МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

«Новосибирский национальный исследовательский государственный университет»

(Новосибирский государственный университет, НГУ)

Структурное подразделение Новосибирского государственного университета –

Высший колледж информатики Университета (ВКИ НГУ)

КАФЕДРА ИНФОРМАТИКИ

**КУРСОВОЙ ПРОЕКТ**

**по дисциплине**

**«Программирование мобильных приложений»**

**на тему «Приложение доставки еды»**

|  |  |
| --- | --- |
| Проверил преподаватель | Жирнов А. А.  «08»декабря2020 г. |
| Выполнил студент 5 курса  гр. 603а2 | Шабинский Д. А.  «05» декабря2020 г. |

Новосибирск 2020 г.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc58074200)

[1 Описание предметной области 4](#_Toc58074201)

[3 Аналоги 6](#_Toc58074202)

[4 Функциональные требования 8](#_Toc58074203)

[5 Нефункциональные требования 9](#_Toc58074204)

[5.1 Требования к программному обеспечению 9](#_Toc58074205)

[5.2 Требования к аппаратному обеспечению 9](#_Toc58074206)

[5.3 Требования к надёжности 9](#_Toc58074207)

[6 Используемые инструменты 10](#_Toc58074208)

[ЗАКЛЮЧЕНИЕ 12](#_Toc58074209)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 13](#_Toc58074210)

[ПРИЛОЖЕНИЕ А 14](#_Toc58074211)

[ПРИЛОЖЕНИЕ Б 20](#_Toc58074212)

# **ВВЕДЕНИЕ**

Объектом исследования является рынок мобильных приложений в сети предприятий общественного питания, рассчитанных на осуществление доставки продукции потребителю.

Цель работы – разработка приложения, предоставляющего данные о товарах в предприятии общественного питания с возможностью оформления заказа.

Данный программный продукт крайне необходим во время пандемии, так как благодаря нему можно снизить уровень заражения вирусом, людям не придется выходить на улицу, чтобы поесть в предприятии общественного питания.

## **1 Описание предметной области**

Доставка еды — услуга, при которой заказчику доставляют еду по указанному им адресу за определённый период времени. Заказ обычно совершается по телефону или интернету (в нашем случае – приложению). Стоимость доставки может варьироваться, либо, что чаще всего, уже включена в стоимость.

В настоящее время доставка еды в крупных городах осуществляется пешими курьерами или малолитражных машинах из-за экономии, так как цена заказа в среднем составляет 600 рублей, что сильно ограничивает возможность расходов на транспорт. Если адрес доставки лежит в пределах одного квартала, доставка может осуществляться пешим курьером.

Пандемия коронавируса прошлась по многим сферам бизнеса. Однако есть и сервисы, которые только выиграли от повсеместной самоизоляции. Речь идет о сервисах доставки готовой еды и продуктов на дом.

Большинство из этих сервисов имеют мобильные приложения, которые позволяют с минимальными усилиями и максимальным комфортом оформить заказ.

Исходя из вышесказанного можно сделать вывод, что приложения для доставки еды очень популярны, особенно в 2020 году. Именно поэтому разрабатываемое приложение имеет основные позиции в меню, такие как:

1. Пицца;
2. Закуски;
3. Напитки;
4. Другое.

Приложения, как правило, имеют возможность регистрации учетной записи, так и в данном программном продукте имеется возможность регистрации, входа, а также просмотра своих предыдущих заказов.

**2 Постановка задачи**

Необходимо разработать приложение, которое позволит выбирать продукты предприятия общественного питания и производить их заказ. Разработка подразумевает:

* создание клиентской части – графического интерфейса для взаимодействия с приложением;
* создание серверной части – базы данных, в которой будут хранится данные аккаунтов пользователей и всех продуктов предприятия, а также создание API, для более удобного способа работы с данными.

Клиентская часть включает в себя следующие возможности:

* регистрация нового пользователя, с возможностью входа в аккаунт в будущем;
* просмотр информации об аккаунте;
* просмотр информации о продуктах предприятия;
* формирование “Корзины” из выбранных продуктов, с последующей возможностью заказа;
* просмотр данных о предприятии.

Серверная часть включает в себя следующие возможности:

* отправка данных нового пользователя на сервер при регистрации;
* проверка и получение данных аккаунта при совпадении логина и пароля при входе;
* получение данных о продуктах предприятия;
* отправка заказа на сервер с помощью API.

## **3 Аналоги**

Основным аналогом приложения является сервис доставки еды " «Додо-пицца», рисунок 1. В этом приложении есть возможность выбрать продукт, добавить его в корзину, а также заказать уже выбранные товары, рисунок 2.

Достоинства:

* приложение адаптировано для пользователей с ограниченными возможностями;
* удобный поиск;
* показаны жидкости, необходимые для определённой детали;
* удобный интерфейс определенного продукта;
* удобный интерфейс профиля пользователя.

Недостатки:

* нельзя привязать банковскую карту до оплаты;
* нельзя удалить свой аккаунт из приложения;
* многочисленные всплывающие окна акций;
* некрасивый дизайн основного меню.

