



UNIVERZITET U SARAJEVU
FAKULTET ZA SAOBRAĆAJ I KOMUNIKACIJE



SEMINARSKI RAD IZ PREDMETA:
Razvoj aplikacija mobilnih uređaja

Tema rada:	KFC Menu Mobile Application
-------------------	------------------------------------

Predmetni nastavnik:	v. prof. Asmir Butković
Asistent (saradnik):	

Student:	Armin Hrelja
Broj indeksa:	8442
Usmjerenje:	KiT
Godina studija:	III godina
Rezultat rada:	

Datum: 19.06.2024.

UVOD

Mobilna aplikacija "KFC Menu" namijenjena je digitalizaciji, optimizaciji i poboljšavanju korisničkog iskustva prilikom naručivanja i dostave hrane. Aplikacija omogućava korisnicima kreiranje profila i uređivanje detalja o profile, čime se omogućava personalizacija usluge. Korisnici mogu pretraživati hranu za narudžbu prema kategorijama, što olakšava pronalazak željenih artikala. Omiljena hrana može se dodati u listu "My Favorites" za brži pristup pri budućim narudžbama. Po završetku narudžbe, korisnicima će biti ispisana poruka da je narudžba dostavljena, čime se osigurava transparentnost i informisanost korisnika.

Korisnici mogu kreirati svoj profil unoseći osnovne informacije kao što su ime, e – mail i lozinka. Ovo omogućava personalizovanu interakciju sa aplikacijom i čuva korisničke preferencije za buduće sesije.

Nakon kreiranja profila, korisnici mogu uređivati svoje podatke u bilo kojem trenutku. To uključuje promjenu osnovnih podataka, ažuriranje adrese za dostavu, ili mijenjanje postavki lozinke.

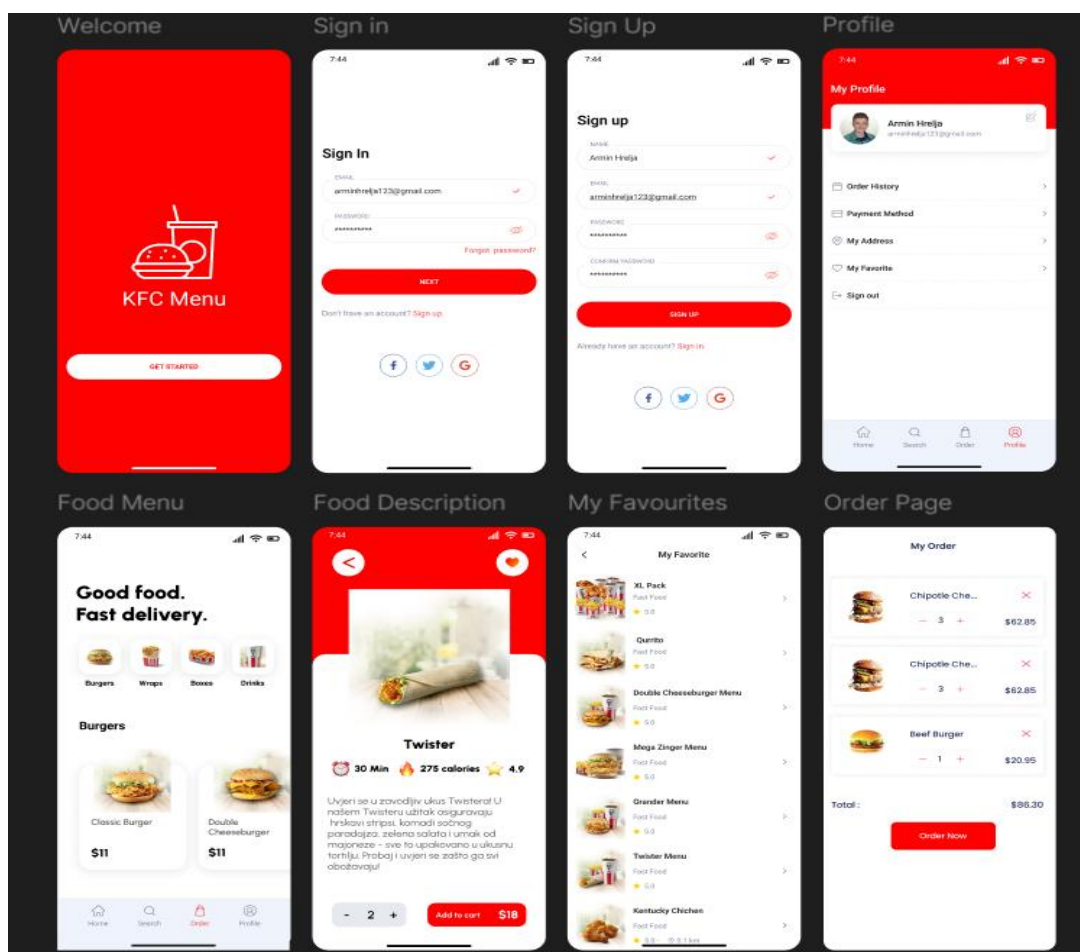
Korisnici mogu pregledavati menu prema kategorijama, čime se olakšava istraživanje dostupnih opcija.

Korisnici mogu označiti svoja omiljena jela i dodati ih u listu "My Favorites". Ovo omogućava brži pristup najdražim jelima i olakšava ponovno naručivanje omiljene hrane.

Nakon što je narudžba uspješno dostavljena, korisnici će dobiti obavijest putem aplikacije. Ova funkcionalnost osigurava da korisnici budu informisani o statusu svoje narudžbe i smanjuje neizvjesnost oko vremena dostave.

PROTOTIP

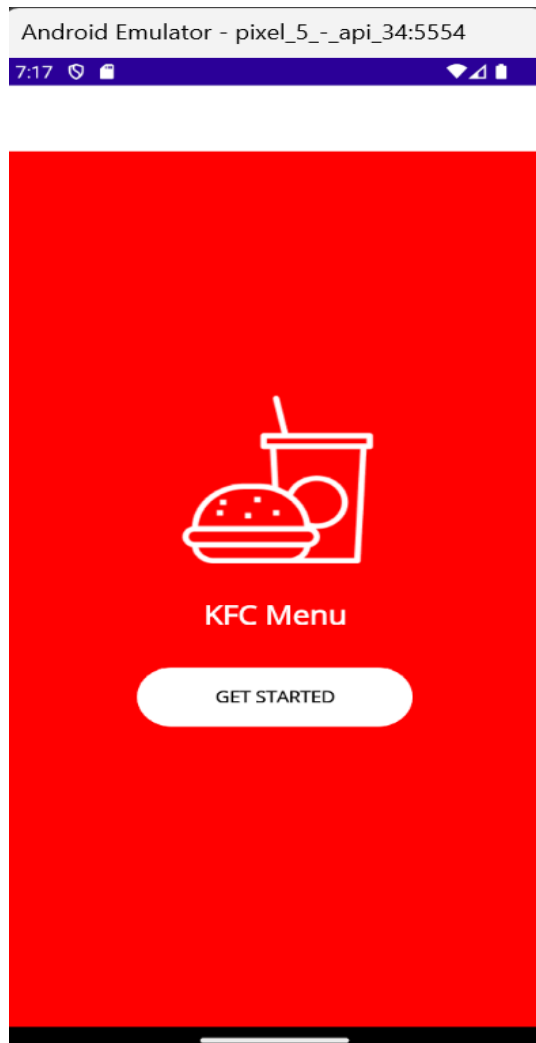
Ovaj KFC Menu mobilni app prototip omogućava korisnicima da lako pregledaju, odaberu i naruče hranu. Početna stranica dočekuje korisnike s jasno istaknutim logom i gumbom za početak, dok ekrani za prijavu i registraciju omogućavaju jednostavno kreiranje i pristup korisničkim računima. Profilna stranica prikazuje osnovne informacije o korisniku i pruža navigacijske opcije za pregled povijesti narudžbi, favorita i postavki plaćanja. Stranica s jelovnikom kategorizira dostupne artikle, a detaljni prikaz hrane nudi informacije o sastojcima, kalorijama i ocjenama. Ekran s favoritima omogućava brz pregled omiljenih artikala, dok stranica za narudžbe omogućava korisnicima da upravljaju svojim narudžbama i završe kupovinu s lakoćom. Sve stranice su vizualno privlačne i intuitivne za korištenje, osiguravajući ugodno korisničko iskustvo.



Slika 1. Prototip projekta (Figma)

GET STARTED PAGE

Ova XAML datoteka definira glavnu stranicu aplikacije pod nazivom "KFC_Menu" koristeći .NET MAUI framework. Stranica sadrži vertikalno složen layout s crvenom pozadinom, gdje su postavljeni logotip slike KFC-a, natpis "KFC Menu" i gumb "GET STARTED". Logotip je centriran na stranici s određenim dimenzijama i margina. Natpis je bijele boje, velikog fonta i podebljan. Gumb je također centriran, bijele boje s crnim tekstom i zaobljenim rubovima. Pritiskom na gumb pokreće se metoda `GetStarted_Clicked`.



Slika 2. Get Started Page

```

namespace KFC_Menu
{
    public partial class MainPage : ContentPage
    {
        public MainPage()
        {
            InitializeComponent();
        }

        private void GetStarted_Clicked(object sender, EventArgs e)
        {
            Navigation.PushAsync(new SignInPage());
        }
    }
}

```

}

Konstruktor MainPage poziva metodu InitializeComponent koja inicijalizira komponentu, što uključuje učitavanje XAML sadržaja za ovu stranicu. Konstruktor MainPage poziva metodu InitializeComponent koja inicijalizira komponentu, što uključuje učitavanje XAML sadržaja za ovu stranicu.

XAML KOD:

```

<?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="KFC_Menu.MainPage"
    Shell.NavBarIsVisible="False">

```

```

    <ContentPage.Content>
        <VerticalStackLayout BackgroundColor="Red">
            <Grid RowDefinitions="*, *, *">

```

```

                <Image Source="logo.png"
                    VerticalOptions="Center"
                    HorizontalOptions="Center"
                    Grid.Row="0"
                    HeightRequest="200"
                    WidthRequest="200"
                    Margin="0,180,0,0"/>

```

Image sa logom aplikacije

```

                <Label Text="KFC Menu"
                    TextColor="White"
                    FontSize="Large"
                    HorizontalOptions="Center"
                    FontAttributes="Bold"
                    Margin="0,0,0,30"
                    Grid.Row="1"/>

```

Label sa imenom aplikacije

```

                <Button Text="GET STARTED"
                    BackgroundColor="White"
                    TextColor="Black"
                    FontAttributes="Bold"
                    CornerRadius="25"
                    WidthRequest="200"
                    HeightRequest="50"
                    HorizontalOptions="Center"
                    VerticalOptions="Center"
                    Grid.Row="2"
                    Clicked="GetStarted_Clicked"/>

```

Button na koji kada se klikne, prebacuje na Sign In Page

```

            </Grid>
        </VerticalStackLayout>
    </ContentPage.Content>
</ContentPage>

