Name : Rajkumar B L
Reg.No : 2047120
Course : MCS 172 – Unix Assignment

Question 02: Socket Inter-Process Communication

A socket is an IPC method of point to point two-way communication between two processes: connecting two nodes on a network. To implement this, I created an application where two separate processes: server and client, communicate with each other through a socket IPC method. Here, the communication context is that a client(Rajkumar) establishes a connection with the server(Pizza-Hut) and orders pizza for his family members to celebrate Christmas.

- Creating and naming socket: socket(), bind()
- Establish Connection: listen(), accept(), connect()
- Communication: read(), write(), send(), recv()

1.Server.c and client.c are compiled in two separate terminals

```
Ubuntu 20.04 LTS
kumarraj@kumarraj:~/MCS_172/Assignment$ tty
/dev/tty1
kumarraj@kumarraj:~/MCS_172/Assignment$ gcc client.c -o client
kumarraj@kumarraj:~/MCS_172/Assignment$ ./client
```

```
Ubuntu 20.04 LTS
kumarraj@kumarraj:~/MCS_172/Assignment$ tty
/dev/tty2
kumarraj@kumarraj:~/MCS_172/Assignment$ gcc server.c -o server
kumarraj@kumarraj:~/MCS_172/Assignment$ ./server
```

2.Connection is established, the server sends the menu to the client

```
Ubuntu 20.04 LTS
kumarraj@kumarraj:~/MCS_172/Assignment$ tty
/dev/tty1
kumarraj@kumarraj:~/MCS_172/Assignment$ gcc client.c -o client
kumarraj@kumarraj:~/MCS_172/Assignment$ ./client

Connection Established ...

============================
       Pizza-Hut Menu
============================
1.Hawaiin      ---> Rs.150
2.Greek-Veg    ---> Rs.200
3.Pepperoni    ---> Rs.250
4.Triple Crown ---> Rs.300
5.BBQchicken   ---> Rs.500
============================
Enter the choice to add to the cart:
```

```
Ubuntu 20.04 LTS
kumarraj@kumarraj:~/MCS_172/Assignment$ tty
/dev/tty2
kumarraj@kumarraj:~/MCS_172/Assignment$ gcc server.c -o server
kumarraj@kumarraj:~/MCS_172/Assignment$ ./server
Server Started
Waiting For a Client...

Connection Established with client...
Client received the menu...
```

3.Client receives the menu and replies to the server with the pizza choice that he wants to add to the cart. The server calculates the pizza's cost based on the client's preference and responds with the total cart value. And asks if the client wants to add more pizzas to the cart?

Ubuntu 20.04 LTS

```
kumarraj@kumarraj:~/MCS_172/Assignment$ tty
/dev/tty1
kumarraj@kumarraj:~/MCS_172/Assignment$ gcc client.c -o client
kumarraj@kumarraj:~/MCS_172/Assignment$ ./client

Connection Established ...

============================
       Pizza-Hut Menu
============================
1.Hawaiin      ---> Rs.150
2.Greek-Veg    ---> Rs.200
3.Pepperoni    ---> Rs.250
4.Triple Crown ---> Rs.300
5.BBQchicken   ---> Rs.500
============================
Enter the choice to add to the cart:
1
Cart total = 150
Would you like to add more pizza's to the cart (y/n):
```

Ubuntu 20.04 LTS

```
kumarraj@kumarraj:~/MCS_172/Assignment$ tty
/dev/tty2
kumarraj@kumarraj:~/MCS_172/Assignment$ gcc server.c -o server
kumarraj@kumarraj:~/MCS_172/Assignment$ ./server
Server Started
Waiting For a Client...

Connection Established with client...
Client received the menu...
Client responded with their choice...
Total Amount in the cart is 150
```

4.Server sends the menu again as the client responded with yes for adding more items to the cart. We loop through step 3 again.

Ubuntu 20.04 LTS

```
kumarraj@kumarraj:~/MCS_172/Assignment$ ./client

Connection Established ...

============================
       Pizza-Hut Menu
============================
1.Hawaiin      ---> Rs.150
2.Greek-Veg    ---> Rs.200
3.Pepperoni    ---> Rs.250
4.Triple Crown ---> Rs.300
5.BBQchicken   ---> Rs.500
============================
Enter the choice to add to the cart:
1
Cart total = 150

Would you like to add more pizza's to the cart (y/n):
y

============================
       Pizza-Hut Menu
============================
1.Hawaiin      ---> Rs.150
2.Greek-Veg    ---> Rs.200
3.Pepperoni    ---> Rs.250
4.Triple Crown ---> Rs.300
5.BBQchicken   ---> Rs.500
============================
Enter the choice to add to the cart:
5
Cart total = 650

Would you like to add more pizza's to the cart (y/n):
```

Ubuntu 20.04 LTS

```
kumarraj@kumarraj:~/MCS_172/Assignment$ ./server
Server Started
Waiting For a Client...

Connection Established with client...
Client received the menu...
Client responded with their choice...
Total Amount in the cart is 150
Client responded with their choice...

Connection Established with client...
Client received the menu...
Client responded with their choice...
Total Amount in the cart is 650
```