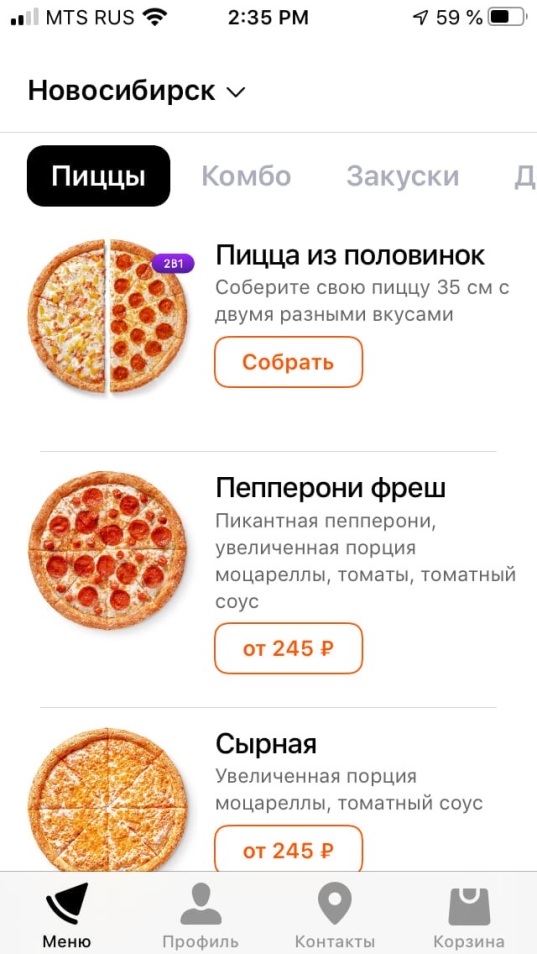


Рисунок 1 – Главное страница приложения «Додо-пицца»

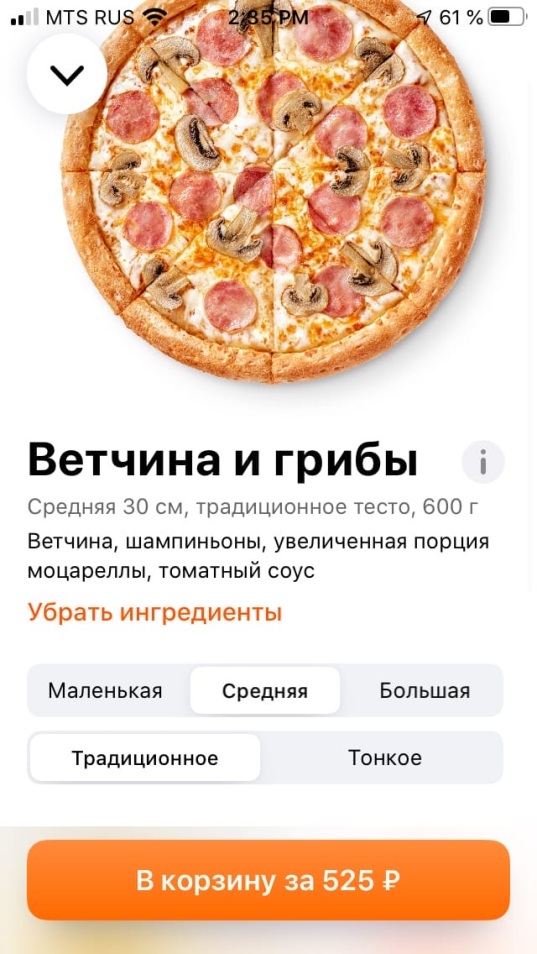


Рисунок 2 – Страница определенного товара.

## **4 Функциональные требования**

Пользователь имеет следующие возможности:

* зарегистрироваться/войти в аккаунт;
* просматривать всю продукцию предприятия;
* просматривать свои прошлые заказы;
* обращение к API, с целью отправки заказа на сервер.

## **5 Нефункциональные требования**

### **5.1 Требования к программному обеспечению**

Для обеспечения функционирования программного средства необходимо:

* смартфон на базе ОС Android 8.0 – API 26 и выше;
* установленное на смартфон приложение «Play Market».

### **5.2 Требования к аппаратному обеспечению**

Для функционирования программного средства необходимы:

* количество оперативной памяти не менее 1 ГБ;
* наличие Интернет-соединения.

### **5.3 Требования к надёжности**

К программному средству предъявляются следующие требования в плане надёжности:

* входные данные должны подвергаться проверке на корректность;
* поступление на ввод некорректных данных должно корректно обрабатываться.

## **6 Используемые инструменты**

При создании приложения были использованы следующие средства и технологии:

* среда разработки Microsoft Visual Studio 2019.

Достоинства: поддержка большинства плагинов для разработки на разных языках, доступен встроенный терминал, скорость.

Недостатки: высокая потребность в оперативной памяти.

Данное программное обеспечение необходимо использовать ввиду его возможностей для разработки проектов на Xamarin.

* платформа Xamarin с открытым исходным кодом, предназначенная для построения современных производительных приложений для iOS, Android и Windows с .NET.
* эмулятор смартфона Pixel XL Oreo 8.0 – API 26, 1GB.

Устройство, необходимое для тестирования приложения на наличие ошибок.

**7 Отладка и тестирование программного средства**

Тестирование продукта производилось методом ручного тестирования с использованием программных средств для проверки, путём моделирования действий пользователя.

Тестирование проводилось на ОС Windows 10 с использованием эмулятора смартфона «Pixel XL Oreo 8.0 – API 26, 1GB».

# **ЗАКЛЮЧЕНИЕ**

По итогу проделанной работы были сформулированы функциональные и нефункциональные требования к программному продукту, а также разработано мобильное приложение.

Также были выполнены следующие виды работ:

* Проанализирована предметная область разрабатываемого приложения;
* Изучена среда разработки Microsoft Visual Studio 2019;
* Изучена платформа Xamarin;
* Разработан интерфейс приложения;
* Изучен механизм работы мобильных приложений;
* Изучен механизм работы серверных Баз Данных.
* Изучен механизм создания API.

