Multi-Client TCP Server:

Implement a TCP server that:

Binds to port 6060.

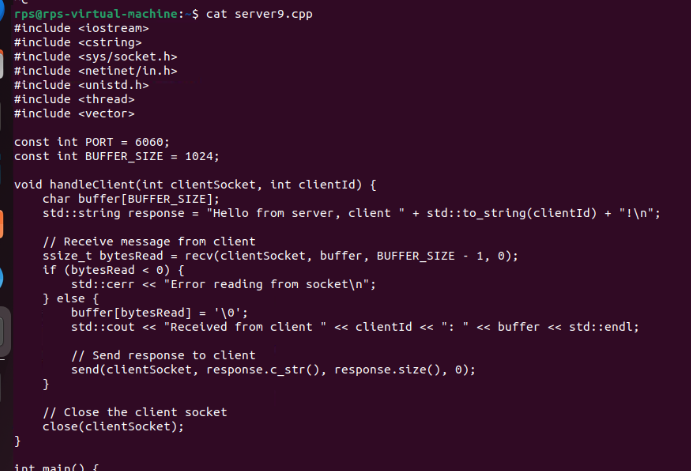
Listens for incoming connections.

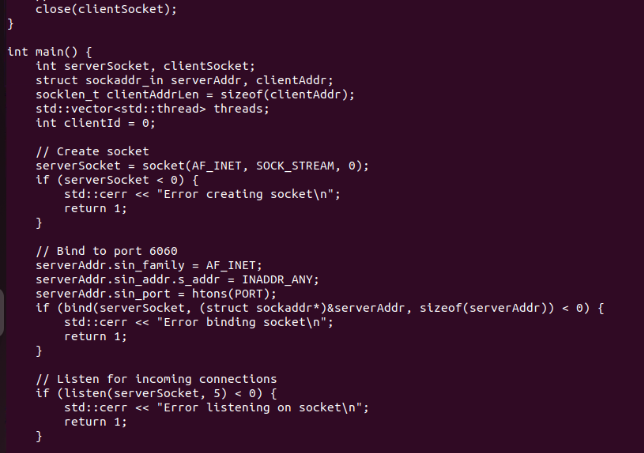
Accepts multiple client connections concurrently.

Receives a message from each client and sends a unique response back.

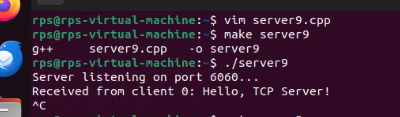
Closes the connections and terminates after handling all clients.

Server Part:



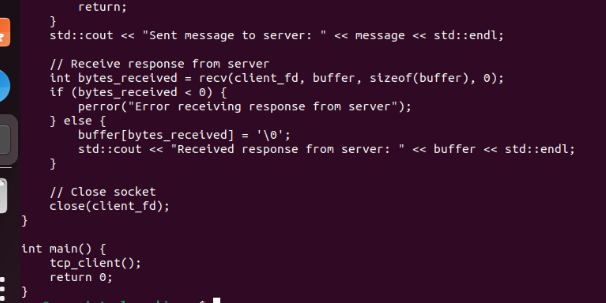


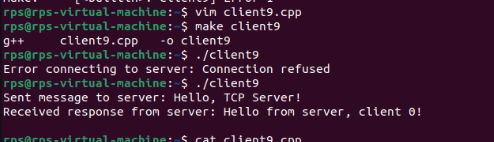




Client Part:







TCP Client with Error Handling:

Create a TCP client that:

Connects to a server at port 5050.

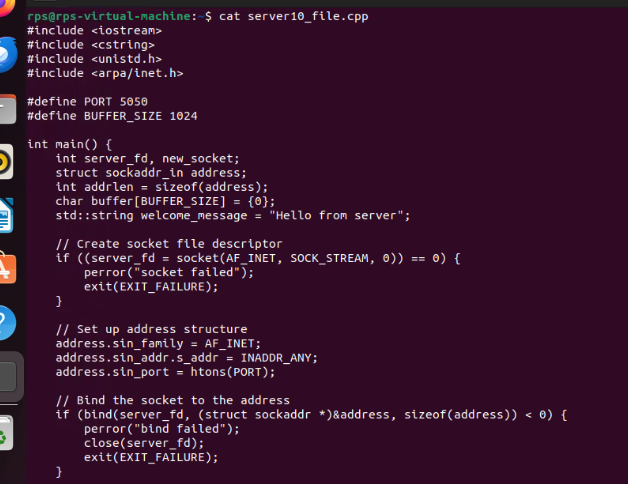
Sends a message to the server.

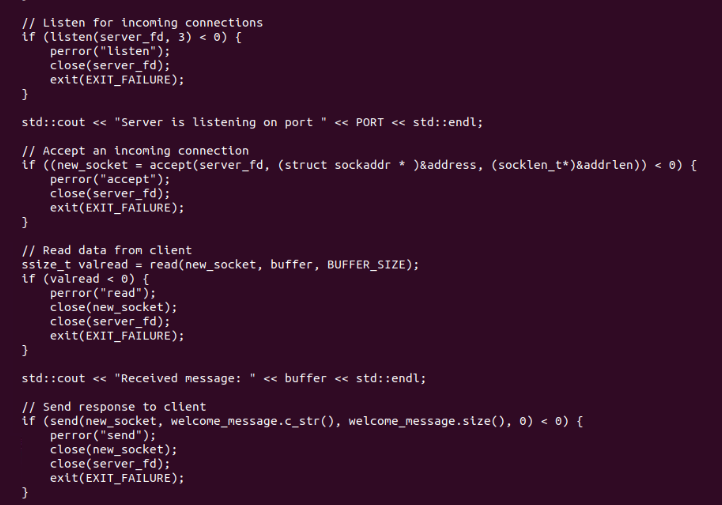
Handles and displays error messages for common issues such as connection failure or data transmission errors.

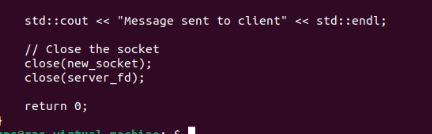
Receives and prints the response message from the server.

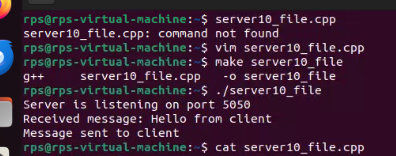
Closes the socket and terminates.

Server Part:

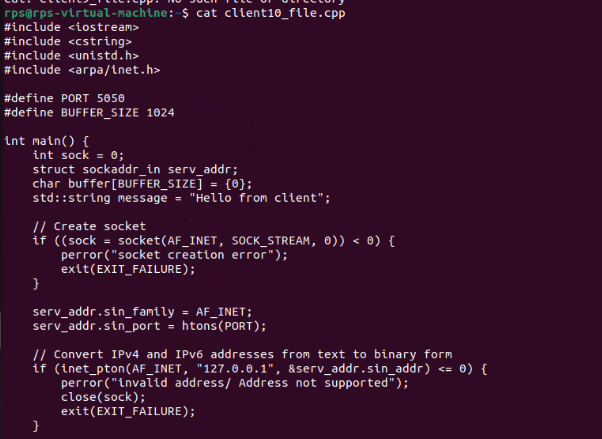


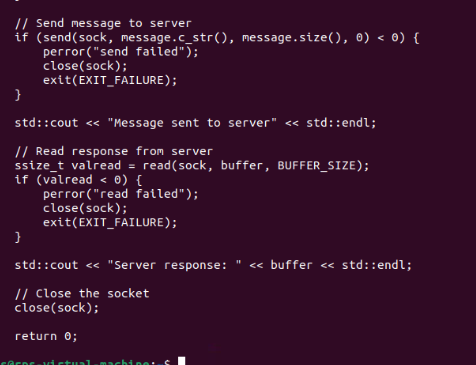


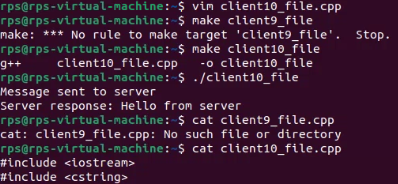




Client Part:







Server with Select for Multiple Clients:

Implement a TCP server that:

Binds to port 4040.

Listens for incoming connections.

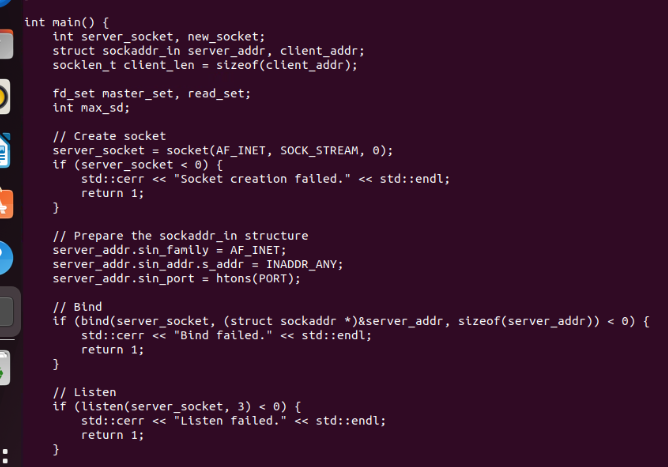
Uses select() to handle multiple client connections.

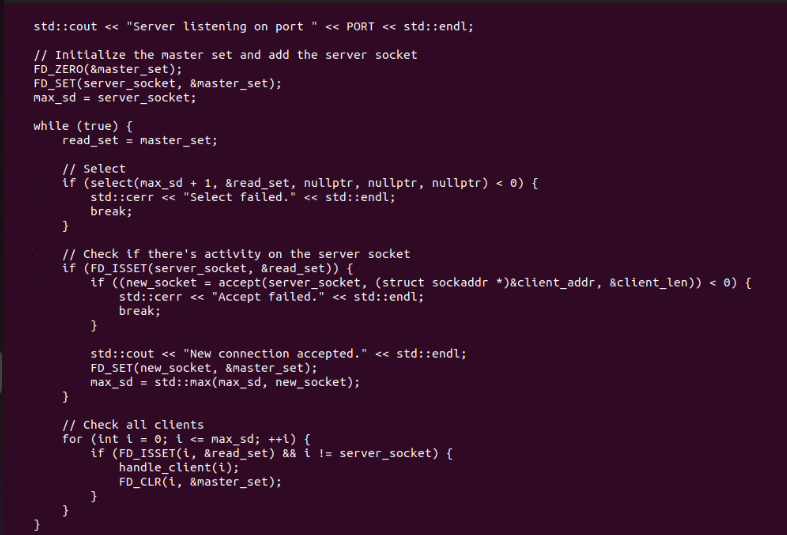
Receives a message from each client and sends a response back.

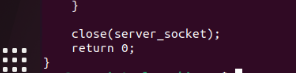
Closes the connections and terminates.

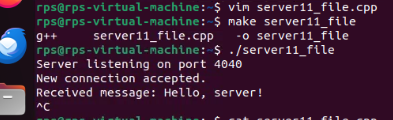
Server Part:



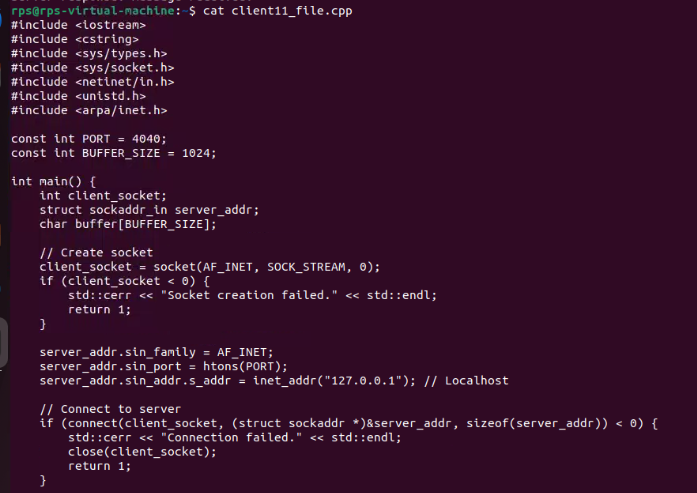


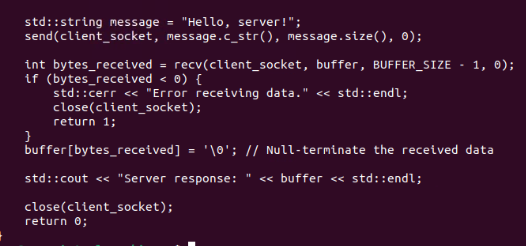


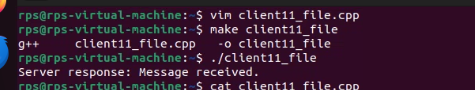




Client Part:







Client with Timeout:

Create a TCP client that:

Connects to a server at port 3030.

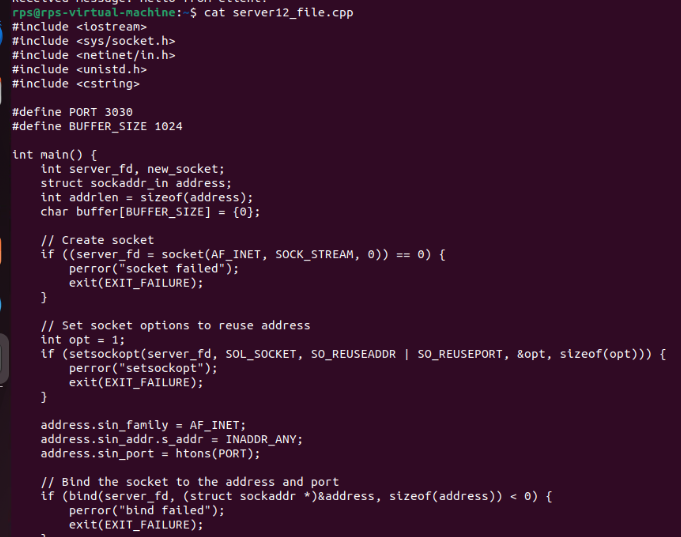
Sends a message to the server.

Implements a timeout mechanism to handle cases where the server does not respond within a specified time.

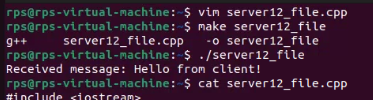
Receives and prints the response message if available.

Closes the socket and terminates.

Server Part:







Client Part:

