SWIFT-MAIL (SMAIL) APPLICATION DEVELOPMENT

A Computer Networks Laboratory Mini Project Report

Submitted By

Sandhiya M

2020506078

Sivaranjani S

2020506091

Bachelor of Technology

In

Information Technology

Department of Information Technology

Madras Institute of Technology

SMAIL APP

ABSTRACT

Communication between people have seen different phases and today it totally based on the electronic message transmission. Internet has made all to connect and communicate anywhere and at any time. This enables us to communicate with the people who is around hundreds of miles apart in an efficient and quick way. Email is an application that is predominantly used today to lead our daily communication with people and have formal communication with office and managers. The part of email in communication has been powerful since it enables secured transmission and reliable transmission through various protocols that exist. Some of the well-known protocols that is used in email communication is Transmission Control Protocol (TCP), Simple Mail Transfer Protocol (SMTP), Post Office Protocol (POP) and Internet Message Access Protocol (IMAP). The SMAIL (Swift-Mail) application aims to develop the email application that authenticates the user credential (Mail Id and password), allows user to view their inbox and send mail with attachments. The SMAIL appl replicates some of the function of Google mail application. As the name suggest Swift (without delay), the app ensures quick login and transmission of the messages and its attachment.

INTRODUCTION

The SMAIL app was developed in the motto to replicate the Gmail application and to enable quick transmission of mail. The drawback of two-step verification of Gmail app is overcome by the SMAIL. SMAIL does not need two step verification mechanism so the application faster and easily accessible from anywhere.

Objective:

- To enable quick transmission of mail.
- To ensure secured transmission.
- To provide better UI/UX for the application.
- To avoid two-step verification delay problem by using application password of length 16 that consists only of characters.

TOOLS USED

- **UI/UX development**
  - ✓ Java Swing
- **SDE Environment**
  - ✓ NetBeans

PROTOCOLS USED

Protocol	**Usage**
Simple Mail Transfer Protocol (SMTP)	To transfer mail from one user to another.
Post Office Protocol (POP)	To retrieve all the emails from the server,

- **Credentials:**

In order to use SMAIL app one must have a mail id and its app password which is different from their regular password. The app password is generated from the security and privacy panel of the account.

Simple Mail Transfer Protocol:

Most internet systems use SMTP as a method to transfer mail from one user to another.

- ✓ SMTP is a push protocol.
- ✓ SMTP is an application layer protocol.
- ✓ The SMTP server is an always-on listening mode.
- ✓ The SMTP process initiates a connection through port 25.

Some SMTP Commands:

HELO – Identifies the client to the server, fully qualified domain name, only sent once per session

MAIL – Initiate a message transfer, fully qualified domain of originator

RCPT – Follows MAIL, identifies an addressee, typically the fully qualified name of the addressee, and for multiple addressees use one RCPT for each address.

DATA – send data line by line.

POST OFFICE PROTOCOL Version 3 (POP3):

- ✓ POP3 client is installed on the recipient system.
- ✓ POP3 server is installed on the recipient's mail server.

Working of POP3 Protocol:

- ✓ To establish the connection between the POP3 server and the POP3 client, the POP3 server asks for the user's name to the POP3 client.
- ✓ If the username is found in the POP3 server, then it sends the ok message.
- ✓ It then asks for the password from the POP3 client; then the POP3 client sends the password to the POP3 server.
- ✓ If the password is matched, then the POP3 server sends the OK message, and the connection gets established.
- ✓ After the establishment of a connection, the client can see the list of mails on the POP3 mail server.
- ✓ In the list of mails, the user will get the email numbers and sizes from the server. Out of this list, the user can start the retrieval of mail.
- ✓ Once the client retrieves all the emails from the server, all the emails from the server are deleted.
- ✓ Therefore, we can say that the emails are restricted to a particular machine, so it would not be possible to access the same mails on another machine. This situation can be overcome by configuring the email settings to leave a copy of mail on the mail server

