

# **Systems Practicum (CS-307)**

## **Final Project**

Under the supervision of  
Prof. Varun Dutt



### **Command-Pedia**

#### **Group 3**

**Saransh Bansal (B20132)**

**Abhay Gupta (B20075)**

**Himakshi Gupta (B20104)**

**Rachit Goel (B20309)**

**Shruti Jain (B20136)**

**Titiksha Behal (B20138)**

**Isha Sukhija (B20105)**

**Vastav Bansal(B20325)**

# Index

<b>Introduction</b>	<b>3</b>
<b>Requirements for building the application</b>	<b>4</b>
<b>Features</b>	<b>6</b>
<b>Diagrams</b>	<b>7</b>
<b>Code Overview</b>	<b>10</b>
<b>Plan</b>	<b>12</b>
<b>Working</b>	<b>13</b>
<b>Future Scope and Improvements</b>	<b>18</b>
<b>Conclusion</b>	<b>19</b>
<b>References</b>	<b>20</b>

# Introduction

In today's fast-paced and information-driven world, easy access to knowledge plays a pivotal role in personal, educational, and professional growth. As the volume of information continues to expand exponentially, there arises a pressing need for efficient, reliable, and convenient platforms that empower individuals to navigate this vast sea of knowledge. The Command Line Encyclopedia is a sophisticated server-client application that utilizes socket programming and thread management to provide users with a seamless command line interface for retrieving information on various topics. This project aims to create an efficient and scalable system where multiple clients can connect to a central server, submit their queries, and receive relevant information in real-time. The user, acting as a client, interacts with the system through a terminal interface. By entering a specific "topic" of interest, the client initiates a request to the server. The server, in turn, processes the query by searching through an internally developed file system, which contains a comprehensive collection of information. Once the requested information is found, the server responds to the client, providing the relevant details.

## Requirements for building the application:

The project is created using React.js framework and hence has the following requirements.

### Technologies Used :

Socket Programming	For Client server interaction
Threading	For entertaining multiple clients
C language	For writing programs
Inbuilt file system	For storing the information about various topics on the server side

## Collaboration :

VS-code Live Share	For live code collaboration.
Git	For version control of the project.
GitHub	For hosting the code and the "Organization" feature was used for the collaboration of the team.

So to run this application we just need an executable environment i.e Windows, linux, macOS, etc. Now follow the given steps:

Note: This project requires the Linux Operating System.

1. Clone the code from the following repository:  
<https://github.com/Saransh019/Command-Pedia>
2. Compile the Server.c file and run it in a terminal window using the following command :

```
$ gcc Server.c -o Server && ./Server
```

3. Open a new terminal window and compile and run the client file using the following command :

```
$ gcc Client.c -o Client && ./Client
```

## Features

Following are the features of our application:

- **Provides an easy access to information :** The application provides the user with a seamless command line interface for retrieving information on various topics. The client can make requests for information on a specific topic which are then processed by the server by searching for information in an internal file system, and the retrieved information is then passed on to the client. The application hence acts as a comprehensive source of easily available knowledge for the client.

- **Multiple connections can be established simultaneously** : The application uses threading to enable the server to connect to multiple clients at once. Multiple users can access the information simultaneously in this manner and this helps make the application scalable for future use.
- **Different operation modes** : The application can be used in two different operation modes, user mode and admin mode. In the user mode, the already available information can be accessed by the client. Whereas, in the admin mode, the client can provide more topics and information to the server which are then stored by the server and can be used for connections.
- **Add information to the server** : The client can also add new topics and their corresponding information to the server's file system. This is done in the admin mode.
- **User registration and validation** : Users have to register themselves with the server using a particular format. The server also validates the user's name to ensure that it is entered in the correct format.
- **Termination** : The user can anytime choose to exit the application by entering 'q' as their input message.

## UML Diagrams

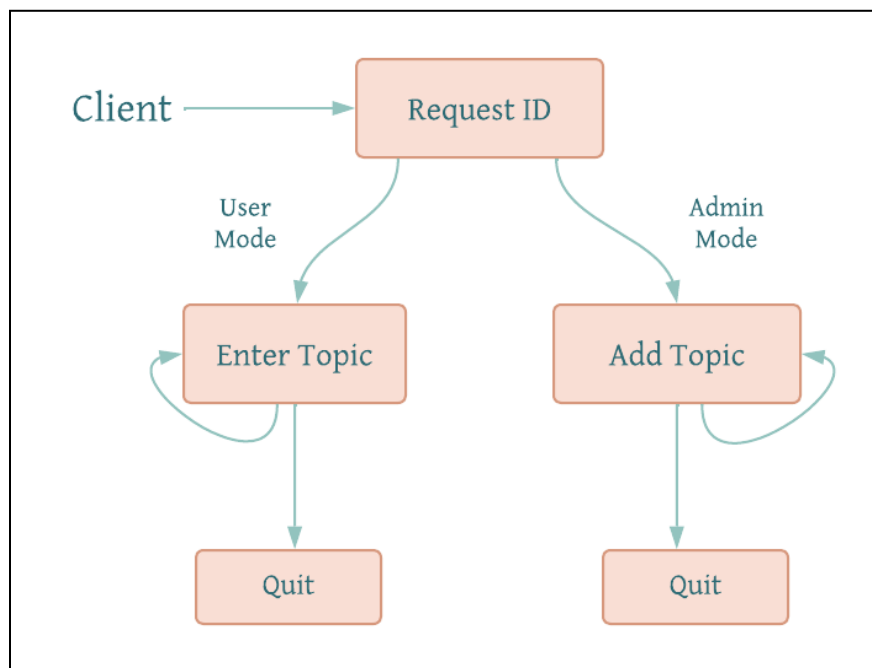


Fig 1 . UML diagram showing overview of application

- The client opens the application and is then required to enter his user-ID.
- After that the client is made to choose between two modes of operation, user mode and admin mode.
- In the user mode, the client is prompted to enter a topic on which he/she requires information.
- If the topic is present in the file system, then the information would be displayed on the screen otherwise a message would be displayed informing the client that the above topic is not available.
- In the admin mode, the client can add more topics to the file system. The client first needs to enter the new topic followed by the relevant information on that topic.
- The client can any time exit the application by choosing to quit.

## Code overview

### Structure :

The project consists of initially 2 folders and 2 major files .

- **Folders:**

1. **Index:** This stores the names of all the topics that the server contains the information about.
2. **Topics:** This contains the information about each topic present in the index folder.

- **Files:**

1. **Server.c :** To implement a basic server that handles multiple client connections, allowing users to search for topics and admins to add new topics.
2. **Client.c :** To establish a connection to a server, allows the user to send messages to the server, and receives messages from the server, using a multi-threaded approach.

# Detailed timeline

Tasks	Start Date	End Date
Topic research and finalization	1 March	7 March
Going through the necessary concepts	10 March	16 March
Planning the project and defining the scope	16 March	22 March
Designing the base architecture	30 March	4 April
Gathering data for the file system	5 April	10 April
Building initial project structure and gathering requirements	11 April	13 April
Developing Server side code	14 April	18 April
Developing client side code	19 April	23 April
Integrating server and client side codes and testing them	24 April	28 April
Fixing bugs	29 April	2 May
Final Testing	6 May	9 May

