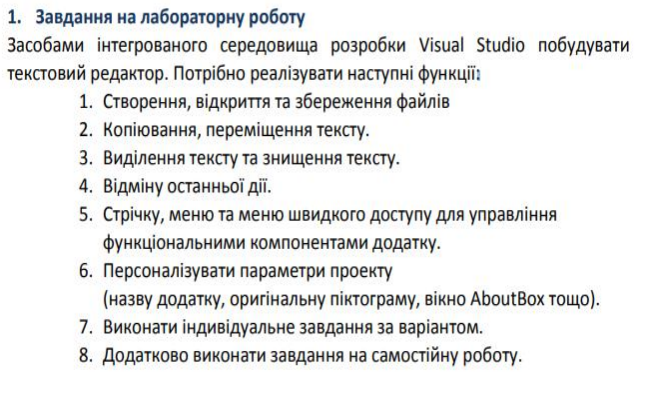
**Лабораторна робота №1**

**Тема роботи:** Інтерфейс віконного додатку.

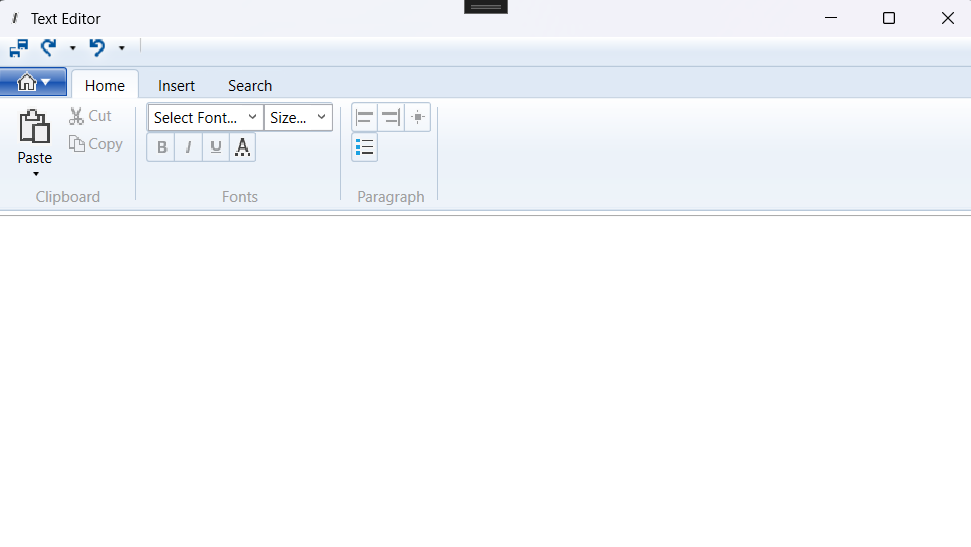
**Мета роботи**: дослідження основних підходів та технологійреалізаціїінтерфейсу універсалього додатку Windows.

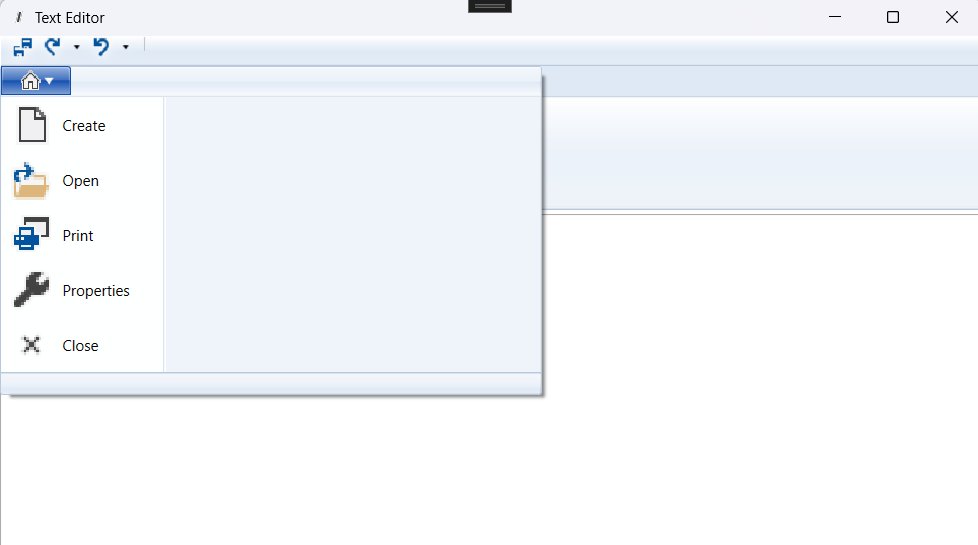
Обладнання: ПК, Visual Studio не нижче v.2010

