HTTP Persistent connection using TCP socket

AIM:

Establish a TCP connection between the client and the server, using persistent HTTP and download your previous semester mark sheets. The header fields of HTTP should be used.

SYNOPSIS:

The given code implements http persistent connection by establishing a TCP connection between a client and the server. The client sends request to the server as a structure containing the header fields as well as the data (student data -> register number and semester). The server receives this request, prints it in it's stdout, then fetches the particular student's record of the given semester from a file called stud.txt which contains the student records in binary format.
[*I wrote a separate C code (stud.c) to get the student records as input and saving it in a file (stud.txt) instead of hardcoding the values.]*
Once the record is fetched, the server sends the response to client with some headers as well as this fetched record. The client receives the response, and displays it on the screen. This process continues till client enters -1 (Multiple objects are sent through a single connection). After that the client is disconnected and server stops.

CODE:

myheader.h [common header file for both client and server]

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

#define PORT 3542
#define SIZE 300
#define N 20
```

```c
typedef struct{
        int regno;
        int age;
        int sem;
        int marks[5];
        char name[SIZE];
        char gender;
}student;

typedef struct{
        char method[N];
        char path[N];
        char version[N];
        char accept[N];
        char connection[N];
        char useragent[N];
        student stu;
}request;

typedef struct{
        int status;
        char statmsg[N];
        char version[N];
        char connection[N];
        char contype[N];
        char datetime[N];
        student stu;
}response;
```

perserver.c

```c
#include "myheader.h"

int main(int argc, char *argv[]){
        response sr;
        strcpy(sr.version, "http/1.1");
        strcpy(sr.connection, "keep-alive");
        strcpy(sr.contype, "text");

        struct sockaddr_in serveraddr;
        int sockfd = socket(AF_INET, SOCK_STREAM, 0);
        serveraddr.sin_family = AF_INET;
        serveraddr.sin_port = htons(PORT);
        serveraddr.sin_addr.s_addr = htonl(INADDR_ANY);
        if (bind(sockfd, (struct sockaddr *) &serveraddr,
sizeof(serveraddr)) < 0) return 0;
        if (listen(sockfd, 2) < 0) return 0;
        printf("Server listening on port: %d\n", PORT);

        int clisock = accept(sockfd, NULL, NULL);
        if (clisock < 0) return 0;

        while(1){
```

```
                request clr;
                int len = recv(clisock, &clr, sizeof(clr), 0);
                printf("CLIENT'S REQUEST:\n%s%s%s\nConnection: %s\nAccept:
%s\nUser-Agent: %s\nRegno: %d\nRequired sem: %d\n\n",
                                clr.method, clr.path, clr.version,
clr.connection, clr.accept, clr.useragent, clr.stu.regno, clr.stu.sem);
                if (clr.stu.sem == -1) break;

                FILE *inf;
                inf = fopen("stud.txt", "rb");
                student s;
                int found = 0;
                while(fread(&s, sizeof(student), 1, inf) == 1){
                        if (s.regno == clr.stu.regno && s.sem ==
clr.stu.sem) {
                                found = 1;
                                sr.stu = s;
                                strcpy(sr.statmsg, "OK");
                                sr.status = 200;
                                break;
                        }
                }   if (!found){
                        sr.stu.regno = -1;
                        sr.stu.sem = -1;
                        strcpy(sr.statmsg, "Not found");
                        sr.status = 404;
                }
                fclose(inf);
                send(clisock, &sr, sizeof(sr), 0);
        }
        close(sockfd);
        return 0;
}
```

perclient.c

```
include "myheader.h"

int main(int argc, char *argv[]){

        request cr;
        strcpy(cr.method, "GET");
        strcpy(cr.path, "/");
        strcpy(cr.version, "http/1.1");
        strcpy(cr.connection, "keep-alive");
        strcpy(cr.accept, "text");

        struct sockaddr_in cliaddr;
        int sockfd = socket(AF_INET, SOCK_STREAM, 0);
        cliaddr.sin_family = AF_INET;
        cliaddr.sin_port = htons(PORT);
        cliaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
```

```
        if (connect(sockfd, (struct sockaddr *) &cliaddr, sizeof(cliaddr))
< 0) return 0;
        printf("Connection made with the server.\n");


        printf("Enter useragent: ");
        scanf("%s", cr.useragent);
        printf("Enter RegNo of the student: ");
        scanf("%d", &cr.stu.regno);

        while(1){

                printf("Enter semester [-1 to exit]: ");
                scanf("%d", &cr.stu.sem);

                send(sockfd, &cr, sizeof(cr), 0);

                if (cr.stu.sem == -1) break;

                response sr;
                int len = recv(sockfd, &sr, sizeof(sr), 0);
                if (len > 0){
                        printf("SERVER'S RESPONSE: \n%s\n%d
%s\nConnection: %s\nContent-Type: %s\n\n", sr.version, sr.status,
sr.statmsg, sr.connection, sr.contype);
                        if (sr.status == 404){
                                printf("Marksheet not found.\n");
                        }
                        else{
                                printf("           MARKSHEET\n");
                                printf("|----------------------------
|\n");
                                printf("|REGNO:  NAME:\t AGE: GENDER:
|\n");
                                printf("|%d    %s\t %d    %c      |\n",
sr.stu.regno, sr.stu.name, sr.stu.age, sr.stu.gender);
                                printf("|----------------------------
|\n");
                                printf("|SEM: %d\tMARKS: ", sr.stu.sem);
                                for (int j = 0; j < 5; j++){
                                        printf("%d ", sr.stu.marks[j]);
                                }
                                printf("|\n|----------------------------
|\n");
                        }
                }
        }
        close(sockfd);
        printf("Disconnected from the server.\n");

        return 0;
}
```

OUTPUT:

./server

```
Server listening on port: 3542
CLIENT'S REQUEST:
GET/http/1.1
Connection: keep-alive
Accept: text
User-Agent: Chrome
Regno: 3001
Required sem: 1

CLIENT'S REQUEST:
GET/http/1.1
Connection: keep-alive
Accept: text
User-Agent: Chrome
Regno: 3001
Required sem: 2

CLIENT'S REQUEST:
GET/http/1.1
Connection: keep-alive
Accept: text
User-Agent: Chrome
Regno: 3001
Required sem: 3

CLIENT'S REQUEST:
GET/http/1.1
Connection: keep-alive
Accept: text
User-Agent: Chrome
Regno: 3001
Required sem: -1
```

./client

```
Enter useragent: Chrome
Enter RegNo of the student: 3001
Enter semester [-1 to exit]: 1
SERVER'S RESPONSE:
http/1.1
200 OK
Connection: keep-alive
Content-Type: text

           MARKSHEET
|-----------------------------|
|REGNO:  NAME:   AGE: GENDER: |
|3001    Jane    19    M      |
|-----------------------------|
|SEM: 1 MARKS: 98 98 98 98 98 |
|-----------------------------|

Enter semester [-1 to exit]: 2
SERVER'S RESPONSE:
http/1.1
200 OK
Connection: keep-alive
Content-Type: text

           MARKSHEET
|-----------------------------|
|REGNO:  NAME:   AGE: GENDER: |
|3001    Jane    19    M      |
|-----------------------------|
|SEM: 2 MARKS: 99 99 99 99 99 |
|-----------------------------|

Enter semester [-1 to exit]: 3
SERVER'S RESPONSE:
http/1.1
200 OK
Connection: keep-alive
Content-Type: text

           MARKSHEET
|-----------------------------|
|REGNO:  NAME:   AGE: GENDER: |
|3001    Jane    19    M      |
|-----------------------------|
|SEM: 3 MARKS: 100 100 100 100 100 |
|-----------------------------|

Enter semester [-1 to exit]: -1
Disconnected from the server.
```

EXTRA: stud.c code [The program created to accept student records]

```
#include<stdio.h>

#define SIZE 300
#define SEM 3
#define N 2

typedef struct{
        int regno;
        int age;
        int sem;
        int marks[5];
        char name[SIZE];
        char gender;
}student;

int main(){
        student s[N];
        FILE *of;
        of = fopen("stud.txt", "w");
        for (int i = 0; i < N; i++){
                printf("Enter Regno, Name, Age, Gender\n");
                scanf("%d", &s[i].regno);
                getchar();
                scanf("%[^\n]%*c", s[i].name);
                scanf("%d", &s[i].age);
                getchar();
                scanf("%c", &s[i].gender);
                for (int k = 0; k < SEM; k++){
                        s[i].sem = k + 1;
                        printf("Enter marks for sem %d, 5 subjects: ", k +
1);
                        for (int j = 0; j < 5; j++)
                                scanf("%d", &s[i].marks[j]);
                        fwrite (&s[i], sizeof(student), 1, of);
                }
        }
        fclose(of);
        student stu;
        FILE *inf;
        inf = fopen("stud.txt", "r");
          while(fread(&stu, sizeof(student), 1, inf)){
                printf("Regno: %d\nName: %s\nAge: %d\nGender: %c\nSem:
%d\nMarks: ",
                                stu.regno, stu.name, stu.age, stu.gender,
stu.sem);
                for (int j = 0; j < 5; j++)
                        printf("%d ", stu.marks[j]);
                printf("\n");
        }
        fclose(inf);
        return 0;
}
```