UNIVERSITY OF WOLVERHAMPTON

BIRATNAGAR INTERNATIONAL COLLEGE

4CS017 – Internet Software Architecture

Lab Report

Name: Bhawanath Sapkota

Group: L4cG2

Student ID: NP02CS4A240060/2513314

Module Leader

_____

Tutor

Arvind Nepal

Reflective report on the strengths and weaknesses of weather app's architecture

Introduction

The backbone of this architecture for the weather application is the mixture between PHP in the backend processing and JavaScript at manipulating webpage data at the client side, enabling the application to do a good balance of data persistence with real-time updates. Here now is a reflection of strong points and weaknesses:

Strengths

Database Caching Mechanism :
It utilizes a MySQL database for efficient storage of the weather data, from which it directly pulls the data, rather than having to make several requests to OpenWeatherMap. It caches data and checks for timestamps to avoid making requests it doesn't need to the API, hence improving performance and lowering API-related costs.

Dynamic User Interface:
JavaScript provides real-time updates, hence making the application interactive and efficient. It provides for better usability through features like switching temperature units, i.e., Celsius/Fahrenheit, and live weather updates of places given by the user.

Default Fallback Mechanism:
A default location can be set to Kathmandu, which can let the user see some of the weather data, but at least that related to him generally speaking. Also, in addition, it keeps the app somewhat more reliable since one is not being thrown into an empty state.

Scalability:
Segregation of the PHP back-end logic and the front-end logic done using JavaScript keeps the app very scalable, with easily hosted multilingual support or for greater location coverage, without many major architectural changes.

Error Handling:
It does have a fairly good number of error-handling methods that can handle failure in database connections, bad requests-which usually means invalid location, API failures, etc. This robustness greatly reduces the chances of crashes and provides meaningful feedback to the user.

Weaknesses

Use of Feared Frontend Framework:

It adds interactive features, but without tools such as React or Vue, it is a bit more work scaling up and maintaining for advanced features, such as offline mode or turning into a full-fledged PWA.

No Asynchronous Job Queuing:
Data from APIs is fetched and saved in real time, which results in slower response time in high traffic. The job queue or background workers could offload such tasks-indexing and so forth-from the main thread, improving performance.

Security Vulnerabilities:
Architecture has no authentication and input sanitization and is thus open to SQL injection or XSS attacks. Strong security practices should be implemented in order to mitigate risks.

Hardcoding of API Key:
The API key is hardcoded in the PHP script, which is a serious vulnerability in its usage. It should be set instead in the environment file or a secure configuration.

JavaScript Static Default Data:
The JavaScript holds hardcoded alternative weather data for Kathmandu, and it can get stale. It can be synced with the backend or live responses from the API to get more precise results.

Conclusion

While the app does balance both categories, it sets up user interactivity well, which is quite impressive. Still, it misses some key features to make it production-ready, like security against attacks, not using hard-coded values, etc. Then again, if these are improved upon, this will make the app more robust, scalable, and easier to maintain.