5.After adding desired pizzas to the cart, the client now responds to the server as no. The server accepts the client's response and replies with an endnote stating the client to be ready with the cart total at the time of delivery. With this, the connection comes to an end.

```
Ubuntu 20.04 LTS
============================
       Pizza-Hut Menu
============================
1.Hawaiin      ---> Rs.150
2.Greek-Veg    ---> Rs.200
3.Pepperoni    ---> Rs.250
4.Triple Crown ---> Rs.300
5.BBQchicken   ---> Rs.500
============================
Enter the choice to add to the cart:
5
Cart total = 650

Would you like to add more pizza's to the cart (y/n):
n

Your order has been placed.
On delivery pay a toatl of RS.650
Thank you for visiting us today.
Have a happy day!


kumarraj@kumarraj:~/MCS_172/Assignment$
```

```
Ubuntu 20.04 LTS
kumarraj@kumarraj:~/MCS_172/Assignment$ ./server
Server Started
Waiting For a Client...

Connection Established with client...
Client received the menu...
Client responded with their choice...
Total Amount in the cart is 150
Client responded with their choice...

Connection Established with client...
Client received the menu...
Client responded with their choice...
Total Amount in the cart is 650
Client responded with their choice...

Connection ended ...
kumarraj@kumarraj:~/MCS_172/Assignment$
```