Добавить перспективы развития и модернизации приложения

# **СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

1. Васюкова А. Т. Организация производства и обслуживания на предприятиях общественного питания / А. Т. Васюкова, Т. Р. Любецкая; ISBN 978-5-394-02181-7, 2014. – 417с.
2. Черников Вячеслав Разработка мобильных приложений на C# для iOS и Android; Москва: Издательство «ДМК Пресс»,2020. – 188с.
3. Официальная документация по Xamarin [Электронный ресурс]

URL: <https://docs.microsoft.com/ru-ru/xamarin/> (дата обращения 05.12.2020 г.)

1. Додо Пицца | Сеть пиццерий №1 в России: примеры продуктов, особенности, вопросы выбора [Электронный ресурс]

URL: <https://dodopizza.ru/moscow> (дата обращения 03.12.2020 г.)

1. Рейтинг приложений доставки еды [Электронный ресурс]

URL: <https://basetop.ru/luchshie-prilozheniya-dostavki-edy-rejting-roskachestva/> (дата обращения 02.12.2020 г.)

# **ПРИЛОЖЕНИЕ А**

/Views/DishDetailPage.xaml.cs – Шаблон страницы определенного продукта.

using Eldoed.Data;

using Eldoed.Models;

using System;

using System.Linq;

using System.Text.RegularExpressions;

using Xamarin.Forms;

namespace Eldoed.Views

{

[QueryProperty("Name", "name")]

public partial class DishDetailPage : ContentPage

{

public Dish \_dish;

public string Name

{

set

{

BindingContext = AllDishesData.AllDishes.FirstOrDefault(m => m.Name == Uri.UnescapeDataString(value));

\_dish = AllDishesData.AllDishes.FirstOrDefault(m => m.Name == Uri.UnescapeDataString(value));

}

get

{

return Name;

}

}

public DishDetailPage()

{

InitializeComponent();

}

protected override bool OnBackButtonPressed()

{

return base.OnBackButtonPressed();

}

private static double PriceToDouble(string price)

{

var pattern = @"^[0-9]+\,\*[0-9]+";

Regex regEx = new Regex(pattern);

var temp = double.Parse(regEx.Match(price).ToString());

return temp;

}

private void orderButton\_Clicked(object sender, EventArgs e)

{

var tempDish = Cart.CartList.FirstOrDefault(m => m.Name == \_dish.Name);

if (tempDish == null)

{

var dish = new DishInCart() {

Name = \_dish.Name,

ImageUrl = \_dish.ImageUrl,

Price = PriceToDouble(\_dish.Price),

Quantity = 1

};

Cart.CartList.Add(dish);

DisplayAlert("Выполнено", $"В вашу корзину добавлен товар: {\_dish.Name}", "ОК");

}

else

{

//Cart.CartList.FirstOrDefault(m => m.Name == \_dish.Name).Quantity += 1;

DisplayAlert("Внимание", $"В вашей корзине уже присутствует данный товар: {\_dish.Name}", "ОК");

}

Application.Current.MainPage.Navigation.PopAsync();

}

}

}

/Data/AllDishesData.cs – Заполнение данными.

using Eldoed.Models;

using Newtonsoft.Json;

using System;

using System.Collections.Generic;

using System.Net.Http;

using System.Threading.Tasks;

namespace Eldoed.Data

{

public static class AllDishesData

{

private const string URL = "https://api-eldoed.herokuapp.com/products";

public static IList<Dish> AllDishes {get; private set;}

static AllDishesData()

{

AllDishes = new List<Dish>();

using (HttpClient httpClient = new HttpClient())

{

using (HttpRequestMessage httpRequestMessage = new HttpRequestMessage())

{

httpRequestMessage.RequestUri = new Uri(URL);

httpRequestMessage.Method = HttpMethod.Get;

httpRequestMessage.Headers.Add("null", "null");

Task<HttpResponseMessage> httpResponse = httpClient.SendAsync(httpRequestMessage);

using (HttpResponseMessage httpResponseMessage = httpResponse?.Result)

{

HttpContent responseContent = httpResponseMessage.Content;

Task<string> responseData = responseContent.ReadAsStringAsync();

string data = responseData.Result;

JSONDishesList dishesList = JsonConvert.DeserializeObject<JSONDishesList>(responseData.Result);

foreach (JSONDishTemplate dish in dishesList.Data)

{

Dish temp = new Dish()

{

Name = dish.Name.ToString(),

Price = dish.Price.ToString() + " руб.",

Details = dish.Description.ToString(),

ImageUrl =

"https://api-eldoed.herokuapp.com/"

+ (dish.ImageData.ToString().Replace(@"server\", "")).Replace(@"\", "/")

};

AllDishes.Add(temp);

switch (dish.Category.ToString())

{

case "pizza":

{

PizzaData.Pizzas.Add(temp); break;

};

case "snacks":

{

SnacksData.Snacks.Add(temp); break;

};

case "drinks":

{

DrinksData.Drinks.Add(temp); break;

};

case "other":

{

OtherData.Other.Add(temp); break;

};

default:

{

break;

};

}

}

}

}

}

}

}

}

/Controls/DishSearchHandler.cs – Поиск продуктов.

using Eldoed.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Xamarin.Forms;

namespace Eldoed.Controls

{

public class DishSearchHandler : SearchHandler

{

public IList<Dish> Dishes { get; set; }

public Type SelectedItemNavigationTarget { get; set; }

protected override void OnQueryChanged(string oldValue, string newValue)

{

base.OnQueryChanged(oldValue, newValue);

if (string.IsNullOrWhiteSpace(newValue))

{

ItemsSource = null;

}

else

{

ItemsSource = Dishes

.Where(dish => dish.Name.ToLower().Contains(newValue.ToLower()))

.ToList<Dish>();

}

}

protected override async void OnItemSelected(object item)

{

base.OnItemSelected(item);

await Task.Delay(1000);

ShellNavigationState state = (App.Current.MainPage as Shell).CurrentState;

await Shell.Current.GoToAsync($"{GetNavigationTarget()}?name={((Dish)item).Name}");

}

string GetNavigationTarget()

{

return (Shell.Current as AppShell).Routes.FirstOrDefault(route => route.Value.Equals(SelectedItemNavigationTarget)).Key;

}

}

}

/AppShell.xaml – Главная страница приложения.

