**Data Structure**

**(Mail Server)**

* **Ziad Hisham Ali (20)**
* **Youssef Ahmed Sayed (63)**
* **Mustafa Mohamed Mahmoud (50)**

1. ***Description******and Features.***

It’s mail project that enables to contact with others...

First of all... The email is signed up if it’s the first time to use the application it’s required your: 1. First name

2. Last name

3. User name

4. Password

Or login if there’s already account on the application and it’s only require your: 1.User name

2. Password

1. **Contacts** : You’re able to add other persons that uses this application and have accounts and this require :

1. First name of the added contact

2. Last name of the added contact

Or remove other contacts that already you have

1. **Compose:** it send the emails to others Only to persons that

Have accounts it require: 1. to whom the email will be sent

2. Subject of the message

3. The body of the message

4. Attachment (Optionally)

1. **Move Emails:** it move an email from place to another place (it delete the email from the current place and put it into the destination place) it require: 1.Selecting an email

2. Choose the destination place

(It can’t be Sent, Inbox or Trash)

1. **Delete Emails:** it delete an email by moving it to the Trash it require only to select an email
2. **Filters:** it sort the emails by any condition you choose (To, Subject, …) Or remove a filter that already exist the account it require : 1. Add filter

2. Choose the condition to be sorted by

1. **Refresh:** it’s only reload the current page that already used
2. **Search:** it search for a specific email that wanted and it may be by the body of the message or subject or from certain user ….
3. **Add Folder:** it enable to add another folder in the application to have some emails it require: 1.Folder name

2. The option that email have

3. Any added option else

1. **Edit:** it edit the account itself it can edit: 1.the first name

2. The last name

3. User name

4. Password

1. **Remove:** it remove a folder whole it require only select a folder
2. **Sign out:** it close the account and to open the account again you must enter the user name and the password

1. ***Assumption.***

The names of the accounts saved by the user name of the account it saved in a folder called **Users** and also there’s a text file that called **Usernames** and this file save: 1.First name

2. Last name

3. User name

4. Password

Of each account and inside each folder in the users folder it saves:

1. Sent (Saves the sent messages it saves them as a folder and inside it the body of the email saved as text file and if there’s attachment it save it also as a file And also it save a text file called **index file** that save the date , from , to , subject and attachment if there’s)
2. Inbox (Saves the messages that received By the same Logic)
3. Trash (Saves the deleted messages By the same logic)
4. Contacts.txt (Saves the contacts that this account have it saves the first name , last name & user name)
5. Any added Folders (folders that added and have messages that specified by specific condition)
6. ***Data Structure Usage.***
7. Index files writing that have the informations of the message & load the folders contents is implemented by **Double Linked List**
8. Load the selected mails to make anything on them (move , delete, … ) & handling of the attachments & the Filters option is implemented by **Single Linked List**
9. Priority lists of the emails is implemented by **Priority Queue**
10. Sending an email to multiple users is implemented by **Queue**
11. Filter and searching is Implemented by **Binary Search**
12. Sorting the emails is implemented by **Quick Sort using Stack**
13. ***Main Module.***

**Package**: eg.edu.alexu.csd.datastructure.mailServer.cs20\_cs50\_cs63

**Classes:** 1. App: the main logic and code present in this class

2. AppGui: the GUI code (buttons, labels …)

3. Contact: the informations of the accounts and contacts (First name, Last name, Password & User name)

4. Filter: the Single linked list of the filters was set & get from this class

5. Folder: the added folders were set & get from this class

6. Mail: the emails information (from, to, subject, attachments, priority) were set & get

7. PriorityQueue: implements the priority queue that used in the main code

8. PriorityQueueNode: implements the node of the priority queue

9. Sort: the quick sort code that based on the stack present in this class

1. ***Pseudo Code.***

* boolean createUser ( IContact contact) {

Contact con = new Contact

Path user = paths.get (“users”)

If (user not exists)

Files.createfile (user)

Path data = paths.get (“users/username.txt")

If (data not exists)

Files.createfile (data)

String url = con.getUsername

Path folder = Paths.get (folder)

If (folder exist)

Return false

Files.createfile (folder)

Buffered bwriter

Writer. Write (con.getUsername)

Writer. Write (con.getFirstname)

writer. Write (con.getLastname)

Writer. Write (con.getPassword)

Return true

}

* boolean loginUser ( String username, String password) {

File Reader users

Users = new File reader (Users/usernames.txt)

BufferedReader bReader = users

String tempUserContent = bReader.readline ()

String[] usercontent = tempUserContent.split(“ ”)

If (usercontent [0] = username)

If (usercontent [1] = password)

Return true

Return false

}

* boolean signin( String email, String password) {

Path user = Paths.get (users)

If (user not exist)

Return false

String url = “users/” + email

Path folder = Paths.get (url)

Path data = Paths.get (users/usernames.txt)

If (folders exist)

Return loginUser (email, password)

Return false

}

* boolean signup( IContact contact) {

boolean check = createUser(contact)

If (check)

Print (“Sign up Successfully”)

Return check

}

* void sendingEmail( mail mail, String folder) {

int counter = 0

String temp = “users” + mail.getTo + “/” + folder +.getSubj

Path data = get temp

String temp2 = “users” + mail.getTo + “/” + folder

Path inbox = get temp2

If (not exist)

Create folder

While (data exist)

Temp += counter

data = get temp

Buffered bwriter = new Buffered (m)

bwriter. Write (current date)

bwriter. Write (mail.getSubj)

for (i=0 to i < mail.getEmail().length)

bwriter. Write (mail.getEmail()[i])

for (i=0 to i < mail.getAttachments().length)

streamcopy(src)

}

* void savingEmailSent (Mail mail, String currentuser, String folder) {

int counter = 0

String temp = “users” + currentuser + “/” + folder +.getSubj

Path data = get temp

String temp2 = “users” + currentuser + “/” + folder

Path inbox = get temp2

If (not exist)

Create folder

While (data exist)

Temp += counter

data = get temp

Buffered bwriter = new Buffered (m)

bwriter.write (current date)

bwriter.write (mail.getSubj)

For (i=0 to i < mail.getEmail ().length)

bwriter.write ( mail.getEmail()[i])

For (i=0 to i < mail.getAttachments().length)

streamcopy (src)

}

* boolean compose( IMail email) {

Mail e = new Mail

If (not usercheck (e.getTo))

Return false

SendingEmail (e,”Inbox”)

SavingEmailSent (e, e.getFrom,”Sent”)

}

* void deleteOrMove(String currentUser ,String folder ,String option ,int[] index) {

Dlinkedlist dmain

dmain = indexload (currentUser, folder)

Slinkedlist smain

smain = loadMails (index ,currentUser ,folder)

Folder x = new Folder

x.setFolder (option)

If (option = Trash)

deleteEmails (smain)

For (i = index. Length – 1 to 0)

dmain.remove (index[i])

File file= (“user/” + currentUser + “/” + folder +ind[i]

Delete (file)

IndexWriter (dmain, currentUser, folder)

else

MoveEmails (smain, x)

For (i = index. Length – 1 to 0)

dmain.remove (index[i])

File file= (“user/” + currentUser + “/” + folder +ind[i]

Delete (file)

IndexWriter (dmain, currentUser, folder)

}

* void deleteEmails( ILinkedlist mails) {

For (i=0 to mails. Size)

Mail m = new Mail

SavingEmailTrashOrMoved (m, currentUser, “Trash”, m.getEmail () [0])

}

* void moveEmails( ILinkedlist mails ,IFolder des) {

Folder destination = (Folder) des

For (i=0 to mails. Size)

Mail m = new Mail

SavingEmailTrashOrMoved (m, currentUser, destination.getFolder (), m.getEmail () [0])

}

* int partition( String[] arr ,int low ,int high ,int k) {

String [] p = arr [high].split(“ ”)

String pivot = p[k]

Int i = low – 1

For (i=low to high)

String s = arr[j].split (“ ”)

If(s[k].compare (pivot) < 0 || s[k].compare (pivot) = 0)

i++

Swap (arr[i], arr[j]

Swap (arr [i+1], arr [high])

}

* String[] sort( String[] list ,int low ,int high ,int k) {

Stack <integer> S

s.push (0)

s.push (-1)

While (s is not empty)

While (low <= high)

Int pivotLocation = partition (list, low, high, k)

s.push (pivotLocation + 1)

s.push (high)

high = pivotLocation – 1

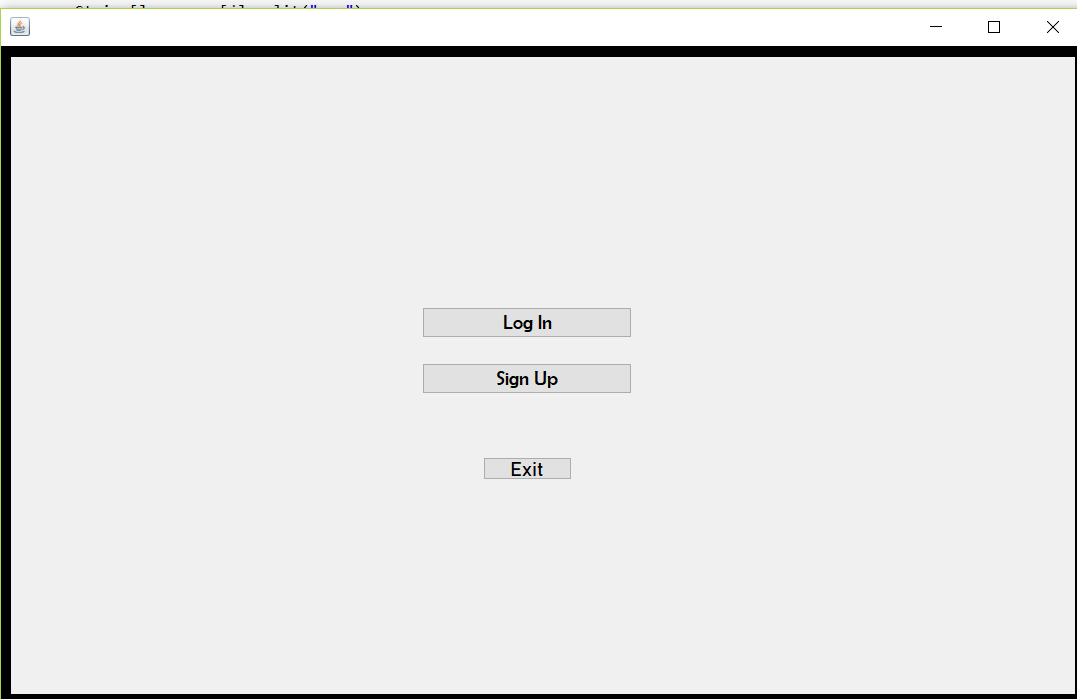
high = s.pop ()

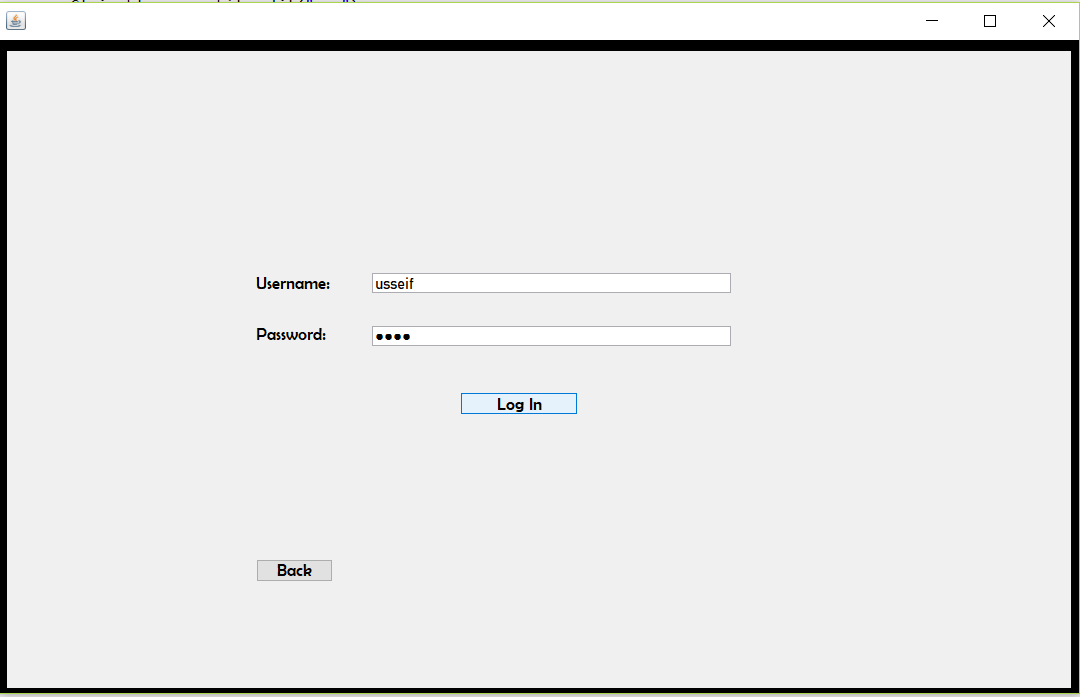
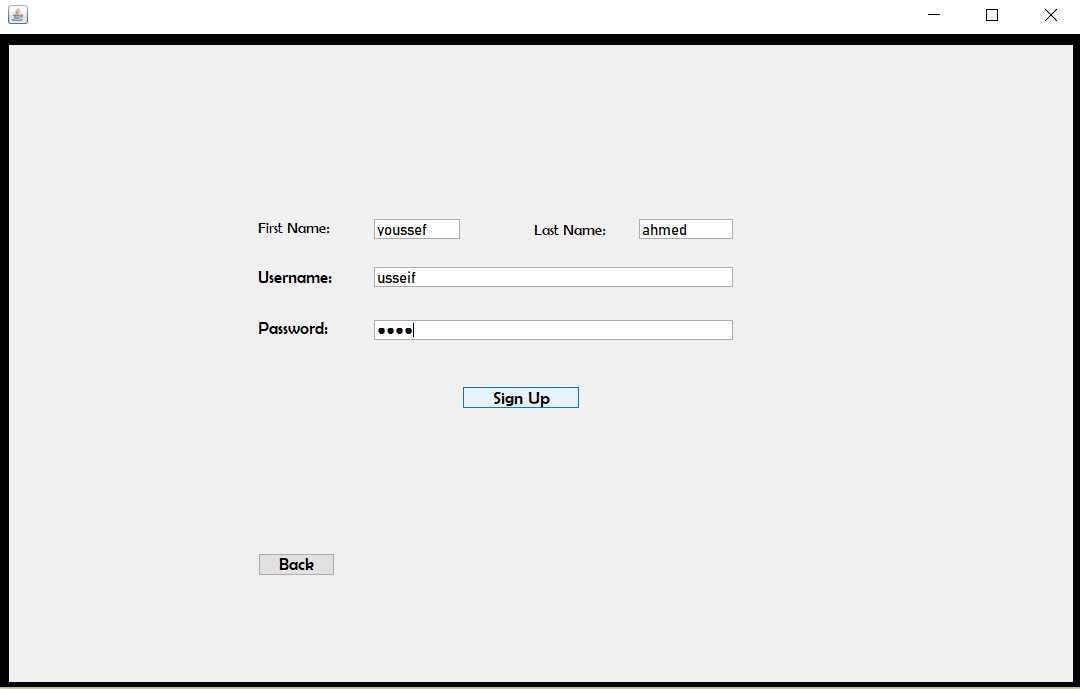
low = s.pop ()

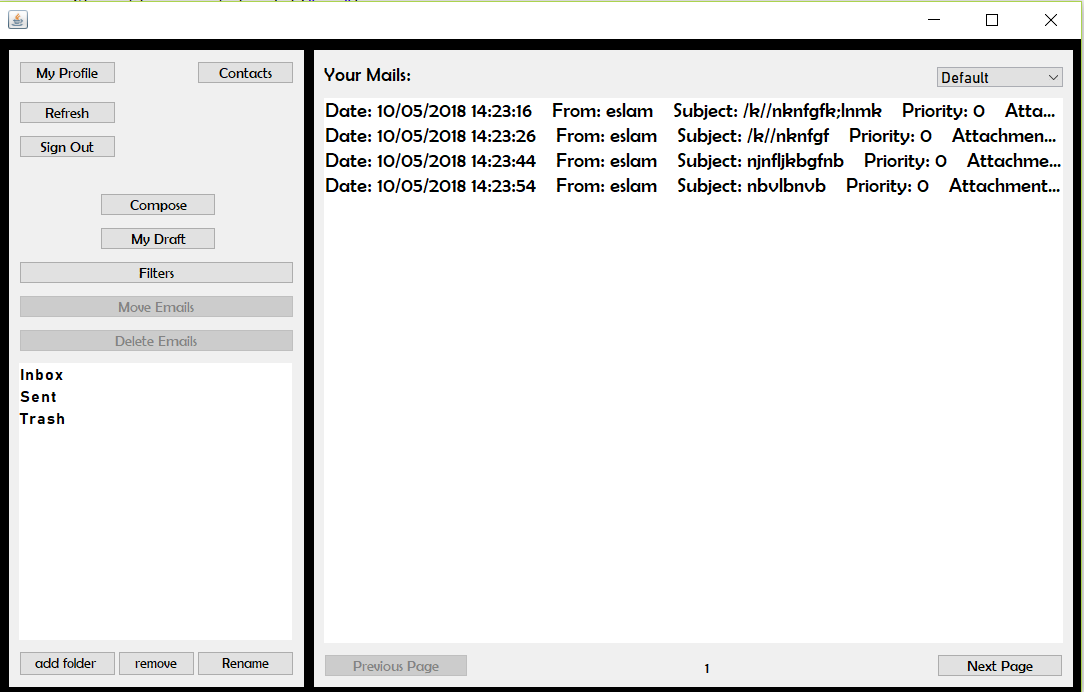
Return list

