Министерство образования и науки РФ

Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский технологический университет «МИСиС»

Институт ИТКН

**Проект**

По дисциплине «Программирование и алгоритмизация»

на тему:

«Мобильное приложение для поиска и покупки авиабилетов»

Выполнил:

студент группы БИВТ-23-5

Разгаров Иван Павлович

Проверил:

к.т.н. доцент Мурадханов С.Э.

Москва 2023

**Оглавление.**

Оглавление

[Введение 2](#_Toc152849003)

[Теоретическая часть 2](#_Toc152849004)

[Практическая часть 2](#_Toc152849005)

[Главная страница 3](#_Toc152849006)

[Разметка страницы в XAML 3](#_Toc152849007)

[Класс страницы в C# 3](#_Toc152849008)

[ViewModel для этой страницы 3](#_Toc152849009)

[Внешний вид (в разработке) 3](#_Toc152849010)

[Страница просмотра результатов поиска 3](#_Toc152849011)

[Разметка страницы в XAML 3](#_Toc152849012)

[Карточка для отображения информации о билете 3](#_Toc152849013)

[ViewModel для страницы 3](#_Toc152849014)

[Внешний вид 3](#_Toc152849015)

[Страница просмотра/покупки билета 3](#_Toc152849016)

[Код страницы в C# 3](#_Toc152849017)

[Внешний вид 3](#_Toc152849018)

# Введение

В данной работе описана разработка мобильного приложения для букинга авиабилетов.

Код доступен по ссылке: <https://github.com/IvanRazgarov/AeroturApp>/

# Теоретическая часть

**Постановка задачи**

Создать приложение, максимально близко повторяющее набор функций и внешний вид уже существующего сайта по поиску билетов. Приложение должно работать на всех популярных мобильных платформах и иметь потенциал для создания версии работающей на персональных компьютерах.

Оригинальный сайт, упомянутый выше можно найти по [ссылке](https://aerotur.aero).

Набор функций, требуемый в минимальном варианте для использования приложения:

Экран ввода критериев поиска билетов (город отправления, город прибытия\*, дата отправления, дата прибытия\*, количество пассажиров, класс полёта. \*-опциональны к заполнению), экран просмотра найденных билетов (на карточке каждого билета отображаются место и дата отлёта и прилёта, авиакомпания предоставитель услуг, кнока «купить» открывающая страницу покупки билета), страница покупки билета (на текущий момент реализуется через страницу оригинального сайта, открываемую во внутреннем браузере, из соображений безопасности).

**Возможные пути реализации**

Для создания многоплатформенных приложений существует несколько фреймворков позволяющий разрабатывать приложения, не создавая специфичный для конкретной платформы код, при этом сохраняя нативные для платформы оформление и управляющие элементы.

* Flutter разработчик: Google 2017г; язык программирования: Dart (by Google)
* React Native разработчик: Facebook 2015г (сейчас Meta Platforms); язык программирования: JavaScript
* Kotlin Multiplatform Mobile разработчик: JetBrains; язык программирования Kotlin (поддерживает только мобильные платформы)
* Xamarin (начиная с .NET 7.0 MAUI) разработчики: Microsoft corporation; язык программирования C#

Для реализации данного проекта был выбран MAUI. Данный фреймворк базируется на платформе .NET, что позволяет разрабатывать приложения для всех устройств, которые поддерживают эту технологию.

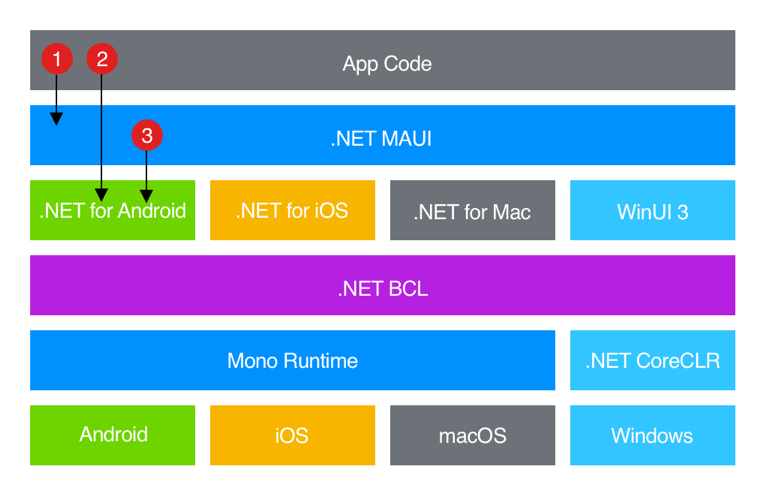
**Multiplatform App UI (MAUI)**

Основан на фреймворке Xamarin с открытым исходным кодом. Использует .NET BCL которая содержит большой набор универсальных функций, таких как работа с XML, базами данных, потоками данных, сетевое взаимодействие и так далее. Позволяет компилировать уже существующий код C# с минимальными изменениями или вообще вез них. Подобный принцип работы позволяет использовать уже существующие библиотеки и ранее написанный код как базу для работы приложения, значительно увеличивая эффективность работы.



**Как работает**

.NET MAUI объединяет API-интерфейсы Android, iOS, macOS, Windows и Tizen в единый API, который предоставляет разработчикам возможность писать код единожды, обеспечивая дополнительный доступ к уникальным функциям платформы.



Также был использован свободно доступный пакет DevExpress.Maui.CollectionView.

**Краткое описание пакета DevExpress.Maui.CollectionView**

Содержит компонент DXCollectionView, который позволяет выводить информацию на экран в виде коллекции объектов, каждый из которых может иметь собственный внешний вид. Позволяет реализовывать бесконечную прокрутку, обновление содержимого по оттягиванию(pull-to-refresh), выбор одного или нескольких элементов, действия при смахивании элемента в сторону, перемещение элементов методом drag-n-drop.

# Практическая часть

Проект основан на структуре MVVM (Model – View – ViewModel), которая позволяет отделять бизнес логику от графического интерфейса. Я использовал стандартный шаблон проекта, предлагаемый компанией-разработчиком.

**По требованиям заказчика были реализованный следующие страницы и сервисы**:

## Главная страница

**Главная страница приложения (на данный момент содержит основную форму поиска)**

Содержит элементы управления, реализующие ввод информации пользователем, и кнопку инициирующую поиск билетов.

### Разметка страницы в XAML

<?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"

xmlns:toolkit="http://schemas.microsoft.com/dotnet/2022/maui/toolkit"

xmlns:vm ="clr-namespace:AeroturApp.Models.ViewModels"

x:Class="AeroturApp.Views.MainPage"

x:DataType="vm:MainViewModel">

<ScrollView>

<VerticalStackLayout Background="{StaticResource PrimaryGradientBrushV}" Padding="0" Margin="0">

<VerticalStackLayout Spacing="8" IsClippedToBounds="True">

<Frame Content="{StaticResource CompanyLogo}" Scale="1.55" BackgroundColor="Transparent"/>

<Frame BackgroundColor="Transparent" BorderColor="Transparent">

<VerticalStackLayout Spacing="5">

<VerticalStackLayout Spacing="5">

<Frame BackgroundColor="White" >

<VerticalStackLayout Padding="0">

<Label Text="Откуда" TextColor="Black" FontSize="30"/>

<Frame Padding="0">

<SearchBar Placeholder="Город вылета"

BackgroundColor="White"

TextColor="Black"

HorizontalOptions="StartAndExpand"

Text="{Binding From\_iata}"

TextTransform="Uppercase">

</SearchBar>

</Frame>

</VerticalStackLayout>

</Frame>

<Frame BackgroundColor="White">

<VerticalStackLayout Padding="0">

<Label Text="Куда" TextColor="Black" FontSize="30"/>

<Frame Padding="0">

<SearchBar Placeholder="Город прилёта"

BackgroundColor="White"

TextColor="Black"

HorizontalOptions="FillAndExpand"

Text="{Binding To\_iata}"

TextTransform="Uppercase">

</SearchBar>

</Frame>

</VerticalStackLayout>

</Frame>

</VerticalStackLayout>

<HorizontalStackLayout HorizontalOptions="Center" Spacing="5" Padding="0" Margin="0">

<Frame BackgroundColor="White" Margin="0" HorizontalOptions="Start" >

<DatePicker x:Name="fromDatePicker"

BackgroundColor="White"

TextColor="Black"

FontSize="21"

Date="{Binding Dep}"

Format="dd'.'MM'.'yyyy"/>

</Frame>

<Frame BackgroundColor="White" Margin="0" HorizontalOptions="Start" >

<DatePicker x:Name="toDatePicker"

BackgroundColor="White"

TextColor="White"

FontSize="21"

Date="{Binding Arr}"

Format="dd'.'MM'.'yyyy"

DateSelected="toDatePicker\_DateSelected"/>

</Frame>

</HorizontalStackLayout>

<Frame BackgroundColor="White">

<toolkit:Expander BackgroundColor="White" >

<toolkit:Expander.Header>

<VerticalStackLayout Padding="0" Margin="0" Spacing="0">

<Label Text="Пассажиры" TextColor="Black" FontSize="30" />

<Label Text="{Binding Total\_people, StringFormat='{0} человек(а)'}" TextColor="Black" FontSize="20"/>

</VerticalStackLayout>

</toolkit:Expander.Header>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="4\*" />

<ColumnDefinition Width="3\*" />

<ColumnDefinition Width="3\*" />

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="40\*" />

<RowDefinition Height="40\*" />

<RowDefinition Height="40\*" />

<RowDefinition Height="40\*" />

</Grid.RowDefinitions>

<Label Text="Взрослые(старше 12 лет)" TextColor="Black" VerticalOptions="Center" HorizontalOptions="StartAndExpand"/>

<Label Grid.Row="1" Text="Дети(2-12 лет)" TextColor="Black" VerticalOptions="Center" HorizontalOptions="StartAndExpand"/>

<Label Grid.Row="2" Text="Младенцы(до 2 лет)" TextColor="Black" VerticalOptions="Center" HorizontalOptions="StartAndExpand"/>

<Label Grid.Row="3" Text="Младенцы с местом( до 2 лет)" TextColor="Black" VerticalOptions="Center" HorizontalOptions="StartAndExpand"/>

<Stepper x:Name="adults" Value="{Binding Adults}" Grid.Column="1" Maximum="12" Minimum="1" />

<Stepper x:Name="children" Value="{Binding Children}" Grid.Column="1" Grid.Row="1" Maximum="4" Minimum="0"/>

<Stepper x:Name="infants" Value="{Binding Infants}" Grid.Column="1" Grid.Row="2" Maximum="2" Minimum="0"/>

<Stepper x:Name="infants\_seat" Value="{Binding Infants\_seat}" Grid.Column="1" Grid.Row="3" Maximum="4" Minimum="0"/>

<Label Grid.Column="2" Text="{Binding Adults}" TextColor="Black" HorizontalOptions="Center" VerticalOptions="Center"/>

<Label Grid.Column="2" Grid.Row="1" Text="{Binding Children}" TextColor="Black" HorizontalOptions="Center" VerticalOptions="Center"/>

<Label Grid.Column="2" Grid.Row="2" Text="{Binding Infants}" TextColor="Black" HorizontalOptions="Center" VerticalOptions="Center"/>

<Label Grid.Column="2" Grid.Row="3" Text="{Binding Infants\_seat}" TextColor="Black" HorizontalOptions="Center" VerticalOptions="Center"/>

</Grid>

</toolkit:Expander>

</Frame>

<Frame BackgroundColor="White">

<VerticalStackLayout>

<Label Text="Класс полёта" TextColor="Black" FontSize="20"/>

<Picker x:Name="flight\_class"

ItemsSource="{Binding PricingTypes}"

SelectedItem="{Binding PricingType}"

SelectedIndex="0"

BackgroundColor="White"

TextColor="Black"

TitleColor="Black"

/>

</VerticalStackLayout>

</Frame>

<Button Text="Найти"

FontSize="30"

TextColor="White"

Background="{StaticResource AccentGradientBrushV}"

Command="{Binding NavigateCommand}"

/>

</VerticalStackLayout>

</Frame>

</VerticalStackLayout>

</VerticalStackLayout>

</ScrollView>

</ContentPage>

### Класс страницы в C#

using AeroturApp.Models.ViewModels;

namespace AeroturApp.Views;

public partial class MainPage : ContentPage

{

public MainPage(MainViewModel model)

{

InitializeComponent();

BindingContext = model;

}

private void toDatePicker\_DateSelected(object sender, DateChangedEventArgs e)

{

(sender as DatePicker).TextColor = Colors.Black;

}

}

### ViewModel для этой страницы

using AeroturApp.Models.DataModels;

using AeroturApp.Services;

using AeroturApp.Views;

using CommunityToolkit.Mvvm.ComponentModel;

using CommunityToolkit.Mvvm.Input;

using System.Collections.ObjectModel;

namespace AeroturApp.Models.ViewModels;

public partial class MainViewModel : ObservableObject

{

private IataCodesService codesService;

private Dictionary<string, string> translatePricingName = new()

{

{ "Эконом", "Economy" },

{ "Бизнес", "Business" },

{ "Первый", "First"}

};

private SearchParams { get; set; }

[ObservableProperty]

