Now I want to create tables for my Access Database. show me how to organize it. This is the data that the system will ask from user(admin healthcare provider);

In adding Patient:  
1.PatientID (this is autogenerated by the database)

2.Last Name

3.First Name

4.Middle Name

5. Purok No.

6. Age

7. Sex

8.Birth Date

9.Contact Number

10. Household Serial Number

11.Medical History

12.Prescription

13.Patient Photo(Optional, if user do not want to put photo the system has reserved default Patient photo)

In adding a drug:

1. DrugID

2. Drug Name

3. Generic Name

4. Brand Name

5.Date Added (This is autogenerated by the system according to present date)

6. Expiry Date

7. Manufacturing Date

8. Category

9. Strength

10. Measurement Type

11. Quantity

12. Drug Photo (Optional, if user do not want to put photo the system has reserved default photo)  
  
  
These information are the to be showed also in all CRUD functionalities.  
Also make a query guide for me, show what are necessary based on my system’s functionalities.  
  
My system can do the following I future;

1. CRUD for Patient Records

2. CRUD for Drug Records

3. Display Low Stock Drug on “Home” by making it as notification to user, if none it dispays “No Low Stock Drug”

4. Display when there is a drug going to be expired tomorrow, if none it will show “No expired drugs”. If the drug has expired it will autoremoved by the system and display on “Home” saying “A drug named \*drug name here\* have been removed

5. Show on “Home” the bar graph Current Distribution of Patients in the barangay per Purok

6. Show on “Home” the Line Graph of Drug Trend in barangay based on the drugs prescribed to patients.

7. Show on “Home” the pie graph of Puroks in Entire Barangay with respect to number of patients per purok.

8. Show on “Home” the newly added Patients and Drugs by flow layout panels

9. Operate CRUD for Patients and Drug/Inventory.

10. In “Analytics” it can generate data based on the want of the user, it can be filtered Medicine demand with respect to puroks Age, Sex, Medical History or to Just Generate Overall distribution of patients in the entire barangay with respect to puroks

11. Can Export Data in PDF depending on what data analytics the user wants to export

12. Has AI API called “HealthBud” so that the user will have an ai assistance if needed

Query for Patient Fields:

1.PatientID (AutoNumber)

2.LastName (Short Text)

3.FirstName (Short Text)

4.MiddleName (Short Text)

5. PurokNumber (Short Text)

6. Age (Number)

7. Sex (Short Text)

8.BirthDate (Date/Time)

9.ContactNumber (Short Text)

10. HouseholdSerialNo (Short Text)

11.MedicalHistory (Short Text)

12.Prescription (Short Text)

13.PatientPhoto (Attachment)

i will work on my database from now on. I will first work in CRUD functionalities of Patient records, Here are the fields in my Access Query for Patient.

Query for Patient Fields:

1.PatientID (AutoNumber)

2.LastName (Short Text)

3.FirstName (Short Text)

4.MiddleName (Short Text)

5. PurokNumber (Short Text)

6. Age (Number)

7. Sex (Short Text)

8.BirthDate (Date/Time)

9.ContactNumber (Short Text)

10. HouseholdSerialNo (Short Text)

11.MedicalHistory (Short Text)

12.Prescription (Short Text)

13.PatientPhoto (Attachment)

I want you to help me set up first for the Create. In my system I have main menu such as; btnHome, btnPatient, btnInventory, btnAnalytics, and btnHealthBud(AI using API). The btnPatient and btnInventory has submenu which are dropdown when each main menu is clicked. All of this main menu are inside my “DashboardForm” and each time user clicks main menu buttons it will just call the user control of corresponding main menu and even in submenu(for btnPatient and btnInventory)

Lets focus first for implementing CRUD in btnPatient using System.Data.Oledb nuget. Where to start coding for CRUD operations of this, will it be in the submenu of btnPatient? Its submmenu are btnNewPatient(Create/Add patient) and btnManagePatient(Read/Update/Delete Patients). Tell me if you need to see the code in both btnPatient submenu for you to be guided. I want this CRUD implementation to be clean considering OOP principles that are applicable in my project for the codes to be neat-looking. Just answer me directly based on my questions here.

ok. now lets get back to the PatientRepository.cs you have implemented(I already copy pasted it to my project). For btnNewPatient you said that i will Create a form that collects all the patient details. All of the main menu are inside my “DashboardForm” and each time user clicks main menu buttons it will just call the user control of corresponding main menu and even in submenu(for btnPatient and btnInventory). Now when user clicks the btnNewPatient it will call the user control named NewPatientControl.cs and there the user can input necessary patient information, database will be displayed but it is in readonly mode so it cant be modified, also this user control has "flowPatientPanel" that will show a user control(has placeholders so the informations can be changed) based on the newly added patients on the database. The code in NewPateintControl is attached below:

Now how am I going to implement the CRUD for btnPatient? tell me if you need additional info just like the user control for btnManagePatient and its code inside

UPDATED PROMPT AS OF April 2, 2025. 8PM  
  
For context I am working with my Smart Electronic Health Record System in OOP2 using C#.

