6-14-2019

HomeBook

Stephen Poth

Simon Stauber

Jake Vargas

SANTA CLARA UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

Date: June 12, 2019

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Stephen Poth
Simon Stauber
Jake Vargas

ENTITLED

HomeBook

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

[illegible]

Thesis Advisor

[illegible]

Department Chair

HomeBook

by

Stephen Poth
Simon Stauber
Jake Vargas

Submitted in partial fulfillment of the requirements
for the degree of
Bachelor of Science in Computer Science and Engineering
School of Engineering
Santa Clara University

Santa Clara, California
June 14, 2019

HomeBook

Stephen Poth
Simon Stauber
Jake Vargas

Department of Computer Engineering
Santa Clara University
June 14, 2019

ABSTRACT

Having a reliable address system is something that is often taken for granted, but simply having one comes with immense benefits. People can easily find locations of friends and family, services like taxis know exactly where to pick customers up, and emergency response units can accurately find the scene of emergencies quickly. However, in many developing countries, these are luxuries only afforded by the wealthy and privileged neighborhoods. Locations are given haphazardly through description, often leading to late arrivals. In some cases this can be a dire possibility. Our solution aims to change this by developing a mobile application that anyone can download onto their mobile device. The application uses GPS coordinates to generate a permanent address, which can then be shared with other people and services.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

The lack of a reliable address system, or not having addresses at all, is a major problem for many of the world's countries. Due to this, simple services that are often taken for granted in developed nations, such as calling an ambulance during an emergency, are inaccurate or take an extensive amount of time to find the correct location. We are currently focusing on solving this issue in Cameroon, where the primary method of giving one's address is done in a descriptive way, typically using objects and local landmarks to direct people to a location.

There currently exists a solution for creating addresses based on latitude and longitude called What3Words [1]. This company sections a map into 3x3 meter blocks and assigns three unique words to describe these blocks' coordinates. However, this solution is not viable for Cameroon and many other poverty-stricken countries because it does not inherently integrate with other services. Local companies would need to buy the rights to the technology from What3Words, something many in impoverished nations can't afford. Another solution that is currently being used by some public establishments in Cameroon are Plus Codes [2], which roughly generates a location on Google Maps. However, this is something that is only accurate within a few miles around the specific location and many local residents find it difficult to use effectively. Despite many other products that utilize GPS systems for different services, all of them need an address to function. There is currently no system in place in Cameroon that creates an address and integrates it with local services effectively.

Finding a solution to unreliable address systems is extremely important in improving the general lifestyle of those affected. For people in Cameroon, they will no longer have to rely on directions that are based on landmarks they see along the way. A working solution could open doors to businesses that deliver, help people find each other in a much simpler manner, and even save lives.

1.2 Solution

Our solution, HomeBook, is a proof-of-concept application that uses GPS coordinates to create and store a permanent home address. Users will create a password protected account and input basic information such as their name, email and phone number. They can then set their location, and store it alongside the other information as a personal contact card within the application. This contact card can be shared with anyone who has a HomeBook account, allowing them to store it as an entry on their in-app home-screen. Account information can be recovered through a secure password recovery feature, and any personal information can be edited or deleted by the user. Finally, our system will be able to be implemented on smart-devices that are available in developing or rural areas, as well as be intuitive for any user to navigate.

Chapter 2

Requirements

Requirements are an essential way for us to describe what our system accomplishes at a high-level, and highlight experiences we need to enable for our client. Requirements will also effectively limit the scope of our project and make sure that an unrealistic product is not promised to our clients. Additionally, we have identified constrains which limit our design. Table 2.1 details our functional requirements, Table 2.2 describes our non-functional requirements, and Table 2.3 lists our design constraints.

Table 2.1: Functional Requirements

Functional Requirements	
Critical	Have the ability to generate and store user's locations
	Allow users to share information and location with other users
	Have a layer of security for log-in authentication
	Allow users to recover password, delete their account, and edit contact information
Recommended	Enable users to see the visual location on a map within the application
	Allow users to update their email and password

Table 2.2: Non-Functional Requirements

Non-Functional Requirements	
Critical	Easy for users to navigate and use the application
	Quickly be able to share address information with other people and services
Recommended	The application should not be memory intensive for mobile devices
	The system should be able to handle large amounts of people and their data

Table 2.3: Design Constraints

Design Constraints
Application must operate on mobile devices in developing countries, specifically Cameroon
Application must operate cross platform, on both Android and iOS

Chapter 3

Use Cases

Our only actor is the user, who will first need to set up an account as shown in Table 3.1. Once all required information is confirmed, the user will then be taken to their home screen (Table 3.1). The home screen, shown in Table 3.3, gives the user the ability to share their information and location (Table 3.2) or search and view other users' information. If a user wants to edit their personal information or change the coordinates of their house, Table 3.4 shows that the user can do this in their personal page. Table 3.5 shows that users are able to reset their password directly from the login page. All that is needed to reset a password is the user's email. Finally, users are able to create other user's profiles as shown in Table 3.6. Once the user fills in the full name, phone number, and sets coordinates, the user profile will get added to the home screen.

Table 3.1: Store Information and Location Use Case

Use Case Name	Store Information
Goal	Be able to store information and location
Actor(s)	User
Precondition(s)	Set up account
Postcondition(s)	Information and location gets stored under user profile
Exception(s)	NA

Table 3.2: Share Information and Location Use Case

Use Case Name	Share Information
Goal	Be able to share location to another user or service
Actor(s)	User
Precondition(s)	Have an account set up
Postcondition(s)	Location gets shared to other user or service
Exception(s)	NA

Table 3.3: View Other Users' Location and Information

Use Case Name	View Other Users' Information
Goal	Be search or view other users' information
Actor(s)	User
Precondition(s)	NA
Postcondition(s)	User can view another user's information and location
Exception(s)	NA

Table 3.4: Edit Personal Information and Location

Use Case Name	Edit Personal Information
Goal	Edit user's personal information or location
Actor(s)	User
Precondition(s)	Have an account set up
Postcondition(s)	Information or location gets changed
Exception(s)	NA

Table 3.5: Reset Password

Use Case Name	Reset Password
Goal	Reset password for user
Actor(s)	User
Precondition(s)	User cannot remember password
Postcondition(s)	Password gets reset
Exception(s)	User does not have account

Table 3.6: Add User

Use Case Name	Add User
Goal	Add user profile to home screen
Actor(s)	User
Precondition(s)	NA
Postcondition(s)	User profile gets added to home screen
Exception(s)	User already exists

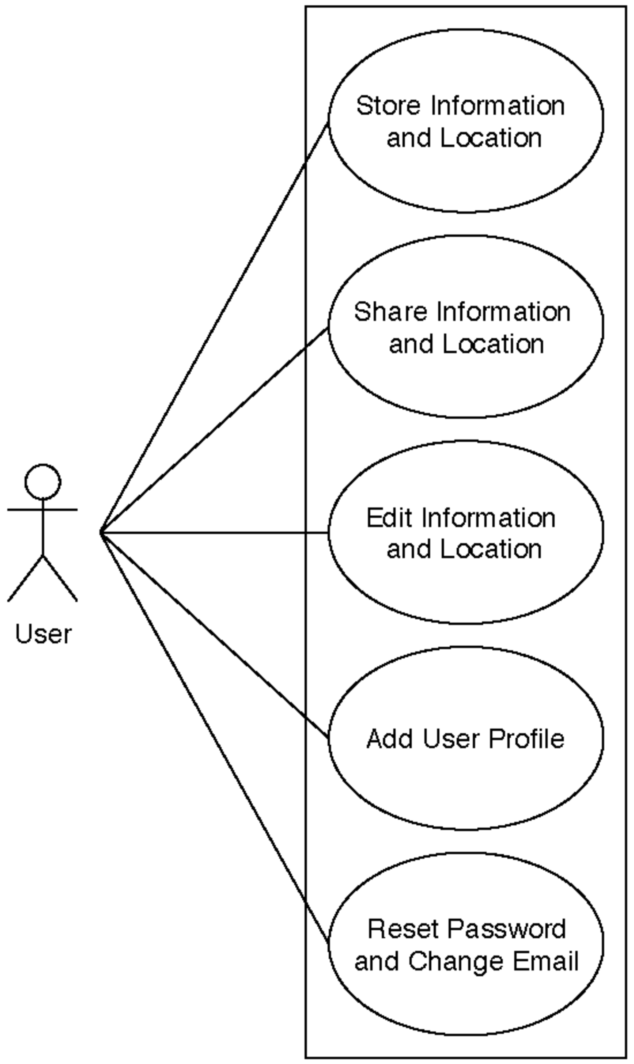

Store Information and Location
Share Information and Location
Edit Information and Location
Add User Profile
Reset Password and Change Email
User

Figure 3.1: Use Cases

Chapter 4

Activity Diagram

Figure 4.1 details the flow of how the user will use our application. Upon entering the application, users will be sent to our login page. From there, users will have three choices. Those options are to login, create and account, or change their password if forgotten. If users want to create an account they will be sent to a page where they can fill in their basic information and use their native maps interface to set a pin wherever they wish. After filling that out, users will be sent to the main home page. The second option is to login to an existing account. Users must correctly enter their email and password. If a user is successful, they will be sent to the main home page. If not completed successfully, users will remain on the login page. Lastly, if a user forgets their password, they will be sent to a form asking for their email address. If this email address is valid, the user will then be sent an email prompting them to change their password.

At the Users Home Page, users will be able to view their own profile, view other contact's profiles, share their coordinates, and add new contacts. In a user's own profile, they can edit their information and coordinates. When a user views other contacts' profiles, they can edit the information and coordinates shown, or delete the contact's profile entirely. Users are also able to share their contact card to any user that has a HomeBook account. Lastly, users can create a new contact from the Home Page. A new form will show up, asking for basic information and coordinates for the contact they want to create.

To exit the app, a user will press the hamburger menu icon and press ”Logout”. This will bring them back to the login page where they would need to enter the proper credentials to log back in.

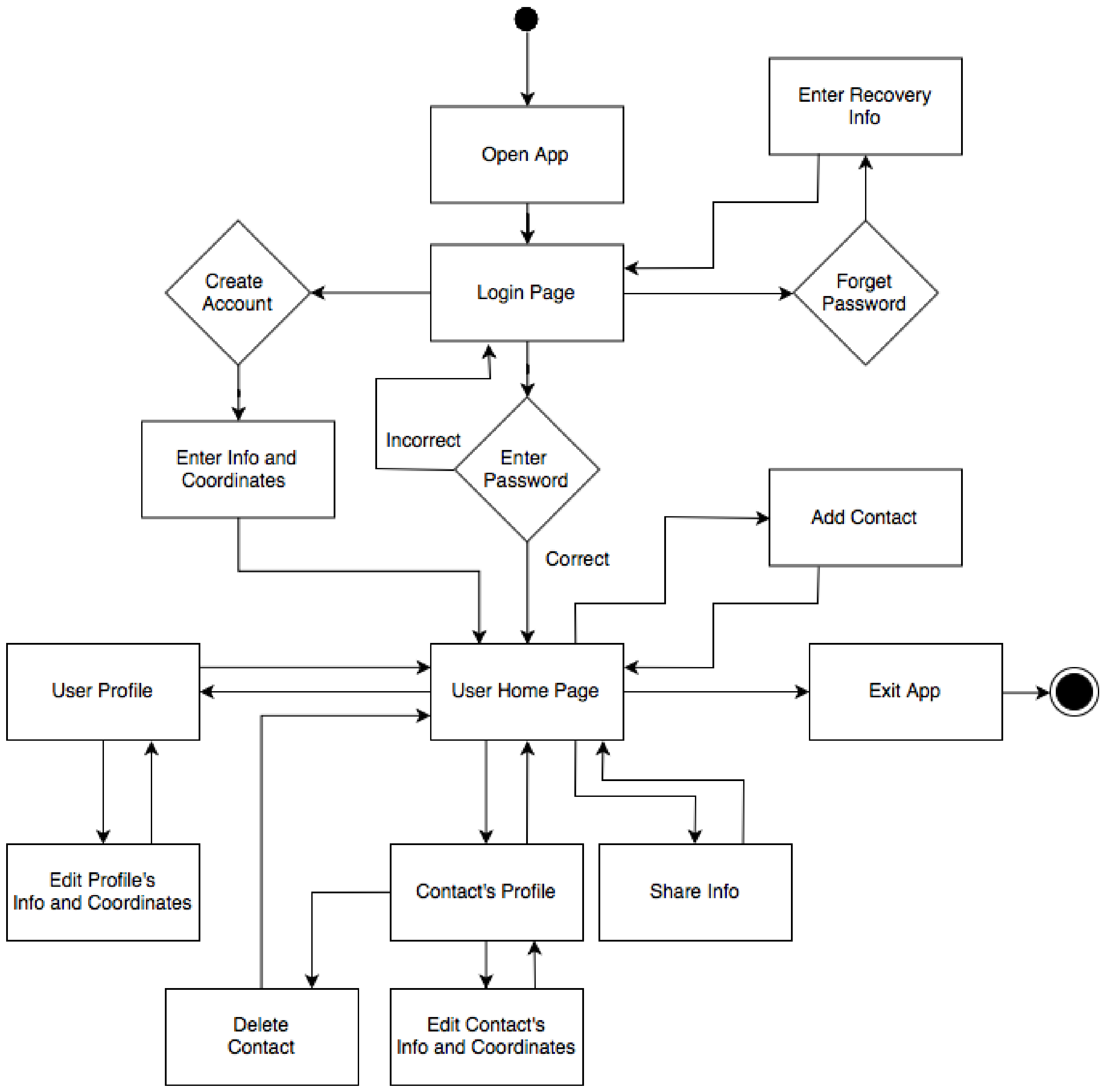


Figure 4.1: Activity Diagram for Users

Chapter 5

User Interface

5.1 Login Screen

As seen in Figure 5.1, the application begins on the login page. If a user has an account, they will enter their email and password. If a user does not have an account, they can click the ”create account” link to get set up. If the user forgot their password, they can click ”Forgot Password” to be sent to another page to recover their password.

5.2 Reset Password

Figure 5.2 shows how the password reset page looks like. A user will simply type in their email and click send. An email will be sent allowing the user to update their password. If an incorrect email is typed in, a warning will pop up to alert the user to their mistake.

5.3 Create Account

Figure 5.3 shows the create account process, in which users will enter their name, password, and phone number. Clicking the confirm button will take them to the next screen, which has them set their location. Passwords must be confirmed correctly or an error will appear, and emails will be checked to see if they are already in use or not. There cannot be more than one account tied to a single email.

5.4 Set Coordinates

After users enter their textual information, they will be sent to the ”Set Your Home” screen as seen in Figure 5.4. A native map interface will allow the user to set a pin wherever they wish, the map can be manipulated as it is a touch-screen. If a user would like to use their current location, they can simply click the ”Current Location” that is highlighted blue, and the map will drop a pin at the present location. After ”Create Account” is pressed, the user is sent back to the login page and a verification email will be sent. A message will pop up to warn the user to finish this step before logging in.

5.5 Home Page

Figure 5.5 shows what the User Home screen looks like. In the top right corner, the user can touch this icon to be sent to another page that deals with sharing their location. If users want to see their own information, they can click their own name at the top. The list of users can be scrolled through, and if the user wants to see any of the information they simply need to click on a name. If a user wants to manually add a new contact to their list, they can touch the plus icon in the bottom right corner of the page. The menu icon in the top left corner will expand to show additional settings that the user can interact with.

5.6 Share Contact

Figure 5.6 shows where the user goes when they want to share their location and contact information with another HomeBook user. They need to input a valid email of another user and press ”search”. A button will show up with the searched user’s name. When pressed, it will send their contact information to that user’s ”Contact Request” page which can be seen in Figure 5.8.

5.7 Settings

The extended menu, shown in Figure 5.7, contains different settings that can be accessed by the user. If they want to check if another user has sent their contact information over, they can touch ”Contact Requests” which will take them to Figure 5.8. The ”Reset Password” and ”Change Email” will take the user to different pages to edit the respective information. The user also has the option to logout and delete their account; however, when trying to delete they need to input their password first as an added layer of security.

5.8 Contact Requests

If a user is sent someone else’s contact information, the ”User Request” page, as shown in Figure 5.8, is where this information will be displayed. Pressing ”Accept” adds the new user’s information to the home screen, and pressing ”Decline” will remove that user without adding their information to the home screen.

5.9 Reset Password

Once users select ”Reset Password” in the extended menu, they are taken to the screen shown in Figure 5.9. To reset their password, the initial password must first be inputted to authenticate the user. The user then inputs the new password twice to assure they entered it correctly. Pressing ”Update Password” after the information is properly entered will update the user’s password.

5.10 Change Email

Once users select ”Change Email” in the extended menu, they are taken to the screen shown in Figure 5.10. To update their email, the initial email and password must be inputted to authenticate that the actual user is trying to reset their email. If all information is entered correctly, the user will be sent a confirmation email to the new email once ”Update Email” is pressed. If the user logs out, they must authenticate the new email before being able to log back in.

5.11 User Profile

The user profile, displayed in Figure 5.11, will show the user’s name, phone number, email, and pinned location. They can also edit this information at any time.

5.12 Other User Profile

Users can store other users’ profiles in the application and Figure 5.12 shows what this will look like. The only functional difference between this page and the personal user profile is the option to delete the user profile.

5.13 Add Contact

Figure 5.13 shows how a user can add another user profile. They will be able to fill in the first name, last name, email, and phone number. Similarly to the ”Set Coordinates” page, users will be able to set pins for the contact they are creating. Once the ”Add profile” button is clicked, that profile will be added to the user’s home page.

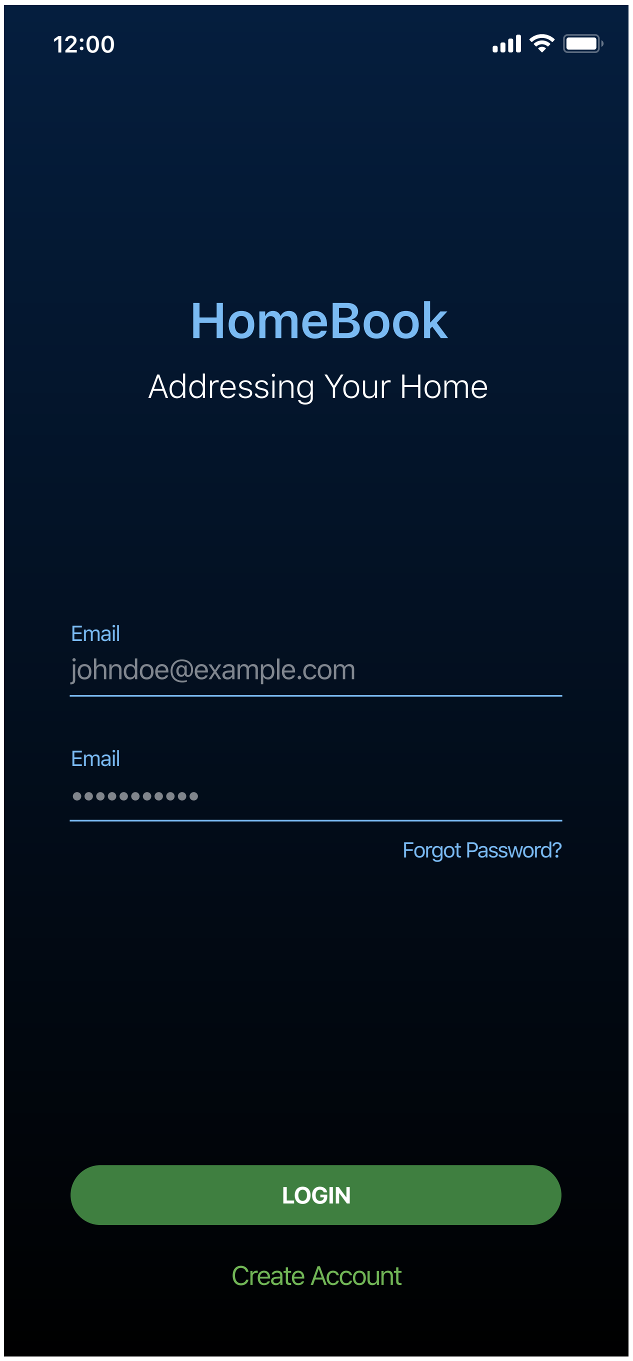


Figure 5.1: Login Screen

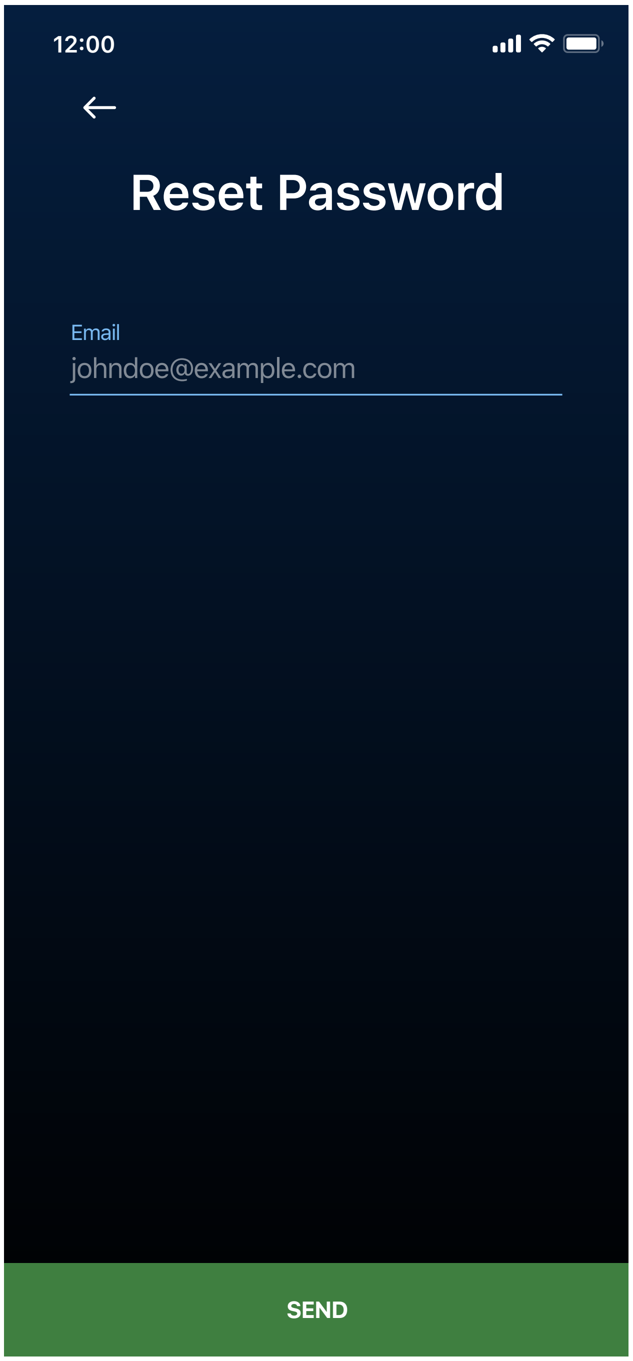
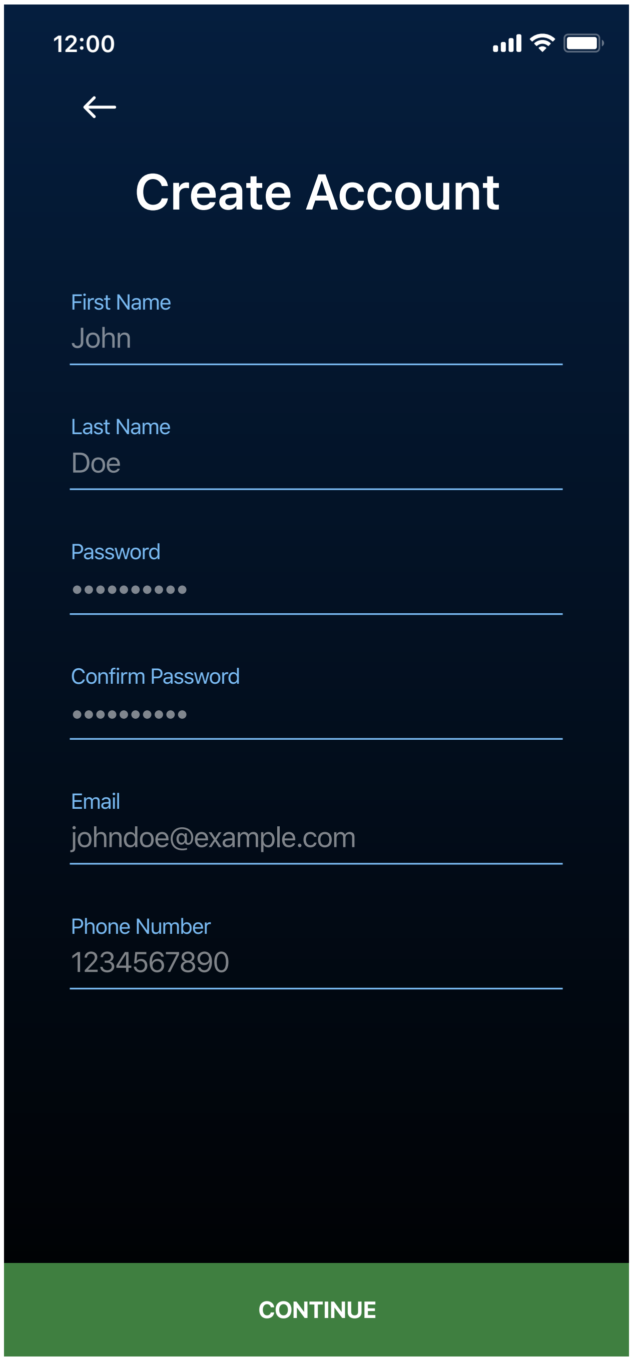


