Design and Development of a Flutter application

Cities Weather App

Henrique Manuel Cordeiro Rodrigues Gardé
up202108725
João Tomás Matos Fernandes Garcia Padrão
up202108766
José Francisco Reis Pedreiras Neves Veiga
up202108753

Master's in Informatics and Computer Engineering

Mobile Computing

2024/2025

Contents

1 Introduction

This paper presents a mobile weather consultation system called the Cities Weather App, developed using the Flutter framework and the Dart programming language. The app is aimed at users who wish to keep track of meteorological data for Portuguese district capitals through an interactive and user-friendly interface.

The system consists of a single Flutter application that enables users to maintain a personalized list of cities. At any time, users can add or remove cities from this list according to their interests. For each selected city, the app allows for a consultation of current weather conditions, including temperature, humidity, wind, precipitation, and atmospheric pressure. Each condition is visually complemented with representative icons for better clarity and experience.

In addition to current conditions, the app provides a separate view for the next day's forecast, including optional icons that describe expected weather variations throughout the day. Users can also explore historical data in graphical form, analyzing trends over the past week for key weather indicators in any chosen city.

The system is built using modern technologies, with a focus on clean architecture, responsive user interface, and intuitive user experience. Integration with external weather APIs ensures that the data is accurate and up-to-date. The use of Flutter allows the application to remain performant across multiple platforms while maintaining a native look and feel.

In the following chapters, we will examine the architecture, design decisions, and implementation of the application, highlighting the tools, libraries, and techniques used to meet and extend the system requirements defined above.

2 System Overview

As mentioned above, the Cities Weather System is composed of a single main component:

- A **Weather App**, developed using Flutter and Dart, which allows users to manage a personalized list of Portuguese district capitals, consult current weather data, view forecasts, and analyze historical trends.

The functionality of the system is centered on the user's ability to interact with real-time and historical meteorological data in an intuitive and responsive mobile interface. The interactions available to the user are described below.

1. **City List Management**: Users can add or remove cities from a personalized list. The app restricts the selection to the district capitals of Portugal, ensuring data relevance and uniformity.

2. **Current Weather Conditions**: Upon selecting a city from the list, the user can retrieve real-time weather conditions, including temperature,

humidity, atmospheric pressure, wind speed, and precipitation. The app displays this information alongside an appropriate weather icon for better visual comprehension.

3. **Next-Day Forecast**: The user can access a separate forecast view for the next day. This forecast includes segmented data for different times of the day with small icons that represent predicted weather changes.

4. **Weekly Historical Trends**: The app offers an interactive graphical view of the weather of the last week for any city on the user list. Users can explore the evolution of various metrics, such as temperature or humidity, through clean and informative data visualizations.

5. **AI-Powered Weather Tips**: After fetching current weather data for a selected city, the app generates a context-sensitive tip. These tips are designed to enhance the user experience by offering useful suggestions based on real-time conditions.

All data interactions are made possible through integration with external weather APIs, ensuring accurate and updated information. The application architecture follows clean coding principles, with separation of concerns, structured data models, and reactive UI components.

Further sections of this report will detail the system architecture, the technologies, and libraries used in the implementation of the Cities Weather App.

3 Architecture

The image below illustrates the **backend architecture** of the application. This UML class diagram focuses on the non-UI components that manage core application logic: data models, external service integration, and local storage.

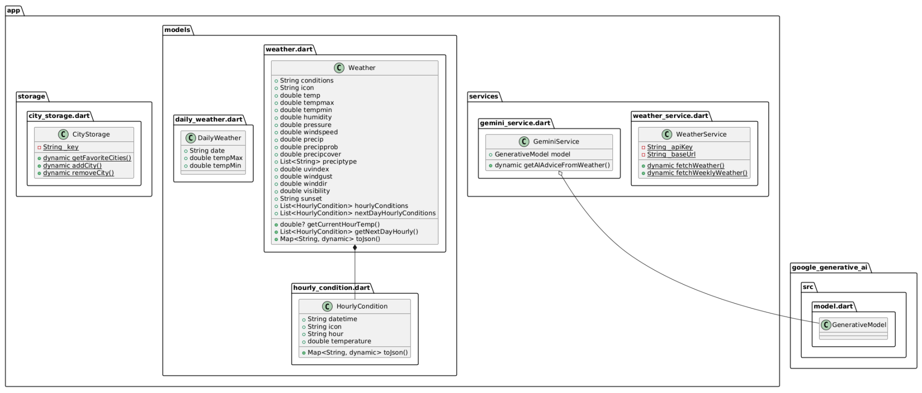


Figure 1: Backend architecture UML diagram of the Cities Weather App

Models (app/models)

