Visitor Cell Phone Application: An Innovative Design to Monitor Visitor Mobility in Acadia National Park

An Interactive Qualifying Project Report

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

Degree of Bachelor of Science

by:

Joseph P. Caltabiano

Jack L. Charbonneau

Mikayla A. Fischler

Gillian T. Nadeau

Advised by:

Frederick Bianchi

Date: August 2018

Table of Contents

I. Abstract

The goal of this project was to demonstrate the feasibility of using a mobile application to collect data about vehicle position and movement in Acadia National Park.

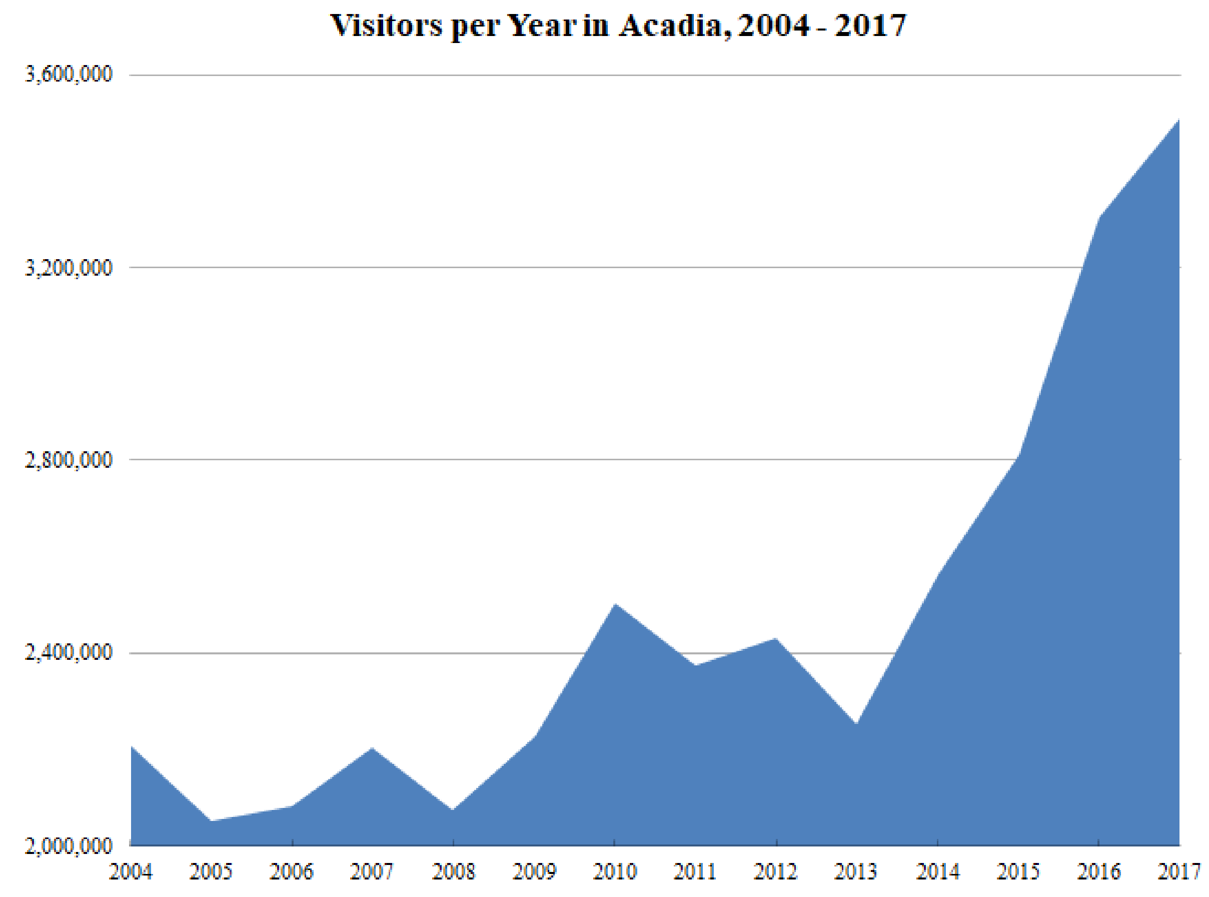
The team developed a mobile application for the Android cell phone. This was done to show a proof of concept for a more effective way to collect data on vehicular traffic throughout the park. The data that was collected and stored by our application contained GPS location, time, and an anonymous user ID. Our project also determined the willingness of volunteers to download a mobile application that would collect data for future research study for the national park. The feasibility of the mobile application was determined and suggestions were made regarding its future implementation. The application was distributed to visitors in the park in order to demonstrate the efficacy of the data collected. To analyze and convey this data, various tools were used including ArcGIS and Google Maps.

II. Acknowledgments

The research team would like to acknowledge and thank Professors Frederick Bianchi and Paula Quinn for their facilitation of the Interactive Qualifying Project. We would also like to thank Friends of Acadia for their assistance with promotion of the mobile application, as well as the park rangers for their collaboration with us to effectively carry out our project. The patience, persistence, and willingness of these people has ensured the success of this project.

III. Executive Summary

Acadia National Park is home to some of the most magnificent views in the United States, resulting in the park hosting the eighth highest number of yearly visitors. Ranked 47 out of the 58 national parks in terms of size, Acadia struggles under the high density of people passing through. Road traffic has become a severe problem. Across the nation, other parks are experiencing congestion as visitation numbers become difficult to manage.



(Figure 1: Visitors per Year in Acadia)

Acadia's size, relative lack of personnel, and tight budget makes it difficult for officials to mitigate traffic. As of the time of writing, the park's traffic monitoring capabilities are lacking. Although others have tried to create methods to collect traffic data and reduce the overall congestion, the style and scale of the data falls short. A previous team from WPI set up car counters at key locations across the roads of Acadia that could count the number of vehicles that drive over them. Acadia added more in recent years. However, there are only fifteen car-counting devices monitor the entire 49,052 acre park. Other national parks have also attempted to use handheld GPS units. These programs can result in extremely useful data, but the units were mostly handed out to hikers, not cars. Additionally, the amount of active GPS units a park can hand out is limited severely by the unit cost and the risk of theft. Locals, park rangers, and locals are becoming increasingly concerned as daily travel, ranger duties, and enjoyment are being hindered.

To further address this issue, Acadia National Park has released a Draft Transportation Plan to the public. Many of the suggested resolutions involve implementing a reservation-based system to control visitation numbers. The park could make more informed management decisions about road congestion if they were able to monitor current traffic and observe changes in the patterns after reservations are introduced. Our team has provided an innovative solution to gathering this location data by developing a mobile application.

The goal of this project was to demonstrate the feasibility of using this mobile application to collect data about vehicle position and movement in Acadia National Park. The data collected from such an application is instrumental in accomplishing the larger goal of protecting Acadia’s resources by reducing traffic congestion throughout the park. In order to complete this goal, three main objectives were accomplished: (1) evaluation of past attempts to monitor and manage park congestion, (2) development and distribution of a visitor tracking application, and (3) demonstration of the value of data collected by the application.

This application was developed using the React Native framework and distributed to visitors with Android phones (React Native). It collects a GPS point and sends the point, along with the time of collection and the application’s assigned unique user ID, to a server and stores the data in a database. The application displays a user’s GPS location, allows them to turn off and on the tracking feature, and delete all their collected data if they ever feel uncomfortable or concerned about privacy. Whether they are time-, seasonal-, or weather-based, countless extrapolation can be made from this data to show park officials the more nuanced patterns of movement through the park.

Data collected by the application is eventually sent to a server. The park has inconsistent data availability, and the app uses different systems to work around this. If there is no signal at all, the phone will continue to collect data and store it locally in the phone until a signal is found. If the phone is just on cell data, the application will store up 25 data points before sending them all at once to reduce data usage. If a user is on wifi, the app will send data points to the server as they are collected. The server is a MariaDB instance, and the data is stored in an SQL database.

A website was also built at <barharboriqp.ky8.io>. Users on the website are able to access the entire set of collected data, separated by UUID, as well as read other metrics about the application and server health.

Setting up a table the Hulls Cove Visitor Center, the team recruited willing visitors to download the application on their phones. Over a period of three days, the application was installed over 30 times.

Our team has also researched the willingness of visitors to download an application whose sole purpose was to track their movements through the park with the intent of alleviating traffic in the future. This survey was handed out at Hulls Cove Visitor Center as well. We found that 103 of 137 visitors surveyed were interested in using such an application to provide data to park rangers.

After collecting location data from our application, we saw that this style of data does provide powerful visualizations of complex and possibly counterintuitive traffic patterns. It also gives the opportunity for countless interpretations. Large amounts of metrics can be pulled from a dataset of this style and can be greatly customized by time and specific location. The data can also be cross-referenced with other datasets such as historical weather patterns, holiday occurrences, cruise ship dockings, and more.

The team recommends that Acadia either incorporates this feature into an existing application, or build a new Acadia official application from the ground up. Apps such as Acadia by Chimani are widely used through the park, and it may be possible for Acadia to collaborate with Chimani to add this feature. Alternatively, Acadia could hire a third-party development team to customize their own specific app. The company OnCell provides a service that allows an organization to drag-and-drop modules and features for a customizable development experience.

OnCell has built apps for museums and other tourist attractions as well as some national forests. The national forests reported that they were able to view useful location data about their visitors.

IV. Authorship

Abstract: Gillian Nadeau
Executive Summary: Joe Caltabiano, Gillian Nadeau
Introduction: Jack Charbonneau, Gillian Nadeau, Joe Caltabiano
Background: Gillian Nadeau, Jack Charbonneau, Mikayla Fischler, Joe Caltabiano
Goals: Jack Charbonneau
Methods: Gillian Nadeau, Jack Charbonneau, Mikayla Fischler, Joe Caltabiano
Analysis: Jack Charbonneau, Joe Caltabiano
User Manual: Mikayla Fischler, Jack Charbonneau, Gillian Nadeau
Conclusions and Recommendations: Gillian Nadeau, Jack Charbonneau

V. Table of Figures

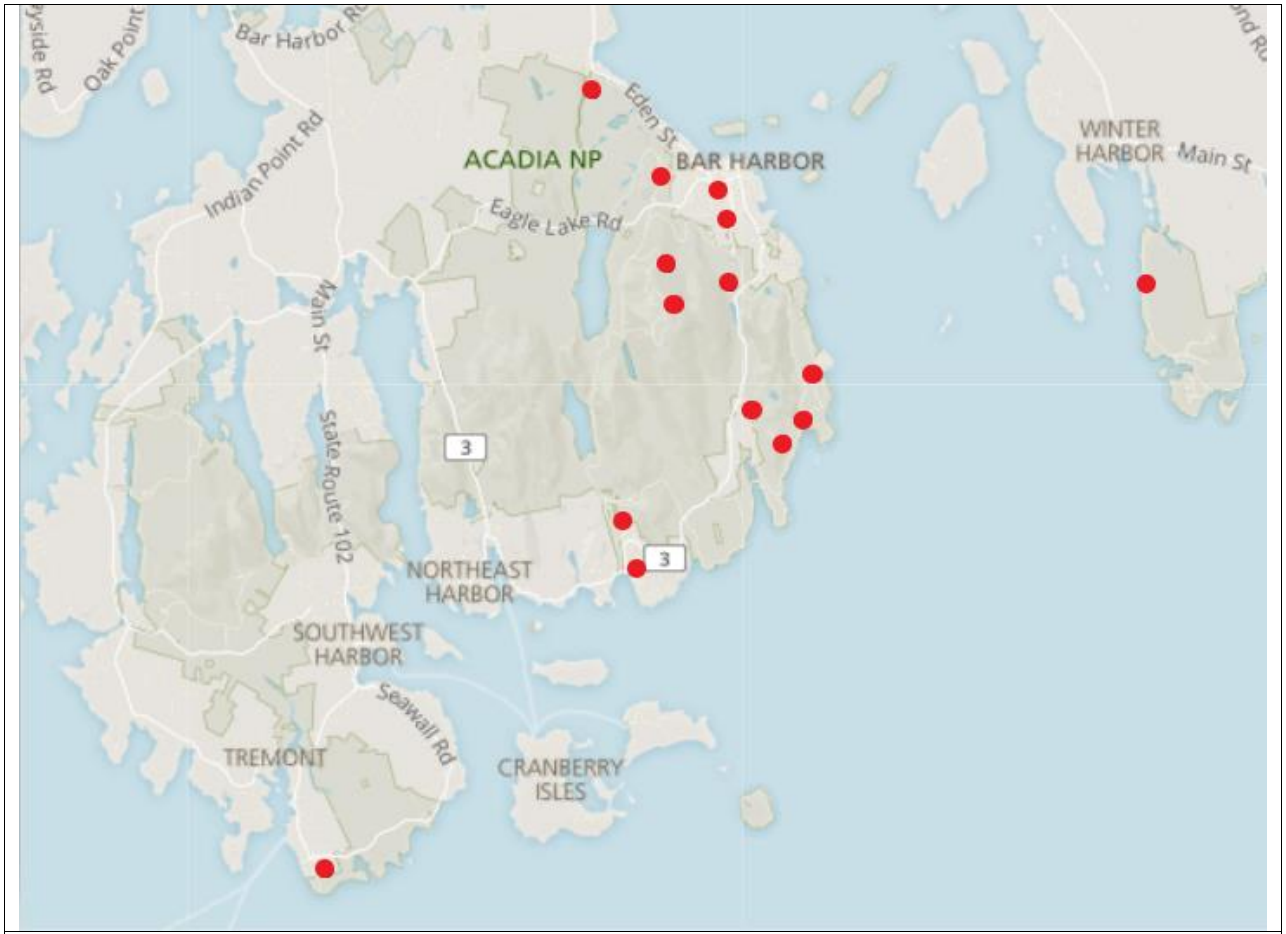
1. Introduction

Over the last several decades, road congestion has become an increasing concern in many of America's national parks. High visitation rates have begun to affect the experiences of locals and visitors, as well as the duties of park rangers. Maintaining the natural beauty of these parks is a core objective of the National Park Service, and unfortunately, this issue has put the parks' ecological security at risk.

In 2014, the U.S. National Park Service published a seven-step Congestion Management System/Process as part of its Congestion Management Toolkit (United States Department of the Interior, 2014). Step two of this process, "determine the location(s), frequency, and impacts of congestion," cannot properly be carried out because of the inadequate amount of data that has been collected. Collecting data on traffic through the park is necessary for park officials to fully understand and address traffic related issues.

There have been several attempts by various parks to gather data about traffic patterns. Solutions like car counters and handheld GPS units have only been able to supply limited data. This is largely in part to their relative inaccuracy and limited deployments. To effectively manage traffic, park officials require more robust and complete datasets.

In Acadia National Park, traffic data has traditionally been collected through the use of traffic counters. These counters merely keep track of the number of cars that pass by at a given location in a period of time. There are many limitations associated with the data collected by traffic counters. For example, they are unable to show the course of a single car, only showing the fact that a car has passed over it. The quantity and utility of metrics that can be pulled from this data is limited. The low number of counters also poses an issue with the accuracy of the data, and can be a poor representation of actual patterns. According to Stephanie Clement, the Conservation Director at Friends of Acadia, there are only fifteen traffic counters in the entire 49,000 acre park.



(Figure 2: Location of traffic counters)

The goal of this project is to contribute to Acadia National Park's traffic congestion management by demonstrating the feasibility of collecting visitor location data through the use of a mobile application. Since most modern smartphones incorporate GPS receivers, a mobile application can be developed to collect GPS data without encountering the setbacks experienced previously with individual GPS units. The issue of cost associated with a mobile application is minimal when compared to previous solutions since there is no need of additional physical tracking devices. The application will not cost more than its development and periodic maintenance. Data will be uploaded automatically from each phone, eliminating the issues of collecting physical devices to download data. The issue of a user having to know how to operate a GPS unit is avoided since participants of this study will be using devices that they own and likely are relatively familiar with. Regardless of the familiarity, the use of the app itself can be heavily simplified to significantly reduce difficulty understanding its operation. The team seeks to capitalize on the quality and depth of data collected through GPS tracking methods while also mitigating the limitations associated with physical GPS units.

Recently, Acadia released a Draft Transportation Plan, which has been in development for roughly four years (National Park Service, 2018). Although still a draft, the plan elaborates on the use of reservation systems within the park. Implementing a system like this may bring problems with congestion migrating to other parts of the park. Our mobile application will help the park monitor the traffic flow after the reservation system is implemented.

By demonstrating the feasibility of a traffic monitoring application, this project seeks to provide Acadia National Park with a highly efficient means of collecting data that is needed to make informed decisions regarding the park's traffic issues. Our long term goal is to show the vehicular traffic patterns throughout the park as well as to give the park the ability to extrapolate from the data according to its specific management needs.

2. Background

The natural beauty of Acadia National Park has made it an astonishing visitor destination since its foundation in 1916. As more people began exploring this hidden gem, the visitation rates increased dramatically, bringing along a spike in vehicular traffic. Traffic congestion throughout the park has become an increased concern to visitors and park rangers alike. This congestion is primarily a problem on the park's main road, Park Loop Road, and at popular tourist attractions such as Cadillac Mountain and Sand Beach. Congestion, however, is not limited to these areas as it also occurs in simple trail parking lots and at scenic viewpoints throughout the park. These areas have heavy visitor traffic which affects the accessibility across the park, preventing visitors from enjoying the park to its full extent. The road to the summit of Cadillac Mountain, for example, was closed a dozen times throughout the summer of 2017, seven of those times being on Labor Day weekend alone (AOMM Enterprises, 2017). Congestion is often so bad that parking lots exceed their capacity, forcing visitors to resort to parking at social pull-offs or the side of the road in the areas surrounding popular attractions. Parking on the roadside can prove to be a safety hazard, since it forces pedestrians to stand and walk in close proximity to moving vehicles. In addition, parking in these areas lead to further congestion as traffic flow is frequently interrupted when visitors come in and out of these makeshift roadside "parking spots". Roadside parking also poses an ecological threat. It can kill vegetation and destroy potential habitats over time from the damage caused by heavy vehicles and off-roading. The park usability and ecological concerns are not likely to correct themselves unless a significant effort is made to address their root cause: mismanaged traffic congestion.

As shown in *Figure 3,* visitor population at Acadia National Park has been increasing over the past ten years. It can assumed that this trend will only continue in the future. As well as causing severe traffic, this influx can cause sound and light pollution across the island. It is critical to the health of the park and enjoyment of future visitors for these effects to be mitigated.

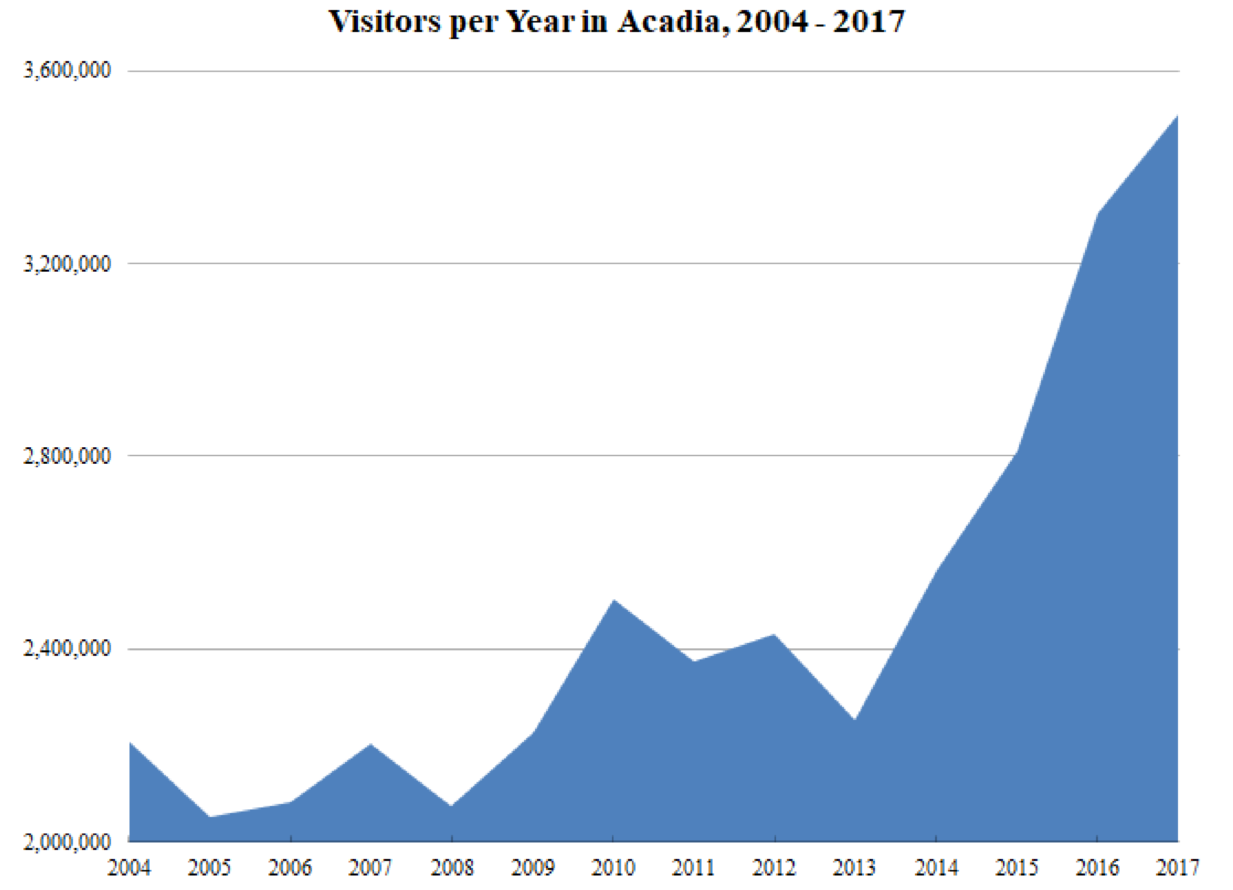


Figure 3: Graph of yearly visitors from 2004 through 2017 in Acadia National Park. Source: NPS Stats

The problem of excessive tourist congestion is not exclusive to Acadia. Across the country, traffic in national parks has become a widespread issues due to the ever increasing tourist populations. In many cases, the roads and parking areas provided have become insufficient under the strain of contemporary traffic flow, and much like Acadia, national parks all across the country have seen a continued increase in visitor attendance over recent years. Yellowstone National Park, for example, has seen consistent growth throughout its history. During the 1990s, Yellowstone saw an average of about 2.8 million tourists per year. However, in the last few years, that number has risen closer to four million. Likewise, Yosemite National

Park and Glacier National Park have each seen a rapid increase in annual visitor population in recent years (NPS Stats).

Visitor population does not stay constant throughout the year, instead varying greatly by month. This continues to exacerbate congestion issues due to a large portion of the annual population visiting in only a few months. In Yosemite National Park, traffic changes with seasons, weather, and holidays. As shown in *Figure 4*, the park is relatively vacant during the winter, but visitation spikes drastically starting in May, continues to rise to a peak in July, the declines until November, at which point the park is relatively vacant until the cycle begins again.

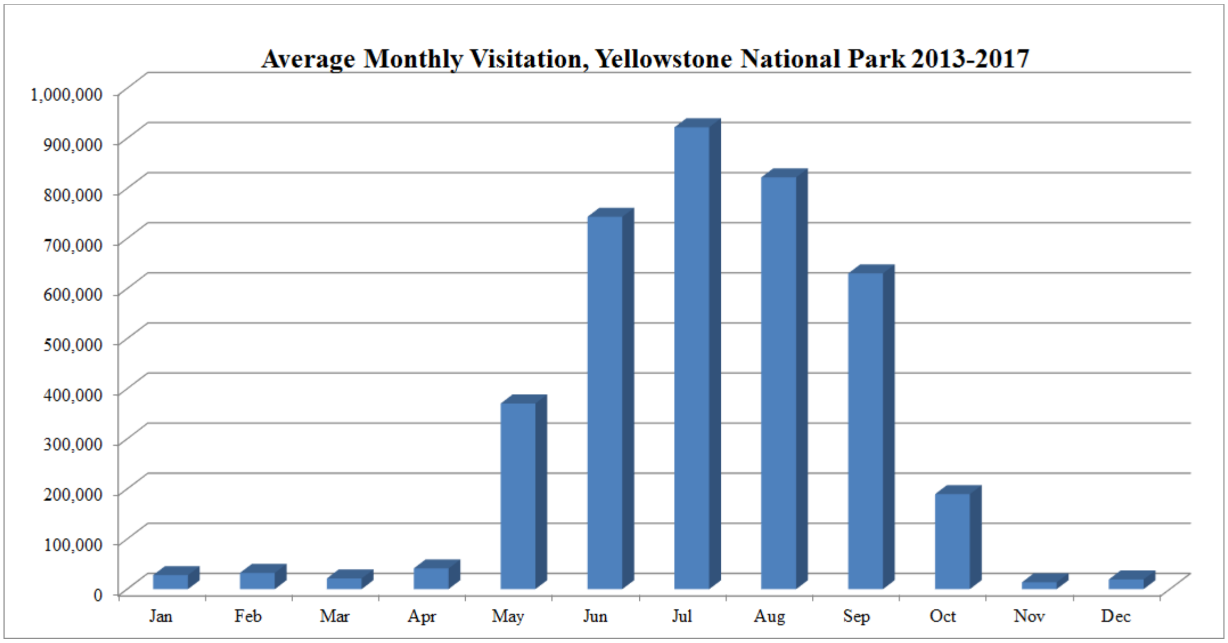


Figure 4: Average visitation by month to Yellowstone National Park from 2013 to 2017. Source: National Park Service, 2018b

One example of how the traffic congestion degrades the experience of the national park guests is how the entrance station delays in Yosemite can range from one to three hours during times of peak visitation (Traffic in Yosemite, 2018). In addition to the traveling time taken to arrive, most visitors must face an additional wait before entering. This problem extends to other parks such as Zion National Park. In comparison to Yellowstone, Zion receives the same number of annual visitors, but is only one fifteenth of the size. Adding to the difficulty, national parks are proposed a 16% budget cut to the Department of the Interior for the fiscal year 2019 (United States Office of Management and Budget, 2018). Underfunding and understaffing of national parks makes it difficult to implement effective solutions. Without an increase of funding these problems are likely to get worse. In 2004, an independent research team from the University of Vermont conducted a traffic study by administering handheld GPS devices to tourists at Acadia and other national parks. Researchers were pleased with the quality of the data collected, but noted that their methodology still had significant limitations. Specifically, researchers stated that cost, technical complexity, and collection of GPS units after use were issues of concern (Hallo et al., 2004).

Since the traffic issues Acadia is experiencing are not unique, it is possible to evaluate the actions that other parks have taken to try and mitigate their traffic issues. Examining the steps other parks have taken to monitor and manage their traffic can provide useful insight on what solutions are most successful, and if they could be tailored to fit Acadia's specific needs. Ideas to have "car free weekends" have been attempted at Acadia and expressed in other national parks, but many times these solutions offer only a temporary fix to the issue. Other ideas that have been tried in place such as Arches National Park include an electric shuttle at no additional fee beyond admissions, an electric Jeep for a moderate additional fee (self-driving longer term), and keeping your private car for a premium additional fee (Kurt Repanshek, 2018). Glacier National Park has implemented similar initiations with the use of a free shuttle travel option (Erin Madison, 2013).

Current traffic flow conditions in Acadia National Park are not ideal, and this is of increasing concern to the park rangers trying to run and maintain the park. Additionally, locals and frequent park visitors are concerned by the air and noise pollution that results from high traffic levels. Likewise, organizations affiliated with the park who are invested in the longevity and health of the park, such as the Friends of Acadia, are concerned with what long term impacts traffic congestion may have if it is not properly managed. This is especially concerning in popular park locations such as Park Loop Road. In addition to its expected crowding, the park is currently undergoing construction on different sections of its car routes, which will surely keep the park at a constant state of congestion (National Parks Service, 2018).

Of all U.S. national parks, Acadia is ranked 47th in terms of size, but 8th in annual visitor population. The large amount of visitation is especially concerning for smaller parks, and specifically, vehicular traffic is the primary concern. Statistically, at Acadia National Park there are approximately three million visitors annually with over 90% driving their cars into the park. Park Loop Road in Acadia is victim to frequent excess traffic, and it is well known that excess traffic causes environmental damage. This results in high-traffic areas being more prone to damage. Finding a solution that would more effectively distribute traffic and minimize uneven concentrations of tourists would benefit the health of the park and the visitor experience. Visitor satisfaction is negatively affected when visitors see many more people than expected in a certain area, which is of common occurrence due to concentrated regions of traffic. (Calvi, Maki, Peters, Shuai, & Wivagg, 2017).

Previous attempts have been made to monitor vehicular movement throughout Acadia National Park in order to help alleviate traffic congestion. In the summer of 2017, a student team from Worcester Polytechnic Institute (WPI) researched the feasibility of implementing a parking reservation system at the top of Cadillac Mountain (Cosmopulos, Gaulin, Jauris, Morisseau, & Quevillon, 2017). They surveyed visitors to collect data on their opinions of the plan, investigated several different methods of implementing the reservations, and determined the cost-efficiency. The team's final solution, an online reservation system enforced by gated lanes, would effectively reduce wait times, overflow, and congestion at the top of Cadillac Mountain. However, this plan would cost over a quarter of a million dollars, and would involve road expansion and the removal of some natural resources. It also would only take reservations for the Cadillac Mountain lot, and would not address congestion throughout the rest of the park, or on the Park Loop Road. During the same time, another research group from WPI investigated the possibility of integrating an intelligent transport system into the park's existing infrastructure. Full-scale implementation would involve sensors at park entrances to count visitors, cameras to monitor parking availability in lots, and a visitor information phone line (Calvi, Maki, Peters, Shuai, & Wivagg, 2017).

While previous traffic monitoring attempts in Acadia have succeeded in quantifying the number of vehicles that pass through an area at a given time, they have failed to provide data with meaningful continuity; the traffic counters used by past IQP teams are effective at counting cars at a single location, but they cannot provide information about a vehicle's movement before or after having passed the counter. To make meaningful conclusions about dynamic traffic patterns based on this static data collected from traffic counters requires far too much extrapolation and inference to provide an appropriate level of reliability and confidence in the results. Therefore, a successful traffic monitoring solution must incorporate a system that facilitates the continuous collection of location data about each unique vehicle that is being monitored. By integrating continuous data collection, researchers can form more reliable conclusions about the broad traffic patterns that exist within the park as they will know when and where visitors go in the park. This IQP is a continuation of the research done previously, but with an added feasibility element and a more accurate way to observe traffic patterns and collect data through the creation of a mobile application.

Continuous data about moving entities is most commonly generated via GPS (Global Positioning System) measurements over time. Historically speaking, GPS has often been implemented using specially designed single-purpose devices. These devices, which are often handheld, rely upon their ability to communicate with GPS satellites in order to accurately calculate position. Though these GPS devices can be useful and are very specialized, they present many difficulties with regards to implementation as noted by previous researchers (Hallo et al., 2004). The first and most pronounced issue is that of cost. Even the cheapest and lowest quality GPS devices are priced around $50 per unit, and at any given time, a research team would need to possess at least one GPS unit for each individual participating in data collection. This limits the amount of data that is collected and can skew the results of patterns developed through the use of a small amount of units. In the long term, GPS device implementation is not feasible since the cost will continue to rise in linear proportion to the number of study participants.

There are several other problems that arise when implementing the handheld GPS method. In addition to the cost, handheld GPS units would also require the visitors to receive some degree of instruction about how to properly use the device. When considering the wide variety of handheld GPS devices on the market, it is unlikely that many research participants would be familiar with the specific model that is entrusted to them. Furthermore, leaving participants in possession of a physical unit requires us to rely on the visitors to return the devices before leaving Acadia National Park. Failure to return to the GPS device would mean that the data gathered would be lost and a cost would be incurred to replace the missing equipment. Despite making initial setup simple, single-purpose GPS devices provide substantial difficulties when considering widespread implementation.

Rapid advances in technology over the course of the last decade have led to a more widespread implementation of GPS capabilities in a variety of devices; the most notable of these is the smartphone. As of January 2018, some 77% of adults in the United States own a smartphone, the vast majority of which are capable of location tracking via GPS (Pew Research Center, n.d.). Rather than purchasing a standalone device for location tracking, GPS capability has been built into a device that most Americans already own.

With these difficulties and opportunities in mind, our team has set out to collect continuous data by leveraging the GPS capabilities of smartphones and other mobile devices. Our team has developed a mobile application for Android and iOS that utilizes the GPS capabilities of these devices to determine the location of visitors within Acadia National Park. Once GPS data has been collected, the devices’ networking capabilities can then be used to send that data to a server where it can be stored for future observation and analysis. This mobile application approach removes the overhead cost associated with the previously mentioned single-purpose GPS device method and greatly reduces the need to educate visitors on how to properly use the device. An additional positive result of eliminating the need to purchase special GPS devices is that data collection will be limited only by the willingness of visitors to participate, not by the number of units in our possession. Utilizing a GPS-based mobile application will allow us to accumulate a substantial amount of continuous traffic data while still avoiding the drawbacks associated with stand-alone GPS tracking devices.

3. Project Goals

The goal of this project was to demonstrate the feasibility of using a mobile application to collect data about vehicle position and movement in Acadia National Park. The data collected from such an application is instrumental in accomplishing the larger goal of protecting Acadia's resources by reducing traffic congestion throughout the park. In order to complete this goal, three main objectives were accomplished: (1) evaluation of past attempts to monitor and manage park congestion, (2) development and distribution of a visitor tracking application, and (3) demonstration of the value of data collected by the application. Much of the first objective has been accomplished by the research that went into the creation of this proposal. The project team's understanding of the success and failures of previous methodologies were continuously refined as we continued to work with the Acadia park staff. The second objective required us to create a mobile application capable of tracking and storing GPS data on both iOS and Android devices. The application is extremely secure to protect the privacy of research participants and was not required for participants to download, but rather volunteers were given additional disclosed information to ensure they feel safe using it. To complete this objective, we established a goal of at least 20 people to get what we considered a significant number of volunteer participants. The third objective was accomplished in order to demonstrate that the data gathered by the application was useful. To demonstrate analysis of the data we used the GPS analyzation software ArcGIS. Figure 5 contains the flowchart of our project goal, primary objectives, and associated tasks.

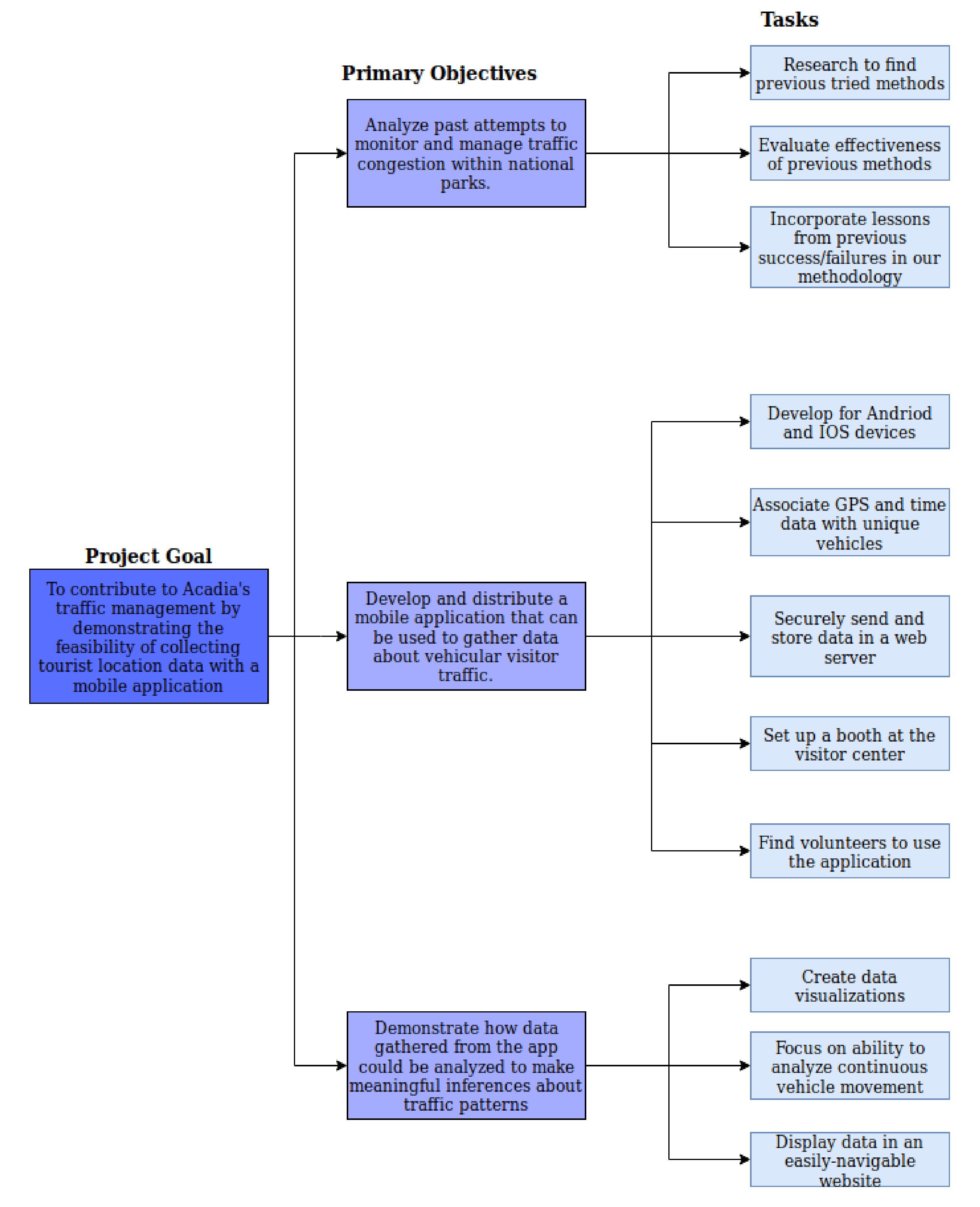

Tasks
Primary Objectives
Project Goal
To contribute to Acadia's traffic management by demonstrating the feasibility of collecting tourist location data with a mobile application
Analyze past attempts to monitor and manage traffic congestion within national parks.
Research to find previous tried methods
Evaluate effectiveness of previous methods
Incorporate lessons from previous success/failures in our methodology
Develop and distribute a mobile application that can be used to gather data about vehicular visitor traffic.
Develop for Andriod and IOS devices
Associate GPS and time data with unique vehicles
Securely send and store data in a web server
Set up a booth at the visitor center
Find volunteers to use the application
Demonstrate how data gathered from the app could be analyzed to make meaningful inferences about traffic patterns
Create data visualizations
Focus on ability to analyze continuous vehicle movement
Display data in an easily-navigable website

Figure 5: Flowchart of Project Goal, Objectives, and Tasks

4. Methods

I. Development

The application development process incorporated two separate but interconnected components. These components include the front-end mobile application and the back-end server. The front-end was developed using the React Native framework which allows the development of a Java Script that was compiled to run on Google's Android platform. It consists of all of the software that will actually be downloaded on a user's mobile device. This includes any visual display the user sees, as well as the code necessary to acquire the user's GPS data, store it locally, and send it to the server. The back-end server is separate from the front-end application and consists of an on-site device running a web-server. This server is responsible for properly handling and storing all of the data that it receives from participants who have downloaded the application.

