WEATHER APP

PRESENTATION

Bhavya Chawla (2022594)

Siddhant Singh (2022497)

Ayaan hasan (2022121)

Jetpack Weather App

A weather app that shows current conditions like temperature, humidity, and wind speed. It supports voice search, location detection, and saves search history. Built with Jetpack Compose and Retrofit, it includes dark/light mode and auto-refreshes data. The app also reads forecasts aloud using text-to-speech.

Features the app supports

Weather App
Features:

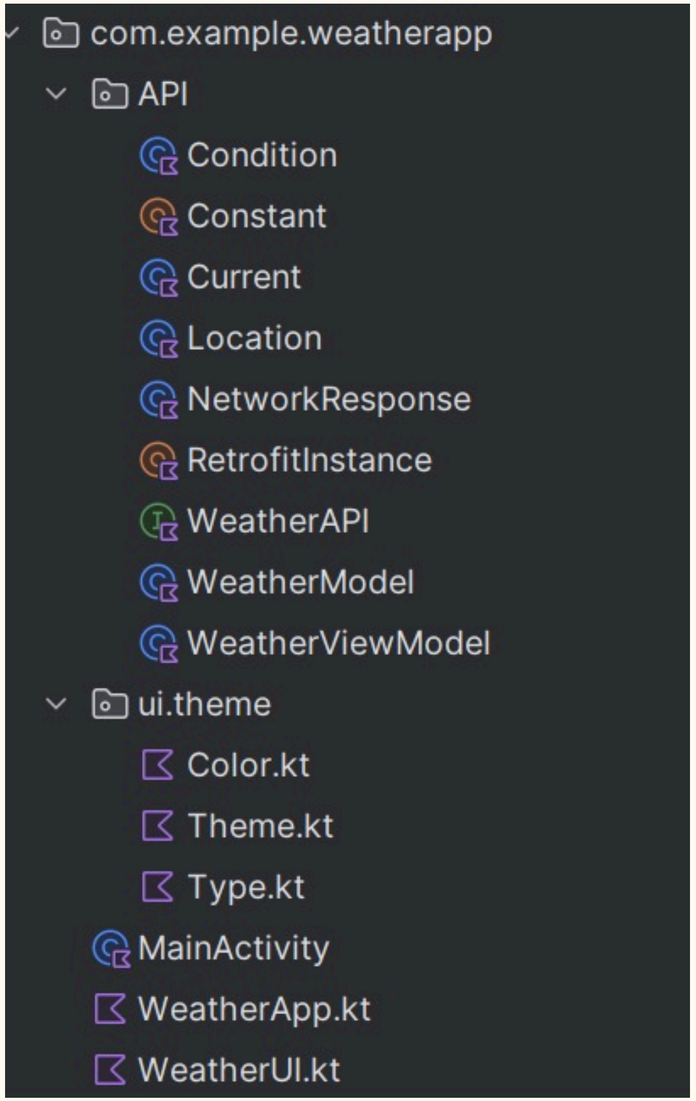
- 🌦️ Current Weather Data – Displays temperature, humidity, wind speed, UV index, and more.
- 📍 Location-Based Search – Uses GPS or manual city input for weather updates.
- 🔍 Search History – Stores recent searches for quick access.
- 🎙️ Voice Search – Speak a city name to fetch weather data.
- 🌙 Dark/Light Mode – Toggle between themes for better readability.
- 🗣️ Text-to-Speech (TTS) – Reads weather details aloud.
- 🔄 Auto-Refresh – Updates weather every 1 minutes.

Tools and Technologies being used

Technical Stack:

- 📱 Jetpack Compose – Modern declarative UI framework.
- 🌐 Retrofit + Gson – For fetching and parsing weather API data.
- 📍 Google Play Services (Location) – For GPS-based city detection.
- 🖼️ Coil – Efficient image loading for weather icons.
- 🗺️ Compose Navigation – Handles screen transitions.
- 💾 SharedPreferences – Stores search history and app settings.

Weather App UI



Project Structure

🌦️ Weather App Overview

🔹 MainActivity.kt

- Entry Point & Core Logic
- Dark Mode toggle with persistence (mutableStateOf, SharedPreferences)
- Text-to-Speech: US English, lifecycle-aware (init, pause, destroy)
- Architecture: Jetpack Compose + ViewModel

🔹 WeatherUI.kt

- Weather Display: Current + detailed (temp, humidity, wind, UV, etc.)
- Jetpack Compose UI: Responsive, card-based, dark/light support
- Search System: Text input, voice, location-based (with permissions)
- Auto-Refresh + TTS Reading
- Robust error handling + adaptive design

🔹 WeatherApp.kt

- NavHost with single "search" screen
- Location Services: Fetch coordinates, geocode to city
- Search History: Persistent, deduplicated, MRU ordering
- Utility: fetchData(), permission-aware, toast feedback

Challenges we faced

- API Integration & Networking.
- Android Permissions & Location Services
- Background Tasks & Auto-Refresh
- Text-to-Speech (TTS) & Voice Search
- Data Persistence (SharedPreferences)

Future Possibilities

- Background Tasks & Auto-🔮 Future Work: ML Integration in Weather App (Android + Kotlin)
- 📌 Why ML Not Yet Integrated
- Most reliable ML-powered weather & AQI APIs are paid
- Free API limitations and budget constraints delayed integration
- Initial focus was on building robust core functionality
- 💡 Planned ML Features (On-Device using Kotlin + TFLite)
- Short-Term Weather Forecasting
- Use LSTM/GRU models to predict next-hour/day weather (temp, rain, humidity)
- Based on historical user/location-specific data
- AQI (Air Quality Index) Prediction
- Train regression model using historical AQI + weather parameters
- Predict upcoming air quality conditions (PM2.5, PM10, etc.)
- Helpful for sensitive groups (asthma, outdoor activity planning)
- Personalized UX with On-Device Learning
- Learn user behavior (e.g., which metric is viewed most)
- Adjust UI recommendations or auto-refresh settings accordingly
- 🛠️ Kotlin + Android Implementation Plan
- 📈 Data: Collect historical weather + AQI data (e.g., OpenAQ, WAQI)
- 🧠 Model: Train in Python → Convert to .tflite
- 📲 Integrate in Kotlin

```
val tflite = Interpreter(loadModelFile("aqi_predictor.tflite"))
val input = floatArrayOf(/* weather params */)
val output = FloatArray(1)
tflite.run(input, output)
val predictedAQI = output[0]
```

References

weatherapi:
WeatherAPI.com provides access to free weather and geo data via a JSON/XML restful API. It allows developers to create desktop, web and mobile applications using this data very easy.

Github Link

Contribution

Bhavy Chawla (2022594):
Basic UI of the App and API call implementation, location based search, presentation

Siddhant Singh (2022497):
advanced features - voice search, auto refresh, text to speech, presentation

Ayaan (2022121) - light and dark mode implementation, history of data , ideation , presentation

Thank You

Bhavya Chawla (2022594)

Siddhant Singh (2022497)

Ayaan (2022121)