Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Кафедра цифрових технологій в енергетиці

ЗВІТ   
з виконання лабораторної роботи №7  
з дисципліни «Декларативне програмування графічних інтерфейсів»

«Розробка сторінкового/віконного WPF-застосунка»

Варіант 18

Виконала: студентка групи ТР-22 Хнюкало В.С.

Перевірив: доцент, к.ф.-м.н.  
Тарнавський Ю.А.

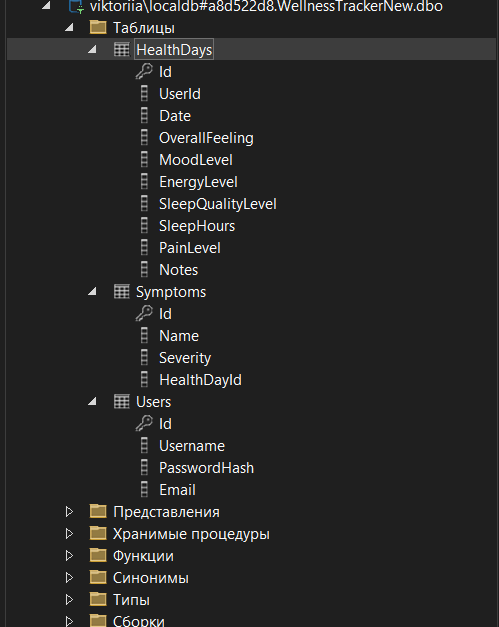
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

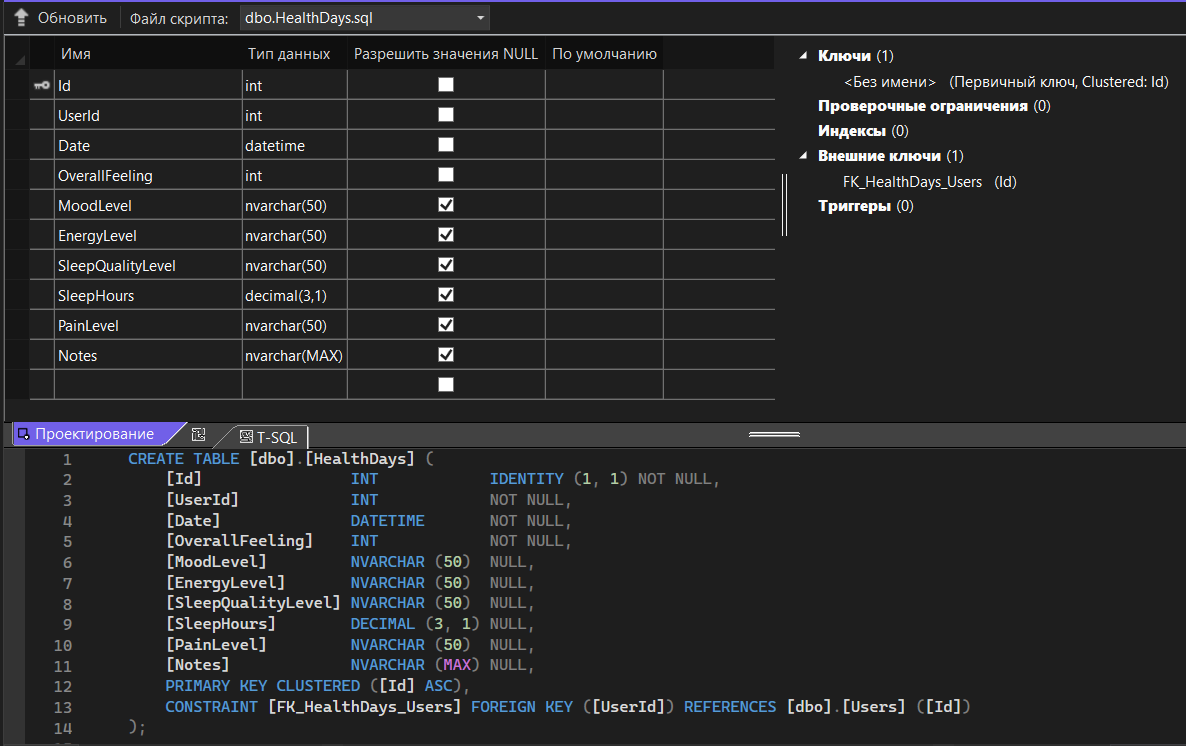
**Мета:** знайомитися з основами розробки сторінки WPF-застосунку, що використовує базу даних.

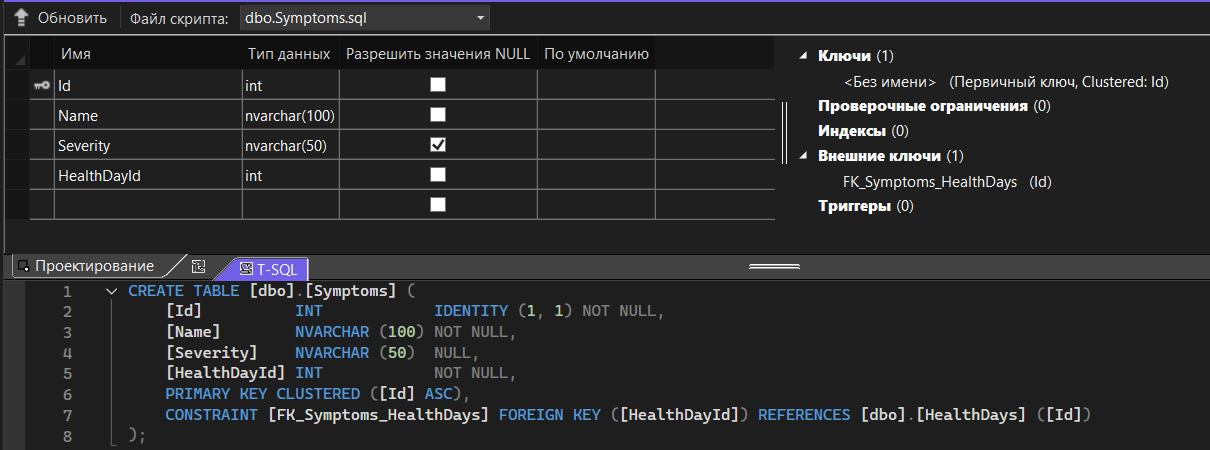
**Обрана тема:** Трекер здоров’я.

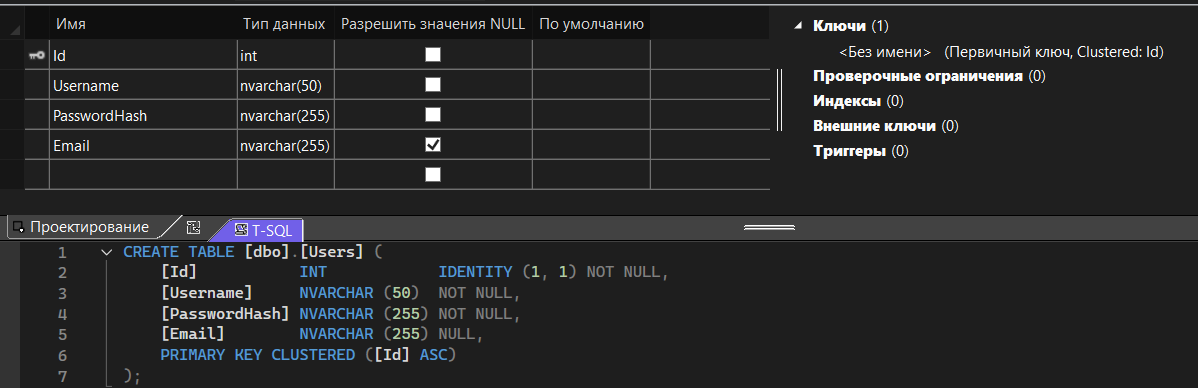
**Хід роботи**

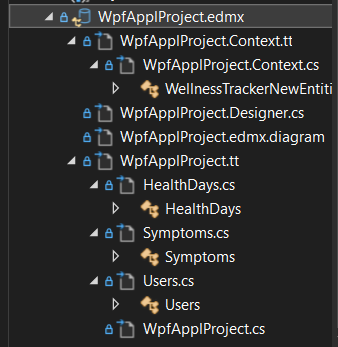
**БД**

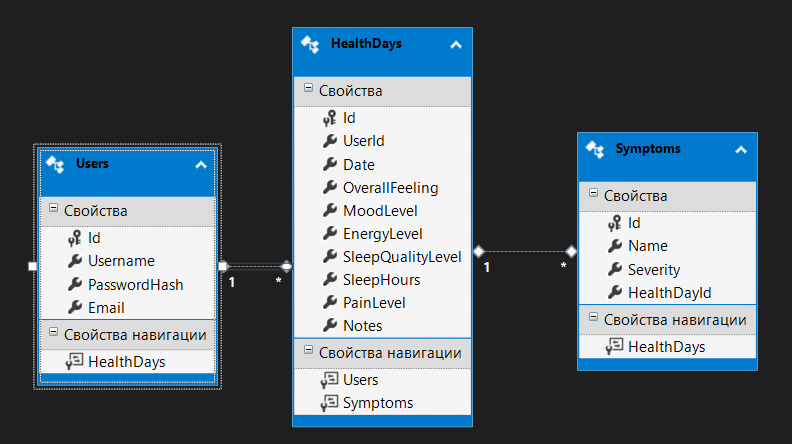
****

****

****

****

****

****

**Styles.xaml**

<ResourceDictionary xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">

<!-- Стиль заголовків таблиці -->

<Style x:Key="RoundedHeaderStyle" TargetType="DataGridColumnHeader">

<Setter Property="Background" Value="#C8E6C9"/>

<Setter Property="FontWeight" Value="SemiBold"/>

<Setter Property="FontSize" Value="14"/>

<Setter Property="HorizontalContentAlignment" Value="Center"/>

<Setter Property="Height" Value="35"/>

<Setter Property="Padding" Value="4"/>

<Setter Property="BorderThickness" Value="0"/>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="DataGridColumnHeader">

<Border Background="{TemplateBinding Background}"

CornerRadius="10"

SnapsToDevicePixels="True"

Padding="{TemplateBinding Padding}">

<ContentPresenter HorizontalAlignment="Center"

