ДОДАТОК А

Графічний матеріал кваліфікаційної роботи

Харківський національний університет радіоелектроніки

Факультет навчально-науковий центр заочної форми навчання

Кафедра електронних обчислювальних машин

КВАЛІФІКАЦІЙНА РОБОТА

перший (бакалаврський)

на тему: Застосунок для мобільного пристрою з використанням .NET MAUI

Виконав:
ст. гр. КІУКІз-21-1
Пересічанська Тетяна Миколаївна

Науковий керівник:
ст. викл.
Дяченко Владислав Олександрович

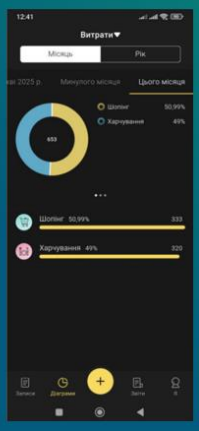
Мета роботи

- Актуальність розробки мобільного застосунку на .NET MAUI
- Аналіз існуючих мобільних застосунків
- Постановка завдання та визначення функціональних вимог
- Аналіз та вибір інструментальних засобів розробки
- Розробка мобільного застосунку
- Тестування роботи застосунку

Актуальність теми

За допомогою .NET MAUI можливо розробляти багатоплатформні застосунки в рамках одного проекту, це дає змогу не витрачати ресурси та час на декілька проектів, створюючи один і той самий застосунок для кожної платформи. При цьому забезпечується гнучкість розробки специфічного для платформи коду і ресурсів там, де це необхідно, через що можливо створити застосунок з більшими можливостями. За його допомогою розробники можуть централізувати більшу частину логіки програми та дизайну інтерфейсу в єдиній кодовій базі, що спрощує процес розробки.

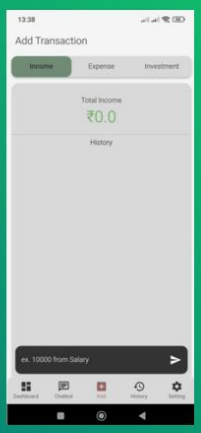
Аналіз існуючих мобільних застосунків



Financing Tracker App

Weather '21

Seeing AI



FinTrack

Технології

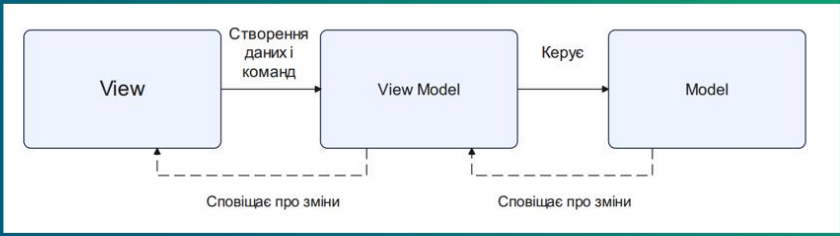
- За допомогою комплексного середовища розробки Visual Studio було розроблено застосунок.
- За допомогою MySQL було створено базу даних для застосунку
- Для розробки застосунку було використано мову програмування C# та мова розмітки XAML.

Переваги та особливості використання платформи .NET MAUI

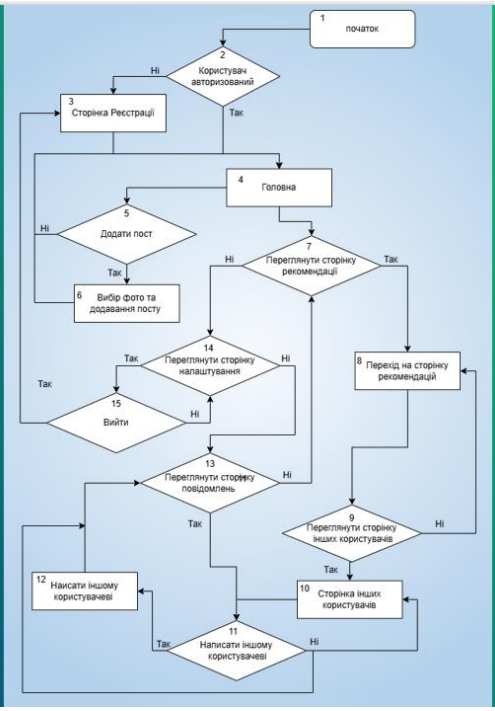
1. Єдина кодова база, яка дозволяє писати один код на C# та XAML для всіх платформ: Android, iOS, Windows, macOS.
2. Спільна бізнес-логіка та UI.
3. Незалежність від платформи в .NET MAUI.
4. Вбудована підтримка шаблонів архітектури.
5. Hot Reload та Live Preview - швидке оновлення інтерфейсу та логіки без повної перекомпіляці.
6. Підтримка сучасних API.
7. Повна підтримка .NET MAUI в Visual Studio 2022.

Архітектура застосунку

Для розробки мобільного застосунку використано архітектурний шаблон MVVM, який є рекомендованим для .NET MAUI. За допомогою MVVM-архітектури можливо розділити логіку інтерфейсу та бізнес-логіку, зручно тестувати окремі компоненти, повторно використовувати ViewModel між платформами та ефективно підтримувати зв'язування даних між UI та логікою.



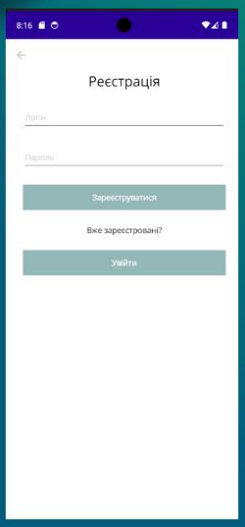
Функціональ на модель застосунку

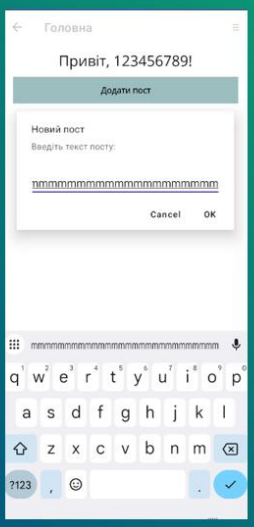


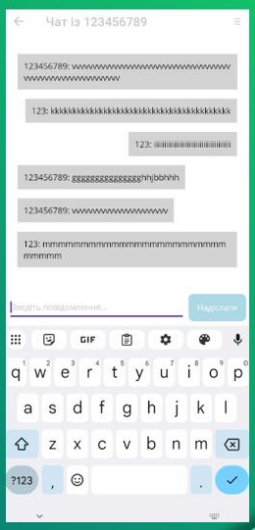
Реалізовано наступні функції

- Реєстрацію та вхід
- Перегляд профілю користувача
- Перегляд стрічки рекомендацій
- Завантаження та видаленя фото
- Обмін повідомленнями
- Адаптивність та продуктивність

Демонстрація роботи застосунку







Висновки

У ході виконання даної роботи розроблено мобільний застосунок з використанням .NET MAUI. Розроблено реєстрацію та авторизацію користувача, створення та видалення постів, обмін повідомленнями між користувачами.

Було виконано тестування роботи застосунку, тестування показало повну працездатність та відповідність застосунку до поставленого завдання. В подальшому застосунок буде розвиватися, будуть розроблені нові функції.

