**1. Introduction:**

We developed a **Mail and Calendar system using client and server** program. In this we developed the mail and calendar systems in which services such as sending, receiving, replying, forwarding, deleting and filtering of mails in mail system and creating, modifying and deleting an event in calendar system. We implemented server program and the other is client program.

Initially, we created a configuration file where the user’s username, password and his mail and calendar structure is stored. When the server starts, the data is loaded from this file to the respective users’ structure.

When User Logouts from the System, the whole data from the structure is written into the respective mail files and calendar files of the user.

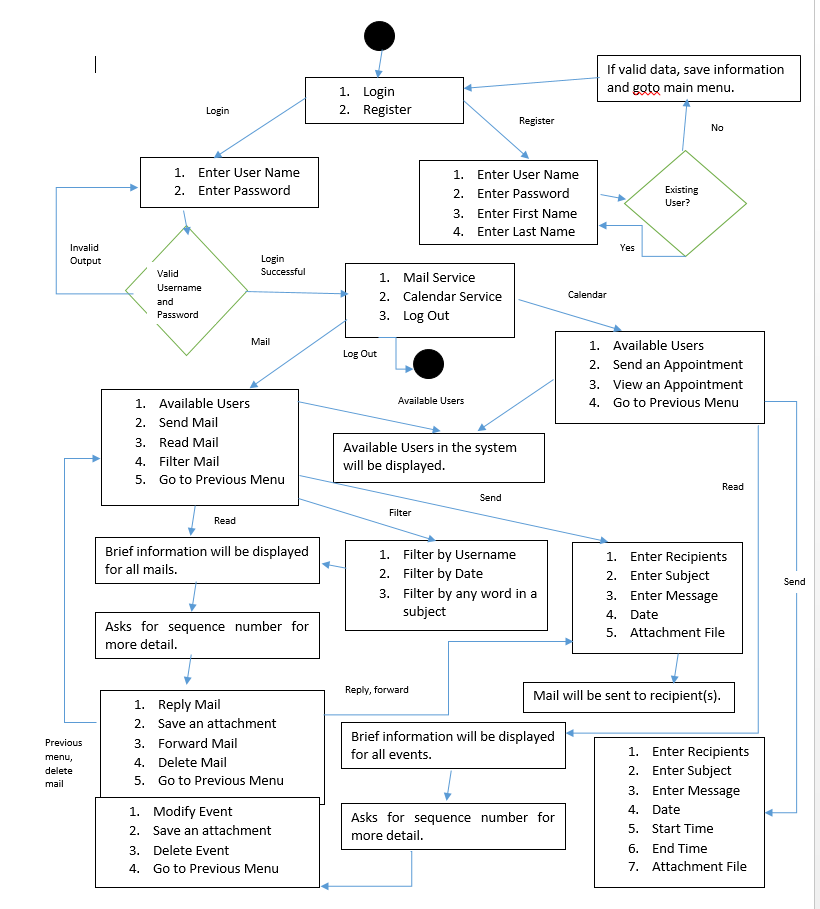
**2. Flow of the project:**

Figure 1 shows the building block of the project. When user calls the server using the port number. He is displayed with -

* **Signup**: If the user is not registered in the system. When user signs up, he is given with two files, one for calendar named cal\_username.txt and the other for mail mail\_username.txt. Usernames are unique, once a name assigned to a client cannot be assigned to another client.

While Signup, structures are assigned for the client, one for mail and other for calendar.

* **Login**: The user is an existing one and need to authenticate the system using his username and password. The validation is done by checking the value of username and password against the information stored in the configuration file.



**Figure 1: Building Blocks of the Project**

After user logins and enters into his account, User can either select

1. **Mail Services**: User can perform operations like
2. Available Users: This will retrieve the information from server structure and print the different usernames who already registered to the system.
3. Send Mail: The whole data which is to be sent to other users are taken into the buffer and is write to the server. On server, the respective mails files are opened based on the recipients list, and whole data is updated into the file structure.

* Multiple recipients are allowed using a token “;” . Server parses the whole name using the string and obtain different recipient names.
* The validations on username is done using the list that is retrieved during the available user call. The recipient names are checked with these values.
* The attachment should be in .txt format and should be one of the text filo on the local data.

1. Read Mail: The data is retrieved from the server structure to client structure. From client structure the data is retrieved and displayed. If any new mails arrive, a notification is shown that a message is arrived. The background thread checks for the new mails and will be contacting server frequently for new mails.

* When user selects view mail, he is displayed all his mails. Only a part of every mail is displayed. User is prompted with the sequence number. On entering the value in the range of displayed messages. It will display the entire content of the mail.

1. Filter Mail: Mails are filtered based on token taken from the user which can be date, subject and sender’s username. The token is sent to the function on client side and all the data is parsed, the token is checked with the data and respective data is retrieved and displayed to the client.
2. **Calendar Services**: User can perform operations like
3. Available Users: This will retrieve the information from server structure and print the different usernames who already registered to the system.
4. Send an Event: User can create an event and can invite other users using the calendar request. The data like the different users who are invited, place of event, date of the event, start and end time of the event information is taken from the sender and is sent to server. Server opens the respective calendar file of the recipient users and check if he is available at that time or not. If any of the recipient user is busy, the sender need to change the date or time or both of that event.

* A unique event id is generated for any event. Based on this id, deletion, modifying is performed.
* Sequence number of an event will be different for different client. It will be allocated to the user based on the available events in his system.
* For any particular event, the different users invited along with the sender’s name is stored in an array.
* The date, start time and end time of the event created should not be less than the local time. These values are validated using the functions on client side.

1. View Appointment: The calendar data is retrieved from the server and loaded in the client structure. On selecting this option, the data from the client structure is displayed to the user. A background thread continuously checks for the new events and will be contacting server for new events.

* Modifying the event can be done by any user who are invited. Once the event is modified which can be change of value in date, start time and end time. The modification is seen in all the recipient structures. The different recipient name can be retrieved from user list array stored in the structure. This modification is done based on event id and a new event id is allocated to the modified event. The old event is deleted form the system.
* The same is the case with delete. Once a user deletes an event from the system. The name of the user who deleted the event will be deleted from the rest of the recipient structure.

1. **Logout**: When user selects an option of logout, he can exit the system and all his mail data will be stored in mail\_username.txt file and calendar data is stored in cal\_username.txt

The structure of **mail\_username.txt** will be:

* If attachment is not present-

Sequence no| read | sender’s name| sender’s first name| sender’s last name| date and Time | subject | message| is Attachment (0)|~

* If attachment is present-

Sequence number | read| sender’s name| sender’s first name| sender’s last name| subject | message| is Attachment (1)| attachment name | attachment body |~

The structure of cal\_username.txt will be:

* If attachment is not present-

Event id | sequence number | read | sender’s name| sender’s first name| sender’s last name| start date | end date | place | subject | message | is Attachment (0)|~

* If attachment is present-

Event id | sequence number | read | sender’s name| sender’s first name| sender’s last name| start date | end date | place | subject | message | is Attachment (1) | attachment file name | attachment body | recipient list|~

* Read will be initially set to 0. When the client reads his mail or an event, it is automatically turned to 1.
* If the is Attachment value is 0, then the mail or an event does not have the attachment. Else 1, it contains the attachment.
* The sender’s last name and first name are automatically updated to the structure by the server.
* We have a third structure which loads the data from configuration file. The configuration file name is configuration.txt which is in the form-

Username | password | first name | last name | mail\_username.txt | cal\_username.txt | is logged in|~

If is logged in value = 0; user does not exists

Logged in value = 1; user is logged in

Logged in value = 2; user logout

* We have event.txt files which will store the event id and will be updated every time when a new event is created.

1. **Challenges:**

* Initially, we started the project the using files, but we changed the whole concept of files and started using structures. Created two structures one for mail and other for client. So, this change is one of the difficult task.
* We created a shared memory for calendar and mail. In the last month of the project we added an attribute called event id which uniquely defines an event. Adding this attribute to the shared memory was hard.
* In our project when the server starts, all the data from the file is retrieved and stored in the structure. This retrieval was done using strtok function which caused many errors in our system.

1. **Limitations:**

* Gracefully killing the server using the second client isn’t working in our system.

