ADWAITH SHINOD

S6 CSE ,Rollno:12

IMPLEMENTATION OF FILE SERVER

CODE

Server

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/wait.h>
#include <errno.h>

#define PORT 8080
#define MAX_PENDING_CONNECTIONS 10
#define MAX_BUFFER_SIZE 1024

void handle_client_request(int client_socket) {
    char buffer[MAX_BUFFER_SIZE];
    ssize_t bytes_received;
    ssize_t bytes_sent;
```

```
pid_t pid = getpid();


// Receive filename from client
bytes_received = recv(client_socket, buffer, sizeof(buffer), 0);
if (bytes_received < 0) {
        perror("Error receiving data from client");
        exit(EXIT_FAILURE);
}


// Null terminate the received data
buffer[bytes_received] = '¥0';


// Check if the requested file exists
FILE* file = fopen(buffer, "rb");
if (file != NULL) {
        // If file exists, send the file contents to the client
        while ((bytes_sent = fread(buffer, 1, sizeof(buffer), file)) > 0) {
                if (send(client_socket, buffer, bytes_sent, 0) != bytes_sent) {
                        perror("Error sending file to client");
                        exit(EXIT_FAILURE);
                }
        }
        fclose(file);
} else {
        // If file doesn't exist, send appropriate message
```

```
            const char* message = "File not found.";
            if (send(client_socket, message, strlen(message), 0) < 0) {
                perror("Error sending message to client");
                exit(EXIT_FAILURE);
            }
        }

        // Send PID of server to client
        snprintf(buffer, sizeof(buffer), "%d", pid);
        if (send(client_socket, buffer, strlen(buffer), 0) < 0) {
            perror("Error sending PID to client");
            exit(EXIT_FAILURE);
        }

        // Close the client socket
        close(client_socket);
    }

    int main() {
        int server_socket, client_socket;
        struct sockaddr_in server_addr, client_addr;
        socklen_t client_addr_len = sizeof(client_addr);
        pid_t pid;

        // Create socket
```

```
if ((server_socket = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
        perror("Error creating socket");
        exit(EXIT_FAILURE);
}

// Initialize server address struct
memset(&server_addr, 0, sizeof(server_addr));
server_addr.sin_family = AF_INET;
server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
server_addr.sin_port = htons(PORT);

// Bind socket to address
if (bind(server_socket, (struct sockaddr*)&server_addr, sizeof(server_addr)) < 0) {
        perror("Error binding socket");
        exit(EXIT_FAILURE);
}

// Listen for incoming connections
if (listen(server_socket, MAX_PENDING_CONNECTIONS) < 0) {
        perror("Error listening on socket");
        exit(EXIT_FAILURE);
}

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

```
    // Accept incoming connections and handle requests
    while (1) {
        // Accept connection from client
        if ((client_socket = accept(server_socket, (struct sockaddr*)&client_addr, &client_addr_len)) <
0) {
            perror("Error accepting connection");
            exit(EXIT_FAILURE);
        }

        // Fork a child process to handle client request
        pid = fork();
        if (pid < 0) {
            perror("Error forking child process");
            exit(EXIT_FAILURE);
        } else if (pid == 0) {
            // Child process
            close(server_socket); // Close server socket in child process
            handle_client_request(client_socket);
            exit(EXIT_SUCCESS);
        } else {
            // Parent process
            close(client_socket); // Close client socket in parent process
            waitpid(-1, NULL, WNOHANG); // Reap zombie processes
        }
    }
```

```
    // Close server socket
    close(server_socket);

    return 0;
}
```

Client

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

#define SERVER_IP "127.0.0.1"
#define PORT 8080
#define MAX_BUFFER_SIZE 1024

int main() {
    int client_socket;
    struct sockaddr_in server_addr;
    char filename[MAX_BUFFER_SIZE];
    ssize_t bytes_received;
```

```
// Create socket
if ((client_socket = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
        perror("Error creating socket");
        exit(EXIT_FAILURE);
}

// Initialize server address struct
memset(&server_addr, 0, sizeof(server_addr));
server_addr.sin_family = AF_INET;
server_addr.sin_port = htons(PORT);
server_addr.sin_addr.s_addr = htonl(INADDR_ANY);

// Connect to server
if (connect(client_socket, (struct sockaddr*)&server_addr, sizeof(server_addr)) < 0) {
        perror("Error connecting to server");
        exit(EXIT_FAILURE);
}

// Get filename from user input
printf("Enter filename: ");
fgets(filename, sizeof(filename), stdin);
filename[strcspn(filename, "¥n")] = '¥0'; // Remove newline character

