To Do (Sprint Backlog)	In Progress	Testing	Done
			Task 1: Create React App and initialize any necessary infrastructure (Eric) [1]
			Task 2: Set up routing (Eric) [2]
			Task 3: Create login skeleton (Eric) [3]
			Task 4: Finish figma for login, sign up, and landing page (Eric, John) [4]
			Task 14: Connect flask app to Postgres DB (Aditya)
			Task 5: Create signup skeleton (John) [5]
			Task 7: Set user information in redux store after user is authenticated (John) [6]
			Task 6: Return JWT when user is authenticated (Aditya) [7]
			Task 8: Create redux action/dispatch to be called when user logs in (John) [8]
			Task 9: Create redux action/dispatch to be called when user logs out (John) [9]
			Task 10: Create redux action/dispatch to be called when user refreshes (John) [10]
			Task 21: Add route for password reset with token embedded in url (John) [11]
			Task 17: Display fetched listings with a card (Eric) [12]
			Task 19: Create Reset Password endpoint (Aditya) [13]
			Task 14: Display Forgot Password component when clicked (John) [14]
			Task 16: Set up input field for resetting password (Eric) [15]
			Task 25: Create form skeleton for editing profile info (Eric) [16]
			Task 26: Send and receive request for editing profile (Eric) [17]
			Task 18: Integrate vertical menu bar to include profile link (John) [18]
			Task 18: Integrate vertical menu bar to include profile link (John) [19]
			Task 24: Query user and change profile info in database (Aditya)
			Task 22: Send email when a user registers (Francisco) [20]
			Task 23: Reset password from email link (Francisco) [21]

[1] Acceptance Criteria
Tester (John)

1. The application should be able to compile and load without any problems after setting up infrastructure.

[2] Acceptance Criteria
Tester (John)

1. The user should be able to route to a different page, depending on what the user enters in the browser

[3] Acceptance Criteria
Tester (John)

1. The user should be able see basic HTML input to log in.
2. The user should be able to enter basic log in information.
3. The user should be able to see an error if he/she is unable to login.
4. The user should be able to see the home page if he/she has been successfully logged in.

[4] Acceptance Criteria
Tester (Aditya, Francisco)

1. The layout of the application should reflect how the app will look in the future.

[5] Acceptance Criteria
Tester (Eric)

1. The user should be able to see basic HTML input regarding signing up.
2. The user should be able to enter signup information.
3. The user should be redirected to the login page upon successfully signing up.
4. The user should see an error message if the sign up did not go through.

[6] Acceptance Criteria:
Tester: Eric

1. When a user signs in, the redux store should be holding basic user information

[7] Acceptance Criteria:
Tester: Francisco

1. If a user enters the proper credentials, then a jwt will be encoded.
2. The jwt will have the user email in it as well

[8] Acceptance Criteria:
Tester: Eric
1. When a user logs in, the users information should be set in redux store
2. This action should be automatically called upon successful login

[9] Acceptance Criteria:
Tester: Eric
1. The user information should no longer be in the redux store once the log out button is called

[10] Acceptance Criteria:
Tester: Eric
1. When a user refreshes a page, the user session should stay the same
2. The token should also be refreshed from another dispatch

[11] Acceptance Criteria:
Tester: Eric

1. When a user clicks the reset password button, an email should be sent to the user's account
2. This email will redirect the user to an encoded route

[12] Acceptance Criteria:
Tester: John

1. When a user logs into the listings page, all the postings should be displayed in card forms

[13] Acceptance Criteria:
1. When a user hits the reset button, the endpoint should receive a request with the new password
2. This password should be swapped out with the old password in the db

[14] Acceptance Criteria
Tester (Aditya)

1. The user should see a button that he/she can click on to see an HTML input to enter an email to reset the password.
2. The user should see a success message upon successfully reseting the password.
3. The user should see an error message if he/she is unable to reset the password.

[15] Acceptance Criteria:
1. When a user inputs a new password, this should be saved in a state for future use

[16] Acceptance Criteria
Tester (John)

1. The user should be able see all of his/her profile information pre-filled inside HTML tags.
2. The user should be able to see reflected changes if he/she makes any change and clicks submit.

[17] Acceptance Criteria:
1. A user should be able to change parts of their profile
2. When the profile gets refreshed, the changed fields should remain the same

[18] Acceptance Criteria
Tester (Eric)

1. The user should be able to see a left menu bar that contains a "Edit Profile" link
2. The user should be able to click on "Edit Profile", which will render the Edit Profile section on the right interface.

[19] Acceptance Criteria:
1. When a user views the navbar, there should be section for profile
2. When a user clicks on the section it should redirect the user

[20] Acceptance Criteria
Tester: Aditya
1. Get SendGrid API key
2. Set up Flask-SendGrid functionality
3. Call mail function when a user registers for out service, after the user has been inserted into the database

[21] Acceptance Criteria

Tester: Aditya

1. Attach token to link that is sent to email
2. Make sure front end can accept token endpoint
3. Allow for new password to be typed at frontend endpoint
4. when new password is sent from frontend to backend, update that row in the database