Министерство образования Московской области

Государственное образовательное учреждение высшего образования Московской области

«Государственный гуманитарно-технологический университет»

**Ликино-Дулевский политехнический колледж – филиал ГГТУ**

**КУРСОВОЙ ПРОЕКТ**

«Разработка программного модуля для Пассажирские перевозки»

МДК 01.01 «Разработка программных модулей»

**Выполнил:**

Яковлев Алексей Алексеевич

студент группы ИСП.21.1А\_\_\_\_\_\_\_\_

09.02.07 Информационные системы и программирование)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

очной формы обучения\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Руководитель:**

Кузьмина Елена Евгеньевна­­­­­­­­­­­­­­­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка

\_\_\_\_\_\_\_\_\_/\_\_\_­­­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

Подпись руководителя

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

Орехово-Зуево

2023 год

**Введение**

Одной из самых востребованных услуг в Москве является перевозка пассажиров. С их перевозкой не всегда справляются общественный и личный автотранспорт, главным образом, когда приходится организовывать коммерческие поездки для больших групп.

Мобильное приложение Swift travel создано для упрощения процесса аренды транспорта и обработки запросов заказчиков.

**Пассажирские перевозки** - это перевозки людей с одного места на другое с использованием различных видов транспорта, таких как автомобили, маршрутки, автобусы. Основные понятия, связанные с пассажирскими перевозками, включают:

**Пассажир** - человек, который путешествует из одного места в другое. Осуществление перевозки пассажиров является целью деятельности пассажирских перевозчиков.

**Пассажирский транспорт** - различные виды транспорта, предназначенные для перевозки пассажиров. Это включает в себя автомобили, автобусы, поезда, метро, самолеты, трамваи, паромы и т.д.

**Пассажирские перевозчики** - компании или организации, занимающиеся предоставлением услуг по перевозке пассажиров. Они могут быть государственными или частными, и обычно работают на основе расписания или графика.

**Маршрут** - определенный путь, который должен пройти транспорт для доставки пассажиров из одной точки в другую. Маршруты могут быть регулярными, то есть постоянно повторяющимися, или одноразовыми.

1. **Назначение разработки**

Разработка проводится на основании приказа о курсовом проектировании. Существует компания которая занимается перевозкой пассажиров, в неё обращаются люди для аренды различных видов авто. Мобильное приложение предназначено для упрощения процесса бронирования поездок.

1. **Требования к программному модулю**

**2.1 Требования к функциональным характеристикам:**

Приложение должно выполнять следующие функции:

* Регистрация и авторизация пользователей
* Сортировка транспорта по его виду
* Просмотр информации о имеющихся авто
* Ввод и отправка данных для заказа транспорта
* Роль администратора для просмотра заявок
* Корзина для пользователя с данными о его заказе

## Требования к составу и параметрам технических средств

Минимальные характеристики устройства

|  |  |
| --- | --- |
| Процессор | 1 ГГц или выше |
| Внутренняя память: | 8 ГБ или выше. |
| Дисплей | 480x800 |
|  |  |
|  |  |
|  |  |

Оптимальные характеристики компьютера

|  |  |
| --- | --- |
| Процессор | 1.5 ГГц или выше. |
| Внутренняя память: | 16 ГБ или выше. |
| Дисплей | 720x1280 |
|  |  |
|  |  |

## Требования к информационной и программной совместимости

Для работы мобильного приложения необходима ОС Android с версией 4.4(уровень API 19-Kit Kat) и выше.

1. **Руководство программиста**

## 3.1 Разработка программного модуля

Мобильное приложение было разработано на Xamarin.Forms. С помощью среды разработки Visual Studio. Использовался язык программирования C#, для разметки использовался XAML.

Xamarin.Forms — это платформа пользовательского интерфейса с открытым кодом. С помощью Xamarin.Forms разработчики могут создавать приложения для Xamarin.Android, Xamarin.iOS и Windows на основе общей базы кода.Платформа позволяет разработчикам создавать пользовательские интерфейсы в XAML с помощью кода программной части в C#. Эти интерфейсы на каждой платформе подготавливаются к просмотру как собственные элементы управления.



Рис. 1 «Xamarin»

XAML — это декларативный язык разметки. При применении к модели программирования .NET XAML упрощает создание пользовательского интерфейса для приложения .NET. Можно создать видимые элементы пользовательского интерфейса в декларативной XAML-разметке, а затем отделить определение пользовательского интерфейса от логики времени выполнения, используя файлы кода программной части, присоединенные к разметке с помощью определений разделяемых классов.



