Технически университет – София

Факултет по компютърни системи и технологии

Курсов проект

по дисциплината: Програмиране за мобилни устройства

Тема: Weather Application

Изготвена от: Цветомир Петков

Курс: III

Специалност: КСИ

Група: 43

Фак. №: 121220107

Съдържание

Увод …………………………………………….. 3

Анализ на съществуващи разработки ………... 3

Проектиране …………………………………. 3-5

Реализация ………………………………….. 5-11

Потребителски ръководство (Резултати) …11-13

Заключение …………………………………… 14

Литература ……………………………………. 14

**Увод**

Приложенията за времето са станали неотменна част от ежедневието ни, тъй като ни предоставят информация за прогнозата за времето, която може да бъде използвана за планиране на дейности, пътувания, спортни събития и други.

Проблемът, който приложенията за времето решават, е предоставянето на точна и актуална информация за метеорологичните условия в определен момент и за предстоящите дни. Това включва информация за температурата, вятъра, влажността, налягането, облачността, вероятността за валежи и други метеорологични параметри. Приложенията за времето използват множество източници на данни, като сателитни и радарни изображения, метеорологични станции и модели за прогнозиране на времето, за да съберат и анализират информацията и да я предоставят на потребителите си.

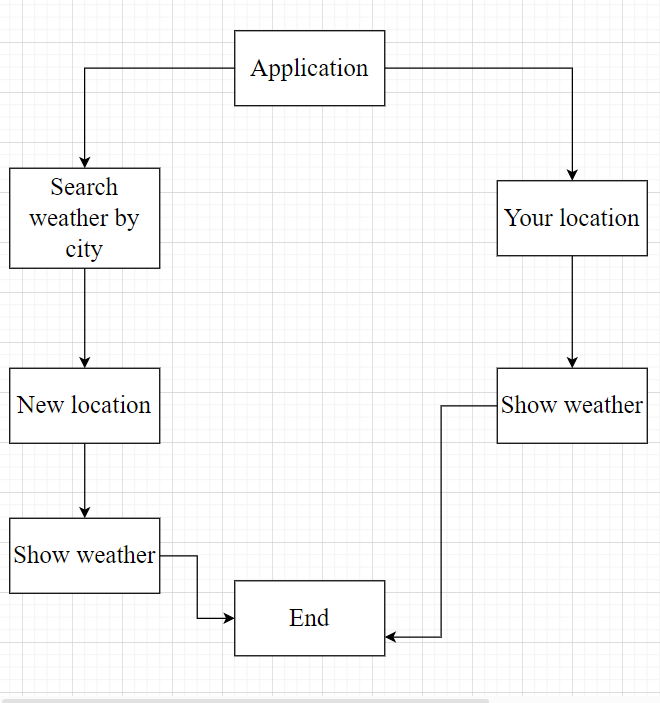
**Анализ на съществуващи разработки**

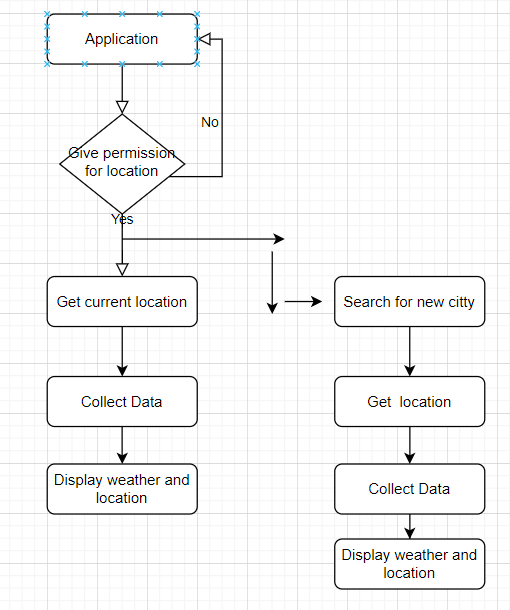
На пазара има множество приложения за времето, като някои от най-популярните са AccuWeather, Weather Underground, The Weather Channel, Dark Sky, и други. Голяма част от тях имат както Android, така и iOS версии, които са налични за безплатно сваляне от Google Play Store и App Store.

**Проектиране**

Приложението е разработено на Visual Studio с cross-platform framework-а .NET MAUI, с помощта на C# и XAML. С помощта на .NET MAUI можете да разработвате приложения, които могат да се изпълняват в Android, iOS, macOS и Windows, от една обща кодова база.

Приложението няма възрастови ограничения, което позволява да се използва от различни възрастови групи. Нужните данни за времето се предоставят от метеорологичните станции по света. Чрез приложението ще имате възможността да виждате метеорологичната обстановка в реално време, чрез достъп до Вашето местоположение или да потърсите чрез търсачката обстановката в друг град. Разполага и с функция за следене на влажността, скоростта на вятъра и почасова прогноза.





**Реализация**

**Генериране на C# класове от JSON – Root.cs**

using Newtonsoft.Json;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace WeatherApp1.Models

{

// Root myDeserializedClass = JsonConvert.DeserializeObject<Root>(myJsonResponse);

public class City

{

public int id { get; set; }

public string name { get; set; }

public Coord coord { get; set; }

public string country { get; set; }

public int population { get; set; }

public int timezone { get; set; }

public int sunrise { get; set; }

public int sunset { get; set; }

}

public class Clouds

{

public int all { get; set; }

}

public class Coord

{

public double lat { get; set; }

public double lon { get; set; }

}

public class List

{

public int dt { get; set; }

public string dateTime => UtcTimeLibrary.UtcTimeStamp.ConvertToUtc(dt);

public Main main { get; set; }

public List<Weather> weather { get; set; }

public Clouds clouds { get; set; }

public Wind wind { get; set; }

public int visibility { get; set; }

public double pop { get; set; }

public Rain rain { get; set; }

public Sys sys { get; set; }

public string dt\_txt { get; set; }

}

public class Main

{

public double temp { get; set; }

public double temperature => Math.Round(temp);

public double feels\_like { get; set; }

public double temp\_min { get; set; }

public double temp\_max { get; set; }

public int pressure { get; set; }

public int sea\_level { get; set; }

public int grnd\_level { get; set; }

public int humidity { get; set; }

public double temp\_kf { get; set; }

}

public class Rain

{

[JsonProperty("3h")]

public double \_3h { get; set; }

}

public class Root

{

public string cod { get; set; }

public int message { get; set; }

public int cnt { get; set; }

public List<List> list { get; set; }

public City city { get; set; }

}

public class Sys

{

public string pod { get; set; }

}

public class Weather

{

public int id { get; set; }

public string main { get; set; }

public string description { get; set; }

public string icon { get; set; }

// public string fullIconUrl => string.Format("https://openweathermap.org/img/wn/{0}@2x.png", icon);

public string customIcon => string.Format("icon\_{0}.png", icon);

}

public class Wind

{

public double speed { get; set; }

public int deg { get; set; }

public double gust { get; set; }

}

}

**ApiService.cs**

using Newtonsoft.Json;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using WeatherApp1.Models;

namespace WeatherApp1.Services

{

public static class ApiService

{

//извличане на данните

public static async Task<Root> GetWeather(double latidute, double longitude ) {

var httpClient = new HttpClient();

var response = await httpClient.GetStringAsync(string.Format("https://api.openweathermap.org/data/2.5/forecast?lat={0}&lon={1}&units=metric&appid=91d3467e7d6a21edb46e31643c4a08ce", latidute, longitude));

return JsonConvert.DeserializeObject<Root>(response); // конвертира json в C# класове

}

public static async Task<Root> GetWeatherByCity(string city)

{

var httpClient = new HttpClient();

var response = await httpClient.GetStringAsync(string.Format("https://api.openweathermap.org/data/2.5/forecast?q={0}&units=metric&appid=91d3467e7d6a21edb46e31643c4a08ce", city));

return JsonConvert.DeserializeObject<Root>(response);

}

}

}

**Дизайн на интерфейса**

<?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="WeatherApp1.WeatherPage"

Title="WeatherPage">

<Grid RowDefinitions="Auto, Auto, Auto, Auto, Auto"

RowSpacing="20">

<StackLayout Grid.Row="0"

Margin="20, 20, 20, 0"

Orientation="Horizontal">

<Frame Padding="10"

CornerRadius="35"

HasShadow="True"

BorderColor="LightGray">

<Label Text="Your location"

FontSize="Default"/>

<Frame.GestureRecognizers>

<TapGestureRecognizer x:Name ="TapLoacation" Tapped ="TapLoacation\_Tapped"/>

</Frame.GestureRecognizers>

</Frame>

<ImageButton Source="search.png"

HeightRequest="30"

WidthRequest="30"

HorizontalOptions="EndAndExpand"

Clicked="ImageButton\_Clicked"/>

</StackLayout>

<StackLayout Grid.Row="1"

Orientation="Vertical">

<Label x:Name="LblCity"

FontSize="40"

HorizontalOptions="Center"

TextColor="Gray"/>

<Label x:Name="LblWeatherDescription"

FontSize="Medium"

HorizontalOptions="Center"

TextColor="SlateGray"/>

<Image Grid.Row="2"

x:Name="ImageWeatherIcon"

VerticalOptions="Start"

HeightRequest="300"

WidthRequest="300"/>

<Grid Grid.Row="3"

RowDefinitions="Auto, Auto"

ColumnDefinitions="0.33\*, 0.33\*, 0.33\*"

Margin="20">

<Image Grid.Row="0"

Grid.Column="0"

HorizontalOptions="Center"

HeightRequest="25"

Source="humidity.png"/>

<Label x:Name="LblHumidity"

Grid.Row="1"

Grid.Column="0"

HorizontalOptions="Center"/>

<Label x:Name="LblTemperature"

Grid.Row="0"

Grid.Column="1"

Grid.RowSpan="2"

HorizontalOptions="Center"

FontSize="40"

FontAttributes="Bold"/>

<Image Grid.Row="0"

Grid.Column="2"

HorizontalOptions="Center"

HeightRequest="25"

Source="wind.png"/>

<Label x:Name="LblWind"

Grid.Row="1"

Grid.Column="2"

HorizontalOptions="Center"/>

</Grid>

<CollectionView Grid.Row="4" x:Name="CvWeather" HeightRequest="100 ">

<CollectionView.ItemsLayout>

<LinearItemsLayout Orientation="Horizontal"

ItemSpacing="15"/>

</CollectionView.ItemsLayout>

<CollectionView.ItemTemplate>

<DataTemplate>

<Grid Grid.Row="4"

RowDefinitions="Auto,Auto,Auto">

<Image Grid.Row="0"

HorizontalOptions="Center"

VerticalOptions="Start"

HeightRequest="50"

WidthRequest="50"

Source="{Binding weather[0].customIcon}"/>

<Label Grid.Row="1"

FontSize="Large"

HorizontalOptions="Center"

Text="{Binding main.temperature}"/>

<Label Grid.Row="2"

FontSize="Small"

HorizontalOptions="Center"

Text="{Binding dateTime}"/>

</Grid>

</DataTemplate>

</CollectionView.ItemTemplate>

</CollectionView>

</StackLayout>

</Grid>

</ContentPage>

**WeatherPage.xaml.cs**

using WeatherApp1.Models;

using WeatherApp1.Services;

namespace WeatherApp1;

public partial class WeatherPage : ContentPage

{

public List<Models.List> WeatherList;

private double latitude;

private double longitude;

public WeatherPage()

{

InitializeComponent();

WeatherList = new List<Models.List>();

}

protected async override void OnAppearing()

{

base.OnAppearing();

await GetLocation();

await GetWeatherDataByLocation(latitude, longitude);

}

public async Task GetLocation() {

var location = await Geolocation.GetLocationAsync();

latitude = location.Latitude;

longitude = location.Longitude;

}

private async void TapLoacation\_Tapped(object sender, EventArgs e) {

await GetLocation();

await GetWeatherDataByLocation(latitude, longitude);

}

public async Task GetWeatherDataByLocation(double latitude, double longitude) {

var result = await ApiService.GetWeather(latitude, longitude);

UpdateUI(result);

}

private async void ImageButton\_Clicked(object sender, EventArgs e) {

var response = await DisplayPromptAsync(title:"", message:"", placeholder:"Search weather by city", accept:"Search", cancel:"Cancel");

if (response != null) {

await GetWeatherDataByCity(response);

}

}

public async Task GetWeatherDataByCity(string city)

{

var result = await ApiService.GetWeatherByCity(city);

UpdateUI(result);

}

public void UpdateUI(Root result) {

foreach (var item in result.list)

{

WeatherList.Add(item);

}

CvWeather.ItemsSource = WeatherList;

LblCity.Text = result.city.name;

LblWeatherDescription.Text = result.list[0].weather[0].description;

LblTemperature.Text = result.list[0].main.temperature + "°C";

LblHumidity.Text = result.list[0].main.humidity + "%";

LblWind.Text = result.list[0].wind.speed + "km/h";

ImageWeatherIcon.Source = result.list[0].weather[0].customIcon;

}

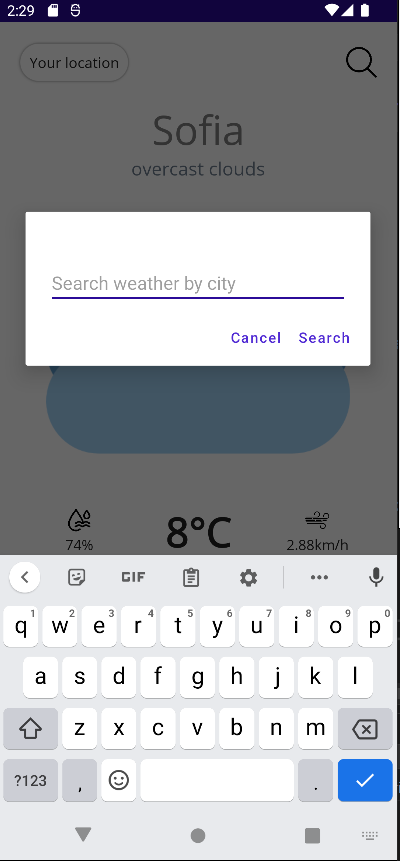
}

**Потребителски ръководство (Резултати)**

Фиг.1 След получаване на съгласие, потребителят може да провери какво е времето в момента.



Фиг.2 Въвеждане на нов град



Фиг.3 След въвеждането на новия град, потребителят може да види данните за него.



**Заключение**

Чрез приложението “Weather Application”, потребителят разполага с възможността да следи прогнозата за времето в реално време. Има опцията да търси и друг град, където и да е по света. Приложението е направено за използване от Android телефони.

**Литература**

* <https://youtube.com/playlist?list=PL3fvNBbJYqs6MTRpTthdTbAL5MKXovm91>
* <https://openweathermap.org/forecast5>
* <https://www.flaticon.com/packs/weather-161?word=clouds%20weather>