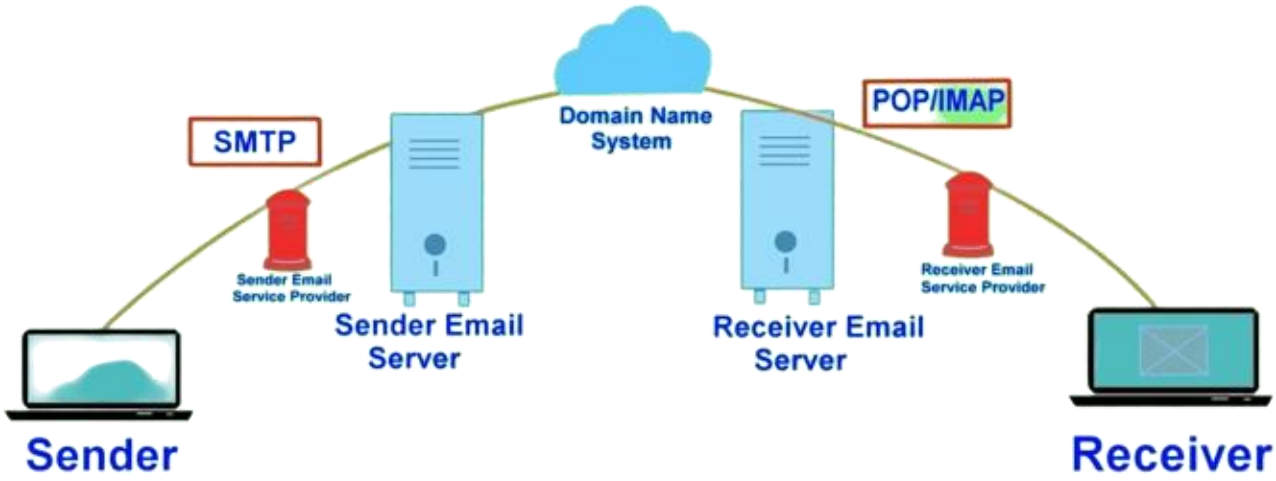
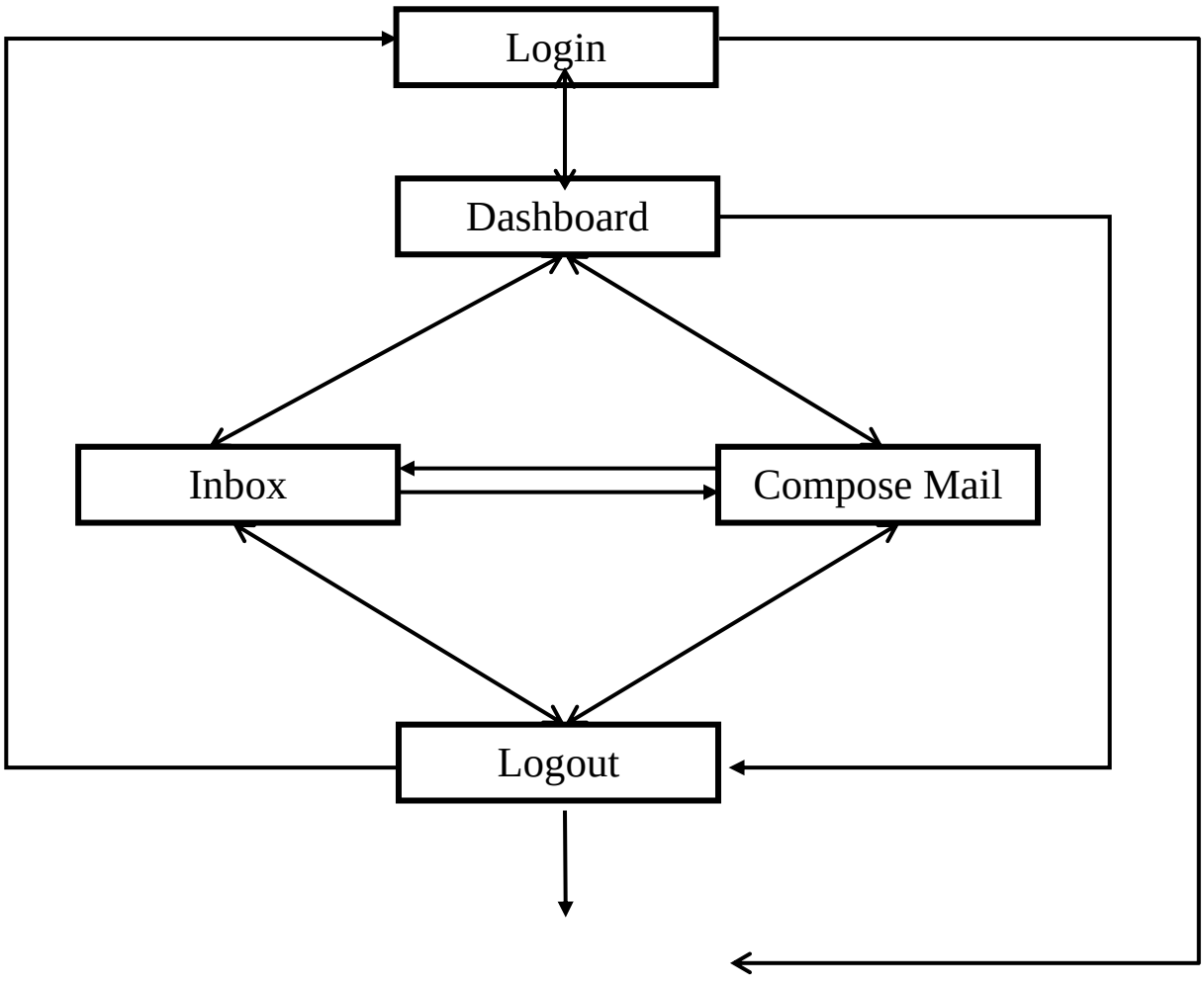


Fig. Working of Mail

ARCHITECTURE



Exit

SMAIL App Architecture

*The double-sided arrows represent that they are navigate able from each other.

LOGIN

This frame enables the user to enter their email address and their email application password. Various validation methods were used to ensure correctness of the input from the user.

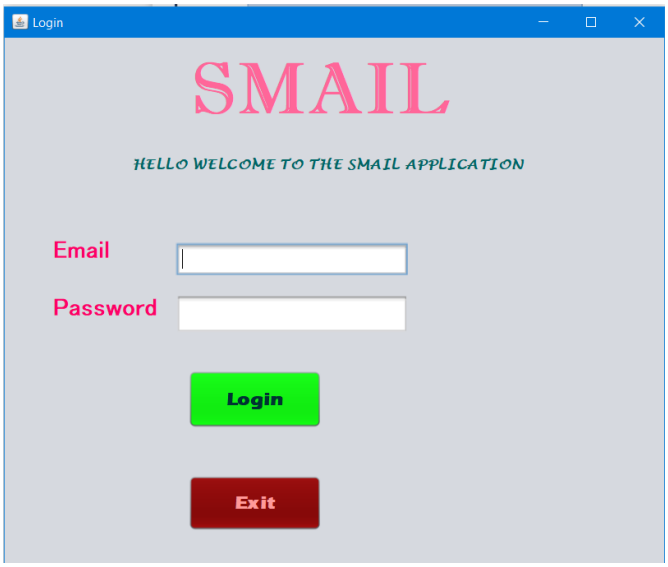


Fig. Login frame

Validation methods:

- **Null input verification:** NULL values are checked and reported with JLabels.

- **Email validation:** Using JAVA regular expression the email pattern is validated.
- **Email and password credential Validation:** The given mail and password are authenticated using **Authenticator class.**

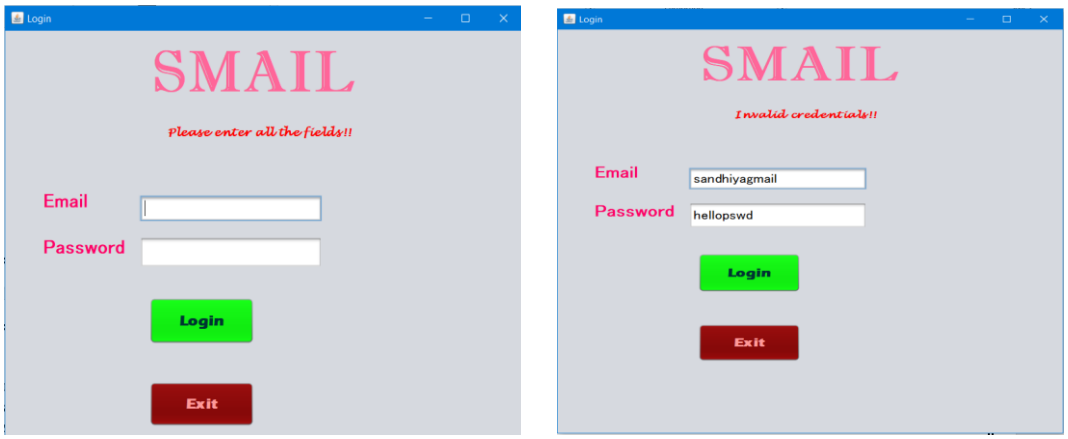


Fig. Validating inputs

When all the validation is found to be true global variables of email and password are set and the login Frame disappears and the Dashboard frame appears.

DASHBOARD

This frame has two navigators namely Inbox and Compose which navigates to their respective page. The global variables email and password are validated for the null values.

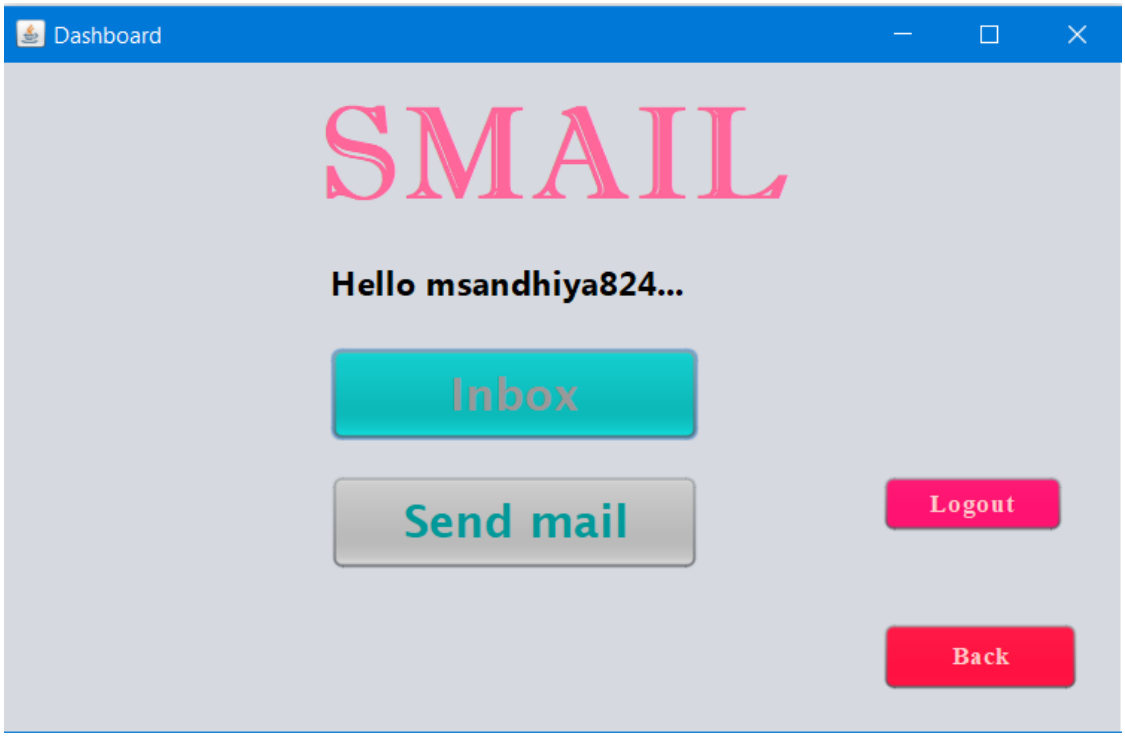


Fig. Dashboard Frame layout

INBOX

- **Filtering**
  - ✓ The inbox has the feature of selecting the month such that the mails received on these months would be displayed.
- **Validation**
  - ✓ Initially, global variables emailid and password are validated for not null values. If null values found, it is reported using the JLabel embedded in the Frame.
- **POP setup**
  - ✓ The POP3 protocol is used to read all the mails from the server and the read mail are filtered according to the given month.
  - ✓ The Session is setup with various properties of the POP3 protocol to ensure secure and reliable retrieval of mail from the server.
- **Reading Mail**
  - ✓ Store class is used to establish the POP3 protocol and connect the user credentials.
  - ✓ Folder class is used to retrieve the Inbox from the server.
  - ✓ Folder is opened and all the mails are fetched from it using Message class.
  - ✓ Now this message class is traversed to find the mails of the required month and displayed in the various components of SWING embedded in the Frame.

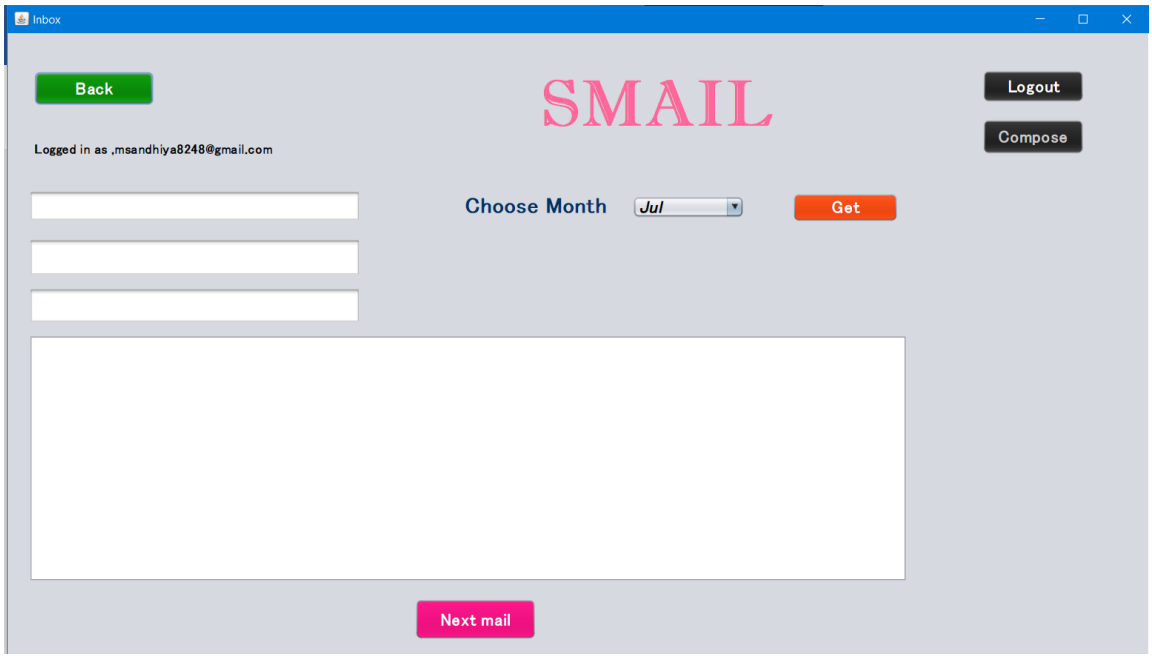


Fig. Inbox Layout

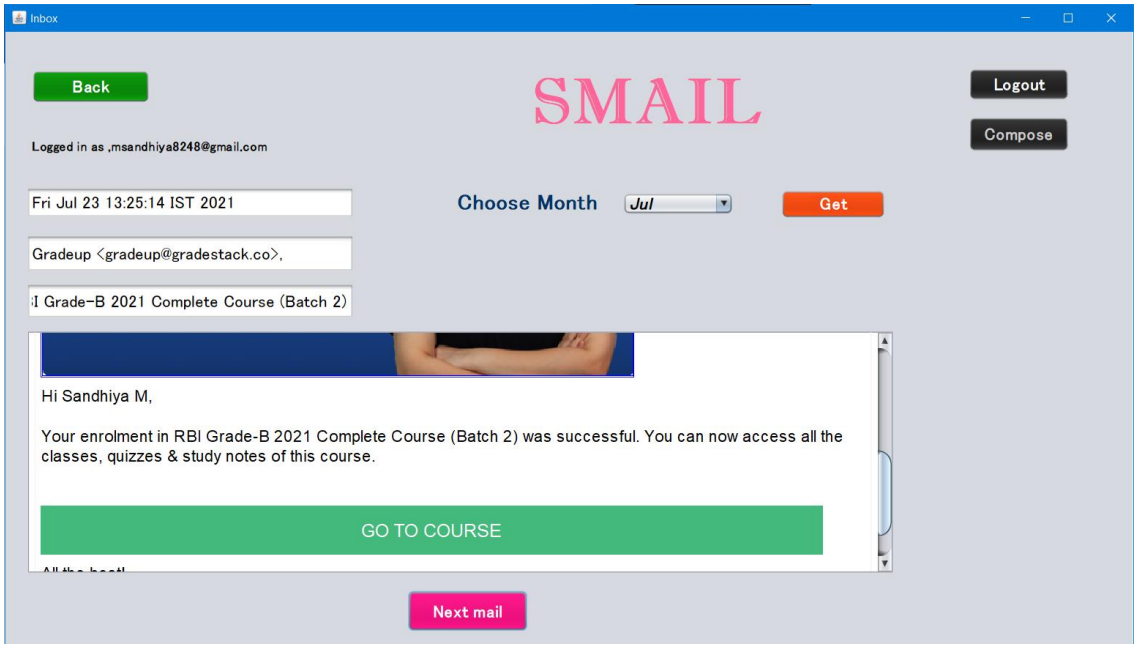


Fig. Inbox with mail display

COMPOSE MAIL

- **Validation**
  - ✓ Initially, global variables emailid and password are validated for not null values. If null values found, it is reported using the JLabel embedded in the Frame.
- **GUI**
  - ✓ This frame provides a UI to user to send the mail with attachments.
  - ✓ User where given UI to give the to-mail address, Subject of the mail, Body of the mail and the file chooser option to choose the file from the local storage.
- **SMTP connection**
  - ✓ SMTP protocol is established and connected to send the mail with the attached files* of any type.
- **Sending Mail**
  - ✓ Mime Message class is used set all the parts of the mail that is to be sent.
  - ✓ Multipart class is used to attach the chosen file from the local directory using file chooser after validating not to be null.
  - ✓ Files are attached to multipart using Data source and Data handler class.
  - ✓ Now, the multipart is added the Mime Message and sent.
  - ✓ Transport function is used to send the set Mime message

*The files are attached with the SMTP message through cloud provider (here OneDrive is used).

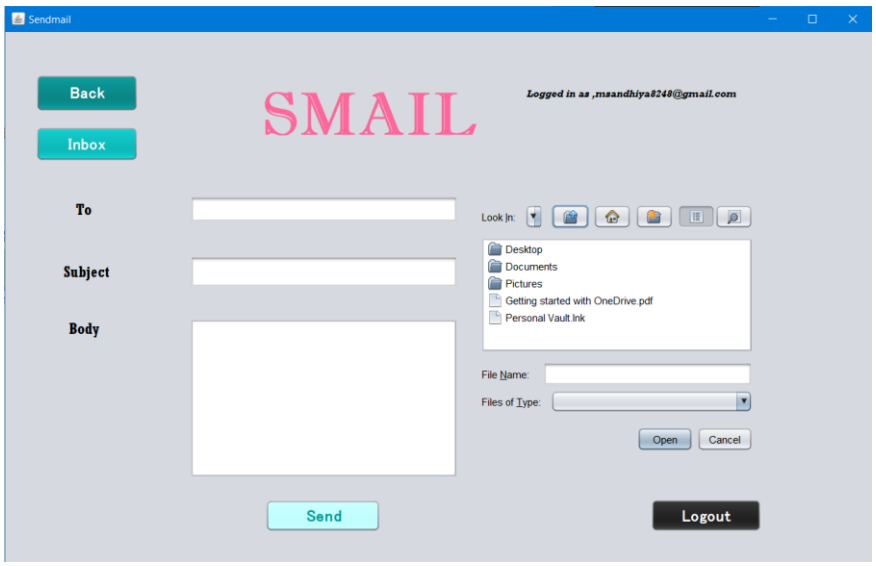


Fig. Compose Frame

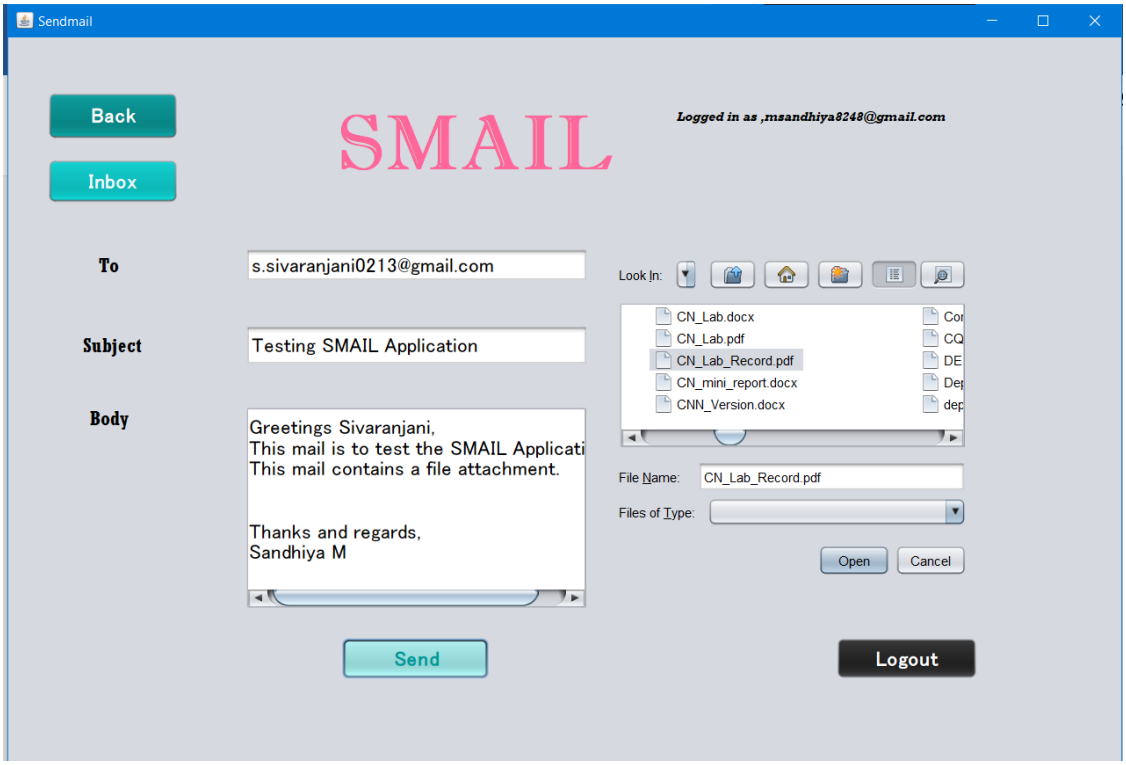


Fig. Sending mail with file attachment

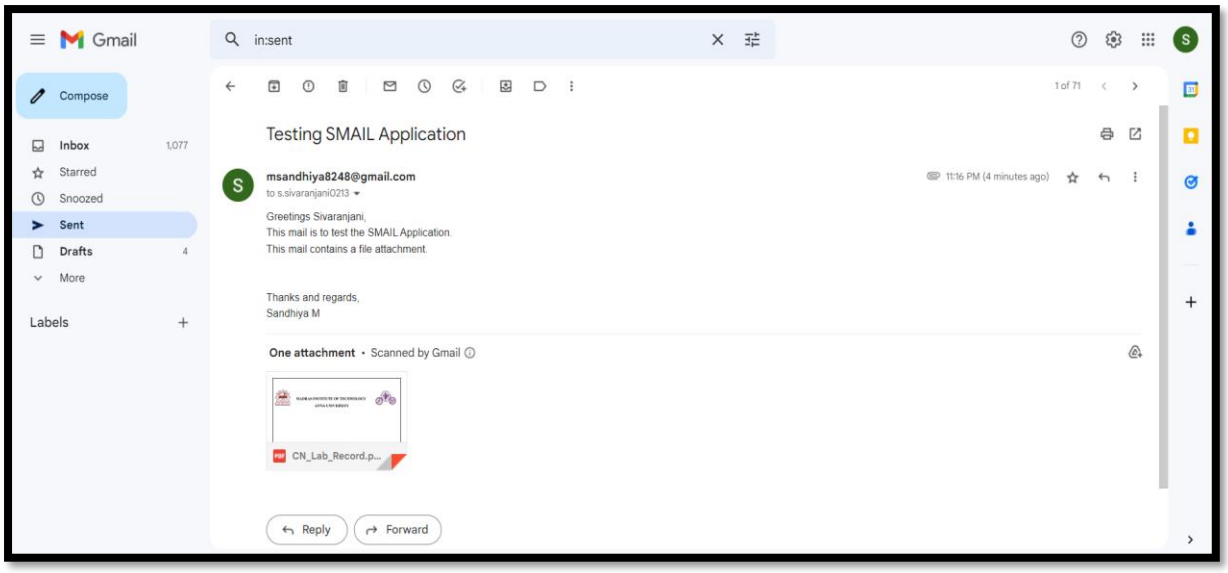
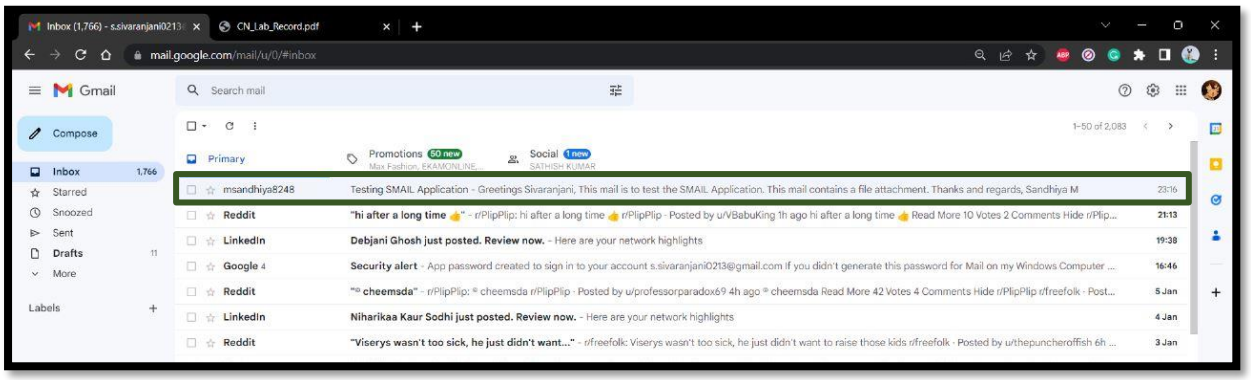


Fig. Verified with the Gmail sent box of the sender



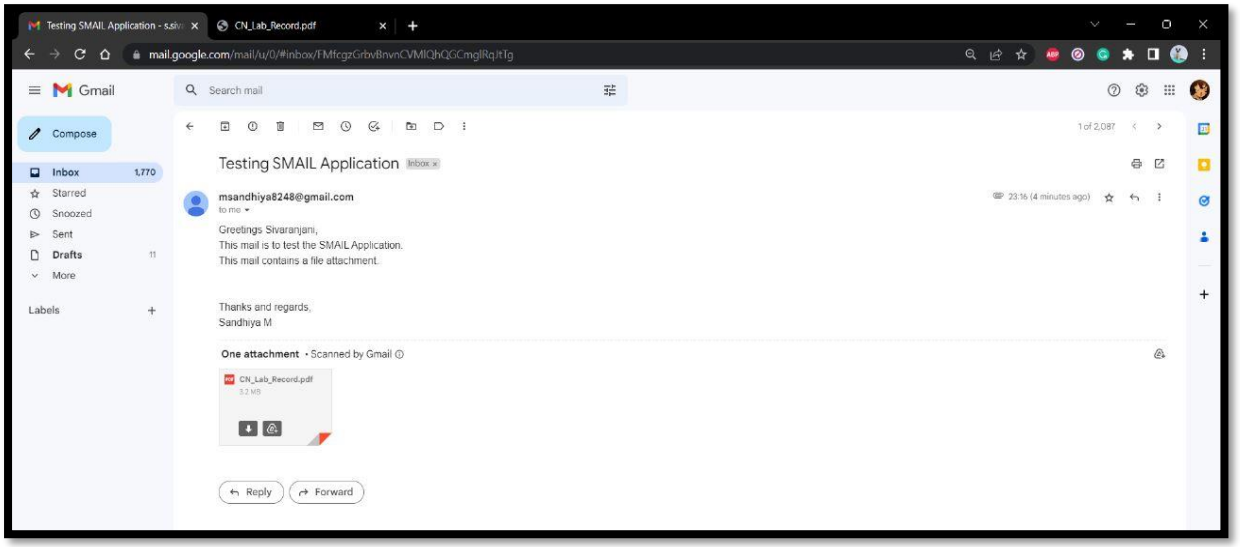


Fig. Received mail verified with Gmail

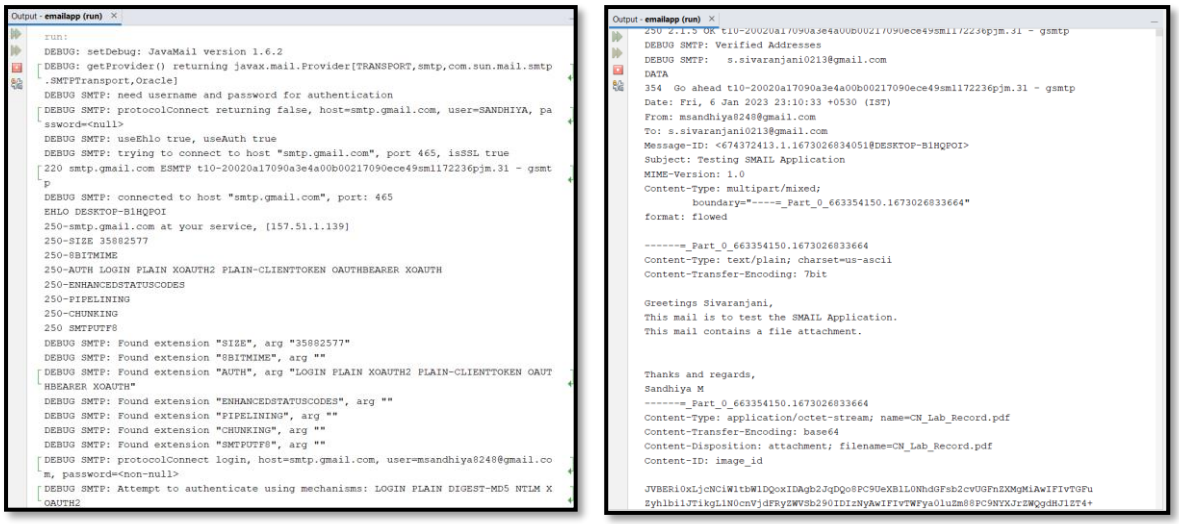


Fig. Backend of sending mail

LOGOUT

- This makes the global variables emailid and password to null values such that navigating to Inbox and Compose would show up the message to login.
- Every frame has a JButton component to set the null values to the global variables.

EXIT

This option is used to dispose all the frames that are visible and stops running the app.

FUTURE DEVELOPMENT

- The app can be improved to show all the messages in the inbox like Gmail.
- The app can be improved to filter the inbox messages with various parameters like by from address, by date, by time etc.
- The app can be deployed and can be even used in mobile phones.

CONCLUSION

Thus, the SMAIL app ensures easy, secure, and reliable transfer of the mail. It enables quick delivery of mail and it also provides better UI/UX. The application has achieved all the objectives mentioned earlier. This application has

achieved to attach the file easily by providing a file chooser menu in the Compose Frame. This application also enables secure transmission by authenticating the mail and passwords of the user in the login frame itself. The app is used for quick mail sending because it does not ask for two-step verification when logged in from another device. But still the mail is secured with the application password generated by the user.