OOP
Mail Server Web Application

Name	ID
Habiba Osama Zaky	19015575
Mai Ahmed Hussien	19016736
Bassant Yasser Salah	19017262
Toka Ashraf Abou Elwafa	19015539

Opening both angular and Spring boot code:

Running Angular:

1- Unzip downloaded file.
2- you must have NodeJS and angular-Cli on your computer.
3- After unzipping the file, open the project file and press right click on the background of the folder. Open **cmd** window.
4- Write **npm install** in cmd window.
5- Then you can open the project by writing **ng serve –open in cmd window (port number: 4200).**

Running Spring Boot project:

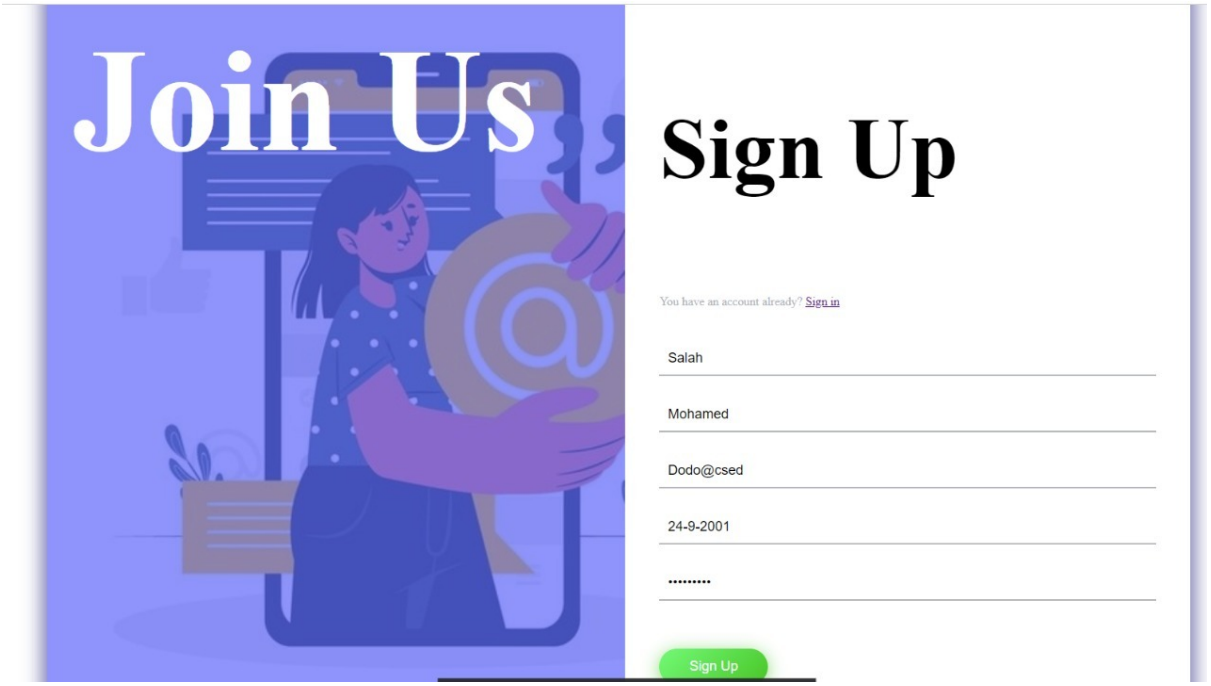
6- Open Eclipse ide or any other ide.
7- Import the whole Spring boot folder into ide.
8- Make sure while running the code that the port number inside (Crossorigin is the same port number of angular project).

How to run mail server:

1-Run Angular Code while Running Spring boot program to be able to receive data from the front end.
2- As shown the picture this page will appear in the browser which contains Email interface as shown in fig.1

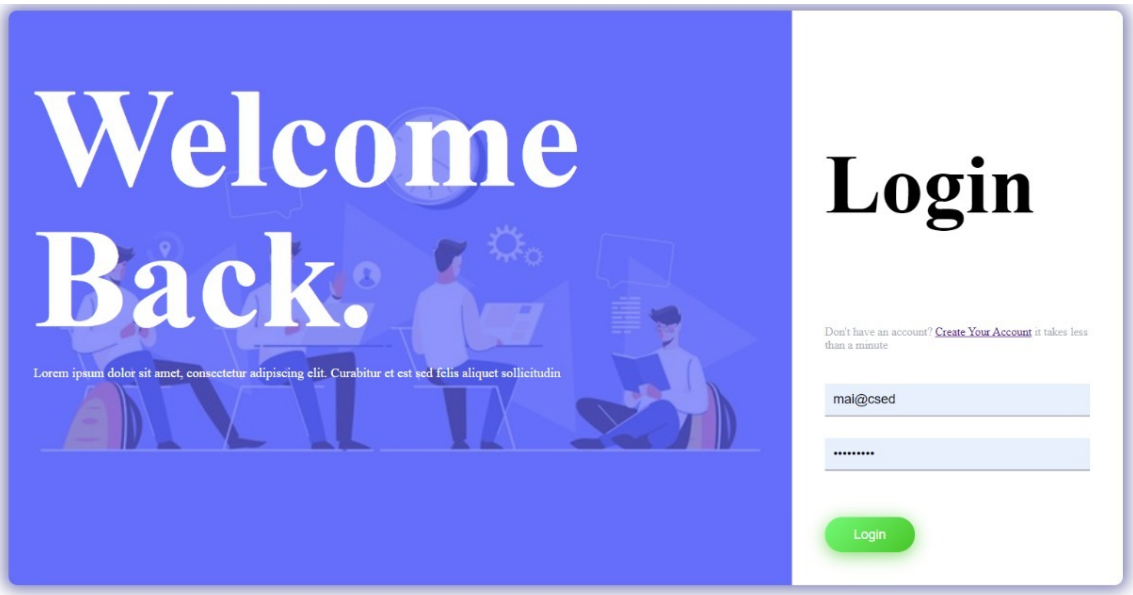
Snapshots of the UI and a User Guide:

1) When you open the application it opens on the login page , Click on the “create new account” link. The user will first sign up in the sign up page and he should at least enter his name and email, and password.

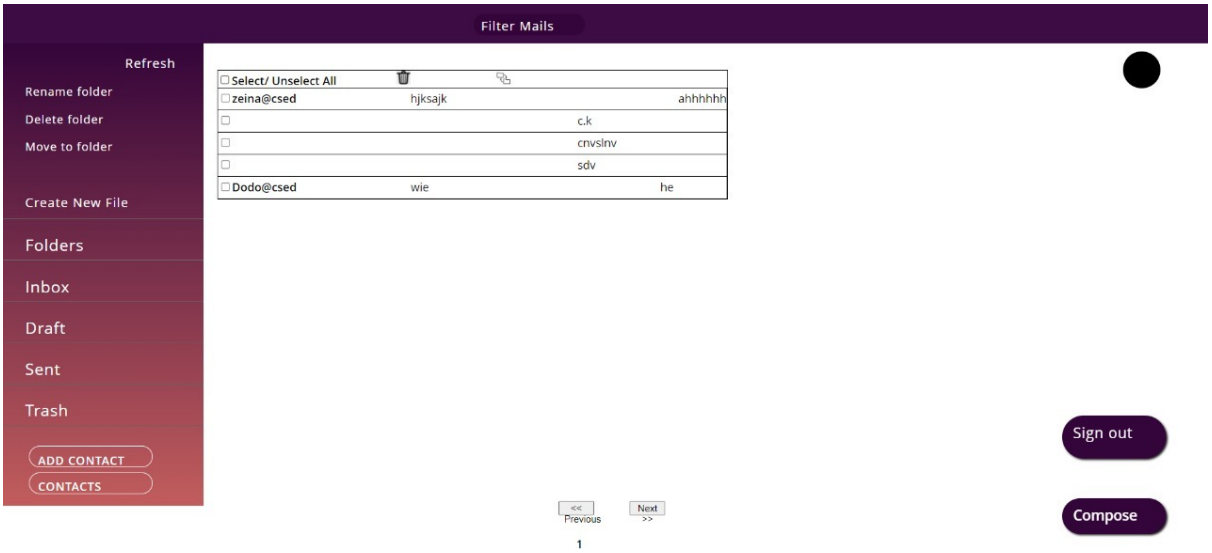


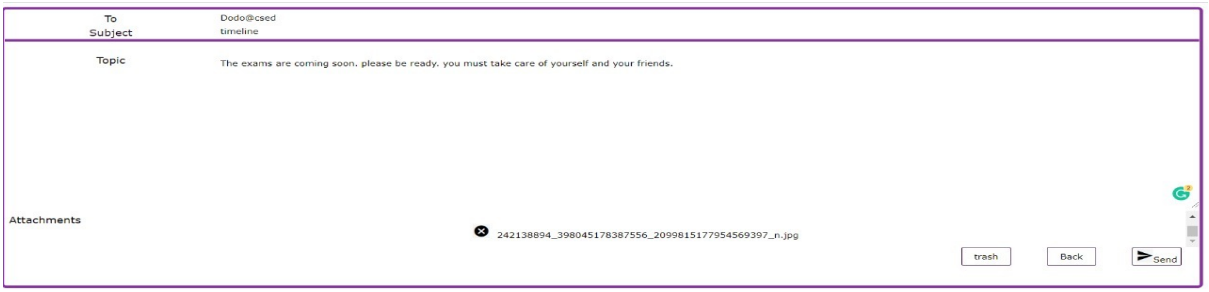
2) Then the home page will appear to him containing all tools he can use in the mail.

3) If the user already has an email, he can easily log in by entering his email and password correctly and then his email will open to his home page (i.e defaulted: Inbox messages).



4) The user can easily compose a mail by clicking the compose button, then he should fill all the required parts of the mail (the subject is optional to be filled), then click send.





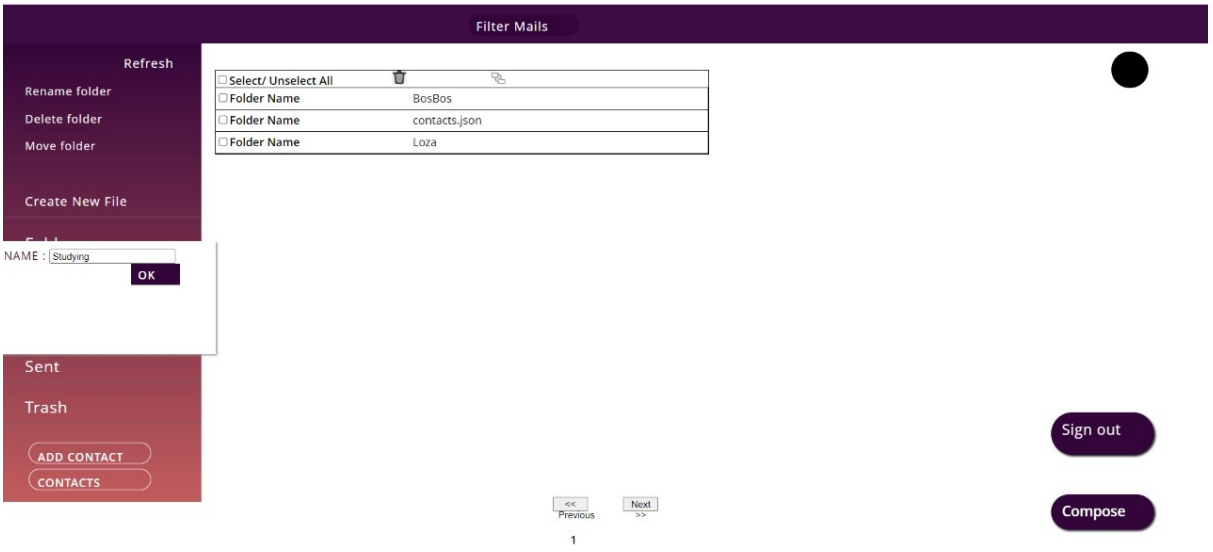
5)If the user didn't click send and he clicked the back button, it will take him to the home page but the message (mail) will be saved as a draft.

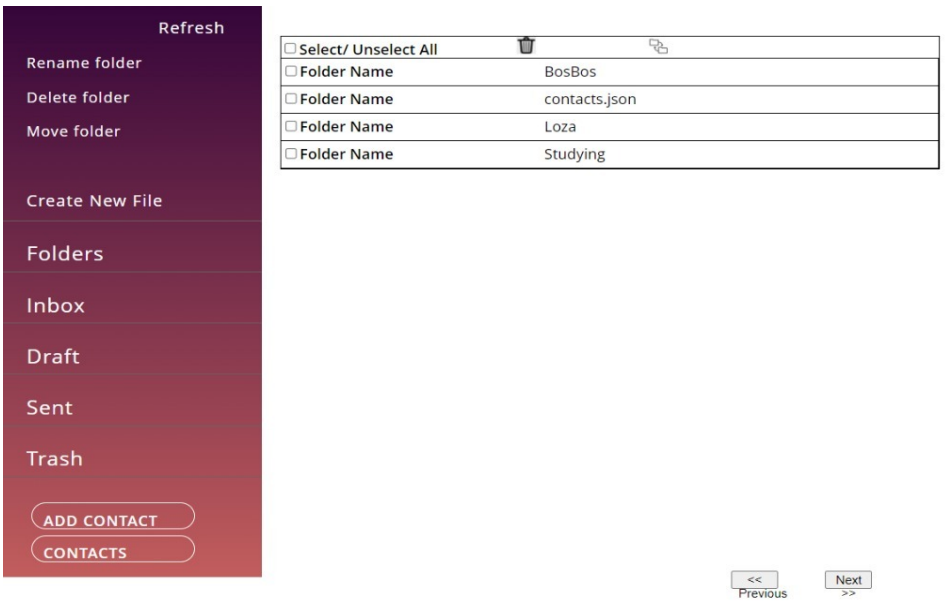
6)If the user wanted to go back to the home page and discard the mail he was writing, he could easily click on the trash button.

7)The user can easily navigate through his inbox, send, draft, trash, add contacts,or added folders by clicking on their corresponding buttons in the toolbar and click on any mail to open.

