Weather App

Overview:

WeatherApp is a ReactJS + .NET 6 application that retrieves a simplified weather report of the provided city and country (optional).

This application consists of mainly the frontend application and the backend API. The frontend application is built using the React components. The backend is a Web Api that is built using .NET 6.

Architecture:

The implementation of this application follows shadows of the Clean Architecture pattern making it easier to develop quality code that performs better, can be changed easily and has a very few dependencies.

Implementation Details:

WeatherApp implementation consists of 5 projects with the requests being served in the order of their explanation. Following are the details of each of the project in details:

|  |  |
| --- | --- |
| Front-End / Web-Api |  |
| Front-End | **Components**   * Common   + Message – displays error/warning message * Weather   + Location – inputs for city & country   + Report – displays weather data   + Weather - <Location />, <Report />   **Services**   * WeatherService – Sends request to the local end-point and receives response |
| Web-Api | **Controller**   * WeatherController   + Index     - WeatherRequestModel     - CancellationToken   + **Models** * WeatherRequestModel   + string City   + string Country   + **Middlewares** * ApiKeyAuthenticationMiddleware – Authenticates requests for API Keys and proceeds further or returns 401 (Unauthorized) * CustomRateLimitingConfiguration – Tracks number of requests for each API Key and returns 429 (too many requests) if limit exceeds |
| Application | **Requests Handlers**   * IRequestHandler – generic request handler that accepts TRequest and returns TResponse   **UseCases**   * Weather   + ICurrentWeatherRequestHandler – interface to communicate with the Application layer   + CurrentWeatherRequestHandler – Implementation that handles request to get current weather. This handler requests the weather service to get the current weather for the given location.   **Services**   * IWeatherService – Interface to server current weather for the given location. |
| Infrastructure | * Services   + WeatherService – Implementation that communicates with the external weather Api client and gets the current weather for the requested location. |
| ExternalWeather Api.Client | * Api Clients   + ExternalWeatherApiClient – Implementation to create a http-client, set the request URL, and send a request message. On receiving the HttpResponseMessage, this client returns it to the WeatherService without any changes. * DTOs   + WeatherData (maps response from the api request)     - string City     - string ErrorMessage     - System       * string CountryCode     - ICollection<Weather> WeatherList       * string Description * Options   + string ApiBaseUrl   + string ApiEndpoint   + string ApiKey |

# Future enhancements

* File logging using SeriLog or NLog nuget package
* API model validation using fluent validation for custom validations and unit testing them
* Extend swagger documentation
* Make external Api client more configurable for the request Uri to be independent of the format
* Implement better and testable UI

Flow Diagram:

