Park-Me! Final Report

Abu Nadim Kabir
Georgia State University

Owusu Bonsu
Georgia State University

Rafael Hernandez Almazan
Georgia State University

Santhosh Ramachandran
Georgia State University

Zexuan Liu
Georgia State University

Abstract

In this present era, social media attracts people to present their views [1] has a lot of impact in our lives. We all are well aware of the fact that social media has a lot of influence over people's mental Health like depression, dementia, schizophrenia etc. Although the usage of social media platform like Facebook, Instagram , twitter and software engineering together is not well understood, these mechanisms influence the software development practices. Software developers use and integrate into a wide range of tools ranging from code editing web-based portals . In our research project we would like to discuss about software engineering practices implemented in our project "Park-Me". We used resourced data from Mapbox API, React and Firebase in our project. Using this data gives the way into utilizing the machine learning models and can be extended to deep learning methods such as CNN, AE [2] and in real live scenarios such as twitter analysis.

1 Introduction

After typing in a destination, our application, "Park Me!" helps users find and reserve a parking spot in a parking lot close to their desired destination. "Park Me!" is a web application that can search for parking lots near their desired destination and show their availability, distance, hours and price. If the parking lots allow it, the product also have a reserve function. Users could reserve a specific parking spot they want by selecting it in the application and then pressing the reserve button. The targeted audiences are customers who have trouble finding a parking place at their destination and worry that they will not find an available spot in a parking lot once they arrive at their destination. This application will also allow users to locate their vehicle in a parking lot much faster.

There are some alternatives to this application currently available on the web like Parker, ParkWhiz, and Honk. However, our product has some compelling features. First of all, Park Me! allows the user to reserve parking places 90 minutes before arriving, which is a pretty long time to help users make plans ahead of time. Second, our approach is user friendly and easy to handle. All customers need to do is type in their desired destination. The system would then search for parking lots around the destination and show all the related information about the parking lots on the map and a list on the application side. Next, customers can choose the parking lots they want and even reserve a particular parking spot. After confirming the parking place and successfully paying the necessary fees, the application would direct the user to the parking lot by providing a google maps link. Third, Park Me! is very flexible. If parking policy allows, users can manipulate or cancel the reservation when they need it. The payment would change based on the change of reservation.

Another flexibility of Park Me! is that it can also search for parking locations within an area rather than near a specific location. For example, you can search the area "Lenox" instead of "Lenox Square Mall". This function would allow you to be more spontaneous. Forth, the user shall be able to save their favorite parking spots. This feature would save some time when the user wants to park in the same location. Lastly, after arriving at the parking location, the parking lot system would scan the user's QR code for entrance. Using QR code is very convenient and efficient. If the QR code does not work, the user can use their phone number

Proceedings of the 24[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130.

or name as proof of reservation.

This application is supported by Express (Node JS), React JS and a list of APIs. For displaying the destination location and available parking spots, the MapBox, and ParkMobile APIs will be utilized. The application will use a payment transaction API like square or stripe to handle and verify all transactions. Finally, we will use Google's Firebase as a DBMS to manage the user's account, favorite spots, reservations and their personal information. We hope this application can help people when they are in a pinch and need to quickly reserve a parking spot.

2 Data

2.1 Data Management System

The product uses Google Firebase, which stores data in the JSON format. Example of how data would look:


```
"User_Account":[
    {
        "first_name": "Owusu",
        "last_name": "Bonsu",
        "username": "OBonsu",
        "email_adresss": owusu@test.com,
        "password": "12345",
        "phone_number" "9999999999",
        "renter_ID": "",
    }
]
```


Figure 1: Data in the JSON format

2.2 Data Specification and Analysis

User account

First Name (String)	Last Name (String)	Username (PK) (String)	Email Address (PK) (String)	Password (String)	Phone Number (String)	Renter_ID (FK)

Figure 2: User Account

Vehicle Details (Vehicle_Details)

License Plate (PK) (String)	Make & Model (String)	Year (int)	State (String)	Color (String)

Figure 3: Save Vehicle details

Reservations

Reservation_ID (PK)	Starts (Date & Time)	Ends (Date & Time)	Duration (int)	Address (String)	Price (int)	Payment Info (String)	Status (Boolean)

Figure 4: Reservation Information

Payment Types (Payment_Types)

Card Number (PK)	Expiration (MM/YY) (Date & Time)	CVC (int)	Full Name (String)	Card Description (String)

Figure 5: Payment Types and information

3 Use Cases

3.1 Show available parking lots nearby for the user after typing in their destination

Actors User, Mapbox

Description 1. The user starts searching for a location using the search feature by typing their desired location.