ДОДАТОК Б

Лістінг програмного коду

```
namespace MauiApp1
{
    public partial class App : Application
    {

        public App()
        {
            InitializeComponent();
            MainPage = new NavigationPage(new HomePage("User"));
        }
    }
}
using MySql.Data.MySqlClient;
using BCrypt.Net;
using Microsoft.Maui.Controls;

namespace MauiApp1
{
    public partial class MainPage : ContentPage
    {
        private string connectionString =
"Server=192.168.0.103;Database=myappdb;User ID=root;Password=;
Connection Timeout=60;";

        public MainPage()
        {
            InitializeComponent();
        }
        private async void OnRegisterClicked(object sender,
EventArgs e)
        {
            var username = usernameEntry.Text;
            var password = passwordEntry.Text;
            var passwordHash =
BCrypt.Net.BCrypt.HashPassword(password);
            using (var connection = new
MySqlConnection(connectionString))
            {
                await connection.OpenAsync();
                var query = "INSERT INTO users (Username,
PasswordHash) VALUES (@username, @passwordHash)";
                using (var command = new MySqlCommand(query,
connection))
                {
                    command.Parameters.AddWithValue("@username",
username);

command.Parameters.AddWithValue("@passwordHash", passwordHash);
                    try
```

```
                    {
                        await command.ExecuteNonQueryAsync();
                        await DisplayAlert("Успіх", "Реєстрація
пройшла успішно!", "OK");
                    }
                    catch (MySqlException ex) when (ex.Number ==
1062)
                    {
                        await DisplayAlert("Помилка",
"Користувач із таким ім'ям уже існує.", "OK");
                    }
                }
            }
        }
        private async void OnLoginClicked(object sender,
EventArgs e)
        {
            var username = usernameEntry.Text;
            var password = passwordEntry.Text;

            using (var connection = new
MySqlConnection(connectionString))
            {
                await connection.OpenAsync();
                var query = "SELECT PasswordHash FROM users
WHERE Username = @username";
                using (var command = new MySqlCommand(query,
connection))
                {
                    command.Parameters.AddWithValue("@username",
username);
                    var passwordHash = (string)await
command.ExecuteScalarAsync();
                    if (passwordHash != null &&
BCrypt.Net.BCrypt.Verify(password, passwordHash))
                    {
                        if (!string.IsNullOrEmpty(username))
                        {
                            await Navigation.PushAsync(new
HomePage(username));
                        }
                    }
                    else
                    {
                        await DisplayAlert("Помилка",
"Неправильний логін або пароль", "OK");
                    }
                }
            }
        }
    }
}
<?xml version="1.0" encoding="utf-8" ?>
```

```
<ContentPage
xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="MauiApp1.MainPage">
    <ScrollView>
        <VerticalStackLayout
           Padding="30,0"
           Spacing="25">
            <Label Text="Реєстрація" FontSize="24"
HorizontalOptions="Center" />
            <Entry x:Name="usernameEntry" Placeholder="Логін" />
            <Entry x:Name="passwordEntry" Placeholder="Пароль"
IsPassword="True" />
            <Button x:Name="RegisterButton"
                Text="Зареєструватися"
                Clicked="OnRegisterClicked"
                BackgroundColor="#94b8b8"
                CornerRadius="0"
                FontAttributes="Bold"
               />
            <Label Text="Вже зареєстровані?"
HorizontalOptions="Center" />
            <Button Text="Увійти"
                Clicked="OnLoginClicked"
                BackgroundColor="#94b8b8"
                CornerRadius="0"
                FontAttributes="Bold"
                 />
        </VerticalStackLayout>
    </ScrollView>
</ContentPage>
using Microsoft.Maui.Controls;
using Microsoft.Maui.Storage;
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using System.Linq;
namespace MauiApp1
{
    public partial class HomePage : ContentPage
    {
        private string Username;
        private string connectionString =
"Server=192.168.0.103;Database=myappdb;User ID=root;Password=;
Connection Timeout=30;";
        public HomePage(string username)
        {
            InitializeComponent();
            Username = username;

            if (string.IsNullOrEmpty(Username) ||
```

```
!IsUserExists(Username))
            {
                MainLayout.Children.Clear();

                var loginButtonStyle = new Style(typeof(Button))
                {
                    Setters =
                        {
                            new Setter { Property =
Button.FontAttributesProperty, Value = FontAttributes.Bold },
                            new Setter { Property =
Button.CornerRadiusProperty, Value = 0 },
                            new Setter { Property =
Button.BackgroundColorProperty, Value =
Color.FromArgb("#94b8b8") },
                            new Setter { Property =
Button.TextColorProperty, Value = Colors.Black }
                        }
                };
                MainLayout.Children.Add(new Button
                {
                    Text = "Увійти",

                    Command = new Command(async () => await
Navigation.PushAsync(new MainPage())),
                    Style = loginButtonStyle
                });
                ToolbarItems.Clear();
            }
            else
            {
                GreetingLabel.Text = $"Привіт, {Username}!";
                PostsListView.ItemsSource = GetPosts();
            }
        }
        private async void OnMenuClicked(object sender,
EventArgs e)
        {
            await MenuHelper.HandleMenuAsync(this, Username,
"Головна");
        }
        private async void AddPostButton_Clicked(object sender,
EventArgs e)
        {
            await AddPostAsync();
        }
        private async void DeletePostButton_Clicked(object
sender, EventArgs e)
        {
            var button = sender as Button;
            var post = button?.BindingContext as Post;

            if (post != null)
```

```
            {
                bool confirm = await DisplayAlert("Видалити
пост?", "Ви впевнені, що хочете видалити цей пост?", "Так",
"Ні");
                if (confirm)
                {
                    DeletePostFromDatabase(post);
                    PostsListView.ItemsSource = GetPosts();
                }
            }
        }
        private async Task AddPostAsync()
        {
            var result = await MediaPicker.PickPhotoAsync();
            if (result != null)
            {
                string selectedImagePath = result.FullPath;
                string text = await DisplayPromptAsync("Новий
пост", "Введіть текст посту:");

                if (!string.IsNullOrEmpty(text))
                {
                    SavePostToDatabase(text, selectedImagePath);
                    await DisplayAlert("Успіх", "Ваш пост було
додано!", "OK");
                    PostsListView.ItemsSource = GetPosts();
                }
                else
                {
                    await DisplayAlert("Помилка", "Текст посту
не може бути порожнім.", "OK");
                }
            }
            else
            {
                await DisplayAlert("Помилка", "Зображення не
вибрано.", "OK");
            }
        }
        private void SavePostToDatabase(string text, string
imagePath)
        {
            try
            {
                using var connection = new
MySqlConnection(connectionString);
                connection.Open();
                string query = "INSERT INTO post (username,
text, image_path) VALUES (@username, @text, @image_path)";
                using var command = new MySqlCommand(query,
connection);
                command.Parameters.AddWithValue("@username",
Username);
```

```
                command.Parameters.AddWithValue("@text", text);
                command.Parameters.AddWithValue("@image_path",
imagePath);
                command.ExecuteNonQuery();
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Помилка при збереженні
посту: {ex.Message}");
            }
        }
        private void DeletePostFromDatabase(Post post)
        {
            try
            {
                using var connection = new
MySqlConnection(connectionString);
                connection.Open();
                string query = "DELETE FROM post WHERE username
= @username AND text = @text AND image_path = @image_path";
                using var command = new MySqlCommand(query,
connection);
                command.Parameters.AddWithValue("@username",
Username);
                command.Parameters.AddWithValue("@text",
post.Text);
                command.Parameters.AddWithValue("@image_path",
post.ImagePath);
                command.ExecuteNonQuery();
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Помилка при видаленні посту:
{ex.Message}");
            }
        }
        private List<Post> GetPosts()
        {
            var posts = new List<Post>();
            try
            {
                using var connection = new
MySqlConnection(connectionString);
                connection.Open();
                string query = "SELECT text, image_path,
created_at FROM post WHERE username = @username ORDER BY
created_at DESC";
                using var command = new MySqlCommand(query,
connection);
                command.Parameters.AddWithValue("@username",
Username);
                using var reader = command.ExecuteReader();
                while (reader.Read())
```

```
                {
                    posts.Add(new Post
                    {
                        Text = reader.GetString("text"),
                        ImagePath =
reader.GetString("image_path"),
                        CreatedAt =
reader.GetDateTime("created_at").ToString("dd.MM.yyyy HH:mm")
                    });
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Помилка при завантаженні
постів: {ex.Message}");
            }
            return posts;
        }
        private bool IsUserExists(string username)
        {
            try
            {
                using var connection = new
MySqlConnection(connectionString);
                connection.Open();
                string query = "SELECT COUNT(*) FROM users WHERE
username = @username";
                using var command = new MySqlCommand(query,
connection);
                command.Parameters.AddWithValue("@username",
username);
                return Convert.ToInt32(command.ExecuteScalar())
> 0;
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Помилка під час перевірки
користувача: {ex.Message}");
                return false;
            }
        }
    }
    public class Post
    {
        public string Text { get; set; }
        public string ImagePath { get; set; }
        public string CreatedAt { get; set; }
    }
}
<ContentPage
xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
             x:Class="MauiApp1.HomePage"
```

```
xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             Title="Головна">
    <ContentPage.ToolbarItems>
        <ToolbarItem Text="☰"
                     IconImageSource="menu.png"
                     Order="Primary"
                     Priority="0"
                     Clicked="OnMenuClicked"/>
    </ContentPage.ToolbarItems>
    <StackLayout Padding="10" x:Name="MainLayout">
        <Label x:Name="GreetingLabel"
               FontSize="24"
               HorizontalOptions="Center" />
        <StackLayout Padding="10">
            <Button Text="Додати пост"
                x:Name="AddPostButton"
                Clicked="AddPostButton_Clicked"
                FontAttributes="Bold"
                CornerRadius="0"
                BackgroundColor="#94b8b8"
                TextColor="Black"/>
        </StackLayout>
        <ListView x:Name="PostsListView"
                  HasUnevenRows="True"
                  SeparatorVisibility="Default">
            <ListView.ItemTemplate>
                <DataTemplate>
                    <ViewCell>
                        <StackLayout Padding="10"
Orientation="Vertical" Spacing="10">
                            <StackLayout
Orientation="Horizontal" HorizontalOptions="End">
                                <Button Text="..."

Clicked="DeletePostButton_Clicked"
                                        BindingContext="{Binding
.}"
                                        TextColor="#001a1a"

BackgroundColor="#f0f5f5"/>
                            </StackLayout>
                            <Grid RowDefinitions="*,Auto">
                                <Image Aspect="AspectFit"
                                    Source="{Binding ImagePath}"
                                    HorizontalOptions="Center"
                                    VerticalOptions="Center" />
                            </Grid>
                            <Label FontSize="16"
                                  FontAttributes="Bold"

HorizontalOptions="StartAndExpand"
                                   Text="{Binding Text}" />
                            <Label FontSize="12"
```

```
                                    TextColor="Gray"
HorizontalOptions="StartAndExpand"
                                    Text="{Binding CreatedAt}" />
                        </StackLayout>
                    </ViewCell>
                </DataTemplate>
            </ListView.ItemTemplate>
        </ListView>
    </StackLayout>
</ContentPage>
using Microsoft.Maui.Controls;
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;

namespace MauiApp1
{
    public partial class RecommendationsPage : ContentPage
    {
        private string Username;
        private string connectionString =
"Server=192.168.0.103;Database=myappdb;User ID=root;Password=;
Connection Timeout=30;";
        private async void OnMenuClicked(object sender,
EventArgs e)
        {
            await MenuHelper.HandleMenuAsync(this, Username,
"Рекомендації");
        }
        public RecommendationsPage(string username)
        {
            InitializeComponent();
            Username = username;
            Title = $"Рекомендації";
            GreetingLabel.Text = $"Рекомендації для {Username}";
            PostsListView.ItemsSource = GetPosts();
        }
        private List<Post1> GetPosts()
        {
            var posts = new List<Post1>();
            try
            {
                using (var connection = new
MySqlConnection(connectionString))
                {
                    connection.Open();
                    string query = @"
                SELECT p.id, p.text, p.image_path, p.created_at,
p.username,
                       p.like_count, p.dislike_count,
                       (SELECT vote_type FROM votes v WHERE
v.post_id = p.id AND v.username = @Username LIMIT 1) as
```

```
user_vote
                FROM post p
                ORDER BY p.created_at DESC";

                    using (var command = new MySqlCommand(query,
connection))
                    {

command.Parameters.AddWithValue("@Username", Username);

                        using (var reader =
command.ExecuteReader())
                        {
                            while (reader.Read())
                            {
                                posts.Add(new Post1
                                {
                                    Id = reader.GetInt32("id"),
                                    Text =
reader.GetString("text"),
                                    ImagePath =
reader.GetString("image_path"),
                                    CreatedAt =
reader.GetDateTime("created_at").ToString("dd.MM.yyyy HH:mm"),
                                    Username =
reader.GetString("username"),
                                    LikeCount =
reader.GetInt32("like_count"),
                                    DislikeCount =
reader.GetInt32("dislike_count"),
                                    UserVote =
reader.IsDBNull(reader.GetOrdinal("user_vote")) ? null :
reader.GetString("user_vote")
                                });
                            }
                        }
                    }
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Помилка при завантаженні
постів: {ex.Message}");
            }
            return posts;
        }
        private async void OnLikeClicked(object sender,
EventArgs e)
        {
            if (sender is ImageButton button &&
button.BindingContext is Post1 post)
            {
                await HandleVote(post, "like");
```

```
            }
        }
        private async void OnDislikeClicked(object sender,
EventArgs e)
        {
            if (sender is ImageButton button &&
button.BindingContext is Post1 post)
            {
                await HandleVote(post, "dislike");
            }
        }
        private async Task HandleVote(Post1 post, string
voteType)
        {
            try
            {
                using (var connection = new
MySqlConnection(connectionString))
                {
                    await connection.OpenAsync();
                    string currentVote = post.UserVote;
                    bool removeVote = currentVote == voteType;
                    string query;
                    if (removeVote)
                    {
                        query = "DELETE FROM votes WHERE post_id
= @PostId AND username = @Username";
                    }
                    else if (currentVote == null)
                    {
                        query = "INSERT INTO votes (post_id,
username, vote_type) VALUES (@PostId, @Username, @VoteType)";
                    }
                    else
                    {
                        query = "UPDATE votes SET vote_type =
@VoteType WHERE post_id = @PostId AND username = @Username";
                    }
                    using (var command = new MySqlCommand(query,
connection))
                    {

command.Parameters.AddWithValue("@PostId", post.Id);

command.Parameters.AddWithValue("@Username", Username);
                        if (!removeVote)

command.Parameters.AddWithValue("@VoteType", voteType);

                        await command.ExecuteNonQueryAsync();
                    }
                    string updateCountsQuery = @"
                UPDATE post p
```

```
                SET
                    like_count = (SELECT COUNT(*) FROM votes
WHERE post_id = p.id AND vote_type = 'like'),
                    dislike_count = (SELECT COUNT(*) FROM votes
WHERE post_id = p.id AND vote_type = 'dislike')
                WHERE p.id = @PostId";

                    using (var updateCommand = new
MySqlCommand(updateCountsQuery, connection))
                    {

updateCommand.Parameters.AddWithValue("@PostId", post.Id);
                        await
updateCommand.ExecuteNonQueryAsync();
                    }
                    if (removeVote)
                        post.UserVote = null;
                    else
                        post.UserVote = voteType;
                    var scrollTargetId = post.Id;
                    var updatedPosts = GetPosts();
                    PostsListView.ItemsSource = null;
                    PostsListView.ItemsSource = updatedPosts;
                    var targetPost =
updatedPosts.FirstOrDefault(p => p.Id == scrollTargetId);
                    if (targetPost != null)
                    {
                        await Task.Delay(100);
                        PostsListView.ScrollTo(targetPost,
ScrollToPosition.Center, animated: false);
                    }
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Помилка при голосуванні:
{ex.Message}");
            }
        }
        private async void OnUsernameClicked(object sender,
EventArgs e)
        {
            if (sender is Button button && button.Text is string
clickedUsername)
            {
                await Navigation.PushAsync(new
UserProfilePage(clickedUsername));
            }
        }
    }
    public class Post1
    {
        public int Id { get; set; }
```

```
        public string Text { get; set; }
        public string ImagePath { get; set; }
        public string CreatedAt { get; set; }
        public string Username { get; set; }
        public int LikeCount { get; set; }
        public int DislikeCount { get; set; }
        public string UserVote { get; set; }
    }
}
<ContentPage
xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="MauiApp1.RecommendationsPage"
             Title="Рекомендації">
    <ContentPage.ToolbarItems>
        <ToolbarItem Text="☰"
             IconImageSource="menu.png"
             Order="Primary"
             Priority="0"
             Clicked="OnMenuClicked"/>
    </ContentPage.ToolbarItems>
    <StackLayout>
        <StackLayout Padding="10">
            <Label x:Name="GreetingLabel"
                   FontSize="24"
                   HorizontalOptions="Center"
                   VerticalOptions="Center" />
        </StackLayout>
        <ListView x:Name="PostsListView"
                  HasUnevenRows="True"
                  SeparatorVisibility="Default">
            <ListView.ItemTemplate>
                <DataTemplate>
                    <ViewCell>
                        <StackLayout Padding="10"
Orientation="Vertical" Spacing="10">
                            <Button FontSize="14"
                                    TextColor="#001a1a"
                                    BackgroundColor="#f0f5f5"
                                    Text="{Binding Username}"
                                    Clicked="OnUsernameClicked"
                                    CornerRadius="0"
                                    HorizontalOptions="Fill"
                                    Padding="10,5" />
                            <Grid RowDefinitions="*,Auto">
                                <Image Aspect="AspectFit"
                                       Source="{Binding
ImagePath}"

HorizontalOptions="Center"
                                       VerticalOptions="Center"
/>
```

```
                            </Grid>
                            <Label FontSize="16"
                                   FontAttributes="Bold"

HorizontalOptions="StartAndExpand"
                                   Text="{Binding Text}"
                                   Padding="30,0" />
                            <Label FontSize="12"
                                   TextColor="Gray"

HorizontalOptions="StartAndExpand"
                                   Text="{Binding CreatedAt}" />
                            <Grid ColumnDefinitions="*,*"
                      Margin="0,10,0,0"
                      HorizontalOptions="FillAndExpand"
                      VerticalOptions="Center">
                                <StackLayout
Orientation="Horizontal"
                                 HorizontalOptions="Start"
                                 VerticalOptions="Center"
                                 Grid.Column="0"
                                 Padding="20,0,0,0">
                                    <ImageButton
Source="like_social_icon.png"
                                     Clicked="OnLikeClicked"
                                     WidthRequest="40"
                                     HeightRequest="40"

BackgroundColor="Transparent"/>
                                    <Label Text="{Binding
LikeCount}"
                                VerticalOptions="Center"
                                FontAttributes="Bold"/>
                                 </StackLayout>
                                 <StackLayout
Orientation="Horizontal"
                                  HorizontalOptions="End"
                                  VerticalOptions="Center"
                                  Grid.Column="1"
                                  Padding="0,0,20,0">
                                    <ImageButton
Source="dislike_social_icon.png"
                                     Clicked="OnDislikeClicked"
                                     WidthRequest="40"
                                     HeightRequest="40"

BackgroundColor="Transparent"/>
                                    <Label Text="{Binding
DislikeCount}"
                                VerticalOptions="Center"
                                FontAttributes="Bold"/>
                                 </StackLayout>
                            </Grid>
```

```
                        </StackLayout>
                    </ViewCell>
                </DataTemplate>
            </ListView.ItemTemplate>
        </ListView>

    </StackLayout>
</ContentPage>


using Microsoft.Maui.Controls;
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;

namespace MauiApp1
{
        public partial class UserProfilePage : ContentPage
        {
            private string Username;
            private string connectionString =
"Server=192.168.0.103;Database=myappdb;User ID=root;Password=;
Connection Timeout=30;";
            private async void OnMenuClicked(object sender,
EventArgs e)
            {
                await MenuHelper.HandleMenuAsync(this, Username,
"Рекомендації");
            }
            public UserProfilePage(string username)
            {
                InitializeComponent();
                Username = username;
                GreetingLabel.Text = $"Профіль користувача
{Username}:";
                PostsListView.ItemsSource = GetPosts();
            }
            private List<Post1> GetPosts()
            {
                var posts = new List<Post1>();
                try
                {
                    using (var connection = new
MySqlConnection(connectionString))
                    {
                        connection.Open();
                        string query = @"
                SELECT p.id, p.text, p.image_path, p.created_at,
p.username,
                       p.like_count, p.dislike_count,
                       (SELECT vote_type FROM votes v WHERE
v.post_id = p.id AND v.username = @Username LIMIT 1) as
user_vote
```

```
                FROM post p
                ORDER BY p.created_at DESC";
                        using (var command = new
MySqlCommand(query, connection))
                        {

command.Parameters.AddWithValue("@Username", Username);
                            using (var reader =
command.ExecuteReader())
                            {
                                while (reader.Read())
                                {
                                    posts.Add(new Post1
                                    {
                                        Id =
reader.GetInt32("id"),
                                        Text =
reader.GetString("text"),
                                        ImagePath =
reader.GetString("image_path"),
                                        CreatedAt =
reader.GetDateTime("created_at").ToString("dd.MM.yyyy HH:mm"),
                                        Username =
reader.GetString("username"),
                                        LikeCount =
reader.GetInt32("like_count"),
                                        DislikeCount =
reader.GetInt32("dislike_count"),
                                        UserVote =
reader.IsDBNull(reader.GetOrdinal("user_vote")) ? null :
reader.GetString("user_vote")
                                    });
                                }
                            }
                        }
                    }
                }
                catch (Exception ex)
                {
                    Console.WriteLine($"Помилка при завантаженні
постів: {ex.Message}");
                }
             return posts;
            }
            private async void OnLikeClicked(object sender,
EventArgs e)
        {
            if (sender is ImageButton button &&
button.BindingContext is Post1 post)
            {
                await HandleVote(post, "like");
            }
        }
```

```
        private async void OnDislikeClicked(object sender,
EventArgs e)
        {
            if (sender is ImageButton button &&
button.BindingContext is Post1 post)
            {
                await HandleVote(post, "dislike");
            }
        }
        private async Task HandleVote(Post1 post, string
voteType)
        {
            try
            {
                using (var connection = new
MySqlConnection(connectionString))
                {
                    await connection.OpenAsync();

                    string currentVote = post.UserVote;
                    bool removeVote = currentVote == voteType;

                    string query;
                    if (removeVote)
                    {
                        query = "DELETE FROM votes WHERE post_id
= @PostId AND username = @Username";
                    }
                    else if (currentVote == null)
                    {
                        query = "INSERT INTO votes (post_id,
username, vote_type) VALUES (@PostId, @Username, @VoteType)";
                    }
                    else
                    {
                        query = "UPDATE votes SET vote_type =
@VoteType WHERE post_id = @PostId AND username = @Username";
                    }

                    using (var command = new MySqlCommand(query,
connection))
                    {

command.Parameters.AddWithValue("@PostId", post.Id);

command.Parameters.AddWithValue("@Username", Username);
                        if (!removeVote)

command.Parameters.AddWithValue("@VoteType", voteType);

                        await command.ExecuteNonQueryAsync();
                    }
                    string updateCountsQuery = @"
```

```
                UPDATE post p
                SET
                    like_count = (SELECT COUNT(*) FROM votes
WHERE post_id = p.id AND vote_type = 'like'),
                    dislike_count = (SELECT COUNT(*) FROM votes
WHERE post_id = p.id AND vote_type = 'dislike')
                WHERE p.id = @PostId";
                    using (var updateCommand = new
MySqlCommand(updateCountsQuery, connection))
                    {

updateCommand.Parameters.AddWithValue("@PostId", post.Id);
                        await
updateCommand.ExecuteNonQueryAsync();
                    }
                    if (removeVote)
                        post.UserVote = null;
                    else
                        post.UserVote = voteType;
                    var scrollTargetId = post.Id;
                    var updatedPosts = GetPosts();
                    PostsListView.ItemsSource = null;
                    PostsListView.ItemsSource = updatedPosts;
                    var targetPost =
updatedPosts.FirstOrDefault(p => p.Id == scrollTargetId);
                    if (targetPost != null)
                    {
                        await Task.Delay(100);
                        PostsListView.ScrollTo(targetPost,
ScrollToPosition.Center, animated: false);
                    }
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Помилка при голосуванні:
{ex.Message}");
            }
        }
        private async void OnMessagesButtonClicked(object
sender, EventArgs e)
        {
            await Navigation.PushAsync(new
NotificationsPage(Username));
        }
    }
}
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage
xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="MauiApp1.UserProfilePage"
```

```
             Title="Профіль користувача">
    <ContentPage.ToolbarItems>
        <ToolbarItem Text="☰"
              IconImageSource="menu.png"
              Order="Primary"
              Priority="0"
              Clicked="OnMenuClicked"/>
    </ContentPage.ToolbarItems>
    <StackLayout Padding="10">
        <Label x:Name="GreetingLabel"
               FontSize="24"
               HorizontalOptions="Center"
               VerticalOptions="Center" />
        <StackLayout Padding="10">
            <Button x:Name="MessagesButton"
                Text="Повідомлення"
                BackgroundColor="#94b8b8"
                TextColor="Black"
                Clicked="OnMessagesButtonClicked"
                FontAttributes="Bold"
                CornerRadius="0"/>
        </StackLayout>
        <ListView x:Name="PostsListView"
                  HasUnevenRows="True"
                  SeparatorVisibility="Default">
            <ListView.ItemTemplate>
                <DataTemplate>
                    <ViewCell>
                        <StackLayout Padding="10"
Orientation="Vertical" Spacing="10">
                            <Grid RowDefinitions="*,Auto">
                                <Image Aspect="AspectFit"
                                  Source="{Binding ImagePath}"
                                  HorizontalOptions="Center"
                                  VerticalOptions="Center" />
                            </Grid>
                            <Label FontSize="16"
                                   FontAttributes="Bold"

HorizontalOptions="StartAndExpand"
                                   Text="{Binding Text}"
                                   Padding="30,0"
                                  />
                            <Label FontSize="12"
                                   TextColor="Gray"

HorizontalOptions="StartAndExpand"
                                   Text="{Binding CreatedAt}" />
                            <Grid ColumnDefinitions="*,*"
Margin="0,10,0,0"
HorizontalOptions="FillAndExpand"
VerticalOptions="Center">
                                <StackLayout
```

```
Orientation="Horizontal"
          HorizontalOptions="Start"
          VerticalOptions="Center"
          Grid.Column="0"
          Padding="20,0,0,0">
                                    <ImageButton
Source="like_social_icon.png"
              Clicked="OnLikeClicked"
              WidthRequest="40"
              HeightRequest="40"
              BackgroundColor="Transparent"/>
                                    <Label Text="{Binding
LikeCount}"
         VerticalOptions="Center"
         FontAttributes="Bold"/>
                                </StackLayout>
                                <StackLayout
Orientation="Horizontal"
          HorizontalOptions="End"
          VerticalOptions="Center"
          Grid.Column="1"
          Padding="0,0,20,0">
                                    <ImageButton
Source="dislike_social_icon.png"
              Clicked="OnDislikeClicked"
              WidthRequest="40"
              HeightRequest="40"
              BackgroundColor="Transparent"/>
                                    <Label Text="{Binding
DislikeCount}"
         VerticalOptions="Center"
         FontAttributes="Bold"/>
                                </StackLayout>
                            </Grid>
                        </StackLayout>
                    </ViewCell>
                </DataTemplate>
            </ListView.ItemTemplate>
        </ListView>
    </StackLayout>
</ContentPage>



using Microsoft.Maui.Controls;
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;

namespace MauiApp1
{
    public partial class MessagePage : ContentPage
    {
```

```
        private List<string> Messages = new List<string>();
        private string CurrentUser;
        private string OtherUser;
        private string connectionString =
"Server=192.168.0.103;Database=myappdb;User ID=root;Password=;
Connection Timeout=30;";
        private async void OnMenuClicked(object sender,
EventArgs e)
        {
            await MenuHelper.HandleMenuAsync(this, CurrentUser,
"Сповіщення");
        }
        public MessagePage(string currentUser, string otherUser)
        {
            InitializeComponent();
            CurrentUser = currentUser;
            OtherUser = otherUser;

            Title = $"Чат із {OtherUser}";
            LoadMessages();
        }
        private void LoadMessages()
        {
            try
            {
                using var conn = new
MySqlConnection(connectionString);
                conn.Open();
                string query = @"
                    SELECT sender, content
                    FROM message
                    WHERE
                        (sender = @user1 AND receiver = @user2)
OR
                        (sender = @user2 AND receiver = @user1)
                    ORDER BY timestamp";
                using var cmd = new MySqlCommand(query, conn);
                cmd.Parameters.AddWithValue("@user1",
CurrentUser);
                cmd.Parameters.AddWithValue("@user2",
OtherUser);
                using var reader = cmd.ExecuteReader();
                while (reader.Read())
                {
                    string sender = reader.GetString("sender");
                    string content =
reader.GetString("content");
                    string fullMessage = $"{sender}: {content}";
                    Messages.Add(fullMessage);
                    AddMessageToView(fullMessage, sender ==
CurrentUser);
                }
            }
```

```
            catch (Exception ex)
            {
                Console.WriteLine($"Помилка при завантаженні
повідомлень: {ex.Message}");
            }
        }
        private void SaveMessage(string content)
        {
            try
            {
                using var conn = new
MySqlConnection(connectionString);
                conn.Open();
                string query = "INSERT INTO message (sender,
receiver, content, timestamp) VALUES (@sender, @receiver,
@content, NOW())";
                using var cmd = new MySqlCommand(query, conn);
                cmd.Parameters.AddWithValue("@sender",
CurrentUser);
                cmd.Parameters.AddWithValue("@receiver",
OtherUser);
                cmd.Parameters.AddWithValue("@content",
content);
                cmd.ExecuteNonQuery();
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Помилка при збереженні
повідомлення: {ex.Message}");
            }
        }
        private void AddMessageToView(string message, bool
isCurrentUser)
        {
            var messageLabel = new Label
            {
                Text = message,
                BackgroundColor = Colors.LightGray,
                Padding = new Thickness(10),
                Margin = new Thickness(5),
                HorizontalOptions = isCurrentUser ?
LayoutOptions.End : LayoutOptions.Start,
                TextColor = Colors.Black
            };
            messageContainer.Children.Add(messageLabel);
        }
        private void OnSendButtonClicked(object sender,
EventArgs e)
        {
            string text = messageEntry.Text;
            if (!string.IsNullOrWhiteSpace(text))
            {
                try
```

```
                {
                    SaveMessage(text);
                    string formattedMessage = $"{CurrentUser}:
{text}";
                    Messages.Add(formattedMessage);
                    AddMessageToView(formattedMessage, true);
                    messageEntry.Text = "";
                }
                catch (Exception ex)
                {
                    Console.WriteLine($"Помилка надсилання:
{ex.Message}");
                }
            }
        }
    }
}
```

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage
xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="MauiApp1.MessagePage"
             Title="Чат">
    <ContentPage.ToolbarItems>
        <ToolbarItem Text="☰"
     IconImageSource="menu.png"
     Order="Primary"
     Priority="0"
     Clicked="OnMenuClicked"/>
    </ContentPage.ToolbarItems>
    <ContentPage.Content>
        <StackLayout Padding="10" Spacing="10">
            <ScrollView HeightRequest="400"
                        BackgroundColor="White">
                <StackLayout x:Name="messageContainer"
                             Padding="10"
                             Spacing="5" />
            </ScrollView>
            <StackLayout Orientation="Horizontal" Spacing="5">
                <Entry x:Name="messageEntry"
                       Placeholder="Введіть повідомлення..."
                       HorizontalOptions="FillAndExpand" />
                <Button Text="Надіслати"
                        Clicked="OnSendButtonClicked"
                        BackgroundColor="LightBlue"
                        TextColor="White" />
            </StackLayout>
        </StackLayout>
    </ContentPage.Content>
```

```
</ContentPage>
using Microsoft.Maui.Controls;
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
namespace MauiApp1
{
    public partial class NotificationsPage : ContentPage
    {
        private string Username;
        private HashSet<string> AddedUserButtons = new();
        private string connectionString =
"Server=192.168.0.103;Database=myappdb;User ID=root;Password=;
Connection Timeout=30;";
        private async void OnMenuClicked(object sender,
EventArgs e)
        {
            await MenuHelper.HandleMenuAsync(this, Username,
"Сповіщення");
        }
        public NotificationsPage(string username)
        {
            InitializeComponent();
            Username = username;
            LoadUsersFromDatabase();
        }
        private void LoadUsersFromDatabase()
        {
            try
            {
                using var connection = new
MySqlConnection(connectionString);
                connection.Open();
                string query = "SELECT DISTINCT username FROM
users";
                using var command = new MySqlCommand(query,
connection);
                using var reader = command.ExecuteReader();
                while (reader.Read())
                {
                    string user = reader.GetString("username");
                    if (user != Username)
                    {
                        AddUserButton(user);
                    }
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Помилка при отриманні списку
користувачів: {ex.Message}");
            }
        }
```

