TASK 13

UDP Socket Programming in C
Create a basic UDP program in C:
Implement both client and server functionalities within the program.
The client sends a message "Hello from Client" to the server.
The server receives the message, prints it, and responds with "Hello from Server."
The client receives the server's response and displays it.

UDP CLIENT CODE

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <arpa/inet.h>

#define PORT 8080
#define MAXLINE 1024

int main() {
    int sockfd;
    char buffer[MAXLINE];
    char *hello = "Hello from Client";
    struct sockaddr_in servaddr;

    // Create socket
    if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
        perror("Socket creation failed");
        exit(EXIT_FAILURE);
    }

    memset(&servaddr, 0, sizeof(servaddr));

    servaddr.sin_family = AF_INET;
    servaddr.sin_port = htons(PORT);
    servaddr.sin_addr.s_addr = INADDR_ANY;

    // Send message to server
    sendto(sockfd, (const char *)hello, strlen(hello), MSG_CONFIRM,
           (const struct sockaddr *)&servaddr, sizeof(servaddr));
```

```
    printf("Message sent to server.\n");

    int n, len;
    len = sizeof(servaddr);

    // Receive response
    n = recvfrom(sockfd, (char *)buffer, MAXLINE, MSG_WAITALL,
                 (struct sockaddr *)&servaddr, &len);
    buffer[n] = '\0';
    printf("Server says: %s\n", buffer);

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

FILE

udp_client.c

UDP SERVER CODE

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>

#define PORT 8080
#define MAXLINE 1024

int main() {
    int sockfd;
    char buffer[MAXLINE];
    char *hello = "Hello from Server";
    struct sockaddr_in servaddr, cliaddr;

    // Create socket
    if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
        perror("Socket creation failed");
```

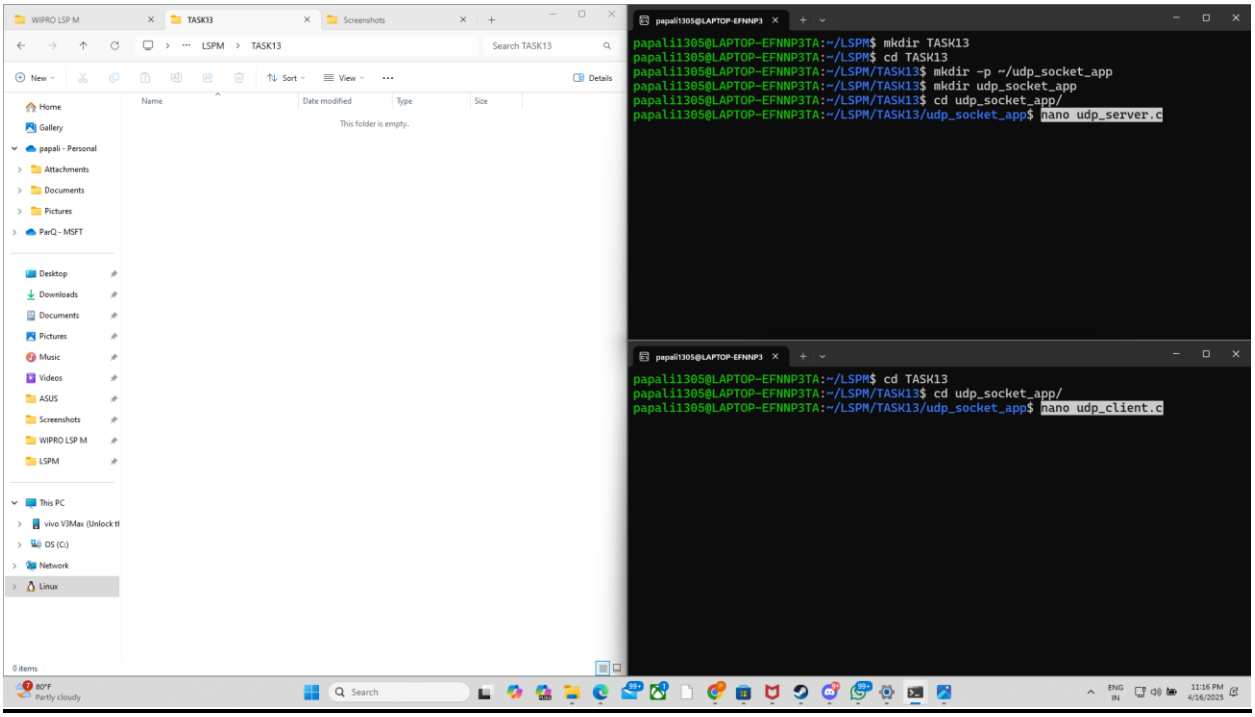
```
        exit(EXIT_FAILURE);
    }

    // Fill server info
    memset(&servaddr, 0, sizeof(servaddr));
    memset(&cliaddr, 0, sizeof(cliaddr));

    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = INADDR_ANY;
    servaddr.sin_port = htons(PORT);

    // Bind
    if (bind(sockfd, (const struct sockaddr *)&servaddr, sizeof(servaddr)) < 0) {
        perror("Bind failed");
        exit(EXIT_FAILURE);
    }

    int len, n;
    len = sizeof(cliaddr);

    // Receive message
    n = recvfrom(sockfd, (char *)buffer, MAXLINE, MSG_WAITALL,
                 (struct sockaddr *)&cliaddr, &len);
    buffer[n] = '\0';
    printf("Client says: %s\n", buffer);

    // Send reply
    sendto(sockfd, (const char *)hello, strlen(hello), MSG_CONFIRM,
           (const struct sockaddr *)&cliaddr, len);
    printf("Hello message sent.\n");

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

FILE

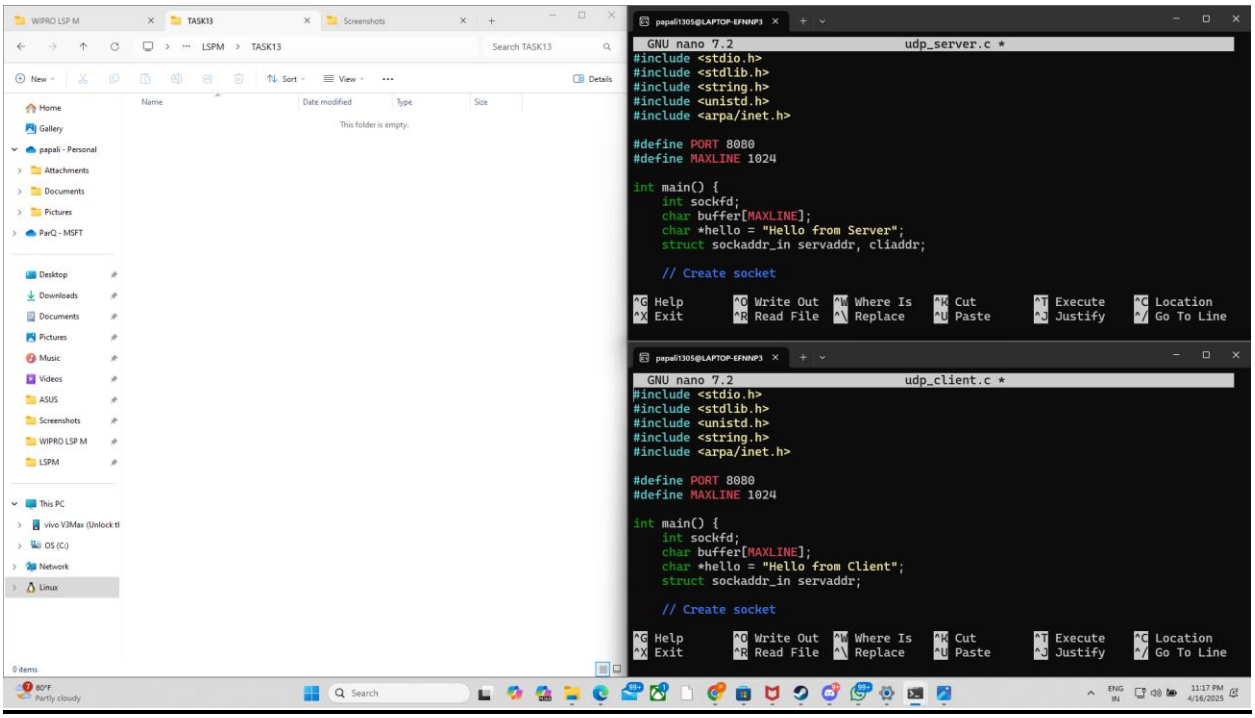
udp_server.c

OUTPUT & SCREENSHOT

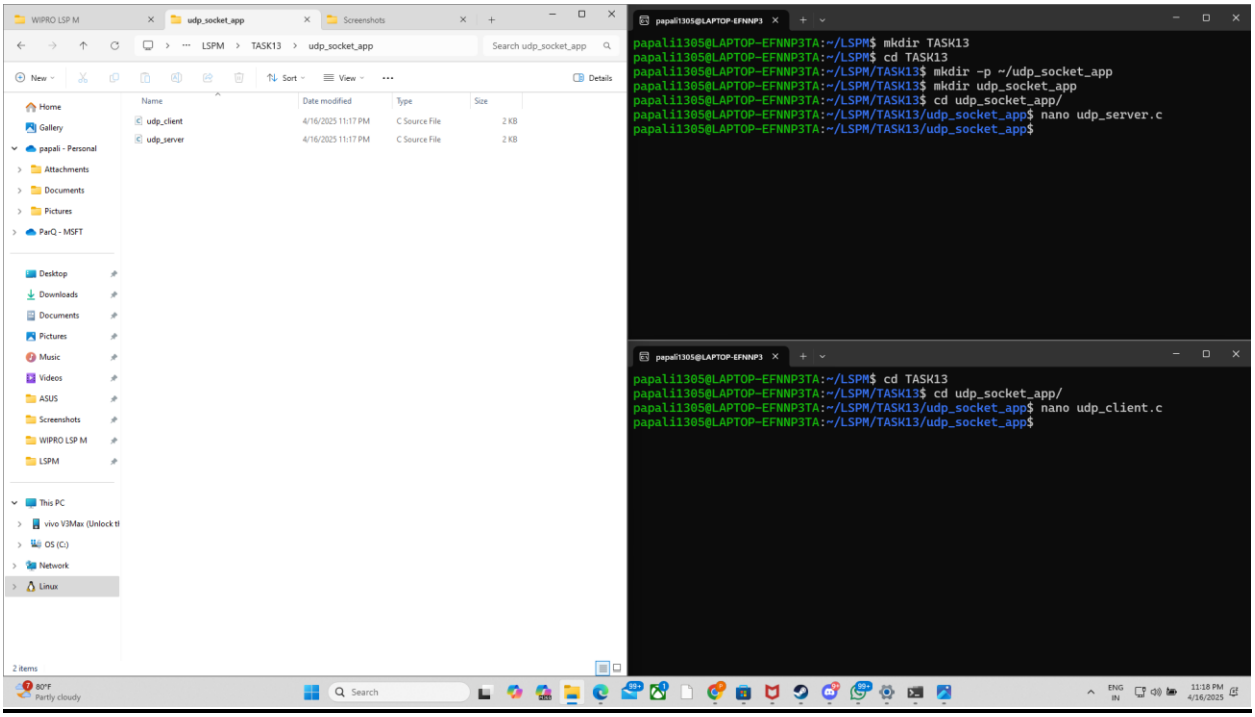
01).



