Requirements and Analysis Document for MailMe

authors

Martin Fredin

Elin Hagman

David Zamanian

Alexey Ryabov

Hampus Jernkrook

date 2021-09-07
version 1.0

1 Introduction

1.1 Purpose of application

The purpose of this project is to create a simple email client. The application will support basic features that can be found in similar email clients.

1.2 General functionality of application

The application will be a standalone cross-platform desktop application with a graphical user interface (GUI) and will be able to run on Linux, Windows and MacOs.

With our application, users will be able to connect with an already existing email account to an email service provider to receive, read and send emails. There will be additional options to reply to and forward received emails, as well as attach files to emails.

1.3 Scope of application

The application intent is not to be able to compete with already existing email clients similar to ours such as Thunderbird or the one offered by Apple. The application should be able to support simple email client features. The key focus will be on constructing a good object-oriented design for the application.

1.1 Definitions, acronyms, and abbreviations

- **ESP (Email service provider)**: The email server of the email domain provider.
- **POP3**: Short for Post Office Protocol version 3. This is the third version of the Post Office Protocol, which is an application-layer internet standard protocol used by email clients to retrieve email from an email server. (Post Office Protocol, 2021)
- **SMTP**: Short for Simple Mail Transfer Protocol. This is an internet standard communication protocol for sending and receiving emails. (Simple Mail Transfer Protocol, 2021)
- **Port**: A communication endpoint of the computer's operating system. The port number depends on the IP address of the host and the transport protocol used for the communication. (Port, 2021)
- **Host**: A device connected to a computer network and assigned at least one internet address. Hosts may work as servers offering services to users or other hosts on the computer network. (Host, 2021)
- **Attachment**: File sent bundled within an email.
- **Account**: An account existing on a supported ESP.

2 Requirements

2.1 User Stories

Green annotation means that a user story is fully implemented. Yellow means it is partially implemented (acceptance criteria may have been added as we discovered certain aspects of the application). Red means that the user story is not implemented at all (at least not so that it can be used in the application by the user). Orange means that an acceptance criteria is almost met, but may fail sometimes due to bugs.

#User story
Story Identifier: add_gmail_account
As a: user
I want to: be able to login to my existing gmail account.
So that: I can use my existing account.

Acceptance:

- User is able to submit an account with account details (email and password) upon launching the application.
- User is able to enter and submit a new email address and password for an existing account in a separate addAccountView.
- User will get feedback if the email was successfully connected.
- User will get feedback if the email could not be added due to rejection from the server.
- User will get feedback if the email provided is not supported by the application.

#User story
Story Identifier: login only once
As a: user
I want to: only have to submit my account details once
So that: I can use my submitted account without having to login to it every time I launch the application.

Acceptance:

- The user's account details are stored locally on the user's computer and read by the application when necessary.
- If the account has already been added and the user tries to add it again, the application's stored info connected to the submitted account should not be overridden, in order to keep other data already saved and linked to the account.

#User story
Story Identifier: choose active account
As a: user
I want to: have a list of my connected accounts.
So that: I can select one as active.

Acceptance:

- The user gets to choose from a list of added accounts when launching the application.
- User is able to get a drop-down of added accounts.
- User is able to select an account in the drop-down.
- The active account state is updated accordingly.
- User receives visual feedback of which account is active

#User story
Story Identifier: email overview
As a: user
I want to: have a list of previews of all emails in the currently selected folder.
So that: I can get an overview of my email feed.

Acceptance:

- User should be able to see a list of emails displaying the sender, subject and date.
- The list should contain all emails within the currently selected folder.
- The list should be retrieved from the collection of emails stored in the application's storage.

#User story
Story Identifier: writing emails
As a: user
I want to: be able to write a new email to any other email user with any other email address.
So that: I can communicate with other email users.

Acceptance:

- User should get a new window to write the new email in.
- User will be able to input the recipient of the new email.
- User will be able to input the subject of the new email.
- User will be able to input the content of the new email.
- User will be able to send that particular email.
- The 'from' field of the written email is automatically sent to the user's active account's email address.
- User should get a notification if the email was successfully sent.
- User should get a notification if the email could not be sent.
- The sent email should be stored in the application's storage under the 'Sent'-folder.

#User story
Story Identifier: email folders
As a: user
I want to: see a list of my email folders.
So that: I get an overview and can navigate between them.

Acceptance:

- User sees a panel with the different email folders.

- User is able to navigate between the folders by clicking on them.
- Clicking on a folder should update the email overview accordingly, with the emails contained in that folder.
- The folder button should be linked to a list of emails, or else retrieve the emails from storage.

#User story
Story Identifier: read emails
As a: user
I want to: be able to read my emails.
So that: I can be up-to-date with my emails.

Acceptance
- User will be able to select a specific email from many emails within an email overview.
- User will be able to open specific emails and read their content with a new reading view.
- The single email button should either be linked to a specific Email object or else retrieve the corresponding email from storage.
- The application is able to show HTML content
- The application is able to read and receive attachments

#User story
Story Identifier: refresh from server
As a: user
I want to: be able to refresh my inbox.
So that: I can see new ones that I have received.

Acceptance
- User will be able to request a refresh from the server by pressing a button.
- The overview with the emails in the inbox should be updated with the emails received from the server, in addition to any emails that were previously in the folder.
- The storage is updated with the newly fetched emails, in addition to any previously stored ones.

#User story
Story Identifier: attach files
As a: user
I want to: be able to attach files to my emails that I am writing
So that: I can send files to other email users.

Acceptance
- User can press an ‘Attach’-button in the writing view to initialize the attach-process.
- Upon pressing the ‘Attach’-button, the user can choose files to attach to the email from their file explorer.
- Emails with attachments can be sent via the server.
- In the ‘Sent’-folder, the user can see what files were attached to a given email when reading it.

- Emails can be stored with information about potential attached files.
- User receives visual feedback that the file is attached

#User story
Story Identifier: save drafts
As a: user
I want to: be able to save email drafts
So that: I can start composing an email and continue to write it at a later time.

Acceptance
- User is notified that the currently written email was saved in ‘Drafts’ when closing down a non-empty email without first sending.
- The closed non-empty email is stored in storage under folder ‘Drafts’.
- The user is able to click an email in the ‘Drafts’ overview to continue writing on that email.
- Each time the user closes down the same non-empty email from the writing view, the corresponding stored email gets updated with the new email components.
- When the user sends an email stored in ‘Drafts’, the email is moved to ‘Sent’.

#User story
Story Identifier: delete account
As a: user
I want to: be able to delete an account from the application’s storage (not server)
So that: I can clean up among my used accounts.

Acceptance
- User can click a button to delete a given account within the application.
- User is notified of the outcome of the deletion of the account.
- The deleted account is removed from the application’s storage, along with all emails and other data connected to that account.
- If the deleted account was the active one, the user is prompted to select another active account or add a new account to use.

#User story
Story Identifier: reply to received
As a: user
I want to: be able to reply to received emails
So that: I can quickly answer other email users that have contacted me.

Acceptance
- User can press a button to reply to the received email.
- A writing view is opened upon pressing the reply button.
- The ‘to’ field of the reply-email is automatically set to the sender of the received email.
- The ‘subject’ field of the reply-email is automatically set to the subject of the received email.
- The content of the received email is concatenated with the content of the user’s reply.
- The functionality in other regards adheres to the same acceptance criteria as writing

emails in general.

#User story
Story Identifier: forward received
As a: user
I want to: be able to forward my received emails
So that: I can quickly pass along a message to another email user.

Acceptance
- User can press a button to forward a received email.
- A writing view is opened upon pressing the forward button.
- The ‘subject’ field of the reply-email is automatically set to the subject of the received email.
- The content of the received email is concatenated with the eventual content of the user’s additional message.
- The functionality in other regards adheres to the same acceptance criteria as writing emails in general.

#User story
Story Identifier: filter on to
As a: user
I want to: be able to filter my emails based on a specific to-address
So that: I can quickly find specific emails.

Acceptance
- User can enter a text within a to-textfield for the filter.
- User is notified about the outcome of the filtering process.
- User can see what filters are currently applied.
- User can clear the filter.
- The overview displays exactly those emails within the selected folder that fulfil the active filtering condition (user-entered substring is part of one of the to-addresses).
- The overview displays all emails within the selected folder upon clearing the filter.

#User story
Story Identifier: filter on from
As a: user
I want to: be able to filter my emails based on a specific from-address
So that: I can quickly find specific emails.

Acceptance
- User can enter a text within a from-textfield for the filter.
- User is notified about the outcome of the filtering process.
- User can see what filters are currently applied.
- User can clear the filter.
- The overview displays exactly those emails within the selected folder that fulfil the active filtering condition (user-entered substring is part of the from-address).
- The overview displays all emails within the selected folder upon clearing the filter.

#User story
Story Identifier: filter on date
As a: user
I want to: be able to filter my emails based on a given time interval
So that: I can quickly find specific emails.

Acceptance
- User can click on one of multiple max dates to apply the date filter.
- User is notified about the outcome of the filtering process.
- User can see what filters are currently applied.
- User can clear the filter.
- The overview displays exactly those emails within the selected folder that fulfil the active filtering condition (the email's received-date is less than or equal to the user-entered max date).
- The overview displays all emails within the selected folder upon clearing the filter.

#User story
Story Identifier: sort new to old
As a: user
I want to: be able to sort my emails by newest date to oldest date
So that: I can quickly find specific emails.

Acceptance
- User is able to click a button/toggle to make the emails be sorted.
- User is notified about the outcome of the sorting process.
- User can see what sorting condition is applied currently.
- User can clear the sorting condition.
- The overview displays exactly those emails within the selected folder that fulfil the sorting condition.
- The overview displays all emails within the selected folder, in any arbitrary order, upon clearing the sorting condition.

#User story
Story Identifier: sort old to new
As a: user
I want to: be able to sort my emails by oldest date to newest date
So that: I can quickly find specific emails.
Acceptance
- User is able to click a button/toggle to make the emails be sorted.
- User is notified about the outcome of the sorting process.
- User can see what sorting condition is applied currently.
- User can clear the sorting condition.
- The overview displays exactly those emails within the selected folder that fulfil the sorting condition.
- The overview displays all emails within the selected folder, in any arbitrary order, upon clearing the sorting condition.

#User story
Story Identifier: search
As a: user

I want to: be able to search for a specific sub-word within the emails
So that: I can quickly find specific emails.

Acceptance

- User can enter a search word into a search box.
- User is notified about the outcome of the searching process.
- User can see what search word is applied currently.
- User can clear the search box.
- The overview displays exactly those emails within the selected folder that match against the search word.
- The overview displays all emails within the selected folder upon clearing the search box.

#User story
Story Identifier: add microsoft account
As a: user
I want to: be able to login to my existing microsoft account (hotmail, outlook, live).
So that: I can use my existing account.

Acceptance

- User is able to submit an account with account details (email and password) upon launching the application.
- User is able to enter and submit a new email address and password for an existing account in a separate addAccountView.
- User will get feedback if the email was successfully connected.
- User will get feedback if the email could not be added due to rejection from the server.
- User will get feedback if the email provided is not supported by the application.

delete and move emails between folders

2.2 Definition of Done

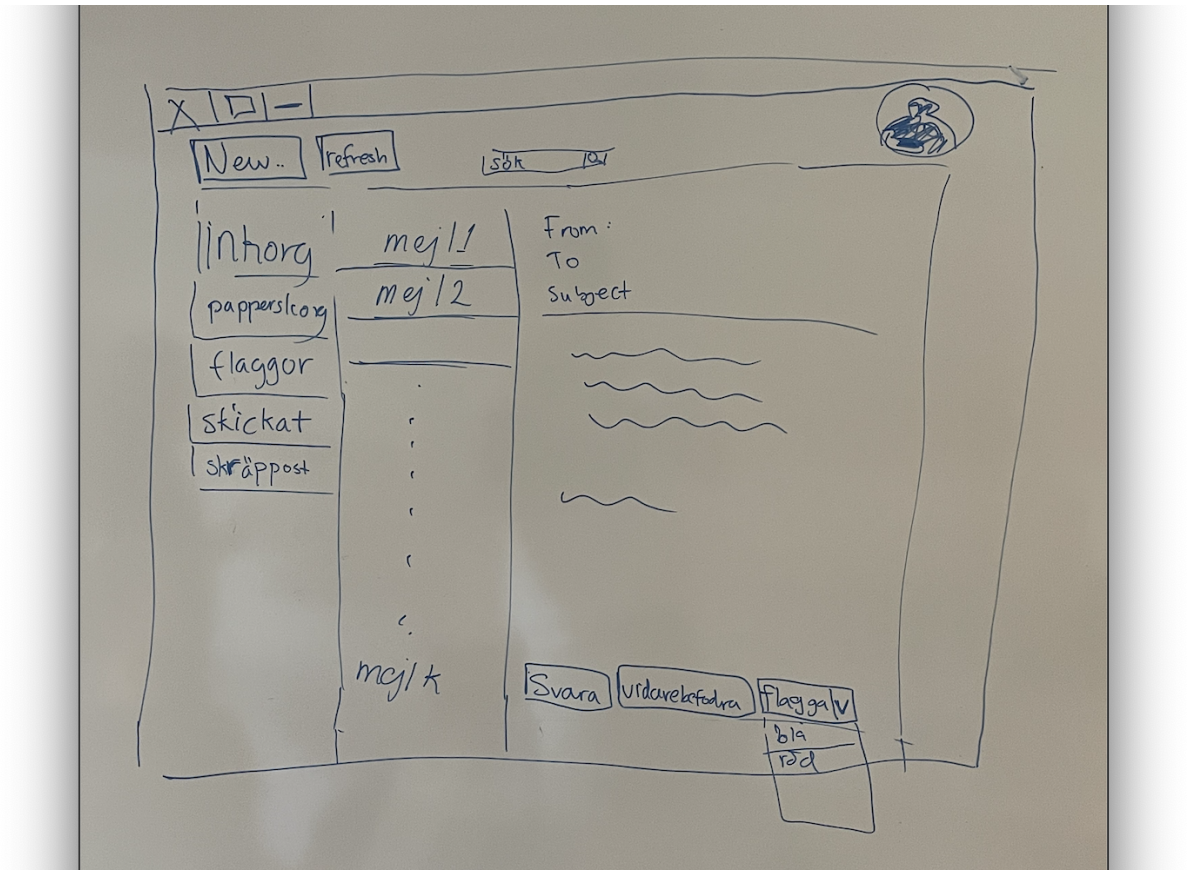
- The code should have been tested via unit tests.
- The code should have been tested by running the application and making sure everything runs.
- The code should be under the version control system git.
- The code should be thoroughly commented, including javadoc for all method signatures and classes.
- All javadoc comments should state the author of the class/method.
- The code should contain clear method and variable names.
- The Application should be available for Linux, OSX, and Windows.

2.3 User interface

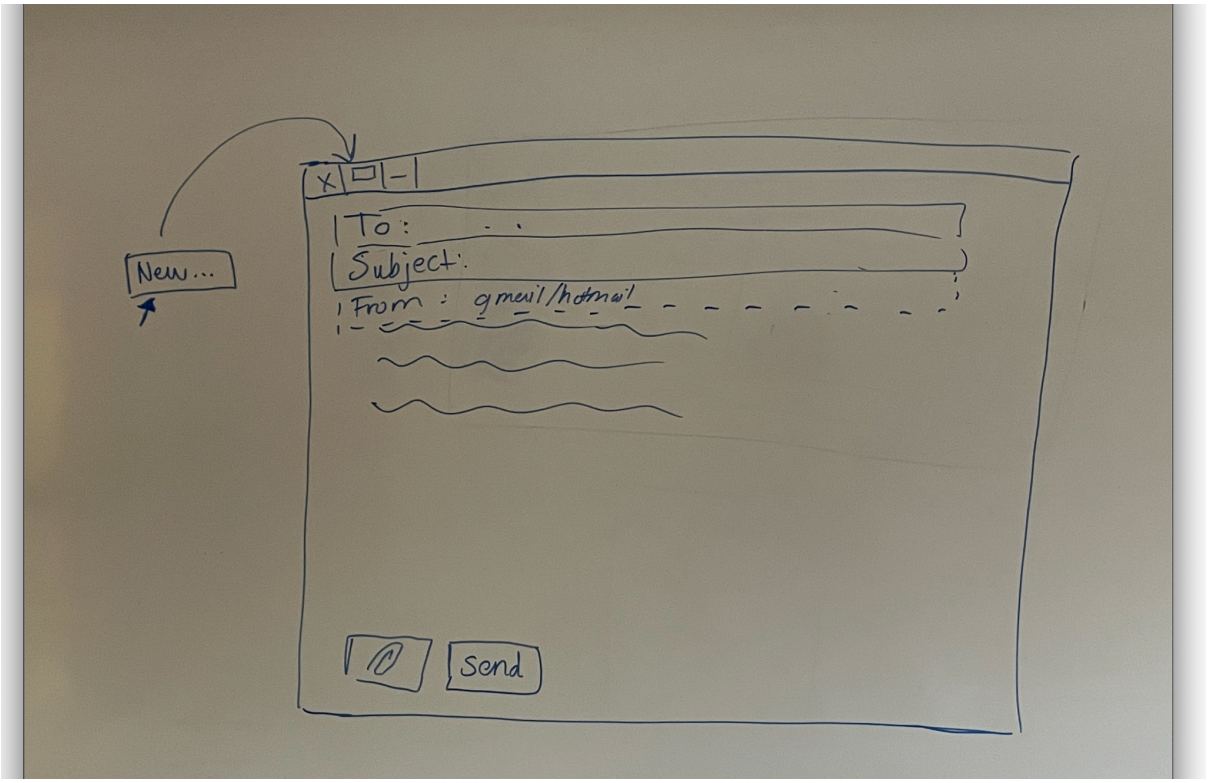
2.3.1 First sketches and drawings for our application interface

When we started planning the design for our application GUI in the first week, we started with some drawings/sketches of how we wanted the interface to look.

If we compare these sketches with the interface we ended up with, there are some small differences in component placements but overall the design is quite familiar.



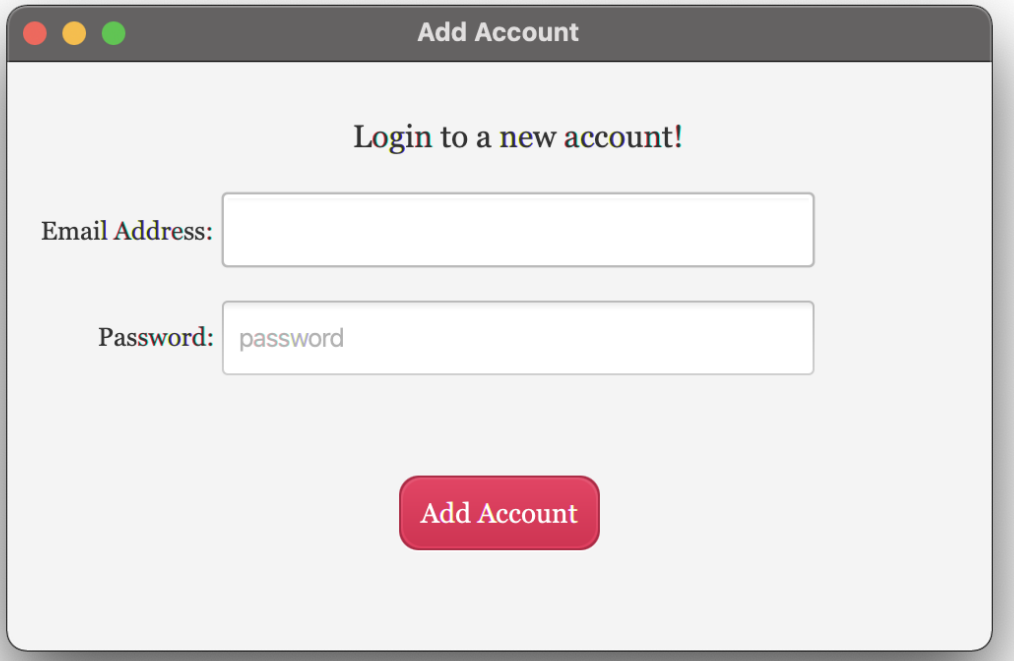
Sketch of the main view from week 1



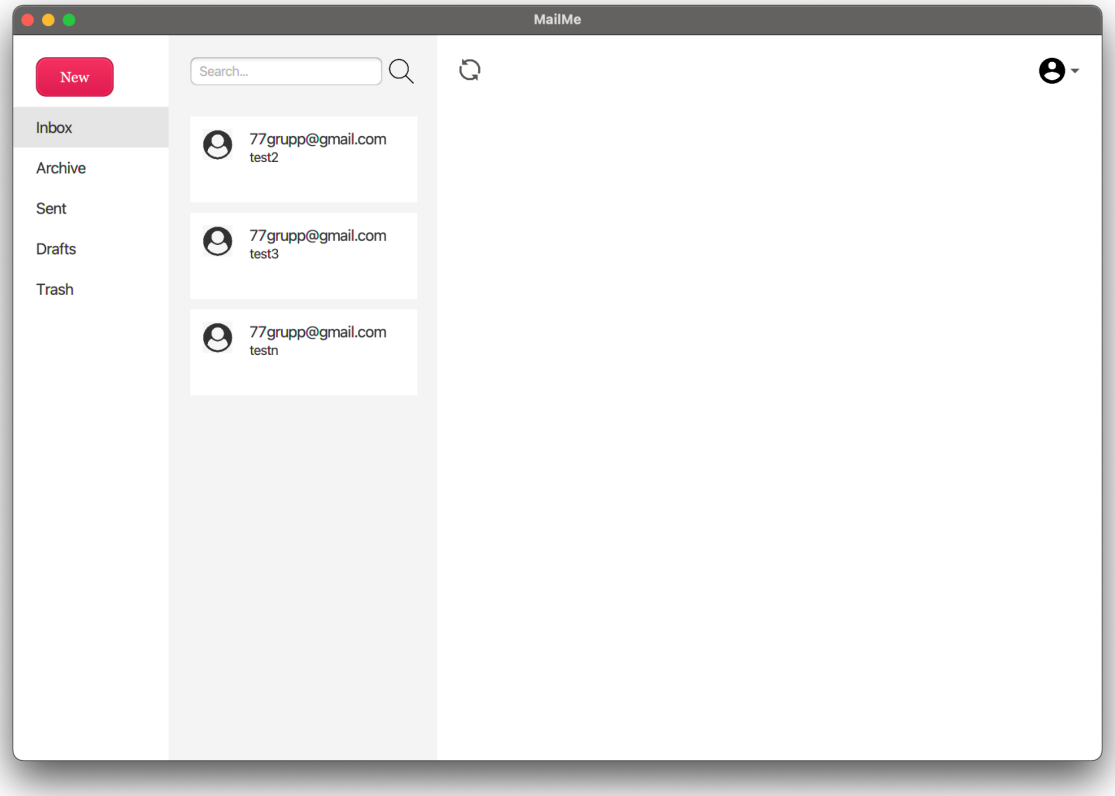
Sketch of the writing view from week 1

2.3.2 The application interface

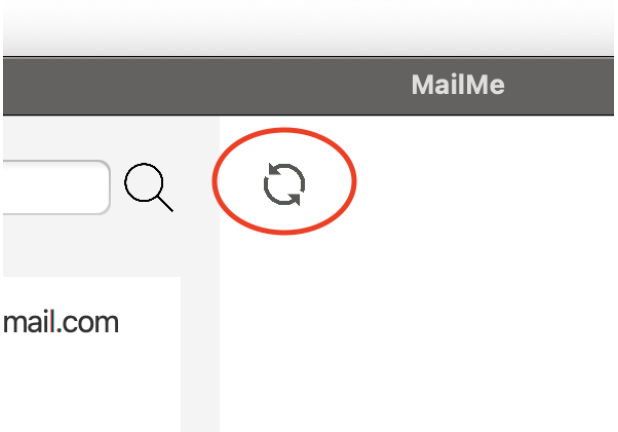
When starting the application for the first time, or when no accounts has yet to be added, This start view will be opened:



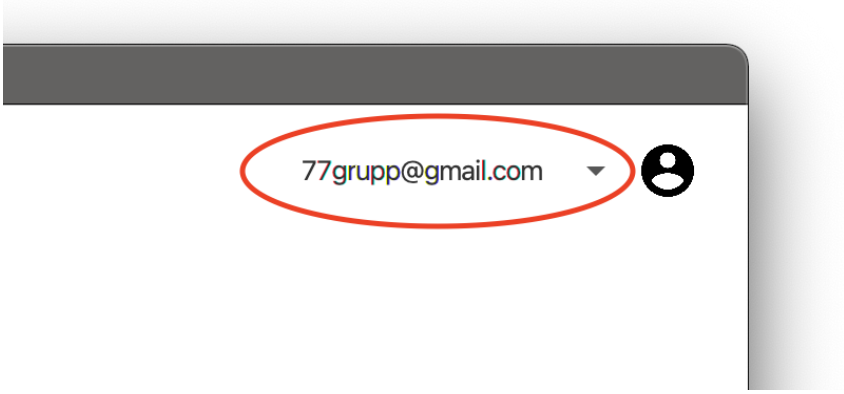
Here you can add a new account with an existing email address. When added, you will be directed to the main view when clicking on Add Account button:



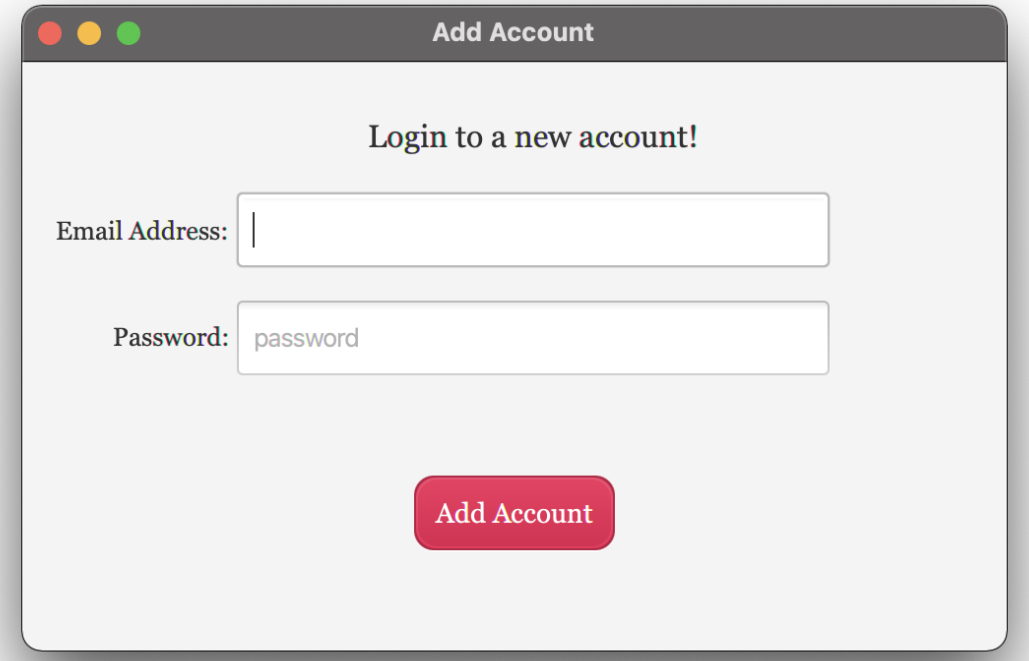
Here you can navigate through all the different mailboxes to the left and all the emails contained in them in the middle part. You can refresh to download new emails if you click on the refresh button next to the search bar:



If you want to add another account, you can click on your email address on the top right and click on “Add New Account”:



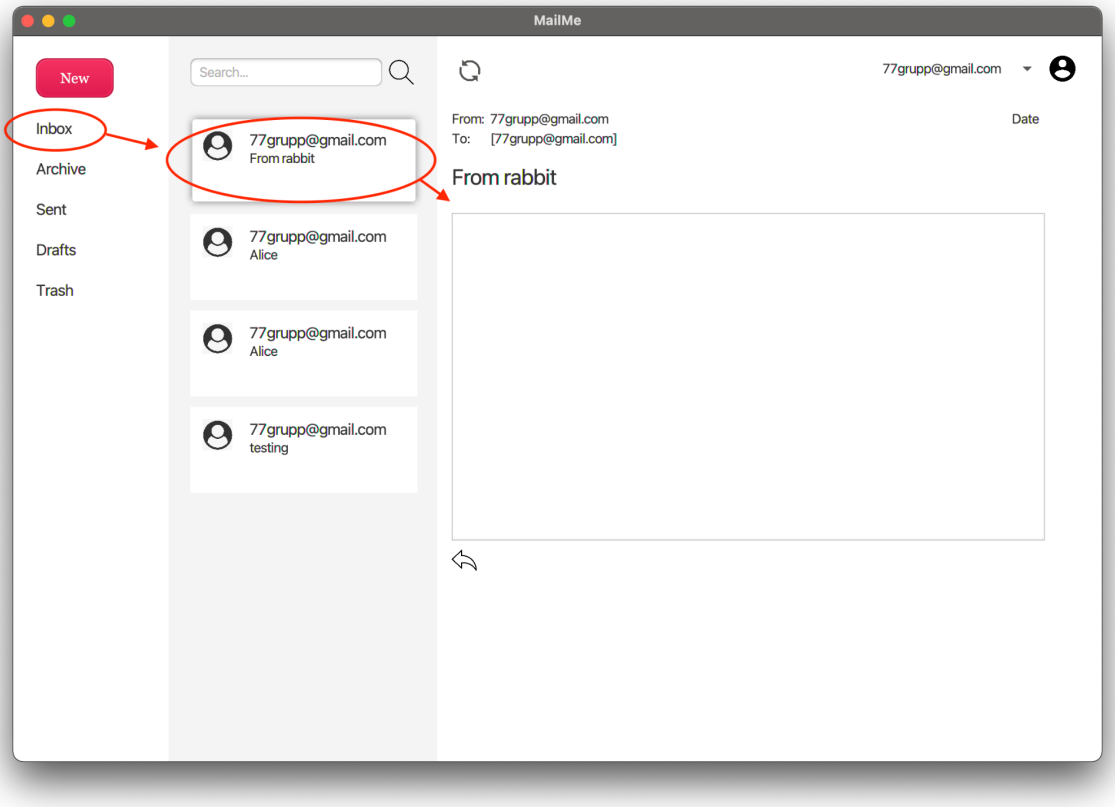
and a new window will pop up:



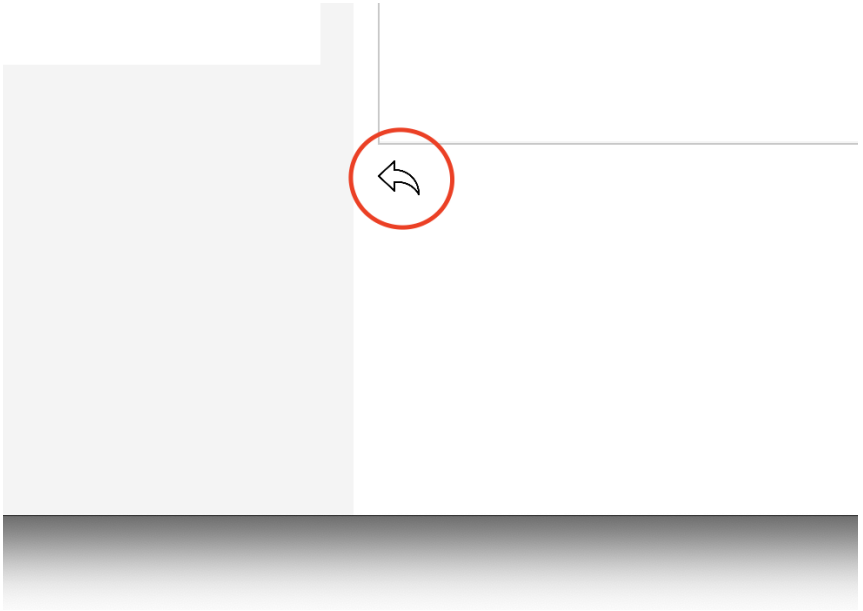
Here you can add your credentials for other email accounts.

If you want to read an email, you click on the inbox and on the email you want to read, now

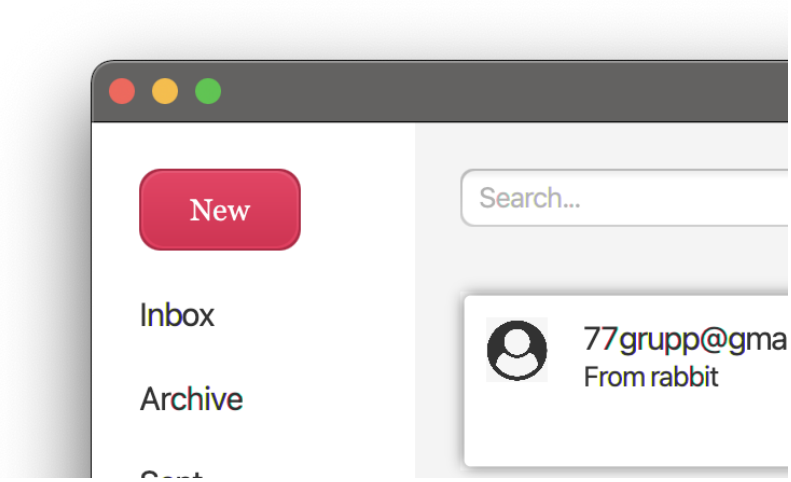
you will see a reading view:



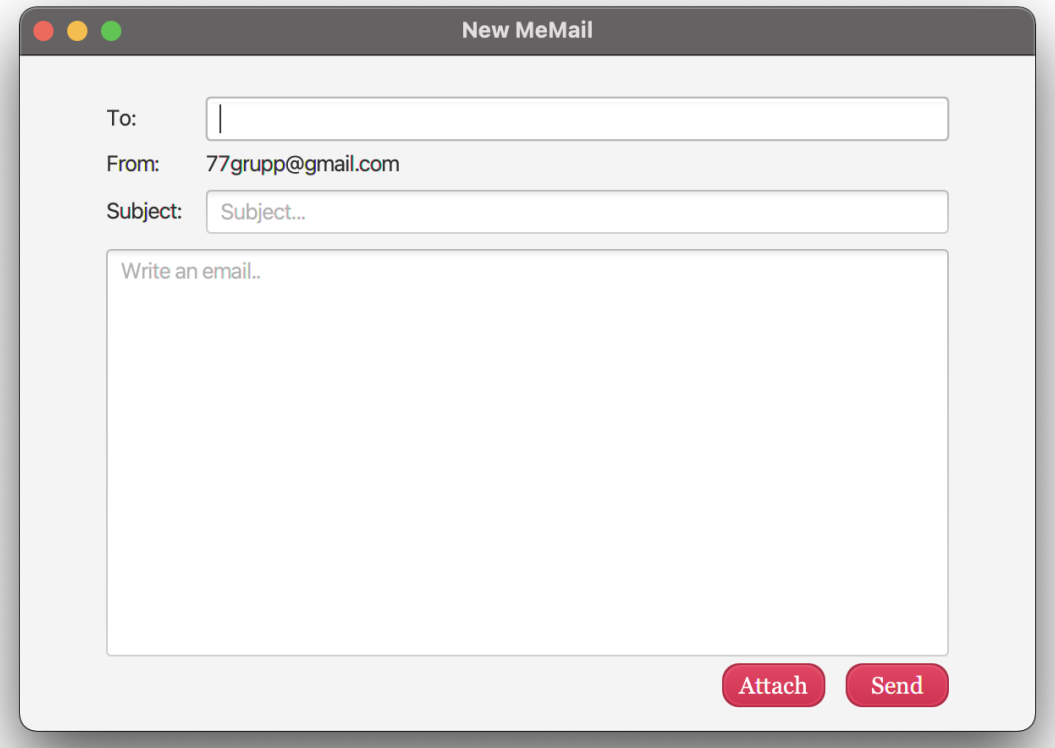
You can reply to the email by clicking on the reply button under the view:



When you click on the new button in the top left corner, a writing view will appear (the same view that appears when clicking en the reply button):



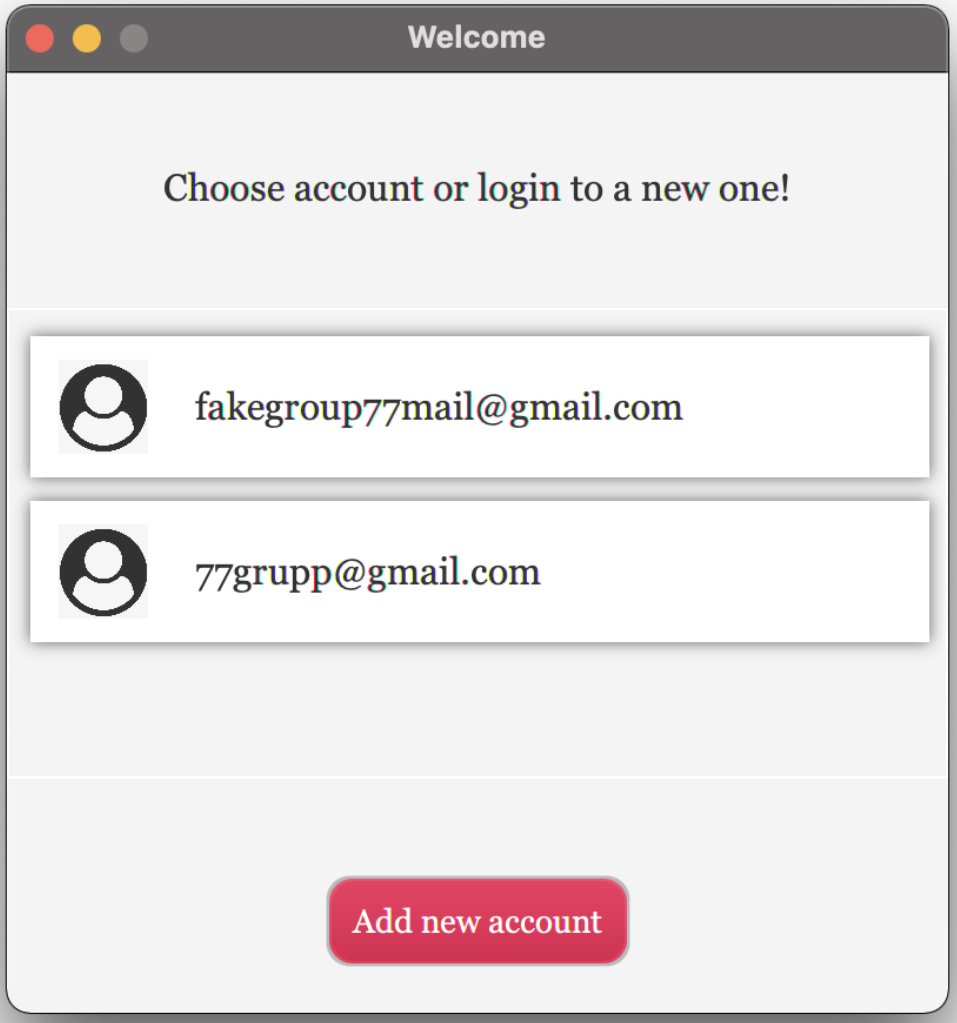
When you click on the button, this writing view appears:



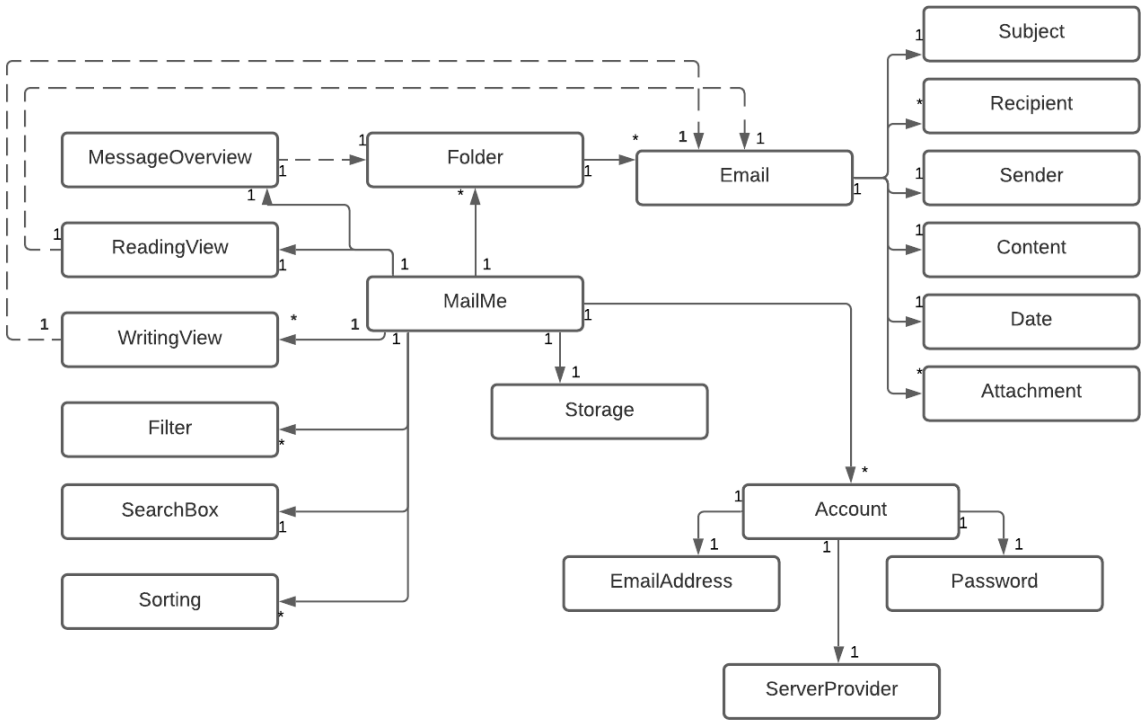
Here you can write new emails, with attachments if needed, and send them away.

If two or more accounts has been added, and when reopening the application, this

choice-screen will be opened, where you can choose which account you want login with(or add a new account):



3 Domain model



3.1 Class responsibilities

- MailMe: This represents the root of the application. This class coordinates and delegates between the different components of the application.
- Storage: This class is responsible for storing application-specific data, such as accounts and emails.
- Account: This class represents the abstract notion of an account, consisting of subparts:
    - EmailAddress: Represents an email address.
    - Password: Represents a password for a given account.
    - ServerProvider: The Email Service Provider that the account is linked to (where the account exists and gets its email service from).
- Email: Represent the abstract notion of an email, consisting of subparts:
    - Subject: Represents the subject of the email.
    - Recipient: Represents the recipient of the email (found in the ‘To’-field).
    - Sender: Represents the sender of the email (found in the From-field).
    - Content: Represents the text content of the email.
    - Date: Represents the sending or receiving date of the email.
    - Attachment: Represents any eventual files that were sent with the email.
- Folder: Represents a specific folder containing emails, such as ‘Inbox’, ‘Sent’ or ‘Drafts’.
- MessageOverview: Shows all emails within a certain folder in a list-like overview.
- ReadingView: Displays the contents of an email.

- WritingView: Allows users to compose their own email.
- Filter: Allows users to filter emails in the current folder based on some filter condition, for example based on some specific date interval.
- SearchBox: Allows users to search for specific emails within the current folder that contain a specific search word.
- Sorting: Allows users to sort the emails according to a specific total ordering.

4 References

Maven. Available at: https://maven.apache.org/index.html (Accessed. 07 10 2021)

JavaFX. Available at: https://openjfx.io/ (Accessed: 07 10 2021)

JavaMail API https://javaee.github.io/javamail/docs/api/ (Accessed: during september, october 2021)

Apache Commons. Available at: http://commons.apache.org/ (Accessed: during september, october 2021)

Post Office Protocol. Available at: https://en.wikipedia.org/wiki/Post_Office_Protocol (Accessed: 07 10 2021)

Simple Mail Transfer Protocol. Available at: https://en.wikipedia.org/wiki/Port_(computer_networking) (Accessed: 07 10 2021)

Host. Available at: https://en.wikipedia.org/wiki/Host_(network) (Accessed: 07 10 2021)

Port. Available at: https://en.wikipedia.org/wiki/Port_(computer_networking) (Accessed: 07 10 2021)