МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М. Є. Жуковського

«Харківський авіаційний інститут»

Факультет програмної інженерії та бізнесу

Кафедра інженерії програмного забезпечення

**Лабораторна робота № 8**

з дисципліни Проєктування інтерфейсу користувача

(назва дисципліни)

на тему: «Оцінка зручності використання інтерфейсу користувача сайту засобами сервісу UsabilityHub»

Виконав: студент 2 курсу групи № 622п

Галузь знань 12 «Інформаційні

технології»

(код та найменування)

Спеціальність 121 «Інженерія

програмного забезпечення»

(код та найменування)

Освітня програма «Інженерія

програмного забезпечення»

Зайченко Я. І.

(прізвище й ініціали студента)

Прийняв: старший викладач на каф. 603

Лучшева О. В.

(посада, науковий ступінь, прізвище й ініціали)

Національна шкала: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Харків – 2023

ЗМІСТ

[Мета роботи: 3](#_Toc154084882)

[Порядок виконання роботи 3](#_Toc154084883)

[Зміст звіту 3](#_Toc154084884)

[Варіант 9 4](#_Toc154084885)

[Короткі теоретичні відомості 5](#_Toc154084886)

[Виконання роботи 6](#_Toc154084887)

[Опис прикладів використання шаблону в типових рішеннях ПП. 6](#_Toc154084888)

[Опис послідовності реалізації проекту у Visual Studio. 7](#_Toc154084889)

[Реалізація використання шаблону 7](#_Toc154084890)

[Висновок 9](#_Toc154084891)

[*Додаток А* Приклад коду шаблону для форми 10](#_Toc154084892)

[*Додаток Б* Код реалізації шаблону для форми 11](#_Toc154084893)

Мета роботи: сформувати навички тестування сайту засобами сервісу UsabilityHub UsabilityHub.

Порядок виконання роботи

1. Зареєструватися на сайті https://usabilityhub.com.
2. Вивчити особливості розроблення тестів: Five Second Test, First Clіck Test, Preference Test, Navigation Test.
3. Провести тестування зручності використання сайту за допомогою п'яти респондентів.
4. Скласти звіт за результатами тестування п'яти респондентів, який включає скрини результатів тестування .
5. Проаналізувати отримані результати для окремих видів тестування.
6. Сформулювати загальні висновки щодо результатів тестування.
7. Оформити звіт, перевірити у викладача та захистити роботу.

Зміст звіту

1. Зміст пояснювальної записки.
2. Вступ (коротко описати особливості тестування юзабіліті користувальницького інтерфейсу та платформи usabilityhub).
3. План тестування
   1. Вибір програмного забезпечення (описати, який програмний продукт (згідно варіанту до 2 л.р.) буде предметом тестування).
   2. Вибір пристроїв (визначити, які типи пристроїв будуть використані: персональний комп'ютер, планшет, мобільний телефон та інш.)
   3. Вибір функцій користувача (визначити, які функції інтерфейсу будуть протестовані (пошук реєстрації на рейс, купівля квитка на конкретний напрямок, пересилання повідомлення тощо).
   4. Вибір видів тестування (описати види тестування), що будуть проведені.
4. Розроблення тестів (опис створення тестів, включаючи скрини)
   1. Five Second Test
   2. First Clіck Test
   3. Preference Test
   4. Navigation Test
5. Аналіз результатів тестування (аналіз усіх видів тестування з висновками по кожному виду тестування, включаючи скрини)
   1. Five Second Test
   2. First Clіck Test
   3. Preference Test
   4. Navigation Test
6. Загальні висновки

Варіант 9

Сайт для покупки квитків на театральні вистави через мережу Інтернет.

