Ministry of Communications
and Information Technology

رواد مصر الرقمية

Project 3: Healthcare Predictive Analytics Project

v1.0

GitHub link: aldwaltly/DEBI-Project

Student name	Student ID
Youssef Mohamed Mohamed	21064291
Mohamed Mahmoud Hussein	21064445
Ahmed Wael Mohamed	21095708
Abdelrahman Ahmed Samir	21066553
Moustafa Hamdy Mohamed	21093187
Amr Yasser Hamdan	21074207

Contents

Project Planning & Management

1. Project Proposal

This project focuses on developing a predictive model for diabetes diagnosis based on a dataset containing relevant health parameters. The goal is to leverage machine learning techniques to predict whether an individual is likely to have diabetes, using medical and demographic data.

Objectives

1. Data Understanding & Preprocessing
    - Analyze the dataset structure and features.
    - Handle missing values, outliers, and data imbalances.
    - Perform feature selection and engineering to enhance model performance.
2. Exploratory Data Analysis (EDA)
    - Identify patterns and correlations between features.
    - Visualize distributions and trends in diabetic vs. non-diabetic patients.
3. Model Development
    - Train multiple machine learning models (e.g., Logistic Regression, Decision Trees, Random Forest, SVM, and Neural Networks).
    - Optimize model parameters using hyperparameter tuning.
    - Evaluate models using performance metrics (e.g., accuracy, precision, recall, F1-score, ROC-AUC).
4. Deployment & Interpretation
    - Develop a user-friendly interface for predictions.
    - Interpret model results to provide actionable insights for healthcare professionals.

Scope of the Project

- Dataset: The project will use structured medical data, such as the PIMA Indian Diabetes Dataset or any other relevant diabetes dataset.
- Features Considered: Age, BMI, blood pressure, glucose levels, insulin levels, skin thickness, pregnancy count, and family history of diabetes.
- Outcome Variable: Binary classification – Diabetic (1) or Non-Diabetic (0).
- Target Audience: Healthcare practitioners, data scientists, and medical researchers aiming to leverage AI for early diabetes detection.

2. Project Plan

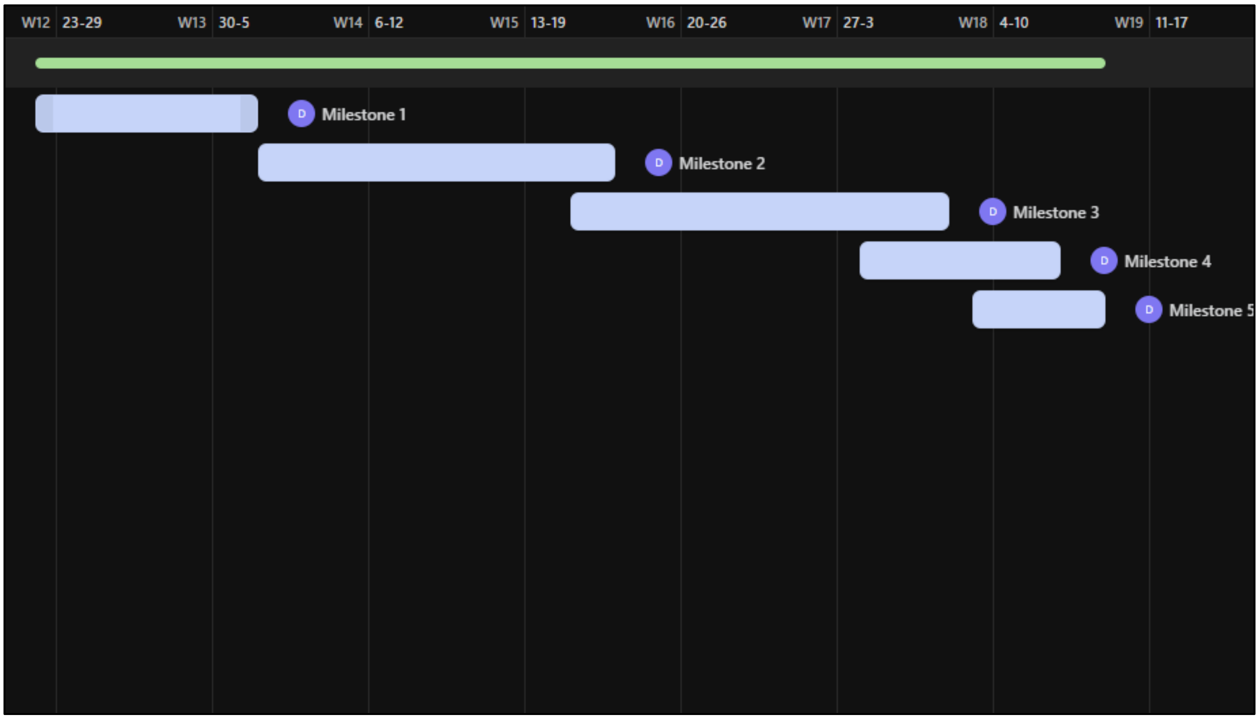

Figure 1 Gnatt chart

Milestone	**Key Deliverables**
1. Data Collection, Exploration & Preprocessing	**EDA Report, Interactive Visualizations, Cleaned Dataset**
2. Data Analysis, Visualization & Feature Engineering	**Data Analysis Report, Visualizations of Health Trends, Feature Engineering Summary**
3. Model Development & Optimization	**Model Evaluation Report, Model Code, Final Model**
4. MLOps, Deployment & Monitoring	**Deployed Model, MLOps Report, Monitoring Setup**
5. Final Documentation & Presentation	**Final Project Report, Final Presentation**

Resource Allocation

1. Human Resources

Role	Responsibilities	Estimated Effort
Data Scientist	Data preprocessing, feature engineering, model development, and evaluation	40%
Machine Learning Engineer	Model optimization, deployment, and API development	25%
Data Analyst	Exploratory Data Analysis (EDA), visualization, and insights generation	15%
Software Developer	Frontend/backend development for user-friendly implementation	10%
Project Manager	Overseeing the project timeline, resource allocation, and coordination	10%

2. Data Resources

Resource	Description
Dataset	Diabetes prediction dataset
Feature Selection	glucose level, BMI, age, Smoking history, etc.

3. Task Assignment & Roles

Team member	Role & Assignment
Mohamed Mahmoud (Leader)	Project management, Model Training, Optimization & Task Coordination
Abdelrahman Ahmed	Model Tuning & Hyperparameter Optimization
Youssef Mohamed	Deployment & AI Development
Ahmed Wael	Model Evaluation & Performance Metrics
Amr Yasser	Exploratory Data Analysis (EDA) & Visualization
Moustafa Hamdy	Data Preprocessing & Feature Engineering

4. Risk Assessment & Mitigation Plan

Risk	Impact	Mitigation Strategy
Data Quality Issues (e.g., missing or biased data)	Poor model accuracy	Perform data preprocessing (imputation, outlier removal) and use a diverse dataset.
Overfitting (model performs well on training but poorly on real-world data)	Poor generalization to new patients	Use cross-validation, regularization (L1/L2), and dropout layers (for neural networks).

Class Imbalance (e.g., more non-diabetic samples than diabetic)	Model favors majority class	Use SMOTE (Synthetic Minority Over-sampling Technique) or adjust class weights.

5. Key Performance Indicators (KPIs) for a Diabetes Prediction Model

For a machine learning model predicting diabetes, relevant KPIs focus on accuracy, reliability, and fairness.

1. Model Performance KPIs

- **Accuracy** → Percentage of correctly predicted cases.
- **Precision & Recall** → Measures false positives and false negatives.
- **F1-Score** → Balances precision and recall.
- **Confusion Matrix** → Measures true positive rate vs. false positive rate.
- **Inference Time** → Speed of making predictions (important if deployed).

2. Data & Fairness KPIs

- **Class Balance Score** → Checks if the model is fair across diabetic/non-diabetic groups.

Literature Review

The literature said “Overall the project planning seems good but choose a good use case for the project as the dataset does not have many features. Also, work on the format and font of the document to make it more professional”. We told him about our use case which is using the model as a predictive system for diabetes diagnosis. He replied, “OK that seems good go ahead and continue working on the project”.

Requirements Gathering

1. Stakeholder analysis

Stakeholder	**Role**	**Needs/Expectations**
Patients (End Users)	Individuals using the system for self-assessment	Accurate predictions, easy-to-use interface, privacy of health data
Doctors & Healthcare Providers	Use predictions to support diagnosis	Reliable predictions, interpretable insights, integration with medical records
Data Scientists & Developers	Build & improve the model	Access to quality data, model interpretability, system scalability
Healthcare Institutions	Hospitals, clinics, research centers	Reduce misdiagnoses, improve patient care, cost-effective deployment

2. User Stories & Use Cases

User Stories

ID	User Role	User Story
US01	Patient	As a patient, I want to input my medical data and receive a diabetes risk prediction so that I can take preventive actions.
US02	Patient	As a patient, I want to see recommendations based on my risk level so that I can improve my lifestyle.
US03	Patient	As a patient, I want my data to be securely stored so that my privacy is protected.
US04	Doctor	As a doctor, I want to review my patient's diabetes risk scores so that I can make informed medical decisions.
US05	Data Scientist	As a data scientist, I want to analyze model performance on different demographics so that I can improve its accuracy.
US06	Data Scientist	As a data scientist, I want access to high-quality, well-labeled data so that I can train a reliable model.

Use Cases

Use Case 1: Patient Checking Diabetes Risk

Actors: Patient, System (Diabetes Prediction Model)

Preconditions:

- The patient has access to the system.
- The patient is willing to input required health data.

Steps:

1. The patient opens the application.
2. The patient enters personal and health-related data (age, BMI, glucose level, etc.).
3. The system processes the data and runs the prediction model.
4. The system displays the risk level (Low, Medium, High) with confidence score.

Postconditions:

- The patient receives diabetes risk assessment.
- The patient can take preventive measures or consult a doctor.

Use Case 2: Data Scientist Improving Model Accuracy

Actors: Data Scientist, System, Patient Data

Preconditions:

- The data scientist has access to anonymized patient data.
- The system logs model performance over time.

Steps:

1. The data scientist loads recent patient data and model predictions.
2. The data scientist analyzes model accuracy, biases, and feature importance.
3. The data scientist retrains the model with additional data if needed.
4. The system updates the model with the improved version.
5. The data scientist tests the updated model for better accuracy and fairness.

Postconditions:

- The model is improved based on new data.
- The system provides more reliable predictions.

Use Case 3: Healthcare Administrator Monitoring System Usage
Actors: Healthcare Administrator, System
Preconditions:

- The system has been deployed in a hospital or clinic.
- The administrator has access to performance reports.

Steps:

1. The administrator logs into the system.
2. The system displays usage statistics (number of predictions, accuracy trends).
3. The administrator reviews system performance and patient feedback.
4. The administrator decides on further actions (e.g., expanding system use).