```

DATABASE SERVICES I MODELS

Ovaj kod predstavlja DatabaseServices klasu koja upravlja svim operacijama baze podataka za KFC Menu aplikaciju, koristeći SQLite za lokalno skladištenje podataka. Klasa implementira Singleton pattern da bi osigurala jedinstvenu instancu u celoj aplikaciji. initializeDatabase metoda kreira tabele ako ne postoje, uključujući korisnike, stavke menija, omiljene stavke, narudžbine i stavke narudžbine. Klasa omogućava dodavanje, ažuriranje, brisanje i preuzimanje korisnika, narudžbina, stavki narudžbine i omiljenih stavki. Takođe upravlja prijavom i odjavom korisnika pomoću metoda setLoggedInUser i signOutUser. Ova klasa centralizuje sve operacije baze podataka, omogućavajući ostalim delovima aplikacije jednostavan pristup potrebnim podacima i funkcionalnostima.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using KFC_Menu.Models;
using SQLite;
using static KFC_Menu.OrderPage;

namespace KFC_Menu.Services
{
    public class DatabaseServices
    {
        private static DatabaseServices instance;
        private static SQLiteAsyncConnection db;
        private static readonly string dbPath =
Path.Combine(Environment.GetFolderPath(Environment.SpecialFolder.LocalApplication
Data), "KFC_Menu.db3");
        private Models.User loggedInUser;

        public static DatabaseServices Instance => instance ??= new
DatabaseServices();
        public static async Task initializeDatabase()
        {
            try
            {
                if (db == null)
                {
                    db = new SQLiteAsyncConnection(dbPath);
                    await db.CreateTableAsync<User>();
                    await db.CreateTableAsync<MenuItem>();
                    await db.CreateTableAsync<Favorite>();
                    await db.CreateTableAsync<Order>();
                    await db.CreateTableAsync<OrderItem>();
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex.Message);
            }
        }

        public static async Task<int> addUser(User user)
        {
            try
            {
                await DatabaseServices.initializeDatabase();
```

```

        var result = await db.InsertAsync(user);
        Console.WriteLine($"User added with result: {result}");
        return result;
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
        return -1;
    }
}

public static Task<User> getUser(string email, string password)
{
    return db.Table<User>().FirstOrDefaultAsync(u => u.Email == email &&
u.Password == password);
}

public Task<User> getLoggedInUser()
{
    return Task.FromResult(loggedInUser);
}

public Task setLoggedInUser(User user)
{
    loggedInUser = user;
    return Task.CompletedTask;
}

public Task signOutUser()
{
    loggedInUser = null;
    return Task.CompletedTask;
}

public async Task<int> addOrder(Order order)
{
    await initializeDatabase();
    return await db.InsertAsync(order);
}

public async Task<int> addOrderItem(OrderItem orderItem)
{
    await initializeDatabase();
    return await db.InsertAsync(orderItem);
}

public async Task<List<Order>> getOrdersForUser(int userId)
{
    await initializeDatabase();
    return await db.Table<Order>().Where(o => o.UserId ==
userId).ToListAsync();
}

public async Task<int> updateOrderItem(OrderItem orderItem)
{
    await DatabaseServices.initializeDatabase();
    return await db.UpdateAsync(orderItem);
}

public async Task<int> removeOrderItem(int orderItemId)
{
    await initializeDatabase();

```

```

        return await db.DeleteAsync<OrderItem>(orderId);
    }

    public async Task<List<OrderItem>> getOrderItem(int userId)
    {
        await DatabaseServices.initializeDatabase();
        return await db.Table<OrderItem>().Where(oi => oi.UserId ==
userId).ToListAsync();
    }
    public async Task addFavorite(int userId, MenuItem item)
    {
        var favorite = new Favorite
        {
            UserId = userId,
            ItemId = item.Id,
            ItemName = item.Name,
            ItemPrice = item.Price,
            ItemImage = item.Image,
            ItemDescription = item.Description,
            ItemCategory = item.Category
        };
        await db.InsertAsync(favorite);
    }

    public async Task<Favorite> getFavoriteItem(int userId, int itemId)
    {
        await DatabaseServices.initializeDatabase();
        return await db.Table<Favorite>().FirstOrDefaultAsync(fi => fi.UserId
== userId && fi.ItemId == itemId);
    }

    public async Task<List<MenuItem>> getFavoritesForUser(int userId)
    {
        await DatabaseServices.initializeDatabase();
        var userFavorites = await db.Table<Favorite>().Where(f => f.UserId ==
userId).ToListAsync();
        return userFavorites.Select(f => new MenuItem
        {
            Id = f.ItemId,
            Name = f.ItemName,
            Price = f.ItemPrice,
            Image = f.ItemImage,
            Description = f.ItemDescription,
            Category = f.ItemCategory
        }).ToList();
    }

    public async Task removeFromFavorites(int userId, int itemId)
    {
        await DatabaseServices.initializeDatabase();
        var favorite = await getFavoriteItem(userId, itemId);
        if (favorite != null)
        {
            await db.DeleteAsync(favorite);
        }
    }
}

}

```


Modeli

Ovi modeli predstavljaju osnovne entitete korišćene u KFC Menu aplikaciji i mapiraju se na odgovarajuće tabele u SQLite bazi podataka. User klasa definiše korisnika sa jedinstvenim ID-em, imenom, e-mail adresom i lozinkom, pri čemu su e-mail i lozinka jedinstveni. Order klasa predstavlja narudžbinu, uključujući ID narudžbine, ID korisnika koji je napravio narudžbinu, datum narudžbine, naziv stavke, cenu, ukupnu cenu i količinu. Favorite klasa definiše omiljene stavke korisnika sa ID-em favorita, ID-em korisnika, ID-em stavke, nazivom, cenom, slikom, opisom i kategorijom stavke. Ovi modeli omogućavaju čuvanje i upravljanje korisničkim podacima, narudžbinama i omiljenim stavkama u aplikaciji.

User.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using SQLite;

namespace KFC_Menu.Models
{
    public class User
    {
        [PrimaryKey, AutoIncrement]
        public int Id { get; set; }
        public string Name { get; set; }
        [Unique]
        public string Email { get; set; }
        [Unique]
        public string Password { get; set; }
    }
}
```

Order.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace KFC_Menu.Models
{
    public class Order
    {
        public int Id { get; set; }
        public int UserId { get; set; }
        public DateTime OrderDate { get; set; }
        public string ItemName { get; set; }
        public decimal Price { get; set; }
        public decimal TotalPrice { get; set; }
        public int Quantity { get; set; }
    }
}
```

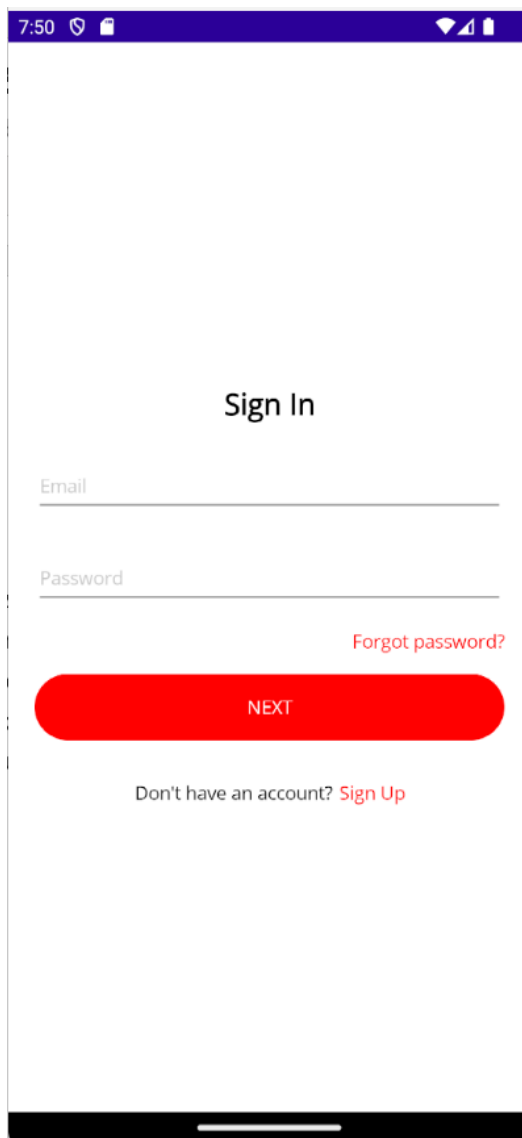
Favorite.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using SQLite;

namespace KFC_Menu.Models
{
    public class Favorite
    {
        [PrimaryKey, AutoIncrement]
        public int Id { get; set; }
        public int UserId { get; set; }
        public int ItemId { get; set; }
        public string ItemName { get; set; }
        public string ItemPrice { get; set; }
        public string ItemImage { get; set; }
        public string ItemDescription { get; set; }
        public string ItemCategory { get; set; }
    }
}
```

SIGN IN PAGE

SignInPage u aplikaciji KFC_Menu predstavlja formu za prijavu korisnika. XAML dio stranice definira bijelu pozadinu s grid layoutom koji sadrži natpis "Sign In", polja za unos emaila i lozinke, gumb "NEXT" za potvrdu prijave, te link "Forgot password?" i opciju za registraciju s tekстом "Don't have an account? Sign Up". U C# kodu, konstruktor inicijalizira stranicu i bazu podataka. Metoda NextClicked asinkrono provjerava korisničke podatke u bazi; ako su ispravni, korisnik se prijavljuje, postavlja kao trenutni korisnik i preusmjerava na MenuPage, a ako nisu, prikazuje se poruka o grešci. Metoda SignUpTapped navigira korisnika na stranicu za registraciju.



Slika 3. Sign In Page

```

using KFC_Menu.Models;
using KFC_Menu.Services;
namespace KFC_Menu;

public partial class SignInPage : ContentPage
{
    public SignInPage()
    {
        InitializeComponent();
        DatabaseServices.initializeDatabase();
    }

    private async void NextClicked(object sender, EventArgs e)
    {
        var user = await DatabaseServices.getUser(emailEntry.Text,
passwordEntry.Text);

        if (user != null)
        {
            await DatabaseServices.Instance.setLoggedInUser(user);
            DisplayAlert("Success", "Signed in successfully", "OK");
            Application.Current.MainPage = new NavigationPage(new
MenuPage());
            ((App)Application.Current).loginSuccessful();
        }
        else
        {
            DisplayAlert("Error", "Invalid email or password", "OK");
        }
    }

    private void SignUpTapped(object sender, EventArgs e)
    {
        Navigation.PushAsync(new SignUpPage());
    }
}

```

U konstruktoru klase inicijaliziraju se komponente stranice i baza podataka putem DatabaseServices.initializeDatabase(). Metoda NextClicked asinkrono provjerava unesene korisničke podatke (email i lozinku) u bazi; ako su podaci ispravni, korisnik se prijavljuje, postavlja kao trenutni korisnik, prikazuje se poruka o uspjehu, te se korisnik preusmjerava na MenuPage. Ako podaci nisu ispravni, prikazuje se poruka o grešci. Metoda SignUpTapped navigira korisnika na stranicu za registraciju SignUpPage.