<https://kontramarka.ua/uk/theatre>

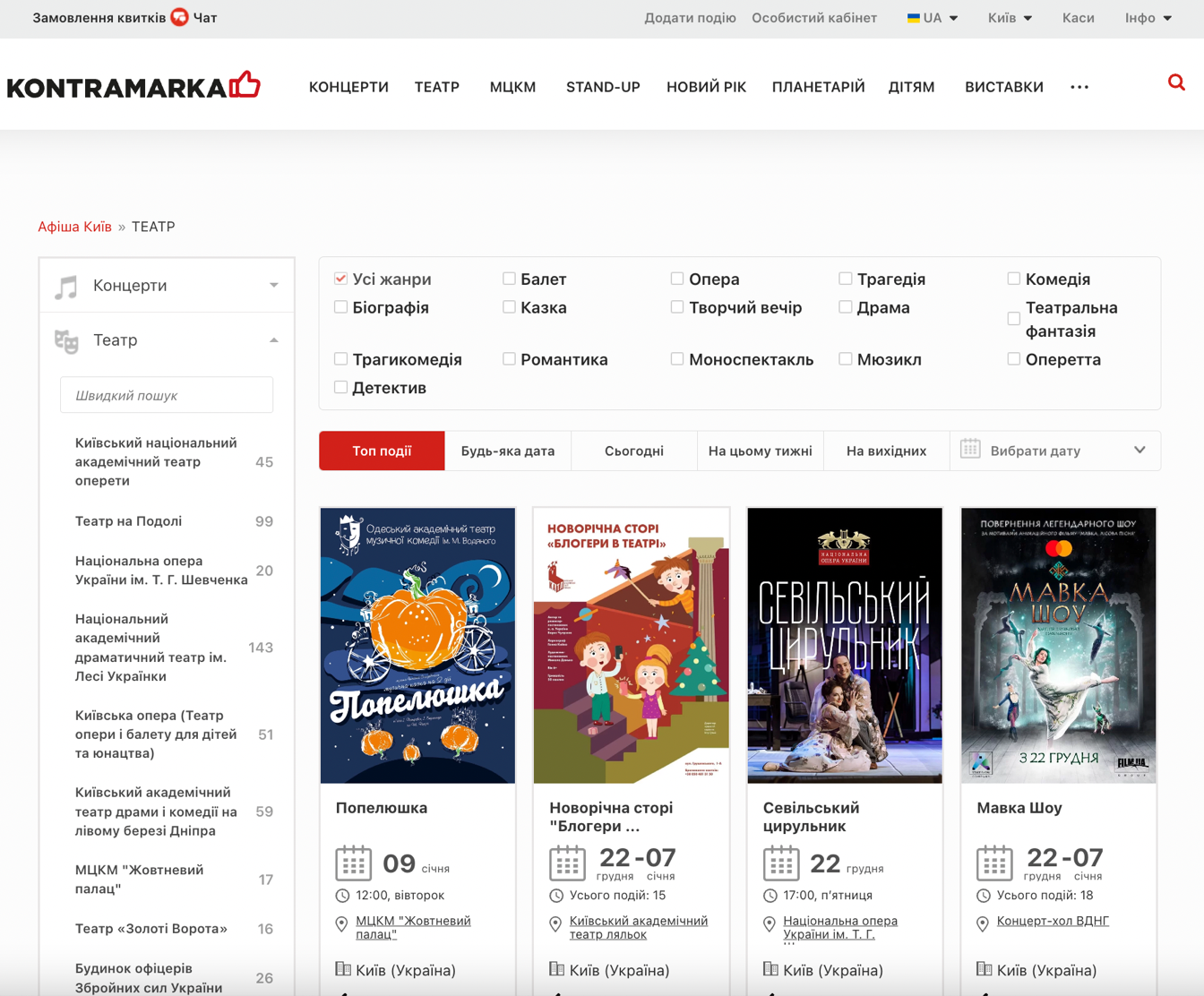


Рисунок 1 – Вид сайту

Короткі теоретичні відомості

**UsabilityHub** - це онлайн платформа для проведення юзабіліті-тестів та проектних опитувань (<https://app.usabilityhub.com/>).

До складу Usabilityhub входять:

* **Five Second Test.** Тестування "5 секунд". Користувачів просять згадати, що вони запам'ятали після 5-ти секундного перегляду сайту або програми.
* **Click Test.** Тест дозволяє визначити наскільки ефективно використання тих чи інших елементів інтерфейсу під час вирішення різних завдань. Карта кліків — інструмент для вимірювання та відображення статистики кліків на сайті. Карта кліків допомагає оцінювати юзабіліті сайту, виявляти найбільш клікабельні елементи, визначати, які елементи дизайну здаються відвідувачам сайтів посиланням.
* **Question.** Тест-питання. Прямі та чесні відгуки від реальних користувачів про дизайн сайту або програми.
* **Navigation Test.** Навігаційний тест. Дозволяє перевірити легкість та ефективність навігації користувачів на сайті або у додатку.
* **Preference Test.** Тест переваг. Надання користувачам вибрати найбільш привабливий для них та зручний дизайн сайту з декілька варіантів.

Виконання роботи

Зареєструватися на сайті <https://app.lyssna.com/users/sign_up> (раніше https://usabilityhub.com): ввести параметри та натиснути кнопку Create account або обрати реєстрацію через Google (Рис. 1).

