Міністерство освіти і науки України

Сумський державний університет

Кафедра комп’ютерних наук

Секція інформаційних технологій проектування

**ЗВІТ**

про виконання лабораторної роботи № 8-10 на тему

«Реалізація програмного забезпечення»

Виконали *Акименко В.В*

*Шальнова Р.Ю*

Варіант *1*

Перевірив *Нагорний В. В.*

Суми

2021

**Мета роботи**

1. Удосконалити навички створення програм з графічним інтерфейсом
2. Навчитися працювати з базою даних MySQL і взаємодіяти з нею за допомогою C#
3. Удосконалити навички з написання програм на мові C#

**Звіт**

| **Номер тест-кейсу** | **Опис тест-кейсу** | **Дії** | **ER – що очікується** | **Статус**  **(пройдений/не пройдений)** |
| --- | --- | --- | --- | --- |
| **UC01 LogIn** | | | | |
| 1.1 | Виконання авторизації кнопкою «Вхід». | Користувач натискає кнопку авторизації. Відкривається форма «Авторизація». Вводиться пароль та натискається кнопка «Вхід», запускається перевірка даних. | Після успішної авторизації форма «Авторизація» закривається. Відображається необхідний інтерфейс головного вікна | Пройдений |
| 1.2 | Невірна авторизація | Користувач виконує авторизацію, вводить неправильний пароль та натискає кнопку «Вхід». Запускається перевірка даних | Після неправильного вводу пароля, виводиться повідомлення про невірно введені дані. Вікно авторизації залишається відкритим. | Пройдений |
| **UC02 Load File** | | | | |
|  | Завантаження даних при натисканні кнопки «Завантажити» | Користувач знаходиться в головному меню програми. Натискає кнопку «Завантажити». | У формі виникає список записів та інформація про них згідно з даними, що знаходяться в базі даних *cloudy.sql* | Пройдений |
|  | Завантаження даних при запуску програми | Користувач запускає програму. | У формі виникає список записів та інформація про них згідно з даними, що знаходяться в базі даних *cloudy.sql* | Пройдений |
| **UC03 Edit Data** | | | | |
| 3.1 | Введення даних для редагування | Користувач обирає необхідний запис із бази даних, після чого натискає кнопку «Редагувати» у головному вікні програми. Після цього відповідні поля заповнюються необхідною інформацією, яку користувач може змінювати. Після введення цих значень натискається кнопка «Зберегти», викликається повідомлення про успішне редагування. | Після всіх виконаних дій викликається повідомлення про успішне редагування. Програма записує змінені дані до БД *cloudy.sql* | Пройдений |
| 3.2 | Неправильне введення даних для редагування | Користувач обирає необхідний запис із бази даних, після чого натискає кнопку «Редагувати» у головному вікні програми. Користувач вводить невірні дані, або дані в неправильному форматі | Після всіх виконаних дій викликається повідомлення про те, що відбулася помилка | Пройдений |
| **UC04 Add Data** | | | | |
| 4.1 | Додавання запису | Користувач вводить необхідну інформацію в поля форми, після чого натискає кнопку «Додати» | Програма записує дані, введені у поля форми в базу даних *cloudy.sql* | Пройдений |
| 4.2 | Неправильне додавання запису | Користувач неправильно вводить необхідну інформацію в поля форми, після чого натискає кнопку «Додати» | Після всіх виконаних дій викликається повідомлення про те, що відбулася помилка | Пройдений |
| **UC05 DeleteData** | | | | |
| 5.1 | Видалення запису | Користувач обирає необхідний запис із бази, після чого натискає кнопку «Видалити». | Після всіх виконаних дій запис видаляється з БД. | Пройдений |
| **UC06 SearchXY** | | | | |
| 6.1 | Створення списку записів за містом Х та місяцем Y | Користувач обирає необхідний місяць та місто, після чого натискає кнопку «Пошук» | У формі, з’являється список записів за містом Х та місяцем Y | Пройдений |
| **UC07 SearchRain** | | | | |
| 7.1 | Створення списку записів за дні, коли температура була більша за 0 та йшов дощ. | Користувач натискає кнопку «Пошук дощових днів» | У формі, з’являється список записів за дні, коли температура була більша за 0 та йшов дощ. | Пройдений |
| **UC08 LogInFile** | | | | |
| 8.1 | Запис у файл списку записів за містом Х та місяцем Y | Користувачу після виконання пункту 6.1 надається можливість зберегти інформацію до файлу, натиснувши кнопку «Створити звіт» | В залежності від вибору користувача програма записує або не записує список у файл MS Word. | Пройдений |
| 8.2 | Запис у файл списку записів за дні, коли температура була більша за 0 та йшов дощ. | Користувачу після виконання пункту 7.1 надається можливість зберегти інформацію до файлу, натиснувши кнопку «Створити звіт» | В залежності від вибору користувача програма записує або не записує список у файл MS Word. | Пройдений |