Code for Client.c and Server.c is attached below: -

```c
/***********************************
* MCS 172  - Assignment
* Filename : server.c
* Author   : Rajkumar B L
* Reg.No   : 2047120
***********************************/
#include <unistd.h>
#include <stdio.h>
#include <sys/socket.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <string.h>
#include <stdbool.h>
#define PORT 8080
int cart();

int main()
{
    int server_fd, new_socket, valread;
    char info_to_customer[250];
    struct sockaddr_in address;
    int opt = 1;
    int addrlen = sizeof(address);
    char buffer[1024] = {0};
    bool s_conct = true;

    char menu[250] = "\n============================\n       Pizza-Hut Menu\n============================\n1.Hawaiin      ---> Rs.150\n2.Greek-Veg    ---> Rs.200\n3.Pepperoni    ---> Rs.250\n4.Triple Crown ---> Rs.300\n5.BBQchicken   ---> Rs.500\n============================";
    int cart_amount = 0;

    // Creating socket file descriptor
    if ((server_fd = socket(AF_INET, SOCK_STREAM, 0)) == 0)
    {
        perror("socket failed");
        exit(EXIT_FAILURE);
    }

    // Forcefully attaching socket to the port
    if (setsockopt(server_fd, SOL_SOCKET, SO_REUSEADDR | SO_REUSEPORT, &opt, sizeof(opt)))
    {
        perror("setsockopt");
        exit(EXIT_FAILURE);
    }
    address.sin_family = AF_INET;
    address.sin_addr.s_addr = INADDR_ANY;
    address.sin_port = htons(PORT);

    // Forcefully attaching socket to the port
    if (bind(server_fd, (struct sockaddr *)&address, sizeof(address)) < 0)
    {
        perror("bind failed");
        exit(EXIT_FAILURE);
    }
    if (listen(server_fd, 3) < 0)
    {
        perror("listen");
        exit(EXIT_FAILURE);
```

```
    }
    if ((new_socket = accept(server_fd, (struct sockaddr *)&address,(socklen_t *)&addrlen)) < 0)
    {
        perror("accept");
        exit(EXIT_FAILURE);
    }

    printf("Server Started\n");
    printf("Waiting For a Client...\n");

    while (s_conct)
    {
        char ch;
        //Sending Menu to the client
        send(new_socket, menu, strlen(menu), 0); //Send 1 Menu

        valread = read(new_socket, buffer, 1024); // Read 1 if the menu is received
        printf("\nConnection Established with client... \n");
        printf("%s\n", buffer);
        memset(buffer, 0, strlen(buffer));

        //Ask the user to enter the choice
        memset(info_to_customer, 0, strlen(info_to_customer));
        strcpy(info_to_customer, "Enter the choice to add to the cart:");
        send(new_socket, info_to_customer, strlen(info_to_customer), 0); //Send 2 - Enter the choice
        memset(info_to_customer, 0, strlen(info_to_customer));

        //Read the option selected by the client
        valread = read(new_socket, buffer, 1024); // Read 2
        printf("Client responded with their choice...\n");

        char ct_amt_st[5];
        cart_amount = cart_amount + cart(buffer);
        printf("Total Amount in the cart is %d\n", cart_amount);
        memset(buffer, 0, sizeof(buffer));
        //Sending current cart total amount to client
        strcpy(info_to_customer, "Cart total = ");
        snprintf(ct_amt_st, 5, "%d", cart_amount);
        strcat(info_to_customer, ct_amt_st);
        send(new_socket, info_to_customer, strlen(info_to_customer), 0); //Send 3 - Sending total cart value
        memset(info_to_customer, 0, sizeof(info_to_customer));

        //Asking customer if they want to add more pizzas to the cart
        strcpy(info_to_customer, "\nWould you like to add more pizza's to the cart (y/n):");
        send(new_socket, info_to_customer, strlen(info_to_customer), 0); //Send 4 - Asking to add more item
        memset(info_to_customer, 0, sizeof(info_to_customer));

        //Read the option selected by the client for more pizza
        valread = read(new_socket, buffer, 1024); // Read 3
        printf("Client responded with their choice...\n");
        //printf("%s", buffer);
        ch=buffer[0];
        if (buffer[0] == 'y')
        {
            s_conct = true;
```

```
            memset(buffer, 0, strlen(buffer));
        }
        else
        {
            memset(info_to_customer, 0, sizeof(info_to_customer));
            strcpy(info_to_customer, "\nYour order has been placed.\nOn delivery pay a toatl of RS.");
            snprintf(ct_amt_st, 5, "%d", cart_amount);
            strcat(info_to_customer, ct_amt_st);
            send(new_socket, info_to_customer, strlen(info_to_customer), 0); //Send end 1 - Sending total cart value
            memset(info_to_customer, 0, sizeof(info_to_customer));

            s_conct = false;
            strcpy(info_to_customer, "\nThank you for visiting us today.\nHave a happy day!");
            send(new_socket, info_to_customer, strlen(info_to_customer), 0); //Send end 2 - sending end note
            memset(buffer, 0, strlen(buffer));
            printf("\nConnection ended ... \n");
        }
    }

    return 0;
}

int cart(char choice[])
{
    int cost=0;
    switch (choice[0])
    {
        case '1':
            cost=150;
            break;
        case '2':
            cost = 200;
            break;
        case '3':
            cost = 250;
            break;
        case '4':
            cost = 300;
            break;
        case '5':
            cost = 500;
            break;
    }
    return cost;
}
```

```c
/***********************************
* MCS 172  - Assignment
* Filename : client.c
* Author   : Rajkumar B L
* Reg.No   : 2047120
***********************************/

#include <stdio.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <string.h>
#include <stdbool.h>
#define PORT 8080

int main()
{
    int sock = 0, valread;
    char info_to_server[100];
    struct sockaddr_in serv_addr;
    char info_buff[1024] = {0};

    bool c_conct=true;

    if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0)
    {
        printf("\n Socket creation error \n");
        return -1;
    }

    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(PORT);

    // Convert IPv4 and IPv6 addresses from text to binary form
    if (inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr) <= 0)
    {
        printf("\nInvalid address/ Address not supported \n");
        return -1;
    }

    if (connect(sock, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0)
    {
        printf("\nConnection Failed \n");
        return -1;
    }
    else
    {
        printf("\nConnection Established ... \n");
    }

    while (c_conct)
    {

        char ch;
        //Menu Received
        valread = read(sock, info_buff, 1024); //Read 1
        printf("%s\n", info_buff);
        memset(info_buff, 0, sizeof(info_buff));

        memset(info_to_server, 0, sizeof(info_to_server));
```

```
        strcpy(info_to_server, "Client received the menu...");
        send(sock, info_to_server, strlen(info_to_server), 0); // Send 1
        memset(info_to_server, 0, sizeof(info_to_server));

        //Reading form server - Asking to enter the choice
        valread = read(sock, info_buff, 1024); //Read 2
        printf("%s\n", info_buff);
        memset(info_buff, 0, sizeof(info_buff));

        //Sending the Choice to the server
        scanf("%s",info_to_server);
        send(sock, info_to_server, strlen(info_to_server), 0); // Send 2 - sending client choice
        memset(info_to_server, 0, sizeof(info_to_server));

        //Reading the total cart value
        valread = read(sock, info_buff, 1024); //Read 3
        printf("%s\n", info_buff);
        memset(info_buff, 0, sizeof(info_buff));

        //Reading - if to add more pizza to cart
        valread = read(sock, info_buff, 1024); //Read 4
        printf("%s\n", info_buff);
        memset(info_buff, 0, sizeof(info_buff));

        //Sending the more pizza Choice to the server
        scanf("%s", info_to_server);
        send(sock, info_to_server, strlen(info_to_server), 0); // Send 3 - sending client choice
        //printf("%s", info_to_server);
        ch = info_to_server[0];
        if (info_to_server[0] == 'y')
        {
            c_conct = true;
            memset(info_to_server, 0, sizeof(info_to_server));
        }
        else
        {
            c_conct = false;
            //Reading - end note
            valread = read(sock, info_buff, 1024); //Read end 1
            printf("%s\n", info_buff);
            memset(info_buff, 0, sizeof(info_buff));

            valread = read(sock, info_buff, 1024); //Read end 2
            printf("%s\n\n", info_buff);
            memset(info_buff, 0, sizeof(info_buff));
            memset(info_to_server, 0, sizeof(info_to_server));
        }
    }

    return 0;
}
```