DCIT 318: PROGRAMMING II

ASSIGNMENT 4

Submission Requirements:

- Complete Visual Studio project folder
- Screenshots of all working features

QUESTION 1

Design and implement a Windows Forms-based Medical Appointment Booking System that connects to a SQL Server database. The application should allow patients to:

- Book appointments
- View available doctors
- Search and filter appointments
- Delete or modify existing bookings

You will use ADO.NET (Data Providers, Data Readers, DataSets, DataAdapters) to connect and interact with SQL Server. You must implement proper data retrieval and update logic using command objects, parameters, and event-driven logic in Windows Forms.

a. Database Design (SQL Server)
Create a SQL Server database `MedicalDB` with the following tables:

Doctors

- DoctorID (int, Primary Key)
- FullName (varchar)
- Specialty (varchar)
- Availability (bit)

Patients

- PatientID (int, Primary Key)
- FullName (varchar)
- Email (varchar)

Appointments

- AppointmentID (int, Primary Key)

- DoctorID (int, Foreign Key)
- PatientID (int, Foreign Key)
- AppointmentDate (datetime)
- Notes (varchar)

Insert sample data into the `Doctors` and `Patients` tables.

b. Create a Windows Forms Application

`MainForm`: Landing form with navigation buttons
`DoctorListForm`: Displays all doctors using DataGridView
`AppointmentForm`: Allows booking an appointment
`ManageAppointmentsForm`: View, update, or delete existing appointments

c. Windows Forms Controls and Events

Use the following controls and hook up appropriate events:

- **TextBox, ComboBox, Button, DateTimePicker, DataGridView**
- **Common events:** `Click`, `SelectedIndexChanged`, `TextChanged`
- Use **EventHandler delegates** to trigger logic when form controls are used
- Each form should be implemented as a **partial class**

d. Connecting to SQL Server
Use `SqlConnection` to connect to SQL Server
Store the **connection string** in `App.config`

- Include `Data Source`, `Initial Catalog`, `Integrated Security` or `User ID`/`Password`

e. Retrieving and Displaying Data
Use `SqlCommand` and `ExecuteReader` to fetch:

- List of doctors (`CommandType.Text`)
- Patient information (`CommandType.Text`)

Bind data to `DataGridView` or `ComboBox` using a `DataReader`

d. Booking an Appointment

Use `SqlCommand` with parameters and `ExecuteNonQuery` to insert appointment
Use parameter placeholders (`@DoctorID`, `@PatientID`, etc.)
Set `SqlParameter.Direction` to `Input`
Add validation logic before booking (e.g., availability check)

e. Viewing and Managing Appointments

Use `SqlDataAdapter` and `DataSet` to populate and display patient's appointments
Implement update (modify appointment date) using `UPDATE` command and `ExecuteNonQuery`
Implement delete using `DELETE` command

f. Exception Handling

Use `try-catch` blocks around all database operations
Display user-friendly error messages using `MessageBox.Show`
Ensure connections are closed in `finally` blocks

QUESTION 2

You are to develop a Windows Forms application that allows a pharmacy to manage its inventory and record sales. The application will support adding new medicines, searching for medicines by name or category, updating stock levels, and recording purchases using stored procedures in SQL Server.

Database Setup

a. **Create a database** named `PharmacyDB`

b. **Create a table** `Medicines` with fields:
`MedicineID` (int, primary key, identity)
`Name` (varchar)

`Category` (varchar)
`Price` (decimal)
`Quantity` (int)

c. **Create a table** `Sales`:
`SaleID` (int, primary key, identity)
`MedicineID` (foreign key)
`QuantitySold` (int)
`SaleDate` (datetime)

d. Write and save stored procedures:

AddMedicine`(@Name, @Category, @Price, @Quantity)`
`SearchMedicine(@SearchTerm)`
`UpdateStock(@MedicineID, @Quantity)`
`RecordSale(@MedicineID, @QuantitySold)`
`GetAllMedicines()`

Database Setup

e. **Design** the Main Form with the following controls:

- Textboxes for Medicine Name, Category, Price, Quantity
- Buttons: `Add Medicine`, `Search`, `Update Stock`, `Record Sale`, `View All`
- DataGridView to display medicine list
- ComboBox or TextBox for search

f. Use Partial Classes for separation of UI and logic.
g. **Add Events** for each button (e.g., btnAddMedicine_Click, btnSearch_Click, etc.)
h. **Attach Event Handlers** using delegates or designer-generated methods.

Database Connection and Operations

i. Create a connection string to connect to `PharmacyDB` using `SqlConnection`
j. **Use `SqlCommand` with CommandType set to StoredProcedure** for all operations.
k. Use appropriate command execution methods:
`ExecuteNonQuery()` for Add/Update/Sale

`ExecuteReader()` for search and view
`ExecuteScalar()` for returning single values if needed

l. Use `SqlDataReader` to load medicine data into the DataGridView.
m. **Handle ParameterDirection** if stored procedures have output parameters.
n. Display friendly error messages using MessageBox.