TEAM-5

TEAM MEMBERS:

AMELIN THOMAS

RITHWIK S MENON

ROHIT DEEPAK

SHALLET MARY T ELDHO

CLINIC MANAGEMENT SYSTEM

Abstract

The Clinic Management System is a highly tuned management system that assists in the automation of clinic operations and activities. The clinic's administrative and financial operations, as well as all medical facilities, are all monitored by this system. The entire system works together to guarantee that critical daily chores and interactions run smoothly.

Functional Requirements

1.There are 4 users

- Admin
- Office staff
- Doctor
- Lab technician

2.Admin

- This user can assign roles and responsibilities to staff according to their designation,
- The user has all the privileges but it cannot register itself on this system as the admin needs to verify the registering user. After the admin verifies the user only when he is eligible to use the the-the system and after that, it has all the privileges.

3.Front office.

- Register the patient with his details(Name,gender,age,Address,Blood group,Contact Number)
- The details of the patients are registered and added to the database and a unique identification number is generated.

- After the patient is registered,the patient is assigned to the doctor that they wish to see,and a token number is given to the patient.

4.Doctor

- The patient list for the day is provided to the doctor from the front office.
- After consultation the prescription and medicines are added by the doctor
- Doctor assigns required tests for the patient.
- The doctor should have access to the lab test results of the patient till date and also the results of the test assigned by the doctor.
- The history of the patient is accessible to the doctor.

5.Lab technician

- The lab tests required for patients are given to the lab from the doctor or front desk.
- When the lab technician gets the notification for the test, they identify the patient based on the unique identification number. The test history is analysed and the tests prescribed on that particular day are performed and the results should be updated to the doctor and the details of the tests performed are updated to the front office.
- Can access the day wise list of patients who need to undergo different tests assigned by the doctor,search for a particular patient and view his/her result.

Non-Functional Requirements

1.The data in the patient management system should be always accurate.

2.The patient details should be protected from unauthorised access.

3.The system should be reliable so that the history of the patient is not lost.

4.The system should be able to handle any number of patients.

5.There shouldn't be any time delay between data entry and data retrieval.

6.The patient management system should be available at any time for the required users

DATABASE DESIGN

```
use DBClinic;

CREATE TABLE Role(
RoleId INT IDENTITY(1,1) PRIMARY KEY,
RoleName VARCHAR(20) NOT NULL,
IsActive bit not null);



CREATE TABLE Users(
UserId INT IDENTITY(1,1) PRIMARY KEY,
UserName VARCHAR(20) NOT NULL,
UserPassword VARCHAR(20) NOT NULL,
```

```
IsActive bit not null,
RoleId INT
FOREIGN KEY (RoleId) REFERENCES Role(RoleId));



create table Departments(
DepartmentId int identity(1,1) primary key,
DepartmentName varchar(30) not null,
IsActive bit not null);

create table Designations(
DesignationId int identity(1,1) primary key,
Designation varchar(20) not null,
IsActive bit not null);

create table Staffs(
StaffId int identity(1,1) primary key,
StaffName varchar(30) not null,
Experience decimal,
JoiningDate date not null,
IsActive bit not null,
DepartmentId int,
DesignationId int,
```

```
FOREIGN KEY (DepartmentId) REFERENCES Departments(DepartmentId),
FOREIGN KEY (DesignationId) REFERENCES Designations(DesignationId));

create table Patients(
PatientId int identity(1,1) primary key,
PatientName varchar(30),
Contact int not null,
Address varchar(30),
Gender varchar(20) not null,
IsActive bit not null);

create table Doctors(
DoctorId int identity(1,1) primary key,
DoctorName varchar(20) not null,
Specialization varchar(20) not null,
IsActive bit not null);

create table Consultings(
PatientId int,
DoctorId int,
ConsultingId int identity(1,1) primary key,
ConsultingDate date,
IsActive bit not null,
FOREIGN KEY (PatientId) REFERENCES Patients(PatientId),
FOREIGN KEY (DoctorId) REFERENCES Doctors(DoctorId));
```

```
create table Tests(
TestId int identity(1,1) primary key,
PatientId int,
ConsultingId int,
TestDate date,
TestNames varchar(90),
IsActive bit not null,
FOREIGN KEY (PatientId) REFERENCES Patients(PatientId),
FOREIGN KEY (ConsultingId) REFERENCES Consultings(ConsultingId));

create table Prescriptions(
PrescriptionId int identity(1,1) primary key,
ConsultingId int,
TestId int,
Medicines varchar(90),
DoctorNotes varchar(100),
IsActive bit not null,
FOREIGN KEY (ConsultingId) REFERENCES Consultings(ConsultingId),
FOREIGN KEY (TestId) REFERENCES Tests(TestId));


create table Results(
ResultId int identity(1,1) primary key,
```

```
TestId int,
Result decimal,
IsActive bit not null,
FOREIGN KEY (TestId) REFERENCES Tests(TestId));

create table TestBills(
TestBillId int identity(1,1) primary key,
ConsultingId int,
TestId int,
TestAmount decimal,
IsActive bit not null,
FOREIGN KEY (TestId) REFERENCES Tests(TestId),
FOREIGN KEY (ConsultingId) REFERENCES Consultings(ConsultingId));


create table Medicines(
MedicineId int identity(1,1) primary key,
PrescriptionId int,
Quantity int,
UnitPrice decimal,
Amount decimal,
IsActive bit not null,
FOREIGN KEY (PrescriptionId) REFERENCES Prescriptions(PrescriptionId));
```

```
create table Bills(
BillId int identity(1,1) primary key,
PatientId int,
ConsultingId int,
MedicineId int,
ConsultingFee decimal,
TotalAmount decimal,
IsActive bit not null,
FOREIGN KEY (PatientId) REFERENCES Patients(PatientId),
FOREIGN KEY (ConsultingId) REFERENCES Consultings(ConsultingId),
FOREIGN KEY (MedicineId) REFERENCES Medicines(MedicineId));

create table Reports(
ReportId int identity(1,1) primary key,
TestId int,
ResultId int,
PatientId int,
ReportDate date,
IsActive bit not null,
FOREIGN KEY (PatientId) REFERENCES Patients(PatientId),
FOREIGN KEY (TestId) REFERENCES Tests(TestId),
FOREIGN KEY (ResultId) REFERENCES Results(ResultId));
```