Analysis Paralysis Group 21 Code dump

Table of Contents

[Main Form: 1](#_Toc493584359)

[New Booking Form: 17](#_Toc493584360)

[Finalize Booking 32](#_Toc493584361)

[Admin Form 41](#_Toc493584362)

[Login Form 68](#_Toc493584363)

[User Control Form 71](#_Toc493584364)

[Client Management Form 76](#_Toc493584365)

[Change Email Messages Form 81](#_Toc493584366)

[Edit Email Settings Form 83](#_Toc493584367)

[Report Viewer 85](#_Toc493584368)

# Main Form:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using System.Data.OleDb;

using Chalique\_Nail\_Studio.Properties;

namespace Chalique\_Nail\_Studio

{

public partial class frmMain : Form

{

// Instance variables

private const string OWNER1 = "Chantenique: 084 557 3751";

private const string OWNER2 = "Angelique: 074 992 7807";

private OleDbConnection conn; // Database connection

private string DB = @"Provider=Microsoft.ACE.OLEDB.12.0; Data Source = ";

private string DBloc, target; // Location variables

private bool backup; // Flag to backup database and settings

private bool Loading; // Flag to prevent method triggers while loading

private string bg = ""; // Background setting

private bool loginFail; // Flag for login failure, prevents operation continuance

private int currentBooking; // Booking currently selected, used to display, edit, delete and finalise bookings

private string cName, bDate; // Client name and booking date

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// ======================= FORM LOAD AND CLOSE EVENTS ====================== //

public frmMain()

{//------------------------------------------------------------------------------------------------------------

DBloc = Directory.GetCurrentDirectory();

if (!File.Exists(DBloc + @"\ChaliqueNailStudio.accdb"))

DBloc = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + @"\Chalique Nail Studio";

if (!File.Exists(DBloc + @"\ChaliqueNailStudio.accdb"))

{

try

{

if (MessageBox.Show("The database file cannot be found. Do you want to select a backed up file?", "Database Not Found", MessageBoxButtons.YesNo) == DialogResult.Yes)

{

OpenFileDialog databasePicker = new OpenFileDialog();

databasePicker.Title = "Select the Database file";

databasePicker.Filter = "Access Database files (\*.accdb)|\*.accdb";

databasePicker.InitialDirectory = @"C:\";

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

{

File.Copy(databasePicker.FileName, DBloc + @"\ChaliqueNailStudio.accdb", true);

}

else throw new Exception();

}

else throw new Exception();

}

catch (Exception)

{

MessageBox.Show("The program cannot run without the database. Please contact an administrator to restore the database", "Fatal error");

Environment.Exit(0);

}

}

DB += DBloc + @"\ChaliqueNailStudio.accdb";

conn = new OleDbConnection(DB);

// Initialise login form

loginFail = false;

Login login = new Login(conn);

login.ShowDialog();

if (login.Canceled)

{

// Exit immediately if login process is canceled on startup

loginFail = true;

Environment.Exit(0);

}

InitializeComponent();

}

private void frmMain\_Shown(object sender, EventArgs e)

{

if (loginFail)

{

this.Close();

}

else

{

// Prevents events being triggered during the loading process

Loading = true;

DateRangeEventArgs g = new DateRangeEventArgs(dateSelect.TodayDate, dateSelect.TodayDate);

dateSelect\_DateSelected(dateSelect, g);

// if settings file does not exist, create a new one

if (!File.Exists(DBloc + @"\settings.cns"))

{

StreamWriter w = File.CreateText(DBloc + @"\settings.cns");

w.Write("DB request backup ;true\r\nDB backup location ;\r\nSelected background; 2");

w.Close();

}

// Load the settings

StreamReader r = File.OpenText(DBloc + @"\settings.cns");

string[] temp = r.ReadLine().Split(';');

backup = Convert.ToBoolean(temp[1]);

if ((temp = r.ReadLine().Split(';'))[1] != "")

target = temp[1];

if ((temp = r.ReadLine().Split(';'))[1] != "")

switch (int.Parse(temp[1]))

{

case 1: purpleToolStripMenuItem\_Click(sender, e);

break;

case 2: pinkToolStripMenuItem\_Click(sender, e);

break;

case 3: pink2ToolStripMenuItem\_Click(sender, e);

break;

default: break;

}

folderBrowserDialog1.SelectedPath = DBloc;

// Checks backup setting and backup location.

if (backup)

if (target == null || target == "" || !Directory.Exists(target))

if (MessageBox.Show("No backup location is chosen. Do you wish to choose a backup location?", "Backup", MessageBoxButtons.YesNo) == DialogResult.Yes)

if (folderBrowserDialog1.ShowDialog() == DialogResult.OK)

{

target = folderBrowserDialog1.SelectedPath;

r.Close();

Loading = false;

writeSettings();

}

r.Close();

Loading = false; // Loading completed, processes continue as normal

}

}

private void frmMain\_FormClosing(object sender, FormClosingEventArgs e)

{

if (backup)

backupDB(); //Do this also on a daily basis

writeSettings();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Write the settings to the settings file

private void writeSettings()

{

if (!Loading)

{

StreamWriter w = File.CreateText(DBloc + @"\settings.cns");

if (backup)

{

w.Write("DB request backup ;true" + "\r\n");

if (Directory.Exists(target))

w.Write("DB backup location ;" + target + "\r\n");

else

w.Write("DB backup location ;\r\n");

}

else

{

w.Write("DB backup location ;false" + "\r\n");

w.Write("DB backup location ;\r\n");

}

w.Write("Selected background ;" + bg);

w.Close();

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Create a backup of the database

private void backupDB()

{

if (target == "" || target == null)

target = DBloc;

File.Copy(DBloc + @"\ChaliqueNailStudio.accdb", target + @"\ChaliqueNailStudio.Backup.accdb", true);

if (File.Exists(DBloc + @"\logfile.txt"))

File.Copy(DBloc + @"\logfile.txt", target + @"\logfile.Backup.txt", true);

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Refresh the bookings table

private void bookingsRefresh\_Tick(object sender, EventArgs e)

{

LoadBookings();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// =========================== MENU STRIP EVENTS =========================== //

//\*\*\*\*\*=================================================================\*\*\*\*\*//

// ======================== BACKGROUND CHANGE EVENTS ======================= //

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

{

this.BackgroundImage = Resources.Light\_Purple;

button\_colour(Color.Thistle);

bg = "1";

writeSettings();

}

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

{

this.BackgroundImage = Resources.Bright\_Pink;

button\_colour(Color.Pink);

bg = "2";

writeSettings();

}

private void pink2ToolStripMenuItem\_Click(object sender, EventArgs e)

{

this.BackgroundImage = Resources.Light\_Pink;

button\_colour(Color.Pink);

bg = "3";

writeSettings();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Change the button colours to match the theme

private void button\_colour(Color c)

{

txtBookDetails.BackColor = c;

btnCancel.BackColor = c;

btnEditBooking.BackColor = c;

btnNewBooking.BackColor = c;

btnFinish.BackColor = c;

btnClients.BackColor = c;

btnAdmin.BackColor = c;

}

//\*\*\*\*\*=================================================================\*\*\*\*\*//

// ========================= USER MANAGEMENT EVENTS ======================== //

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Open the user management dialog

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

{

// Initialise login form

loginFail = false;

Login login = new Login(conn);

login.ShowDialog();

if (login.Canceled)

{

loginFail = true;

}

if (!loginFail)

{

if (login.acclvl == 1 || login.acclvl == 0)

{

//show the users dialog

User\_control user = new User\_control(conn);

user.ShowDialog();

}

else MessageBox.Show("You do not have access to the User Management system!", "ACCESS DENIED");

}

}

//\*\*\*\*\*=================================================================\*\*\*\*\*//

// ============================== HELP EVENTS ============================== //

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Show the help manual \*\*\*Message box until help manual is compiled

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

{

//helpMethod();

openHelpMenu();

}

private void helpMethod()

{

string help = "First select a date on the calendar.\n" +

"Then select a booking from the list.\n" +

"If there are no bookings in the list you need to create one,\nclick the New Booking button.";

MessageBox.Show(help, "HELP", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

public void openHelpMenu()

{

if (File.Exists(DBloc + @"\Manual\Help Manual.pdf"))

System.Diagnostics.Process.Start(DBloc + @"\Manual\Help Manual.pdf");

else if (File.Exists(DBloc + @"\Manual\Help Manual.docx"))

System.Diagnostics.Process.Start(DBloc + @"\Manual\Help Manual.docx");

else MessageBox.Show("No help file has been found!");

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Message box containing details about the probgram

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

{

string about = "Chalique Nail Studio\nVersion: 1.0\n© 2017 Analysis Paralysis - Group 21\r\n\r\n"

+ "Project developers:\r\n"

+ "Robin de Klerk \t 079 030 3144 \r\n"

+ "Ahmed Yusuf Patel\t 063 448 3161 \r\n"

+ "Nicole Roberts \t 084 571 7752 \r\n"

+ "Franco Verster \t 083 254 1119 \r\n"

+ "Arno Pretorius \t 074 748 4914 \r\n"

+ "Naomi Pretorius \t 060 780 9777";

MessageBox.Show(about, "About Chalique Nails Studio", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Close the program

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

{

this.Close();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Toggle the backup setting

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

{

if (backup)

{

backup = false;

MessageBox.Show("Backups have been disabled", "Toggle Backups");

}

else

{

backup = true;

MessageBox.Show("Backups have been enabled", "Toggle Backups");

}

writeSettings();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Change the Database backup location, can be changed to a removable device

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

{

if (folderBrowserDialog1.ShowDialog() == DialogResult.OK)

{

target = folderBrowserDialog1.SelectedPath;

writeSettings();

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// ========================== LOAD BOOKING EVENTS ========================== //

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Reload the bookings table when selected date changes

private void dateSelect\_DateSelected(object sender, DateRangeEventArgs e)

{

LoadBookings();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Load all the bookings into the bookings table for the 5 days after the date selected

private void LoadBookings()

{

// Display in grid view for selection (Table Layout Panel, tlpBookings)

DateTime date = dateSelect.SelectionRange.Start;

// Remove bookings textboxes from the tablelayoutpanel

if (!Loading)

for (int j = 0; j < 5; j++)

for(int i = 1; i < 20; i++)

{

Control a;

if ((a = tlpBookings.GetControlFromPosition(j, i)) != null)

tlpBookings.Controls.Remove(a);

}

txtDay1.Text = date.ToShortDateString();

RetrieveBookings(date, 0);

date = date.AddDays(1);

txtDay2.Text = date.ToShortDateString();

RetrieveBookings(date, 1);

date = date.AddDays(1);

txtDay3.Text = date.ToShortDateString();

RetrieveBookings(date, 2);

date = date.AddDays(1);

txtDay4.Text = date.ToShortDateString();

RetrieveBookings(date, 3);

date = date.AddDays(1);

txtDay5.Text = date.ToShortDateString();

RetrieveBookings(date, 4);

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Load the bookings for the specified date into the bookings table

private void RetrieveBookings(DateTime date, int col)

{

conn.Open();

int c = 1;

OleDbCommand cmd = new OleDbCommand($@"SELECT \* FROM Bookings WHERE DateValue(BookingDateTime) = #{date.ToShortDateString()}# ORDER BY BookingDateTime ", conn);

OleDbDataReader read = cmd.ExecuteReader();

while (read.Read())

{

OleDbCommand com = new OleDbCommand($@"SELECT \* FROM Clients WHERE ClientID = {read["ClientID"].ToString()}", conn);

OleDbDataReader r = com.ExecuteReader();

r.Read();

string bID, cName, bTime, tech;

int techID;

bID = read["BookingID"].ToString();

cName = r["ClientName"].ToString();

bTime = (Convert.ToDateTime(read["BookingDateTime"].ToString())).TimeOfDay.ToString();

if ((techID = int.Parse(read["EmployeeID"].ToString())) != 0)

