LINUX MULTI-THREADED CLIENT-SERVER USING SHARED MEMORY PROJECT USER-MANUAL DOCUMENT

Table of Contents

[**1.** **INTRODUCTION** 3](#_Toc168435044)

[**2.** **SYSTEM REQUIREMENTS** 3](#_Toc168435045)

[**3.** **INSTALLATION** 3](#_Toc168435046)

[**4.** **SERVER SETUP** 3](#_Toc168435047)

[**5.** **CLIENT CONNECTION** 3](#_Toc168435048)

[**6.** **DATA RETRIEVAL** 4](#_Toc168435049)

[**7.** **TERMINATION** 4](#_Toc168435050)

[**8.** **TESTING** 4](#_Toc168435051)

[**9.** **TROUBLESHOOTING** 4](#_Toc168435052)

# **INTRODUCTION**

Welcome to the Simple Client-Server Communication System user manual. This guide provides step-by-step instructions for setting up and using the client-server system implemented using sockets and shared memory in the C programming language.

# **SYSTEM REQUIREMENTS**

1. ***Server Side***
2. Unix-like operating system (e.g., Linux, macOS)
3. C compiler (e.g., gcc)
4. POSIX-compliant libraries (for socket programming and shared memory)
5. ***Client Side***
   1. Unix-like operating system
   2. C compiler
   3. POSIX-compliant libraries

# **INSTALLATION**

* Download the source code for the client and server programs.
* Extract the files to your desired location.

# **SERVER SETUP**

1. Open a terminal window.
2. Navigate to the directory containing the server source code.
3. Compile the server program using the following command:

gcc server.c -o server

1. Run the compiled server program:

./server

1. The server will start listening for incoming client connections on a default port (8080).

# **CLIENT CONNECTION**

1. Open a terminal window.
2. Navigate to the directory containing the client source code.
3. Compile the client program using the following command:

gcc client.c -o client

1. Run the compiled client program with the following command:
   * 1. ./client

# **DATA RETRIEVAL**

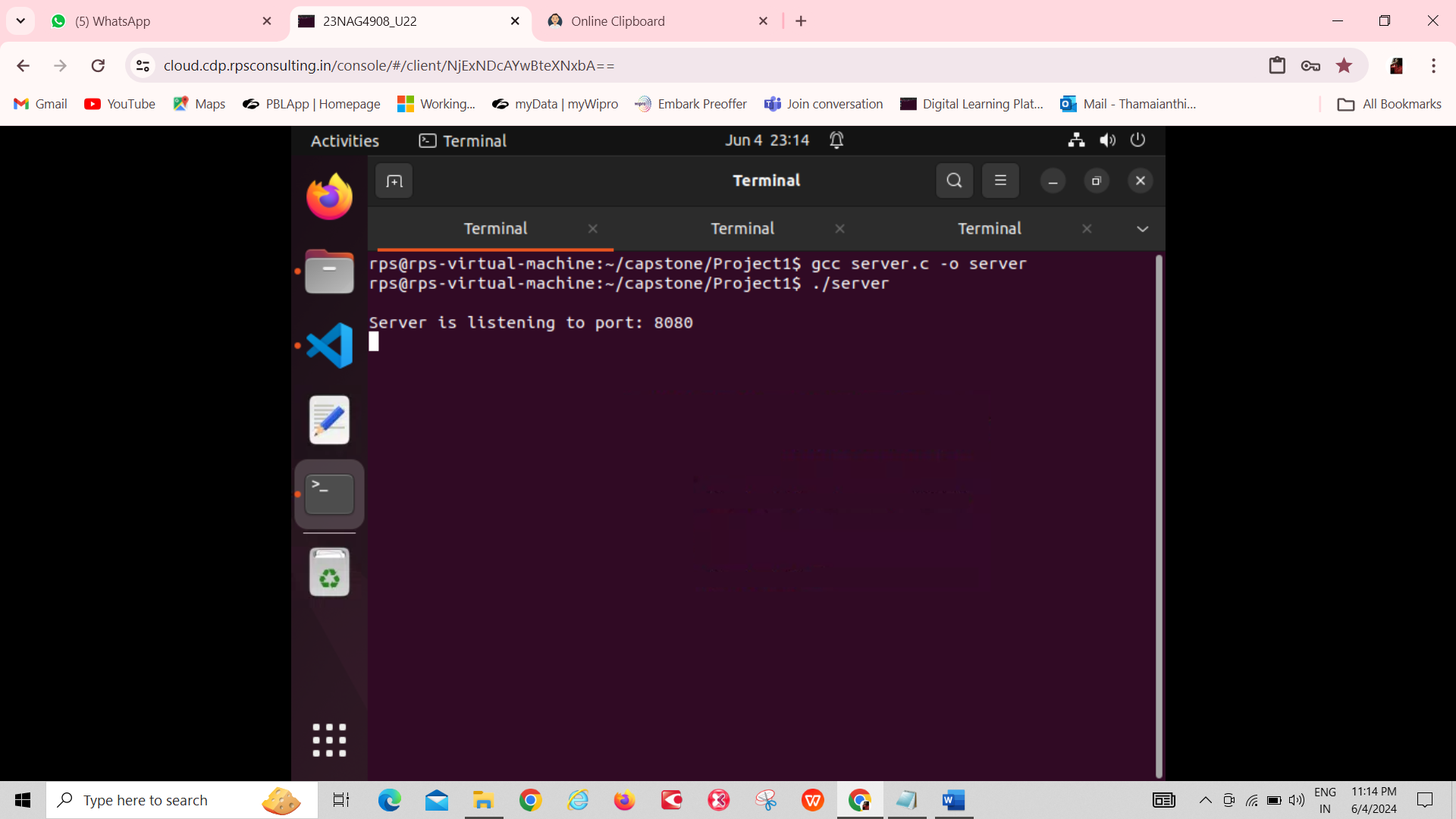
* Once the client program is running, it will connect to the server automatically.
* The client will start receiving random numbers generated by the server.
* Random numbers will be displayed on the client's terminal at regular intervals.

# **TERMINATION**

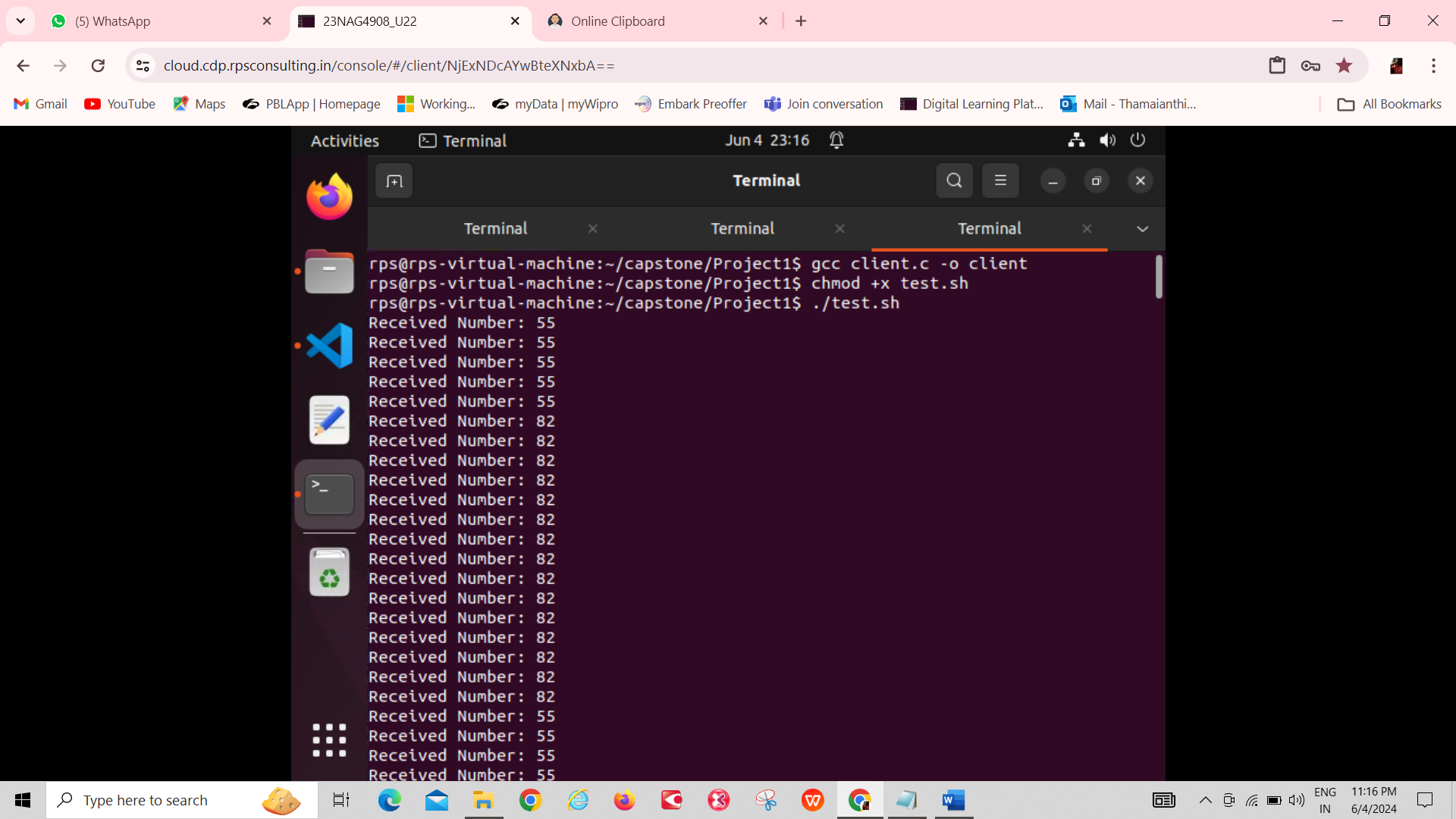
* To terminate the server program, press Ctrl + C in the terminal where the server is running.
* To terminate the client program, press Ctrl + C in the terminal where the client is running.

# **TESTING**

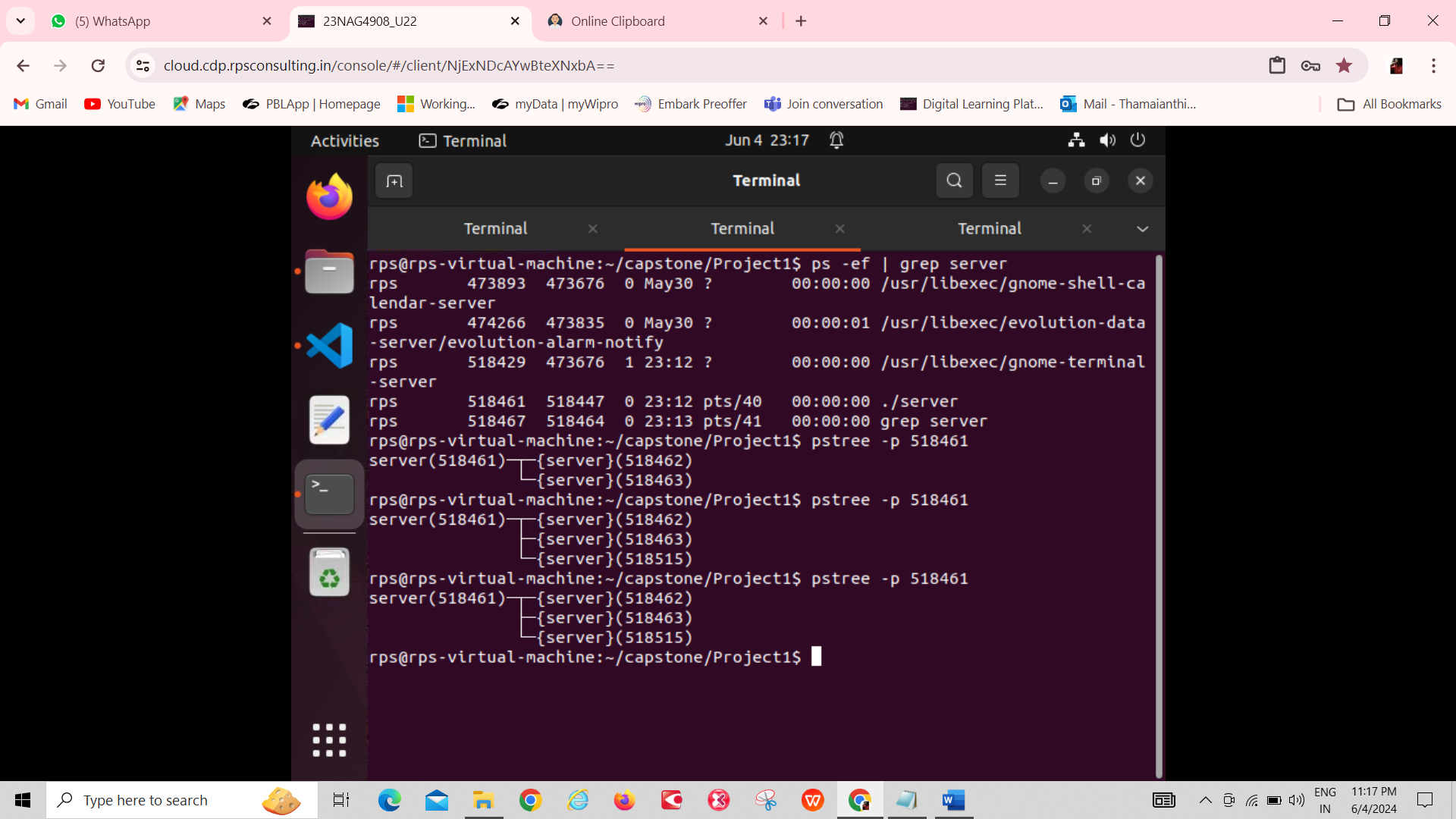
**1.Server**



**2.Client**



**3.Thread-creation**



# **TROUBLESHOOTING**

1. ***Connection Issues:***

* Ensure that the server is running and accessible from the client's machine.
* Check the server's IP address and port configuration.

1. ***Compilation Errors:***

* Check for any compilation errors when compiling the server and client programs.
* Ensure that all dependencies are installed correctly.

1. ***Runtime Errors:***

* Look for any runtime errors or segmentation faults when running the server and client programs.
* Review the code for potential bugs or issues.