// Send filename to server
if (send(client_socket, filename, strlen(filename), 0) < 0) {
```

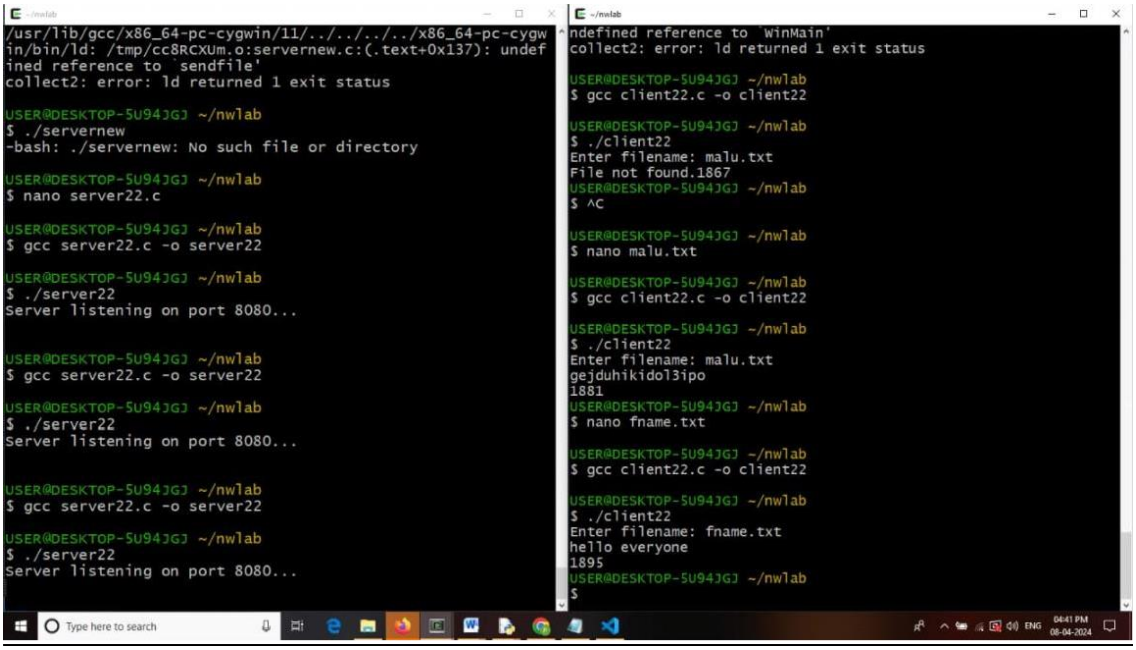
```
        perror("Error sending filename to server");
        exit(EXIT_FAILURE);
    }

    // Receive data from server
    char buffer[MAX_BUFFER_SIZE];
    while ((bytes_received = recv(client_socket, buffer, sizeof(buffer), 0)) > 0) {
        fwrite(buffer, 1, bytes_received, stdout);
    }
    if (bytes_received < 0) {
        perror("Error receiving data from server");
        exit(EXIT_FAILURE);
    }

    // Close socket
    close(client_socket);

    return 0;
}
```

OUTPUT


/usr/lib/gcc/x86_64-pc-cygwin/11/../../../../x86_64-pc-cygwin/bin/ld: /tmp/cc8RCXUm.o:servernew.c:(.text+0x137): undefined reference to `sendfile'
collect2: error: ld returned 1 exit status
USER@DESKTOP-5U94JGJ ~/nwlab
$ ./servernew
-bash: ./servernew: No such file or directory
USER@DESKTOP-5U94JGJ ~/nwlab
$ nano server22.c
USER@DESKTOP-5U94JGJ ~/nwlab
$ gcc server22.c -o server22
USER@DESKTOP-5U94JGJ ~/nwlab
$ ./server22
Server listening on port 8080...
USER@DESKTOP-5U94JGJ ~/nwlab
$ gcc server22.c -o server22
USER@DESKTOP-5U94JGJ ~/nwlab
$ ./server22
Server listening on port 8080...
USER@DESKTOP-5U94JGJ ~/nwlab
$ gcc server22.c -o server22
USER@DESKTOP-5U94JGJ ~/nwlab
$ ./server22
Server listening on port 8080...
ndefined reference to `WinMain'
collect2: error: ld returned 1 exit status
USER@DESKTOP-5U94JGJ ~/nwlab
$ gcc client22.c -o client22
USER@DESKTOP-5U94JGJ ~/nwlab
$ ./client22
Enter filename: malu.txt
File not found.1867
USER@DESKTOP-5U94JGJ ~/nwlab
$ ^C
USER@DESKTOP-5U94JGJ ~/nwlab
$ nano malu.txt
USER@DESKTOP-5U94JGJ ~/nwlab
$ gcc client22.c -o client22
USER@DESKTOP-5U94JGJ ~/nwlab
$ ./client22
Enter filename: malu.txt
gejduhikidol3ipo
1881
USER@DESKTOP-5U94JGJ ~/nwlab
$ nano fname.txt
USER@DESKTOP-5U94JGJ ~/nwlab
$ gcc client22.c -o client22
USER@DESKTOP-5U94JGJ ~/nwlab
$ ./client22
Enter filename: fname.txt
hello everyone
1895
USER@DESKTOP-5U94JGJ ~/nwlab
$
Type here to search
ENG
04:41 PM
08-04-2024