

WEATHER APP

[Chosen Technology Stack 3](#_Toc200895199)

[Frontend Technologies 3](#_Toc200895200)

[Backend Technologies 4](#_Toc200895201)

[Application Architecure 5](#_Toc200895202)

[Description of Application Functionality 6](#_Toc200895203)

[User Interface Description 7](#_Toc200895204)

[Summary 9](#_Toc200895205)

[Sources:- 10](#_Toc200895206)

# Chosen Technology Stack

For this weather application, we decided to work mainly with OpenWeatherMap API for backend and .Net MAUI and XAML for frontend.

## Frontend Technologies

.NET MAUI (Multi-platform App UI)

.NET MAUI is Microsoft’s latest framework for developing native applications using a single codebase. It helps developers build applications that run smoothly across Android, iOS, macOS, and Windows. MAUI hides complexity of platform-specific APIs into a unified development experience, significantly reducing time and effort in cross-platform development.

Formerly, Xamarin.Forms, .NET MAUI introduces enhanced performance, a more coherent API surface, and better integration with modern .NET versions, making it the future of Microsoft’s cross-platform development stack.

Key Features include;

Single Project Structure – One project targets all platforms.

Hot Reload – Instantly view UI changes without rebuilding.

Platform Services – Access device features like GPS, camera, sensors with shared code.

Dependency Injection – Built-in support for clean architecture.

Performance – Native compilation for high-speed execution.

XAML (Extensible Application Markup Language)

XAML is a declarative markup language used for designing user interfaces in .NET MAUI applications. It allows developers to separate UI design from business logic, typically following the MVVM (Model-View-ViewModel) design pattern. XAML enhances maintainability, facilitates UI reuse, and improves developer productivity with features like data binding and control templates.

Key Features include;

Declarative Syntax – Define UI layout clearly and concisely.

Data Binding – Bind UI elements directly to ViewModel properties.

Templates & Styles – Create reusable visual components.

Animations & Transitions – Define visual behavior in markup.

MVVM Support – Seamless integration with ViewModels for logic separation.

C# and .NET 9.0

C# is the primary programming language used for developing MAUI applications. It offers strong typing, modern object-oriented features, and deep integration with the .NET ecosystem. The upcoming .NET 9.0 runtime introduces performance improvements, enhanced diagnostics, and expanded support for native device features. Together, C# and .NET 9.0 provide a robust foundation for scalable and maintainable applications.

.NET MAUI Essentials

Previously known as Xamarin.Essentials, .NET MAUI Essentials is a library that provides a collection of cross-platform APIs to access native device capabilities such as battery status, device sensors, geolocation, file system, and connectivity. In .NET MAUI, these APIs are integrated by default, enabling simplified access to hardware features across all supported platforms without writing platform-specific code.

## Backend Technologies

OpenWeatherMap API

In the WeatherNow app, OpenWeatherMap acts as the external backend service that supplies weather data. The app does not store or generate weather information on its own — instead, it sends a request to the OpenWeatherMap API and receives data in real-time, which is then displayed through the frontend.

# Application Architecure

1. User Interface (UI) with XAML

* XAML defines the layout and visual structure of the app.
* It includes UI elements like the Search Bar, Weather Display, Temperature Label, Weather Icon, and Forecast Cards.
* All these elements are arranged using stack layouts, grids, or scroll views.

2. User Interaction

* The user enters a city name into a search field.
* They tap a button (or use "Enter") to trigger a command.
* UI elements respond immediately to user input with animations or data loading indicators.

3. Data Binding (XAML ⇄ C# ViewModel)

* XAML UI elements are bound to properties in the ViewModel.
* For example, when the ViewModel receives new temperature data, the corresponding label on the screen updates automatically.
* This is done using MVVM (Model-View-ViewModel), where the ViewModel acts as the bridge between the UI and backend logic.

4. Live Updates

* When new weather data is received (e.g., after a city is searched), the frontend updates:
  + City name
  + Temperature
  + Description (e.g., “clear sky”)
  + Weather icon
  + Humidity, wind speed, etc.

5. Cross-Platform Display

* Thanks to .NET MAUI, the same frontend code works across:
  + Android phones
* The layout automatically adjusts to screen size and orientation.

6. Visual Feedback

* The app gives visual feedback to the user:
  + Loading indicators when fetching data
  + Error messages if a city isn't found
  + Vibrations or animations for better experience (optional using MAUI Essentials)

# Description of Application Functionality

The WeatherNow App is a cross-platform mobile and desktop application designed to provide users with accurate and real-time weather information. Built using .NET MAUI, the app allows users to search for weather conditions in any city and view key environmental metrics through a clean and interactive interface.

Core Functionalities:

1. City-Based Search

Users can enter the name of any city into the search bar at the top of the screen. After typing the desired city, the user clicks the Search button. The app sends a request to the OpenWeatherMap API to retrieve the weather information for that specific city. The screen updates in real time to display:

* Temperature
* Weather condition (e.g., "Cloudy", "Sunny")
* Humidity and wind speed
* A relevant weather icon

2. "My Location" Weather Fetching

By clicking the "My Location" button, the app uses the device's GPS to get the current geographic coordinates. It then sends those coordinates to the OpenWeatherMap API to get the weather for the user’s exact location, even if they haven’t entered a city manually. This feature is powered by .NET MAUI Essentials’ Geolocation API.

Additional Behaviors:

The app displays loading feedback during data fetching to inform the user that the weather is being retrieved. If the city name is invalid or there's no internet connection, the app handles errors gracefully with an appropriate message.

# User Interface Description

The WeatherNow App features a clean, modern, and user-friendly interface designed using XAML in .NET MAUI. The layout is simple yet visually effective, prioritizing ease of use, clarity of information, and responsiveness across devices.

* Minimalist Layout: The UI avoids clutter by focusing only on essential weather elements, making the experience fast and intuitive.
* Central Alignment: Key weather information (temperature, city name, icon) is centrally aligned for quick visibility.
* Color Scheme:
  + Primary Color: Purple top bar or accent for branding and contrast
  + Background: Light or white background for readability
  + Icons: Colored weather icons (yellow sun, gray clouds, etc.) for immediate recognition

Layout Components

1. Search Bar
   * Located at the top of the screen
   * Allows users to input a city name
   * Triggers the weather search functionality
2. City and Weather Information Display
   * Displays:
     + City name
     + Temperature (°C)
     + Weather description (e.g., "Cloudy", "Sunny")
     + Weather icon
   * Positioned prominently in the center for quick scanning
3. Additional Weather Details
   * Includes humidity and wind speed
   * Arranged below the main temperature and icon
   * Displayed with consistent styling and spacing
4. Forecast Cards (if implemented)
   * Horizontally scrollable section
   * Displays time-based forecasts (e.g., next few hours)
   * Includes mini icons, temperature, and time stamps
5. Error Messages / Alerts
   * Shown when the user enters an invalid city or if there's no internet connection
   * Uses toast messages, dialogs, or in-line alerts

# Summary

WeatherApp is a cross-platform weather application built using .NET MAUI, enabling users to view real-time weather information for any city worldwide. The app offers a clean and responsive user interface, accurate data integration via the OpenWeatherMap API, and is structured using the MVVM (Model-View-ViewModel) design pattern. It runs on Android, iOS, Windows, and macOS from a single codebase.

# Sources

Microsoft. “.NET MAUI Documentation.” Microsoft Learn, Microsoft, <https://learn.microsoft.com/en-us/dotnet/maui/>. Accessed 16 June 2025.

Microsoft. “XAML Overview for .NET MAUI.” Microsoft Learn, Microsoft, <https://learn.microsoft.com/en-us/dotnet/maui/user-interface/xaml/>. Accessed 16 June 2025.

Microsoft. “Platform Integration with .NET MAUI Essentials.” Microsoft Learn, Microsoft, <https://learn.microsoft.com/en-us/dotnet/maui/platform-integration/>. Accessed 16 June 2025.

OpenWeather. “Weather API.” OpenWeatherMap, OpenWeather Ltd., https://openweathermap.org/api. Accessed 16 June 2025.

Microsoft. “Visual Studio IDE.” Visual Studio, Microsoft, <https://visualstudio.microsoft.com/>. Accessed 16 June 2025.

GitHub. “GitHub Docs.” GitHub, <https://docs.github.com/en>. Accessed 16 June 2025.

Sharma, Tapas. “.NET MAUI Weather App – Full Tutorial Playlist.” YouTube, uploaded by Tapas Sharma, <https://youtube.com/playlist?list=PL3fvNBbJYqs6MTRpTthdTbAL5MKXovm91>. Accessed 16 June 2025.