# Assingment-5.

## Group Details:

1. Adarsh Dhakar → 22CS01040
2. Avik Sarkar → 22CS01060
3. Debargha Nath → 22CS01070
4. Soham Chakraborty → 22CS02002

## Github Repository Link:

https://github.com/adarshdhakar/cn_lab_sheet5/

## Images Directory Link:

https://github.com/adarshdhakar/cn_lab_sheet5/images

## Report.pdf Link:

https://github.com/adarshdhakar/cn_lab_sheet5/Report.pdf

## Demo Video Link:

https://github.com/adarshdhakar/cn_lab_sheet5/Demo.mp4

**1.**

## Why C++ over C ?

- Use of **string** makes it easier to handle text messages instead of relying on character arrays where we would have used strcpy() and strcat().
- Use of **set<int>** to maintain list of active clients efficiently.

## What all libraries used ? Funtionalities provided by these.

**#include <bits/stdc++.h>**

Functions used: string, set, iostream

**#include <netinet/in.h>**

Provides sturctures for internet addresses **(sockaddr_in)**

**#include <netdb.h>** (only in Client)

Contains **gethostbyname()** to resolve hostnames to IP addresses.

**#include <pthread.h>**

Used for multi-threading:

- Creating threads for handling multiple clients on server.
- Creating separate threads for reading and writing for the client.

## Explaination of Flow and Working:

**Server:**

- Initialization
- Accepting Clients
- Broadcasting Messages

**Client:**

- Connection
- Messaging

**2.**

**i) Develop a Chat Server program using threads which can:**

- **Handle multiple clients at the same time.**
- **A client can join/disconnect from the chat.**
- **Two clients can chat via server.**
- **A client can choose to broadcast the message to all clients alive.**
- **Add more functionalities → Timeout**

**ii) Design the Chat Server program (in place of threads) use select() system call to connect multiple clients.**

**iii) Design the Client program for the chat-server.**