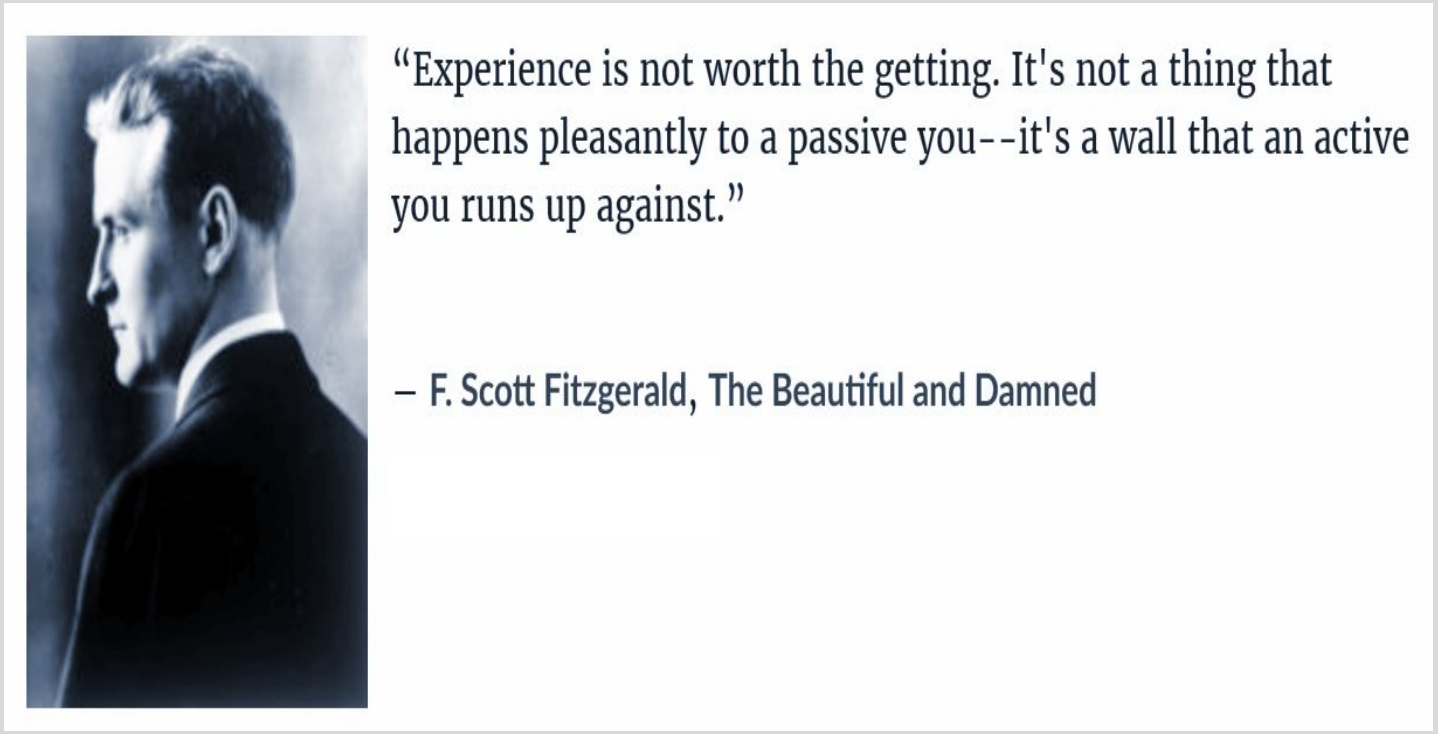
UrWrld

Constantine Firinidis, Chris Hamburger, Christopher Ferguson

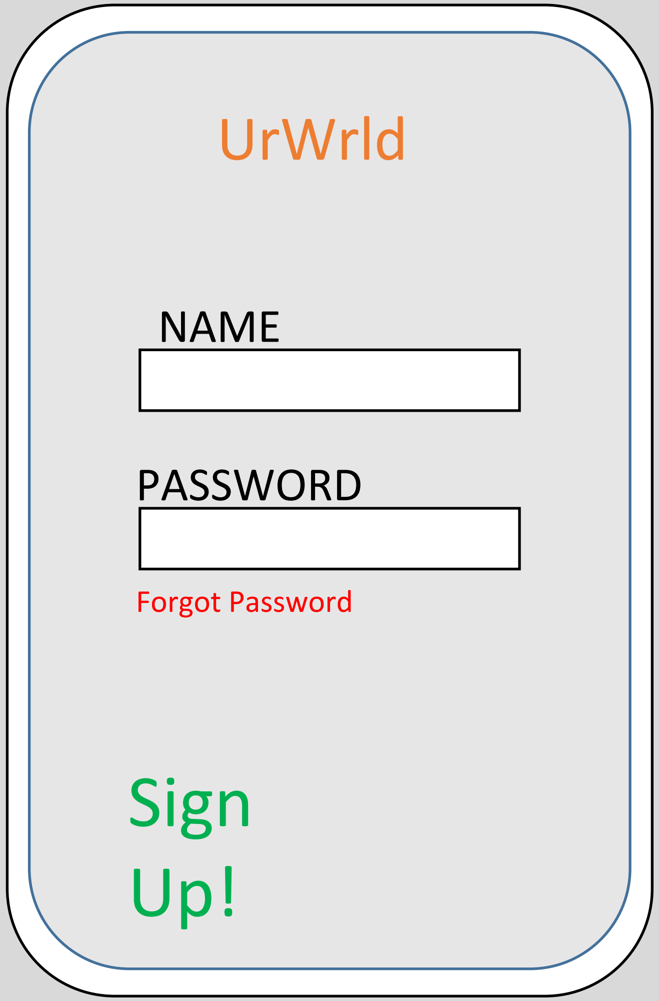


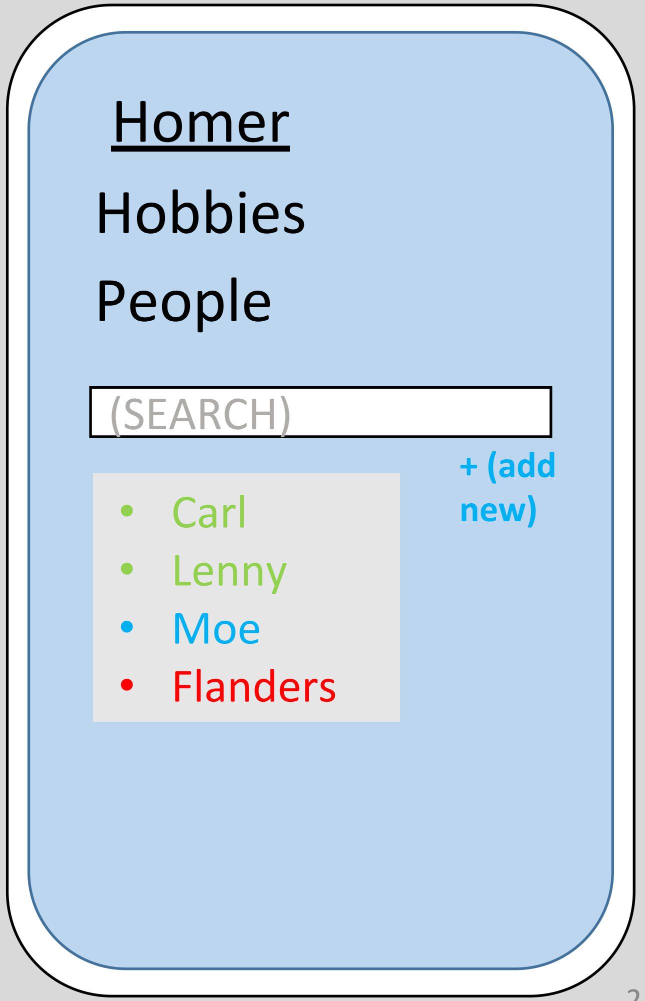
"Experience is not worth the getting. It's not a thing that happens pleasantly to a passive you--it's a wall that an active you runs up against."

– F. Scott Fitzgerald, The Beautiful and Damned

Our Initial Idea

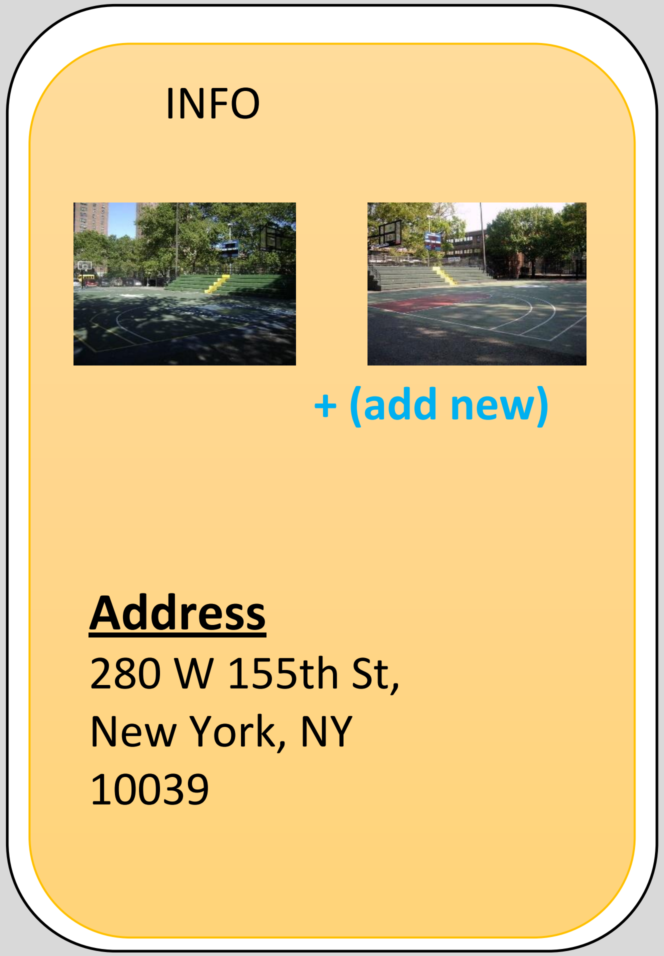
- An app that allows users to define their hobby spaces
- Allow them to check in and check out of these locations
- To view other users currently checked into these locations
- To ultimately define their world, their space, and their level of interaction with our app and the users on it





Defining the Space

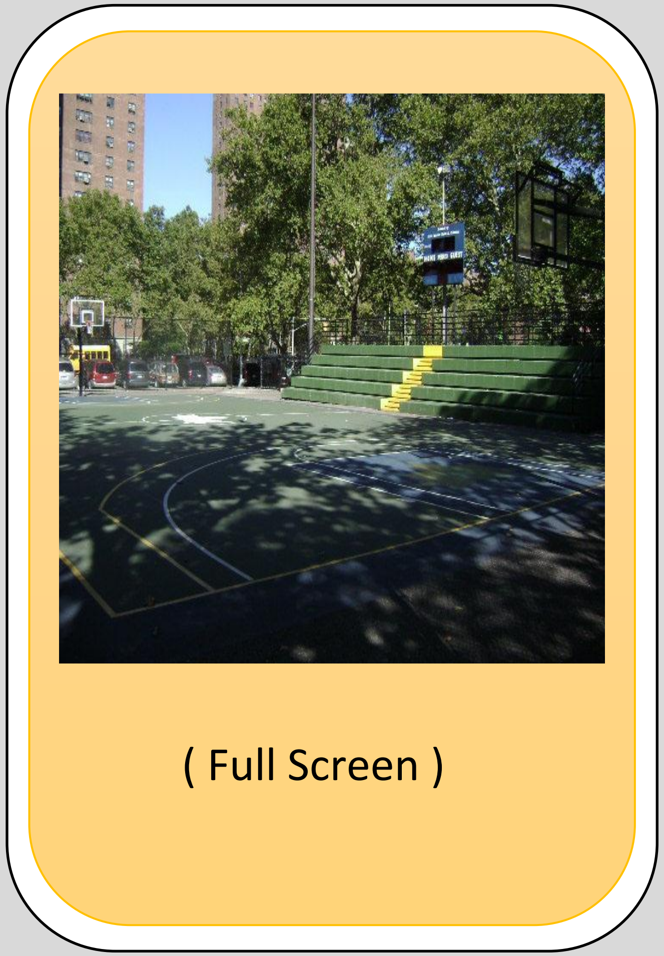
- Users define the space
- Seamless GPS integration to retieve a space's address
- Allows users to upload photos
- Other users can view the space and the various hobbies offered there



INFO

+ (add new)

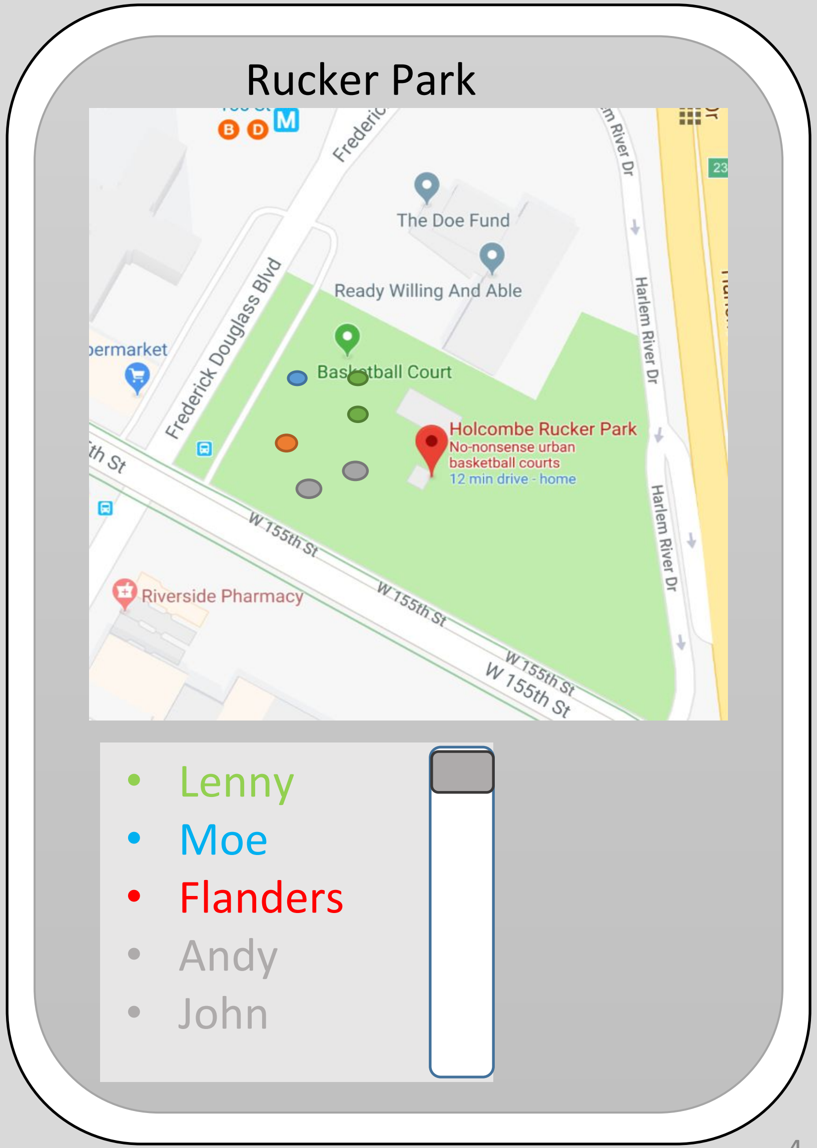
Address
280 W 155th St,
New York, NY
10039



(Full Screen)

Users at a Space

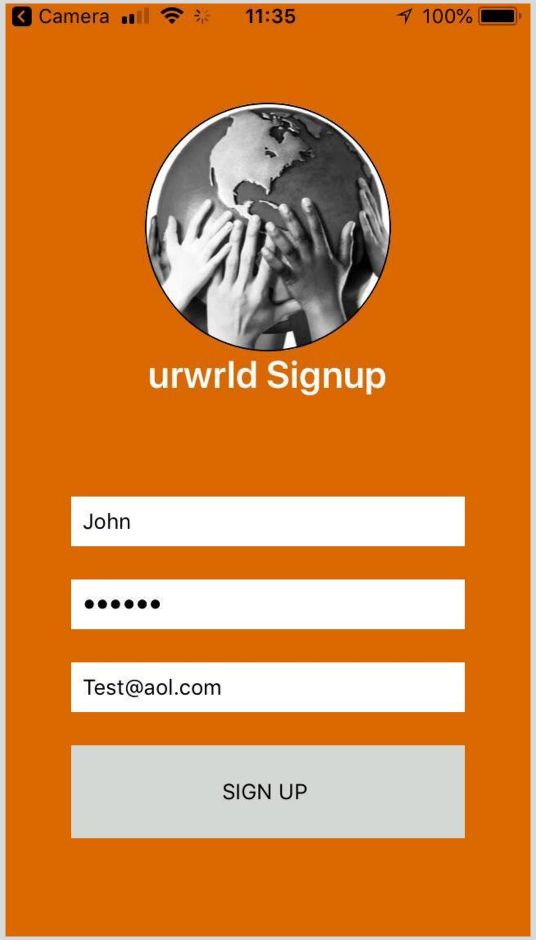
- If users are at a space and are checked in, they will be listed on a map
- We'd get their GPS coordinates and display them on a map of the space
- They choose their level of involvement
  - share what they wish to share with other users

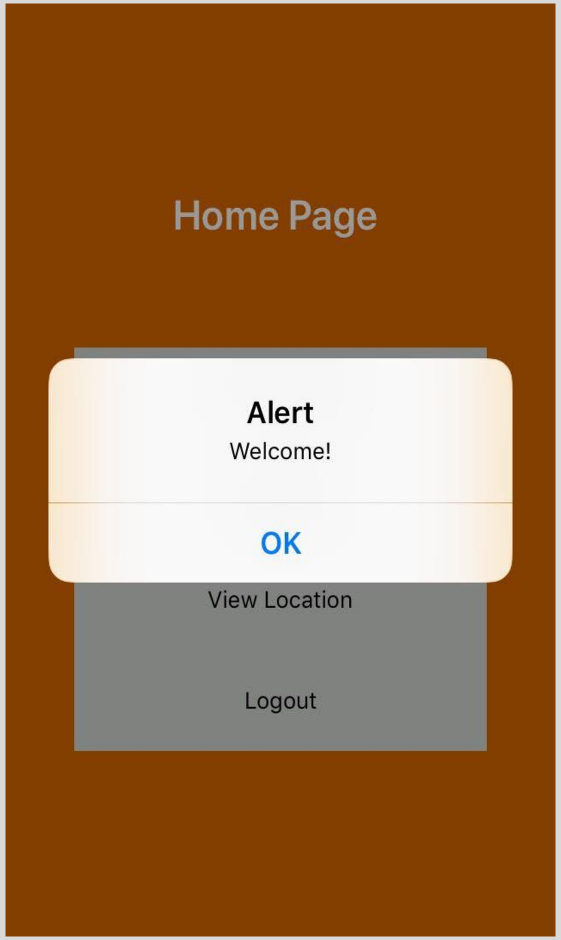

Rucker Park

- Lenny
- Moe
- Flanders
- Andy
- John

What We Achieved

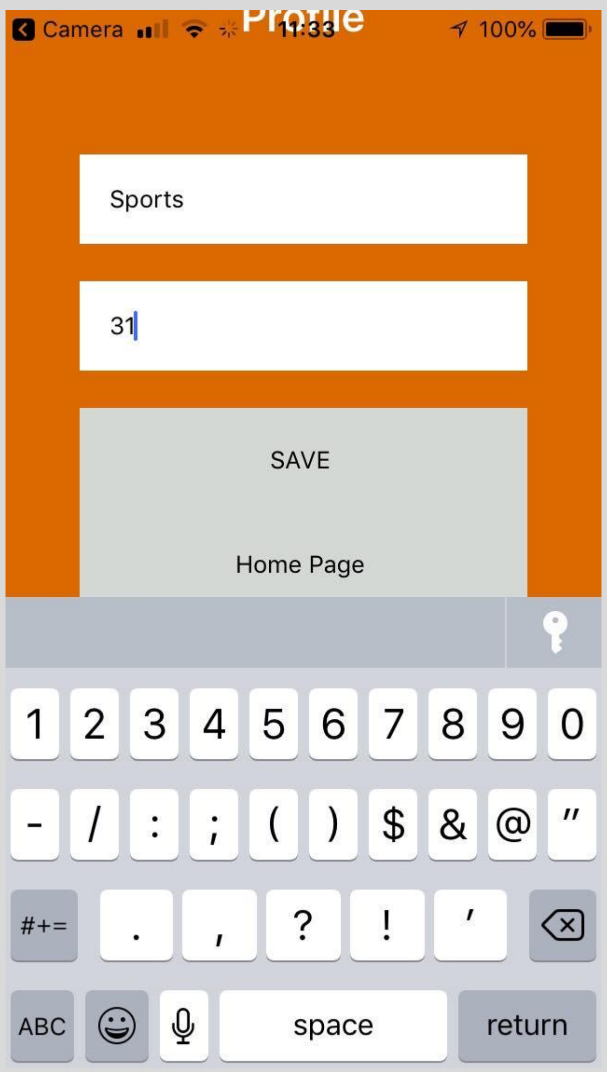
- Users can sign up and log in
- We have partially sanitized inputs
  - checks for duplicate user
  - checks for valid email
  - password must be 6 characters long, username must be between 4 and 10
- Deficiencies
  - Username not case sensitive ie. chris and Chris two different users
  - Email domains not confirmed ie. dum@dum.dum valid email