XAML KOD:

```
<?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="KFC_Menu.SignInPage"
  Shell.NavBarIsVisible="False">

  <ContentPage.Content>
    <Grid BackgroundColor="White" Padding="20"
      RowDefinitions="*, auto, auto, *">
      <StackLayout Grid.Row="1" Spacing="15">
        <Label Text="Sign In"
          FontSize="Large"
          FontAttributes="Bold"
          TextColor="Black"
          HorizontalOptions="Center"/>
        <Entry Placeholder="Email"
          x:Name="emailEntry"
          Text=""
          Keyboard="Email"
          Margin="0,10,0,0"/>
        <Entry Placeholder="Password"
          x:Name="passwordEntry"
          IsPassword="True"
          Text=""
          Margin="0,10,0,0"/>
        <StackLayout Orientation="Horizontal" HorizontalOptions="End">
          <Label Text="Forgot password?"
            TextColor="red"
            VerticalOptions="Center"/>
        </StackLayout>
        <Button Text="NEXT"
          BackgroundColor="Red"
          TextColor="White"
          CornerRadius="25"
          HeightRequest="50"
          Clicked="NextClicked" />
      </StackLayout>
      <StackLayout Grid.Row="2" Orientation="Horizontal"
        HorizontalOptions="Center" Margin="30">
        <Label Text="Don't have an account?"/>
        <Label Text="Sign Up"
          TextColor="red"
          Margin="5,0,0,0" >
          <Label.GestureRecognizers>
            <TapGestureRecognizer Tapped="SignUpTapped" />
          </Label.GestureRecognizers>
        </Label>
      </StackLayout>
    </Grid>
  </ContentPage.Content>
</ContentPage>
```

Label sa imenom Sign In

Entry polje u koje se unosi email od profila

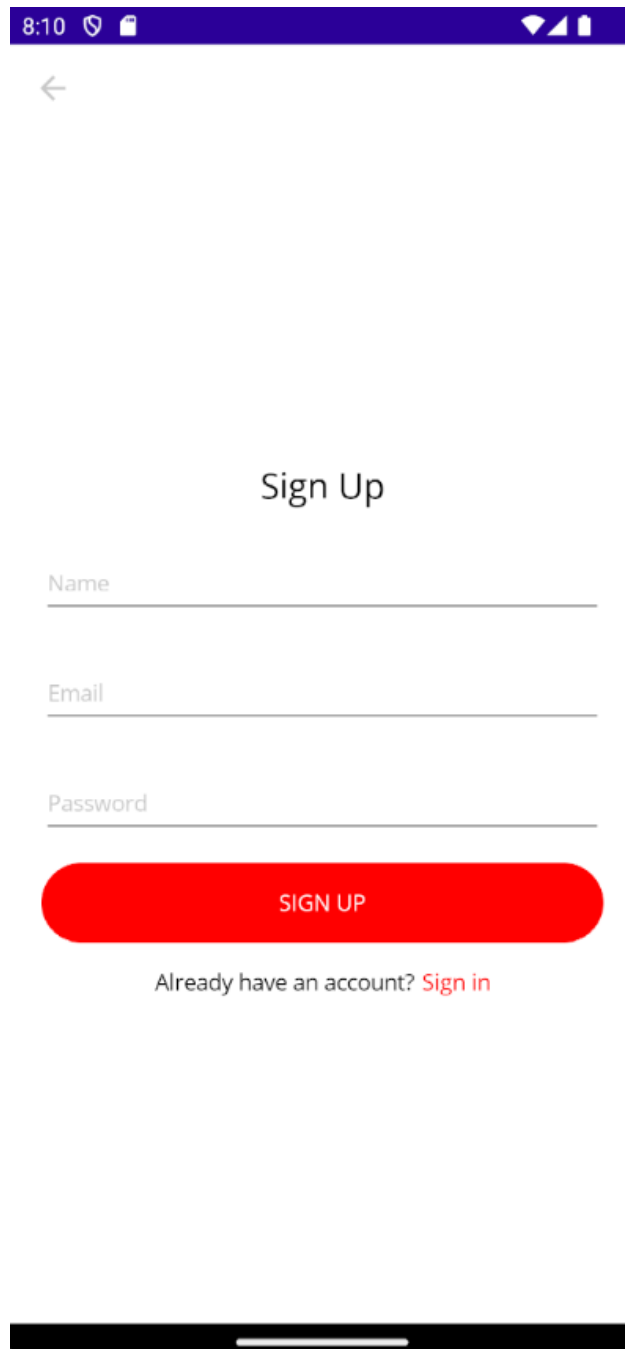
Entry polje u koje se unos password od profila

Button NEXT na koji kada se klikne provjerava da li podaci o tom korisniku postoje i zatim prebacuje na Menu Page

Sign Up label nad kojim je urađen TapGesture i kada se klikne na njega vodi na Sign up

SIGN UP PAGE

SignUpPage u aplikaciji KFC_Menu omogućava korisnicima registraciju novog računa. XAML dio definira izgled stranice koja sadrži natpis "Sign Up", polja za unos imena, emaila i lozinke, te gumb "SIGN UP". Tu je i opcija za prebacivanje na stranicu za prijavu s tekстом "Already have an account? Sign in". U C# kodu, konstruktor inicijalizira komponentu stranice. Metoda SignInTapped navigira korisnika na stranicu za prijavu (SignInPage). Metoda SignUpClicked kreira novi objekt korisnika s unesenim podacima, dodaje ga u bazu podataka pomoću DatabaseServices.addUser, prikazuje poruku o uspješnom kreiranju računa, te preusmjerava korisnika na stranicu za prijavu.



Slika 4. Sign Up Page

```

using KFC_Menu.Models;
using KFC_Menu.Services;

namespace KFC_Menu;

public partial class SignUpPage : ContentPage
{
    public SignUpPage()
    {
        InitializeComponent();
    }

    private void SignInTapped(object sender, EventArgs e)
    {
        Navigation.PushAsync(new SignInPage());
    }

    private void SignUpClicked(object sender, EventArgs e)
    {
        var user = new User
        {
            Name = nameEntry.Text,
            Email = emailEntry.Text,
            Password = passwordEntry.Text,
        };

        DatabaseServices.addUser(user);
        DisplayAlert("Success", "Account created successfully", "OK");
        Navigation.PushAsync(new SignInPage());
    }
}

```

Konstruktor `SignUpPage` inicijalizira komponentu stranice. Metoda `SignInTapped` se aktivira kada korisnik dodirne tekst "Sign in" i navigira korisnika na stranicu za prijavu (`SignInPage`). Metoda `SignUpClicked` se aktivira kada korisnik klikne na gumb "SIGN UP". Ova metoda kreira novi objekt korisnika s unesenim imenom, emailom i lozinkom, zatim dodaje korisnika u bazu podataka koristeći `DatabaseServices.addUser`. Nakon uspješne registracije prikazuje se poruka o uspjehu, a korisnik se preusmjerava na stranicu za prijavu.

XAML KOD:

```
<?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="KFC_Menu.SignUpPage"
  Shell.NavBarIsVisible="False">

  <ContentPage.Content>
    <Grid BackgroundColor="White" Padding="20"
      RowDefinitions="*, auto, auto, *"
      <StackLayout Grid.Row="1" Spacing="15">
        <Label Text="Sign Up"
          FontSize="Large"
          TextColor="Black"
          HorizontalOptions="Center"/>
        <Entry Placeholder="Name"
          x:Name="nameEntry"
          Text=""
          Margin="0,10,0,0" />
        <Entry Placeholder="Email"
          x:Name="emailEntry"
          Text=""
          Keyboard="Email"
          Margin="0,10,0,0" />
        <Entry Placeholder="Password"
          x:Name="passwordEntry"
          IsPassword="True"
          Text=""
          Margin="0,10,0,0"/>
        <Button Text="SIGN UP"
          BackgroundColor="Red"
          TextColor="White"
          CornerRadius="25"
          HeightRequest="50"
          Clicked="SignUpClicked"/>
      </StackLayout>
      <StackLayout Grid.Row="2" Orientation="Horizontal"
        HorizontalOptions="Center">
        <Label Text="Already have an account?"/>
        <Label Text="Sign in"
          TextColor="Red"
          Margin="5,0,0,0">
          <Label.GestureRecognizers>
            <TapGestureRecognizer Tapped="SignInTapped"/>
          </Label.GestureRecognizers>
        </Label>
      </StackLayout>
    </Grid>
  </ContentPage.Content>
</ContentPage>
```

Label sa imenom Sign Up

Entry polje u koje se unosi ime i prezime korisnika

Entry polje u koje se unosi email korisnika

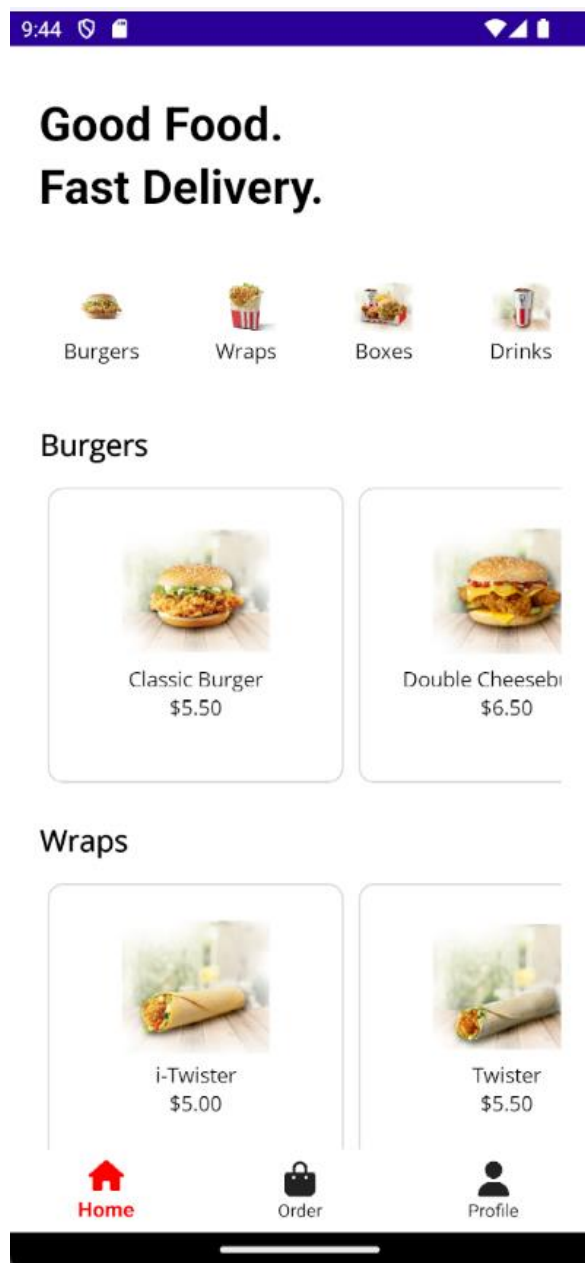
Entry polje u koje se unosi lozinka korisnika za profil

Button SIGN UP na koji kada se klikne, sačuva podatke o profilu u bazu i prebacuje korisnika na Sign In Page da se prijavi na profil

Label koji ima TapGesture i kada se klikne na njega odvodi na Sign In Page

MENU PAGE

Ova MenuPage stranica u aplikaciji KFC_Menu prikazuje jelovnik s različitim kategorijama proizvoda poput burgera, wrapova, kutija s jelima i pića. Gornji dio stranice sadrži zaglavlje s promotivnim tekstom i horizontalno pomičnu navigaciju za odabir kategorija, dok donji dio stranice prikazuje stavke izabrane kategorije. Podaci o stavkama su učitani u ObservableCollection, a svaka kategorija prikazuje svoje stavke pomoću CollectionView kontrola. Kad korisnik odabere kategoriju, stranica se pomiče do odgovarajuće sekcije. Klikom na stavku, korisnik se preusmjerava na detaljnu stranicu te stavke, dok klikom na gumb za dodavanje u favorite, stavka se dodaje u korisnikov popis favorita.