<?xml version="1.0" encoding="utf-8" ?>

<Shell xmlns="http://xamarin.com/schemas/2014/forms"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

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

xmlns:d="http://xamarin.com/schemas/2014/forms/design"

xmlns:controls="clr-namespace:Eldoed.Controls"

xmlns:views="clr-namespace:Eldoed.Views"

mc:Ignorable="d"

x:Class="Eldoed.AppShell"

BackgroundColor="#ff4400"

FlyoutBackgroundColor="#FCD27D"

TitleColor="White"

DisabledColor="Gray"

UnselectedColor="Black"

FlyoutBehavior="Flyout"

FlyoutHeaderBehavior="CollapseOnScroll">

<!-- Static Resources -->

<Shell.Resources>

<Style x:Key="BaseStyle"

TargetType="Element">

<Setter Property="Shell.BackgroundColor"

Value="#455A64" />

<Setter Property="Shell.ForegroundColor"

Value="White" />

<Setter Property="Shell.TitleColor"

Value="White" />

<Setter Property="Shell.DisabledColor"

Value="#B4FFFFFF" />

<Setter Property="Shell.UnselectedColor"

Value="#95FFFFFF" />

</Style>

<Style TargetType="ShellItem"

BasedOn="{StaticResource BaseStyle}" />

<Style x:Key="MenuShell"

TargetType="Element"

BasedOn="{StaticResource BaseStyle}">

<Setter Property="Shell.BackgroundColor"

Value="#ffa500" />

</Style>

<Style x:Key="ProfileShell"

TargetType="Element"

BasedOn="{StaticResource BaseStyle}">

<Setter Property="Shell.BackgroundColor"

Value="#ff8c00" />

</Style>

<Style x:Key="ContactsShell"

TargetType="Element"

BasedOn="{StaticResource BaseStyle}">

<Setter Property="Shell.BackgroundColor"

Value="#ff5a00" />

</Style>

<Style x:Key="CartShell"

TargetType="Element"

BasedOn="{StaticResource BaseStyle}">

<Setter Property="Shell.BackgroundColor"

Value="#ff4500" />

</Style>

<Style x:Key="AboutShell"

TargetType="Element"

BasedOn="{StaticResource BaseStyle}">

<Setter Property="Shell.BackgroundColor"

Value="#96d1ff" />

</Style>

</Shell.Resources>

<!-- Header -->

<Shell.FlyoutHeader>

<controls:FlyoutHeader />

</Shell.FlyoutHeader>

<!-- Template each flyout item-->

<Shell.ItemTemplate>

<DataTemplate>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="0.3\*" />

<ColumnDefinition Width="0.7\*" />

</Grid.ColumnDefinitions>

<Image Source="{Binding Icon}"

Margin="5"

HeightRequest="45" />

<Label Text="{Binding Title}"

FontFamily="Arial"

Grid.Column="1"

TextColor="#282828"

FontSize="Medium"

FontAttributes="Bold"

VerticalTextAlignment="Center"/>

</Grid>

</DataTemplate>

</Shell.ItemTemplate>

<!-- Template each MenuItem -->

<Shell.MenuItemTemplate>

<DataTemplate>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="0.3\*" />

<ColumnDefinition Width="0.7\*" />

</Grid.ColumnDefinitions>

<Image Source="{Binding Icon}"

Margin="5"

HeightRequest="45" />

<Label Text="{Binding Text}"

FontFamily="serif"

Grid.Column="1"

TextColor="#454545"

FontSize="Medium"

FontAttributes="Bold, Italic"

VerticalTextAlignment="Center"/>

</Grid>

</DataTemplate>

</Shell.MenuItemTemplate>

<!-- Flyout items -->

<FlyoutItem Route="flyout"

Title="Flyout"

FlyoutDisplayOptions="AsMultipleItems">

<Tab Title="Меню"

Route="menu"

Icon="menu1.ico">

<ShellContent Route="pizza"

Style="{StaticResource MenuShell}"

Title="Пицца"

Icon="pizzaicon.png"

ContentTemplate="{DataTemplate views:PizzaPage}" />

<ShellContent Route="drinks"

Style="{StaticResource MenuShell}"

Title="Напитки"

Icon="drinkicon.png"

ContentTemplate="{DataTemplate views:DrinksPage}" />

<ShellContent Route="snack"

Style="{StaticResource MenuShell}"

Title="Закуски"

Icon="snackicon.png"

ContentTemplate="{DataTemplate views:SnacksPage}" />

<ShellContent Route="other"

Style="{StaticResource MenuShell}"

Title="Другое"

Icon="othericon.png"

ContentTemplate="{DataTemplate views:OtherPage}" />

</Tab>

<ShellContent Route="profile"

Style="{StaticResource ProfileShell}"

Title="Профиль"

Icon="profile.ico"

