Самченко ІТІНФ-20-1

База хвороб: назва, симптоми, процедури, перелік рекомендованих ліків із зазначенням необхідної кількості.

База медикаментів: назва, форми випуску, дозування, взаємозамінність.

Формування рецепту після огляду хворого. Висновок порівняльних діаграм

**Database.cs**

using System;

using System.Collections.Generic;

using Microsoft.Data.SqlClient;

using System.Text.RegularExpressions;

namespace DatabaseCore

{

public class Database

{

private string connetionString; //Строка, содержащая данные для подключения к БД

private SqlConnection con; //Переменная подключения к БД

private SqlCommand sc = new SqlCommand(); //Переменная, содержащая команды (query) БД

public Database(string connetionString) //Конструктор БД

{

this.connetionString = connetionString;

con = new SqlConnection(connetionString);

sc.Connection = con;

}

public void InsertNewPatient(string name\_patient, string phone) //Добавить в БД нового пациента

{

try

{

con.Open();

sc.CommandText = "SELECT COUNT(\*) " +

"FROM Patient " +

"WHERE Name\_Patient='" + name\_patient + "' " +

"AND Phone='" + phone + "'";

int patientCount = (int)sc.ExecuteScalar(); //Проверка, есть ли уже человек с такими ФИО и телефоном в таблице

if (patientCount == 0) //Если такого человека нет, добавляем его в таблицу

{

sc.CommandText = "INSERT INTO Patient " +

"VALUES (" + "'" + name\_patient + "'" + ", " + "'" + phone + "'" + ")";

sc.ExecuteNonQuery();

}

con.Close();

}

catch (Exception)

{

throw new Exception("Не могу подключиться к базе данных!");

}

}

private int FindPossibleDisease(Dictionary<int, string> diseases, string symptoms, string complaint) //Выбор самой вероятной болезни по имеющимся симптомам

{

List<string> keyWords = new List<string>(Regex.Split(symptoms, @"[/\s;]+"));

keyWords.RemoveAll(keyWord => keyWord.Length <= 2);

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

keyWords[i] = keyWords[i].Substring(0, 3);

int max = 0;

int maxId = 0;

foreach (int id in diseases.Keys)

{

int evaluation = 0;

foreach (string keyWord in keyWords)

if (complaint.ToLower().Contains(keyWord) && diseases[id].Contains(keyWord))

evaluation += 1;

if (evaluation > max)

{

max = evaluation;

maxId = id;

}

}

return maxId;

}

public int Diagnose(string complaint) //Определение болезни

{

try

{

con.Open();

Dictionary<int, string> diseases = new Dictionary<int, string>();

string symptoms = "";

sc.CommandText = "SELECT ID\_Dis, Symptom FROM Disease";

SqlDataReader sqlreader = sc.ExecuteReader();

while (sqlreader.Read())

{

diseases[sqlreader.GetInt32(sqlreader.GetOrdinal("ID\_Dis"))] = sqlreader.GetString(sqlreader.GetOrdinal("Symptom"));

symptoms += sqlreader.GetString(sqlreader.GetOrdinal("Symptom")) + ";";

}

sqlreader.Close();

int id\_dis = FindPossibleDisease(diseases, symptoms, complaint);

con.Close();

return id\_dis;

}

catch (Exception)

{

throw new Exception("Не могу подключиться к базе данных!");

}

}

public int InsertNewInspection(int id\_dis, string complaint, string name\_patient, string phone) //Добавить в БД новый осмотр

{

DateTime datetime = DateTime.Now;

try

{

con.Open();

sc.CommandText = "INSERT INTO Inspection (ID\_Patient, ID\_Dis, Date, Complain) " +

"SELECT ID\_Patient, " + id\_dis + " AS ID\_Dis, '" + datetime.ToString("yyyy-MM-dd HH:mm:ss.fff") + "' AS Date, '" + complaint + "' AS Complain " +

"FROM Patient " +

"WHERE Name\_Patient='" + name\_patient + "' " +

"AND Phone='" + phone + "' " +

"SELECT @@IDENTITY AS 'Identity'";

int id\_ins = Convert.ToInt32(sc.ExecuteScalar());

con.Close();

return id\_ins;

}

catch (Exception)

{

throw new Exception("Не могу подключиться к базе данных!");