Figure 5.2: Reset Password


12:00

Create Account

First Name
John

Last Name
Doe

Password
••••••••••

Confirm Password
••••••••••

Email
johndoe@example.com

Phone Number
1234567890

Figure 5.3: Create Account

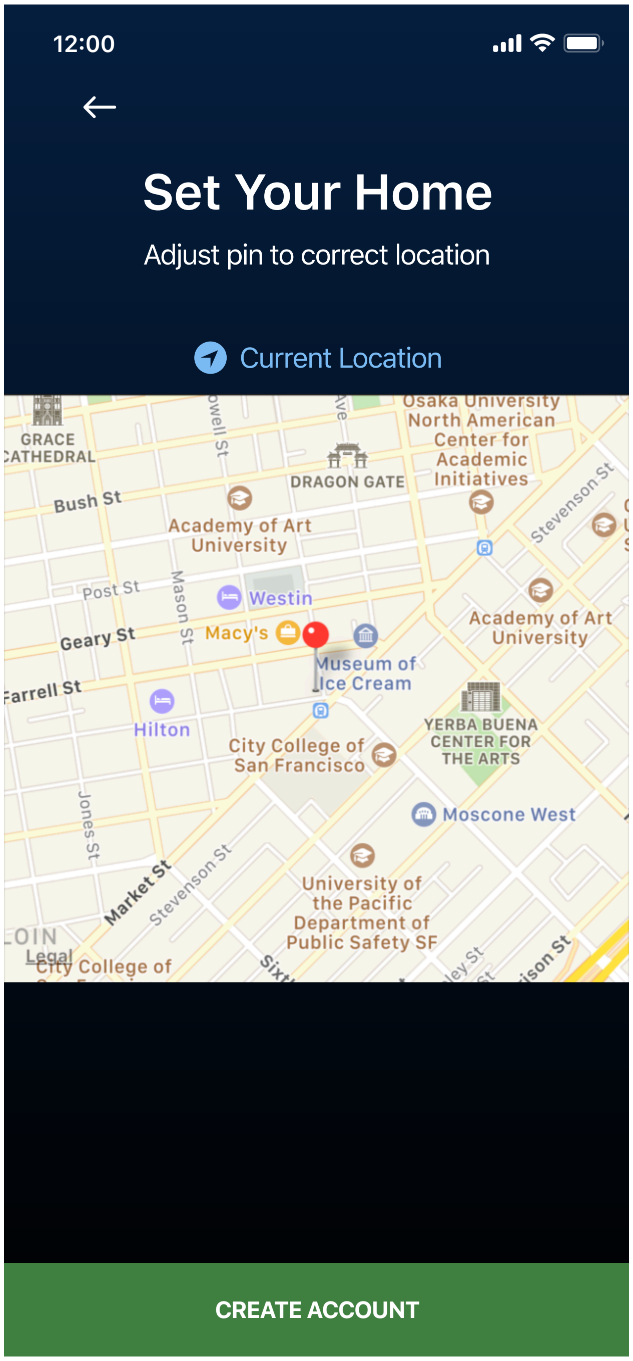
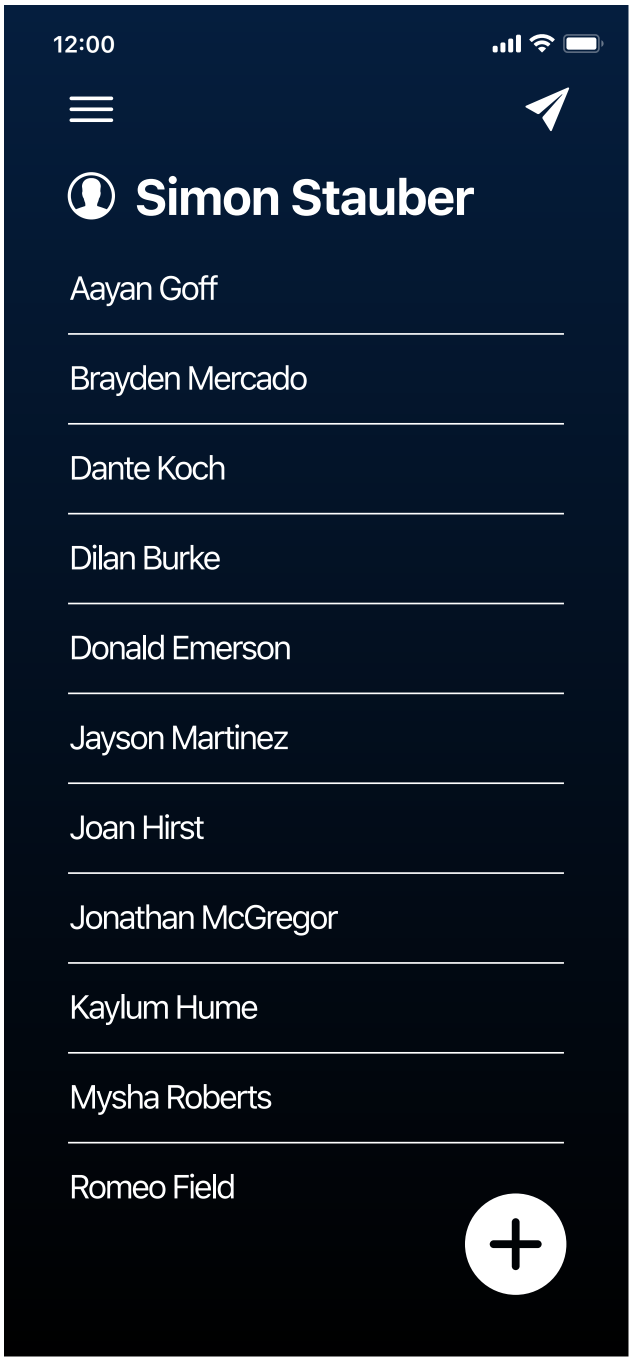