2. If the system was not able to find any results, the user will be asked to try again to type another search

3. The system will load the results in the map view and list view based by the closest the distance to their destination

4. The user is now able to see available parking spots near their destination and may add any filtering for customized results

Alternative paths Views all the search results without any filters Exception Paths: The system was not able to find any results based on the user's search

Precondition User logins in the system

Postcondition User is able to see all the available parking spots

3.2 The user wants to view the details of the parking lot near their destination

Actors User, Mapbox

Description 1. The user completes the use case 1 2. The user selects their desired parking lot to view the specific details regarding the location on the list view

Alternative Path Views this parking lot by searching its name

Exception Path Details of the parking lot is not available

Precondition Complete use case 3.1

Postcondition User is able to view the details of the parking lot

3.3 The user wants to reserve a parking lot before driving to the lot

Actors User, Mapbox

Description 1. The user completes the use case 3.1 or use case 3.2

2. The user selects the applicable parking spot who would like to reserve

3. The user chooses the space number and selects how much time to reserve the spot

4. If the system was not able to continue with the request, the user will be informed that the spot or the time frame has been occupied or taken, and has to choose a different selection

5. The system will request the user to enter their payment information and confirm the reservation details

6. If the system did not proceed the transaction, the user must choose a different payment method or try it again

7. The system will generate a confirmation order with details about the reservation and send them through their email

Alternative Path Pay at arrival instead of paying in advance or use a different payment method

Exception Path Spot gets full before checkout

Precondition Complete use case 3.1 or 3.2

Postcondition User is provided with directions to this spot

3.4 The user wants to show its reservation QR code to the parking lot system

Actors User, Mapbox

Description 1. The user completes the use case 3.3

2. The user can either retrieve the QR code by

a. Opening their confirmation email and locate the QR code

b. Logging into the website on their phone and go under reservations

3. User shows this code to the administrator or machine get scanned their reservation

Alternative Path User uses his name as proof of reservation or confirmation order

Exception Path QR code is not working or is expired or invalid

Precondition Complete the use case 3.3

Postcondition User is able to park at their spot

3.5 The user wants to add a favorite location of parking lot to their account

Actors User, Mapbox

Description 1. The user completes the use case 3.2

2. The user chooses their prefered parking spot location by pressing the “favorite” button and the system will store the information in the user’s account

3. If the system was not able to store the information, the user will be asked to try it again or to log in/register in case if the user does not own an account

Alternative Path User adds this to their favorites and takes it out of the favorites

Exception Path The user is not logged in or does not have an activated account

Precondition The user must be logged in or have registered to have an account available

Postcondition User is able to retrieve this spot much faster than before

3.6 The user wants to view parking lots with specific filtering requirements

Actors User, Mapbox

Description 1. Complete use case 3.2

2. User adds filters such as gated, covered/open space etc.,

3. User selects a specific parking lot to view the details

Alternative Path User searches for this location

Exception Path User does not get any results for the specified requirements

Precondition Complete use case 3.2

Postcondition User can view the details of the spot and make a decision

3.7 The user wants to see available parking lots near by their current location

Actors User, Mapbox, Internet Browser

Description 1. The user selects to search by pressing the location button on the top right side of the map

2. The browser will ask the user to accept sharing their current location with the system

3. If the user does not accept the permissions, the system will not retrieve their information and the user should try it again by pressing the button or refreshing the website

4. The user is provided with the list of lots and the map will also populate those spots based on their current location

Alternative Path User searches

Exception Path User did not grant permission to access location

Precondition User should consider the browser restrictions in order to provide the location services to Park Me!

Postcondition User is able to see all the available lots

3.8 User wants to save their vehicle information to their account

Actors User, Mapbox

Description 1. The user logs in to their account

2. If the user does not have an activated account, the user must first register to continue

3. The user goes to their profile and selects "Vehicles" section

4. The user enters credentials about the vehicle such as license, title number etc., and submits the information

5. The system will verify that the fields are correctly filled in, and will save the information to their account, otherwise it will ask the user to correct the missing fields

Alternative Path User needs to register first to add vehicle information

Exception Path User tries to the exits the registering screen without saving

Precondition User has an account or registers to a new account

Postcondition User's vehicle is registered with Park Me!

3.9 The user wants to get a notification if their reservation to the parking lot is almost over

Actors User, Mapbox

Description 1. Complete use case 3.3

2. The system will ask the user a time period before the reservation ends to be notified via email

3. The user selects the time period, and confirms the information to be submitted

4. The system will notify the user

Alternative Path User is able to extend their reservation time

Exception Path User does not have notifications turned on

Precondition Complete use case 3.3

Postcondition User picked up the car before the end time or completes use case 3.10

3.10 The user wants to modify their reservation

Actors User, Mapbox

Description 1. Complete the use case 3.3

2. The user can either

   a. Add more time to the reservation in the website

   b. Add more time in the parking facility

   c. Cancels the reservation in use case 3.12

3. The user will not be able to modify the reservation if

a. The reservation has exceeded the time allowed

b. The payment method was declined, or the information is incorrect

c. The reservation is over, and needs to reserve a new one

d. The parking spot is taken for another user

4. The user verifies all the changes and pays for the new charges

5. The system confirms the user and emails the updated reservation

Alternative Path User changes the location or cancels the reservation in use case 3.12

Exception Path Parking lot do not have availability at the new time

Precondition Complete the use case 3.3

Postcondition Specified time has been changed and accepted by the parking lot

3.11 The user wants to find available parking spots within a specific location (ex. within Chamblee) using the search function

Actors User, Mapbox

Description 1. The user starts searching for a specific location like city (Chamblee) or zip code by using the search feature

2. If the system was not able to find any results, the user will be asked to try again to type another search

3. The system will load the results in the map view and list view based by the specific location

4. The user is now able to see available parking spots near the specific location and may add any filtering for customized results

Alternative Path Searches by zip code of a city

Exception Path Searched location is not found

Precondition User needs to login

Postcondition User can view all the available parking lots in the city

3.12 The user wants to cancel their reservation

Actors User, Mapbox

Description 1. The user completes the use case 3.3

2. If the user has not logged in, the user must log in first to retrieve the reservation

3. The user navigates to their account, and chooses reservations section

4. The user chooses the reservation and chooses the cancel option

5. The system will ask to confirm if the user wants to cancel the reservation

6. The system confirms the cancelation and emails the cancelled confirmation to the user

Alternative Path User cancels by calling the parking lot or in person

Exception Path The parking policy does not allow the user to cancel or the system was not able to proceed the cancellation request

Precondition Complete the use case 3.3

Postcondition User cancels his reservation successfully

3.13 The user wants to register an account

Actors User, Firebase

Description 1. The user selects register on top right of the interface

2. The user fills out all the required fields of the form, and match all the required parameters

3. The system will verify that the information is correct, and submit the information to the database

4. The system will notify the user if

a. The user has already created an account before with the email

b. The information does not match with the field requirements such as passwords

c. The user did not fill any of the required fields

d. There is an error with proceeding their request

5. The user has now an active account in the system, will be directed to the homepage and have an active session

Alternative Path The user cancels the registration form or logs in if the user has an account

Exception Path User gets notified that there was an error while proceeding their request

Precondition The user must not have an active account

Postcondition The user successfully created an account

3.14 The user wants to login with their account

Actors User, Firebase

Description 1. The user selects login on top right of the interface

2. The user fills out all the required fields of the form, and match all the required parameters

3. The system will verify that the information is correct, and submit the information to the database

4. The system will notify the user if

a. The information does not match with the field requirements such as passwords

b. The user did not fill any of the required fields

c. There is an error with proceeding their request

d. The system did not find any account with the given information

5. The user has now logged into the system, will be directed to the homepage and have an active session

Alternative Path The user cancels the login or registers if they do not have an active account

Exception Path User gets notified that there was an error while proceeding their request

Precondition The user must have an active account

Postcondition The user successfully logged in

4 System Modeling

4.1 Class Diagram Model

There are 4 classes in our class diagram. This diagram can represent saveVehicle() getCurrentLocation(), seeNearbyParkingPlaces(), reserveParkingLot(), payParkingFee(), modifyReservation(), cancelReservation(), etc.