These classes define the data structures used throughout the application:

- **Weather**: Represents a full daily weather snapshot. It includes fields for temperature, humidity, wind, and precipitation, along with a list of `HourlyCondition` entries and helper methods for accessing the current hour temperature or converting to JSON.
- **HourlyCondition**: Represents weather data for a specific hour, including temperature, hour string, and icon.
- **DailyWeather**: A simple model used to represent maximum and minimum temperatures for a given date. It is mainly used to feed the weekly temperature chart.

The `Weather` class has a composition relationship with `HourlyCondition`, as it contains a list of hourly forecasts.

Services (app/services)

These classes encapsulate external logic:

- **WeatherService**: Handles HTTP requests to the Visual Crossing Weather API. It includes static methods for fetching current and historical weather data.
- **GeminiService**: Integrates with Google's Generative AI SDK to produce weather-based user advice. This class contains a reference to `GenerativeModel`, illustrating the system's use of external AI models.

Storage (app/storage)

- **CityStorage**: A utility class that uses `SharedPreferences` to persist the user's favorite cities. It exposes static methods to retrieve, add, and remove cities.

Summary

This backend-focused design follows a clean and modular architecture, where each layer is responsible for a distinct set of concerns:

- Models handle data representation and transformation.
- Services handle external integrations and business logic.
- Storage manages persistence and user personalization.

Such organization facilitates scalability, testability, and maintainability of the application.

4 Features and Technologies

The Cities Weather App was developed using the Flutter framework and Dart programming language. It integrates several technologies to deliver a responsive, modern user experience across supported platforms. The main features and tools used include:

- **Flutter and Dart**: Used to develop the app's user interface and business logic. Although Flutter supports multiple platforms (Android, iOS, Web, Desktop), this project targets Android specifically.
- **Visual Crossing Weather API**: Provides reliable and up-to-date meteorological data for Portuguese district capitals, including both current and historical weather.
- **Google Generative AI (Gemini)**: Delivers context-aware AI-based suggestions based on live weather data, enhancing user interaction.
- **SharedPreferences**: Stores the user's list of favorite cities locally on the device.
- **Data Visualization**: Includes an interactive chart that displays temperature evolution across the last 7 days.
- **Clean Architecture**: Implements a modular separation between models, services, storage, and presentation layers.

5 Screens & How to use

The Cities Weather App offers a clean and intuitive navigation experience through a small set of well-defined screens.

Upon launch, users are greeted by a **welcome screen** featuring the app logo, slogan, and a single *Get Started* button (Figure 2). The main screen allows users to search for Portuguese district capitals and manage a personalized list of favorite cities. Basic weather information is shown for each favorite, including temperature, condition, and min/max values.

Tapping on a city opens the **weather details screen** (Figure 4). This screen displays real-time data such as temperature, wind speed, humidity, precipitation, sunset time, and visibility. It also includes a graphical chart showing the temperature evolution over the past week (Figure 6) and an **AI-generated suggestion** based on the current weather.