Slika 5. Food Menu Page

```
using System.Collections.ObjectModel;
using KFC_Menu.Models;
using KFC_Menu.Services;
using SQLite;
```

```
namespace KFC_Menu;
```

```
public partial class MenuPage : ContentPage
{
```

```
    public ObservableCollection<MenuItem> Burgers { get; set; }
    public ObservableCollection<MenuItem> Wraps { get; set; }
    public ObservableCollection<MenuItem> Boxes { get; set; }
    public ObservableCollection<MenuItem> Drinks { get; set; }
    public ProfilePage profilePage;
    public MenuPage()
    {
        InitializeComponent();

        profilePage = new ProfilePage();

        LoadData();
    }
}
```

```
private async void LoadData()
{
```

```
    Burgers = new ObservableCollection<MenuItem>
    {
        new MenuItem {Name = "Classic Burger", Price = "$5.50", Image =
"classic_burger.png", Description = "Uživajte u klasičnom burgeru s ukusnim
mesom, svježom salatom i kremastim umakom.", Category = "Burgers"},
        new MenuItem {Name = "Double Cheeseburger", Price = "$6.50", Image =
"double_cheeseburger.png", Description = "Dvostruki užitek s dvostrukim sirom i
sočnim mesom u svakom zalogaju.", Category = "Burgers"},
        new MenuItem {Name = "Grander Burger", Price = "$7", Image =
"grander_burger.png", Description = "Veličanstveni burger s većim komadom mesa za
prave gurmane.", Category = "Burgers"},
        new MenuItem {Name = "Mini Cheeseburger", Price = "$3.50", Image =
"mini_cheeseburger.png", Description = "Mali, ali moćan, s topljenim sirom i
sočnim mesom.", Category = "Burgers"},
        new MenuItem {Name = "Mini Chickenburger", Price = "$3.75", Image =
"mini_chickenburger.png", Description = "Ukusan mini burger s hrskavom piletinom
i svježim povrćem.", Category = "Burgers"},
        new MenuItem {Name = "Zinger Burger", Price = "$6.00", Image =
"zinger_burger.png", Description = "Pikantni Zinger Burger za ljubitelje
začinjene hrane.", Category = "Burgers"},
    };
}
```

```

Wraps = new ObservableCollection<MenuItem>
{
    new MenuItem {Name = "i-Twister", Price = "$5.00", Image =
    "i_twister.png", Description = "Svježi wrap s piletinom, povrćem i ukusnim
    umakom.", Category = "Wraps"},
    new MenuItem {Name = "Twister", Price = "$5.50", Image =
    "twister.png", Description = "Klasični Twister wrap s piletinom, salatam i
    rajčicom.", Category = "Wraps"},
    new MenuItem {Name = "Qurrito", Price = "$6.0", Image =
    "qurrito.png", Description = "Savršena kombinacija burrita i quesadille s
    piletinom i sirom.", Category = "Wraps"},
    new MenuItem {Name = "Boxmaster", Price = "$6.50", Image =
    "boxmaster.png", Description = "Izdašan wrap s hrskavom piletinom, sirom i
    hrskavim krompirom.", Category = "Wraps"},
};

```

```

Boxes = new ObservableCollection<MenuItem>
{
    new MenuItem {Name = "Classic Burger Box", Price = "$8.50", Image =
    "classic_burger_box.png", Description = "Klasična kutija s burgerom, krompirom i
    pićem.", Category = "Boxes"},
    new MenuItem {Name = "Double Cheeseburger Box", Price = "$9.50",
    Image = "double_cheeseburger_box.png", Description = "Kutija s dvostrukim
    cheeseburgerom, krompirom i pićem.", Category = "Boxes"},
    new MenuItem {Name = "Grander Box", Price = "$10.00", Image =
    "grander_box.png", Description = "Kutija s Grander Burgerom, krompirom i
    salatam.", Category = "Boxes"},
    new MenuItem {Name = "Tower Box", Price = "$10.50", Image =
    "tower_box.png", Description = "Impresivna kutija s velikim burgerom, krompirom i
    pićem.", Category = "Boxes"},
    new MenuItem {Name = "Twister Box", Price = "$9.00", Image =
    "twister_box.png", Description = "Kutija s Twister wrapom, krompirom i pićem.",
    Category = "Boxes"},
    new MenuItem {Name = "Zinger Box", Price = "$9.50", Image =
    "zinger_burger_box.png", Description = "Pikantna kutija s Zinger Burgerom,
    krompirom i pićem.", Category = "Boxes"},
};

```

```

Drinks = new ObservableCollection<MenuItem>
{
    new MenuItem {Name = "Coca Cola", Price = "$1.50", Image =
    "coca_cola.png", Description = "Osvježavajuće gazirano piće s okusom Cole.",
    Category = "Drinks"},
    new MenuItem {Name = "Fanta", Price = "$1.50", Image = "fanta.png",
    Description = "Slatko gazirano piće s okusom naranče.", Category = "Drinks"},
    new MenuItem {Name = "Sprite", Price = "$1.50", Image = "sprite.png",
    Description = "Osvježavajuće gazirano piće s okusom limuna i limete.", Category =
    "Drinks"},
    new MenuItem {Name = "Water", Price = "$1.00", Image = "voda.png",
    Description = "Čista, osvježavajuća voda za hidrataciju.", Category = "Drinks"},
    new MenuItem {Name = "Cappuccino", Price = "$2.50", Image =
    "cappuccino.png", Description = "Bogat i kremast cappuccino za ljubitelje kave.",
    Category = "Drinks"},
};

BindingContext = this;
}

```

```

public void CategoryClicked(object sender, EventArgs e)
{
    var frame = sender as Frame;
    var label = frame.Content as StackLayout;
    var category = (label.Children[1] as Label).Text;

    var yOffset = category switch
    {
        "Burgers" => 0,
        "Wraps" => BurgersSection.Height,
        "Boxes" => BurgersSection.Height + WrapsSection.Height,
        "Drinks" => BurgersSection.Height + WrapsSection.Height +
BoxesSection.Height,
        _ => 0
    };

    ContentScrollView.ScrollToAsync(0, yOffset, true);
}

```

```

private async void ItemTapped(object sender, EventArgs e)
{
    var frame = sender as Frame;
    var item = frame.BindingContext as MenuItem;

    if (item != null)
    {
        await Navigation.PushAsync(new ItemDetailsPage(item));
    }
}

private async void addToFavoritesClicked(object sender, EventArgs e)
{
    var button = sender as Button;
    var item = button.BindingContext as MenuItem;
    var loggedInUser = DatabaseServices.Instance.GetLoggedInUser();

    if (loggedInUser != null && item != null)
    {
        await DatabaseServices.Instance.AddFavorite(loggedInUser.Id, item);
        await DisplayAlert("Favorites", $"{item.Name} has been added to your
favorites", "OK");
    }
}
}

```

```

public class MenuItem
{
    [PrimaryKey, AutoIncrement]
    public int Id { get; set; }
    public string Name { get; set; }
    public string Price { get; set; }
    public string Image { get; set; }
    public string Description { get; set; }
    public string Category { get; set; }
}

```

Ovaj kod predstavlja MenuPage klasu u aplikaciji KFC_Menu koja prikazuje jelovnik restorana KFC s različitim kategorijama proizvoda kao što su burgeri, wrapovi, kutije s jelima i pića. Podaci o stavkama u meniju učitavaju se u ObservableCollection kolekcije koje se vežu za korisničko sučelje. Kategorije se prikazuju u horizontalno pomičnoj navigaciji, a stavke svake kategorije u horizontalnim popisima unutar ScrollView kontrole. Klikom na stavke u meniju korisnik se preusmjerava na stranicu s detaljima odabrane stavke. Također, postoji funkcionalnost dodavanja stavki u favorite, gdje se korisniku prikazuje obavijest nakon uspjeha dodavanja.

XAML KOD:

```
<?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="KFC_Menu.MenuPage"
    Shell.NavBarIsVisible="False"
    Shell.TabBarBackgroundColor="Gainsboro"
    Shell.TabBarTitleColor="Red">

    <ContentPage.Content>
        <Grid RowDefinitions="Auto, Auto, *">

            <!-- Header Text -->
            <StackLayout Grid.Row="0" Padding="20,10,20,0" Margin="0,20,0,20">
                <Label Text="Good Food." FontSize="32" FontFamily="Poppins" FontAttributes="Bold"
                    HorizontalOptions="Start"/>
                <Label Text="Fast Delivery." FontSize="32" FontFamily="Poppins" FontAttributes="Bold"
                    HorizontalOptions="Start"/>
            </StackLayout>

            <!-- Category Navigation -->
            <StackLayout Grid.Row="1">
                <ScrollView Orientation="Horizontal" BackgroundColor="White"
                    HeightRequest="80" Margin="0,10,0,0">
                    <HorizontalStackLayout Padding="10">
                        <Frame Margin="5" BorderColor="White"
                            CornerRadius="10" BackgroundColor="white" HeightRequest="100">
                            <StackLayout VerticalOptions="Center">
                                <Image Source="burger_category.png" HeightRequest="40" WidthRequest="40"/>
                                <Label Text="Burgers" HorizontalOptions="Center"/>
                            </StackLayout>
                            <Frame.GestureRecognizers>
                                <TapGestureRecognizer Tapped="CategoryClicked" CommandParameter="Burgers"/>
                            </Frame.GestureRecognizers>
                        </Frame>


```

Labeli sa porukama na vrhu stranice

Frame sa kategorijom za burgere sa TapGesture koji klikom na nju skrola do te kategorije u meniju

```

<Frame Margin="5" BorderColor="White"
    CornerRadius="10" BackgroundColor="White" HeightRequest="100">
    <StackLayout VerticalOptions="Center">
        <Image Source="wraps_category.png" HeightRequest="40" WidthRequest="40"/>
        <Label Text="Wraps" HorizontalOptions="Center"/>
    </StackLayout>
    <Frame.GestureRecognizers>
        <TapGestureRecognizer Tapped="CategoryClicked" CommandParameter="Wraps"/>
    </Frame.GestureRecognizers>
</Frame>

```

Frame sa kategorijom za wraps sa TapGesture koji klikom na nju skrola do te kategorije u meniju

```

<Frame Margin="5" BorderColor="White"
    CornerRadius="10" BackgroundColor="White" HeightRequest="100">
    <StackLayout VerticalOptions="Center">
        <Image Source="boxes_category.png" HeightRequest="40" WidthRequest="40"/>
        <Label Text="Boxes" HorizontalOptions="Center"/>
    </StackLayout>
    <Frame.GestureRecognizers>
        <TapGestureRecognizer Tapped="CategoryClicked" CommandParameter="Boxes"/>
    </Frame.GestureRecognizers>
</Frame>

```

Frame sa kategorijom za boxes sa TapGesture koji klikom na nju skrola do te kategorije u meniju

```

<Frame Margin="5" BorderColor="White"
    CornerRadius="10" BackgroundColor="White" HeightRequest="100">
    <StackLayout VerticalOptions="Center">
        <Image Source="drinks_category.png" HeightRequest="40" WidthRequest="40"/>
        <Label Text="Drinks" HorizontalOptions="Center"/>
    </StackLayout>
    <Frame.GestureRecognizers>
        <TapGestureRecognizer Tapped="CategoryClicked" CommandParameter="Drinks"/>
    </Frame.GestureRecognizers>
</Frame>

