

Mobile Weather App

Sprawozdanie i dokumentacja zostało napisane przy użyciu języka znaczników Markdown, a następnie zostało zapisane do formatu PDF Cały projekt jak i sprawozdanie z dokumentacją można znaleźć pod [linkiem na moim profilu GH](#)

Spis treści

1. [Sprawozdanie](#)
 - [Cele i wymagania](#)
 - [Wykorzystane narzędzia](#)
 - [Zobrazowanie działania aplikacji](#)
 - [Podsumowanie](#)
2. [Dokumentacja techniczna](#)
 - [Instalacja](#)
 - [Uruchomienie aplikacji](#)
 - [Utworzenie pakietu instalacyjnego dla środowiska Android](#)

Sprawozdanie

Cele i wymagania

1. Aplikacja powinna być napisana do uruchomienia w środowisku mobilnym Android.
2. Użytkownik powinien mieć możliwość sprawdzenia pogody aktualnej oraz na kilka dni do przodu.
3. Użytkownik musi mieć możliwość wyszukiwania pogody dla wybranego przez siebie miasta.
4. Aplikacja powinna pokazać podstawowe dane o pogodzie na obecny dzień takie jak:
 - Temperatura
 - Warunki pogodowe
 - Wilgotność powietrza
 - Prędkość i kierunek wiatru
 - Ciśnienie atmosferyczne
5. Poza danymi z punktu 4 aplikacja powinna pokazać warunki pogodowe, temperaturę i datę na 3 kolejne dni.
6. Użytkownik powinien mieć możliwość zapisania oraz usuwania wyszukiwanych miast w pamięci aplikacji, aby łatwo i szybko przełączać widoki dla innych lokalizacji - maksymalnie 5 zapisanych miast.

Wykorzystane narzędzia

1. [Język programowania Python](#) Python to język programowania wysokiego poziomu ogólnego przeznaczenia, o rozbudowanym pakiecie bibliotek standardowych, którego ideą przewodnią jest czytelność i klarowność kodu źródłowego. Jego składnia cechuje się przejrzystością i zwięzłością.
2. [Framework Kivy](#) Kivy to bezpłatna i otwarta platforma Python do tworzenia aplikacji mobilnych i innego oprogramowania aplikacji wielodotykowych z naturalnym interfejsem użytkownika. Jest

rozprowadzany na warunkach licencji MIT i może działać na systemach Android, iOS, Linux, macOS i Windows.

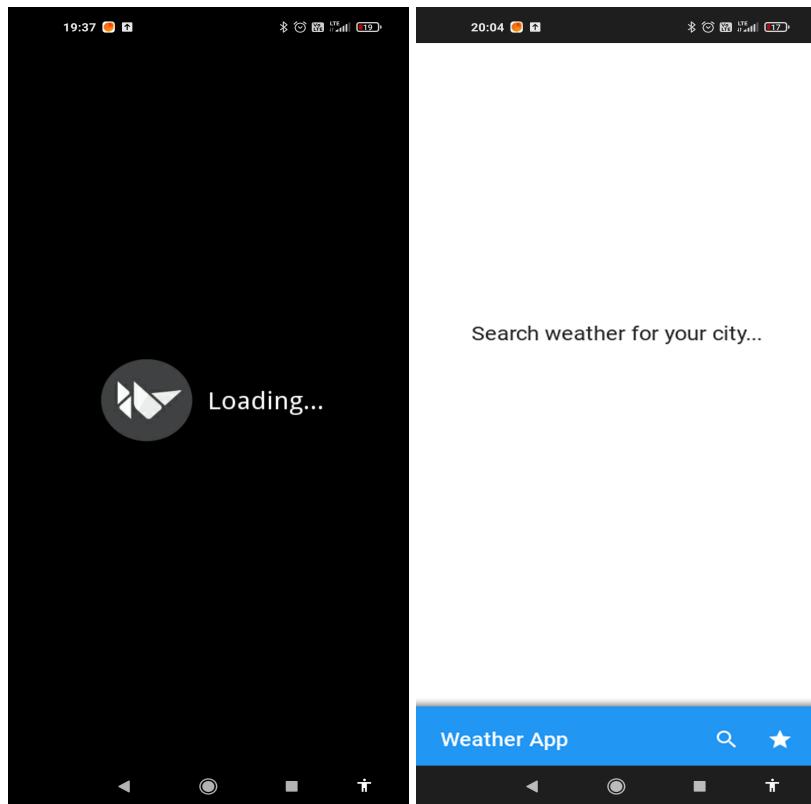
3. [Pakiet komponentów KivyMD](#) Jest zbiorem widżetów (Material Designs) do użytku z Kivy i innymi frameworkami aplikacji graficznych obsługujących dotyk.
4. [OpenWeatherMap API](#) OpenWeather to portal udostępniający dla każdego punktu na kuli ziemskiej historyczne, bieżące i prognozowane dane pogodowe za pośrednictwem interfejsów API. Posiada płatne i darmowe pakiety.
5. [Buildozer](#) Buildozer to narzędzie, które ma na celu łatwe tworzenia pakietów aplikacji mobilnych. Automatyzuje cały proces komplikacji, pobiera wymagania wstępne, takie jak python-for-android, Android SDK, NDK itp. Buildozer zarządza plikiem o nazwie buildozer.spec w katalogu aplikacji, opisując wymagania i ustawienia aplikacji, takie jak tytuł, ikona, dołączone moduły itp. Użyje pliku specyfikacji do stworzenia pakietu dla Androida, iOS i innych.