```
        private void AddUserButton(string username)
        {
            if (!AddedUserButtons.Contains(username))
            {
                var userButton = new Button
                {
                    Text = username,
                    Style = (Style)Resources["UserButtonStyle"]
                };
                userButton.Clicked += async (sender, e) =>
                {
                    await Navigation.PushAsync(new
MessagePage(Username, username));
                };
                buttonContainer.Children.Add(userButton);
                AddedUserButtons.Add(username);
            }
        }
    }
}
```

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage
xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="MauiApp1.NotificationsPage"
             Title="Сповіщення">
    <ContentPage.ToolbarItems>
        <ToolbarItem Text="☰"
        IconImageSource="menu.png"
        Order="Primary"
        Priority="0"
        Clicked="OnMenuClicked"/>
    </ContentPage.ToolbarItems>
    <ContentPage.Resources>
        <Style x:Key="UserButtonStyle" TargetType="Button">
            <Setter Property="BackgroundColor" Value="#94b8b8"
/>
            <Setter Property="TextColor" Value="Black" />
            <Setter Property="CornerRadius" Value="0" />
            <Setter Property="FontAttributes" Value="Bold" />
            <Setter Property="Padding" Value="10,5" />
        </Style>
    </ContentPage.Resources>
    <ScrollView>
        <StackLayout Padding="10">
            <Label Text="Список користувачів:"
                   FontSize="24"
                   HorizontalOptions="Center"
                   VerticalOptions="Center" />
            <StackLayout x:Name="buttonContainer"
                         Padding="10"
```

```
                        Spacing="10" />
        </StackLayout>
    </ScrollView>
</ContentPage>

using Microsoft.Maui.Controls;
using System;
namespace MauiApp1
{
    public partial class SettingsPage : ContentPage
    {
        private string Username;
        private async void OnMenuClicked(object sender,
EventArgs e)
        {
            await MenuHelper.HandleMenuAsync(this, Username,
"Налаштування");
        }
        public SettingsPage(string username)
        {
            InitializeComponent();
            Username = username;
            settingsLabel.Text = $"Налаштування для {Username}";

        }
        private async void OnLogoutButtonClicked(object sender,
EventArgs e)
        {
            await Navigation.PushAsync(new MainPage());
        }
    }
}
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage
xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="MauiApp1.SettingsPage"
             Title="Налаштування">
    <ContentPage.ToolbarItems>
        <ToolbarItem Text="☰"
           IconImageSource="menu.png"
           Order="Primary"
           Priority="0"
           Clicked="OnMenuClicked"/>
    </ContentPage.ToolbarItems>
    <ScrollView>
        <StackLayout
            Padding="30,0"
            Spacing="25">
            <Label x:Name="settingsLabel"
                   FontSize="24"
                   HorizontalOptions="Center"
```

```
                    VerticalOptions="Center" />
                <Button x:Name="logoutButton"
                    Text="Вихід"
                    FontAttributes="Bold"
                    BackgroundColor="#94b8b8"
                    TextColor="Black"
                    Clicked="OnLogoutButtonClicked"
                    CornerRadius="0"
                    />
        </StackLayout>
    </ScrollView>
</ContentPage>
```