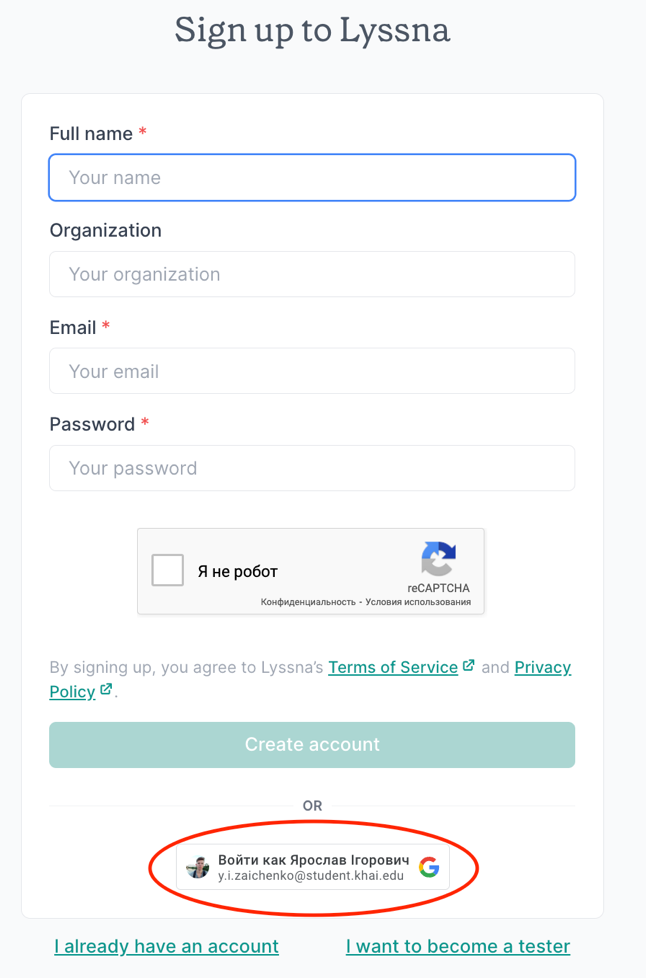


Рисунок 2 – реєстрація акаунту за допомогою пошти

Натиснути на кнопку Create new test

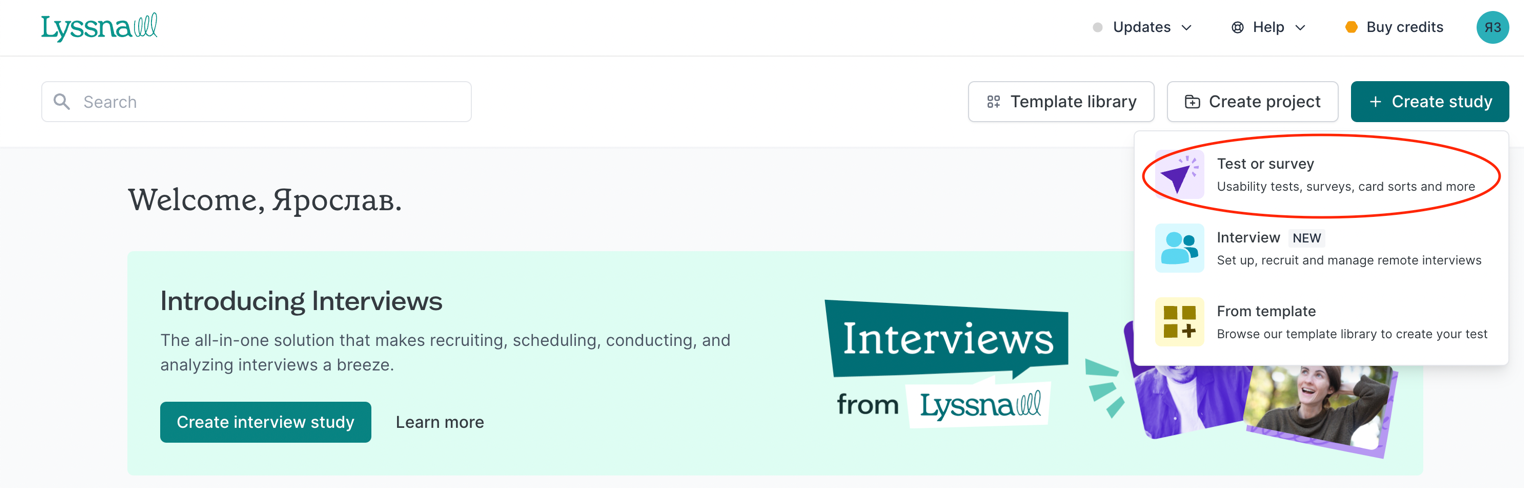


Рисунок 3 – Створення нового опитування

Ввести ім’я тесту, мову, тип пристрою

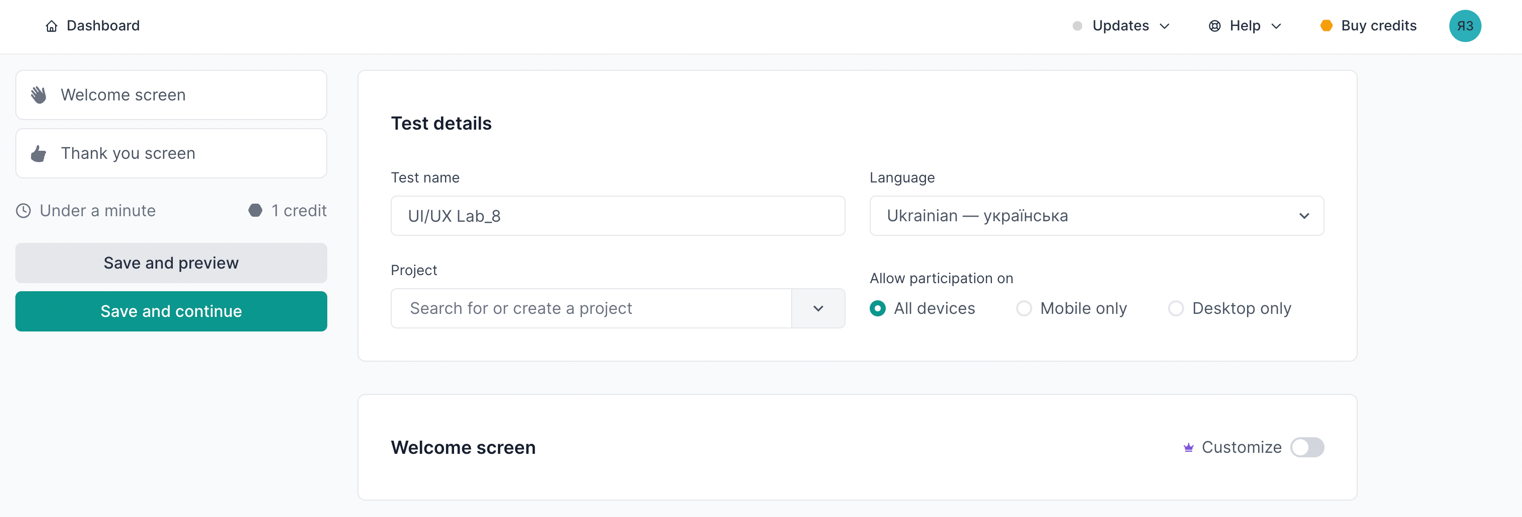


Рисунок 4 – Введення параметрів

Натискаємо на ім’я тесту та обираємо його вид

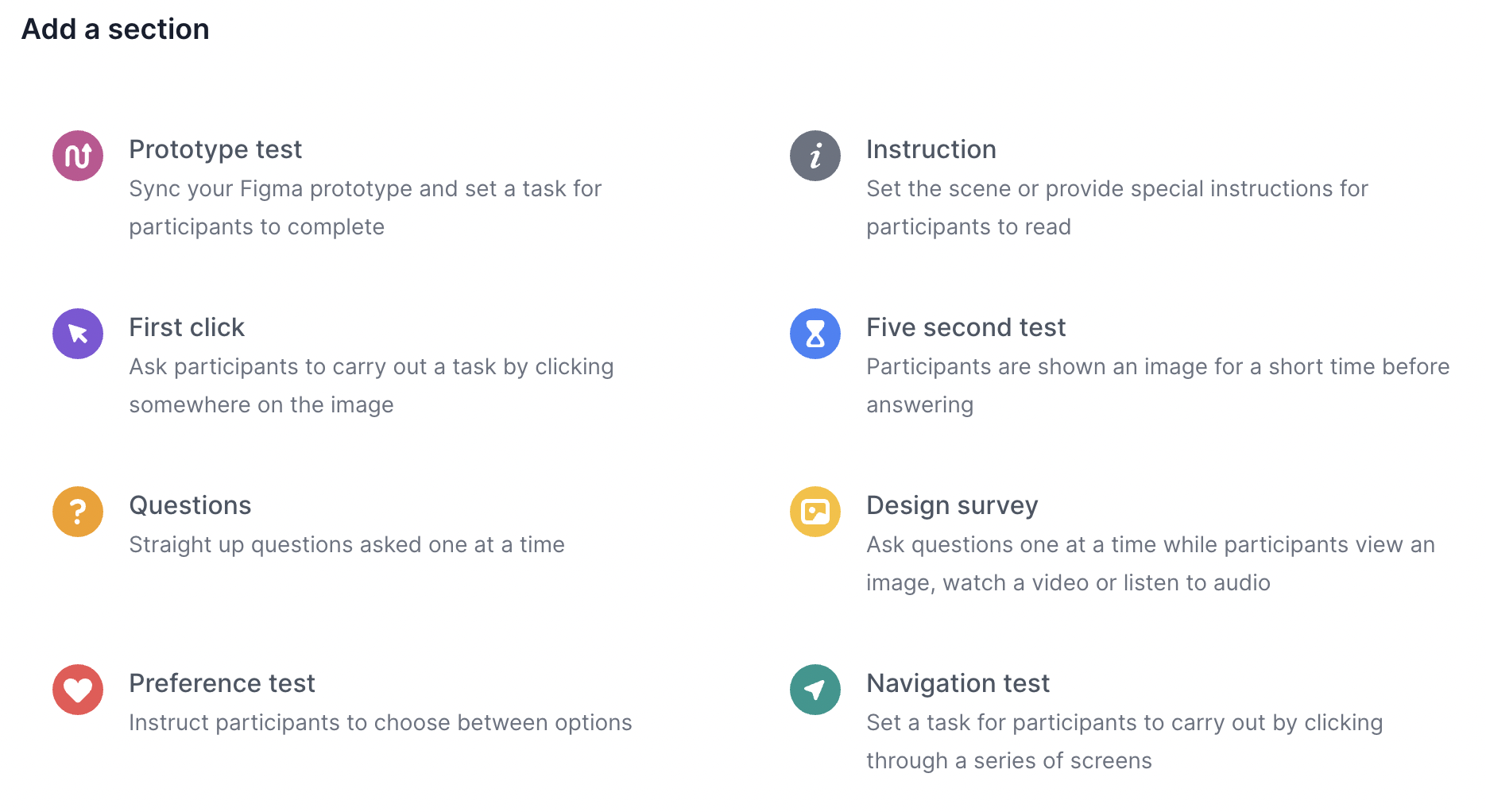


Рисунок 5 – визначення типу тестування

Вводимо параметри тесту

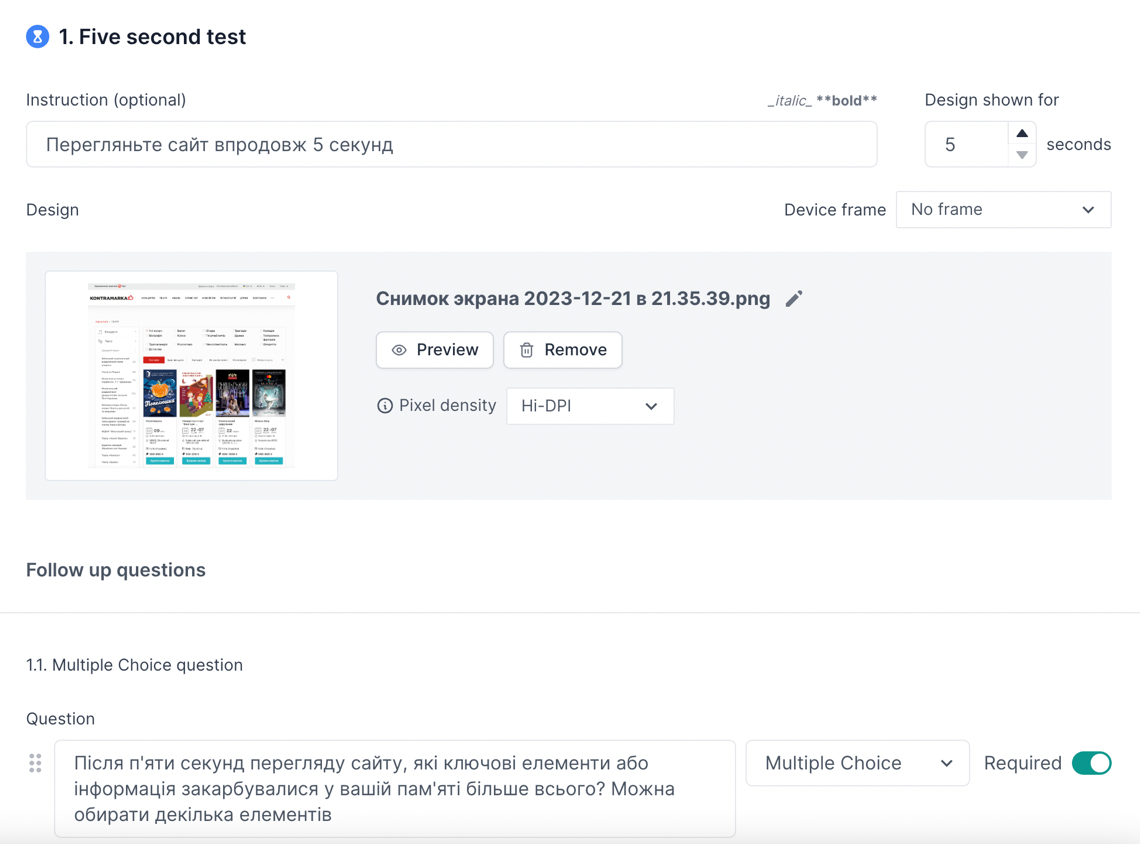


Рисунок 6 – створення Five Second Test

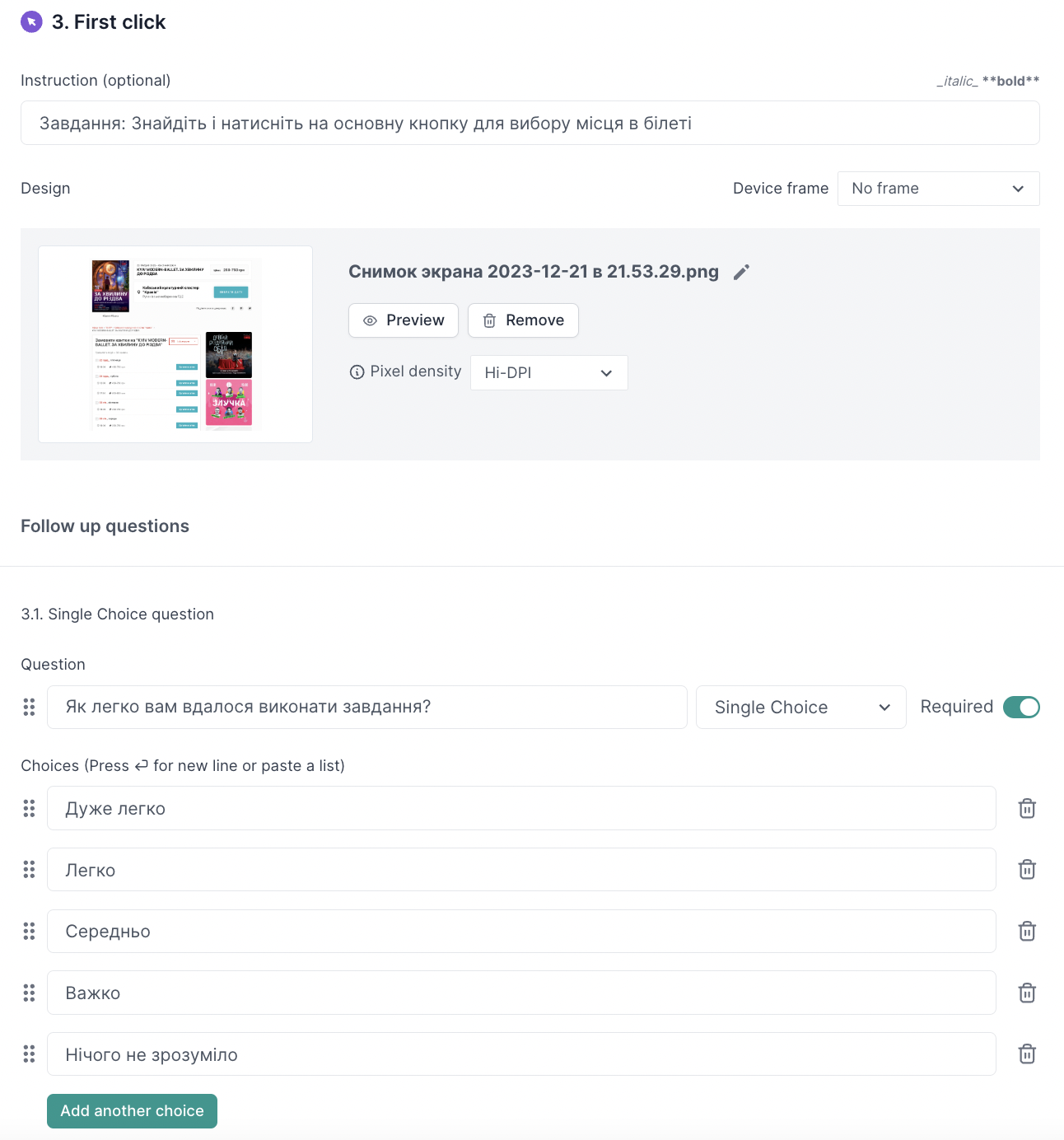


Рисунок 7 – створення First Clіck Test

Опис прикладів використання шаблону в типових рішеннях ПП.

Шаблон інтерфейсу користувача з використанням концепції меню та розкриваючої або спливаючої панелі може бути застосований в різних програмних продуктах для полегшення вибору значень, забезпечення зручності користувача та збільшення ефективності взаємодії. Ось кілька прикладів застосування подібного шаблону:

1. Онлайн-магазин електроніки:

* Меню: категорії товарів (наприклад, смартфони, ноутбуки, аксесуари).
* Панель: розкривна або спливаюча панель з фільтрами (бренд, ціновий діапазон, характеристики).
* Вибір значень: користувач може вибрати категорію в меню, а потім налаштувати параметри в розкривній панелі для точного вибору товару.

1. Інтернет-банкінг:

* Меню: рахунки, перекази, кредити.
* Панель: додаткові опції для кожної категорії (історія операцій, розширені налаштування кредиту).
* Вибір значень: користувач може швидко переглядати основні функції через меню, а для більш детального вибору використовувати розкривні панелі.

1. Управління завданнями в проектному менеджері:

* Меню: проекти, завдання, команди.
* Панель: додаткові фільтри та опції (стан завдань, терміни виконання, відповідальні особи).
* Вибір значень: користувач може швидко переглядати свої проекти та завдання через меню, а за допомогою розкривних панелей точно налаштовувати фільтрацію і пошук.

1. Система управління персоналом:

* Меню: кадри, відділи, звіти.
* Панель: додаткові фільтри (посада, стаж роботи, заробітна плата).
* Вибір значень: кадровий менеджер може використовувати меню для швидкого переходу між розділами, а панелі фільтрів для зручного вибору певних характеристик співробітників.

Ці приклади показують, як шаблон з меню та розкриваючою або спливаючою панеллю може бути використаний в різних контекстах для полегшення навігації та вибору значень.

Опис послідовності реалізації проекту у Visual Studio.

Для початку розставимо компоненти WinForms на вікні: *StripMenu*, *StripMenuItem*, *ComboBox* та *TextBox*. Розташування компонентів на формі подано на рисунку 1.