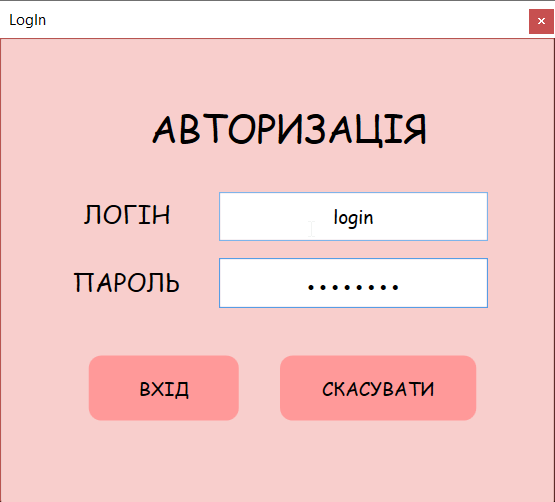


Рисунок 1 – Авторизація



Рисунок 2 – Головна форма програми



Рисунок 3 - Редагування



Рисунок 4 – Додавання



Рисунок 5 – Додавання



Рисунок 6 – Видалення



Рисунок 7 - Видалення



Рисунок 8 - Видалення



Рисунок 9 – Реалізація пошуку за містом Х та місяцем Y



Рисунок 10 – Реалізація пошуку дощових записів

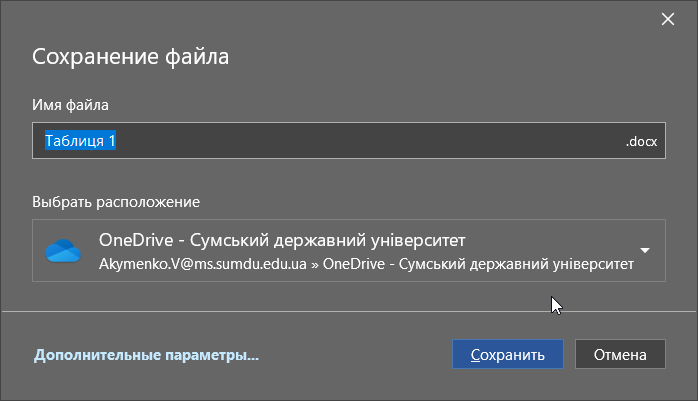


Рисунок 11 – Формування звіту

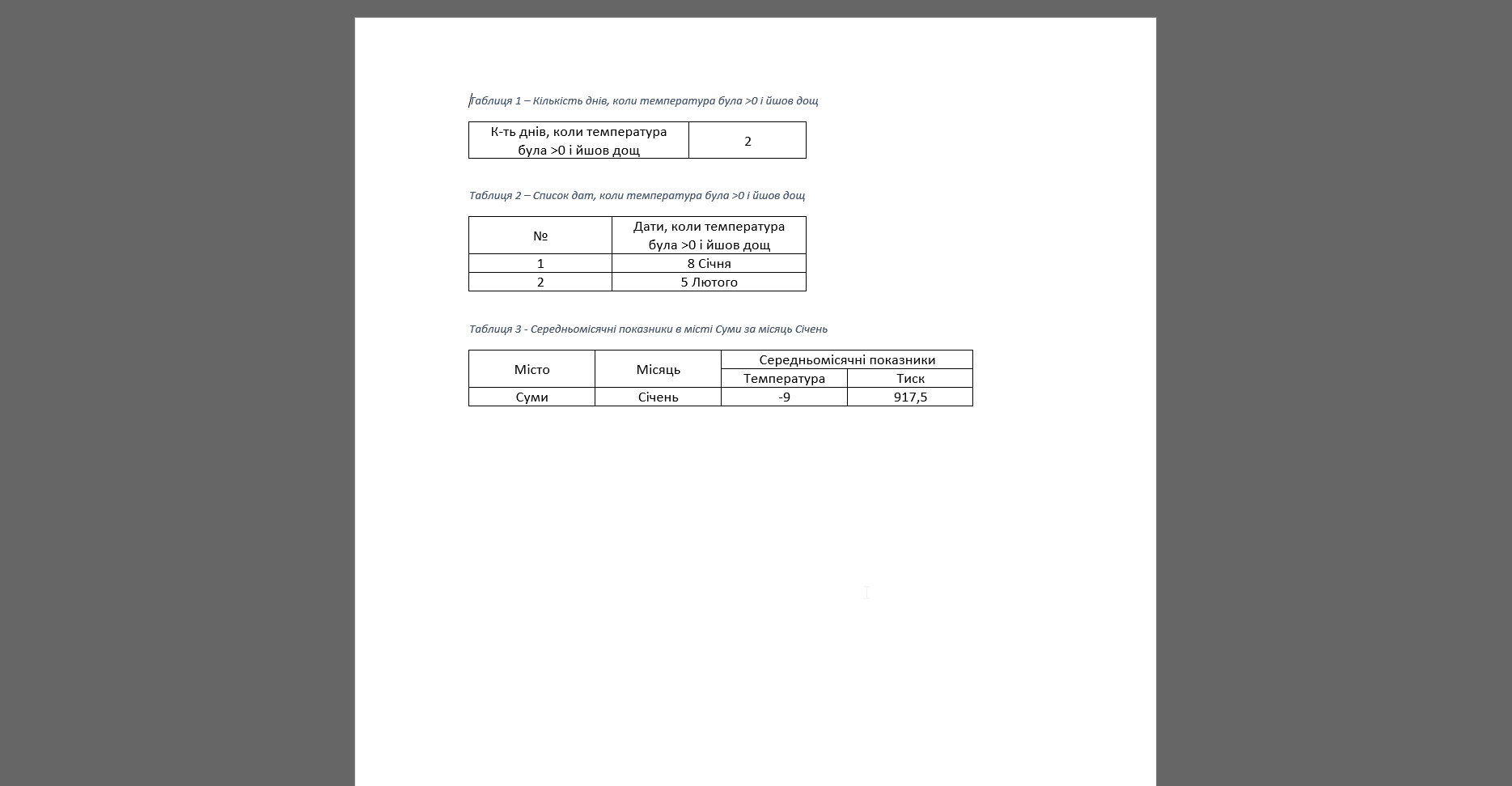


Рисунок 12 – Готовий звіт

**Код програми**

**Authorization.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using MySql.Data;

using MySql.Data.MySqlClient;

namespace cloudy

{

public class Authorization

{

public bool loged\_in { get; set; }

MySqlConnection conn;

MySqlCommand command;

MySqlDataReader reader;

public Authorization()

{

string connStr = "server=localhost; user=admin; password=; database=cloudy; port=3306;";

conn = new MySqlConnection(connStr);

command = new MySqlCommand();

command.Connection = conn;

command.Connection.Open();

}

public bool CheckLogIn(string login, string password)

{

try

{

command.CommandText = $"SELECT \* FROM editors WHERE login = '{login}' AND password = '{password}';";

reader = command.ExecuteReader();

if(reader.HasRows)

{

return true;

}

else

{

return false;

}

}

catch

{

return false;

}

}

}

}

**DataAccess.cs**

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using MySql.Data;

using MySql.Data.MySqlClient;

using System.Windows;

namespace cloudy

{

class DataAccess

{

MySqlConnection conn;

MySqlCommand command;

MySqlDataReader reader;

public DataAccess()

{

string connStr = "server=localhost; user=admin; password=; database=cloudy; port=3306;";

conn = new MySqlConnection(connStr);

command = new MySqlCommand();

command.Connection = conn;

command.Connection.Open();

}

private List<Weather> GetList(string commandText)

{

try

{

command.CommandText = commandText;

reader = command.ExecuteReader();

List<Weather> result = new List<Weather> { };

while (reader.Read())

{

result.Add(new Weather(reader.GetString(0), reader.GetInt16(1), reader.GetString(2), reader.GetInt32(3), reader.GetBoolean(4) == false ? "Нема" : "Є", reader.GetUInt32(5)));

}

reader.Close();

if (result.Count == 0)

{

return null;

}

else

{

return result;

}

}

catch

{

return null;

}

}

public List<Weather> GetWeathers()

{

return GetList("SELECT \* FROM cloudy;");

}

public int AddToBase(Weather weather)

{

try

{

command.CommandText = $"SELECT \* FROM cloudy WHERE city = '{weather.city}' AND date = {weather.day} " +

$"AND month = '{weather.month}';";

reader = command.ExecuteReader();

if (!reader.HasRows)

{

reader.Close();

command.CommandText = "INSERT INTO cloudy(city,date,month,temperature,precipitation,pressure) VALUES(@city, @date, @month, @temperature, @precipitation, @pressure)";

command.Parameters.Clear();

command.Parameters.AddWithValue("@city", weather.city);

command.Parameters.AddWithValue("@date", weather.day);

command.Parameters.AddWithValue("@month", weather.month);

command.Parameters.AddWithValue("@temperature", weather.temperature);

command.Parameters.AddWithValue("@precipitation", weather.precipitation == "Є" ? true : false);

command.Parameters.AddWithValue("@pressure", weather.pressure);

command.ExecuteNonQuery();

return 1;

}

else

{

reader.Close();

return -1;

}

}

catch

{

reader.Close();

return 0;

}

}

public bool DeleteFromBase(Weather weather)

{

try

{

command.CommandText = $"DELETE FROM cloudy WHERE city = '{weather.city}' AND date = {weather.day} " +

$"AND month = '{weather.month}' AND temperature = {weather.temperature} AND precipitation = {(weather.precipitation == "Є" ? true : false)} " +

$"AND pressure = {weather.pressure};";

if (command.ExecuteNonQuery() == 0)

{

throw new Exception();

}

return true;

}

catch

{

return false;

}

}

public List<Weather> SelectXY(string city, string month)

{

try

{

string commandText;

if (city != String.Empty && month != String.Empty)

{

commandText = $"SELECT \* FROM cloudy WHERE city = '{city}' AND month = '{month}';";

}

else if (city != String.Empty)

{

commandText = $"SELECT \* FROM cloudy WHERE city = '{city}';";

}

else

{

throw new Exception();

}

return GetList(commandText);

}

catch

{

return null;

}

}

public List<Weather> GetRainData()

{

return GetList($"SELECT \* FROM cloudy WHERE temperature > 0 AND precipitation = 1;");

}

}

}

**Globals.cs**

namespace cloudy

{

class Weather

{

public string city { get; set; }

public short day { get; set; }

public string month { get; set; }

public int temperature { get; set; }

public string precipitation { get; set; }

public uint pressure { get; set; }

public Weather(string city, short day, string month, int temp, string prec, uint pres)

{

this.city = city;

this.day = day;

this.month = month;

this.temperature = temp;

this.precipitation = prec;

this.pressure = pres;

}

}

}

**InfoWeatherForm.xaml.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace cloudy

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public static Authorization authorization = new Authorization();

private List<Weather> weathers { get; set; }

private List<Weather> selectXYList { get; set; }

private List<Weather> rainDataList { get; set; }

private SelectData selectData = new SelectData();

private Weather selectedWeather { get; set; }

private DataAccess dataAccess = new DataAccess();

private string buttonPressed;

public static string selectedCity = String.Empty;

public static string selectedMonth = String.Empty;

public MainWindow()

{

InitializeComponent();

}

private void MainWindow\_Loaded(object sender, RoutedEventArgs e)

{

this.Show();

LoadBase();

for(int i = 1; i <= 31; i++)

{

day\_box.Items.Add(Convert.ToString(i));

}

}

private void DisplayList(List<Weather> list)

{

WeatherTable.Items.Clear();

foreach(Weather element in list)

{

WeatherTable.Items.Add(element);

}

}

private bool LoadBase()

{

try

{

List<Weather> result = dataAccess.GetWeathers();

ClearList();

if (result == null)

{

MessageBox.Show("База даних порожня", "Увага!", MessageBoxButton.OK, MessageBoxImage.Warning);

}

else

{

foreach (Weather element in result)

{

AddToList(element);

if (!city\_x.Items.Contains(element.city))

{

city\_x.Items.Add(element.city);

}

}

}

WeatherTable.SelectedItem = null;

return true;

}

catch

{

MessageBox.Show("Не вдалось прочитати базу даних", "Помилка!", MessageBoxButton.OK, MessageBoxImage.Warning);

return false;

}

}

public void ToggleDisplay()

{

EditForm.Visibility = (EditForm.Visibility == Visibility.Visible) ? Visibility.Hidden : Visibility.Visible;

UserForm.Visibility = (UserForm.Visibility == Visibility.Visible) ? Visibility.Hidden : Visibility.Visible;

}

private void ClearList()

{

weathers = new List<Weather> { };

WeatherTable.Items.Clear();

city\_x.Items.Clear();

}

private void ClearBoxes()

{

city\_box.Text = String.Empty;

day\_box.Text = String.Empty;

month\_box.Text = String.Empty;

precip\_box.Text = String.Empty;

pressure\_box.Text = String.Empty;

temp\_box.Text = String.Empty;

}

private bool UserInputOk()

{

return !String.IsNullOrWhiteSpace(city\_box.Text) && !String.IsNullOrWhiteSpace(day\_box.Text) &&

!String.IsNullOrWhiteSpace(month\_box.Text) && !String.IsNullOrWhiteSpace(temp\_box.Text) &&

!String.IsNullOrWhiteSpace(precip\_box.Text) && !String.IsNullOrWhiteSpace(pressure\_box.Text);

}

private bool AddToList(Weather weather)

{

try

{

WeatherTable.Items.Add(weather);

weathers.Add(weather);

return true;

}

catch

{

MessageBox.Show("Не вдалось додати запис", "Помилка!", MessageBoxButton.OK, MessageBoxImage.Warning);

return false;

}

}

private bool DeleteFromList(Weather weather)

{

try

{

if (dataAccess.DeleteFromBase(weather))

{

return true;

}

else

{

MessageBox.Show("Не вдалось видалити запис з бази даних", "Помилка!", MessageBoxButton.OK, MessageBoxImage.Warning);

return false;

}

}

catch

{

MessageBox.Show("Не вдалось видалити запис", "Помилка!", MessageBoxButton.OK, MessageBoxImage.Warning);

return false;

}

}

private void AddButton\_Click(object sender, RoutedEventArgs e)

{

try

{

if(buttonPressed == (string)EditButton.Content)

{

throw new Exception("Увімкнутий режим редагування");

}

Weather weather = new Weather(city\_box.Text, Convert.ToInt16(day\_box.Text), month\_box.Text, Convert.ToInt16(temp\_box.Text), precip\_box.Text, Convert.ToUInt32(pressure\_box.Text));

int result = dataAccess.AddToBase(weather);

if (result == 1)

{

LoadBase();

ClearBoxes();

}

else if(result == 0)

{

throw new Exception("Не вдалось додати запис");

}

else

{

throw new Exception("Записи на цю дату в даному місті вже існують");

}

}

catch(Exception ex)

{

MessageBox.Show(ex.Message, "Помилка", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

private void SaveButton\_Click(object sender, RoutedEventArgs e)

{

try

{

if (!UserInputOk())

{

throw new Exception("Неправильні вхідні дані");

}

if (buttonPressed == (string)EditButton.Content)

{

if(DeleteFromList(selectedWeather))

{

buttonPressed = String.Empty;

AddButton\_Click(sender, e);

}

else

{

throw new Exception("Не вдалось відредагувати запис");

}

}

else

{

AddButton\_Click(sender, e);

}

ClearBoxes();

WeatherTable.SelectedItem = null;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Помилка!", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

private void LogInButton\_Click(object sender, RoutedEventArgs e)

{

LogInForm logInForm = new LogInForm();

logInForm.ShowDialog();

if(MainWindow.authorization.loged\_in)

{

ToggleDisplay();

}

}

private void EditButton\_Click(object sender, RoutedEventArgs e)

{

try

{

buttonPressed = (string)EditButton.Content;

selectedWeather = (Weather)WeatherTable.SelectedItem;

if(selectedWeather == null)

{

throw new Exception();

}

city\_box.Text = selectedWeather.city.Trim();

day\_box.Text = Convert.ToString(selectedWeather.day).Trim();

month\_box.Text = selectedWeather.month;

temp\_box.Text = Convert.ToString(selectedWeather.temperature).Trim();

precip\_box.Text = selectedWeather.precipitation.Trim();

pressure\_box.Text = Convert.ToString(selectedWeather.pressure).Trim();

}

catch

{

MessageBox.Show("Ви не обрали запис", "Помилка!", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

private void DeleteButton\_Click(object sender, RoutedEventArgs e)

{

try

{

selectedWeather = (Weather)WeatherTable.SelectedItem;

if (selectedWeather == null)

{

throw new Exception();

}

DeleteFromList(selectedWeather);

LoadBase();

ClearBoxes();

WeatherTable.SelectedItem = null;

}

catch

{

MessageBox.Show("Ви не обрали запис", "Помилка!", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

private void SearchButton\_Click(object sender, RoutedEventArgs e)

{

try

{

selectedCity = city\_x.Text;

selectedMonth = month\_y.Text;

selectXYList = dataAccess.SelectXY(selectedCity, selectedMonth);

if(selectXYList == null)

{

throw new Exception("Жодного запису не знайдено");

}

else

{

ClearList();

city\_x.Text = String.Empty;

month\_y.Text = String.Empty;

DisplayList(selectXYList);

}

}

catch(Exception ex)

{

MessageBox.Show(ex.Message, "Помилка!", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

private void SearchRainButton\_Click(object sender, RoutedEventArgs e)

{

try

{

rainDataList = dataAccess.GetRainData();

if (rainDataList == null)

{

throw new Exception("Жодного запису не знайдено");

}

else

{

DisplayList(rainDataList);

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Помилка!", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

private void ReportButton\_Click(object sender, RoutedEventArgs e)

{

try

{

if (selectXYList == null)

{

throw new Exception("Здійсніть спочатку пошук записів за містом та місяцем, або оберіть потрібні місто та місяць");

}

if (rainDataList == null)

{

rainDataList = dataAccess.GetRainData();

}

selectData.WriteData(selectXYList, rainDataList);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Помилка!", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

private void RefreshButton\_Click(object sender, RoutedEventArgs e)

{

LoadBase();

}

}

}

**LogInForm.xaml.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace cloudy

{

/// <summary>

/// Логика взаимодействия для LogInForm.xaml

/// </summary>

public partial class LogInForm : Window

{

public LogInForm()

{

InitializeComponent();

}

private void Continue\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

private void Enter\_Click(object sender, RoutedEventArgs e)

{

MainWindow.authorization.loged\_in = MainWindow.authorization.CheckLogIn(login.Text.Trim(), password.Password.Trim());

if (MainWindow.authorization.loged\_in)

{

this.Close();

}

else

{

MessageBox.Show("Неправильний логін або пароль", "Помилка!");

}

}

}

}

**SelectData.cs**

using System;

using System.Collections.Generic;

using System.Windows;

namespace cloudy

{

class SelectData

{

Microsoft.Office.Interop.Word.Application wordApp;

Microsoft.Office.Interop.Word.Document wordDoc;

string filePath;

public void WriteData(List<Weather> selectXY, List<Weather> rainData)

{

try

{

filePath = Environment.CurrentDirectory.ToString();

wordApp = new Microsoft.Office.Interop.Word.Application();

wordDoc = wordApp.Documents.Add(filePath + "\\Шаблон\_Пошуку.dotx");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message + char.ConvertFromUtf32(13) +

"Помістіть файл Шаблон\_Пошуку.dotx" + char.ConvertFromUtf32(13) +

"у каталог із ехе-файлом програми і повторіть збереження", "Помилка", MessageBoxButton.OK, MessageBoxImage.Error);

}

try

{

ReplaceText(MainWindow.selectedCity, "[X]");

ReplaceText(Convert.ToString(AverageTemp(selectXY)), "[T]");

ReplaceText(MainWindow.selectedMonth, "[Y]");

ReplaceText(Convert.ToString(AveragePres(selectXY)), "[P]");

ReplaceText(rainData);

wordDoc.Save();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message + char.ConvertFromUtf32(13) +

"Помилка збереження відібраних даних", "Помилка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void ReplaceText(string textToReplace, string replacedText)

{

Object missing = Type.Missing;

Object wrap = Microsoft.Office.Interop.Word.WdFindWrap.wdFindContinue;

Object replace = Microsoft.Office.Interop.Word.WdReplace.wdReplaceAll;

Microsoft.Office.Interop.Word.Range content;

content = wordDoc.Range(wordDoc.Content.Start, wordDoc.Content.End);

Microsoft.Office.Interop.Word.Find find = wordApp.Selection.Find;

find.Text = replacedText;

find.Replacement.Text = textToReplace;

find.Execute(

FindText: Type.Missing,

MatchCase: false,

MatchWholeWord: false,

MatchWildcards: false,

MatchAllWordForms: false,

Forward: true,

Wrap: wrap,

Format: false,

ReplaceWith: missing,

Replace: replace

);

}

private void ReplaceText(List<Weather> rainData)

{

if (rainData == null)

{

return;

}

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

{

string month = rainData[i].month;

month = month.Replace("ень", "ня");

month = month.Replace("пад", "пада");

month = month.Replace("ий", "ого");

wordDoc.Tables[2].Rows.Add();

wordDoc.Tables[2].Cell(2 + i, 1).Range.Text = Convert.ToString(i + 1);

wordDoc.Tables[2].Cell(2 + i, 2).Range.Text = rainData[i].day + " " + month;

}

wordDoc.Tables[1].Cell(1, 2).Range.Text = Convert.ToString(rainData.Count);

}

private double AverageTemp(List<Weather> weathers)

{

if(weathers == null)

{

return 0;

}

double result = 0;

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

{

result += weathers[i].temperature;

}

return result / weathers.Count;

}

private double AveragePres(List<Weather> weathers)

{

if (weathers == null)

{

return 0;

}

double result = 0;

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

{

result += weathers[i].pressure;

}

return result / weathers.Count;

}

}

}

**Висновки:**

Під час виконання лабораторної роботи ми здобули навички створення додатків з графічним інтерфейсом, навички взаємодії з базою даних MySQL, а також навички написання програм з використанням мови програмування с#.