```

```

</HorizontalStackLayout>
</ScrollView>
</StackLayout>

```

Frame sa kategorijom za drinks sa TapGesture koji klikom na nju skrola do te kategorije u meniju

```

<!-- Content Area -->
<ScrollView x:Name="ContentScrollView" Grid.Row="2" Margin="0,10,0,0">
    <VerticalStackLayout Padding="10">

        <!-- Burgers -->
        <StackLayout x:Name="BurgersSection" Padding="10">
            <Label Text="Burgers" FontSize="20" FontAttributes="Bold" Margin="0,0,0,10"/>
            <CollectionView ItemsSource="{Binding Burgers}" ItemsLayout="HorizontalList">
                <CollectionView.ItemTemplate>
                    <DataTemplate>
                        <Frame BorderColor="LightGray" Margin="5"
                            CornerRadius="10" BackgroundColor="White" WidthRequest="200"
                            HeightRequest="200">
                            <StackLayout>
                                <Image Source="{Binding Image}" HeightRequest="100"
                                    WidthRequest="100" HorizontalOptions="Center"/>
                                <Label Text="{Binding Name}" HorizontalOptions="Center"/>
                                <Label Text="{Binding Price}" HorizontalOptions="Center"/>
                            </StackLayout>
                            <Frame.GestureRecognizers>
                                <TapGestureRecognizer Tapped="ItemTapped"/>
                            </Frame.GestureRecognizers>
                        </Frame>
                    </DataTemplate>
                </CollectionView.ItemTemplate>
            </CollectionView>
        </StackLayout>

        <!-- Wraps -->
        <StackLayout x:Name="WrapsSection" Padding="10">
            <Label Text="Wraps" FontSize="20" FontAttributes="Bold" Margin="0,0,0,10"/>
            <CollectionView ItemsSource="{Binding Wraps}" ItemsLayout="HorizontalList">
                <CollectionView.ItemTemplate>
                    <DataTemplate>
                        <Frame BorderColor="LightGray" Margin="5"
                            CornerRadius="10" BackgroundColor="White" WidthRequest="200"
                            HeightRequest="200">
                            <StackLayout>
                                <Image Source="{Binding Image}" HeightRequest="100"
                                    WidthRequest="100" HorizontalOptions="Center"/>
                                <Label Text="{Binding Name}" HorizontalOptions="Center"/>
                                <Label Text="{Binding Price}" HorizontalOptions="Center"/>
                            </StackLayout>
                        </DataTemplate>
                    </CollectionView.ItemTemplate>
                </CollectionView>
            </StackLayout>
        </ScrollView>

```

CollectionView za burgere

Osnovni podaci za svaki od MenuItem u kategoriji burgera prikazani preko Binding

TapGesture za klik na bilo koji od MenuItem za više detalja o tom itemu

CollectionView za wraps

Osnovni podaci za svaki od MenuItem u kategoriji wraps prikazani preko Binding

```

<Frame.GestureRecognizers>
  <TapGestureRecognizer Tapped="ItemTapped"/>
</Frame.GestureRecognizers>
</Frame>
</DataTemplate>
</CollectionView.ItemTemplate>
</CollectionView>
</StackLayout>

<!-- Boxes -->
<StackLayout x:Name="BoxesSection" Padding="10">
  <Label Text="Boxes" FontSize="20" FontAttributes="Bold" Margin="0,0,0,10"/>
  <CollectionView ItemsSource="{Binding Boxes}" ItemsLayout="HorizontalList">
    <CollectionView.ItemTemplate>
      <DataTemplate>
        <Frame BorderColor="LightGray" Margin="5"
          CornerRadius="10" BackgroundColor="White" WidthRequest="200"
          HeightRequest="200">
          <StackLayout>
            <Image Source="{Binding Image}" HeightRequest="100"
              WidthRequest="100" HorizontalOptions="Center"/>
            <Label Text="{Binding Name}" HorizontalOptions="Center"
              HorizontalTextAlignment="Center"/>
            <Label Text="{Binding Price}" HorizontalOptions="Center"/>
          </StackLayout>
          <Frame.GestureRecognizers>
            <TapGestureRecognizer Tapped="ItemTapped"/>
          </Frame.GestureRecognizers>
        </DataTemplate>
      </CollectionView.ItemTemplate>
    </CollectionView>
  </StackLayout>

  <!-- Drinks -->
  <StackLayout x:Name="DrinksSection" Padding="10">
    <Label Text="Drinks" FontSize="20" FontAttributes="Bold" Margin="0,0,0,10"/>
    <CollectionView ItemsSource="{Binding Drinks}" ItemsLayout="HorizontalList">
      <CollectionView.ItemTemplate>
        <DataTemplate>
          <Frame BorderColor="LightGray" Margin="5"
            CornerRadius="10" BackgroundColor="White" WidthRequest="200"
            HeightRequest="200">
            <StackLayout>

```

TapGesture za klik na bilo koji od Menultem za više detalja o tom itemu

CollectionView za boxes

Osnovni podaci za svaki od Menultem u kategoriji boxes prikazani preko Binding

TapGesture za klik na bilo koji od Menultem za više detalja o tom itemu

CollectionView za drinks


```

        <Image Source="{Binding Image}" HeightRequest="100"
WidthRequest="100" HorizontalOptions="Center"/>
        <Label Text="{Binding Name}" HorizontalOptions="Center"/>
        <Label Text="{Binding Price}" HorizontalOptions="Center"/>
    </StackLayout>
    <Frame.GestureRecognizers>
        <TapGestureRecognizer Tapped="ItemTapped"/>
    </Frame.GestureRecognizers>
</Frame>
</DataTemplate>
</CollectionView.ItemTemplate>
</CollectionView>
</StackLayout>
</VerticalStackLayout>
</ScrollView>
</Grid>
</ContentPage.Content>
</ContentPage>

```

Osnovni podaci za svaki
od MenuItem u
kategoriji boxes
prikazani preko Binding

TapGesture za klik na bilo koji od
MenuItem za više detalja o tom
itemu

ITEM DETAILS PAGE

Ova stranica ItemDetailsPage u aplikaciji KFC_Menu prikazuje detalje pojedinačne stavke iz menija. Korisnik može videti sliku, naziv, opis i količinu stavke, kao i dodati je u omiljene ili u korpu. Stranica sadrži dugmad za povećanje ili smanjenje količine i za dodavanje stavke u korpu, pri čemu se ukupna cena automatski ažurira. Korisnici mogu kliknuti na dugme za povratak na prethodnu stranicu ili označiti stavku kao omiljenu. Aplikacija proverava da li je korisnik prijavljen, omogućava dodavanje stavki u korpu i omiljene, te prikazuje odgovarajuće obaveštenja o uspešnosti tih akcija.



Slika 6. Item Details Page

```
using KFC_Menu.Models;
using KFC_Menu.Services;
using static KFC_Menu.OrderPage;
```

```
namespace KFC_Menu;
```

```
public partial class ItemDetailsPage : ContentPage
{
```

```
    private MenuItem currentItem;
    private int quantity;
    private User loggedInUser;

    public ItemDetailsPage(MenuItem item)
    {
        InitializeComponent();
        currentItem = item;
        quantity = 1;
        BindingContext = currentItem;

        ItemQuantity.Text = quantity.ToString();
        UpdateAddToCartButtonText();
        LoadLoggedInUser();
    }
```

```
    public async void LoadLoggedInUser()
    {
        loggedInUser = await DatabaseServices.Instance.getLoggedInUser();
        ItemQuantity.Text = quantity.ToString();
        UpdateFavoriteButtonState();
    }
    private async void BackButtonClicked(object sender, EventArgs e)
    {
        await Navigation.PopAsync();
    }
```

```
    private async void FavoriteButtonClicked(object sender, EventArgs e)
    {
        try
        {
            loggedInUser = await
DatabaseServices.Instance.getLoggedInUser();

            if (loggedInUser != null)
            {
                var favoriteItem = await
DatabaseServices.Instance.getFavoriteItem(loggedInUser.Id, currentItem.Id);
                if (favoriteItem == null)
                {
                    var favorite = new Favorite
                    {
                        UserId = loggedInUser.Id,
                        ItemId = currentItem.Id,
                        ItemName = currentItem.Name,
                        ItemPrice = currentItem.Price,
                        ItemImage = currentItem.Image,
                        ItemDescription = currentItem.Description,
                        ItemCategory = currentItem.Category
                    };

                    await DatabaseServices.Instance.addFavorite(loggedInUser.Id,
currentItem);

                    await DisplayAlert("Added", $"{currentItem.Name} has been
added to favorites", "OK");
                }
            }
        }
        catch { }
    }
```

```

        UpdateFavoriteButtonState();
    }
    else
    {
        await
DatabaseServices.Instance.removeFromFavorites(loggedInUser.Id, currentItem.Id);
        await DisplayAlert("Removed", $"{currentItem.Name} has been
removed from favorites", "OK");
        UpdateFavoriteButtonState();
    }
}
else
{
    await DisplayAlert("Error", "User is not logged in.",
"OK");
}
}
catch (ObjectDisposedException ex)
{
    await DisplayAlert("Error", "An error occurred: " +
ex.Message, "OK");
}
catch (Exception ex)
{
    await DisplayAlert("Error", "An unexpected error occurred: " +
ex.Message, "OK");
}
}
}

```

```

private void DecreaseQuantityClicked(object sender, EventArgs e)
{
    if (quantity > 1)
    {
        quantity--;
        ItemQuantity.Text = quantity.ToString();
        UpdateAddToCartButtonText();
    }
}

```

```

private void IncreaseQuantityClicked(object sender, EventArgs e)
{
    quantity++;
    ItemQuantity.Text = quantity.ToString();
    UpdateAddToCartButtonText();
}

```

```

private void UpdateAddToCartButtonText()
{
    string priceString = currentItem.Price.Replace("$", "").Trim();
    if (decimal.TryParse(priceString, out decimal price))
    {
        decimal total = quantity * price;
        AddToCartButton.Text = $"Add to cart  ${total}";
    }
}

```

```

private async void UpdateFavoriteButtonState()
{
    if (loggedInUser == null) return;

    var favoriteItem = await
DatabaseServices.Instance.getFavoriteItem(loggedInUser.Id, currentItem.Id);
    if (favoriteItem == null)
    {
        FavoriteButton.Source = "heart_regular.svg";
    }
    else
    {
        FavoriteButton.Source = "heart_solid.svg";
    }
}

```

```

private async void AddToCartButtonClicked(object sender, EventArgs e)
{
    loggedInUser = await DatabaseServices.Instance.getLoggedInUser();
    if (loggedInUser == null)
    {
        await DisplayAlert("Error", "You need to be logged in to add
items to the cart.", "OK");
        return;
    }

    var orderItem = new OrderItem
    {
        UserId = loggedInUser.Id,
        Id = currentItem.Id,
        OrderDate = DateTime.Now,
        ItemName = currentItem.Name,
        ItemImage = currentItem.Image,
        Price = decimal.Parse(currentItem.Price.Replace("$",
"").Trim()),
        Quantity = quantity
    };
    await DatabaseServices.Instance.addOrderItem(orderItem);

    await DisplayAlert("Added", $"{currentItem.Name} has been added to
your order", "OK");
}
}

```

ItemDetailsPage je stranica koja prikazuje detalje odabrane stavke menija. Konstruktor inicijalizuje stranicu sa stavkom menija, postavlja početnu količinu na 1 i prikazuje osnovne informacije o stavci. Metoda LoadLoggedInUser asinhrono učitava informacije o prijavljenom korisniku i ažurira UI komponente. Metode za dugmad omogućavaju korisniku da se vrati unazad, dodaje ili uklanja stavku iz omiljenih, menja količinu stavke i dodaje je u korpu, uz ažuriranje UI u skladu sa izmenama. Metoda OnAppearing osigurava ažuriranje podataka o korisniku pri prikazivanju stranice.

XAML KOD:

```
<?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="KFC_Menu.ItemDetailsPage">
    <ContentPage.Content>
        <Grid RowDefinitions="Auto, *, *">
            <!--Red Bacground with Back and Favorite Buttons -->
            <StackLayout BackgroundColor="Red" Padding="20" HeightRequest="250"
                VerticalOptions="Start">
                <Grid ColumnDefinitions="*, *" ColumnSpacing="250">
                    <Frame WidthRequest="50" HeightRequest="50" CornerRadius="25"
                        BackgroundColor="White"
                        Padding="10" HorizontalOptions="Start"
                        VerticalOptions="Start">
                        <ImageButton Source="angle_left_solid.svg"
                            Clicked="BackButtonClicked" WidthRequest="30"
                                HeightRequest="30" HorizontalOptions="Start"
                                VerticalOptions="Start"/>
                    </Frame>
                    <Frame Grid.Column="1" WidthRequest="50" HeightRequest="50"
                        CornerRadius="25" BackgroundColor="White"
                        Padding="10" HorizontalOptions="Start"
                        VerticalOptions="Start">
                        <ImageButton x:Name="FavoriteButton" WidthRequest="30"
                            HeightRequest="30" HorizontalOptions="End"
                            Clicked="FavoriteButtonClicked"/>
                    </Frame>
                </Grid>
            </StackLayout>
        </Grid>
    </ContentPage.Content>
</ContentPage>
```

Prikaz ImageButtona za
vracanje na prethodnu
stranicu i za dodavanje
Menuitem u Favorites

```
<!-- Item Details -->
<AbsoluteLayout Grid.Row="1">
    <Image Source="{Binding Image}" HeightRequest="250"
        HorizontalOptions="Center"
        VerticalOptions="CenterAndExpand"
        AbsoluteLayout.LayoutFlags="All" AbsoluteLayout.LayoutBounds="0.5, 0.3, -1, -1"/>
    <VerticalStackLayout Padding="10" Spacing="10"
        AbsoluteLayout.LayoutFlags="All" AbsoluteLayout.LayoutBounds="0, 0.9, 1, 0">
        <Label Text="{Binding Name}" FontSize="24"
            FontAttributes="Bold" HorizontalOptions="Center"/>
        <Label Text="{Binding Description}" FontSize="18"
            HorizontalOptions="Center" HorizontalTextAlignment="Center"
            VerticalTextAlignment="Center"/>
    </VerticalStackLayout>
</AbsoluteLayout>
```

Prikaz osnovnih detalja o
Menuitem koristeći
AbsoluteLayout i Binding

```

<!-- Quantity and Add to Cart Buttons -->
<HorizontalStackLayout Grid.Row="2" Spacing="10"
HorizontalOptions="Center" VerticalOptions="End" Margin="0,0,0,20">
    <Button Text="-" Grid.Column="0"
Clicked="DecreaseQuantityClicked" WidthRequest="40" FontSize="15"
HeightRequest="40" BackgroundColor="SlateGray"
TextColor="White" CornerRadius="20"/>
    <Label x:Name="ItemQuantity" Text="1" HorizontalOptions="Center"
VerticalOptions="Center" FontSize="18" Padding="10"/>
    <Button Text="+" Clicked="IncreaseQuantityClicked"
WidthRequest="40" FontSize="15"
HeightRequest="40" BackgroundColor="SlateGray"
TextColor="White" CornerRadius="20"/>
    <Button x:Name="AddToCartButton" Text="Add to cart"
BackgroundColor="Red"
TextColor="White" HorizontalOptions="End"
Clicked="AddToCartButtonClicked"
WidthRequest="150" CornerRadius="20"
FontAttributes="Bold" Margin="40,0,0,0"/>
</HorizontalStackLayout>
</Grid>
</ContentPage.Content>
</ContentPage>

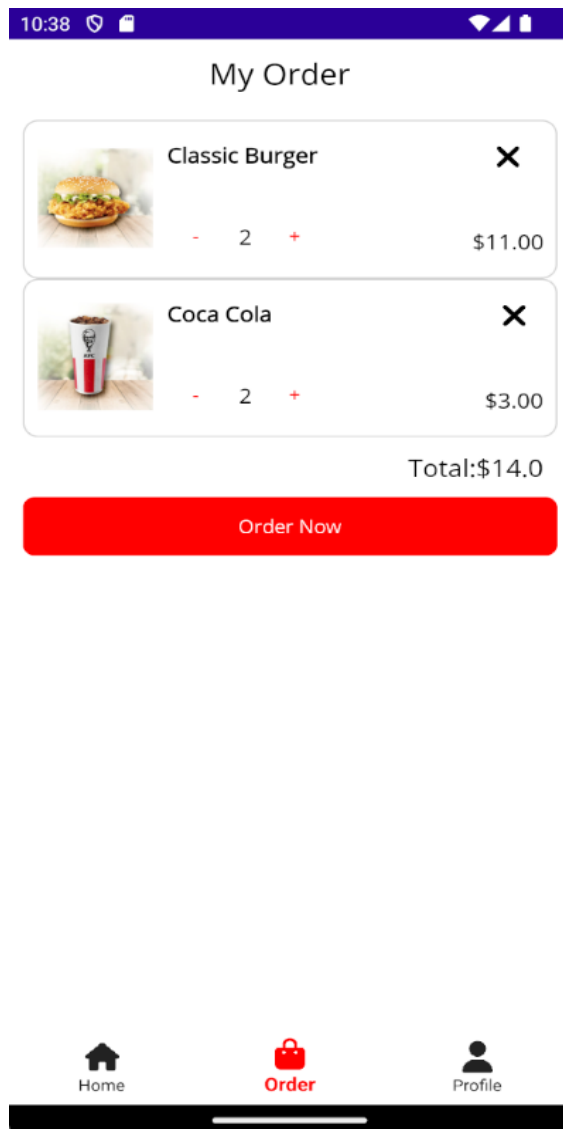
```

Add to Cart Button koji
dodaje MenuItem u Order

Button za smanjenje i
povećavanje broja porcija tog
MenuItem za dodavanje u Order

ORDER PAGE

OrderPage je stranica koja omogućava korisniku da pregleda svoju trenutnu narudžbinu, manipuliše količinama stavki, uklanja stavke iz narudžbine i plasira konačnu narudžbu. Prikazuje sve stavke u kolekciji sa njihovim imenom, slikom, količinom, ukupnom cenom i opcijom za brisanje. Korisnik može povećavati ili smanjivati količinu stavki što automatski ažurira ukupnu cenu narudžbine. Nakon što korisnik potvrdi narudžbu klikom na dugme "Order Now", narudžba se evidentira u bazi podataka, a korisniku se prikazuje potvrda da je narudžba uspešno primljena, nakon čega se lista narudžbine čisti za novu sesiju. Stranica takođe osigurava ažuriranje prikaza kada se korisnik prijavi ili odjavi.



Slika 7. Order Page


```
using System.Collections.ObjectModel;
using KFC_Menu.Models;
using KFC_Menu.Services;
using SQLite;
```

```
namespace KFC_Menu;
```

```
public partial class OrderPage : ContentPage
{
```

```
    private static ObservableCollection<OrderItem> orderItems;
    private User loggedInUser;
    private int quantity;
    public OrderPage()
    {
        InitializeComponent();
        LoadLoggedInUser();
        quantity = 1;
        BindingContext = this;
    }
```

```
    private async void LoadLoggedInUser()
    {
        loggedInUser = await DatabaseServices.Instance.getLoggedInUser();
        if (loggedInUser != null)
        {
            LoadOrderItems();
        }
    }
```

```
    private async void LoadOrderItems()
    {
        var items = await
DatabaseServices.Instance.getOrderItem(loggedInUser.Id);
        BindingContext = items;
        orderItems = new ObservableCollection<OrderItem>(items);
        OrderCollectionView.ItemsSource = orderItems;
        UpdateTotalPrice();
    }
```

```
    private void UpdateTotalPrice()
    {
        decimal totalPrice = orderItems.Sum(item => item.Quantity *
item.Price);
        TotalPriceLabel.Text = $"${totalPrice}";
    }
```

```
    private async void IncreaseQuantityClicked(object sender, EventArgs e)
    {
        var button = sender as Button;
        var orderItem = button.BindingContext as OrderItem;
        orderItem.Quantity++;
        await DatabaseServices.Instance.updateOrderItem(orderItem);
        UpdateTotalPrice();
    }
```

```

private async void DecreaseQuantityClicked(object sender, EventArgs e)
{
    var button = sender as Button;
    var orderItem = button.BindingContext as OrderItem;
    if (orderItem.Quantity > 1)
    {
        orderItem.Quantity--;
        await DatabaseServices.Instance.updateOrderItem(orderItem);
        UpdateTotalPrice();
    }
}

```

```

private async void RemoveItemClicked(object sender, EventArgs e)
{
    var button = sender as ImageButton;
    var orderItem = button.BindingContext as OrderItem;
    await DatabaseServices.Instance.removeOrderItem(orderItem.Id);
    orderItems.Remove(orderItem);
    UpdateTotalPrice();
}

```

```

private async void OrderClicked(object sender, EventArgs e)
{
    var newOrder = new Order
    {
        UserId = loggedInUser.Id,
        OrderDate = DateTime.Now,
        TotalPrice = orderItems.Sum(item => item.Quantity *
item.Price)
    };

    await DatabaseServices.Instance.addOrder(newOrder);

    await DisplayAlert("Order", "Your order has been successfully placed
and will be delivered soon.", "OK");

    orderItems.Clear();
    UpdateTotalPrice();
}

```

```

public class OrderItem
{
    [PrimaryKey, AutoIncrement]
    public int Id { get; set; }
    public int UserId { get; set; }
    public int OrderId { get; set; }
    public int ItemId { get; set; }
    public DateTime OrderDate { get; set; }
    public string ItemName { get; set; }
    public string ItemImage { get; set; }
    public decimal Price { get; set; }
    public int Quantity { get; set; }
    public decimal TotalPrice => Price * Quantity;
}
}

```

Fajl `OrderPage.xaml.cs` implementira logiku koja upravlja funkcionalnostima prikazanim na `OrderPage.xaml` stranici. Po učitavanju stranice, inicijalizuje se prijavljeni korisnik i učitavaju se stavke njegove trenutne narudžbine iz baze podataka. `LoadLoggedInUser` metoda proverava da li postoji prijavljeni korisnik i, ako postoji, poziva `LoadOrderItems` da bi se prikazale sve stavke narudžbine korisnika.

Metoda `LoadOrderItems` dobavlja stavke narudžbine iz baze podataka koristeći korisnikov ID, nakon čega se te stavke postavljaju kao izvor podataka za `OrderCollectionView`, koji je prikazan u `OrderPage.xaml`. Takođe se računa i ažurira ukupna cena narudžbine pomoću `UpdateTotalPrice` metode, koja sabira cene svih stavki množenih sa njihovim količinama.

Korisnik može interaktivno manipulirati stavkama u narudžbini: povećavati ili smanjivati količinu svake stavke koristeći odgovarajuće dugmiće, ažurirajući automatski ukupnu cenu. Takođe postoji mogućnost uklanjanja pojedinačnih stavki iz narudžbine klikom na "x" dugme pored svake stavke. Kada korisnik klikne na dugme "Order Now", kreira se nova narudžba sa podacima o korisniku, trenutnom datumu i ukupnoj ceni svih stavki u narudžbini. Nova narudžba se dodaje u bazu podataka, a korisniku se prikazuje potvrda uspešno postavljene narudžbine, nakon čega se lista narudžbine čisti za sledeću sesiju ili novu narudžbinu. Metode su takođe sinhronizovane sa događajima `OnAppearing` i `OnDisappearing` kako bi osigurale ažuriranje prikaza pri promeni stanja stranice.

XAML KOD:

```
<?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="KFC_Menu.OrderPage"
    Shell.NavBarIsVisible="False">
    <ContentPage.Content>
        <StackLayout Padding="10">
            <Label Text="My Order" FontSize="Large"
                VerticalTextAlignment="Center" Margin="130,0,0,20"/>

            <CollectionView x:Name="OrderCollectionView">
                <CollectionView.ItemTemplate>
                    <DataTemplate>
                        <Frame BorderColor="LightGray" CornerRadius="10"
                            BackgroundColor="White" Padding="10">
                            <Grid ColumnDefinitions="90, *, Auto"
                                RowDefinitions="Auto, Auto">
                                    <Image Source="{Binding ItemImage}"
                                        HeightRequest="90" WidthRequest="90" Grid.RowSpan="2" Margin="0,0,20,0"/>
                                    <StackLayout Grid.Column="1" Spacing="20">
                                        <Label Text="{Binding ItemName}"
                                            FontSize="16" FontAttributes="Bold" Margin="0,5,0,10"/>
                                        <StackLayout Orientation="Horizontal"
                                            Spacing="10">
                                            <Button Text="-" WidthRequest="40"
                                                HeightRequest="40" BackgroundColor="White" TextColor="Red"
                                                Clicked="DecreaseQuantityClicked" />
                                            <Label x:Name="ItemQuantity"
                                                Text="{Binding Quantity}" FontSize="16" VerticalOptions="Center"
                                                HorizontalOptions="Center"/>
                                            <Button Text="+" WidthRequest="40"
                                                HeightRequest="40" BackgroundColor="White" TextColor="Red"
                                                Clicked="IncreaseQuantityClicked" />
                                        </StackLayout>
                                    </StackLayout>
                                </Grid>
                            </Frame>
                        </DataTemplate>
                    </CollectionView.ItemTemplate>
                </CollectionView>

                <StackLayout Grid.Column="2" Grid.RowSpan="2"
                    VerticalOptions="CenterAndExpand" Spacing="40" HorizontalOptions="End">
                    <ImageButton Source="xmark_solid.svg"
                        WidthRequest="25" HeightRequest="25" Clicked="RemoveItemClicked"/>
                    <Label Text="{Binding TotalPrice,
                        StringFormat='{0:C}'}" FontSize="16" VerticalOptions="Center"/>
                </StackLayout>
            </Grid>
        </StackLayout>

        <StackLayout Orientation="Horizontal" HorizontalOptions="End"
            Padding="10,10">
            <Label Text="Total:" FontSize="Medium"/>
            <Label x:Name="TotalPriceLabel" FontSize="Medium"/>
        </StackLayout>

        <Button Text="Order Now" BackgroundColor="Red" TextColor="White"
            Clicked="OrderClicked"/>
    </ContentPage.Content>
</ContentPage>
```

Prikaz slike, imena i broja porcija za MenuItems u Orderu preko Binding svojstva kao i - i + znaka za povećavanje i smanjivanje broja porcija

ImageButton za brisanje MenuItem iz Ordera i TotalPrice tog MenuItem za taj broj porcija

Ukupna cijena kompletne narudžbe

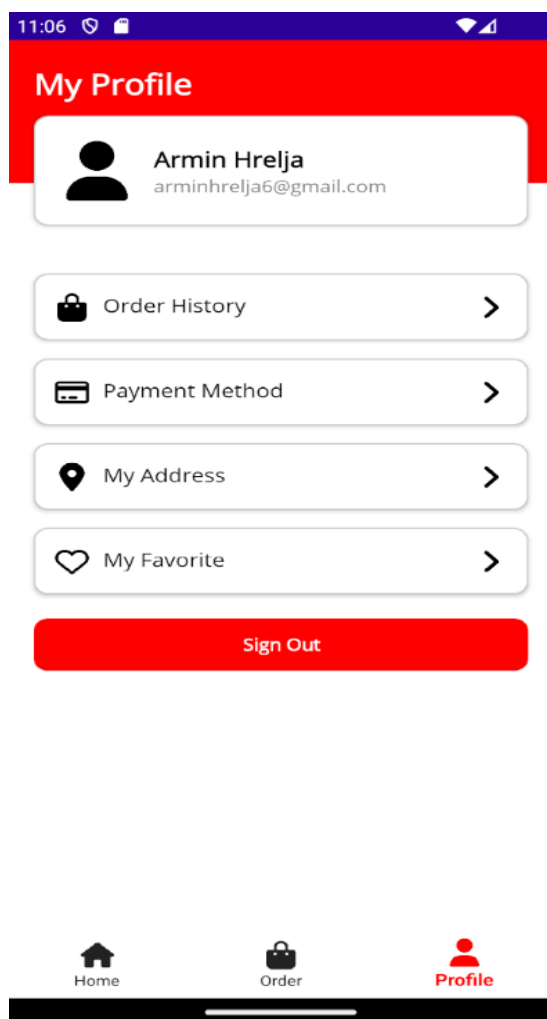
Order Now Button za kreiranje narudžbe i prikaz poruke da je narudžba uspješna

PROFILE PAGE

ProfilePage.xaml.cs implementira funkcionalnosti povezane sa prikazom korisničkog profila na stranici ProfilePage.xaml. Prilikom inicijalizacije stranice, učitava se profil prijavljenog korisnika kroz LoadUserProfile metodu koja koristi DatabaseServices.Instance.getLoggedInUser() za dobavljanje podataka iz baze podataka. Ako je korisnik uspešno prijavljen, postavljaju se podaci o korisniku (ime i email) na odgovarajuće Label kontrole, a loggedInUser se takođe postavlja kao BindingContext stranice radi upotrebe podataka iz databaze u XAML-u.

Na stranici su implementirane opcije kao što su "Order History", "Payment Method", "My Address" i "My Favorite", svaka u okviru sa odgovarajućim ikonama i nazivom. Klikom na "My Favorite", korisnik se preusmerava na FavoritesPage. Klikom na dugme "Sign Out", korisnik se odjavljuje pomoću DatabaseServices.Instance.signOutUser() metode, nakon čega se korisnik preusmerava na početnu stranicu za prijavu (SignInPage). Ako korisnik nije prijavljen prilikom učitavanja stranice, automatski se preusmerava na stranicu za prijavu kako bi mogao da se prijavi ili registruje.

Ova stranica pruža korisnicima pregled njihovog profila, omogućava im da pregledaju omiljene stavke, izvrše odjavu i pristupe drugim opcijama vezanim za korisnički nalog.



Slika 8. Profile Page

```
using System.Collections.ObjectModel;
using KFC_Menu.Models;
using KFC_Menu.Services;
```

```
namespace KFC_Menu;
```

```
public partial class ProfilePage : ContentPage
{
```

```
    private User loggedInUser;
    public ObservableCollection<MenuItem> Favorites { get; set; }
    public ProfilePage()
    {
        InitializeComponent();
        LoadUserProfile();
    }
```

```
    private async void LoadUserProfile()
    {
        loggedInUser = await DatabaseServices.Instance.getLoggedInUser();
        if (loggedInUser != null)
        {
            nameLabel.Text = loggedInUser.Name;
            emailLabel.Text = loggedInUser.Email;
            BindingContext = loggedInUser;
        }
        else
        {
            await Navigation.PushAsync(new SignInPage());
        }
    }
```

```
    private async void MyFavoritesTapped(object sender, EventArgs e)
    {
        Navigation.PushAsync(new FavoritesPage());
    }

    private async void SignOutClicked(object sender, EventArgs e)
    {
        await DatabaseServices.Instance.signOutUser();
        Application.Current.MainPage = new NavigationPage(new SignInPage());
    }
}
```

Ovaj fajl ProfilePage.xaml.cs implementira funkcionalnosti povezane sa prikazom korisničkog profila i upravljanjem korisničkim akcijama u okviru aplikacije. Prilikom inicijalizacije stranice u konstruktoru ProfilePage(), poziva se InitializeComponent() za inicijalizaciju XAML komponenti i zatim LoadUserProfile() metoda koja asinhrono učitava podatke korisničkog profila iz baze podataka pomoću DatabaseServices.Instance.GetLoggedInUser(). Ako je korisnik uspešno prijavljen (loggedInUser nije null), postavljaju se podaci o korisniku (ime i email) na odgovarajuće Label kontrole (nameLabel i emailLabel) i loggedInUser se postavlja kao BindingContext stranice kako bi se podaci o korisniku koristili u XAML-u.

Metoda MyFavoritesTapped() omogućava korisniku da pređe na stranicu FavoritesPage kada pritisne opciju "My Favorite", koristeći Navigation.PushAsync(). U SignOutClicked() metodi, korisnik se odjavljuje pozivom DatabaseServices.Instance.SignOutUser(), nakon čega se preusmerava na početnu stranicu za prijavu (SignInPage) koja se postavlja kao nova glavna stranica aplikacije kroz Application.Current.MainPage.

Ovaj fajl omogućava korisnicima da pregledaju svoj profil, pristupe omiljenim stavkama i izvrše odjavu iz aplikacije, prateći MVC (Model-View-Controller) ili sličan arhitektonski obrazac za upravljanje korisničkim interakcijama i prikazom podataka.

XAML KOD:

```
<?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="KFC_Menu.ProfilePage"
  Shell.NavBarIsVisible="False"
  BackgroundColor="White">
  <ContentPage.Content>
    <Grid>
      <!-- Red background on top -->
      <BoxView Color="Red" HeightRequest="120" VerticalOptions="Start"
HorizontalOptions="FillAndExpand"/>

      <!-- Content -->
      <StackLayout Padding="20" VerticalOptions="Start">
        <!-- Page Title -->
        <Label Text="My Profile"
          FontSize="24"
          FontAttributes="Bold"
          TextColor="White"
          HorizontalOptions="Start"
          Grid.Row="0" />

        <!-- User Info Frame -->
        <Frame BackgroundColor="White" CornerRadius="10"
          Padding="20" HasShadow="True"
          Margin="0,10,0,20" VerticalOptions="Start">
          <StackLayout Orientation="Horizontal">
            <Image Source="user_solid.svg" WidthRequest="50"
HeightRequest="50" Margin="0,0,15,0" VerticalOptions="Center" />
            <StackLayout VerticalOptions="CenterAndExpand">
              <Label x:Name="nameLabel" FontSize="Medium"
FontAttributes="Bold"/>
              <Label x:Name="emailLabel" FontSize="Small"
TextColor="Gray" />
            </StackLayout>
          </StackLayout>
        </Frame>

        <!-- Menu Options -->
        <StackLayout Spacing="15" Margin="0,20,0,0">
          <!-- Order History -->
          <Frame BackgroundColor="White" CornerRadius="10" Padding="15"
HasShadow="True" >
            <StackLayout Orientation="Horizontal" Spacing="10">
              <Image Source="bag_shopping_solid.svg"
WidthRequest="24" HeightRequest="24" />
              <Label Text="Order History" FontSize="16"
TextColor="Black" />
            <StackLayout>
              <Image Source="angle_right_solid.svg"
WidthRequest="24" HeightRequest="24" HorizontalOptions="EndAndExpand"/>
            </StackLayout>
          </Frame>
          <!-- Payment Method -->
          <Frame BackgroundColor="White" CornerRadius="10" Padding="15"
HasShadow="True" >
            <StackLayout Orientation="Horizontal" Spacing="10">
              <Image Source="credit_card_regular.svg"
WidthRequest="24" HeightRequest="24" />
              <Label Text="Payment Method" FontSize="16"
TextColor="Black" />
            </StackLayout>
          </Frame>
        </StackLayout>
      </StackLayout>
    </Grid>
  </ContentPage.Content>
</ContentPage>
```

Label sa nazivom stranice

Frame sa podacima o profile korisnika


```

        <Image Source="angle_right_solid.svg"
WidthRequest="24" HeightRequest="24" HorizontalOptions="EndAndExpand"/>
    </StackLayout>
</Frame>
<!-- My Address -->
<Frame BackgroundColor="White" CornerRadius="10" Padding="15"
HasShadow="True" >
    <StackLayout Orientation="Horizontal" Spacing="10">
        <Image Source="location_dot_solid.svg"
WidthRequest="24" HeightRequest="24" />
        <Label Text="My Address" FontSize="16"
TextColor="Black" />
        <Image Source="angle_right_solid.svg"
WidthRequest="24" HeightRequest="24" HorizontalOptions="EndAndExpand"/>
    </StackLayout>
</Frame>
<!-- My Favorite -->
<Frame BackgroundColor="White" CornerRadius="10" Padding="15"
HasShadow="True" >
    <StackLayout Orientation="Horizontal" Spacing="10">
        <Image Source="heart_regular.svg" WidthRequest="24"
HeightRequest="24" />
        <Label Text="My Favorite" FontSize="16"
TextColor="Black" />
        <Image Source="angle_right_solid.svg"
WidthRequest="24" HeightRequest="24" HorizontalOptions="EndAndExpand"/>
        <StackLayout.GestureRecognizers>
            <TapGestureRecognizer Tapped="MyFavoritesTapped"
/>
        </StackLayout.GestureRecognizers>
    </StackLayout>
</Frame>
</StackLayout>
<Button Text="Sign Out"
TextColor="White"
FontAttributes="Bold"
BackgroundColor="Red"
CornerRadius="10"
VerticalOptions="EndAndExpand"
Margin="0,20,0,0"
Clicked="SignOutClicked" />
</StackLayout>
</Grid>
</ContentPage.Content>
</ContentPage>

```

Frame na koji kada se
klikne vodi na My
Favorites Page

Sign Out Button na
koji kada se klikne
vraća na Sign In Page

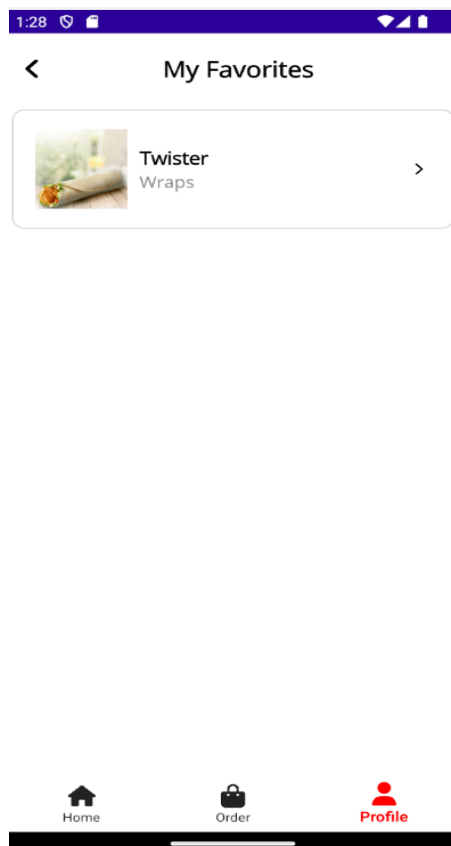
MY FAVORITES PAGE

Stranica `FavoritesPage` omogućava korisnicima da pregledaju svoje omiljene stavke koje su prethodno sačuvali. Pri inicijalizaciji, u konstruktoru `FavoritesPage()`, poziva se `InitializeComponent()` za inicijalizaciju XAML komponenti i zatim `LoadFavorites()` metoda koja asinhrono učitava omiljene stavke korisnika iz baze podataka pozivom `DatabaseServices.Instance.getFavoritesForUser(loggedInUser.Id)`. Ako korisnik nije prijavljen, prikazuje se upozorenje da je potrebno biti prijavljen da bi se videli omiljeni predmeti.

U `LoadFavorites()` metodi, nakon što se učitaju omiljeni predmeti, postavlja se `Favorites` kao nova `ObservableCollection` koja sadrži listu `MenuItem` objekata dobijenih iz baze podataka. Ova kolekcija se zatim postavlja kao `ItemsSource` za `FavoritesCollectionView`, što omogućava prikaz svake omiljene stavke unutar `CollectionView`.

Pritiskom na strelicu (`ImageButton`) pored svake stavke u `FavoritesCollectionView` (u metodi `FavoriteItemClicked`), korisnik može da pregleda detalje određenog omiljenog predmeta pozivom `Navigation.PushAsync(new ItemDetailsPage(menuItem))`, gde se otvara nova stranica `ItemDetailsPage` sa detaljima o odabranom predmetu.

Ova stranica omogućava korisnicima da pregledaju, upravljaju i pristupe detaljima svojih omiljenih stavki unutar aplikacije, što je korisno za personalizovanje iskustva kupovine i brz pristup preferiranim proizvodima ili uslugama.



Slika 9. My Favorites Page

```

using System.Collections.ObjectModel;
using KFC_Menu.Models;
using KFC_Menu.Services;

namespace KFC_Menu;

public partial class FavoritesPage : ContentPage
{
    private User loggedInUser;
    public ObservableCollection<MenuItem> Favorites { get; set; }
    public FavoritesPage()
    {
        InitializeComponent();
        LoadFavorites();
    }

    private async void BackButtonClicked(object sender, EventArgs e)
    {
        await Navigation.PopAsync();
    }

    private async void FavoriteItemClicked(object sender, EventArgs e)
    {
        var menuItem = (sender as ImageButton).BindingContext as MenuItem;
        if (menuItem != null)
        {
            await Navigation.PushAsync(new ItemDetailsPage(menuItem));
        }
    }

    private async void LoadFavorites()
    {
        loggedInUser = await DatabaseServices.Instance.GetLoggedInUser();

        if (loggedInUser == null)
        {
            await DisplayAlert("Error", "You need to be logged in to see your favorites", "OK");
        }

        var favorites = await DatabaseServices.Instance.GetFavoritesForUser(loggedInUser.Id);

        Favorites = new ObservableCollection<MenuItem>(favorites);
        BindingContext = this;
        FavoritesCollectionView.ItemsSource = Favorites;
    }
}

```

Ovaj kod predstavlja Xamarin.Forms stranicu FavoritesPage koja omogućava korisnicima da pregledaju svoje omiljene stavke.

U konstruktoru FavoritesPage() inicijalizuje se XAML komponenta pozivom InitializeComponent() i odmah se poziva LoadFavorites() metoda koja asinhrono učitava omiljene stavke korisnika iz baze podataka pozivom DatabaseServices.Instance.getFavoritesForUser(loggedInUser.Id).

Metoda BackButtonClicked omogućava korisniku da se vrati unazad pritiskom na dugme za povratak. Metoda FavoriteItemClicked se aktivira kada korisnik pritisne na strelicu pored omiljene stavke u FavoritesCollectionView, što otvara detalje te stavke na novoj stranici ItemDetailsPage.

Metoda LoadFavorites() prvo dobavlja trenutno prijavljenog korisnika pozivom DatabaseServices.Instance.getLoggedInUser(). Ako korisnik nije prijavljen, prikazuje se upozorenje putem DisplayAlert. Nakon toga, učitavaju se omiljene stavke tog korisnika iz baze podataka i smještaju se u ObservableCollection Favorites, koja se zatim postavlja kao BindingContext za FavoritesPage, omogućujući prikaz svih omiljenih stavki u FavoritesCollectionView.

XAML KOD:

```
<?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="KFC_Menu.FavoritesPage"
  Shell.NavBarIsVisible="False">
```

```
  <ContentPage.Content>
```

```
    <StackLayout>
```

```
      <StackLayout Orientation="Horizontal" Padding="10">
```

```
        <ImageButton Source="angle_left_solid.svg"
          WidthRequest="25"
          HeightRequest="25"
          Background="Transparent"
          Clicked="BackButtonClicked"
          VerticalOptions="Center"
          HorizontalOptions="Start"/>
```

ImageButton za vraćanje
na Profile Page

Label sa imenom stranice
My Favorites

```
        <Label Text="My Favorites" FontSize="Large" Padding="10"
          VerticalOptions="Center"
          FontAttributes="Bold" Margin="90,0,0,0"/>
      </StackLayout>
```

```
    <CollectionView x:Name="FavoritesCollectionView">
      <CollectionView.ItemTemplate>
        <DataTemplate>
          <Frame BorderColor="LightGray" Margin="5"
            CornerRadius="10"
            BackgroundColor="White" HeightRequest="120">
            <AbsoluteLayout>
              <StackLayout Orientation="Horizontal"
                AbsoluteLayout.LayoutBounds="0,0,1,1"
                AbsoluteLayout.LayoutFlags="All">
                <Image Source="{Binding Image}"
                  WidthRequest="80"
                  HeightRequest="90" Aspect="AspectFill"
                  VerticalOptions="CenterAndExpand"
                  HorizontalOptions="Start"/>
                <StackLayout Padding="10,0,0,0"
                  VerticalOptions="CenterAndExpand">
                  <Label Text="{Binding Name}"
                    FontSize="18" FontAttributes="Bold" />
                  <Label Text="{Binding Category}"
                    FontSize="16" TextColor="Gray" />
                </StackLayout>
              </StackLayout>
              <ImageButton Source="angle_right_solid.svg"
                Clicked="FavoriteItemClicked"
                HeightRequest="15"
                AbsoluteLayout.LayoutBounds="7.5,
                9.5, AutoSize, AutoSize"
                AbsoluteLayout.LayoutFlags="PositionProportional"/>
            </AbsoluteLayout>
          </Frame>
        </DataTemplate>
      </CollectionView.ItemTemplate>
    </CollectionView>
  </StackLayout>
</ContentPage.Content>
</ContentPage>
```

CollectionView za prikazivanje svih
Menuitema u Favorites Page. Menuitem
je prikazan preko Binding svojstva i
korišten je AbsoluteLayout za raspored
elemenata Menuitema

TABBAR ZA PRELAŽENJE IZMEĐU STRANICA

Ovaj TabBar u AppShell.xaml fajlu definiše navigacionu strukturu za Xamarin.Forms aplikaciju koristeći Shell komponentu. TabBar sadrži tri Tab elementa, svaki sa svojim ShellContent, koji predstavljaju glavne stranice aplikacije: "Home", "Order", i "Profile".

Shell: Korijenski element koji omogućava jednostavniju navigaciju i upravljanje stranicama u aplikaciji. Postavljeni atributi onemogućavaju bočni izbornik (FlyoutBehavior="Disabled") i sakrivaju navigacijsku traku (NavBarIsVisible="False"). Takođe, boja naslova i prednji plan tab bara su postavljeni na crvenu (TabBarTitleColor="Red" i TabBarForegroundColor="Red").

TabBar: Sadrži tri taba:

- Tab za Home: Ima ikonu house_solid.svg i prikazuje stranicu MenuPage bez navigacione trake.
- Tab za Order: Ima ikonu bag_shopping_solid.svg i prikazuje stranicu OrderPage bez navigacione trake.
- Tab za Profile: Ima ikonu user_solid.svg i prikazuje stranicu ProfilePage bez navigacione trake.

Svaki Tab koristi ShellContent da definiše koja se stranica prikazuje kada je tab aktivan, koristeći ContentTemplate za povezivanje sa odgovarajućom XAML stranicom.

```
<?xml version="1.0" encoding="UTF-8" ?>
<Shell
  x:Class="KFC_Menu.AppShell"
  xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
  xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
  xmlns:local="clr-namespace:KFC_Menu"
  Shell.FlyoutBehavior="Disabled"
  Shell.NavBarIsVisible="False"
  Shell.TabBarTitleColor="Red"
  Shell.TabBarForegroundColor="Red">

  <TabBar>
    <Tab Title="Home" Icon="house_solid.svg">
      <ShellContent Title="Home" ContentTemplate="{DataTemplate
local:MenuPage}"
                    Route="MenuPage" Shell.NavBarIsVisible="False" />
    </Tab>
    <Tab Title="Order" Icon="bag_shopping_solid.svg">
      <ShellContent Title="Order" ContentTemplate="{DataTemplate
local:OrderPage}"
                    Route="OrderPage" Shell.NavBarIsVisible="False" />
    </Tab>
    <Tab Title="Profile" Icon="user_solid.svg">
      <ShellContent Title="Profile" ContentTemplate="{DataTemplate
local:ProfilePage}"
                    Route="ProfilePage" Shell.NavBarIsVisible="False" />
    </Tab>
  </TabBar>
</Shell>
```