AeroAspire - SDE Intern

Gokul Krishna S

Week 3 – Day 5 (October 11)

Questions/Reflections:

1. Explain full request-response cycle including network, HTTP headers.

- Request-Response Cycle (Network & HTTP Headers)
- **Step-by-step:**
- **Client initiates a request:** You click a button or load a page. The browser or app prepares an HTTP request.
- **DNS resolution:** The domain name (e.g., example.com) is translated to an IP address so your device knows where to send the request.
- **TCP connection:** Your device opens a network connection to the server (often over HTTPS for security).
- **HTTP request sent:** The client sends an HTTP request, which includes:
- Method (GET, POST, etc.)
- URL (the resource you want)
- Headers (extra info like Content-Type, Authorization, User-Agent)
- Body (for methods like POST, contains data)

- **Server processes request:** The server receives the request, reads headers and body, and does its work (e.g., queries a database).
- **Server sends response:** The server replies with:
- Status line (HTTP/1.1 200 OK)
- Response headers (e.g., Content-Type, Set-Cookie, Cache-Control)
- Body (HTML, JSON, etc.)

- **Client receives response:** The browser/app reads the response, processes headers, and displays data or takes action.
- Headers are key for passing extra info:
- **Request headers:** Tell the server about the client, what data is accepted, authentication, etc.
- **Response headers:** Tell the client about the data, caching, cookies, and more.

2. What do you need to consider for security (e.g. input validation, sanitization, auth) even if simple app.

- **Input validation:** Always check and validate user input (e.g., form fields, query params) to prevent bad data or attacks.
- **Sanitization:** Clean input to remove dangerous characters (e.g., escaping HTML to prevent XSS).
- **Authentication:** Make sure users are who they say they are (e.g., login with password, tokens).
- **Authorization:** Check if the user is allowed to do what they're trying to do (e.g., only admins can delete data).
- **HTTPS:** Use encrypted connections to protect data in transit.
- **Limit exposure:** Only expose necessary endpoints and data.
- **Error handling:** Don't leak sensitive info in error messages.

3. How would you monitor errors in production?

- To keep your app healthy, you need to know when things go wrong:
- **Logging:** Record errors, warnings, and important events to log files or a logging service.
- **Error tracking tools:** Use services like Sentry, Rollbar, or custom dashboards to collect and alert on errors.
- **Health checks:** Set up automated checks to make sure endpoints are working.
- **Alerts:** Configure notifications (email, Slack, etc.) for critical errors or downtime.
- **Metrics:** Track response times, error rates, and other key stats.