Zobrazowanie działania aplikacji

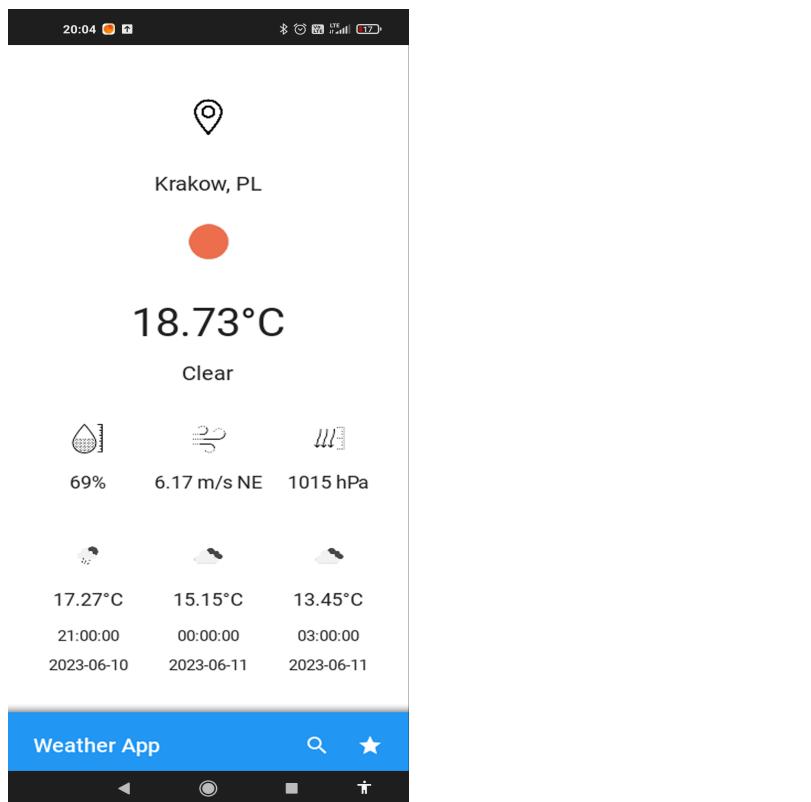
1. Widok ekranu głównego po instalacji aplikacji.



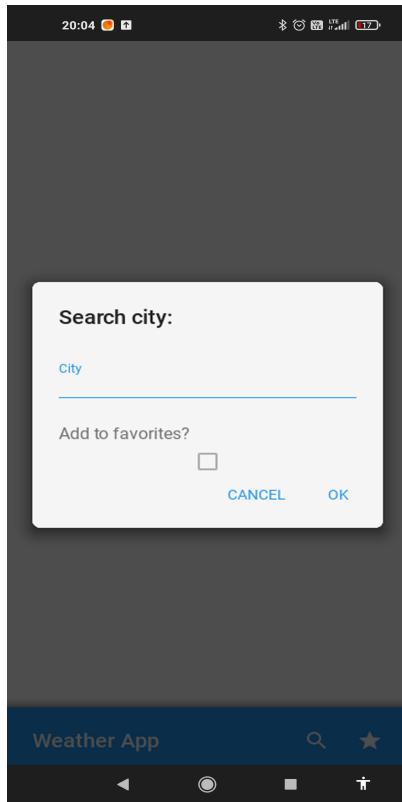
2. Widok z ładowania aplikacji i po pierwszym uruchomieniu.



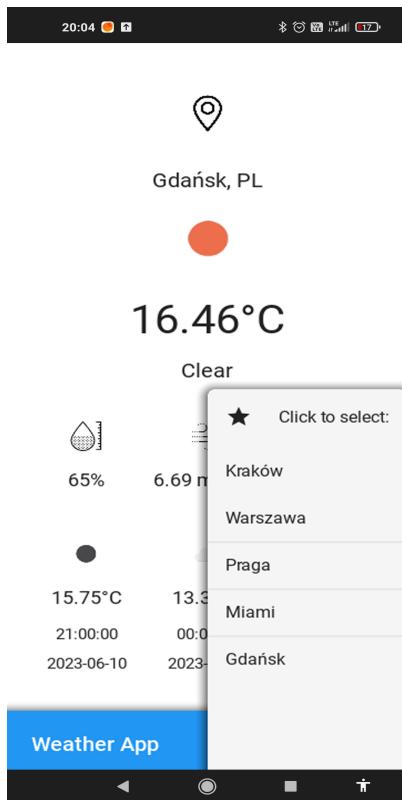
3. Widok kompletnych danych pogodowych dla wybranego miasta.



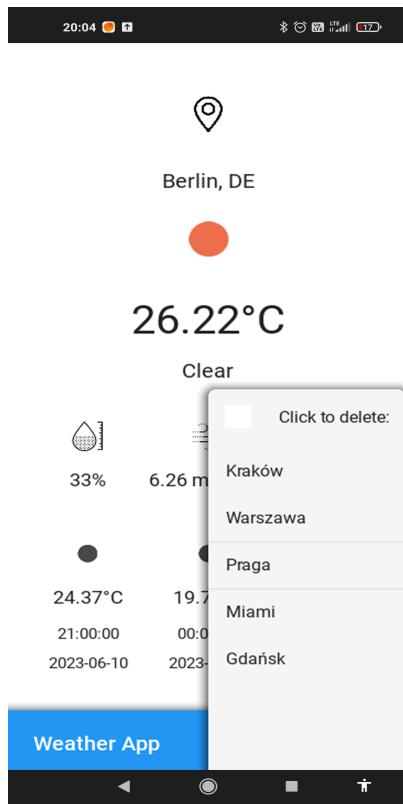
4. Widok wyszukiwania miasta.



5. Widok listy zapisanych miast.



6. Widok listy zapisanych miast z opcją usuwania.



Podsumowanie

Projekt został wykonany zgodnie z założeniami zawartymi na początku dokumentu z małym wyjątkiem - zamiast pokazania pogody na kilka dni do przodu, pokazano informacje pogodowe na za 3,6 oraz 9 godzin. Ma to związek z limitami darmowego API od OpenWeatherMap, które zostało wykorzystane w tej aplikacji.

Projekt stanowi dobrą bazę do rozbudowywania aplikacji o kolejne funkcjonalności takie jak na przykład pokazanie danych godzinowych, historycznych, wyświetlenia danych na wykresach czy tworzenie widgetów na ekran główny telefonu.

Dzięki wykorzystaniu KivyMD, do aplikacji można dodawać w łatwy sposób gotowe komponenty, które można dostosować do swoich potrzeb bez konieczności tworzenia modułów od podstaw.

Aplikacja została uruchomiona i testowana przy użyciu telefonu Xiaomi Mi 10T Pro z systemem Android w wersji 10 QKQ1.200419.002 oraz przy użyciu emulatora systemu Android o nazwie BlueStacks X.

Dokumentacja techniczna

Instalacja

1. Pobierz repozytorium z kodem i otwórz folder z projektem przy użyciu dowolnego IDE (na przykład Visual Studio Code).
2. Utwórz wirtualne środowisko i aktywuj je.

```
# Utworzenie środowiska wirtualnego
python -m venv /path/to/new/virtual/environment
```

```
# Aktywacja środowiska wirtualnego  
.env\Scripts\activate
```

3. Używając menedżera pakietów [pip](#), zainstaluj wymagane biblioteki.

```
pip install -r requirements.txt
```

Uruchomienie aplikacji

1. Utwórz konto oraz idywidualny klucz API na portalu [Open Weather Map](#)
2. Wprowadź Twój klucz API w pliku [main.py](#)

```
# Replace API_KEY with your OpenWeatherMap API key  
API_KEY = 'YOUR_OPEN_WEATHER_MAP_API_KEY'
```

3. Uruchom aplikację używając poniższego polecenia (pamiętaj o aktywacji środowiska wirtualnego opisanego w sekcji wyżej):

```
python3 main.py
```

Utworzenie pakietu instalacyjnego dla środowiska Android

1. Aby utworzyć aplikację gotową do uruchomienia w środowisku Android, należy przygotować pakiet instalacyjny z rozszerzeniem [.apk](#), który będzie można uruchomić na urządzeniu mobilnym.
2. Aby stworzyć pakiet instalacyjny, możemy wykorzystać system operacyjny Windows oraz narzędzie [Google Colab](#) lub system operacyjny Linux.
3. W wybranym systemie należy uruchomić poniższe polecenia:

```
!pip install buildozer
```

```
!pip install cython==0.29.19
```

```
!sudo apt-get install -y \ python3-pip \ build-essential \ git \ python3 \  
python3-dev \ ffmpeg \ libsdl2-dev \ libsdl2-image-dev \ libsdl2-mixer-dev \  
libsdl2-ttf-dev \ libportmidi-dev \ libswscale-dev \ libavformat-dev \ libavcodec-  
dev \ zlib1g-dev
```

```
!sudo apt-get install -y \ libgstreamer1.0 \ gstreamer1.0-plugins-base \ gstreamer1.0-plugins-good
```

```
!sudo apt-get install build-essential libsqlite3-dev sqlite3 bzip2 libbz2-dev zlib1g-dev libssl-dev openssl libgdbm-dev libgdbm-compat-dev liblzma-dev libreadline-dev libncursesw5-dev libffi-dev uuid-dev libffi6
```

```
!sudo apt-get install libffi-dev
```

```
!buildozer init
```

4. Po wykonaniu ostatniego polecenia zostanie utworzony plik [buildozer.spec](#). Przed przystąpieniem do kolejnego kroku należy edytować utworzony plik, aby nadać tytuł aplikacji, określić wymagane pakiety z konkretnymi wersjami oraz wskazać potrzebne uprawnienia.

```
# Title of your application
title = Weather App
...
# Application requirements
# Comma separated e.g. requirements = sqlite3,kivy
requirements = python3,kivy
...
# Permissions
android.permissions = android.permission.INTERNET,
(name=android.permission.WRITE_EXTERNAL_STORAGE;maxSdkVersion=18)
```

5. Jako ostatni krok należy użyć komendy wskazanej poniżej. (UWAGA! Proces jest czasochłonny - może trwać nawet ponad 30 min w przypadku komputera ze słabymi podzespołami)

```
!buildozer -v android debug
```