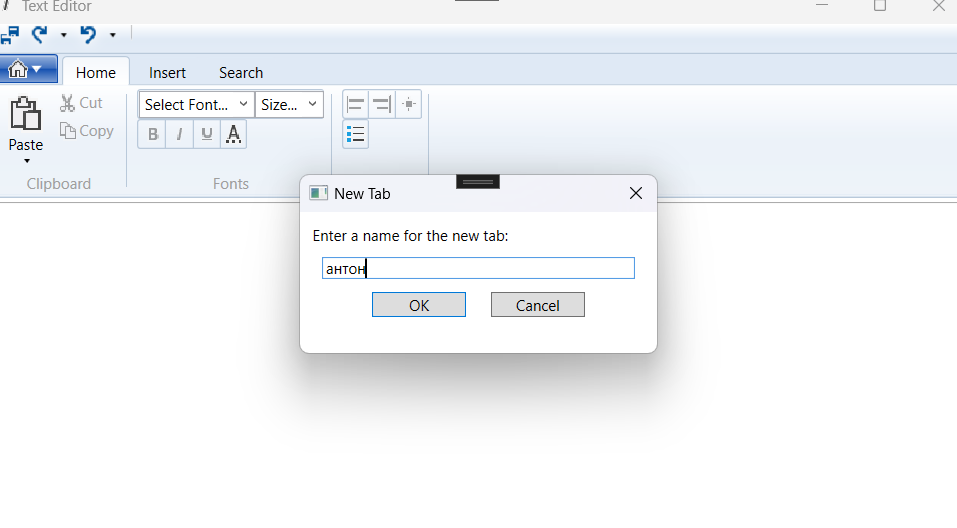


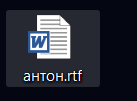


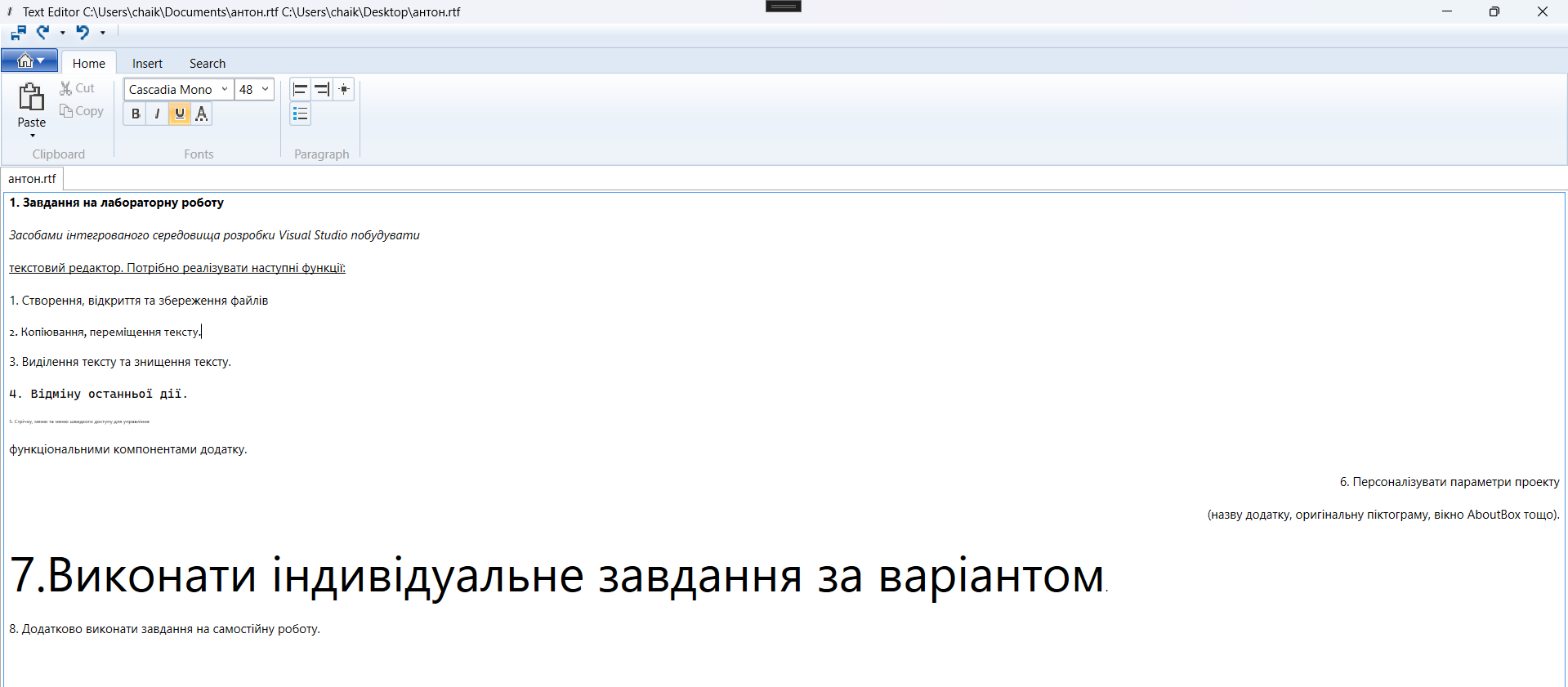


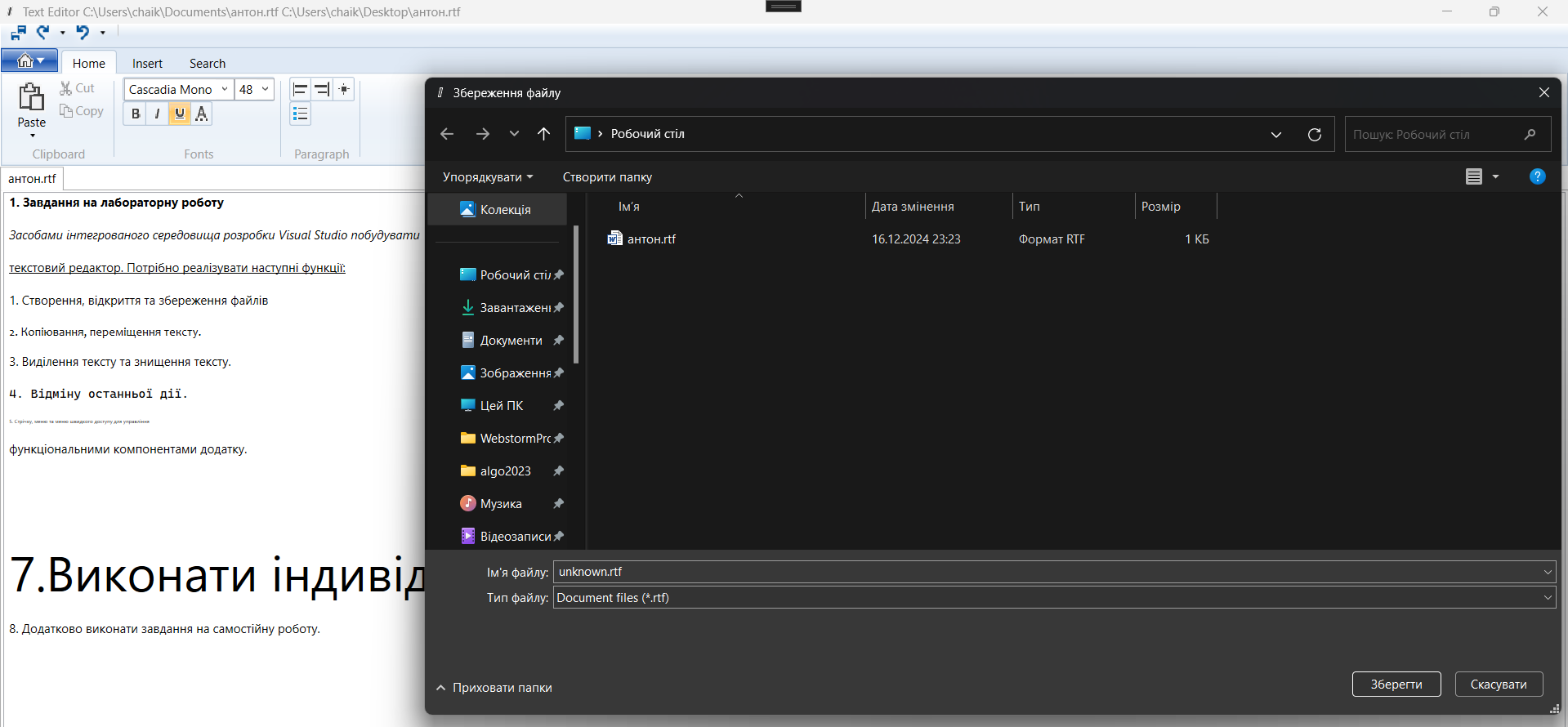




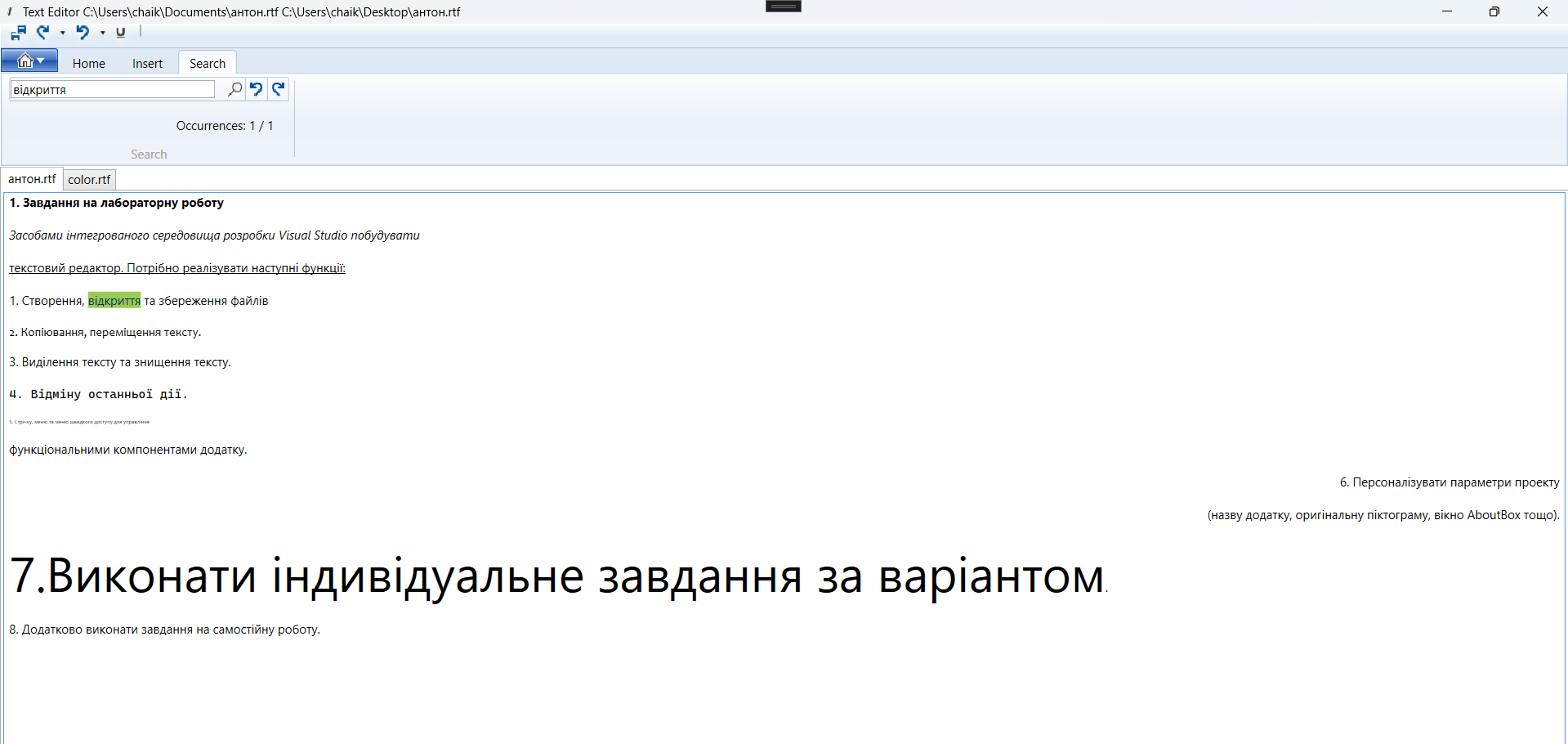












using Microsoft.Win32;

using System.DirectoryServices;

using System.IO;

using System.Text;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Documents;

using System.Windows.Media;

using static System.Net.Mime.MediaTypeNames;

namespace TextEditor;

