***Технически университет – София***

*Факултет по компютърни системи и технологии*

*Специалност - КСИ*

*КУРСОВ ПРОЕКТ*

*по дисциплината : Програмиране за мобилни устройства*

*тема : Приложение за прогноза за времето*

*изготвена от : преподавател :*

*Динко Христов гл. ас. Невен Николов*

*(III курс, 42.гр. фак. №: 121220082)*

*Дата: 24.04.2023*

***Съдържание***

1. **..........................................................................Увод**
2. **...............Анализ на съществуващи разработки**
3. **...........................................................Проектиране**
4. **........................................Софтуерна реализация**
5. **............Потребителско ръководство(тестване)**
6. **.............................................................Заключение**
7. **………………..................Използвана литература**

**Увод**

Мобилните телефони и приложенията за времето направиха много лесен и удобен достъп до прогноза за времето, но настръхвам, когато някой каже. „Току-що проверих телефона си и... ахх ахх ахх.“ Прогнозирането на времето е трудно и на приложението на телефона ви липсва едно много ценно нещо: човешка интерпретация.

Чувам някои от шегите, които казвате сега: „Прогнозата винаги е грешна. Можете да грешите 50% и пак да запазите работата си. Няма нужда да гледам времето по новините, имам мобилния си телефон. Но чудили ли сте се някога откъде това приложение за времето на телефона ви получава информация?

Метеоролози или Националната метеорологична служба прекарват часове в изливане на данни от компютърни модели, сателити, радари и наблюдения на повърхността. Прогнозирането на времето е изкуство, колкото и наука. Наличието на образование и опит позволява по-добра интерпретация на данните от модела.

**Анализ на съществуващи разработки**

Приложението е направено изрично за устройства, работещи с операционната система Android и като такова, то има много конкуренти, които могат да бъдат намерени в Google Play магазина на всяко едно Android устройство. По-известни са Carrot Weather и Weather Channel, които работят, както и на Android така и на iOS.

По-рано ексклузивен продукт за iOS, който не можехте да получите на Android, Carrot Weather отдавна е преодолял разделението на платформата; сега злонамереният повелител на изкуствения интелект Керът може да донесе на потребителите на Android прогнозата за времето, заедно с голяма помощ от подигравки и сарказъм. Потребителите на iOS все още се радват на функции като възможността да добавят джаджа Carrot Weather към заключените си екрани, както и поддръжка за дейности на живо в iOS 16.1. Но периодичните актуализации добавят други функции и бонуси за първокласни абонати, като данни за приливи и отливи или прогнози за до пет местоположения.

Безплатната, поддържана от реклами версия на приложението Weather Channel е достъпна за Android и iOS. Приложението TWC ви предоставя изобилие от метеорологични данни като температура, вятър и видимост на почасова или дневна база, с налична разширена 10-дневна прогноза. Интерактивните карти могат да покажат най-новите доплерови радарни данни, за да покажат валежи и метеорологични събития и предупреждения за лошо време.

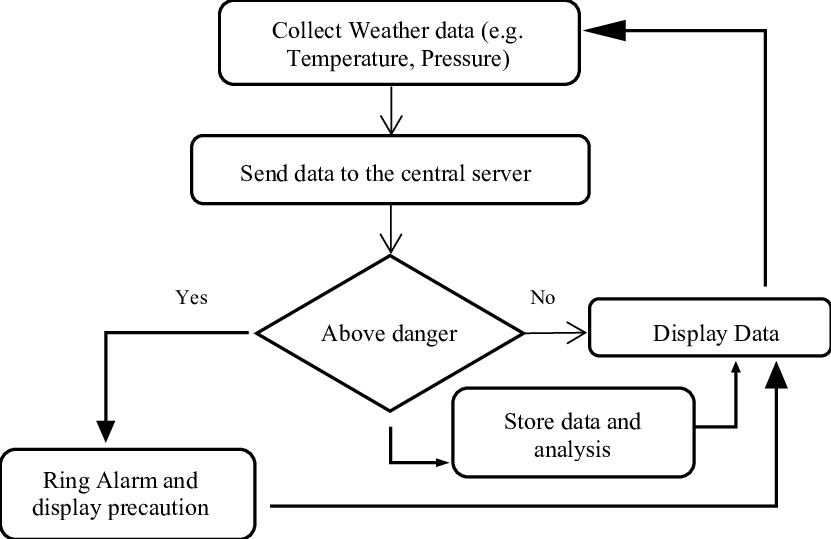
**Проектиране**

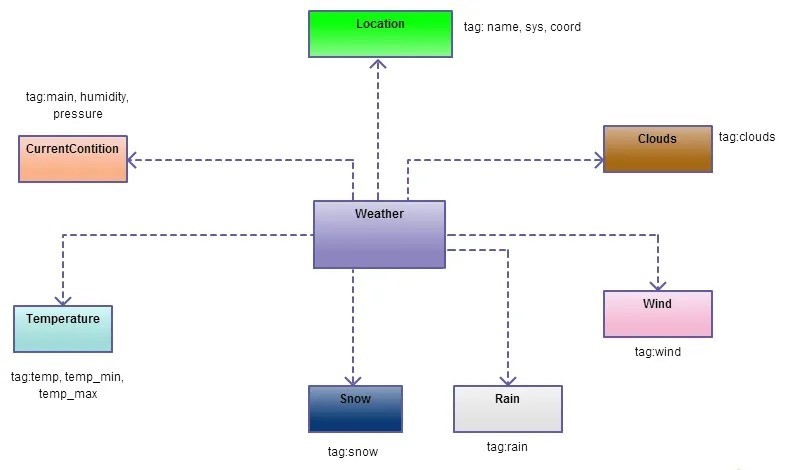
Приложението е разработено с помощта на “Android Studio”, приложение с отворен код за разработка на Android приложения. Като такова, в основата си то използва езикът за програмиране Java за осъществяване на автоматизирани дейности в себе си и XML файлове за описание на външния вид на потребителския си интерфейс. Също така е важно да се отбележи, че приложението е направено да поддържа устройства, работещи с Android 4.4 “KitKat” и нагоре, което означава, че към днешна дата приблизително 99 процента от устройствата, използващи се от хората по света и работещи на Android операционната система, ще могат да го използват.

Продуктът ще се използва от всеки човек, който притежава телфон, компютър или каквото и да е устройтво с Android операционна система.

Данните, които ще използва потребителят ще са предоставени от Метеорологичните станции по света и ще ни предоставят нужните данни за най-точна прогноза на времето.

Можем да достъпваме данните чрез кликване или натискане на бутон на екрана и въвеждане на мястото, което искаме да проверим и каква ще е прогнозата на него.





**Софтуерна реализация**

Както е споменато по-рано в този документ, приложението е реализирано с помощта на езика за програмиране Java и XML файлове, описващи външния вид на потребителския му интерфейс. Следващият текст представлява откъси от кода, използван за връзка между действията, нужни за правилното функциониране на приложението:

Фиг №1. Клас “Weather”:

package com.example.weatherapp.models;

public class Weather {

private double temperature;

private double humidity;

private double precipitation;

public double getTemperature() {

return temperature;

}

public void setTemperature(double temperature) {

this.temperature = temperature;

}

public double getHumidity() {

return humidity;

}

public void setHumidity(double humidity) {

this.humidity = humidity;

}

public double getPrecipitation() {

return precipitation;

}

public void setPrecipitation(double precipitation) {

this.precipitation = precipitation;

}

}

Фиг №2 Интерфейс „WeatherService“:

package com.example.weatherapp.services;

import com.example.weatherapp.models.Weather;

public interface WeatherService {

Weather getWeather(double latitude, double longitude);

}

Фиг №3 Клас „WeatherServiceImpl“, които имплементира интерфейса “WeatherService”:

package com.example.weatherapp.services;

import com.example.weatherapp.models.Weather;

public class WeatherServiceImpl implements WeatherService {

@Override

public Weather getWeather(double latitude, double longitude) {

// TODO: Use an API to retrieve weather data for the specified latitude and longitude

Weather weather = new Weather();

weather.setTemperature(75.0);

weather.setHumidity(50.0);

weather.setPrecipitation(0.0);

return weather;

}

}

Фиг №4 Клас „Main Activity“:

package com.example.weatherapp;

import androidx.appcompat.app.AppCompatActivity;

import android.location.Location;

import android.location.LocationListener;

import android.location.LocationManager;

import android.os.Bundle;

import android.widget.TextView;

import com.example.weatherapp.models.Weather;

import com.example.weatherapp.services.WeatherService;

import com.example.weatherapp.services.WeatherServiceImpl;

public class MainActivity extends AppCompatActivity implements LocationListener {

private LocationManager locationManager;

private TextView temperatureTextView;

private TextView humidityTextView;

private TextView precipitationTextView;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

temperatureTextView = findViewById(R.id.temperatureTextView);

humidityTextView = findViewById(R.id.humidityTextView);

precipitationTextView = findViewById(R.id.precipitationTextView);

locationManager = (LocationManager) getSystemService(LOCATION\_SERVICE);

locationManager.requestLocationUpdates(LocationManager.GPS\_PROVIDER, 0, 0, this);

}

@Override

public void onLocationChanged(Location location) {

double latitude = location.getLatitude();

double longitude = location.getLongitude();

WeatherService weatherService = new WeatherServiceImpl();

Weather weather = weatherService.getWeather(latitude, longitude);

temperatureTextView.setText(String.format("%.1f °F", weather.getTemperature()));

humidityTextView.setText(String.format("%.0f%%", weather.getHumidity()));

precipitationTextView.setText(String.format("%.2f in", weather.getPrecipitation()));

}

@Override

public void onProviderEnabled(String provider) {}

@Override

public void onProviderDisabled(String provider) {}

@Override

public void onStatusChanged(String provider, int status, Bundle extras) {}

}

Фиг №5 “activity\_main.xml”:

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".MainActivity">

<TextView

android:id="@+id/temperatureTextView"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textSize="24sp"

android:text="Temperature: " />

<TextView

android:id="@+id/humidityTextView

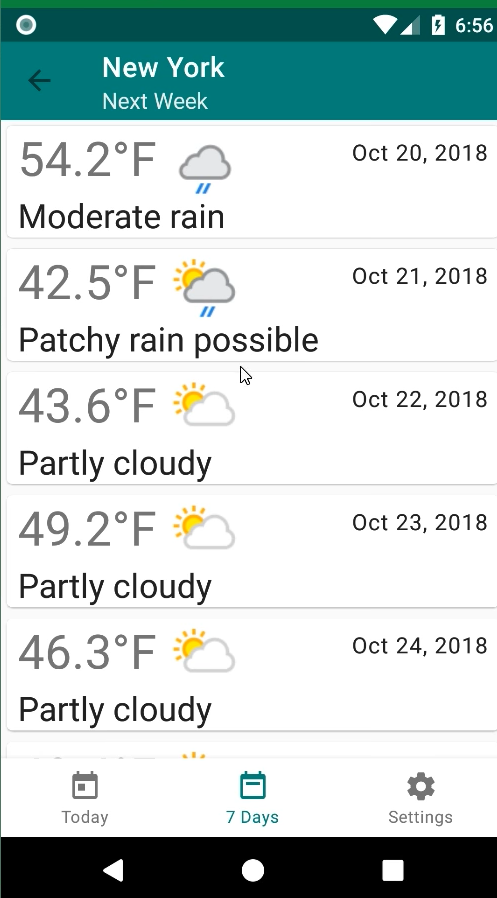
**Потребителско ръководство(Тестване)**

На следващите снимки е показано как да се работи с предоставеното приложение към курсовата работа:

Фиг 1. Показване на прогнозата за днешния ден.(във Фаренхайт)



Фиг. 2. Показване на цялата седмична прогноза.(във Фаренхайт)

****

Всички следващи възможни действия са еднакви и представляват просто повтаряне на описаните по-рано такива. С това приключва описанието на функционалностите на приложението в потребителското ръководство.

**Заключение**

Приложението “Weather App”, разработено за този курсов проект е напълно достатъчно за нуждите на потребителя като следене на прогнозата за настоящия ден или за прогнозата за цялата седмица. Имаме нужда от интернет за да можем да следим прогнозата в реално време и д авиждаме дали имаме промяна в температурата или вероянтост за валежи. Също така неговият код е с отворен достъп и това позволява неговите функционалности да бъдат надграждани от доброволци и така то да бъде подобрявано безплатно.

**Използвана литература**

1. [Best weather apps | Tom's Guide (tomsguide.com)](https://www.tomsguide.com/round-up/best-weather-apps)
2. [How to Build a Weather App in Android? - GeeksforGeeks](https://www.geeksforgeeks.org/how-to-build-a-weather-app-in-android/)
3. [Weather Wise: Problems with Weather Apps (ktvh.com)](https://www.ktvh.com/news/weather-wise/weather-wise-problems-with-weather-apps)
4. [A Data Model for a Weather App | Vertabelo Database Modeler](https://vertabelo.com/blog/a-data-model-for-a-weather-app/)
5. [Make a Weather App for Android | Android Studio | Kotlin - YouTube](https://www.youtube.com/watch?v=gj0g1a75Lmo)