TASK 12

TCP Socket Programming in C
Develop a simple client-server application using TCP sockets in C:
The server listens on a specified port and accepts a client connection.
Once connected, the server sends a message "Welcome to the Server!" to the client.
The client receives the message and prints it to the console.

CLIENT CODE

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

#define PORT 8080

int main() {
    int sock = 0;
    struct sockaddr_in serv_addr;
    char buffer[1024] = {0};

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

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

    // Convert IP address to binary
    if (inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr) <= 0) {
        printf("Invalid address/ Address not supported\n");
        return -1;
    }

    // Connect to server
    if (connect(sock, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0) {
        printf("Connection Failed\n");
        return -1;
    }

    // Read server message
    read(sock, buffer, sizeof(buffer));
    printf("Server says: %s\n", buffer);

    close(sock);
    return 0;
}
```

FILES

client.c

SERVER CODE

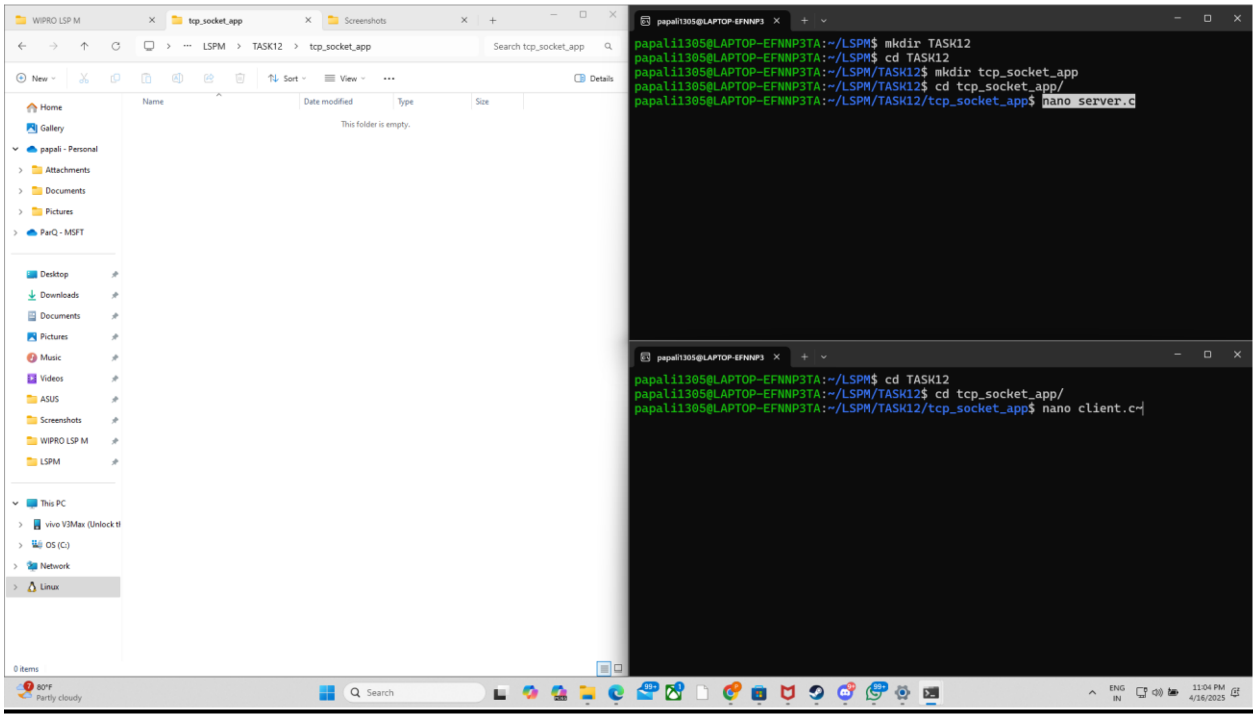
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <netinet/in.h>
#include <sys/socket.h>

#define PORT 8080

int main() {
    int server_fd, new_socket;
    struct sockaddr_in address;
    int opt = 1;
    int addrlen = sizeof(address);
    char *message = "Welcome to the Server!";

    // Create socket file descriptor
    if ((server_fd = socket(AF_INET, SOCK_STREAM, 0)) == 0) {
        perror("socket failed");
        exit(EXIT_FAILURE);
    }
    // Attach socket to the port 8080
    setsockopt(server_fd, SOL_SOCKET, SO_REUSEADDR | SO_REUSEPORT, &opt,
sizeof(opt));

    address.sin_family = AF_INET;
    address.sin_addr.s_addr = INADDR_ANY; // listen on all interfaces
    address.sin_port = htons(PORT);

    // Bind socket
    if (bind(server_fd, (struct sockaddr *)&address, sizeof(address)) < 0) {
        perror("bind failed");
        exit(EXIT_FAILURE);
    }

    // Listen
    if (listen(server_fd, 3) < 0) {
        perror("listen failed");
        exit(EXIT_FAILURE);
    }

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

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

    // Send message to client
    send(new_socket, message, strlen(message), 0);
    printf("Message sent to client.\n");

    close(new_socket);
    close(server_fd);

    return 0;
}
```