List<string> pricingTypes = ["Эконом", "Бизнес", "Первый"];

[ObservableProperty]

string pricingType = "Эконом";

[ObservableProperty]

ObservableCollection<IATA\_City> suggestionsFrom = new();

[ObservableProperty]

IATA\_City suggestionFrom = new();

[ObservableProperty]

ObservableCollection<IATA\_City> suggestionsTo = new();

[ObservableProperty]

IATA\_City suggestionTo = new();

[ObservableProperty]

int adults = 1;

[ObservableProperty]

int children = 0;

[ObservableProperty]

int infants = 0;

[ObservableProperty]

int infants\_seat = 0;

partial void OnAdultsChanged(int value)

{

Total\_people = value + children + infants + infants\_seat;

}

partial void OnChildrenChanged(int value)

{

Total\_people = adults + value + infants + infants\_seat;

}

partial void OnInfantsChanged(int value)

{

Total\_people = adults + children + value + infants\_seat;

}

partial void OnInfants\_seatChanged(int value)

{

Total\_people = adults + children + infants + value;

}

[ObservableProperty]

int total\_people=1;

[ObservableProperty]

string from\_iata = "MOW";

[ObservableProperty]

string to\_iata = "TAS";

[ObservableProperty]

DateTime dep = DateTime.Now;

//[ObservableProperty]

DateTime? arr = null;

public DateTime? Arr

{

get

{

if(arr == null || arr <= DateTime.Now) return null;

else return arr;

}

set => SetProperty(ref arr, value);

}

[ObservableProperty]

string date1 = DateTime.Now.ToString("dd'.'MM'.'yyyy");

[ObservableProperty]

string date2 = DateTime.Now.ToString("dd'.'MM'.'yyyy");

public MainViewModel()

{

//codesService = iataCodesService;

/\*searchParams = new SearchParams()

{

locale = "RU",

instance = "aerotur.aero.dev",

adults = 1,

children = 0,

infants = 0,

infants\_seat = 0,

flight\_class = "Economy",

from = "LED",

fromType = "city",

to = "OSS",

toType = "city",

date1 = DateTime.Now.ToString("yyyy'-'MM'-'dd"),

date2 = null,

asGrouped = 0

};\*/

}

[RelayCommand]

Task Navigate()

=> Shell.Current.GoToAsync($"{nameof(SearchResultPage)}", new Dictionary<string, object>()

{

{"SearchParams", new SearchParams()

{

locale = "RU",

instance = "aerotur.aero.dev",

adults = Adults,

children = Children,

infants = Infants,

infants\_seat = Infants\_seat,

flight\_class = translatePricingName[PricingType??"Эконом"],

from = From\_iata,

fromType = "city",

to = To\_iata,

toType = "city",

date1 = Dep.ToString("yyyy'-'MM'-'dd"),//DateTime.Now.AddDays(1).ToString("yyyy'-'MM'-'dd"),

date2 = Arr?.ToString("yyyy'-'MM'-'dd") ?? null,

asGrouped = 0

}

}

});

[RelayCommand]

Task GetSuggestions(string input)

=> GetIataCode(input);

private async Task GetIataCode(string input)

{

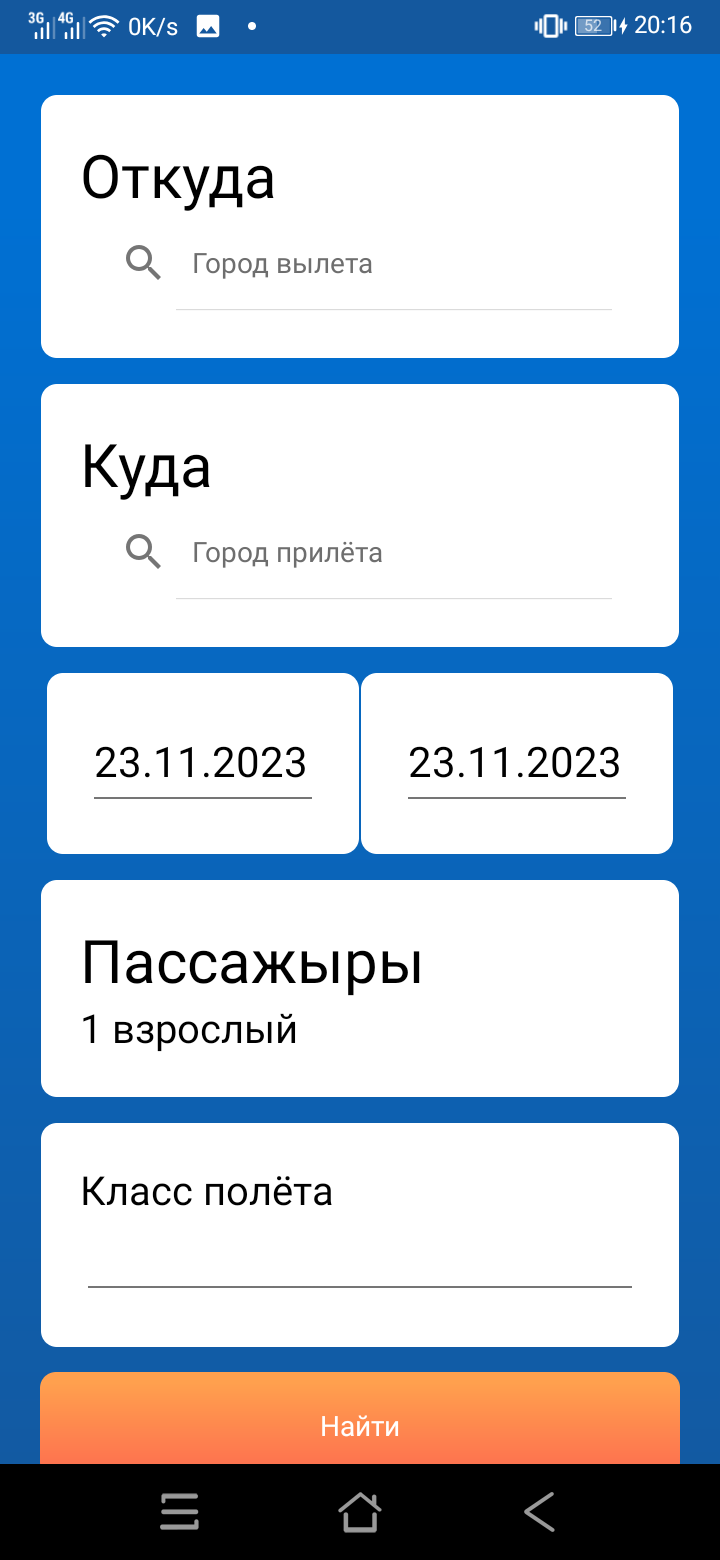
SuggestionsFrom = await codesService.GetCityCodeByPart(input);

SuggestionsTo = await codesService.GetCityCodeByPart(input);

}

}

### Внешний вид (в разработке)



## Страница просмотра результатов поиска

Страница выводящая все найденные по запросу билеты с информацией о них и возможностью перейти на страницу покупки конкретного билета.

### Разметка страницы в XAML

<?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"

xmlns:toolkit="http://schemas.microsoft.com/dotnet/2022/maui/toolkit"

xmlns:vm="clr-namespace:AeroturApp.Models.ViewModels"

xmlns:m="clr-namespace:AeroturApp.Models.DataModels"

xmlns:dx="clr-namespace:DevExpress.Maui.CollectionView;assembly=DevExpress.Maui.CollectionView"

xmlns:local="clr-namespace:AeroturApp"

x:Class="AeroturApp.Views.SearchResultPage"

Title="Результаты поиска"

x:DataType="vm:SearchResultsViewModel"

>

<Shell.BackButtonBehavior >

<BackButtonBehavior Command="{Binding ReturnToPreviousCommand}"/>

</Shell.BackButtonBehavior>

<dx:DXCollectionView

ItemsSource="{Binding Flights}"

IsPullToRefreshEnabled="True"

IndicatorColor="Orange"

IsRefreshing="{Binding IsBusy}"

PullToRefreshCommand="{Binding RetrySearchCommand}"

Background="{StaticResource PrimaryGradientBrushV}"

ItemTemplate="{StaticResource FlightCard\_small}"

>

</dx:DXCollectionView>

</ContentPage>

### Карточка для отображения информации о билете

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

<?xaml-comp compile="true" ?>

<ResourceDictionary

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

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

xmlns:m="clr-namespace:AeroturApp.Models.DataModels"

xmlns:toolkit="http://schemas.microsoft.com/dotnet/2022/maui/toolkit"

xmlns:vm="clr-namespace:AeroturApp.Models.ViewModels">

<DataTemplate x:Key="FlightCard\_big" x:DataType="m:Variant">

<Frame Padding="1" Margin="5" BackgroundColor="{AppThemeBinding Light={StaticResource Black}, Dark={StaticResource White}, Default={StaticResource White}}">

<VerticalStackLayout>

<Frame Margin="0">

<VerticalStackLayout >

<HorizontalStackLayout>

<Label Text="{Binding price}" FontSize="30"/>

<Label Text="{Binding currency}" FontSize="30"/>

</HorizontalStackLayout>

<HorizontalStackLayout HorizontalOptions="Center" Padding="0" Spacing="10" Margin="0">

<VerticalStackLayout Spacing="10" >

<Label Text="{Binding legs[0].segments[0].departure\_airport}" HorizontalOptions="Center"/>

<Label Text="{Binding legs[0].segments[0].ddt, StringFormat='{0:dd} {0:MMM}'}" HorizontalOptions="Center"/>

<!--<Label Text="{Binding expected\_ticket\_count}" HorizontalOptions="Start" TextColor="white"/>-->

</VerticalStackLayout>

<Label Text="{Binding delta\_travel\_time, StringFormat='Время в пути: {0:%h} ч. {0:%m} м.'}"/>

<VerticalStackLayout Spacing="10" >

<Label Text="{Binding arrival\_city}" HorizontalOptions="Center"/>

<Label Text="{Binding arrival\_date\_time, StringFormat='{0:dd} {0:MMM}'}" HorizontalOptions="Center"/>

</VerticalStackLayout>

</HorizontalStackLayout>

<Frame Padding="0" Margin="0">

<HorizontalStackLayout>

<!--<Grid.RowDefinitions>

<RowDefinition Height="0.5\*"/>

<RowDefinition Height="0.5\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="0.75\*"/>

<ColumnDefinition Width="0.25\*"/>

</Grid.ColumnDefinitions>-->

<!--<Button Text="other trips"

Grid.Column="1"

Grid.RowSpan="2"

HorizontalOptions="CenterAndExpand"

VerticalOptions="Center"

Background="{StaticResource AccentGradientBrushV}"/>-->

</HorizontalStackLayout>

</Frame>

<HorizontalStackLayout>

<Label Text="transit delta time placeholder" HorizontalOptions="Center"/>

</HorizontalStackLayout>

</VerticalStackLayout>

</Frame>

</VerticalStackLayout>

</Frame>

</DataTemplate>

<DataTemplate x:Key="FlightCard\_small" x:DataType="m:Variant">

<Frame Padding="5" Margin="5" BackgroundColor="{AppThemeBinding Light={StaticResource White}, Dark={StaticResource Black}, Default={StaticResource White}}">

<VerticalStackLayout>

<Frame CornerRadius="8">

<VerticalStackLayout >

<HorizontalStackLayout HorizontalOptions="CenterAndExpand" Padding="0" Spacing="10">

<VerticalStackLayout Spacing="1" HorizontalOptions="StartAndExpand">

<Label Text="{Binding legs[0].segments[0].departure\_airport}" FontSize="25" HorizontalOptions="Center"/>

<Label Text="{Binding legs[0].segments[0].ddt, StringFormat='{0:%d} {0:MMM}'}" FontSize="18" HorizontalOptions="Center"/>

</VerticalStackLayout>

<VerticalStackLayout HorizontalOptions="CenterAndExpand">

<Label Text="{Binding legs[0].segments[0].operating\_company}" FontSize="25" HorizontalOptions="Center"/>

</VerticalStackLayout>

<VerticalStackLayout Spacing="1" HorizontalOptions="EndAndExpand">

<Label Text="{Binding arrival\_city}" FontSize="25" HorizontalOptions="Center"/>

<Label Text="{Binding arrival\_date\_time, StringFormat='{0:%d} {0:MMM}'}" FontSize="18" HorizontalOptions="Center"/>

</VerticalStackLayout>

</HorizontalStackLayout>

<Label Text="{Binding delta\_travel\_time, StringFormat='Время в пути: {0:%h} ч. {0:%m} м.'}" HorizontalOptions="CenterAndExpand"/>

</VerticalStackLayout>

</Frame>

<HorizontalStackLayout Spacing="5" HorizontalOptions="CenterAndExpand">

<Label Text="{Binding price, StringFormat='{0:C}'}" FontSize="30" HorizontalOptions="Center" VerticalOptions="Center"/>

<Button Text="Купить"

HorizontalOptions="End"

FontSize="30"

Background="{StaticResource AccentGradientBrushV}"

Command="{Binding Source={RelativeSource AncestorType={x:Type vm:SearchResultsViewModel}}, Path=NavigateToSelectedCommand}"