VerticalAlignment="Center"/>

</Border>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

<!-- Стиль кнопок -->

<Style TargetType="Button">

<Setter Property="FontSize" Value="14"/>

<Setter Property="FontWeight" Value="SemiBold"/>

<Setter Property="Foreground" Value="White"/>

<Setter Property="Padding" Value="8,4"/>

<Setter Property="Margin" Value="5"/>

<Setter Property="Cursor" Value="Hand"/>

<Setter Property="BorderThickness" Value="0"/>

<Setter Property="Background">

<Setter.Value>

<LinearGradientBrush StartPoint="0,0" EndPoint="0,1">

<GradientStop Color="#4CAF50" Offset="0"/>

<GradientStop Color="#388E3C" Offset="1"/>

</LinearGradientBrush>

</Setter.Value>

</Setter>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="Button">

<Border Background="{TemplateBinding Background}"

CornerRadius="8"

Padding="{TemplateBinding Padding}">

<ContentPresenter HorizontalAlignment="Center"

VerticalAlignment="Center"/>

</Border>

</ControlTemplate>

</Setter.Value>

</Setter>

<Style.Triggers>

<Trigger Property="IsMouseOver" Value="True">

<Setter Property="Background">

<Setter.Value>

<LinearGradientBrush StartPoint="0,0" EndPoint="0,1">

<GradientStop Color="#66BB6A" Offset="0"/>

<GradientStop Color="#43A047" Offset="1"/>

</LinearGradientBrush>

</Setter.Value>

</Setter>

<Setter Property="Effect">

<Setter.Value>

<DropShadowEffect Color="Black" BlurRadius="8" ShadowDepth="2"/>

</Setter.Value>

</Setter>

</Trigger>

</Style.Triggers>

</Style>

<!-- Стиль полів вводу (TextBox) -->

<Style TargetType="TextBox">

<Setter Property="FontSize" Value="14"/>

<Setter Property="Padding" Value="6"/>

<Setter Property="Margin" Value="5"/>

<Setter Property="BorderBrush" Value="#A5D6A7"/>

<Setter Property="BorderThickness" Value="1"/>

<Setter Property="Background" Value="White"/>

<Setter Property="Foreground" Value="Black"/>

<Setter Property="Height" Value="32"/>

<Setter Property="VerticalContentAlignment" Value="Center"/>

<Setter Property="SnapsToDevicePixels" Value="True"/>

</Style>

<!-- Стиль для ComboBoxItem (елементи списку) -->

<Style TargetType="ComboBoxItem">

<Setter Property="FontSize" Value="14"/>

<Setter Property="Padding" Value="8,4"/>

<Setter Property="Foreground" Value="Black"/>

<Setter Property="Background" Value="White"/>

<Setter Property="HorizontalContentAlignment" Value="Left"/>

<Setter Property="VerticalContentAlignment" Value="Center"/>

<Setter Property="SnapsToDevicePixels" Value="True"/>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="ComboBoxItem">

<Border x:Name="Border"

Background="{TemplateBinding Background}"

Padding="{TemplateBinding Padding}">

<ContentPresenter />

</Border>

<ControlTemplate.Triggers>

<Trigger Property="IsMouseOver" Value="True">

<Setter TargetName="Border" Property="Background" Value="#C8E6C9"/>

</Trigger>

<Trigger Property="IsSelected" Value="True">

<Setter TargetName="Border" Property="Background" Value="#A5D6A7"/>

<Setter Property="Foreground" Value="Black"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

<!-- Оновлений стиль ComboBox -->

<Style TargetType="ComboBox">

<Setter Property="FontSize" Value="14"/>

<Setter Property="Padding" Value="6,2"/>

<Setter Property="Margin" Value="5"/>

<Setter Property="Foreground" Value="Black"/>

<Setter Property="Background" Value="White"/>

<Setter Property="BorderBrush" Value="#A5D6A7"/>

<Setter Property="BorderThickness" Value="1"/>

<Setter Property="Height" Value="32"/>

<Setter Property="SnapsToDevicePixels" Value="True"/>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="ComboBox">

<Grid>

<!-- Кнопка розгортання -->

<ToggleButton Name="ToggleButton"

Background="{TemplateBinding Background}"

BorderBrush="{TemplateBinding BorderBrush}"

BorderThickness="{TemplateBinding BorderThickness}"

Focusable="False"

IsChecked="{Binding IsDropDownOpen, Mode=TwoWay, RelativeSource={RelativeSource TemplatedParent}}"

ClickMode="Press">

<Grid>

<!-- Вибраний текст -->

<ContentPresenter Margin="6,0,30,0"

VerticalAlignment="Center"

HorizontalAlignment="Left"

Content="{TemplateBinding SelectionBoxItem}"

ContentTemplate="{TemplateBinding SelectionBoxItemTemplate}"

ContentTemplateSelector="{TemplateBinding ItemTemplateSelector}" />

<!-- Стрілка -->

<Viewbox HorizontalAlignment="Right"

VerticalAlignment="Center"

Margin="0,0,10,0"

Width="12" Height="6">

<Canvas Width="12" Height="6">

<Path x:Name="Arrow"

Stroke="Black"

StrokeThickness="1.5"

Data="M 0 0 L 6 6 L 12 0" />

</Canvas>

</Viewbox>

</Grid>

</ToggleButton>

<!-- Випадаючий список -->

<Popup Name="Popup"

Placement="Bottom"

IsOpen="{TemplateBinding IsDropDownOpen}"

AllowsTransparency="True"

Focusable="False"

PopupAnimation="Slide"

Width="{Binding ActualWidth, RelativeSource={RelativeSource TemplatedParent}}">

<Border Background="White"

BorderBrush="#A5D6A7"

BorderThickness="1"

CornerRadius="8"

Padding="4"

MinWidth="{Binding ActualWidth, RelativeSource={RelativeSource TemplatedParent}}">

<ScrollViewer>

<StackPanel IsItemsHost="True" />

</ScrollViewer>

</Border>

</Popup>

</Grid>

<!-- Тригери для візуальної реакції -->

<ControlTemplate.Triggers>

<Trigger Property="IsMouseOver" Value="True">

<Setter TargetName="ToggleButton" Property="BorderBrush" Value="#66BB6A"/>

</Trigger>

<Trigger Property="IsFocused" Value="True">

<Setter TargetName="ToggleButton" Property="BorderBrush" Value="#388E3C"/>

</Trigger>

<Trigger Property="IsEnabled" Value="False">

<Setter TargetName="ToggleButton" Property="Opacity" Value="0.5"/>

</Trigger>

<Trigger Property="IsDropDownOpen" Value="True">

<Setter TargetName="Arrow" Property="Visibility" Value="Collapsed"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

<!-- Стиль міток -->

<Style TargetType="Label">

<Setter Property="FontWeight" Value="SemiBold"/>

<Setter Property="FontSize" Value="14"/>

<Setter Property="Margin" Value="5,10,5,2"/>

</Style>

<!-- Стиль заголовку -->

<Style x:Key="TitleText" TargetType="TextBlock">

<Setter Property="FontSize" Value="24"/>

<Setter Property="FontWeight" Value="Bold"/>

<Setter Property="TextAlignment" Value="Center"/>

<Setter Property="TextWrapping" Value="Wrap"/>

<Setter Property="Margin" Value="0,0,0,20"/>

</Style>

<!-- Ефект тіні -->

<DropShadowEffect x:Key="DefaultShadow"

Color="Black"

BlurRadius="6"

ShadowDepth="2"

Opacity="0.2"/>

<!-- Стиль комірок таблиці -->

<Style TargetType="DataGridCell">

<Setter Property="TextBlock.TextAlignment" Value="Center"/>

<Setter Property="HorizontalContentAlignment" Value="Center"/>

<Setter Property="VerticalContentAlignment" Value="Center"/>

</Style>

</ResourceDictionary>

**MainWindow**

**MainWindow.xaml**

<Window x:Class="WpfApplProject.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:WpfApplProject"

mc:Ignorable="d"

WindowStartupLocation="CenterScreen"

Title="eMood" Height="600" Width="850"

Icon="/Images/logo.png">

<Grid>

<Frame x:Name="frame1" Margin="3" NavigationUIVisibility="Hidden"/>

</Grid>

</Window>

**MainWindow.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 WpfApplProject

{

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

frame1.Navigate(new LoginPage()); // стартова сторінка

}

}

}

**LoginPage**

**LoginPage.xaml**

<Page x:Class="WpfApplProject.LoginPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="LoginPage">

<Page.Background>

<LinearGradientBrush StartPoint="0,0" EndPoint="0,1">

<GradientStop Color="#4CAF50" Offset="0"/>

<GradientStop Color="#A5D6A7" Offset="1"/>

</LinearGradientBrush>

</Page.Background>

<Grid>

<Border Width="400"

Padding="30"

Background="White"

CornerRadius="20"

BorderBrush="#C8E6C9"

BorderThickness="2"

HorizontalAlignment="Center"

VerticalAlignment="Center">

<StackPanel>

<!-- Заголовок -->

<TextBlock Text="Вхід до системи"

FontSize="28"

FontWeight="Bold"

Foreground="#2E7D32"

HorizontalAlignment="Center"

Margin="0,0,0,30"/>

<!-- Email -->

<TextBlock Text="Електронна пошта"

FontSize="14"

Foreground="Gray"

Margin="5,0,0,3"/>

<TextBox x:Name="emailBox"

Height="40"

FontSize="14"

Background="#EFFBEF"

BorderBrush="#C5E1A5"

BorderThickness="1"

Padding="8"

Margin="0,0,0,15"/>

<!-- Пароль -->

<TextBlock Text="Пароль"

FontSize="14"

Foreground="Gray"

Margin="5,0,0,3"/>

<PasswordBox x:Name="passwordBox"

Height="40"

FontSize="14"

Background="#EFFBEF"

BorderBrush="#C5E1A5"

BorderThickness="1"

Padding="8"

Margin="0,0,0,25"/>

<!-- Кнопки -->

<StackPanel Orientation="Horizontal" HorizontalAlignment="Center">

<!-- Стиль -->

<StackPanel.Resources>

<Style TargetType="Button">

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="Button">

<Border Background="{TemplateBinding Background}"

CornerRadius="10"

Padding="5"

BorderThickness="0">

<ContentPresenter HorizontalAlignment="Center"

VerticalAlignment="Center"/>

</Border>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

</StackPanel.Resources>

<Button Content="Увійти"

Width="120"

Height="40"

Margin="5"

Background="#66BB6A"

Foreground="White"

FontWeight="SemiBold"

Click="LoginButton\_Click"/>

<Button Content="Реєстрація"

Width="120"

Height="40"

Margin="5"

Background="#A5D6A7"

Foreground="#1B5E20"

FontWeight="SemiBold"