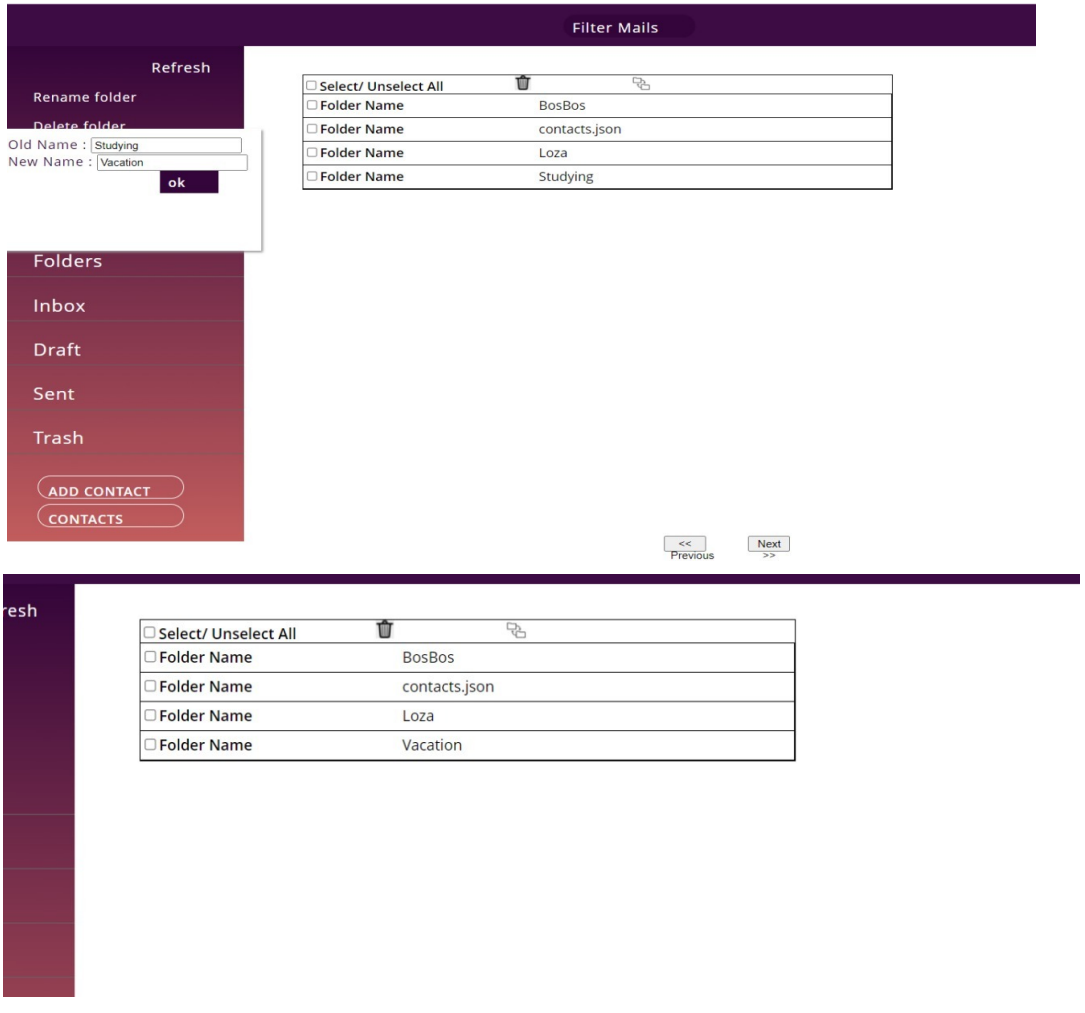
8)The user can add, rename, or remove a folder.

Add Folder:

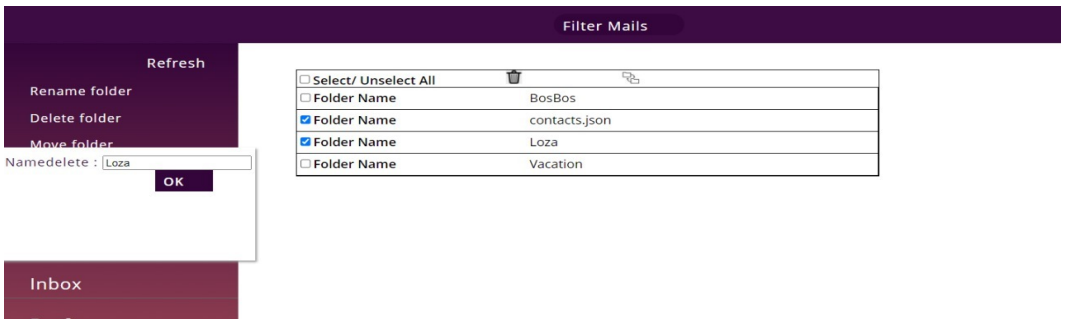




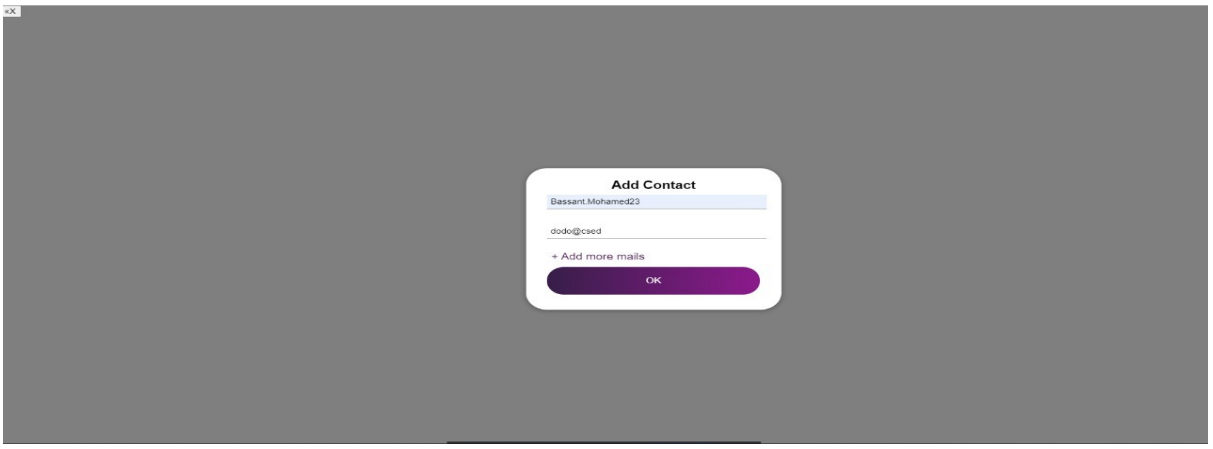
Rename Folder:



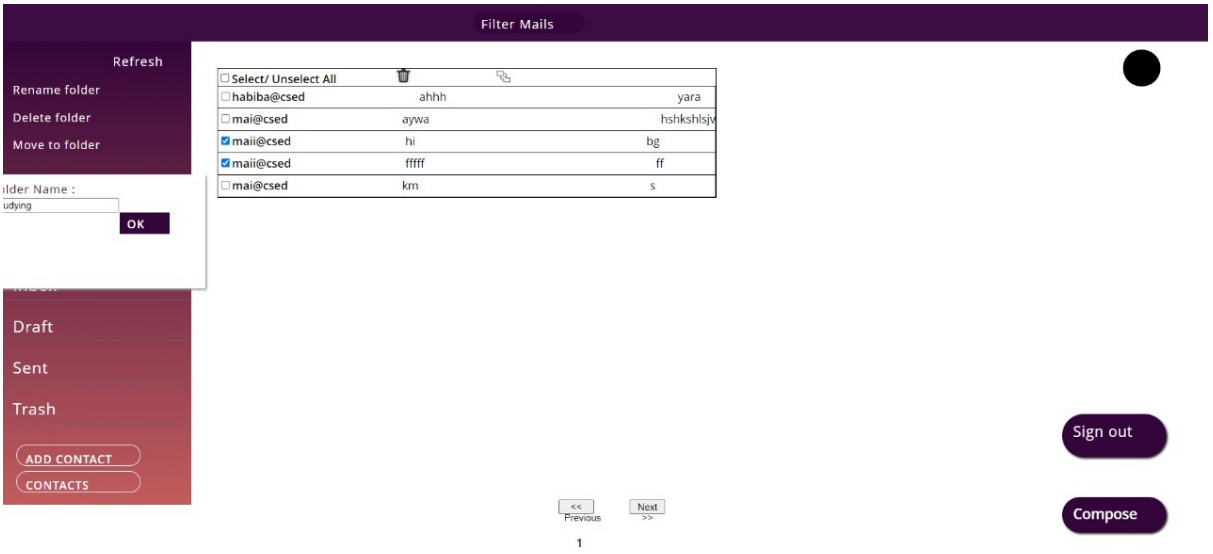
Delete Folder:

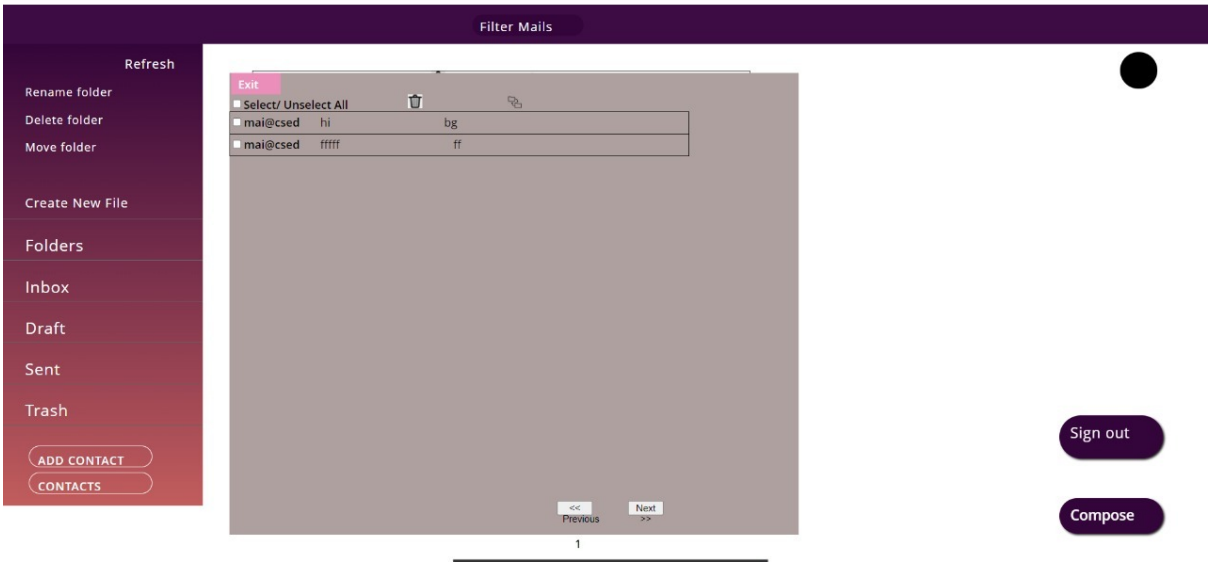


9) The user can add and remove contacts.

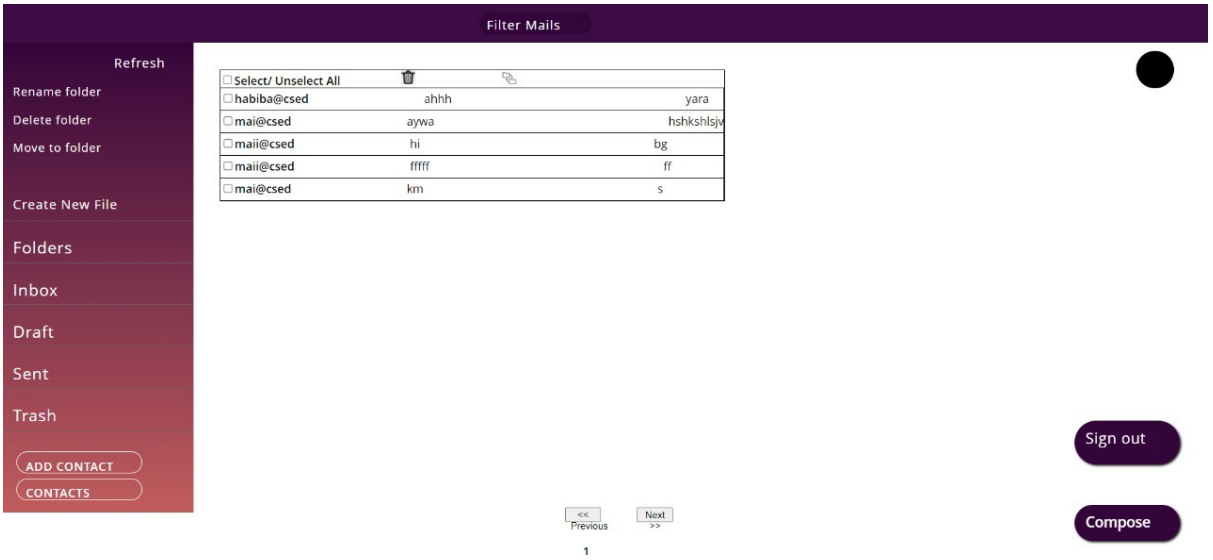


10) The user can move specific mails from a folder to any other folder.