{

com = new OleDbCommand($@"SELECT EmployeeName FROM Employees WHERE EmployeeID = {techID}", conn);

r = com.ExecuteReader();

r.Read();

tech = r[0].ToString();

}

else tech = "No preference";

TextBox t1 = new TextBox()

{

Multiline = true,

Text = $"{cName}\r\n{bTime} #{bID}\r\n{tech}",

Name = "b" + bID,

ReadOnly = true,

BackColor = Color.White

};

t1.Height = 46;

t1.Click += new EventHandler(SelectBooking\_Click);

tlpBookings.Controls.Add(t1, col, c++);

}

conn.Close();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// ========================== BUTTON CLICK EVENTS ========================== //

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Open the Admin dialog

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

{

// Initialise login form

loginFail = false;

Login login = new Login(conn);

login.ShowDialog();

if (login.Canceled)

loginFail = true;

if (!loginFail)

{

if (login.acclvl == 1 || login.acclvl == 0)

{

Admin admin = new Admin(conn, login.user, target);

admin.ShowDialog();

}

else MessageBox.Show("You do not have access to the Admin system!", "ACCESS DENIED");

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Open the Clients dialog

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

{

Client\_Management cManagement = new Client\_Management(conn);

cManagement.Show();

LoadBookings();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Open the New booking dialog

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

{//------------------------------------------------------------------------------------------------------------

New\_Booking booking = new New\_Booking(conn);

booking.ShowDialog();

if (booking.Complete)

{

//MessageBox.Show("Booking Successful");

txtBookDetails.Visible = false;

btnEditBooking.Visible = false;

btnCancel.Visible = false;

btnFinish.Visible = false;

// reload bookings grid

LoadBookings();

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Open the New booking dialog, sending the booking to be edited

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

{

// Open the New Booking form, sending the Booking ID

try

{

New\_Booking booking = new New\_Booking(conn, currentBooking);

booking.ShowDialog();

if (booking.Complete)

{

MessageBox.Show("The booking has been successfully changed.");

}

txtBookDetails.Visible = false;

btnEditBooking.Visible = false;

btnCancel.Visible = false;

btnFinish.Visible = false;

}

catch (Exception ex)

{

MessageBox.Show("An error has occured while attempting to edit the booking.\r\n" + ex, "Edit error");

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Delete the selected booking

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

{

if (MessageBox.Show($"Do you want to cancel the booking for {cName} on {bDate} ?", "Confirm cancelation", MessageBoxButtons.YesNo) == DialogResult.Yes)

{

conn.Open();

OleDbCommand cmd = new OleDbCommand($"SELECT \* FROM Payment WHERE BookingID = {currentBooking}", conn);

OleDbDataReader read = cmd.ExecuteReader();

if (!read.Read())

{

try

{

cmd = new OleDbCommand($"DELETE \* FROM TreatmentDetailPerBooking WHERE BookingID = {currentBooking}", conn);

cmd.ExecuteNonQuery();

cmd = new OleDbCommand($"DELETE \* FROM Bookings WHERE BookingID = {currentBooking}", conn);

cmd.ExecuteNonQuery();

conn.Close();

LoadBookings();

MessageBox.Show("The booking has been cancelled!");

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("There was an error while deleting the booking!\r\n" + ex, "Error deleting booking");

}

}

else MessageBox.Show("This booking has already been finalised. Cant Cancel.", "Cancellation error");

}

txtBookDetails.Visible = false;

btnEditBooking.Visible = false;

btnCancel.Visible = false;

btnFinish.Visible = false;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Open the Email settings dialog

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

{

loginFail = false;

Login login = new Login(conn);

login.ShowDialog();

if (login.Canceled)

loginFail = true;

if (!loginFail)

{

if (login.acclvl == 1 || login.acclvl == 0)

{

Edit\_email\_settings myEmailsettings = new Edit\_email\_settings();

myEmailsettings.Show();

}

else MessageBox.Show("You do not have access to the Admin system!", "ACCESS DENIED");

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Open the Email messages dialog

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

{

loginFail = false;

Login login = new Login(conn);

login.ShowDialog();

if (login.Canceled)

loginFail = true;

if (!loginFail)

{

if (login.acclvl == 1 || login.acclvl == 0)

{

Change\_Email\_Message myDefaultEmail = new Change\_Email\_Message();

myDefaultEmail.Show();

}

else MessageBox.Show("You do not have access to the Admin system!", "ACCESS DENIED");

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Open the Finish bookings dialog, sending the booking ID

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

{

// Open the Finish booking dialog. send current booking number

conn.Open();

OleDbCommand cmd = new OleDbCommand($"SELECT \* FROM Payment WHERE BookingID = {currentBooking}", conn);

OleDbDataReader read = cmd.ExecuteReader();

if (!read.Read())

{

conn.Close();

Finish\_booking input = new Finish\_booking(conn, currentBooking);

input.ShowDialog();

txtBookDetails.Visible = false;

btnEditBooking.Visible = false;

btnCancel.Visible = false;

btnFinish.Visible = false;

}

else MessageBox.Show("This booking has already been finalised. Cannot Finalise again.", "Finalisation error", MessageBoxButtons.OK, MessageBoxIcon.Error);

conn.Close();

}

private void frmMain\_HelpRequested(object sender, HelpEventArgs hlpevent)

{

helpMethod();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// ============================= BOOKING EVENTS ============================ //

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Set the current booking

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

{

// Select booking detail for the selected booking, -> buttons visible

txtBookDetails.Text = ""; // Detail text of booking, including treatments

txtBookDetails.Visible = true;

btnEditBooking.Visible = true;

btnCancel.Visible = true;

btnFinish.Visible = true;

currentBooking = int.Parse((sender as TextBox).Text.Split('\r')[1].Split('#')[1]);

try

{

LoadBookingData();

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("Error showing booking details:\r\n" + ex, "Error");

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Show the detail for the current booking

private void LoadBookingData()

{

// Load booking detail into txtBookDetails, including treatments

if (currentBooking == 0)

throw new Exception("No booking selected");

conn.Open();

OleDbCommand cmd = new OleDbCommand($@"SELECT \* FROM Bookings WHERE BookingID = {currentBooking}", conn);

OleDbDataReader read = cmd.ExecuteReader();

string cID, cNum, cEmail, bTime, bTech, bTimeEst;

int techID;

if (read.Read())

{

OleDbCommand com = new OleDbCommand($@"SELECT \* FROM Clients WHERE ClientID = {cID = read["ClientID"].ToString()}", conn);

OleDbDataReader r = com.ExecuteReader();

r.Read();

cName = r["ClientName"].ToString();

cNum = r["TelNumber"].ToString();

cEmail = r["EmailAddress"].ToString();

bDate = (read.GetDateTime(2)).ToShortDateString();

bTime = (read.GetDateTime(2)).ToShortTimeString();

bTimeEst = read["TimeEstimate"].ToString();

if ((techID = int.Parse(read["EmployeeID"].ToString())) != 0)

{

com = new OleDbCommand($@"SELECT EmployeeName FROM Employees WHERE EmployeeID = {techID}", conn);

r = com.ExecuteReader();

r.Read();

bTech = r["EmployeeName"].ToString();

}

else bTech = "No preference";

txtBookDetails.Text = $"Client:\t{cName}\r\nTel:\t{cNum}\r\nEmail:\t{cEmail}\r\nDate:\t{bDate} Time: {bTime}\r\nTechnician:\t{bTech}\r\nTimes estimate:\t{bTimeEst} minutes\r\n\r\nTreatments:";

com = new OleDbCommand($@"SELECT TreatmentID FROM TreatmentDetailPerBooking WHERE BookingID = {currentBooking}", conn);

r = com.ExecuteReader();

while (r.Read())

{

cmd = new OleDbCommand($@"SELECT \* FROM Treatments WHERE TreatmentID = {r["TreatmentID"].ToString()}", conn);

read = cmd.ExecuteReader();

read.Read();

txtBookDetails.AppendText($"\r\n-{read["Description"].ToString()}\tR{read["TreatmentCost"].ToString()}");

}

}

else

throw new Exception("Error reading from the Bookings table.");

conn.Close();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// ========================== EVENTS ========================== //

}

}

# New Booking Form:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

using System.IO;

using Chalique\_Nail\_Studio.Properties;

using System.Net.Mail;

using System.Net.NetworkInformation;

using System.Net.Security;

using System.Net.Sockets;

namespace Chalique\_Nail\_Studio

{

public partial class New\_Booking : Form

{

private OleDbConnection conn;

private string DBloc;

public bool Complete { get; set; } // Booking is fully completed

private bool selected = false; // Client is selected

private bool edit = false; // Existing booking is being edited

private bool editLoading = false; // True while the booking to be edited is still loading

private bool clientEdit = false; // Client details have been altered

public DateTime Date { get; set; } // Date of booking

public static int ID { get; set; } // Booking ID

public int clientID; // Client ID

private string clientName; // Name of client

private string descript; // Description of booking

private int techID; // Employee ID of preffered technician; 0 if no preference

public New\_Booking(OleDbConnection con)

{

commonConstructor(con);

clientName = "";

descript = "";

Complete = false;

}

//if date has been selected

public New\_Booking(OleDbConnection con, DateTime bookingDate)

{

commonConstructor(con);

Date = bookingDate;

clientName = "";

descript = "";

//set date to date received

pickDate.SelectionStart = bookingDate;

Complete = false;

}

//if editing a booking

public New\_Booking(OleDbConnection con, int bookingID)

{

commonConstructor(con);

ID = bookingID;

try

{

editLoading = true;

conn.Open();

OleDbCommand cmd = new OleDbCommand("SELECT \* FROM Bookings WHERE BookingID = " + ID, conn);

OleDbDataReader read = cmd.ExecuteReader();

if (read.Read())

{

clientID = read.GetInt32(1);

Date = read.GetDateTime(2);

techID = read.GetInt32(3);

pickDate.SetDate(read.GetDateTime(2));

string bTime = (read.GetDateTime(2)).ToString("HH:mm");

cmbTime.SelectedIndex = cmbTime.FindString(bTime);

int timeEstimate = int.Parse(read["TimeEstimate"].ToString());

if (timeEstimate != 0)

{

cmbTimeEstimate.SelectedIndex = cmbTimeEstimate.FindString($"{timeEstimate}");

}

// Retrieve client details for booking

selected = true;

cmd = new OleDbCommand("SELECT \* FROM Clients WHERE ClientID = " + clientID, conn);

read = cmd.ExecuteReader();

read.Read();

txtClientName.Text = clientName = read["ClientName"].ToString();

txtClientNumber.Text = read["TelNumber"].ToString();

txtClientEmail.Text = read["EmailAddress"].ToString();

if (techID != 0)

{

cmd = new OleDbCommand($@"SELECT EmployeeName FROM Employees WHERE EmployeeID = {techID}", conn);

read = cmd.ExecuteReader();

read.Read();

cmbPrefTech.SelectedIndex = cmbPrefTech.FindString(read["EmployeeName"].ToString());

}

else cmbPrefTech.SelectedIndex = 0;

cmd = new OleDbCommand("SELECT \* FROM TreatmentDetailPerBooking WHERE BookingID = " + ID, conn);

read = cmd.ExecuteReader();

while (read.Read())

{

OleDbCommand com = new OleDbCommand($"SELECT \* FROM Treatments WHERE TreatmentID = " + read["TreatmentID"].ToString(), conn);

OleDbDataReader read1 = com.ExecuteReader();

read1.Read();

lbxTreatmentList.Items.Add(read1["Description"].ToString());

}

}

else throw new Exception("No booking found for Booking ID " + ID);

conn.Close();

editLoading = false;

edit = true;

Complete = true;

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("An error has occured while editing the booking: " + ex);

this.Close();

}

}

// This is shared by all constructors

private void commonConstructor(OleDbConnection con)

{

// Component initialization

InitializeComponent();

conn = con;