**User Guide**

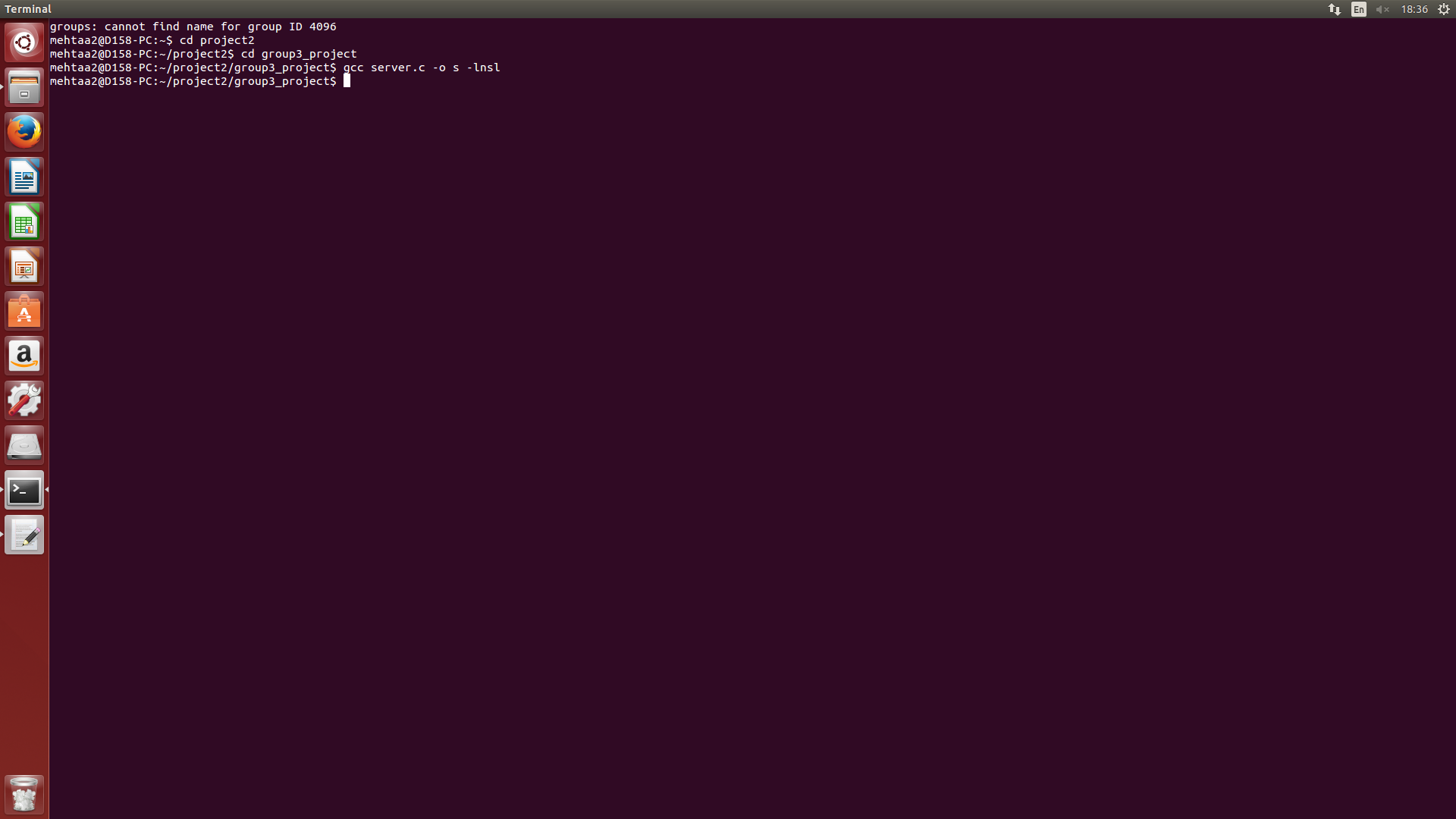
Step1: Download the .rar file which contains file client.c, server.c, event.txt and configuration.txt.

Step 2: Extract the rar files to the home folder.

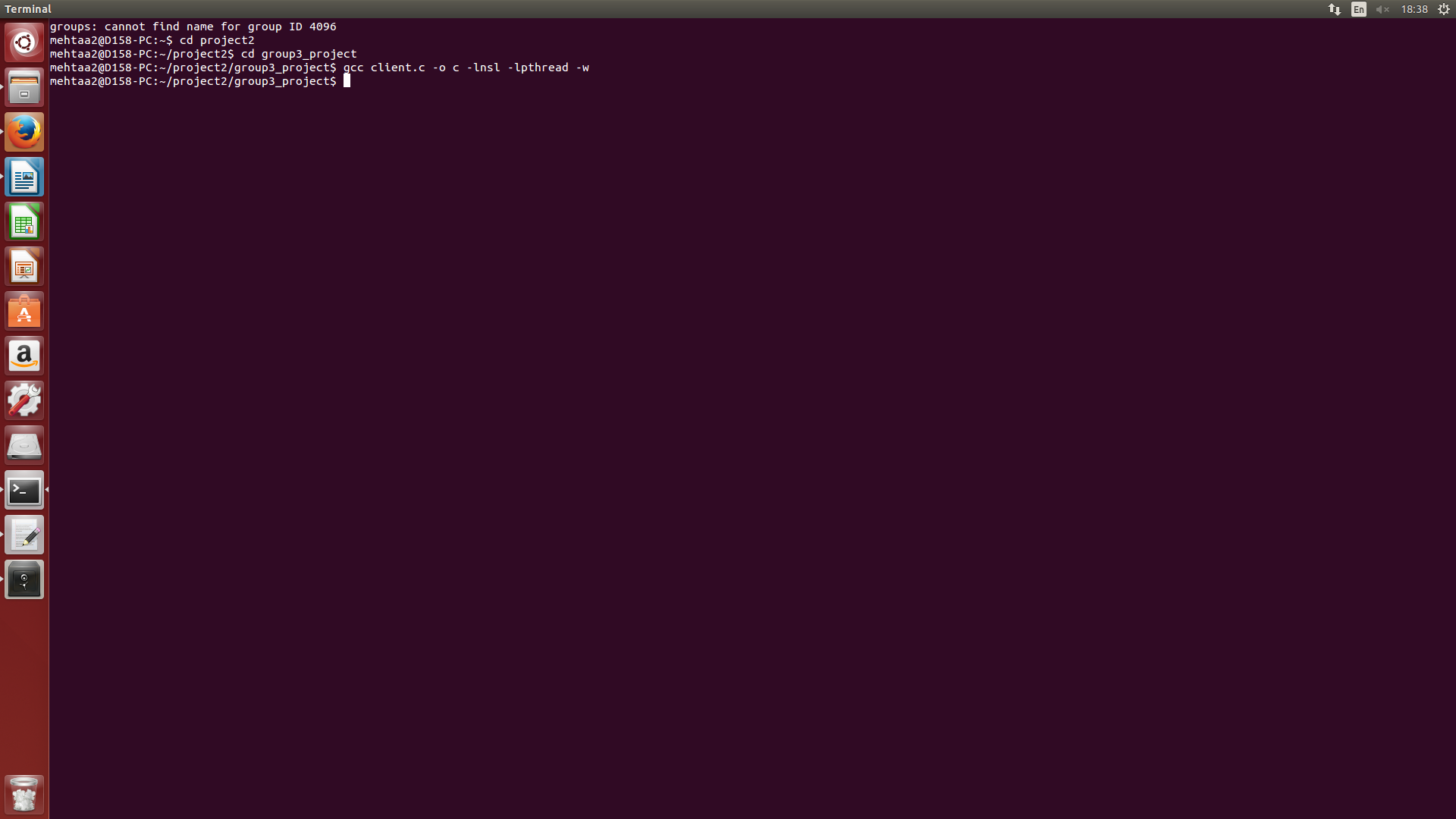
Step 3: Open the terminal.

Step 4: Compile the server and client in two different terminals.

Compiling Server : In Terminal 1, type “gcc server.c -o server -lnsl” and hit enter

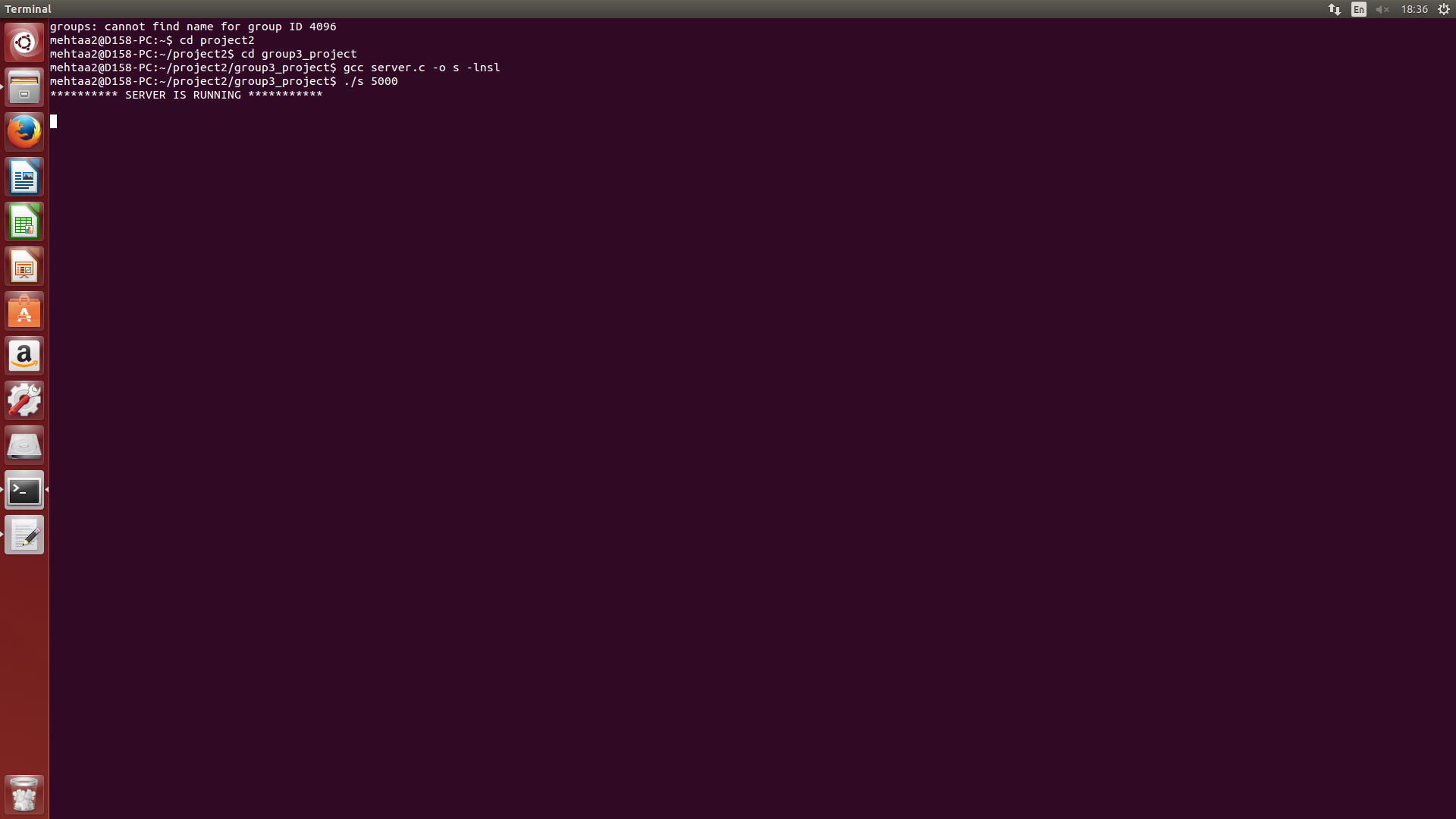


Compiling Client: In Terminal2, type “ gcc client.c -o client -lnsl -lpthread” and hit enter

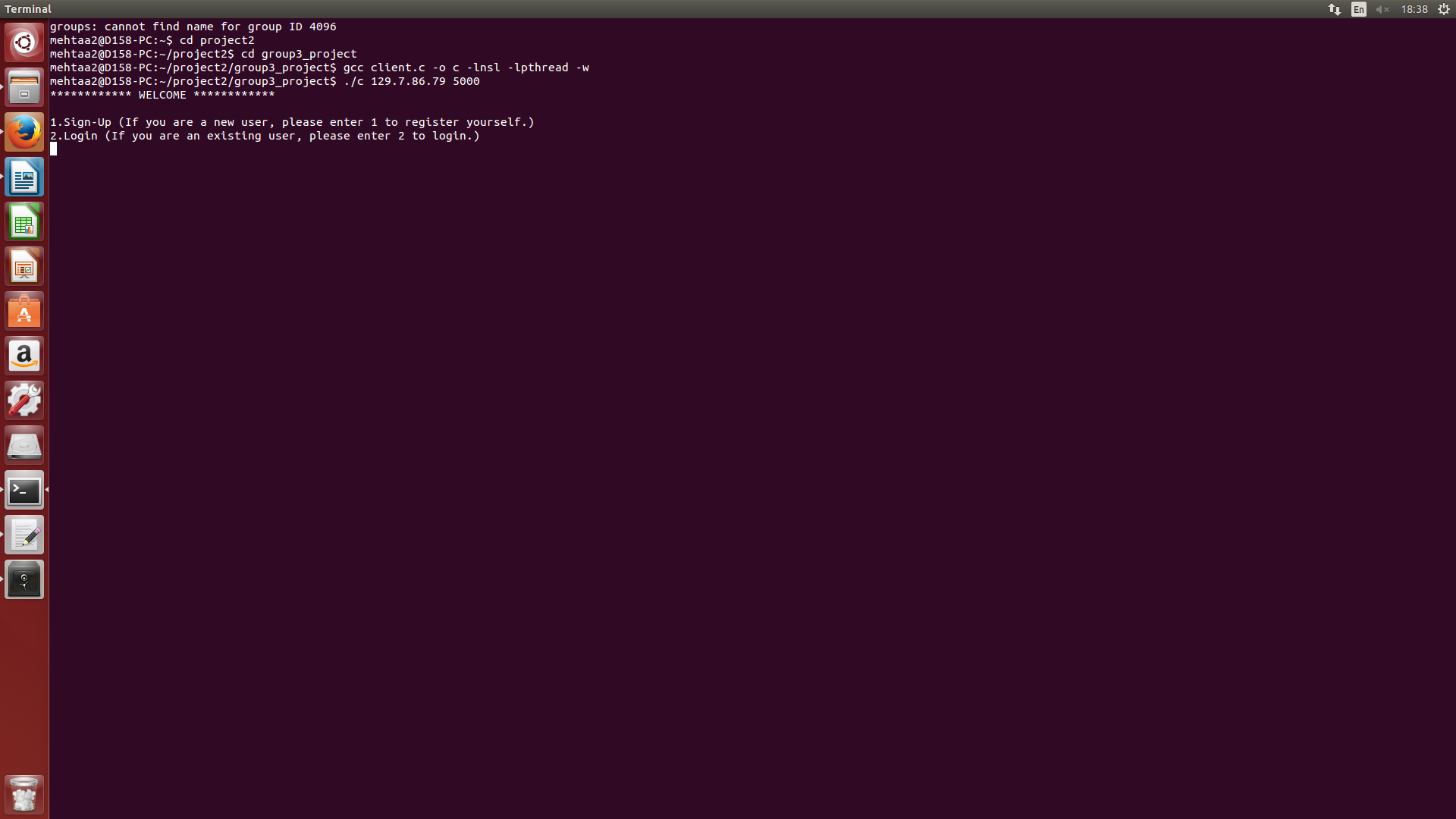


Step 5: Run the server and client for communication in between them.