Figure 5.4: Set Coordinates



Simon Stauber

Aayan Goff

Brayden Mercado

Dante Koch

Dilan Burke

Donald Emerson

Jayson Martinez

Joan Hirst

Jonathan McGregor

Kaylum Hume

Mysha Roberts

Romeo Field

Figure 5.5: User Home Page

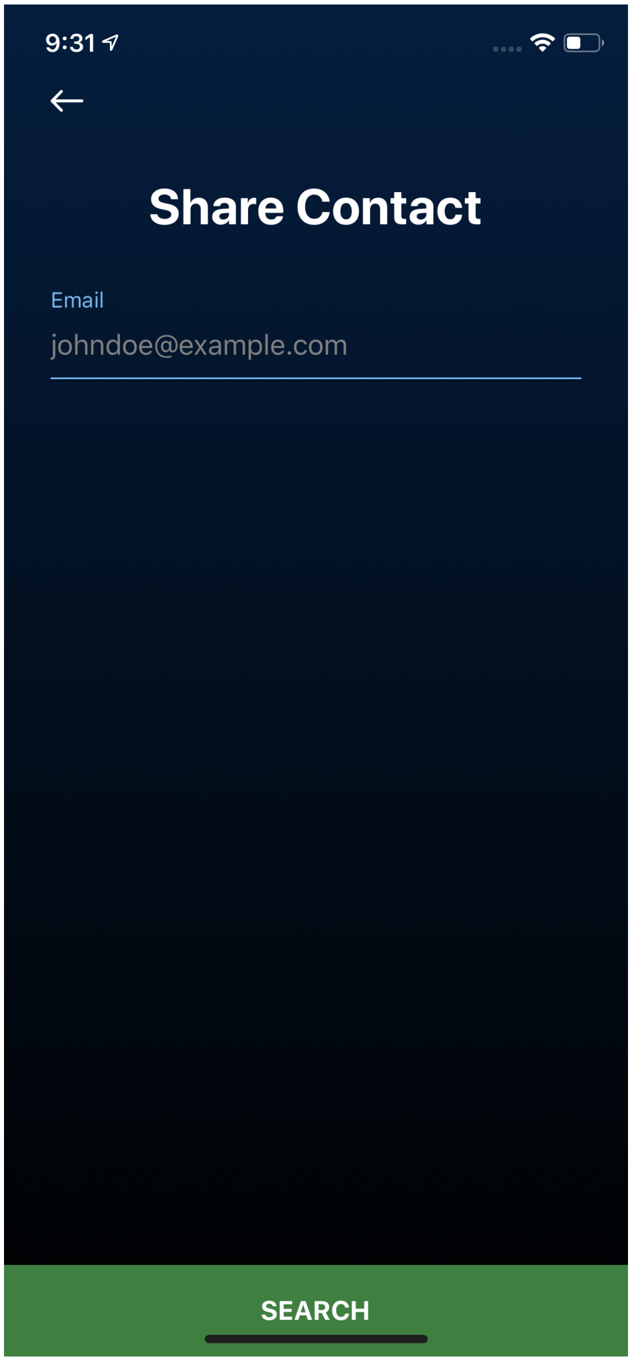


Figure 5.6: Share Contact

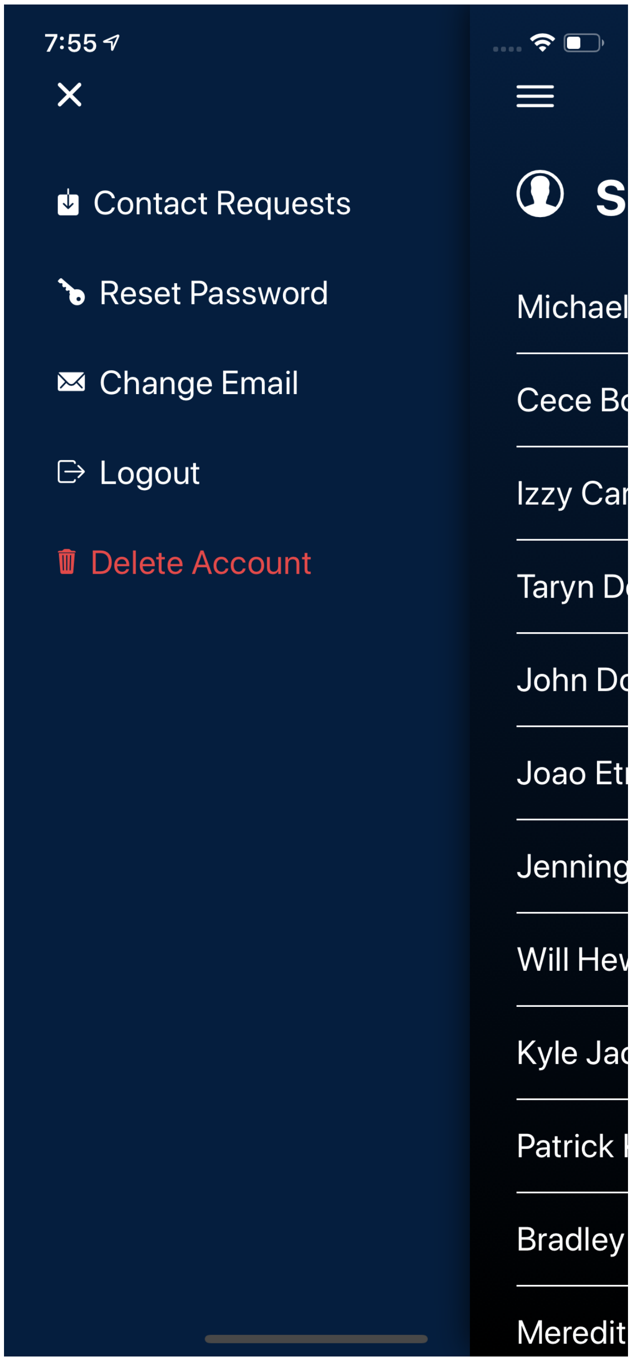


Figure 5.7: Settings

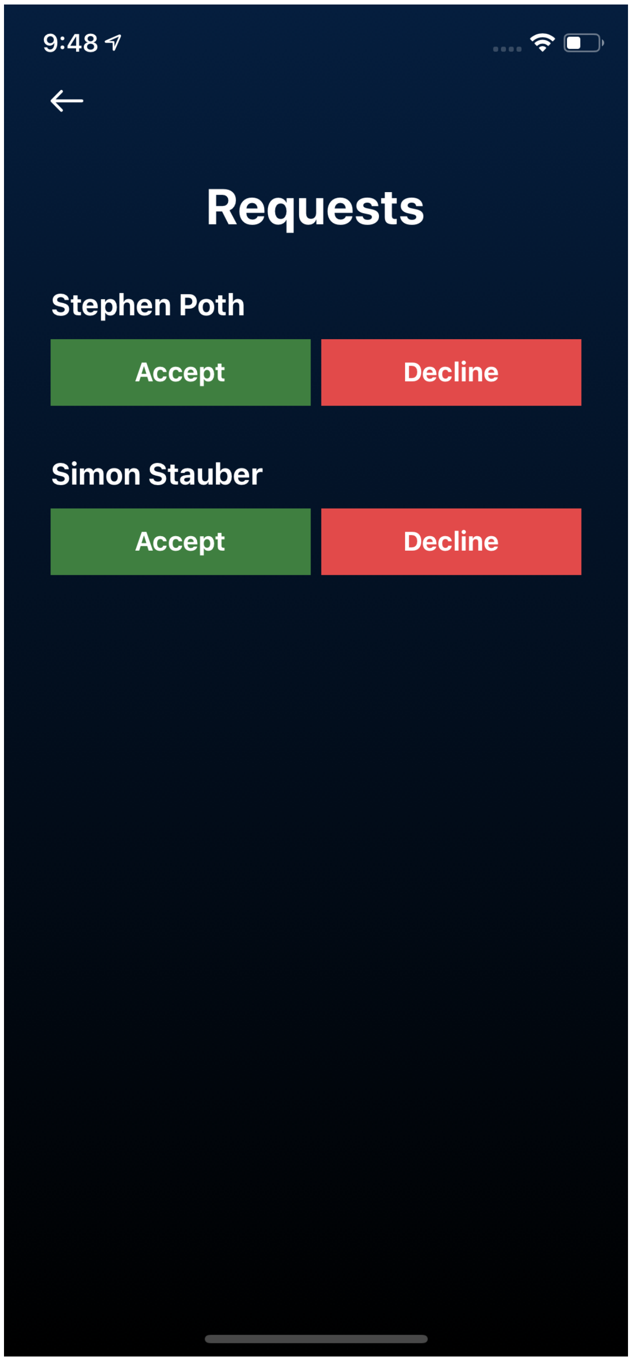


Figure 5.8: Contact Requests

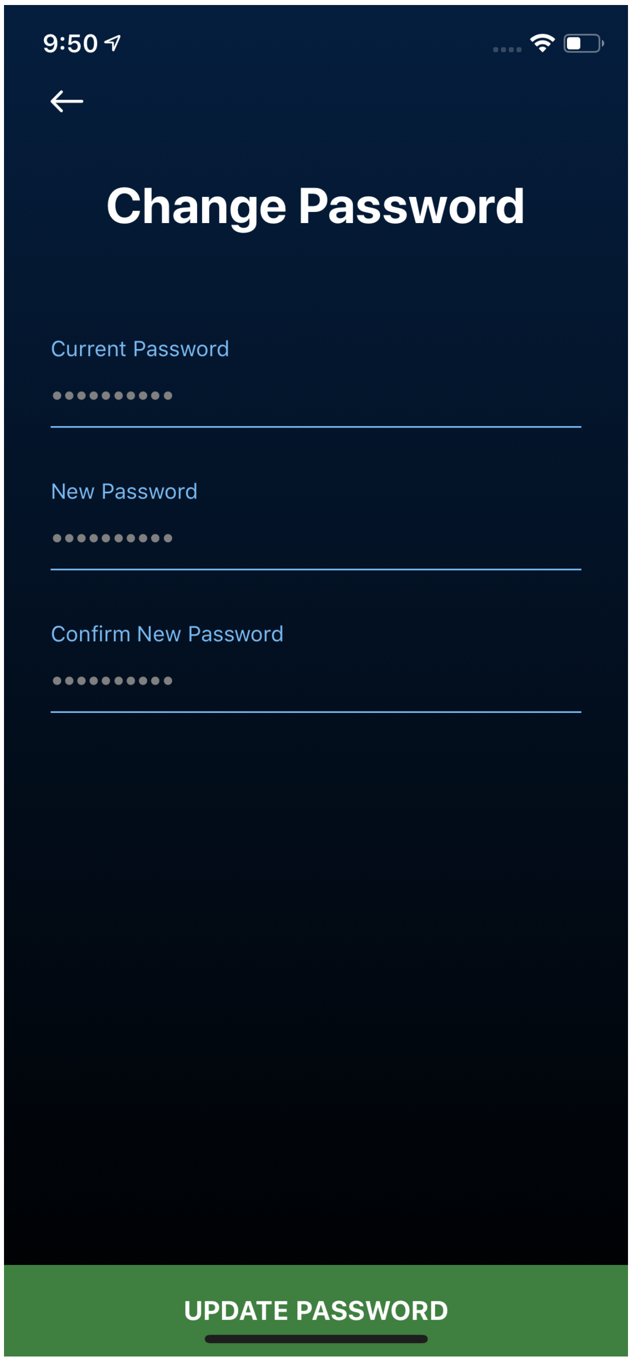


Figure 5.9: Reset Password

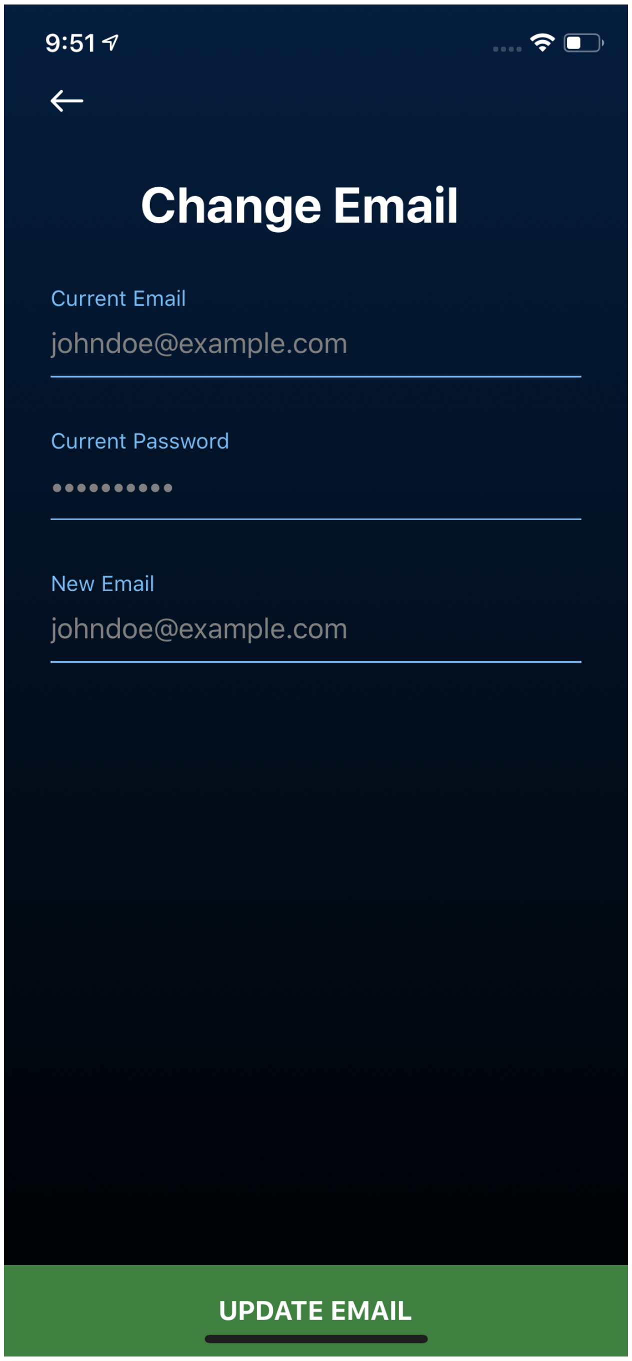
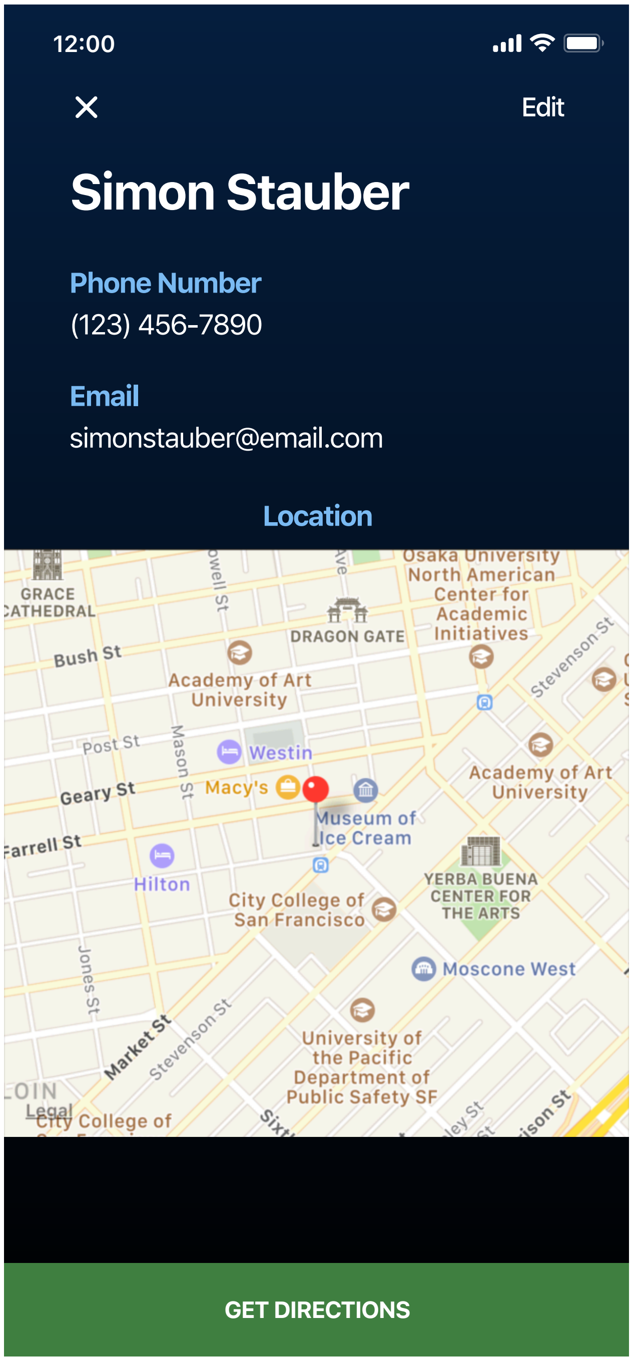


Figure 5.10: Change Email


12:00

Edit

Simon Stauber

Phone Number

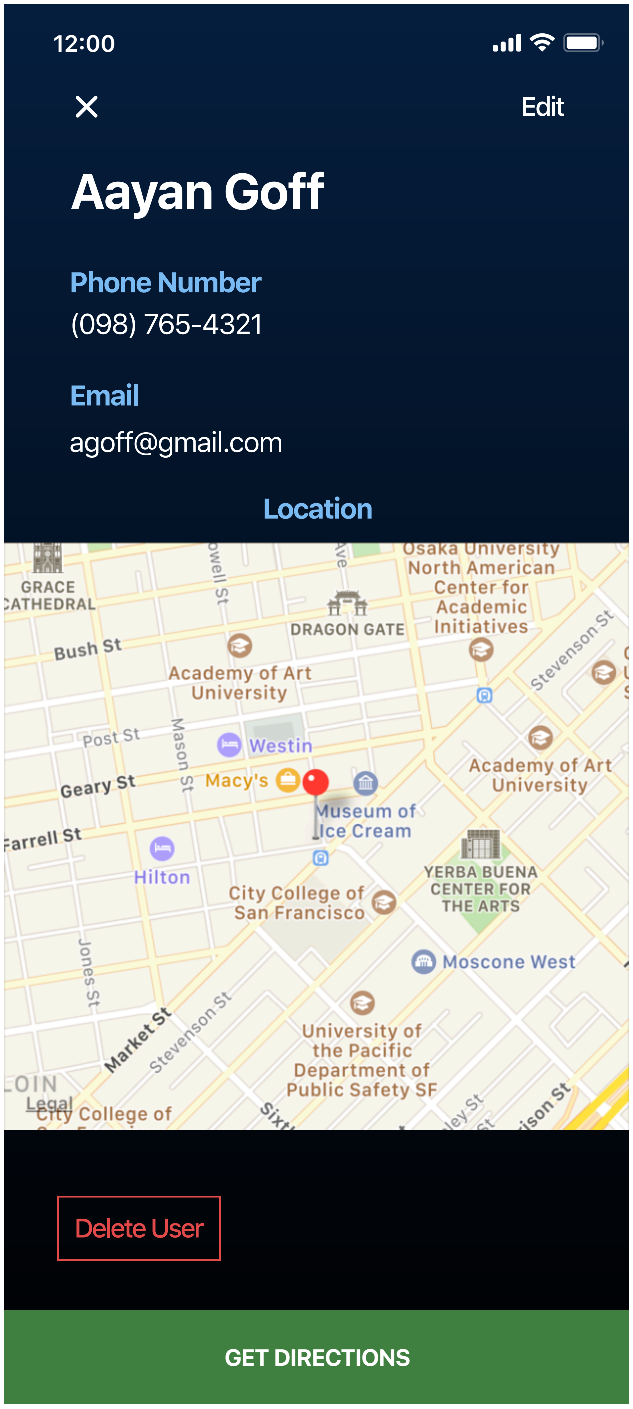
(123) 456-7890

Email

simonstauber@email.com

Location

Figure 5.11: User Profile


12:00

Edit

Aayan Goff

Phone Number

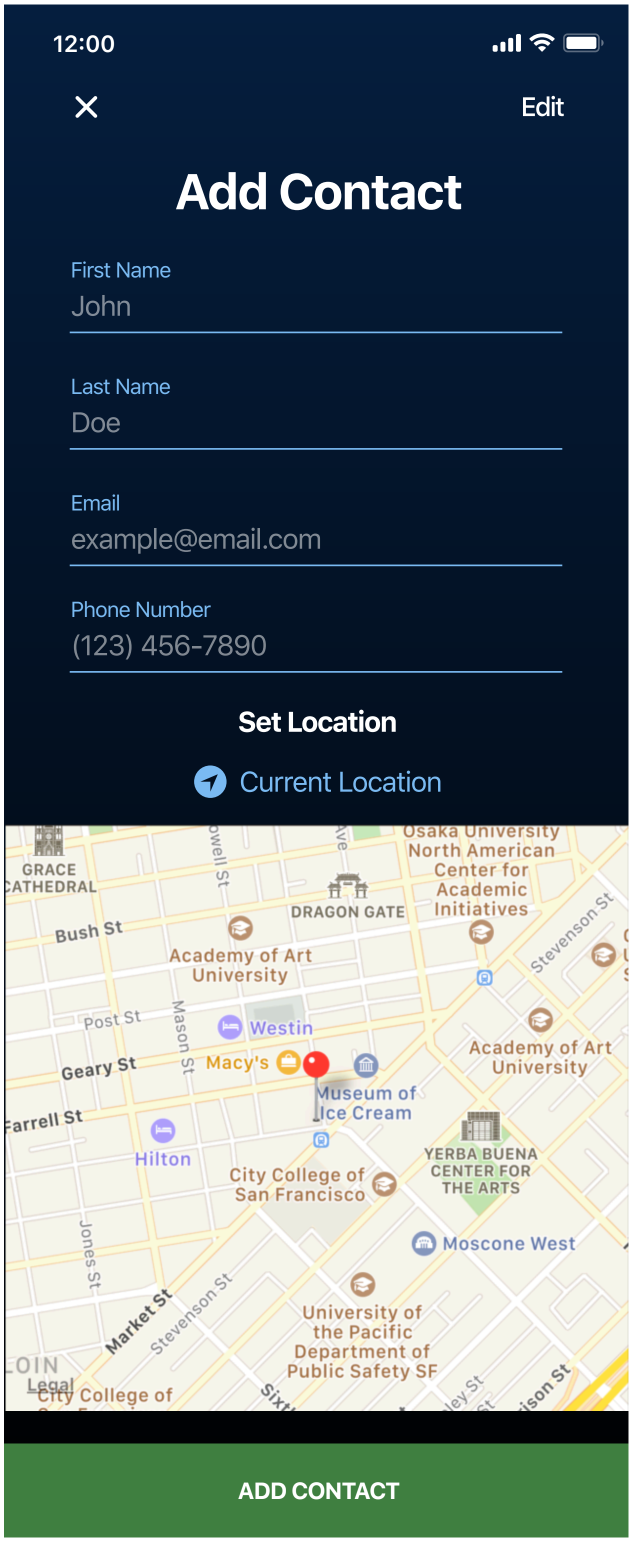
(098) 765-4321

Email

agoff@gmail.com

Location

Figure 5.12: Other User Profile



12:00

Edit

Add Contact

First Name
John

Last Name
Doe

Email
example@email.com

Phone Number
(123) 456-7890

Set Location

Current Location

ADD CONTACT

GRACE CATHEDRAL
Bush St
Academy of Art University
DRAGON GATE
Osaka University North American Center for Academic Initiatives
Stevenson St
Post St
Mason St
Westin
Geary St
Macy's
Museum of Ice Cream
Academy of Art University
Farrell St
Hilton
City College of San Francisco
YERBA BUENA CENTER FOR THE ARTS
Moscone West
Jones St
Market St
Stevenson St
University of the Pacific
Department of Public Safety SF

Figure 5.13: Add Contact

Chapter 6

Technologies Used

We wanted to make sure the framework we chose to build for this application would be cross-platform compatible, which is the main reason in using React-Native as the UI framework. React-Native is known for its cross platform capabilities, and it was the optimal choice for our project. It allowed us to move forward quickly and more efficiently with our project, as many redundancies between platforms were eliminated. In terms of state management, we used Redux due to its versatility. It allowed for all the pages to update and render automatically whenever the state changed. Google's Cloud Firestore and Firebase were used as the database and server to securely store all user information, and were selected for their simple integration with React-Native and scalability. Their API had easily implemented functions which provided a wider range of uses. An example of this is the built-in authentication platform. It enabled email/password authentication and email verification to work seamlessly with the rest of our application. Additionally, because the passwords are handled by Firebase directly, they are not stored in the databases connected to the application itself, which adds another layer of security. Finally, Git was used as our version control through the duration of the project and GitHub was the collaborative platform we used to keep all the code properly updated.

Table 6.1: Technologies Used

Technology	**Purpose**
Git	Version Control
React-Native	UI Framework
Redux	State Preservation
Firestore	Database
Firebase	Server

Chapter 7

Architectural Diagram

We used a Client-Server architecture as the basis for our design. The server handles querying and editing our database. Meanwhile, clients display the information retrieved from the server. Lastly, our database contains the information and locations from the users. Users can access our system through their mobile devices, after logging into a created account or creating a new account. If users lose their phone, the information stored in the database can be resent to the user when they reconnect to their account.

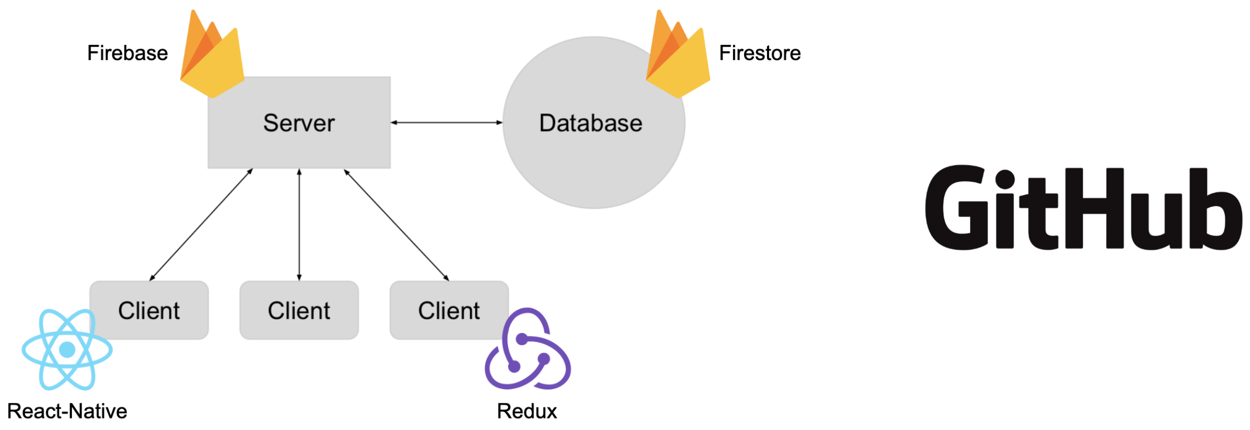


Figure 7.1: Architectural Diagram

Chapter 8

Design Rationale

Starting with the architecture, we used a client-server model because it allows for a single point of access for all user information. Our design also requires an easy method to quickly add and remove a large amount of clients, which is a strength found with this type of architecture.

Moving on to the technologies, used using React-Native to build our application for its cross-platform compatibility. However, it is important to note that our target clientele are those in developing countries, where the majority of smartphones are Android. Because of this, we have focused our testing and implementation on Android devices. Firestore and Firebase were used for our back-end because of their versatility. The API were easy to implement and had a lot of additional functionalities that worked well with our application. The built-in authentication is a great example of this, as it is simple and secure. One additional function of our application is the ability to see coordinates on a map, which we have implemented using the native map API. For Apple, it is their Maps API and for Google, it is Google's Maps API. Each SDK is an open source utility found online that is easy to implement, and something that will improve the usability of our application.

For our back-end development, a lot of the heavy lifting is done through the Firebase and Firestore API. The language that is used by this is Node.JS which is open-source and cross-platform - suiting our needs perfectly. Redux and React.JS were helpful with the state management and styling. We didn't need to manually update states, as Redux automatically updated all pages within the application whenever there was a change in state. React.JS was cross platform, so many of the styling we initially created did not have to be adjusted to fit the other platform, which saved a lot of time.

Chapter 9

Test Plan

The majority of testing was unit testing. Every time we created a new function, we would test how it would affect our overall project. We would look at how the new function would affect things like database security, database entries, other functions, how it affected the application aesthetically, and how it affected the current navigation stack for the different screens. We sent the function unique inputs with the goal of trying to break our code. Only through this could we be sure that the system was robust. These unit tests were helpful in finding bugs in the current function, or other functions. Unit testing was important because we needed to be clear how our code was specifically affecting different parts of our application.

Another aspect that we tested heavily was login security. We used Firebase's authentication to ensure the users security. This is important because we did not want other users to be able to access a users information, either accidentally or with malice. Therefore, we made sure the app was impossible to enter without the users emails and password. We also did test to ensure that users could not create an account with the email of another person. We ran several unit tests with the goal of trying to penetrate a user's account, or create accounts with another person's email. This helped us ensure that users information was safe, and led us to create warnings that guided people through the process of restoring the account to the correct person.

Our group also did a great deal of beta testing. Throughout the development, we showed HomeBook to family, friends, and colleagues. We conducted beta testing for two reasons: the first being to detect bugs that our group had missed, and the second being to ensure that we were building an intuitive UI. Since we were actively developing the product, the UI was familiar to us, but we wanted to make sure it was intuitive to others. To address this, we made sure to show HomeBook to people with both technical and non-technical backgrounds. This was effective because we were building this application for a large group of people with ranging levels of technology literacy. Beta testing did an excellent job accounting for this, and helping structure HomeBook to be user friendly.

Chapter 10

Risk Analysis

The risk analysis table shows different risks and consequences involved with the design and implementation of our application. We had identified the accompanying probability and severity to each risk, and quantified them on a scale from one to ten. The product of these two values was then used to determine the total impact the risk has towards completing the project. Each risk had mitigation strategies to reduce the likelihood of the issue occurring. Throughout our design process, many of these risks occurred at some point. The largest problem we faced was the scope creep in the beginning. We were overly ambitious with some of the design choices, which ended up taking time away from other, more important features. Bugs were always common, however, in terms of severity it wasn't a problem to work through them. Poor team health slowed us down during the winter months, however it was always only one member who was afflicted. The rest of the team was able to pick up where the other left off, and made sure to stick to our timeline as best as possible.

Table 10.1: Risk Analysis

Name of Risk	**Consequences**	**Probability**	**Severity (0-10)**	**Impact**	**Mitigation Strategies**
Time	System not complete by deadline	0.1	8	0.8	Follow Gantt chart Schedule weekly team meetings Re-evaluate necessity of features
Bugs	Must allocate time dedicated to debugging	1.0	2	2	Maintain thorough documentation of code Perform rigorous unit testing
Team Health	Illnesses push back our development timeline	0.4	4	1.6	Make sure each member understands entirety of design Consider personal health a priority
Scope creep	Unrealistic requirements are continuously added to the project	0.4	9	3.6	Perform code reviews Frequent, effective communication between team members and client Allow customer to provide feedback
Loss of work	Accidents in handling project code can result in deletion	0.1	10	1	Save work often and in different places Use git as version control system to keep records of code
Technical limitations	Unable to implement certain requirements and/or features	0.6	3	1.8	Familiarize ourselves with technologies used Be honest regarding technical skills with self, team and client

Chapter 11

Development Timeline

The development timeline in Figure 11.1 shows the rough time-frame we followed to complete our senior project on time. A highlighted square indicates a month that a task was worked on. The dark blue are major tasks that were needed to be completed, while the lighter blue represents a minor task that needed focus for the major task to be completed correctly.

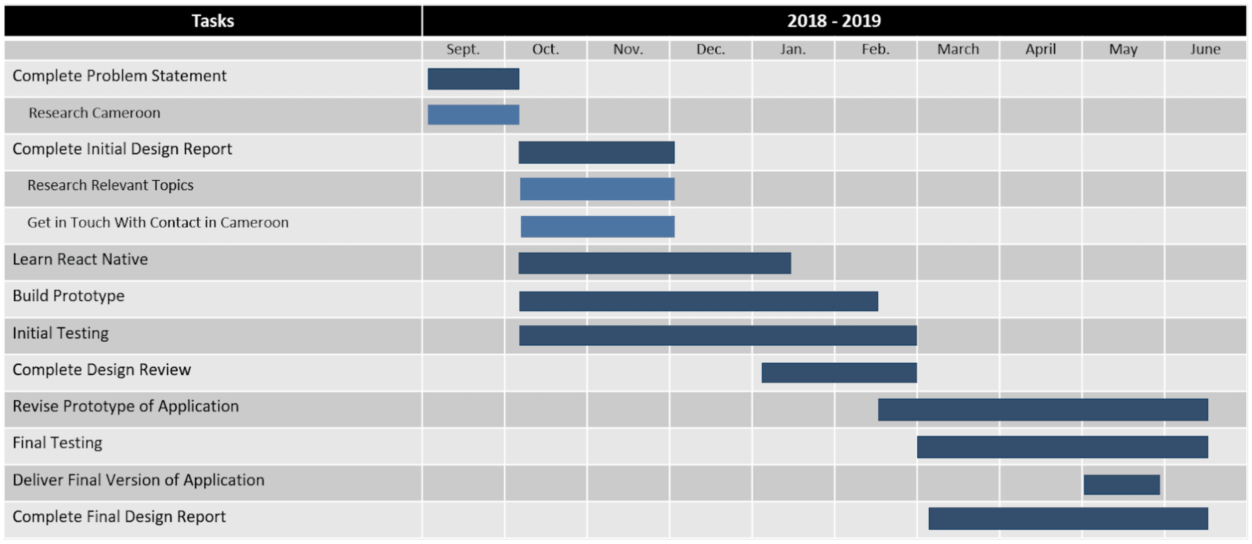


Figure 11.1: Development Timeline

Chapter 12

Societal Issues

12.1 Ethical

The main ethical question that pertains to HomeBook is the possibility of personal information being lost or handled poorly. We realize that sensitive information, such as where people are living is something important to keep safe. In light of this, we made sure that passwords are never seen by anyone with access to the back-end database. Our password handler is done through Google itself and never crosses paths with our database of information. In addition to this, in terms of front-end security, we have included email-password authentication and email verification. When sensitive actions are being taken in the application itself, such as deleting an account or changing a password, verification is always in place.

12.2 Social

The purpose of HomeBook is to vastly improve the lives of people in developing nations by giving them the benefits of having a permanent address. There is a slight social aspect when looking at sending profiles to other users, but this is a small portion of the purpose of the application. We have included a check to make sure someone accepts before adding contact information to their home screen.

12.3 Political

Our application has no relations to political issues in any way other than to hopefully advocate the use of a reliable address system for any developing country.

12.4 Economic

The application is intended to be free for all users, the only cost would be incorporating a widespread database that scales to the number of users. With this being the only cost, we don't believe the economic impact would be an issue.

In fact, the application was intended for those who were less fortunate and couldn't afford something like a home address, which is mainly found in the wealthy areas of developing nations.

12.5 Health and Safety

HomeBook is a proof-of-concept as is, however we realize that in the future there could be some concerns regarding the use of HomeBook and services such as emergency response units. Our hope is that eventually, HomeBook can be used to generate addresses that emergency response units can use to better navigate to areas in need. We have implemented HomeBook on the API of Apple Maps and eventually Google Maps, which have a presence worldwide. Each have reliable map and direction functions, which are already a positive step compared to the current description based systems used in many developing nations. The other thing to keep in mind as future work towards incorporating services to HomeBook continues, is to make sure that sharing contacts never fails, especially with a service like an emergency response. However, as it currently is, HomeBook has little impact on the health and safety of others.

12.6 Manufacturability

HomeBook is a mobile application that has been built, which addresses the manufacturability issue. The only problem that could arise is with scalability. With a huge amount of users, the costs for database and server usage will slowly start to increase.

12.7 Sustainability

HomeBook is an extremely sustainable app in the sense that it has been built to be able to scale to any number of users, allowing for it to be used far into the future with little to no maintenance. The only thing required to keep HomeBook working is paying for the database storage, which was chosen due to it's scalability and low cost.

12.8 Environmental Impact

HomeBook is a mobile application, so there is no environmental issue in this context.

12.9 Usability

We designed HomeBook's navigation based on multiple user walkthroughs, which gave us the structure that user's were able to understand the fastest. Clear icons that represented actions and low cognitive loads were utilized to achieve a clean, easy to learn interface.

12.10 Lifelong Learning

This project had a great impact on the entire team's want to always be learning new technologies and techniques. The first few months were difficult to learn all new languages and frameworks, but our ability to create an entire application from these languages and frameworks were a huge inspiration to continue creating and learning.

12.11 Compassion

Compassion is what led us to create this application. We learned about the issue of not having a permanent address and what the repercussions of this issue are, so we decided to create a product that solves this problem. People were dying because emergency response units could not find the correct house in time, and we couldn't stand the thought of that happening. Our hope is that this application is a step in the right direction in righting this wrong, and that the successful integration of services is in the near future.

Chapter 13

Conclusion

Our project was to create a proof-of-concept to address a global problem that stemmed from a seemingly simple problem - not having a permanent home address. We strove to create an application that worked as an address book, and that generated a home address based on latitude and longitude coordinates. This, along with other common personal information, could then be shared with other people whenever the user wanted. We included authentication to the account, as well as re-authentication whenever a sensitive action was taken in-app.

13.1 Lessons Learned

One of the main takeaways from this process was project management. Staying on task can be difficult at the best of times, so being clear on deadlines with everyone within the team is essential. It is more efficient to iron out a clear plan with specific short-term goals, rather than a broad plan over a long period of time. Short coding sprints might be a more effective way to staying on task and closer to target dates, rather than marathon sessions.

It is also important to clearly define the project scope and stick to it. We were over-aggressive in the beginning with some of our goals, and trying to accomplish everything spread us thin. It would have been more efficient to start small with essential functions, rather than try and do everything at once. It didn't help with hitting setbacks regarding our database and server early on, which pushed us behind schedule. It was a good choice to start shaving away some of the extra functionalities and focus on what would make a solid foundation for future work on this project.

13.2 Future Work

The first thing that needs to be completed is the Android Map API integration. Functionality wise, everything else works the same as it does on the iOS version of the application. However, we ran into trouble with integrating Google's Map API. Other than that, our iOS version of the application is running - so continued tests would be the next step.

In the future, HomeBook needs to integrate a services capability. Currently, HomeBook works peer to peer, so the next step would be to integrate peer to service. This would include services like taxis, emergency response, or delivery services. This is the most important step, and something that could really affect many people positively around the world. This means scalability also needs to increase as the usefulness of the application increases.

Chapter 14

References

[1] “what3words — Addressing the World” 3 Word Address: ///Wounds.client.face, what3words.com/wounds.client.face.

[2] “Addresses for Everyone.” Plus Codes, plus.codes/.

[3] schwarzmller, Maximillian. “Online Courses - Anytime, Anywhere.” Udemy, www.udemy.com/react-native-the-practical-guide/learn/lecture/8567852?start=0#overview.

[4] “Getting Started React Native.” React Native Blog ATOM, facebook.github.io/react-native/docs/getting-started.html.

[5] Google, Google, firebase.google.com/.

[6] “Google Maps Platform Documentation.” Google Maps Platform — Google Developers, Google, developers.google.com/maps/documentation/.

[7] “React - A JavaScript Library for Building User Interfaces.” - A JavaScript Library for Building User Interfaces, reactjs.org/.

[8] “GitHub Guides.” GitHub Guides, guides.github.com/.

[9] “Usage with React Redux.” Redux, redux.js.org/basics/usage-with-react.

[10] Oblador. “Oblador/React-Native-Vector-Icons.” GitHub, 7 June 2019, github.com/oblador/react-native-vector-icons.

[11] TerryGLee. “Install Visual Studio.” Microsoft Docs, docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2019.

[12] React Native Navigation - Truly Native Navigation for IOS and Android, wix.github.io/react-native-navigation/#/.

[13] “Install Android Studio — Android Developers.” Android Developers, developer.android.com/studio/install.

[14] React-Native-Community. “React-Native-Community/React-Native-Maps.” GitHub, 11 June 2019, github.com/react-native-community/react-native-maps.

[15] laki944. “laki944/React-Native-Navigation-Directions.” GitHub, 30 Apr. 2019, github.com/laki944/react-native-navigation-directions.