}

}

public List<Inspection> GetAllInspections() //Получить список всех осмотров

{

List<Inspection> inspections = new List<Inspection>();

try

{

con.Open();

sc.CommandText = "SELECT Inspection.ID\_Ins, Patient.Name\_Patient, Disease.Name\_Dis, FORMAT(Inspection.Date, N'dd.MM.yyyy') AS Date, Inspection.Complain " +

"FROM Inspection " +

"JOIN Patient ON(Inspection.ID\_Patient = Patient.ID\_Patient) " +

"JOIN Disease On(Inspection.ID\_Dis = Disease.ID\_Dis)";

SqlDataReader sqlreader = sc.ExecuteReader();

while (sqlreader.Read())

{

inspections.Add(new Inspection(sqlreader.GetInt32(sqlreader.GetOrdinal("ID\_Ins")),

sqlreader.GetString(sqlreader.GetOrdinal("Name\_Patient")),

sqlreader.GetString(sqlreader.GetOrdinal("Name\_Dis")),

sqlreader.GetString(sqlreader.GetOrdinal("Date")).ToString(),

sqlreader.GetString(sqlreader.GetOrdinal("Complain"))));

}

sqlreader.Close();

con.Close();

return inspections;

}

catch (Exception ex)

{

throw new Exception(ex.Message);

}

}

public string GetDisease(int id\_ins) //Получить болезнь из определённого осмотра

{

try

{

con.Open();

sc.CommandText = "SELECT Disease.Name\_Dis " +

"FROM Disease " +

"JOIN Inspection ON (Inspection.ID\_Dis = Disease.ID\_Dis) " +

"WHERE Inspection.ID\_Ins = " + id\_ins;

SqlDataReader sqlreader = sc.ExecuteReader();

sqlreader.Read();

string disease = sqlreader.GetString(sqlreader.GetOrdinal("Name\_Dis"));

sqlreader.Close();

con.Close();

return disease;

}

catch (Exception)

{

throw new Exception("Не могу подключиться к базе данных!");

}

}

public Dictionary<string, int> GetCountedDiseases() //Получить количество заражений каждой болезнью

{

Dictionary<string, int> countedDiseases = new Dictionary<string, int>();

try

{

con.Open();

sc.CommandText = "SELECT Disease.Name\_Dis, COUNT(Disease.Name\_Dis) AS Count " +

"FROM Inspection " +

"JOIN Disease ON(Inspection.ID\_Dis = Disease.ID\_Dis) " +

"GROUP BY Disease.Name\_Dis " +

"ORDER BY Count DESC";

SqlDataReader sqlreader = sc.ExecuteReader();

while (sqlreader.Read())

{

countedDiseases[sqlreader.GetString(sqlreader.GetOrdinal("Name\_Dis"))] = sqlreader.GetInt32(sqlreader.GetOrdinal("Count"));

}

sqlreader.Close();

con.Close();

return countedDiseases;

}

catch (Exception)

{

throw new Exception("Не могу подключиться к базе данных!");

}

}

public Dictionary<string, int> GetCountedPatients() //Получить количество осмотров каждого пациента

{

Dictionary<string, int> countedPatients = new Dictionary<string, int>();

try

{

con.Open();

sc.CommandText = "SELECT Patient.Name\_Patient, COUNT(Patient.Name\_Patient) AS Count " +

"FROM Inspection " +

"JOIN Patient ON(Inspection.ID\_Patient = Patient.ID\_Patient) " +

"GROUP BY Patient.Name\_Patient " +

"ORDER BY Count DESC";

SqlDataReader sqlreader = sc.ExecuteReader();

while (sqlreader.Read())

{

countedPatients[sqlreader.GetString(sqlreader.GetOrdinal("Name\_Patient"))] = sqlreader.GetInt32(sqlreader.GetOrdinal("Count"));

}

sqlreader.Close();

con.Close();

return countedPatients;

}

catch (Exception)

{

throw new Exception("Не могу подключиться к базе данных!");

}

}

public Dictionary<string, int> GetCountedMedication() //Получить количество попаданий каждого препарата в рецепт

{

Dictionary<string, int> countedMedication = new Dictionary<string, int>();

try

{

con.Open();

sc.CommandText = "SELECT Medication.Name\_Med, COUNT(Medication.Name\_Med) AS Count " +