Click="GoToRegister\_Click"/>

</StackPanel>

</StackPanel>

</Border>

</Grid>

</Page>

**LoginPage.xaml.cs**

using System;

using System.Linq;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Navigation;

namespace WpfApplProject

{

public partial class LoginPage : Page

{

public LoginPage()

{

InitializeComponent();

}

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

{

string email = emailBox.Text.Trim();

string password = passwordBox.Password.Trim();

if (string.IsNullOrEmpty(email) || string.IsNullOrEmpty(password))

{

MessageBox.Show("Будь ласка, введіть логін та пароль.");

return;

}

using (var db = new WellnessTrackerNewEntities())

{

var user = db.Users.FirstOrDefault(u => u.Email == email);

if (user == null)

{

MessageBox.Show("Користувача не знайдено.");

return;

}

// Тут використовується перевірка через хеш

bool isPasswordCorrect = PasswordHelper.VerifyPassword(password, user.PasswordHash);

if (!isPasswordCorrect)

{

MessageBox.Show("Невірний пароль.");

return;

}

MessageBox.Show($"Вітаємо, {user.Username}!");

// Збереження ID користувача для інших сторінок

App.Current.Properties["UserId"] = user.Id;

NavigationService?.Navigate(new PageMain()); // Переход куди потрібно

}

}

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

{

NavigationService?.Navigate(new RegisterPage()); // Перехід до сторінки реєстрації

}

}

}

**RegisterPage**

**RegisterPage.xaml**

<Page x:Class="WpfApplProject.RegisterPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="RegisterPage">

<Page.Background>

<LinearGradientBrush StartPoint="0,0" EndPoint="0,1">

<GradientStop Color="#4CAF50" Offset="0"/>

<GradientStop Color="#A5D6A7" Offset="1"/>

</LinearGradientBrush>

</Page.Background>

<Grid>

<Border Width="450"

Padding="30"

Background="White"

CornerRadius="20"

BorderBrush="#C8E6C9"

BorderThickness="2"

HorizontalAlignment="Center"

VerticalAlignment="Center">

<StackPanel>

<!-- Заголовок -->

<TextBlock Text="Реєстрація"

FontSize="28"

FontWeight="Bold"

Foreground="#2E7D32"

HorizontalAlignment="Center"

Margin="0,0,0,30"/>

<!-- Ім'я користувача -->

<TextBlock Text="Ім’я користувача"

FontSize="14"

Foreground="Gray"

Margin="5,0,0,3"/>

<TextBox x:Name="usernameBox"

Height="40"

FontSize="14"

Background="#EFFBEF"

BorderBrush="#C5E1A5"

BorderThickness="1"

Padding="8"

Margin="0,0,0,15"/>

<!-- Email -->

<TextBlock Text="Email"

FontSize="14"

Foreground="Gray"

Margin="5,0,0,3"/>

<TextBox x:Name="emailBox"

Height="40"

FontSize="14"

Background="#EFFBEF"

BorderBrush="#C5E1A5"

BorderThickness="1"

Padding="8"

Margin="0,0,0,15"/>

<!-- Пароль -->

<TextBlock Text="Пароль"

FontSize="14"

Foreground="Gray"

Margin="5,0,0,3"/>

<PasswordBox x:Name="passwordBox"

Height="40"

FontSize="14"

Background="#EFFBEF"

BorderBrush="#C5E1A5"

BorderThickness="1"

Padding="8"

Margin="0,0,0,15"/>

<!-- Підтвердження пароля -->

<TextBlock Text="Підтвердження пароля"

FontSize="14"

Foreground="Gray"

Margin="5,0,0,3"/>

<PasswordBox x:Name="confirmPasswordBox"

Height="40"

FontSize="14"

Background="#EFFBEF"

BorderBrush="#C5E1A5"

BorderThickness="1"

Padding="8"

Margin="0,0,0,25"/>

<!-- Кнопки -->

<StackPanel Orientation="Vertical" HorizontalAlignment="Center">

<StackPanel.Resources>

<Style TargetType="Button">

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="Button">

<Border Background="{TemplateBinding Background}"

CornerRadius="10"

Padding="5"

BorderThickness="0">

<ContentPresenter HorizontalAlignment="Center"

VerticalAlignment="Center"/>

</Border>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

</StackPanel.Resources>

<Button Content="Зареєструватися"

Width="200"

Height="40"

Margin="5"

Background="#66BB6A"

Foreground="White"

FontWeight="SemiBold"

Click="RegisterButton\_Click"/>

<Button Content="Назад до входу"

Width="200"

Height="40"

Margin="5"

Background="#A5D6A7"

Foreground="#1B5E20"

FontWeight="SemiBold"

Click="BackButton\_Click"/>

</StackPanel>

</StackPanel>

</Border>

</Grid>

</Page>

**RegisterPage.xaml.cs**

using System;

using System.Linq;

using System.Windows;

using System.Windows.Controls;

namespace WpfApplProject

{

public partial class RegisterPage : Page

{

public RegisterPage()

{

InitializeComponent();

}

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

{

string username = usernameBox.Text.Trim();

string email = emailBox.Text.Trim();

string password = passwordBox.Password;

string confirmPassword = confirmPasswordBox.Password;

if (string.IsNullOrEmpty(username) || string.IsNullOrEmpty(email) || string.IsNullOrEmpty(password))

{

MessageBox.Show("Будь ласка, заповніть усі поля.");

return;

}

if (password != confirmPassword)

{

MessageBox.Show("Паролі не співпадають.");

return;

}

using (var db = new WellnessTrackerNewEntities())

{

if (db.Users.Any(u => u.Email == email))

{

MessageBox.Show("Користувач з таким email вже існує.");

return;

}

var newUser = new Users

{

Username = username,

Email = email,

PasswordHash = PasswordHelper.ComputeSha256Hash(password)

};

db.Users.Add(newUser);

db.SaveChanges();

}

MessageBox.Show("Реєстрація пройшла успішно!");

NavigationService?.Navigate(new LoginPage());

}

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

{

NavigationService?.Navigate(new LoginPage());

}

}

}

**PasswordHelper.cs**

using System.Security.Cryptography;

using System.Text;

public static class PasswordHelper

{

public static string ComputeSha256Hash(string rawData)

{

using (SHA256 sha256Hash = SHA256.Create())

{

byte[] bytes = sha256Hash.ComputeHash(Encoding.UTF8.GetBytes(rawData));

StringBuilder builder = new StringBuilder();

foreach (var t in bytes)

builder.Append(t.ToString("x2"));

return builder.ToString();

}

}

public static bool VerifyPassword(string inputPassword, string storedHash)

{

string inputHash = ComputeSha256Hash(inputPassword);

return inputHash == storedHash;

}

}

**PageMain**

**PageMain.xaml**

<Page x:Class="WpfApplProject.PageMain"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="Головна">

<!-- Градієнтний фон -->

<Page.Background>

<LinearGradientBrush StartPoint="0,0" EndPoint="0,1">

<GradientStop Color="#4CAF50" Offset="0"/>

<GradientStop Color="#A5D6A7" Offset="1"/>

</LinearGradientBrush>

</Page.Background>

<!-- Стилі -->

<Page.Resources>

<ResourceDictionary>

<ResourceDictionary.MergedDictionaries>

<ResourceDictionary Source="Styles.xaml"/>

</ResourceDictionary.MergedDictionaries>

</ResourceDictionary>

</Page.Resources>

<Grid>

<!-- Кнопка Вийти (в правому верхньому куті) -->

<Button Content="Вийти"

HorizontalAlignment="Right"

VerticalAlignment="Top"

Margin="0,10,10,0"

Background="#FE2E2E"

Foreground="White"

FontWeight="SemiBold"

BorderThickness="0"

Padding="10,5"

Click="Logout\_Click"/>

<!-- Центрований вміст -->

<StackPanel MaxWidth="400"

Margin="20"

HorizontalAlignment="Center"

VerticalAlignment="Center">

<TextBlock Text="Ласкаво просимо до застосунку!"

FontSize="26"

FontWeight="Bold"

TextAlignment="Center"

TextWrapping="Wrap"

Margin="0,0,0,30"

Foreground="#CEF6D8"/>

<Button Content="📄 Переглянути історію"

MaxWidth="250"

Margin="0,0,0,10"

Height="40"

Click="ViewHistory\_Click"/>

<Button Content="➕ Додати запис"

MaxWidth="250"

Height="40"

Click="AddEntry\_Click"/>

</StackPanel>

</Grid>

</Page>

**PageMain.xaml.cs**

using System.Windows;

using System.Windows.Controls;

namespace WpfApplProject

{

public partial class PageMain : Page

{

public PageMain()

{

InitializeComponent();

}

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

{

NavigationService?.Navigate(new PageTable1());

}

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

{

NavigationService?.Navigate(new PageTable2());

}

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

{

App.Current.Properties.Remove("UserId");

((MainWindow)Application.Current.MainWindow).frame1.Navigate(new LoginPage());

}

}

}

**PageTable1 (сторінка записів з командами)**

**PageTable1.xaml**

<Page x:Class="WpfApplProject.PageTable1"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:command="clr-namespace:WpfApplProject.Commands"

xmlns:local="clr-namespace:WpfApplProject"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="PageTable1">

<Page.CommandBindings>

<CommandBinding Command="command:DataCommands.Undo" Executed="UndoCommandBinding\_Executed" CanExecute="UndoCommandBinding\_CanExecute" />

<CommandBinding Command="command:DataCommands.New" Executed="NewCommandBinding\_Executed" CanExecute="NewCommandBinding\_CanExecute" />

<CommandBinding Command="command:DataCommands.Replace" Executed="ReplaceCommandBinding\_Executed" CanExecute="ReplaceCommandBinding\_CanExecute" />

<CommandBinding Command="command:DataCommands.Save" Executed="SaveCommandBinding\_Executed" CanExecute="SaveCommandBinding\_CanExecute" />

<CommandBinding Command="command:DataCommands.Find" Executed="FindCommandBinding\_Executed" CanExecute="FindCommandBinding\_CanExecute" />

<CommandBinding Command="command:DataCommands.Delete" Executed="DeleteCommandBinding\_Executed" CanExecute="DeleteCommandBinding\_CanExecute" />

<CommandBinding Command="command:DataCommands.Update" Executed="UpdateCommandBinding\_Executed" CanExecute="UpdateCommandBinding\_CanExecute" />

</Page.CommandBindings>

<Page.Resources>

<ResourceDictionary>

<ResourceDictionary.MergedDictionaries>

<ResourceDictionary Source="Styles.xaml"/>

</ResourceDictionary.MergedDictionaries>

</ResourceDictionary>

</Page.Resources>

<Page.Background>

<LinearGradientBrush StartPoint="0,0" EndPoint="0,1">

<GradientStop Color="White" Offset="0"/>

<GradientStop Color="#C8E6C9" Offset="1"/>

</LinearGradientBrush>

</Page.Background>

<DockPanel>

<!-- Меню -->

<Menu DockPanel.Dock="Top">

<MenuItem Header="Дія">

<MenuItem Header="Скасувати" Command="command:DataCommands.Undo"/>

<Separator/>

<MenuItem Header="Створити" Command="command:DataCommands.New"/>

<MenuItem Header="Редагувати" Command="command:DataCommands.Replace"/>

<Separator/>

<MenuItem Header="Зберегти" Command="command:DataCommands.Save"/>

<MenuItem Header="Знайти" Command="command:DataCommands.Find"/>

<MenuItem Header="Видалити" Command="command:DataCommands.Delete"/>

<MenuItem Header="Оновити" Command="command:DataCommands.Update"/>

</MenuItem>

<MenuItem Header="Звіт"/>

</Menu>

<!-- Панель інструментів -->

<ToolBarTray DockPanel.Dock="Top">

<ToolBar>

<Button ToolTip="Назад" Click="BackButton\_Click" >

<Image Source="Images/Back.png" Width="24" Height="24"/>

</Button>

<Button ToolTip="Скасувати" Command="command:DataCommands.Undo">

<Image Source="Images/Undo.png" Width="24" Height="24"/>

</Button>

<Button ToolTip="Створити" Command="command:DataCommands.New">

<Image Source="Images/New.png" Width="30" Height="30"/>

</Button>

<Button ToolTip="Редагувати" Command="command:DataCommands.Replace">

<Image Source="Images/Edit.png" Width="30" Height="30"/>

</Button>

<Button ToolTip="Зберегти" Command="command:DataCommands.Save">

<Image Source="Images/Save.png" Width="30" Height="30"/>

</Button>

<Button ToolTip="Знайти" Command="command:DataCommands.Find">

<Image Source="Images/Find.png" Width="30" Height="30"/>

</Button>

<Button ToolTip="Видалити" Command="command:DataCommands.Delete">

<Image Source="Images/Delete.png" Width="30" Height="30"/>

</Button>

<Button ToolTip="Оновити" Command="command:DataCommands.Update">

<Image Source="Images/Update.png" Width="30" Height="30"/>

</Button>

</ToolBar>

</ToolBarTray>

<!-- Таблиця історії -->

<Grid Margin="10">

<DataGrid x:Name="historyGrid"

AutoGenerateColumns="False"

CanUserAddRows="False"

IsReadOnly="True"

HeadersVisibility="Column"

RowHeight="30"

FontSize="14"

AlternatingRowBackground="#F1F8E9"

GridLinesVisibility="Horizontal"

ColumnHeaderHeight="35"

ColumnHeaderStyle="{StaticResource RoundedHeaderStyle}">

<DataGrid.Columns>

<DataGridTextColumn Header="Дата" Binding="{Binding Date, StringFormat={}{0:dd.MM.yyyy HH:mm}}" Width="\*" />

<DataGridTextColumn Header="Самопочуття" Binding="{Binding OverallFeeling}" Width="Auto"/>

<DataGridTextColumn Header="Настрій" Binding="{Binding MoodLevel}" Width="Auto"/>

<DataGridTextColumn Header="Енергія" Binding="{Binding EnergyLevel}" Width="Auto"/>

<DataGridTextColumn Header="Якість сну" Binding="{Binding SleepQualityLevel}" Width="Auto"/>

<DataGridTextColumn Header="Годин сну" Binding="{Binding SleepHours}" Width="Auto"/>

<DataGridTextColumn Header="Рівень болю" Binding="{Binding PainLevel}" Width="Auto"/>

<DataGridTextColumn Header="Симптоми" Binding="{Binding SymptomSummary}" Width="2\*"/>

<DataGridTextColumn Header="Нотатки" Binding="{Binding Notes}" Width="2\*"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</DockPanel>

</Page>

**PageTable1.xaml.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Input;

using Microsoft.VisualBasic;

namespace WpfApplProject

{

public partial class PageTable1 : Page

{

private bool isDirty = true;

public PageTable1()

{

InitializeComponent();

LoadHealthHistory();

}

private void LoadHealthHistory()

{

try

{

int userId = (int)App.Current.Properties["UserId"];

using (var db = new WellnessTrackerNewEntities())

{

var data = db.HealthDays

.Where(h => h.UserId == userId)

.OrderByDescending(h => h.Date)

.ToList()

.Select(h => new

{

h.Id,

h.Date,

h.OverallFeeling,

h.MoodLevel,

h.EnergyLevel,

h.SleepQualityLevel,

h.SleepHours,

h.PainLevel,

SymptomSummary = string.Join(", ", h.Symptoms.Select(s => $"{s.Name} ({s.Severity})")),

h.Notes

})

.ToList();

historyGrid.ItemsSource = data;

}

}

catch (Exception ex)

{

MessageBox.Show("Помилка завантаження даних: " + ex.Message);

}

}

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

{

((MainWindow)Application.Current.MainWindow).frame1.Navigate(new PageMain());

}

private void UndoCommandBinding\_CanExecute(object sender, CanExecuteRoutedEventArgs e)

{

e.CanExecute = historyGrid != null && historyGrid.SelectedItem != null;

}

private void UndoCommandBinding\_Executed(object sender, ExecutedRoutedEventArgs e)

{

historyGrid.SelectedItem = null;

}

private void NewCommandBinding\_CanExecute(object sender, CanExecuteRoutedEventArgs e) => e.CanExecute = true;

private void NewCommandBinding\_Executed(object sender, ExecutedRoutedEventArgs e)

{

((MainWindow)Application.Current.MainWindow).frame1.Navigate(new PageTable2());

}

private void ReplaceCommandBinding\_CanExecute(object sender, CanExecuteRoutedEventArgs e) => e.CanExecute = historyGrid != null && historyGrid.SelectedItem != null;

private void ReplaceCommandBinding\_Executed(object sender, ExecutedRoutedEventArgs e)

{

if (historyGrid.SelectedItem != null)

{

dynamic selected = historyGrid.SelectedItem;

App.Current.Properties["EditEntryId"] = selected.Id; // передача ID

((MainWindow)Application.Current.MainWindow).frame1.Navigate(new PageTable2());

}

}

private void SaveCommandBinding\_CanExecute(object sender, CanExecuteRoutedEventArgs e) => e.CanExecute = isDirty;

private void SaveCommandBinding\_Executed(object sender, ExecutedRoutedEventArgs e)

{

try

{

var data = historyGrid.ItemsSource;

if (data == null)

{

MessageBox.Show("Немає даних для збереження.");

return;

}

// Вибір місця збереження

Microsoft.Win32.SaveFileDialog saveFileDialog = new Microsoft.Win32.SaveFileDialog

{

FileName = "HealthHistory",

DefaultExt = ".csv",

Filter = "CSV файли (\*.csv)|\*.csv|Текстові файли (\*.txt)|\*.txt|Усі файли (\*.\*)|\*.\*"

};

if (saveFileDialog.ShowDialog() == true)

{

using (var writer = new System.IO.StreamWriter(saveFileDialog.FileName, false, System.Text.Encoding.UTF8))

{

// заголовок

writer.WriteLine("Дата,Самопочуття,Настрій,Енергія,Якість сну,Годин сну,Рівень болю,Симптоми,Нотатки");

foreach (var item in data)

{

dynamic row = item;

string Escape(string s) => s?.Replace("\"", "\"\"") ?? "";

string line = $"{row.Date:yyyy-MM-dd HH:mm},{row.OverallFeeling},{Escape(row.MoodLevel)},{Escape(row.EnergyLevel)}," +

$"{Escape(row.SleepQualityLevel)},{row.SleepHours},{Escape(row.PainLevel)}," +

$"\"{Escape(row.SymptomSummary)}\",\"{Escape(row.Notes)}\"";

writer.WriteLine(line);

}

}

MessageBox.Show("Дані успішно збережено!");

isDirty = false;

}

}

catch (Exception ex)

{

MessageBox.Show($"Помилка під час збереження: {ex.Message}");

}

}

private void FindCommandBinding\_CanExecute(object sender, CanExecuteRoutedEventArgs e) => e.CanExecute = true;

private void FindCommandBinding\_Executed(object sender, ExecutedRoutedEventArgs e)

{

string keyword = Interaction.InputBox("Введіть ключове слово для пошуку", "Пошук", "");

if (!string.IsNullOrWhiteSpace(keyword))

{

var collectionView = CollectionViewSource.GetDefaultView(historyGrid.ItemsSource);

if (collectionView != null)

{

collectionView.Filter = item =>

{

var obj = item.ToString();

return obj != null && obj.ToLower().Contains(keyword.ToLower());

};

}

}

}

private void DeleteCommandBinding\_CanExecute(object sender, CanExecuteRoutedEventArgs e) => e.CanExecute = isDirty;

private void DeleteCommandBinding\_Executed(object sender, ExecutedRoutedEventArgs e)

{

var result = MessageBox.Show("Видалити всі записи? Натисніть 'Ні' для видалення лише виділеного.", "Підтвердження", MessageBoxButton.YesNoCancel);

using (var db = new WellnessTrackerNewEntities())

{

int userId = (int)App.Current.Properties["UserId"];

if (result == MessageBoxResult.Yes)

{

var all = db.HealthDays.Where(h => h.UserId == userId).ToList();

foreach (var hd in all)

{

db.Symptoms.RemoveRange(hd.Symptoms); // видаляємо пов’язані симптоми

}

db.HealthDays.RemoveRange(all);

db.SaveChanges();

LoadHealthHistory();

MessageBox.Show("Усі записи видалено.");

}

else if (result == MessageBoxResult.No && historyGrid.SelectedItem != null)

{

dynamic selected = historyGrid.SelectedItem;

int idToDelete = selected.Id;

var toDelete = db.HealthDays.Include("Symptoms").FirstOrDefault(h => h.Id == idToDelete);

if (toDelete != null)

{

db.Symptoms.RemoveRange(toDelete.Symptoms); // видаляємо пов’язані симптоми

db.HealthDays.Remove(toDelete);

db.SaveChanges();

LoadHealthHistory();

MessageBox.Show("Запис видалено.");

}

}

}

}

private void UpdateCommandBinding\_CanExecute(object sender, CanExecuteRoutedEventArgs e) => e.CanExecute = true;

private void UpdateCommandBinding\_Executed(object sender, ExecutedRoutedEventArgs e)

{

LoadHealthHistory();

MessageBox.Show("Дані оновлено");

}

}

}

**DataCommands.сs**

using System.Windows.Input;

using static System.Data.Entity.Infrastructure.Design.Executor;

namespace WpfApplProject.Commands

{

public static class DataCommands

{

public static RoutedCommand Undo { get; set; }

public static RoutedCommand New { get; set; }

public static RoutedCommand Replace { get; set; }

public static RoutedCommand Save { get; set; }

public static RoutedCommand Find { get; set; }

public static RoutedCommand Delete { get; set; }

public static RoutedCommand Update { get; set; }

static DataCommands()

{

// Undo — Ctrl+Z

InputGestureCollection undoGestures = new InputGestureCollection();

undoGestures.Add(new KeyGesture(Key.Z, ModifierKeys.Control, "Ctrl+Z"));

Undo = new RoutedCommand("Undo", typeof(DataCommands), undoGestures);

// New — Ctrl+N

InputGestureCollection newGestures = new InputGestureCollection();

newGestures.Add(new KeyGesture(Key.N, ModifierKeys.Control, "Ctrl+N"));

New = new RoutedCommand("New", typeof(DataCommands), newGestures);

// Replace — Ctrl+R

InputGestureCollection replaceGestures = new InputGestureCollection();

replaceGestures.Add(new KeyGesture(Key.R, ModifierKeys.Control, "Ctrl+R"));

Replace = new RoutedCommand("Replace", typeof(DataCommands), replaceGestures);

// Save — Ctrl+S

InputGestureCollection saveGestures = new InputGestureCollection();

saveGestures.Add(new KeyGesture(Key.S, ModifierKeys.Control, "Ctrl+S"));

Save = new RoutedCommand("Save", typeof(DataCommands), saveGestures);

// Find — Ctrl+F

InputGestureCollection findGestures = new InputGestureCollection();

findGestures.Add(new KeyGesture(Key.F, ModifierKeys.Control, "Ctrl+F"));

Find = new RoutedCommand("Find", typeof(DataCommands), findGestures);

// Delete — Ctrl+D

InputGestureCollection deleteGestures = new InputGestureCollection();

deleteGestures.Add(new KeyGesture(Key.D, ModifierKeys.Control, "Ctrl+D"));

Delete = new RoutedCommand("Delete", typeof(DataCommands), deleteGestures);

// Update — Ctrl+U

InputGestureCollection updateGestures = new InputGestureCollection();

updateGestures.Add(new KeyGesture(Key.U, ModifierKeys.Control, "Ctrl+U"));

Update = new RoutedCommand("Update", typeof(DataCommands), updateGestures);

}

}

}

**PageTable2 (додати запис)**

**PageTable2.xaml**

<Page x:Class="WpfApplProject.PageTable2"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:sys="clr-namespace:System;assembly=mscorlib"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="PageTable2">

<!-- Градієнтний фон -->

<Page.Background>

<LinearGradientBrush StartPoint="0,0" EndPoint="0,1">

<GradientStop Color="White" Offset="0"/>

<GradientStop Color="#C8E6C9" Offset="1"/>

</LinearGradientBrush>

</Page.Background>

<!-- Підключення стилів -->

<Page.Resources>

<ResourceDictionary>

<ResourceDictionary.MergedDictionaries>

<ResourceDictionary Source="Styles.xaml"/>

</ResourceDictionary.MergedDictionaries>

</ResourceDictionary>

</Page.Resources>

<!-- Основний вміст -->

<ScrollViewer VerticalScrollBarVisibility="Auto">

<StackPanel Margin="20">

<!-- Назад -->

<Button ToolTip="Назад"

Click="BackButton\_Click"

Background="Transparent"

BorderThickness="0"

Padding="0"

HorizontalAlignment="Left">

<Image Source="Images/Back.png" Width="35" Height="35"/>

</Button>

<!-- Дата -->

<Label Content="Дата:"/>

<DatePicker x:Name="datePicker" SelectedDate="{x:Static sys:DateTime.Today}" Margin="0,0,0,10"/>

<!-- Загальне самопочуття з зірочками -->

<StackPanel Margin="0,10">

<TextBlock Text="Загальне самопочуття:" FontWeight="Bold"/>

<StackPanel Orientation="Horizontal" VerticalAlignment="Center">

<ItemsControl x:Name="starRating">

<ItemsControl.ItemsPanel>

<ItemsPanelTemplate>

<StackPanel Orientation="Horizontal"/>

</ItemsPanelTemplate>

</ItemsControl.ItemsPanel>

</ItemsControl>

<TextBlock x:Name="ratingLabel" Margin="10,0,0,0" VerticalAlignment="Center"/>

</StackPanel>

</StackPanel>

<!-- Настрій -->

<Label Content="Настрій:"/>

<ComboBox x:Name="moodLevelBox" Margin="0,0,0,10">

<ComboBoxItem Content="Чудовий"/>

<ComboBoxItem Content="Гарний"/>

<ComboBoxItem Content="Нейтральний"/>

<ComboBoxItem Content="Поганий"/>

<ComboBoxItem Content="Дуже поганий"/>

</ComboBox>

<!-- Енергія -->

<Label Content="Енергія:"/>

<ComboBox x:Name="energyLevelBox" Margin="0,0,0,10">

<ComboBoxItem Content="Повний енергії"/>

<ComboBoxItem Content="Нормальний"/>

<ComboBoxItem Content="Втомлений"/>

<ComboBoxItem Content="Дуже виснажений"/>

</ComboBox>

<!-- Якість сну -->

<Label Content="Якість сну:"/>

<ComboBox x:Name="sleepQualityLevelBox" Margin="0,0,0,10">

<ComboBoxItem Content="Глибокий сон"/>

<ComboBoxItem Content="Нормальний сон"/>

<ComboBoxItem Content="Переривчастий сон"/>

<ComboBoxItem Content="Поганий сон"/>

</ComboBox>

<!-- Години сну -->

<Label Content="Години сну:"/>

<TextBox x:Name="sleepHoursBox" Width="100" Margin="0,0,0,10"/>

<!-- Рівень болю -->

<Label Content="Рівень болю:"/>

<ComboBox x:Name="painLevelBox" Margin="0,0,0,10">

<ComboBoxItem Content="Немає болю"/>

<ComboBoxItem Content="Слабкий"/>

<ComboBoxItem Content="Помірний"/>

<ComboBoxItem Content="Відчутний"/>

<ComboBoxItem Content="Сильний"/>

<ComboBoxItem Content="Дуже сильний"/>

</ComboBox>

<!-- Нотатки -->

<Label Content="Нотатки:"/>

<TextBox x:Name="notesBox" Height="60" TextWrapping="Wrap" AcceptsReturn="True" Margin="0,0,0,10"/>

<!-- Додати симптоми -->

<Label Content="Симптоми:"/>

<StackPanel Orientation="Horizontal">

<TextBox x:Name="symptomNameBox" Width="200" Margin="0,0,10,0"/>

<ComboBox x:Name="symptomSeverityBox" Width="150">

<ComboBoxItem Content="Дуже легка"/>

<ComboBoxItem Content="Легка"/>

<ComboBoxItem Content="Помірна"/>

<ComboBoxItem Content="Сильна"/>

<ComboBoxItem Content="Дуже сильна"/>

</ComboBox>

<Button Content="Додати" Click="AddSymptom\_Click" Margin="10,0,0,0"/>

</StackPanel>

<!-- Таблиця симптомів -->

<DataGrid x:Name="symptomsGrid"

AutoGenerateColumns="False"

CanUserAddRows="False"

IsReadOnly="True"

Margin="0,10,0,10"

RowBackground="White"

AlternatingRowBackground="#F1F8E9"

HeadersVisibility="Column"

BorderThickness="0"

ColumnHeaderStyle="{StaticResource RoundedHeaderStyle}">

<DataGrid.Columns>

<DataGridTextColumn Header="Назва" Binding="{Binding Name}" Width="\*" />

<DataGridTextColumn Header="Інтенсивність" Binding="{Binding Severity}" Width="Auto" />

</DataGrid.Columns>

</DataGrid>

<!-- Зберегти запис -->

<Button Content="Зберегти запис" HorizontalAlignment="Center" Click="SaveHealthDay\_Click" Margin="0,20,0,0"/>

</StackPanel>

</ScrollViewer>

</Page>

**PageTable2.xaml.cs**

using System;

using System.Collections.ObjectModel;

using System.Linq;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

namespace WpfApplProject

{

public partial class PageTable2 : Page

{

private ObservableCollection<Symptoms> symptomsList = new ObservableCollection<Symptoms>();

private double currentRating = 2.5;

private int? editingId = null;

public PageTable2()

{

InitializeComponent();

symptomsGrid.ItemsSource = symptomsList;

datePicker.SelectedDate = DateTime.Now;

InitializeStarRating();

TryLoadForEdit();

}

private void TryLoadForEdit()

{

if (App.Current.Properties.Contains("EditEntryId"))

{

editingId = (int)App.Current.Properties["EditEntryId"];

using (var db = new WellnessTrackerNewEntities())

{

var health = db.HealthDays.Find(editingId);

if (health != null)

{

datePicker.SelectedDate = health.Date;

currentRating = health.OverallFeeling / 2.0;

InitializeStarRating();

moodLevelBox.Text = health.MoodLevel;

energyLevelBox.Text = health.EnergyLevel;

sleepQualityLevelBox.Text = health.SleepQualityLevel;

sleepHoursBox.Text = health.SleepHours?.ToString();

painLevelBox.Text = health.PainLevel;

notesBox.Text = health.Notes;

symptomsList.Clear();

foreach (var s in health.Symptoms)

{

symptomsList.Add(new Symptoms { Name = s.Name, Severity = s.Severity });

}

}

}

}

}

private void InitializeStarRating()

{

starRating.Items.Clear();

for (int i = 1; i <= 5; i++)

{

double value = i;

Image starImage = new Image

{

Width = 32,

Height = 32,

Margin = new Thickness(2),

Source = GetStarImageSource(i, currentRating)

};

Button starButton = new Button

{

Content = starImage,

Tag = value,

Padding = new Thickness(0),

BorderThickness = new Thickness(0),

Background = Brushes.Transparent,

Cursor = Cursors.Hand

};

starButton.Click += (s, e) =>

{

Point position = Mouse.GetPosition(starButton);

currentRating = position.X < starButton.ActualWidth / 2 ? value - 0.5 : value;

InitializeStarRating();

};

starRating.Items.Add(starButton);

}

ratingLabel.Text = $"{currentRating \* 2:0.0} / 10";

}

private BitmapImage GetStarImageSource(int position, double rating)

{

string imageName;

if (rating >= position)

imageName = "star\_full.png";

else if (rating >= position - 0.5)

imageName = "star\_half.png";

else

imageName = "star\_empty.png";

return new BitmapImage(new Uri($"/Images/{imageName}", UriKind.Relative));

}

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

{

if (!string.IsNullOrWhiteSpace(symptomNameBox.Text) && symptomSeverityBox.SelectedItem is ComboBoxItem selectedSeverity)

{

symptomsList.Add(new Symptoms

{

Name = symptomNameBox.Text,

Severity = selectedSeverity.Content.ToString()

});

symptomNameBox.Clear();

symptomSeverityBox.SelectedIndex = -1;

}

else

{

MessageBox.Show("Будь ласка, введіть назву симптому та виберіть інтенсивність.");

}

}

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

{

if (datePicker.SelectedDate == null)

{

MessageBox.Show("Будь ласка, оберіть дату.");

return;

}

using (var db = new WellnessTrackerNewEntities())

{

try

{

int userId = (int)App.Current.Properties["UserId"];

HealthDays healthDay;

if (editingId.HasValue)

{

healthDay = db.HealthDays.Include("Symptoms").FirstOrDefault(h => h.Id == editingId.Value);

db.Symptoms.RemoveRange(healthDay.Symptoms);

}

else

{

healthDay = new HealthDays();

db.HealthDays.Add(healthDay);

}

healthDay.UserId = userId;

healthDay.Date = datePicker.SelectedDate.Value.Date + DateTime.Now.TimeOfDay;

healthDay.OverallFeeling = (int)Math.Round(currentRating \* 2);

healthDay.MoodLevel = (moodLevelBox.SelectedItem as ComboBoxItem)?.Content.ToString();

healthDay.EnergyLevel = (energyLevelBox.SelectedItem as ComboBoxItem)?.Content.ToString();

healthDay.SleepQualityLevel = (sleepQualityLevelBox.SelectedItem as ComboBoxItem)?.Content.ToString();

healthDay.SleepHours = decimal.TryParse(sleepHoursBox.Text, out decimal hours) ? (decimal?)hours : null;

healthDay.PainLevel = (painLevelBox.SelectedItem as ComboBoxItem)?.Content.ToString();

healthDay.Notes = notesBox.Text;

foreach (var symptom in symptomsList)

{

symptom.HealthDays = healthDay;

db.Symptoms.Add(symptom);

}

db.SaveChanges();

MessageBox.Show("Запис успішно збережено!");

ClearForm();

App.Current.Properties.Remove("EditEntryId");

}

catch (Exception ex)

{

MessageBox.Show($"Помилка при збереженні: {ex.Message}\n{ex.InnerException}");

}

}

}

private void ClearForm()

{

datePicker.SelectedDate = DateTime.Now;

currentRating = 2.5;

InitializeStarRating();

moodLevelBox.SelectedIndex = -1;

energyLevelBox.SelectedIndex = -1;

sleepQualityLevelBox.SelectedIndex = -1;

sleepHoursBox.Clear();

painLevelBox.SelectedIndex = -1;

notesBox.Clear();

symptomsList.Clear();

editingId = null;

}

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

{

App.Current.Properties.Remove("EditEntryId");

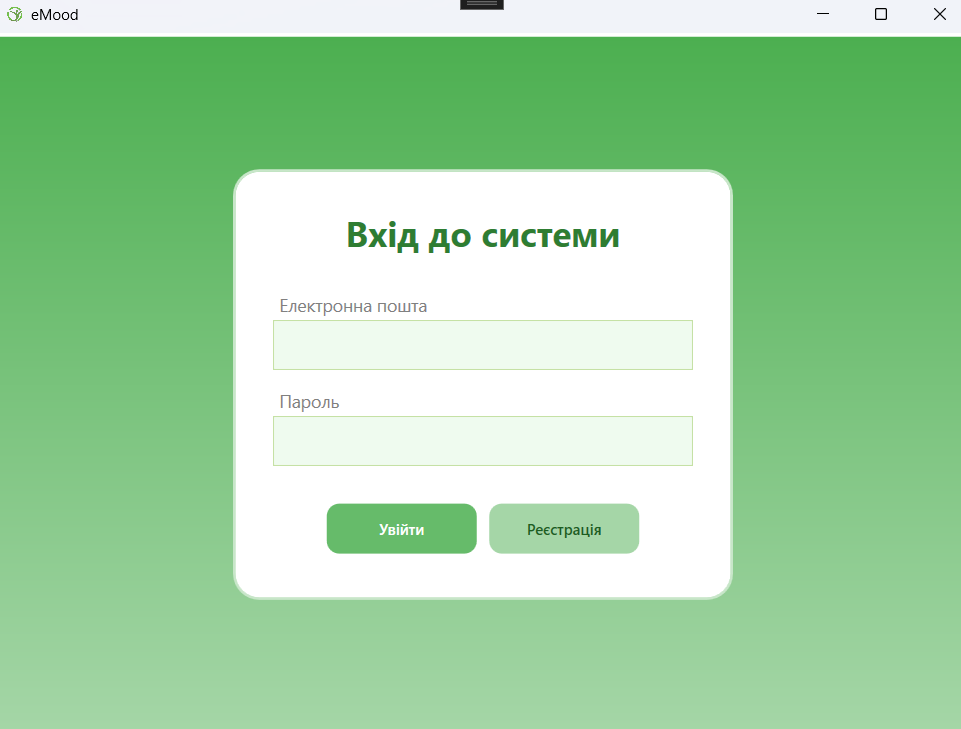
NavigationService?.GoBack();

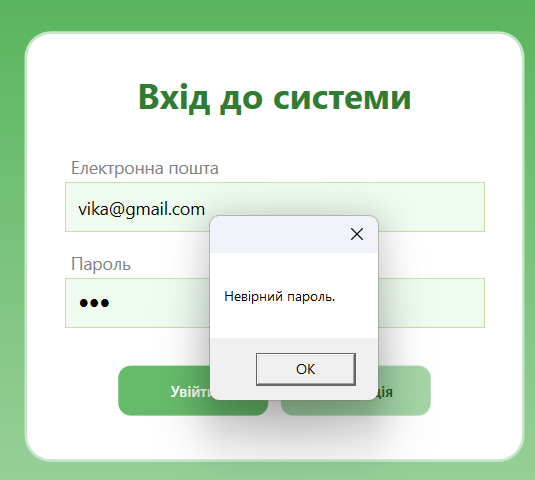
}

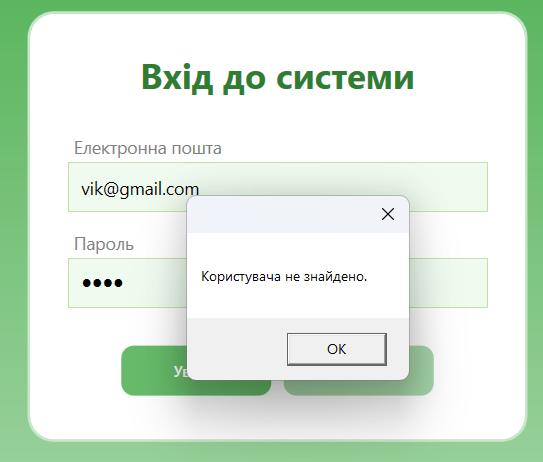
}

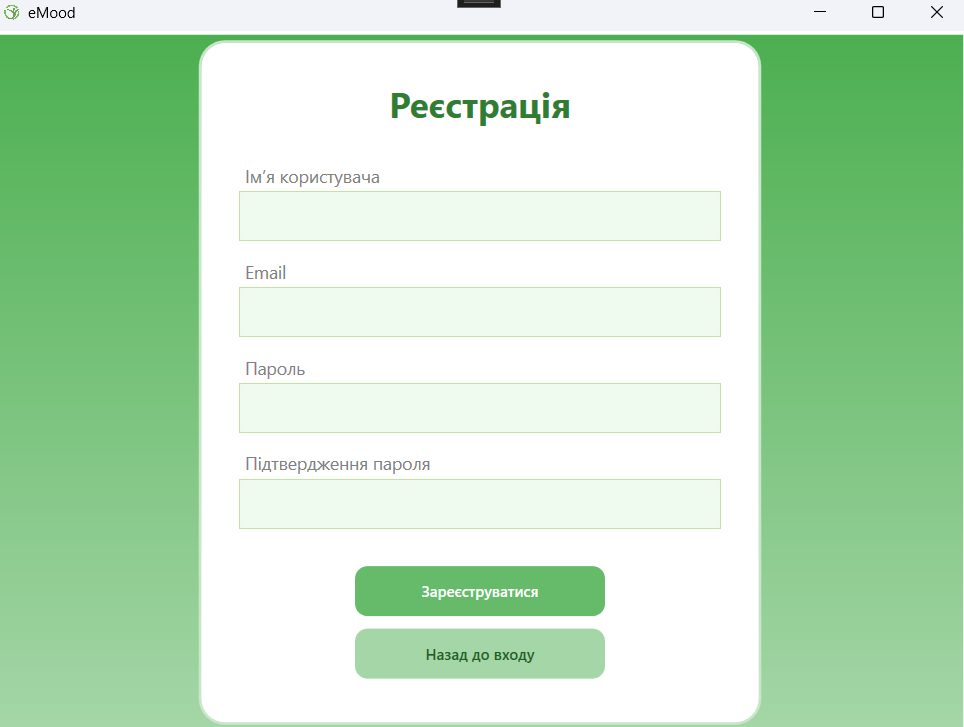
}

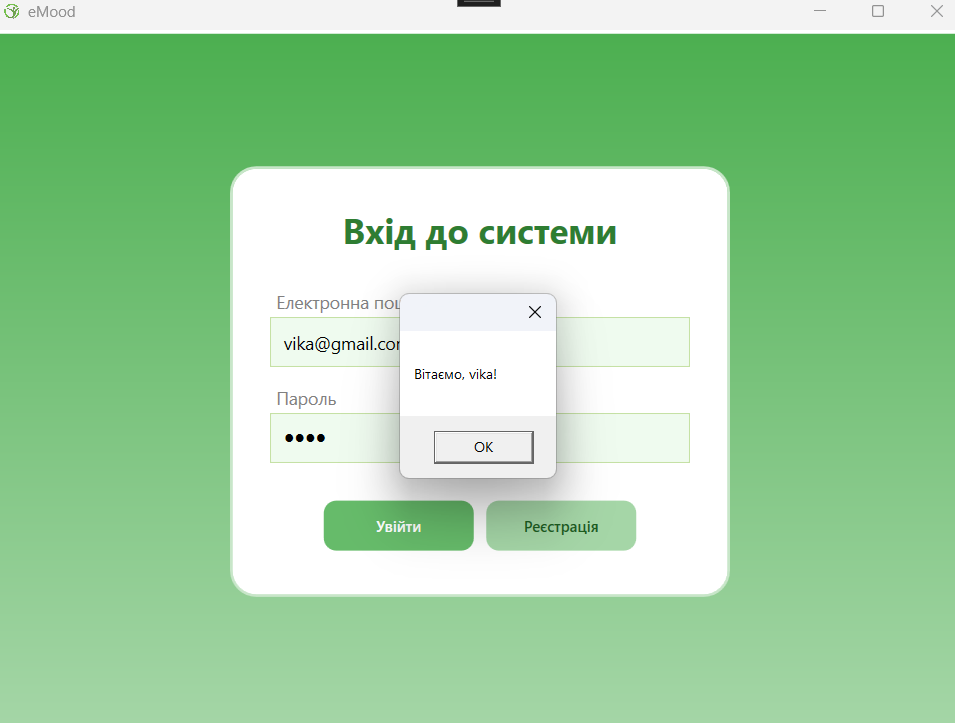
**Результат роботи**

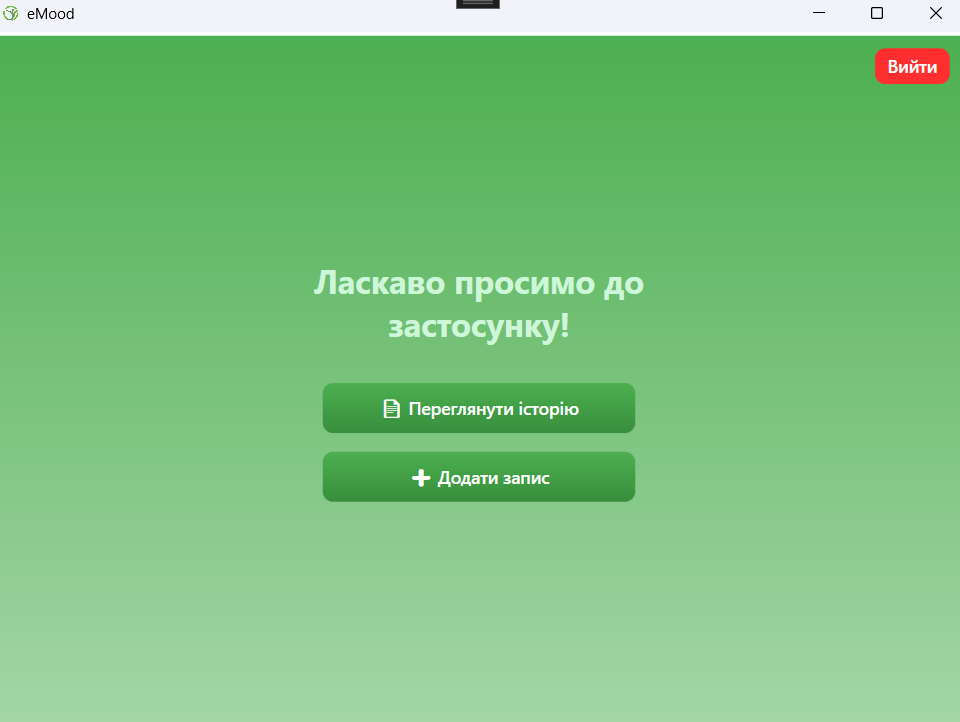


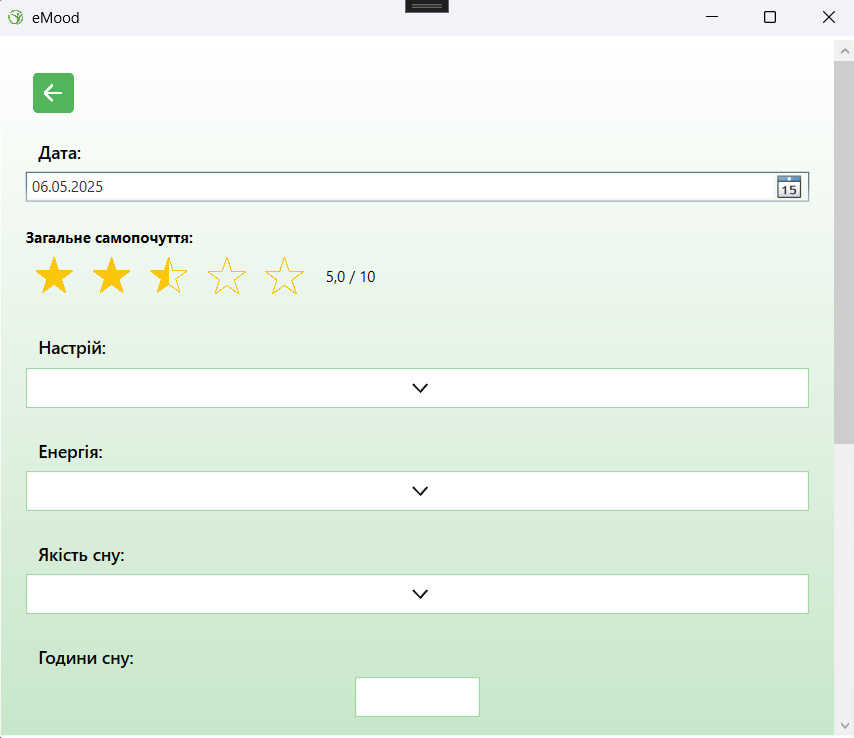


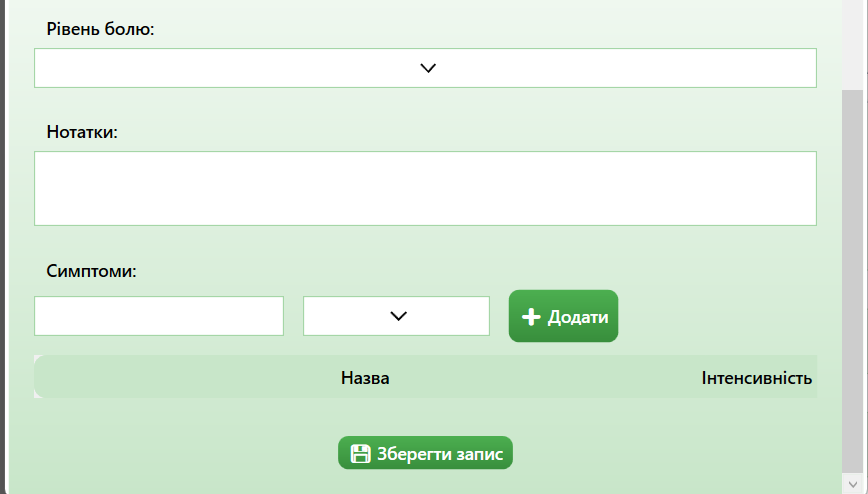


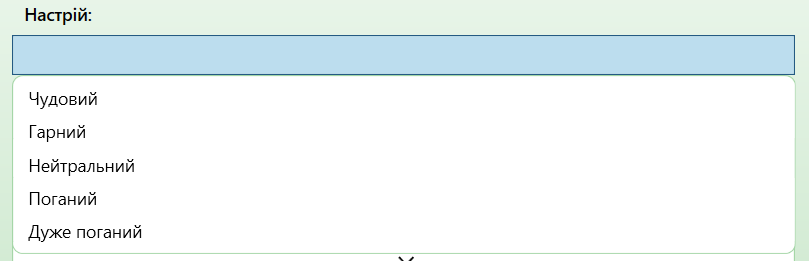


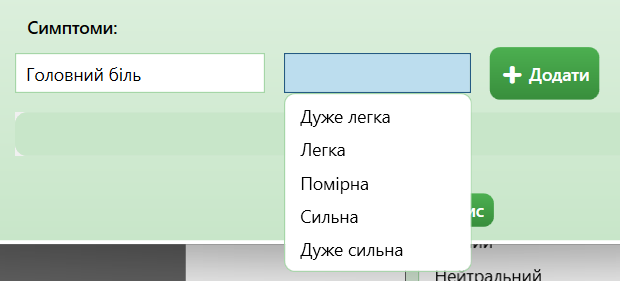


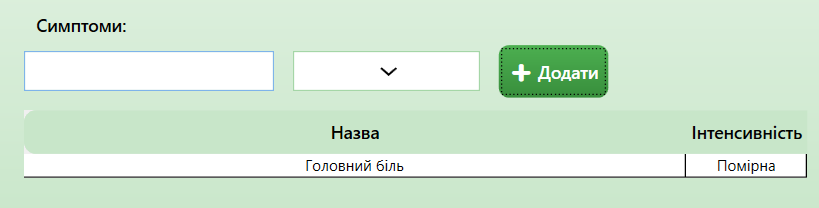


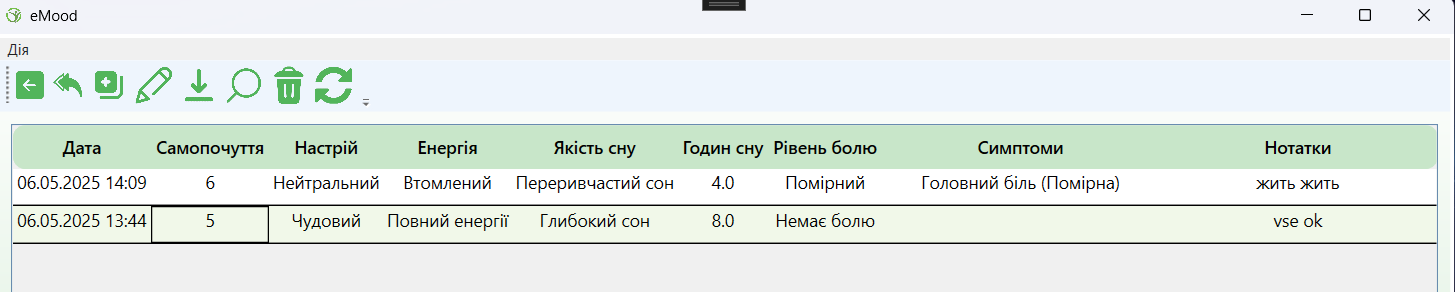


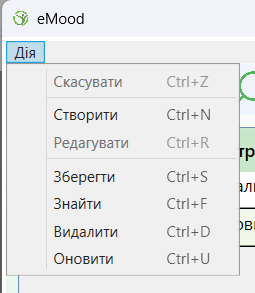
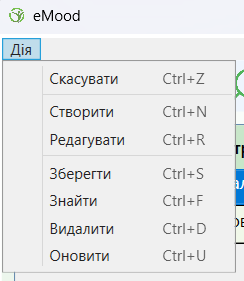










Щодо команд:



Повернення на сторінку:



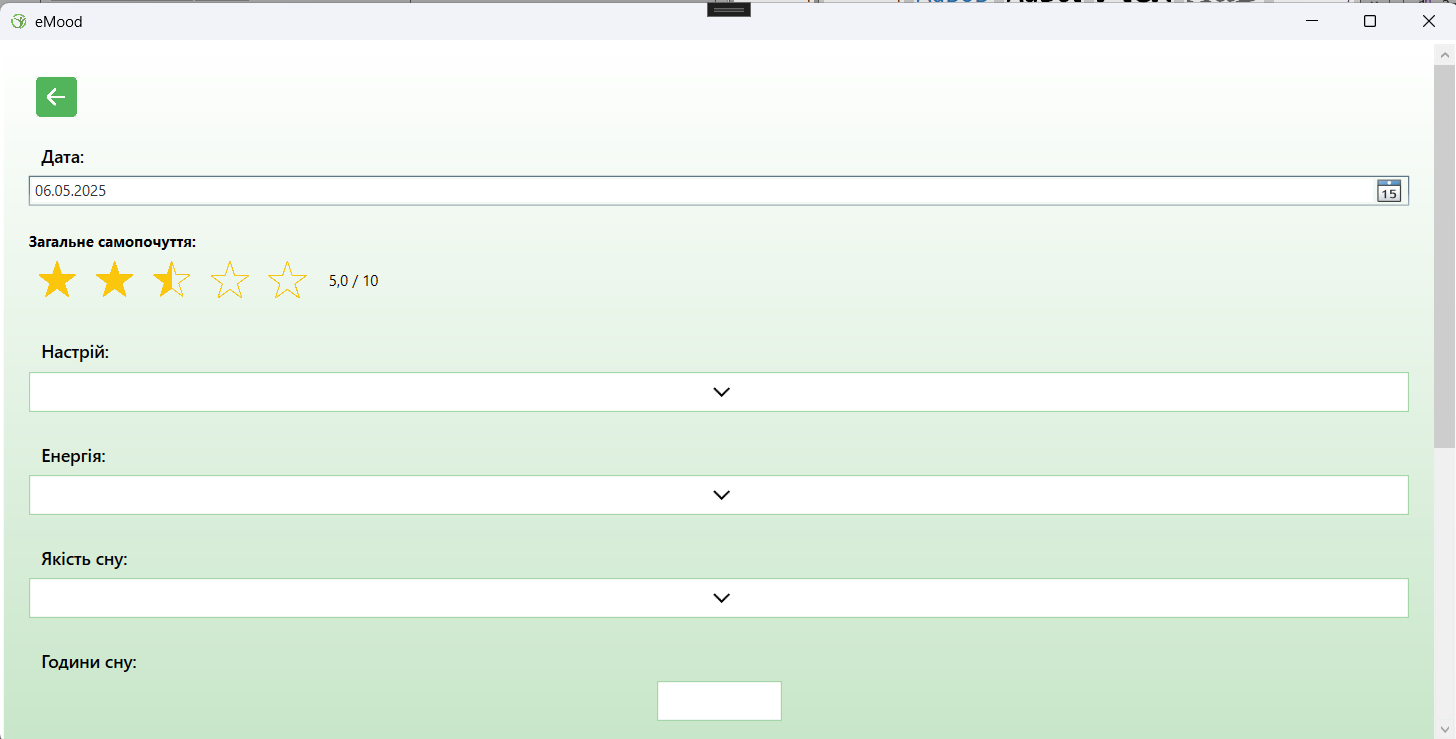
Скасування виділення рядка:



Додавання нового запису:



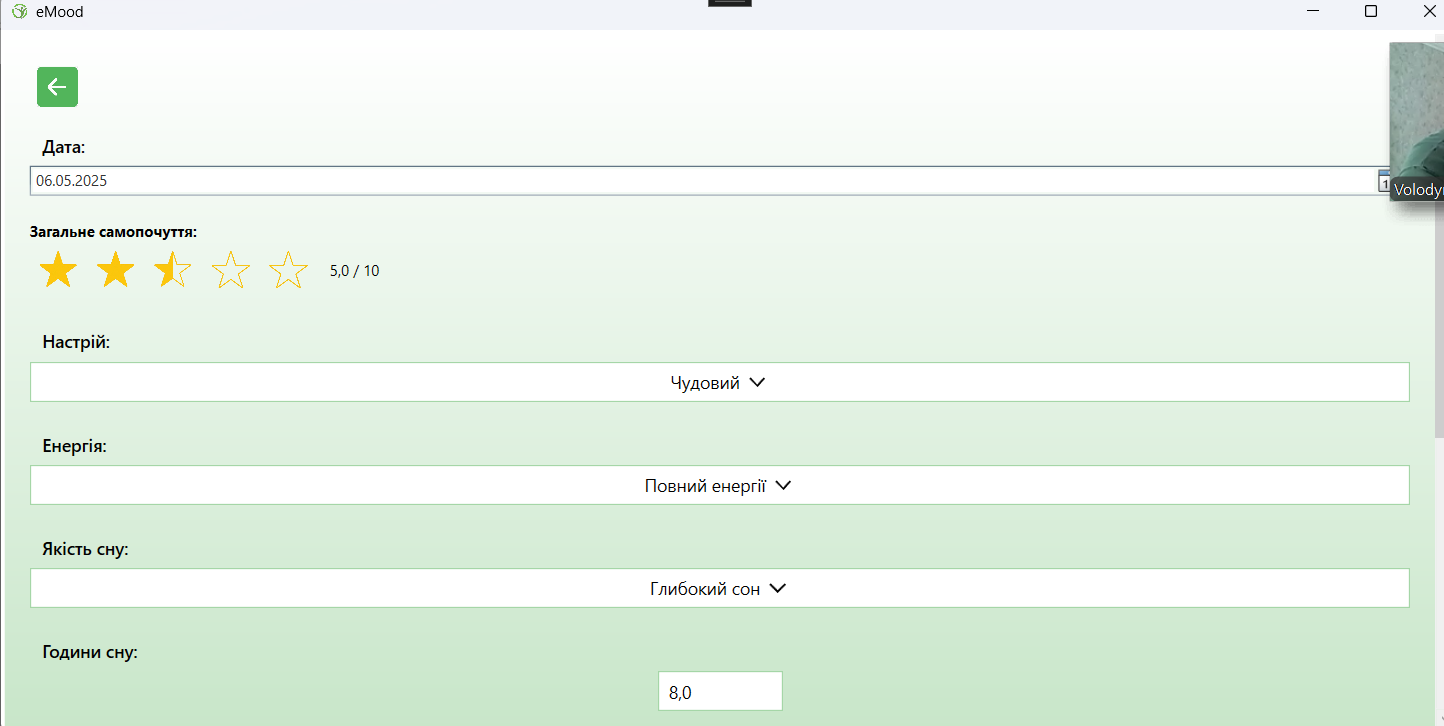
Відкривається сторінка додавання запису:



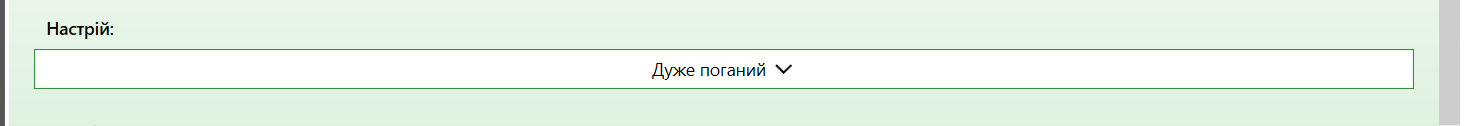
Редагування існуючого запису:



Обираємо запис і натискаємо на олівець:



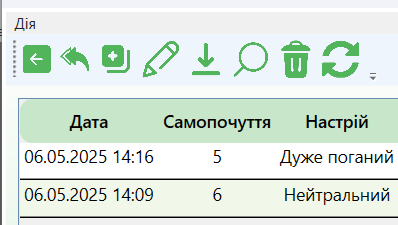
Змінимо «чудовий» на «дуже поганий» та збережемо



Повертаємось і оновлюємо за допомогою:



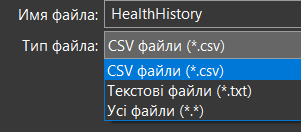
Отримуємо наступне:

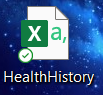


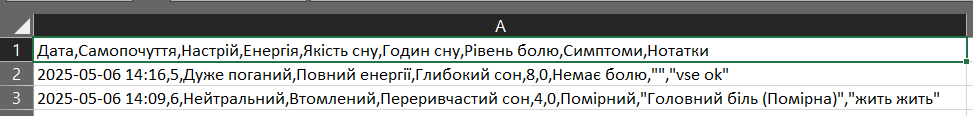
Реалізовано збереження:



Приклад роботи:

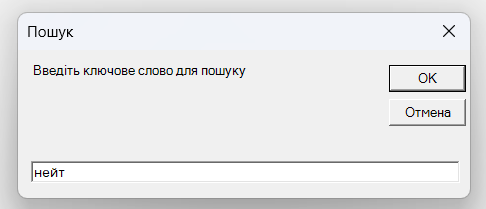


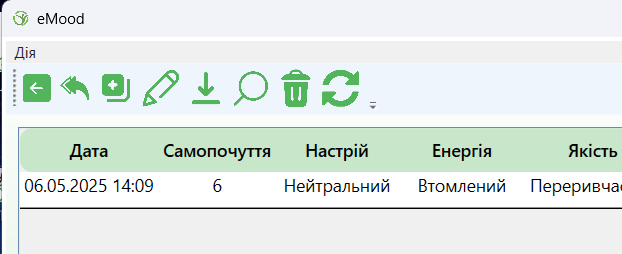




Пошук:







Реалізоване видалення:



