CSE183 Fall 2020 - Assignment 9

In this assignment you write an E-Mail system as a Single Page Full Stack Web App using the NERP Stack: Node.js, Express, React, and PostgreSQL, plus Material-UI.

This assignment is worth 20% of your final grade.

Submissions are due NO LATER than 23:59, Thursday December 10, 2020 (~ 2.5 weeks)

Installation

See instructions in Assignment 3 and ensure you are running the current LTS version of Node.js. You will also need to have downloaded and installed Docker Desktop: https://www.docker.com/products/docker-desktop.

Setup

Download the starter code archive from Canvas and expand into an empty folder. I recommend, if you have not already done so, creating a folder for the class and individual folders beneath that for each assignment.

The starter code archive contains some files that you will modify:

```
backend/api/openapi.yaml
backend/src/app.js
backend/test/backend.test.js
backend/sql/data.sql
backend/sql/schema.sql
backend/sql/indexes.sql

frontend/public/src/App.js
frontend/public/test/dummy.test.js
```

And some that you should ***not*** modify:

```
backend/.env
backend/.eslintrc.js
backend/docker-compose.yml
backend/package.json
backend/src/server.js
backend/src/dummy.js
backend/sql/database.sql

frontend/.eslintrc.js
frontend/package.json
frontend/public/index.html
frontend/public/favicon.ico
frontend/public/src/index.js
frontend/public/src/dummy.js
frontend/public/test/dummy.test.js
frontend/public/test/globalSetup.js
frontend/public/test/globalTeardown.js
```

To setup the development environment, ***navigate to the folder where you extracted the starter code*** and run the following command in this folder and the `frontend` and `backend` sub-folders:

```
$ npm install
```


This will take some time to execute as `npm` downloads and installs all the packages required to build the assignment.

To start the frontend and backend dev servers, run the following command:

```
$ npm start
```

The first time this runs it will take a while to download and install the backend Docker PostgreSQL image and start the database service for your server to run against.

The frontend and backend servers can be run individually from their respective folders by running `npm start` in separate console / terminal windows.

To execute tests run the following command in either the `frontend` or `backend` folder:

```
$ npm test
```

To run the linter against your code, run the following command in either the `frontend` or `backend` folder:

```
$ npm run lint
```

Web App Specification

This system must be a Single Page Web Application using the following technologies.

Browser:	React & Material-UI
Server:	Node.js & Express
Storage Tier:	PostgreSQL

The server must present an OpenAPI constrained API to the SPA running in the browser and the server just store its information in a PostgreSQL database.

A “mobile first” approach should be taken; the principal operations being:

- Login Screen
- Mailbox Viewer
- Mailbox Selection
- Mail Viewer
- Mail Composer
- Settings
- Search

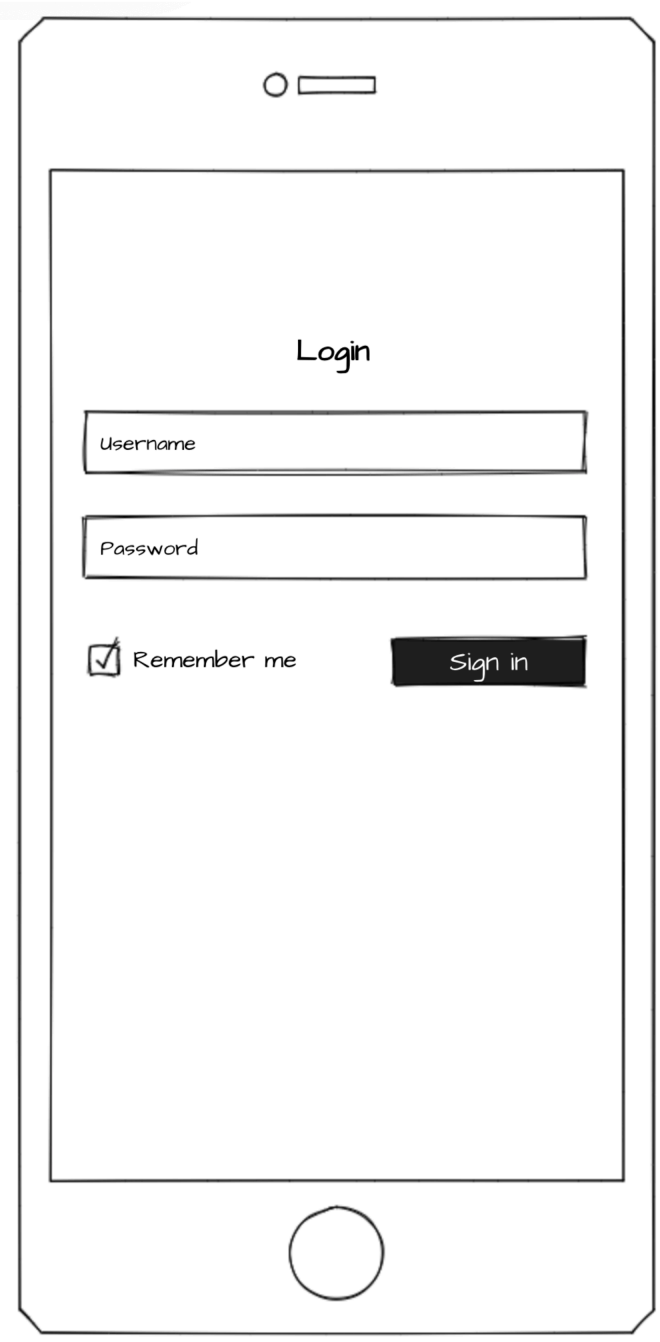
For the desktop version, the principal operations are as for mobile, but composition and settings should be modal dialogs rather than full screen.

Each is examined in detail on the following pages.

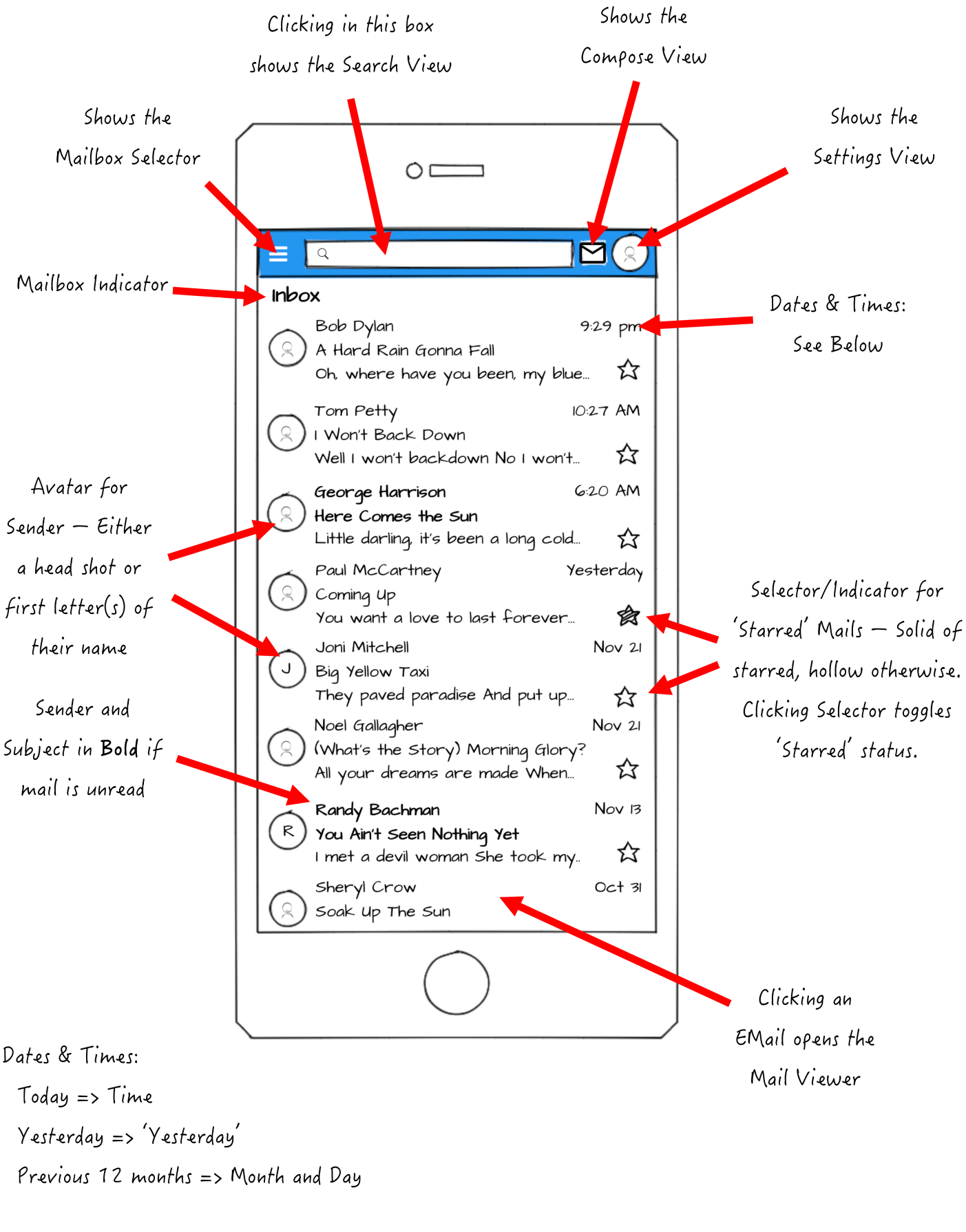
Login Screen

We have yet to cover techniques and technologies required to support authentication and user sessions in Web Applications. This page will be updated once we have.

For now, be aware the concept of a user is important when designing your Web Application. The logged in user should only be able to see their own e-mails.



Mobile: Mailbox Viewer



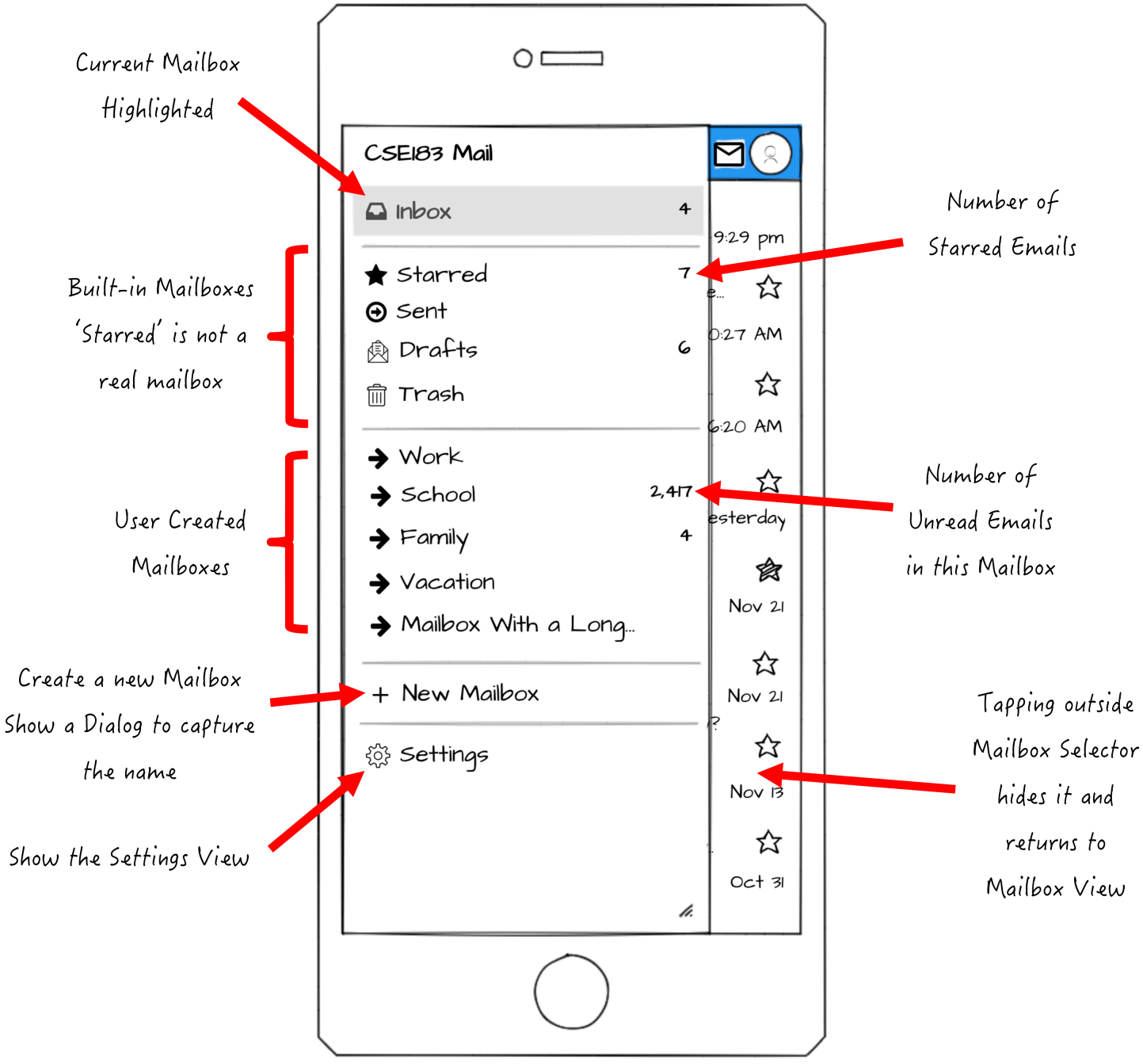
Dates & Times:

Today => Time

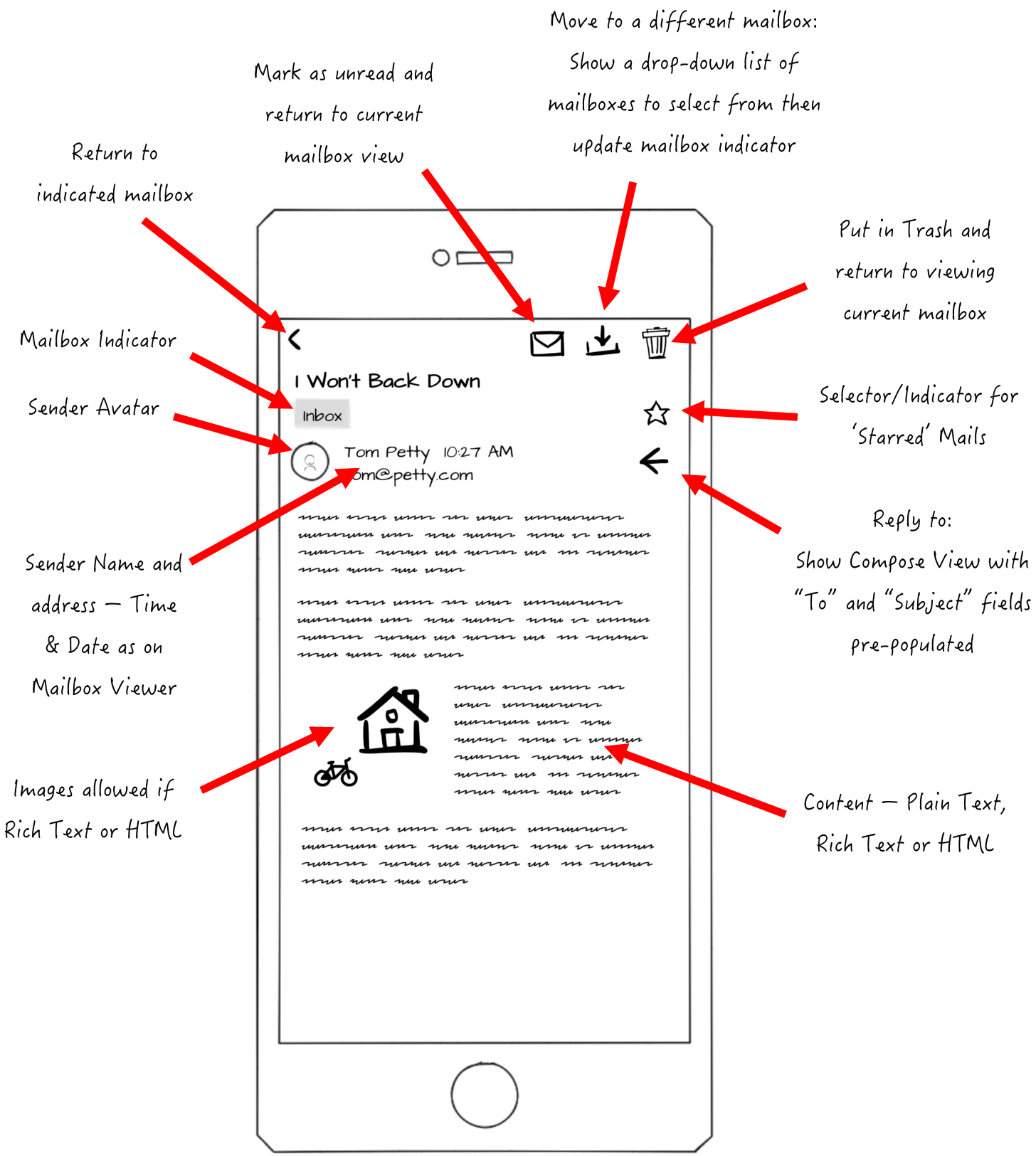
Yesterday => 'Yesterday'

Previous 12 months => Month and Day

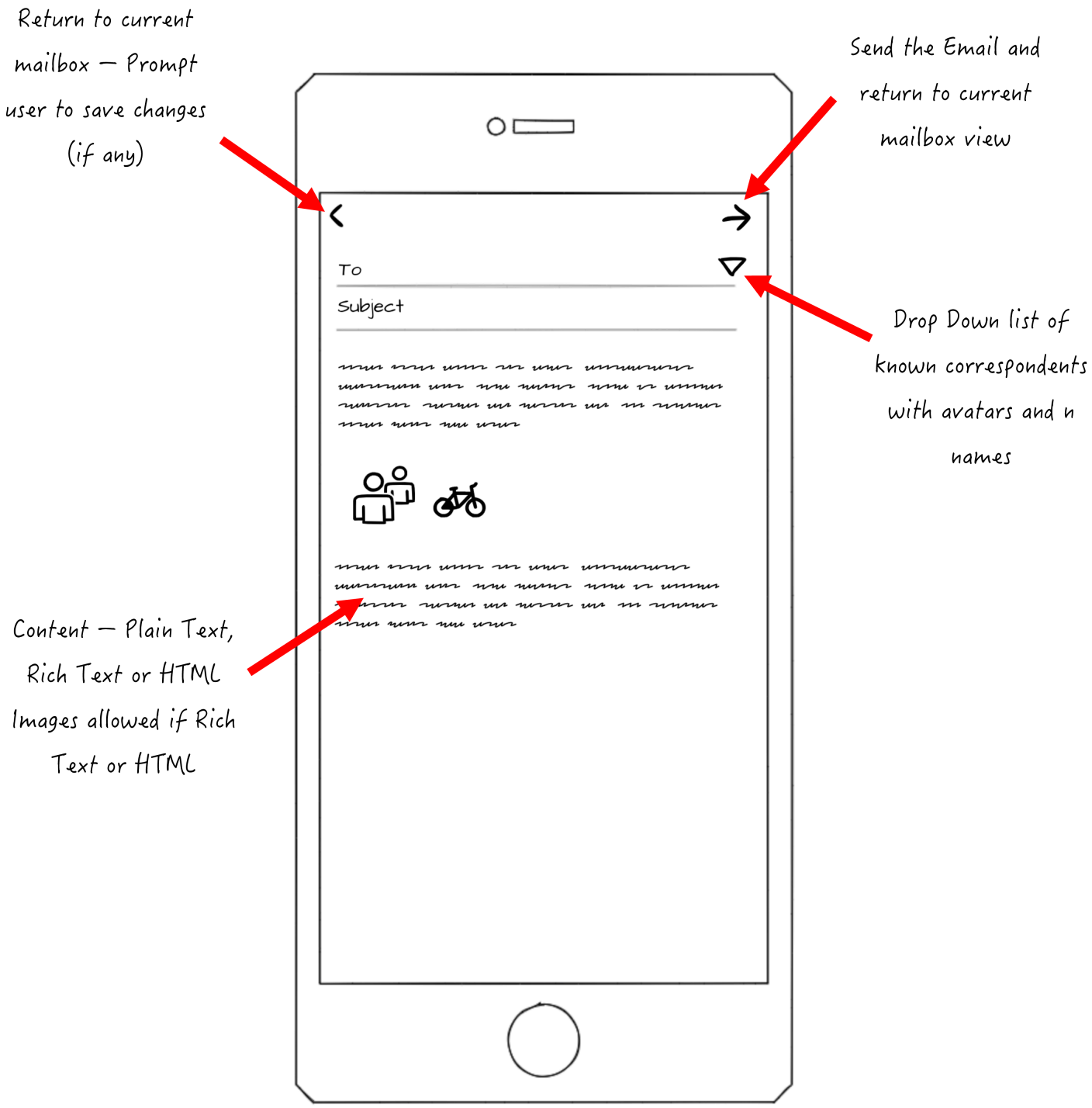
Mobile: Mailbox Selector



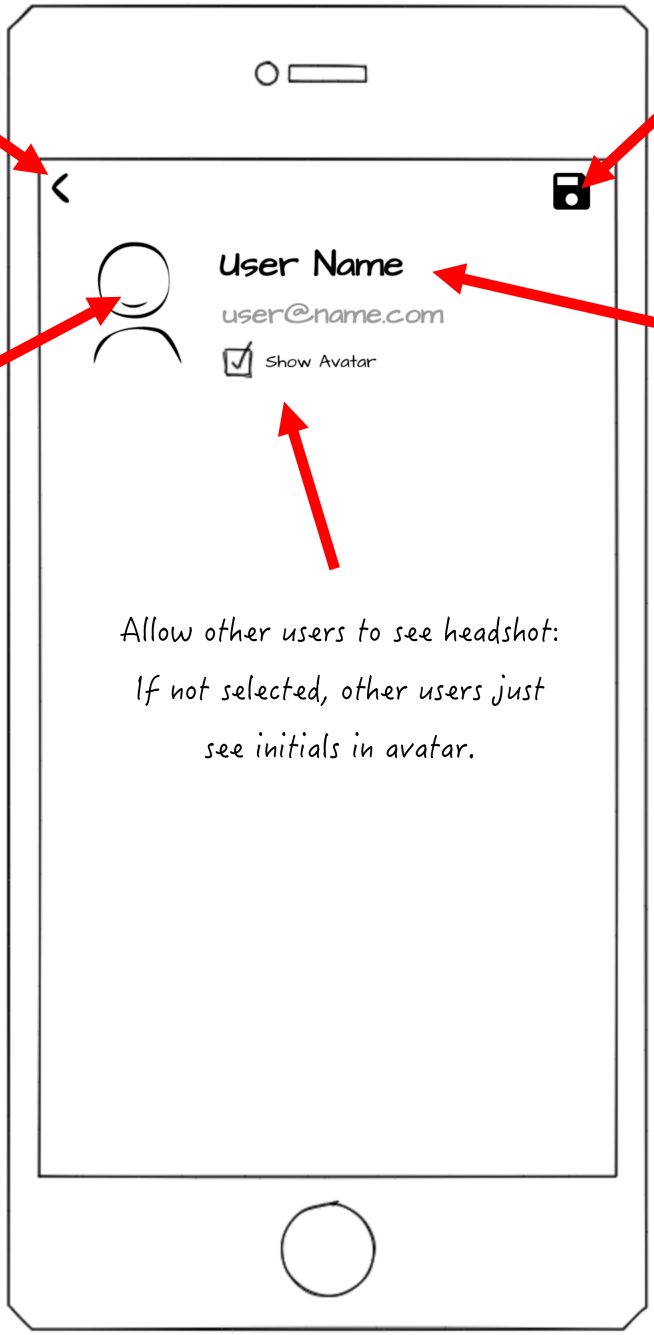
Mobile: Mail Viewer



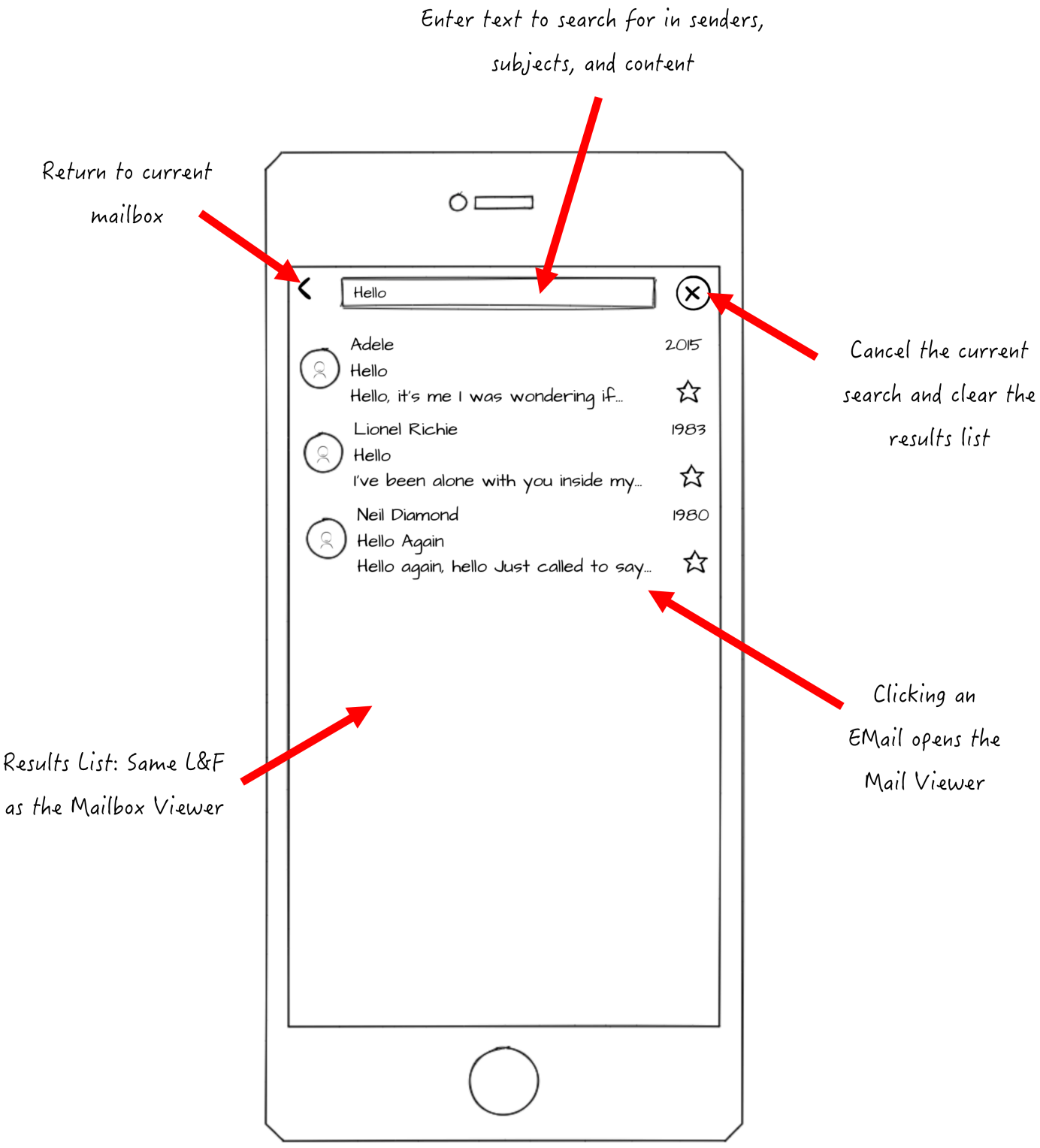
Mobile: Composer



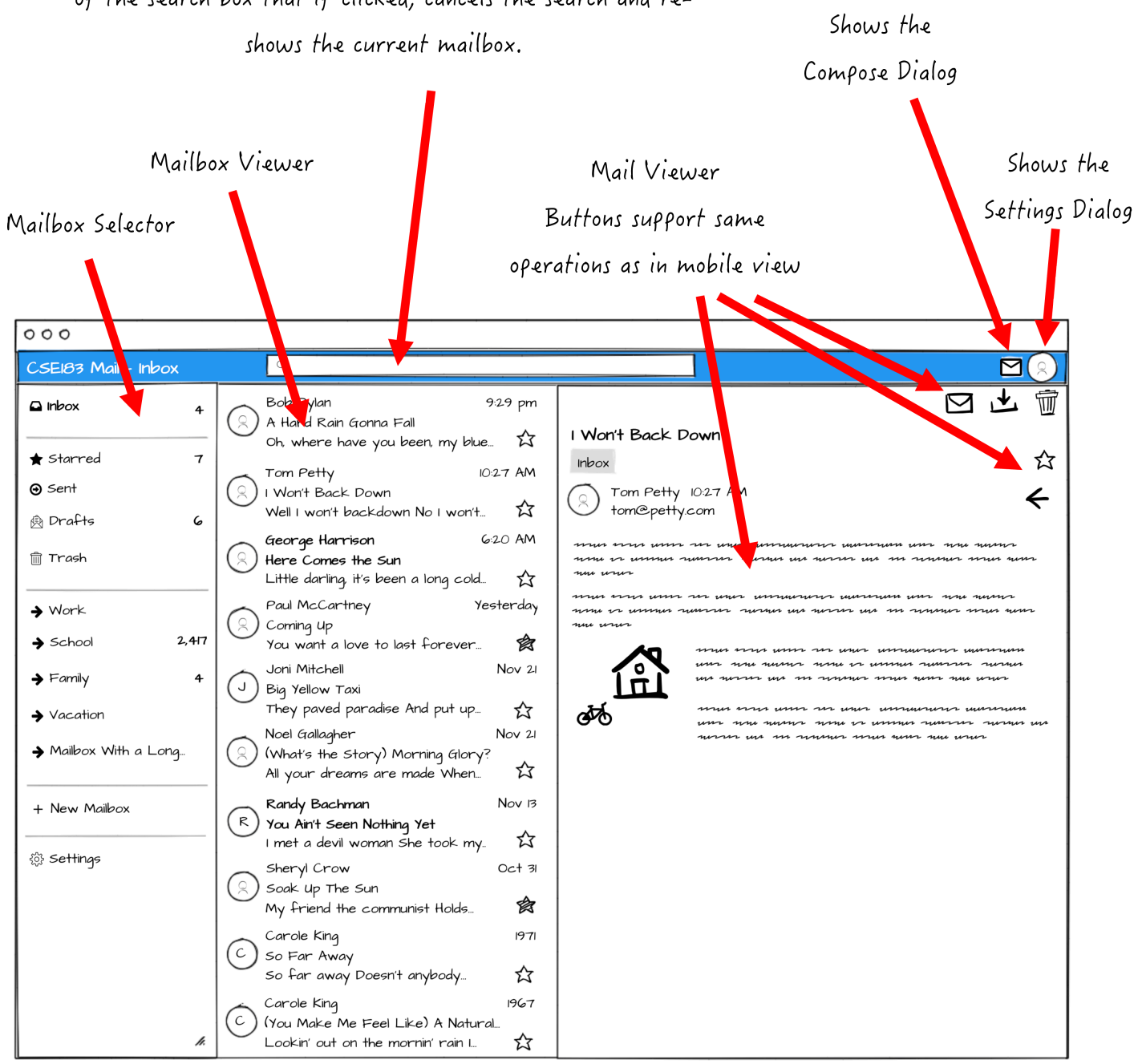
Mobile: Settings



Mobile: Search



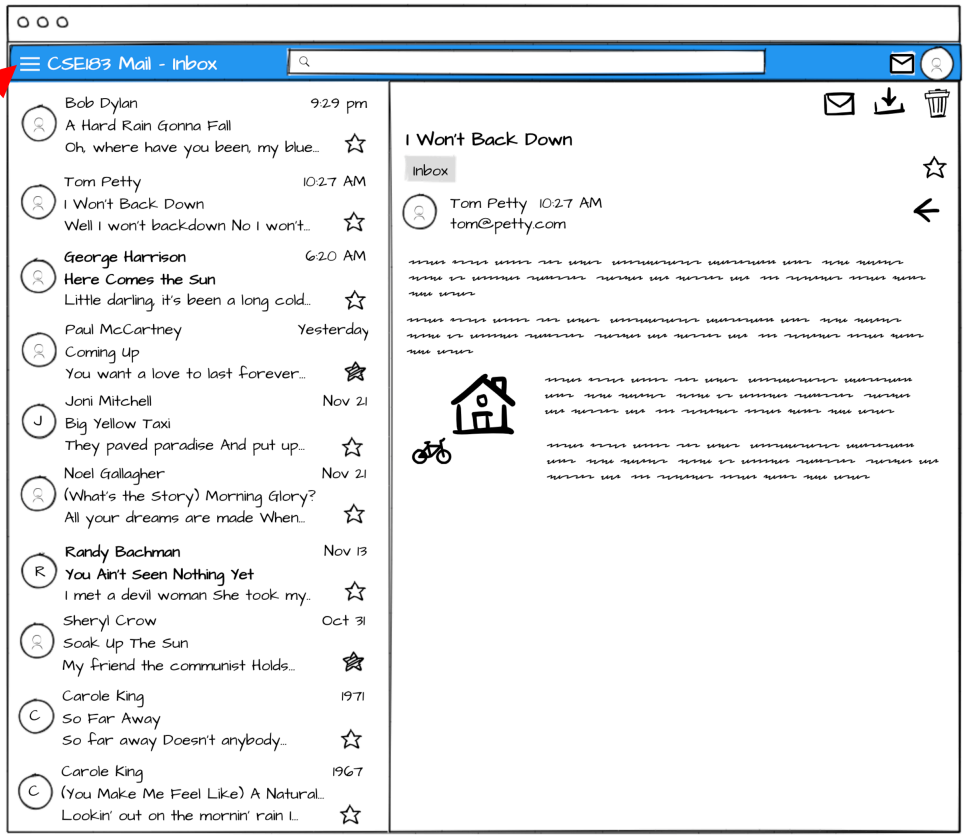
Desktop: Wide View

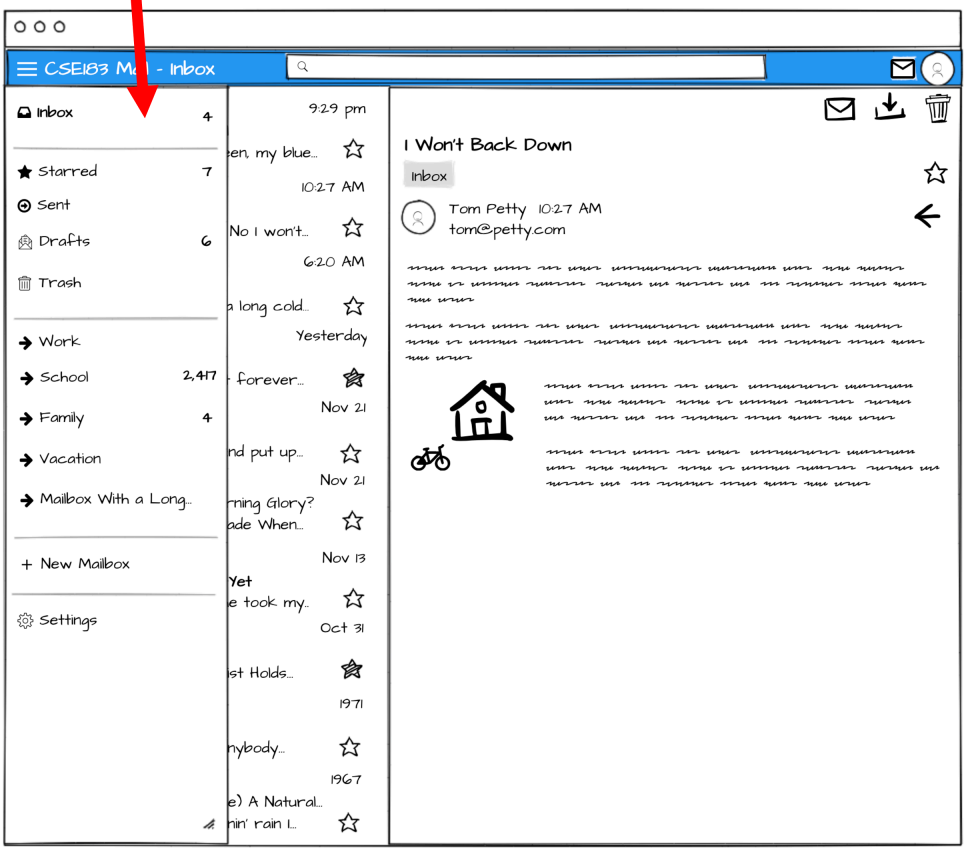


Desktop: Narrow View & Mailbox Selection Popup

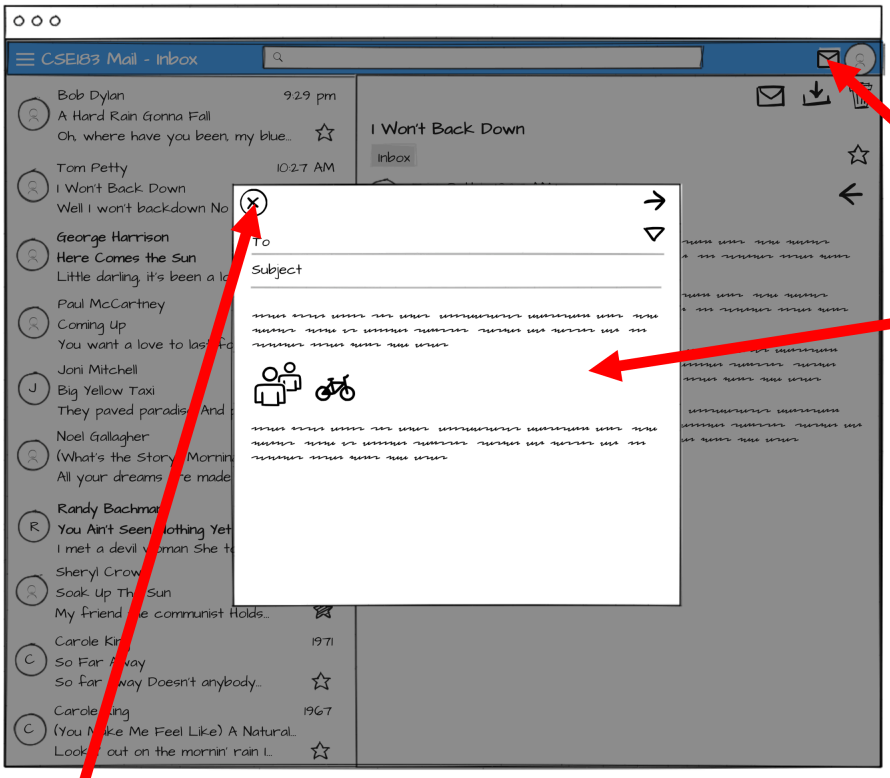
Mailbox Selector is hidden, and Menu Button is shown when browser is too narrow

Clicking the Menu Button shows the Mailbox Selector above the Mailbox Viewer.



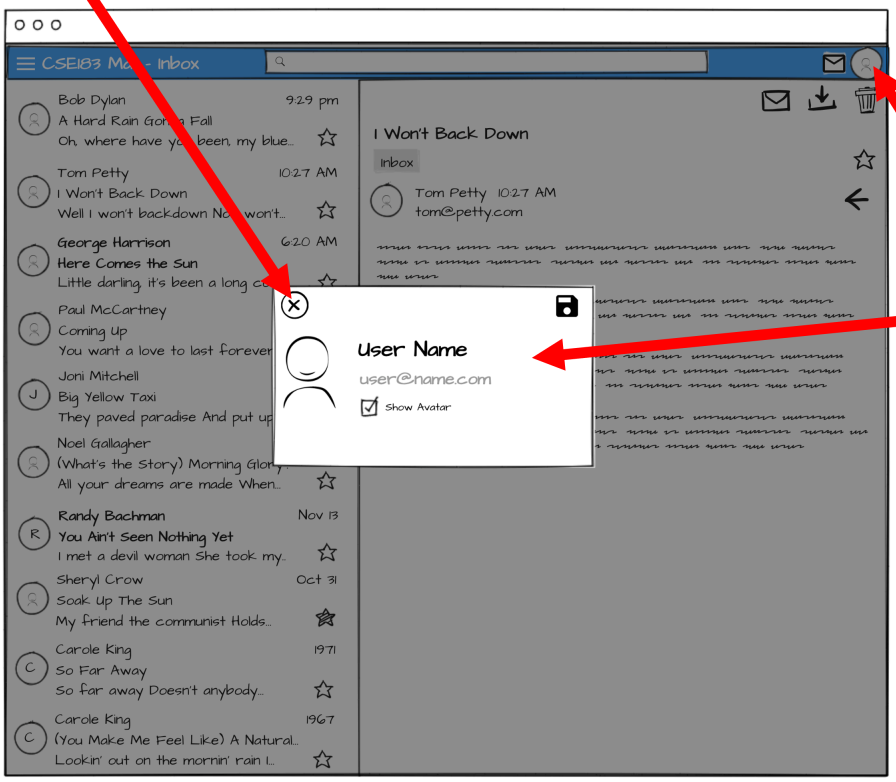


Desktop: Compose & Settings Dialogs



Clicking the Compose Button shows the Compose Dialog — buttons have same operations as in mobile view

Dialog Close Button



Clicking the Settings Button shows the Settings Dialog — buttons have the same operations as in mobile view

Development Guidelines

Whilst you can construct the user interface any way you like, this finished front end must be a Single Page Application; React Routes may prove useful for the mail reading and composition pages.

Background

E-Mails in this system can have any properties you deem necessary to complete the assignment.

Note that your API must be constrained by an OpenAPI version 3.0.3 schema. You should endeavor to make this schema as restrictive as possible.

For example, your schema should specify the acceptable format of the e-mail id property. If an id in any other form is presented, your system should reject it simply by performing the validation provided in the starter code.

When you start the backend, use a browser to visit http://localhost:3010/v0/api-docs/ where you can manually test the API as demonstrated in class.

Initially, the only operation available will be:

Which if executed will return the current data and time and the date and time the backend database was initialised.

Database Schema

The supplied database schema can be found in `backend/sql/schema.sql`. It defines a single table '`dummy`'. You will want to define the tables you need in this file.

Optionally, you can also define indexes to be built against your tables in: `backend/sql/indexes.sql`

Resetting Docker

If you run into problems with your dev database, or simply want to re-set it to its initial state, issue the following commands:

```
$ docker stop $(docker ps -aq)
$ docker rm $(docker ps -aq)
```

Requirements

Features are awarded points based on their complexity. A breakdown of the feature points can be found at the end of this document. There are 300 marks available, but the maximum score is capped at 200 which equates to 20 points (100%) on the assignment.

You are also required to make a short YouTube video of your Web App demonstrating the features you have successfully implemented - more details to follow.

What steps should I take to tackle this?

You should base your implementation on your Assignment 6 and Assignment 8 solutions. The "Simplest Solution" to Assignment 6 presented in class is a good starting point if your submission was incomplete, as is The Books Database Example if you Assignment 8 submission was below par.

There is a huge amount of information available on OpenAPI 3.0.3 at https://swagger.io/specification/ and http://spec.openapis.org/oas/v3.0.3. Also consult the Petstore example at https://petstore3.swagger.io/ and make good use of the on-line schema validator at https://editor.swagger.io/.

A good resource for querying JSON stored in PostgreSQL: https://www.postgresqltutorial.com/postgresql-json/

Many Material-UI Examples and sandboxes for experimentation are available: https://material-ui.com/

How much code will I need to write?

This is a difficult question to answer, but not including your OpenAPI Schema, something in the order of 600 to 800 lines of JavaScript might be a reasonable estimate.

Grading scheme

The following aspects will be assessed:

1. (100%) **Does it work?**

    a. Maximum of 20 points available for implemented features.

2. (-100%) **Did you give credit where credit is due?**

    a. Your submission is found to contain code segments copied from on-line resources and you failed to give clear and unambiguous credit to the original author(s) in your source code (-100%). You will also be subject to the university academic misconduct procedure as stated in the class academic integrity policy.

    b. Your submission is determined to be a copy of a past or present student's submission (-100%)

    c. Your submission is found to contain code segments copied from on-line resources that you did give a clear an unambiguous credit to in your source code, but the copied code constitutes too significant a percentage of your submission:

        - < 80% copied code — No deduction
        - 33% to 66% copied code — (-50%)
        - > 66% copied code — (-100%)

What to submit

Run the following command to create the submission archive:

```
$ npm run zip
```

You must also paste a link to your YouTube video into the canvas submission – more details to follow.

****** UPLOAD** `Assignment9.Submission.zip` **TO THE CANVAS ASSIGNMENT AND SUBMIT ******

Assignment 9 - Mark Distribution	**Grand Total:**	**300**
Single User	**Total:**	**250**
Mobile: Mailbox View		100
Clicking menu button shows the mailbox selector	2	
Clicking in search box shows the search view	2	
Clicking compose button shows the Compose View	2	
Clicking settings button shows the Settings View	2	
Settings button shows avatar of user	3	
Mailbox indicator is correctly set	3	
Clicking anywhere on an email opens the mail viewer	3	
Dates & Times are correctly formatted	10	
Starred selector / indicator is set correctly	5	
Clicking Starred selector / indicator changes starred status	10	
Unread mails have sender and subject in bold	3	
Avatars of senders are set appropriately	5	
Layout of email is correct - avatar, sender, date, subject and first row of content	50	
Mobile: Mailbox Selector		28
Shows correctly - above mailbox view menu bar	2	
Tapping outside Mailbox Selector hides it and returns to Mailbox View	2	
Current mailbox is highlighted	3	
Number of unread emails in each mailbox is shown and correct	5	
Number of starred emails is shown and correct	3	
User Created mailboxes are shown	5	
Clicking "New mailbox" shows a dialog to capture the name	3	
After creating new mailbox, it shows in Mailbox Selector	3	
Clicking "Settings" shows settings view	2	
Mobile: Mail Viewer		32
Mailbox indicator is set correctly	2	
Clicking chevron in top left returns to indicated mailbox	2	
Clicking "mark as unread" does so and returns to indicated mailbox	2	
Clicking "Move to" shows list of mailboxes to move mail to	2	
Selecting mailbox to move mail to does so and updates mailbox indicator	2	
Clicking trash button moves mail to trash and returns to currently indicated mailbox	2	
Starred selector/indicator is set correctly	2	

Starred selector/indicator works correctly	2	
HTML or Rich Text Content	10	
Sender name, email address and avatar correct	2	
Received date is set correctly	2	
Clicking Reply To button shows compose view with "To" and "Subject" Fields set correctly	2	
Mobile: Composer		19
Clicking chevron in top left returns to current mailbox without sending the email	2	
Clicking send button returns to current mailbox after sending the email	2	
Drop-down list of known corespondents works - selection populates the "To" field	10	
HTML or Rich Text Content	5	
Mobile: Settings		12
Clicking chevron in top left returns to current mailbox without sending the email	2	
After clicking chevron in top left prompt user to save changes, if any	2	
Clicking save button saves changes and returns to current mailbox	2	
Clicking headshowt shwos dialog to change image URL	2	
Selecting new image shows it correctly in headshot	2	
User name can be changed	2	
Mobile: Search		14
Clicking chevron in top left returns to current mailbox	2	
Clicking cancel button cancels current search ancdclears the results list	2	
Clickng an email opens the email viewer	2	
Results list has same Look and Feel as Mailbox Viewer	2	
Entering text to search for works for subjects	2	
Entering text to search for works for senders	2	
Entering text to search for works for contents	2	
Desktop: Wide View		24
Mailbox Selector visible and working as for Mobile	2	
Mailbox View visible as for Mobile	2	
Mail Viewer is Embedded in desktop view	2	
Mail Viewer buttons support same operations as in mobile view	2	
Clicking compose button shows compose dialog	2	
Clicking settings button shows settings dialog	2	
Entering text into search box filters mail in Mailbox Viewer	10	
Clicking the search Cancel button returns to selected mailbox	2	

Desktop: Narrow View			7
	Mailbox Selector is hidden or narrow screens and meu button appears	5	
	Mailbox Selector appears above mailbox viewer when menu button is clicked	2	
Desktop: Compose			7
	Compose dialog buttons have same operation as mobile	5	
	Compose Dialog has a close button in the top left	2	
Desktop: Settings			7
	Settings dialog buttons have same operation as mobile	5	
	Settings Dialog has a close button in the top left	2	
Multi User		**Total:**	**50**
Login Screen			
	Only authorised Users are allowed in	10	
Mobile & Desktop Views			
	Only show mailboxes and mail for the logged in user	10	
Mobile: Settings			
	Show Avatar option setting prevents / allows other users from seeing headshot	10	
Stretch			
	Users can send mail to each other	10	
	Sending an email from one user to another shows up in the receiver's mailbox without manual refresh	10	