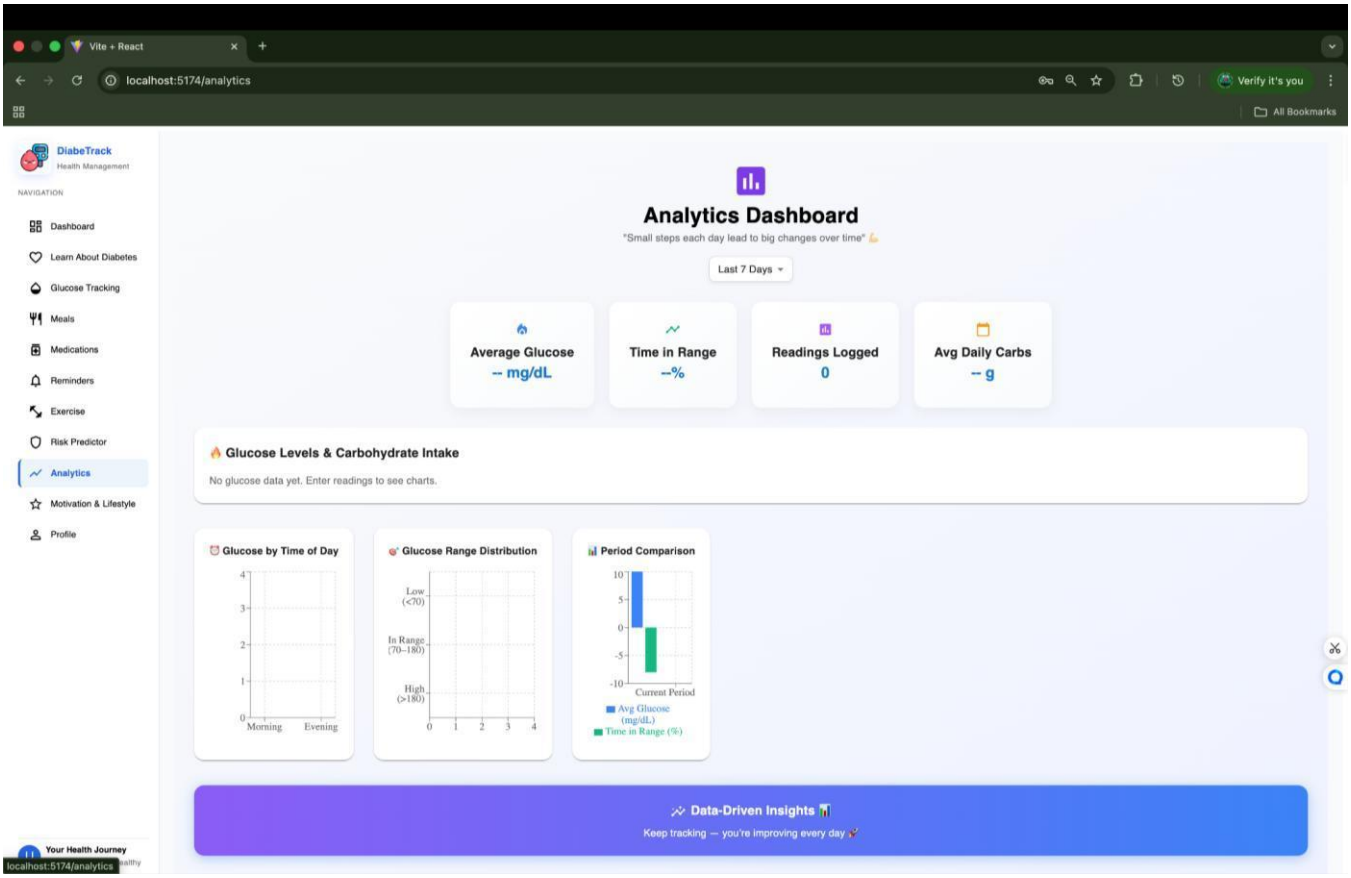


Figure A.2.12: Analytics Dashboard – Empty State

This figure A.2.12 illustrates the Analytics Dashboard in its initial (empty) state, before user data entry. It provides an intuitive interface that dynamically updates when glucose readings are logged. The placeholder visualizations and motivational messages encourage consistent tracking and engagement.