The development phase included the creation of an the application and server which was tested via both virtual simulation and on physical devices. The development process continued throughout the duration of the project, we continued to reevaluate the functionality of our application in order to address any bugs or concerns that arose as the application was distributed and used on a larger scale. This was especially important for the server application because the increase in participants meant that it is under more strain than it encountered during the pre-deployment testing.

Once the application and server were fully developed and tested with smaller populations, it was deemed fit for distribution to visitors as mentioned above. The app was deemed fit based upon its functionality during tests and its incorporation of essential privacy features. The application was compiled into an apk file and hosted on our web server. Since the the visitor center generally has enough cell connection to access the web server, this distribution method was deemed sufficient.

II. Administrative Interface

The administrative interface is used by a group of authorized members to view data, delete data, and adjust the server configurations. In order to protect the system and its data, there is an account-based login system. Accounts are only created by the existing super-admin, which is Mikayla Fischler. There is no way for non-users to register or gain access to the system. Passwords are stored in a heavily secured form using 2048 iterations of the Blowfish cryptographic hashing algorithm. Once logged-in, users are greeted with a dashboard showing a system overview. It contains network information and statistics, server information and statistics, the status of system services, database statistics, device statistics, and data statistics. The device statistics contains a count of authorized devices, enabled devices, disabled devices, and the number of devices active in the last 24 hours. The data statistics shows the time of start of collection, the number of GPS data points collected, the number of unique paths, and the number of user data wipes.

Once logged in, users can view the raw collected data, which consists of a list of collected data points, in addition to a delete button to delete individual points. On another page, there is a list of device keys (the UUIDs), which can be enabled or disabled in addition to being deleted. Next, we have a system test page which allows for creation of test UUIDs and test data points. There are then pages to view these as with the actual device keys and data points.

For configuration and administration, users with elevated privileges can lockout all users except for the super admin in the case of an account being stolen. Additionally, device reporting can be disabled or enabled, which can prevent devices from uploading data. Next is the deletion section, which allows for the deletion of all GPS data, all device keys, all test data, or all test keys. The count of mobile app initiated user wipes can also be reset to 0. Finally, users with the proper permissions can modify the application key that is used to authenticate devices looking for a new UUID.

In order to keep the system running smoothly, we have a paginated log viewing interface as well. This allows for users to see system logs, account logs, admin logs, user logs, login logs, warnings, and errors. These labels can be used as filters to only show one category, or all can be viewed at once. The system logs contain changes to configuration and mass deletion, the account logs include lockout events and user creation. Admin logs contain individual data point deletions and UUID deletions for test and actual data. The user logs contain requests for new UUIDs, but do not show any information relating to the user's identity. As would be expected, the login logs contain a history of attempted logins and the IP that they originated from. Warnings contain a list of important events, such as core configuration changes, user data deletion, and application key changes. Finally, the error logs contain a list of errors that have occured, where they originated from, what the message was, and what the error code is in order to facilitate quick and easy debugging.

III. Backend Data Collection

The backend data collection encompases the app-to-server interaction. The system data is stored on a MariaDB server instance. MariaDB is open source software that allows for the easy creation, manipulation, and access of databases using SQL syntax. Our databases store all location data, user identifiers, test data, test identifiers, administrative user accounts, the authentication token, configurations, and logs.

The mobile app is able to interface with the storage system using multiple scripts on the server that are accessible through TLS 1.2 encrypted HTTP requests. Once the app is freshly installed on a device, it sends a request to a server side script to attain a UUID (universally unique identifier). In order to maintain security, UUIDs are only handed out to devices with the legitimate application key, which is a predetermined key present in all of the instances of our mobile app. Once the UUID is received, the app can then report location data using either a single data point reporting script, or a batch reporting script. If at any point a user decides they want to opt out of the study or wipe their location data, they can press a button in the app that will send a request to the server to wipe all location data associated with their UUID.

The collected data in the database provides the tracking information of the users. The data stored in the database has no way to be linked back to a person's identity, as all it stores is their UUID, which only their phone and the authorized devices list knows. Using this data, we can form the paths of the users, since each user has a distinct UUID and can be differentiated from any other user.

In addition to normal functionality, the server will log errors and other notable events. Errors are stored both in a text log file on the server and in the database for easier access and simplified debugging. Other events that are logged are the creation of new UUIDs, the full deletion of user data initiated by that user's device, and administrative events. By keeping track of the number of full user wipes, we can see how many users decided to opt out.

A visual representation of communication between the application and the server is shown below in Figure 6.

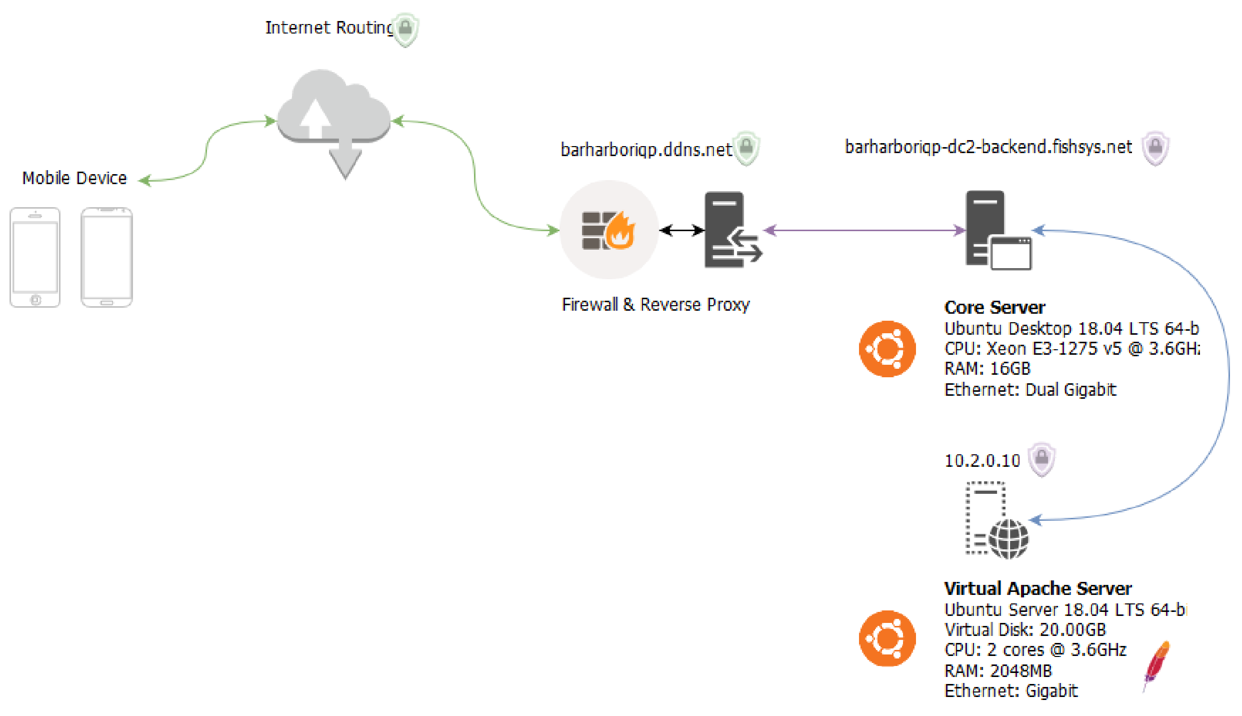


Figure 6: Network Diagram

IV. Application

The mobile application was written using the React Native framework which is an extension of ReactJS. ReactJS is a JavaScript library commonly used for front end web development. React Native takes this framework and allows it to be used in the creation of mobile applications rather than just web applications. The app also makes use of JSX which is a syntax used for embedding XML within JavaScript. XML is a markup language which allows developers to add a variety of custom styles to their user interfaces.

The primary function of the application is to collect a latitude, longitude, and time from the users mobile device and send requests to a back end web server to upload the collected data. The application will automatically upload the data points that it has collected if the user has wifi connection. If the user has data service like LTE or 3G the application will store up to 25 data points locally before attempting to upload to the server. If the user has no cell service then the application will store data points until a connection is found. The practice of prioritizing network requests while on a wifi connection is intended to minimize the data usage of our application. In order to allow data collection while in the background, our application makes use of Transistorsoft's React Native Background Geolocation API. It is quite difficult to run background process in a React Native application, and this API made that much easier. We felt that it was extremely important to include background geolocation as a feature because this allowed users to enjoy their phone however they normally would while still fully participating in our research. Using the API ensured that our app did not interrupt their ordinary cell phone use in any way.

5. Analysis

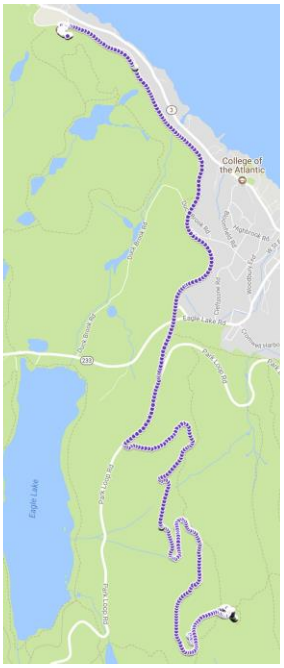
I. Application:

Across a period of three days, the application was distributed to 35 volunteers at the Hulls Cove Visitor Center. The team set up a small booth to distribute the app and typically worked the booth from about 10:30am to 1:30pm since the Visitor Center is most busy during that period of time. Figure 7 shows the booth used for app distribution.

Figure 7: App Distribution Booth

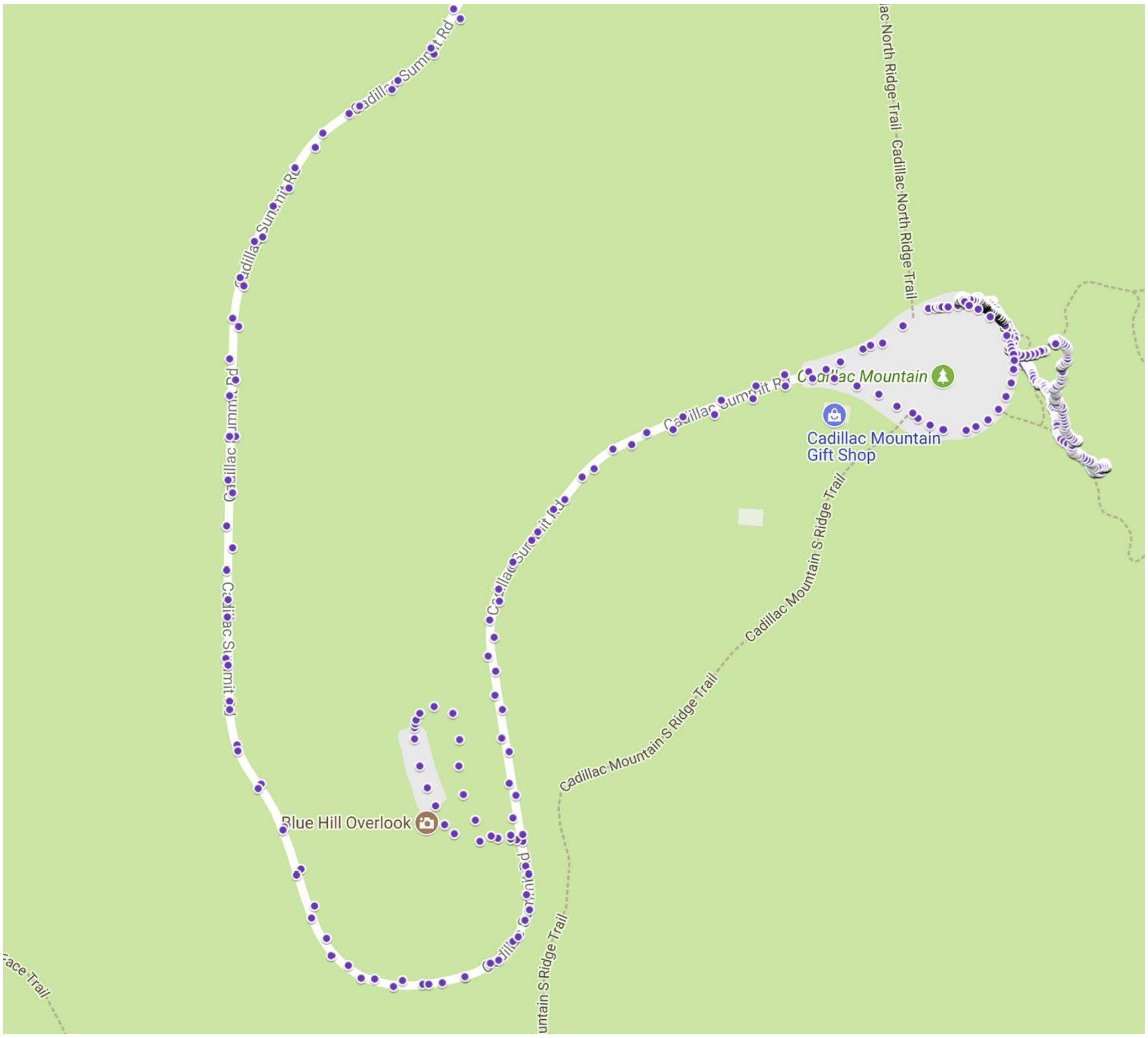
On the first day of distribution, the application's code contained a bug that occured when the app lacked wifi connection when being run for the first time. Unfortunately, this prevented the proper function of our application on the majority of devices from the first day. We were able to fix the application later that night and distribute an updated, working version on days two and three.

One example of a data set collected by a user of our application can be viewed below:



(Figure 8: Data collected by a single user of the mobile application)

The path shown here begins up north at the Visitor Center and continues a bit south towards the summit of Cadillac Mountain. A zoomed in picture of the path, depicted below, allows a better understanding of the data it contains.


Cadillac Summit Rd
Cadillac North Ridge Trail
Cadillac North Ridge Trail
Cadillac Mountain
Cadillac Mountain Gift Shop
Cadillac Mountain S Ridge Trail
Cadillac Mountain S Ridge Trail
Blue Hill Overlook
Face Trail
untain S Ridge Trail

(Figure 9: Zoomed view of figure above)

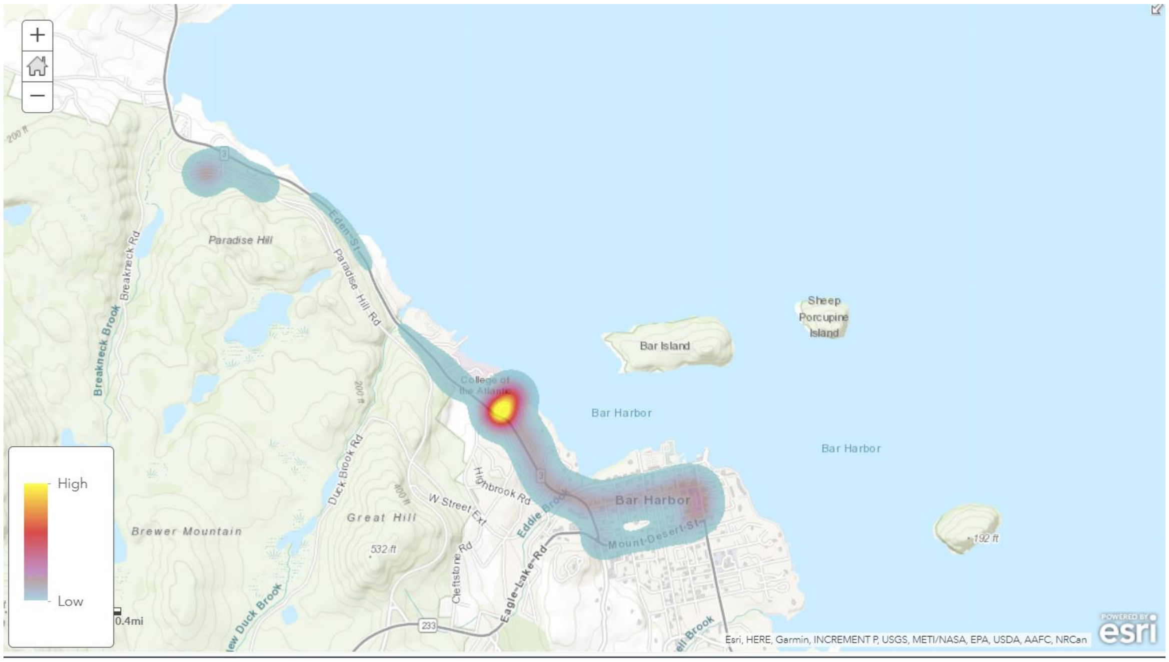
When zooming in, it becomes rather obvious that the path is merely a set of closely positioned data points. Each data point contains a latitude, longitude, timestamp, and UUID (Universal Unique Identifier). Because each data point contains a latitude, longitude, and timestamp, the data allows us to see exactly where the device is, where it came from, where it goes, and how long it spends in each location. The UUID simply allows us to differentiate among the various users of our application.

Although meaningful analysis is difficult with such a limited sample size, the data gathered from our application allows us to make projections about how data of this type could be analyzed in the future if the application was distributed to a much wider audience.

II. Application Data:

In order to analyze the data collected by the application, the team used Google Maps and ArcGIS. Google Maps provided a quick way to plot the collected GPS locations to show an individual's movement. In this way, some of the better examples collected were plotted to demonstrate the resolution of data. However, Google Maps caps the total amount of data that can be uploaded to 250 MB. Maps also lacks analyzation tools beyond simple visualization.

ArcGIS is the premier GIS software used by many professionals globally. Its expansive library of tools and options allows for deep analyzation. ArcGIS is organized by layers on a map. A .csv containing GPS points, time, and UUIDs of virtually unlimited size can be uploaded to any number of layers. The number of data points that can be displayed is limited by the computer, not the software itself, and so can be in the billions of points. A large dataset collected over a long period of time can be displayed in ArcGIS as a heatmap to show the highest and lowest densities of visitors. Since the collected data includes a time, data can be separated and displayed according to a desired period of time.



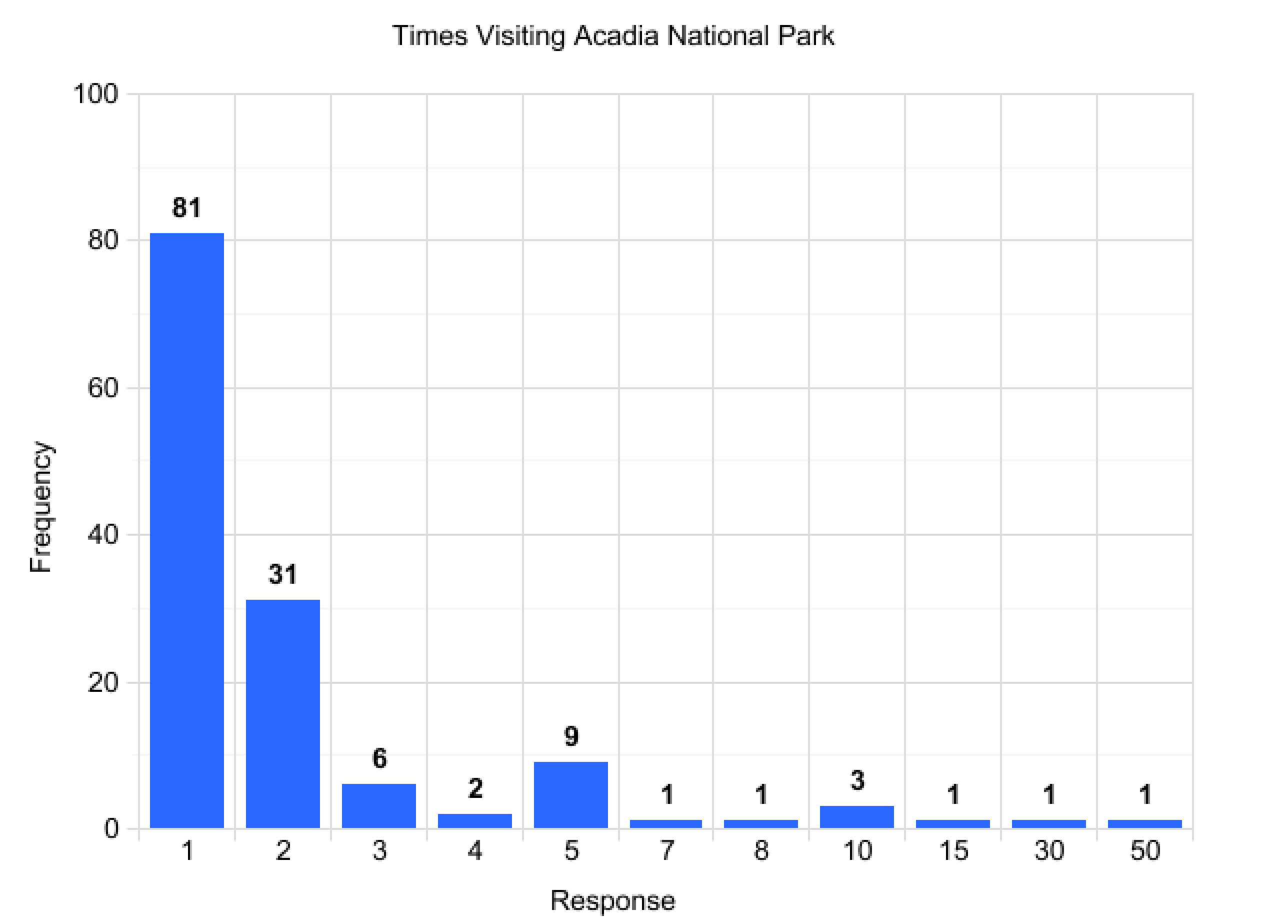
(Figure 10: Heatmap of example data)

ArcGIS also provides the ability for data to be animated over time. Data points with different UUIDs can be differentiated either by icon or by color. A user can set a range, with times and dates, for what data they want shown. Within this range, they can choose an interval, such as 10 seconds. Starting at the beginning of the range, the map will display all the data points that were collected in each 10 second interval sequentially. This essentially creates a view of all the visitors moving through the park over time.

In the long term, analyzation of this data would be done by a larger professional operation. Acadia, or whatever developer is responsible for the application, would most likely use their own tools and techniques to look at patterns and generate metrics. Alternatively, Acadia could hire a third-party contractor that specializes in GIS analytics.

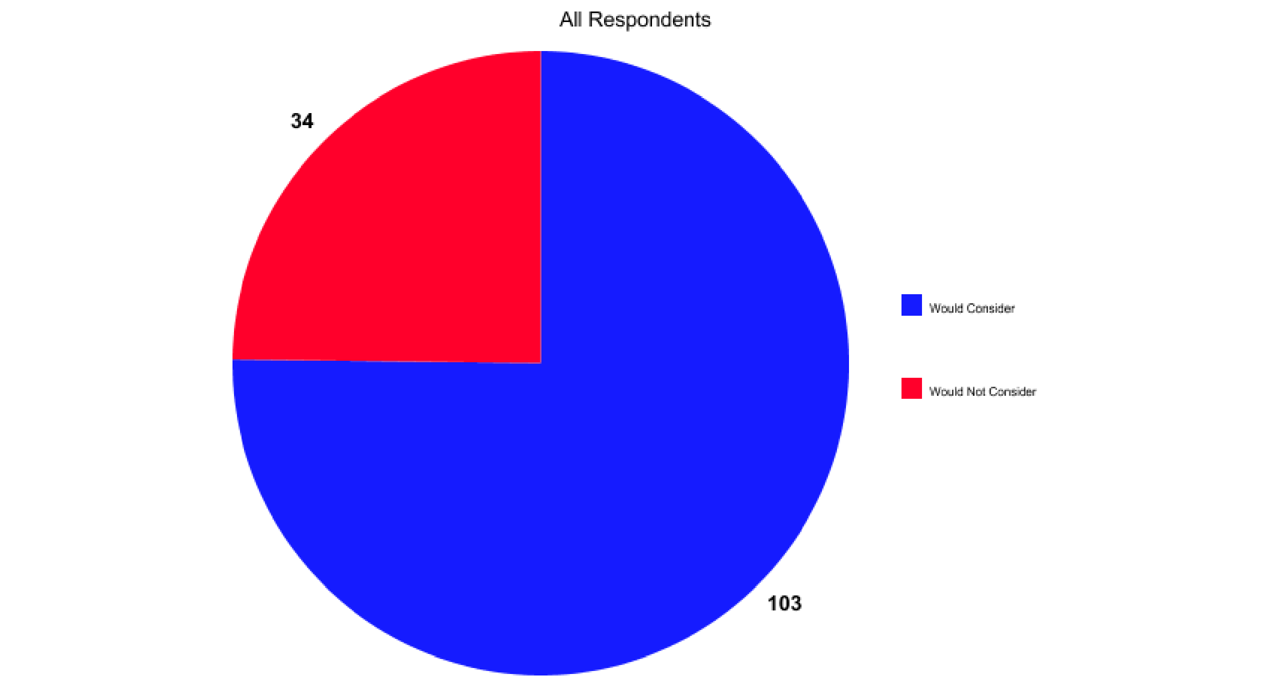
III. Survey:

The three question survey shown in Appendix 1 was distributed to 137 participants at the Hull’s Cove Visitor Center. When asked how many times they had visited Acadia National Park, 81 respondents (59.12%) said that it was their first time visiting the park. Of all the respondents, 94.16% reported having been to the park five or fewer times. On average, respondents reported about 2.58 park visits, but this number is brought up significantly by a few outliers who reported having been to the park 15, 30, or even 50 times. The complete data set of responses to this question can be seen in the chart below.



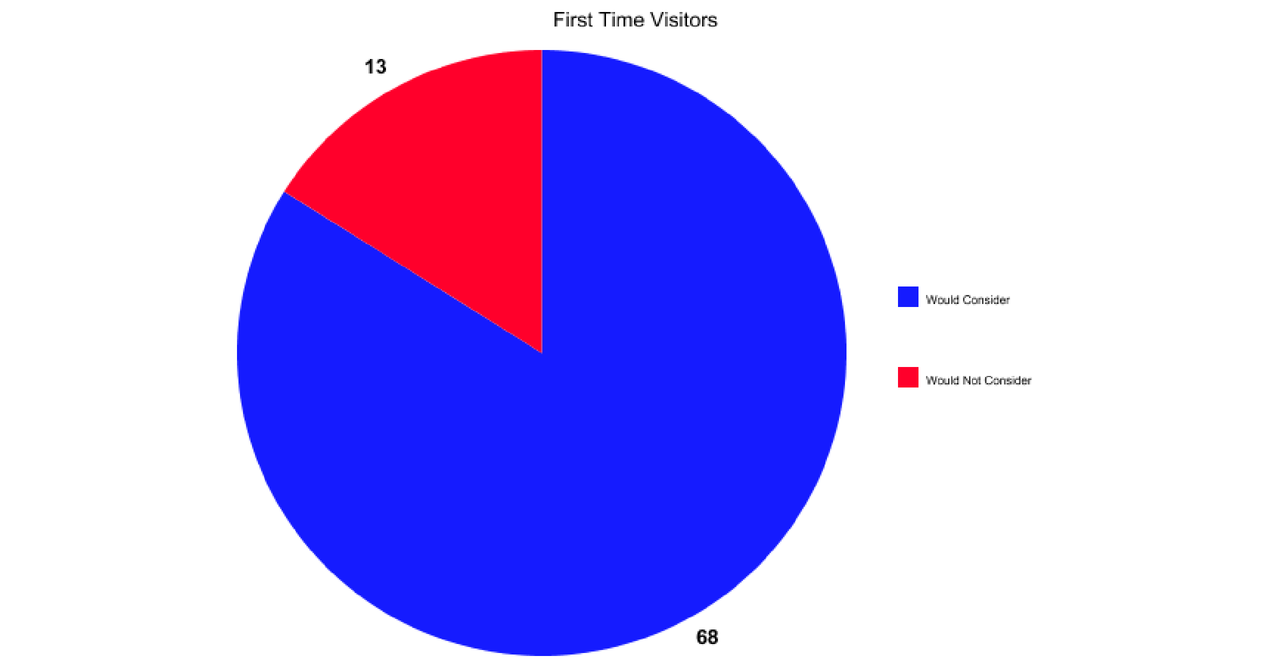
(Figure 11: Survey Response)

When asked about mobile applications, 103 participants (75.18%) stated they would consider downloading a vehicle tracking application whose data was collected and used by park officials. Although such data was not formally collected on the survey, several participants who responded “No” to this question made comments regarding their lack of cell phone use or their disinterest in any mobile application technology. The ratio of “Yes” and “No” responses can be viewed visually in the chart below.



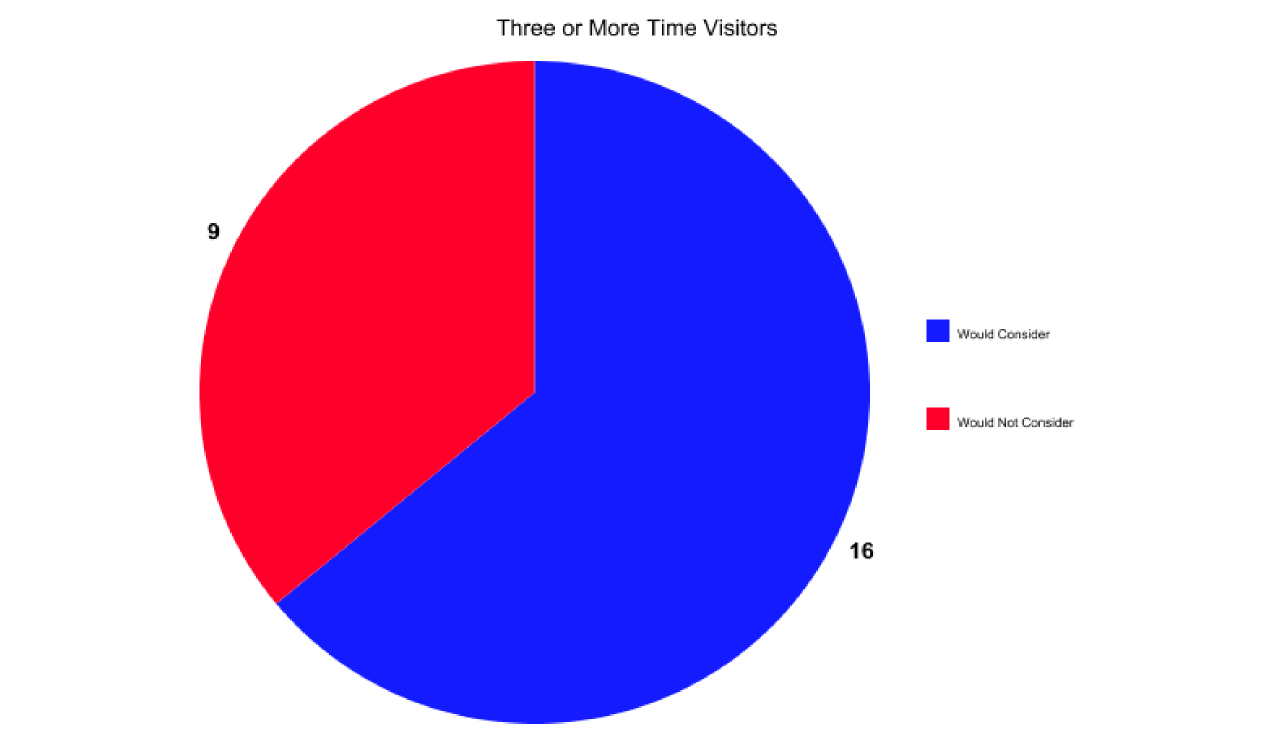
(Figure 12: Answers to interest in downloading)

Interestingly, first time park visitors are more likely than others to consider downloading an application. Of the 81 first time visitors, 68 of them (83.95%) reported that they would be willing to consider using the application.



(Figure 13: First time park-goers response to downloading app)

Of the 25 visitors who had been to the park three or more times, only 16 of them (64%) responded “Yes” to question three.



(Figure 14: More than one time park-goers response to downloading app)

Although results may just be due to a relatively small sample size (n=25), it may be possible that frequent park visitors are less inclined to use the application because they already know the park well and could would not be lured in by features that newer users could find helpful. Alternatively, it could just be that first time park goers are more enthusiastic and motivated to help the park in any way possible.

Regardless of the reasoning behind people's responses, a 75% “Yes” response on question three indicates that there is a very large potential marking for a vehicle tracking application. Taking into consideration that the park had about 3.5 million visitors in 2017, the potential market size for one year can be estimated around 2.6 million if the result from our survey is assumed to be representative of the entire population. Even if only a small fraction of this market ever actually downloaded the app, it is certain that enough data could be collected over a period of a few years that meaningful inferences could be made regarding broad traffic patterns throughout the park.

IV. Challenges

The team encountered a few unexpected challenges during the application development process. The first challenge encountered was allowing GPS to be tracked while the app is running in the background. If the app had to always run in the foreground, it would be a big inconvenience for our users since it would prohibit them from doing anything else on their phones while they use the app. Unfortunately for us, it was unexpectedly difficult to allow an app to run in the background when developing with React Native. After a bit of research, the team ultimately decided to purchase and implement the React Native Background Geolocation API developed by Transistorsoft. This API essentially allowed us to replace the current GPS location function with Transistorsoft's version of the function which is very similar, but runs in the background by default.

The other major problem the team ran into during development was building and deploying an ipa file for use on iOS devices. Although creating the iOS app was certainly possible, it proved more difficult than we expected, especially when considering the team's lack of prior app development experience. In the interest of time, we ultimately decided to stop development on the iOS app and focus entirely on the android platform. Despite our lack of success in building the iOS app, we are confident that it could be done by a group with more time and/or experience.

6. User Manual

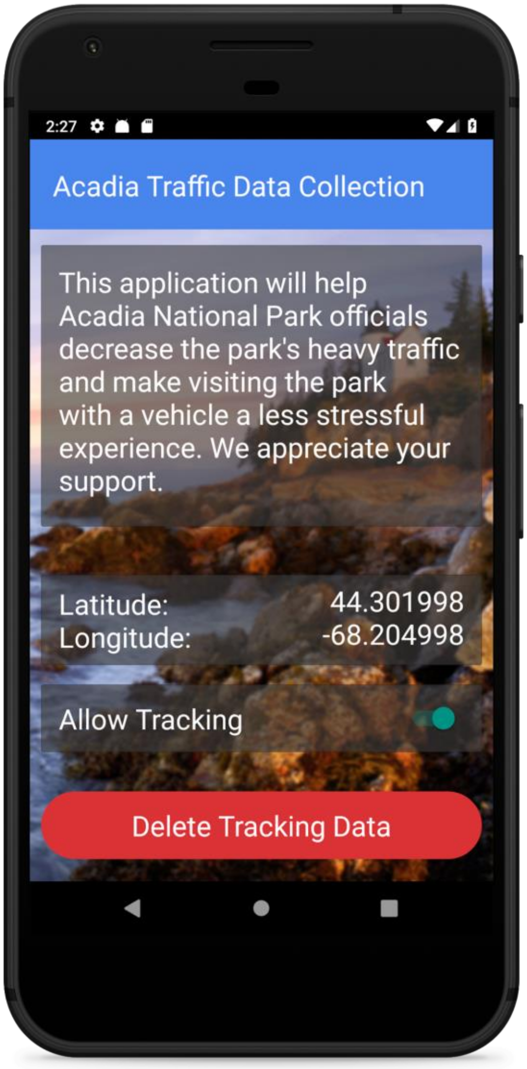
Table of Contents

I. Mobile Application Installation

To install the Acadia Traffic Solutions application onto an Android device, users followed these steps:

1. Navigate to https://acadia-app.ky8.io
2. Click the download link
3. Allow the .apk file to be downloaded
4. When the download has finished click on the file and select install
5. If necessary, allow your browser as a source for downloading and installing apk files.
6. Once the installation is finished the process is complete and the application can be opened

II. Mobile Application- Home View



This application will help Acadia National Park officials decrease the park's heavy traffic and make visiting the park with a vehicle a less stressful experience. We appreciate your support.

Latitude: 44.301998
Longitude: -68.204998

Allow Tracking ●

(Figure 15: Mobile application home screen)

III. Mobile Security

All data about volunteer app users is completely anonymous. Each mobile device is identified by a unique randomly generated series of letters and numbers that cannot be tied back to the user in any way. No personal data aside from location is ***ever*** collected from the app. All network requests made by the mobile application are heavily encrypted using industry standard TLS 1.2, this ensures that data cannot be spied on as it is sent across the web.

IV. Mobile Application- Button/Switch

I. Stop Tracking Switch

Our application included a feature that allowed users to stop the app from tracking them whenever they desire. This feature was built-in specifically with locals in mind. People who live on Mount Desert Island may feel uncomfortable having the application track them back to their houses, so we felt that this feature was important to include. The user is free to resume tracking whenever they want by simply flipping the switch back into the on position.

II. Delete All Data Button

This feature was created for those users who completely regret their participation in the study. It gives volunteers the option to opt out of the data collection process and completely wipe all data that has already collected.

V. Collection Bounds

There are two conditional statements in the code that draw a rectangular box around Mount Desert Island. If the application user travels outside of those bounds, the app will stop tracking and collecting data. This feature was made to prevent the application from tracking users back to their homes. This feature is also useful for avoiding the collection of useless data, since we are only interested in traffic patterns within the park.

7. Conclusions/Recommendations

As our methods and analysis progressed, the team made key findings towards mitigating traffic congestion in Acadia National Park. The findings from our project show the willingness of park goers to download a mobile app that tracks location, specific traffic hotspots based on time of the day and weather, and how this mobile application can be used to show traffic patterns in real time, as well as collect long-term traffic data.

The results of our survey concluded that 103 people out of the 137 asked would be willing to download and use a phone application that could track your vehicle movements while in the park to assist Acadia in researching traffic congestion, this is an outstanding 75 percent approval rating. The acceptance rate of our application with the public shows that people are willing to assist the park in our proposed methodology. After three days distributing the mobile application at the visitors center, 35 people downloaded the application. We were able to collect enough data to show the behavior of individual users as they move within the park. In the future as data is collected and stored over time, more patterns will start to emerge. By demonstrating the feasibility and functionality of our traffic monitoring application we are providing Acadia National Park with a more efficient way to collect data and eventually provide the park with data that is needed to make informed decisions in response to the parks' vehicular traffic.

With regards to long term integration of a vehicle tracking application, the team has come up with a few suggestions. The first suggestion is to speak with developers of a pre-existing Acadia application and see if they would be willing to incorporate vehicle tracking features into their application. This is an appealing option because it greatly reduces the need to market the application. Some independently made Acadia apps, such as Chimani, already receive tens of thousands of downloads annually (Chimani). The drawback to this option is that it requires cooperation between park officials and independent application developers.

A more realistic option for long term implementation would be for Acadia National Park or Friends of Acadia to hire a private contracting company to develop the application from the ground up. This option is appealing because it would allow Acadia to tailor the app to its specific needs and have complete control over its development. In addition to vehicle tracking capabilities, the park would be free to add any other features that it wants. The downside to this development method is that the new application would have to be marketed from the ground up.

An appealing alternative to hiring a private development company would be for Acadia National Park or Friends of Acadia to develop their own app using a company such as OnCell (OnCell). OnCell provides an easy-to-use method for creating mobile applications in a way similar to how WordPress or Weebly allows users to create websites. Using the OnCell platform, Acadia could decide which features it wants to incorporate into its application. Furthermore, OnCell has experience working with museums, National Forests, and National Parks, so the company is likely familiar with the features that Acadia would look to implement.

Regardless of which option is chosen, it is imperative that Acadia National Park seeks to create a vehicle tracking application in the very near future. Traffic problems in the park are well documented and show no signs of alleviating in the near future. The data collected from a vehicle tracking application could provide park officials with vital data as they continue to search for ways to combat the growing problem of traffic congestion.

8. References

Angela Calvi, Colin Maki, Jackson Noah Peters, Mingqi Shuai, Daniel Wivagg (2017,July 28). Intelligent Transportation in Acadia National Park. *Worcester Polytechnic Institute Electronic Projects Collection*. Retrieved from https://web.wpi.edu/Pubs/E-project/Available/E-project-072817-085848/

AOMM Enterprises. (2017, September 11). Traffic shuts Cadillac Mountain summit road 49 times so far. Retrieved April 04, 2018, from https://acadiaonmymind.com/2017/09/traffic-shuts-cadillac-mountain-summit-road-49-times-so-far/

“ArcGIS Online.” Arcgis.com, www.arcgis.com/index.html .

Chimani, Inc. (n.d.). Chimani: Enhancing the Outdoors. Retrieved from https://www.chimani.com/

Eric Riley Cosmopulos, Jesse Trevor Gaulin, Hannah M Jauris, Michael Robert Morisseau, Elizabeth A Quevillon (2017, July 27). Preparing Acadia National Park for Modern Tourist Congestion. *Worcester Polytechnic Institute Electronic Projects Collection.* Retrieved from https://web.wpi.edu/Pubs/E-project/Available/E-project-072717-222628/

Erin Madison (2013, August 14). Glacier National Park grapples with traffic. *USA Today.* Retrieved from https://www.usatoday.com/story/travel/destinations/2013/08/14/glacier-national-parks-shuttles-traffic/2653981/

Hallo, J. C., Manning, R. E., Valliere, W., & Budruk, M. (2004). *172 Proceedings of the 2004 Northeastern Recreation Research Symposium GTR-NE-326 A CASE STUDY COMPARISON OF VISITOR SELF-REPORTED AND GPS RECORDED TRAVEL ROUTES* [Scholarly project].

Kurt Repanshek (2018, January 14). "Arches For The People" Proposes Solution to Arches National Park's Congestion Woes. National Parks Traveler. Retrieved from http://nationalparkstraveler.org/2018/01/arches-people-proposes-solution-arches-national-parks-congestion-woes

National Geographic. (2009, November 05). This Small Seaside National Park Is One of the Most Visited. Retrieved from https://www.nationalgeographic.com/travel/national-parks/acadia-national-park/

National Park Service. (2018). *Current Conditions.* Acadia National Park Service. U.S. Government. Retrieved on 2018, April 4 from https://www.nps.gov/acad/planyourvisit/conditions.htm

National Park Service. (2018, March 05). Average Monthly Visitation, Yellowstone National Park 2013-2017 [Chart]. In *Yellowstone Visitation Statistics*. Retrieved April 2, 2018, from https://www.nps.gov/yell/planyourvisit/visitationstats.htm

"National Park Service." *National Parks Service*, U.S. Department of the Interior, parkplanning.nps.gov/projectHome.cfm?projectID=58482.

NPS Stats. (n.d.). National Park Service Visitor Use Statistics. Retrieved April 02, 2018, from https://irma.nps.gov/Stats/OnCell. "Connecting People,Places and Stories."

OnCell | Mobile App & Audio Tour Builder for Museums & Destinations, www.oncell.com

Pew Research Center. (n.d.). Percentage of U.S. adults who own a smartphone from 2011 to 2018. In Statista - The Statistics Portal. Retrieved April 4, 2018, from https://www.statista.com/statistics/219865/percentage-of-us-adults-who-own-a-sma rtphone/

React Native · A framework for building native apps using React. (n.d.). Retrieved from https://facebook.github.io/react-native/

Traffic in Yosemite National Park. (2018, February 15). Retrieved April 04, 2018, from https://www.nps.gov/yose/planyourvisit/traffic.htm

United States, Department of the Interior. (2014). *Congestion Management Toolkit*. National Park Service.

United States Government. Office of Management and Budget. (2018). *An American Budget: Efficient, Effective, Accountable. Fiscal Year 2019* https://www.whitehouse.gov/wp-content/uploads/2018/02/budget-fy2019.pdf

Appendix A

Traffic Tracking Survey

1) Are you over 18 years old?

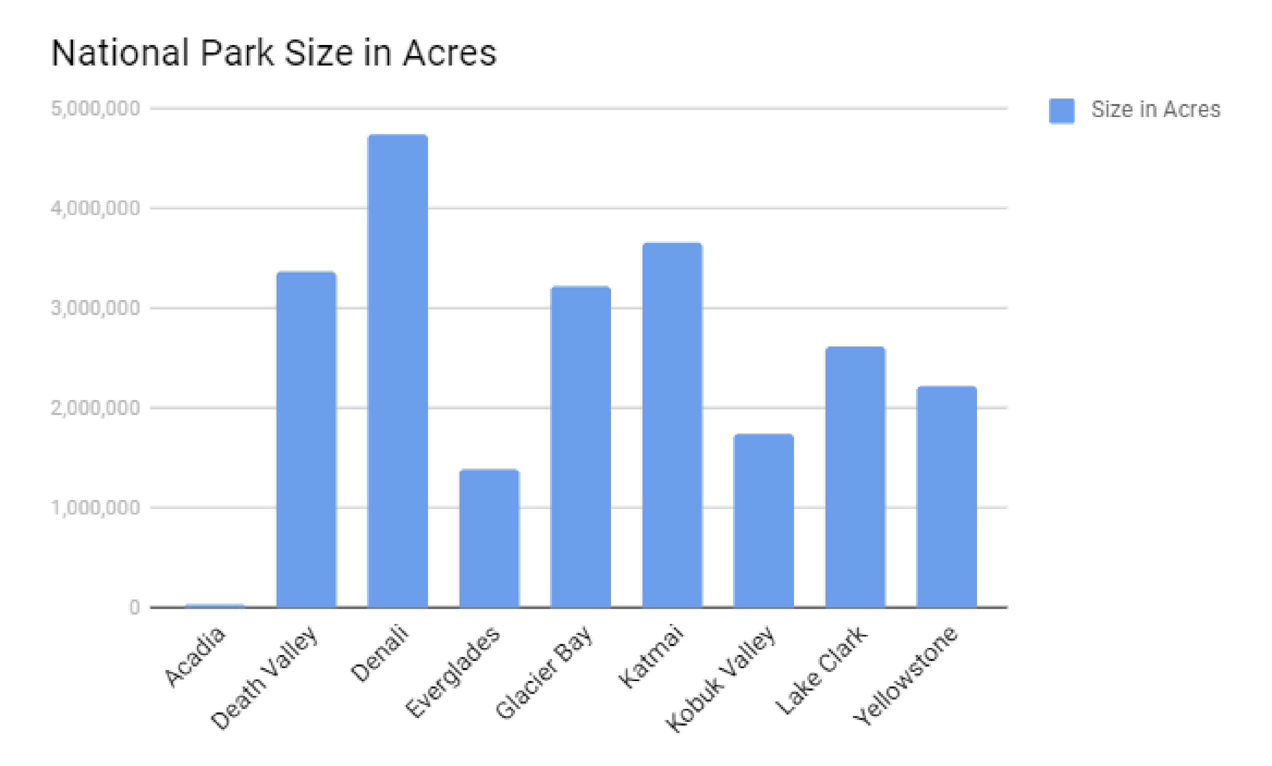
Yes___ No___

2) How many times have you visited Acadia National Park? ________

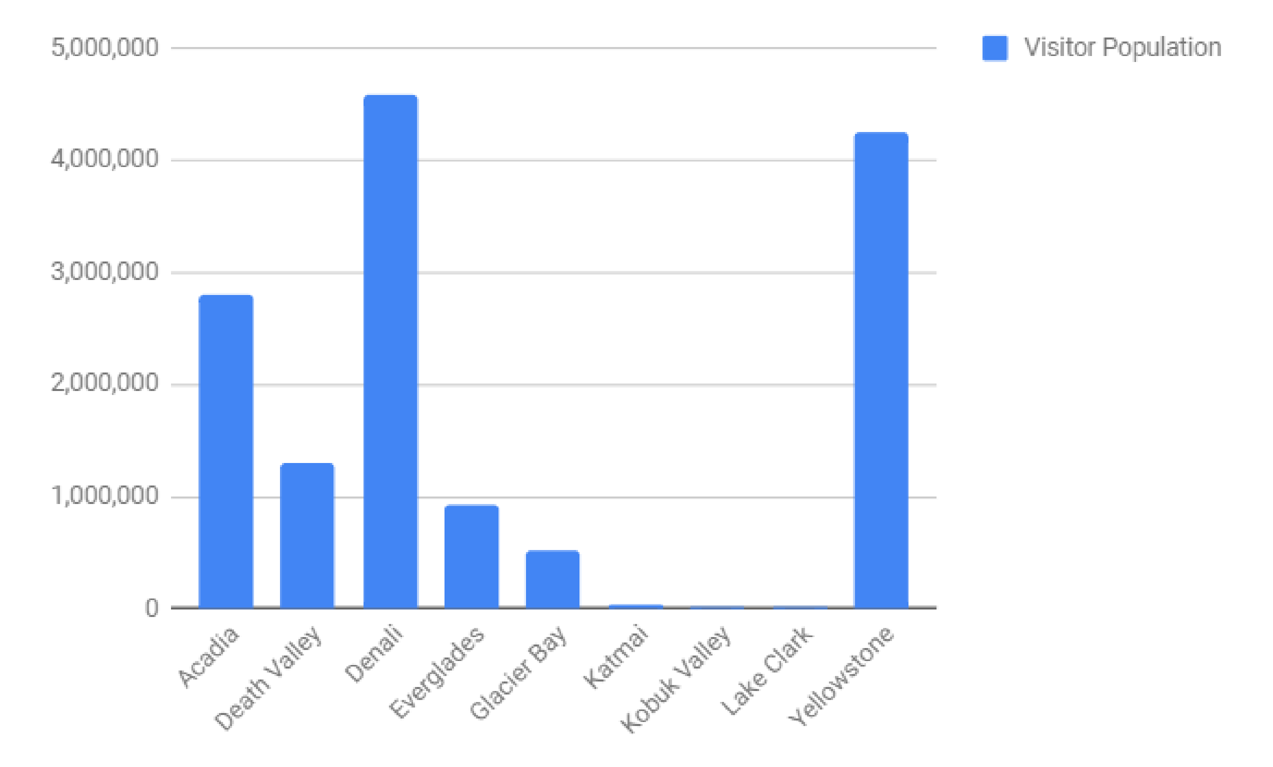
3) If there was a phone application that could track your vehicle movements while in the park in order to assist Acadia in researching traffic congestion, would you consider downloading and use that application.?

Yes___ No____

Appendix B

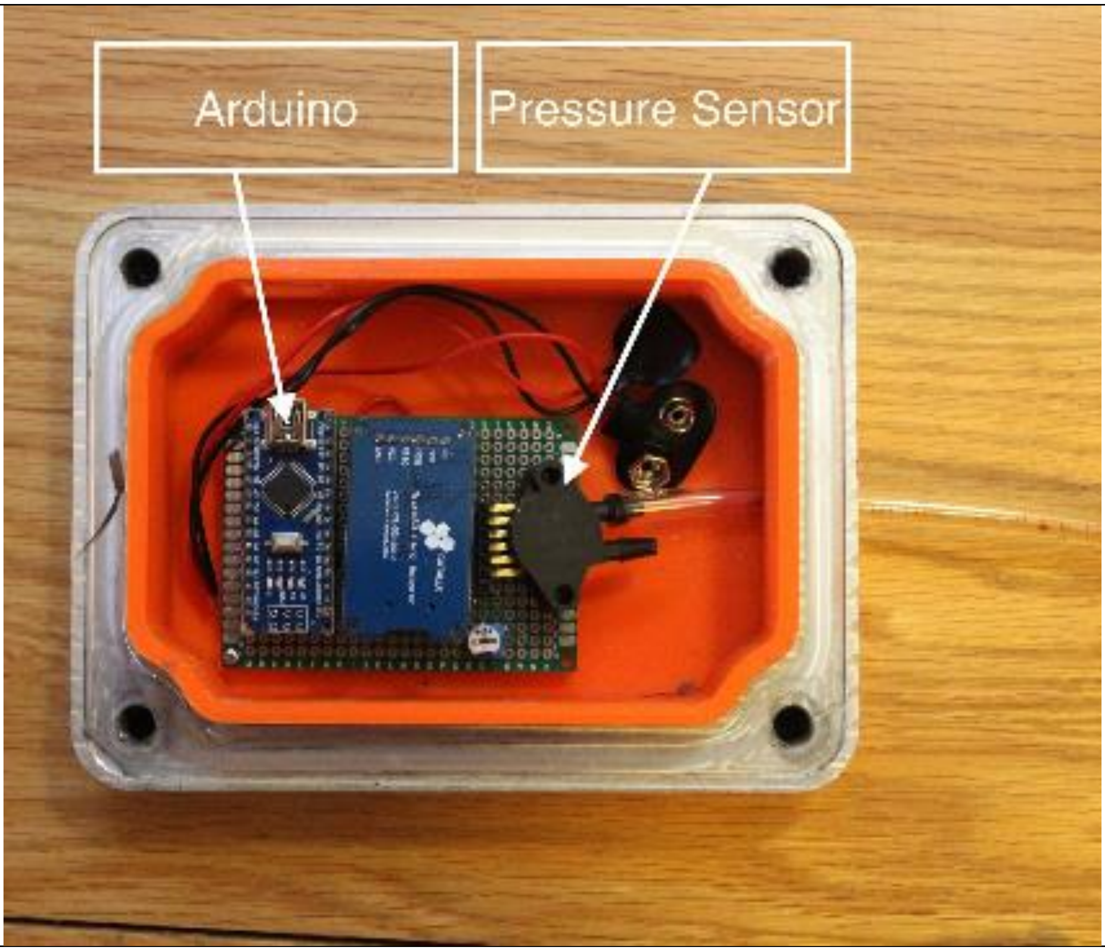


(Figure 16: National Park comparison of size in acres)

