**Appendence**

**Landing Page**

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.Messaging;

namespace pinnotedeskup

{

public partial class giris : Form

{

public giris()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

kayit\_ekran\_2 formm2 = new kayit\_ekran\_2();

formm2.Show();

this.Hide();

}

private void button2\_Click(object sender, EventArgs e)

{

hesaba\_giris\_ekranı\_3 formm3 = new hesaba\_giris\_ekranı\_3();

formm3.Show();

this.Hide();

}

}

}

**Register Page**

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 static pinnotedeskup.Program;

using System.Text.RegularExpressions;

using System.Data.SqlClient;

namespace pinnotedeskup

{

public partial class kayit\_ekran\_2 : Form

{

//DEĞİŞKEN ATAMALARI

string pas\_ctrl = @"^(?=.\*[0-9])(?=.\*[\W\_])(?!.\*\s).{8,}$";

string tel\_ctrl = @"^(?:\+?90|0)?[ -]?\d{3}[ -]?\d{3}[ -]?\d{4}$";

bool name\_ctrl = true;

bool surname\_ctrl = true;

bool email\_ctrl = true;

bool password\_ctrl = true;

bool telnum\_ctrl = true;

bool birth\_ctrl = true;

private DateTime dateTimePicker1\_Value;

SqlConnection baglanti = new SqlConnection("Data Source = SEDEN\\SQLEXPRESS; Initial Catalog = PinNoteDB; Integrated Security = True; Encrypt=False;");

public kayit\_ekran\_2()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(name\_box.Text))

{

MessageBox.Show("İsim boş bırakılamaz");

name\_box.Focus();

return;

}

if (!Regex.IsMatch(name\_box.Text, @"^[a-zA-Z]+\s\*[a-zA-Z]\*$"))

{

MessageBox.Show("İsim özel karakter veya boşluk içeremez");

name\_box.Text = "";

name\_box.Focus();

return;

}

// Soyisim kontrolü

if (string.IsNullOrWhiteSpace(surname\_box.Text))

{

MessageBox.Show("Soyisim boş bırakılamaz");

surname\_box.Focus();

return;

}

if (!Regex.IsMatch(surname\_box.Text, @"^[a-zA-Z]+\s\*[a-zA-Z]\*$"))

{

MessageBox.Show("Soyisim özel karakter ve boşluk içeremez");

surname\_box.Text = "";

surname\_box.Focus();

return;

}

// Şifre kontrolü

if (string.IsNullOrWhiteSpace(pass\_box.Text))

{

MessageBox.Show("Şifre boş bırakılamaz");

pass\_box.Focus();

return;

}

if (!Regex.IsMatch(pass\_box.Text, pas\_ctrl))

{

MessageBox.Show("Parola uygun değil");

pass\_box.Text = "";

pass\_box.Focus();

return;

}

// E-posta kontrolü

if (string.IsNullOrWhiteSpace(mail\_box.Text))

{

MessageBox.Show("E-posta boş bırakılamaz");

mail\_box.Focus();

return;

}

//girilen veriler uygunsa db kontrolü yap

if (name\_ctrl && surname\_ctrl && email\_ctrl && password\_ctrl && telnum\_ctrl && birth\_ctrl)

{

//mail db da mevcut mu

string sorgu = "SELECT COUNT(\*) FROM USER\_DATA WHERE E\_MAIL = @Email";

SqlCommand command = new SqlCommand(sorgu, baglanti);

command.Parameters.AddWithValue("@Email", mail\_box.Text);

try

{

baglanti.Open();

int count = (int)command.ExecuteScalar();

if (count > 0)

{

email\_ctrl = false;

MessageBox.Show("mail daha önceden kullanılmış");

name\_box.Text = ""; mail\_box.Focus();

}

else

{

email\_ctrl = true;

}

}

catch (Exception ex)

{

MessageBox.Show("Error: " + ex.Message);

}

//tel no db mevcut mu

string sorgu2 = "SELECT COUNT(\*) FROM USER\_DATA WHERE PHONE = @Phone";

SqlCommand command1 = new SqlCommand(sorgu2, baglanti);

command1.Parameters.AddWithValue("@Phone", phone\_num\_box.Text);

try

{

baglanti.Open();

int count = (int)command1.ExecuteScalar();

if (count > 0)

{

telnum\_ctrl = false;

MessageBox.Show("telefon numarası daha önceden kullanılmış");

phone\_num\_box.Text = "";

phone\_num\_box.Focus();

}

else

{

if (Regex.IsMatch(phone\_num\_box.Text, tel\_ctrl)) { telnum\_ctrl = true; }//tel no özel karakter içeriyo mu

else { MessageBox.Show("Phone number is incorrect or missing"); phone\_num\_box.Text = ""; phone\_num\_box.Focus(); }

}

}

catch (Exception ex)

{

Console.WriteLine("Error: " + ex.Message);

}

//birth kontrol

if (dateTimePicker1\_Value < DateTime.Today) { birth\_ctrl = true; }

else

{

MessageBox.Show("Date of birth cannot be in the future tense", "Invalid birth date");

}

//bilgilerin uygun olduğu durumda kayıt oluştur ve ana sayfaya yönlendir

if (name\_ctrl && surname\_ctrl && email\_ctrl && password\_ctrl && telnum\_ctrl && birth\_ctrl)

{

user\_data.Name = name\_box.Text;

user\_data.Surname = surname\_box.Text;

user\_data.Email = mail\_box.Text;

user\_data.Password = pass\_box.Text;

user\_data.PhoneNumber = phone\_num\_box.Text;

user\_data.BirthDate = birth.Value;

//tabloya kullanıcı bilgilerini ekle

string insertdata = "INSERT INTO USER\_DATA (NAME, SURNAME, MAIL, PHONE, BIRTH\_DATE, REGISTER\_DATE) VALUES (@NAME, @SURNAME, @MAIL, @PHONE, @BIRTH, @REGISTER)";

using (SqlCommand command2 = new SqlCommand(insertdata, baglanti))

{

command2.Parameters.AddWithValue("@NAME", user\_data.Name);

command2.Parameters.AddWithValue("@SURNAME", user\_data.Surname);

command2.Parameters.AddWithValue("@MAIL", user\_data.Email);

command2.Parameters.AddWithValue("@PHONE", user\_data.PhoneNumber);

command2.Parameters.AddWithValue("@BIRTH", user\_data.BirthDate);

command2.Parameters.AddWithValue("@REGISTER", DateTime.Now);

command2.ExecuteNonQuery();

}

//TABLOYA EKLENDİKTEN SONRA ID BİLGİSİNİ ÇEK

string idcekme = "SELECT ID FROM USER\_DATA WHERE MAIL = @Email";

using (SqlCommand command3 = new SqlCommand(idcekme, baglanti))

{

command3.Parameters.AddWithValue("@Email", user\_data.Email);

user\_data.Id = Convert.ToInt32(command3.ExecuteScalar());

}

//MessageBox.Show(Convert.ToString(user\_data.Id));

//tabloya password ekleme

string insertpassword = "INSERT INTO USER\_PASSWORD (ID, PASSWORD, LAST\_UPDATE) VALUES (@ID, @PASSWORD, @LAST\_UPDATE)";

using (SqlCommand command3 = new SqlCommand(insertpassword, baglanti))

{

command3.Parameters.AddWithValue("@ID", user\_data.Id);

command3.Parameters.AddWithValue("@PASSWORD", user\_data.Password);

command3.Parameters.AddWithValue("@LAST\_UPDATE", DateTime.Now);

command3.ExecuteNonQuery();

}

//yeni kullanıcı için tabloları oluştur

// Tabloları oluştur

string note\_table = $"NOTES\_{user\_data.Id}";

string create\_note\_table = $"CREATE TABLE {note\_table} ({"NOTE\_ID INT PRIMARY KEY, EDIT BIT"})";

using (SqlCommand command4 = new SqlCommand(create\_note\_table, baglanti))

{

command4.ExecuteNonQuery();

}

string reminder\_table = $"REMINDER\_{user\_data.Id}";

string create\_reminder\_table = $"CREATE TABLE {reminder\_table} ({"REMINDER\_ID INT PRIMARY KEY, REMINDER\_EDIT BIT"})";

using (SqlCommand command5 = new SqlCommand(create\_reminder\_table, baglanti))

{

command5.ExecuteNonQuery();

}

string friends\_table = $"FRIENDS\_{user\_data.Id}";

string create\_friends\_table = $"CREATE TABLE {friends\_table} ({"FriendID INT, AddedDate DATETIME"})";

using (SqlCommand command6 = new SqlCommand(create\_friends\_table, baglanti))

{

command6.ExecuteNonQuery();

}

string requests\_table = $"REQUESTS\_{user\_data.Id}";

string create\_requests\_table = $"CREATE TABLE {requests\_table} ({"RequesterID INT, RequestDate DATETIME"})";

using (SqlCommand command7 = new SqlCommand(create\_requests\_table, baglanti))

{

command7.ExecuteNonQuery();

}

//ilk not ve hatırlatıcıyı ekle

string firstnote\_table = $"NOTES\_{user\_data.Id}";

string firstnoteadd = $"INSERT {firstnote\_table} (NOTE\_ID, EDIT) VALUES (1000003,0)";

using (SqlCommand command8 = new SqlCommand(firstnoteadd, baglanti))

{

command8.ExecuteNonQuery();

}

string firstreminder\_table = $"REMINDER\_{user\_data.Id}";

string firstreminderadd = $"INSERT {firstreminder\_table} (REMINDER\_ID, REMINDER\_EDIT)VALUES (1000,0)";

using (SqlCommand command9 = new SqlCommand(firstreminderadd, baglanti))

{

command9.ExecuteNonQuery();

}

string notify\_table = $"NOTIFY\_{user\_data.Id}";

string notifytable = $@"CREATE TABLE {notify\_table} (BILDIRIM\_ID INT IDENTITY(1,1) PRIMARY KEY, PAYLASAN\_ID INT NOT NULL, BILDIRIM\_DATE DATETIME NOT NULL, PAYLASILAN\_ID INT NOT NULL, PAYLASILAN\_TUR CHAR(1) NOT NULL)";

using (SqlCommand command10 = new SqlCommand(notifytable, baglanti))

{

command10.ExecuteNonQuery();

}

baglanti.Close();

homepage homepage = new homepage();

homepage.Show();

this.Hide();