Рис. 2 «XAML»

 Visual Studio — это стартовая площадка для написания, отладки и сборки кода, а также последующей публикации приложений. Помимо стандартного редактора и отладчика, которые есть в большинстве сред IDE, Visual Studio включает в себя компиляторы, средства автозавершения кода, графические конструкторы и многие другие функции для улучшения процесса разработки.



Рис. 3 «Visual Studio»

C# — объектно-ориентированный язык программирования общего назначения.

Разработан в 1998–2001 годах группой инженеров компании Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота как язык разработки приложений для платформы Microsoft .NET Framework и .NET Core.



Рис. 4 «C#»

Процесс разработки приложения:

1.Установлены пакеты NuGet для работы с платформой Firebase. В проекте используются сервисы Firebase Realtime Database и Firebase Authentication.

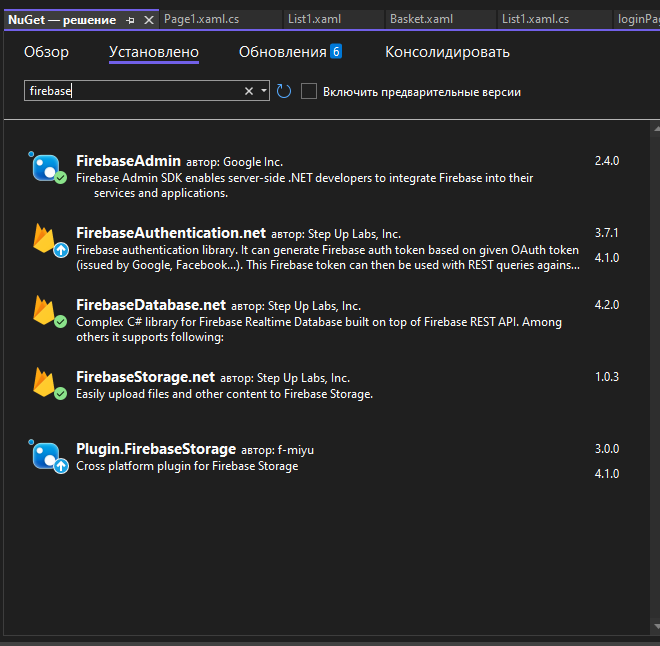


Рис. 5 «Пакеты для работы с firebase»

2.В папку drawable добавлены изображения которые будут использованы в проекте.

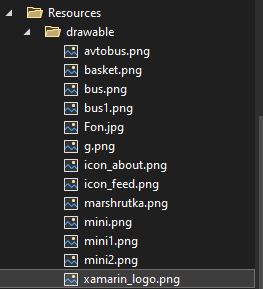


Рис. 6 «Изображения используемые в проекте»

3.Создана страница регистрации.

**Разметка страницы:**

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

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

x:Class="KP.Page1"

>

<Grid>

<Image Source="g.png" Aspect="AspectFill" />

<Frame BackgroundColor="Transparent" HorizontalOptions="Center" VerticalOptions="Center">

<StackLayout HorizontalOptions="Center" VerticalOptions="Center">

<Frame x:Name="ab" BorderColor="White" BackgroundColor="Transparent" Padding="0" CornerRadius="0" HasShadow="False" Margin="0,0, 0, 10">

<Entry x:Name="entry1" WidthRequest="200" Placeholder="Введите email" PlaceholderColor="White" HorizontalTextAlignment="Center" Completed="Login\_Completed1" TextColor="White" FontSize="19" FontAttributes="Bold"/>

</Frame>

<Frame x:Name ="NameFr" BorderColor="White" BackgroundColor="Transparent" Padding="0" CornerRadius="0" HasShadow="False" Margin="0,0, 0, 10">

<Entry x:Name="entryName" WidthRequest="200" Placeholder="Введите Имя" PlaceholderColor="White" HorizontalTextAlignment="Center" Completed="Login\_Completed1" TextColor="White" FontSize="19" FontAttributes="Bold"/>

</Frame>

<Frame x:Name="ac" BorderColor="White" BackgroundColor="Transparent" Padding="0" CornerRadius="0" HasShadow="False" Margin="0,0, 0, 10">

<Entry x:Name="entry2" WidthRequest="200" IsPassword="True" Placeholder="Введите пароль" PlaceholderColor="White" HorizontalTextAlignment="Center" Completed="Password\_Completed2" TextColor="White" FontSize="19" FontAttributes="Bold"/>

</Frame>

<Frame x:Name="ad" BorderColor="White" BackgroundColor="Transparent" Padding="0" CornerRadius="0" HasShadow="False" Margin="0,0, 0, 20">

<Entry x:Name="entry3" WidthRequest="200" IsPassword="True" Placeholder="Подтвердите пароль" PlaceholderColor="White" HorizontalTextAlignment="Center" Completed="Password\_Completed3" TextColor="White" FontSize="19" FontAttributes="Bold"/>

</Frame >

<Label x:Name ="ErrorName" Text="" TextColor="red" FontSize="Body"/>

<Button

x:Name="ButtenRegistration"

Margin="-15,0, 0, 0"

HorizontalOptions="Center"

VerticalOptions="Center"

Text="Уже есть аккаунт?&#10;Войти"

TextColor="LightSkyBlue"

BackgroundColor="Transparent"

Clicked="ButtonRegistration\_Click"

/>

<StackLayout Orientation="Horizontal" Margin="0,0, 0, 10">

<CheckBox x:Name="checkBox" CheckedChanged="CheckBox\_CheckedChanged" Color="AliceBlue" />

<StackLayout Orientation="Vertical">

<Label Text="Соглашаюсь на обработку" TextColor="white" FontSize="Body"/>

<Label Text="персональных данных" TextColor="white" FontSize="Body"/>

</StackLayout>

</StackLayout>

<Button x:Name="ButtenAccount\_registration" Text="Зарегистрироваться" BackgroundColor="LightSkyBlue" TextColor="White" FontAttributes="Bold" CornerRadius="40" HorizontalOptions="Center" VerticalOptions="Center" Command="{Binding SubmitCommand}" Clicked="ButtenAccount\_registration\_Clicked"/>

</StackLayout>

</Frame>

</Grid>

</ContentPage>

**С# код:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Xamarin.Forms;

using Newtonsoft.Json;

using System.IO;

using Google.Cloud.Firestore;

using Google.Apis.Auth.OAuth2;

using Firebase;

using Firebase.Database;

using Google.Cloud.Firestore.V1;

using Grpc.Core;

using Firebase.Database.Query;

using FirebaseAdmin;

using FireSharp.Interfaces;

using Firebase.Auth;

using FireSharp.Response;

using Xamarin.Essentials;

using static Google.Rpc.Context.AttributeContext.Types;

using FirebaseAdmin.Auth;

using System.Text.RegularExpressions;

using Xamarin.Forms.PlatformConfiguration.AndroidSpecific;

namespace KP

{

public partial class Page1 : ContentPage

{

public static string login;

public string password;

public string password1;

public static string Name;

public Page1()

{

InitializeComponent();

ButtenAccount\_registration.IsEnabled = false;

ButtenAccount\_registration.BackgroundColor = Color.Gray;

}

private void Login\_Completed1(object sender, EventArgs e)

{

login = entry1.Text;

}

private void Password\_Completed2(object sender, EventArgs e)

{

password = entry2.Text;

}

private void Password\_Completed3(object sender, EventArgs e)

{

password1 = entry3.Text;

}

private void CheckBox\_CheckedChanged(object sender, CheckedChangedEventArgs e)

{

CheckBox checkBox = (CheckBox)sender;

ButtenAccount\_registration.IsEnabled = e.Value;

if(ButtenAccount\_registration.IsEnabled==false)

{

ButtenAccount\_registration.BackgroundColor= Color.Gray;

}

else

{

ButtenAccount\_registration.BackgroundColor = Color.LightSkyBlue;

}

}

private async void ButtenAccount\_registration\_Clicked(object sender, EventArgs e)

{

registration();

}

private bool IsValidEmail(string email)

{

string emailPattern = @"^([\w\.\-]+)@([\w\-]+)((\.(\w){2,3})+)$";

return Regex.IsMatch(email, emailPattern);

}

private async void registration()

{

login = entry1.Text + "";

password = entry2.Text + "";

password1 = entry3.Text + "";

Name = entryName.Text + "";

login = entry1.Text;

password = entry2.Text;

Name = entryName.Text;

password1 = entry3.Text;

if (string.IsNullOrWhiteSpace(login) || string.IsNullOrWhiteSpace(password) || string.IsNullOrWhiteSpace(password1) || string.IsNullOrWhiteSpace(Name) || password.Length < 6

|| password1 != password || Name.Length < 2)

{

if (string.IsNullOrWhiteSpace(login) || string.IsNullOrWhiteSpace(password) || string.IsNullOrWhiteSpace(password1) || string.IsNullOrWhiteSpace(Name))

{

if (string.IsNullOrWhiteSpace(password))

{

ac.BorderColor = Color.Red;

}

else

{

ac.BorderColor = Color.White;

}

if (string.IsNullOrWhiteSpace(password1))

{

ad.BorderColor = Color.Red;

}

else

{

ad.BorderColor = Color.White;

}

if (string.IsNullOrWhiteSpace(login))

{

ab.BorderColor = Color.Red;

}

else

{

ab.BorderColor = Color.White;

}

if (string.IsNullOrWhiteSpace(Name))

{

NameFr.BorderColor = Color.Red;

}

else

{

NameFr.BorderColor = Color.White;

}

ErrorName.Text = "Поля не должны быть пустыми";

return;

}

if (password.Length < 6)

{

ac.BorderColor = Color.Red;

ErrorName.Text = "Пароль должно содержать минимум 6 символов";

}

else

{

ac.BorderColor = Color.White;

ErrorName.Text = "";

}

if (password1 != password)

{

ad.BorderColor = Color.Red;

ErrorName.Text = "Пароли не совпадают";

}

else

{

ad.BorderColor = Color.White;

ErrorName.Text = "";

}

if (Name.Length < 2)

{

NameFr.BorderColor = Color.Red;

ErrorName.Text = "Имя должно содержать минимум 2 символа";

}

else

{

NameFr.BorderColor = Color.White;

ErrorName.Text = "";

}

if (!IsValidEmail(login))

{

ab.BorderColor = Color.Red;

ErrorName.Text = "Некорректный Email";

}

else

{

ab.BorderColor = Color.White;

}

}

else

{

string apiKey = "AIzaSyDh4J-cAw60c5IyCK89C-B9a7M6\_aKf8eU";

FirebaseAuthProvider firebaseAuthProvider = new FirebaseAuthProvider(new FirebaseConfig(apiKey));

FirebaseAuthLink link1 = await firebaseAuthProvider.CreateUserWithEmailAndPasswordAsync(login, password);

await Navigation.PushModalAsync(new List1());

}

}

private async void ButtonRegistration\_Click(object sender, EventArgs e)

{

await Navigation.PushModalAsync(new loginPage());

}

}

}

4.Создана страница авторизации .

**Разметка страницы:**

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

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

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

x:Class="KP.loginPage">

<ContentPage.Content>

<Grid VerticalOptions="FillAndExpand">

<Image Source="g.jpg" Aspect="AspectFill" />

<Frame BackgroundColor="Transparent" HorizontalOptions="Center" VerticalOptions="Center" >

<StackLayout HorizontalOptions="Center" VerticalOptions="Center" >

<Frame x:Name="ab" BorderColor="White" BackgroundColor="Transparent" Padding="0" CornerRadius="0" HasShadow="False" Margin="0,0, 0, 25" >

<Entry x:Name ="entry1" WidthRequest="300" Placeholder="Введите email" PlaceholderColor="White" HorizontalTextAlignment="Center" Completed="Login\_Completed1" TextColor="White" FontSize="19" FontAttributes="Bold"/>

</Frame>

<Frame x:Name="ac" BorderColor="White" BackgroundColor="Transparent" Padding="0" CornerRadius="0" HasShadow="False" Margin="0,0, 0, 10" >

<Entry x:Name ="entry2" WidthRequest="300" IsPassword="True" Placeholder="Введите пароль" PlaceholderColor="White" HorizontalTextAlignment="Center" Completed="Password\_Completed2" TextColor="White" FontSize="19" FontAttributes="Bold" />

</Frame>

<Frame Padding="0" CornerRadius="0" HasShadow="False" Margin="0,0, 0, 10" BackgroundColor="Transparent">

<Label x:Name="Txt" Text="" TextColor="red" FontSize="16" FontAttributes="Bold" />

</Frame>

<Button

x:Name="ButtenRegistration"

Margin="0,0,0,10"

HorizontalOptions="Center"

VerticalOptions="Center"

Text="Ещё нет аккаунта?&#10;Pегистрация"

TextColor="LightSkyBlue"

BackgroundColor="Transparent"

Clicked="ButtonRegistration\_Click1" />

<Button x:Name="ButtenAccount\_registration" Text="Войти" BackgroundColor="LightSkyBlue" TextColor="White" FontAttributes="Bold" CornerRadius="40" HorizontalOptions="Center" VerticalOptions="Center" Command="{Binding SubmitCommand}" Clicked="ButtenAccount\_registration\_Clicked" Margin="0,0,0,10" HeightRequest="70" WidthRequest="140"/>

</StackLayout>

</Frame>

</Grid>

</ContentPage.Content>

</ContentPage>

**Код C#:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Xamarin.Forms;

using Newtonsoft.Json;

using System.IO;

using Google.Cloud.Firestore;

using Google.Apis.Auth.OAuth2;

using Firebase;

using Firebase.Database;

using Google.Cloud.Firestore.V1;

using Grpc.Core;

using Firebase.Database.Query;

using FirebaseAdmin;

using FireSharp.Interfaces;

using Firebase.Auth;

using FireSharp.Response;

using Xamarin.Essentials;

using static Google.Rpc.Context.AttributeContext.Types;

using Google.Api;

namespace KP

{

public partial class loginPage : ContentPage

{

public static string login;

public string password;

public string loginBD;

public string passwordBD;

public loginPage()

{

InitializeComponent();

}

private void Login\_Completed1(object sender, EventArgs e)

{

Entry entry1 = (Entry)sender;

login = entry1.Text;

}

private void Password\_Completed2(object sender, EventArgs e)

{

Entry entry2 = (Entry)sender;

password = entry2.Text;

}

private async void ButtonRegistration\_Click1(object sender, EventArgs e)

{

await Navigation.PushModalAsync(new Page1());

}

private async void ButtenAccount\_registration\_Clicked(object sender, EventArgs e)

{

string apiKey = "AIzaSyDh4J-cAw60c5IyCK89C-B9a7M6\_aKf8eU";

FirebaseAuthProvider firebaseAuthProvider = new FirebaseAuthProvider(new FirebaseConfig(apiKey));

try

{

login = entry1.Text;

password = entry2.Text;

FirebaseAuthLink link1 = await firebaseAuthProvider.SignInWithEmailAndPasswordAsync(login,password);

await Navigation.PushModalAsync(new List1());

}

catch (FirebaseAuthException ex)

{

Txt.Text = "Неверный логин или пароль";

if (ex.Reason == AuthErrorReason.WrongPassword || ex.Reason == AuthErrorReason.UnknownEmailAddress)

{

Txt.Text = "Неверный логин или пароль";

}

}

}

}

}

## 3.1 Разработка базы данных

* В приложении используется облачная база данных Firebase

Realtime Database, в которой хранится информация о транспорте и заказах пользователей.

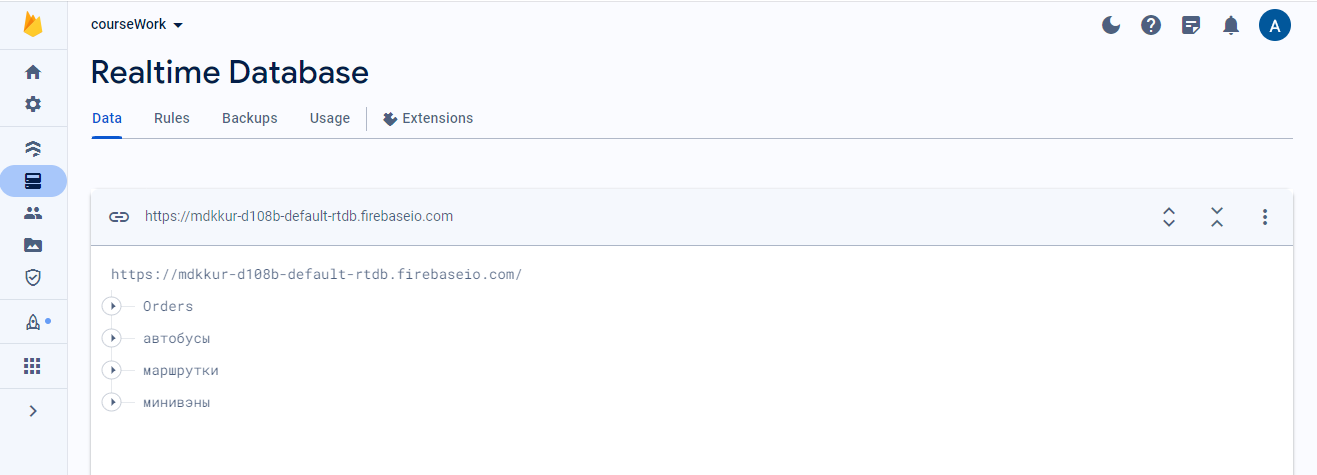


Рис. 1 «Структура базы данных»

* В приложении используется облачная база данных Firebase

Realtime Database, в которой хранится информация о транспорте и

заказах пользователей