"FROM Recept " +

"JOIN Medication ON(Recept.ID\_Med = Medication.ID\_Med) " +

"GROUP BY Medication.Name\_Med " +

"ORDER BY Count DESC";

SqlDataReader sqlreader = sc.ExecuteReader();

while (sqlreader.Read())

{

countedMedication[sqlreader.GetString(sqlreader.GetOrdinal("Name\_Med"))] = sqlreader.GetInt32(sqlreader.GetOrdinal("Count"));

}

sqlreader.Close();

con.Close();

return countedMedication;

}

catch (Exception)

{

throw new Exception("Не могу подключиться к базе данных!");

}

}

public void InsertNewRecept(int id\_ins) //Добавить в БД новый рецепт

{

Random random = new Random();

try

{

con.Open();

sc.CommandText = "INSERT INTO Recept " +

"SELECT Inspection.ID\_Ins, " +

"Treatment.ID\_Med, " +

"IIF(Disease.Complexity=1, ABS(CHECKSUM(NewId())) % 4 + 1, IIF(Disease.Complexity=2, ABS(CHECKSUM(NewId())) % 2 + 3, 4)) AS Count\_receptions, " +

"IIF(Disease.Complexity=1, ABS(CHECKSUM(NewId())) % 15 + 7, ABS(CHECKSUM(NewId())) % 8 + 14) AS Count\_days, " +

"Disease.Complexity " +

"FROM Disease " +

"JOIN Inspection ON (Inspection.ID\_Dis = Disease.ID\_Dis) " +

"JOIN Treatment ON (Disease.ID\_Dis = Treatment.ID\_Dis) " +

"WHERE Inspection.ID\_Ins = " + id\_ins;

sc.ExecuteNonQuery();

con.Close();

}

catch (Exception)

{

throw new Exception("Не могу подключиться к базе данных!");

}

}

public List<Recept> GetRecept(int id\_ins) //Получить рецепт

{

List<Recept> recept = new List<Recept>();

try

{

con.Open();

sc.CommandText = "SELECT Medication.Name\_Med, Recept.Dosage, Recept.Count\_receptions, Recept.Count\_days " +

"FROM Recept " +

"JOIN Medication ON (Recept.ID\_Med = Medication.ID\_Med) " +

"WHERE Recept.ID\_Ins = " + id\_ins;

SqlDataReader sqlreader = sc.ExecuteReader();

while (sqlreader.Read())

{

recept.Add(new Recept(

sqlreader.GetString(sqlreader.GetOrdinal("Name\_Med")),

sqlreader.GetInt32(sqlreader.GetOrdinal("Dosage")),

sqlreader.GetInt32(sqlreader.GetOrdinal("Count\_receptions")),

sqlreader.GetInt32(sqlreader.GetOrdinal("Count\_days"))

));

}

sqlreader.Close();

con.Close();

return recept;

}

catch (Exception)

{

throw new Exception("Не могу подключиться к базе данных!");

}

}

}

public class Inspection //Класс осмотра

{

public int ID\_Ins { get; private set; }

public string Name\_Patient { get; private set; }

public string Name\_Dis { get; private set; }

public string Date { get; private set; }

public string Complain { get; private set; }

public Inspection(int id\_ins, string name\_patient, string name\_dis, string date, string complain)

{

ID\_Ins = id\_ins;

Name\_Patient = name\_patient;

Name\_Dis = name\_dis;

Date = date;

Complain = complain;

}

}

public class Recept //Класс рецепта

{

public string Name\_Med { get; private set; }

public int Dosage { get; private set; }

public int Count\_receptions { get; private set; }

public int Count\_days { get; private set; }

public Recept(string name\_Med, int dosage, int count\_receptions, int count\_days)

{

Name\_Med = name\_Med;

Dosage = dosage;

Count\_receptions = count\_receptions;

Count\_days = count\_days;

}

}

}

**ResizeControl.cs**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace ResizeControl

{

public class FormResizeControl

{

//Изменение размера элемента

public static void ResizeControl(Control control, Rectangle originalControlRect, float originalFontSize, float widthRatio, float heightRatio)

{

float newWidth = originalControlRect.Location.X \* widthRatio;

float newHeight = originalControlRect.Location.Y \* heightRatio;

control.Location = new Point((int)newWidth, (int)newHeight);

control.Width = (int)(originalControlRect.Width \* widthRatio);

control.Height = (int)(originalControlRect.Height \* heightRatio);

float fontRatio = Math.Min(widthRatio, heightRatio);

float newFontSize = originalFontSize \* fontRatio;

try

{

Font newFont = new Font(control.Font.FontFamily, newFontSize);

control.Font = newFont;

}

catch (ArgumentException) { }

}

}

}

**Program.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Kursach

{

static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

Application.SetHighDpiMode(HighDpiMode.SystemAware);

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new AppointmentForm());

}

}

}

**Apointment.cs**

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Windows.Forms;

using System.Text.RegularExpressions;

using ResizeControl;

using DatabaseCore;

namespace Kursach

{

public partial class AppointmentForm : Form

{

private Rectangle originalFormRect;

private Rectangle originalFullNameLabelRect;

private Rectangle originalFullNameTextBoxRect;

private Rectangle originalPhoneNumberLabelRect;

private Rectangle originalPhoneNumberTextBoxRect;

private Rectangle originalComplaintLabelRect;

private Rectangle originalComplaintTextBoxRect;

private Rectangle originalMakeAnAppointmentButtonRect;

private Rectangle originalAppoitmentLabelRect;

private Rectangle originalEmptyTextFieldLabel1Rect;

private Rectangle originalEmptyTextFieldLabel2Rect;

private Rectangle originalEmptyTextFieldLabel3Rect;

private float originalFullNameLabelFontSize;

private float originalFullNameTextBoxFontSize;

private float originalPhoneNumberLabelFontSize;

private float originalPhoneNumberTextBoxFontSize;

private float originalComplaintLabelFontSize;

private float originalComplaintTextBoxFontSize;

private float originalMakeAnAppointmentButtonFontSize;

private float originalAppoitmentLabelFontSize;

private float originalEmptyTextFieldLabel1FontSize;

private float originalEmptyTextFieldLabel2FontSize;

private float originalEmptyTextFieldLabel3FontSize;

public AppointmentForm()

{

InitializeComponent();

}

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

{

originalFormRect = new Rectangle(this.Location, this.Size);

originalFullNameLabelRect = new Rectangle(FullNameLabel.Location, FullNameLabel.Size);

originalFullNameTextBoxRect = new Rectangle(FullNameTextBox.Location, FullNameTextBox.Size);

originalPhoneNumberLabelRect = new Rectangle(PhoneNumberLabel.Location, PhoneNumberLabel.Size);

originalPhoneNumberTextBoxRect = new Rectangle(PhoneNumberTextBox.Location, PhoneNumberTextBox.Size);

originalComplaintLabelRect = new Rectangle(ComplaintLabel.Location, ComplaintLabel.Size);

originalComplaintTextBoxRect = new Rectangle(ComplaintTextBox.Location, ComplaintTextBox.Size);

originalMakeAnAppointmentButtonRect = new Rectangle(MakeAnAppointmentButton.Location, MakeAnAppointmentButton.Size);

originalAppoitmentLabelRect = new Rectangle(AppoitmentLabel.Location, AppoitmentLabel.Size);

originalEmptyTextFieldLabel1Rect = new Rectangle(EmptyTextFieldLabel1.Location, EmptyTextFieldLabel1.Size);

originalEmptyTextFieldLabel2Rect = new Rectangle(EmptyTextFieldLabel2.Location, EmptyTextFieldLabel2.Size);

originalEmptyTextFieldLabel3Rect = new Rectangle(EmptyTextFieldLabel3.Location, EmptyTextFieldLabel3.Size);

originalFullNameLabelFontSize = FullNameLabel.Font.Size;

originalFullNameTextBoxFontSize = FullNameTextBox.Font.Size;

originalPhoneNumberLabelFontSize = PhoneNumberLabel.Font.Size;

originalPhoneNumberTextBoxFontSize = PhoneNumberTextBox.Font.Size;

originalComplaintLabelFontSize = ComplaintLabel.Font.Size;

originalComplaintTextBoxFontSize = ComplaintTextBox.Font.Size;

originalMakeAnAppointmentButtonFontSize = MakeAnAppointmentButton.Font.Size;

originalAppoitmentLabelFontSize = AppoitmentLabel.Font.Size;

originalEmptyTextFieldLabel1FontSize = EmptyTextFieldLabel1.Font.Size;

originalEmptyTextFieldLabel2FontSize = EmptyTextFieldLabel2.Font.Size;

originalEmptyTextFieldLabel3FontSize = EmptyTextFieldLabel3.Font.Size;

