4. System Analysis & Design

4 .1 Problem Statement & Objectives

Problem Statement

The COVID-19 pandemic has significantly impacted healthcare systems, leading to overburdened hospitals, limited ICU bed availability, and inadequate allocation of medical resources. The unpredictable nature of the virus has made it difficult for healthcare facilities to anticipate patient influx and manage critical care units effectively. This project aims to develop a data-driven solution that predicts ICU admissions and resource needs, helping hospitals optimize patient care and preparedness strategies.

Project Objectives

Main Objective:

Build a machine learning model to predict the likelihood of COVID-19 patients being at high risk based on their health data.

Sub-Objectives:

- **Predict patient hospitalization needs**: Develop an AI-powered model to determine whether a COVID-19 patient requires hospitalization, ICU admission, or ventilator support.
- **Improve healthcare resource management**: Analyze trends in COVID-19 hospitalizations to help healthcare administrators efficiently allocate resources and plan for patient surges.
- **Enhance decision-making for medical professionals**: Provide doctors with real-time predictive insights to assist in prioritizing patient care and early intervention.
- **Support government policy planning**: Offer data-driven recommendations to government agencies for better pandemic response and public health strategies.
- **Develop an accessible and user-friendly system**: Ensure that healthcare professionals can easily interpret and interact with the predictive system for quick decision-making.

4.2 Use Case Diagram & Descriptions

Use Case Diagram

A **Use Case Diagram** represents how different actors (users) interact with the system. Below are the primary actors and their interactions:

- **Hospital Administrators –** Manage hospital resources based on predictions.
- **Doctors & Healthcare Workers –** Input patient data and receive critical insights.
- **Government Agencies –** Access regional reports for healthcare planning.

- **Patients** – Check hospital availability for ICU and general admission.
- **System (AI Model)** – Processes data and provides predictive insights.

Use Case Descriptions

Use Case	**Actor(s)**	**Description**
Predict ICU & Ventilator Demand	Hospital Administrator, Doctor	System predicts hospital resource needs based on patient data and displays recommendations.
Monitor COVID-19 Trends	Government Agencies	Provides reports on infection rates, hospital capacities, and patient trends regionally.
Patient Admission Request	Patient, Hospital Administrator	Patients can check ICU availability, and administrators can approve or redirect admissions.
Data Entry & Processing	Doctors, Healthcare Workers	Medical staff input patient information for real-time processing and analysis.
System Alerts & Notifications	Hospital Administrator, Doctor	The system sends alerts if a hospital is at critical capacity or ICU demand is predicted to rise.
Generate Reports & Insights	Hospital Administrator, Government	Decision-makers receive predictive analytics and trend reports for strategic planning.

4.2.3.. Functional & Non-Functional Requirements:

Functional Requirements:

1. Data Entry:
   The system must allow doctors to enter patient data (e.g., age, sex, chronic diseases).

2. Data Processing:
   The system must process the data using a machine learning model to generate predictions.

3. Display Results:
   The system must display prediction results to the doctor in a clear manner.

4. Data Storage:
   The system must store patient data in a database.

Non-Functional Requirements:

1. **Performance**:
   The system must process data and generate predictions in less than 5 seconds.

2. **Availability:**
   The system must be available 24/7.

3. **Security:**
   Patient data must be encrypted and protected.

4. **Scalability:**
   The system must handle an increasing number of users and data.

4.2.3.Software Architecture:

High-Level Design:

- Architecture Style:

  The MVC (Model-View-Controller) architecture will be used to separate the user interface from the business logic and database.

- System Components:
  1. View (User Interface):

     A web interface that allows doctors to input data and view results.

  2. Controller (Business Logic):

     Handles user requests and interacts with the model and database.

  3. Model (Data & ML Model):

     Contains the machine learning model and database.

  4. Database:

     Stores patient data and results.

4 .2 Database Design & Data Modeling:

4.2.1. ER Diagram (Entity-Relationship Diagram):

- **Entities:**
  1. **Patient:**

     Attributes: (PatientID (Primary Key), Sex, Age, Pregnant, Tobacco)

  2. **MedicalHistory:**

     Attributes: (MedicalHistoryID (Primary Key), Diabetes, COPD, Asthma, Hypertension, Cardiovascular, RenalChronic, Obesity, OtherDisease, Immunosuppressed, PatientID (Foreign Key))

  3. **TestResult:**

     Attributes: (TestID (Primary Key), Classification, Pneumonia, PatientID (Foreign Key)).

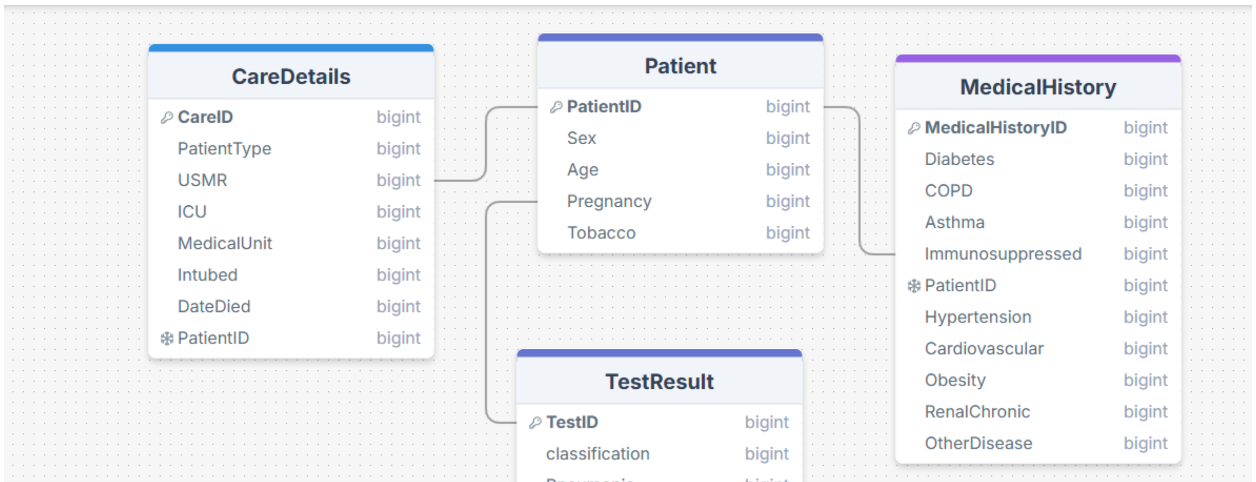
  4. **CareDetails:**

     Attributes: (CareID (PK), PatientType, USMR, MedicalUnit, Intubed, ICU, DateDied, PatientID (FK)).

- **Relationships :**
  1. Patient → MedicalHistory:
     Relationship Type: One-to-One (Each patient has one medical history).
     Foreign Key:PatientID in MedicalHistory references PatientID in Patient.

  2. Patient → TestResult:
     Relationship Type:One-to-One (Each patient has one test result).
     Foreign Key:PatientID in TestResult references PatientID in Patient.

  3. Patient → CareDetails:
     Relationship Type: One-to-One (Each patient has one care detail).
     Foreign Key:PatientID in CareDetails references PatientID in Patient.



ERD Diagram

4.2.2. Logical Schema:

- **Tables:**
    1. Patient:
       Columns: (PatientID (PK, INT), Sex (INT), Age (INT), Pregnancy (INT), Tobacco (INT)).
    2. MedicalHistory:
       Columns:(MedicalHistoryID (PK, INT), Diabetes (INT), COPD (INT), Asthma (INT), Hypertension (INT), Cardiovascular (INT), RenalChronic (INT), Obesity (INT), OtherDisease (INT), Immunosuppressed (INT), PatientID (FK, INT)).
    3. TestResult:
       Columns: (TestID (PK, INT), Classification (INT), Pneumonia (INT), PatientID (FK, INT)).
    4. CareDetails:
       Columns:(CareID (PK, INT), PatientType (INT), USMR (INT), MedicalUnit (INT), Intubed (INT), ICU (INT), DateDied (DATE), PatientID (FK, INT)).

4.2.3. Physical Schema:

SQL Table Creation:

1. Patient Table

```
CREATE TABLE Patient (
  PatientID INT AUTO_INCREMENT PRIMARY KEY,
  Sex INT,
  Age INT,
  Pregnancy INT,
  Tobacco INT
);
```

2. MedicalHistory Table

```
CREATE TABLE MedicalHistory (
  MedicalHistoryID INT AUTO_INCREMENT PRIMARY KEY,
  Diabetes INT,
  COPD INT,
  Asthma INT,
  Hypertension INT,
  Cardiovascular INT,
  RenalChronic INT,
  Obesity INT,
  OtherDisease INT,
  Immunosuppressed INT,
  PatientID INT,
  FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);
```

3. TestResult Table

```
CREATE TABLE TestResult (
    TestID INT AUTO_INCREMENT PRIMARY KEY,
    Classification INT,
    Pneumonia INT,
    PatientID INT,
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);
```

4. CareDetails Table

```
CREATE TABLE CareDetails (
    CareID INT AUTO_INCREMENT PRIMARY KEY,
    PatientType INT,
    USMR INT,
    MedicalUnit INT,
    Intubed INT,
    ICU INT,
    DateDied DATE,
    PatientID INT,
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);
```

4.2.3. Project Documentation:

Project Description:

Objective:Build a database to store COVID-19 patient data for predicting high-risk patients.

Tables:4 tables (Patient, MedicalHistory, TestResult, CareDetails).

Relationships:Each table is linked to the Patient table via a Foreign Key.

4 .3. Data Flow & System Behavior:

4.3.1. DFD (Data Flow Diagram)

Context-Level DFD:

- External Entities:
    1. Doctor:

       Enters patient data and receives predictions.

    2. ML Model: Processes data and generates predictions.

- Main Process:

  Prediction System:

  Receives data from the doctor and returns predictions.

Detailed levels (DFD):

- Processes:
    1. Enter Data:

       The doctor enters patient data.

    2. Process Data:

       The system processes the data using the ML model.

    3. Return Results:

       The system displays the results to the doctor.

- Data Stores:
    1. Database:

       Stores patient data.

4.3.2. Sequence Diagrams

Sequence of Interactions:

1. Doctor Enters Data:

   Doctor → System: Send patient data.

   System → Database: Store data.

   System → ML Model: Process data.

   ML Model → System: Return predictions.

System → Doctor: Display results.

4.3.3. Activity Diagram

- Workflow:
  1. Start:

     The doctor enters patient data.
  2. Process Data:

     The system processes the data using the ML model.
  3. Display Results:

     The system displays the results to the doctor.
  4. End:

     The process ends.

4.3.4. State Diagram

- System States:
  1. Idle:

     The system is waiting.
  2. Processing:

     The system is processing data.
  3. Displaying Results:

     The system is displaying results.
  4. End:

     The process ends.

4.3.5. Class Diagram

- System Structure
  1. Class: Patient:

     Attributes: PatientID, Sex, Age, Pregnancy, Tobacco.

     Methods: EnterData(), GetPrediction().

  2. Class: MedicalHistory

     Attributes: MedicalHistoryID, Diabetes, COPD, Asthma, Hypertension, Cardiovascular, RenalChronic, Obesity, OtherDisease, Immunosuppressed.

     Methods: StoreData(), RetrieveData().

  3. Class: TestResult

     Attributes: TestID, Classification, Pneumonia.

     Methods: ProcessData(), GeneratePrediction().

  4. Class: CareDetails

     Attributes: CareID, PatientType, USMR, MedicalUnit, Intubed, ICU, DateDied.

     Methods: StoreData(), RetrieveData().