Date: 19 June 2024 Day: Wednesday

Overview:

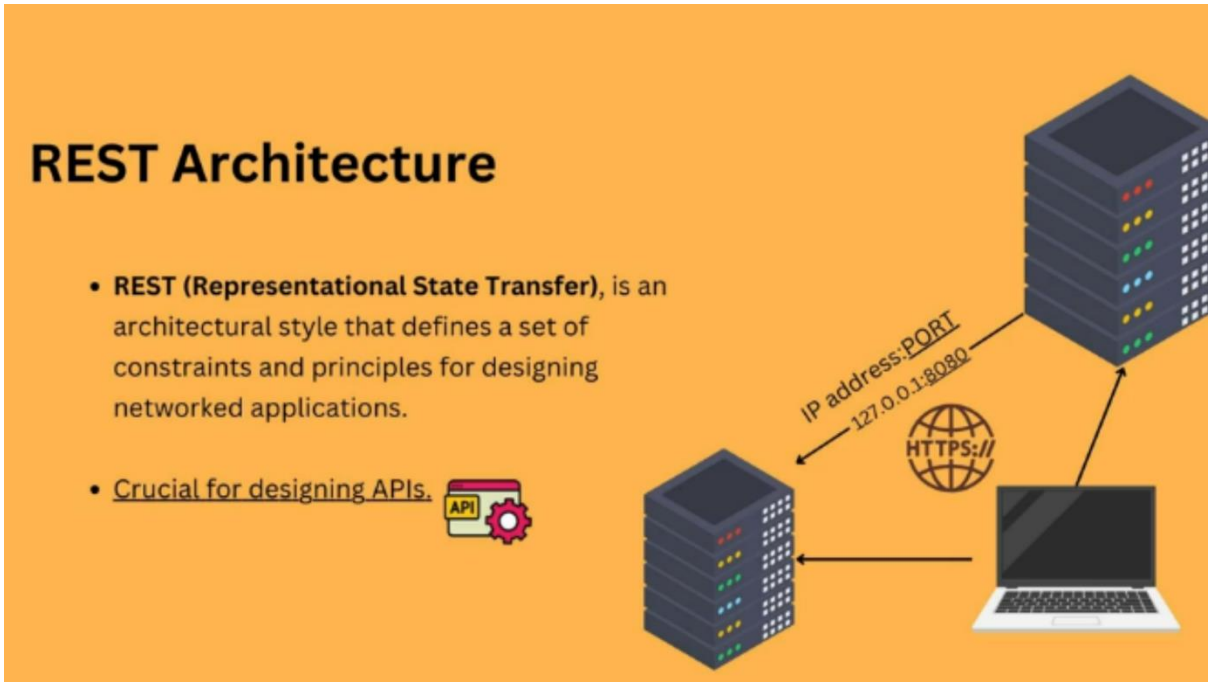
Day 11 of the internship focused on understanding the fundamentals of the web and REST APIs, essential for developing modern web applications using Node.js. This included exploring HTTP fundamentals, utilizing the Node.js http module for server-side operations, and implementing HTTP request and response handling.

Learning Objectives:

- Understanding How the Web Works:
- Explored the architecture and components of the World Wide Web, including clients, servers, browsers, and URLs.
- Studied the role of HTTP (Hypertext Transfer Protocol) in facilitating communication between clients (browsers) and servers.

Introduction to REST APIs:

- Discussed REST (Representational State Transfer) principles and its significance in designing scalable and interoperable web services.
- Explored RESTful API conventions, including resource endpoints, HTTP methods (GET, POST, PUT, DELETE), status codes, and payload formats (JSON, XML).



Node.js HTTP Module:

- Covered the Node.js http module for creating HTTP servers and handling HTTP requests and responses:
- Creating HTTP Servers: Utilized http.createServer() to create a server that listens for incoming requests.
- Handling Requests and Responses: Implemented request event listeners (request) to handle incoming client requests and generate appropriate responses.

Activities and Insights:

- Implementation of HTTP Fundamentals:
- Developed Node.js applications to demonstrate HTTP server creation and basic request-response handling using the http module.
- Practiced parsing request URLs, extracting query parameters, and processing headers and body content in HTTP requests.

Exploring REST API Concepts:

- Designed and implemented RESTful API endpoints using Node.js, adhering to REST principles for resource management and CRUD operations.
- Integrated middleware functions for request validation, authentication, and error handling to enhance API security and reliability.

Documentation and Best Practices:

- Documented HTTP server configurations, API endpoint definitions, and usage examples for client interaction and testing.
- Discussed best practices for designing RESTful APIs, including URI design, versioning, response formatting, and error handling strategies.