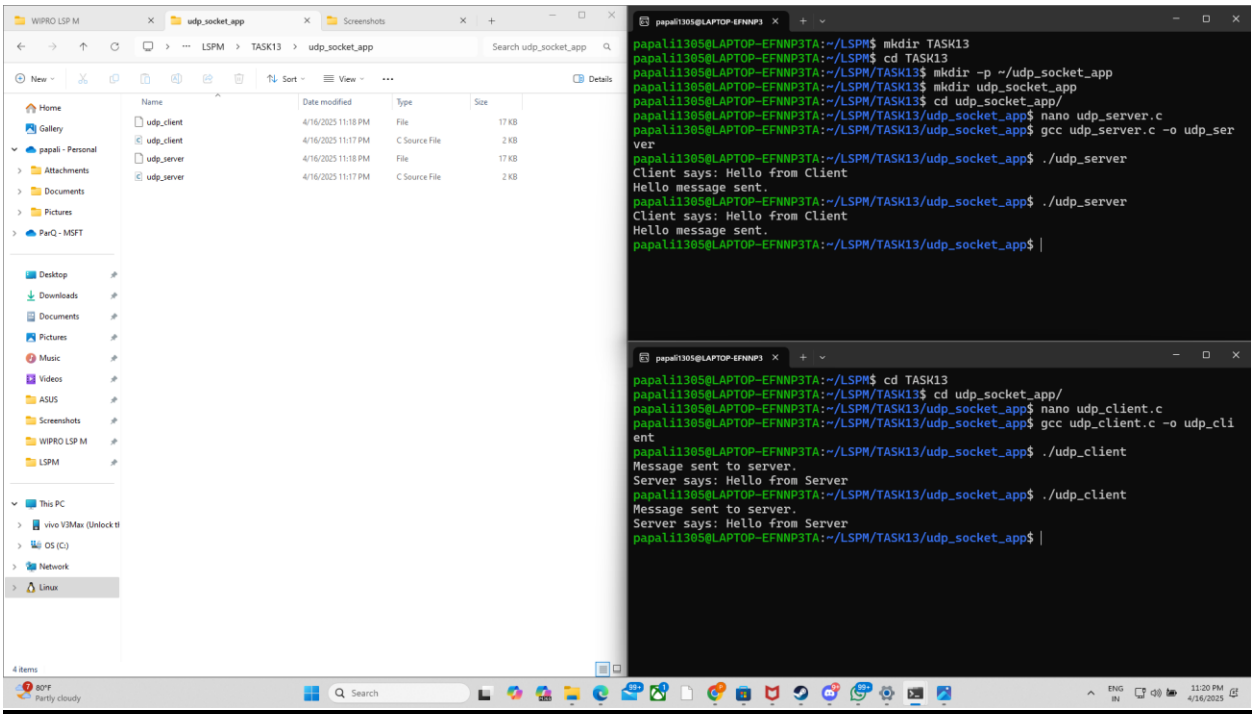
02).



03).


papali1305@LAPTOP-EFNNP3TA:~/LSPM$ mkdir TASK13
papali1305@LAPTOP-EFNNP3TA:~/LSPM$ cd TASK13
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13$ mkdir -p ~/udp_socket_app
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13$ mkdir udp_socket_app
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13$ cd udp_socket_app/
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13/udp_socket_app$ nano udp_server.c
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13/udp_socket_app$
papali1305@LAPTOP-EFNNP3TA:~/LSPM$ cd TASK13
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13$ cd udp_socket_app/
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13/udp_socket_app$ nano udp_client.c
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13/udp_socket_app$

04).


papali1305@LAPTOP-EFNNP3TA:~/LSPM$ mkdir TASK13
papali1305@LAPTOP-EFNNP3TA:~/LSPM$ cd TASK13
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13$ mkdir -p ~/udp_socket_app
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13$ mkdir udp_socket_app
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13$ cd udp_socket_app/
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13/udp_socket_app$ nano udp_server.c
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13/udp_socket_app$ gcc udp_server.c -o udp_server
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13/udp_socket_app$ ./udp_server
Client says: Hello from Client
Hello message sent.
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13/udp_socket_app$ ./udp_server
Client says: Hello from Client
Hello message sent.
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13/udp_socket_app$
papali1305@LAPTOP-EFNNP3TA:~/LSPM$ cd TASK13
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13$ cd udp_socket_app/
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13/udp_socket_app$ nano udp_client.c
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13/udp_socket_app$ gcc udp_client.c -o udp_client
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13/udp_socket_app$ ./udp_client
Message sent to server.
Server says: Hello from Server
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13/udp_socket_app$ ./udp_client
Message sent to server.
Server says: Hello from Server
papali1305@LAPTOP-EFNNP3TA:~/LSPM/TASK13/udp_socket_app$