CSCE 662 - Design Document for MP1(Tiny SNS)
Name: Cherukuri Venkata Surya Krishna

Introduction

Overview of the Client/Server Architecture

- The system architecture is based on the client/server model, a distributed application structure that partitions tasks or workloads between providers of a resource or service, called servers, and service requesters, called clients.
- This model enhances the scalability and maintenance by separating concerns, allowing clients and servers to be developed and updated independently.

System Architecture

High-Level Architecture

- The architecture includes a gRPC server (tsd.cc) and multiple gRPC clients (tsc.cc), interacting over a well-defined set of procedures specified in sns.proto.
- The server handles multiple client requests concurrently, managing user data and session states, while clients provide a user interface for interaction with the server.

sns.proto

Overview: This Protocol Buffers (protobuf) file defines the structure and services for a social networking application using proto3 syntax, facilitating robust client-server communication.

Services:

- **SNSService:** This is the main service provided by the server, encompassing several key functionalities:
  - **Login:** Authenticates a user based on provided credentials.
  - **List:** Fetches a list of all users and the followers for the authenticated user.
  - **Follow:** Allows a user to follow another user.
  - **UnFollow:** Allows a user to unfollow another user.
  - **Timeline:** Establishes a bidirectional streaming connection for real-time message updates between the client and server.

Messages:

- **ListReply:**
  - **all_users (repeated string):** A list of all registered usernames.
  - **followers (repeated string):** A list of usernames that follow the requesting user.

- **Request:**
  - **username (string):** The username of the user making the request.
  - **arguments (repeated string):** Additional arguments that may be needed for various requests like Follow or UnFollow.

- **Reply:**
  - **msg (string):** A message from the server that could be a confirmation or error message related to the request.

- **Message:**
  - **username (string):** The username of the user who sent the message.
  - **msg (string):** The content of the message.
  - **timestamp (google.protobuf.Timestamp):** The exact time the message was sent, useful for ordering messages chronologically in a timeline.

tsc.cc (Client)

1. MakeMessage Function

- **Purpose**: Constructs a Message object using the provided username and message text. It also attaches the current timestamp to this message.
- **Process**:
  - A new Message object is created.
  - User's username and msg are set.
  - A timestamp is generated and set to the current time.
  - The message object is returned.

2. Client Class Functions

- **connectTo**:
  - **Purpose**: Establishes a connection to the server using the hostname and port. It attempts to log in to verify the connection.
  - **Process**: Creates a channel using grpc::CreateChannel, initializes a stub, and attempts to log in via the Login function.

- **processCommand**:
  - **Purpose**: Processes a command entered by the user. This could be one of several commands like FOLLOW, UNFOLLOW, LIST, or TIMELINE.
  - **Process**: Parses the command and dispatches to the appropriate function such as Follow, UnFollow, List, or handles timeline processing.

- **processTimeline**:

  - **Purpose**: Manages the user's timeline, allowing them to see messages and post new ones.
  - **Process**: Enters a loop where it reads from and writes messages to the server using a bidirectional stream.

- **Login, List, Follow, UnFollow**:

  - These functions make corresponding gRPC calls to the server to perform operations related to logging in, listing users, following a user, and unfollowing a user, respectively.

Server-side (tsd.cc)

1. SNSServiceImpl Class Functions

- **List:**

  - **Purpose:** Retrieves and sends a list of all users and the requester's followers.
  - **Process:** Iterates through the client_db to compile and return the necessary user data.

- **Follow and UnFollow:**

  - **Purpose:** Modifies follow relationships between users.
  - **Process:** Verifies the existence of the target user and updates the follow/followers lists. Handles special cases like self-following or non-existent user following.

- **Login:**

  - **Purpose:** Manages user login requests.
  - **Process:** Registers new users into client_db or addresses login conflicts if the user is already logged in.

- **Timeline:**

  - **Purpose:** Facilitates a live message feed for users.
  - **Process:** Keeps a continuous bidirectional stream open for the user, managing the flow of incoming and outgoing messages.

2. Main Functions

- **RunServer:**

  - **Purpose:** Sets up and runs the gRPC server.
  - **Process:** Configures the server's listening parameters, registers service implementations, and launches the server to handle incoming client connections.

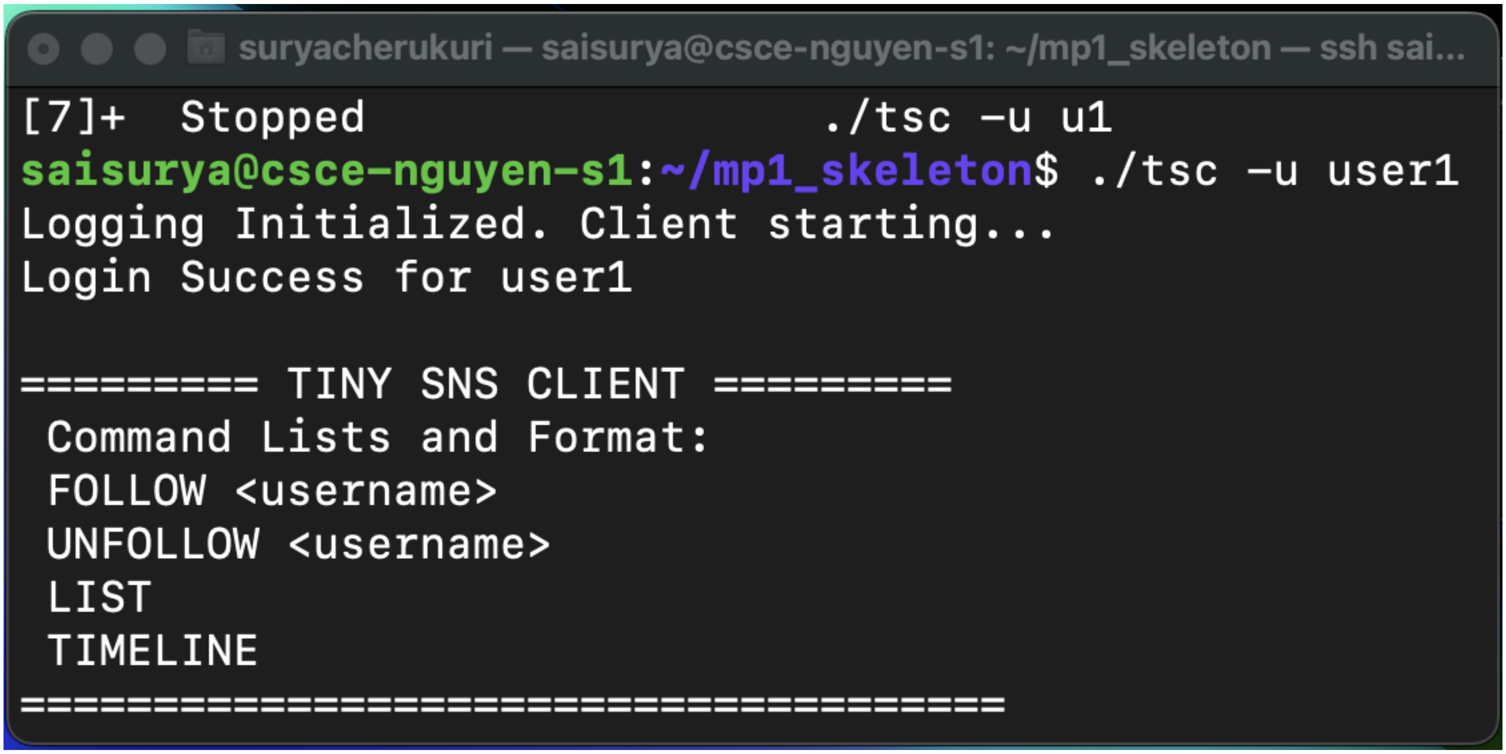
- **main:**

  - **Purpose:** Entry point for server execution.
  - **Process:** Handles command-line arguments for server configuration and starts the server through RunServer.

Output (TestCases)

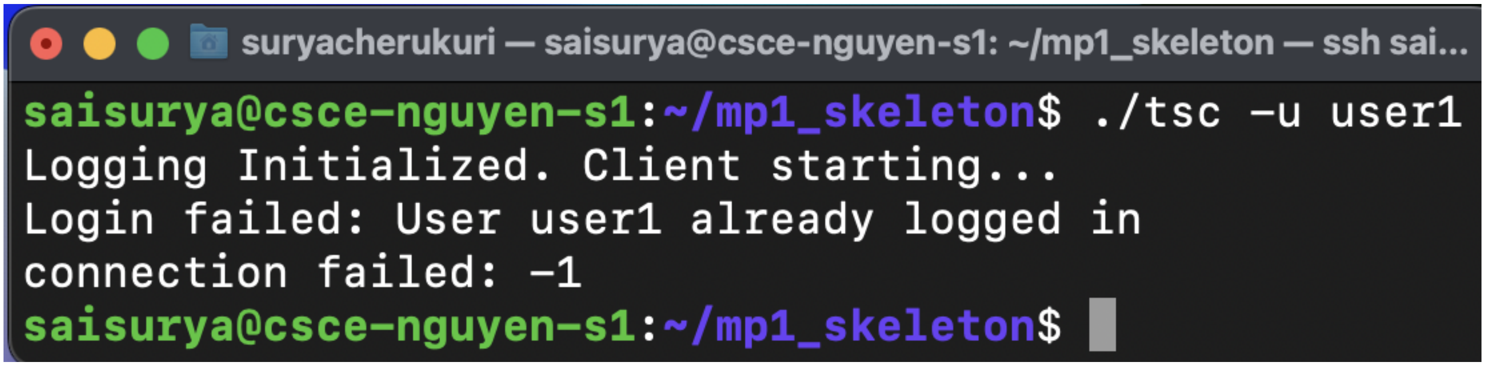
TestCase 1

- **Command & Output**:
  - **./tsc -u user1**


suryacherukuri — saisurya@csce-nguyen-s1: ~/mp1_skeleton — ssh sai...

```
[7]+  Stopped                        ./tsc -u u1
saisurya@csce-nguyen-s1:~/mp1_skeleton$ ./tsc -u user1
Logging Initialized. Client starting...
Login Success for user1

========= TINY SNS CLIENT =========
 Command Lists and Format:
 FOLLOW <username>
 UNFOLLOW <username>
 LIST
 TIMELINE
=====================================
```

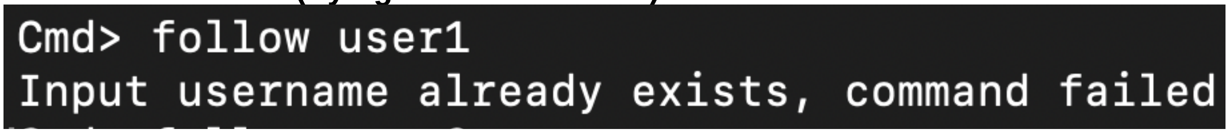
  - **./tsc -u user1 (user1 is already logged in, but trying to login again)**


suryacherukuri — saisurya@csce-nguyen-s1: ~/mp1_skeleton — ssh sai...

```
saisurya@csce-nguyen-s1:~/mp1_skeleton$ ./tsc -u user1
Logging Initialized. Client starting...
Login failed: User user1 already logged in
connection failed: -1
saisurya@csce-nguyen-s1:~/mp1_skeleton$ 
```

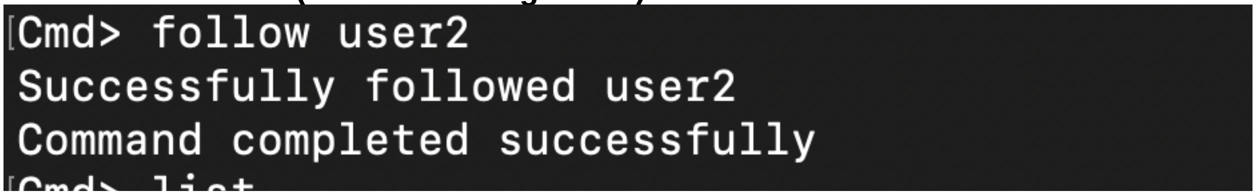
TestCase 2

- **Conditions**:
  - **user1 and user2 are already logged in.**
- **Command & Output**:
  - **follow user1 (trying to follow ourself)**



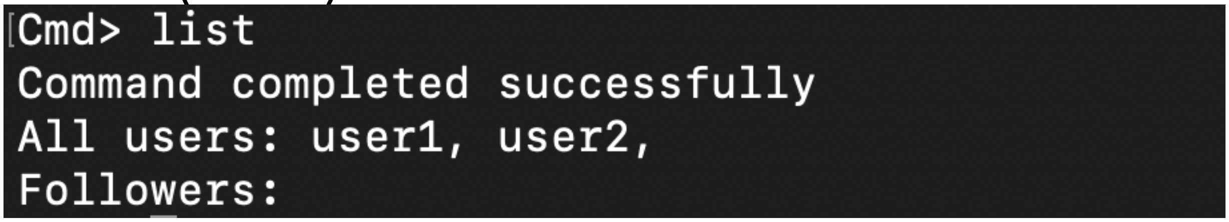
```
Cmd> follow user1
Input username already exists, command failed
```

  - **follow user2 (user1 following user2)**



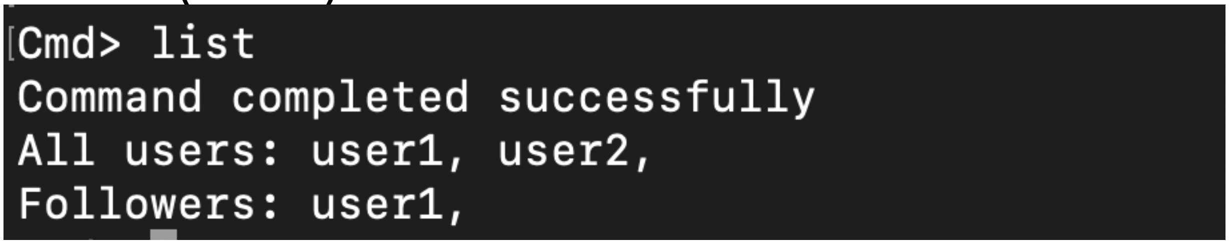
```
[Cmd> follow user2
Successfully followed user2
Command completed successfully
```

  - **LIST (for user1)**



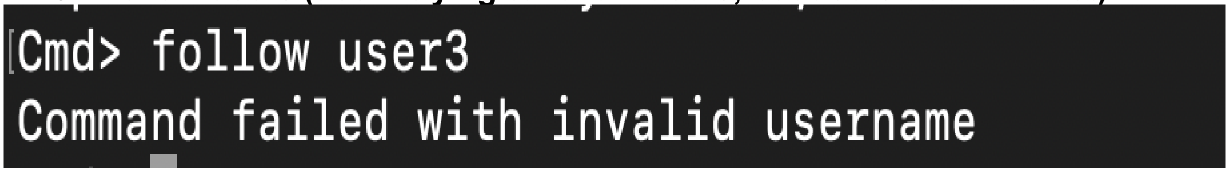
```
[Cmd> list
Command completed successfully
All users: user1, user2,
Followers:
```

  - **LIST (for user2)**



```
[Cmd> list
Command completed successfully
All users: user1, user2,
Followers: user1,
```

  - **Follow user3 (user1 trying to follow user3, but user3 doesn't exist)**



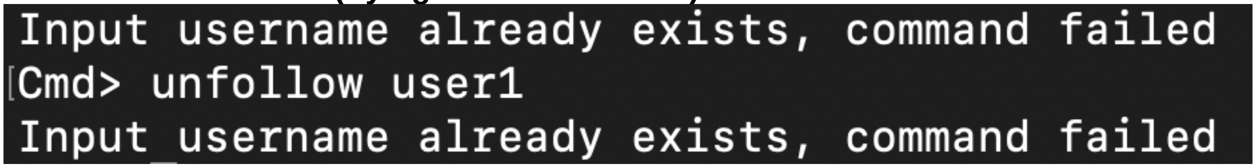
```
[Cmd> follow user3
Command failed with invalid username
```

TestCase 3

- **Conditions**:

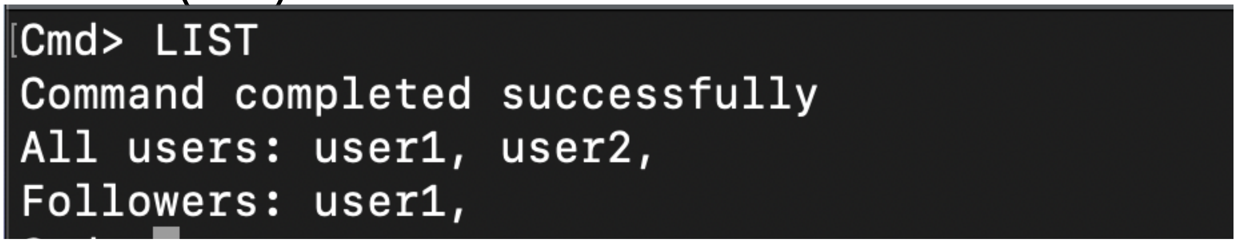
- **user1 and user2 are already logged in.**
- **user1 following user2**
- **user2 following user1**

- **Command & Output**:
  - **unfollow user1 (trying to follow ourself)**



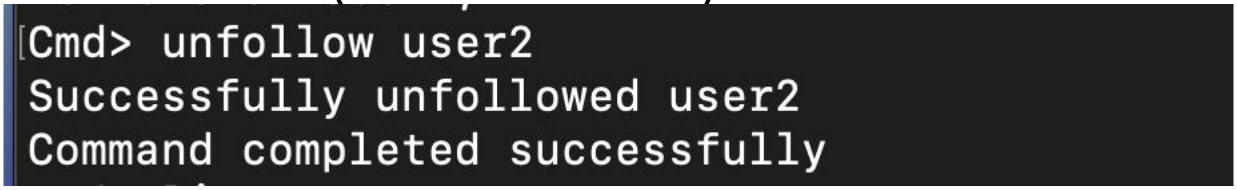
```
Input username already exists, command failed
[Cmd> unfollow user1
Input username already exists, command failed
```

  - **LIST (user2)**



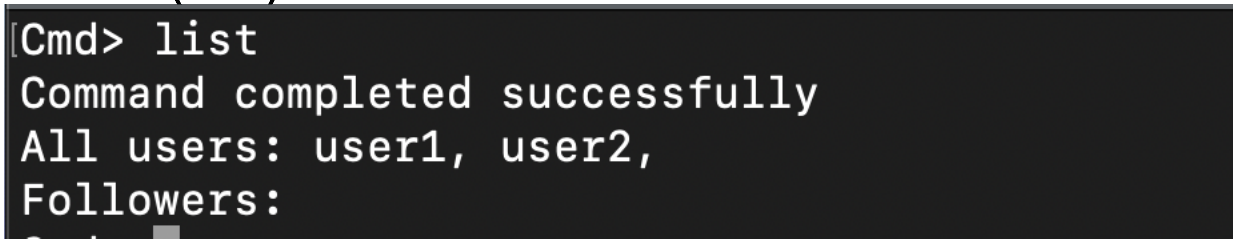
```
[Cmd> LIST
Command completed successfully
All users: user1, user2,
Followers: user1,
```

  - **unfollow user2 (user1 unfollowed user2)**



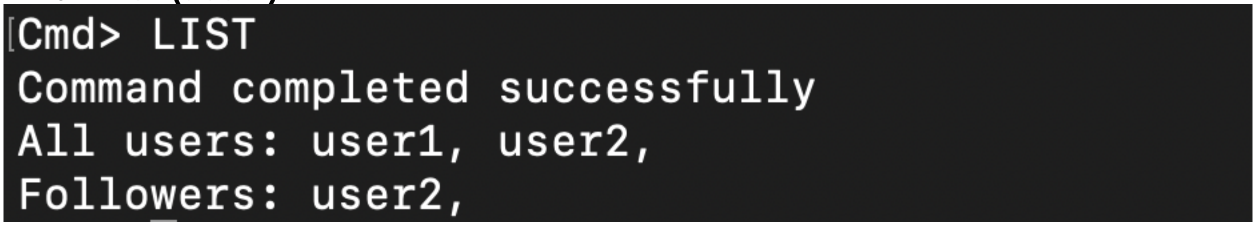
```
[Cmd> unfollow user2
Successfully unfollowed user2
Command completed successfully
```

  - **LIST(user2)**



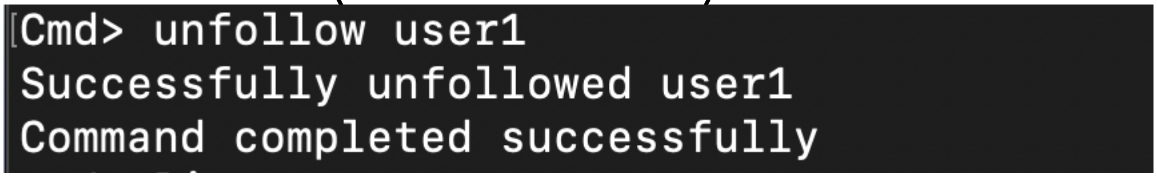
```
[Cmd> list
Command completed successfully
All users: user1, user2,
Followers:
```

  - **LIST(user1)**



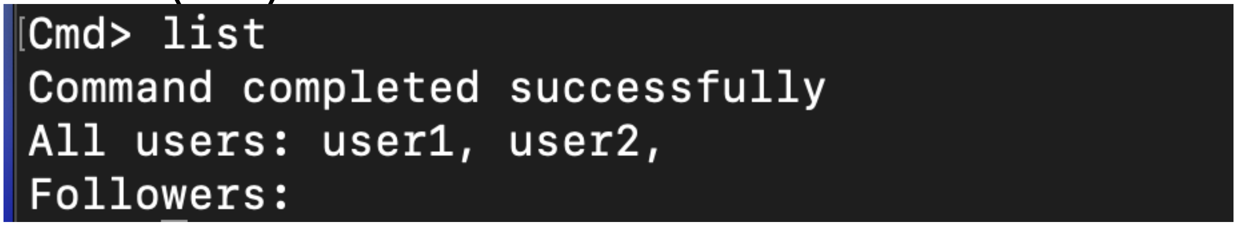
```
[Cmd> LIST
Command completed successfully
All users: user1, user2,
Followers: user2,
```

- **unfollow user1 (user2 unfollowed user1)**



```
[Cmd> unfollow user1
Successfully unfollowed user1
Command completed successfully
```

- **LIST(user1)**

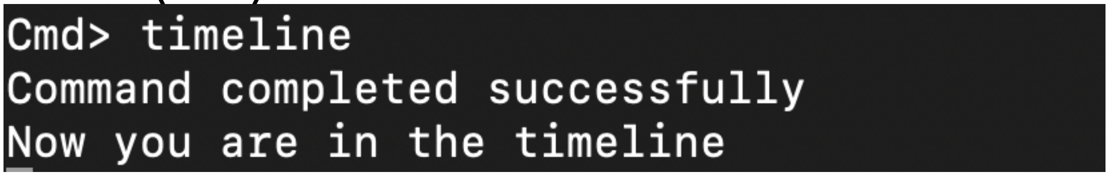


```
[Cmd> list
Command completed successfully
All users: user1, user2,
Followers:
```

TestCase 4

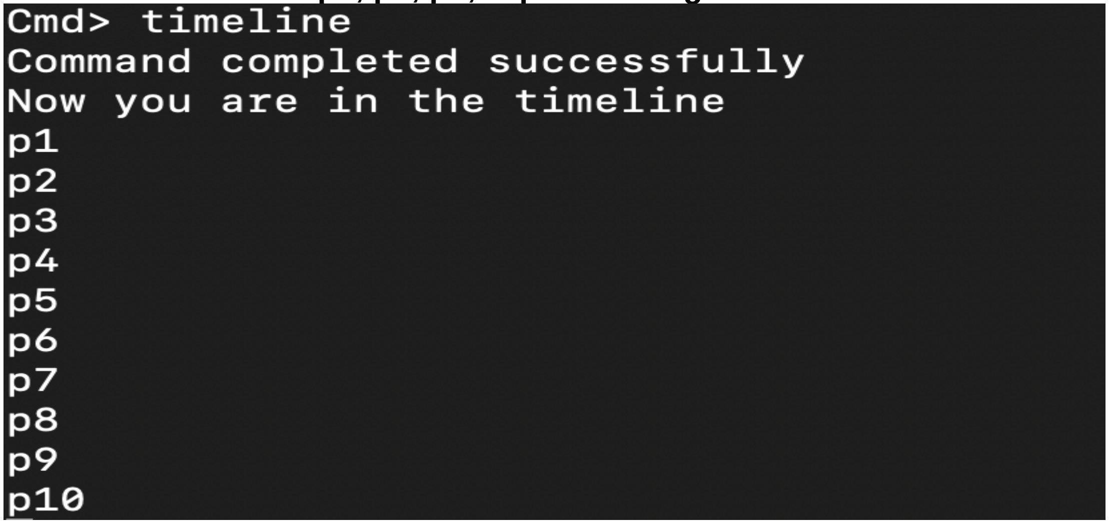
- **Conditions**:
  - **User1, user2, and user3 are already logged in.**
  - **user1 following user2**
  - **user2 following user1**
  - **user3 is not following user1 yet**

- **Command & Output**:
  - **TIMELINE(user1)**



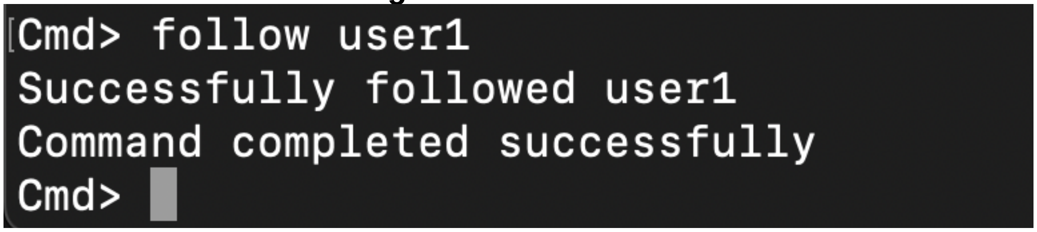
```
Cmd> timeline
Command completed successfully
Now you are in the timeline
```

- **user1 entered p1, p2, p3, ... p10 messages in his timeline**



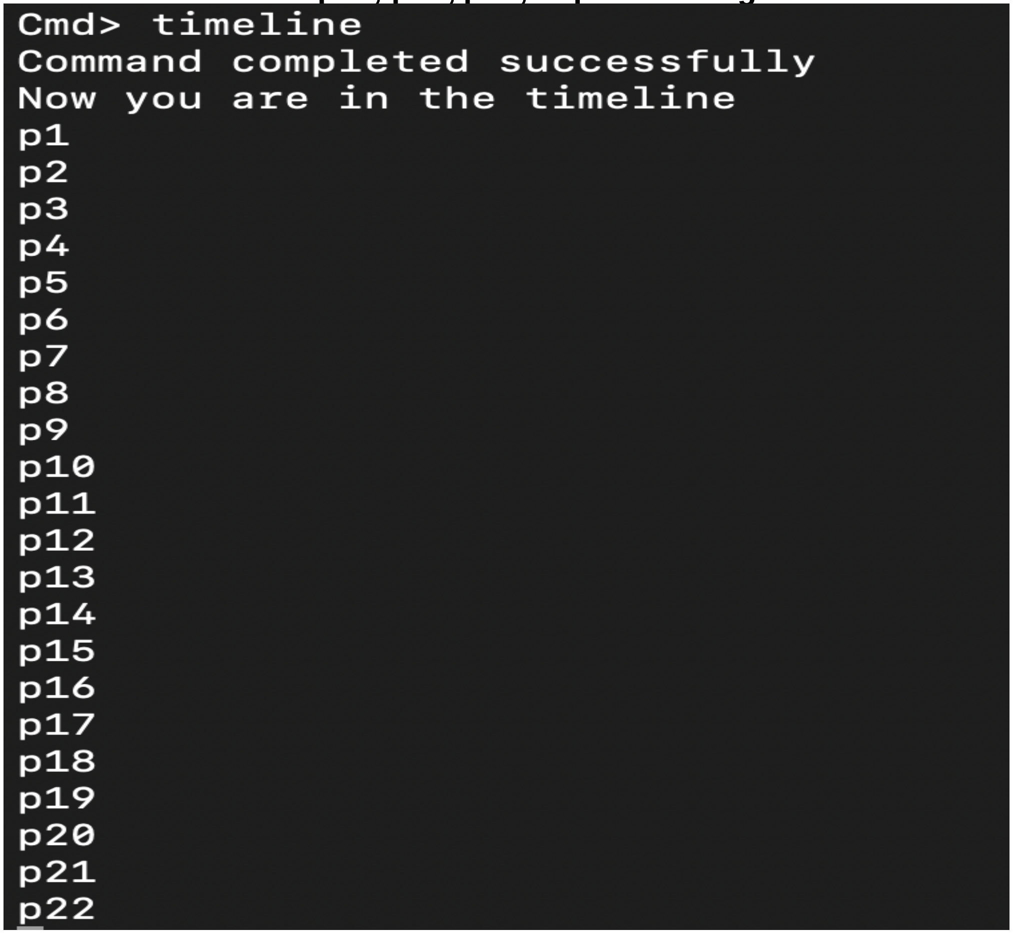
```
Cmd> timeline
Command completed successfully
Now you are in the timeline
p1
p2
p3
p4
p5
p6
p7
p8
p9
p10
```

- **user3 now following user1**



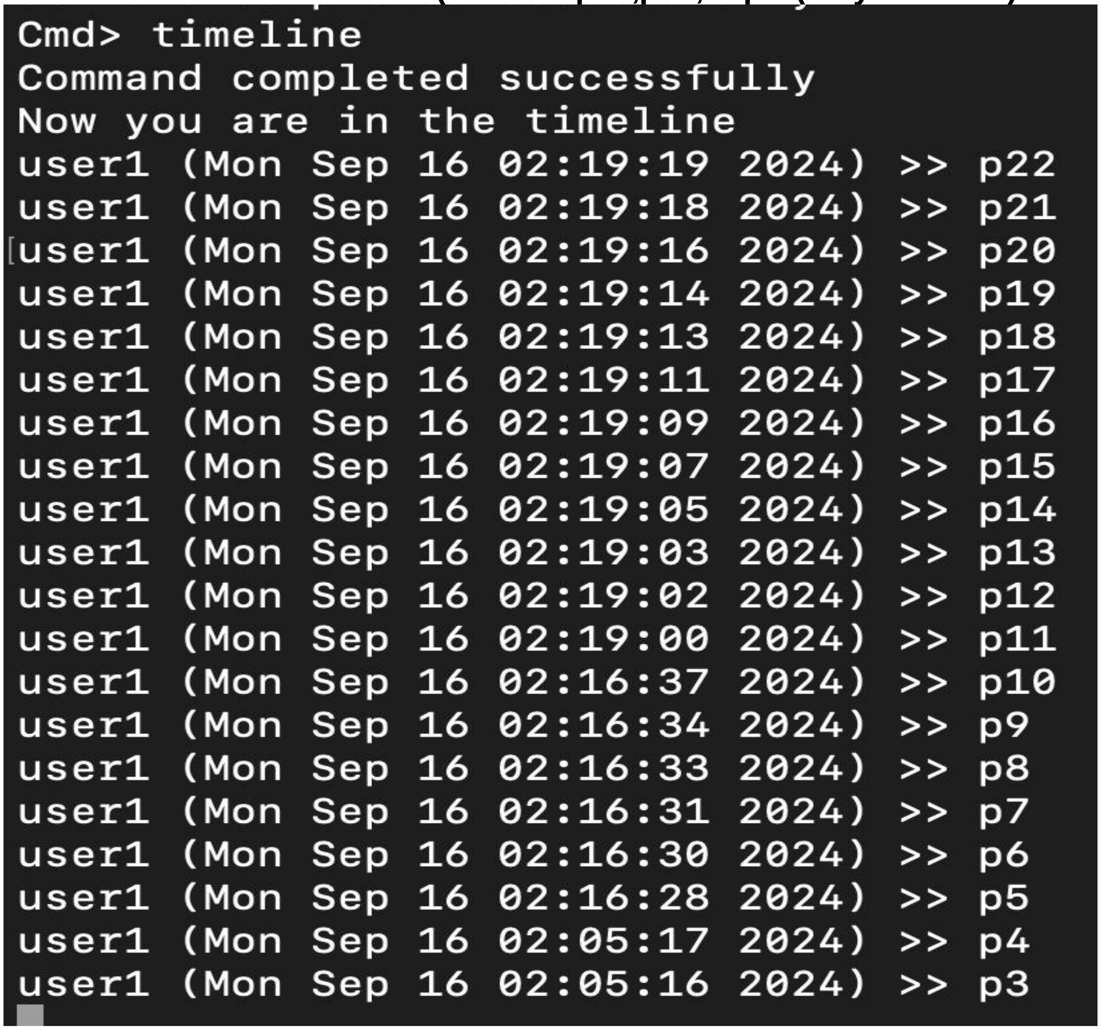
```
Cmd> follow user1
Successfully followed user1
Command completed successfully
Cmd>
```

- **user1 entered p11, p12, p13, ... p22 messages in his timeline**



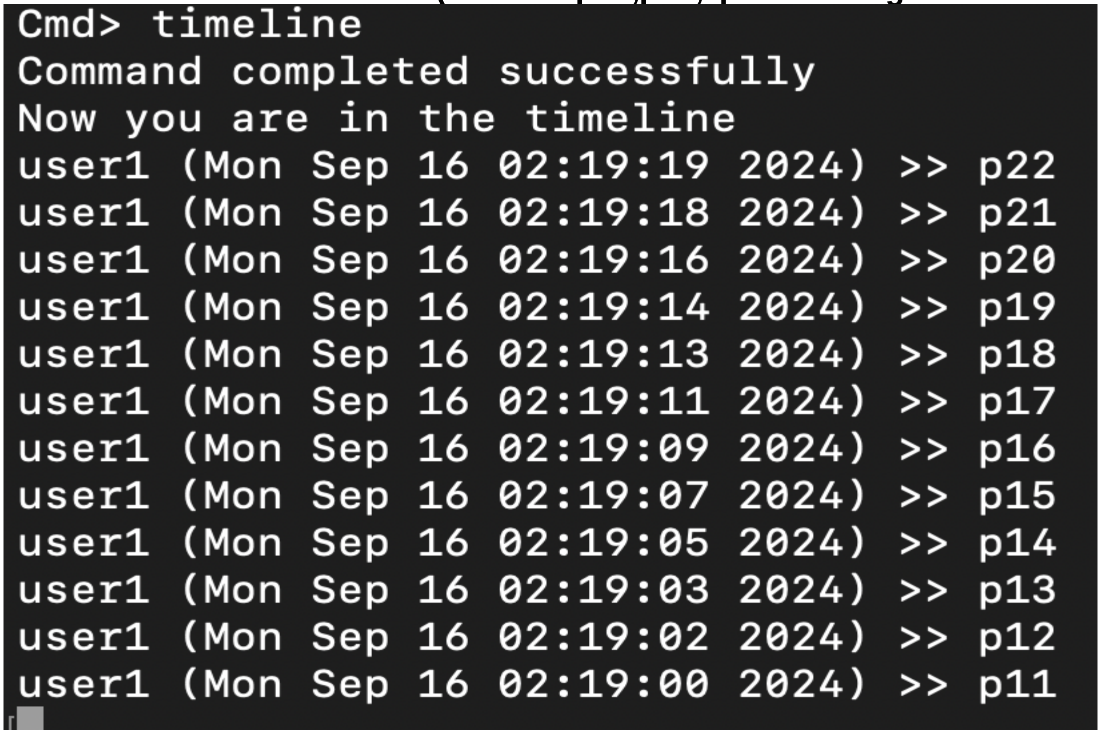
```
Cmd> timeline
Command completed successfully
Now you are in the timeline
p1
p2
p3
p4
p5
p6
p7
p8
p9
p10
p11
p12
p13
p14
p15
p16
p17
p18
p19
p20
p21
p22
```

- **user2 entered timeline(he sees p22,p21,.....p3 (only 20 latest) messages in his timeline)**



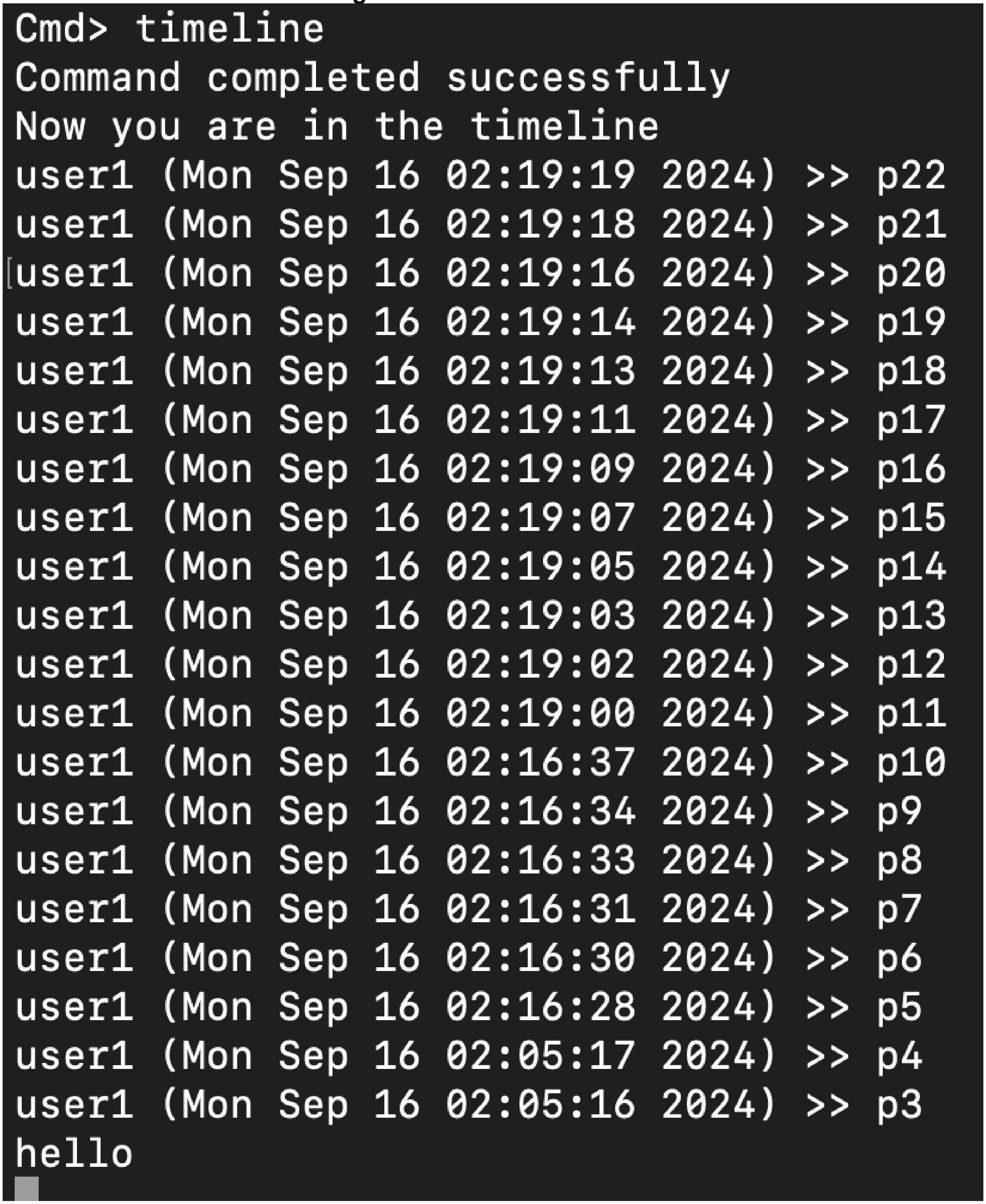
```
Cmd> timeline
Command completed successfully
Now you are in the timeline
user1 (Mon Sep 16 02:19:19 2024) >> p22
user1 (Mon Sep 16 02:19:18 2024) >> p21
user1 (Mon Sep 16 02:19:16 2024) >> p20
user1 (Mon Sep 16 02:19:14 2024) >> p19
user1 (Mon Sep 16 02:19:13 2024) >> p18
user1 (Mon Sep 16 02:19:11 2024) >> p17
user1 (Mon Sep 16 02:19:09 2024) >> p16
user1 (Mon Sep 16 02:19:07 2024) >> p15
user1 (Mon Sep 16 02:19:05 2024) >> p14
user1 (Mon Sep 16 02:19:03 2024) >> p13
user1 (Mon Sep 16 02:19:02 2024) >> p12
user1 (Mon Sep 16 02:19:00 2024) >> p11
user1 (Mon Sep 16 02:16:37 2024) >> p10
user1 (Mon Sep 16 02:16:34 2024) >> p9
user1 (Mon Sep 16 02:16:33 2024) >> p8
user1 (Mon Sep 16 02:16:31 2024) >> p7
user1 (Mon Sep 16 02:16:30 2024) >> p6
user1 (Mon Sep 16 02:16:28 2024) >> p5
user1 (Mon Sep 16 02:05:17 2024) >> p4
user1 (Mon Sep 16 02:05:16 2024) >> p3
```

- **user3 entered timeline(he sees p22,p21,..p11 messages in his timeline)**



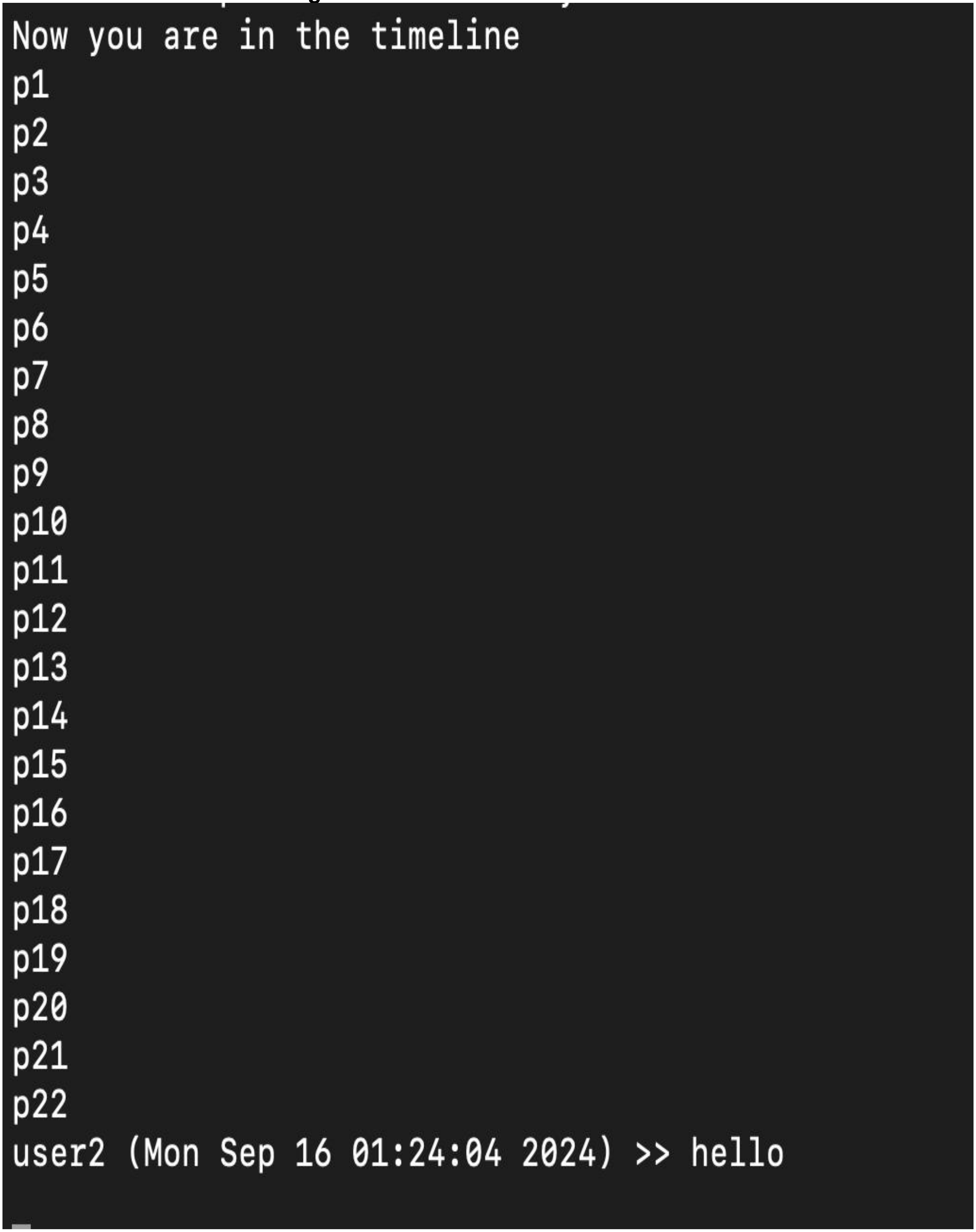
```
Cmd> timeline
Command completed successfully
Now you are in the timeline
user1 (Mon Sep 16 02:19:19 2024) >> p22
user1 (Mon Sep 16 02:19:18 2024) >> p21
user1 (Mon Sep 16 02:19:16 2024) >> p20
user1 (Mon Sep 16 02:19:14 2024) >> p19
user1 (Mon Sep 16 02:19:13 2024) >> p18
user1 (Mon Sep 16 02:19:11 2024) >> p17
user1 (Mon Sep 16 02:19:09 2024) >> p16
user1 (Mon Sep 16 02:19:07 2024) >> p15
user1 (Mon Sep 16 02:19:05 2024) >> p14
user1 (Mon Sep 16 02:19:03 2024) >> p13
user1 (Mon Sep 16 02:19:02 2024) >> p12
user1 (Mon Sep 16 02:19:00 2024) >> p11
```

- **user2 enters “hello” msg in his timeline**



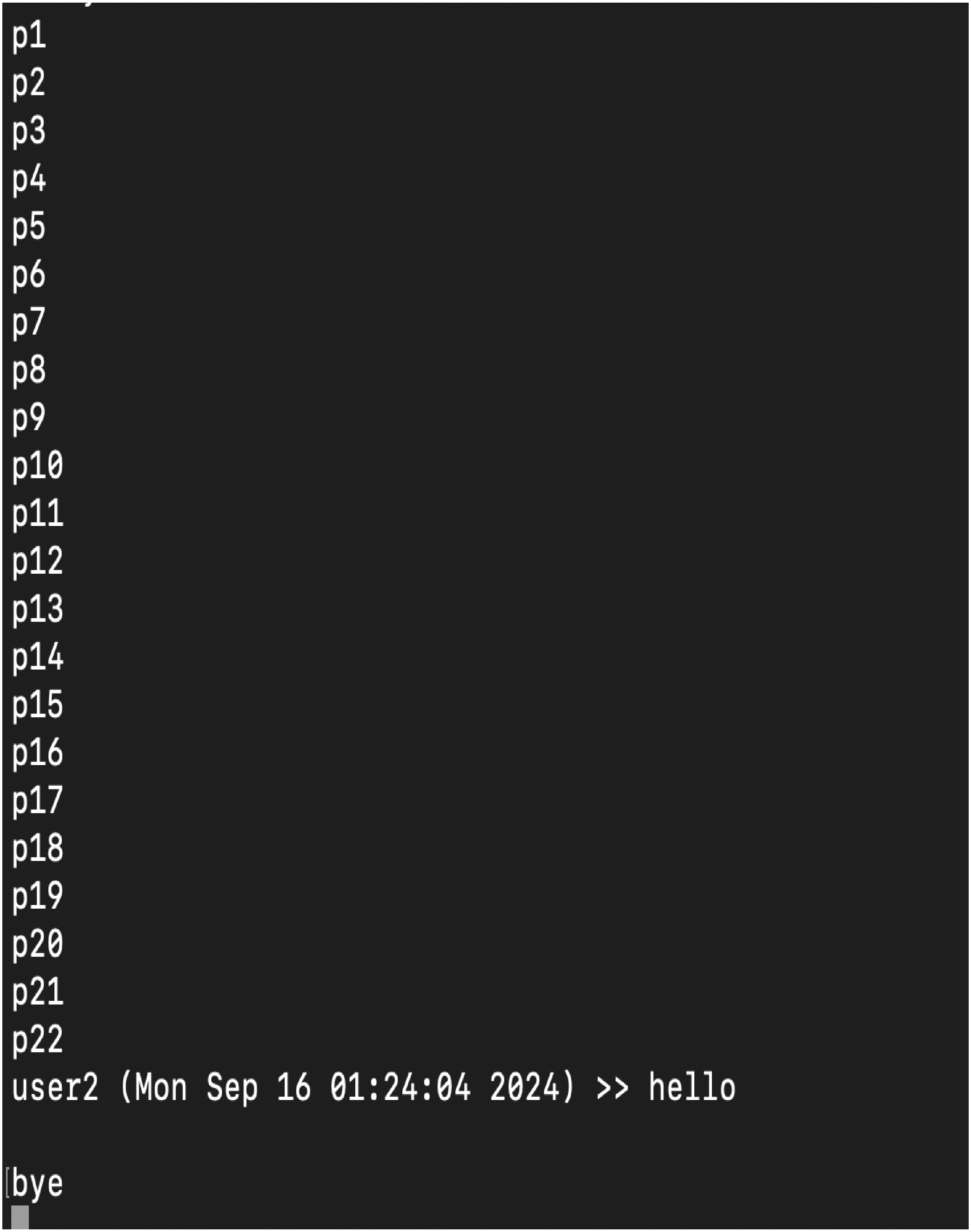
```
Cmd> timeline
Command completed successfully
Now you are in the timeline
user1 (Mon Sep 16 02:19:19 2024) >> p22
user1 (Mon Sep 16 02:19:18 2024) >> p21
user1 (Mon Sep 16 02:19:16 2024) >> p20
user1 (Mon Sep 16 02:19:14 2024) >> p19
user1 (Mon Sep 16 02:19:13 2024) >> p18
user1 (Mon Sep 16 02:19:11 2024) >> p17
user1 (Mon Sep 16 02:19:09 2024) >> p16
user1 (Mon Sep 16 02:19:07 2024) >> p15
user1 (Mon Sep 16 02:19:05 2024) >> p14
user1 (Mon Sep 16 02:19:03 2024) >> p13
user1 (Mon Sep 16 02:19:02 2024) >> p12
user1 (Mon Sep 16 02:19:00 2024) >> p11
user1 (Mon Sep 16 02:16:37 2024) >> p10
user1 (Mon Sep 16 02:16:34 2024) >> p9
user1 (Mon Sep 16 02:16:33 2024) >> p8
user1 (Mon Sep 16 02:16:31 2024) >> p7
user1 (Mon Sep 16 02:16:30 2024) >> p6
user1 (Mon Sep 16 02:16:28 2024) >> p5
user1 (Mon Sep 16 02:05:17 2024) >> p4
user1 (Mon Sep 16 02:05:16 2024) >> p3
hello
```

- **user1 sees “hello” msg in his timeline**



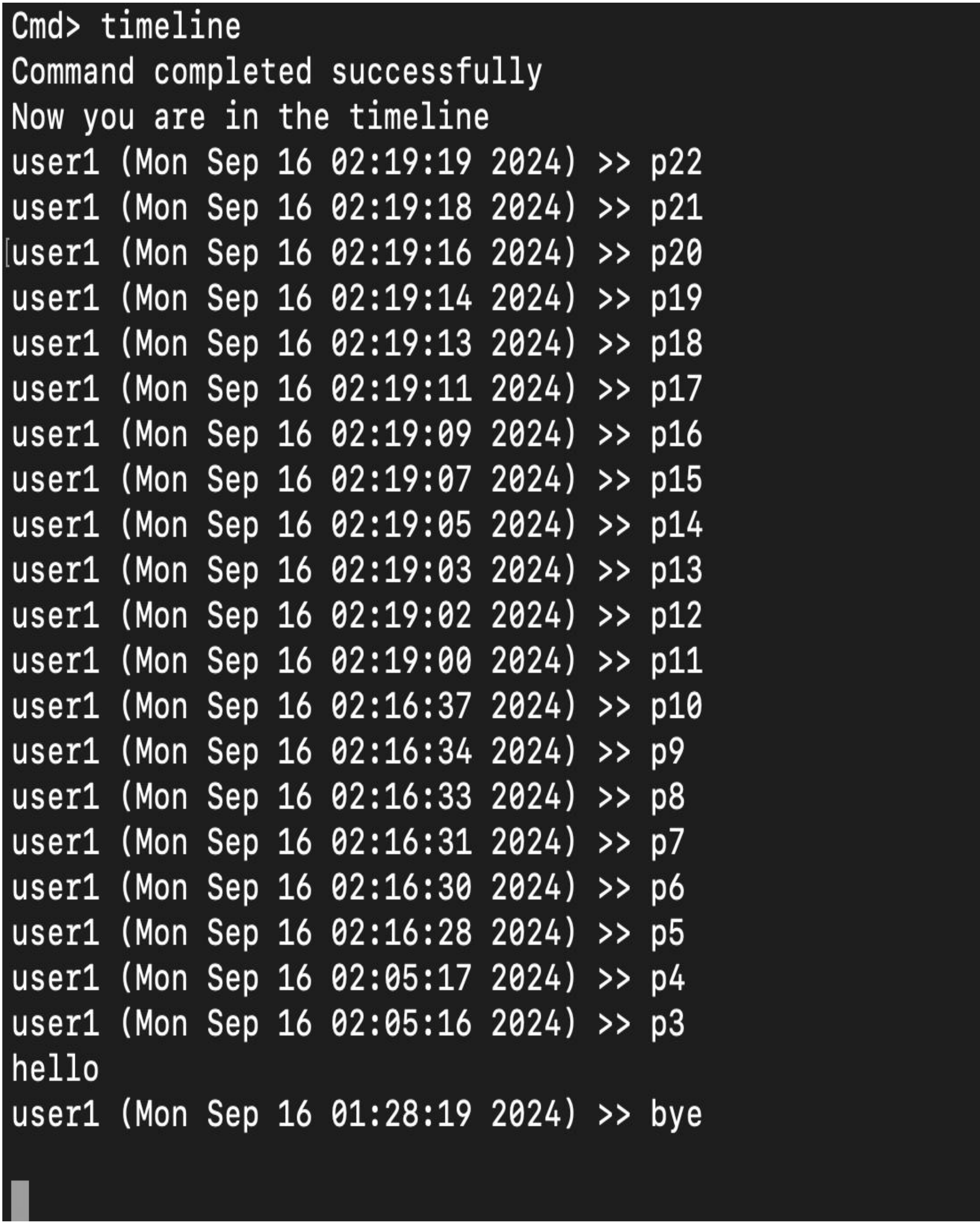
```
Now you are in the timeline
p1
p2
p3
p4
p5
p6
p7
p8
p9
p10
p11
p12
p13
p14
p15
p16
p17
p18
p19
p20
p21
p22
user2 (Mon Sep 16 01:24:04 2024) >> hello
```

- **user1 entered “bye” msg in his timeline**



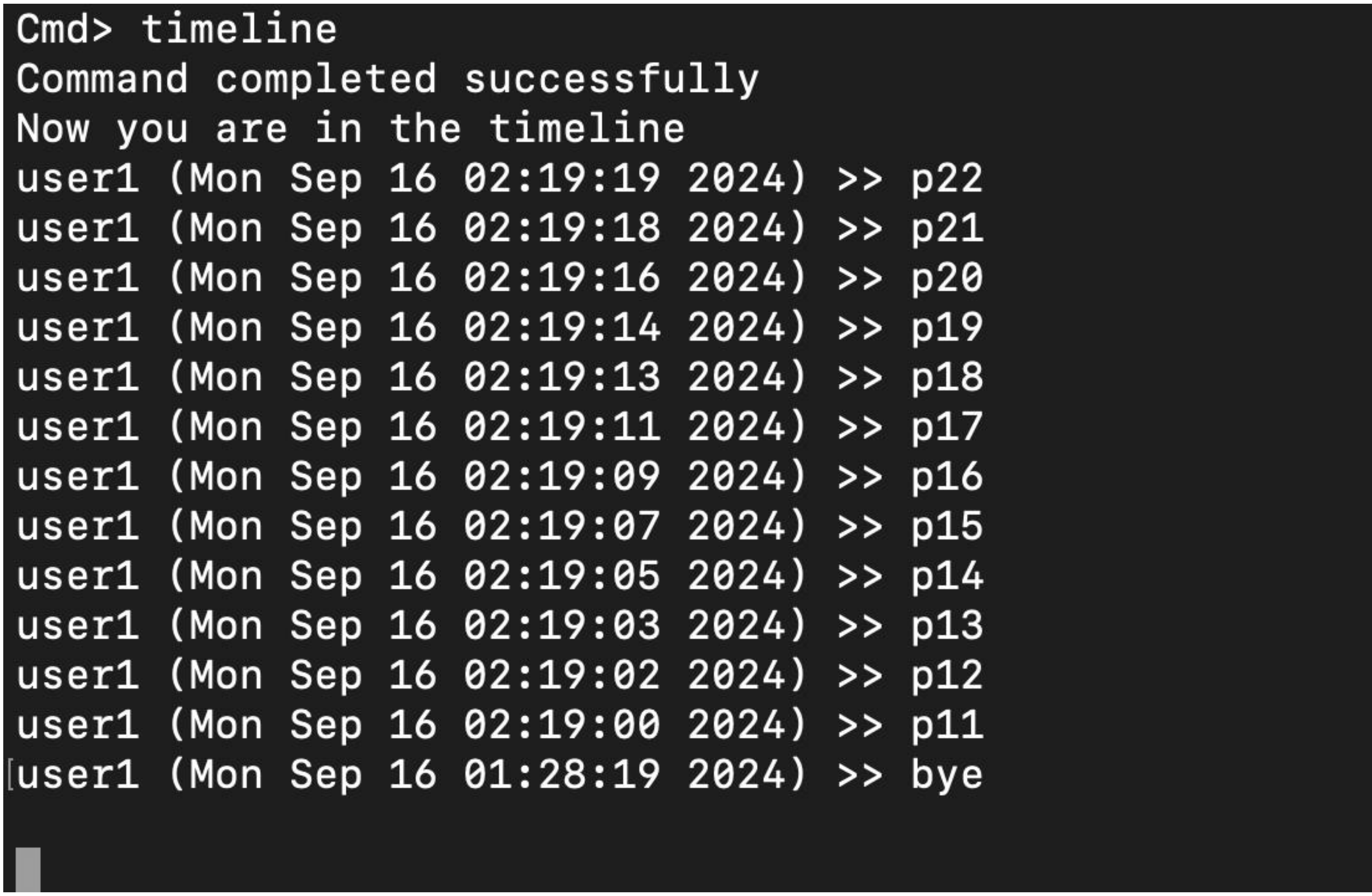
```
p1
p2
p3
p4
p5
p6
p7
p8
p9
p10
p11
p12
p13
p14
p15
p16
p17
p18
p19
p20
p21
p22
user2 (Mon Sep 16 01:24:04 2024) >> hello

bye
```

- **user2 sees “bye” msg in his timeline**



```
Cmd> timeline
Command completed successfully
Now you are in the timeline
user1 (Mon Sep 16 02:19:19 2024) >> p22
user1 (Mon Sep 16 02:19:18 2024) >> p21
user1 (Mon Sep 16 02:19:16 2024) >> p20
user1 (Mon Sep 16 02:19:14 2024) >> p19
user1 (Mon Sep 16 02:19:13 2024) >> p18
user1 (Mon Sep 16 02:19:11 2024) >> p17
user1 (Mon Sep 16 02:19:09 2024) >> p16
user1 (Mon Sep 16 02:19:07 2024) >> p15
user1 (Mon Sep 16 02:19:05 2024) >> p14
user1 (Mon Sep 16 02:19:03 2024) >> p13
user1 (Mon Sep 16 02:19:02 2024) >> p12
user1 (Mon Sep 16 02:19:00 2024) >> p11
user1 (Mon Sep 16 02:16:37 2024) >> p10
user1 (Mon Sep 16 02:16:34 2024) >> p9
user1 (Mon Sep 16 02:16:33 2024) >> p8
user1 (Mon Sep 16 02:16:31 2024) >> p7
user1 (Mon Sep 16 02:16:30 2024) >> p6
user1 (Mon Sep 16 02:16:28 2024) >> p5
user1 (Mon Sep 16 02:05:17 2024) >> p4
user1 (Mon Sep 16 02:05:16 2024) >> p3
hello
user1 (Mon Sep 16 01:28:19 2024) >> bye
```

- **user3 sees “bye” msg in his timeline**



```
Cmd> timeline
Command completed successfully
Now you are in the timeline
user1 (Mon Sep 16 02:19:19 2024) >> p22
user1 (Mon Sep 16 02:19:18 2024) >> p21
user1 (Mon Sep 16 02:19:16 2024) >> p20
user1 (Mon Sep 16 02:19:14 2024) >> p19
user1 (Mon Sep 16 02:19:13 2024) >> p18
user1 (Mon Sep 16 02:19:11 2024) >> p17
user1 (Mon Sep 16 02:19:09 2024) >> p16
user1 (Mon Sep 16 02:19:07 2024) >> p15
user1 (Mon Sep 16 02:19:05 2024) >> p14
user1 (Mon Sep 16 02:19:03 2024) >> p13
user1 (Mon Sep 16 02:19:02 2024) >> p12
user1 (Mon Sep 16 02:19:00 2024) >> p11
[user1 (Mon Sep 16 01:28:19 2024) >> bye
```

Conclusion:

- **Tested all the testcases provided in the excel sheet and all output the correct result.**
- **Tested with large number of clients (10).**
- **Tested on TAMU server so, I changed the home path for pkg config path**

```
export PKG_CONFIG_PATH =
/home/grads/s/saisurya/.local/lib/pkgconfig:/home/grads/s/saisurya/grpc/third_party/re2:/home/grads/s/saisurya/.local/share/p kgconfig/
```

- **However, I changed the config path to /home/csce662 directory while submitting.**
  export PKG_CONFIG_PATH = /home/csce662/.local/lib/pkgconfig:/home/csce662/grpc/third_party/re2:/home/csce662/.local/share/pkgconfig/