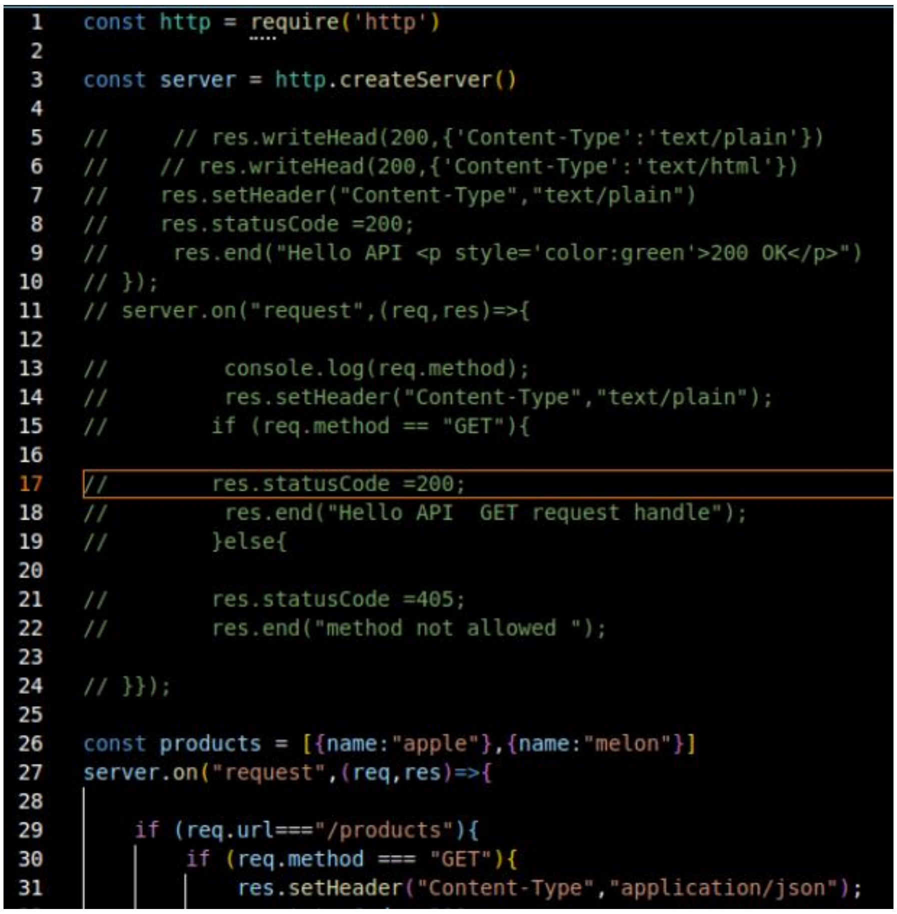
Date: 20 June 2024 **Day: Thursday**

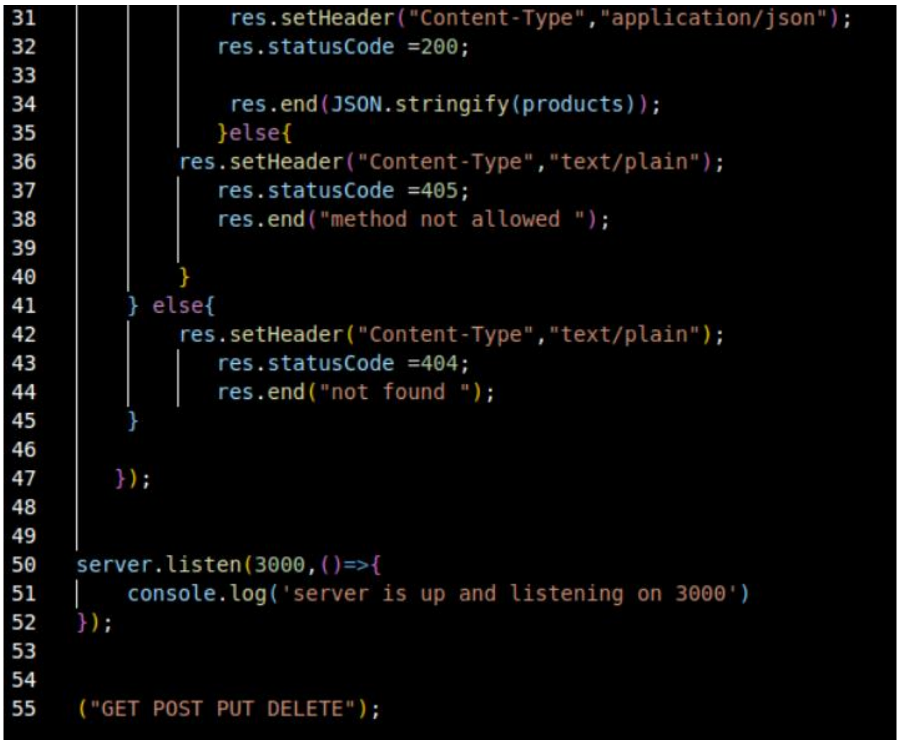
Day 12 of the internship focused on practical implementation of handling GET requests in Node.js applications. This involved exploring routing and URL handling concepts to effectively respond to client requests and returning JSON responses from GET endpoints.

Learning Objectives:

- **Handling GET Requests in Node.js:**
    - Implemented server-side logic to handle GET requests using Node.js and the Express.js framework for routing.
    - Explored different approaches to define routes and handle specific HTTP GET requests for retrieving data from servers.
- **Exploring Routing and URL Handling:**
    - Discussed routing mechanisms in Node.js and Express.js to map HTTP methods (GET) to specific request handlers (controllers).
    - Utilized route parameters and query parameters in URLs to customize GET requests and fetch targeted data from servers.
- **Returning JSON Responses:**
    - Practiced formatting and returning JSON (JavaScript Object Notation) responses from GET endpoints:
    - Serialized JavaScript objects into JSON format using built-in methods or libraries like JSON.stringify().
    - Integrated JSON responses with Express.js response objects to send structured data to clients requesting information.



```
const http = require('http')

const server = http.createServer()

//     // res.writeHead(200,{'Content-Type':'text/plain'})
//    // res.writeHead(200,{'Content-Type':'text/html'})
//    res.setHeader("Content-Type","text/plain")
//    res.statusCode =200;
//     res.end("Hello API <p style='color:green'>200 OK</p>")
// });
// server.on("request",(req,res)=>{

//         console.log(req.method);
//         res.setHeader("Content-Type","text/plain");
//        if (req.method == "GET"){

//        res.statusCode =200;
//         res.end("Hello API  GET request handle");
//        }else{

//        res.statusCode =405;
//        res.end("method not allowed ");

// }});

const products = [{name:"apple"},{name:"melon"}]
server.on("request",(req,res)=>{

    if (req.url==="/products"){
        if (req.method === "GET"){
            res.setHeader("Content-Type","application/json");
```



```
            res.setHeader("Content-Type","application/json");
```

```
           res.statusCode =200;

            res.end(JSON.stringify(products));
           }else{
        res.setHeader("Content-Type","text/plain");
           res.statusCode =405;
           res.end("method not allowed ");

        }
    } else{
        res.setHeader("Content-Type","text/plain");
           res.statusCode =404;
           res.end("not found ");
    }

   });


server.listen(3000,()=>{
    console.log('server is up and listening on 3000')
});


("GET POST PUT DELETE");
```

Activities and Insights:

- **Implementation of GET Request Handling:**
    - Developed Express.js applications to define routes and middleware functions for handling specific GET requests.
    - Implemented CRUD (Create, Read, Update, Delete) operations using GET requests to fetch data from databases or external APIs.

- **JSON Response Formatting:**
    - Formatted JavaScript objects into JSON strings using JSON.stringify() for consistent and structured data representation in API responses.
    - Configured Express.js routes and middleware to send JSON responses (res.json()) with appropriate HTTP status codes (200 OK, 404 Not Found) based on request outcomes.