}

private void AppointmentForm\_Resize(object sender, EventArgs e)

{

ResizeChildrenControls();

}

private void ResizeChildrenControls() //Изменение размеров всех элементов формы при изменении размера формы

{

float widthRatio = (float)this.ClientRectangle.Width / (float)originalFormRect.Width;

float heightRatio = (float)this.ClientRectangle.Height / (float)originalFormRect.Height;

FormResizeControl.ResizeControl(FullNameLabel, originalFullNameLabelRect, originalFullNameLabelFontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(FullNameTextBox, originalFullNameTextBoxRect, originalFullNameTextBoxFontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(PhoneNumberLabel, originalPhoneNumberLabelRect, originalPhoneNumberLabelFontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(PhoneNumberTextBox, originalPhoneNumberTextBoxRect, originalPhoneNumberTextBoxFontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(ComplaintLabel, originalComplaintLabelRect, originalComplaintLabelFontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(ComplaintTextBox, originalComplaintTextBoxRect, originalComplaintTextBoxFontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(MakeAnAppointmentButton, originalMakeAnAppointmentButtonRect, originalMakeAnAppointmentButtonFontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(AppoitmentLabel, originalAppoitmentLabelRect, originalAppoitmentLabelFontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(EmptyTextFieldLabel1, originalEmptyTextFieldLabel1Rect, originalEmptyTextFieldLabel1FontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(EmptyTextFieldLabel2, originalEmptyTextFieldLabel2Rect, originalEmptyTextFieldLabel2FontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(EmptyTextFieldLabel3, originalEmptyTextFieldLabel3Rect, originalEmptyTextFieldLabel3FontSize, widthRatio, heightRatio);

}

private void SignInAsDoctorButton\_MouseHover(object sender, EventArgs e) //Закрашивание красным кнопки при наведении

{

SignInAsDoctorButton.ImageIndex = 1;

SignInAsDoctorButton.BackColor = Color.FromArgb(203, 239, 253);

}

//Возврат прозрачной кнопки

private void SignInAsDoctorButton\_MouseLeave(object sender, EventArgs e) => SignInAsDoctorButton.ImageIndex = 0;

private bool IsFullNameCorrect() //Проверка правильности введения ФИО (правильный формат: "Фамилия И. О.")

{

if (String.IsNullOrEmpty(FullNameTextBox.Text))

{

EmptyTextFieldLabel1.Text = "Это поле не может быть пустым";

EmptyTextFieldLabel1.Visible = true;

return false;

}

else if (FullNameTextBox.Text.Split(' ').Length != 3

|| !Regex.IsMatch(FullNameTextBox.Text, @"^[а-яА-Я. ]+$")

|| FullNameTextBox.Text.Split(' ')[1].Length != 2

|| FullNameTextBox.Text.Split(' ')[2].Length != 2)

{

EmptyTextFieldLabel1.Text = "Введите вашу полную фамилию и инициалы через пробел";

EmptyTextFieldLabel1.Visible = true;

return false;

}

FullNameCorrection();

return true;

}

private void FullNameTextBox\_LostFocus(object sender, EventArgs e) => IsFullNameCorrect();

private void FullNameTextBox\_TextChanged(object sender, EventArgs e) => EmptyTextFieldLabel1.Visible = false;

private bool IsPhoneNumberCorrect() //Проверка правильности введения номера телефона

{

if (PhoneNumberTextBox.Text.Length <= 4)

{

EmptyTextFieldLabel2.Visible = true;

PhoneNumberTextBox.Text = "+380";

return false;

}

else if (PhoneNumberTextBox.Text.Length != 13 || !long.TryParse(PhoneNumberTextBox.Text[1..], out long n))

{

EmptyTextFieldLabel2.Visible = true;

return false;

}

return true;

}

private void PhoneNumberTextBox\_LostFocus(object sender, EventArgs e) => IsPhoneNumberCorrect();

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

{

if (PhoneNumberTextBox.Focused)

EmptyTextFieldLabel2.Visible = false;

if (PhoneNumberTextBox.Text.Length > 13)

