**NAME: ZUFRA JAHAN**

**INTERNSHIP TASK 4**

**CODE:**

#include <iostream>

#include <thread>

#include <vector>

#include <winsock2.h>

#include <ws2tcpip.h>

#include <fstream>

#include <sstream>

#pragma comment(lib, "Ws2\_32.lib")

using namespace std;

const int PORT = 8080;

const int MAX\_BUFFER\_SIZE = 4096;

void handle\_client(SOCKET client\_socket) {

char buffer[MAX\_BUFFER\_SIZE];

memset(buffer, 0, MAX\_BUFFER\_SIZE);

int bytes\_received = recv(client\_socket, buffer, MAX\_BUFFER\_SIZE, 0);

if (bytes\_received == SOCKET\_ERROR) {

cerr << "Error reading from socket\n";

closesocket(client\_socket);

return;

}

cout << "Received request:\n" << buffer << endl;

istringstream request(buffer);

string method, path;

request >> method >> path;

if (path == "/") {

path = "/index.html";

}

path = path.substr(1);

ifstream file(path);

if (!file.is\_open()) {

string response = "HTTP/1.1 404 Not Found\r\nContent-Length: 0\r\n\r\n";

send(client\_socket, response.c\_str(), response.size(), 0);

}

else {

stringstream file\_content;

file\_content << file.rdbuf();

string response = "HTTP/1.1 200 OK\r\nContent-Length: " + to\_string(file\_content.str().size()) + "\r\n\r\n" + file\_content.str();

send(client\_socket, response.c\_str(), response.size(), 0);

}

closesocket(client\_socket);

}

int main() {

WSADATA wsa\_data;

int result = WSAStartup(MAKEWORD(2, 2), &wsa\_data);

if (result != 0) {

cerr << "WSAStartup failed: " << result << "\n";

return 1;

}

SOCKET server\_socket = socket(AF\_INET, SOCK\_STREAM, 0);

if (server\_socket == INVALID\_SOCKET) {

cerr << "Socket creation failed: " << WSAGetLastError() << "\n";

WSACleanup();

return 1;

}

sockaddr\_in address;

address.sin\_family = AF\_INET;

address.sin\_addr.s\_addr = INADDR\_ANY;

address.sin\_port = htons(PORT);

if (bind(server\_socket, (sockaddr\*)&address, sizeof(address)) == SOCKET\_ERROR) {

cerr << "Bind failed: " << WSAGetLastError() << "\n";

closesocket(server\_socket);

WSACleanup();

return 1;

}

if (listen(server\_socket, 10) == SOCKET\_ERROR) {

cerr << "Listen failed: " << WSAGetLastError() << "\n";

closesocket(server\_socket);

WSACleanup();

return 1;

}

cout << "Server listening on port " << PORT << endl;

vector<thread> threads;

while (true) {

SOCKET client\_socket = accept(server\_socket, nullptr, nullptr);

if (client\_socket == INVALID\_SOCKET) {

cerr << "Accept failed: " << WSAGetLastError() << "\n";

continue;

}

threads.emplace\_back(handle\_client, client\_socket);

}

for (auto& thread : threads) {

if (thread.joinable()) {

thread.join();

}

}

closesocket(server\_socket);

WSACleanup();

return 0;

}

