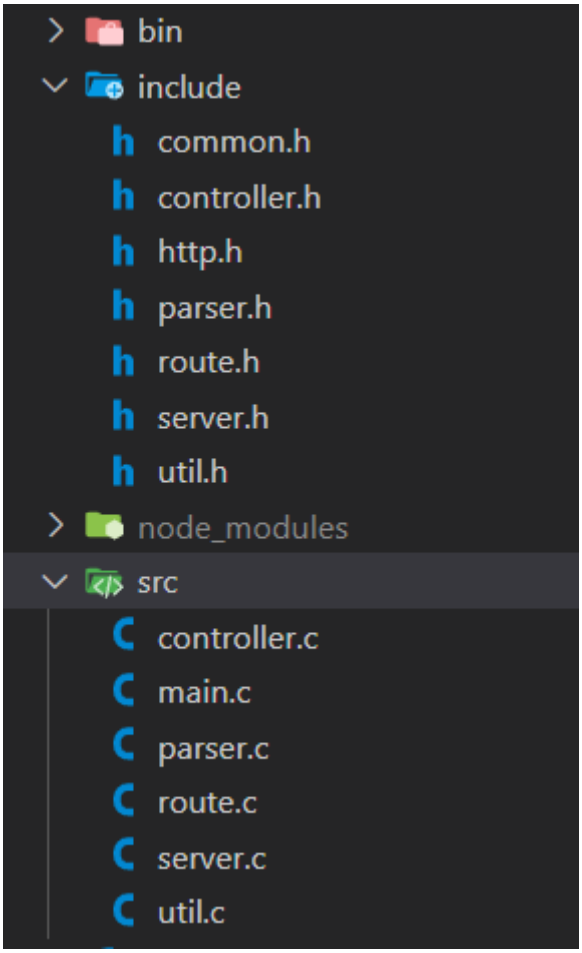
SisterJS

(disclaimer ini asli gangerti bener atau enggak, nikmati aja kode ampas saya)

NOTE: ini dibuat di ubuntu jadi keknya bakalan error kalo di windows ehe

Files



Kurang lebih begini terdapat 6 file .c dari source code yang mana

server.c: main server

util.c: isinya utilitas yang sering dipake

parser.c: parser buat

- Application/json
- Text/plain
- Application/x-www-form-urlencoded

route.c: untuk routing

controller.c: untuk handling request

- GET
- POST
- PUT
- DELETE

contoh dari controller tiap request

```c
void GET_example(int client_socket, HttpRequest *req, HttpResponse* res) {
    handle_request(client_socket, req, res);
}
```

```c
void POST_example(int client_socket, HttpRequest* req, HttpResponse* res) {
    handle_request(client_socket, req, res);
}
```

```c
void PUT_example(int client_socket, HttpRequest *req, HttpResponse* res) {
    handle_request(client_socket, req, res);
}
```

```c
void DELETE_example(int client_socket, HttpRequest *req, HttpResponse* res) {
    handle_request(client_socket, req, res);
}
```

Fungsi untuk handle request-nya

```c
void handle_request(int client_socket, HttpRequest* req, HttpResponse* res){
    bool is_valid = true;

    memset(res->response, 0, sizeof(res->response));
    if (req->content_type[0] == '\0' && (strcmp(req->method, "POST") == 0 || strcmp(req->method, "PUT")
== 0)) {
        sprintf(res->response, "{\"status\": \"error\", \"message\": \"Content-Type header
missing\"}");
        send_response(client_socket, "400 Bad Request", "application/json", res->response);
        return;
    }

    if (strcmp(req->method, "GET") == 0) {
        if (strlen(req->params) != 0) {
            snprintf(res->response, sizeof(res->response), "GET with params: %.2000s", req->params);
        } else {
            snprintf(res->response, sizeof(res->response), "GET Only");
        }
        send_response(client_socket, "200 OK", "text/plain", res->response);
        return;
    }

    if (strcmp(req->method, "POST") == 0 || strcmp(req->method, "PUT") == 0){
        parse_body(req->content_type, req->body, res->keys, res->values, &res->total_data);


        if (!is_valid) {
            strcat(res->status, "Unsupported content type");
            sprintf(res->response, "{\"status\": \"error\", \"message\": \"Unsupported content
type\"}");
            send_response(client_socket, "400 Bad Request", "application/json", res->response);
            return;
        }
        printf("method sebelum http: %s\n", req->method);

        generate_response_http(req, res);
        send_response(client_socket, "200 OK", "application/json", res->response);
        memset(res->response, 0, sizeof(res->response));
        return;
    }

    if (strcmp(req->method, "DELETE") == 0){
        if (res->total_data > 0) {
            strcat(res->response, "{\"status\": \"deleted\", \"data\": {");
            for (int i = 0; i < res->total_data; i++) {
                int pair_length = strlen(res->keys[i]) + strlen(res->values[i]) + 6;
                char* pair = malloc(pair_length * sizeof(char));
                sprintf(pair, "\"%s\": \"%s\"", res->keys[i], res->values[i]);
                strcat(res->response, pair);
                if (i < res->total_data - 1) {
                    strcat(res->response, ", ");
                }
                free(pair);
            }
            strcat(res->response, "}}");
        } else {
            strcat(res->response, "{\"status\": \"deleted\", \"data\": {}}");
        }
        send_response(client_socket, "200 OK", "application/json", res->response);
        return;
    }

    sprintf(res->response, "{\"status\": \"error\", \"message\": \"Unsupported HTTP method\"}");
    send_response(client_socket, "405 Method Not Allowed", "application/json", res->response);
}
```

