Use your preferred language to build a program to support following activities through command line interface from standard input/output.
* Login. create a user account if not exist
* Send. send a message to another user in the system.
* Read. read a new message to this user.
* Reply. reply to the current read message.
* Forward. forward the current read message to another user.
* Broadcast. send a broadcast message to all users in the system.

The program is not required to have a persistent state so when it starts, you can assume it is in a clean condition with no existing data.

a sample interaction is below (>- your input, <- program output):

```
>- send def “message”
<- error: please login first.
>- login abc
<- abc logged in.
>- login def
<- def logged in.
>- send abc "test message 1"
<- message sent.
>- send abc "test message 2"
<- message sent.
-> login abc
<- abc logged in, 2 new messages.
>- read
<- from def: “test message 1”
>- reply "read"
<- message sent to def
>- reply “read again”
<- message sent to def
>- read
<- from def: “test message 2”
>- forward def
<- message forwarded to def
>- broadcast “hello world”
>- login def
<- def logged in, 4 new messages.
>- read
<- from abc: “read”
>- read
<- from abc: “read again”
>- read
<- from def, abc: “test message 2”
```

>- read
<- from abc: “hello world”
>- send fgh “test message 3”
<- error: User does not exist.

Please provide the following when you submit:
* program source code with necessary comments
* readme for how to build/run the program.
* test cases
* assumptions you made
Make sure your submission does not require installation of external dependency to run.

Bonus tasks:

1. Message threads.
   >- login abc
   <- abc logged in.
   >- login def
   <- def logged in.
   >- send abc “text message 1”
   <- message sent.
   >- send abc “text message 2”
   <- message sent.
   >- login abc
   <- abc logged in, 2 new messages.
   >- read
   <- message thread #1:
   from def: “text message 1”
   >- reply “okay”
   <- message sent to def
   >- reply “i read this”
   <- message sent to def
   >- read
   <- message thread #2:
   from def: “text message 2”
   >- reply “read”
   <- message sent to def
   >- login def
   <- def logged in, 3 new messages.
   >- read
   <- message thread #1:
   from def: “test message 1”
   from abc: “okay.”, ”I read this”

2. Viewing all messages available and choosing one to read.
   >- login abc
   <- abc logged in.
   >- login def
   <- def logged in.
   >- send abc “text message 1”
   <- message sent.
   >- send abc “text message 2”
   <- message sent.
   >- send abc “text message 3”
   <- message sent.
   >- login abc
   <- abc logged in, 3 new messages. Choose a number from 1 to 3 to pick the message to read. Pick 0 to cancel.
   >- read 2
   <- from def: “test message 2”
   >- read 0
   <- Message reading cancelled.
   >- read
   <- 2 new messages. Choose a number from 1 to 2 to pick the message to read. Pick 0 to cancel.
3. Combining bonus task 1 and 2 in similar fashion. You can define the input/output for this.
4. Multi-user support without a database, across different terminal sessions.