FILES

server.c

OUTPUT & SCREENSHOTS

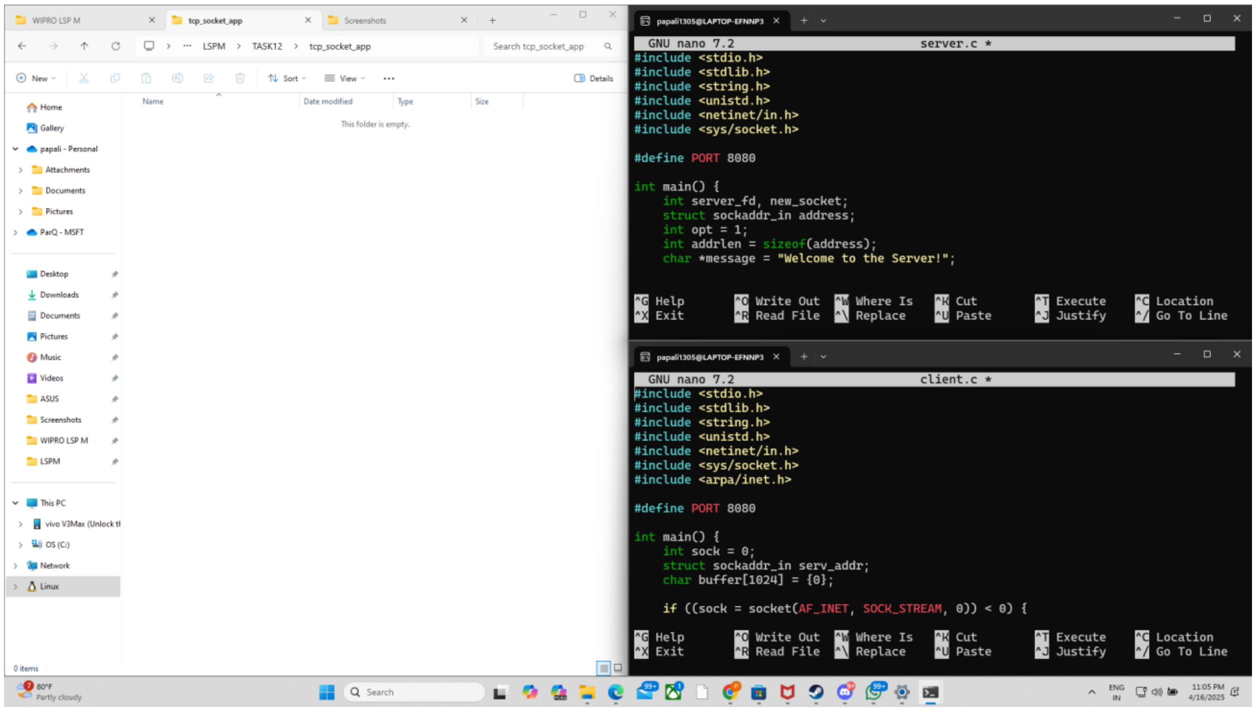
01).



```
papali1305@LAPTOP-EFNNP3TA:~/LSPM$ mkdir TASK12
papali1305@LAPTOP-EFNNP3TA:~/LSPM$ cd TASK12
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12$ mkdir tcp_socket_app
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12$ cd tcp_socket_app/
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12/tcp_socket_app$ nano server.c
```

```
papali1305@LAPTOP-EFNNP3TA:~/LSPM$ cd TASK12
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12$ cd tcp_socket_app/
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12/tcp_socket_app$ nano client.c~
```