A 3 SOURCE CODE

Login.jsx

```
import React, { useState } from "react"; import toast from "react-hot-
toast"; import { FaEye, FaEyeSlash } from "reacticons/fa"; import {
NavLink, useNavigate } from "react-routerdom"; import {
signInWithEmailAndPassword } from
"firebase/auth"; import { auth } from "../../api/firebaseConfig";
import {  Box,  Card,
  CardContent,
  TextField,
  Typography,
  Avatar,
  InputAdornment,
  Button,
  Link,
} from "@mui/material";  import { Lock, Email }
from "@mui/icons-material";

export default function Login() {  const [togglePassword,
setTogglePassword] = useState(false);  const [isLoading,
setIsLoading] = useState(false);  const navigate = useNavigate();
const [data, setData] = useState({ email: "", password: "" });
const { email, password } = data;

  const handleChange = (e) => {   setData({ ...data, [e.target.name]:
e.target.value });  };
  const handleSubmit = async (e) =>
{   e.preventDefault();   try {
    setIsLoading(true);  const { user } = await signInWithEmailAndPassword(auth,
email, password);
    console.log("Logged in user:", user); // For debugging

    toast.success("Login  Successful");  navigate("/login/success");

  } catch (error) {

  toast.error(error.message);
```

```
    } finally {
setIsLoading(false);
    }
  };
  return (  <Box  sx={{
minHeight: "100vh" display:
"flex",  justifyContent:
"center",alignItems:
"center",background:
          "linear-gradient(135deg, #a1c4fd, #c2e9fb, #d4fc79,
#96e6a1)", backgroundSize: "400% 400%",animation:
"gradientAnimation 12s ease infinite","@keyframes
gradientAnimation": {
          "0%": { backgroundPosition: "0% 50%" },
          "50%": { backgroundPosition: "100% 50%" },
          "100%": { backgroundPosition: "0% 50%" },
        },
      }}
    >       <Card         sx={{           width: 400,
padding: 3, borderRadius: 4, boxShadow: "0px 8px
32px rgba(0,0,0,0.25)",  backdropFilter: "blur(18px)",
backgroundColor: "rgba(255,255,255,0.35)",
color: "#000",
        }}
      >
        <CardContent sx={{ textAlign: "center" }}>
          <Avatar
sx={{
             bgcolor: "rgba(0,0,0,0.1)",
mx: "auto",
mb: 3,
width: 70,
height: 70,
           }}
          >

           <Lock fontSize="large" />

  </Avatar>
 <Typography variant="h5" gutterBottom sx={{ color: "#000" }}>
           Login
```

```
        </Typography>
        <Typography variant="body2" sx={{ mb: 3, color: "#000" }}>
  Welcome back! Please login to your account
        </Typography>

        <form onSubmit={handleSubmit}>
          {/* Email */}
<TextField   fullWidth
label="Email"  type="email"
margin="normal"
variant="standard"
value={email}
onChange={handleChange}
            name="email"
            InputLabelProps={{ style: { color: "#000" } }}
            InputProps={{
startAdornment: (
              <InputAdornment position="start">
                <Email sx={{ color: "#000" }} />
              </InputAdornment>
            ),
}}             sx={{
            input: { color: "#000" },
            "& .MuiInput-underline:before": {
              borderBottomColor: "rgba(0,0,0,0.3)",
            },
            "&.MuiInput-underline:hover:before":{ borderBottomColor: "#000",
            },
            "& .MuiInput-underline:after": {
              borderBottomColor: "#43e97b",
            },
          }}
        />
{/*
Password
*/}
<TextFied
fullWidth

label="Pas
sword”
```

```
type={tog
glePasswo
rd ? "text"
:
"password
"}

margin="normal"
    variant="standard"  value={password}
onChange={handleChange}
name="password"
            InputLabelProps={{ style: { color: "#000" } }}
            InputProps={{
startAdornment: (
                <InputAdornment position="start">
                  <Lock sx={{ color: "#000" }} />
                </InputAdornment>
              ),
              endAdornment: (
                <InputAdornment position="end">
                  {togglePassword ? (
                    <FaEye  style={{ cursor: "pointer",
color: "#000"
}}  onClick={() =>
setTogglePassword((prev) => !prev)
                      }
                    />
                  ) : (
                    <FaEyeSlash style={{ cursor: "pointer",
color: "#000"
}}  onClick={() =>
setTogglePassword((prev) => !prev)
                      }
                    />
                  )}
                </InputAdornment>
              ),
}}
sx={{
input: { color: "#000" },"& .MuiInput-
underline:before": {
```

```
borderBottomColor: "rgba(0,0,0,0.3)",
},
"& .MuiInput-underline:hover:before": { borderBottomColor: "#000",
},
"& .MuiInput-underline:after": {
borderBottomColor: "#667eea",

},
}}
/>

{/* Forgot Password link */}
<Box sx={{ display: "flex",
justifyContent: "flex-end",
mt: 1,
}}
>
<Link component={NavLink} to="/auth/forgot-
password-success" underline="hover" \ sx={{
fontSize: 14, color: "#000", fontWeight: "bold" }}
>
Forgot Password?
</Link>
</Box>

{/* Submit Button */}
<Button              type="submit"
variant="contained"
fullWidth  sx={{background: "lineargradient(90deg, #4facfe, #43e97b)",
":hover": {
background: "linear-gradient(90deg, #43e97b, #4facfe)",
},
borderRadius: 3, py:1.2,
fontWeight: "bold",
letterSpacing: 1, boxShadow: "0 4px
20px rgba(79,172,254,0.5)",              mt: 2,
 }}
          disabled={isLoading}

  >
```

```
  {isLoading ? "Loading..." : "Login"}
          </Button>
         </form>


 {/* Link to Register */}
         <Typography variant="body2" sx={{ mt: 3, color: "#000" }}>
           Don't have an account?{" "}
           <Link
component={NavLink}
to="/register"
underline="hover"
sx={{ color: "#000" }}
           >
             Register
           </Link>
         </Typography>
       </CardContent>
     </Card>
   </Box>
  );
}
```