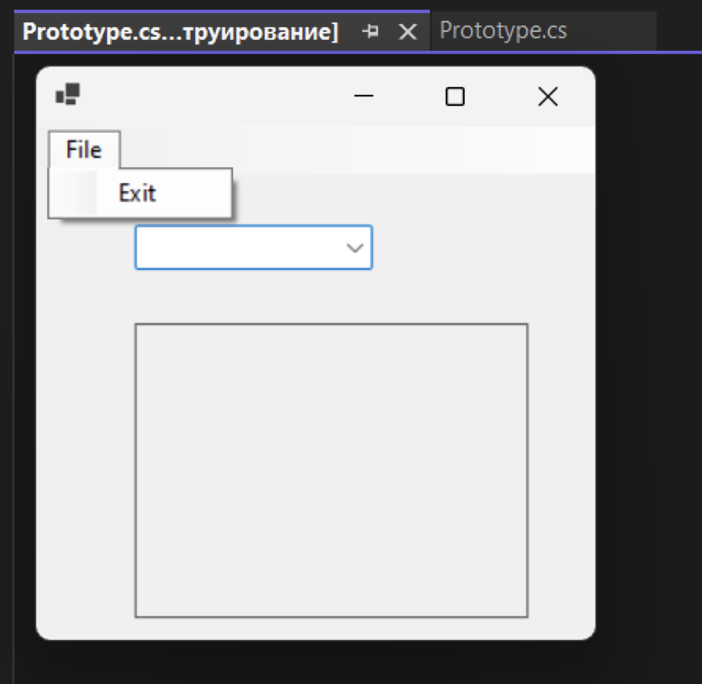


Рисунок 7 – Конструктор шаблону форми в середовищі розробки

Після створення форми можна перейти до написання коду для шаблону. Приклад поданий в Додатку А

Реалізація використання шаблону

На рисунку 2 подано приклад використання створеного шаблону. В програмі використовується концепція меню для відкриття та зберігання файлів. В якості елементу розкривної або спливаючої панель, що містить більш складний інтерфейс користувача для вибору значень використовується *ComboBox* з вибором заготовлених значень або введення з клавіатури власного значення розміру шрифту.

Також реалізовано контекстне меню для текстового поля, для вибору типів форматування тексту.

Шаблон відповідає вимозі повідомлення користувачем додатку значення через вибір визначених варіантів та введення відмінних значень з клавіатури.

Код реалізованого прикладу шаблону подано в Додатку Б.

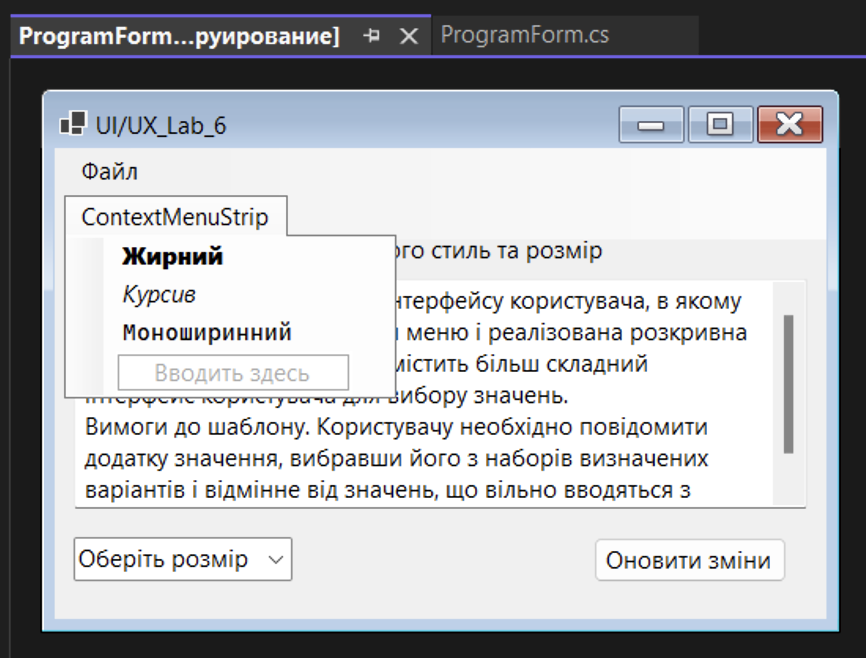


Рисунок 8 – Реалізація створеного шаблону

Висновок

На лабораторній роботі було розроблено шаблон дизайну інтерфейсу користувача, який використовує концепцію меню та розкривної або спливаючої панелі. Шаблон був створений з вимогою надати користувачеві зручний інтерфейс для вибору значень, використовуючи визначені варіанти та вільний ввід з клавіатури.

Приклад використання шаблону може бути запрограмований у різних сценаріях, таких як управління товарами в онлайн-магазині, вибір банківських операцій в інтернет-банкінгу або фільтрація завдань у системі управління проектами.

Проект був реалізований в середовищі розробки Visual Studio. Послідовно були виконані етапи створення інтерфейсу, налаштування меню та панелі, а також додавання логіки вибору значень.