CommandParameter="{Binding booking\_url}"/>

</HorizontalStackLayout>

<Frame BackgroundColor="#31000000" Padding="0" Margin="0">

<toolkit:Expander Margin="0" Padding="0" IsClippedToBounds="True" IsExpanded="{Binding is\_expanded}">

<toolkit:Expander.Header>

<Frame BackgroundColor="White" Padding="0" Margin="0" CornerRadius="0" IsClippedToBounds="True">

<Label Text="Детали"/>

</Frame>

</toolkit:Expander.Header>

<Frame Margin="0" Padding="5" CornerRadius="8" >

<VerticalStackLayout HorizontalOptions="FillAndExpand">

<VerticalStackLayout>

<HorizontalStackLayout>

<Label Text="Туда "/>

<Path VerticalOptions="Center" Opacity="0.25" Fill="#1864B0" Data="M2.408 8.719l1.108-3.586H1.898l-1.002.896H0L.72 4.5 0 2.97h.896l1.002.897h1.618L2.408.281h.897l2.004 3.586h2.46c.07.012.176.024.317.035.234.035.375.07.422.106.105.07.21.135.316.193.164.07.211.17.14.299 0 .188-.11.334-.333.44-.211.093-.434.152-.668.175l-.334.018h-2.32L3.305 8.719h-.897z"/>

<Label Text="{Binding delta\_travel\_time, StringFormat='{0:%h} ч. {0:%m} м.'}"/>

</HorizontalStackLayout>

<HorizontalStackLayout>

<Label Text="{Binding legs[0].segments[0].departure\_airport}"/>

<Path Fill="Black" Data="M0 9l10 0c1 0 1 0-2-3l1-1 3 3c2 1 2 2 0 3l-3 3-1-1c3-3 3-3 2-3l-10 0Z"/>

<Label Text="{Binding arrival\_city}"/>

</HorizontalStackLayout>

</VerticalStackLayout>

<VerticalStackLayout BindableLayout.ItemsSource="{Binding legs[0].segments}" Margin="0" Padding="0" Spacing="10" ZIndex="1">

<BindableLayout.ItemTemplate>

<DataTemplate x:DataType="m:Segment">

<VerticalStackLayout Padding="0" Margin="0" VerticalOptions="Fill" HorizontalOptions="Fill">

<Frame Margin="0" >

<VerticalStackLayout HorizontalOptions="FillAndExpand" Margin="0" Padding="0" Spacing="10">

<HorizontalStackLayout HorizontalOptions="CenterAndExpand" Spacing="200">

<Label Text="{Binding operating\_company}" HorizontalOptions="Start" VerticalOptions="Center"/>

<VerticalStackLayout HorizontalOptions="End" VerticalOptions="Center">

<Label Text="{Binding aircraft\_type}"/>

<Label Text="{Binding flight\_number\_full}"/>

</VerticalStackLayout>

</HorizontalStackLayout>

<Grid HorizontalOptions="FillAndExpand">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition/>

<ColumnDefinition Width="auto"/>

</Grid.ColumnDefinitions>

<VerticalStackLayout VerticalOptions="Center">

<Label Text="{Binding ddt, StringFormat='{0:%h}:{0:mm}'}"/>

<Label Text="{Binding ddt, StringFormat='{0:M}'}"/>

</VerticalStackLayout>

<Path Grid.Column="1" Grid.RowSpan="3" HorizontalOptions="Center" VerticalOptions="Center"

Data=""/>

<Label Grid.Column="2" Text="{Binding departure\_airport}" HorizontalOptions="Center" VerticalOptions="Center"/>

<Label Grid.Row="1" Grid.Column="2" Text="{Binding travel\_time, StringFormat='Время перелёта:/n {0:%h} ч. {0:%m} м.'}"

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

<VerticalStackLayout Grid.Row="2" VerticalOptions="Center">

<Label Text="{Binding adt, StringFormat='{0:%h}:{0:mm}'}"/>

<Label Text="{Binding adt, StringFormat='{0:M}'}"/>

</VerticalStackLayout>

<Label Grid.Row="2" Grid.Column="2" Text="{Binding arrival\_airport}"

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

</Grid>

</VerticalStackLayout>

</Frame>

<Border Padding="0" Margin="0"

Stroke="#61B8FF"

StrokeShape="RoundRectangle 50"

AbsoluteLayout.LayoutFlags="All"

AbsoluteLayout.LayoutBounds="0.25,0.875,1,1">

<Frame Padding="0" Margin="0">

<HorizontalStackLayout>

<Path/>

<Label Text=""/>

</HorizontalStackLayout>

</Frame>

</Border>

</VerticalStackLayout>

</DataTemplate>

</BindableLayout.ItemTemplate>

</VerticalStackLayout>

</VerticalStackLayout>

</Frame>

</toolkit:Expander>

</Frame>

</VerticalStackLayout>

</Frame>

</DataTemplate>

</ResourceDictionary>

### ViewModel для страницы

using AeroturApp.Models.DataModels;

using AeroturApp.Services;

using CommunityToolkit.Mvvm.ComponentModel;

using CommunityToolkit.Mvvm.Input;

using System.Collections.ObjectModel;

namespace AeroturApp.Models.ViewModels;

//[QueryProperty(nameof(DebugInfo), nameof(DebugInfo))]

public partial class SearchResultsViewModel : ObservableObject, IQueryAttributable

{

[ObservableProperty]

private SearchParams searchParams;

public void ApplyQueryAttributes(IDictionary<string, object> query)

{

SearchParams = query[nameof(SearchParams)] as SearchParams;

Task.Run(() => GetSearchForFlights(SearchParams));

}

//private SearchReturn results;

//public void ApplyQueryAttributes(IDictionary<string, object> query)

//{

// DebugInfo = query[nameof(DebugInfo)].ToString();

// OnPropertyChanged(nameof(DebugInfo));

//}

private List<Variant> variants = new();

[ObservableProperty]

public string debugInfo = "No debug info";

[ObservableProperty]

private bool isBusy = false;

[ObservableProperty]

private ObservableCollection<Variant> \_flights;

[ObservableProperty]

private Variant \_flight = new();

private WebAPIClient \_client;

public SearchResultsViewModel(WebAPIClient client)

{

\_client = client;

/\*searchParams = new SearchParams()

{

locale = "RU",

instance = "aerotur.aero.dev",

adults = 1,

children = 0,

infants = 0,

infants\_seat = 0,

flight\_class = "Economy",

from = "LED",

fromType = "city",

to = "OSS",

toType = "city",

date1 = DateTime.Now.AddDays(1).ToString("yyyy'-'MM'-'dd"),

date2 = null,

asGrouped = 0

};

GetSearchForFlights(searchParams);

AddPartToCollection(8);\*/

}

[RelayCommand]