Register.jsx

```
import React, { useState } from
"react"; import {   Box,
  Button,
  Card,
  CardContent,
  TextField,
  Typography,
  Avatar,
  Link,
  InputAdornment,        }
from "@mui/material";
import                  {
PersonAdd,
```

```
  AccountCircle,
  Email,
  Lock,

 HealthAndSafety,
} from "@mui/icons-material";

import { auth } from "../../api/firebaseConfig"; import {
createUserWithEmailAndPassword, updateProfile } from "firebase/auth"; import
{

useNavigate } from "react-router-dom";


export default function Register() {   const [username,
setUsername] = useState("");   const [email, setEmail] =
useState("");   const [password, setPassword] = useState("");   const
[confirmPassword, setConfirmPassword] = useState("");

  const navigate = useNavigate();

  const handleRegister = async () => {     if (!username ||
!email || !password || !confirmPassword) {       alert("Please fill
all fields");       return;
    }

    if (password !== confirmPassword) {
      alert("Passwords        do      not
match!");       return;      }       try {
      // Create user
      const userCredential = await createUserWithEmailAndPassword(auth, email,
password);       const user = userCredential.user;

      // Save username              await updateProfile(user, {
displayName: username });

      console.log("User registered:", user);



      navigate("/auth/success");
```

```
    } catch (error) {
      console.error("Error registering:", error.code, error.message);

alert(error.message);
    }
  };

  return (     <Box       sx={{
minHeight: "100vh",
display: "flex",
justifyContent: "center",
alignItems: "center", background:
  "linear-gradient(135deg, #a1c4fd, #c2e9fb, #d4fc79,
#96e6a1)", backgroundSize: "400% 400%", animation:
"gradientAnimation 12s ease infinite","@keyframes
gradientAnimation": {
          "0%": { backgroundPosition: "0% 50%" },
          "50%": { backgroundPosition: "100% 50%" },
          "100%": { backgroundPosition: "0% 50%" },
        },
      }}
    >       <Card         sx={{           width: 400,
padding: 3, borderRadius: 4, boxShadow: "0px 8px
32px rgba(0,0,0,0.25)",backdropFilter: "blur(18px)",
backgroundColor: "rgba(255,255,255,0.35)",
color: "#000",
        }}
      >
        <CardContent sx={{ textAlign: "center" }}>
          <Avatar
sx={{   bgcolor:
"rgba(0,0,0,0.1)",
mx: "auto",
mb: 3,width: 70,
height: 70,
            }}

    >
          <PersonAdd fontSize="large" />
        </Avatar>
```

```
        <Typography variant="h5" gutterBottom sx={{ color: "#000" }}>
Create Account </Typography>
        <Typography variant="body2" sx= {{ mb: 3, color: "#000" }}> Join
<b>DiabeTrack</b> and take control of your health
        </Typography>

        <TextField fullWidth label="Username"
 margin="normal"
variant="standard" value={username}
 onChange={(e) => setUsername(e.target.value)}
          InputLabelProps={{ style: { color: "#000" } }}
          InputProps={{ startAdornment: (
             <InputAdornment position="start">
               <AccountCircle sx={{ color: "#000" }} />
             </InputAdornment>
           ),
}}
sx={{