DBloc = Directory.GetCurrentDirectory();

if (!File.Exists(DBloc + @"\ChaliqueNailStudio.accdb"))

DBloc = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + @"\Chalique Nail Studio";

commonConstructor();

}

// This is shared by all constructors

private void commonConstructor()

{

Clear();

// Preferred technician comboBox

conn.Open();

cmbPrefTech.Items.Clear();

cmbPrefTech.Items.Add("No preference");

// Select all employees that are techicians

OleDbCommand cmd = new OleDbCommand("SELECT EmployeeName, IsTech FROM Employees WHERE IsTech = TRUE", conn);

OleDbDataReader read = cmd.ExecuteReader();

while(read.Read())

{

cmbPrefTech.Items.Add(read[0].ToString()); // Add them to the technician list

}

cmbTime.SelectedIndex = 6;

cmbTimeEstimate.SelectedIndex = 1;

// Treatment list combobox

cmbTreatment.Items.Clear();

cmd = new OleDbCommand("SELECT Description FROM Treatments", conn);

read = cmd.ExecuteReader();

while(read.Read())

cmbTreatment.Items.Add(read["Description"].ToString());

// Name textbox autocomplete settings

txtClientName.AutoCompleteMode = AutoCompleteMode.SuggestAppend;

txtClientName.AutoCompleteSource = AutoCompleteSource.CustomSource;

cmbTreatment.DropDownWidth = 250;

conn.Close();

// Style preset

StreamReader r = File.OpenText(DBloc + @"\settings.cns");

r.ReadLine();

r.ReadLine();

string[] temp;

if ((temp = r.ReadLine().Split(';'))[1] != "")

switch (int.Parse(temp[1]))

{

case 1:

this.BackgroundImage = Resources.Light\_Purple;

button\_colour(Color.Thistle);

break;

case 2:

this.BackgroundImage = Resources.Bright\_Pink;

button\_colour(Color.Pink);

break;

case 3:

this.BackgroundImage = Resources.Light\_Pink;

button\_colour(Color.Pink);

break;

default: break;

}

r.Close();

//SuggestStrings will have the logic to return array of strings either from cache/db

try

{

string[] arr = SuggestNames(txtClientName.Text);

if (arr != null)

{

AutoCompleteStringCollection collection = new AutoCompleteStringCollection();

collection.AddRange(arr);

txtClientName.AutoCompleteCustomSource = collection; //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* UNTESTED

}

}

catch (Exception)

{ }

}

private void button\_colour(Color c)

{

// Include here all buttons that will have their colour changed \*\*\*

btnNewBooking.BackColor = c;

btnCancel.BackColor = c;

btnClear.BackColor = c;

lblTimeEstimate.BackColor = c;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// ======================= SEARCH CLIENTS EVENTS =========================== //

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

{

if (!editLoading)

if (selected)

{

clientEdit = true;

}

}

private string[] SuggestNames(string t)

{

conn.Open();

OleDbCommand cmd = new OleDbCommand("SELECT ClientName FROM Clients ORDER BY ClientName", conn);

OleDbDataReader reader = cmd.ExecuteReader();

List<string> names = new List<string>();

while(reader.Read())

{

names.Add(reader[0].ToString());

}

conn.Close();

if (names.Count > 0)

return names.ToArray();

else return null;

}

private void txtClientName\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

selectClient();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// ========================== SELECT CLIENT EVENT ========================== //

private void selectClient()

{

if (txtClientName.Text != "")

{

conn.Open();

OleDbCommand cmd = new OleDbCommand($"SELECT \* FROM Clients WHERE ClientName LIKE '%{txtClientName.Text}%'", conn);

OleDbDataReader reader = cmd.ExecuteReader();

if (reader.Read())

{

txtClientEmail.Text = reader["EmailAddress"].ToString();

txtClientNumber.Text = reader["TelNumber"].ToString();

clientName = txtClientName.Text;

clientID = int.Parse(reader["ClientID"].ToString());

selected = true;

cmbTime.Focus();

}

conn.Close();

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// ========================== GATHER DETAIL EVENTS ========================= //

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

{

if (!editLoading)

if (selected)

{

clientEdit = true;

}

}

private void cmbTreatment\_SelectedValueChanged(object sender, EventArgs e)

{

}

private void addClient()

{

clientName = txtClientName.Text;

if (MessageBox.Show($"Do you want to add this client?\r\nName: {clientName}\tCell number: {txtClientNumber.Text}\r\nEmail: {txtClientEmail.Text}", "New Client", MessageBoxButtons.YesNo) == DialogResult.Yes)

{

conn.Open();

// Write new client details to table

OleDbCommand cmd = new OleDbCommand($"INSERT INTO Clients (ClientName, TelNumber, EmailAddress) VALUES('{clientName}', '{txtClientNumber.Text}', '{txtClientEmail.Text}')", conn);

cmd.ExecuteNonQuery();

// Read Client ID and set client as selected

cmd = new OleDbCommand($"SELECT ClientID From Clients WHERE ClientName = '{clientName}'", conn);

OleDbDataReader read = cmd.ExecuteReader();

read.Read();

clientID = int.Parse(read[0].ToString());

selected = true;

clientEdit = false;

conn.Close();

}

else throw new Exception("You must first select a client or add a new client.");

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// ========================= FINISH BOOKING EVENTS ========================= //

public void CompletionDetails()

{

// Notification method

try

{

if (Complete)

{

// Send notification of completion to client

//------------------------------------------------Email functionality added here------------------------------------------------

//Reads email and password from a textfile

string details = "";

string emailText = "";

string email = "";

string password = "";

SmtpClient client = new SmtpClient("smtp.gmail.com", 587);

client.EnableSsl = true;

client.Timeout = 60000;

client.Port = 587;

client.DeliveryMethod = SmtpDeliveryMethod.Network;

client.UseDefaultCredentials = false;

StreamReader myReader = new StreamReader(DBloc + "\\emailDetails.cns");

while (!myReader.EndOfStream)

{

details = myReader.ReadLine();

}

string[] allDetails = details.Split(':');

email = allDetails[0].ToString();

password = allDetails[1].ToString();

myReader.Close();

client.Credentials = new System.Net.NetworkCredential(email, password);

MailMessage msg = new MailMessage();

msg.To.Add(txtClientEmail.Text);

msg.From = new MailAddress("chaliquenailstudio@gmail.com");

//change email heading and body. Body reads from textfile when it is created

msg.Subject = "Appointment at Chalique Nail Studio";

myReader = new StreamReader(DBloc + "\\EmailMessage.cns");

emailText = myReader.ReadToEnd();

myReader.Close();

client.SendCompleted += new SendCompletedEventHandler(SendCompletedCallback);

msg.Body = emailText + $"Booking number:\t{ID}\r\nDate of Booking:\t{Date.ToString("dddd, dd MMMM, yyyy")}\r\nTime of Booking:\t{cmbTime.Text}\r\nPreferred technician:\t{cmbPrefTech.Text}";

client.SendMailAsync(msg);

}

else

{

MessageBox.Show("The booking is not complete.");

}

}

catch (Exception ex)

{

notifyDone.ShowBalloonTip(1500, "Booking not Completed!", $"{ex}", ToolTipIcon.Error);

}

}

private static void SendCompletedCallback(object sender, AsyncCompletedEventArgs e)

{

// Get the unique identifier for this asynchronous operation.

//String token = (string)e.UserState;

if (e.Cancelled)

{ }

if (e.Error != null)

{ }

else

{

MessageBox.Show($"Booking {ID} notification sent");

}

}

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

{

try

{

// Client name

if (txtClientName.Text != "" && !selected)

selectClient();

else if (txtClientName.Text == "")

throw new Exception("Please indicate a client for the treatment");

// Check all controlls for values

if (lbxTreatmentList.Items.Count <= 0)

{

throw new Exception("Please fill all the fields");

}

if (txtClientEmail.Text != "")

{

var test = new MailAddress(txtClientEmail.Text);

}

if (txtClientNumber.Text != "")

try

{

string t = txtClientNumber.Text.Replace(" ", "").Replace("(", "").Replace(")", "").Replace("+", "");

Int64 test1 = Int64.Parse(t);

}

catch (OverflowException)

{

throw new Exception("The telephone number is too long.");

}

catch (FormatException)

{

throw new Exception("The telephone number may not contain letters or special characters.");

}

if (!selected)

addClient();

OleDbCommand com;

if (clientEdit)

{

if (selected)

{

// Update Clients table with new info

conn.Open();

string message = $"UPDATE Clients SET ClientName = '{txtClientName.Text}', TelNumber = '{txtClientNumber.Text}', EmailAddress = '{txtClientEmail.Text}' WHERE ClientID = {clientID}";

//MessageBox.Show(message);

com = new OleDbCommand(message, conn);

com.ExecuteNonQuery();

conn.Close();

}

}

int hour, min;

hour = int.Parse(cmbTime.Text.Split(':')[0]);

min = int.Parse(cmbTime.Text.Split(':')[1]);

Date = Convert.ToDateTime(pickDate.SelectionStart.ToShortDateString()).AddHours(hour).AddMinutes(min);

//Ask if all input is correct

DialogResult d = MessageBox.Show("Is the booking correct?\n" + $"{clientName}\t{Date}\n{descript}", "Booking entry", MessageBoxButtons.YesNo);

if (!(d == DialogResult.Yes))

throw new Exception("");

int timeEst;

switch (cmbTimeEstimate.SelectedIndex)

{

case 0: timeEst = 30;

break;

case 1: timeEst = 60;

break;

case 2: timeEst = 90;

break;

case 3: timeEst = 120;

break;

case 4: timeEst = 150;

break;

default: timeEst = 0;

break;

}

// Store data

if (edit)

{

// Update booking and client tables

conn.Open();

com = new OleDbCommand($"UPDATE Bookings SET (BookingDateTime = #{Date}#, EmployeeID = {techID}, TimeEstimate = {timeEst}) WHERE BookingID = {ID}", conn);

//com.ExecuteNonQuery();

// Delete old entries from TreatmentDetailPerBooking

com = new OleDbCommand("DELETE \* FROM TreatmentDetailPerBooking WHERE BookingID = " + ID, conn);

com.ExecuteNonQuery();

}

else

{

// Insert into booking table

conn.Open();

com = new OleDbCommand($"INSERT INTO Bookings (ClientID, BookingDateTime, EmployeeID, TimeEstimate) VALUES({clientID}, #{Date}#, {techID}, {timeEst})", conn);

com.ExecuteNonQuery();

com = new OleDbCommand($"SELECT BookingID FROM Bookings WHERE ClientID = {clientID} AND BookingDateTime = #{Date}#", conn);

OleDbDataReader read = com.ExecuteReader();

read.Read();

ID = int.Parse(read[0].ToString());

}

foreach (var s in lbxTreatmentList.Items)

{

// Read data from the treatments table to store in the TreatmentDetailPerBooking table

com = new OleDbCommand($"SELECT \* FROM Treatments WHERE Description LIKE '%{s}%'", conn);

OleDbDataReader read = com.ExecuteReader();

read.Read();

string tID = read["TreatmentID"].ToString();

com = new OleDbCommand($"INSERT INTO TreatmentDetailPerBooking VALUES ({ID}, {tID}) ", conn);

com.ExecuteNonQuery();

}

//If all data could be stored

conn.Close();

Complete = true;

if (txtClientEmail.Text != "")

{

CompletionDetails();

}

this.Close();

}

catch (Exception ex)

{

conn.Close();

if (ex.Message != "")

MessageBox.Show("Please make sure all the data has been input\r\n//"+ ex, "ERROR", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// =============================== CANCEL EVENT ============================ //

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

{

Complete = false;

this.Close();

}

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

{

commonConstructor();

txtClientName.Focus();

}

private void Clear()

{

// Reset all input fields and variables

txtClientName.Clear();

txtClientNumber.Clear();

txtClientEmail.Clear();

selected = false;

Complete = false;

edit = false;

clientEdit = false;

clientName = "";

clientID = 0;

ID = 0;

descript = "";

techID = 0;

cmbTime.SelectedIndex = -1;

cmbTreatment.SelectedIndex = -1;

cmbTimeEstimate.SelectedIndex = -1;

cmbPrefTech.SelectedIndex = -1;

pickDate.SelectionStart = DateTime.Now;

lbxTreatmentList.Items.Clear();

}

private string CheckDoubleBookings(DateTime slot)

{

DateTime time;

time = slot;

return "";

}

private void lbxTreatmentList\_DoubleClick(object sender, EventArgs e)

{

if (lbxTreatmentList.SelectedItem != null)

{

cmbTreatment.Items.Add(lbxTreatmentList.SelectedItem);

lbxTreatmentList.Items.Remove(lbxTreatmentList.SelectedItem);

}

}

private void DropDown(object sender, EventArgs e)

{

(sender as ComboBox).DroppedDown = true;

}

private void cmbPrefTech\_SelectedIndexChanged(object sender, EventArgs e)

{

if (cmbPrefTech.SelectedIndex == 0)

{

// Select employee ID 0 // no preference

techID = 0;

}

else

{

if (!editLoading)

{

// Read employee ID by selected name

conn.Open();

OleDbCommand cmd = new OleDbCommand($"SELECT EmployeeID From Employees WHERE EmployeeName LIKE '%{cmbPrefTech.Text}%'", conn);

OleDbDataReader read = cmd.ExecuteReader();

read.Read();

techID = int.Parse(read[0].ToString());

conn.Close();

}

}

}

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

{

if (cmbTreatment.SelectedIndex != -1)

{

// Add to description

if (cmbTreatment.SelectedText != "")

{

lbxTreatmentList.Items.Add(cmbTreatment.Text);

cmbTreatment.Items.Remove(cmbTreatment.SelectedItem);

DropDown(sender, e);

}

}

else DropDown(sender, e);

}

private void cmbTreatment\_DropDownClosed(object sender, EventArgs e)

{

if (cmbTreatment.SelectedIndex != -1)

{

// Add to description

if (cmbTreatment.SelectedText != "")

{

lbxTreatmentList.Items.Add(cmbTreatment.Text);

cmbTreatment.Items.Remove(cmbTreatment.SelectedItem);

DropDown(sender, e);

}

}

}

private void pickDate\_DateChanged(object sender, DateRangeEventArgs e)

{

if (pickDate.SelectionStart.DayOfWeek == DayOfWeek.Saturday)

{

cmbTime.Items.Clear();

string[] temp = { "07:00", "07:30", "08:00", "08:30", "09:00", "09:30", "10:00", "10:30", "11:00", "11:30", "12:00", "12:30" };

cmbTime.Items.AddRange(temp);

cmbTime.SelectedIndex = 6;

cmbTimeEstimate.SelectedIndex = 0;

}

else if (cmbTime.Items.Count < 13)

{

cmbTime.Items.Clear();

string[] temp = { "07:00", "07:30", "08:00", "08:30", "09:00", "09:30", "10:00", "10:30", "11:00", "11:30", "12:00", "12:30", "13:00", "13:30", "14:00", "14:30", "15:00", "15:30", "16:00", "16:30" };

cmbTime.Items.AddRange(temp);

cmbTime.SelectedIndex = 6;

cmbTimeEstimate.SelectedIndex = 0;

}

}

}

}

# Finalize Booking

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

using System.IO;

using Chalique\_Nail\_Studio.Properties;

namespace Chalique\_Nail\_Studio

{

public partial class Finish\_booking : Form

{

public OleDbConnection conn;

public string DBloc;

private int currentBooking;

private double totalCost = 0;

public bool Complete;

public double individualCost = 0;

public double price1;

// global variables

public int bookID;

public int amountPaid;

public string paymentMethod;

public DateTime dt;

// global variables(payment)

public Finish\_booking(OleDbConnection con, int bookingID)

{

InitializeComponent();

conn = con;

DBloc = Directory.GetCurrentDirectory();

if (!File.Exists(DBloc + @"\ChaliqueNailStudio.accdb"))

DBloc = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + @"\Chalique Nail Studio";

currentBooking = bookingID;

txtBookingNumber.Text = $"{currentBooking}";

comboxBoxData();

readData();

Style();

txtFinalPrice.Enabled = true;

// allow userbility with textbox

}

private void Style()

{

// Style preset

StreamReader r = File.OpenText(DBloc + @"\settings.cns");

r.ReadLine();

r.ReadLine();

string[] temp;

if ((temp = r.ReadLine().Split(';'))[1] != "")

switch (int.Parse(temp[1]))

{

case 1:

this.BackgroundImage = Resources.Light\_Purple;

button\_colour(Color.Thistle);

break;

case 2:

this.BackgroundImage = Resources.Bright\_Pink;

button\_colour(Color.Pink);

break;

case 3:

this.BackgroundImage = Resources.Light\_Pink;

button\_colour(Color.Pink);

break;

default: break;

}

r.Close();

}

private void button\_colour(Color c)

{

// Include here all buttons that will have their colour changed

btnFinalize.BackColor = c;

btnCancel.BackColor = c;

btnAddTreatment.BackColor = c;

}

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

{

try

{

// finalizes the booking

Complete = true;

//notifyDone.ShowBalloonTip(1500, "Booking Finalised", $"The booking for {clientName} has been finalised and payment received.\r\nBooking finalised", ToolTipIcon.None);

this.Close();

paymentMethod = cmbPayment.Text;

amountPaid = Convert.ToInt16(txtFinalPrice.Text);

dt = DateTime.Now;

conn.Open();

OleDbDataAdapter adapt = new OleDbDataAdapter(@"SELECT BookingID, AmountPaid, PaymentMethod, DateOfPayment FROM Payment", conn);

OleDbCommand cmd = new OleDbCommand(@"INSERT Into Payment(BookingID, AmountPaid, PaymentMethod, DateOfPayment) Values ( " + currentBooking + " , '" +

amountPaid + "', ' " + paymentMethod + "' , '" + dt + "')", conn);

adapt.InsertCommand = cmd;

adapt.InsertCommand.ExecuteNonQuery();

// Inserts data into Payment table

OleDbCommand cmds = new OleDbCommand("DELETE \* FROM TreatmentDetailPerBooking WHERE BookingID = " + currentBooking, conn);

cmds.ExecuteNonQuery();

foreach (var s in lbxTreatments.Items)

{

// Read data from the treatments table to store in the TreatmentDetailPerBooking table

OleDbCommand cm = new OleDbCommand($"SELECT \* FROM Treatments WHERE Description LIKE '%{s}%'", conn);

OleDbDataReader read = cm.ExecuteReader();

read.Read();

string tID = read["TreatmentID"].ToString();

OleDbCommand c = new OleDbCommand($"INSERT INTO TreatmentDetailPerBooking VALUES ({currentBooking}, {tID}) ", conn);

c.ExecuteNonQuery();

}

conn.Close();

// MessageBox.Show("The booking was finalised on: " + dt);

}

catch(Exception exc)

{

conn.Close();

MessageBox.Show("An error occured: " + exc.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

Complete = false;

this.Close();

MessageBox.Show("Finish booking cancelled", "Cancelled", MessageBoxButtons.OK, MessageBoxIcon.Information);

// notification

}

public void comboxBoxData()

{

try

{

string holder = "";

conn.Open();

string testQuery = "SELECT Description FROM Treatments";

OleDbCommand cmds = new OleDbCommand(testQuery, conn);

OleDbDataReader drs = cmds.ExecuteReader();

while (drs.Read())

{

holder = drs["Description"].ToString();

cmbTreatments.Items.Add(holder);

// populate additional treatment(s) combo box

}

conn.Close();

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("Finish err 01\r\n"+ ex, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

public void readData()

{

try

{

conn.Open();

cmbPayment.SelectedIndex = 0;

OleDbCommand cmd = new OleDbCommand("SELECT \* FROM Bookings WHERE BookingID = " + currentBooking, conn);

OleDbDataReader read = cmd.ExecuteReader();

if (read.Read())

{

int clientID = read.GetInt32(1);

// Retrieve client details for booking

cmd = new OleDbCommand("SELECT \* FROM Clients WHERE ClientID = " + clientID, conn);

read = cmd.ExecuteReader();

read.Read();

txtClientName.Text = read["ClientName"].ToString();

txtClientNumber.Text = read["TelNumber"].ToString();

txtClientEmail.Text = read["EmailAddress"].ToString();

cmd = new OleDbCommand("SELECT \* FROM TreatmentDetailPerBooking WHERE BookingID = " + currentBooking, conn);

read = cmd.ExecuteReader();

while (read.Read())

{

OleDbCommand com = new OleDbCommand($"SELECT \* FROM Treatments WHERE TreatmentID = " + read["TreatmentID"].ToString(), conn);

OleDbDataReader read1 = com.ExecuteReader();

read1.Read();

string treatment = read1["Description"].ToString();

lbxTreatments.Items.Add(treatment);

if(cmbTreatments.FindString(treatment) != -1)

{

cmbTreatments.Items.RemoveAt(cmbTreatments.FindString(treatment));

}

price1 = double.Parse(read1[2].ToString());

totalCost += price1;

//totalCost += double.Parse(read1[2].ToString());

txtFinalPrice.Text = Convert.ToString(totalCost);

}

//txtFinalPrice.Text = $"R {totalCost:0.00}"; // Requires testing \*\*\*

}

conn.Close();

// import data for client name, telephone number and email address based on the client ID

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("Finish err 02\r\n"+ ex, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

try

{

if (cmbTreatments.SelectedIndex != -1)

{

lbxTreatments.Items.Add(cmbTreatments.SelectedItem);

}

// List box adds (additions) selected from combox box items to listbox(additions)

string value = cmbTreatments.Text;

// retrieving data from the combo box

conn.Open();

OleDbCommand cmd = new OleDbCommand($@"SELECT TreatmentCost FROM Treatments WHERE Description = '{value}'", conn);

OleDbDataReader rdr = cmd.ExecuteReader();

rdr.Read();

price1 = double.Parse(rdr["TreatmentCost"].ToString());

totalCost += price1;

txtFinalPrice.Text = Convert.ToString(totalCost);

conn.Close();

cmbTreatments.Items.Remove(cmbTreatments.SelectedItem);

}

catch (Exception exc)

{

conn.Close();

MessageBox.Show("An error occured!" + exc.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

// basic calculation function

}

private void lbxTreatments\_MouseDoubleClick(object sender, MouseEventArgs e)

{

if (lbxTreatments.SelectedItem != null)

{

string value = lbxTreatments.SelectedItem.ToString();

conn.Open();

OleDbCommand cmd = new OleDbCommand($@"SELECT TreatmentCost FROM Treatments WHERE Description = '{value}'", conn);

OleDbDataReader rdr = cmd.ExecuteReader();

rdr.Read();

price1 = double.Parse(rdr["TreatmentCost"].ToString());

totalCost -= price1;

txtFinalPrice.Text = Convert.ToString(totalCost);

conn.Close();

// closes the database

cmbTreatments.Items.Add(lbxTreatments.SelectedItem);

// remove the selected item

cmbTreatments.Items.Remove(cmbTreatments.SelectedItem);

lbxTreatments.Items.Remove(lbxTreatments.SelectedItem);

}

else

{

MessageBox.Show("Please select a valid treatent to remove", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

// Function to double click remove and basic calculation function

}

private void cmbTreatments\_SelectedIndexChanged(object sender, EventArgs e)

{

///// N/A

}

}

}

# Admin Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

using System.IO;

namespace Chalique\_Nail\_Studio

{

public partial class Admin : Form

{

OleDbConnection conn;

private string DBloc;

private string Target;

private string person;

private int expCode;

private bool edit = false;

private DataGridViewRow myRows;

public Admin(OleDbConnection con, string user, string target)