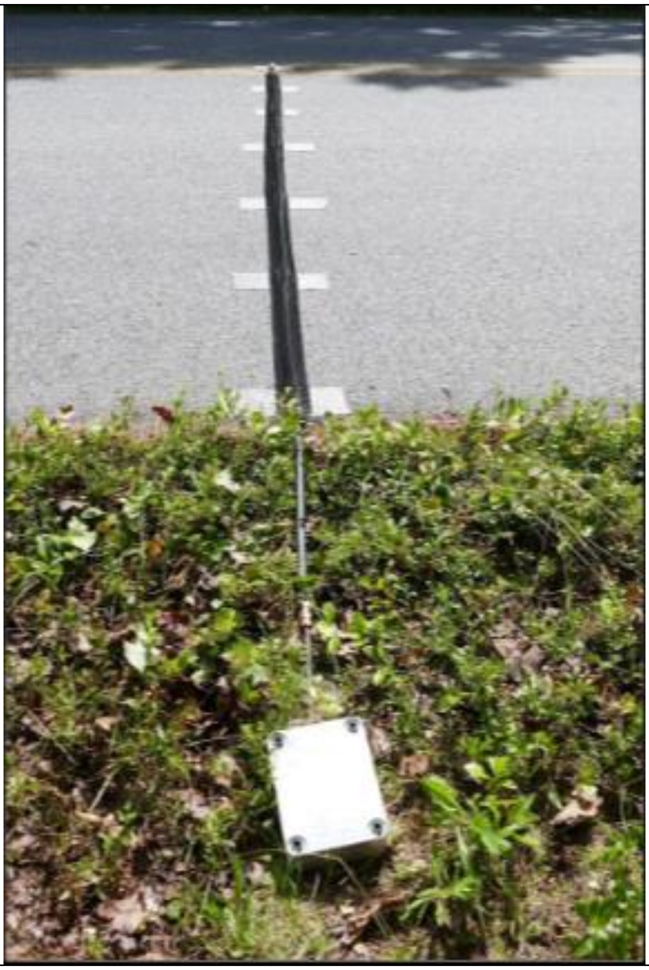


(Figure 17: National Park comparison of visitor population)

Appendix C

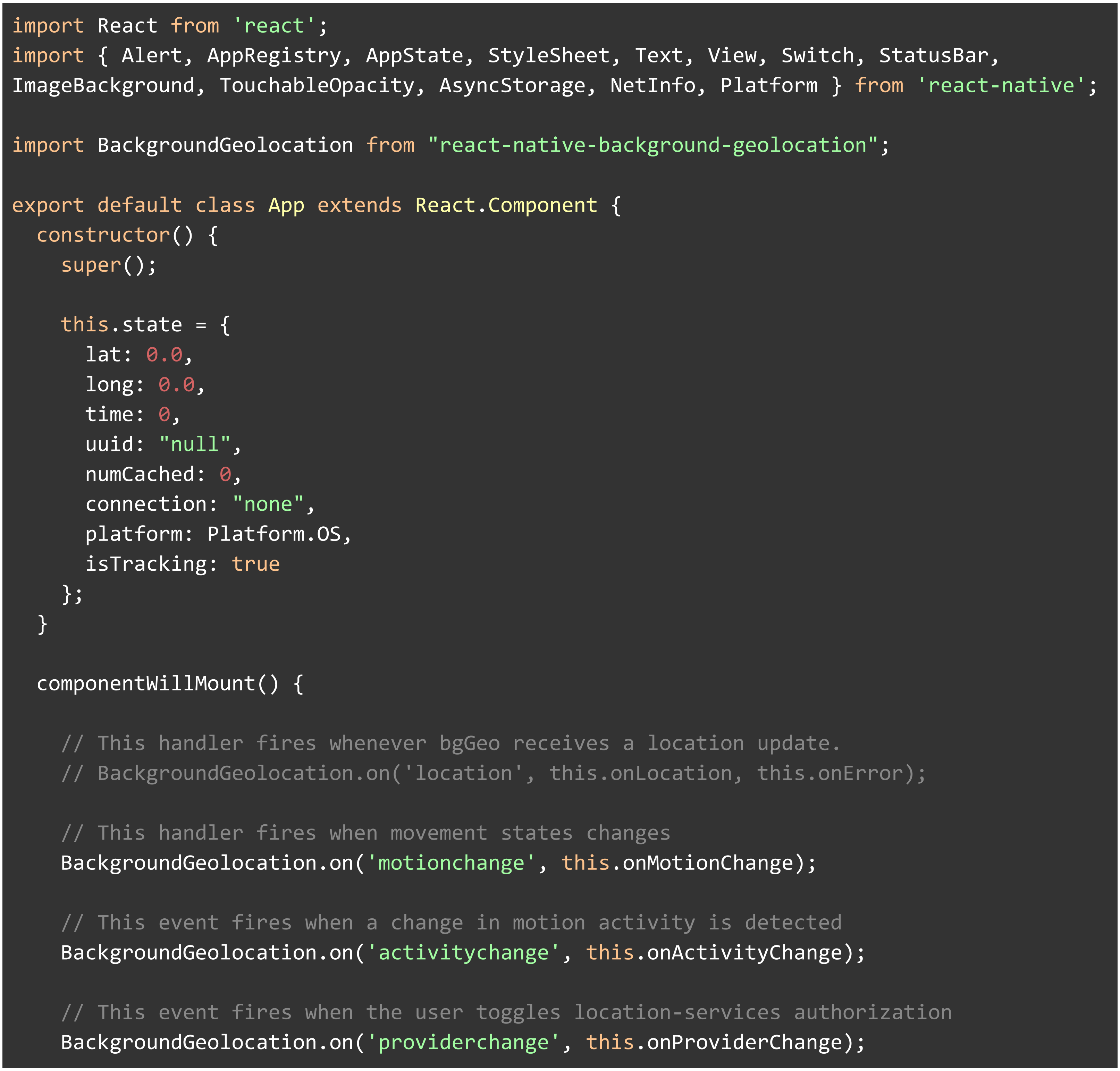


(Figure 18: WPI car counter)

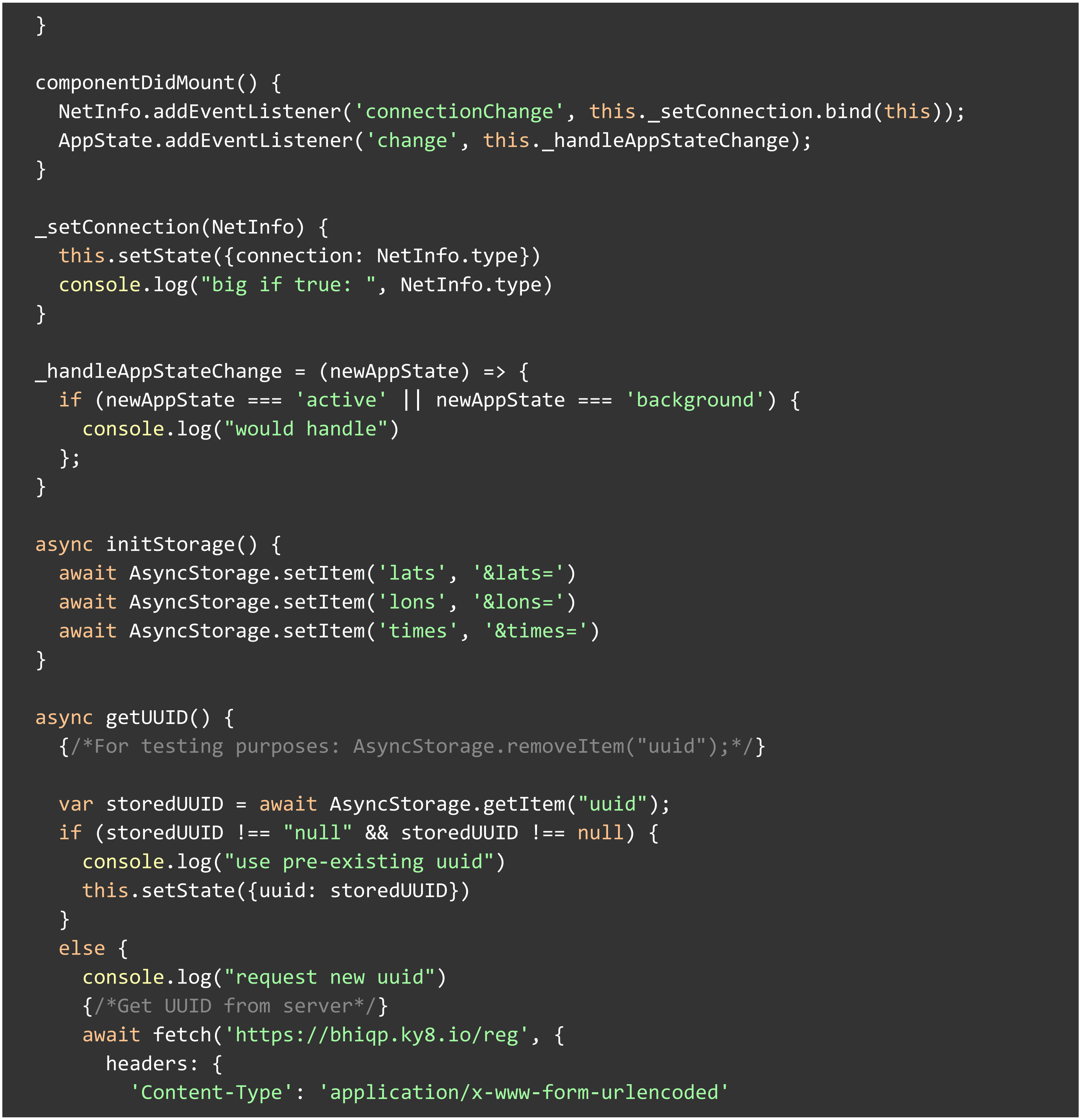


(Figure 19: Traffic Counter)

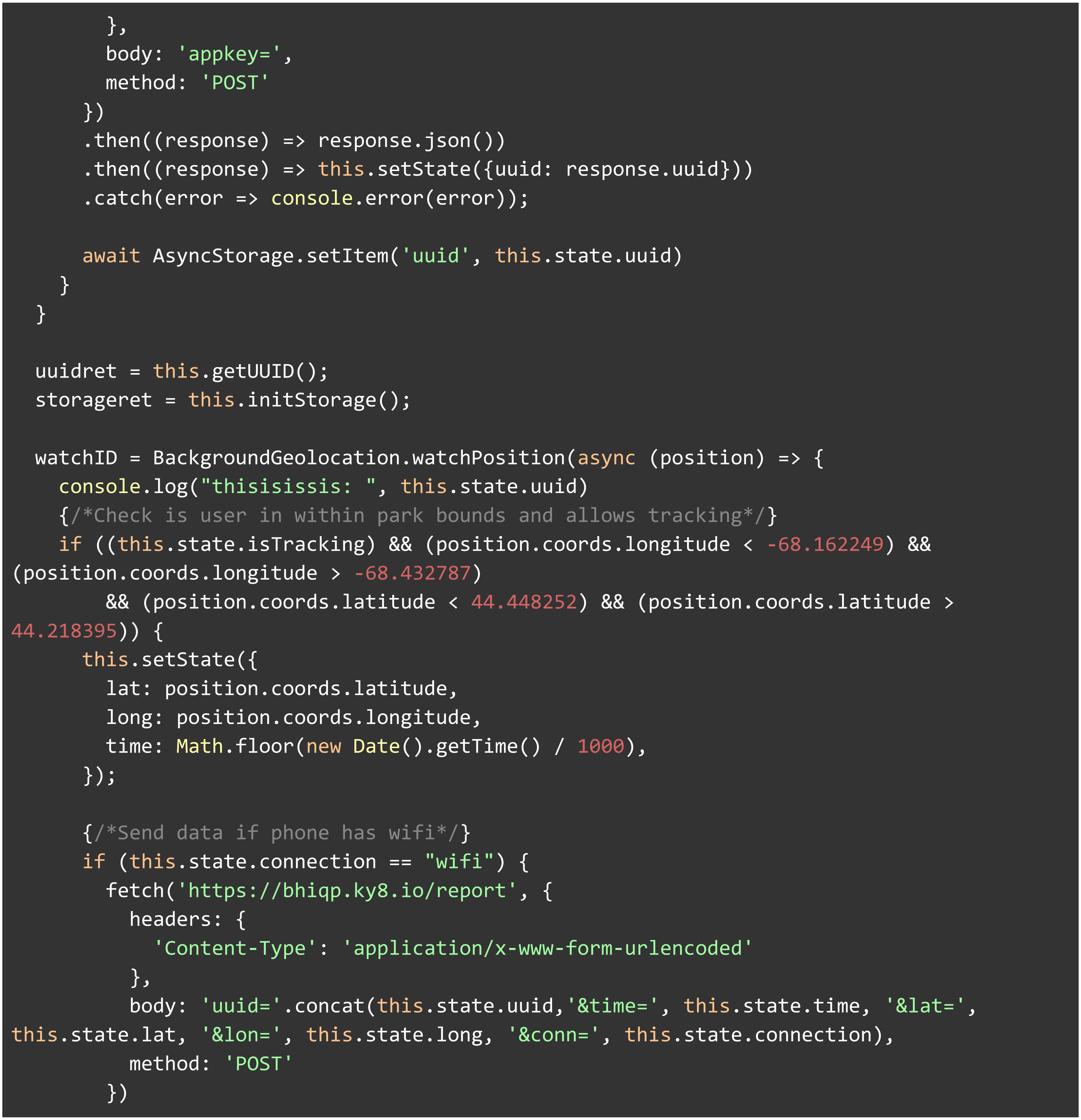
Appendix D



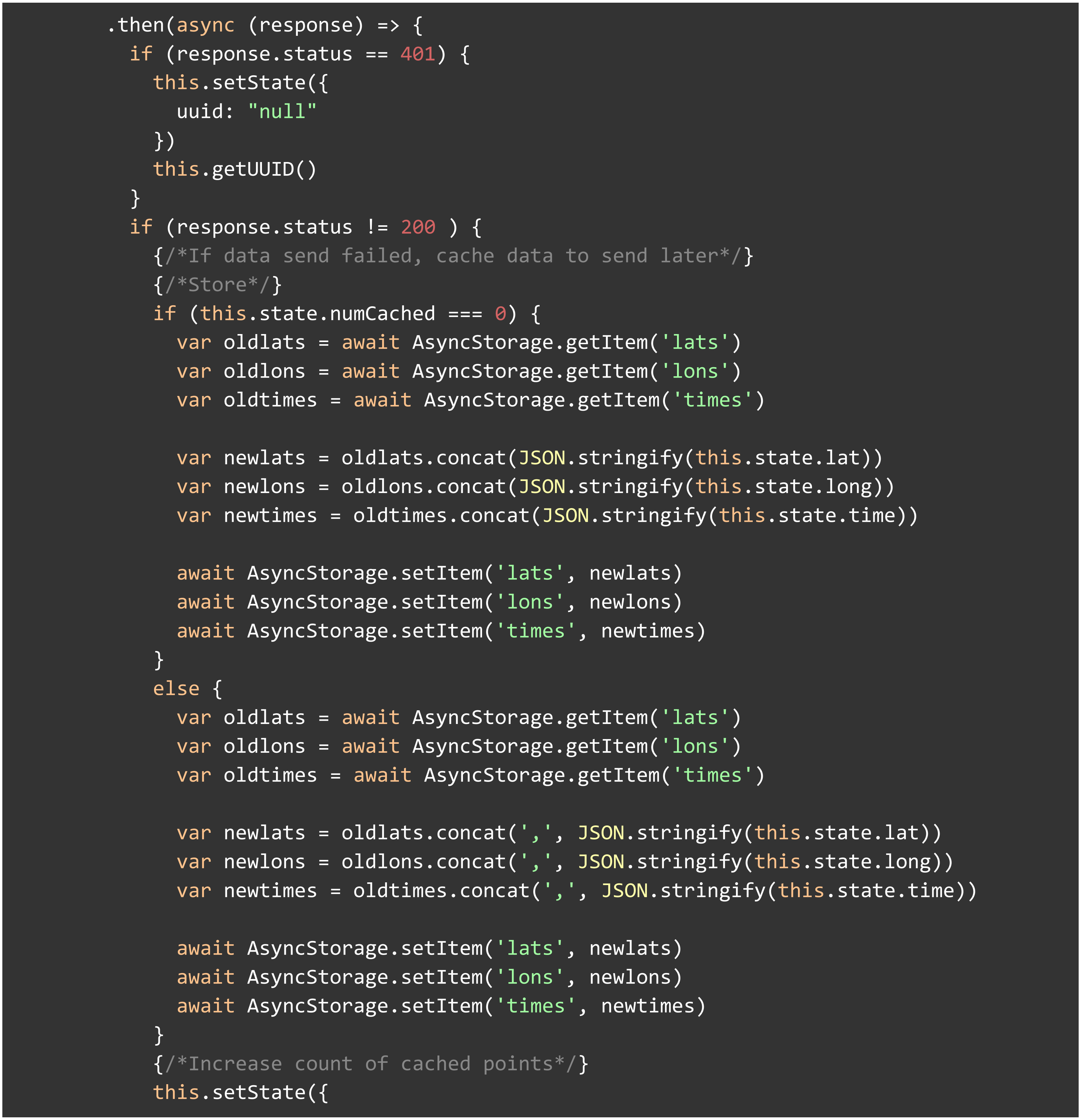
```
import React from 'react';
import { Alert, AppRegistry, AppState, StyleSheet, Text, View, Switch, StatusBar,
ImageBackground, TouchableOpacity, AsyncStorage, NetInfo, Platform } from 'react-native';

import BackgroundGeolocation from "react-native-background-geolocation";

export default class App extends React.Component {
  constructor() {
    super();

    this.state = {
      lat: 0.0,
      long: 0.0,
      time: 0,
      uuid: "null",
      numCached: 0,
      connection: "none",
      platform: Platform.OS,
      isTracking: true
    };
  }

  componentWillMount() {

    // This handler fires whenever bgGeo receives a location update.
    // BackgroundGeolocation.on('location', this.onLocation, this.onError);

    // This handler fires when movement states changes
    BackgroundGeolocation.on('motionchange', this.onMotionChange);

    // This event fires when a change in motion activity is detected
    BackgroundGeolocation.on('activitychange', this.onActivityChange);

    // This event fires when the user toggles location-services authorization
    BackgroundGeolocation.on('providerchange', this.onProviderChange);
    BackgroundGeolocation.ready({
      desiredAccuracy: 10,
      distanceFilter: 10,
      logLevel: BackgroundGeolocation.LOG_LEVEL_VERBOSE,
      stopOnTerminate: false,
      startOnBoot: false,
    }, (state) => {
      console.log("- BackgroundGeolocation is configured and ready: ", state.enabled);

      if(!state.enabled) {
        BackgroundGeolocation.start(function() {
          console.log("- Start success");
        });
      }
    });
  }

  componentWillUnmount() {
    BackgroundGeolocation.removeListeners();
    AppState.removeEventListener('change', this._handleAppStateChange);
  }

  onLocation(location) {
    console.log('- [event] location: ', location);
  }
  onError(error) {
    console.warn('- [event] location error ', error);
  }
  onActivityChange(activity) {
    console.log('- [event] activitychange: ', activity);  // eg: 'on_foot', 'still', 'in_vehicle'
  }
  onProviderChange(provider) {
    console.log('- [event] providerchange: ', provider);
  }
  onMotionChange(location) {
    console.log('- [event] motionchange: ', location.isMoving, location);
```