ContentTemplate="{DataTemplate views:ProfilePage}" />

<ShellContent Route="contacts"

Style="{StaticResource ContactsShell}"

Title="Контакты"

Icon="contacts.ico"

ContentTemplate="{DataTemplate views:ContactsPage}" />

<ShellContent Route="cart"

Style="{StaticResource CartShell}"

Title="Корзина"

Icon="cart.ico"

ContentTemplate="{DataTemplate views:CartPage}" />

</FlyoutItem>

<MenuItem Text="Случайная позиция"

IconImageSource="random.png"

Command="{Binding RandomPageCommand}" />

<MenuItem Text="Помощь"

IconImageSource="help.png"

Command="{Binding HelpCommand}"

CommandParameter="https://gabajoba.ru/" />

</Shell>

# **ПРИЛОЖЕНИЕ Б**

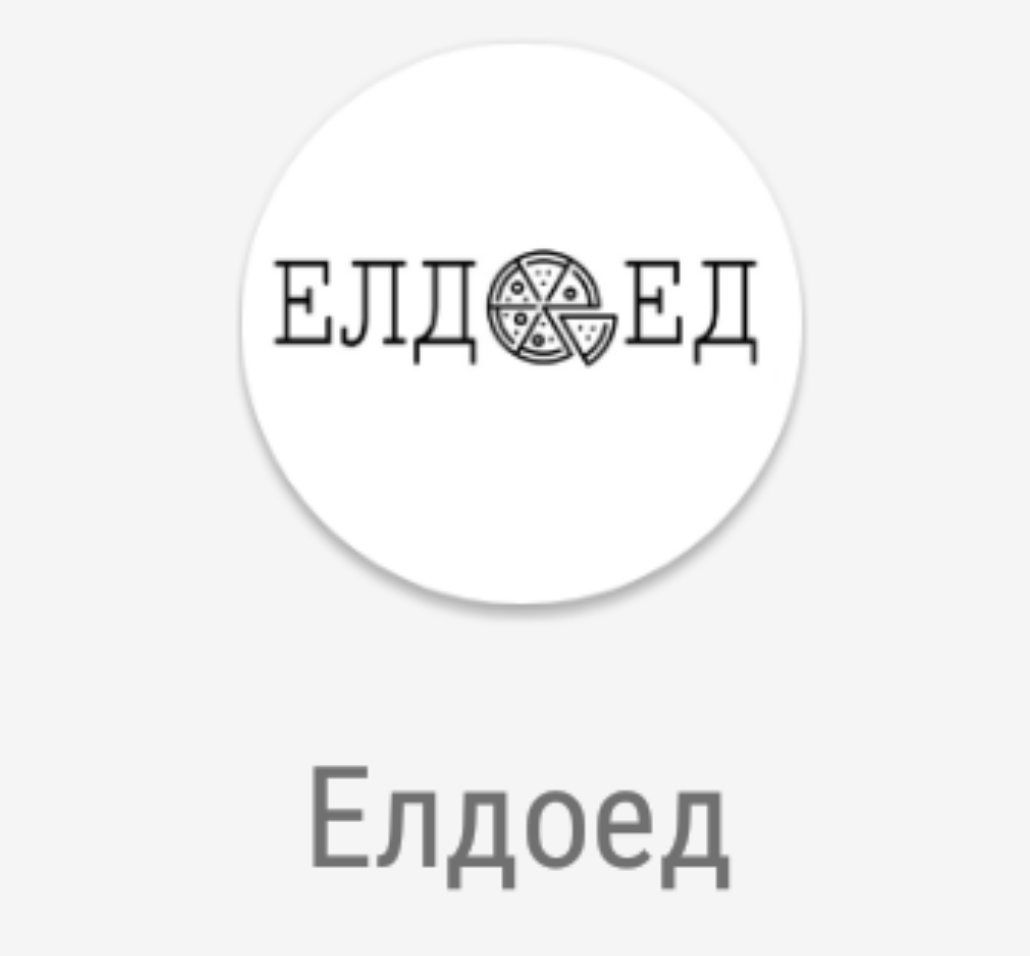


Рисунок 3 – Иконка приложения.

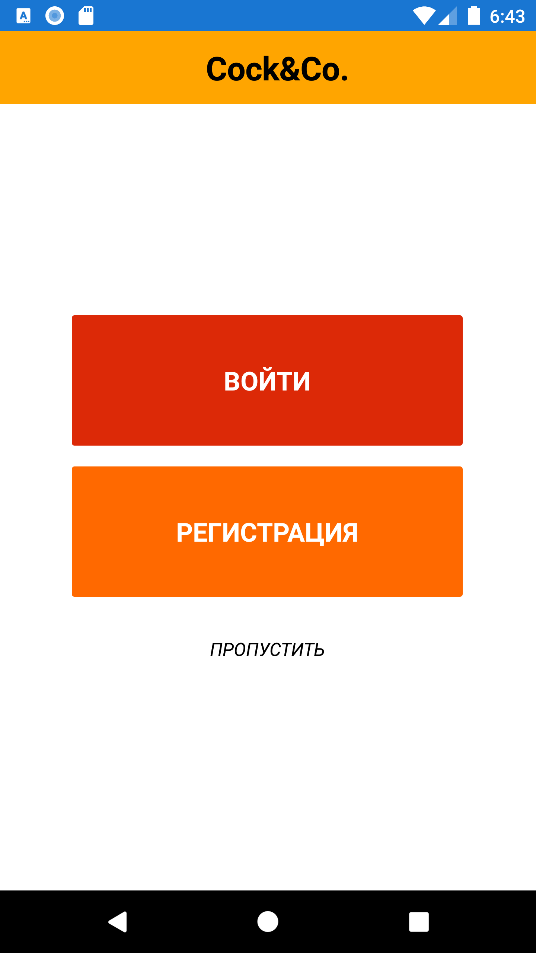


Рисунок 4 – Начальная страница приложения.