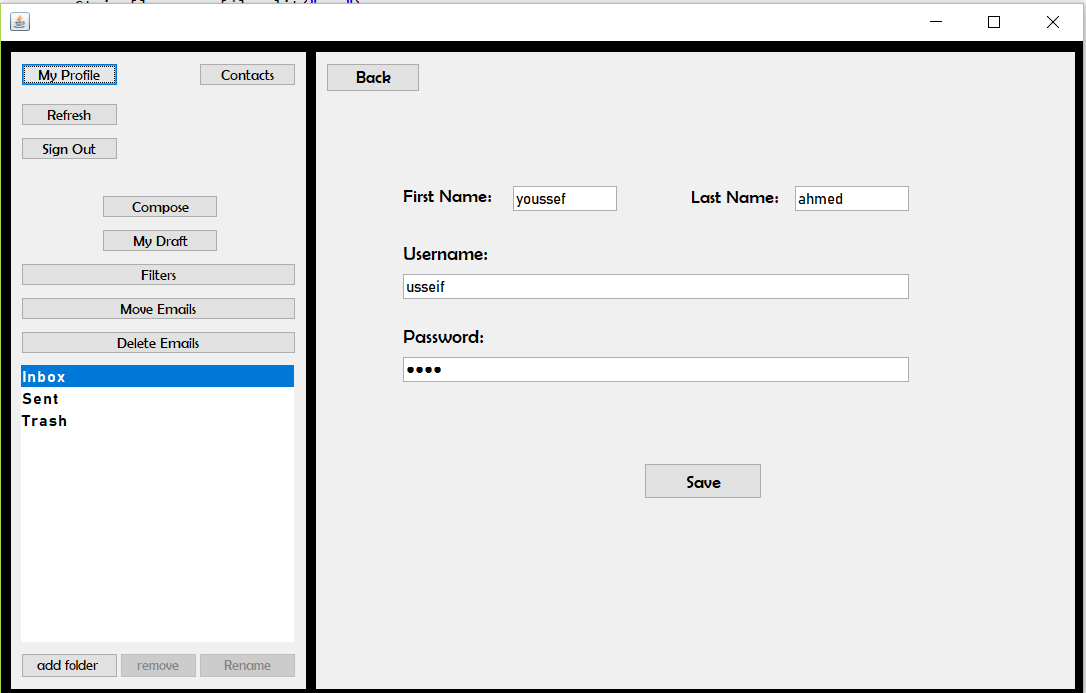
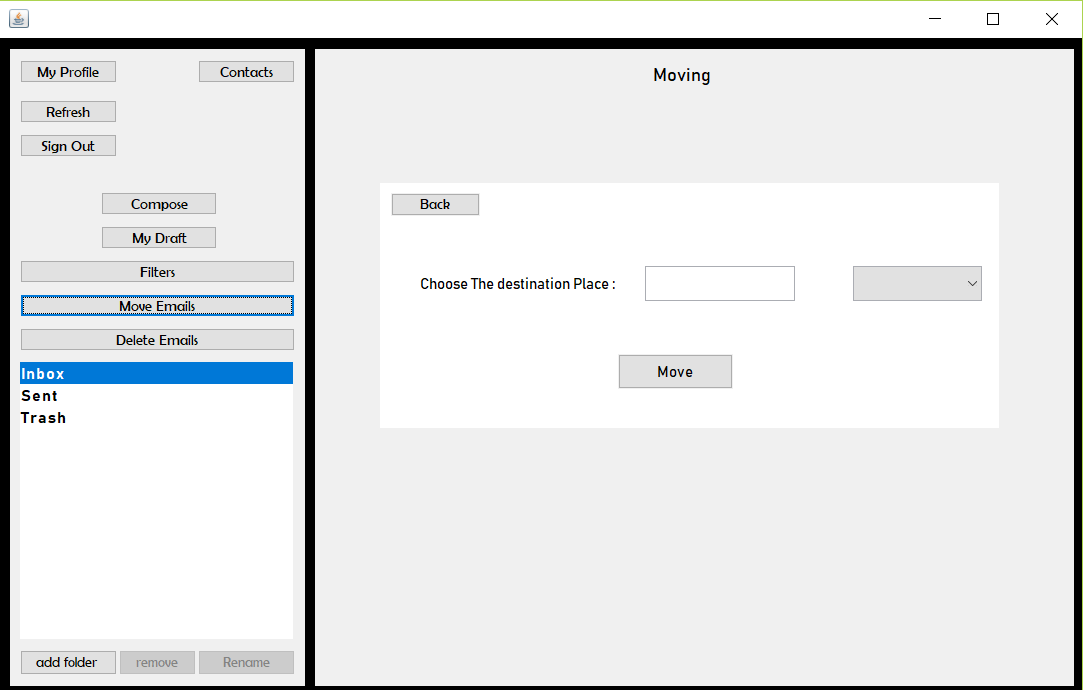