{

PhoneNumberTextBox.Text = PhoneNumberTextBox.Text[..^1];

PhoneNumberTextBox.SelectionStart = PhoneNumberTextBox.Text.Length;

PhoneNumberTextBox.SelectionLength = 0;

}

}

private void ComplaintTextBox\_LostFocus(object sender, EventArgs e)

{

if (String.IsNullOrEmpty(ComplaintTextBox.Text))

EmptyTextFieldLabel3.Visible = true;

}

private void ComplaintTextBox\_TextChanged(object sender, EventArgs e) => EmptyTextFieldLabel3.Visible = false;

private void FullNameCorrection() //Исправление ФИО (делает первую букву фамилии и буквы инициалов большими)

{

string[] fullName = FullNameTextBox.Text.Split(' ');

FullNameTextBox.Text = char.ToUpper(fullName[0][0]) + fullName[0].Substring(1) + ' ' +

char.ToUpper(fullName[1][0]) + fullName[1].Substring(1) + ' ' +

char.ToUpper(fullName[2][0]) + fullName[2].Substring(1);

}

private void MakeAnAppointmentButton\_Click(object sender, EventArgs e) //Нажатие на кнопку "Записаться"

{

bool allFieldsFilled = true;

if (!IsFullNameCorrect())

allFieldsFilled = false;

if (!IsPhoneNumberCorrect())

allFieldsFilled = false;

if (String.IsNullOrEmpty(ComplaintTextBox.Text))

{

EmptyTextFieldLabel3.Visible = true;

allFieldsFilled = false;

}

if (allFieldsFilled) //Если все поля заполнены правильно

{

//В конструктор БД передаётся строка подключения

Database database = new Database("Data Source=ESFORGARY;Initial Catalog=DoctorsHandbook;Integrated Security=SSPI;TrustServerCertificate=True");

try

{

int id\_dis = database.Diagnose(ComplaintTextBox.Text);

if (id\_dis == 0)

MessageBox.Show("Не могу найти болезнь с такими симптомами, попробуйте описать их подробнее");

else //Если болезнь определена

{

//Добавляем нового пациента, осмотр и рецепт

database.InsertNewPatient(FullNameTextBox.Text, PhoneNumberTextBox.Text);

int id\_ins = database.InsertNewInspection(id\_dis, ComplaintTextBox.Text, FullNameTextBox.Text, PhoneNumberTextBox.Text);

database.InsertNewRecept(id\_ins);

//Получение рецепта и болезни, переход на форму "Рецепт" с полученными данными

List<Recept> recept = database.GetRecept(id\_ins);

string disease = database.GetDisease(id\_ins);

this.Hide();

var receptForm = new ReceptForm(recept, disease);

receptForm.Closed += (s, args) => this.Close();

receptForm.Show();

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

}

private void SignInAsDoctorButton\_Click(object sender, EventArgs e) //При нажатии на "Войти как доктор"

{

//Переход на форму "Войти как доктор"

this.Hide();

var signInAsDoctorForm = new SignInAsDoctorForm();

signInAsDoctorForm.Closed += (s, args) => this.Close();

signInAsDoctorForm.Show();

}

}

}

**DoctorForm.cs**

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Windows.Forms;

using System.Windows.Forms.DataVisualization.Charting;

using ResizeControl;

using DatabaseCore;

namespace Kursach

{

public partial class DoctorForm : Form

{

private Rectangle originalFormRect;

private Rectangle originalChart1Rect;

private Rectangle originalChart2Rect;

private Rectangle originalChart3Rect;

private Rectangle originalInspectionsDataGridViewRect;

private float originalChart1FontSize;

private float originalChart2FontSize;

private float originalChart3FontSize;

private float originalInspectionsDataGridViewFontSize;

public DoctorForm()

{

InitializeComponent();

}

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

{

originalFormRect = new Rectangle(this.Location, this.Size);

originalChart1Rect = new Rectangle(chart1.Location, chart1.Size);

originalChart2Rect = new Rectangle(chart2.Location, chart2.Size);

originalChart3Rect = new Rectangle(chart3.Location, chart3.Size);

originalInspectionsDataGridViewRect = new Rectangle(InspectionsDataGridView.Location, InspectionsDataGridView.Size);

originalChart1FontSize = chart1.Font.Size;

originalChart2FontSize = chart2.Font.Size;

originalChart3FontSize = chart3.Font.Size;

originalInspectionsDataGridViewFontSize = InspectionsDataGridView.Font.Size;