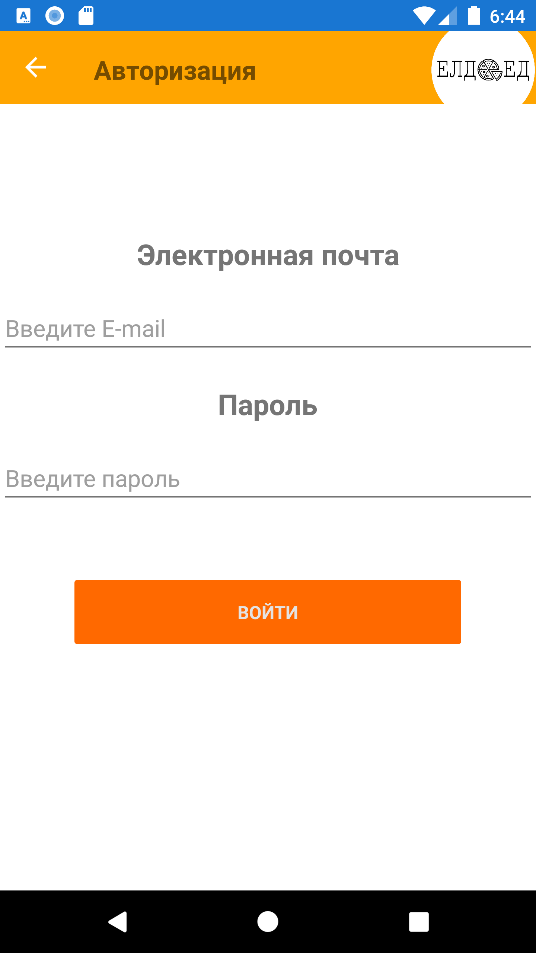


Рисунок 5 – Страница авторизации.

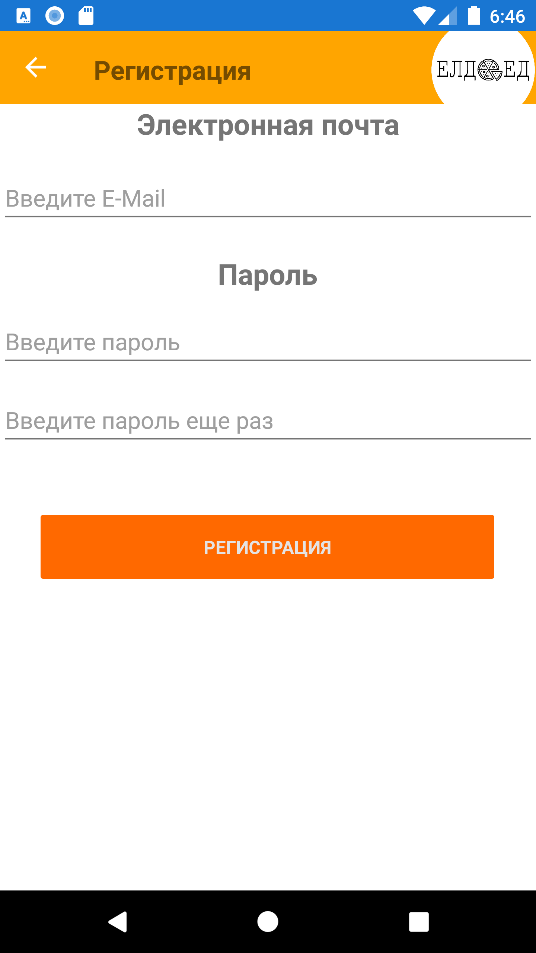


Рисунок 6 – Страница регистрации.

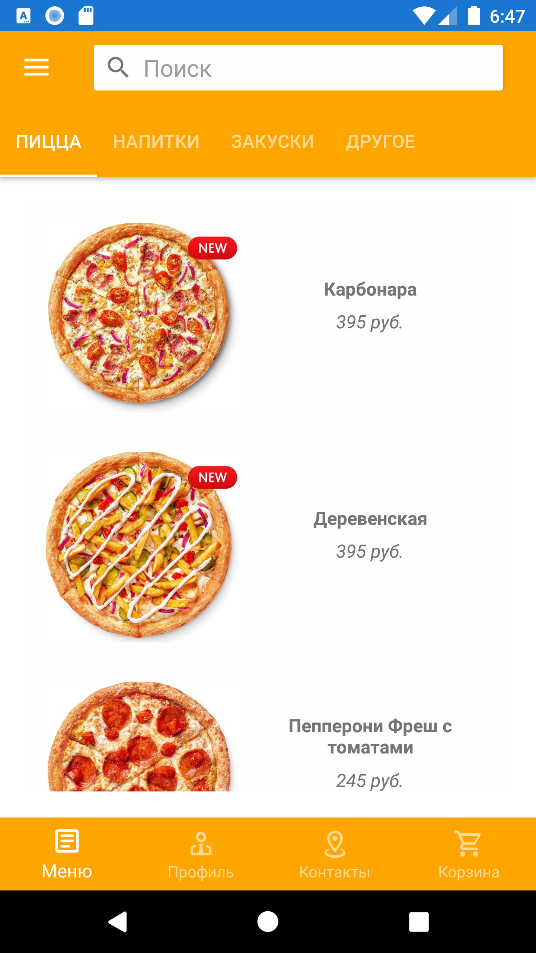


Рисунок 7 – Страница меню продукции (Пицца).