}

}

}

}

}

**Log In Page**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Security.Policy;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static pinnotedeskup.Program;

namespace pinnotedeskup

{

public partial class hesaba\_giris\_ekranı\_3 : Form

{

/\*kullanicidan bilgileri alma\*/

string password="";

string mail="";

/\*SQL Control\*/

string user\_ID;

string user\_psw;

/\*eslesme durumunu kontrol etme\*/

bool isim;

bool sifre;

/\*hata mesajı icerigi\*/

string warning\_message;

/\*SQL baglantisi\*/

bool psw;

SqlConnection baglanti = new SqlConnection("Data Source = SEDEN\\SQLEXPRESS; Initial Catalog = PinNoteDB; Integrated Security = True; Encrypt=False;");

/\*DB SADECE OKUMA MODUNDA AÇ!!!!!!!!!!\*/

public hesaba\_giris\_ekranı\_3()

{

InitializeComponent();

mailbox.Focus();

}

private void textBox3\_TextChanged(object sender, EventArgs e)

{

mailbox.Focus();

mail= mailbox.Text;

}

private void textBox4\_TextChanged(object sender, EventArgs e)

{

password= textBox4.Text;

}

private void button1\_Click\_1(object sender, EventArgs e)

{

baglanti.Open();

if (mail.Length > 0 && password.Length > 0)

{

/\*mail control\*/

SqlCommand mail\_ctrl = new SqlCommand("SELECT ID FROM USER\_DATA WHERE MAIL = @Mail", baglanti);

mail\_ctrl.Parameters.AddWithValue("@Mail", mail);

SqlDataReader sqlDataReader\_mail = mail\_ctrl.ExecuteReader();

if (sqlDataReader\_mail.Read())

{

user\_data.Id = Convert.ToInt32(sqlDataReader\_mail["ID"]);

}

else

{

//hata msj goster

MessageBox.Show("wrong mail!");

mailbox.Focus();

return;

}

sqlDataReader\_mail.Close();

/\*password control\*/

SqlCommand psw\_ctrl = new SqlCommand("SELECT PASSWORD FROM USER\_PASSWORD WHERE ID = @UserID", baglanti);

psw\_ctrl.Parameters.AddWithValue("@UserID", user\_data.Id);

SqlDataReader sqlDataReader\_psw = psw\_ctrl.ExecuteReader();

if (sqlDataReader\_psw.Read())

{

user\_psw = sqlDataReader\_psw["PASSWORD"].ToString();//eğer değer varsa atama yapacak

}

sqlDataReader\_psw.Close();

/\*giris durumu\*/

if (user\_psw == password)

{

try

{

// SQL komutu

string sql = "SELECT NAME, SURNAME, PHONE, MAIL, BIRTH\_DATE, PHOTO FROM USER\_DATA WHERE ID = @id";

using (SqlCommand vericekme = new SqlCommand(sql, baglanti))

{

// Parametre ekleme

vericekme.Parameters.AddWithValue("@id", user\_data.Id);

using (SqlDataReader reader = vericekme.ExecuteReader())

{

if (reader.Read())

{

user\_data.Name = reader["NAME"].ToString();

user\_data.Surname = reader["SURNAME"].ToString();

user\_data.PhoneNumber = reader["PHONE"].ToString();

user\_data.Email = reader["MAIL"].ToString();

user\_data.BirthDate = (DateTime)reader["BIRTH\_DATE"];

// Fotoğraf okuma, varsa

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

{

byte[] photoArray = (byte[])reader["PHOTO"];

// Fotoğrafı işlemek için gerekli kodlar burada olabilir

}

}

else

{

MessageBox.Show("No data found.");

}

}

}

}

catch (Exception ex)

{

MessageBox.Show("An error occurred: " + ex.Message);

} /\*homepage yonlendir\*/

homepage homepage = new homepage();

homepage.Show();

this.Hide();

}

else

{

//uyari mesaji goster ve yonlendir

MessageBox.Show("Wrong password!");

textBox4.Text = "";

textBox4.Focus();

}

}

else

{

MessageBox.Show("mail or password can not be empty");

}

baglanti.Close();

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

dogrulama\_kodu\_4 dogrulama = new dogrulama\_kodu\_4();

dogrulama.Show();

this.Hide();

}

}

}

**Verify Page**

using System;

using System.Data.SqlClient;

using System.Net;

using System.Net.Mail;

using System.Text;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

using static pinnotedeskup.Program;

namespace pinnotedeskup

{

public partial class dogrulama\_kodu\_4 : Form

{

string mail;

string verify\_code;

string verify\_check;

Random random = new Random();

public dogrulama\_kodu\_4()

{

InitializeComponent();

}

private void button2\_Click(object sender, EventArgs e)

{

verify\_code = verify\_maker(6); // 6 karakterlik doğrulama kodu oluştur

mail =mail\_box.Text;

//email adresi db de mevcut mu kontrol et

SqlConnection baglanti = new SqlConnection("Data Source = SEDEN\\SQLEXPRESS; Initial Catalog = PinNoteDB; Integrated Security = True; Encrypt=False;");

baglanti.Open();

SqlCommand mail\_ctrl = new SqlCommand("SELECT ID FROM USER\_DATA WHERE MAIL = @Mail", baglanti);

mail\_ctrl.Parameters.AddWithValue("@Mail", mail);

SqlDataReader sqlDataReader\_mail = mail\_ctrl.ExecuteReader();

if (sqlDataReader\_mail.Read())

{

// Doğrulama kodunu yolla

SendEmail(verify\_code, mail\_box.Text);

}

else

{

//hata msj goster

MessageBox.Show("wrong mail!");

}

sqlDataReader\_mail.Close();

}

public static string verify\_maker(int verify\_code\_len)

{

const string verify\_char = "1234567890qwertyuopasdfghjklizxcvbnmQWERTYUIOPASDFGHJKLZXCVBNM@\*-+/!?.-\_&%$";

StringBuilder sb = new StringBuilder();

Random random = new Random();

for (int i = 0; i < verify\_code\_len; i++)

{

int idx = random.Next(verify\_char.Length);

sb.Append(verify\_char[idx]);}

return sb.ToString();}

private void SendEmail(string verify\_code, string user\_mail)

{

string subjectEmail = "Verify Code";

string bodyEmail = "Your verify code for reset password: " + verify\_code;

using (SmtpClient smtp = new SmtpClient("smtp.outlook.com")) // SMTP sunucusu

{

smtp.Port = 587; // SMTP port numarası

smtp.Credentials = new NetworkCredential("pinnote\_app@hotmail.com", "XqWchh6s!Y"); // E-posta hesabı bilgileri

smtp.EnableSsl = true; // SSL/TLS

// E-posta oluştur

MailMessage mailMessage = new MailMessage("pinnote\_app@hotmail.com", user\_mail)

{

Subject = subjectEmail,

Body = bodyEmail

};

try

{

smtp.Send(mailMessage);

MessageBox.Show("Mail sent successfully");

}

catch (Exception ex)

{

MessageBox.Show("Error sending email: " + ex.Message);

}

}

}

private void verify\_code\_box\_TextChanged(object sender, EventArgs e)

{

verify\_check = verify\_code\_box.Text;

}

private void button3\_Click(object sender, EventArgs e)

{

if (verify\_code == verify\_check)

{

sifre\_sifirlama\_5 sifre\_Sifirlama\_5 = new sifre\_sifirlama\_5();

sifre\_Sifirlama\_5.Show();

this.Hide();

}

}

}

}

**Password Reset**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static pinnotedeskup.Program;

namespace pinnotedeskup

{

public partial class sifre\_sifirlama\_5 : Form

{

public sifre\_sifirlama\_5()

{

InitializeComponent();

}

private void psw\_button\_Click(object sender, EventArgs e)

{

if(new\_psw1.Text == new\_psw2.Text)

{

//password içeriğinin uygunluğunu kontrol et

//db de password olarak psw1 ata

SqlConnection baglanti = new SqlConnection("Data Source = SEDEN\\SQLEXPRESS; Initial Catalog = PinNoteDB; Integrated Security = True; Encrypt=False;");

baglanti.Open();

SqlCommand VeriGuncelle = new SqlCommand("update PASSWORD\_TABLE PASSWORD=@password,LAST\_UPDATE=@lastupdate where ID=@id", baglanti);

VeriGuncelle.Parameters.AddWithValue("@id", user\_data.Id);

VeriGuncelle.Parameters.AddWithValue("@password", new\_psw1.Text);

VeriGuncelle.Parameters.AddWithValue("@lastupdate", DateTime.Now);

baglanti.Close();

MessageBox.Show("password güncellendi");

}

else

{

MessageBox.Show("Passwordlar eslesmiyor");

new\_psw1.Text = "";

new\_psw2.Text = "";

new\_psw1.Focus();

}

giris giris = new giris();

giris.Show();

this.Hide();

}

}

}

**Homepage**

using pinnotedeskup.wgt;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Security.Cryptography;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static pinnotedeskup.Program;

namespace pinnotedeskup

{

public partial class homepage : Form

{

SqlConnection baglanti = new SqlConnection("Data Source = SEDEN\\SQLEXPRESS; Initial Catalog = PinNoteDB; Integrated Security = True; Encrypt=False;");

public homepage()

{

InitializeComponent();

//MessageBox.Show(user\_data.Name + user\_data.Email);

//note

baglanti.Open();

string table\_name = "NOTES\_" + user\_data.Id;

List<int> note\_id\_list = new List<int>();

List<bool> note\_edit\_list = new List<bool>();

//database veri çekme(note)

notetablewriter();

//database veri çekme (reminder)

remindertablewriter();

//db name çekme

using (SqlCommand name\_ctrl = new SqlCommand("SELECT NAME FROM USER\_DATA WHERE ID = @id", baglanti))

{

name\_ctrl.Parameters.AddWithValue("@id", user\_data.Id);

using (SqlDataReader sqlDataReader\_nameread = name\_ctrl.ExecuteReader())

{

while (sqlDataReader\_nameread.Read())

{

string alinan\_veri = Convert.ToString(sqlDataReader\_nameread["NAME"]);

label2.Text = "Merhaba " + alinan\_veri + "!";

}

}

}

//takvim verisi

List<int> reminderIds = new List<int>();

string reminderTable = $"REMINDER\_{user\_data.Id}";

using (SqlCommand cmd = new SqlCommand($"SELECT REMINDER\_ID FROM {reminderTable}", baglanti))

{

using (SqlDataReader reader = cmd.ExecuteReader())

{

while (reader.Read())

{

reminderIds.Add(reader.GetInt32(0));

}

}

}

foreach (int reminderId in reminderIds)

{

using (SqlCommand innerCmd = new SqlCommand("SELECT REMINDER\_DATE FROM REMINDERS WHERE REMINDER\_ID = @id", baglanti))

{

innerCmd.Parameters.AddWithValue("@id", reminderId);

using (SqlDataReader innerReader = innerCmd.ExecuteReader())

{

while (innerReader.Read())

{

DateTime reminderDate = innerReader.GetDateTime(0);

monthCalendar1.AddBoldedDate(reminderDate);

}

}

}

}

monthCalendar1.UpdateBoldedDates();

//masaustu bildirim

foreach (int reminderId in reminderIds)

{

using (SqlCommand innerCmd = new SqlCommand("SELECT REMINDER\_DATE,REMINDER\_NAME, REMINDER\_BODY FROM REMINDERS WHERE REMINDER\_ID = @id", baglanti))

{

innerCmd.Parameters.AddWithValue("@id", reminderId);

using (SqlDataReader innerReader = innerCmd.ExecuteReader())

{

while (innerReader.Read())

{

DateTime reminderDate = innerReader.GetDateTime(0);

string reminderName = innerReader.GetString(1);

string reminderbody = innerReader.GetString(2);

// Şu anki zaman ile hatırlatıcı tarihi arasındaki fark

TimeSpan difference = reminderDate - DateTime.Now;

// Eğer fark 0'a eşitse bildirim göster

if (difference == TimeSpan.Zero)

{

// Hatırlatıcı zamanı geldiğinde bildirim göster

DialogResult result = MessageBox.Show(reminderName, reminderbody, MessageBoxButtons.OKCancel);

if (result == DialogResult.OK)

{

// "OK" butonuna tıklanırsa yapılacak işlemler

// Örneğin, başka bir formu açabilirsiniz

reminder\_olusturma\_10 rem = new reminder\_olusturma\_10(reminderId);

rem.Show();

this.Hide();

}

else if (result == DialogResult.Cancel)

{

}

}

}

}

}

}

}

//yan menu

private void pictureBox2\_Click(object sender, EventArgs e)

{

//home

homepage homepage = new homepage();

homepage.Show();

this.Hide();

}

private void pictureBox4\_Click(object sender, EventArgs e)

{

//notlar

notes notes = new notes();

notes.Show();

this.Hide();

}

private void pictureBox3\_Click(object sender, EventArgs e)

{

//hatırlatıcı

reminder\_liste reminder\_Liste = new reminder\_liste();

reminder\_Liste.Show();

this.Hide();

}

private void pictureBox6\_Click(object sender, EventArgs e)

{

//arkadaslar

Friends friends = new Friends();

friends.Show();

this.Hide();

}

private void pictureBox7\_Click(object sender, EventArgs e)

{

//profil

profile profile = new profile();

profile.Show();

this.Hide();

}

private void pictureBox1\_Click(object sender, EventArgs e)

{

//home

homepage homepage1 = new homepage();

homepage1.Show();

this.Hide();

}

//

private void button1\_Click\_1(object sender, EventArgs e)

{

//bildirimler

}

private void remindergrid\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

if (remindergrid.Columns[e.ColumnIndex] is DataGridViewButtonColumn && e.RowIndex >= 0)

{

try

{

int reminderIdColumnIndex = remindergrid.Columns["ReminderID"].Index;

if (!int.TryParse(remindergrid.Rows[e.RowIndex].Cells[reminderIdColumnIndex].Value.ToString(), out int reminderId))

{

MessageBox.Show("Reminder ID could not be parsed.");

return;

}

if (remindergrid.Columns[e.ColumnIndex].Name == "rem\_edit")

{

EditReminder(reminderId);

}

else if (remindergrid.Columns[e.ColumnIndex].Name == "delete\_rem")

{

DeleteReminder(reminderId);

remindertablewriter();

}

}

catch (Exception ex)

{

MessageBox.Show("An error occurred: " + ex.Message);

}

}

}

private void EditReminder(int reminderId)

{

reminder\_olusturma\_10 reminder\_Olusturma\_10 = new reminder\_olusturma\_10(reminderId);

reminder\_Olusturma\_10.Show();

this.Hide();

}

private void DeleteReminder(int reminderId)

{

string tablename = $"REMINDER\_{user\_data.Id}";

string sqlsorgustr = $"DELETE FROM {tablename} WHERE REMINDER\_ID = @delete";

try

{

using (SqlCommand cmd = new SqlCommand(sqlsorgustr, baglanti))

{

cmd.Parameters.AddWithValue("@delete", reminderId);

cmd.ExecuteNonQuery();

}

this.Refresh();

MessageBox.Show("Reminder deleted successfully.");

}

catch (Exception ex)

{

MessageBox.Show($"Failed to delete reminder: {ex.Message}");

}

}

private void notegrid\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

if (notegrid.Columns[e.ColumnIndex] is DataGridViewButtonColumn && e.RowIndex >= 0)

{

try

{

int noteIdColumnIndex = notegrid.Columns["NoteID"].Index;

if (!int.TryParse(notegrid.Rows[e.RowIndex].Cells[noteIdColumnIndex].Value.ToString(), out int noteId))

{

MessageBox.Show("Note ID could not be parsed.");

return;

}

if (notegrid.Columns[e.ColumnIndex].Name == "edit")

{

EditNote(noteId);

this.Refresh();

}

else if (notegrid.Columns[e.ColumnIndex].Name == "delete")

{

DeleteNote(noteId);

notetablewriter();

this.Refresh();

}

}

catch (Exception ex)

{

MessageBox.Show("An error occurred: " + ex.Message);

}

}

}

private void EditNote(int noteId)

{

newnote newNoteForm = new newnote(noteId);

newNoteForm.Show();

this.Hide();

}

private void DeleteNote(int noteId)

{

string tablename = $"NOTES\_{user\_data.Id}";

string sqlsorgustr = $"DELETE FROM {tablename} WHERE NOTE\_ID = @delete";

try

{

using (SqlCommand cmd = new SqlCommand(sqlsorgustr, baglanti))

{

cmd.Parameters.AddWithValue("@delete", noteId);

cmd.ExecuteNonQuery();

MessageBox.Show("Note deleted successfully.");

}

}

catch (Exception ex)

{

MessageBox.Show($"Failed to delete note: {ex.Message}");

}

}

private void notetablewriter ()

{

string notetablename = "NOTES\_"+user\_data.Id;

List<int> note\_id\_list = new List<int>();

List<bool> note\_edit\_list = new List<bool>();

using (SqlCommand noteid\_ctrl = new SqlCommand($"SELECT NOTE\_ID, EDIT FROM {notetablename}", baglanti))

{

using (SqlDataReader sqlDataReader\_noteid = noteid\_ctrl.ExecuteReader())

{

while (sqlDataReader\_noteid.Read())

{

note\_id\_list.Add(sqlDataReader\_noteid.GetInt32(0));

note\_edit\_list.Add(sqlDataReader\_noteid.GetBoolean(1));

}

sqlDataReader\_noteid.Close();

}

}

//note tablosundan verileri çekme

foreach (int note\_id in note\_id\_list)

{

SqlCommand table\_read = new SqlCommand("SELECT NOTE\_TITLE, NOTE\_BODY, UPDATE\_TIME FROM NOTES WHERE NOTEID = @id", baglanti);

table\_read.Parameters.AddWithValue("@id", note\_id);

SqlDataReader sqlDataReader\_tableread = table\_read.ExecuteReader();

while (sqlDataReader\_tableread.Read())

{

string title = sqlDataReader\_tableread.GetString(0);

string body = sqlDataReader\_tableread.GetString(1);

DateTime update = sqlDataReader\_tableread.GetDateTime(2);

notegrid.Rows.Add(note\_id, title, body, update, "Edit", "Del");

}

sqlDataReader\_tableread.Close();

}

}

private void remindertablewriter ()

{

//reminder

string reminder\_name = "REMINDER\_" + user\_data.Id;

List<int> reminder\_id = new List<int>();

List<bool> reminder\_edit = new List<bool>();

using (SqlCommand reminderid\_ctrl = new SqlCommand($"SELECT REMINDER\_ID, REMINDER\_EDIT FROM {reminder\_name}", baglanti))

{

using (SqlDataReader sqlDataReader\_reminderid = reminderid\_ctrl.ExecuteReader())

{

while (sqlDataReader\_reminderid.Read())

{

reminder\_id.Add(sqlDataReader\_reminderid.GetInt32(0));

reminder\_edit.Add(sqlDataReader\_reminderid.GetBoolean(1));

}

}

}

//reminder tablosundan verileri çekme

foreach (int i in reminder\_id)

{

SqlCommand table\_read = new SqlCommand("SELECT REMINDER\_NAME, REMINDER\_BODY, REMINDER\_DATE FROM REMINDERS WHERE REMINDER\_ID = @id", baglanti);

table\_read.Parameters.AddWithValue("@id", i);

using (SqlDataReader reader = table\_read.ExecuteReader())

{

while (reader.Read())

{

string title = reader.GetString(0);

string body = reader.GetString(1);

DateTime hatirlatma = reader.GetDateTime(2);

remindergrid.Rows.Add(i, title, body, hatirlatma, "Edit", "Del");

}

}

}

}

private void button1\_Click(object sender, EventArgs e)

{

newnote newnote = new newnote(null);

newnote.Show();

this.Hide();

}

private void button2\_Click(object sender, EventArgs e)

{

reminder\_olusturma\_10 reminder\_Olusturma\_10 = new reminder\_olusturma\_10(null);

reminder\_Olusturma\_10 .Show();

this .Hide();

}

private void pictureBox5\_Click(object sender, EventArgs e)

{

IsteklerEkran isteklerEkran = new IsteklerEkran();

isteklerEkran.Show();

}

}

}

**Notes List**

using Microsoft.Graph.Models;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static pinnotedeskup.Program;

namespace pinnotedeskup

{

public partial class notes : Form

{

SqlConnection baglanti = new SqlConnection("Data Source = SEDEN\\SQLEXPRESS; Initial Catalog = PinNoteDB; Integrated Security = True; Encrypt=False;");

public notes()

{

InitializeComponent();

string table\_name = "NOTES\_" + user\_data.Id;

List<int> note\_id = new List<int>();

List<bool> edit = new List<bool>();

//database veri çekme(reminderid)

baglanti.Open();

SqlCommand id\_ctrl = new SqlCommand($"SELECT Note\_ID, EDIT FROM {table\_name}", baglanti);

SqlDataReader sqlDataReader\_noteid = id\_ctrl.ExecuteReader();

while (sqlDataReader\_noteid.Read())

{

note\_id.Add(sqlDataReader\_noteid.GetInt32(0));

edit.Add(sqlDataReader\_noteid.GetBoolean(1));

}

sqlDataReader\_noteid.Close();

//note tablosundan verileri çekme

foreach (int id in note\_id)

{

string query = "SELECT NOTE\_TITLE, NOTE\_BODY, UPDATE\_TIME, IMP\_LEVEL, CREATER\_ID FROM NOTES WHERE NOTEID = @NoteId";

SqlCommand table\_read = new SqlCommand(query, baglanti);

table\_read.Parameters.AddWithValue("@NoteId", id);

SqlDataReader sqlDataReader\_tableread = table\_read.ExecuteReader();

string title = "", body = "", creater = "";

DateTime update = DateTime.Now;

int imp = 0, createrid = 0;

if (sqlDataReader\_tableread.Read())

{

title = sqlDataReader\_tableread.GetString(0);

body = sqlDataReader\_tableread.GetString(1);

update = sqlDataReader\_tableread.GetDateTime(2);

imp = sqlDataReader\_tableread.GetInt32(3);

createrid = sqlDataReader\_tableread.GetInt32(4);

}

sqlDataReader\_tableread.Close();

if (createrid != 0)

{

SqlCommand creater\_find = new SqlCommand("SELECT NAME, SURNAME FROM USER\_DATA WHERE ID=@CreatorId", baglanti);

creater\_find.Parameters.AddWithValue("@CreatorId", createrid);

SqlDataReader read = creater\_find.ExecuteReader();

if (read.Read())

{

string creatername = read.GetString(0);

string creatersurname = read.GetString(1);

creater = creatername + " " + creatersurname;

}

read.Close();

}

dataGridView.Rows.Add(id, title, body, update, imp, creater,"Edit","Del");

}

}

private void button1\_Click\_1(object sender, EventArgs e)

{

//oluşturma sayfasına yönlendir

newnote newnote = new newnote(null);

newnote.Show();

this.Close();

}

private void pictureBox2\_Click\_1(object sender, EventArgs e)

{

homepage homepage2 = new homepage();

homepage2.Show();

this.Hide();

}

private void pictureBox4\_Click(object sender, EventArgs e)

{

this.Controls.Clear();

this.InitializeComponent();

}

private void pictureBox3\_Click\_1(object sender, EventArgs e)

{

reminder\_liste reminder\_Liste = new reminder\_liste();

reminder\_Liste.Show();

this.Hide();

}

private void pictureBox6\_Click\_1(object sender, EventArgs e)

{

Friends friends = new Friends();

friends.Show();

this.Hide();

}

private void pictureBox7\_Click\_1(object sender, EventArgs e)

{

profile profile2 = new profile();

profile2.Show();

this.Hide();

}

private void dataGridView\_CellContentClick\_1(object sender, DataGridViewCellEventArgs e)

{

if (dataGridView.Columns[e.ColumnIndex] is DataGridViewButtonColumn && e.RowIndex >= 0)

{

int noteIdColumnIndex = 0;

int noteId = Convert.ToInt32(dataGridView.Rows[e.RowIndex].Cells[noteIdColumnIndex].Value);

if (dataGridView.Columns[e.ColumnIndex].Name == "edit")

{

// Düzenleme butonu işlemi

newnote newNoteForm = new newnote(noteId);

newNoteForm.Show();

this.Hide();

}

else if (dataGridView.Columns[e.ColumnIndex].Name == "delete")

{

// Silme butonu işlemi

string deltablename = "NOTES\_" + user\_data.Id;

try

{

using (SqlCommand delnote = new SqlCommand($"DELETE FROM {deltablename} WHERE NOTE\_ID=@notid", baglanti))

{

delnote.Parameters.AddWithValue("@notid", noteId);

delnote.ExecuteNonQuery();

MessageBox.Show("Note deleted successfully.");

}

this.Refresh();

}

catch (Exception ex)

{

MessageBox.Show("Error deleting note: " + ex.Message);

}

}

}

}

}

}

**Create Note**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static pinnotedeskup.Program;

namespace pinnotedeskup

{

public partial class newnote : Form

{

int onemdrc = 1;

int? note\_id=null;

int newnote\_id;

Dictionary<int, bool> paylasim = new Dictionary<int, bool>();

SqlConnection baglanti = new SqlConnection("Data Source = SEDEN\\SQLEXPRESS; Initial Catalog = PinNoteDB; Integrated Security = True; Encrypt=False;");

public newnote(int? noteid)

{

InitializeComponent();

if (noteid!=null)

{

note\_id = noteid.Value;

//not bilgilerini ekrana yazdir

string sql = $"SELECT NOTE\_TITLE, NOTE\_BODY, IMP\_LEVEL FROM NOTES WHERE NOTEID=@id ";

using (SqlCommand sqlCommand = new SqlCommand(sql, baglanti))

{

baglanti.Open();

sqlCommand.Parameters.Add("@id", SqlDbType.Int).Value = note\_id.Value;

SqlDataReader sqlDataReader\_noteread = sqlCommand.ExecuteReader();

while (sqlDataReader\_noteread.Read())

{

note\_title\_box.Text = sqlDataReader\_noteread.GetString(0);

note\_body.Text = sqlDataReader\_noteread.GetString(1);

int imp = sqlDataReader\_noteread.GetInt32(2);

if (imp == 1) { comboBox1.SelectedIndex = 0; }

else if (imp == 2) { comboBox1.SelectedIndex = 1; }

else if (imp == 3) { comboBox1.SelectedIndex = 2; }

else if (imp == 4) { comboBox1.SelectedIndex = 3; }

else if (imp == 5) { comboBox1.SelectedIndex = 4; }

}

baglanti.Close();

}

}

else

{

note\_body.Text = "";

note\_title\_box.Text = "";

}

}

private void button1\_Click(object sender, EventArgs e)

{

int onemdrc = Convert.ToInt32(comboBox1.SelectedValue);

if (note\_id!=null)

{

//değişiklikleri kaydet

if (!string.IsNullOrWhiteSpace(note\_title\_box.Text) && !string.IsNullOrWhiteSpace(note\_body.Text))

{

string updateSql = "UPDATE NOTES SET NOTE\_TITLE = @NoteTitle, NOTE\_BODY = @NoteBody, IMP\_LEVEL = @ImpLevel, UPDATE\_TIME = @UpdateTime WHERE NOTEID = @NoteId;";

using (SqlCommand command = new SqlCommand(updateSql, baglanti))

{

// Parametreleri ayarla

command.Parameters.AddWithValue("@NoteId", note\_id); // Güncellenecek notun ID'si

command.Parameters.AddWithValue("@NoteTitle", note\_title\_box.Text);

command.Parameters.AddWithValue("@NoteBody", note\_body.Text);

command.Parameters.AddWithValue("@ImpLevel", onemdrc); // Önem derecesi

command.Parameters.AddWithValue("@UpdateTime", DateTime.Now); // Güncelleme zamanı

baglanti.Open();

command.ExecuteNonQuery();

baglanti.Close();

MessageBox.Show("Degisiklikler kaydedildi.");

homepage homepage = new homepage();

homepage.Show();

this.Hide();

}

}

}

else

{

if (!string.IsNullOrWhiteSpace(note\_title\_box.Text) && !string.IsNullOrWhiteSpace(note\_body.Text))

{

baglanti.Open();

//notu notlar tablosuna ekler

int impLevel = Convert.ToInt32(comboBox1.SelectedValue);

string insertnote = @"INSERT INTO NOTES (NOTE\_TITLE, NOTE\_BODY, IMP\_LEVEL, UPDATE\_TIME, CREATER\_ID) VALUES (@Notetitle, @Notebody, @Imp\_level, @Updatetime, @Createrid);

SELECT SCOPE\_IDENTITY();"; // son identitiy değeri alır

using (SqlCommand command = new SqlCommand(insertnote, baglanti))

{

command.Parameters.AddWithValue("@Notetitle", note\_title\_box.Text);

command.Parameters.AddWithValue("@Notebody", note\_body.Text);

command.Parameters.AddWithValue("@Imp\_level", impLevel);

command.Parameters.AddWithValue("@Updatetime", DateTime.Now);

command.Parameters.AddWithValue("@Createrid", user\_data.Id);

newnote\_id = Convert.ToInt32(command.ExecuteScalar());

}

//notu kullanıcının tablosuna ekler

string noteidadd = "NOTES\_"+user\_data.Id;

string insertnoteadd = $"INSERT INTO {noteidadd} (NOTE\_ID, EDIT) VALUES (@noteid, @edit)";

using (SqlCommand sqlnote = new SqlCommand(insertnoteadd, baglanti))

{

sqlnote.Parameters.AddWithValue("@noteid", newnote\_id);

sqlnote.Parameters.AddWithValue("@edit", true);

sqlnote.ExecuteNonQuery();

MessageBox.Show("Saved!");

homepage homepage1 = new homepage();

homepage1.Show();

this.Hide();

}

baglanti.Close();

}

else { MessageBox.Show("Title or body can not be empty"); }

}

//tabloya ekleme

foreach (var kvp in paylasim)

{

string tablenamearkds = "NOTES\_" + kvp.Key;

using (SqlCommand arkdsnot = new SqlCommand($"INSERT INTO {tablenamearkds} (NOTE\_ID, EDIT) VALUES (@noteid, @edit)", baglanti))

{

arkdsnot.Parameters.AddWithValue("@noteid", kvp.Key);

arkdsnot.Parameters.AddWithValue("@edit", kvp.Value);

baglanti.Open();

arkdsnot.ExecuteNonQuery();

baglanti.Close();

}

}

}

private void pictureBox2\_Click(object sender, EventArgs e)

{

homepage homepage = new homepage();

homepage.Show();

this.Hide();

}

private void pictureBox4\_Click(object sender, EventArgs e)

{

notes notes = new notes();

notes.Show();

this.Hide();

}

private void pictureBox3\_Click(object sender, EventArgs e)

{

reminder\_liste reminder\_Liste = new reminder\_liste();

reminder\_Liste.Show();

this.Hide();

}

private void pictureBox6\_Click(object sender, EventArgs e)

{

Friends friends = new Friends();

friends.Show();

this.Hide();

}

private void pictureBox7\_Click(object sender, EventArgs e)

{

profile profile = new profile();

profile.Show();

this.Hide();

}

private void button2\_Click(object sender, EventArgs e)

{

int friendctrl = Convert.ToInt32(textBox1.Text);

bool editper = checkBox1.Checked;

List<int> friends = new List<int>();

baglanti.Open();

string friendstable = "FRIENDS\_" + user\_data.Id;

//arkadaşlar listesini çek

using (SqlCommand sqlfriend = new SqlCommand($"SELECT FriendID FROM {friendstable}",baglanti))

{

friends.Add(Convert.ToInt32(sqlfriend.ExecuteScalar()));

}

//yazılan id listede var mı kontrol et

bool eklimi = friends.Contains(friendctrl);

string name = string.Empty;

string surname = string.Empty;

if (eklimi)

{

//isim verisi çekme

using (SqlCommand namecekme = new SqlCommand("SELECT NAME, SURNAME FROM USER\_DATA WHERE ID=@id", baglanti))

{

namecekme.Parameters.AddWithValue ("@id", friendctrl);

SqlDataReader sqlDataReader\_name = namecekme.ExecuteReader();

while (sqlDataReader\_name.Read())

{

name = sqlDataReader\_name.GetString(0);

surname = sqlDataReader\_name.GetString(1);

}

}

paylasim.Add(friendctrl,editper);

MessageBox.Show("Not paylaşıldı!");

}

else

{

MessageBox.Show("Arkadaş listenizde bulunmamaktadır.");

}

baglanti.Close();

}

}

}

**Reminder List**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static pinnotedeskup.Program;

namespace pinnotedeskup

{

public partial class reminder\_liste : Form

{

SqlConnection baglanti = new SqlConnection("Data Source = SEDEN\\SQLEXPRESS; Initial Catalog = PinNoteDB; Integrated Security = True; Encrypt=False;");

public reminder\_liste()

{

InitializeComponent();

string reminderName = "REMINDER\_" + user\_data.Id;

List<int> reminderIds = new List<int>();

List<bool> reminderEdits = new List<bool>();

baglanti.Open();

// ID ve edit bilgilerini çekme

using (SqlCommand command = new SqlCommand($"SELECT REMINDER\_ID, REMINDER\_EDIT FROM {reminderName}", baglanti))

{

using (SqlDataReader reader = command.ExecuteReader())

{

while (reader.Read())

{

reminderIds.Add(reader.GetInt32(0));

reminderEdits.Add(reader.GetBoolean(1));

}

}

}

// Detayları çekme

for (int i = 0; i < reminderIds.Count; i++)

{

using (SqlCommand detailCommand = new SqlCommand("SELECT REMINDER\_NAME, REMINDER\_BODY, REMINDER\_IMPLVL, REMINDER\_DATE, REMINDER\_CREATER FROM REMINDERS WHERE REMINDER\_ID = @id", baglanti))

{

detailCommand.Parameters.AddWithValue("@id", reminderIds[i]);

using (SqlDataReader detailReader = detailCommand.ExecuteReader())

{

while (detailReader.Read())

{

string title = detailReader.GetString(0);

string body = detailReader.GetString(1);

int implvl = detailReader.GetInt32(2);

DateTime reminderDate = detailReader.GetDateTime(3);

int createrId = detailReader.GetInt32(4);

dataGridView.Rows.Add(reminderIds[i], title, body, reminderDate, implvl, createrId, reminderEdits[i], "Del");

}

}

}

}

}

private void button1\_Click\_1(object sender, EventArgs e)

{

reminder\_olusturma\_10 newreminder\_ = new reminder\_olusturma\_10(null);

newreminder\_.Show();

this.Hide();

}

private void pictureBox2\_Click(object sender, EventArgs e)

{

homepage homepage = new homepage();

homepage.Show();

this.Hide();

}

private void pictureBox4\_Click(object sender, EventArgs e)

{

notes notes = new notes();

notes.Show();

this.Hide();

}

private void pictureBox6\_Click(object sender, EventArgs e)

{

Friends friends = new Friends();

friends.Show();

this.Hide();

}

private void pictureBox7\_Click(object sender, EventArgs e)

{

profile profile = new profile();

profile.Show();

this.Hide();

}

private void dataGridView\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

if (dataGridView.Columns[e.ColumnIndex] is DataGridViewButtonColumn && e.RowIndex >= 0)

{

// Sütun isimlerine göre işlem türünü kontrol et

if (dataGridView.Columns[e.ColumnIndex].Name == "edit")

{

// Düzenleme işlemi

int reminderId = Convert.ToInt32(dataGridView.Rows[e.RowIndex].Cells[0].Value);

reminder\_olusturma\_10 reminder\_Olusturma\_10 = new reminder\_olusturma\_10(reminderId);

reminder\_Olusturma\_10.Show();

this.Hide();

}

else if (dataGridView.Columns[e.ColumnIndex].Name == "delete")

{

// Silme işlemi

int reminderId = Convert.ToInt32(dataGridView.Rows[e.RowIndex].Cells[0].Value);

string remdeletename = "REMINDER\_" + user\_data.Id;

try

{

using (SqlCommand delreminder = new SqlCommand($"DELETE FROM {remdeletename} WHERE REMINDER\_ID=@remid", baglanti))

{

delreminder.Parameters.AddWithValue("@remid", reminderId);

delreminder.ExecuteNonQuery();

MessageBox.Show("Reminder deleted successfully.");

}

this.Refresh();

}

catch (Exception ex)

{

MessageBox.Show("Error deleting reminder: " + ex.Message);

}

}

}

}

}

}

**Reminder Create**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Net.Mail;

using System.Net;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static pinnotedeskup.Program;

using static System.Windows.Forms.VisualStyles.VisualStyleElement.Button;

using Microsoft.Graph.Models;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

using System.Globalization;

namespace pinnotedeskup

{

public partial class reminder\_olusturma\_10 : Form

{

int onemdrc=1;

string mailaddress = string.Empty;

int? reminderID = null;

List<int> paylasilanlar = new List<int>();

List<int> addedfriendsids = new List <int>();

bool duzenleme;

bool mail;

List<int> friends = new List<int>();

string mail\_body;

List<bool>mailyolla = new List<bool>();

List<bool> duzenizin = new List<bool>();

int index;

SqlConnection baglanti = new SqlConnection("Data Source = SEDEN\\SQLEXPRESS; Initial Catalog = PinNoteDB; Integrated Security = True; Encrypt=False;");

public reminder\_olusturma\_10(int? reminderId)

{

baglanti.Open();

InitializeComponent();

//reminderid null değilse çalışacak

if (reminderId != null)

{

reminderID = reminderId.Value;

//not bilgilerini ekrana yazdir

string sql = $"SELECT REMINDER\_NAME, REMINDER\_BODY, REMINDER\_IMPLVL FROM REMINDERS WHERE REMINDER\_ID=@id ";

using (SqlCommand sqlCommand = new SqlCommand(sql, baglanti))

{

sqlCommand.Parameters.Add("@id", SqlDbType.Int).Value = reminderID.Value;

SqlDataReader sqlDataReader\_reminder\_read = sqlCommand.ExecuteReader();

while (sqlDataReader\_reminder\_read.Read())

{

reminder\_title\_box.Text = sqlDataReader\_reminder\_read.GetString(0);

reminder\_box.Text = sqlDataReader\_reminder\_read.GetString(1);

int imp = sqlDataReader\_reminder\_read.GetInt32(2);

if (imp == 1) { comboBox1.SelectedIndex = 0; }

else if (imp == 2) { comboBox1.SelectedIndex = 1; }

else if (imp == 3) { comboBox1.SelectedIndex = 2; }

else if (imp == 4) { comboBox1.SelectedIndex = 3; }

else if (imp == 5) { comboBox1.SelectedIndex = 4; }

}

sqlDataReader\_reminder\_read.Close();

}

}

else

{

reminder\_box.Text = "";

reminder\_title\_box.Text = "";

reminderId = null;

date.Value = DateTime.Now;

saat.SelectedIndex = 9;

comboBox1.SelectedIndex = 1;

}

//arkadaslar listesini yazdır

string tablename = "FRIENDS\_" + user\_data.Id;

using (SqlCommand arkadaslariyazdir = new SqlCommand($"SELECT FriendID FROM {tablename}", baglanti))

{

SqlDataReader reader = arkadaslariyazdir.ExecuteReader();

while (reader.Read())

{

friends.Add(reader.GetInt32(0)); // FriendID değerlerini listeye ekle

}

reader.Close();

}

foreach (int id in friends)

{

using (SqlCommand listboxyazdir = new SqlCommand("SELECT NAME, SURNAME, MAIL FROM USER\_DATA WHERE ID=@id", baglanti))

{

listboxyazdir.Parameters.AddWithValue("@id", id);

SqlDataReader arkadasisim = listboxyazdir.ExecuteReader();

while (arkadasisim.Read())

{

string isim = arkadasisim.GetString(0);

string soyisim = arkadasisim.GetString(1);

mailaddress = arkadasisim.GetString(2);

addfriend.Rows.Add(id, isim, soyisim,false,false,"Add"); // DataGridView'e veri ekler

}

arkadasisim.Close();

}

}

baglanti.Close();

}

//add friend table

private void addfriend\_CellContentDoubleClick(object sender, DataGridViewCellEventArgs e)

{

int id = Convert.ToInt32(addfriend.Rows[e.RowIndex].Cells[0].Value);

int index = 0;

for (int i = 0; i<addedfriendsids.Count; i++)

{

if (addedfriendsids[i] == id)

{

index = i; break;

}

}

if (e.RowIndex >= 0 && (e.ColumnIndex == addfriend.Columns["edit"].Index))

{

duzenleme = Convert.ToBoolean(addfriend.Rows[e.RowIndex].Cells["edit"].Value);

duzenizin[index] = duzenleme;

}

if (e.ColumnIndex == addfriend.Columns["mail"].Index)

{

mail = Convert.ToBoolean(addfriend.Rows[e.RowIndex].Cells["mail"].Value);

mailyolla[index] = duzenleme;

}

}

private void addfriend\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

if (e.RowIndex >= 0 && (e.ColumnIndex == addfriend.Columns["editcheck"].Index || e.ColumnIndex == addfriend.Columns["mail\_bildirim"].Index))

{

addfriend.CommitEdit(DataGridViewDataErrorContexts.Commit);

}

// Tıklanan hücrenin buton olup olmadığını ve "add" isimli buton olup olmadığını kontrol et

if (e.RowIndex >= 0 && e.ColumnIndex == addfriend.Columns["add"].Index)

{

try

{

int arkdsId = Convert.ToInt32(addfriend.Rows[e.RowIndex].Cells[0].Value);

// Listede bu ID yoksa ekleme işlemini yap

if (!addedfriendsids.Contains(arkdsId))

{

string arkdsname = Convert.ToString(addfriend.Rows[e.RowIndex].Cells[1].Value);

string arkdssur = Convert.ToString(addfriend.Rows[e.RowIndex].Cells[2].Value);

bool edt = Convert.ToBoolean(addfriend.Rows[e.RowIndex].Cells[3].Value);

bool m = Convert.ToBoolean(addfriend.Rows[e.RowIndex].Cells[4].Value);

// Arkadaş bilgilerini yeni tabloya ekle

added\_friend.Rows.Add(arkdsId, arkdsname, arkdssur, edt, m, "Add");

// ID'yi listeye ekle

addedfriendsids.Add(arkdsId) ;

duzenizin.Add(edt);

mailyolla.Add(m);

MessageBox.Show("Arkadaş eklendi: " + arkdsname + " " + arkdssur);

}

else

{

MessageBox.Show("Bu arkadaş zaten listeye eklenmiş.");

}

}

catch (Exception ex)

{

MessageBox.Show("Hata oluştu: " + ex.Message);

}

}

}

//added friend table

private void added\_friend\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

int id = Convert.ToInt32(addfriend.Rows[e.RowIndex].Cells[0].Value);

for (int i = 0; i < addedfriendsids.Count; i++)

{

if (addedfriendsids[i] == id)

{

index = i; break;

}

}

// Tıklanan hücrenin buton olup olmadığını ve "del" isimli buton olup olmadığını kontrol et

if (e.RowIndex >= 0 && e.ColumnIndex == added\_friend.Columns["del"].Index)

{

try

{

addedfriendsids.Remove(addedfriendsids[index]);

mailyolla.Remove(mailyolla[index]);

duzenizin.Remove(duzenizin[index]);

}

catch (Exception ex)

{

MessageBox.Show("Hata oluştu: " + ex.Message);

}

}

}

private void added\_friend\_CellContentDoubleClick(object sender, DataGridViewCellEventArgs e)

{

int id = Convert.ToInt32(added\_friend.Rows[e.RowIndex].Cells[0].Value);

for (int i = 0; i < addedfriendsids.Count; i++)

{

if (addedfriendsids[i] == id)

{

index = i; break;

}

}

if (e.RowIndex >= 0 && (e.ColumnIndex == added\_friend.Columns["edit"].Index))

{

duzenleme = Convert.ToBoolean(added\_friend.Rows[e.RowIndex].Cells["edit"].Value);

duzenizin[index] = duzenleme;

}

if (e.ColumnIndex == added\_friend.Columns["mail"].Index)

{

mail = Convert.ToBoolean(added\_friend.Rows[e.RowIndex].Cells["mail"].Value);

mailyolla[index] = duzenleme;

}

}

//yan menu

private void pictureBox2\_Click(object sender, EventArgs e)

{

homepage homepage = new homepage();

homepage.Show();

this.Hide();

}

private void pictureBox4\_Click(object sender, EventArgs e)

{

notes notes = new notes();

notes.Show();

this.Hide();

}

private void pictureBox3\_Click(object sender, EventArgs e)

{

reminder\_liste reminderlist = new reminder\_liste();

reminderlist.Show();

this.Hide();

}

private void pictureBox6\_Click(object sender, EventArgs e)

{

Friends friends = new Friends();

friends.Show();

this.Hide();

}

private void pictureBox7\_Click(object sender, EventArgs e)

{

profile profile = new profile();

profile.Show();

this.Hide();

}

//kaydet ve mail yolla

private void SendEmail(string paylasan, string paylasılanmailadresi, string mailbody)

{

baglanti.Open();

string subjectEmail = paylasan + " share a reminder with you.";

string bodyEmail = mailbody;

using (SmtpClient smtp = new SmtpClient("smtp.outlook.com")) // SMTP sunucusu

{

smtp.Port = 587; // SMTP port numarası

smtp.Credentials = new NetworkCredential("pinnote\_app@hotmail.com", "XqWchh6s!Y"); // E-posta hesabı bilgileri

smtp.EnableSsl = true; // SSL/TLS

// E-posta oluştur

MailMessage mailMessage = new MailMessage("pinnote\_app@hotmail.com", paylasılanmailadresi)

{

Subject = subjectEmail,

Body = bodyEmail

};

try

{

smtp.Send(mailMessage);

MessageBox.Show("Mail sent successfully");

}

catch (Exception ex)

{

MessageBox.Show("Error sending email: " + ex.Message);

}

}

baglanti.Close();

}

private void button1\_Click\_2(object sender, EventArgs e)

{

baglanti.Open();

int onemdrc = Convert.ToInt32(comboBox1.SelectedValue);

if (reminderID != null)

{

//değişiklikleri kaydet

if (!string.IsNullOrWhiteSpace(reminder\_title\_box.Text) && !string.IsNullOrWhiteSpace(reminder\_box.Text))

{

string updateSql = @"UPDATE REMINDERS SET REMINDER\_NAME = @remname, REMINDER\_BODY = @remBody, REMINDER\_DATE=@remdate, REMINDER\_IMPLVL = @ImpLevel WHERE REMINDER\_ID = @remId;";

using (SqlCommand command = new SqlCommand(updateSql, baglanti))

{

// parametreleri ayarla

command.Parameters.AddWithValue("@remId", reminderID);

command.Parameters.AddWithValue("@remname", reminder\_title\_box.Text);

command.Parameters.AddWithValue("@remBody", reminder\_box.Text);

command.Parameters.AddWithValue("@ImpLevel", onemdrc);

command.Parameters.AddWithValue("@remdate", date.Value);

command.ExecuteNonQuery();

}

}

baglanti.Close();

MessageBox.Show("Reminder saved successfully.");

}

else

{

DateTime hatirlatma = date.Value;

if (!string.IsNullOrWhiteSpace(reminder\_title\_box.Text) && !string.IsNullOrWhiteSpace(reminder\_box.Text))

{

// Notları REMINDERS tablosuna ekle

string insertnote = "INSERT INTO REMINDERS (REMINDER\_NAME, REMINDER\_BODY, REMINDER\_DATE, REMINDER\_IMPLVL, REMINDER\_CREATER) VALUES (@Notetitle, @Notebody, @Reminderdate, @Imp\_level, @Createrid)";

using (SqlCommand command = new SqlCommand(insertnote, baglanti))

{

command.Parameters.AddWithValue("@Notetitle", reminder\_title\_box.Text);

command.Parameters.AddWithValue("@Notebody", reminder\_box.Text);

command.Parameters.AddWithValue("@Reminderdate", hatirlatma);

command.Parameters.AddWithValue("@Imp\_level", onemdrc);

command.Parameters.AddWithValue("@Createrid", user\_data.Id);

command.ExecuteNonQuery();

}

// En son eklenen Reminder'ın ID'sini al

string getReminderIdQuery = "SELECT IDENT\_CURRENT('REMINDERS')"; // Varsayılan olarak son eklenen ID'yi döner

int reminderID;

using (SqlCommand sqlsorgu = new SqlCommand(getReminderIdQuery, baglanti))

{

reminderID = Convert.ToInt32(sqlsorgu.ExecuteScalar()); // ID değerini döndür

}

// REMINDER\_ID'yi kullanıcının REMINDER tablosuna ekle

string note\_table\_name = "REMINDER\_" + user\_data.Id;

string insert\_note\_id = $"INSERT INTO {note\_table\_name} (REMINDER\_ID, REMINDER\_EDIT) VALUES (@Noteid, @Edit)";

using (SqlCommand command1 = new SqlCommand(insert\_note\_id, baglanti))

{

command1.Parameters.AddWithValue("@Noteid", reminderID);

command1.Parameters.AddWithValue("@Edit", 1);

command1.ExecuteNonQuery();

}

MessageBox.Show("Reminder saved successfully.");

//paylasilan arkadaslara reminder ekleme

try

{

int a = 0;

foreach (int id in addedfriendsids)

{

string note\_table\_name1 = $"REMINDER\_{id}";

string insert\_rem\_id = $"INSERT INTO {note\_table\_name1} (REMINDER\_ID, REMINDER\_EDIT) VALUES (@remid, @Edit)";

using (SqlCommand command1 = new SqlCommand(insert\_rem\_id, baglanti))

{

command1.Parameters.AddWithValue("@remid", reminderID);

command1.Parameters.AddWithValue("@Edit", duzenizin[a]);

command1.ExecuteNonQuery();

}

if (mailyolla[a])

{

string paylasan = user\_data.Name + " " + user\_data.Surname;

mail\_body = paylasan + " shared a reminder with you.\n" + reminder\_title\_box.Text + "\n" + reminder\_box.Text + "\n" + date.Value;

SendEmail(paylasan, mailaddress, mail\_body);

}

a++;

}

homepage homepage = new homepage();

homepage.Show();

this.Hide();

}

catch (Exception ex)

{

Console.WriteLine("An error occurred: " + ex.Message);

}

}

else

{

MessageBox.Show("Title or body can not be empty");

}

}

homepage homepage1 = new homepage();

homepage1.Show();

this.Hide();

}

}

}

**Friends**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using Microsoft.Graph.Models;

using Microsoft.VisualBasic;

using pinnotedeskup.wgt;

using static System.Net.Mime.MediaTypeNames;

using static pinnotedeskup.Program;

namespace pinnotedeskup

{

public partial class Friends : Form

{

SqlConnection baglanti = new SqlConnection("Data Source = SEDEN\\SQLEXPRESS; Initial Catalog = PinNoteDB; Integrated Security = True; Encrypt=False;");

public Friends()

{

InitializeComponent();

baglanti.Open();

