Phy Vathanak SE-G3

LAB W3

EXERCISE 1 – Refactoring

Q1 – What challenges did you face when using the native http module that Express.js helped you solve?

With express I don't need manual routing, and it provides middleware to work well with JSON data.

Q2 – How does Express simplify route handling compared to the native HTTP server?

Native HTTP we need to check the method (get,post,...) when get request to identify which method use but in express it provides function to handle this.

Q3 – What does middleware mean in Express, and how would you replicate similar behavior using the native module?

Middleware are functions that run before routing to check the logic is easier because native module, we need to call function every time when handle route.

REFLECTIVE QUESTIONS

Middleware & Architecture

1. What are the advantages of using middleware in an Express application?

- save time we can call function only one time for many route

- allow to manage validation or authentication before route

2. How does separating middleware into dedicated files improve the maintainability of your code?

- each file responsible for one task so it easy to update or debug or scale the application

- easy to read the code

3. If you had to scale this API to support user roles (e.g., admin vs student), how would you modify the middleware structure?

I will add modify on authenticate middleware after verify token it will get the user data and then we check the role which admin or student. And i will create one more middleware to allow specific role can access to a route.

Query Handling & Filtering

4. How would you handle cases where multiple query parameters conflict or are ambiguous (e.g., minCredits=4 and maxCredits=3)?

I would use middleware to validate query parameters before route. If conflict it would return a bad request and error message.

5. What would be a good strategy to make the course filtering more user-friendly (e.g., handling typos in query parameters like “falll” or “dr. smtih”)?

Security & Validation

6. What are the limitations of using a query parameter for authentication (e.g., ?token=xyz123)? What alternatives would be more secure?

Limitation: token is visible in browser so it can change frequently to the correct one or would be stolen by other.

More Secure: using post method instead because it not visible in browser and should encrypt it also

7. Why is it important to validate and sanitize query inputs before using them in your backend logic?

Because

- Protect backend logic and prevent crashes caused by invalid or unexpected inputs
- Ensure data integrity by allowing only valid and well-formed data to be processed and sent to the client
- Prevent security such as injection attacks, by blocking malicious input from users.

Abstraction & Reusability

8. Can any of the middleware you wrote be reused in other projects? If so, how would you package and document it?

Yes, it can put middle in specific file and standalone module. And then add the document how to use and the feature of each middleware. And include unit text to ensure reliability across project.

9. How could you design your route and middleware system to support future filters (e.g., course format, time slot)?

- Create middleware that accepts options.
- Implement filters for each part
- Routes to accept any query filters

Bonus – Real-World Thinking

10. How would this API behave under high traffic? What improvements would you need to make for production readiness (e.g., rate limiting, caching)?

Under high traffic

- Server can down
- Take a long time to get the data

Improvement:

- Increase server capability
- Using queue to get the APT to avoid server down
- Use rate limiting for user to access in a limitation