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

\_fontSize.ItemsSource = FontSizes;

}

private List<TextRange> \_occurrences = new List<TextRange>();

private int \_currentOccurrenceIndex = -1;

public bool IsSaved = false;

public RichTextBox RichTextBox => TabControl.SelectedContent as RichTextBox;

public bool IsBald =>

RichTextBox.FontStyle == FontStyles.Oblique;

public bool IsItalic =>

RichTextBox.FontStyle == FontStyles.Italic;

public bool IsUnderline =>

RichTextBox.Selection.GetPropertyValue(Inline.TextDecorationsProperty) is TextDecorationCollection decorations &&

decorations.Contains(TextDecorations.Underline[0]);

public double[] FontSizes => [

3.0, 4.0, 5.0, 6.0, 6.5, 7.0, 7.5, 8.0, 8.5, 9.0, 9.5, 10.0, 10.5, 11.0, 11.5, 12.0, 12.5,

13.0,13.5,14.0, 15.0,16.0, 17.0, 18.0, 19.0, 20.0, 22.0, 24.0, 26.0, 28.0, 30.0,32.0, 34.0,

36.0, 38.0, 40.0, 44.0, 48.0, 52.0, 56.0, 60.0, 64.0, 68.0, 72.0, 76.0,80.0, 88.0, 96.0, 104.0,

112.0, 120.0, 128.0, 136.0, 144.0

];

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

{

string tabName = DialogBox.ShowDialog("Enter a name for the new tab:", "New Tab");

if (!string.IsNullOrEmpty(tabName))

{

TabItem newTab = new TabItem();

newTab.Header = tabName + ".rtf";

RichTextBox richTextBox = new RichTextBox();

newTab.Content = richTextBox;

TabControl.Items.Add(newTab);

TabControl.SelectedItem = newTab;

}

}

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

{

OpenFileDialog dlg = new OpenFileDialog();

dlg.Filter = "Document files (\*.rtf)|\*.rtf|Text files (\*.txt)|\*.txt";

var result = dlg.ShowDialog();

if (result == true)

{

var richTextBox = new RichTextBox

{

HorizontalScrollBarVisibility = ScrollBarVisibility.Auto

};

string fileName = dlg.FileName;

string fileContent = "";

using (FileStream file = new FileStream(fileName, FileMode.Open))

{

TextRange textRange = new TextRange(richTextBox.Document.ContentStart, richTextBox.Document.ContentEnd);

if (fileName.EndsWith(".rtf"))

{

textRange.Load(file, DataFormats.Rtf);

}

else

{

textRange.Load(file, DataFormats.Text);

}

fileContent = textRange.Text;

}

AddNewTab(richTextBox, fileName, fileContent);

}

}

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

{

if (RichTextBox is null) return;

SaveFileDialog savefile = new SaveFileDialog();

savefile.FileName = "unknown.rtf";

savefile.Filter = "Document files (\*.rtf)|\*.rtf";

if (savefile.ShowDialog() == true)

{

TextRange t = new TextRange(RichTextBox.Document.ContentStart, RichTextBox.Document.ContentEnd);

this.Title = this.Title + " " + savefile.FileName;

FileStream file = new FileStream(savefile.FileName, FileMode.Create);

t.Save(file, System.Windows.DataFormats.Rtf);

file.Close();

}

IsSaved = true;

}

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

{

if (TabControl.SelectedItem is TabItem selectedTab)

{

if (!IsSaved && MessageBox.Show("Do you want to save changes?", "Message", MessageBoxButton.YesNo) == MessageBoxResult.Yes)

{

btnSave\_Click(sender, e);

}

TabControl.Items.Remove(selectedTab);

}

}

void ApplyPropertyValueToSelectedText(DependencyProperty formattingProperty, object

value)

{

if (value == null)

return;

RichTextBox.Selection.ApplyPropertyValue(formattingProperty, value);

}

private void FontFamily\_SelectionChange(object sender, SelectionChangedEventArgs e)

{

try

{

FontFamily editValue = (FontFamily)e.AddedItems[0];

ApplyPropertyValueToSelectedText(TextElement.FontFamilyProperty, editValue);

}

catch (Exception) { }

}

private void FontSize\_SelectionChange(object sender, SelectionChangedEventArgs e)

{

try

{

ApplyPropertyValueToSelectedText(TextElement.FontSizeProperty, e.AddedItems[0]);

}

catch (Exception) { }

}

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

{

ClearAllTextHighlights();

if (RichTextBox is null) return;

string searchText = searchBox.Text;

\_occurrences.Clear();

\_currentOccurrenceIndex = -1;

OccurrenceStatusLabel.Content = $"Occurrences: {\_currentOccurrenceIndex + 1} / {\_occurrences.Count}";

if (!string.IsNullOrEmpty(searchText))

{

TextRange textRange = new TextRange(RichTextBox.Document.ContentStart, RichTextBox.Document.ContentEnd);

TextPointer start = textRange.Start;

ClearAllTextHighlights();

while (start != null && start.CompareTo(textRange.End) < 0)

{

TextPointer found = FindTextInRichTextBox(start, searchText);

if (found != null)

{

TextPointer end = found.GetPositionAtOffset(searchText.Length);

TextRange highlightRange = new TextRange(found, end);

highlightRange.ApplyPropertyValue(TextElement.BackgroundProperty, Brushes.Yellow);

\_occurrences.Add(highlightRange);

start = end;

}

else

{

break;

}

}

if (\_occurrences.Count > 0)

{

\_currentOccurrenceIndex = 0;

GoToOccurrence();

}

}

}

private TextPointer FindTextInRichTextBox(TextPointer position, string searchText)

{

while (position != null)

{

if (position.GetPointerContext(LogicalDirection.Forward) == TextPointerContext.Text)

{

string textRun = position.GetTextInRun(LogicalDirection.Forward);

int indexInRun = textRun.IndexOf(searchText, StringComparison.OrdinalIgnoreCase);

if (indexInRun >= 0)

{

return position.GetPositionAtOffset(indexInRun);

}

}

position = position.GetNextContextPosition(LogicalDirection.Forward);

}

return null;

}

private void ClearAllTextHighlights()

{

if (RichTextBox is null) return;

TextRange documentRange = new TextRange(RichTextBox.Document.ContentStart, RichTextBox.Document.ContentEnd);

documentRange.ApplyPropertyValue(TextElement.BackgroundProperty, Brushes.Transparent);

}

private void GoToOccurrence()

{

if (RichTextBox is null) return;

if (\_currentOccurrenceIndex < 0 || \_currentOccurrenceIndex >= \_occurrences.Count) return;

TextRange currentResult = \_occurrences[\_currentOccurrenceIndex];

currentResult.ApplyPropertyValue(TextElement.BackgroundProperty, Brushes.Yellow);

RichTextBox.Selection.Select(currentResult.Start, currentResult.End);

OccurrenceStatusLabel.Content = $"Occurrences: {\_currentOccurrenceIndex + 1} / {\_occurrences.Count}";

}

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

{

if (\_occurrences.Count > 0 && \_currentOccurrenceIndex < \_occurrences.Count - 1)

{

\_currentOccurrenceIndex++;

GoToOccurrence();

}

}

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

{

if (\_occurrences.Count > 0 && \_currentOccurrenceIndex > 0)

{

\_currentOccurrenceIndex--;

GoToOccurrence();

}

}

private void AddNewTab(RichTextBox richTextBox, string fileName, string fileContent = "")

{

TabItem newTab = new TabItem();

newTab.Header = Path.GetFileName(fileName);

if (!string.IsNullOrEmpty(fileContent))

{

TextRange textRange = new TextRange(richTextBox.Document.ContentStart, richTextBox.Document.ContentEnd);

textRange.Text = fileContent;

}

newTab.Content = richTextBox;

TabControl.Items.Add(newTab);

TabControl.SelectedItem = newTab;

}

private void TabCtrl\_OnSelectionChanged(object sender, RoutedEventArgs e)

{

\_occurrences.Clear();

\_currentOccurrenceIndex = -1;

OccurrenceStatusLabel.Content = $"Occurrences: {\_currentOccurrenceIndex + 1} / {\_occurrences.Count}";

}

}

Висновок ознайомилась основними підходами та технологіямиреалізації інтерфейсу універсалього додатку Windows