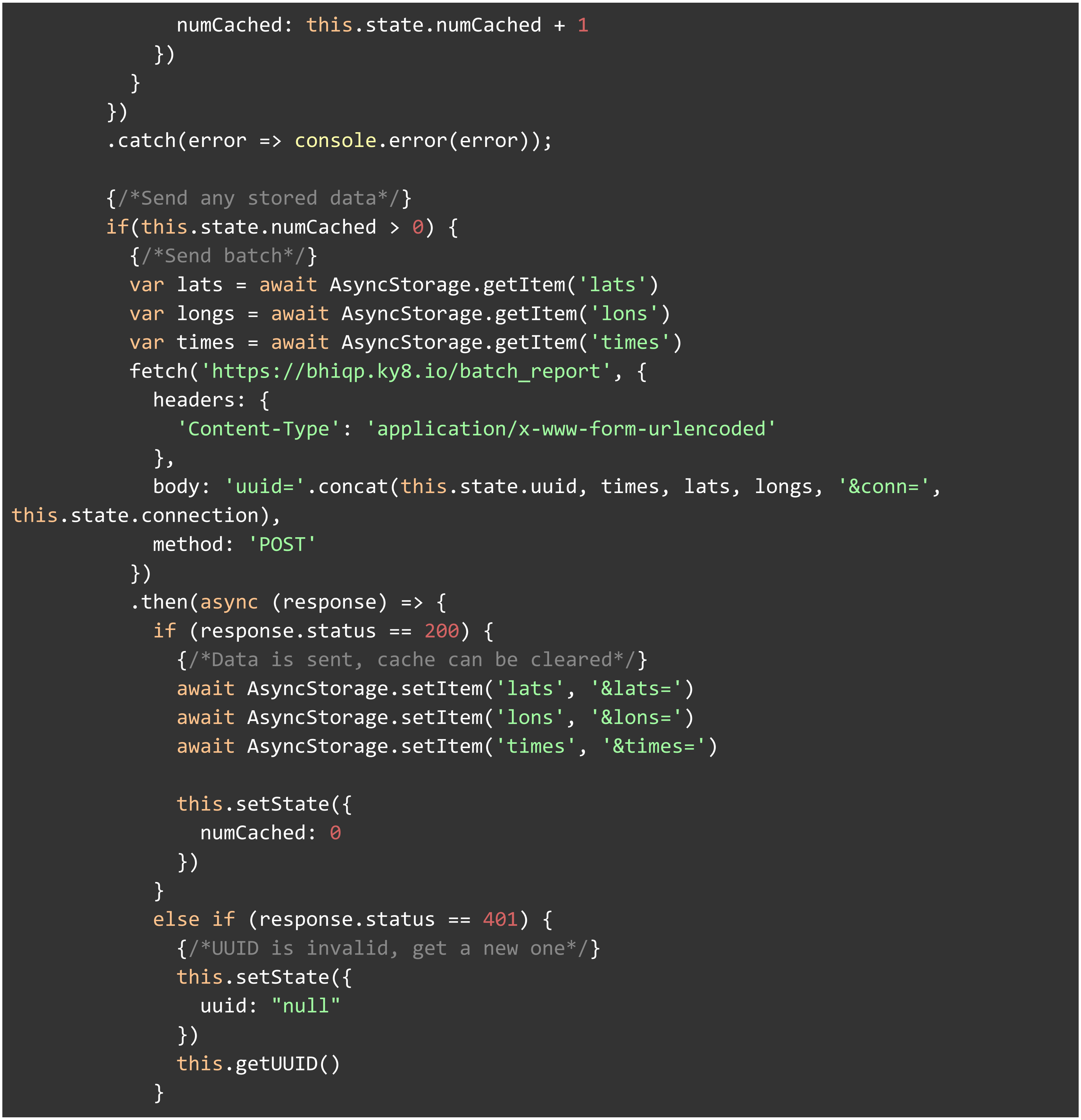
```
}

componentDidMount() {
  NetInfo.addEventListener('connectionChange', this._setConnection.bind(this));
  AppState.addEventListener('change', this._handleAppStateChange);
}

_setConnection(NetInfo) {
  this.setState({connection: NetInfo.type})
  console.log("big if true: ", NetInfo.type)
}

_handleAppStateChange = (newAppState) => {
  if (newAppState === 'active' || newAppState === 'background') {
    console.log("would handle")
  };
}

async initStorage() {
  await AsyncStorage.setItem('lats', '&lats=')
  await AsyncStorage.setItem('lons', '&lons=')
  await AsyncStorage.setItem('times', '&times=')
}

async getUUID() {
  {/*For testing purposes: AsyncStorage.removeItem("uuid");*/}

  var storedUUID = await AsyncStorage.getItem("uuid");
  if (storedUUID !== "null" && storedUUID !== null) {
    console.log("use pre-existing uuid")
    this.setState({uuid: storedUUID})
  }
  else {
    console.log("request new uuid")
    {/*Get UUID from server*/}
    await fetch('https://bhiqp.ky8.io/reg', {
      headers: {
        'Content-Type': 'application/x-www-form-urlencoded'
```



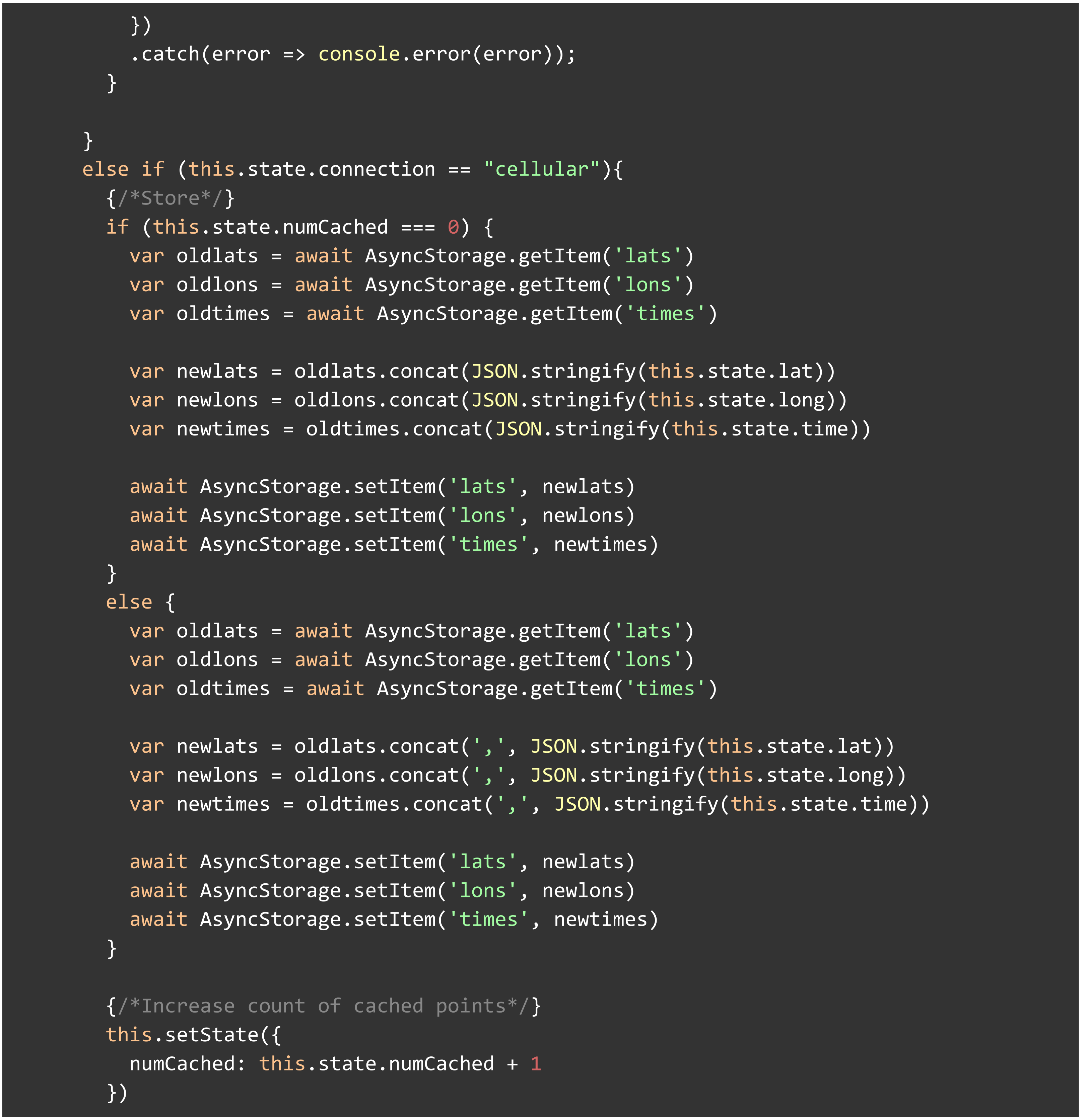
```
        },
        body: 'appkey=',
        method: 'POST'
      })
      .then((response) => response.json())
      .then((response) => this.setState({uuid: response.uuid}))
      .catch(error => console.error(error));

      await AsyncStorage.setItem('uuid', this.state.uuid)
    }
  }

  uuidret = this.getUUID();
  storageret = this.initStorage();

  watchID = BackgroundGeolocation.watchPosition(async (position) => {
    console.log("thisisissis: ", this.state.uuid)
    {/*Check is user in within park bounds and allows tracking*/}
    if ((this.state.isTracking) && (position.coords.longitude < -68.162249) &&
(position.coords.longitude > -68.432787)
        && (position.coords.latitude < 44.448252) && (position.coords.latitude >
44.218395)) {
      this.setState({
        lat: position.coords.latitude,
        long: position.coords.longitude,
        time: Math.floor(new Date().getTime() / 1000),
      });

      {/*Send data if phone has wifi*/}
      if (this.state.connection == "wifi") {
        fetch('https://bhiqp.ky8.io/report', {
          headers: {
            'Content-Type': 'application/x-www-form-urlencoded'
          },
          body: 'uuid='.concat(this.state.uuid,'&time=', this.state.time, '&lat=',
this.state.lat, '&lon=', this.state.long, '&conn=', this.state.connection),
          method: 'POST'
        })
```



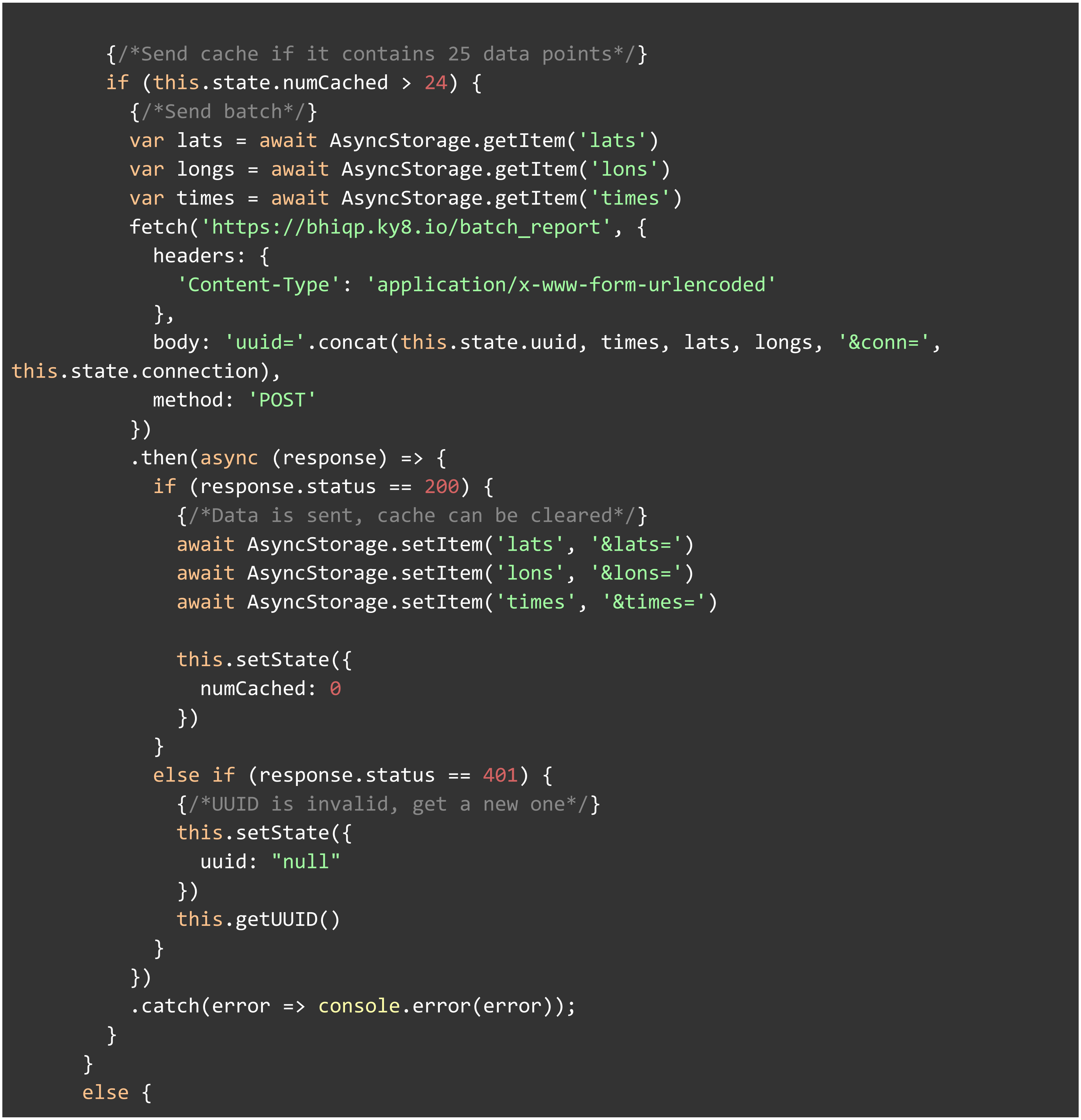
```
        .then(async (response) => {
          if (response.status == 401) {
            this.setState({
              uuid: "null"
            })
            this.getUUID()
          }
          if (response.status != 200 ) {
            {/*If data send failed, cache data to send later*/}
            {/*Store*/}
            if (this.state.numCached === 0) {
              var oldlats = await AsyncStorage.getItem('lats')
              var oldlons = await AsyncStorage.getItem('lons')
              var oldtimes = await AsyncStorage.getItem('times')

              var newlats = oldlats.concat(JSON.stringify(this.state.lat))
              var newlons = oldlons.concat(JSON.stringify(this.state.long))
              var newtimes = oldtimes.concat(JSON.stringify(this.state.time))

              await AsyncStorage.setItem('lats', newlats)
              await AsyncStorage.setItem('lons', newlons)
              await AsyncStorage.setItem('times', newtimes)
            }
            else {
              var oldlats = await AsyncStorage.getItem('lats')
              var oldlons = await AsyncStorage.getItem('lons')
              var oldtimes = await AsyncStorage.getItem('times')

              var newlats = oldlats.concat(',', JSON.stringify(this.state.lat))
              var newlons = oldlons.concat(',', JSON.stringify(this.state.long))
              var newtimes = oldtimes.concat(',', JSON.stringify(this.state.time))

              await AsyncStorage.setItem('lats', newlats)
              await AsyncStorage.setItem('lons', newlons)
              await AsyncStorage.setItem('times', newtimes)
            }
            {/*Increase count of cached points*/}
            this.setState({
```



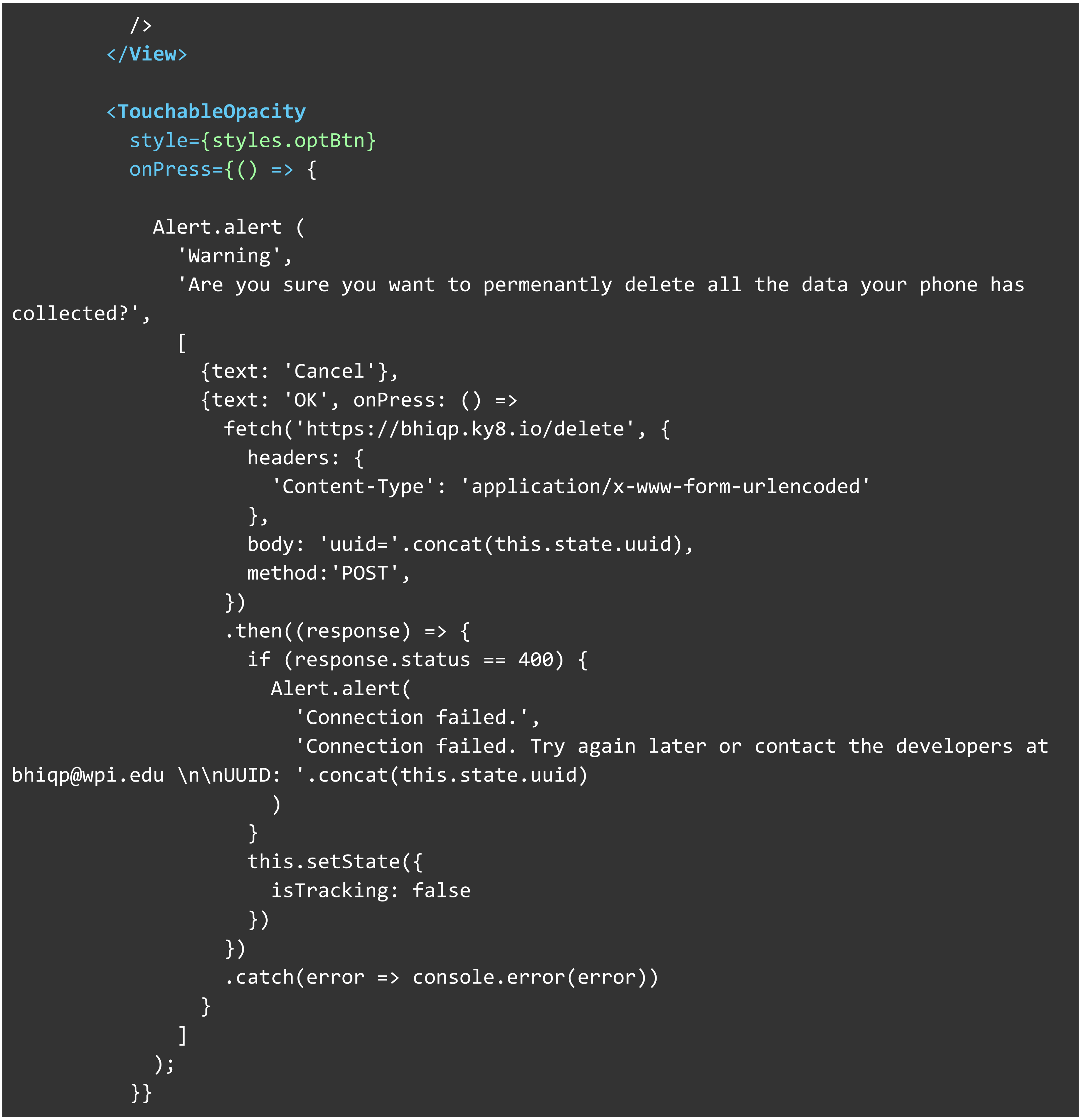
```
              numCached: this.state.numCached + 1
            })
          }
        })
        .catch(error => console.error(error));

        {/*Send any stored data*/}
        if(this.state.numCached > 0) {
          {/*Send batch*/}
          var lats = await AsyncStorage.getItem('lats')
          var longs = await AsyncStorage.getItem('lons')
          var times = await AsyncStorage.getItem('times')
          fetch('https://bhiqp.ky8.io/batch_report', {
            headers: {
              'Content-Type': 'application/x-www-form-urlencoded'
            },
            body: 'uuid='.concat(this.state.uuid, times, lats, longs, '&conn=',
this.state.connection),
            method: 'POST'
          })
          .then(async (response) => {
            if (response.status == 200) {
              {/*Data is sent, cache can be cleared*/}
              await AsyncStorage.setItem('lats', '&lats=')
              await AsyncStorage.setItem('lons', '&lons=')
              await AsyncStorage.setItem('times', '&times=')

              this.setState({
                numCached: 0
              })
            }
            else if (response.status == 401) {
              {/*UUID is invalid, get a new one*/}
              this.setState({
                uuid: "null"
              })
              this.getUUID()
            }
```



```
          })
          .catch(error => console.error(error));
        }

      }
      else if (this.state.connection == "cellular"){
        {/*Store*/}
        if (this.state.numCached === 0) {
          var oldlats = await AsyncStorage.getItem('lats')
          var oldlons = await AsyncStorage.getItem('lons')
          var oldtimes = await AsyncStorage.getItem('times')

          var newlats = oldlats.concat(JSON.stringify(this.state.lat))
          var newlons = oldlons.concat(JSON.stringify(this.state.long))
          var newtimes = oldtimes.concat(JSON.stringify(this.state.time))

          await AsyncStorage.setItem('lats', newlats)
          await AsyncStorage.setItem('lons', newlons)
          await AsyncStorage.setItem('times', newtimes)
        }
        else {
          var oldlats = await AsyncStorage.getItem('lats')
          var oldlons = await AsyncStorage.getItem('lons')
          var oldtimes = await AsyncStorage.getItem('times')

          var newlats = oldlats.concat(',', JSON.stringify(this.state.lat))
          var newlons = oldlons.concat(',', JSON.stringify(this.state.long))
          var newtimes = oldtimes.concat(',', JSON.stringify(this.state.time))

          await AsyncStorage.setItem('lats', newlats)
          await AsyncStorage.setItem('lons', newlons)
          await AsyncStorage.setItem('times', newtimes)
        }

        {/*Increase count of cached points*/}
        this.setState({
          numCached: this.state.numCached + 1
        })
```

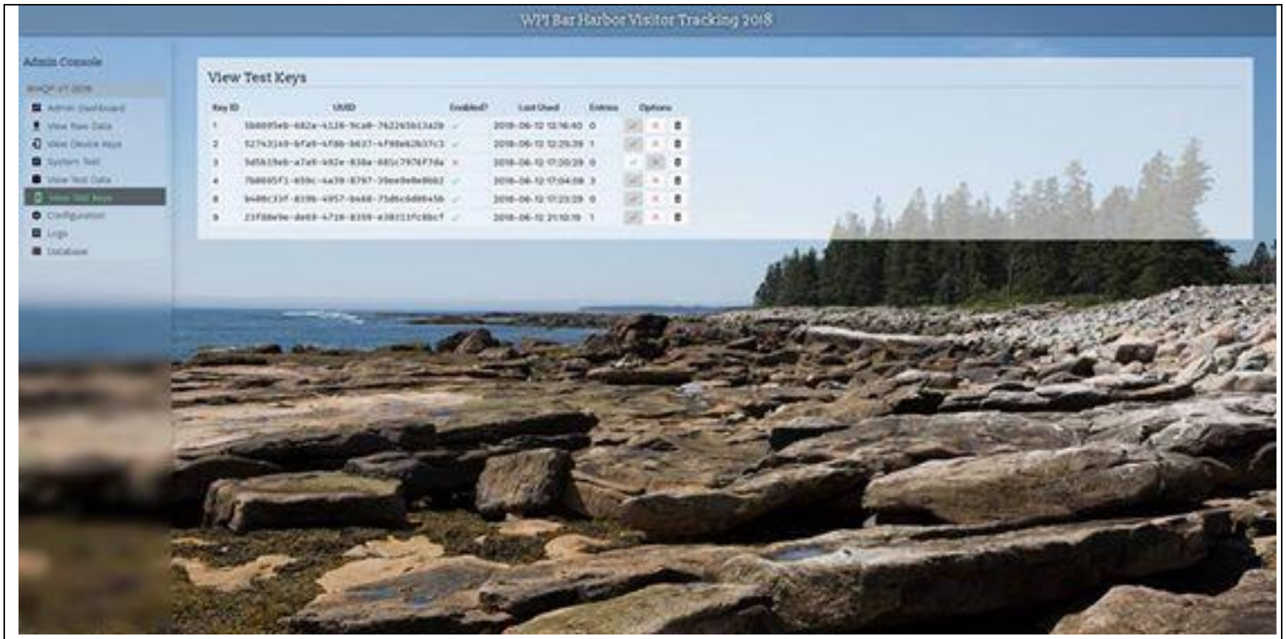


```
        {/*Send cache if it contains 25 data points*/}
        if (this.state.numCached > 24) {
          {/*Send batch*/}
          var lats = await AsyncStorage.getItem('lats')
          var longs = await AsyncStorage.getItem('lons')
          var times = await AsyncStorage.getItem('times')
          fetch('https://bhiqp.ky8.io/batch_report', {
            headers: {
              'Content-Type': 'application/x-www-form-urlencoded'
            },
            body: 'uuid='.concat(this.state.uuid, times, lats, longs, '&conn=',
this.state.connection),
            method: 'POST'
          })
          .then(async (response) => {
            if (response.status == 200) {
              {/*Data is sent, cache can be cleared*/}
              await AsyncStorage.setItem('lats', '&lats=')
              await AsyncStorage.setItem('lons', '&lons=')
              await AsyncStorage.setItem('times', '&times=')

              this.setState({
                numCached: 0
              })
            }
            else if (response.status == 401) {
              {/*UUID is invalid, get a new one*/}
              this.setState({
                uuid: "null"
              })
              this.getUUID()
            }
          })
          .catch(error => console.error(error));
        }
      }
      else {
        {/*Store*/}
        if (this.state.numCached === 0) {
          var oldlats = await AsyncStorage.getItem('lats')
          var oldlons = await AsyncStorage.getItem('lons')
          var oldtimes = await AsyncStorage.getItem('times')

          var newlats = oldlats.concat(JSON.stringify(this.state.lat))
          var newlons = oldlons.concat(JSON.stringify(this.state.long))
          var newtimes = oldtimes.concat(JSON.stringify(this.state.time))

          await AsyncStorage.setItem('lats', newlats)
          await AsyncStorage.setItem('lons', newlons)
          await AsyncStorage.setItem('times', newtimes)
        }
        else {
          var oldlats = await AsyncStorage.getItem('lats')
          var oldlons = await AsyncStorage.getItem('lons')
          var oldtimes = await AsyncStorage.getItem('times')

          var newlats = oldlats.concat(',',
JSON.stringify(this.state.lat))
          var newlons = oldlons.concat(',', JSON.stringify(this.state.long))
          var newtimes = oldtimes.concat(',', JSON.stringify(this.state.time))

          await AsyncStorage.setItem('lats', newlats)
          await AsyncStorage.setItem('lons', newlons)
          await AsyncStorage.setItem('times', newtimes)
        }

        {/*Increase count of cached points*/}
        this.setState({
          numCached: this.state.numCached + 1
        })
      }

    }
  },
  (error) => console.error(error),
  {interval: 1500, desiredAccuracy: 10, persist: false})

  render() {

    return (

      <View style={styles.container}>

        <View style={styles.statusBar}>
          <StatusBar/>
        </View>

        <View style={styles.headerView}>

          <Text style = {styles.headerText}>
            {'Acadia Traffic Solutions'}
          </Text>

        </View>

        <View style = {styles.infoPara}>
          <Text style={styles.text}>
            Using this application will help Acadia National Park officials decrease the
park's heavy traffic and
            make visiting the park with a vehicle a less stressful experience.
            We appreciate your support.
          </Text>
        </View>

        <View style = {styles.allowTracking}>
          <Text style={styles.text}>Allow Tracking</Text>
        </View>

        <View style = {styles.switchPosition}>
          <Switch
            style={styles.switch}
            value={this.state.isTracking}
            onValueChange={(value) => this.setState({isTracking: value})}
```



```
          />
        </View>

        <TouchableOpacity
          style={styles.optBtn}
          onPress={() => {

            Alert.alert (
              'Warning',
              'Are you sure you want to permenantly delete all the data your phone has 
collected?',
              [
                {text: 'Cancel'},
                {text: 'OK', onPress: () =>
                  fetch('https://bhiqp.ky8.io/delete', {
                    headers: {
                      'Content-Type': 'application/x-www-form-urlencoded'
                    },
                    body: 'uuid='.concat(this.state.uuid),
                    method:'POST',
                  })
                  .then((response) => {
                    if (response.status == 400) {
                      Alert.alert(
                        'Connection failed.',
                        'Connection failed. Try again later or contact the developers at 
bhiqp@wpi.edu \n\nUUID: '.concat(this.state.uuid)
                      )
                    }
                    this.setState({
                      isTracking: false
                    })
                  })
                  .catch(error => console.error(error))
                }
              ]
            );
          }}
          >
          <Text style={styles.optBtnText}>Delete Tracking Data</Text>
        </TouchableOpacity>

        <View style = {styles.bottom}>
          <Text style = {styles.bottomText}>
            App Icon credits: Nick Roach
          </Text>
        </View>

      </View>

    )

  }
}

AppRegistry.registerComponent('App', () => App);

const styles = StyleSheet.create({

  container: {
    flex: 1,
    backgroundColor: '#ffffff'
  },

  text: {
    color: '#000000',
    marginLeft: 0,
    fontSize: 20,
  },

  statusBar: {
    height: Platform.OS === 'ios' ? 20 : StatusBar.currentHeight,
    backgroundColor: '#4885ed'
  },

  headerView: {
    position: 'absolute',
    top: 0,
    left: 0,
    right: 0,
    height: 56,
    justifyContent: 'center',
    backgroundColor: '#4885ed'
  },

  headerText: {
    position: 'absolute',
    top: 15,
    left: 20,
    color: '#fff',
    alignContent: 'flex-start',
    fontSize: 20,
    justifyContent: 'center',
  },

  infoPara: {
    position: 'absolute',
    top: '15%',
    marginLeft: 20,
    marginRight: 20,
  },

  bottom: {
    position: 'absolute',
    top: '95%',
    marginBottom: 5,
    marginLeft: 20,
    marginLeft: 0
  },

  bottomText: {
    fontSize: Platform.OS === 'ios' ? 10 : 15,
    marginLeft: '7%'
  },
  allowTracking: {
    position: 'absolute',
    top: '60%',
    marginLeft: '7%',
  },

  switch: {
  },

  switchPosition: {
    position: 'absolute',
    top: '60%',
    right: '10%',
  },

  optBtn: {
    position: 'absolute',
    bottom: '15%',
    height: 60,
    left: 10,
    right: 10,
    backgroundColor: '#db3236',
    alignItems: 'center',
    justifyContent: 'center',
    borderRadius: 45
  },

  optBtnText: {
    justifyContent: 'center',
    alignItems: 'center',
    fontSize: 20,
    color: '#ffffff',
  }
})
```

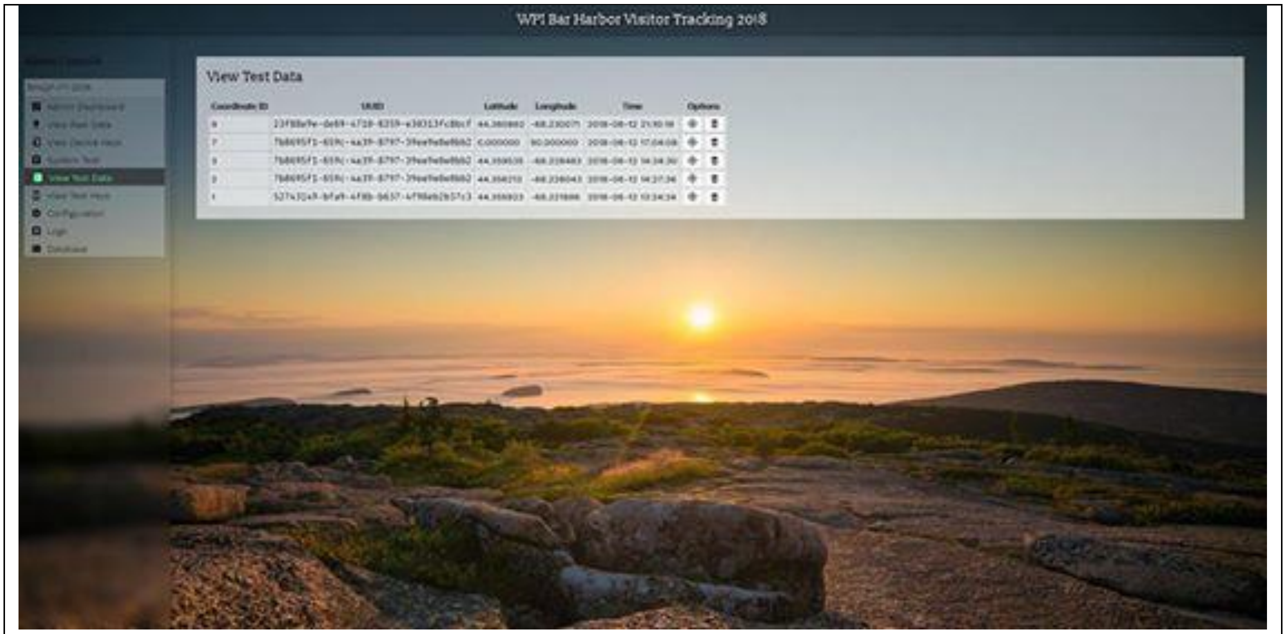
Appendix D



View Test Keys

Key ID	UUID	Enabled?	Last Used	Entries	Options
1	5b6695eb-682a-4326-9ca8-762265613a2b	✓	2018-06-12 12:16:40	0	✓ ✗ 🗑
2	527432a9-bfa9-4f8b-b637-4f98eb2b57c3	✓	2018-06-12 12:25:39	1	✓ ✗ 🗑
3	5d5b19eb-a7a8-482e-838a-681c7976f7da	✗	2018-06-12 17:20:29	0	✓ ✗ 🗑
4	7b8695f1-659c-4a39-8797-39ee9e8e8b62	✓	2018-06-12 17:04:08	3	✓ ✗ 🗑
8	b488c33f-8396-4957-8468-75d6c6d8b456	✓	2018-06-12 17:23:29	0	✓ ✗ 🗑
9	23f88e9e-de69-4728-8359-e38313fc8bcf	✓	2018-06-12 21:10:19	1	✓ ✗ 🗑

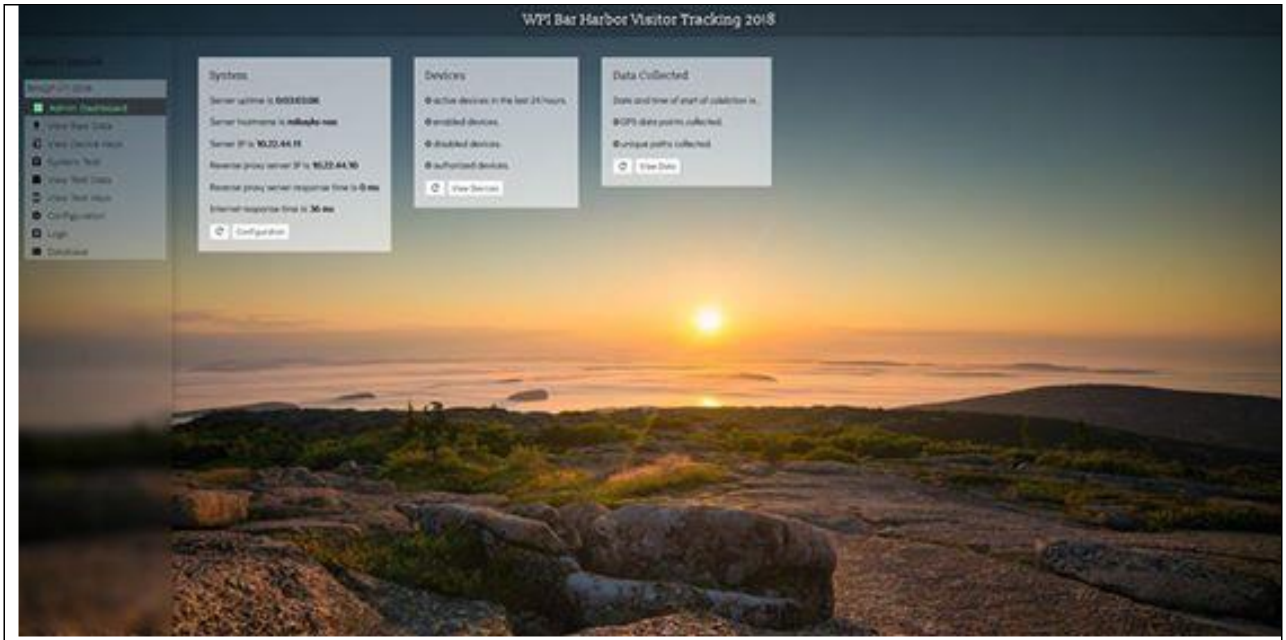
Figure 20: Device Keys Admin View



View Test Data

Coordinate ID	UUID	Latitude	Longitude	Time	Options
8	23f88e9e-de69-4728-8359-e38313fc8bcf	44.360862	-68.230071	2018-06-12 21:10:18	⌖ 🗑
7	7b8695f1-659c-4a39-8797-39ee9e8e8b62	0.000000	90.000000	2018-06-12 17:04:08	⌖ 🗑
3	7b8695f1-659c-4a39-8797-39ee9e8e8b62	44.359535	-68.228463	2018-06-12 14:34:30	⌖ 🗑
2	7b8695f1-659c-4a39-8797-39ee9e8e8b62	44.358213	-68.238043	2018-06-12 14:37:36	⌖ 🗑
1	527432a9-bfa9-4f8b-b637-4f98eb2b57c3	44.358923	-68.221888	2018-06-12 12:34:34	⌖ 🗑

Figure 21: Device Data Admin View



System

Server uptime is 0:03:03:06

Server hostname is mikayla-nas

Server IP is 10.22.44.11

Reverse proxy server IP is 10.22.44.10

Reverse proxy server response time is 0 ms

Internet response time is 36 ms

Devices

0 active devices in the last 24 hours.

0 enabled devices.

0 disabled devices.

0 authorized devices.

Data Collected

Date and time of start of collection is .

0 GPS data points collected.

0 unique paths collected.

Figure 22: Server Info Admin View