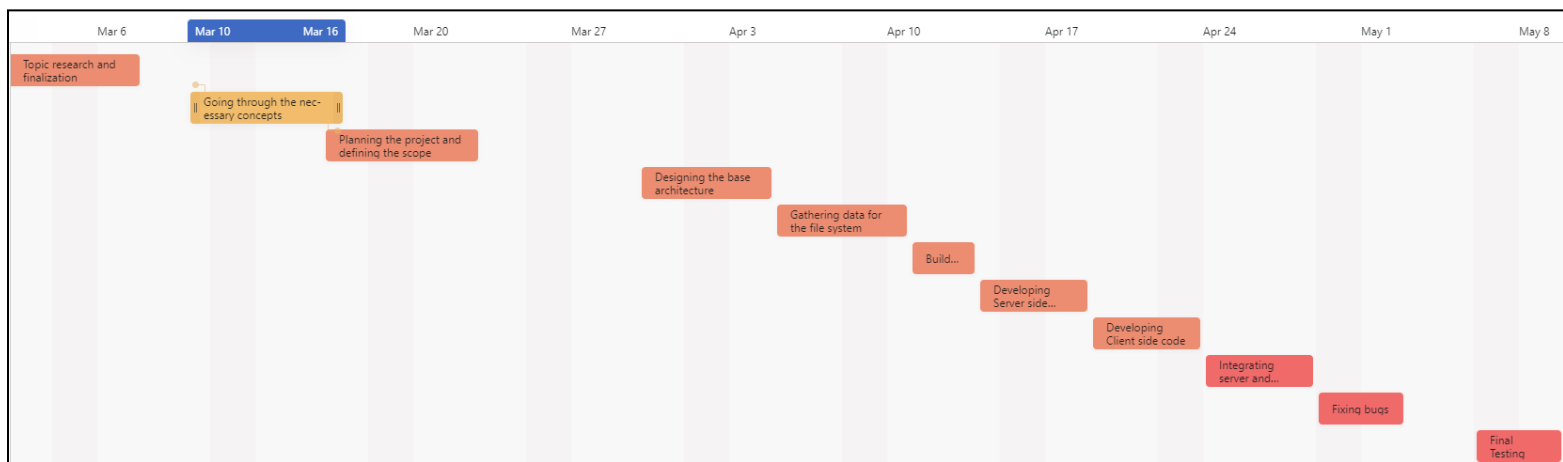
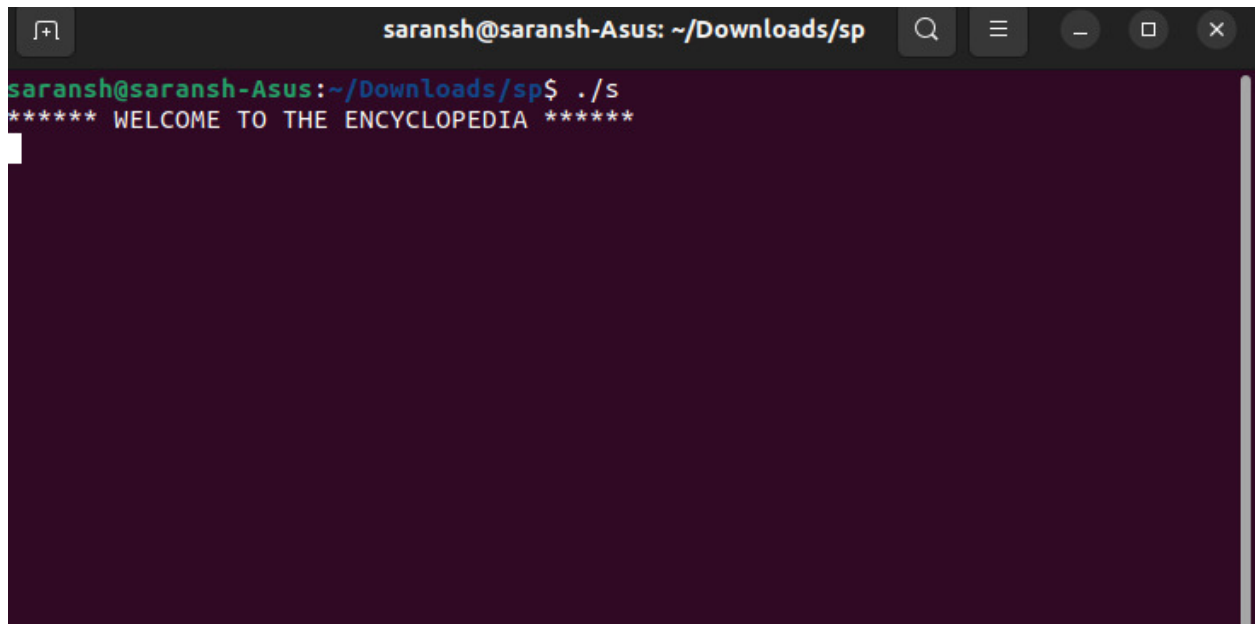


Fig 2. Gantt chart

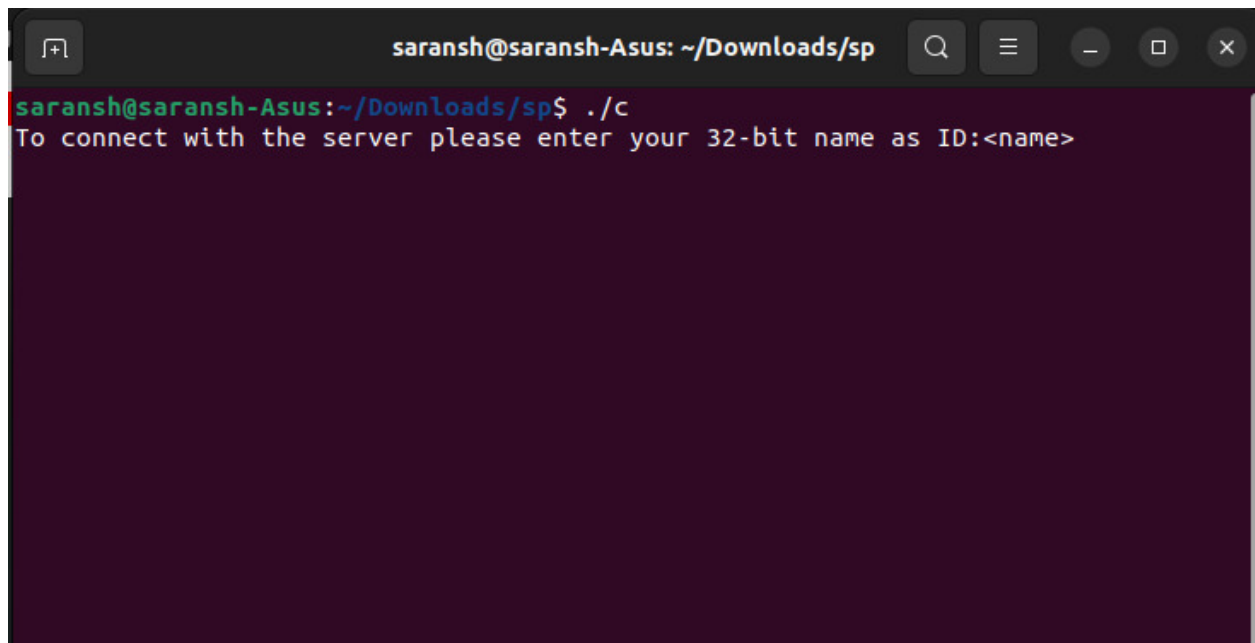
# Working

1. Start the server.

A terminal window titled 'saransh@saransh-Asus: ~/Downloads/sp' with standard window controls. The prompt is 'saransh@saransh-Asus:~/Downloads/sp\$'. The user enters './s', and the terminal outputs '\*\*\*\*\* WELCOME TO THE ENCYCLOPEDIA \*\*\*\*\*'.

```
saransh@saransh-Asus: ~/Downloads/sp
saransh@saransh-Asus:~/Downloads/sp$ ./s
***** WELCOME TO THE ENCYCLOPEDIA *****
```

2. The Client is required to enter their 32 bit name as ID.

A terminal window titled 'saransh@saransh-Asus: ~/Downloads/sp' with standard window controls. The prompt is 'saransh@saransh-Asus:~/Downloads/sp\$'. The user enters './c', and the terminal outputs 'To connect with the server please enter your 32-bit name as ID:<name>'.

```
saransh@saransh-Asus: ~/Downloads/sp
saransh@saransh-Asus:~/Downloads/sp$ ./c
To connect with the server please enter your 32-bit name as ID:<name>
```



3. There are two modes: User mode and Admin mode. The client is required to choose one of them. In user mode, the client can fetch the required information and in the admin mode, the client can input the information which will be stored in the form of input files.

```
saransh@saransh-Asus: ~/Downloads/sp
saransh@saransh-Asus:~/Downloads/sp$ ./c
To connect with the server please enter your 32-bit name as ID:<name>
ID:saransh

Welcome saransh to the command line encyclopedia

Select mode:
press I for user mode
press A for admin mode

format-> mode:<type of mode>

>
```

4. If the client selects the user mode, he has to enter the name of the topic on which information has to be fetched.

```
saransh@saransh-Asus: ~/Downloads/sp
saransh@saransh-Asus:~/Downloads/sp$ ./c
To connect with the server please enter your 32-bit name as ID:<name>
ID:saransh

Welcome saransh to the command line encyclopedia

Select mode:
press I for user mode
press A for admin mode

format-> mode:<type of mode>

> mode:I
> Ok. Type a topic to search:> scheduling
```

- If that topic is present in the server, the required information would be displayed. The client can then either quit or choose another topic to fetch information.

```
saransh@saransh-Asus: ~/Downloads/sp
saransh@saransh-Asus:~/Downloads/sp$ ./c
To connect with the server please enter your 32-bit name as ID:<name>
ID:saransh

Welcome saransh to the command line encyclopedia

Select mode:
press I for user mode
press A for admin mode

format-> mode:<type of mode>

> mode:I
> Ok. Type a topic to search:> scheduling
> The process scheduling is the activity of the process manager that handles the
  removal of the running process from the CPU and the selection of another proces
  s on the basis of a particular strategy.>
To search another topic press 'n'
To quit press 'q'
```

- If that topic is not present in the server, the terminal would display a message indicating that the topic is not present in the server.

```
saransh@saransh-Asus: ~/Downloads/sp
saransh@saransh-Asus:~$ cd Downloads/sp
saransh@saransh-Asus:~/Downloads/sp$ ./c
To connect with the server please enter your 32-bit name as ID:<name>
ID:saransh

Welcome saransh to the command line encyclopedia

Select mode:
press I for user mode
press A for admin mode

format-> mode:<type of mode>

> I
> Enter valid mode format.
> mode:I
> Ok. Type a topic to search:> mongo
> Sorry, Topic not present
>
To search another topic press 'n'
To quit press 'q'
```

- The Client exits the terminal on pressing quit.

```
saransh@saransh-Asus: ~/Downloads/sp
saransh@saransh-Asus:~/Downloads/sp$ ./c
To connect with the server please enter your 32-bit name as ID:<name>
ID:saransh

Welcome saransh to the command line encyclopedia

Select mode:
press I for user mode
press A for admin mode

format-> mode:<type of mode>

> mode:I
> Ok. Type a topic to search:> scheduling
> The process scheduling is the activity of the process manager that handles the
  removal of the running process from the CPU and the selection of another proces
  s on the basis of a particular strategy.>
To search another topic press'n'
To quit press 'q'

> q
Exited Successfully
saransh@saransh-Asus:~/Downloads/sp$
```

- The server logs the actions of the client, and if the client quits, it waits for another client to join.

```
saransh@saransh-Asus: ~/Downloads/sp
saransh@saransh-Asus:~/Downloads/sp$ ./s
***** WELCOME TO THE ENCYCLOPEDIA *****
msgtype: ID
msg: saransh
saransh has selected I mode
'scheduling'
topic info: The process scheduling is the activity of the process manager that h
andles the removal of the running process from the CPU and the selection of anot
her process on the basis of a particular strategy.
''
saransh has left

```

5. If the client selects the admin mode, he has to enter the name of the topic and the information on that topic.

```
saransh@saransh-Asus: ~/Downloads/sp
saransh@saransh-Asus:~/Downloads/sp$ ./c
To connect with the server please enter your 32-bit name as ID:<name>
ID:saransh

Welcome saransh to the command line encyclopedia

Select mode:
press I for user mode
press A for admin mode

format-> mode:<type of mode>

> mode:A
> you are in ADMIN mode. You can add topic in the following format:
<topic> ; <info>
>
```

6. The topic and information get added successfully. He can then either enter a new topic and information or quit the terminal.

```
saransh@saransh-Asus: ~/Downloads/sp
saransh@saransh-Asus:~/Downloads/sp$ ./c
To connect with the server please enter your 32-bit name as ID:<name>
ID:saransh

Welcome saransh to the command line encyclopedia

Select mode:
press I for user mode
press A for admin mode

format-> mode:<type of mode>

> mode:A
> you are in ADMIN mode. You can add topic in the following format:
<topic> ; <info>
> sql;Structured Query Language
> Topic added successfully!!!

Press q to quit
Press n to add more topics
```

## Future Scope and Improvements

- Expansion of Information Database: Continuously updating and expanding the internal file system with new and relevant information will enhance the breadth and depth of knowledge available to users. Regularly incorporating new data sources and implementing mechanisms for automatic updates will ensure the system remains up-to-date.
- User Authentication and Personalization: Implementing user authentication mechanisms can enable personalized experiences, such as saving user preferences, history, and customized content recommendations. This can enhance user engagement and satisfaction with the system.
- Performance Optimization: Continuously monitoring and optimizing the system's performance, including response time, scalability, and resource utilization, will ensure smooth operation even under high loads. This can involve optimizing database queries, caching frequently accessed data, or implementing load balancing techniques.

## Conclusion

The Command-Pedia project successfully created a server-client application using socket programming and thread management. The project provides users with a convenient command line interface for accessing a comprehensive collection of information in real-time. Through this project, we gained valuable insights and learnings. We developed a strong understanding of socket programming and its role in establishing reliable client-server communication. Additionally, we learned how to effectively manage threads to enhance system performance and responsiveness.

The project showcased the importance of scalability and responsiveness in server-client applications, allowing multiple clients to connect simultaneously. The implementation of a comprehensive file system and efficient query processing contributed to the project's success.

Overall, the Command-Pedia project demonstrated the power of socket programming and thread management in creating efficient information retrieval systems. The experience gained from this project will undoubtedly benefit future endeavors in developing similar applications.

## References

1. <https://en.wikipedia.org/wiki/N-back>
2. <https://www.millisecond.com/download/library/nback/>
3. <https://pubmed.ncbi.nlm.nih.gov/17470009/#:~:text=The%20N%2Dback%20task%20requires,behavioral%20tests%20of%20construct%20validity>
4. <https://www.frontiersin.org/articles/10.3389/fpsyg.2017.00352/full>