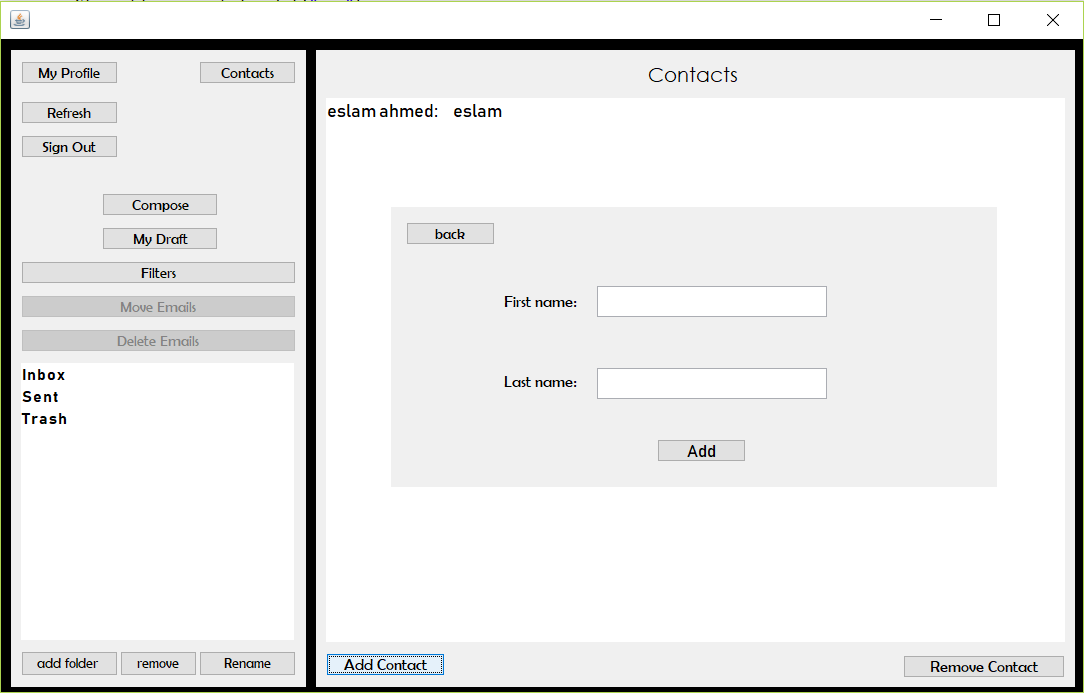
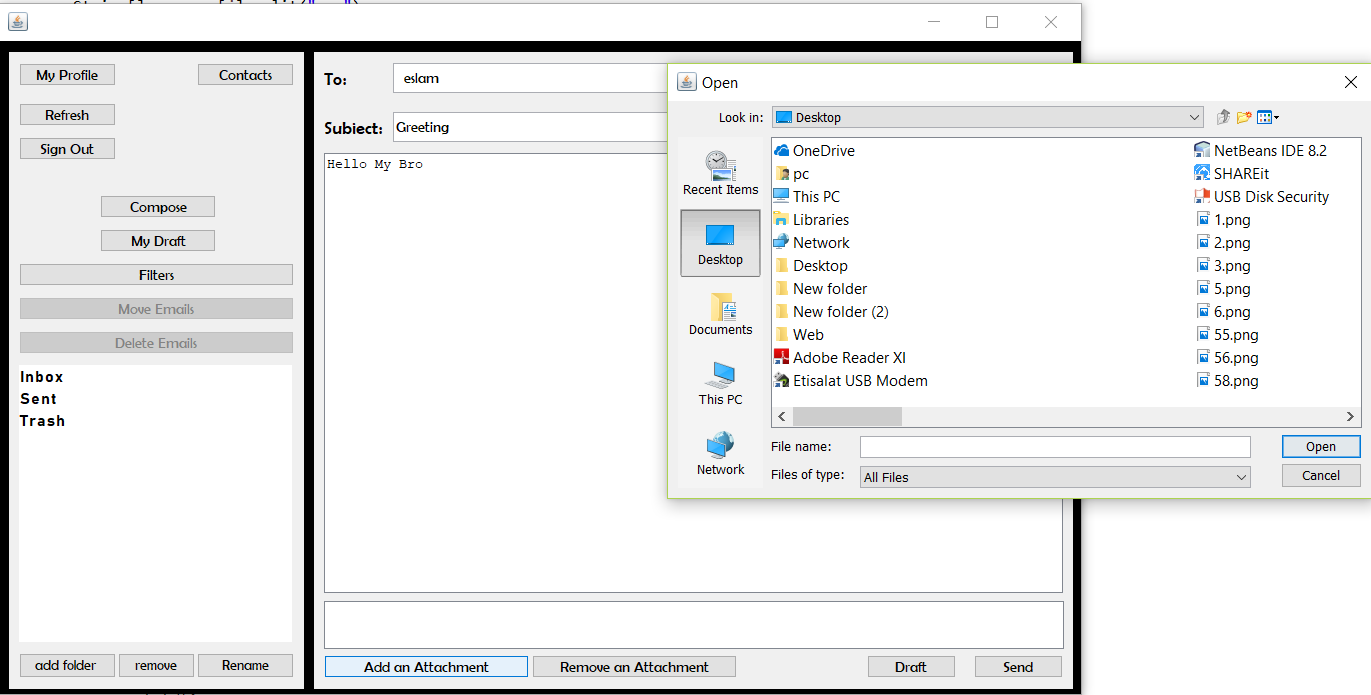
}

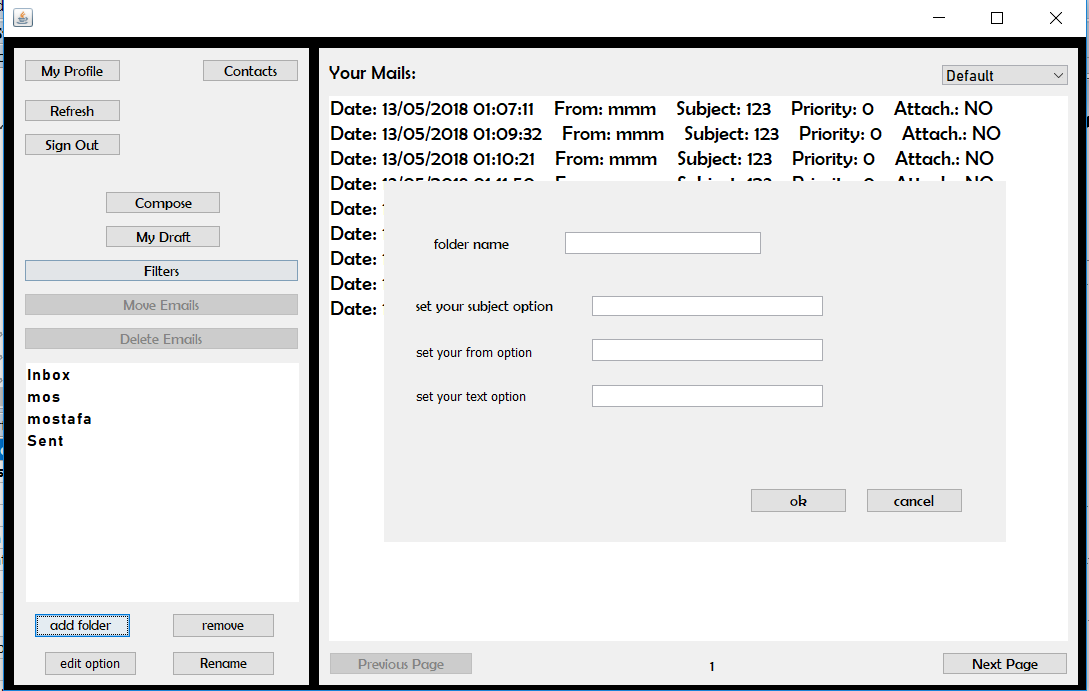
1. ***Sample Runs.***

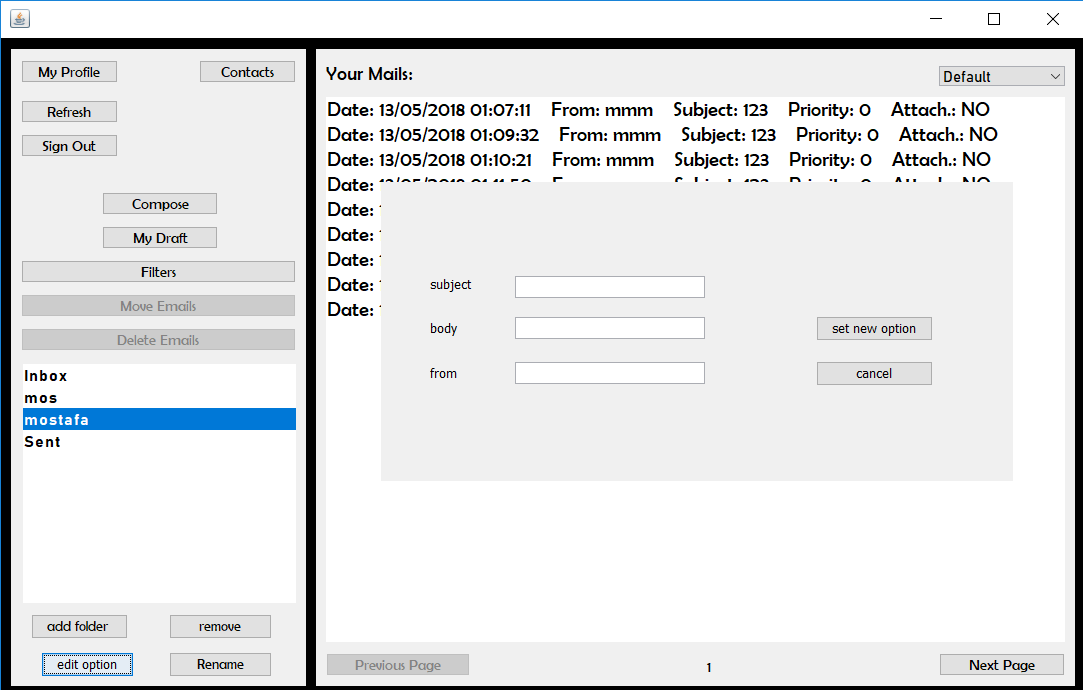












1. ***User Manual.***

* Sigin up if you don’t have previous account by Entering ( First name, Last name, Username ,Password)
* Login if already you have an account by Entering (username ,password)
* To Send a message : 1. Select Compose

2. Enter username of account

3. Enter the subject of message

4. Write the body of message

5. Add Attachment if you want

* To Add a contact : 1.Select contact

2. Select add contact

3. Enter first & last name of contact

* To move an mail : 1. Select the wanted message

2. Select Move Emails

3. Choose destination from jumboBox

4. Click Move

* To delete an email : 1.Select the wanted message

2. Choose Delete Emails

* To Add a folder : 1. Select add folder

2. Enter folder name

3. Enter the option you wanted

5. Click Ok

* To edit a folder option : 1. Select a folder from the list

2. . Select edit option

3. Enter the option you wanted

5. Click Set Option

* To edit the account : 1. Select My Profile

2. Edit the First name, Last name, Password Or User name

* To add a filter (Search) : 1. Select Filters

2. Select the wanted folder

3. Select what you want to search about

* To rename a folder : 1. Select the wanted folder

2. Select rename

3. Enter the new name

* To Reload a page : Select Refresh
* To Sigin out : Select Sigin out