{

InitializeComponent();

conn = con;

DBloc = Directory.GetCurrentDirectory();

if (!File.Exists(DBloc + @"\ChaliqueNailStudio.accdb"))

DBloc = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + @"\Chalique Nail Studio";

Target = target;

person = user;

button\_colour(Color.LightBlue);

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*CHANGE BUTTON COLOUR\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

private void button\_colour(Color c)

{

// Include here all buttons that will have their colour changed \*\*\*

btnReport.BackColor = c;

btnNewTreatment.BackColor = c;

btnAddExpense.BackColor = c;

btnAddExpenseType.BackColor = c;

btnEditexp.BackColor = c;

btnDeleteExp.BackColor = c;

btnUpdateExp.BackColor = c;

dgvMonthView.BackgroundColor = c;

dgvExpList.BackgroundColor = c;

dgvIncomes.BackgroundColor = c;

dgvIncomeMonthly.BackgroundColor = c;

}

//--------------------------------------------------------------------------------------RELATED TO ALL TAB PAGES------------------------------------------------------------------------------

private void tabListExpenses\_Enter(object sender, EventArgs e)

{

//lbxMonthExpensesList.Items.Clear();

ListAll();

}

private void tabTypeExpenses\_Enter(object sender, EventArgs e)

{

ReadExpenseTypesEdit();

txtExpenseType.Focus();

conn.Close();

}

private void tabNewExpense\_Enter(object sender, EventArgs e)

{

//Calls method that display the expense types from the Database

LoadExpenseTypes();

ListCurrentMonthExpsenses();

btnDeleteExp.Visible = false;

btnEditexp.Visible = false;

btnUpdateExp.Visible = false;

conn.Close();

}

private void tabEditAndDeleteIncomes\_Enter(object sender, EventArgs e)

{

//Calls a method to display all payments from the database

ListCurrentMonthIncome();

btnDeleteIncome.Visible = false;

btnEditIncome.Visible = false;

btnUpdate.Visible = false;

conn.Close();

}

private void tabTreatments\_Enter(object sender, EventArgs e)

{

//Calls method that display the treatments from the Database

ReadTreatments();

conn.Close();

}

//-------------------------------------------------------------------------------------------------------------END INFORMATION RELATED TO ALL TAB PAGES---------------------------------------------------

//-----------------------------------------------------------------------------------------------METHOD RELATED TO ADD, EDIT AND DELETE EXPENSES & INCOMES (TAB PAGE 3 & 4)------------------------------------------------------

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Add to a log file: Edit payment; add, edit, remove expense

private void WriteLog(string logText, double oldAmt, double newAmt)

{

if (!File.Exists(DBloc + @"\logfile.txt"))

{

if (Target == null || Target == "")

Target = DBloc;

if (File.Exists(Target + @"\logfile.Backup.txt"))

{

File.Copy(Target + @"\logfile.Backup.txt", DBloc + @"\logfile.txt", true);

}

}

StreamWriter w = File.AppendText(DBloc + @"\logfile.txt");

string datetime = DateTime.Now.ToString("dd/MM/yyyy HH:mm:ss");

w.WriteLine($"{datetime}, {person}, {logText}, New amount: {newAmt: 0.00}, Old amount: {oldAmt: 0.00};");

w.Close();

}

//--------------------------------------------------------------------------------------------------METHOD END RELATED TO TAB PAGE 3 & 4)------------------------------------------------------------------------

//--------------------------------------------------------------------------------------------------All EXPENSES AND INCOMES TAB INFORMATION (TAB PAGE 1)-----------------------------------------------------------------------------

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*lists all the current months expenses in the first tab page\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public void ListAll()

{

DateTime test = dateExpenseSelect.SelectionStart;//which datetimepicker do i choose this from?

DateTime start = new DateTime(test.Year, test.Month, 1, 0, 0, 0, 0);

DateTime end = start.AddMonths(1).AddMinutes(-1);

try

{

conn.Open();

//---------------------------------------------------------------------------Expenses-----------------------------------------------------------------------------------------

OleDbDataAdapter adapter = new OleDbDataAdapter($@"SELECT B.DateOfExpense AS [Date], A.ExpenseDescript AS [Description], B.Cost

FROM ExpenseType A INNER JOIN OperationalExpense B

ON A.ExpenseCode = B.ExpenseCode

WHERE B.DateOfExpense BETWEEN #{start.ToShortDateString()}# AND #{end.ToShortDateString()}#

ORDER BY B.DateOfExpense", conn);

DataSet ds = new DataSet();

adapter.Fill(ds, "Expenses");

dgvMonthView.DataSource = ds;

dgvMonthView.DataMember = "Expenses";

dgvMonthView.Columns[1].Width = 250;

foreach (DataGridViewColumn col in dgvMonthView.Columns)

{

col.HeaderCell.Style.Alignment = DataGridViewContentAlignment.MiddleCenter;

}

conn.Close();

}

catch (Exception e)

{

MessageBox.Show("There was an error\n" + e.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

try {

//---------------------------------------------------------------------------Incomes-----------------------------------------------------------------------------------------

conn.Open();

OleDbDataAdapter adt = new OleDbDataAdapter($@"SELECT B.DateOfPayment AS [Date], C.ClientName AS [Description], B.AmountPaid AS [Amount Paid], B.SaleID AS [Sale ID]

FROM (Bookings A INNER JOIN Payment B ON B.BookingID = A.BookingID)

INNER JOIN Clients C ON A.ClientID = C.ClientID

WHERE B.DateOfPayment BETWEEN #{start.ToShortDateString()}# AND #{end.ToShortDateString()}#

ORDER BY B.DateOfPayment",conn);

DataSet d = new DataSet();

adt.Fill(d,"Incomes");

dgvIncomeMonthly.DataSource = d;

dgvIncomeMonthly.DataMember = "Incomes";

dgvIncomeMonthly.Columns[1].Width = 250;

foreach (DataGridViewColumn col in dgvIncomeMonthly.Columns)

{

col.HeaderCell.Style.Alignment = DataGridViewContentAlignment.MiddleCenter;

}

conn.Close();

}

catch(Exception e)

{

MessageBox.Show("There was an error\n" + e.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//------------------------------------------------------------------------------------------------END TAB PAGE 1-----------------------------------------------------------------------------------------------

//----------------------------------------------------------------------------------------------------EXPENSE TYPES TAB INFORMATION (TAB PAGE 2)-------------------------------------------------------------------------------------------

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Method that displays information from the database in the listBox on tab page 2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

public void ReadExpenseTypesEdit()

{

try

{

//Clears the contens of the listBox on tabPage2

lbxExpenses.Items.Clear();

//Opens the DB connection

conn.Open();

//Command and datareader to read expense types from the DB into the listBox

using (OleDbCommand command = new OleDbCommand())

{

command.Connection = conn;

command.CommandText = "SELECT ExpenseDescript FROM ExpenseType ORDER BY ExpenseDescript ASC";

//reads information from the database

using (OleDbDataReader reader = command.ExecuteReader())

{

while (reader.Read())

{

lbxExpenses.Items.Add(reader["ExpenseDescript"].ToString());

}

}

}

//Close DB connection

conn.Close();

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Button click adds an expense into the database, Calls method that displays the expense in the listbox on tab page 2 as well\*\*\*\*\*\*\*\*\*\*\*\*\*//

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

{

try

{

string expType = txtExpenseType.Text;

//check if the textBox is empty or not before adding information in the database

if (string.IsNullOrEmpty(expType))

{

MessageBox.Show("Please enter an expense type!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

if (MessageBox.Show("Are you sure you would like to add " + expType + " as an expense?", "Request", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

try

{

//Opens the DB connection

conn.Open();

//Command to insert expense type into Database

OleDbCommand cmd = new OleDbCommand($@"INSERT INTO ExpenseType(ExpenseDescript) VALUES ('{expType}')", conn);

cmd.ExecuteNonQuery();

//Expense type that is added is shown in messageBox

MessageBox.Show("You have successfully added " + expType + " as an expense", "Success", MessageBoxButtons.OK, MessageBoxIcon.Information);

//Close DB connection

conn.Close();

//Calls method that reads the database and displays the expense type in the listBox

ReadExpenseTypesEdit();

//Clears the contents of the textBox

txtExpenseType.Clear();

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

else

{

MessageBox.Show(expType + " was not added", "Declined", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

txtExpenseType.Focus();

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Method to remove an expense when you double click the listbox expense type\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

private void lbxExpenses\_DoubleClick(object sender, EventArgs e)

{

if(lbxExpenses.SelectedItem != null)

{

string expName = "";

foreach (string val in lbxExpenses.SelectedItems)

{

expName = val;

}

if (MessageBox.Show("Do you wish to remove " + expName + " as an expense?", "Request", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

try

{

conn.Open();

//Add here the check for expenses of this type and deny deletion if any exist (Can also give option to replace expense type)\*\*\*\*\*\*\*\*\*\*\*\*//

//OleDbDataAdapter adapter = new OleDbDataAdapter($@"SELECT ExpenseDescript FROM ExpenseType ORDER BY ExpenseDescript ASC", conn);

OleDbCommand cmd = new OleDbCommand($@"DELETE FROM ExpenseType WHERE ExpenseDescript = '{expName}'", conn);

cmd.ExecuteNonQuery();

//MessageBox.Show(expName + " was successfully removed as an expense", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

conn.Close();

ReadExpenseTypesEdit();

}

catch(Exception ef)

{

MessageBox.Show("There was an error\n" + ef.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

else

{

MessageBox.Show(expName + " was not removed", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

else

{

MessageBox.Show("You have not selected a valid expense type to remove", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

}

}

//-------------------------------------------------------------------------------------------------------------------END TAB PAGE 2--------------------------------------------------------------------------

//----------------------------------------------------------------------------------------------------NEW EXPENSE TAB INFORMATION (TAB PAGE 3)-------------------------------------------------------------------------------------------

public void clearAll()

{

cmbExpenseType.Text = "Select the type of expense to add";

txtExpenseAmt.Text = null;

dateExpenseSelect.TodayDate = DateTime.Now;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Displays expense type from the database in the comboBox on tabNewExpense\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public void LoadExpenseTypes()

{

try

{

//Clear content of ComboxBox

cmbExpenseType.Items.Clear();

//Opens the DB connection

conn.Open();

//Command and datareader to read expense types from the DB into the comboBox

using (OleDbCommand command = new OleDbCommand())

{

command.Connection = conn;

command.CommandText = "SELECT ExpenseDescript FROM ExpenseType ORDER BY ExpenseDescript ASC";

//whenever you want to get some data from the database

using (OleDbDataReader reader = command.ExecuteReader())

{

while (reader.Read())

{

cmbExpenseType.Items.Add(reader["ExpenseDescript"].ToString());

}

}

}

//Close DB connection

conn.Close();

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*shows all the current months expenses in the datagridview\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public void ListCurrentMonthExpsenses()

{

try

{

conn.Open();

DateTime test = dateExpenseSelect.SelectionStart;

DateTime start = new DateTime(test.Year, test.Month, 1, 0, 0, 0, 0);

DateTime end = start.AddMonths(1).AddMinutes(-1);

OleDbDataAdapter adapter = new OleDbDataAdapter($@"SELECT B.DateOfExpense AS [Date], A.ExpenseDescript AS [Description],B.Cost, B.ExpenseID

FROM ExpenseType A INNER JOIN OperationalExpense B

ON A.ExpenseCode = B.ExpenseCode

WHERE B.DateOfExpense BETWEEN #{start.ToShortDateString()}# AND #{end.ToShortDateString()}#

ORDER BY B.DateOfExpense", conn);

DataSet ds = new DataSet();

adapter.Fill(ds, "Expenses");

dgvExpList.DataSource = ds;

dgvExpList.DataMember = "Expenses";

dgvExpList.Columns[1].Width = 250;

foreach (DataGridViewColumn col in dgvExpList.Columns)

{

col.HeaderCell.Style.Alignment = DataGridViewContentAlignment.MiddleCenter;

}

conn.Close();

}

catch(Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*reads the expense code type from the database so that we work across multiple tables\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public void ExpenseCodeType()

{

try

{

expCode = -1;

conn.Open();

OleDbCommand cmd = new OleDbCommand($@"SELECT ExpenseCode FROM ExpenseType WHERE ExpenseDescript = '{cmbExpenseType.Text}'", conn);

OleDbDataReader read = cmd.ExecuteReader();

read.Read();

expCode = int.Parse(read["ExpenseCode"].ToString());

//MessageBox.Show(expCode.ToString());

conn.Close();

}

catch(Exception ex)

{

conn.Close();

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*method that loads all the current expenses from the database\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public void LoadCurrentExpense(string expID)

{

try

{

cmbExpenseType.Enabled = false;

conn.Open();

dgvExpList.ClearSelection();

edit = true;

OleDbCommand cmd = new OleDbCommand($@"SELECT B.DateOfExpense, A.ExpenseDescript, B.Cost

FROM ExpenseType A INNER JOIN OperationalExpense B

ON A.ExpenseCode = B.ExpenseCode

WHERE B.ExpenseID = {expID}", conn);

OleDbDataReader reader = cmd.ExecuteReader();

reader.Read();

//int.Parse(read["ExpenseCode"].ToString());

dateExpenseSelect.SelectionStart = Convert.ToDateTime(reader["DateOfExpense"].ToString());

dateExpenseSelect.SelectionEnd = Convert.ToDateTime(reader["DateOfExpense"].ToString());

cmbExpenseType.Text = reader["ExpenseDescript"].ToString();

txtExpenseAmt.Text = reader["Cost"].ToString();

conn.Close();

}

catch(Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

//make button submit\_changes.visible = true, and button new\_expense.visible = false;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Add expense to database\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

{

try

{

ExpenseCodeType();

conn.Open();

DateTime dtExp;

dtExp = dateExpenseSelect.SelectionRange.Start;

double oldVal = 0;

if (txtExpenseAmt.Text != "")

try

{

//string t = txtExpenseAmt.Text.Replace(" ", "").Replace("(", "").Replace(")", "").Replace("+", "");

Int64 test1 = Int64.Parse(txtExpenseAmt.Text);

}

catch(Exception)

{

conn.Close();

throw new Exception("Please enter a valid number");

}

double newVal = Convert.ToDouble(txtExpenseAmt.Text);

string logtext = $"Add expense {cmbExpenseType.SelectedItem.ToString()} on the {dateExpenseSelect.SelectionStart.ToShortDateString()} for date: {dtExp.ToShortDateString()}";

if (MessageBox.Show("Are you sure that you would like to add " + cmbExpenseType.SelectedItem.ToString() + " totalling R" + txtExpenseAmt.Text + " as an expense", "Request", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

try

{

OleDbCommand command = new OleDbCommand($@"INSERT INTO OperationalExpense (DateOfExpense, Cost, ExpenseCode) VALUES ('{dtExp}', {double.Parse(txtExpenseAmt.Text)}, {expCode})", conn);

command.ExecuteNonQuery();

conn.Close();

WriteLog(logtext, oldVal, newVal);

dgvExpList.Refresh();

ListCurrentMonthExpsenses();

cmbExpenseType.Items.Clear();

LoadExpenseTypes();

clearAll();

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

else

{

MessageBox.Show(cmbExpenseType.SelectedItem.ToString() + " was not added", "Request", MessageBoxButtons.OK, MessageBoxIcon.Question);

}

conn.Close();

}

catch(Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*btnEdit click that allows you to update the expenses in tab page 3\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

{

try

{

myRows = dgvExpList.SelectedRows[0];

LoadCurrentExpense(myRows.Cells[3].Value.ToString());

btnUpdateExp.Visible = true;

btnDeleteExp.Visible = false;

btnAddExpense.Visible = false;

btnEditexp.Visible = false;

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*btnUpdate that allows you to update an expense\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

{

try

{

DateTime dtExp;

dtExp = dateExpenseSelect.SelectionRange.Start;

//DataGridViewRow myRows = dgvExpList.SelectedRows[0];

//ExpenseID();

string expenseID = myRows.Cells[3].Value.ToString();

double oldVal = Convert.ToDouble(myRows.Cells[2].Value.ToString());

double newVal = Convert.ToDouble(txtExpenseAmt.Text);

string logtext = $"Edit expense {myRows.Cells[1].Value.ToString()}, ID: {expenseID} for date: {myRows.Cells[0].Value.ToString()}";

string message = "Are you sure you want to update " + myRows.Cells[1].Value.ToString() + " on the " + myRows.Cells[0].Value.ToString() + " to R" + txtExpenseAmt.Text + "?";

if (MessageBox.Show(message, "Request", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

try

{

conn.Open();

OleDbCommand cmd = new OleDbCommand($@"UPDATE OperationalExpense SET DateOfExpense = '{dtExp}', Cost = {double.Parse(txtExpenseAmt.Text)} WHERE ExpenseID = {expenseID}", conn);

cmd.ExecuteNonQuery();

conn.Close();

WriteLog(logtext, oldVal, newVal);

ListCurrentMonthExpsenses();

clearAll();

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

else

{

MessageBox.Show(myRows.Cells[1].Value.ToString() + " was not updated", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

dgvExpList.Refresh();

txtExpenseAmt.Text = "";

cmbExpenseType.Text = "Select the type of expense to add";

LoadExpenseTypes();

btnUpdateExp.Visible = false;

btnDeleteExp.Visible = false;

btnAddExpense.Visible = true;

btnEditexp.Visible = false;

cmbExpenseType.Enabled = true;

edit = false;

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Remove the selected expense from the database when expense is selected on the datagridview

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

{

try

{

//Selected expense row

DataGridViewRow myRow = dgvExpList.SelectedRows[0];

//Details for the log file

double oldVal = Convert.ToDouble(myRow.Cells[2].Value.ToString());

double newVal = 0;

string logtext = $"Remove expense {myRow.Cells[1].Value.ToString()}, ID: {myRow.Cells[3].Value.ToString()} for date: {myRow.Cells[0].Value.ToString()}";

//

string expIDDel = (myRow.Cells[3].Value.ToString());

string delItem = myRow.Cells[1].Value.ToString();

string message = "Are you sure that you would like to remove " + delItem + " of R" + myRow.Cells[2].Value.ToString() + " on the " + myRow.Cells[0].Value.ToString();

if (MessageBox.Show(message, "Request", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

try

{

conn.Open();

OleDbCommand cmd = new OleDbCommand($@"DELETE FROM OperationalExpense WHERE ExpenseID = {expIDDel}", conn);

cmd.ExecuteNonQuery();

conn.Close();

ListCurrentMonthExpsenses();

WriteLog(logtext, oldVal, newVal);

clearAll();

//MessageBox.Show(delItem + " was removed successfully", "Success", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

else

{

MessageBox.Show(myRow.Cells[1].Value.ToString() + " was not removed", "Failed", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

dgvExpList.Refresh();

btnUpdateExp.Visible = false;

btnDeleteExp.Visible = false;

btnAddExpense.Visible = true;

btnEditexp.Visible = false;

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Allows you to edit or delete an expense when an expense is selected from the datagridview\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

{

btnDeleteExp.Visible = true;

btnEditexp.Visible = true;

try

{

DataGridViewRow myRow = dgvExpList.SelectedRows[0];

if (myRow == null)

throw new Exception("No entry has been chosen!");

btnDeleteExp.Visible = true;

btnEditexp.Visible = true;

}

catch (Exception ex)

{

MessageBox.Show("" + ex);

}

}

//--------------------------------------------------------------------------------------END TAB PAGE 3-----------------------------------------------------------------------------------------------------

//-------------------------------------------------------------------------------------BEGIN TAB PAGE 4-----------------------------------------------------------------------------------------------------

public void ListCurrentMonthIncome()

{

try

{

//set the date time picker from selected date to one month after selected date

DateTime test = dateExpenseSelect.SelectionStart;

DateTime start = new DateTime(test.Year, test.Month, 1, 0, 0, 0, 0);

DateTime end = start.AddMonths(1).AddMinutes(-1);

//add columns to datagridview

conn.Open();

string message = $@"SELECT B.DateOfPayment AS [Date], C.ClientName AS [Description], B.AmountPaid AS [Amount Paid], B.SaleID AS [Sale ID]

FROM (Bookings A INNER JOIN Payment B ON B.BookingID = A.BookingID)

INNER JOIN Clients C ON A.ClientID = C.ClientID

WHERE B.DateOfPayment BETWEEN #{start.ToShortDateString()}# AND #{end.ToShortDateString()}#

ORDER BY B.DateOfPayment

";

OleDbDataAdapter adapter = new OleDbDataAdapter(message, conn);

DataSet ds = new DataSet();

adapter.Fill(ds, "Incomes");

dgvIncomes.DataSource = ds;

dgvIncomes.DataMember = "Incomes";

dgvIncomes.Columns[1].Width = 250;

foreach (DataGridViewColumn col in dgvIncomes.Columns)

{

col.HeaderCell.Style.Alignment = DataGridViewContentAlignment.MiddleCenter;

}

conn.Close();

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Method Loading Current Income\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

public void LoadCurrentIncome(string SID)

{

try

{

//set the date time picker from selected date to one month after selected date

DateTime test = dateExpenseSelect.SelectionStart;

DateTime start = new DateTime(test.Year, test.Month, 1, 0, 0, 0, 0);

DateTime end = start.AddMonths(1).AddMinutes(-1);

txtDescr.Enabled = false;

conn.Open();

edit = true;

OleDbCommand cmd = new OleDbCommand($@"SELECT B.DateOfPayment AS [Date], C.ClientName AS [Description], B.AmountPaid AS [Amount Paid], B.SaleID AS [Sale ID]

FROM (Bookings A INNER JOIN Payment B ON B.BookingID = A.BookingID)

INNER JOIN Clients C ON A.ClientID = C.ClientID

WHERE SaleID = {SID}", conn);

OleDbDataReader reader = cmd.ExecuteReader();

reader.Read();

dateExpenseSelect.SelectionStart = Convert.ToDateTime(reader["DateOfPayment"].ToString());

dateExpenseSelect.SelectionEnd = Convert.ToDateTime(reader["DateOfPayment"].ToString());

txtDescr.Text = reader["ClientName"].ToString();

txtIncomeAmt.Text = reader["AmountPaid"].ToString();

conn.Close();

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Method when datagridview is selected\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

{

//set buttons visible when datagridview row is selected

btnEditIncome.Visible = true;

btnDeleteIncome.Visible = true;

try

{

DataGridViewRow myRow = dgvIncomes.SelectedRows[0];

if (myRow == null)

throw new Exception("No entry has been chosen!");

btnEditIncome.Visible = true;

btnDeleteIncome.Visible = true;

}

catch (Exception ex)

{

MessageBox.Show("" + ex);

}

}

//-----------------------------------------------------------------------CHANGE PAYMENTS TAB (edit and delete incomes) (TAB PAGE 4)--------------------------------------------------------------------------------------------------------

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

{

try

{

//add desctription to textbox and set editible false

DataGridViewRow myRow = dgvIncomes.SelectedRows[0];

txtDescr.Text = myRow.Cells[1].Value.ToString();

txtDescr.Enabled = false;

//add amount to textbox and keep editible true

txtIncomeAmt.Text = myRow.Cells[2].Value.ToString();

btnUpdate.Visible = true;

btnEditIncome.Visible = false;

btnDeleteIncome.Visible = false;