Postconditions:

- The administrator ensures the system is beneficial and compliant.
- The hospital improves patient care with AI-driven insights.

3. Functional Requirements

Requirement ID	**Requirement Statements**
FR1	Import, normalize, and automate the cleaning and preprocessing of the data. Handle missing values and outliers. The data should be anonymized to ensure privacy.
FR2	Perform data analysis to identify trends and patterns. Use statistical methods such as correlation analysis, hypothesis testing, or feature importance analysis to identify key factors affecting healthcare outcomes.
FR3	Show visualizations to highlight trends, outliers, and significant patterns in health metrics. It also should include an interactive and responsive dashboard to easily view the data.
FR4	Implement the best ML model for the data, split the data into training, validation, and testing, and tune hyperparameters.
FR5	Calculate the performance matrices such as accuracy, F1-score, AUC-ROC, precision, and recall, and evaluate confusion matrix.
FR6	Deploy the model as a REST API or web service accessible to professionals. Implement MLOps practices for versioning, experiment tracking, and reproducibility.
FR7	Set-up continuous monitoring to detect model drift or performance degradation over time and alerts for retraining or updating models based on incoming data or decreased accuracy.

4. Non-Functional Requirements

Measure	Details
Performance	The prediction should have low latency (minimum delay) to ensure real time or near real time response.
Usability	Have a user-friendly dashboard for users and support interactive visualization.
Reliability & Availability	The system has continuous monitoring and implements alerts to target a system availability of 99.99%
Security & Privacy	Implement access controls and authentication and anonymize patient's data.
Maintainability	Implement modular design and apply version control practices for easier updates, integration, and prevent unintended errors.

System Analysis & Design

1. Problem Statement & Objectives – Define the problem being solved and project goals.

Problem Statement

Traditional healthcare systems rely on manual data analysis for diagnosing diseases like **diabetes**, which can lead to delays in medical decisions. By integrating **machine learning (ML)** with a **web-based system**, we can provide **fast and accurate predictions**, assisting patients and doctors in early detection and treatment.

Objectives

- Develop a machine learning model to predict diabetes based on medical data.
- Build a web-based platform where patients can enter their medical records.
- Provide interactive graphs and reports for a better understanding of the results.
- Ensure data security and privacy compliance.

Use case Diagram & Descriptions

Use Case Diagram

The system will have three main actors:

- Patient – Enters medical data and views results.
- ML Model – Analyzes data and generates predictions.
- System (Website) – Handles user interaction, sends data to the model, and displays results.

Use Case Descriptions

Use Case: Diabetes Prediction

- Actor: Patient
- Preconditions: The patient must provide all required medical inputs.
- Steps:
    1. The patient enters medical data (glucose level, BMI, age, etc.).

2. The website sends data to the ML Model.
3. The model processes the input and returns a prediction.
4. The system displays the result as a graph and a message indicating whether the patient is at risk for diabetes.

- Expected Outcome: The patient receives a clear, data-driven diagnosis.

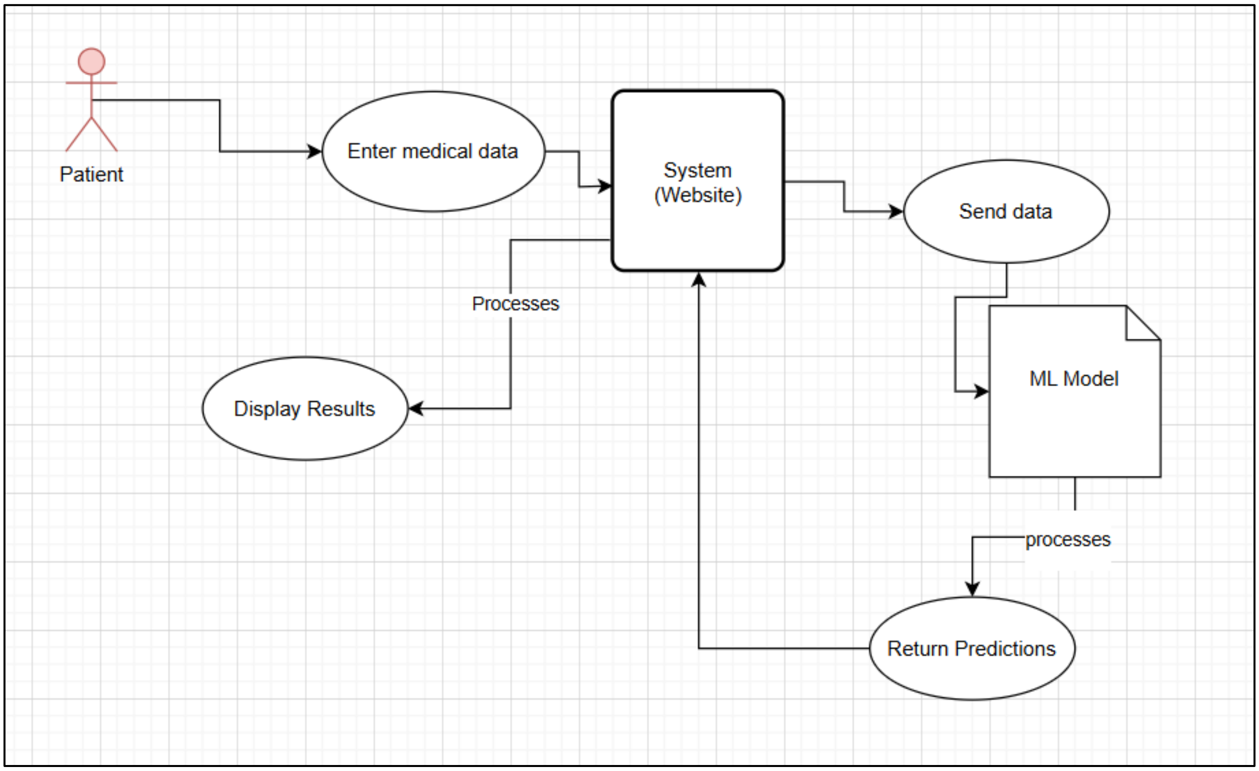

Figure 2 Use Case Diagram

Functional & Non-Functional Requirements:

Functional Requirements:

Requirement ID	**Requirement Statement**
FR1	The website should allow patients to input their medical data.
FR2	The system should send the data to the AI model for prediction.
FR3	The result should be displayed in a graph and a health status message.
FR4	The result should be accurate
FR5	The system should store patient records securely.

Non-Functional Requirements:

Requirement	Requirement Statement
Performance	Predictions should be generated in ≤ 3 seconds.
Security	Patient data must be encrypted.
Scalability	The system should handle large datasets efficiently.
Usability	The website should be responsive and user-friendly.

Software Architecture

Architecture Type:
We will use MVC (Model-View-Controller) architecture:

- Model: Machine Learning model (predicts diabetes risk).
- View: Website frontend (where patients input data and view results).
- Controller: Backend API (handles requests and connects frontend with ML model).

System Components:
Frontend (User Interface) – Built with Angular to create a modern and responsive experience.
Backend (API & Logic Processing) – Developed using Flask or FastAPI to process data and interact with the AI model.
Database (Storage Layer) – Uses PostgreSQL or MongoDB to store patient records securely.
Machine Learning Model – Implemented with Scikit-learn or TensorFlow for predicting diabetes risk.

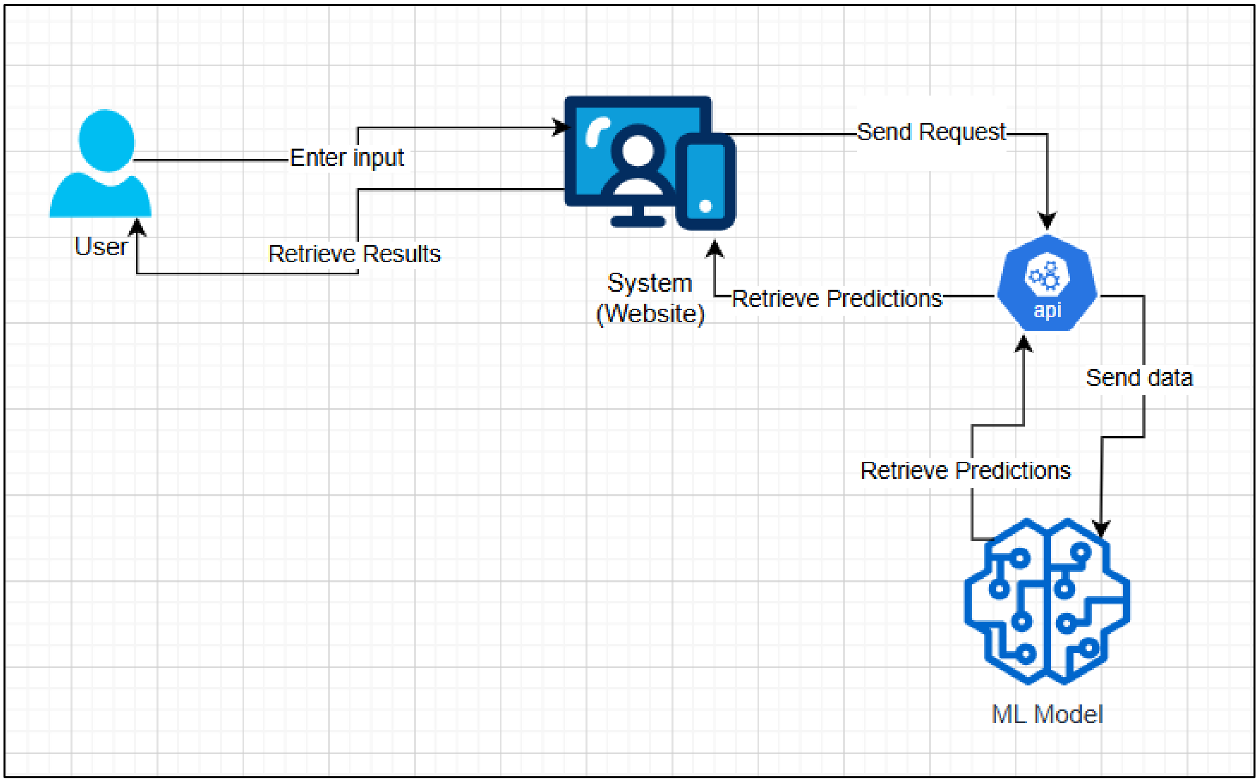

Figure 3 Use Case Diagram 2

2. Database Design & Data Modeling

1. Introduction

This document presents the database design and data modeling for the Diabetes Prediction Project. The goal is to structure the database efficiently to store and retrieve patient data, medical parameters, and prediction results while ensuring scalability and data integrity.

2. Entity-Relationship Diagram (ERD)

The ERD represents the database structure, showing entities, attributes, and relationships. Key entities include Patients, Medical Tests, Predictions, and Doctors.

3. Database Schema

The database consists of the following tables:

Patients

- PatientID (Primary Key)
- Name
- Age
- Gender
- FamilyHistory
- ContactInfo

Medical Tests

- TestID (Primary Key)
- PatientID (Foreign Key)
- GlucoseLevel
- BMI
- BloodPressure
- InsulinLevel
- SkinThickness
- Pregnancies
- TestDate

Predictions

- PredictionID (Primary Key)
- PatientID (Foreign Key)
- TestID (Foreign Key)
- PredictionResult (0: Non-Diabetic, 1: Diabetic)
- ConfidenceScore
- PredictionDate

Doctors

- DoctorID (Primary Key)
- Name
- Specialization
- ContactInfo

4. Normalization

Normalization is applied to eliminate redundancy and improve data integrity. The tables follow Third Normal Form (3NF) to ensure efficient storage and avoid anomalies.

5. Justification for Design Choices

The database design ensures:
Scalability: Supports a growing dataset efficiently.

Data Integrity: Foreign keys maintain relationships between tables.
Performance Optimization: Indexed keys allow fast retrieval of patient records.
Security: Sensitive data is stored separately to enhance privacy.

3. Data Flow & System Behavior

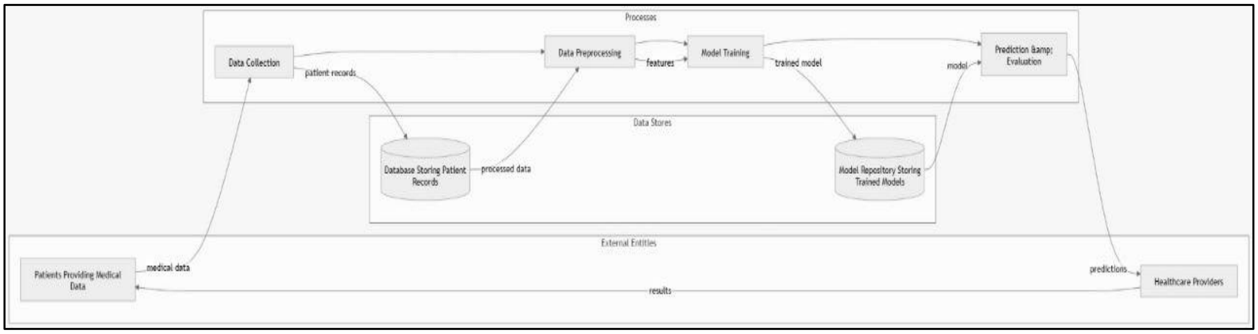

Figure 4 DFD Diagram

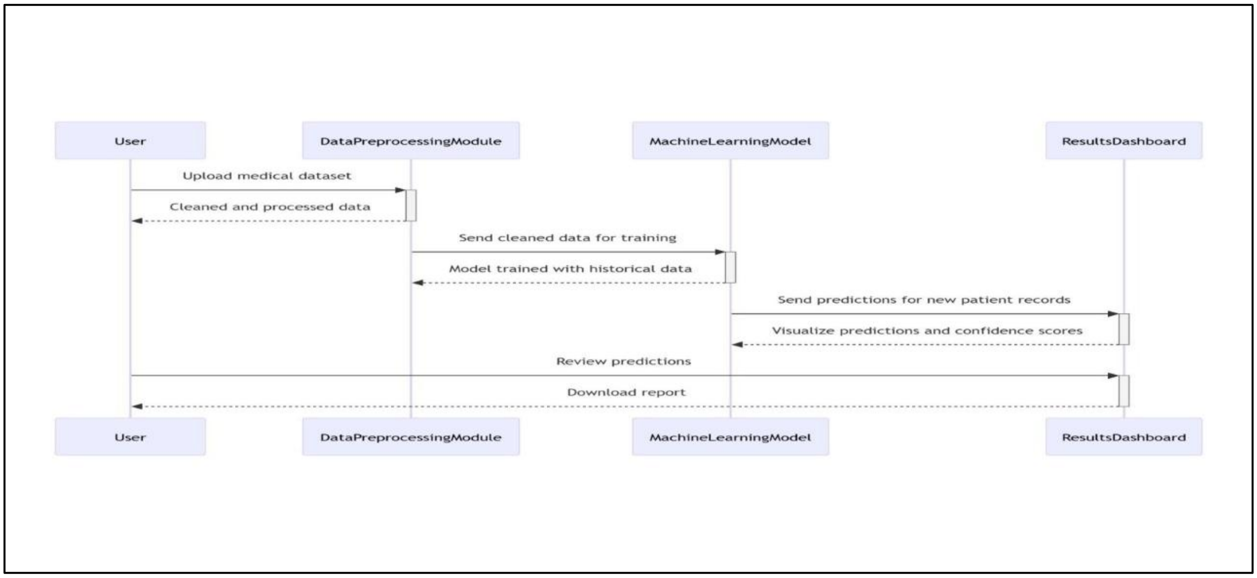

Figure 5 Sequence Diagram

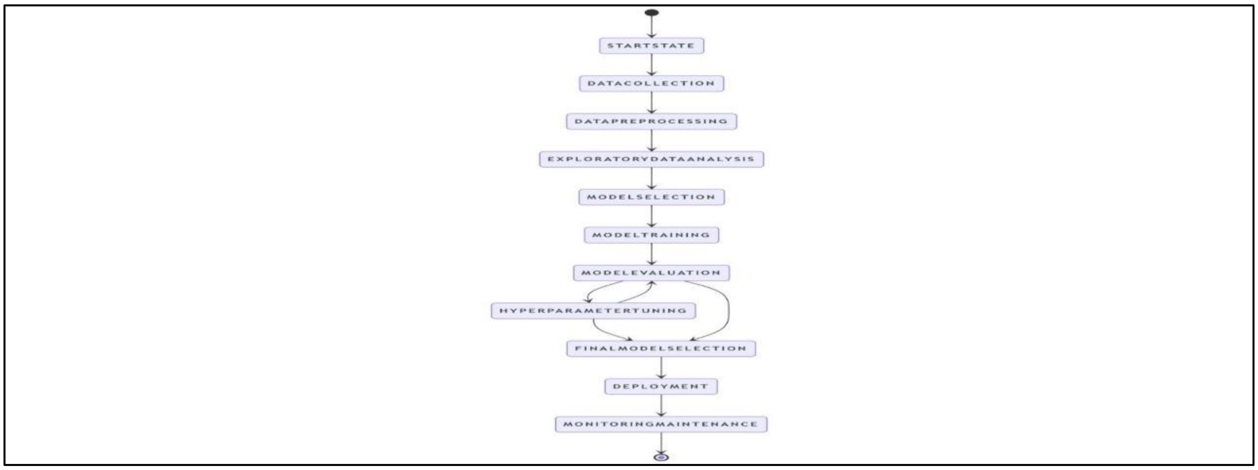

Figure 6 State Diagram

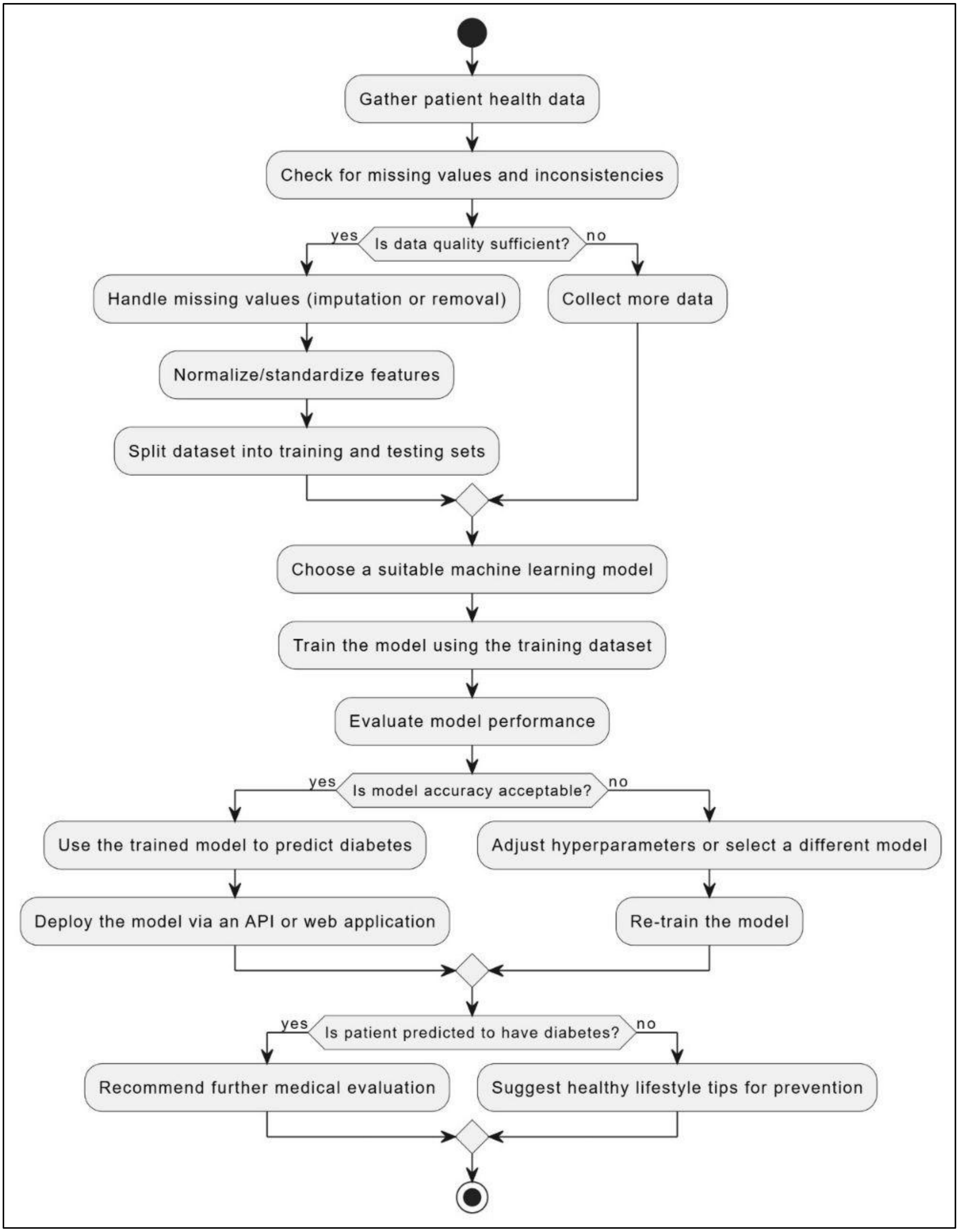
Gather patient health data
Check for missing values and inconsistencies
Is data quality sufficient?
yes
no
Handle missing values (imputation or removal)
Collect more data
Normalize/standardize features
Split dataset into training and testing sets
Choose a suitable machine learning model
Train the model using the training dataset
Evaluate model performance
Is model accuracy acceptable?
yes
no
Use the trained model to predict diabetes
Adjust hyperparameters or select a different model
Deploy the model via an API or web application
Re-train the model
Is patient predicted to have diabetes?
yes
no
Recommend further medical evaluation
Suggest healthy lifestyle tips for prevention

Figure 7 Activity Diagram

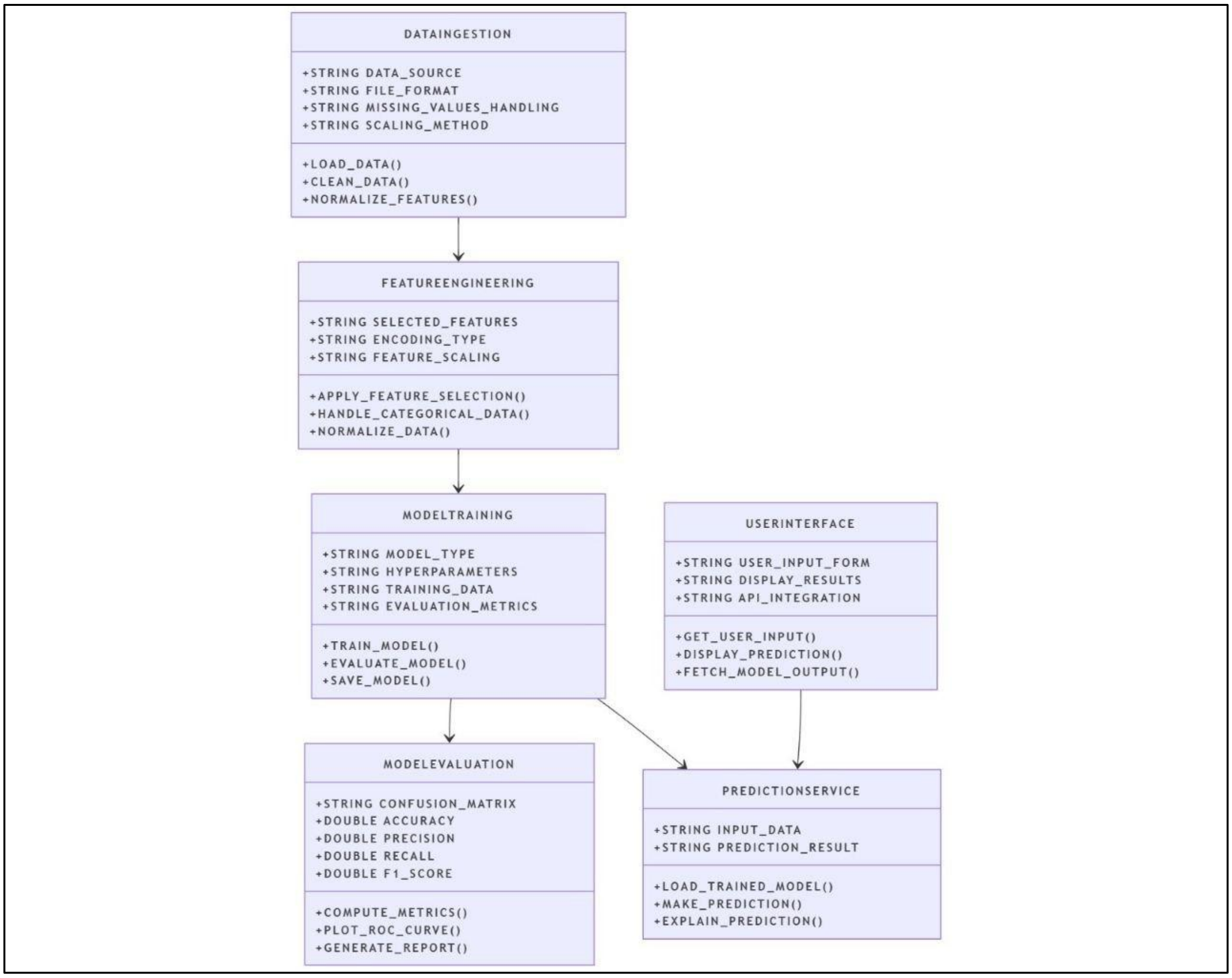

Figure 8 Class Diagram

4. UI/UX Design & Prototyping

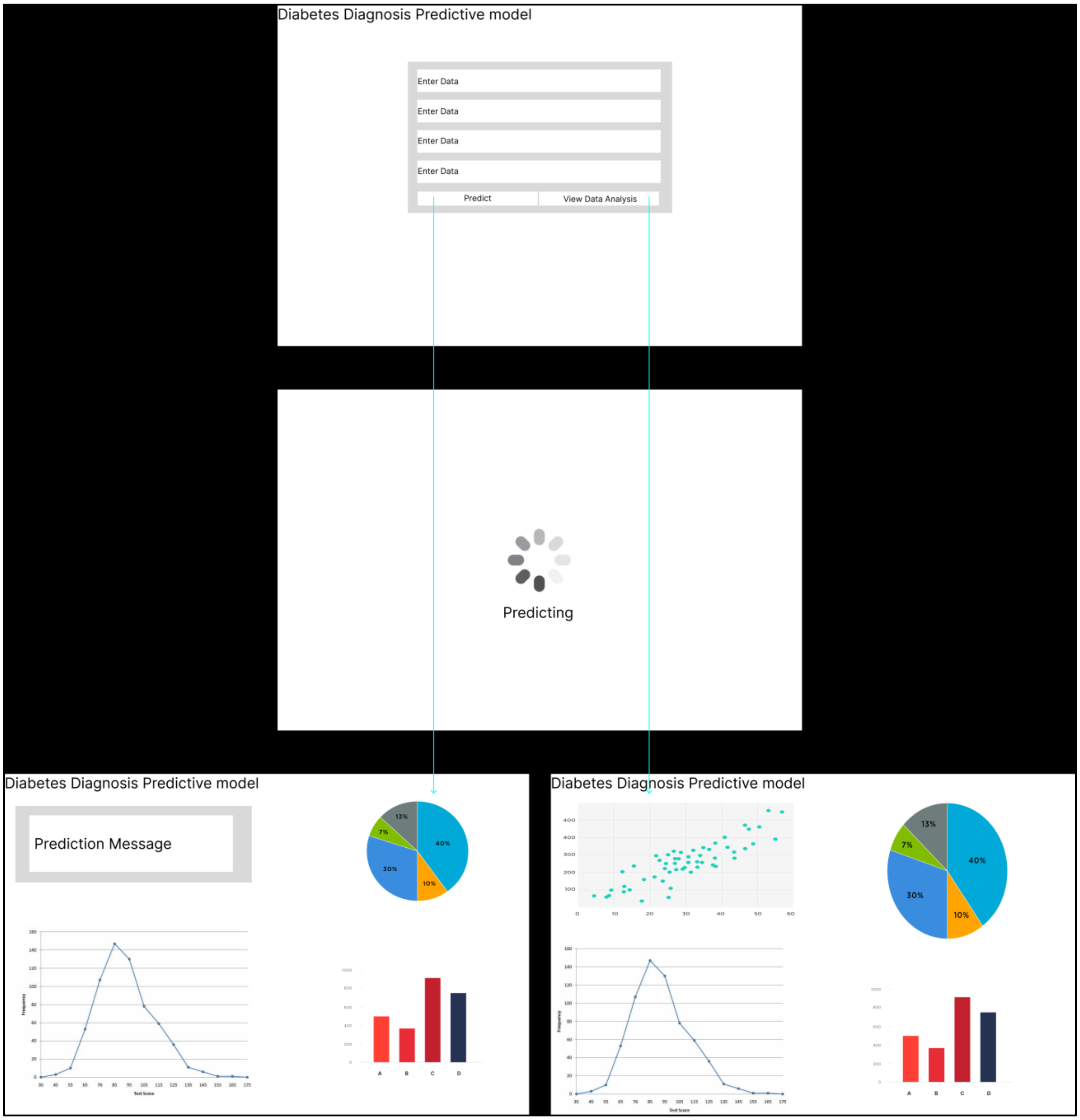

Figure 9 UI Design

The above is a system navigation showing how we plan to build the UI. However, we haven't decided on the actual design, colors, typography, and accessibility yet.

5. System Deployment & Integration

Technology Stack

A technology stack is a collection of technologies used to create a software application. It includes different layers that work together, from the user-facing interface to the databases and server logic behind the scenes.

Backend Technologies

The backend handles the logic, database interactions, user authentication, and communication with servers. It forms the foundation of your application.

- **Flask** – A lightweight and flexible framework, ideal for building machine learning APIs and handling requests efficiently.

Frontend Technologies

The frontend is what users see and interact with. It covers the user interface (UI) and user experience (UX).

- **Angular** – A robust framework for building dynamic web applications, providing a structured and scalable approach.

Database Technologies

Databases store and manage data for your application. The choice between relational (structured) and non-relational (flexible) databases is critical.

- **MySQL** – Chosen for its reliability, scalability, and ability to handle structured health-related data securely.

Model Deployment

- **TensorFlow Serving** – Used to efficiently deploy and manage the predictive model for real-time diabetes risk assessment.

Deployment Diagram

The deployment diagram illustrates how different software components are distributed across hardware. It includes:

- **User Interface (Angular Web App):** Hosted on web, allowing users to input data and receive predictions.
- **Backend Server (Flask API):** Deployed on a server, responsible for processing user requests and communicating with the model.
- **Database (MySQL):** Securely hosted on a database server to store user data and predictions.
- **Model Server (TensorFlow Serving):** Handles real-time prediction requests and interacts with the backend API.

Component Diagram

The component diagram provides a high-level view of the system's main components and their interactions:

- **Frontend (Angular Web App):** Collects user input and displays results.
- **Backend (Flask API):** Routes requests between the frontend, model, and database.
- **Predictive Model (TensorFlow Serving):** Processes input data and provides diabetes risk predictions.
- **Database (MySQL):** Stores patient data, prediction history, and analytics.
- **Authentication Module:** Ensures secure access to the system, preventing unauthorized use.

Explanation of Chosen Technologies

- **Flask:** Chosen because it is lightweight, easy to use, and well-suited for building APIs that handle machine learning models.
- **Angular:** Selected for its ability to create structured, scalable, and interactive web applications with modular components.
- **MySQL:** Preferred for its structured data storage, reliability, and ability to handle large datasets efficiently.
- **TensorFlow Serving:** Ideal for deploying and managing machine learning models in production with low latency and high performance.

This structure ensures a seamless integration of all components, providing a smooth user experience and efficient model deployment.