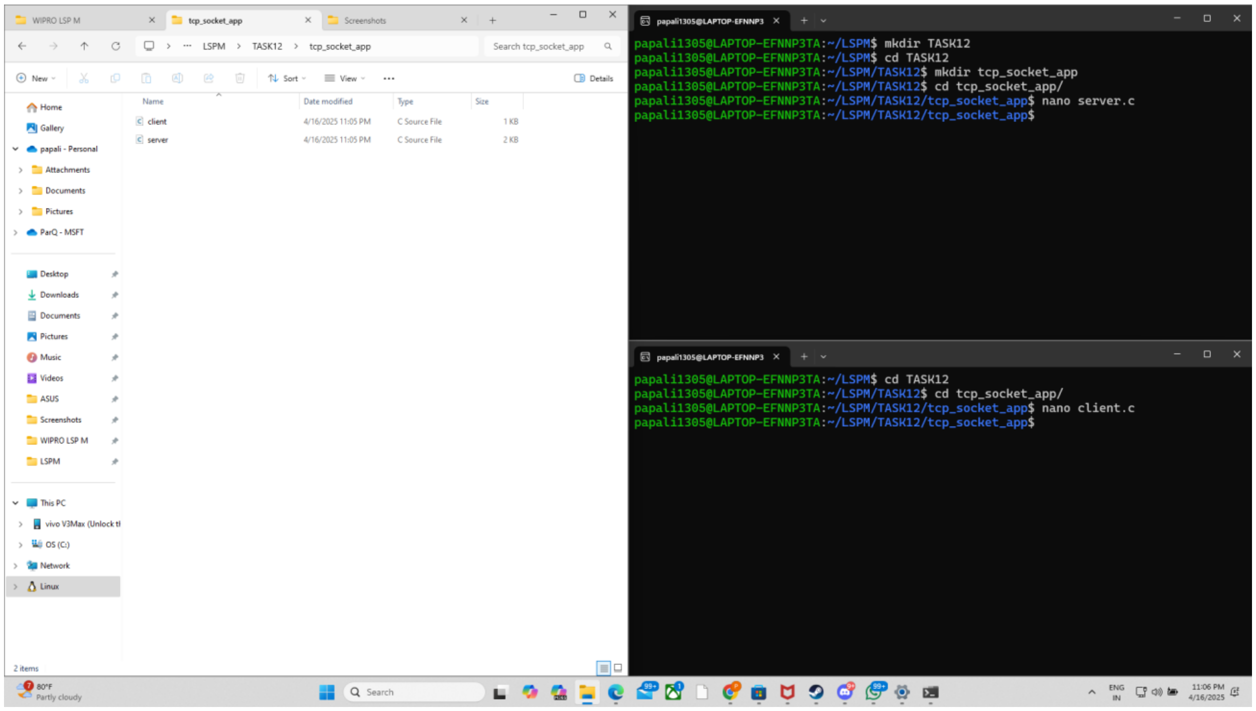
02).



```
GNU nano 7.2                          server.c *
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <netinet/in.h>
#include <sys/socket.h>

#define PORT 8080

int main() {
    int server_fd, new_socket;
    struct sockaddr_in address;
    int opt = 1;
    int addrlen = sizeof(address);
    char *message = "Welcome to the Server!";

^G Help       ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line
```

```
GNU nano 7.2                          client.c *
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <netinet/in.h>
#include <sys/socket.h>
#include <arpa/inet.h>

#define PORT 8080

int main() {
    int sock = 0;
    struct sockaddr_in serv_addr;
    char buffer[1024] = {0};

    if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0) {

^G Help       ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line
```

03).

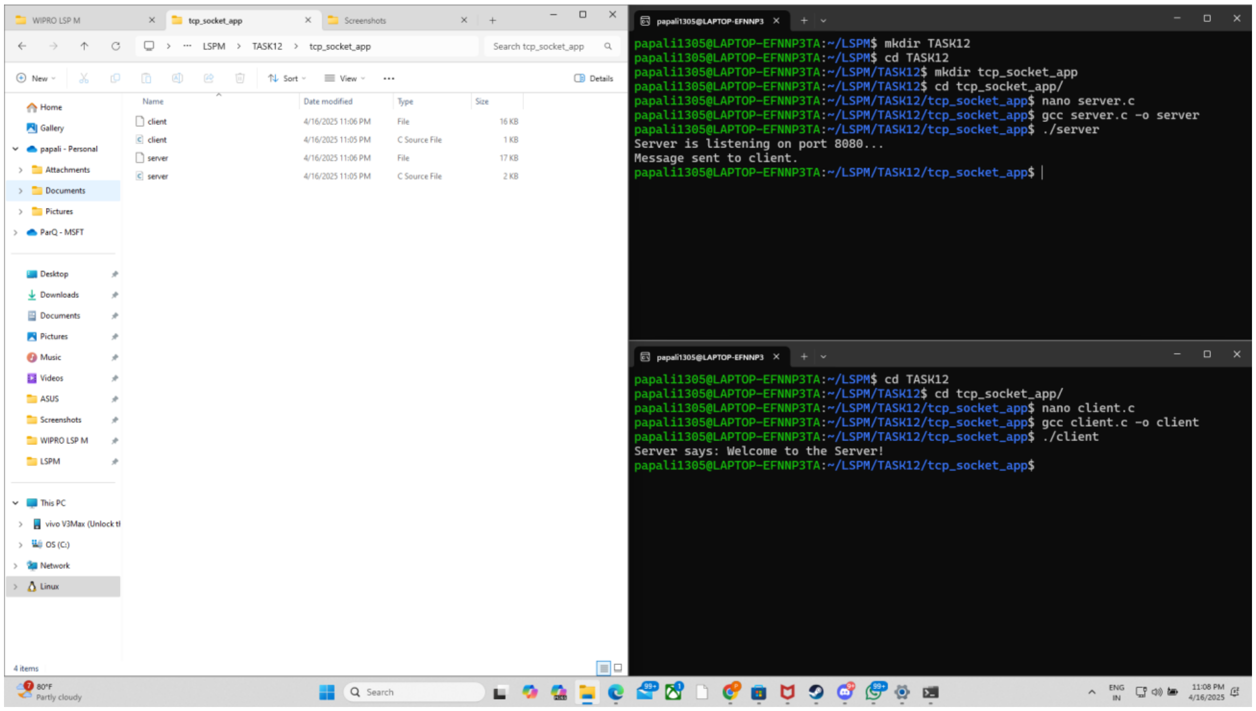


Name	Date modified	Type	Size
client	4/16/2025 11:05 PM	C Source File	1 KB
server	4/16/2025 11:05 PM	C Source File	2 KB

```
papali1305@LAPTOP-EFNNP3TA:~/LSPM$ mkdir TASK12
papali1305@LAPTOP-EFNNP3TA:~/LSPM$ cd TASK12
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12$ mkdir tcp_socket_app
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12$ cd tcp_socket_app/
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12/tcp_socket_app$ nano server.c
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12/tcp_socket_app$
```

```
papali1305@LAPTOP-EFNNP3TA:~/LSPM$ cd TASK12
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12$ cd tcp_socket_app/
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12/tcp_socket_app$ nano client.c
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12/tcp_socket_app$
```

04).



Name	Date modified	Type	Size
client	4/16/2025 11:06 PM	File	16 KB
client	4/16/2025 11:05 PM	C Source File	1 KB
server	4/16/2025 11:06 PM	File	17 KB
server	4/16/2025 11:05 PM	C Source File	2 KB

```
papali1305@LAPTOP-EFNNP3TA:~/LSPM$ mkdir TASK12
papali1305@LAPTOP-EFNNP3TA:~/LSPM$ cd TASK12
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12$ mkdir tcp_socket_app
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12$ cd tcp_socket_app/
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12/tcp_socket_app$ nano server.c
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12/tcp_socket_app$ gcc server.c -o server
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12/tcp_socket_app$ ./server
Server is listening on port 8080...
Message sent to client.
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12/tcp_socket_app$
```

```
papali1305@LAPTOP-EFNNP3TA:~/LSPM$ cd TASK12
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12$ cd tcp_socket_app/
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12/tcp_socket_app$ nano client.c
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12/tcp_socket_app$ gcc client.c -o client
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12/tcp_socket_app$ ./client
Server says: Welcome to the Server!
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK12/tcp_socket_app$
```