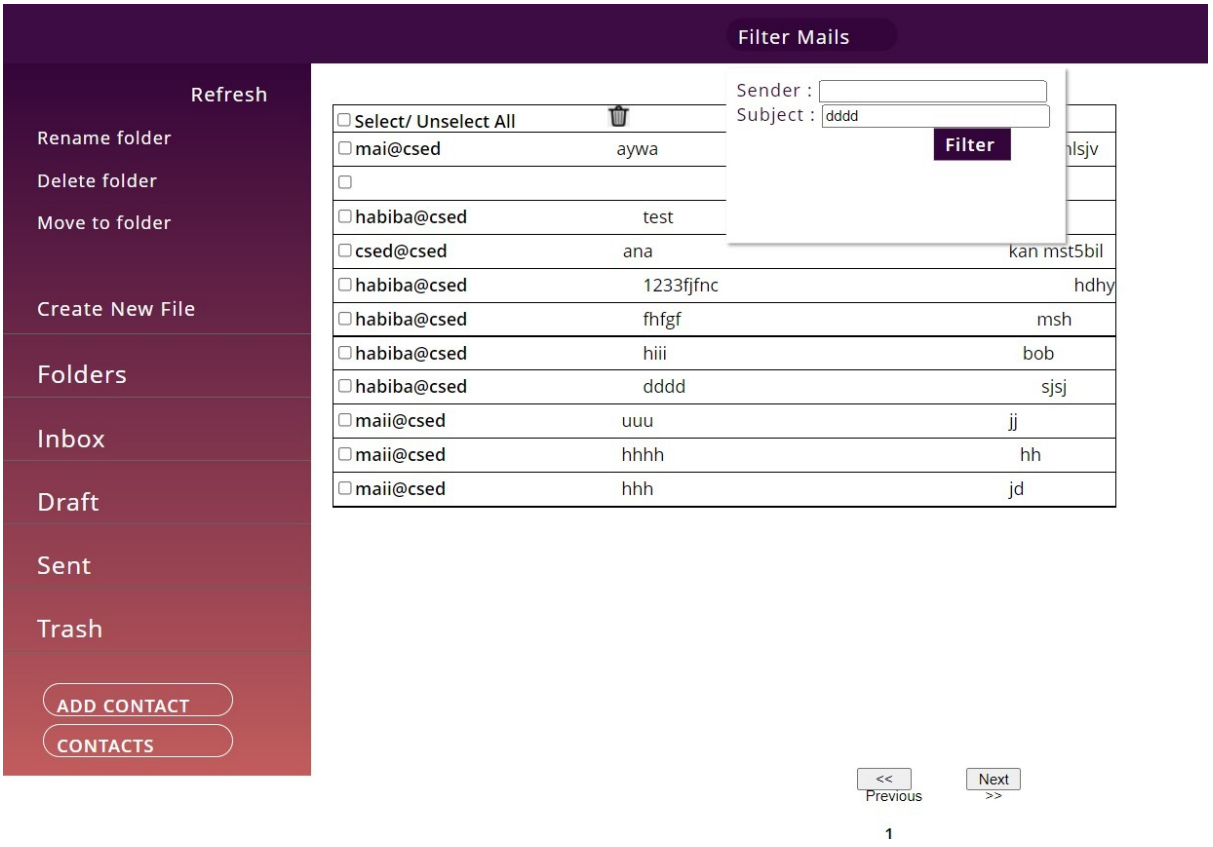


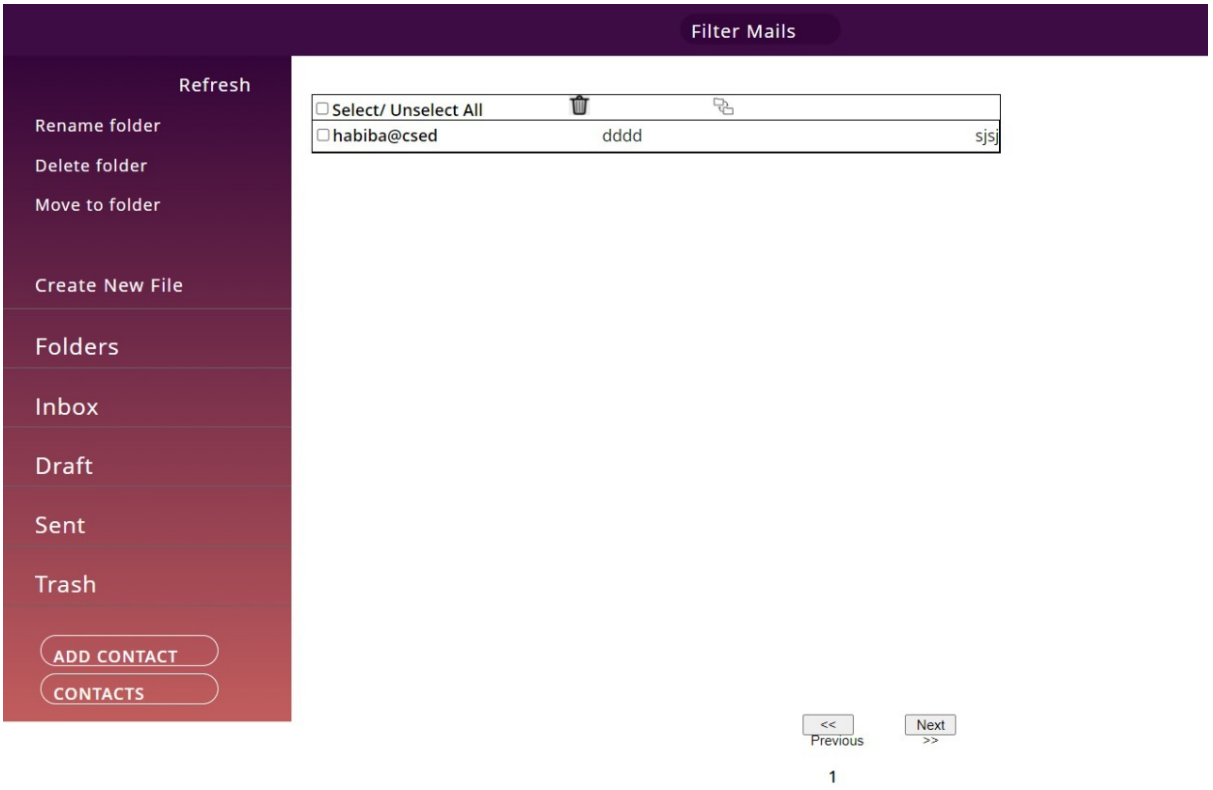
11) The user can easily delete emails,and they will be saved in trash, and this mail will be deleted after 30 days.



12)The user can send mail to any number of other emails.

13) The user can filter his mails concerning the sender or the subject.





14)The user should select the mails before rename, move, or delete.

15) If you want to refresh the page, use the refresh button.

16)the user can't add contact or send message to account which doesn't exist in our data

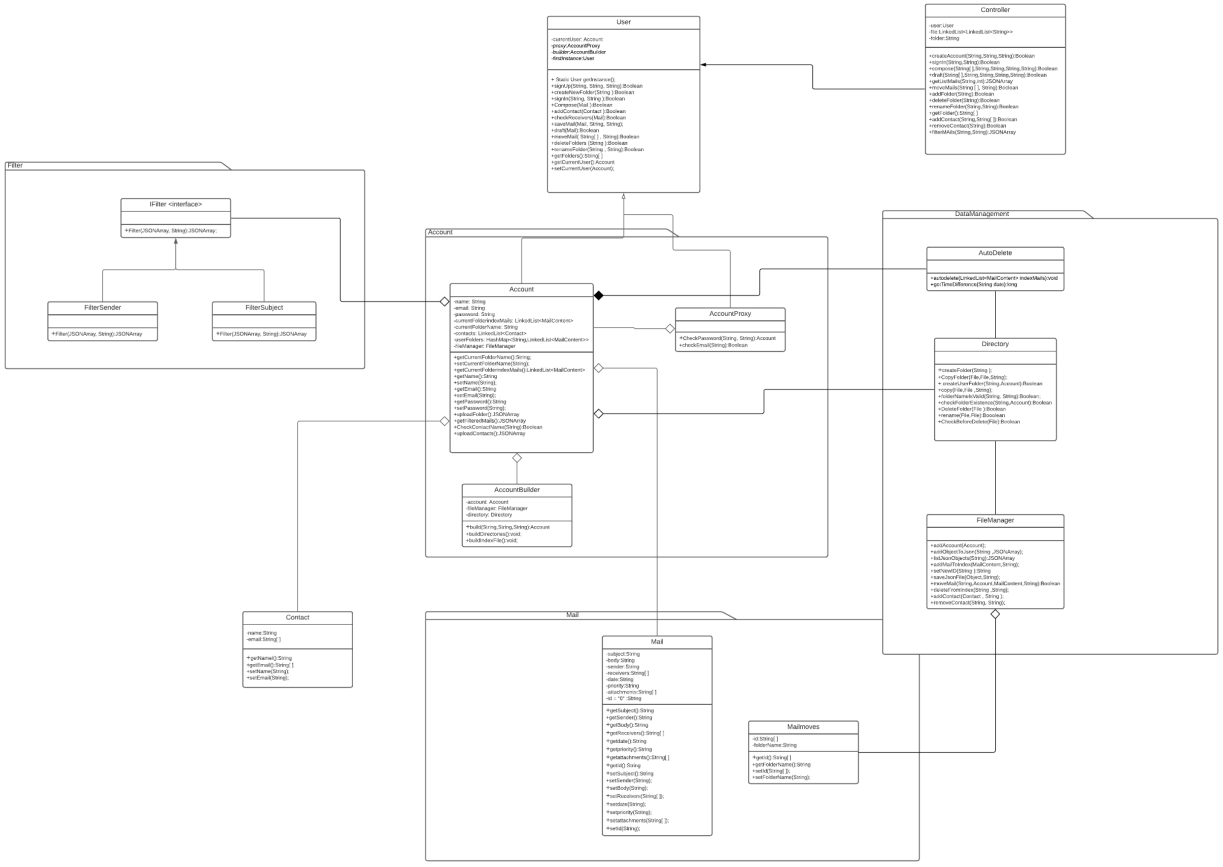
Assumptions:

1- Attachment can appear only before sending the mail, and the user can see what is inside it before sending, and there is no priority for the mails.

2) The user can't sort or search for specific mail, or edit a contact after being added.

UML Class Diagram:

Here is the link:
https://lucid.app/lucidchart/4109452d-add5-434b-94ea-d6c142770e58/edit?invitationId=inv_c1d44f93-eea2-42a7-9e24-699f0462e75b



how we have applied the required design pattern in the code:

Purpose:

Proxy Design: It is used to control the access and logging of clients and after it finishes(checks access) it passes the access to the service body.

Builder design : it is used to construct a complex object step by step and the final step will return the object .the process of constructing an object should be generic so that it can be used to create different representations of the same object

Filter: is used to filter a set of objects using different criteria and changing them in a decoupled way through logical operations

singleton: is responsible to create an object while making sure that only single object gets created. This class provides a way to access its only object which can be accessed directly without need to instantiate the object of the class.

Implementation of design patterns:

1- proxy is used in signup and login as in sign up it first checks the email if present before or not then if it wasn't present before it accesses the request to the account class. And in case you sign in, check the email and password first, then according to it enable access to the server(account) or not.

2- Builder is used to create the complex attribute of the account and also in mails .as it creates the account object whose attributes name email , password , name and create the directories of the account.

3- the filter is used to to filter emails according to subject and sender

4- the singleton class in our implementation is user class as it can be accessed by `user = User.getInstance();`