```
using System;
using System.Collections.Generic;
using System.Net.Http;
using System.Text.Json;
using Xamarin.Essentials;
using Xamarin.Forms;

namespace Skyye
{
    public partial class Airports : ContentPage
    {
        SearchBar searchBar = null;

        class AirportsData
        {
            // Root myDeserializedClass =
JsonConvert.DeserializeObject<Root>(myJsonResponse);
            public class Agent
            {
            }

            public class Client
            {
                public string ip { get; set; }
                public Geo geo { get; set; }
                public Connection connection { get; set; }
                public Device device { get; set; }
                public Agent agent { get; set; }
                public Karma karma { get; set; }
            }

            public class Connection
            {
                public string type { get; set; }
                public int isp_code { get; set; }
                public string isp_name { get; set; }
            }

            public class Device
            {
            }

            public class Geo
            {
                public string country_code { get; set; }
                public string country { get; set; }
                public string continent { get; set; }
                public string city { get; set; }
                public double lat { get; set; }
                public double lng { get; set; }
                public string timezone { get; set; }
            }

            public class Karma
```

```
{
    public bool is_blocked { get; set; }
    public bool is_crawler { get; set; }
    public bool is_bot { get; set; }
    public bool is_friend { get; set; }
    public bool is_regular { get; set; }
}

public class Key
{
    public int id { get; set; }
    public string api_key { get; set; }
    public string type { get; set; }
    public DateTime expired { get; set; }
    public DateTime registered { get; set; }
    public int limits_by_hour { get; set; }
    public int limits_by_minute { get; set; }
    public int limits_by_month { get; set; }
    public int limits_total { get; set; }
}

public class Params
{
    public string lang { get; set; }
}

public class Request
{
    public string lang { get; set; }
    public string currency { get; set; }
    public int time { get; set; }
    public string id { get; set; }
    public string server { get; set; }
    public string host { get; set; }
    public int pid { get; set; }
    public Key key { get; set; }
    public Params @params { get; set; }
    public int version { get; set; }
    public string method { get; set; }
    public Client client { get; set; }
}

public class Response
{
    public string icao_code { get; set; }
    public string country_code { get; set; }
    public string iata_code { get; set; }
    public double lng { get; set; }
    public string name { get; set; }
    public double lat { get; set; }
}

public class Root
{
```

```
                public Request request { get; set; }
                public List<Response> response { get; set; }
                public string terms { get; set; }
            }
        }

        public Airports()
        {
            InitializeComponent();

            NavigationPage.SetHasNavigationBar(this, false);
            NavigationPage.SetBackButtonTitle(this, null);

            PopulateAirports();
        }

        private void OnFilterTextChanged(object sender,
Xamarin.Forms.TextChangedEventArgs e)
        {
            searchBar = (sender as SearchBar);

            if (ListViewAirports.DataSource != null)
            {
                ListViewAirports.DataSource.Filter = FilterService;
                ListViewAirports.DataSource.RefreshFilter();
            }
        }

        private bool FilterService(object obj)
        {
            if (searchBar == null || searchBar.Text == null)
            {
                return true;
            }

            // var list = obj as AirportsData.Response;
            var list = (AirportsData.Root)obj;

            if
(list.iata_code.ToLower().Contains(searchBar.Text.ToLower()) ||
list.name.ToLower().Contains(searchBar.Text.ToLower()))
            {
                return true;
            }
            else
            {
                return false;
            }
        }

        private void CloseTapGestureRecognizer_Tapped(object sender,
EventArgs e)
        {
            Navigation.PopModalAsync();
```

```
        }

        async void PopulateAirports()
        {
            var client = new HttpClient();

            // client.Timeout =
TimeSpan.FromSeconds(App.HttpClient_TimeOut);
            client.BaseAddress = new Uri("https://airlabs.co/api/v9/
airports");

            var content = new FormUrlEncodedContent(new[]
            {
                new KeyValuePair<string, string>("api_key",
App.WebServiceAppID)
            });

            var response = await client.PostAsync("https://
airlabs.co/api/v9/airports", content);

            if (response.IsSuccessStatusCode)
            {
                var result = await
response.Content.ReadAsStringAsync();

                // AirportsData.Root data =
JsonSerializer.Deserialize<AirportsData.Root>(result);
                AirportsData.Root data =
JsonSerializer.Deserialize<AirportsData.Root>(result);

                ListViewAirports.ItemsSource = data.response;

                LottieAirports.IsVisible = false;

                SearchBoxAirports.IsVisible = true;
                ListViewAirports.IsVisible = true;
            }
            else
            {
                await App.Current.MainPage.DisplayAlert("Error",
"Error", "Error");
                // DependencyService.Get<Toast>().Show("Unable to
Get Ads!");
            }
        }

        void ListViewAirports_ItemTapped(System.Object sender,
Syncfusion.ListView.XForms.ItemTappedEventArgs e)
        {
            Preferences.Set("SchedulesAirportCode", (e.ItemData as
AirportsData.Response).iata_code);
            Preferences.Set("SchedulesAirportName", (e.ItemData as
AirportsData.Response).name);
```

```
            MessagingCenter.Send<object>(this,
App.RefreshSchedules);

            Navigation.PopModalAsync();
        }
    }
}
```