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

{

try

{

DataGridViewRow myRow = dgvIncomes.SelectedRows[0];

double oldVal = Convert.ToDouble(myRow.Cells[2].Value);

double newVal = Convert.ToDouble(txtIncomeAmt.Text);

string logtext = $"Edit Income {myRow.Cells[2].Value.ToString()} for salesID: {myRow.Cells[3].Value.ToString()} on date: {myRow.Cells[0].Value.ToString()}";

string message = "Are you sure you want to change the income from R" + myRow.Cells[2].Value.ToString() + " to R" + txtIncomeAmt.Text + "?";

if (MessageBox.Show(message, "Request", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

try

{

//updates information selected

conn.Open();

OleDbCommand cmd = new OleDbCommand($@"UPDATE Payment SET AmountPaid = {double.Parse(txtIncomeAmt.Text)} WHERE SaleID = {myRow.Cells[3].Value}", conn);

cmd.ExecuteNonQuery();

conn.Close();

WriteLog(logtext, oldVal, newVal);

ListCurrentMonthIncome();

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

else

{

MessageBox.Show(myRow.Cells[2].Value.ToString() + " was not updated", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

//refresh page, and set all to original state

dgvIncomes.Refresh();

txtIncomeAmt.Text = "";

txtDescr.Text = "";

btnEditIncome.Visible = false;

btnDeleteIncome.Visible = false;

btnUpdate.Visible = false;

txtDescr.Enabled = true;

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Delete selected item\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

{

try

{

DataGridViewRow myRow = dgvIncomes.SelectedRows[0];

double oldVal = Convert.ToDouble(myRow.Cells[2].Value);

double newVal = 0;

string logtext = $"Remove payment of {myRow.Cells[1].Value.ToString()} of R{myRow.Cells[2].ToString()} paid on: {myRow.Cells[0].Value.ToString()}?";

string message = "Are you sure that you would like to remove the payment made by " + myRow.Cells[1].Value.ToString() + " of R" + myRow.Cells[2].Value.ToString() + " paid on " + myRow.Cells[0].Value.ToString() + "?";

if (MessageBox.Show(message, "Request", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

try

{

conn.Open();

OleDbCommand cmd = new OleDbCommand($@"DELETE FROM Payment WHERE SaleID = {myRow.Cells[3].Value}", conn);

cmd.ExecuteNonQuery();

conn.Close();

WriteLog(logtext, oldVal, newVal);

ListCurrentMonthIncome();

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

else

{

MessageBox.Show(myRow.Cells[1].Value.ToString() + " was not removed", "Failed", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

//refresh page, and set all to original state

dgvIncomes.Refresh();

txtIncomeAmt.Text = "";

txtDescr.Text = "";

btnEditIncome.Visible = false;

btnDeleteIncome.Visible = false;

btnUpdate.Visible = false;

txtDescr.Enabled = true;

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//-----------------------------------------------------------------------------------------------------END TAB PAGE 4-----------------------------------------------------------------------------------------------

//----------------------------------------------------------------------------------------------------TREATMENTS TAB INFORMATION (TAB PAGE 5)-------------------------------------------------------------------------------------------

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Method that displays information from the database in the listBox on tab page 2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

private void ReadTreatments()

{

//Clears the contens of the Treatments listBox

lbxTreatments.Items.Clear();

try

{

//Opens the DB connection

conn.Open();

//Command and datareader to read treatments from the DB into the listBox

using (OleDbCommand command = new OleDbCommand())

{

command.Connection = conn;

command.CommandText = "SELECT Description FROM Treatments";

//reads information from the database

using (OleDbDataReader reader = command.ExecuteReader())

{

while (reader.Read())

{

lbxTreatments.Items.Add(reader["Description"].ToString());

}

}

}

//Close DB connection

conn.Close();

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

private void dateExpenseSelect\_DateChanged(object sender, DateRangeEventArgs e)

{

if (!edit)

ListCurrentMonthExpsenses();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Method to remove a treatment when you double click the treatments listbox

private void lbxTreatments\_DoubleClick(object sender, EventArgs e)

{

if (lbxTreatments.SelectedItem != null)

{

string treatName = "";

foreach (string val in lbxTreatments.SelectedItems)

{

treatName = val;

}

if (MessageBox.Show("Do you wish to remove " + treatName + " as a treatment?", "Request", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

try

{

conn.Open();

//OleDbDataAdapter adapter = new OleDbDataAdapter($@"SELECT ExpenseDescript FROM ExpenseType ORDER BY ExpenseDescript ASC", conn);

OleDbCommand cmd = new OleDbCommand($@"DELETE FROM Treatments WHERE Description = '{treatName}'", conn);

cmd.ExecuteNonQuery();

MessageBox.Show(treatName + " was successfully removed as a treatment", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

conn.Close();

ReadExpenseTypesEdit();

lbxTreatments.Items.Clear();

ReadTreatments();

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

else

{

MessageBox.Show(treatName + " was not removed", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

else

{

MessageBox.Show("You have not selected a valid expense type to remove", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Button click adds a treatment into the database\*\*\*\*\*\*\*\*\*\*\*\*\*//

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

{

try

{

if (((txtTreatmentDescription.Text) == null) && ((txtTreatmentCost.Text == null)))

{

MessageBox.Show("Please enter a treatment description or treatment cost", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

if (MessageBox.Show("Are you sure you want to add " + txtTreatmentDescription.Text + " as an expense", "Request", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

conn.Open();

OleDbCommand cmd = new OleDbCommand($@"INSERT INTO Treatments (Description, TreatmentCost) VALUES ('{txtTreatmentDescription.Text}', {double.Parse(txtTreatmentCost.Text)})", conn);

cmd.ExecuteNonQuery();

conn.Close();

txtTreatmentDescription.Text = null;

txtTreatmentCost.Text = null;

lbxTreatments.Items.Clear();

ReadTreatments();

}

else

{

MessageBox.Show(txtTreatmentDescription.Text + " was not added", "Treatment not added", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

IEReport myReport = new IEReport();

myReport.Show();

}

//-----------------------------------------------------------------------------------------------------------END TAB PAGE 5--------------------------------------------------------------------------------------

}

}

# Login Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using System.Data.OleDb;

using System.Net.Mail;

namespace Chalique\_Nail\_Studio

{

public partial class Login : Form

{

private int inval;

private OleDbConnection conn;

public bool Canceled { get; set; }

public int acclvl { get; set; }

public string user { get; set; }

public Login(OleDbConnection con)

{

//----- Default constructor for general login -----//

InitializeComponent();

Canceled = false;

conn = con;

}

private void edtPass\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

btnLogin\_Click(this, new EventArgs());

}

}

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

{

bool valid = false;

valid = login(); // Call login method

if (valid == true)

{

// If login is valid close the login dialog

this.Close();

}

else

{

// If login fails then give an error and increment fail counter

MessageBox.Show("The login details are invalid!", "Login Failed", MessageBoxButtons.OK, MessageBoxIcon.Error);

inval++;

}

if (inval >= 3)

{

// If login fails 3 times Notify senior manager

try

{

//------------------------------------------------Email functionality added here------------------------------------------------

//Reads email and password from a textfile

string details = "";

string email = "";

string password = "";

SmtpClient client = new SmtpClient("smtp.gmail.com", 587);

client.EnableSsl = true;

client.Timeout = 60000;

client.Port = 587;

client.DeliveryMethod = SmtpDeliveryMethod.Network;

client.UseDefaultCredentials = false;

string DBloc = Directory.GetCurrentDirectory();

if (!File.Exists(DBloc + @"\ChaliqueNailStudio.accdb"))

DBloc = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + @"\Chalique Nail Studio";

if (File.Exists(DBloc + "\\emailDetails.cns"))

{

StreamReader myReader = new StreamReader(DBloc + "\\emailDetails.cns");

while (!myReader.EndOfStream)

{

details = myReader.ReadLine();

}

string[] allDetails = details.Split(':');

email = allDetails[0].ToString();

password = allDetails[1].ToString();

myReader.Close();

client.Credentials = new System.Net.NetworkCredential(email, password);

MailMessage msg = new MailMessage();

msg.To.Add("chaliquenailstudio@gmail.com");

msg.From = new MailAddress("chaliquenailstudio@gmail.com");

//change email heading and body. Body reads from textfile when it is created

msg.Subject = "System login failure";

msg.Body = $"Login failed 3 times:\r\nDate of attempt:\t{DateTime.Now.ToString("HH:mm:ss dd MMMM yyyy")}\r\nPerson making attempt:\t{edtUser.Text}";

client.SendMailAsync(msg);

}

}

catch (Exception ef)

{

MessageBox.Show(ef.Message);

}

Canceled = true;

this.Close();

}

}

private bool login()

{

//----- This method compares the details given to the employee entries from the file -----//

conn.Open();

OleDbCommand cmd = new OleDbCommand($"SELECT \* FROM Employees WHERE Username = '{edtUser.Text}'", conn);

OleDbDataReader read = cmd.ExecuteReader();

if (read.Read())

if (!read["EmployeeName"].Equals(null))

{

if (read["Password"].ToString().Equals(edtPass.Text))

{

user = read["EmployeeName"].ToString();

acclvl = int.Parse(read["Access"].ToString());

conn.Close();

return true;

}

}

conn.Close();

return false;

}

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

{

// Set a check to see whether login succeeded and close the dialog

Canceled = true;

this.Close();

}

}

}

# User Control Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using System.Data.OleDb;

namespace Chalique\_Nail\_Studio

{

public partial class User\_control : Form

{

private int emplvl;

private int tabLock = 0;

private OleDbConnection conn;

public User\_control(OleDbConnection con)

{

InitializeComponent();

txtPassword.PasswordChar = '\*';

radioButton2.Checked = true;

tabControl1.SelectTab(0); // Change to first tab and disable other tabs

(tabControl1.TabPages[1] as TabPage).Enabled = false;

(tabControl1.TabPages[2] as TabPage).Enabled = false;

conn = con;

}

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

{

//----- Set the new employee type and switch tabs -----//

if (radioButton1.Checked)

{

emplvl = 1; // Set new employee level as Owner

}

else

{

emplvl = 2; // Set new employee level as general staff

cbxTech.Checked = true;

}

// Change to second tab and disable other tabs

(tabControl1.TabPages[0] as TabPage).Enabled = false;

(tabControl1.TabPages[1] as TabPage).Enabled = true;

tabLock = 1;

tabControl1.SelectTab(1);

}

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

{

//----- Switch to the Remove Employee tab -----//

lbxEmp.Items.Clear();

lbxEmp.Items.Add("ID \t Name");

if (radioButton1.Checked)

{

emplvl = 1;

}

else

{

emplvl = 2;

}

(tabControl1.TabPages[0] as TabPage).Enabled = false;

(tabControl1.TabPages[2] as TabPage).Enabled = true;

tabLock = 2;

tabControl1.SelectTab(2);

listEmp();

}

// ADD EMPLOYEE /////////////////////////////////////////////////////////////////////////////////////////

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

{

//----- Add the employee to the file -----//

string emps = "";

switch (emplvl)

{

case 1: emps = "an owner";

break;

case 2: emps = "a technician";

break;

}

try

{

// Check the details and prompt to make sure the employee must be added

if (!txtUser.Text.Equals("") && !txtPassword.Text.Equals(""))

{

string tech;

if (cbxTech.Checked && emplvl == 1) { tech = "with technician status"; }

else tech = "";

if (MessageBox.Show($"Are you sure you want to add employee {txtUser.Text} as {emps} with password {txtPassword.Text} {tech}?",

"New Employee Confirmation", MessageBoxButtons.YesNo) == DialogResult.Yes)

{

conn.Open();

OleDbCommand cmd = new OleDbCommand($"SELECT \* FROM Employees WHERE Username = '{txtUser.Text}'", conn);

OleDbDataReader read = cmd.ExecuteReader();

if (read.Read())

{

throw new Exception("This employee Username is already in use!");

}

if (txtPassword.Text.Length > 32)

{

throw new Exception("Your Password is too long.");

}

cmd = new OleDbCommand($@"INSERT INTO Employees (EmployeeName, [Password], [Username], [Access], IsTech)

VALUES('{txtName.Text}', '{txtPassword.Text}', '{txtUser.Text}', {emplvl}, {cbxTech.Checked})", conn);

cmd.ExecuteNonQuery();

conn.Close();

if (MessageBox.Show("Successfuly added " + txtName.Text + " as an " + emps + ".\r\nDo you wish to continue?", "Success", MessageBoxButtons.YesNo) == DialogResult.No)

this.Close();

}

btnAddCancel\_Click(sender, e);

}

else MessageBox.Show("Please make sure all fields are filled in", "Empty input");

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show("Please try again." + ex.Message, "There was a problem");

}

}

// EMPLOYEE REMOVAL /////////////////////////////////////////////////////////////////////////////////////

private void listEmp()

{

//----- List the employees in a listBox -----//

// List employees of the selected level from the senior employee file

conn.Open();

OleDbCommand cmd = new OleDbCommand($"SELECT \* FROM Employees WHERE Access = {emplvl}", conn);

OleDbDataReader read = cmd.ExecuteReader();

while (read.Read())

{

lbxEmp.Items.Add(read["Username"].ToString() + "\t" + read["EmployeeName"]);

}

conn.Close();

}

private void lbxEmp\_SelectedIndexChanged(object sender, EventArgs e)

{

if (lbxEmp.SelectedIndex != -1)

btnRem.Visible = true;

}

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

{

//----- Remove the selected emplyee -----//

if (lbxEmp.SelectedIndex < 1)

MessageBox.Show("No user has been selected!");

else

{

string t = Convert.ToString(lbxEmp.SelectedItem).Split('\t')[0];

conn.Open();

OleDbCommand cmd = new OleDbCommand($"DELETE \* FROM Employees WHERE Username = '{t}'", conn);

cmd.ExecuteNonQuery();

conn.Close();

MessageBox.Show("The employee has been removed.", "Romoval success");

lbxEmp.Items.Clear();

lbxEmp.Items.Add("ID \t Name");

listEmp();

}

}

// TAB CANCEL ///////////////////////////////////////////////////////////////////////////////////////////

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

{

//----- Cancel and go back to the first tab -----//

txtUser.Clear();

txtName.Clear();

txtPassword.Clear();

(tabControl1.TabPages[0] as TabPage).Enabled = true;

(tabControl1.TabPages[1] as TabPage).Enabled = false;

tabLock = 0;

tabControl1.SelectTab(0);

}

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

{

//----- Cancel and go back to the first tab -----//

lbxEmp.Items.Clear();

btnRemove.Visible = false;

(tabControl1.TabPages[0] as TabPage).Enabled = true;

(tabControl1.TabPages[2] as TabPage).Enabled = false;

tabLock = 0;

tabControl1.SelectTab(0);

}

private void tabControl1\_Selecting(object sender, TabControlCancelEventArgs e)

{

if (tabControl1.SelectedIndex != tabLock)

e.Cancel = true;

}

private void cbxPassword\_CheckedChanged(object sender, EventArgs e)

{

if(cbxPassword.Checked == true)

{

txtPassword.PasswordChar = '\0';

//show text of password

}

else

{

txtPassword.PasswordChar = '\*';

//show password as asterisks

}

}

}

}

# Client Management Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

using System.IO;

using Chalique\_Nail\_Studio.Properties;

namespace Chalique\_Nail\_Studio

{

public partial class Client\_Management : Form

{

private OleDbConnection conn;

private string DBloc;

private bool selected;

private string clientName;

private int clientID;

public Client\_Management(OleDbConnection con)

{

// Component initialization

InitializeComponent();

conn = con;

DBloc = Directory.GetCurrentDirectory();

try

{

if (!File.Exists(DBloc + @"\ChaliqueNailStudio.accdb"))

DBloc = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + @"\Chalique Nail Studio";

// Name textbox autocomplete settings

txtName.AutoCompleteMode = AutoCompleteMode.SuggestAppend;

txtName.AutoCompleteSource = AutoCompleteSource.CustomSource;

txtDelName.AutoCompleteMode = AutoCompleteMode.SuggestAppend;

txtDelName.AutoCompleteSource = AutoCompleteSource.CustomSource;

txtName.Focus();

selected = false;

// Load names into textboxes

string[] arr = SuggestNames();

if (arr != null)

{

AutoCompleteStringCollection collection = new AutoCompleteStringCollection();

collection.AddRange(arr);

txtName.AutoCompleteCustomSource = collection;

txtDelName.AutoCompleteCustomSource = collection;

}

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

try

{

if(selected/\*txtDelName.Text != ""\*/)

{

conn.Open();

OleDbCommand cmnd;

OleDbDataReader reader;

cmnd = new OleDbCommand("SELECT BookingID from Bookings WHERE ClientID = " + clientID, conn);

reader = cmnd.ExecuteReader();

//reader.Read();

int bID;

while (reader.Read())

{

bID = int.Parse(reader["BookingID"].ToString());

cmnd = new OleDbCommand("SELECT DateOfPayment FROM Payment WHERE BookingID = " + bID, conn);

reader = cmnd.ExecuteReader();

if (reader.Read())

throw new Exception("This client has had a booking finished on " + reader[0].ToString() + " and can not be removed.");

cmnd = new OleDbCommand("DELETE \* FROM TreatmentDetailPerBooking WHERE BookingID = " + bID, conn);

cmnd.ExecuteNonQuery();

cmnd = new OleDbCommand("DELETE \* FROM Bookings WHERE BookingID = " + bID, conn);

cmnd.ExecuteNonQuery();

}

cmnd = new OleDbCommand("DELETE \* FROM Clients WHERE ClientID = " + clientID, conn);

cmnd.ExecuteNonQuery();

MessageBox.Show("Delete Successful", "Success", MessageBoxButtons.OK, MessageBoxIcon.Information);

conn.Close();

txtDelName.Text = "";

}

else

{

MessageBox.Show("Please enter the client name you wish to delete", "Request", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

catch(Exception ex)

{

conn.Close();

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

try

{

if (selected/\*txtName.Text != "" && txtNum.Text != "" && txtEmail.Text != ""\*/)

{

conn.Open();

OleDbCommand cmnd = new OleDbCommand(@"UPDATE Clients set ClientName = '" + txtName.Text + "', TelNumber = '" + txtNum.Text + "', EmailAddress = '" + txtEmail.Text + "' WHERE ClientID = " + clientID, conn);

cmnd.ExecuteNonQuery();

MessageBox.Show("Client Updated", "Success", MessageBoxButtons.OK, MessageBoxIcon.Information);

conn.Close();

}

else

{

MessageBox.Show("Please fill all the textboxes", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private string[] SuggestNames()

{

conn.Open();

OleDbCommand cmd = new OleDbCommand("SELECT ClientName FROM Clients ORDER BY ClientName", conn);

OleDbDataReader reader = cmd.ExecuteReader();

List<string> names = new List<string>();

while (reader.Read())

{

names.Add(reader[0].ToString());

}

conn.Close();

if (names.Count > 0)

return names.ToArray();

else return null;

}

private void txtName\_KeyDown(object sender, KeyEventArgs e)

{

try

{

if (e.KeyCode == Keys.Enter)

selectClient(sender);

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void selectClient(object sender)

{

// We use a substitute textbox so this method can be used both for change and delete

try

{

TextBox t = (sender as TextBox);

if (t.Text != "")

{

conn.Open();

OleDbCommand cmd = new OleDbCommand($"SELECT \* FROM Clients WHERE ClientName LIKE '%{t.Text}%'", conn);

OleDbDataReader reader = cmd.ExecuteReader();

if (reader.Read())

{

if (t.Name == "txtName")

{

txtEmail.Text = reader["EmailAddress"].ToString();

txtNum.Text = reader["TelNumber"].ToString();

}

clientName = t.Text;

clientID = int.Parse(reader["ClientID"].ToString());

selected = true;

}

conn.Close();

}

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void tabControl1\_SelectedIndexChanged(object sender, EventArgs e)

{

try

{

if (tabControl1.SelectedIndex == 0)

txtName.Focus();

else

txtDelName.Focus();

}

catch (Exception ex)

{

MessageBox.Show("There was an error\n" + ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

}

# Change Email Messages Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

namespace Chalique\_Nail\_Studio

{

public partial class Change\_Email\_Message : Form

{

private string DBloc;

private string initialmessage;

private string reminder;

public Change\_Email\_Message()

{

InitializeComponent();

DBloc = Directory.GetCurrentDirectory();

if (!File.Exists(DBloc + @"\ChaliqueNailStudio.accdb"))

DBloc = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + @"\Chalique Nail Studio";

if (File.Exists(DBloc + @"\EmailMessage.cns"))

{

StreamReader r = File.OpenText(DBloc + @"\EmailMessage.cns");

initialmessage = r.ReadToEnd();

r.Close();

}

else

{

initialmessage =

@"Dear Client

Your booking at Chalique Nail Studio has been confirmed. The details of the booking will follow below.

We now take CASH AND CARD😬😬

Thank you

Have a lovely day🌸

Chalique Nail Studio";

}

txtEmail.Text = initialmessage;

if (File.Exists(DBloc + @"\EmailReminder.cns"))

{

StreamReader r = File.OpenText(DBloc + @"\EmailReminder.cns");

reminder = r.ReadToEnd();

r.Close();

}

else

{

reminder =

@"Dear Client

Please remember your appointment at Chalique Nail Studio for tomorrow💅🏼

We now take CASH AND CARD😬😬

Thank you

Have a lovely day🌸";

}

txtReminder.Text = reminder;

txtEmail.Focus();

txtEmail.SelectionStart = txtEmail.Text.Length - 1;

txtEmail.SelectionLength = 0;

}

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

{

//Write the new text to the EmailMessage file

StreamWriter w = File.CreateText(DBloc + @"\\EmailMessage.cns");

w.Write(txtEmail.Text);

w.Close();

//Write the new text to the EmailReminder file

w = File.CreateText(DBloc + @"\\EmailReminder.cns");

w.Write(txtReminder.Text);

w.Close();

MessageBox.Show("Email messages updated", "Success", MessageBoxButtons.OK, MessageBoxIcon.Information);

this.Close();

}

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

{

this.Close();

}

}

}

# Edit Email Settings Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

namespace Chalique\_Nail\_Studio

{

public partial class Edit\_email\_settings : Form

{

private string email;

private string password;

private string details;

private string path;

public Edit\_email\_settings()

{

InitializeComponent();

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Auto loads the email address and password from the textfile into the textboxes\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

path = Path.Combine(Directory.GetCurrentDirectory() + "\\emailDetails.cns");

details = "";

StreamReader myReader = new StreamReader(path);

while(!myReader.EndOfStream)

{

details = myReader.ReadLine();

}

string[] allDetails = details.Split(':');

email = allDetails[0].ToString();

password = allDetails[1].ToString();

txtEmail.Text = email;

txtPassword.Clear();

txtPassword.Text = password;

txtPassword.PasswordChar = '\*';

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

}

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

{

string[] allDetails = details.Split(':');

StreamWriter myWriter = new StreamWriter(path);

allDetails[0] = txtEmail.Text;

allDetails[1] = txtPassword.Text;

myWriter.WriteLine(allDetails[0] + ":" + allDetails[1]);

myWriter.Close();

this.Close();

}

private void cBxShowPassword\_CheckedChanged(object sender, EventArgs e)

{

if(cBxShowPassword.Checked == true)

{

txtPassword.PasswordChar = '\0';

//show text of password

}

else

{

txtPassword.PasswordChar = '\*';

//show password as asterisks

}

}

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

{

this.Close();

}

}

}

# Report Viewer

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Chalique\_Nail\_Studio

{

public partial class IEReport : Form

{

public IEReport()

{

InitializeComponent();

}

private void crystalReportViewer1\_Load(object sender, EventArgs e)

{

crystalReportViewer1.Refresh();

crystalReportViewer1.RefreshReport();

}

}

}