//arkdas id çekme

string table\_name = "FRIENDS\_" + user\_data.Id;

List<int> friendsids = new List<int>();

using (SqlCommand friendslist = new SqlCommand($"SELECT FriendID FROM {table\_name}",baglanti))

{

try

{

using(SqlDataReader friends\_list =friendslist.ExecuteReader())

{

while (friends\_list.Read())

{

int friendID = friends\_list.GetInt32(0);

friendsids.Add(friendID);

}

friends\_list.Close();

}

}

catch (Exception ex)

{

MessageBox.Show("Error: " + ex.Message);

}

}

//arkadas bilgilerini çekme

foreach (int friendID in friendsids)

{

using (SqlCommand friendsinfo = new SqlCommand("SELECT NAME, SURNAME, PHOTO FROM USER\_DATA WHERE ID=@ID", baglanti))

{

try

{

friendsinfo.Parameters.AddWithValue("@ID", friendID);

using (SqlDataReader sqlDataReader\_friendsinfo = friendsinfo.ExecuteReader())

{

while (sqlDataReader\_friendsinfo.Read())

{

string name = (sqlDataReader\_friendsinfo.GetString(0));

string surname = (sqlDataReader\_friendsinfo.GetString(1));

if (!sqlDataReader\_friendsinfo.IsDBNull(2))

{

byte[] imageData = (byte[])sqlDataReader\_friendsinfo["PHOTO"];

using (MemoryStream ms = new MemoryStream(imageData))

{

System.Drawing.Image profilephoto = System.Drawing.Image.FromStream(ms);

dataGridView1.Rows.Add(profilephoto, name + " " + surname, friendID);

}

}

else

{

dataGridView1.Rows.Add(null, name + " " + surname, friendID);

}

}

sqlDataReader\_friendsinfo.Close();

}

}

catch (Exception ex)

{

MessageBox.Show("Error: " + ex.Message);

}

}

}

}

private void dataGridView1\_CellContentClick\_1(object sender, DataGridViewCellEventArgs e)

{

if (dataGridView1.Columns[e.ColumnIndex] is DataGridViewButtonColumn && e.RowIndex >= 0)

{

int idColumnIndex = 2;

int id = Convert.ToInt32(dataGridView1.Rows[e.RowIndex].Cells[idColumnIndex].Value);

friendprofile friendprofile = new friendprofile(id);

friendprofile.Show();

this.Hide();

}

}

private void button1\_Click(object sender, EventArgs e)

{

//add friends

string add\_id = Interaction.InputBox("Add friends id:", "Add Friend!");

if (!string.IsNullOrEmpty(add\_id))

{

// ID'nin USER\_DATA tablosunda mevcut mu

string idvarmi = "SELECT COUNT(1) FROM USER\_DATA WHERE ID = @add\_id";

using (SqlCommand checkid = new SqlCommand(idvarmi, baglanti))

{

checkid.Parameters.AddWithValue("@add\_id", int.Parse(add\_id));

// Kullanıcı ID'sinin var olup olmadığını kontrol edin

int userExists = (int)checkid.ExecuteScalar();

if (userExists > 0)

{

string tablename = "FRIENDS\_" + user\_data.Id;

string eklimi = $"SELECT COUNT(1) FROM {tablename} WHERE FriendID = @add\_id";

using (SqlCommand checkfriend = new SqlCommand(eklimi, baglanti))

{

checkfriend.Parameters.AddWithValue("@add\_id", int.Parse(add\_id));

int ekli = (int)checkfriend.ExecuteScalar();

if(ekli <= 0)

{

// Kullanıcı ID'si mevcutsa arkadaşlık isteğini ekleyin

string table\_name = "REQUESTS\_" + add\_id;

string insertid = "INSERT INTO " + table\_name + " (RequesterID, RequestDate) VALUES (@myid, @date)";

using (SqlCommand addfriend = new SqlCommand(insertid, baglanti))

{

// Parametreler oluşturulur ve değerleri atanır

addfriend.Parameters.AddWithValue("@myid", user\_data.Id);

addfriend.Parameters.AddWithValue("@date", DateTime.Now);

try

{

addfriend.ExecuteNonQuery();

MessageBox.Show("Request sent!");

}

catch (SqlException ex)

{

MessageBox.Show("An error occurred while sending the request: " + ex.Message);

}

}

}

else

{

MessageBox.Show("This person is already on your list.");

}

}

}

else

{

// Kullanıcı ID'si mevcut değilse hata mesajı göster

MessageBox.Show("The specified user ID does not exist.");

}

}

}

}

private void pictureBox2\_Click\_1(object sender, EventArgs e)

{

homepage homepage = new homepage();

homepage.Show();

this.Hide();

}

private void button2\_Click(object sender, EventArgs e)

{

IsteklerEkran isteklerEkran = new IsteklerEkran();

isteklerEkran.Show();

}

private void pictureBox4\_Click\_1(object sender, EventArgs e)

{

notes notes = new notes(); notes.Show(); this.Hide();

}

private void pictureBox3\_Click(object sender, EventArgs e)

{

reminder\_liste reminder\_Liste = new reminder\_liste();

reminder\_Liste.Show();

this.Hide();

}

private void pictureBox6\_Click(object sender, EventArgs e)

{

Friends friends = new Friends();

friends.Show();

this.Hide();

}

private void pictureBox7\_Click(object sender, EventArgs e)

{

profile profile = new profile();

profile.Show();

this.Hide();

}

}

}

**Friend Profile**

using Microsoft.Graph.Models;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static pinnotedeskup.Program;

namespace pinnotedeskup

{

public partial class friendprofile : Form

{

SqlConnection baglanti = new SqlConnection("Data Source = SEDEN\\SQLEXPRESS; Initial Catalog = PinNoteDB; Integrated Security = True; Encrypt=False;");

public friendprofile(int friendID)

{

InitializeComponent();

baglanti.Open();

//profil foto

using (SqlCommand friendsinfo = new SqlCommand("SELECT NAME, SURNAME, MAIL, PHOTO FROM USER\_DATA WHERE ID=@ID", baglanti))

{

try

{

friendsinfo.Parameters.AddWithValue("@ID", friendID);

using (SqlDataReader sqlDataReader\_friendsinfo = friendsinfo.ExecuteReader())

{

while (sqlDataReader\_friendsinfo.Read())

{

textBox1.Text=friendID.ToString();

textBox2.Text = (sqlDataReader\_friendsinfo.GetString(0));

textBox3.Text = (sqlDataReader\_friendsinfo.GetString(1));

textBox4.Text = sqlDataReader\_friendsinfo.GetString(2);

if (!sqlDataReader\_friendsinfo.IsDBNull(3))

{

byte[] imageData = (byte[])sqlDataReader\_friendsinfo["PHOTO"];

using (MemoryStream ms = new MemoryStream(imageData))

{

pictureBox1.Image = System.Drawing.Image.FromStream(ms);

}

}

}

sqlDataReader\_friendsinfo.Close();

}

}

catch (Exception ex)

{

MessageBox.Show("Error: " + ex.Message);

}

}

baglanti.Close();

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

//ortak notlar

/\*

string mytable\_name = "NOTES\_" + user\_data.Id;

string friendtable\_name = "NOTES\_" + friendID;

List<int> noteid1 = new List<int>();

List<int> noteid2 = new List<int>();

List<int> ortaknotes = new List<int>();

using (SqlCommand noteid\_ctrl = new SqlCommand($"SELECT NOTE\_ID FROM {mytable\_name}", baglanti))

{

using (SqlDataReader sqlDataReader\_noteid = noteid\_ctrl.ExecuteReader())

{

while (sqlDataReader\_noteid.Read())

{

noteid1.Add(sqlDataReader\_noteid.GetInt32(0));

}

sqlDataReader\_noteid.Close();

}

}

using (SqlCommand noteid\_ctrl = new SqlCommand($"SELECT NOTE\_ID FROM {friendtable\_name}", baglanti))

{

using (SqlDataReader sqlDataReader\_noteid = noteid\_ctrl.ExecuteReader())

{

while (sqlDataReader\_noteid.Read())

{

noteid2.Add(sqlDataReader\_noteid.GetInt32(0));

}

sqlDataReader\_noteid.Close();

}

}

foreach (int noteid in noteid1)

{

if (noteid2.Contains(noteid))

{

ortaknotes.Add(noteid);

}

}

//note tablosundan verileri çekme

foreach (int note\_id in ortaknotes)

{

SqlCommand table\_read = new SqlCommand("SELECT NOTE\_TITLE, NOTE\_BODY, UPDATE\_TIME FROM NOTES WHERE NOTEID = @id", baglanti);

table\_read.Parameters.AddWithValue("@id", note\_id);

SqlDataReader sqlDataReader\_tableread = table\_read.ExecuteReader();

while (sqlDataReader\_tableread.Read())

{

string title = sqlDataReader\_tableread.GetString(0);

string body = sqlDataReader\_tableread.GetString(1);

DateTime update = sqlDataReader\_tableread.GetDateTime(2);

this.ortaknotes.Rows.Add(note\_id, title, body, update);

MessageBox.Show(Convert.ToString(note\_id));

}

sqlDataReader\_tableread.Close();

}

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

//ortak hatırlatıcılar

/\*

string mytable\_namerem = "REMINDER\_" + user\_data.Id;

string friendtable\_namerem = "REMINDER\_" + friendID;

List<int> remid1 = new List<int>();

List<int> remid2 = new List<int>();

List<int> ortakrems = new List<int>();

using (SqlCommand noteid\_ctrlrem = new SqlCommand($"SELECT REMINDER\_ID FROM {mytable\_namerem}", baglanti))

{

using (SqlDataReader sqlDataReader\_noteidrem = noteid\_ctrlrem.ExecuteReader())

{

while (sqlDataReader\_noteidrem.Read())

{

remid1.Add(sqlDataReader\_noteidrem.GetInt32(0));

}

sqlDataReader\_noteidrem.Close();

}

}

using (SqlCommand noteid\_ctrlrem = new SqlCommand($"SELECT REMINDER\_ID FROM {friendtable\_namerem}", baglanti))

{

using (SqlDataReader sqlDataReader\_noteidrem = noteid\_ctrlrem.ExecuteReader())

{

while (sqlDataReader\_noteidrem.Read())

{

remid2.Add(sqlDataReader\_noteidrem.GetInt32(0));

}

sqlDataReader\_noteidrem.Close();

}

}

foreach (int id in remid1)