Framework yang kubuat ini cuman bisa di-port 8080 gabisa diganti lmao, yaudah biarin aja yang penting jalan

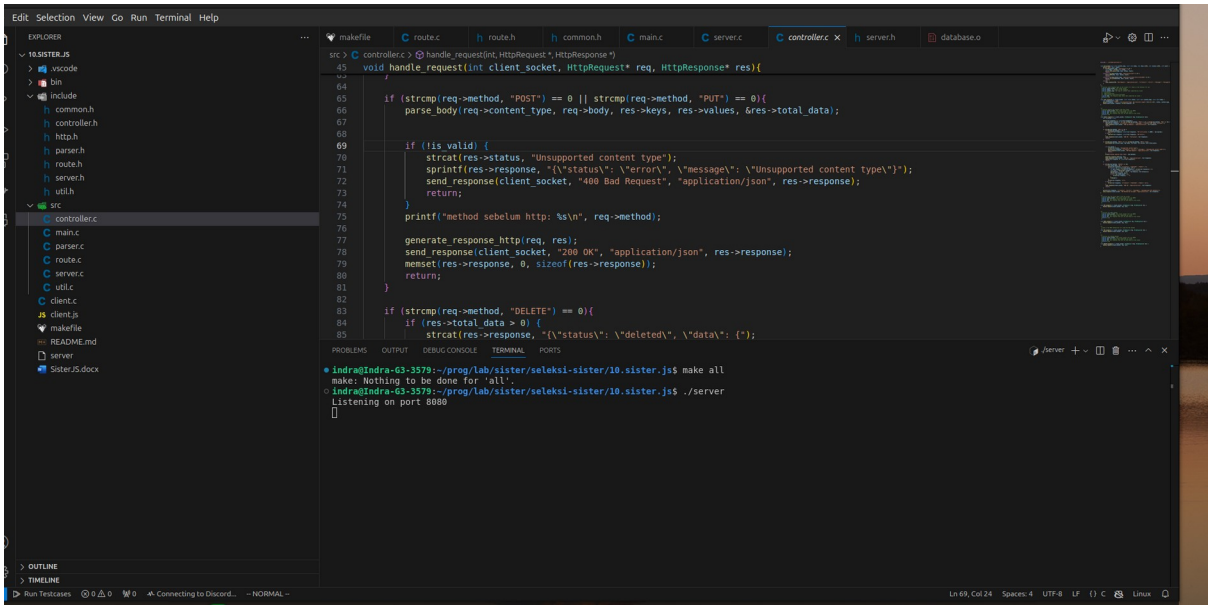
“if it’s work don’t touch it”

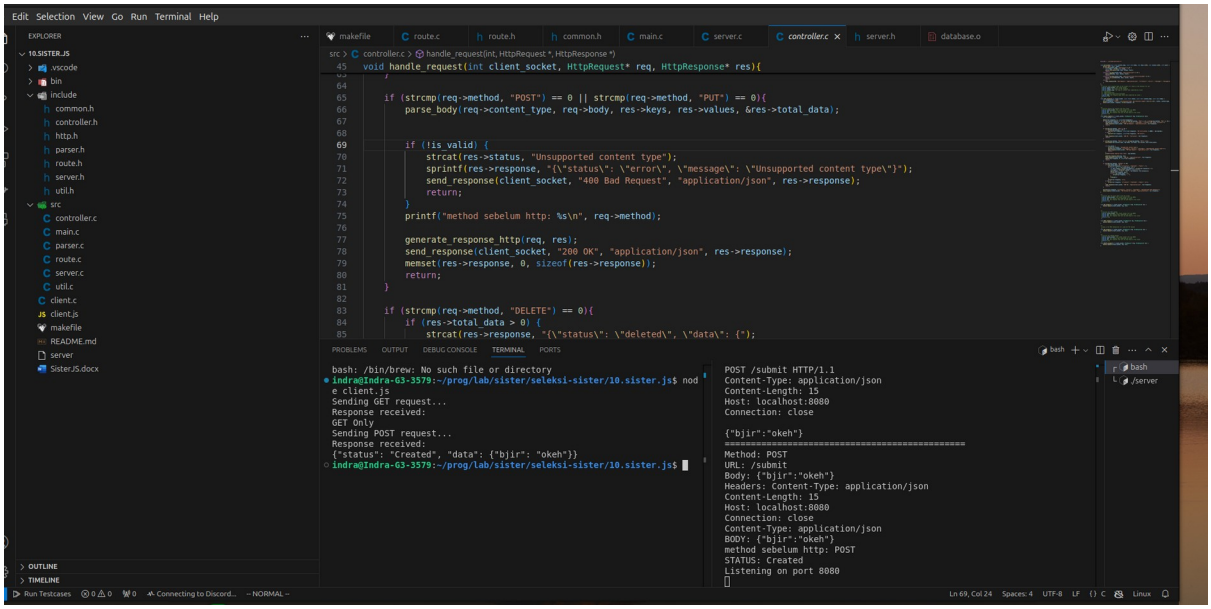
Cuplikan kode client, tapi pake javascript ngehe.

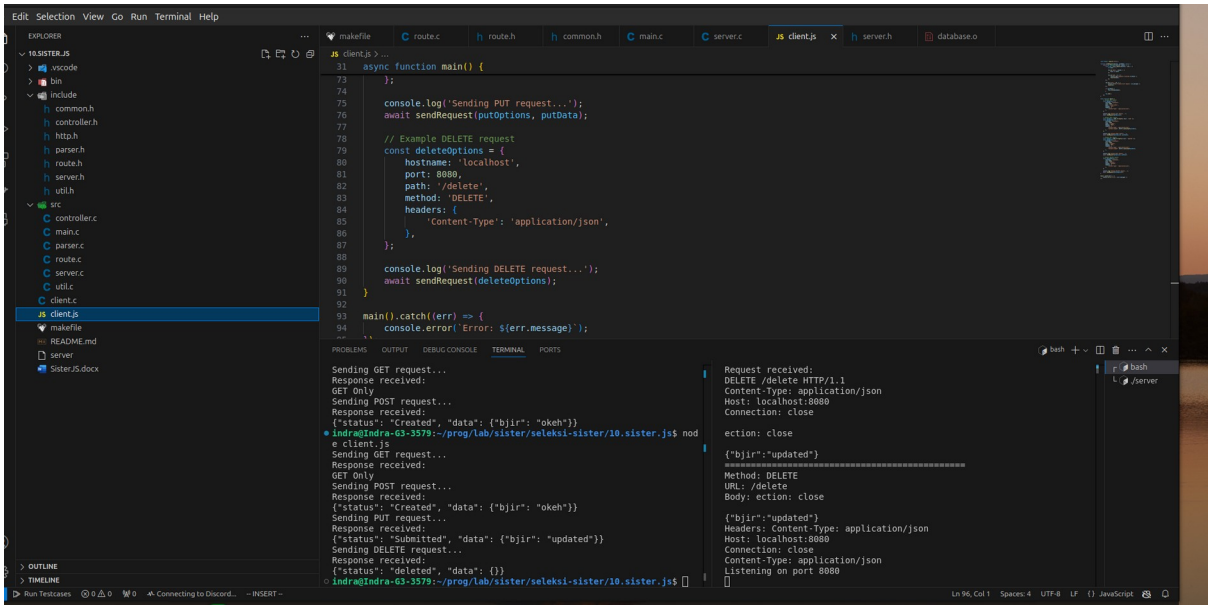
```
const http = require('http');

function sendRequest(options, postData = null) {
    return new Promise((resolve, reject) => {
        const req = http.request(options, (res) => {
            let data = '';

            res.on('data', (chunk) => {
                data += chunk;
            });

            res.on('end', () => {
                console.log(`Response received:\n${data}`);
                resolve(data);
            });
        });

        req.on('error', (e) => {
            console.error(`Problem with request: ${e.message}`);
            reject(e);
        });

        if (postData) {
            req.write(postData);
        }

        req.end();
    });
}

async function main() {
    // Example GET request
    const getOptions = {
        hostname: 'localhost',
        port: 8080,
        path: '/nilai-akhir',
        method: 'GET',
        headers: {
            'Content-Type': 'application/json',
        },
    };

    console.log('Sending GET request...');
    await sendRequest(getOptions);

    // Example POST request
    const postData = JSON.stringify({ bjir: 'okeh' });
    const postOptions = {
        hostname: 'localhost',
        port: 8080,
        path: '/submit',
        method: 'POST',
        headers: {
            'Content-Type': 'application/json',
            'Content-Length': Buffer.byteLength(postData),
        },
    };

    console.log('Sending POST request...');
    await sendRequest(postOptions, postData);
}

main().catch((err) => {
    console.error(`Error: ${err.message}`);
});
```

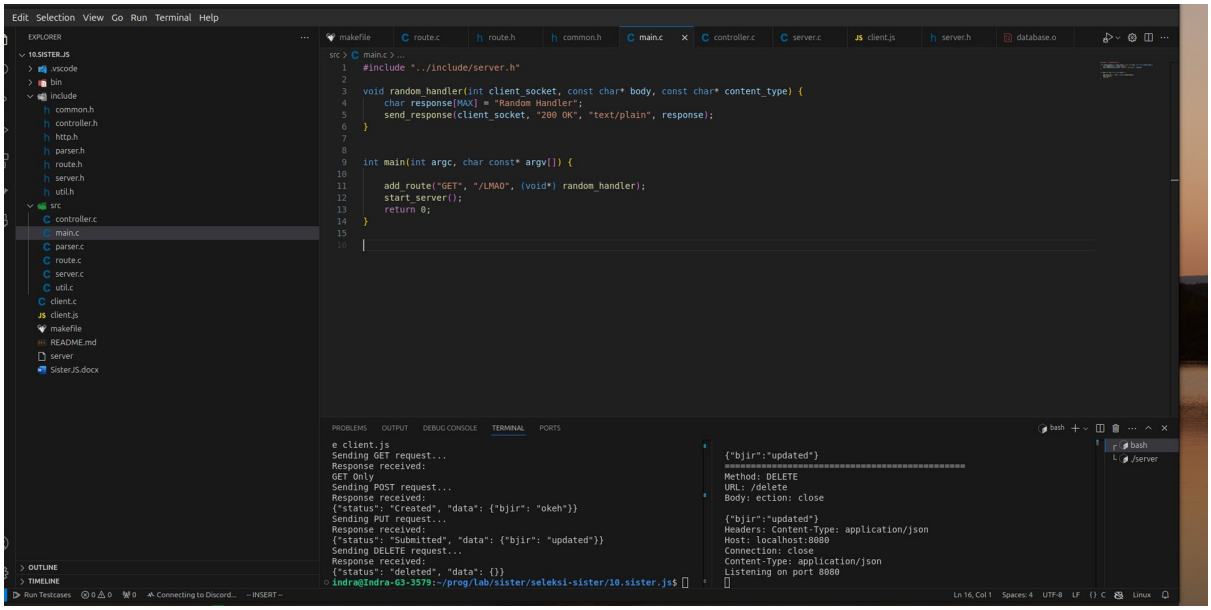
Source code yang saya buat ini tidak menggunakan database. Hanya dibuat untuk untuk terhubung saja, sumpah saya skill issue banget. Udah skill issue, tapi ngide ngerjain pake C, aneh emang orang gila satu ini (ya, itu saya).

Ini screenshot saat program jalan, langsung semuanya PUT, DELETE, GET, POST.









Sekian terima kasih, dan mohon maaf kalau ini jelek karena saya skill issue