This is the data that the system will ask from user(admin healthcare provider);

In adding Patient: 1.PatientID (this is autogenerated by the database)

2.Last Name

3.First Name

4.Middle Name

5. Purok No.

6. Age

7. Sex

8.Birth Date

9.Contact Number

10. Household Serial Number

11.Medical History

12.Prescription

13.Patient Photo(Optional, if user do not want to put photo the system has reserved default Patient photo)

In adding a drug:

1. DrugID

2. Drug Name

3. Generic Name

4. Brand Name

5.Date Added (This is autogenerated by the system according to present date)

6. Expiry Date

7. Manufacturing Date

8. Category

9. Strength

10. Measurement Type

11. Quantity

12. Drug Photo (Optional, if user do not want to put photo the system has reserved default photo)

These information are  the to be showed also in all CRUD functionalities. Also make a query guide for me, show what are necessary based on my system’s functionalities. My system can do the following I future;

1. CRUD for Patient Records

2. CRUD for Drug Records

3. Display Low Stock Drug on “Home” by making it as notification to user, if none it dispays “No Low Stock Drug”

4. Display when there is a drug going to be expired tomorrow, if none it will show “No expired drugs”. If the drug has expired it will autoremoved by the system and display on “Home” saying “A drug named drug name here have been removed

5. Show on “Home” the bar graph Current Distribution of Patients in the barangay per Purok

6. Show on “Home” the Line Graph of Drug Trend in barangay based on the drugs prescribed to patients.

7. Show on “Home” the pie graph of Puroks in Entire Barangay with respect to number of patients per purok.

8. Show on “Home” the newly added Patients and Drugs by flow layout panels

9. Operate CRUD for Patients and Drug/Inventory.

10. In “Analytics” it can generate data based on the want of the user, it can be filtered Medicine demand with respect to puroks Age, Sex, Medical History or to Just Generate Overall distribution of patients in the entire barangay with respect to puroks

11. Can Export Data in PDF depending on what data analytics the user wants to export

12. Has AI API called “HealthBud” so that the user will have an ai assistance if needed

Query for Patient Fields: 1.PatientID (AutoNumber) 2.LastName (Short Text) 3.FirstName (Short Text) 4.MiddleName (Short Text) 5. PurokNumber (Short Text) 6. Age (Number) 7. Sex (Short Text) 8.BirthDate (Date/Time) 9.ContactNumber (Short Text) 10. HouseholdSerialNo (Short Text) 11.MedicalHistory (Short Text) 12.Prescription (Short Text) 13.PatientPhoto (Attachment) i will work on my database from now on. I will first work in CRUD functionalities of Patient records, Here are the fields in my Access Query for Patient. Query for Patient Fields: 1.PatientID (AutoNumber) 2.LastName (Short Text) 3.FirstName (Short Text) 4.MiddleName (Short Text) 5. PurokNumber (Short Text) 6. Age (Number) 7. Sex (Short Text) 8.BirthDate (Date/Time) 9.ContactNumber (Short Text) 10. HouseholdSerialNo (Short Text) 11.MedicalHistory (Short Text) 12.Prescription (Short Text) 13.PatientPhoto (Attachment) I want you to help me set up first for the Create. In my system I have main menu such as; btnHome, btnPatient, btnInventory, btnAnalytics, and btnHealthBud(AI using API). The btnPatient and btnInventory has submenu which are dropdown when each main menu is clicked. All of this main menu are inside my “DashboardForm” and each time user clicks main menu buttons it will just call the user control of corresponding main menu and even in submenu(for btnPatient and btnInventory) Lets focus first for implementing CRUD in btnPatient using System.Data.Oledb nuget. Where to start coding for CRUD operations of this, will it be in the submenu of btnPatient? Its submmenu are btnNewPatient(Create/Add patient) and btnManagePatient(Read/Update/Delete Patients). Tell me if you need to see the code in both btnPatient submenu for you to be guided. I want this CRUD implementation to be clean considering OOP principles that are applicable in my project for the codes to be neat-looking. Just answer me directly based on my questions here.

I just created a repository called PatientRepository and this is the current update of my code in the repo: using System; using System.Collections.Generic; using System.Linq; using System.Text; using System.Threading.Tasks; using System.Data; using System.Data.OleDb; namespace SEHRS.Repositories { internal class PatientRepository { private string connectionString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=|DataDirectory|\\SEHRS(db).accdb;"; } } Just answer me directly always  
  
  
If I am to work with CRUD operations in my btnInventory that has submenu btnNewDrug(Add drug) and btnManageInventory(Read, Update, Delete) am I going to create another repository and name it DrugRepository? just answer me directly always  
  
  
  
  
the btnNewPatient is still not functioning correctly. in my datagrid view the database is not load when user clicks the btnNewPatient, it must be loaded in data grid view named "dgvPatientsRecords", displays "PatientQuery" I have created in Access(the query does not have any data yet). the query contained by the table named “PatientName”, “PatientDemograpahic” and “PatientHistory”. The query for Patient has fields:

1.PatientID (AutoNumber)

2.LastName (Short Text)

3.FirstName (Short Text)

4.MiddleName (Short Text)

5. PurokNumber (Short Text)

6. Age (Number)

7. Sex (Short Text)

8.BirthDate (Date/Time)

9.ContactNumber (Short Text)

10. HouseholdSerialNo (Short Text)

11.MedicalHistory (Short Text)

12.Prescription (Short Text)

13.PatientPhoto (Attachment)

Help me for "Create" functionality, make your directions clear and straightforward, if you need more additional information such as code or the functionalities of the form just tell me

Error message says:

Error loading patient database: Could not find file 'C:OOP2 SecondSem\FinalProject - WinForms\SEHRS(LatestUpdate)\bin\Debug\net8.0-Windows\SEHRS(db).accdb'.

Do the connection string for me, this is my database file path:

C:\OOP2 SecondSem\FinalProject - WinForms\SEHRS(LatestUpdate)\(Database)

this is the name of the database:

SEHRS(db)

private void btnConTest\_Click(object sender, EventArgs e)

Healthcare Provider(this is the only user/admin of system)

-LastName(String)

-FirstName(String)  
-MiddleName(String)  
-ContactNumber(String)  
  
PatientQuery

-PatienID(AutoNumber(Primary Key) on table PatientName but Number to tables PatientDemographic and PatientHistory)  
-LastName(Short Text)

-FirstName(Short Text)  
-MiddleName(Short Text)  
-PurokNumber (Short Text)  
-Age(Number)  
-Sex(Short Text)  
-BirthDate(Date/Time)  
-ContactNumber(Short Text)  
-HouseholdSerialNo(Short Text)  
-MedicalHistory(Short Text)  
-Prescription(Short Text)  
-PatientPhoto(OlE Object)

Inventory  
-DrugID(AutoNumber(Primary key) in DrugName table Number to tables DrugDates and DrugDetail)  
-DrugName (Short Text)  
-GenericName (Short Text)  
-BrandName (Short Text)  
-DateAdded(Date/Time)  
-ExpiryDate(Date/Time)  
-ManufacturingDate(Date/Time)  
-Category(Short Text)  
-Strength(Short Text)  
-MeasurementType(Short Text)  
- Quantity(Number)  
-DrugPhoto(OLE object)

DrugName table composed of

-DrugID(Primary Key)  
-DrugName (Short Text)  
-GenericName (Short Text)  
-BrandName (Short Text)  
  
DrugDates table composed of

-DrugID(Foreign Key)  
-DateAdded(Date/Time)  
-ExpiryDate(Date/Time)  
-ManufacturingDate(Date/Time)

DrugDetail table composed of

-DrugID(Primary Key)  
-Category(Short Text)  
-Strength(Short Text)  
-MeasurementType(Short Text)  
- Quantity(Number)  
-DrugPhoto(OLE object)  
  
  
I do not yet implemented the full CRUD in my system Smart Electronic Health Record System C# winforms but I want you to help me organize my code so it will look neat specially prioritizing to implement inheritance if there is necessary, creating necessary methods or class for future full CRUD functionality. I will send you my code in 2 forms and 9 user controls simultaneously for you to take note of these all and I will just say “DONE” when I am done pasting. After that I will be asking for assistance in implementing CRUD.  
  
Now what i want you to help me is by implementing possible OOP principles applicable in my project like inheritance, interface, polymophism, abstraction, and others because our teacher is gonna check these. What do you think are we going to implement first? Lets go for inheritance?  
  
I am not yet done with my CRUD implementations. Some data are temporary or those hardcoded patients and drugs. I am having a hard time understanding you what to implement inheritance first  
  
Take not that i am still note done with my CRUD operation for patient and drug, some data are hardcoded. What could be the best way to start? I want to prioritize inheritance and when methods are needed in the future but cannot be integrated yet due to incompleteness of my code just create the method blank and add comments for future use and for what.

namespace SEHRS

{

public partial class LoginForm : BaseForm

{

public LoginForm()

{

InitializeComponent();

this.lblErrorMessage.BringToFront();

if (DesignMode)

{

this.lblErrorMessage.Visible = true; // Visible in designer

}

else

{

this.lblErrorMessage.Visible = false; // Hidden at runtime

}

}

public DashboardForm Opens

{

get => default;

set

{

}

}

private void tbxUsername\_Click(object sender, EventArgs e)

{

}

private void tbxPassword\_Click(object sender, EventArgs e)

{

}

private void AuthenticateUser(string username, string password)

{

string correctUsername = "1";

string correctPassword = "1";

if (username == correctUsername && password == correctPassword)

{

this.DialogResult = DialogResult.OK; // Set login success result

this.Close(); // Close LoginForm

}

else

{

lblErrorMessage.Text = "Invalid username or password!";

lblErrorMessage.Visible = true;

}

}

private void btnLogin\_Click(object sender, EventArgs e)

{

lblErrorMessage.Visible = false; // Hide previous errors

if (string.IsNullOrWhiteSpace(tbxUsername.Text) || string.IsNullOrWhiteSpace(tbxPassword.Text))

{

lblErrorMessage.Text = "Username and Password are required!";

lblErrorMessage.Visible = true;

return;

}

// Call AuthenticateUser (this must be defined)

AuthenticateUser(tbxUsername.Text, tbxPassword.Text);

}

// private void LoginForm\_Load(object sender, EventArgs e)

// {

// pnlContainer.Left = (this.ClientSize.Width - pnlContainer.Width) / 2;

//pnlContainer.Top = (this.ClientSize.Height - pnlContainer.Height) / 2;

// }

private void pnlLogin\_Paint(object sender, PaintEventArgs e)

{

}

}

}

First i will send you reference because I am already done for Create in CRUD in managing patients but now i want to manage CRUD for Drug Inventory. I will send you first my code for NewPatientControl and PatientProfileControl so it will become your reference in creating implementations for NewDrugControl and DrugProfileControl. Also i will send you my database attributes and its datatypes. I will just say DONE when I am done copy pasting

Thank you now i am done in my CRUD operations in my Patient Record and Drug Inventory. Now what I want you to help me is to implement my analytics. Here user can generate data in overall distributions of patients per purok and it can also be filtered by generating data according to a specific health condition and it shows the distribution per purok if it has present case.  
  
private void cbxFilterPatientAnalytics\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void SearchCondition\_TextChanged(object sender, EventArgs e)

{

}

private void btnSearchCondition\_Click(object sender, EventArgs e)

{

}

In cbxFilterPatientAnalytics user can select items Medical Condition

Overall Distribution, what i want is when user clicks chose Medical Condition it will show the SearchCondition and btnSearchCondition and let user input a specific medical health condition so that the pltview will now display data according to the filter (per purok). Help me implement this

I want you to help me implement my HomeControl in which when the program runs this is the interface the user will see In my HomeControl I want to display the overall distribution of patients by purok into HomeControls pltHomeViewOverall and any Specific Medical condition for HomeControl pltHomeViewSpecific. Now help me what is the clean option to this for me to not code it over and over. Here are the current codes for your idea:

using ReaLTaiizor.Controls;

using System;

using System.Collections.Generic;

using System.Data.OleDb;

using System.Diagnostics;

using System.Drawing.Drawing2D;

using System.Globalization;

using System.Windows.Forms;

namespace SEHRS

{

public partial class ManagePatientsControl : UserControl

{

public ManagePatientsControl()

{

InitializeComponent();

DisplayPatientProfiles();

RoundPictureBox(picBoxPatientOutput);

// Set initial states

ResetPanelColors();

pnlViewPatient.BackColor = Color.FromArgb(255, 224, 192); // Start in view mode

SetFieldsReadOnly(true); // Start with fields read-only

}

public void DisplayPatientProfiles()

{

PatientProfileService.LoadAndDisplayPatients(flowPatientProfilePanel);

}

private void LoadAndDisplayPatients(string searchTerm = "")

{

try

{

flowPatientProfilePanel.SuspendLayout();

flowPatientProfilePanel.Controls.Clear();

string query = "SELECT \* FROM PatientQuery";

var parameters = new List<OleDbParameter>();

if (!string.IsNullOrEmpty(searchTerm))

{

query += " WHERE FirstName LIKE ? OR LastName LIKE ? OR PatientID LIKE ?";

parameters.Add(new OleDbParameter("@term1", $"%{searchTerm}%"));

parameters.Add(new OleDbParameter("@term2", $"%{searchTerm}%"));

parameters.Add(new OleDbParameter("@term3", $"%{searchTerm}%"));

}

using (OleDbConnection conn = new OleDbConnection(GetConnectionString()))

using (OleDbCommand cmd = new OleDbCommand(query, conn))

{

cmd.Parameters.AddRange(parameters.ToArray());

conn.Open();

using (OleDbDataReader reader = cmd.ExecuteReader())

{

while (reader.Read())

{

var profile = PatientProfileService.CreatePatientProfile(reader);

// Lightweight event handler

profile.ProfileDoubleClicked += (sender, e) =>

{

if (InvokeRequired)

{

Invoke(new Action(() => DisplayPatientDetails(((PatientProfileControl)sender).Tag)));

}

else

{

DisplayPatientDetails(((PatientProfileControl)sender).Tag);

}

};

flowPatientProfilePanel.Controls.Add(profile);

}

}

}

}

catch (Exception ex)

{

MessageBox.Show($"Error loading patients: {ex.Message}");

}

finally

{

flowPatientProfilePanel.ResumeLayout(true);

Application.DoEvents();

}

}

// Method to load patients into the flowPatientPanel

public void LoadPatients(List<(string Name, string ID, string DOB)> patients)

{

if (flowPatientProfilePanel == null)

throw new InvalidOperationException("flowPatientPanel is not initialized.");

flowPatientProfilePanel.Controls.Clear(); // Clear existing controls to prevent duplicates

foreach (var patient in patients)

{

// Create a new PatientProfileControl

PatientProfileControl profileControl = new PatientProfileControl

{

PatientName = patient.Name,

PatientBirthDate = patient.DOB,

PatientID = patient.ID

};

// Add the PatientProfileControl to the flowPatientPanel

flowPatientProfilePanel.Controls.Add(profileControl);

}

}

// Add these methods anywhere in your ManagePatientsControl class (good practice is near the top with other private methods)

private string GetConnectionString()

{

return @"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=""C:\OOP2 SecondSem\FinalProject - WinForms\SEHRS(LatestUpdate)\SEHRS\Resources\SEHRSDB\SEHRS(db).accdb""";

}

private Image ByteArrayToImage(byte[] imageBytes)

{

using (MemoryStream ms = new MemoryStream(imageBytes))

{

return Image.FromStream(ms);

}

}

private void OnPatientProfileDoubleClick(object sender, EventArgs e, PatientProfileControl profile)

{

dynamic data = profile.Tag;

tbxFNameOutput.Text = data.FirstName;

tbxLNameOutput.Text = data.LastName;

tbxMNameOutput.Text = data.MiddleName;

tbxPatientIDNumOutput.Text = data.PatientID;

tbxBirthOutput.Text = ((DateTime)data.BirthDate).ToString("dd/MM/yyyy");

tbxPurokNumOutput.Text = data.PurokNumber;

tbxAgeOutput.Text = data.Age;

tbxSexOutput.Text = data.Sex;

tbxContactOutput.Text = data.ContactNumber;

tbxSerNumOutput.Text = data.HouseholdSerialNo;

tbxMedicalHistoryOutput.Text = data.MedicalHistory;

tbxPrescriptionOutput.Text = data.Prescription;

if (data.PatientPhoto != null)

{

picBoxPatientOutput.Image = PatientProfileService.ByteArrayToImage(data.PatientPhoto);

}

else

{

picBoxPatientOutput.Image = CreateDefaultProfileImage(); // fallback method you already have

}

}

private PatientProfileControl CreatePatientProfile(OleDbDataReader reader)

{

var profile = new PatientProfileControl

{

FirstName = reader["FirstName"].ToString(),

LastName = reader["LastName"].ToString(),

MiddleName = reader["MiddleName"].ToString(),

PatientID = reader["PatientID"].ToString(),

PurokNumber = reader["PurokNumber"].ToString(),

Age = reader["Age"].ToString(),

Sex = reader["Sex"].ToString(),

ContactNumber = reader["ContactNumber"].ToString(),

HouseholdSerialNo = reader["HouseholdSerialNo"].ToString(),

MedicalHistory = reader["MedicalHistory"].ToString(),

Prescription = reader["Prescription"].ToString(),

PatientName = $"{reader["FirstName"]} {reader["LastName"]}",

PatientBirthDate = Convert.ToDateTime(reader["BirthDate"]).ToString("dd/MM/yyyy")

};

if (reader["PatientPhoto"] != DBNull.Value)

{

byte[] imageBytes = (byte[])reader["PatientPhoto"];

profile.PatientPhoto = ByteArrayToImage(imageBytes);

}

// Attach the double click event handler

profile.DoubleClick += (sender, e) =>

{

var clickedProfile = (PatientProfileControl)sender;

DisplayPatientDetails(clickedProfile);

// Debug output

Console.WriteLine($"Double-clicked: {clickedProfile.FirstName} {clickedProfile.LastName}");

Console.WriteLine($"Setting tbxFNameOutput to: {clickedProfile.FirstName}");

};

return profile;

}

private void DisplayPatientDetails(dynamic patientData)

{

try

{

// Use BeginInvoke for thread safety

if (InvokeRequired)

{

BeginInvoke(new Action(() => DisplayPatientDetails(patientData)));

return;

}

// Basic Info

tbxFNameOutput.Text = patientData.FirstName;

tbxLNameOutput.Text = patientData.LastName;

tbxMNameOutput.Text = patientData.MiddleName;

tbxPatientIDNumOutput.Text = patientData.PatientID;

tbxBirthOutput.Text = patientData.BirthDate.ToString("dd/MM/yyyy");

// Demographic Info

tbxPurokNumOutput.Text = patientData.PurokNumber;

tbxAgeOutput.Text = patientData.Age;

tbxSexOutput.Text = patientData.Sex;

tbxContactOutput.Text = patientData.ContactNumber;

tbxSerNumOutput.Text = patientData.HouseholdSerialNo;

// Medical Info

tbxMedicalHistoryOutput.Text = patientData.MedicalHistory;

tbxPrescriptionOutput.Text = patientData.Prescription;

// Photo - handle in background to prevent UI freeze

Task.Run(() =>

{

Image photo = null;

if (patientData.PatientPhoto != null)

{

photo = PatientProfileService.ByteArrayToImage(patientData.PatientPhoto);

}

else

{

photo = CreateDefaultProfileImage();

}

if (picBoxPatientOutput.InvokeRequired)

{

picBoxPatientOutput.Invoke(new Action(() => picBoxPatientOutput.Image = photo));

}

else

{

picBoxPatientOutput.Image = photo;

}

});

}

catch (Exception ex)

{

MessageBox.Show($"Error displaying patient details: {ex.Message}");

}

}

private Image CreateDefaultProfileImage()

{

Bitmap bmp = new Bitmap(100, 100);

using (Graphics g = Graphics.FromImage(bmp))

{

g.Clear(Color.LightGray);

using (Font font = new Font("Arial", 20))

{

g.DrawString("No Image", font, Brushes.White, new PointF(10, 35));

}

}

return bmp;

}

private void cyberButton1\_Click(object sender, EventArgs e) { }

private void panel7\_Paint(object sender, PaintEventArgs e) { }

private void bigLabel1\_Click(object sender, EventArgs e) { }

private void panel13\_Paint(object sender, PaintEventArgs e) { }

private void flowPatientPanel\_Paint(object sender, PaintEventArgs e) { }

private void btnSearchPatientMode\_Click(object sender, EventArgs e)

{

// Get search term and load matching patients

string searchTerm = tbxSearchPatientMode.Text.Trim();

LoadAndDisplayPatients(searchTerm);

}

private void btnChangePhoto\_Click(object sender, EventArgs e)

{

try

{

using (OpenFileDialog openFileDialog = new OpenFileDialog())

{

openFileDialog.Title = "Select Patient Photo";

openFileDialog.Filter = "Image Files|\*.jpg;\*.jpeg;\*.png;\*.gif;\*.bmp";

openFileDialog.Multiselect = false;

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

// Dispose current image if it exists

if (picBoxPatientOutput.Image != null)

{

picBoxPatientOutput.Image.Dispose();

}

// Load the selected image directly into the PictureBox

picBoxPatientOutput.Image = Image.FromFile(openFileDialog.FileName);

}

}

}

catch (Exception ex)

{

MessageBox.Show("Error uploading photo: " + ex.Message, "Upload Error",

MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void RoundPictureBox(PictureBox picBox)

{

GraphicsPath path = new GraphicsPath();

path.AddEllipse(0, 0, picBox.Width, picBox.Height);

picBox.Region = new Region(path);

}

private void picBoxOutput\_Click(object sender, EventArgs e)

{

}

private void tbxFNameOutput\_Click(object sender, EventArgs e)

{

}

private void SetFieldsReadOnly(bool readOnly)

{

tbxFNameOutput.ReadOnly = readOnly;

tbxLNameOutput.ReadOnly = readOnly;

tbxMNameOutput.ReadOnly = readOnly;

tbxPatientIDNumOutput.ReadOnly = true; // ID should never be editable

tbxBirthOutput.ReadOnly = readOnly;

tbxAgeOutput.ReadOnly = readOnly;

tbxSexOutput.ReadOnly = readOnly;

tbxPurokNumOutput.ReadOnly = readOnly;

tbxSerNumOutput.ReadOnly = readOnly;

tbxContactOutput.ReadOnly = readOnly;

tbxMedicalHistoryOutput.ReadOnly = readOnly;

tbxPrescriptionOutput.ReadOnly = readOnly;

// Enable/disable photo change button based on mode

btnChangePatientPhoto.Enabled = !readOnly;

}

private void btnPatientHistoryMode\_Click(object sender, EventArgs e)

{

ResetPanelColors();

pnlViewPatient.BackColor = Color.FromArgb(255, 224, 192);

// Enable search controls

tbxSearchPatientMode.Enabled = true;

btnSearchPatientMode.Enabled = true;

btnSaveEditPatient.Visible = false; // Hide save button in view mode

btnDeletePatientRecord.Visible = false; // Hide delete button in view mode

// Make fields read-only

SetFieldsReadOnly(true);

// Focus on search textbox

tbxSearchPatientMode.Focus();

// LoadAndDisplayPatients();

}

private void btnEditPatientRecMode\_Click(object sender, EventArgs e)

{

ResetPanelColors();

pnlEditPatient.BackColor = Color.FromArgb(255, 224, 192);

// Enable search controls

tbxSearchPatientMode.Enabled = true;

btnSearchPatientMode.Enabled = true;

btnSaveEditPatient.Visible = true;

btnDeletePatientRecord.Visible = false;

// Make fields editable

SetFieldsReadOnly(false);

// Focus on search textbox

tbxSearchPatientMode.Focus();

}

private void btnDeletePatientRecMode\_Click(object sender, EventArgs e)

{

ResetPanelColors();

pnlDeletePatient.BackColor = Color.FromArgb(255, 224, 192);

// Enable search controls

tbxSearchPatientMode.Enabled = true;

btnSearchPatientMode.Enabled = true;

btnSaveEditPatient.Visible = false;

btnDeletePatientRecord.Visible = true;

// Make fields read-only

SetFieldsReadOnly(true);

// Focus on search textbox

tbxSearchPatientMode.Focus();

}

private void tbxSearchPatientMode\_Click(object sender, EventArgs e)

{

string searchTerm = tbxSearchPatientMode.Text.Trim(); // Fixed the control name here

if (string.IsNullOrEmpty(searchTerm))

{

MessageBox.Show("Please enter a name or ID to search.");

return;

}

LoadAndDisplayPatients(searchTerm);

}

private void btnSearchPatientProfile\_Click(object sender, EventArgs e)

{

}

private void ResetPanelColors()

{

pnlViewPatient.BackColor = SystemColors.Control; // Default color

pnlEditPatient.BackColor = SystemColors.Control;

pnlDeletePatient.BackColor = SystemColors.Control;

}

private void btnSaveEditPatient\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(tbxPatientIDNumOutput.Text))

{

MessageBox.Show("No patient selected to update.", "Error",

MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

var result = MessageBox.Show("Are you sure you want to save changes to this patient record?",

"Confirm Update",

MessageBoxButtons.YesNo,

MessageBoxIcon.Question);

if (result == DialogResult.Yes)

{

try

{

// Clean the PatientID by removing any non-numeric characters

string cleanPatientId = new string(tbxPatientIDNumOutput.Text.Where(char.IsDigit).ToArray());

if (string.IsNullOrEmpty(cleanPatientId))

{

MessageBox.Show("Invalid Patient ID format", "Error",

MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

int patientId = Convert.ToInt32(cleanPatientId);

// Parse birth date in dd/MM/yyyy format

if (!DateTime.TryParseExact(tbxBirthOutput.Text, "dd/MM/yyyy",

CultureInfo.InvariantCulture, DateTimeStyles.None, out DateTime birthDate))

{

MessageBox.Show("Please enter a valid date in dd/MM/yyyy format", "Invalid Date",

MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Convert age to integer

if (!int.TryParse(tbxAgeOutput.Text, out int age))

{

MessageBox.Show("Please enter a valid age (numeric value)", "Invalid Age",

MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

string updateQuery = @"UPDATE PatientQuery SET

FirstName = ?,

LastName = ?,

MiddleName = ?,

BirthDate = ?,

PurokNumber = ?,

Age = ?,

Sex = ?,

ContactNumber = ?,

HouseholdSerialNo = ?,

MedicalHistory = ?,

Prescription = ?,

PatientPhoto = ?

WHERE PatientID = ?";

using (OleDbConnection conn = new OleDbConnection(GetConnectionString()))

using (OleDbCommand cmd = new OleDbCommand(updateQuery, conn))

{

// Add parameters with explicit data types

cmd.Parameters.Add("@FirstName", OleDbType.VarWChar).Value = tbxFNameOutput.Text;

cmd.Parameters.Add("@LastName", OleDbType.VarWChar).Value = tbxLNameOutput.Text;

cmd.Parameters.Add("@MiddleName", OleDbType.VarWChar).Value = tbxMNameOutput.Text;

cmd.Parameters.Add("@BirthDate", OleDbType.DBDate).Value = birthDate;

cmd.Parameters.Add("@PurokNumber", OleDbType.VarWChar).Value = tbxPurokNumOutput.Text;

cmd.Parameters.Add("@Age", OleDbType.Integer).Value = age;

cmd.Parameters.Add("@Sex", OleDbType.VarWChar).Value = tbxSexOutput.Text;

cmd.Parameters.Add("@ContactNumber", OleDbType.VarWChar).Value = tbxContactOutput.Text;

cmd.Parameters.Add("@HouseholdSerialNo", OleDbType.VarWChar).Value = tbxSerNumOutput.Text;

cmd.Parameters.Add("@MedicalHistory", OleDbType.LongVarWChar).Value = tbxMedicalHistoryOutput.Text;

cmd.Parameters.Add("@Prescription", OleDbType.LongVarWChar).Value = tbxPrescriptionOutput.Text;

// Handle photo

byte[] photoBytes = ImageToByteArray(picBoxPatientOutput.Image);

if (photoBytes != null && photoBytes.Length > 0 &&

!picBoxPatientOutput.Image.Equals(CreateDefaultProfileImage()))

{

cmd.Parameters.Add("@PatientPhoto", OleDbType.LongVarBinary).Value = photoBytes;

}

else

{

cmd.Parameters.Add("@PatientPhoto", OleDbType.LongVarBinary).Value = DBNull.Value;

}

cmd.Parameters.Add("@PatientID", OleDbType.Integer).Value = patientId;

conn.Open();

int rowsAffected = cmd.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Patient record updated successfully!", "Success",

MessageBoxButtons.OK, MessageBoxIcon.Information);

LoadAndDisplayPatients();

}

else

{

MessageBox.Show("No records were updated.", "Warning",

MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

}

catch (Exception ex)

{

MessageBox.Show($"Error updating patient record: {ex.Message}", "Error",

MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

// Helper method to convert Image to byte array

private byte[] ImageToByteArray(Image image)

{

try

{

// Check if the image is null

if (image == null)

{

return null;

}

// Create a new bitmap to avoid GDI+ errors (this also avoids issues with disposed images)

using (Bitmap bmp = new Bitmap(image))

using (MemoryStream ms = new MemoryStream())

{

// Save in PNG format which is more reliable than JPEG

bmp.Save(ms, System.Drawing.Imaging.ImageFormat.Png);

return ms.ToArray();

}

}

catch (Exception ex)

{

Console.WriteLine($"Image conversion error: {ex.Message}");

return null;

}

}

private void tbxMNameOutput\_Click(object sender, EventArgs e)

{

}

private void tbxLNameOutput\_Click(object sender, EventArgs e)

{

}

private void tbxContactOutput\_TextChanged(object sender, EventArgs e)

{

}

private void tbxSerNumOutput\_TextChanged(object sender, EventArgs e)

{

}

private void tbxSexOutput\_Click(object sender, EventArgs e)

{

}

private void tbxAgeOutput\_Click(object sender, EventArgs e)

{

}

private void tbxPurokNumOutput\_Click(object sender, EventArgs e)

{

}

private void tbxPatientIDNumOutput\_Click(object sender, EventArgs e)

{

}

private void tbxMedicalHistoryOutput\_Click(object sender, EventArgs e)

{

}

private void tbxPrescriptionOutput\_Click(object sender, EventArgs e)

{

}

private void btnDeletePatientRecord\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(tbxPatientIDNumOutput.Text))

{

MessageBox.Show("No patient selected to delete.", "Error",

MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

var result = MessageBox.Show("Are you sure you want to permanently delete this patient record?",

"Confirm Deletion",

MessageBoxButtons.YesNo,

MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

try

{

// Clean the PatientID by removing any non-numeric characters

string cleanPatientId = new string(tbxPatientIDNumOutput.Text.Where(char.IsDigit).ToArray());

if (string.IsNullOrEmpty(cleanPatientId))

{

MessageBox.Show("Invalid Patient ID format", "Error",

MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

int patientId = Convert.ToInt32(cleanPatientId);

using (OleDbConnection conn = new OleDbConnection(GetConnectionString()))

{

conn.Open();

// Delete from PatientHistory first (due to potential foreign key constraints)

string deleteHistoryQuery = "DELETE FROM PatientHistory WHERE PatientID = ?";

using (OleDbCommand cmd = new OleDbCommand(deleteHistoryQuery, conn))

{

cmd.Parameters.Add("@PatientID", OleDbType.Integer).Value = patientId;

cmd.ExecuteNonQuery();

}

// Delete from PatientDemographic

string deleteDemographicQuery = "DELETE FROM PatientDemographic WHERE PatientID = ?";

using (OleDbCommand cmd = new OleDbCommand(deleteDemographicQuery, conn))

{

cmd.Parameters.Add("@PatientID", OleDbType.Integer).Value = patientId;

cmd.ExecuteNonQuery();

}

// Finally delete from PatientName

string deleteNameQuery = "DELETE FROM PatientName WHERE PatientID = ?";

using (OleDbCommand cmd = new OleDbCommand(deleteNameQuery, conn))

{

cmd.Parameters.Add("@PatientID", OleDbType.Integer).Value = patientId;

int rowsAffected = cmd.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Patient record deleted successfully!", "Success",

MessageBoxButtons.OK, MessageBoxIcon.Information);

// Clear form fields and refresh the patient list

ClearFormFields();

LoadAndDisplayPatients();

}

else

{

MessageBox.Show("No records were deleted. Patient may not exist.", "Warning",

MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

}

}

catch (Exception ex)

{

MessageBox.Show($"Error deleting patient record: {ex.Message}", "Error",

MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

private void ClearFormFields()

{

// Clear all text fields

tbxFNameOutput.Clear();

tbxLNameOutput.Clear();

tbxMNameOutput.Clear();

tbxPatientIDNumOutput.Clear();

tbxBirthOutput.Clear();

tbxAgeOutput.Clear();

tbxSexOutput.Clear();

tbxPurokNumOutput.Clear();

tbxSerNumOutput.Clear();

tbxContactOutput.Clear();

tbxMedicalHistoryOutput.Clear();

tbxPrescriptionOutput.Clear();

// Reset photo to default

picBoxPatientOutput.Image = CreateDefaultProfileImage();

}

private void pnlViewPatient\_Paint(object sender, PaintEventArgs e)

{

}

private void pnlEditPatient\_Paint(object sender, PaintEventArgs e)

{

}

private void pnlDeletePatient\_Paint(object sender, PaintEventArgs e)

{

}

}

}

Search key letters

Show last date visited

SEPHOMORE

TWILIO:  
ACC SID: AC6c7f9d2d8dc6ba2f99900379adc692c2  
AUTH TOKEN: 907e5dc301c9181b164e06b07c571885  
Phone number: +17012039137  
  
  
  
VONAGE:  
API key: 61876d35  
API secret 1: jtRHFXsfGd1zEQam