{

if (remid2.Contains(id))

{

ortakrems.Add(id);

}

}

//note tablosundan verileri çekme

foreach (int remid in ortakrems)

{

SqlCommand table\_readrem = new SqlCommand("SELECT NOTE\_TITLE, NOTE\_BODY, UPDATE\_TIME FROM NOTES WHERE NOTEID = @id", baglanti);

table\_readrem.Parameters.AddWithValue("@id", remid);

SqlDataReader sqlDataReader\_tablereadrem = table\_readrem.ExecuteReader();

while (sqlDataReader\_tablereadrem.Read())

{

string title = sqlDataReader\_tablereadrem.GetString(0);

string body = sqlDataReader\_tablereadrem.GetString(1);

DateTime update = sqlDataReader\_tablereadrem.GetDateTime(2);

ortakreminders.Rows.Add(remid, title, body, update);

}

sqlDataReader\_tablereadrem.Close();

}

\*/

}

private void pictureBox2\_Click(object sender, EventArgs e)

{

homepage homepage = new homepage();

homepage.Show();

this.Hide();

}

private void pictureBox4\_Click(object sender, EventArgs e)

{

notes notes = new notes();

notes.Show();

this.Hide();

}

private void pictureBox3\_Click(object sender, EventArgs e)

{

reminder\_liste reminderlist = new reminder\_liste();

reminderlist.Show();

this.Hide();

}

private void pictureBox6\_Click(object sender, EventArgs e)

{

Friends friends = new Friends();

friends.Show();

this.Hide();

}

private void pictureBox7\_Click(object sender, EventArgs e)

{

profile profile = new profile();

profile.Show();

this.Hide();

}

private void button1\_Click(object sender, EventArgs e)

{

try

{

string tableDeleteName = "FRIENDS\_" + user\_data.Id;

string sql = $"DELETE FROM {tableDeleteName} WHERE FriendID = @id";

baglanti.Open();

using (SqlCommand delFriend = new SqlCommand(sql, baglanti))

{

// friend\_id değişkeninin integer olduğunu varsayarak

int id = Convert.ToInt32(friend\_id.Text);

// Parametreyi AddWithValue kullanarak ekleyin

delFriend.Parameters.AddWithValue("@id", id);

int rowsAffected = delFriend.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Deleted!");

}

else

{

MessageBox.Show("No record found with the specified ID.");

}

}

}

catch (Exception ex)

{

MessageBox.Show("Bir hata oluştu: " + ex.Message);

}

}

}

}

**Profile**

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.SqlClient;

using static pinnotedeskup.Program;

using System.IO;

namespace pinnotedeskup

{

public partial class profile : Form

{

/\*sql baglanti\*/

SqlConnection baglanti = new SqlConnection("Data Source = SEDEN\\SQLEXPRESS; Initial Catalog = PinNoteDB; Integrated Security = True; Encrypt=False;");

public profile()

{

InitializeComponent();

user\_id\_box.Text = Convert.ToString(user\_data.Id);

try

{

// SQL komutu

string sql = "SELECT NAME, SURNAME, PHONE, MAIL, BIRTH\_DATE, PHOTO FROM USER\_DATA WHERE ID = @id";

using (SqlCommand vericekme = new SqlCommand(sql, baglanti))

{

// Parametre ekleme

vericekme.Parameters.AddWithValue("@id", user\_data.Id);

// Bağlantıyı aç

baglanti.Open();

using (SqlDataReader reader = vericekme.ExecuteReader())

{

if (reader.Read())

{

usr\_name\_box.Text = reader["NAME"].ToString();

user\_surname\_box.Text = reader["SURNAME"].ToString();

user\_phonenum\_box.Text = reader["PHONE"].ToString();

user\_mail\_box.Text = reader["MAIL"].ToString();

// Fotoğraf okuma, varsa

if (!reader.IsDBNull(reader.GetOrdinal("PHOTO"))) // Burada doğru sütun adı kullanılıyor

{

byte[] imageData = (byte[])reader["PHOTO"];

using (MemoryStream ms = new MemoryStream(imageData))

{

System.Drawing.Image profilephoto = System.Drawing.Image.FromStream(ms);

profile\_pic\_box.Image = profilephoto;

}

}

else

{

profile\_pic\_box.Image = null; // veya varsayılan bir resim atayabilirsiniz

}

}

else

{

MessageBox.Show("No user found with the provided ID.");

}

}

}

}

catch (Exception ex)

{

MessageBox.Show("An error occurred: " + ex.Message);

}

finally

{

baglanti.Close(); // Bağlantıyı her durumda kapat

}

}

private void pp\_change\_button\_Click(object sender, EventArgs e)

{

OpenFileDialog openFileDialog = new OpenFileDialog();

openFileDialog.Filter = "Image files (\*.jpg, \*.jpeg, \*.png) | \*.jpg; \*.jpeg; \*.png";

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

{

profile\_pic\_box.ImageLocation = openFileDialog.FileName;

resim\_textbox.Text = openFileDialog.FileName;

SaveImageToDatabase(openFileDialog.FileName);

}

}

private byte[] ImageToByteArray(string imagePath)

{

using (var imageStream = new FileStream(imagePath, FileMode.Open, FileAccess.Read))

{

var binaryReader = new BinaryReader(imageStream);

return binaryReader.ReadBytes((int)imageStream.Length);

}

}

private void SaveImageToDatabase(string imagePath)

{

byte[] imageBytes = ImageToByteArray(imagePath);

string sql = "UPDATE USER\_DATA SET PHOTO = @Photo WHERE ID = @Id";

using (SqlCommand command = new SqlCommand(sql, baglanti))

{

command.Parameters.AddWithValue("@Photo", imageBytes);

command.Parameters.AddWithValue("@Id", user\_data.Id);

baglanti.Open();

int result = command.ExecuteNonQuery();

if (result > 0)

{

MessageBox.Show("Profile photo updated successfully.");

}

else

{

MessageBox.Show("No record was updated.");

}

}

}

private void pictureBox2\_Click(object sender, EventArgs e)

{

homepage homepage = new homepage();

homepage.Show();

this.Hide();

}

private void pictureBox4\_Click(object sender, EventArgs e)

{

notes notes = new notes();

notes.Show();

this.Hide();

}

private void pictureBox3\_Click(object sender, EventArgs e)

{

reminder\_liste reminder\_Liste = new reminder\_liste();

reminder\_Liste.Show();

this.Hide();

}

private void pictureBox6\_Click(object sender, EventArgs e)

{

Friends friends = new Friends();

friends.Show();

this.Hide();

}

private void pictureBox7\_Click(object sender, EventArgs e)

{

profile profile = new profile();

profile.Show();

this.Hide();

}

}

}

**Requests**

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Windows.Forms;

using static pinnotedeskup.Program;

namespace pinnotedeskup.wgt

{

public partial class IsteklerEkran : Form

{

private DateTime lastCheck = DateTime.Now;

SqlConnection baglanti = new SqlConnection("Data Source = SEDEN\\SQLEXPRESS; Initial Catalog = PinNoteDB; Integrated Security = True; Encrypt=False;");

public IsteklerEkran()

{

InitializeComponent();

LoadRequests();

}

private void LoadRequests()

{

string tableName = "REQUESTS\_" + user\_data.Id;

List<int> requestIds = new List<int>();

baglanti.Open();

// İstek ID'lerini çekme

using (SqlCommand requestList = new SqlCommand($"SELECT RequesterID FROM {tableName}", baglanti))

{

using (SqlDataReader reader = requestList.ExecuteReader())

{

while (reader.Read())

{

requestIds.Add(reader.GetInt32(0));

}

}

}

// Kişi bilgilerini çekme

foreach (int reqID in requestIds)

{

using (SqlCommand friendsInfo = new SqlCommand("SELECT NAME, SURNAME FROM USER\_DATA WHERE ID = @ID", baglanti))

{

friendsInfo.Parameters.AddWithValue("@ID", reqID);

using (SqlDataReader reader = friendsInfo.ExecuteReader())

{

while (reader.Read())

{

string name = reader.GetString(0);

string surname = reader.GetString(1);

istek.Rows.Add(reqID, name, surname, "Accept", "Decline");

}

}

}

}

baglanti.Close();

}

private void button1\_Click(object sender, EventArgs e)

{

this.Close();

}

private void istek\_CellContentClick\_1(object sender, DataGridViewCellEventArgs e)

{

if (e.RowIndex >= 0 && istek.Columns[e.ColumnIndex] is DataGridViewButtonColumn)

{

int friendId = Convert.ToInt32(istek.Rows[e.RowIndex].Cells[0].Value);

if (istek.Columns[e.ColumnIndex].Name == "accept")

{

AcceptRequest(friendId);

}

else if (istek.Columns[e.ColumnIndex].Name == "decline")

{

RemoveRequest(friendId);

}

}

}

private void AcceptRequest(int friendId)

{

string tableName = "FRIENDS\_" + user\_data.Id;

baglanti.Open();

using (SqlCommand command = new SqlCommand($"INSERT INTO {tableName} (FriendID, AddedDate) VALUES (@friendId, @date)", baglanti))

{

command.Parameters.AddWithValue("@friendId", friendId);

command.Parameters.AddWithValue("@date", DateTime.Now);

command.ExecuteNonQuery();

}

RemoveRequest(friendId);

baglanti.Close();

}

private void RemoveRequest(int friendId)

{

string tableName = "REQUESTS\_" + user\_data.Id;

baglanti.Open();

using (SqlCommand command = new SqlCommand($"DELETE FROM {tableName} WHERE RequesterID = @friendId", baglanti))

{

command.Parameters.AddWithValue("@friendId", friendId);

command.ExecuteNonQuery();

}

// DataGridView'dan satırı kaldırma

foreach (DataGridViewRow row in istek.Rows)

{

if (Convert.ToInt32(row.Cells[0].Value) == friendId)

{

istek.Rows.Remove(row);

break;

}

}

baglanti.Close ();

}

}

}

**Program.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace pinnotedeskup

{

internal static class Program

{

/// <summary>

/// Uygulamanın ana girdi noktası.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new giris());

}

public static class user\_data

{

public static int Id { get; set; }

public static string Name { get; set; }

public static string Surname { get; set; }

public static string Email { get; set; }

public static string Password { get; set; }

public static string PhoneNumber { get; set; }

public static DateTime BirthDate { get; set; }

}}}