**Question-4**

**Write a simple web server in C (not C++) which accepts HTTP requests (eg. GET requests for HTML pages) and return**

**responses (eg. HTML pages showing parameters passed with GET request).**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#include <netinet/in.h>

#define PORT 8080

#define BUFFER\_SIZE 1024

void handleRequest(int clientSocket) {

char buffer[BUFFER\_SIZE] = {0};

char response[BUFFER\_SIZE] = {0};

char \*htmlHeader = "HTTP/1.1 200 OK\nContent-Type: text/html\n\n";

char \*htmlBody = "<html><body><h1>Hello, World!</h1></body></html>";

recv(clientSocket, buffer, BUFFER\_SIZE, 0);

printf("Received HTTP Request:\n%s\n", buffer);

strcat(response, htmlHeader);

strcat(response, htmlBody);

send(clientSocket, response, strlen(response), 0);

close(clientSocket);

}

int main() {

int serverSocket, clientSocket;

struct sockaddr\_inaddress;

int addrlen = sizeof(address);

if ((serverSocket = socket(AF\_INET, SOCK\_STREAM, 0)) == 0) {

perror("socket failed");

exit(EXIT\_FAILURE);

}

if (setsockopt(serverSocket, SOL\_SOCKET, SO\_REUSEADDR | SO\_REUSEPORT, &(int){1}, sizeof(int)) < 0) {

perror("setsockopt");

exit(EXIT\_FAILURE);

}

address.sin\_family = AF\_INET;

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

address.sin\_port = htons(PORT);

if (bind(serverSocket, (struct sockaddr \*)&address, sizeof(address)) < 0) {

perror("bind failed");

exit(EXIT\_FAILURE);

}

if (listen(serverSocket, 3) < 0) {

perror("listen");

exit(EXIT\_FAILURE);

}

printf("Server listening on port %d...\n", PORT);

while (1) {

if ((clientSocket = accept(serverSocket, (struct sockaddr \*)&address, (socklen\_t\*)&addrlen)) < 0) {

perror("accept");

exit(EXIT\_FAILURE);

}

handleRequest(clientSocket);

}

return 0;

}

***Instructions***

**Compilation:**

Open a terminal or command prompt and navigate to the directory containing the web\_server.cfile.

Compile the program using **gcc**:

**Bash:**

gcc web\_server.c –o web\_server.c

This command will compile the web\_server.c file and create an executable named web\_server.c

### Execution:

After successful compilation, you can run the executable by typing:

***Bash:***

./web\_server

This command will start the web server, and it will begin listening for incoming HTTP requests on port 8080.

### Testing:

You can test the server by opening a web browser and navigating tohttp://localhost:8080/?test=123. You should see a simple HTML page displaying the query string “test=123”.

### Stopping the Server:

To stop the server, you can press ctrl + c in the terminal where the server is running. This will terminate the server process.

That's it! You have compiled and executed the simple web server program written in C.