Running Server: ./server 7000



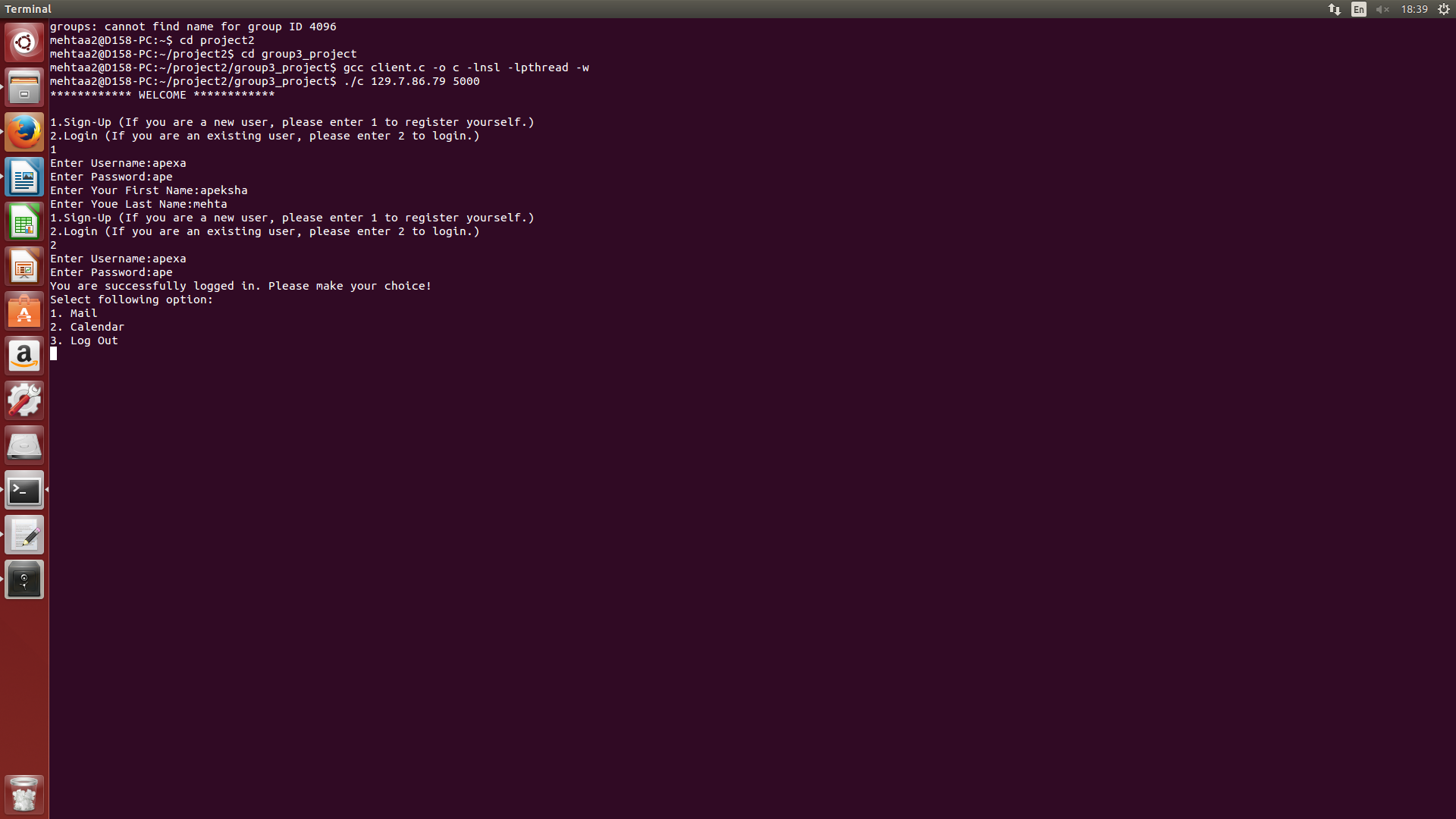
Running Client: ./client 127.0.0.1 7000



Step 6: Connection between the client and server started.

Step 7: Client is given with Two options , SignUp and Login

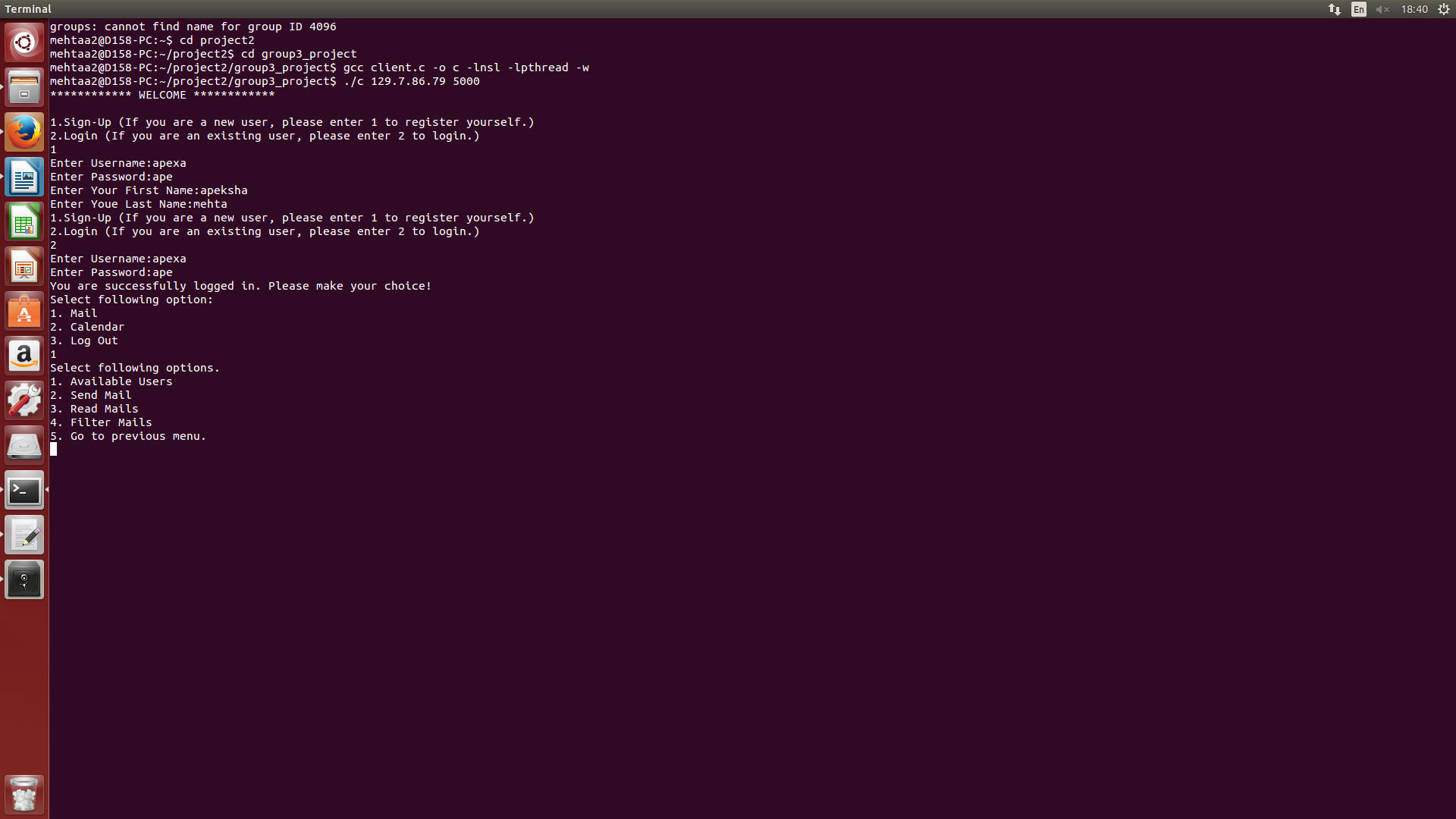
7.1 User need to select Sign Up if he doesnot have an existing account in the system. When user selects this option, he is asked to give username(unique) , password, first name and last name. After user enters every value, we go back to step 7. If user doesn’t select a unique name, he will be prompted again with the username.