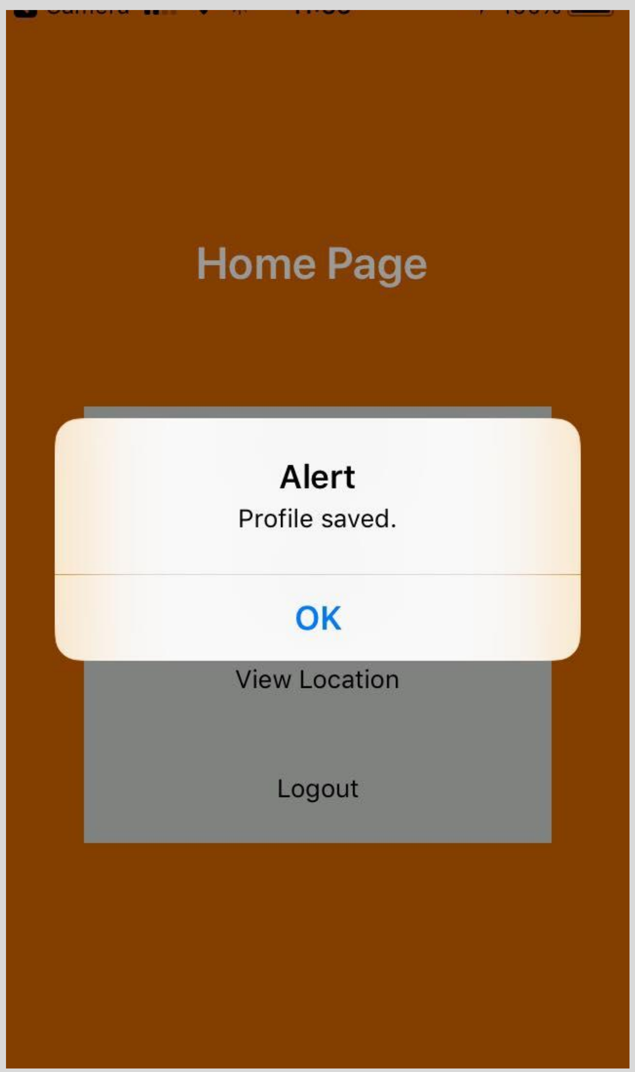




What We Achieved(cont'd)

- Users can edit a profile
  - Form with fields for hobby and age
  - Input is saved to our database
- Deficiencies
  - Inputs not currently sanitized
  - Old input will be overwritten if new input saved

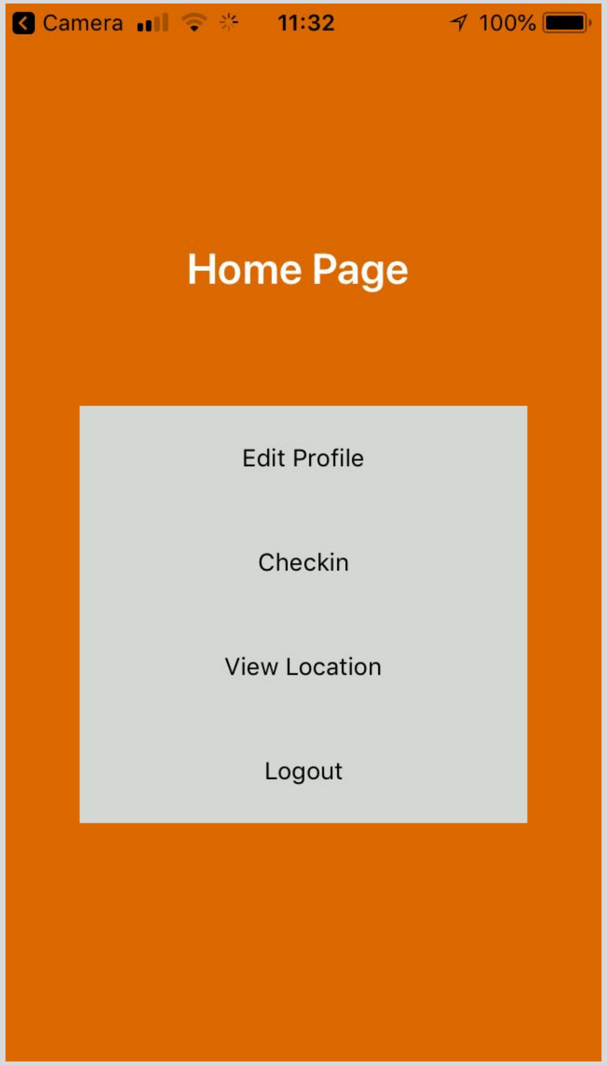


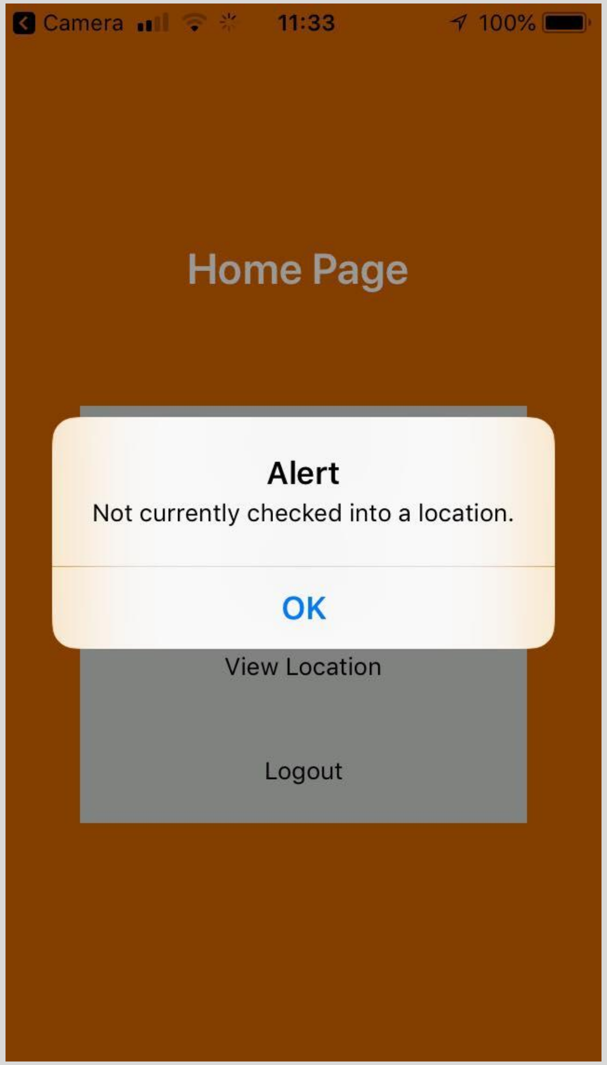


Alert
Profile saved.

What We Achieved(cont'd)

- Users can check in to a location
  - Retrieves a user's GPS and sends to backend to confirm a location with the GPS matches
  - Adds them to a specific location table in our database
- Once checked in they can view the location
  - Checks if currently checked in
- Deficiencies
  - GPS fixed to 3 decimal places, lacks precision
  - Logout does not automatically check out a user
  - Location must be added by us in the database manually



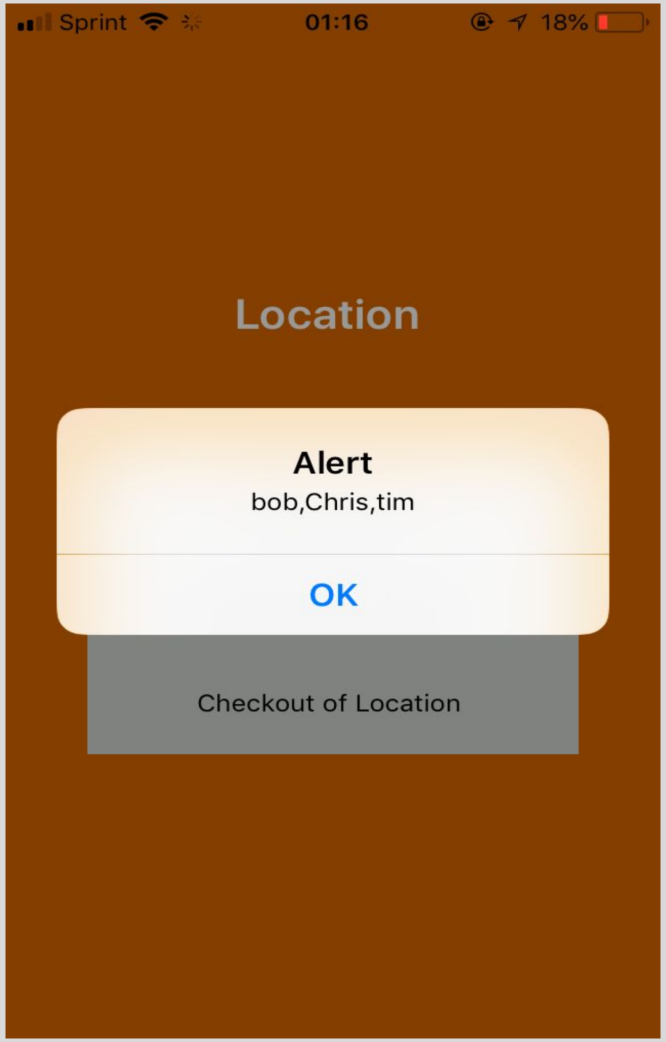


Alert

Not currently checked into a location.

What We Achieved(cont)

- If checked in, users can view their location
- Can only view users currently checked in
  - Retrieves array of users at a location from our database
- Deficiencies
  - Displays as a native alert and lacks formatting
  - Does not show user hobby
  - Does not show name of location or address



Our Road to Development

Tech We Used

React Native

Expo

mongoDB

MySQL™

WampServer

node JS®

Express JS

heroku

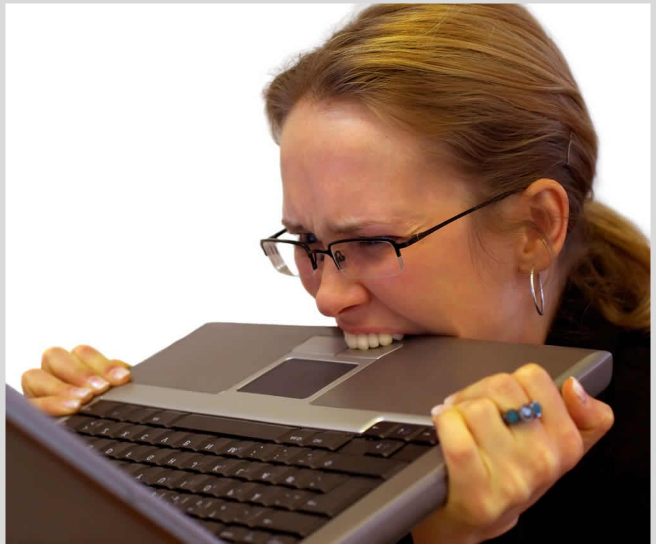
What went poorly?

- Use of Github for stories and issues and tasks
  - We all had limited exposure to github in a team-based capacity
  - Trouble integrating this into our project in a way that was efficient and organized and followed the AGILE SDLC
- Expo installation for React Native project development
  - Chris F. had problems getting Expo to work on his machine
- Geofencing libraries
  - Geofencing would allow us to confirm if a user was or wasn't in a location defined by a geographically confined location
  - The libraries for React Native are not official and lacked documentation as well as debugging errors
- Initially we used a WAMP local server for our database
  - This limited database accessibility to only part of the team and made testing our database difficult when we were separated
- Backend development stagnated for a bit
  - Node.js and Express.js were easy to set up but adhering and implenting a RESTful backend was difficult
- Communicating and being on the same page
  - We did not communicate well intially which made cohesiveness and smooth workflow hard to attain

What went well?

- React Native front end development
  - React Native itself was easy to set up and get started with
  - create react native app command gives you a solid foundation to build upon and allowed us to get a basic front end UI without much headache
- Building with Expo (for those that it worked for)
  - Expo built the react native app quickly and gave a convenient QR code to scan which allowed easy debugging and development with real time edits reflected on your mobile
- Backend server set up via Heroku
  - Once we had our API endpoints defined, hosting a database and our backend was easy with Heroku
  - Every push to our Heroku app repo deployed our updated backend

Our Story…



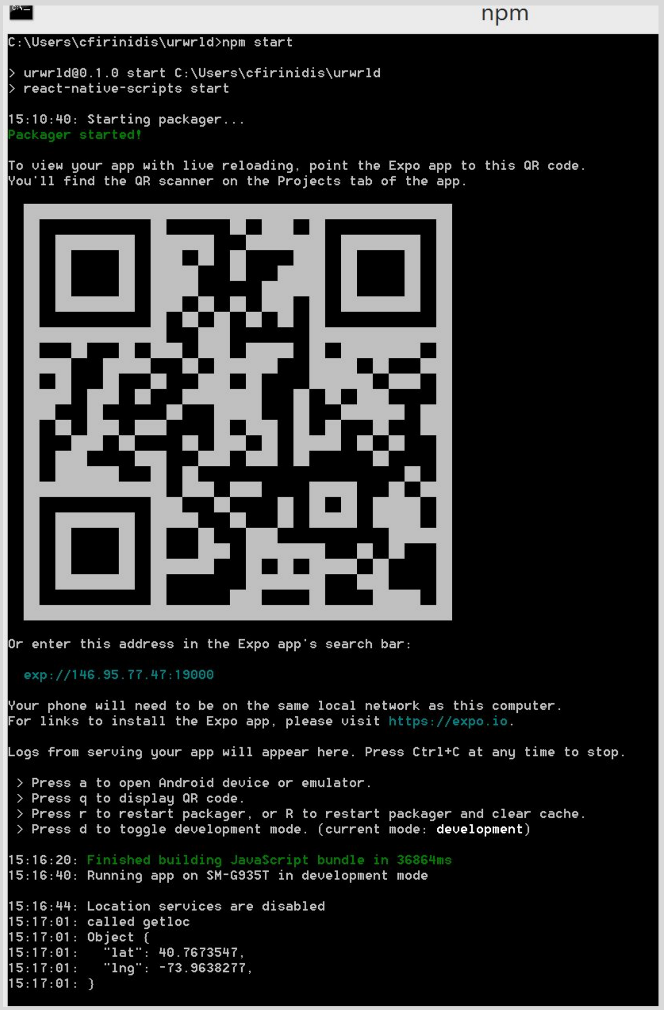
…of enduring frustration and adversity. Of overcoming miscommunication. Of persevering and achieving cohesiveness. Of overcoming together and reaching for team success.

Getting Started: Setting Up the Tools

- Our product was intended to be used as a mobile application and so we decided on React Native to target the two large mobile operating systems using native components
- Downloading and installing the framework was fairly easy for Chris H. on a Linux OS however Constantine and Chris F. had trouble getting them to work with Windows
- We initially chose Django for the backend due to easy setup and option of the Django Rest Framework making a REST api quickly
- We had problems communicating expectations however
    - Chris H. thought he was supposed to work on Django backend while Chris F. and Constantine worked on developing frontend but miscommunication prevailed and Chris developed a small backend while Constantine and Chris F. developed something more full stack with Node.js and Express.js
- We ultimately decided on Node and Express for the backend ease of setting up and maintaining a web server and to maintain a Javascript theme

Getting Started: Working with Expo

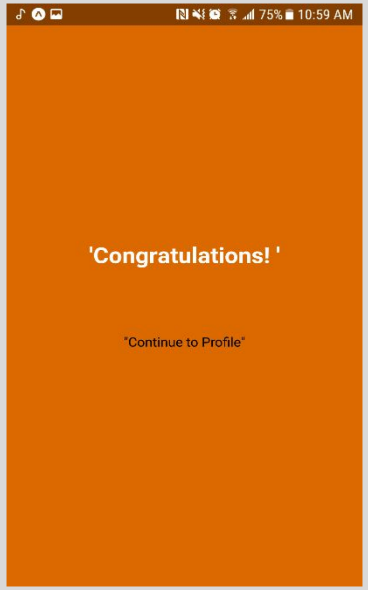
- We used Expo to build the React Native front end
- Required a host app to scan the QR code but allowed easier developing
- This worked for Constantine and Chris H. but Chris F. had some issues setting it up
- These stumbling blocks slowed down development and set us up with a stressful communication avenue



```
npm
C:\Users\cfirinidis\urwrld>npm start

> urwrld@0.1.0 start C:\Users\cfirinidis\urwrld
> react-native-scripts start

15:10:40: Starting packager...
Packager started!

To view your app with live reloading, point the Expo app to this QR code.
You'll find the QR scanner on the Projects tab of the app.

Or enter this address in the Expo app's search bar:

  exp://146.95.77.47:19000

Your phone will need to be on the same local network as this computer.
For links to install the Expo app, please visit https://expo.io.

Logs from serving your app will appear here. Press Ctrl+C at any time to stop.

 > Press a to open Android device or emulator.
 > Press q to display QR code.
 > Press r to restart packager, or R to restart packager and clear cache.
 > Press d to toggle development mode. (current mode: development)

15:16:20: Finished building JavaScript bundle in 36864ms
15:16:40: Running app on SM-G935T in development mode

15:16:44: Location services are disabled
15:17:01: called getloc
15:17:01: Object {
15:17:01:   "lat": 40.7673547,
15:17:01:   "lng": -73.9638277,
15:17:01: }
```

Expo

Cons/Chris.H

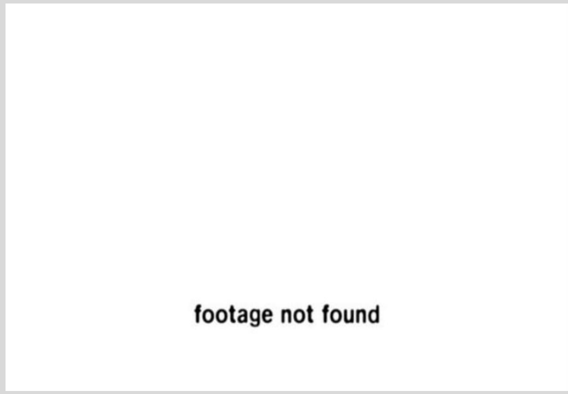


10:59 AM

'Congratulations! '

"Continue to Profile"

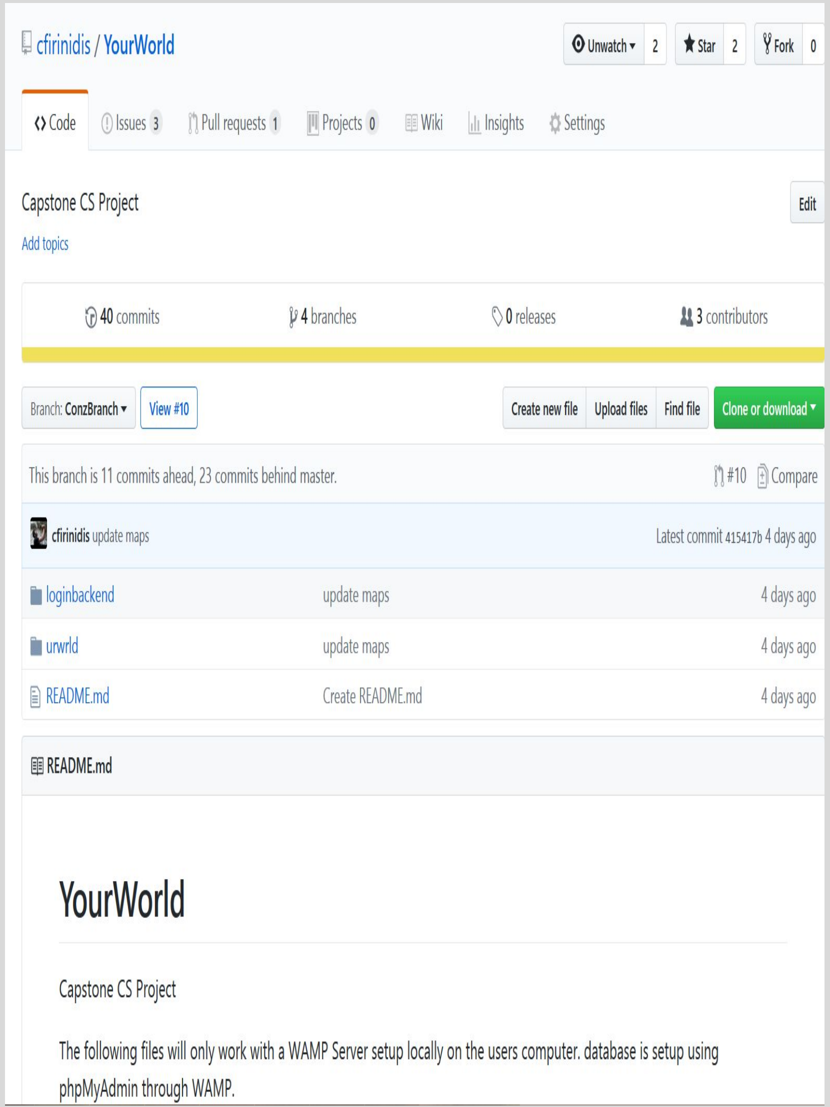
Chris.F



footage not found

Staying On *The* SAME ✓❧❧❧(*Page*)

- We encountered trouble using Github for tracking issues, stories, and tasks
  - Not all of us had used Github before, or command line tools which increased the learning curve for our team collectively
- Deciding on a database left us on a different page with Mongo vs WAMP
- Miscommunication set us back and caused more than one commit and merge over each other
- Once we settled into a groove later in the project it became more organized
  - Used Github to make sure each branch was only changing a certain file that the others were not
- There were still problems using issues and communicating through Github



Capstone CS Project

This branch is 11 commits ahead, 23 commits behind master.

cfirinidis update maps — Latest commit 415417b 4 days ago

loginbackend	update maps	4 days ago
urwrld	update maps	4 days ago
README.md	Create README.md	4 days ago

README.md

YourWorld

Capstone CS Project

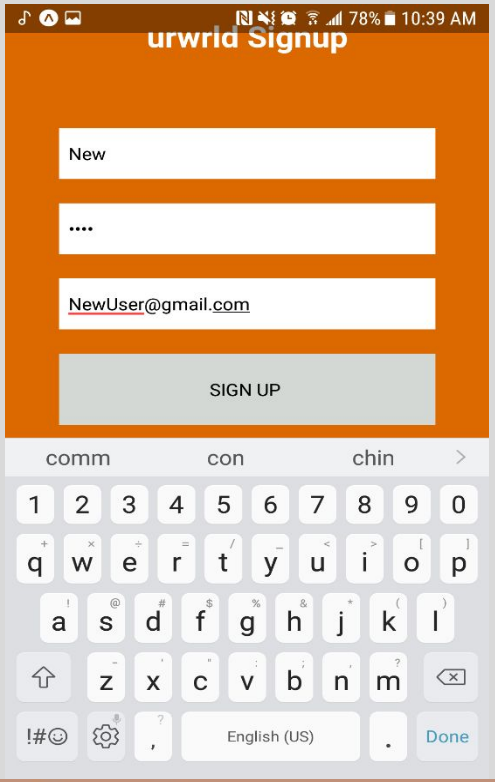
The following files will only work with a WAMP Server setup locally on the users computer. database is setup using phpMyAdmin through WAMP.

Chris.H → mongoDB

Chris.F /Con → WampServer

Success Keeps Us Going

- After our initial bumps we had a working front end UI
- It made calls to backend API endpoints which we could monitor via Nodemon
  - Showed us this was actually working
- This saved data in a local WAMP database which was good and bad
  - Good: we had something set up that worked
  - Bad: only worked on Constantine's local machine
- We needed to setup a wholly functioning backend



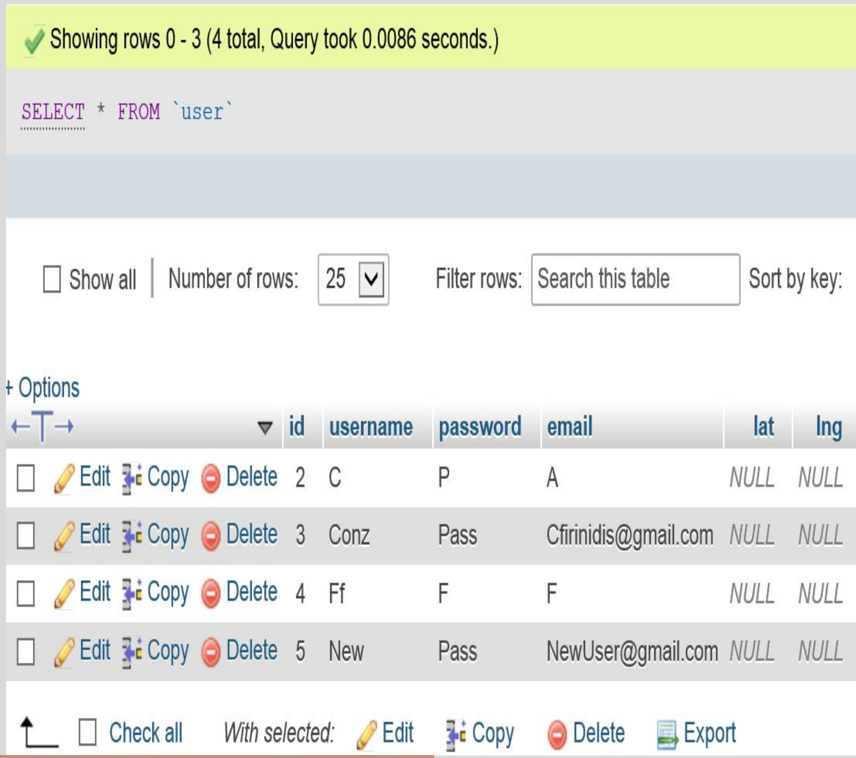
urwrid Signup

New

····

NewUser@gmail.com

SIGN UP

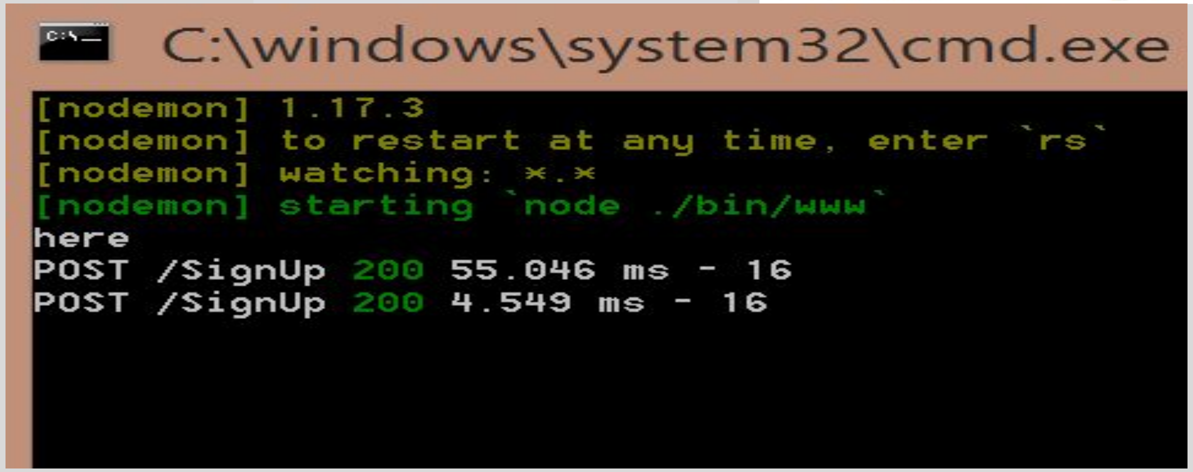

Showing rows 0 - 3 (4 total, Query took 0.0086 seconds.)

```
SELECT * FROM `user`
```

Show all
Number of rows: 25
Filter rows: Search this table
Sort by key:
+ Options

	id	username	password	email	lat	lng
Edit Copy Delete	2	C	P	A	NULL	NULL
Edit Copy Delete	3	Conz	Pass	Cfirinidis@gmail.com	NULL	NULL
Edit Copy Delete	4	Ff	F	F	NULL	NULL
Edit Copy Delete	5	New	Pass	NewUser@gmail.com	NULL	NULL

Check all
With selected: Edit Copy Delete Export



```
[nodemon] 1.17.3
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: *.*
[nodemon] starting `node ./bin/www`
here
POST /SignUp 200 55.046 ms - 16
POST /SignUp 200 4.549 ms - 16
```

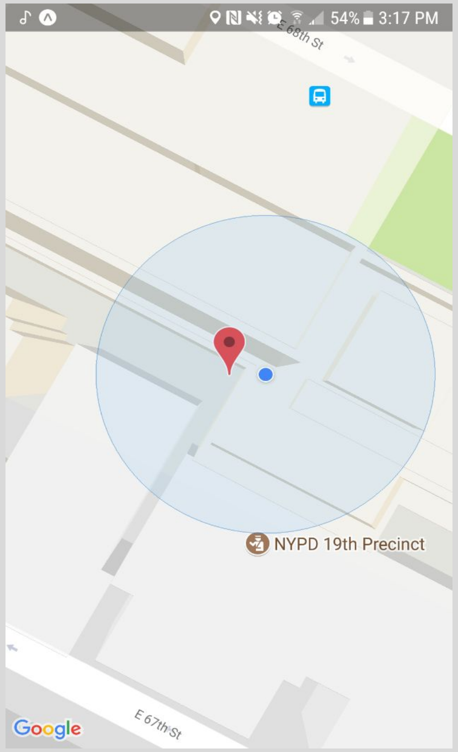
Figuring Things Out… Slowly

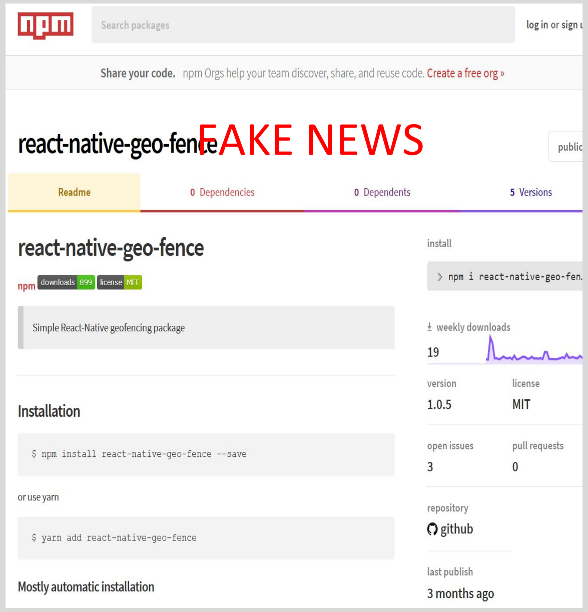
- We started communicating better we divided tasks more efficiently
  - Chris H. was developing a backend and with a suggestion from the App Nexus team started working on using Heroku to host the backend
    - Set up Heroku add-on database using Mysql through ClearDb
  - Constantine and Chris F. worked on getting a user's location using Google's API
    - The aim: to achieve a check in and check out system
- Commits in github were being made with organization and we were communicating our changes to the code base
- We were working cohesively and deliberating tasks effectively
  - There was still a learning curve to use our chosen tech stack but we had built up more team-based focus which would allow us to overcome tool-based stumbling blocks
- We were now able to sign up a user, log in a user, and edit a user's profile

Chris H

C & C

The Maps Dead-end: Or How We Learned to Stop Worrying About A Map and Love the User's GPS




Search packages
log in or sign u

Share your code. npm Orgs help your team discover, share, and reuse code. Create a free org »

react-native-geo-fence FAKE NEWS

public

Readme | 0 Dependencies | 0 Dependents | 5 Versions

react-native-geo-fence

npm downloads 899 license MIT

Simple React-Native geofencing package

Installation

```
$ npm install react-native-geo-fence --save
```

or use yarn

```
$ yarn add react-native-geo-fence
```

Mostly automatic installation

install

```
> npm i react-native-geo-fen
```

weekly downloads
19

version	license
1.0.5	MIT

open issues	pull requests
3	0

repository
github

last publish
3 months ago

- We thought we could use React Native libraries to establish a geofence however this was Fake News!
  - React Native Libraries aren't all maintained or documented
  - Lead to libraries not working when we tried to implement geofencing
- Went back to the drawing board: learned to love the user's GPS
  - We realized we did not need to show the user a map and only needed their GPS points
- Used users GPS coordinates to check them in and out of locations
  - Focused on developing this in the backend and passing the latitude and longitude to the server
  - Created a functional check in and check out system
    - Checking in finds a location in our database and adds the username to that table
    - Checking out finds the table and removes the row matching the username from our database

Possible Features for Future Development

- We’d like to add more details to a user’s profile
  - Append to hobbies in the database rather than overwrite on new input in the profile editing
    - More details like activity level or stats - like number of times visiting a location
  - A profile picture option
- Automatic check in and checkout system
  - This would make use of a geofence type tech that would refresh after a set amount of time to get a user’s new location
  - If they left a location or entered one this would automatically check them in
  - Reduces clutter of the UI by removing a check out and check in button
- Preferences a user would be able to activate or deactivate
  - Visibility to other users and blocked users
  - Enable/disable features like automatic checking in and checking out
- Uploading a picture to a location space
  - Possibly use geotagging photos to confirm it’s for the correct location
  - Defining a space with more detail with type of hobby or sports offered, and number of courts within a space
- Allow a user to check in with the current hobby they are participating in
  - Let other users know “UserXYZ” is playing basketball
- A friends list
  - Add friends to a list and get notifications when they check in nearby

What We Learned?

- Communication is an integral building block and paramount to success. Most of our stumbles were rooted here, from using Github to deciding our tech stack, and was compounded by the lack of exposure to the tools required for our project. When faced with a disagreement, sometimes its wisest to concede for the sake of efficiency and development.

- One's expectation is often not equivalent to the eventual reality. While similar to our initially proposed idea, what we produced isn't as full-fledged as we intended. We learned the time it takes to grow an idea into something we can develop and flower into a final product is hard to gauge and it's better to be realistic. After paring down our planned product features, we were able to achieve something more manageable and attainable within a reasonable timeframe.

- We changed course a few times with respect to the dev-tools we used and would have saved time if we were more decisive in the early stages. Researching and agreeing on a tech stack as early as possible is better than having to adapt later on but this requires a clear vision which may not always be realistic. For example our adoption of Heroku as a platform as a service alleviated the struggle of hosting data locally. however we'd not have known about it until we encountered the issue of remotely hosting our backend.

- We learned how to adhere to RESTful design and glue a front-end with a back-end which better allowed us to manipulate data in a database. We discovered how to make a request from the front end and fetch a API endpoint on our backend to successfully return the necessary data. We learned how to store and retrieve data from a mysql database without using a local machine.

- None of us had used much javascript prior to this course, yet we utilized React-Native, Node.js, Express.js and Heroku in a way that worked together to create a functional application.