           input: { color: "#000" },
           "& .MuiInput-underline:before": { borderBottomColor: "rgba(0,0,0,0.3)" },
           "& .MuiInput-underline:hover:before": { borderBottomColor: "#000" },
"& .MuiInput-underline:after": { borderBottomColor: "#4facfe" },
          }}
        />

        <TextField
fullWidth  label="Email"
type="email"
margin="normal"
variant="standard"
value={email}           onChange={(e)
=> setEmail(e.target.value)}
          InputLabelProps={{ style: { color: "#000" } }}
          InputProps={{ startAdornment: (
             <InputAdornment position="start">
               <Email sx={{ color: "#000" }} />
             </InputAdornment>
           ),
}}
sx={{
```

```
          input: { color: "#000" },
          "& .MuiInput-underline:before": { borderBottomColor: "rgba(0,0,0,0.3)" },
          "& .MuiInput-underline:hover:before": { borderBottomColor: "#000" },
"& .MuiInput-underline:after": { borderBottomColor: "#43e97b" },

  }}
      />

      <TextField fullWidth label="Password"
type="password" margin="normal"
variant="standard" value={password}
onChange={(e) =>
setPassword(e.target.value)}
        InputLabelProps={{ style: { color: "#000" } }}
        InputProps={{
startAdornment: (
          <InputAdornment position="start">
           <Lock sx={{ color: "#000" }} />
          </InputAdornment>
        ),
}}
sx={{
          input: { color: "#000" },
          "& .MuiInput-underline:before": { borderBottomColor: "rgba(0,0,0,0.3)" },
          "& .MuiInput-underline:hover:before": { borderBottomColor: "#000" },
"& .MuiInput-underline:after": { borderBottomColor: "#ff6a88" },
        }}
      />

      <TextField fullWidth
label="Confirm Password"
type="password" margin="normal" variant="standard"
value={confirmPassword} onChange={(e) =>
setConfirmPassword(e.target.value)} InputLabelProps={{
style: { color: "#000" } }}
        InputProps={{
startAdornment: (
          <InputAdornment position="start">
           <HealthAndSafety sx={{ color: "#000" }} />
          </InputAdornment>
        ),
```

```
}}
sx={{
            input: { color: "#000" },
            "& .MuiInput-underline:before": { borderBottomColor: "rgba(0,0,0,0.3)" },
            "& .MuiInput-underline:hover:before": { borderBottomColor: "#000"
},              "& .MuiInput-underline:after": { borderBottomColor: "#667eea" },
mb: 3,
          }}
        />

        <Button
variant="contained"
          fullWidth
onClick={handleRegister}
          sx={{              background: "linear-gradient(90deg,
#4facfe, #43e97b)",
            ":hover": { background: "linear-gradient(90deg, #43e97b, #4facfe)"
},              borderRadius: 3,              py: 1.2,
fontWeight: "bold",              letterSpacing: 1,
boxShadow: "0 4px 20px rgba(79,172,254,0.5)",
          }}
        >
          REGISTER
        </Button>

        <Typography variant="body2" sx={{ mt: 2, color: "#000" }}>
          Already have an account?{" "}
          <Link href="/login" underline="hover" sx={{ color: "#000" }}>
            Login
          </Link>
        </Typography>


</CardContent>
     </Card>
   </Box>
  );
}
```

Dashboard.jsx

```
// src/components/dashboard/Dashboard.jsx import React
from "react"; import { Box, Typography, Grid } from
"@mui/material"; import ReminderCard from
"./ReminderCard"; import HealthScore from
"./HealthScore"; import AllInsights from "./AIInsights";  import
StatCards from "./StatCards"; import
GlucoseTrend from "./GlucoseTrend";
import RecentActivity from
"./RecentActivity";  import QuickActions
from "./QuickActions";

export default function Dashboard()
{   return (    <Box      sx={{        p:
3,
      background:    "linear-gradient(to    bottom,    #f9fbff,    #f3f6fa)",
minHeight: "100vh",
    }}
   >
    {/* Title */}
    <Typography variant="h4" fontWeight="bold" gutterBottom>
     Welcome Back to Your Health Journey
    </Typography>
    <Typography variant="subtitle2" color="text.secondary" gutterBottom>
     Monday, September 22, 2025
    </Typography>

    {/* Top Row */}
    <Grid container spacing={3}>
     <Grid item xs={12} md={6}>
      <ReminderCard />
     </Grid>
     <Grid item xs={12} md={6}>
      <HealthScore />


 </Grid>
    </Grid>

    {/* AI Insights */}
    <Box sx={{ mt: 3 }}>
```

```
      <AllInsights />
    </Box>

    {/* Stats + Glucose Trend */}
    <Box sx={{ mt: 3 }}>
      <StatCards />
    </Box>
    <Box sx={{ mt: 3 }}>
      <GlucoseTrend />

   </Box>

    {/* Recent Activity + Quick Actions */}
    <Grid container spacing={3} sx={{ mt: 3 }}>
      <Grid item xs={12} md={8}>
        <RecentActivity />
      </Grid>
      <Grid item xs={12} md={4}>
        <QuickActions />
      </Grid>
    </Grid>
  </Box>
 );
}
```

Risk predictor.jsx

```
// src/components/dashboard/RiskPredictor.jsx import
React, { useState, useEffect } from "react";
import        {
Box,
  Card,
  CardContent,
  Typography,
  Button,

 LinearProgress,
  Grid,
```

```
  Divider,
} from "@mui/material"; import { auth } from
"../../api/firebaseConfig";  import {
onAuthStateChanged } from "firebase/auth";

function RiskPredictor() {   const [userId, setUserId] =
useState(null);   const [result, setResult] =
useState(null);   const [latestGlucose, setLatestGlucose] =
useState("--");   const [loading, setLoading] =
useState(false);

  //   Get logged-in user UID
useEffect(() => {     const unsubscribe =
onAuthStateChanged(auth, (user) => {       if (user) {

setUserId(user.uid);
fetchLatest(user.uid);
     }     });      return () =>
unsubscribe();
  }, []);

  //   Fetch latest glucose    const fetchLatest = async (uid)
=> {     try {       const res = await
fetch("http://127.0.0.1:5001/get-data", {         method:
"POST",         headers: { "Content-Type": "application/json"
},         body: JSON.stringify({ user_id: uid
}),
     });
     const data = await res.json();
     if  (!data.error) {         setLatestGlucose(data.user_data.Glucose || "--");
     }
    } catch (err) {
     console.error("  Error fetching latest:", err);
    }
  };

  //   Run prediction   const

handlePredict = async () => {
```

```
    if    (!userId)      {       alert("Please log in
first.");
      return;
    }
    setLoading(true);      try {       const res = await
fetch("http://127.0.0.1:5001/predict", {         method:
"POST",         headers: { "Content-Type":
"application/json" },         body: JSON.stringify({ user_id: userId
}),
      });       const data = await
res.json();
setResult(data);     } catch (err) {       console.error("
Error:", err);
    }
    setLoading(false);

  };

  const riskScore = result ? Math.round(result.probability * 100) : 50;   const
isHighRisk = result && result.prediction === 1;

  return (
    <Box p={3}>
      <Typography variant="h5" gutterBottom fontWeight="bold">          Diabetes Risk
Predictor
      </Typography>
      <Typography variant="subtitle1" color="text.secondary" gutterBottom>
        AI-powered analysis of your diabetes management and health risks
</Typography>

      {/* Risk Summary */}
      <Card         sx={{          p: 3,          mb: 3,
bgcolor: isHighRisk ? "#fff3e0" : "#e8f5e9",          border:
"1px solid #eee",          borderRadius: 3,
        }}
      >
        <Typography          variant="h6"
color={isHighRisk ? "error" :
"success.main"}          fontWeight="bold"
```

```
    >
      {result
       ? isHighRisk
         ? "  High Risk"
         : "  Low Risk"
       : "No Prediction Yet"}
      </Typography>
      <Typography variant="body2" gutterBottom>
       {result
        ? isHighRisk
          ? "Several risk factors identified. Consider consulting your healthcare
provider."
          : "You are at low risk. Keep maintaining healthy habits!"
        : "Run prediction to see your risk analysis."}
      </Typography>
      <Typography variant="h4" fontWeight="bold">
{riskScore}/100
      </Typography>        <LinearProgress
variant="determinate"          value={riskScore}
sx={{ height: 10, borderRadius: 5, mt: 1 }}
      />
    </Card>

    {/* Stats Row */}
    <Grid container spacing={2} mb={3}>
      <Grid item xs={4}>
       <Card sx={{ p: 2, textAlign: "center", borderRadius: 3 }}>
        <Typography variant="h6">{latestGlucose} mg/dL</Typography>
        <Typography variant="body2" color="text.secondary">
         Avg Glucose
        </Typography>
       </Card>
      </Grid>
      <Grid item xs={4}>
       <Card sx={{ p: 2, textAlign: "center", borderRadius: 3 }}>
        <Typography variant="h6">100%</Typography>
        <Typography variant="body2" color="text.secondary">
         Time in Range
        </Typography>
       </Card>
```

```
      </Grid>


 <Grid item xs={4}>
        <Card sx={{ p: 2, textAlign: "center", borderRadius: 3 }}>
          <Typography variant="h6">0%</Typography>
          <Typography variant="body2" color="text.secondary">
            Med Adherence
          </Typography>
        </Card>
       </Grid>
     </Grid>
     {/* Risk Factor Analysis */}
     <Card sx={{ borderRadius: 3, mb: 3 }}>
       <CardContent>
        <Typography variant="h6" gutterBottom>             Risk
Factor Analysis
        </Typography>
        <Divider sx={{ mb: 2 }} />

   <Typography variant="body2">
           Glucose Control: {latestGlucose !== "--" ? "Tracked" : "Not logged"}
        </Typography>
        <Typography variant="body2">• Medication Adherence: 0/100</Typography>
<Typography variant="body2">• Lifestyle Factors: Basic Tracking</Typography>
</CardContent>
     </Card>
     {/* Recommendations */}
     <Card sx={{ borderRadius: 3, mb: 3 }}>
       <CardContent>
        <Typography variant="h6" gutterBottom>             Personalized
Recommendations
        </Typography>
        <Divider sx={{ mb: 2 }} />
        <Typography variant="body2" color="error" fontWeight="bold">
          Improve Medication Adherence (HIGH)
        </Typography>
        <Typography variant="body2" color="text.secondary" gutterBottom>
Missing medications can lead to poor glucose control and complications.
</Typography>
```

```
        <Typography variant="body2" color="warning.main" fontWeight="bold">
Increase Physical Activity (MEDIUM)
        </Typography>


   <Typography variant="body2" color="text.secondary">
          Regular exercise helps improve insulin sensitivity and glucose control.
        </Typography>
      </CardContent>
    </Card>

    {/* Predict Button */}
    <Box textAlign="center">
      <Button
variant="contained"
color="primary"
onClick={handlePredict}
disabled={loading}
      >
        {loading ? "Predicting..." : "Run Prediction"}
      </Button>
    </Box>

   </Box>
  );
}

export default RiskPredictor;
```

PLAGIARISM REPORT

RE-2022-666414.docx

Martinsville City Public Schools

Document Details

Submission ID
trn:oid:::30406:514608690

Submission Date
Oct 18, 2025, 1:34 PM GMT+5:30

Download Date
Oct 18, 2025, 1:37 PM GMT+5:30

File Name
RE-2022-666414.docx

File Size
1.6 MB

7 Pages

3,018 Words

18,984 Characters

turnitin Page 1 of 11 - Cover Page Submission ID trn:oid:::30406:514608690

5% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text

Match Groups

- **14** Not Cited or Quoted **5%**
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations **0%**
  Matches that are still very similar to source material
- **0** Missing Citation **0%**
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted **0%**
  Matches with in-text citation present, but no quotation marks

Top Sources

1% Internet sources
3% Publications
2% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

turnitin Page 2 of 11 - Integrity Overview Submission ID trn:oid:::30406:514608690

REFERENCES

[1] A. Ahmed, "The effectiveness of wearable devices using artificial intelligence in blood glucose forecasting or prediction, Healthcare, vol. 11, no. 9, p. 1241, 2023, doi: 10.3390/healthcare11091241.

[2] X. Wang, Y. Li, and Z. Chen, "Effectiveness and safety of AI-driven closed-loop systems in diabetes," Diabetes Therapy, vol. 16, no. 4, pp. 845–857, 2025, doi: 10.1007/s13300-025-01540-7.

[3] M. Mansour and H. Basha, "Wearable devices for glucose monitoring: A review of state-of-the-art technologies," Sensors, vol. 24, no. 1, p. 45, 2024, doi: 10.3390/s24010045.

[4] J. Ali, S. Khan, and R. Fatima, "A comprehensive remote monitoring system for diabetes using wearables, smartphones and machine learning," MDPI Informatics, vol. 12, no. 2, p. 55, 2025, doi:10.3390/informatics12020055.

[5] H. Zhou, Q. Xu, and J. Li, "A robust machine learning framework for diabetes prediction across diverse populations using PIMA and BD datasets," Compute. Biol. Med., vol. 178, p. 107689, 2025, doi: 10.1016/j.compbiomed.2025.107689.

[6] S. Pikul in, I. Yehezkel and R. Moskov itch, "Enhanced blood glucose levels prediction with a smartwatch," PLOS ONE, vol. 19, no. 7, p. e0307136, 2024, doi: 10.1371/journal.pone.0307136.

[7] H. Naz and S. Ahuja, "Deep learning approach for diabetes prediction using PIMA Indian dataset," Int. J. Adv. Compute. Sci. Appl., vol. 11,no. 9, 2020, doi: 10.14569/IJACSA.2020.0110916.

[8] P. S. Moon and S. Raut, "Machine learning approach for diabetes prediction using PIMA dataset," in Proc. ACM ICCAIML, 2023, doi:10.1145/3647444.3652479.

[9] C. Rodriguez-León, J. Hernández, and A. Canoe, "Mobile and wearable technology for the monitoring of diabetes-related parameters: A systematic review," JMIR mHealth uHealth, vol. 9, no. 6, p. E25138, 2021, doi: 10.2196/25138.

[10] I. Rodríguez-Rodríguez and A. Pérez, "IoMT innovations in the management of diabetes: Predictive models and device integration," Expert System. Application., vol. 241, p. 122837, 2024, doi: 10.1016/j.eswa.2023.122837.

[11] B. Al-Khalifa, L. Alzubaidi and J. Santamaría, "Evaluation of machine learningbased regression for continuous glucose monitoring in a large cohort of patients," Heliyon, vol. 11, no. 3, p. e02345, 2025, doi: 10.1016/j.heliyon. 2025.e02345.

[12] M. Zeynali, K. Alipour, and B. Tarvirdizadeh,"Non-invasive blood glucose monitoring using PPG signals," Sci. Rep., vol. 15, p. 9834,2025, doi: 10.1038/s41598–024– 84265–8.

[13] K. Zhang, J. Liu, and W. Sun, "Future horizons in diabetes: Integrating artificial intelligence and emerging technologies," Front. Endocrinol., vol. 16, p. 1583227, 2025, doi: 10.3389/fendo.2025.1583227.

[14] A. Patel and M. Sharma, "Use of technology in prediabetes and precision prevention," J. Diabetes Investigation., vol. 16, no. 5, pp. 2234–2246, 2025, doi: 10.1111/jdi.70057.

[15] S. D. Alvarez, P. Lopez, and R. Martinez, "Clinical impact of flash glucose monitoring: A digital health review," JMIR Diabetes, vol. 8,no. 1, p. e42389, 2023, doi: 10.2196/42389.

[16] S. L. Cichosz, M. C. Riddle, and H.-H. Lervang, "Explainable machine learning models to predict weekly glucose fluctuations," BMJ Open Diabetes Res. Care, vol. 12, no. 1, p. e004345, 2024, doi:10.1136/bmjdrc-2023-004345.

[17] A. Ahmed, R. Khan, and S. Fatima, "Artificial intelligence-driven wearable device features for diabetes monitoring," J. Med. Internet Res., vol. 24, no. 8, p. e36010, 2022, doi: 10.2196/36010.

[18] A. Y. Alhaddad and M. Thomas, "Advances in wearable technology and noninvasive sensors for glucose monitoring," Front. Bioeng.Biotechnology., vol. 10, p. 876672, 2022, doi: 10.3389/fbioe.2022.876672.

[19] M. Naresh and S. Kumar, "Non-invasive glucose prediction and classification using dual wavelength short near-infrared system," Heliyon, vol. 10, no. 4, p. e15432, 2024, doi: 10.1016/j.heliyon. 2024.e15432.

[20] A. Henne belle, P. Dubois, and J. Laurent, "Secure and privacy-preserving automated ML-Edge-IoT-Blockchain system for diabetes prediction," IEEE Access, vol. 10, pp. 119845–119856, 2022, doi: 10.1109/ACCESS.2022.3217645.