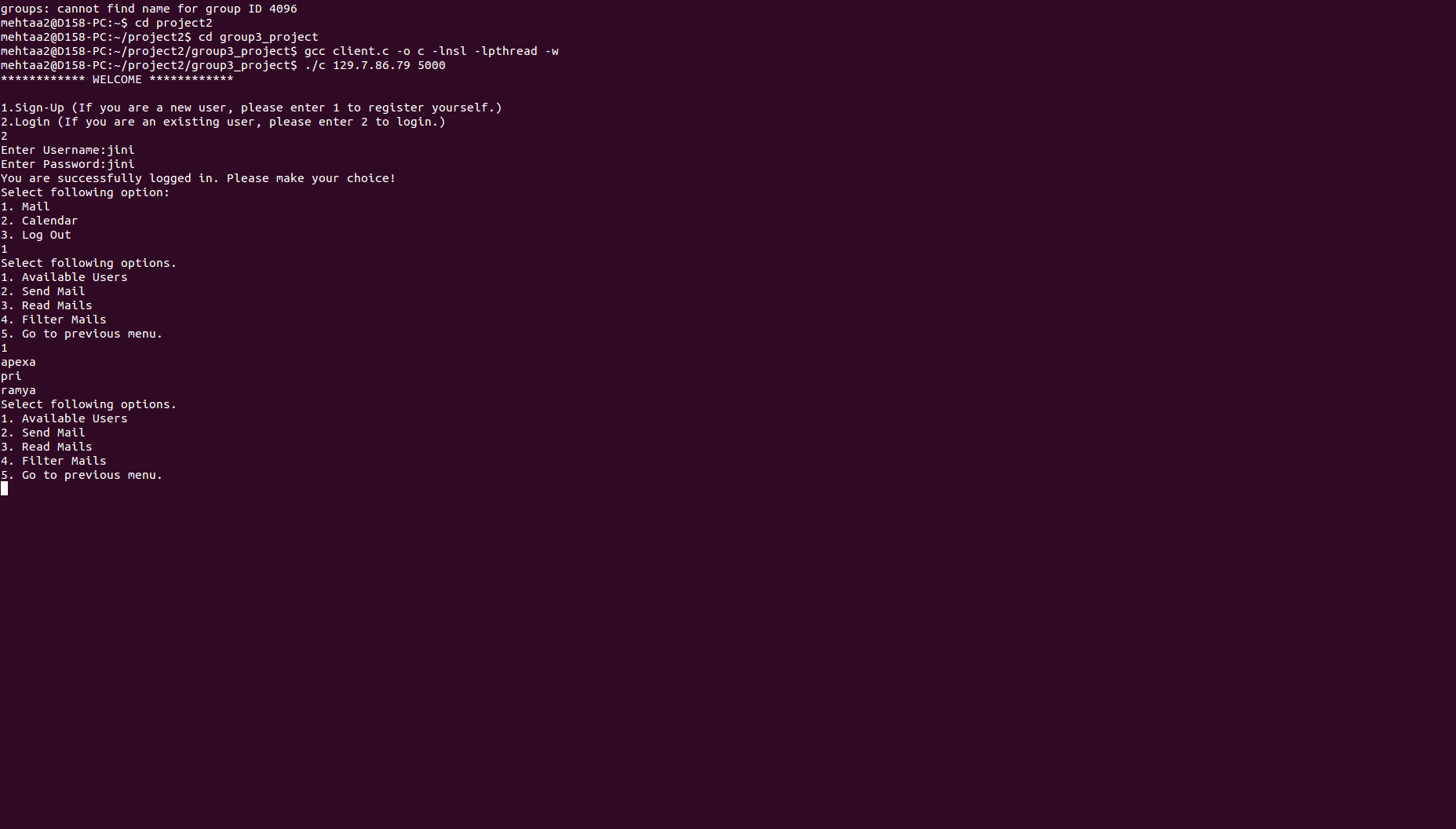
7.2 With the details like username and password, user can login into his account. If the details of the username and password doesn’t match the database. He is not authenticated and is prompted again for the details.

Step8: After User successfully logins, he will have three services, mail, calendar and Logout.

8.1 Mail Service: User enter into mail service where a user has options like viewing the available users, send mail to the existing users, view his mails, filter mails and go back to previous menu.

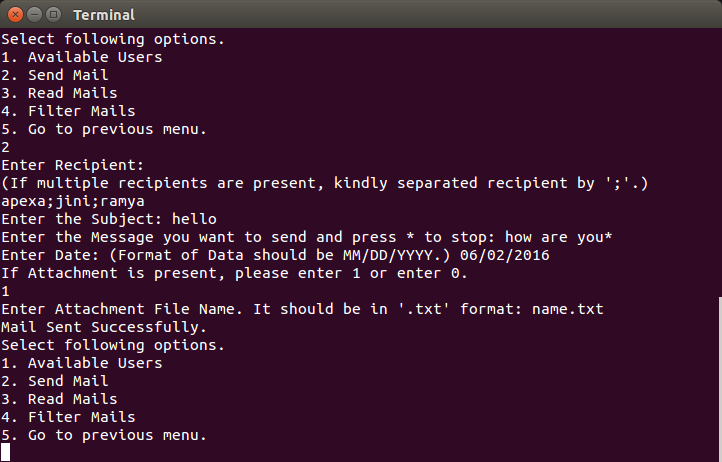


8.1.1. Available Users: User has an option to view all the available users in the system to which he can send a mail to.



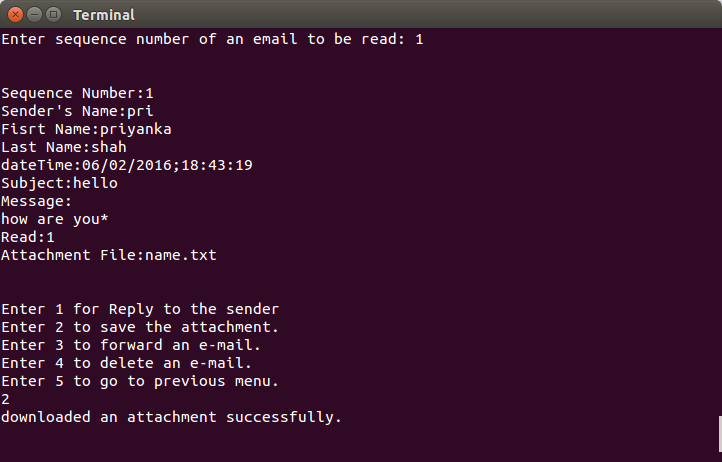
8.1.2 Send Mail: User can send mail to himself and other users.

* Recipient list: User is asked for recipient list which can be one or more, and is validated. If user types a wrong username, he is asked again to type the list of usernames.
* Subject: User should enter the data into subject
* Message: Message which can be multiple lines. Enter \* to end.
* Date: Date which should be mm/dd/yyyy format. If month and date values exceed the limit. User is asked again for the date.
* Attachment: Press 1 if u want to attach any attachment which should be in .txt format and from the local text files. 0 if no attachment is needed.

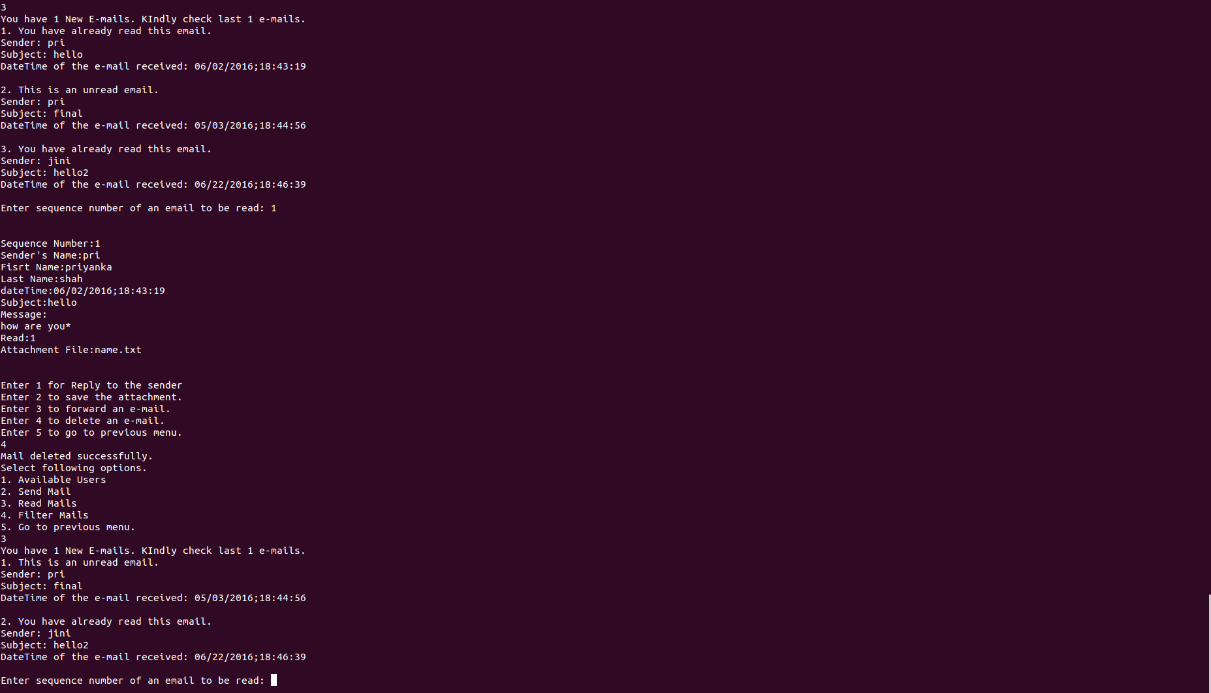


8.1.3 Read mails: Displays all the mails the user received. The User can select any message to read in details. On selecting a message, the user is given with options

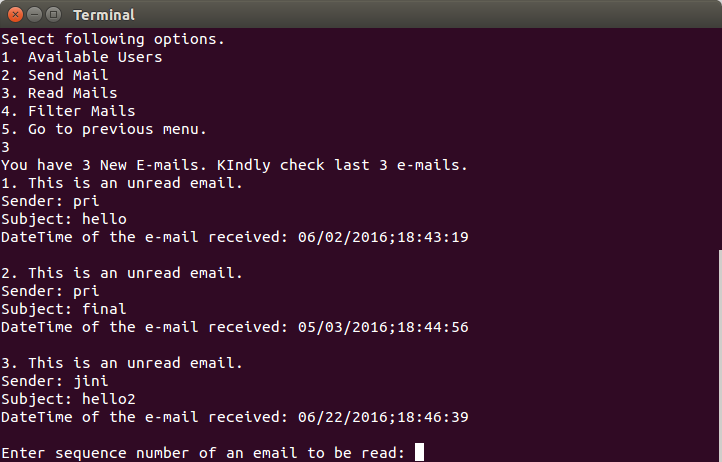
* Reply: Sends a mail to the sender of that selected message. User is asked with information like subject, message and attachment if needed.
* Save: This option helps in saving the attachment present in the mail body to the local disk with the same name as the attachment name.



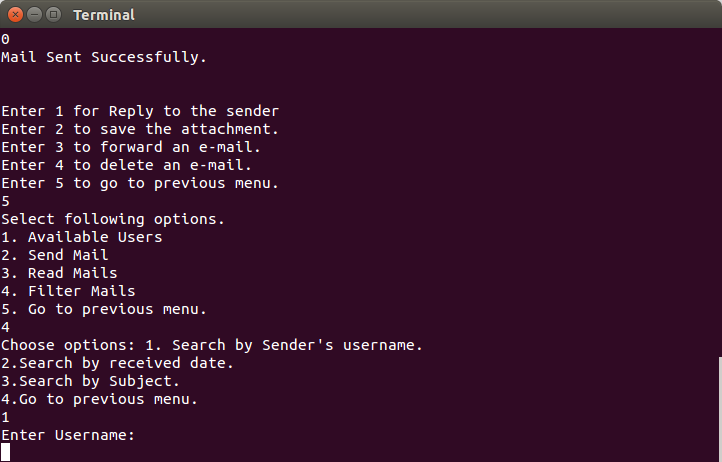
* Forward: User is asked to give the list of username to which that selected message need to be sent. The same message and subject is sent along with the message user types.
* Delete: Deletes the selected message from the inbox of the User.



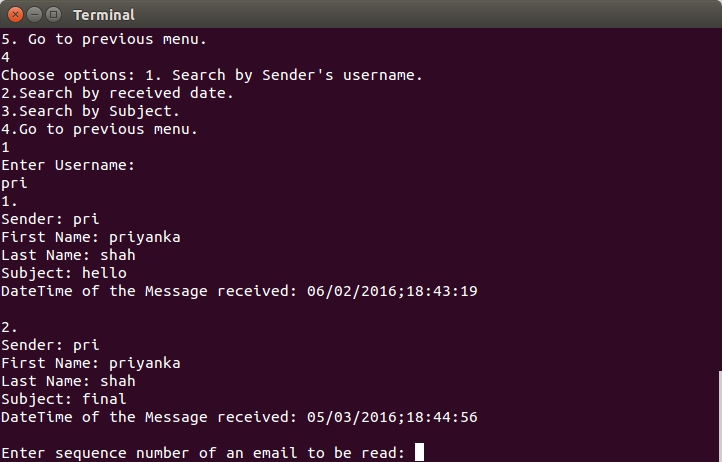
* Go to previous Menu: GO back to step 8.1



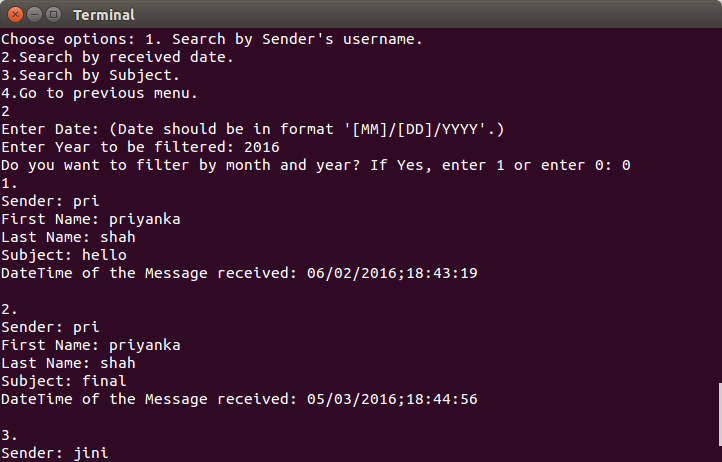
8.1.4 Filter Mail: User can manage the mails and select only the mails of interest by filtering. The different filtering options are:



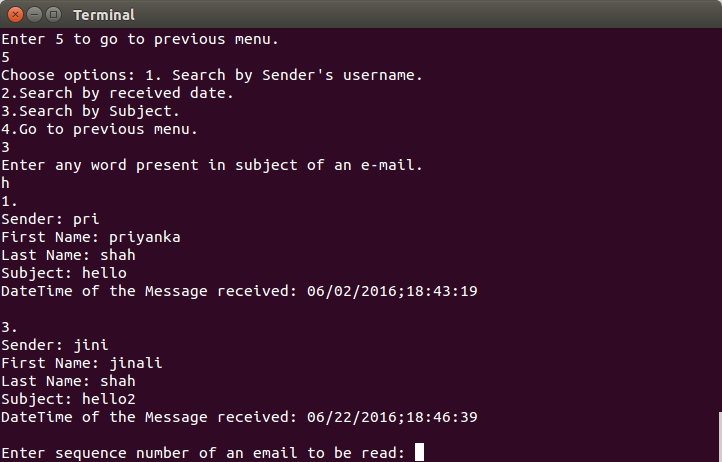
* Search by sender’s name: User is asked with name he want to search the inbox with. On entering the information all the mails sent by the entered name are retrieved. User can view the results in detailed and 8.1.3 repeats.



* Search by received date: User can search the mails based on the year the mail it has been received and then the priority comes to month and date.
  + - * User is asked with year the mails are to be searched with.
      * After that user can select 1 to narrow down the search to month. So, the mails are retrieved based on the year and specific month of that year. Else 0 to check only mails based on year.
      * User is given another option, if user select 1, then he can narrow his search for date. So, user can select a mail of a specific day of a particular month of a year. Else 0 to check mails based on month and year.
      * User can select a specific message from the results and 8.1.3 repeats.



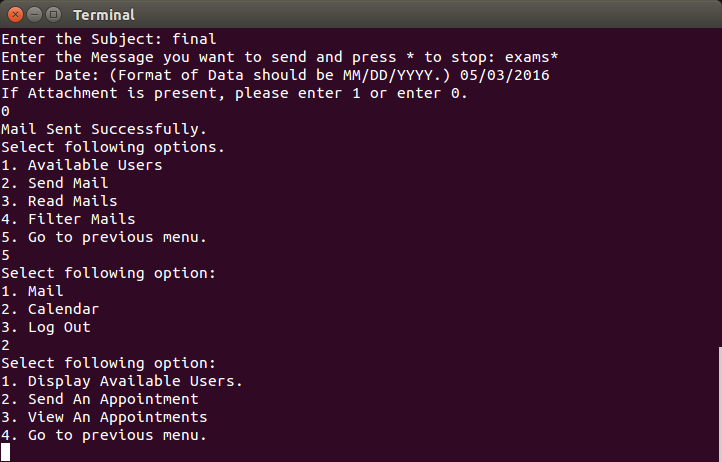
* + - Search by subject: User can search the mails based on the subject in the message. User is prompted for any keyword in the subject and after giving this information, user is given with the mails which match the word in the subject.



* + - Go to previous menu: Takes the control to step 8.1

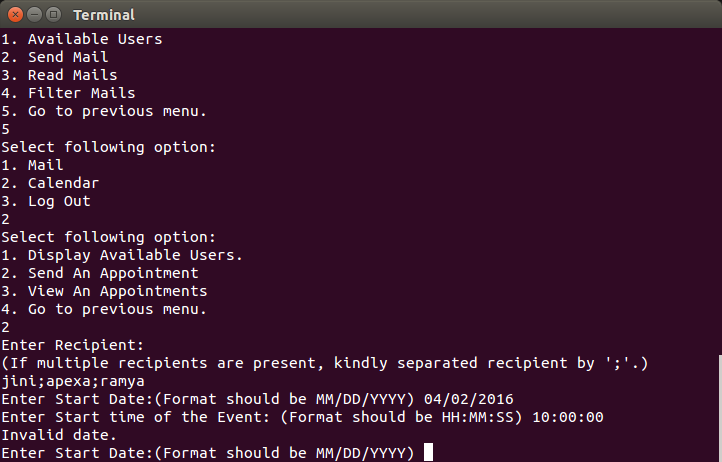
8.1.5: Go to previous Menu: Takes control to step 8.

8.2 Calendar Service: User enters into calendar service and have services like display available users, send an appointment, view an appointment and go to previous menu



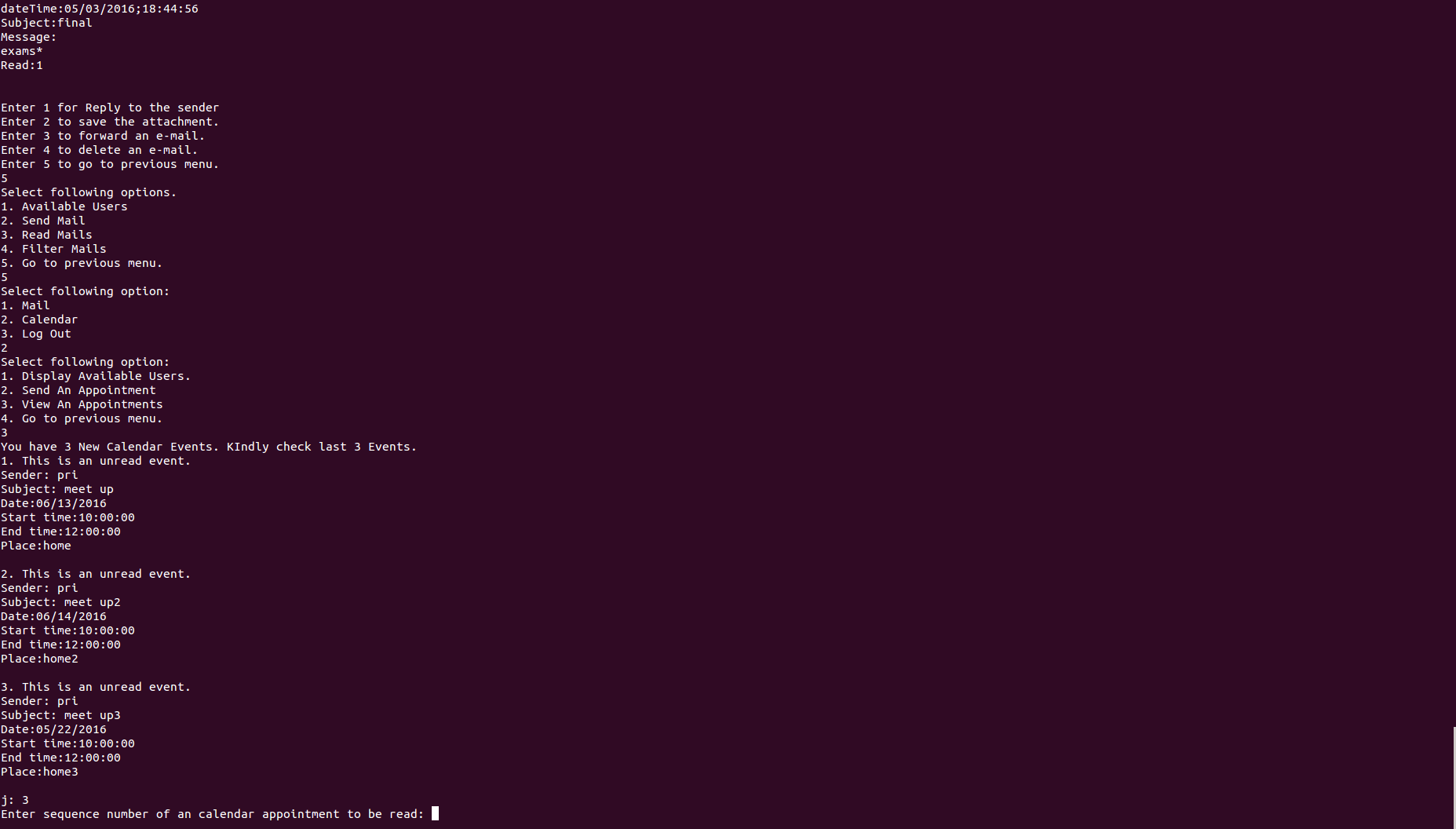
8.2.1 Available Users: This displays the list of available users in the system to which an event can be sent to. This is same as the available user in mail service.

8.2.2 Send an Appointment: User can send an event request to the other users. The user is prompted with

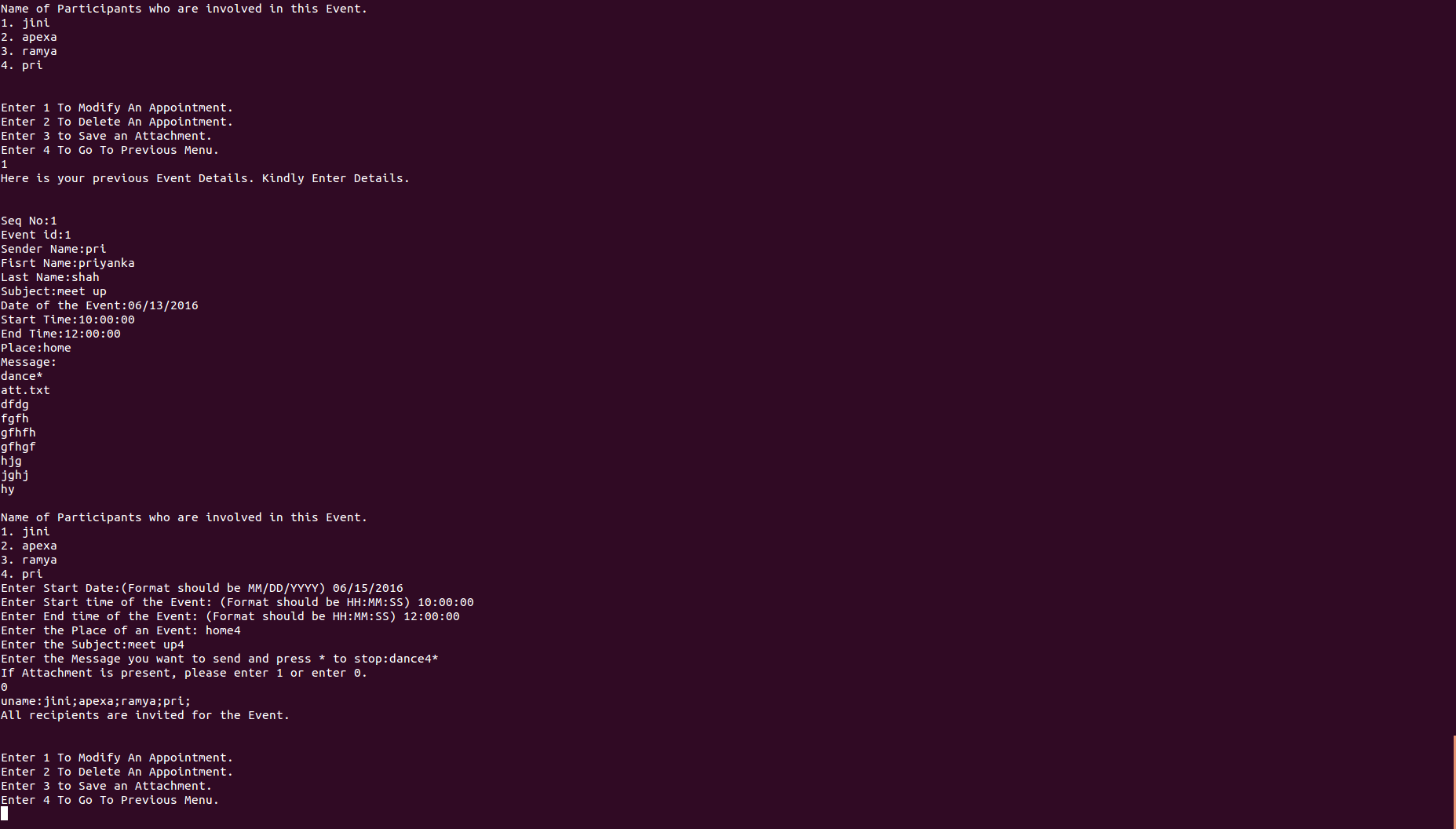


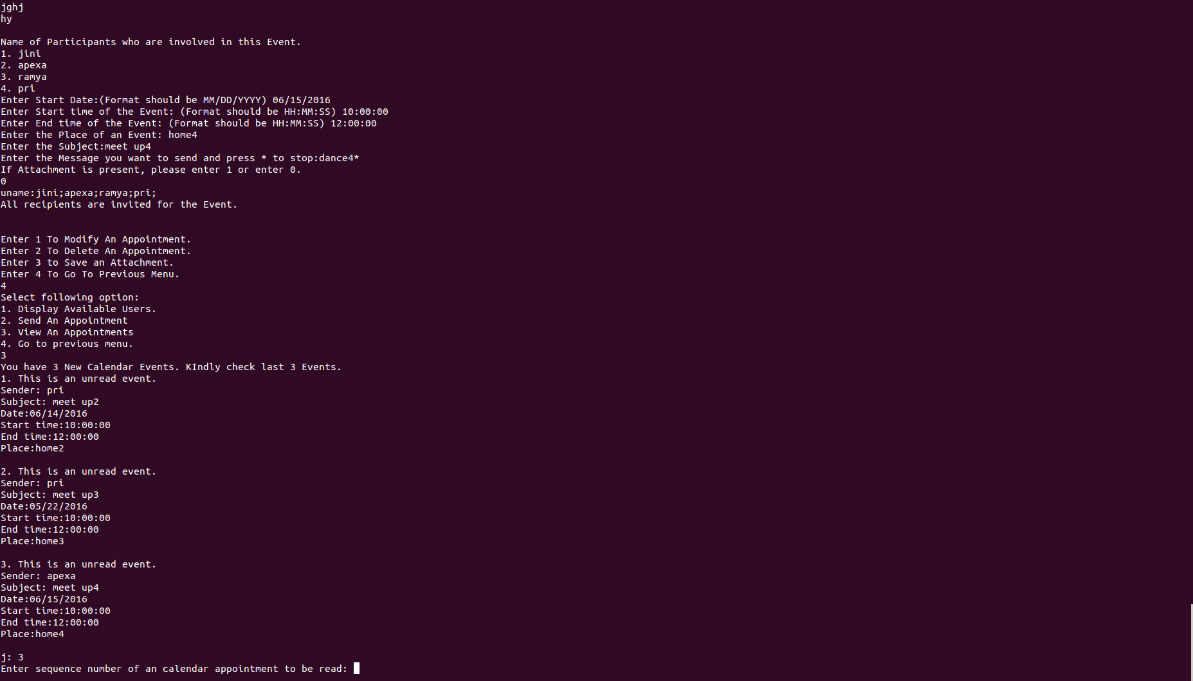
* Recipient list: the different users the event request should be sent to. It can be one or more recipients. If any of the username in the list is not valid, the user is asked with the list again.
* Start date of the event: When the evet starts. It should be in the format of mm/dd/yyyy. Start Time: The time when the event starts. It should be in the format hh:mm:ss.
  + If user enters date and start time prior to the present date and time, it shows an invalid message and the date and start time should be inputted again. If the value of hour, minute and second is beyond the range. The start time should be inputted again.
* End Time of the event: The time when the event ends. It should be in the format of hh:mm:ss.
  + If the value of hour, minute and second is beyond the range. The end time should be inputted again.
* Place of the event: Venue of the event.
* Subject: The subject of the event should be entered.
* Message: Message can be multiple lines and press \* end the message.
* User must press 1 if he is interested in sending an attachment, else 0. The attachment must be in .txt format and from the local text files.
* Go to step 8.2

8.2.3 View an appointment: Displays all the appointments. Enter the sequence number to read the event in detail. User can modify the event, delete the event, save an attachment and go back

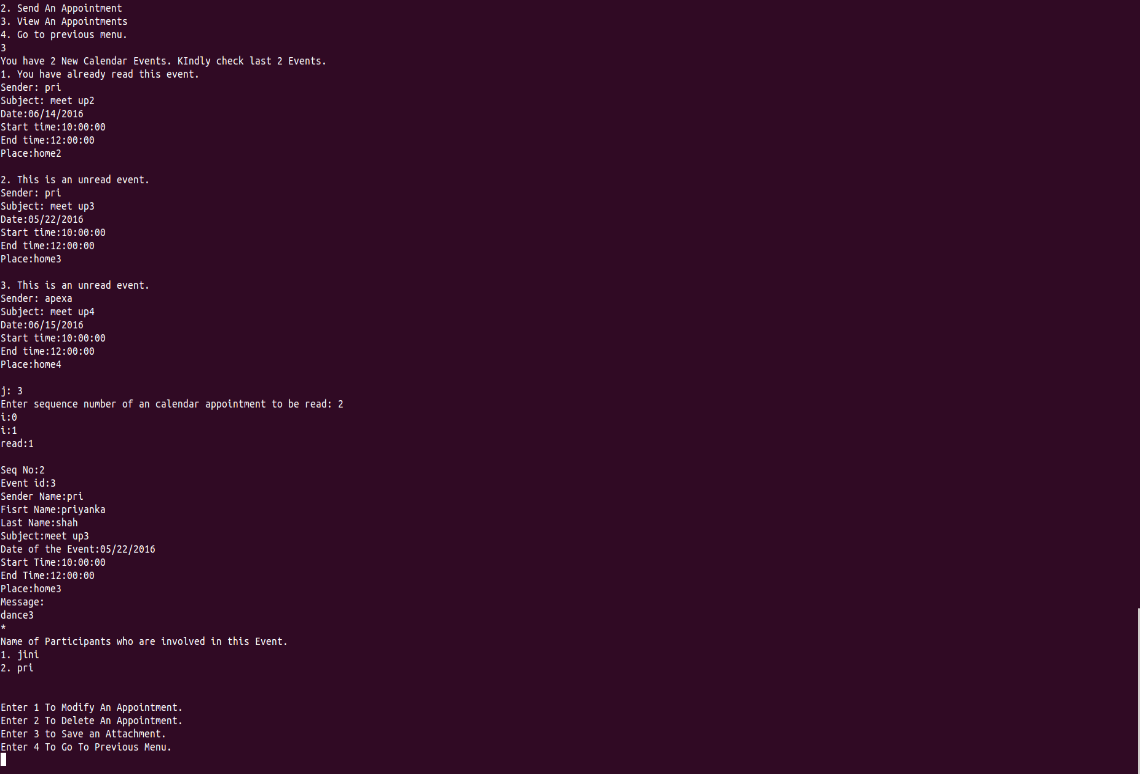


* Modify: If the user is not feasible with the selected event ‘s date and time of the event. He can change the date or time or both. The modification will be reflected in all the recipient list.



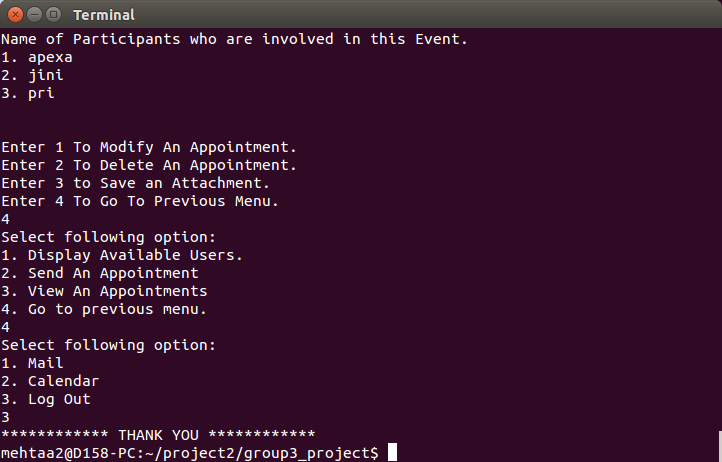


* Delete: Deletes the selected event from the Inbox. This means that the user is not interested in attending this event.



* Save: This option helps in saving the attachment text file to the local disk using the same attachment name.
* Go back to previous menu: Takes the control back to step 8.2

8.3 Logout: Gracefully exits the client from the system by saving all the data to the file.

****