Module 7 - Description

Tested on both the systems below-
Macbook Pro (13 inch, 2017, Two Thunderbolt 3 ports)
Operating system: mac Catalina (10.15.7)
Processor: 2.3 GHz Dual-Core Intel Core i5
Memory 8GB 2133 MHz LPDDR3
Graphics Intel Iris Plus Graphics 540 1536 MB
Safari Version 14.0
Terminal 2.1.0

Operating system: Windows 10 home (20H2) – created on this system
OS build:19042.662
Experience: Windows feature experience pack 120.2212.551.0
HP ENVY X360 convertible 15-DS1XXX
Processor AMD Rizen 5 4500U with Radeon graphics 2.38GHz
Memory 16GB
Browser version: Microsoft edge (87.0.664.57)
Tested using windows CMD terminal
TypeScript – 4.1.2
Node.js version (v14.15.1)

Recursive list of files in the client directory present at time of testing is given in client/file-directory-list.txt

Set up project:
Packet.json file has all these dependencies listed below.
```
"devDependencies": {
   "awesome-typescript-loader": "^5.2.1",
   "babel-plugin-syntax-dynamic-import": "^6.18.0",
   "css-loader": "^3.3.0",
   "html-loader": "^0.5.5",
   "html-webpack-plugin": "^3.2.0",
   "style-loader": "^1.0.1",
   "ts-loader": "^6.2.1",
   "typescript": "^3.7.3",
   "webpack": "^4.41.2",
   "webpack-cli": "^3.3.10",
   "webpack-dev-server": "^3.9.0"
},
  "dependencies": {
   "@material-ui/core": "^4.6.0",
   "@material-ui/icons": "^4.5.1",
   "@types/material-ui": "^0.21.7",
   "@types/react": "^16.9.16",
   "axios": "^0.19.0",
   "normalize.css": "^8.0.1",
```

```
"react": "^16.11.0",
"react-dom": "^16.11.0"
}
```
1. To install these dependencies, navigate to the client folder in terminal and run: **npm install**
2. Update **userEmail = "**<youremail@domain.com>**"**and **serverAddress= "http://localhost" ,** where you should replace <youremail@domain.com> with your own email.
3. Go to client directory, **npm run build** in terminal
4. Open another terminal and start the mailbag server with command **npm run dev** in the server directory. Make sure the server has started listening on port 80
5. Open to client/dist/index.html in browser
6. The mailbag application should load all the inboxes on the left side and all clients on the right side.

Expected behavior
1. You can click on a mailbox on the left to display all the emails in it, for e.g., clicking on mailbox "INBOX", should display all the emails from the mailbox INBOX.
2. You can click on any email, and it should get the email from the server and display it.
3. The client allows us to **"Reply"** to an email or **"Delete"** an email.
4. You can click on "**New Message**" button on top left and it should open a compose message screen with a To:, Subject: and Body: portions. Once you fill in the details and message body, you can click on "**Send**" button in the bottom and it give a brief "Waiting for server" message following which it should return to the INBOX screen and the message would be sent to server.
5. You can click on "**New Contact**" button on top and it takes you to a new screen allowing you to add a name and email. By clicking the save button, the contact is intimated to the server, and added to the contact list on the right.
6. You can view a contact by clicking on their name in the contacts list panel on the right. This shows us the name and email of the client.
7. We can "**Delete"** the contact, "**Update"** the contact, or "**Send Email"** to the contact.
8. Choosing "**Delete"** should send a HTTP Delete request to the server.
9. Choosing "**Update"** should send a HTTP PUT request to the server.
10. Choosing "**Send Email"** should take you to compose page and allow you to send an email using the HTTP POST method.

Benefits of AJAX
- We use Axios module to use the Asynchronous Javascript and XML functionality. AJAX is very helpful at the client because it improves responsiveness significantly. When one element of the UI is being changed, the other elements need not be refreshed. This makes server requests lightweight and user experience smoother.
- Client does not need to reload whole webpage for each GET/PUT/POST/DELETE request. It only reloads a component of the page

New function added
- Update a contact: Client has a new button to update a contact. This calls a updateContact() method in state.ts which in turn instantiates a contactWorker to update the contact. The contactWorker makes a PUT request to the server with contact details in the body and receives the updated contact. The client updates the state of contacts list and displays it.

Updating contact1

Result of updating contact1 to contact2

Mailboxes loaded