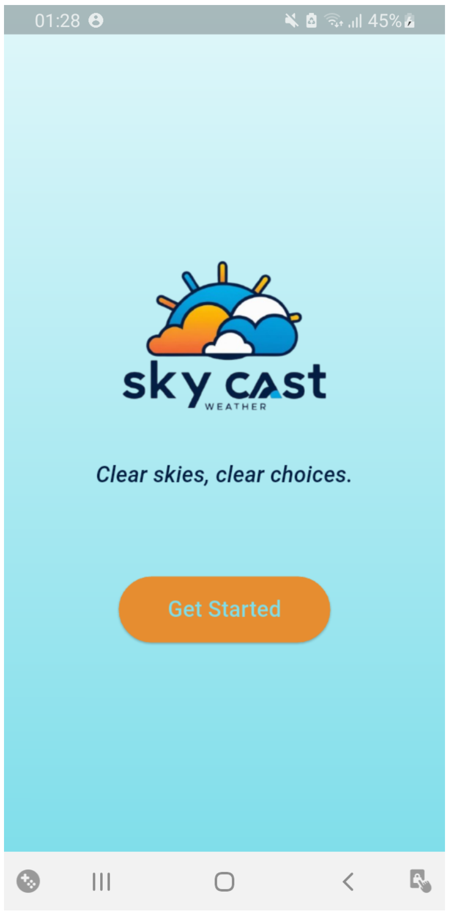


Figure 2: Welcome screen

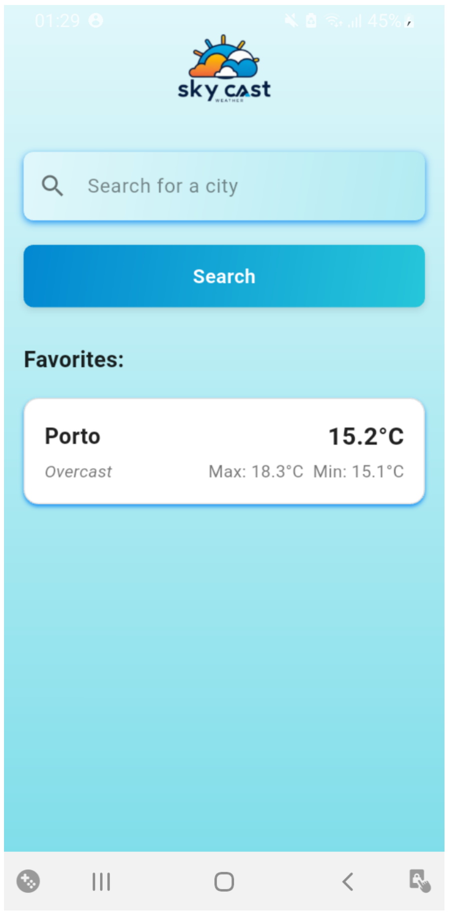
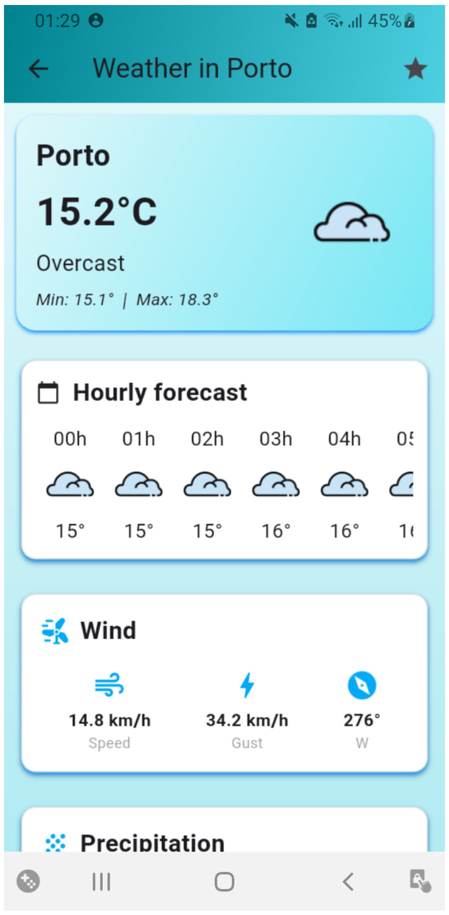


Figure 3: Main screen with search and favorites



Porto

15.2°C

Overcast

Min: 15.1° | Max: 18.3°

Hourly forecast

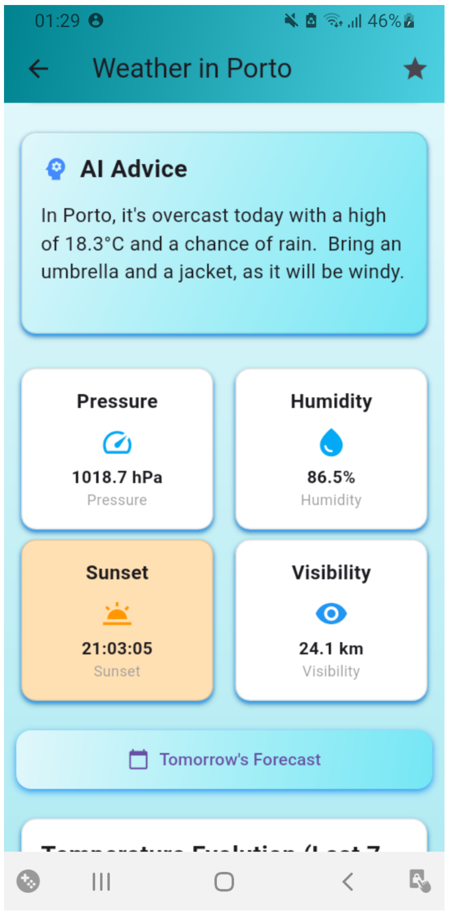
00h	01h	02h	03h	04h	05
15°	15°	15°	16°	16°	1