Усі скріншоти процесу розробки та реалізованого шаблону додаються до звіту, що надає повну ілюстрацію кожного етапу проекту. В результаті цієї роботи був успішно реалізований шаблон, який відповідає вимогам та надає користувачеві зручність та функціональність для вибору значень.

*Додаток А*  
Приклад коду шаблону для форми

using System;

using System.Windows.Forms;

namespace UIExample

{

public partial class MainForm : Form

{

public MainForm()

{

InitializeUI();

}

private void InitializeUI()

{

// MenuStrip

MenuStrip menuStrip = new MenuStrip();

ToolStripMenuItem fileMenu = new ToolStripMenuItem("File");

ToolStripMenuItem exitMenuItem = new ToolStripMenuItem("Exit");

exitMenuItem.Click += (sender, e) => Close();

fileMenu.DropDownItems.Add(exitMenuItem);

menuStrip.Items.Add(fileMenu);

Controls.Add(menuStrip);

// ComboBox for selecting values

ComboBox valuesComboBox = new ComboBox();

//valuesComboBox.DropDownStyle = ComboBoxStyle.DropDownList; // Prevents manual input

valuesComboBox.Items.AddRange(new object[] { "Option 1", "Option 2", "Option 3" });

valuesComboBox.Location = new System.Drawing.Point(50, 50);

Controls.Add(valuesComboBox);

// Panel for more complex UI

Panel complexPanel = new Panel();

complexPanel.BorderStyle = BorderStyle.FixedSingle;

complexPanel.Size = new System.Drawing.Size(200, 150);

complexPanel.Location = new System.Drawing.Point(50, 100);

// Add more UI elements to the complexPanel as needed

Controls.Add(complexPanel);

}

}

static class Program

{

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new MainForm());

}

}

}

*Додаток Б*  
Код реалізації шаблону для форми

namespace \_UI\_UX\_Lab\_6

{

public partial class ProgramForm : Form

{

public ProgramForm()

{

InitializeComponent();

Check\_Empty();

}

bool bold = false;

bool italic = false;

bool mono = false;

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

{

ChangeFormat();

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

Check\_Empty();

}

public void Check\_Empty()

{

if (textBox1.Text.Length == 0)

{

comboBox\_Size.Enabled = false;

Update\_button.Enabled = false;

}

else

{

comboBox\_Size.Enabled = true;

Update\_button.Enabled = true;

}

}

public int GetSize()

{

int i = comboBox\_Size.SelectedIndex;

int size = 0;

try

{

switch (i)

{

case 0: size = 8; break;

case 1: size = 10; break;

case 2: size = 11; break;

case 3: size = 12; break;

case 4: size = 14; break;

case 5: size = 16; break;

case 6: size = 20; break;

default: size = Convert.ToInt16(comboBox\_Size.Text); break;

}

}

catch (Exception) { MessageBox.Show("Не обрано розмір тексту", "Помилка"); return Convert.ToInt16(textBox1.Font.Size); }

return size;

}

void ChangeFormat()

{

Font currentFont = textBox1.Font;

FontStyle style = FontStyle.Regular;

if (bold)

style |= FontStyle.Bold;

if (italic)

style |= FontStyle.Italic;

int size = GetSize();

if (mono)

{

FontFamily fontFamily = new FontFamily("Consolas");

currentFont = new Font(fontFamily, size, style);

}

else

{

currentFont = new Font("Segoe UI", size, style);

}

textBox1.Font = currentFont;

}

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

{

bold = !bold;

жирнийToolStripMenuItem.Checked = bold;

}

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

{

italic = !italic;

курсивToolStripMenuItem.Checked = italic;

}

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

{

mono = !mono;

моношириннийToolStripMenuItem.Checked = mono;

}

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

{

OpenFile();

}

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

{

SaveFile();

}

private void SaveFile()

{

using (SaveFileDialog saveFileDialog = new SaveFileDialog())

{

saveFileDialog.Filter = "Текстові файли (\*.txt)|\*.txt|Всі файли (\*.\*)|\*.\*";

saveFileDialog.Title = "Зберегти текстовий файл";

saveFileDialog.FileName = "Мій\_текстовий\_файл.txt";

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

{

try

{

File.WriteAllText(saveFileDialog.FileName, textBox1.Text);

MessageBox.Show("Файл успішно збережено.", "Успіх");

}

catch (Exception ex)

{

MessageBox.Show($"Помилка при збереженні файлу: {ex.Message}", "Помилка");

}

}

}

}

private void OpenFile()

{

using (OpenFileDialog openFileDialog = new OpenFileDialog())

{

openFileDialog.Filter = "Текстові файли (\*.txt)|\*.txt|Всі файли (\*.\*)|\*.\*";

openFileDialog.Title = "Відкрити текстовий файл";

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

{

try

{

string content = File.ReadAllText(openFileDialog.FileName);

textBox1.Text = content;

MessageBox.Show("Файл успішно відкрито.", "Успіх");

}

catch (Exception ex)

{

MessageBox.Show($"Помилка при відкритті файлу: {ex.Message}", "Помилка");

}

}

}

}

}

}