Task ReturnToPrevious() => Shell.Current.GoToAsync($"..");

//[RelayCommand]

//void LoadMoreItems() => AddPartToCollection();

[RelayCommand]

Task RetrySearch() => GetSearchForFlights(SearchParams);

[RelayCommand]

Task NavigateToSelected(string link)

=> Shell.Current.GoToAsync($"PaymentPage", new Dictionary<string, object>()

{

["link"]=link

});

public async Task GetSearchForFlights(SearchParams pars)

{

IsBusy = true;

var res = await \_client.SearchForFlights(pars);

if (!res.is\_valid || res.variants == null)

{

DebugInfo=res.error\_msg;

IsBusy = false;

return;

}

variants = new List<Variant>(res.variants);

Flights = new ObservableCollection<Variant>(variants);

IsBusy = false;

}

public async Task GetSearchForFlights()

{

IsBusy = true;

var res = await \_client.SearchForFlights(SearchParams);

if (!res.is\_valid || res.variants == null)

{

DebugInfo = "";

IsBusy = false;

return;

}

variants = new List<Variant>(res.variants);

Flights = new ObservableCollection<Variant>(variants);

await Task.Yield();

IsBusy = false;

}

public void AddPartToCollection()

{

if (variants != null)

{

var length = variants.Count;

var len = (int)length / 4;

if (len <= 0) { return; }

var part = variants[0..^len];

variants.RemoveRange(0, len);

foreach (var item in part)

{

Flights.Add(item);

}

}

}

public void AddPartToCollection(int partSize)

{

if (variants != null)

{

var length = variants.Count;

var len = (int)length / partSize;

if (len <= 0) { return; }

var part = variants[0..^len];

variants.RemoveRange(0, len);

foreach (var item in part)

{

Flights.Add(item);

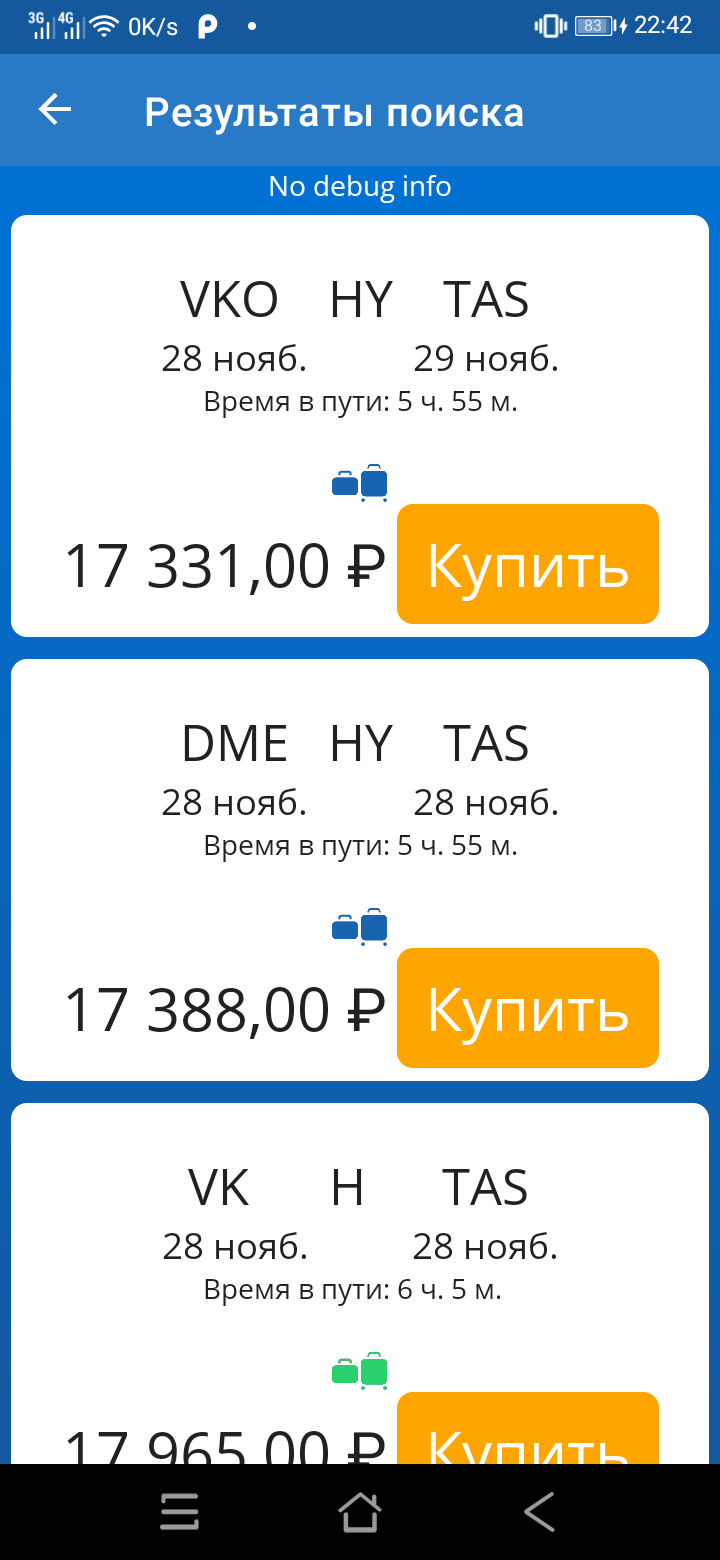
}

}

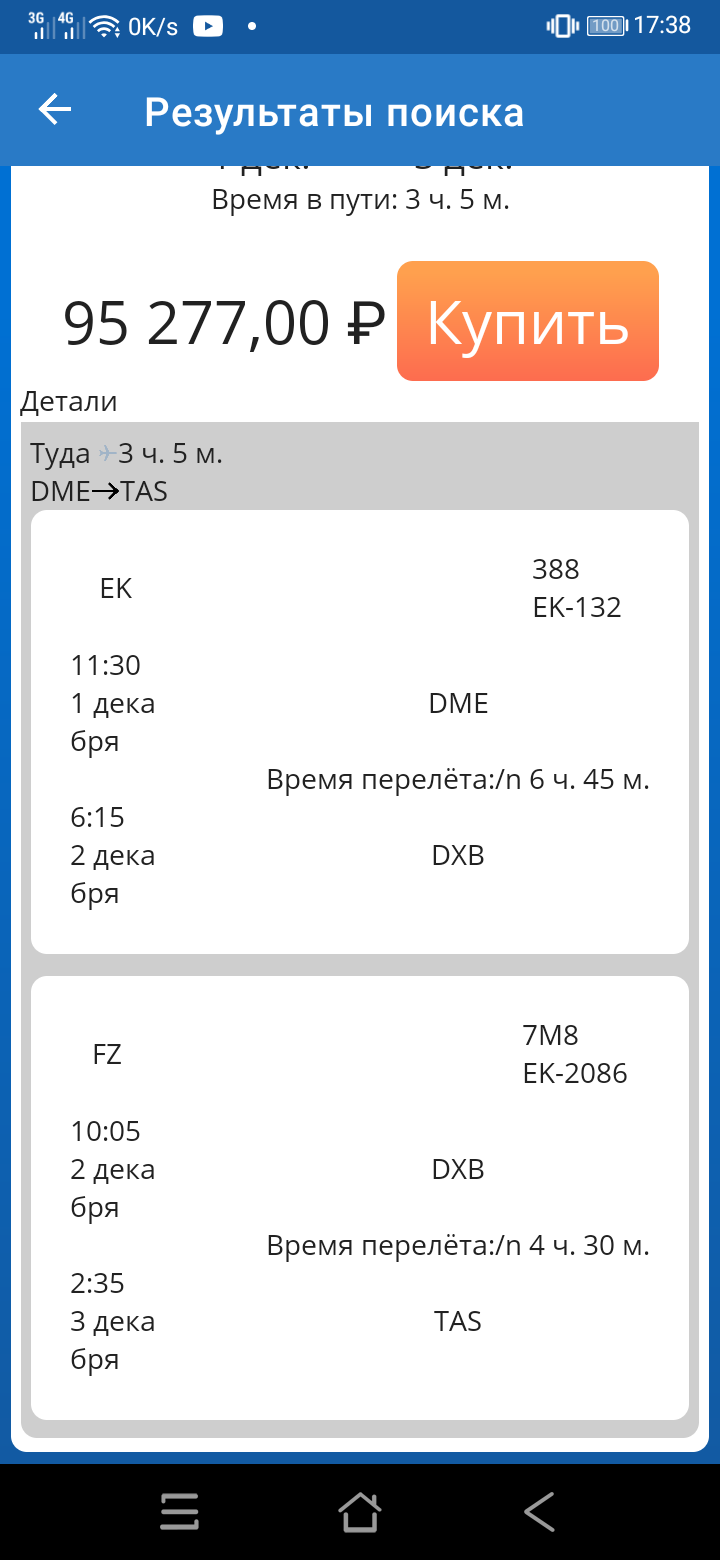
}

}

### Внешний вид



Внешний вид карточки конкретного билета



## Страница просмотра/покупки билета

На данный момент реализованная через открытие страницы с главного сайта в невидимом браузере, в целях безопасности.

### Код страницы в C#

Так как страница содержит всего один элемент, я посчитал нецелесообразным размечать её с использованием XAML

using System.Diagnostics;

namespace AeroturApp.Views;

public class PaymentPage : ContentPage, IQueryAttributable

{

private string link = "";

public void ApplyQueryAttributes(IDictionary<string, object> query)

{

link = query["link"] as string;

Debug.WriteLine(link);

}

public PaymentPage()

{

Content = new WebView();

}

protected override void OnAppearing()

{

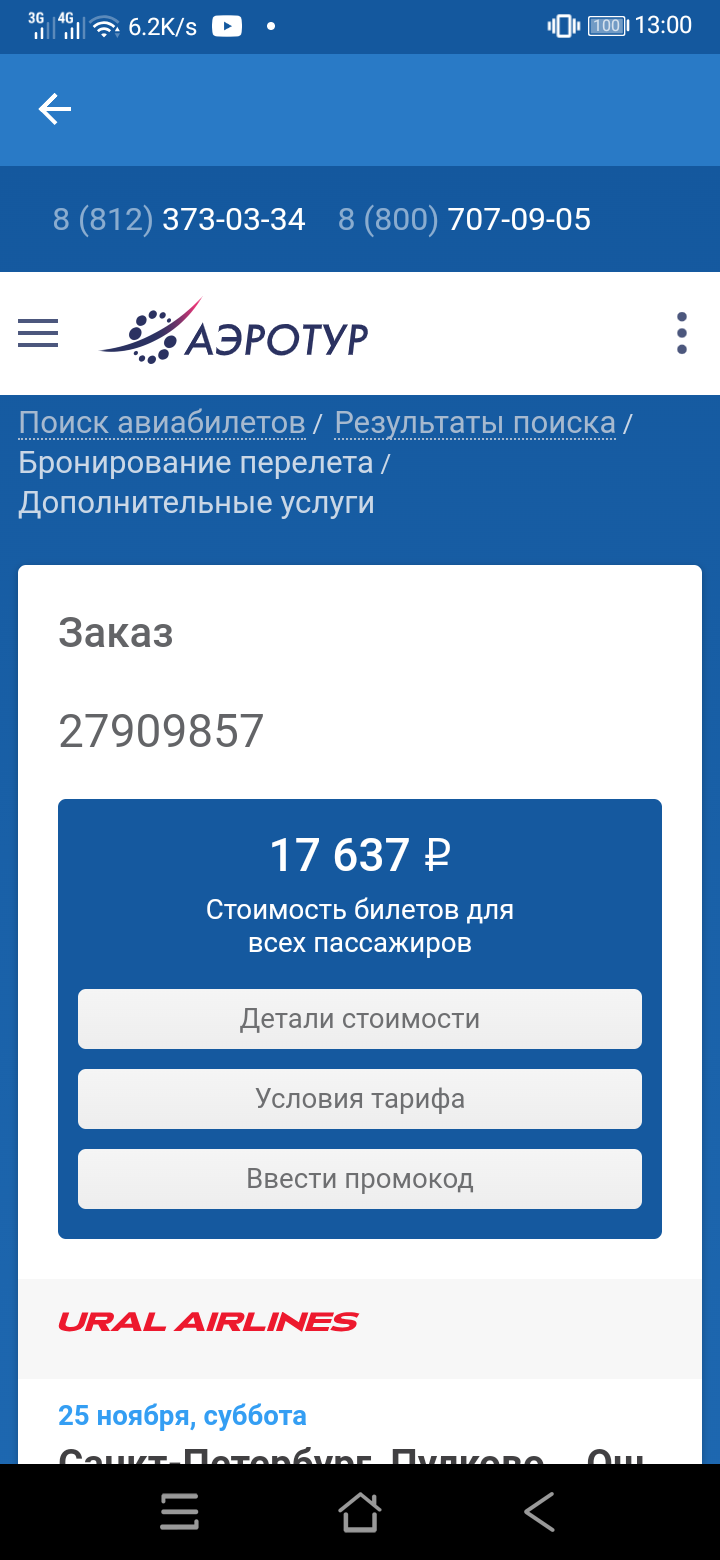
base.OnAppearing();

(Content as WebView).Source = link;

