### **Reflective Report**

### **Server-Side Caching in Weather Application**

**Introduction**

The weather application prototype retrieves and shows the weather of a specified city. In the initial stage, the system used to retrieve the weather information from the OpenWeatherMap API directly on every user request. The caching mechanism using PHP and MySQL is implemented in the server side in order to reduce dependency on other services, thereby reducing the costs.

**Strengths of the Architecture**

Reduces the number of calls to APIs: The primary objective of designing the architecture was a reduction in external API requests frequency. The data is cached about the weather in each city, and there's a MySQL database.It queries OpenWeatherMap if the cache is expired. That is, every 30 minutes.

This system can scale easily because it is using a relational database to store weather data. Cities along with their corresponding weather data can be stored in the cache without affecting the performance of the system.

**API Requests using cURL:** The use of cURL in PHP allows a powerful method of making HTTP requests to external services. This makes the system more dependable as it can handle potential problems such as timeouts and network failure.

**Weaknesses of the Architecture**

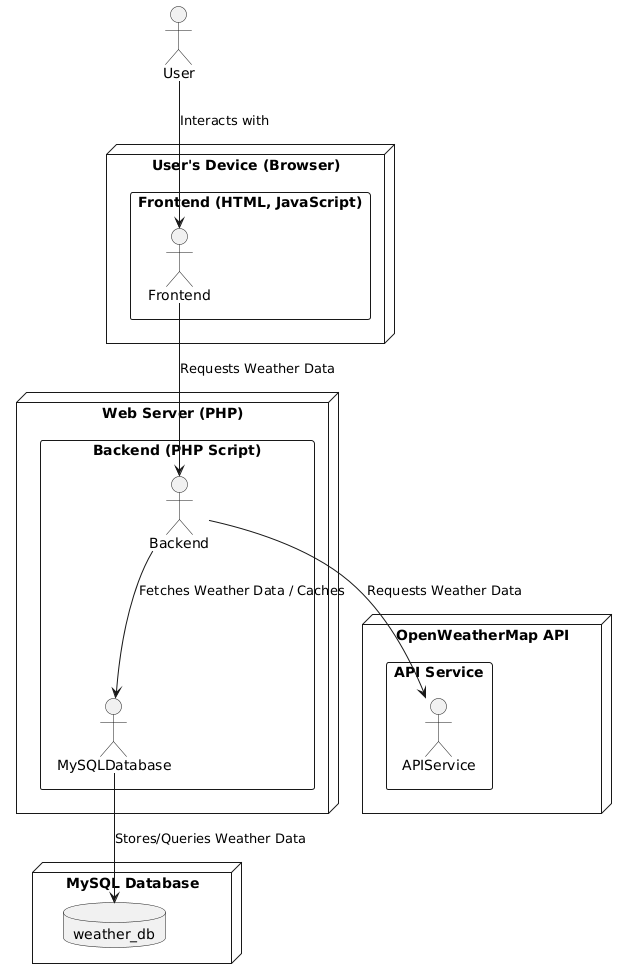
**Static Cache Expiry Time:** A fixed expiry time for cache, set to 30 minutes, is the foundation for the architecture being proposed. Such fixed expiry time may not necessarily be ideal under all circumstances, as some data might change really fast, whereas others remain practically stable.

**Data Staleness:** There still exists a slight possibility of service staleness-even with caching-instating outdated weather data in instances where the cached version is fetched just before some user request arrives.

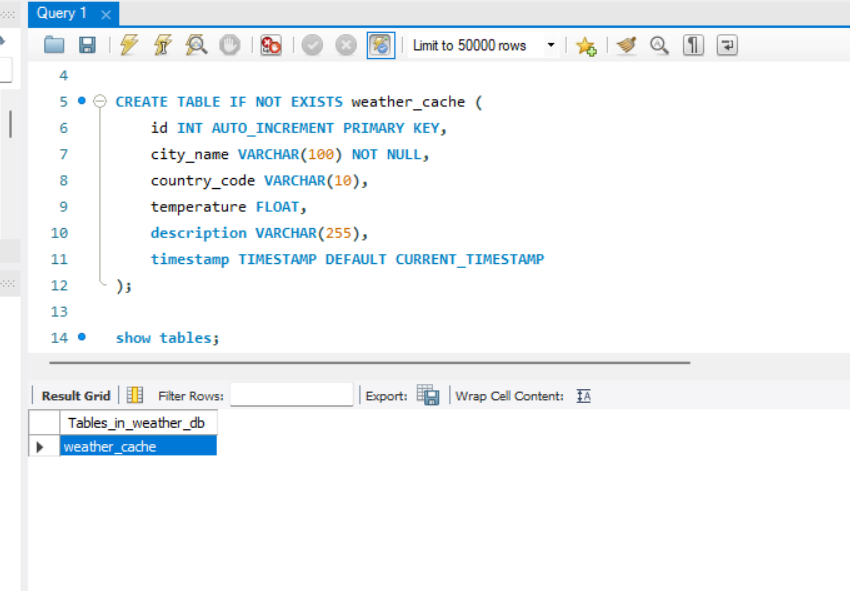
This system only stores the basic weather information, such as temperature and description. Other parameters, like wind speed, humidity, and icons for the type of weather, can be used to enhance the experience.

**Single Source of Data:** The application depends entirely on the OpenWeatherMap API for weather data. If the API is down or changes its data structure, the system will be affected. Fallback mechanisms or diversifying data sources may reduce reliance on a single external API.

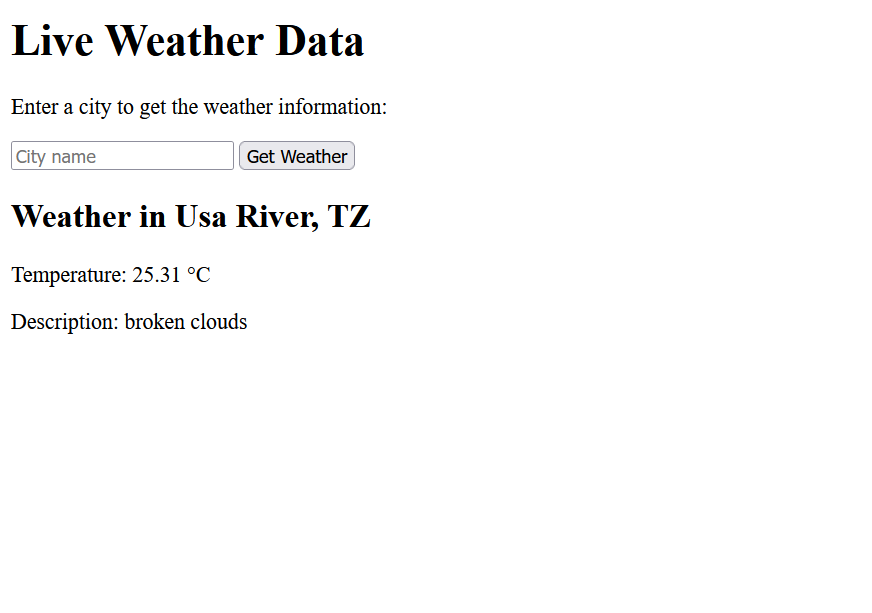
**UML Diagram**



Saving in Mysql:



**Api Response:**



**Conclusion**

The server-side caching architecture drastically enhances the performance and scalability of the weather application. As the data is cached in MySQL, the external API calls will not be required every time. As a result, the weather data will come fast, reducing cost. This design can still be improved on because the policy governing the cache is not that flexible, it may not support data storage beyond basic types, or it could just have other fallback mechanisms for redundancy.