Рисунок 8 – Страница меню продукции (Напитки).

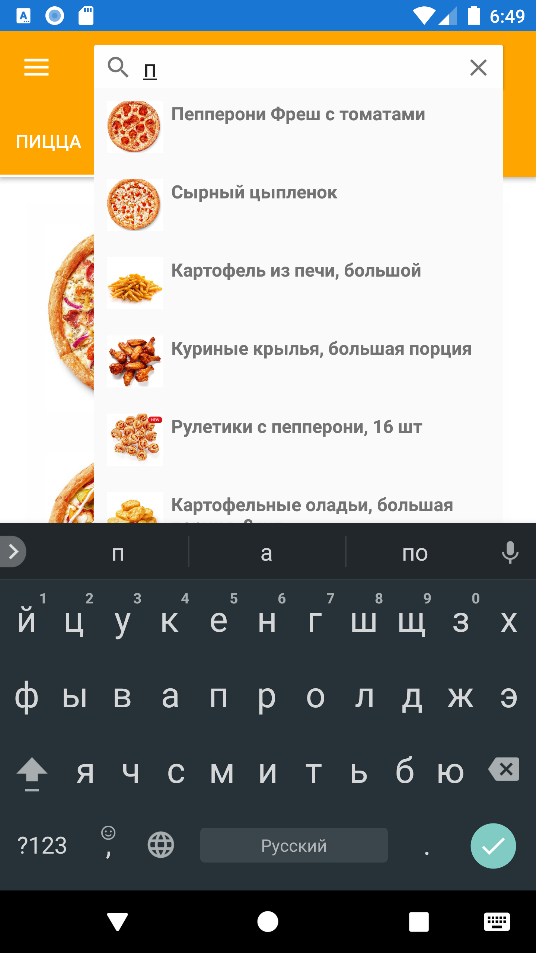


Рисунок 9 – Пример работы поиска в меню.



Рисунок 10 – Страница определенного продукта.

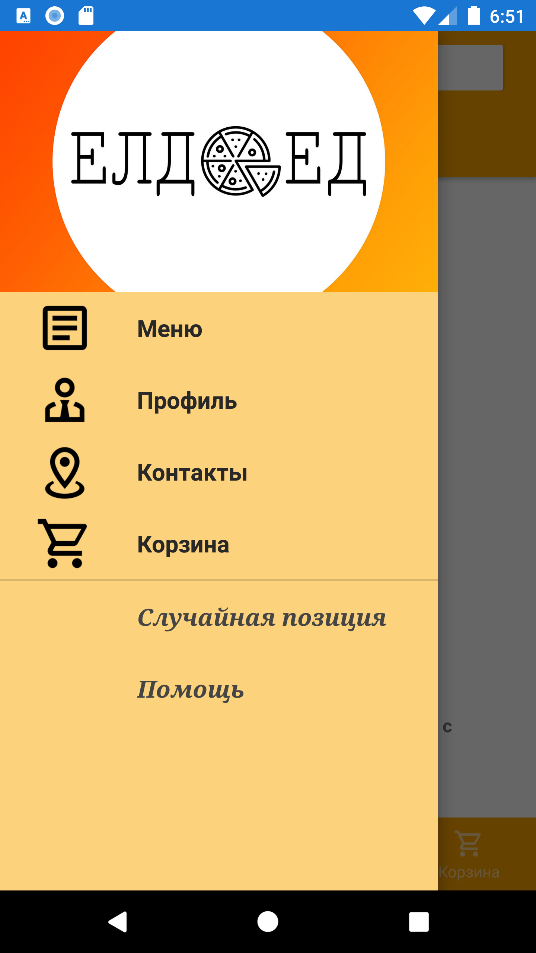


Рисунок 11 – Боковое меню приложения.

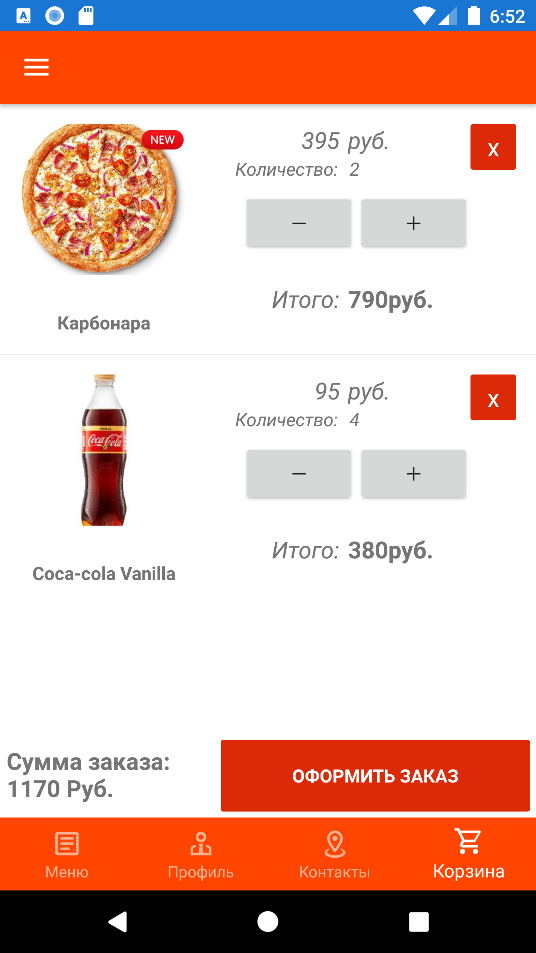


Рисунок 12 – Страница корзины.

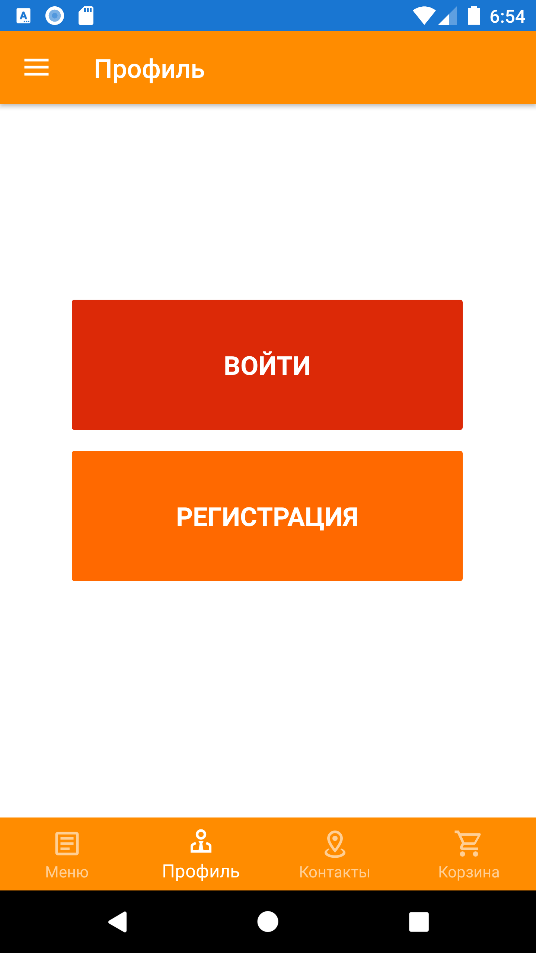


Рисунок 13 – Страница профиля (Неавторизированный пользователь).

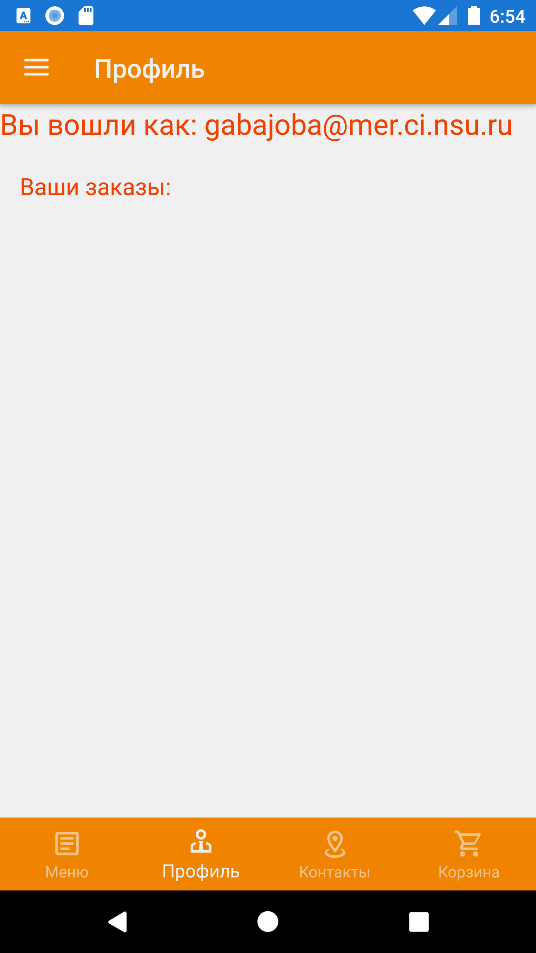


Рисунок 14 – Страница профиля (Авторизированный пользователь).

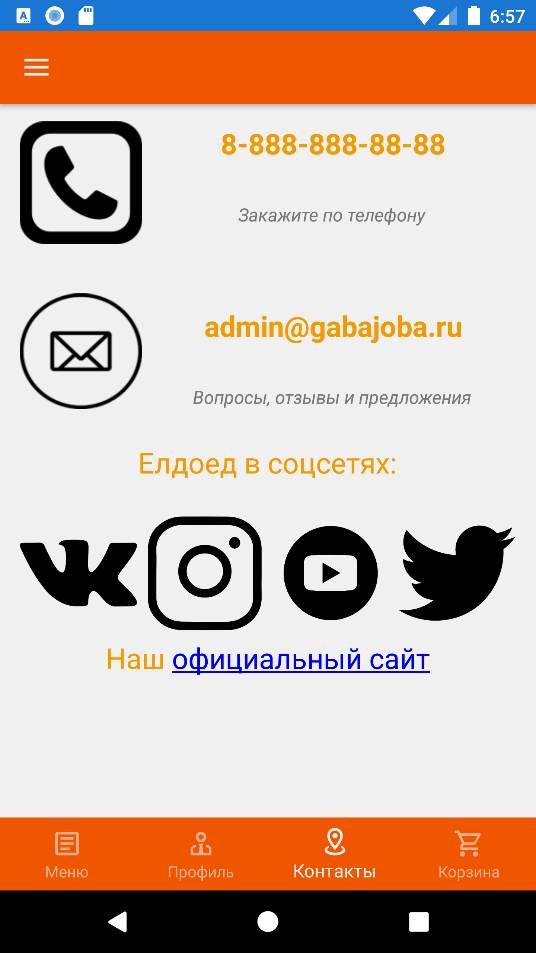


Рисунок 15 – Страница контактов.