//В конструктор БД передаётся строка подключения

Database database = new Database("Data Source=ESFORGARY;Initial Catalog=DoctorsHandbook;Integrated Security=SSPI;TrustServerCertificate=True");

try

{

InspectionsDataGridView.DataSource = database.GetAllInspections(); //Заполнение таблицы

Dictionary<string, int> data = database.GetCountedDiseases();

DrawChart(chart1, data, "Заболеваемость"); //Диаграмма заболеваемости

data = database.GetCountedPatients();

DrawChart(chart2, data, "Посещаемость"); //Диаграмма посещаемости

data = database.GetCountedMedication();

DrawChart(chart3, data, "Медикаменты"); //Диаграмма медикаментов

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void DrawChart(Chart chart, Dictionary<string, int> data, string name) //Создание диаграммы

{

int chartPieces = 5; //Количество частей на диаграмме

int i = 0;

chart.Titles.Add(name);

foreach (string value in data.Keys)

{

if (i < chartPieces)

chart.Series["series"].Points.AddXY(value, data[value]);

else

break;

i++;

}

}

private void DoctorForm\_Resize(object sender, EventArgs e)

{

ResizeChildrenControls();

}

private void ResizeChildrenControls() //Изменение размеров всех элементов формы при изменении размера формы

{

float widthRatio = (float)this.ClientRectangle.Width / (float)originalFormRect.Width;

float heightRatio = (float)this.ClientRectangle.Height / (float)originalFormRect.Height;

FormResizeControl.ResizeControl(chart1, originalChart1Rect, originalChart1FontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(chart2, originalChart2Rect, originalChart2FontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(chart3, originalChart3Rect, originalChart3FontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(InspectionsDataGridView, originalInspectionsDataGridViewRect, originalInspectionsDataGridViewFontSize, widthRatio, heightRatio);

}

}

}

**ReceptForm.cs**

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Windows.Forms;

using ResizeControl;

using DatabaseCore;

namespace Kursach

{

public partial class ReceptForm : Form

{

private Rectangle originalFormRect;

private Rectangle originalDiagnoseLabelRect;

private Rectangle originalReceptLabelRect;

private Rectangle originalReceptListBoxRect;

private float originalDiagnoseLabelFontSize;

private float originalReceptLabelFontSize;

private float originalReceptListBoxFontSize;

private List<Recept> recepts; //Рецепты

private string disease; //Болезнь

public ReceptForm(List<Recept> recepts, string disease) //Получение рецептов и названия болезни

{

InitializeComponent();

this.recepts = recepts;

this.disease = disease;

}

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

{

originalFormRect = new Rectangle(this.Location, this.Size);

originalDiagnoseLabelRect = new Rectangle(DiagnoseLabel.Location, DiagnoseLabel.Size);

originalReceptLabelRect = new Rectangle(ReceptLabel.Location, ReceptLabel.Size);

originalReceptListBoxRect = new Rectangle(ReceptListBox.Location, ReceptListBox.Size);

originalDiagnoseLabelFontSize = DiagnoseLabel.Font.Size;

originalReceptLabelFontSize = ReceptLabel.Font.Size;

originalReceptListBoxFontSize = ReceptListBox.Font.Size;

DiagnoseLabel.Text += disease;

foreach (Recept recept in recepts) //Заполнение списка рецептов

ReceptListBox.Items.Add("Препарат \"" + recept.Name\_Med +

"\" принимать " + recept.Dosage +

" дозы " + recept.Count\_receptions +

" раз(а) в день на протяжении " + recept.Count\_days +

" дней");

}

private void ReceptForm\_Resize(object sender, EventArgs e)

{

ResizeChildrenControls();

}

private void ResizeChildrenControls() //Изменение размеров всех элементов формы при изменении размера формы

{

float widthRatio = (float)this.ClientRectangle.Width / (float)originalFormRect.Width;

float heightRatio = (float)this.ClientRectangle.Height / (float)originalFormRect.Height;

FormResizeControl.ResizeControl(DiagnoseLabel, originalDiagnoseLabelRect, originalDiagnoseLabelFontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(ReceptLabel, originalReceptLabelRect, originalReceptLabelFontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(ReceptListBox, originalReceptListBoxRect, originalReceptListBoxFontSize, widthRatio, heightRatio);