Wind

14.8 km/h	34.2 km/h	276°
Speed	Gust	W

Precipitation

Figure 4: Current weather and hourly forecast

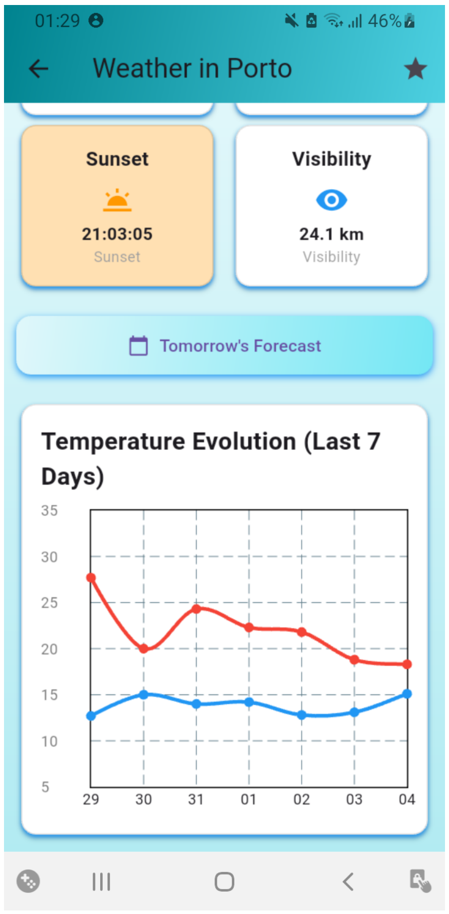


AI Advice

In Porto, it's overcast today with a high of 18.3°C and a chance of rain. Bring an umbrella and a jacket, as it will be windy.

Pressure	Humidity
1018.7 hPa	86.5%
Pressure	Humidity
Sunset	**Visibility**
21:03:05	24.1 km
Sunset	Visibility

Figure 5: AI-based advice section & other info



Sunset

21:03:05

Sunset

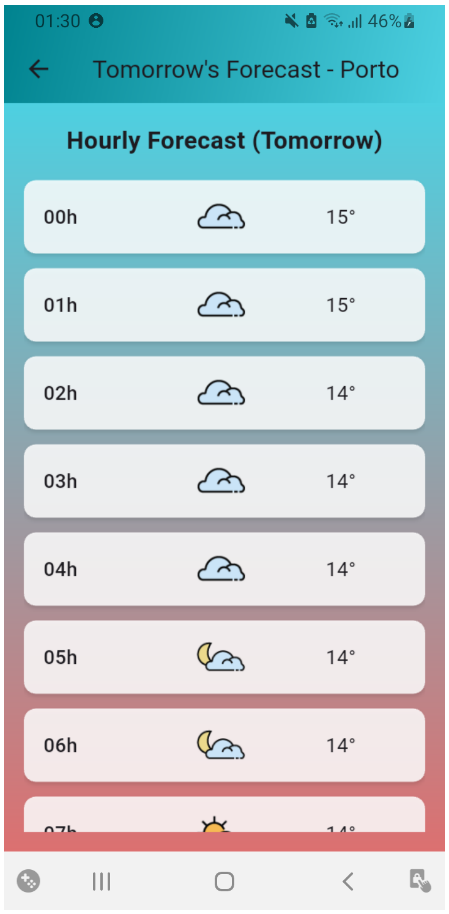
Visibility

24.1 km

Visibility

Temperature Evolution (Last 7 Days)

Figure 6: Temperature evolution chart (7 days)



Tomorrow's Forecast - Porto

Hourly Forecast (Tomorrow)

00h	15°
01h	15°
02h	14°
03h	14°
04h	14°
05h	14°
06h	14°

Figure 7: Tomorrow's hourly forecast

6 Conclusion

This application successfully demonstrates how a simple and intuitive user interface, combined with external APIs and modular architecture, can provide a smooth and informative weather-checking experience. Designed with scalability and cross-platform potential in mind, the application showcases modern Flutter development practices while delivering valuable functionality to users in real time.