}

}

### Внешний вид



## Сервис работы с веб АПИ

Информация о билетах и данные для перевода ИАТА кодов в названия поступает из удалённых источников, в ответ на POST запросы с соответствующими данными.

using AeroturApp.Models.DataModels;

using System.Diagnostics;

using System.Net;

using System.Net.Http.Json;

using System.Text.Json;

using System.Text.RegularExpressions;

#if ANDROID

using Xamarin.Android.Net;

#endif

namespace AeroturApp.Services

{

public class WebAPIClient

{

private HttpClient;

//private HttpRequestMessage? mainReqMessage;

private static CookieContainer cookieContainer = new CookieContainer();

#if ANDROID

private AndroidMessageHandler httpMessageHandler = new AndroidMessageHandler()

{

AllowAutoRedirect = true,

UseCookies = true,

CookieContainer = cookieContainer

};

#else

private HttpClientHandler httpMessageHandler = new HttpClientHandler()

{

AllowAutoRedirect = true,

UseCookies = true,

CookieContainer = cookieContainer

};

#endif

private const string AEROTUR\_API\_URL = "https://api-dev.aerotur.aero/api/";

private const string AVIASALES\_API\_URL = "https://api.travelpayouts.com";

public WebAPIClient()

{

httpClient = new HttpClient(httpMessageHandler, false)

{

//BaseAddress = new Uri(AEROTUR\_API\_URL)

};

}

/// <summary>

/// Пармаметры поиска, кроме date2 не должны быть пустыми. Возвращает объект результатов.

/// </summary>

/// <param name="creterias"></param>

/// <returns></returns>

public async Task<SearchReturn> SearchForFlights(SearchParams creterias)

{

/\*var searchParams = new SearchParams()

{

locale = "RU",

instance = "aerotur.aero.dev",

adults = 1,

children = 0,

infants = 0,

infants\_seat = 0,

flight\_class = "Economy",

from = "LED",

fromType = "city",

to = "OSS",

toType = "city",

date1 = "2023-11-04",

date2 = null,

asGrouped = 0

};\*/

var mainReqMessage = new HttpRequestMessage(HttpMethod.Post, AEROTUR\_API\_URL+"flights");

mainReqMessage.Content = JsonContent.Create(creterias);

var searchRes = await httpClient.SendAsync(mainReqMessage);

if (!searchRes.IsSuccessStatusCode)

{

return new SearchReturn();

//throw new Exception("Status code: "+searchRes.StatusCode+' '+searchRes.Content.ReadAsStringAsync().Result);

}

var resText = await searchRes.Content.ReadAsStringAsync();

var res = new SearchReturn();

try

{

res = JsonSerializer.Deserialize<SearchReturn>(resText);

}

catch (Exception ex)

{

mainReqMessage = new HttpRequestMessage(HttpMethod.Post, AEROTUR\_API\_URL + "flights");

creterias.date1 = DateTime.Parse(creterias.date1).AddDays(1).ToString("yyyy'-'MM'-'dd");

mainReqMessage.Content = JsonContent.Create(creterias);

searchRes = await httpClient.SendAsync(mainReqMessage);

resText = await searchRes.Content.ReadAsStringAsync();

res = JsonSerializer.Deserialize<SearchReturn>(resText);

}

catch

{

res = new SearchReturn() { error\_msg = resText + " No flights for date or tomorrow " + creterias };

//res.is\_valid=false;

//res.error\_msg =ex.Message+" , "+creterias.date1 +" , "+ resText;

}

foreach(var variant in res.variants)

{

foreach(var leg in variant.legs)

{

foreach(var segment in leg.segments)

{

var i = 0;

if (i == leg.segments.Count - 1) segment.is\_last=true;

}

}

}

return res;

}

public async Task<List<T>> GetIATAFromNet<T>()

{

var tail = "/data/ru/";

var payload = new Dictionary<Type, Action>()

{

{typeof(IATA\_Airport),()=>tail="airports.json?\_gl=1\*ajmg2k\*\_ga\*MjEzNDk5MDg2MC4xNjk3NTY5NDQ0\*\_ga\_1WLL0NEBEH\*MTcwMDMzOTg3OC4xNC4wLjE3MDAzMzk4NzguNjAuMC4w" },

{typeof(IATA\_City), ()=>tail="cities.json?\_gl=1\*1o8h4ms\*\_ga\*MjEzNDk5MDg2MC4xNjk3NTY5NDQ0\*\_ga\_1WLL0NEBEH\*MTcwMDMzOTg3OC4xNC4xLjE3MDAzNDA2MDIuNjAuMC4w" }

};

var request = new HttpRequestMessage(HttpMethod.Get, AVIASALES\_API\_URL+tail+payload);

var searchRes = await httpClient.SendAsync(request);

if (!searchRes.IsSuccessStatusCode)

{

return new List<T>();

//throw new Exception("Status code: "+searchRes.StatusCode+' '+searchRes.Content.ReadAsStringAsync().Result);

}

var resText = await searchRes.Content.ReadAsStringAsync();

var res = new List<T>();

try

{

res = JsonSerializer.Deserialize<List<T>>(resText);

}

catch (Exception ex)

{

Debug.WriteLine(ex);

}

return res;

}

}

}

## Дальнейшие доработки проекта

Очень много работы над графической составляющей. Существует готовый дизайн, к которому нужно максимально приблизится. Увидеть целевой дизайн приложения можно по [ссылке](https://www.figma.com/file/htxSDE0D4TznwmhVWqtg7j/✈-Ticket-Search?type=design&node-id=67-8946&mode=design&t=BQdbug81ShC69OYY-0).

Требуется создать и настроить сервис перевода ИАТА кодов городов, аэропортов, стран в их названия.

На момент написания документа отображение объектов на странице результатов поиска не оптимизированно для мобильных устройств. Требуется найти доступный способ оптимизации процесса рендеринга элементов коллекции.

Страница покупки билета по сути не реализована. Создание этой страницы с соответствующими механизмами и уровнем защиты может стать темой отдельного проекта.

# Заключение

В результате проведённой работы было создано приложение, реализующее минимально необходимый набор функций. Приложение нормально функционирует на всех вышеперечисленных платформах. Результат удовлетворяет заказчика при условии продолжения доработок.

# Использованные источники

Официальная документация Microsoft по языку C#: <https://learn.microsoft.com/ru-ru/dotnet/csharp/>

Официальная документация Microsoft по MAUI: <https://learn.microsoft.com/ru-ru/dotnet/maui/what-is-maui?view=net-maui-8.0>

Решения отдельных проблем с сайтов Stack Overflow и Habr

Книга по языку C#: Мурадханов С.Э. Широков А.И. учебник «Информатика и программирование.»