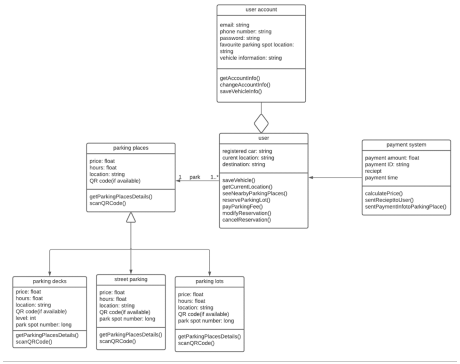


Figure 6: Class Diagram of Park Me

5 Architecture Modeling

5.1 Behavioral modeling

5.1.1 Sequence Diagram 1

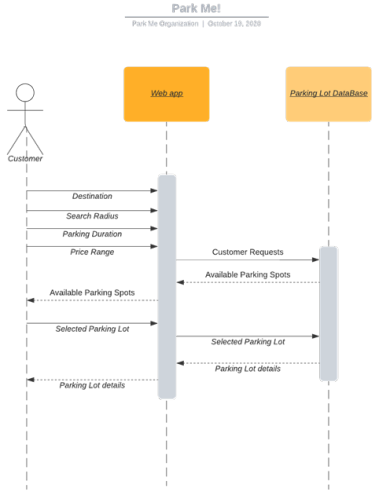


Figure 7: Use Case- The user wants to view the details of the parking lot near their destination

User Case Name: The user wants to view the details of the parking lot near their destination

Actors: User, Mapbox

Description:

1. The user completes the use case 1: Show available parking lots nearby for the user after typing in their destination

2. The user selects their desired parking lot to view the specific details regarding the location on the list view

Alternative Path: Views this parking lot by searching

its name

Exception Path: Details of the parking lot is not available

Precondition: Complete use case 1

Postcondition: User is able to view the details of the parking lot

5.1.2 Sequence Diagram 2

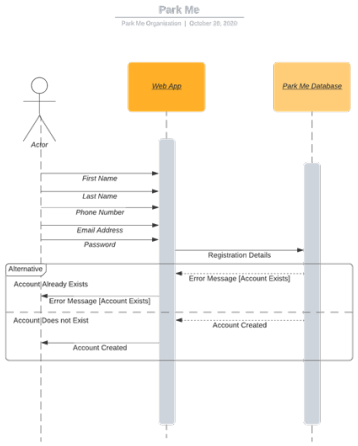


Figure 8: Use Case- The user wants to register an account

User Case Name: The user wants to register an account

Actors: User, Firebase

Description:

1. The user selects register on top right of the interface

2. The user fills out all the required fields of the form, and match all the required parameters

3. The system will verify that the information is correct, and submit the information to the database

4. The system will notify the user if

5. The user has already created an account before with the email

6. The information does not match with the field requirements such as passwords

7. The user did not fill any of the required fields

8. There is an error with proceeding their request

9. The user has now an active account in the system, will be directed to the homepage and have an active session

Alternative paths: The user cancels the registration form or logs in if the user has an account

Exception Paths: User gets notified that there was an error while proceeding their request

Precondition: The user must not have an active account

Postconditions: The user successfully created an account

5.1.3 Sequence Diagram 3

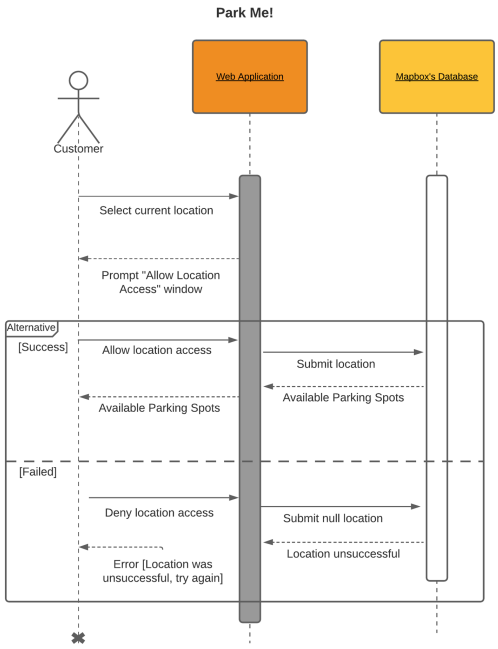


Figure 9: Use Case- The user wants to see available parking lots near by their current location

User Case Name: The user wants to see available parking lots near by their current location

Actors: User, Mapbox, Internet Browser

Description: 1. The user selects to search by pressing the location button on the top right side of the map

2. The browser will ask the user to accept sharing their current location with the system

3. If the user does not accept the permissions, the system will not retrieve their information and the user should try it again by pressing the button or refreshing the website

4. The user is provided with the list of lots and the map will also populate those spots based on their current location

Alternative Path: User searches

Exception Path: User did not grant permission to ac-

cess location

Precondition: User should consider the browser restrictions in order to provide the location services to Park Me!

Postcondition: User is able to see all the available lots

5.1.4 Sequence Diagram 4

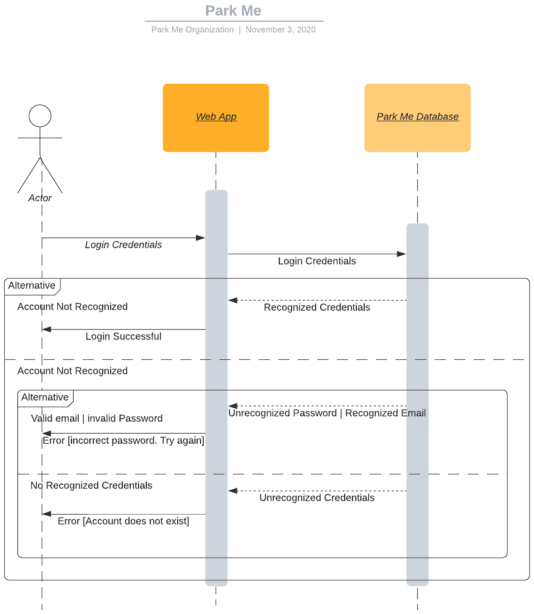


Figure 10: Use Case- The user wants to login with their account

User Case Name: The user wants to login with their account

Actors: User, Firebase

Description: 1. The user selects login on top right of the interface

2. The user fills out all the required fields of the form, and match all the required parameters

3.The system will verify that the information is correct, and submit the information to the database

4. The system will notify the user if the information does not match with the field requirements such as passwords

5. The user did not fill any of the required fields

6. There is an error with proceeding their request

7. The system did not find any account with the given information

8. The user has now logged into the system, will be directed to the homepage and have an active session

Alternative paths: The user cancels the login or registers if they do not have an active account

Exception Paths: User gets notified that there was an error while proceeding their request

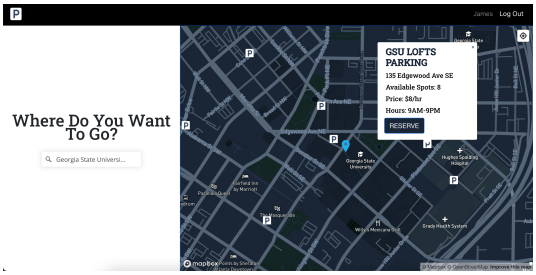
Precondition: The user must have an active account

Postconditions: The user successfully logged in

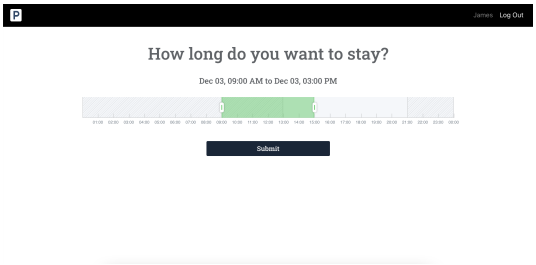
6 Implementation

6.1 Frontend

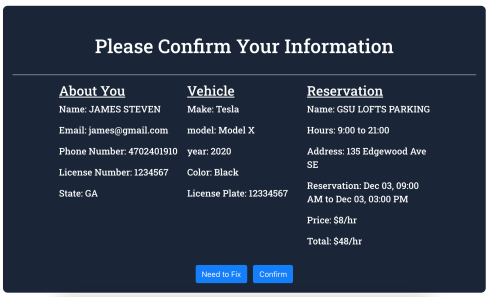
Homepage By clicking the parking location, it will show details such as address, hours, price, ect.



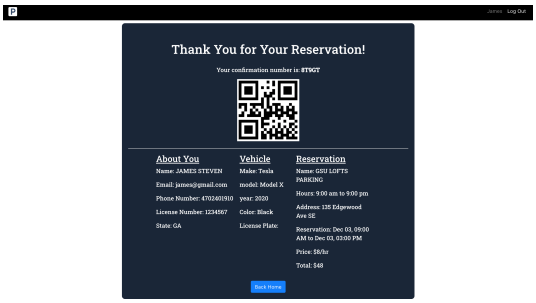
Choose Time Period By dragging the bar, the user can choose reservation time period.



Confirm Before the final reservation, the application will show user the information about this order.

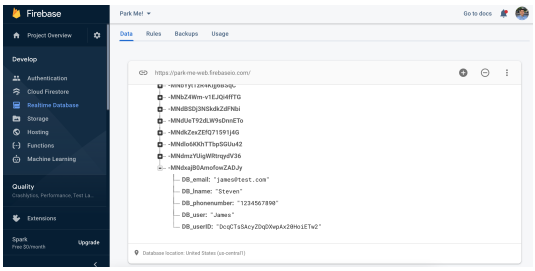


Reservation Info After successfully reserving a parking place, the application will show a page contains all the details about this order.



6.2 Backend

Database This page shows how the data in firebase looks like.



7 Testing

Use Cases used for testing are User Log In, User Register and User Search for location.

7.1 Executable Unit Tests

7.1.1 Renders login Form

```
test('Renders the login form', () => {
    const { getAllByText, getByPlaceholderText } = render(<Login />);

    getAllByText("Log in");
    getByPlaceholderText("Email");
    getByPlaceholderText("Password");

  });
```

Purpose of Test Makes sure that the login form is rendered completely, making sure that the email, password, and login button all show

Test Environment Jest/react testing library

Test Steps Render page

Scan document for log in button, email and password fields

Test Input n/a

Expected Result Test should pass if all elements are present and rendered

7.1.2 User inputs valid login information

```
test('Allows for user input', () => {
    const { getAllByText, getByPlaceholderText, getByLabelText, getByRole } = render(<Login />);

    const inputemail = getByPlaceholderText("Email");
    const inputpassword = getByPlaceholderText("Password");
    fireEvent.change(inputemail, { target: { value: "test@test.com" }});
    fireEvent.change(inputemail, { target: { value: "testtesttest" }})
    fireEvent.click(getByRole('button', { name: "Log in" }));
})
```

Purpose of Test Tests to see if user is able to log in using valid login credentials

Test Environment Jest/react testing library

Test Steps Render page

Input test values into email and password fields

Click “Log in” button

Test Input Email: test@test.com

Password: test@test.com

Expected Result Test should pass if login credentials are in valid format

Likely Problems/BugsRevealed Does not navigate to homepage

7.1.3 Renders registration Form

```
test('Renders the registration form', () => {
    const { getAllByText, getByPlaceholderText } = render(<Register />);

    getByPlaceholderText("First Name");
    getByPlaceholderText("Last Name");
    getByPlaceholderText("Email Address");
    getByPlaceholderText("Phone Number");
    getByPlaceholderText("Password");
    getByPlaceholderText("Retype Password");
```

Purpose of Test Makes sure that the login form is rendered completely, making sure that the email, password, first and last name, phone number, and registration button all render correctly

Test Environment Jest/react testing library

Test Steps Render page

Scan document for First Name, Last Name, Email Address, Phone Number, Password, Retype Password

Test Input n/a

Expected Result Test should pass if all elements are valid and meet formatting requirements for the form element

7.1.4 User inputs valid registration information

```
test('Allows for user input', () => {
    const { getAllByText, getByPlaceholderText, getByLabelText, getByRole } = render(<Register />);

    const inputemail = getByPlaceholderText("Email Address");
    const inputpassword = getByPlaceholderText("Password");
    const inputFirstName = getByPlaceholderText("First Name");
    const inputLastName = getByPlaceholderText("Last Name");
    const inputpasswordRetype = getByPlaceholderText("Retype Password");
    const inputphoneNumber = getByPlaceholderText("Phone Number");

    fireEvent.change(inputemail, { target: { value: "test@test.com" }});
    fireEvent.change(inputpassword, { target: { value: "testtesttest" }});
    fireEvent.change(inputFirstName, { target: { value: "Owusu" }})
    fireEvent.change(inputLastName, { target: { value: "Bonsu" }})
    fireEvent.change(inputpasswordRetype, { target: { value: "testtesttest" }})
    fireEvent.change(inputphoneNumber, { target: { value: "5555555555" }})

    fireEvent.click(getByRole('button', { name: "Register" }));
})
```

Purpose of Test Tests to see if user is able to create an account if they fill out the registration form correctly

Test Environment Jest/react testing library

Test Steps Render page

Input test values into all fields

Click “Register” button

Test Input Email: test@test.com

Password: testtesttest

RetypePassword: testtesttest

FirstName: Owusu

LastName: Bonsu

Phone Number: 555-555-5555

Expected Result Test should pass if registration details are entered correctly and meet formatting requirements

Likely Problems/BugsRevealed No message shown this.props.registerUser is not a function is shown

7.1.5 Search bar is rendered

```
test('Renders the search', () => {
    const { getByText, getByPlaceholderText } = render(<Mapbox />

    getByText("Where Do You Want To Go?");
    getByPlaceholderText("Destination");

});
```

Purpose of Test Tests to make sure search bar is rendered along with prompt asking user where they want to go

Test Environment Jest/react testing library

Test Steps Render page

Scan for text “Where Do You Want To Go?” and for search bar

Test Input n/a

Expected Result Test should pass if elements are found

Likely Problems/BugsRevealed

7.1.6 User is able to search for location and get results

```
test('Allows for user input', () => {
    const { getByText, getByPlaceholderText } = render(<Mapbox />);

    const input = getByPlaceholderText("Destination");
    fireEvent.change(input, { target: { value: "Student Center East" }});

    fireEvent.keyDown(input, { key: 'Enter', code: 'Enter' });
})

test('Asks for and finds user location', () =>{
```

Purpose of Test Tests to make sure user is able to input text into search bar and get results

Test Environment Jest/react testing library

Test Steps Render page

Scan for search bar

Enter example information into search bar

Simulate “Enter” keystroke

Test Input Destination: “Student Center East”

Expected Result Test should pass if able to type into search bar and get results

Likely Problems/BugsRevealed Unable to differentiate between successful search and unsuccessful search at the moment

7.2 Test Results and Documentation

7.2.1 Test Results

```
PASS src/Login.test.js
PASS src/Mapbox.test.js
PASS src/Register.test.js

Test Suites: 3 passed, 3 total
Tests:       6 passed, 6 total
Snapshots:   0 total
Time:        3.16s
Ran all test suites.
```

7.2.2 Test Documentation

Bug	The test uncovered the bug	Description of the bug	Action was taken to fix the bug
Does not navigate to homepage	User inputs valid login information	When the user logs in, user is not navigated to the homepage	We are working on performing navigation action on login
user's name is not displayed in the dashboard	User inputs valid login information	After the user logs in, the user's first name is not displayed in the top bar/dashboard	Working on capturing the session and the displaying the user's name
No message shown	User inputs valid registration information	When the user inputs all the information, no sign of registered notification/message is shown	We are working on displaying an alert message to notify the user
this.props.registerUser is not a function is shown	User inputs valid registration information	When the user completes registration successfully, this.props.registerUser is not a function is show like an error	This is a firebase and react issue and working on finding the root cause of this message
Does not navigate to homepage	User inputs valid registration information	When the user registers, user is not navigated to the homepage	We are working on performing navigation action on registration

References

[1] S. T. Sadasivuni and Y. Zhang, "Using gradient methods to predict twitter users' mental health with both covid-19 growth patterns and tweets," *second IEEE International Conference on Humanized Computing and Communication with Artificial Intelligence (HCCAI 2020) September 21-23, 2020 Irvine, CA, USA*.

[2] J. K. Mandivarapu, B. Camp, and R. J. Estrada, "Self-net: Lifelong learning via continual self-modeling," *Frontiers in Artificial Intelligence*, vol. 3, p. 19, 2020.