}

}

}

**SingAsDoctorForm.cs**

using System;

using System.Drawing;

using System.Windows.Forms;

using ResizeControl;

namespace Kursach

{

public partial class SignInAsDoctorForm : Form

{

private string doctorPassword = "1111"; //Пароль

private Rectangle originalFormRect;

private Rectangle originalSignUpAsDoctorLabelRect;

private Rectangle originalDoctorPasswordLabelRect;

private Rectangle originalDoctorPasswordTextBoxRect;

private Rectangle originalSignInButtonRect;

private Rectangle originalEmptyTextFieldLabelRect;

private float originalSignUpAsDoctorLabelFontSize;

private float originalDoctorPasswordLabelFontSize;

private float originalDoctorPasswordTextBoxFontSize;

private float originalSignInButtonFontSize;

private float originalEmptyTextFieldLabelFontSize;

public SignInAsDoctorForm()

{

InitializeComponent();

}

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

{

originalFormRect = new Rectangle(this.Location, this.Size);

originalSignUpAsDoctorLabelRect = new Rectangle(SignUpAsDoctorLabel.Location, SignUpAsDoctorLabel.Size);

originalDoctorPasswordLabelRect = new Rectangle(DoctorPasswordLabel.Location, DoctorPasswordLabel.Size);

originalDoctorPasswordTextBoxRect = new Rectangle(DoctorPasswordTextBox.Location, DoctorPasswordTextBox.Size);

originalSignInButtonRect = new Rectangle(SignInButton.Location, SignInButton.Size);

originalEmptyTextFieldLabelRect = new Rectangle(EmptyTextFieldLabel.Location, EmptyTextFieldLabel.Size);

originalSignUpAsDoctorLabelFontSize = SignUpAsDoctorLabel.Font.Size;

originalDoctorPasswordLabelFontSize = DoctorPasswordLabel.Font.Size;

originalDoctorPasswordTextBoxFontSize = DoctorPasswordTextBox.Font.Size;

originalSignInButtonFontSize = SignInButton.Font.Size;

originalEmptyTextFieldLabelFontSize = EmptyTextFieldLabel.Font.Size;

}

private void SignUpAsDoctorForm\_Resize(object sender, EventArgs e)

{

ResizeChildrenControls();

}

private void ResizeChildrenControls() //Изменение размеров всех элементов формы при изменении размера формы

{

float widthRatio = (float)this.ClientRectangle.Width / (float)originalFormRect.Width;

float heightRatio = (float)this.ClientRectangle.Height / (float)originalFormRect.Height;

FormResizeControl.ResizeControl(SignUpAsDoctorLabel, originalSignUpAsDoctorLabelRect, originalSignUpAsDoctorLabelFontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(DoctorPasswordLabel, originalDoctorPasswordLabelRect, originalDoctorPasswordLabelFontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(DoctorPasswordTextBox, originalDoctorPasswordTextBoxRect, originalDoctorPasswordTextBoxFontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(SignInButton, originalSignInButtonRect, originalSignInButtonFontSize, widthRatio, heightRatio);

FormResizeControl.ResizeControl(EmptyTextFieldLabel, originalEmptyTextFieldLabelRect, originalEmptyTextFieldLabelFontSize, widthRatio, heightRatio);

}

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

{

EmptyTextFieldLabel.Visible = false;

}

private void SignInButton\_Click(object sender, EventArgs e) //При нажатии на "Войти"

{

if (String.IsNullOrEmpty(DoctorPasswordTextBox.Text)) //Сообщает, если пароль пустой

{

EmptyTextFieldLabel.Text = "Это поле не может быть пустым";

EmptyTextFieldLabel.Visible = true;

}

else if (DoctorPasswordTextBox.Text != doctorPassword) //Сообщает, если пароль неправильный

{

EmptyTextFieldLabel.Text = "Неправильный пароль";

EmptyTextFieldLabel.Visible = true;

}

else

{

//Переход на форму "Доктор"

this.Hide();

var doctorForm = new DoctorForm();

doctorForm.Closed += (s, args) => this.Close();

doctorForm.Show();

}

}

}

}

Изображение выглядит как текст, снимок экрана, внутренний, ноутбук

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание



