CSCE 438 MP1 Design Document

Jonathan Lor

Client

- The means for the user to interface with the system.
    - Accepts and processes commands entered by the user.
    - Connects to the server via a stub, an abstraction which allows the client to call server methods as if they were on the client itself.
    - Most client methods return an IReply object specifying the status of the request and any content it contains.

connectTo()

- Creates a channel with the given (or default) hostname and port.
- Creates a new stub and assigns it to the `stub_` member variable.
- Calls `Login()` to attempt a login.
- Returns a 1 if successful, and -1 otherwise

processCommand()

- Splits the user input into command and argument variables
- Handles different commands accordingly by calling the appropriate functions
- Returns an IReply object specifying the status and result of the command

List()

- Lists all active users and followers of the client who entered the command

- Calls the server side List function using the `stub_`
- Populates and returns the IReply object with the received information

Follow()

- Attempts to process a follow request by calling the server-side Follow() method using the `stub_` .
- Depending on the server response, relay to the user the result of the call
- Return and IReply object with all relevant information

UnFollow()

- Nearly identical to `Follow()` in the client-side implementation.

Login()

- Attempts to connect to the server via `stub_` .
- Returns IReply of whether or not the connection succeeded.

Timeline()

- Multithreaded handling of simultaneous read and writes.
- Defines two threads, reader & writer to handle timeline viewing and posting, respectively.
- Interfaces with the server through a ClientReaderWriter object.

Server

- The logic, in-memory (client_db) and persistent (timeline .txt files) storage for the SNS application.
- Member functions accept pointers to request and replies as defined in the `sns.proto` file.

- Member functions return grpc Status, which allow the client methods to relay the correct information to the user.

List()

- Upon receiving a List request from the client, traverses client_db and the requesting user's followers vector and populates the list_reply object
- Returns OK status if successful

Follow()

- Receives a request from a user, which specifies another user to follow in its arguments.
- Upon validating the argument, attempts to perform the follow operation
- The follow operation adds the requesting user, u1 to the requested user, u2's followers vector
- Likewise, u2 is added to u1's following vector.
- Returns OK status if successful.

UnFollow()

- Identical to Follow() in the way request and arguments are validated
- The UnFollow operation attempts to find the requesting user, u1 in the requested user, u2's followers vector, and removes if found
- Likewise, u2 is erased from u1's following vector
- Returns OK status if successful

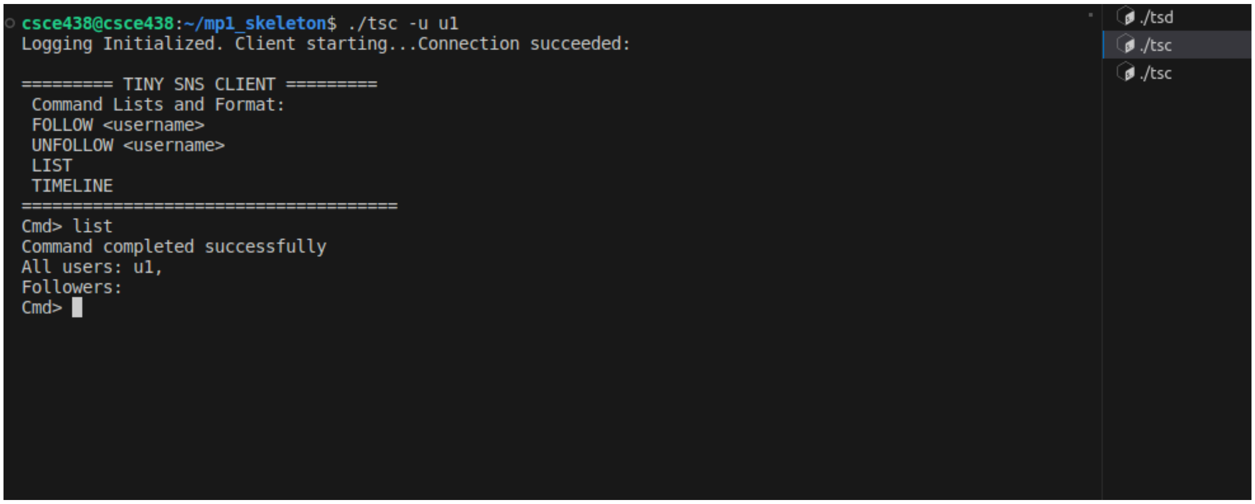
Login()

- Accepts request, and checks if the username exists in client_db
- Adds user if they don't currently exist in client_db
- Does not allow duplicate users.
- Returns OK status if successful.

Timeline()

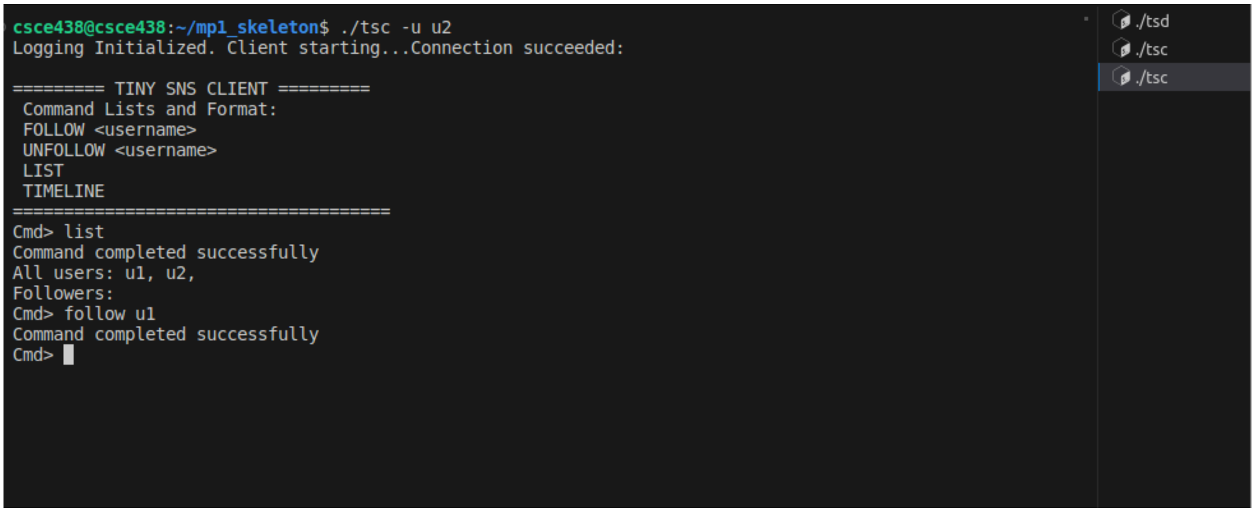
- Note: Didn’t complete implementation on the server-side
- Manages timeline .txt files.
- Read and writes to them accordingly using ServerReaderWriter.
- Returns OK status if successful.

Screenshots - List, Follow, Unfollow



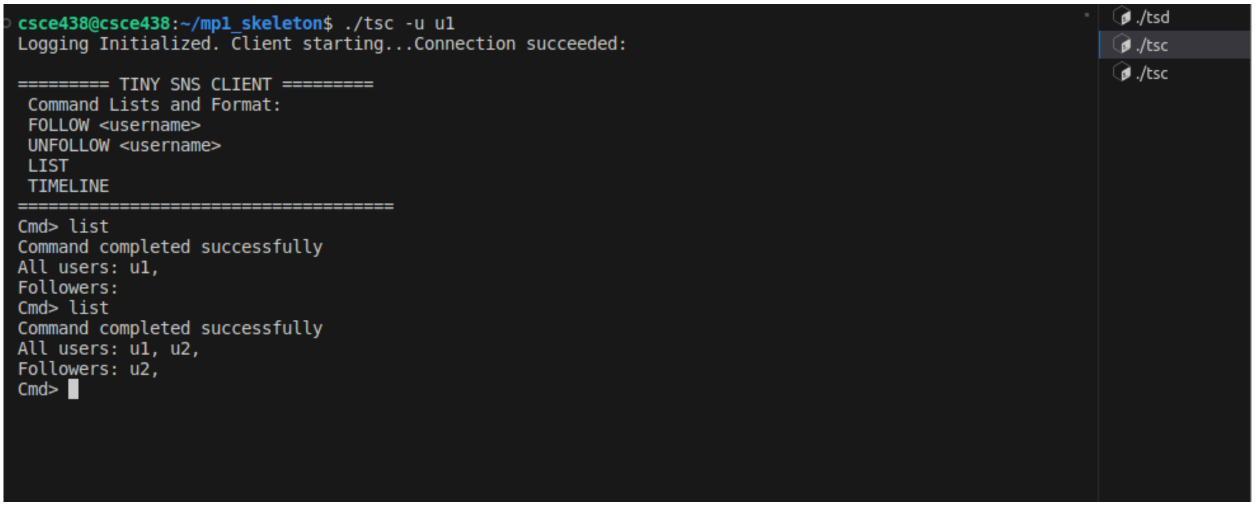
```
csce438@csce438:~/mp1_skeleton$ ./tsc -u u1
Logging Initialized. Client starting...Connection succeeded:

========= TINY SNS CLIENT =========
 Command Lists and Format:
 FOLLOW <username>
 UNFOLLOW <username>
 LIST
 TIMELINE
=====================================
Cmd> list
Command completed successfully
All users: u1,
Followers:
Cmd> 
```



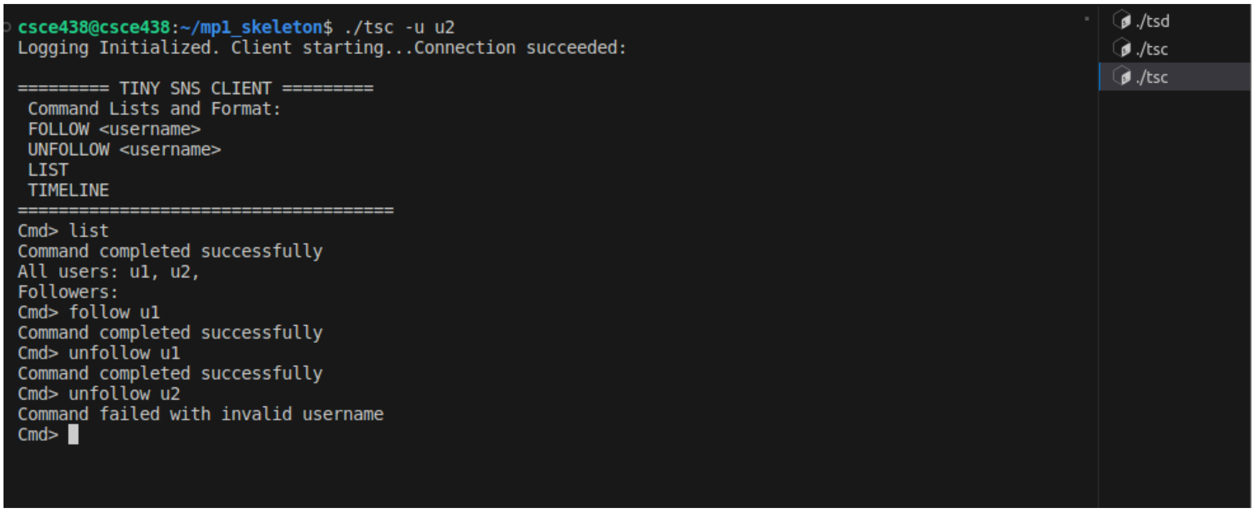
```
csce438@csce438:~/mp1_skeleton$ ./tsc -u u2
Logging Initialized. Client starting...Connection succeeded:

========= TINY SNS CLIENT =========
 Command Lists and Format:
 FOLLOW <username>
 UNFOLLOW <username>
 LIST
 TIMELINE
=====================================
Cmd> list
Command completed successfully
All users: u1, u2,
Followers:
Cmd> follow u1
Command completed successfully
Cmd> 
```



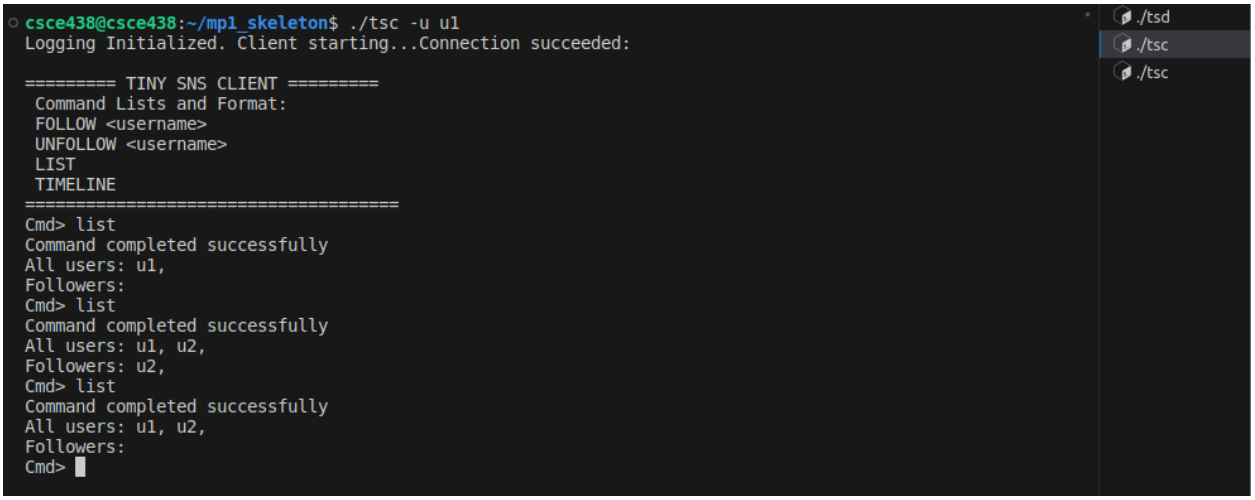
```
csce438@csce438:~/mp1_skeleton$ ./tsc -u u1
Logging Initialized. Client starting...Connection succeeded:

========= TINY SNS CLIENT =========
 Command Lists and Format:
 FOLLOW <username>
 UNFOLLOW <username>
 LIST
 TIMELINE
=====================================
Cmd> list
Command completed successfully
All users: u1,
Followers:
Cmd> list
Command completed successfully
All users: u1, u2,
Followers: u2,
Cmd> 
```



```
csce438@csce438:~/mp1_skeleton$ ./tsc -u u2
Logging Initialized. Client starting...Connection succeeded:

========= TINY SNS CLIENT =========
 Command Lists and Format:
 FOLLOW <username>
 UNFOLLOW <username>
 LIST
 TIMELINE
=====================================
Cmd> list
Command completed successfully
All users: u1, u2,
Followers:
Cmd> follow u1
Command completed successfully
Cmd> unfollow u1
Command completed successfully
Cmd> unfollow u2
Command failed with invalid username
Cmd> 
```



```
csce438@csce438:~/mp1_skeleton$ ./tsc -u u1
Logging Initialized. Client starting...Connection succeeded:

========= TINY SNS CLIENT =========
 Command Lists and Format:
 FOLLOW <username>
 UNFOLLOW <username>
 LIST
 TIMELINE
=====================================
Cmd> list
Command completed successfully
All users: u1,
Followers:
Cmd> list
Command completed successfully
All users: u1, u2,
Followers: u2,
Cmd> list
Command completed successfully
All users: u1, u2,
Followers:
Cmd> 
```