Department of Electrical, Computer, and Software Engineering

Part IV Research Project

Final Report

Project Number: 99

Airbnb for Parking

Aiden Burgess

Sreeniketh Raghavan

Project Supervisor: Andrew Meads

Date: 14/10/2021

Declaration of Originality

This report is my own unaided work and was not copied from nor written in collaboration with any other person.

Signature: [illegible]

Name: Aiden Burgess

ABSTRACT: The shared economy represents a new system for utilising existing assets. Platforms such as Uber and Airbnb utilise cars and apartments to generate income for their owners. However, there is currently no dominant platform for rental parking. This represents an opportunity in the shared economy space for a short-term rental parking solution. In this project we aim to create an MVP which investigates and explores the rental parking space by creating a mobile compatible application. The literature review focuses on pricing, mobile technologies, and existing applications. The implications of this existing research are discussed to produce desirable features and possible improvements above existing solutions. Finally, the areas of research are defined and a project timeline to produce the MVP is shown.

1. Introduction

The idea for this project was formed during a long search for parking on the edge of the Auckland CBD. Even the side streets in the suburbs had no spaces, but the driveways were empty. The theory of this idea is that owners of the driveways would be willing to rent them for money. Despite the perceived demand, no parking sharing platform has become ubiquitous in New Zealand or globally.

This project aims to explore the possibility of a short-term rental parking solution as a part of the shared economy. The shared economy has been widely discussed as a new system for utilising existing assets. This new system utilises the assets of individuals and communities, to create income for their owners. Examples of these assets include cars (Uber) and homes (Airbnb). In this case, the asset is a parking space. Parking is a necessity for individuals and businesses. However, it is often not available in the right place, at the right time. In this project we aim to create an MVP which investigates and explores the rental parking space by creating a mobile compatible application.

"The sharing economy differs from the traditional business model in its dominant reliance on the internet platform, its non-ownership of assets, its ability to access idle resources, its lower prices with more customized products/services and its non-conventional workforce as product/service providers" [1] [2].

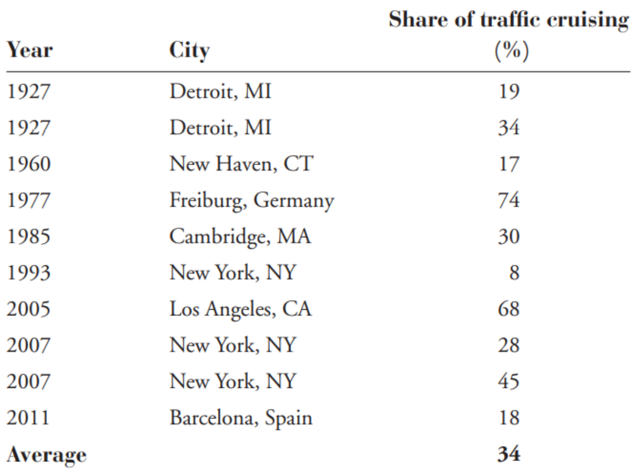
In the literature review section, we will summarise the existing literature on aspects related to developing a rental parking mobile application. These include pricing, mobile application development frameworks, technologies to implement mobile applications, mobile map technology, usability of mobile applications, and existing applications.

We then discuss why our application will be mobile compatible, what features will be included in the application, and how the application will be different to existing applications. Finally, the statement of research intent will be presented, along with a planned timeline of the project.

2. Literature Review

2.1. Pricing

Ten studies conducted in eight cities between 1927 and 2011 found that an average of 34% of cars in congested downtown traffic were cruising for parking (Fig. 1.) [3] [4]. This represents a large pricing issue in the parking industry.



Year	City	Share of traffic cruising (%)
1927	Detroit, MI	19
1927	Detroit, MI	34
1960	New Haven, CT	17
1977	Freiburg, Germany	74
1985	Cambridge, MA	30
1993	New York, NY	8
2005	Los Angeles, CA	68
2007	New York, NY	28
2007	New York, NY	45
2011	Barcelona, Spain	18
Average		**34**

Fig. 1. Cruising information for cities *[3]*

The price of a parking space is not just the price of a parking but should also include waiting time (cruising time) and walk time to destination. "This waiting cost is the time drivers spend circling the block searching for an open space." [4].

It is important to optimise pricing as there are negative consequences when parking is both under-priced and overpriced [4]. When parking is under-priced, many people are willing to park, but there are not enough spaces, so this can increase cruise times. It also means that the revenue is not being maximised. When parking is overpriced, "curb spaces remain empty, nearby stores lose potential customers, employees lose jobs, and governments lose tax revenue" [4].

In 2011, San Francisco switched from a static price per hour parking to prices which "vary by time of day and from block to block." [4]. Philip Goodwin proposed two strategies based on price and quantity. One strategy was to change the price of the parking to reduce traffic. Another was to have a target traffic level and adjust prices to achieve that target [5].

Dynamic pricing is changing the pricing of a product or service over time, across consumers, or across products/services. Gibbs et al found that “dynamic pricing has been adopted by some of the leading sharing economy firms and has become an integral part of the sharing economy culture” [6].

Airbnb recently released an AI powered pricing system which considers unique features of the listings and offers a price for each date in the future [6]. However, it gives users flexibility to also choose a price higher or lower than recommended. This contrasts with Uber/Lyft where they have spot pricing which cannot be changed.

Disruptive products usually have lower pricing (e.g. Airbnb, Uber, Lyft) and other benefits, as they often lack some aspects of the traditional products. An example of this is hotels have staff whereas Airbnb listings do not.

There are two main types of owners: professional and non-professional [6]. Professional owners usually own more than one property and have greater returns and occupancy rates. Nonprofessional owners are less likely to change the price of their property to match demand.

Rob Stock identified three participants in the sharing economy: sharing opportunists, “those driven by economic necessity”, and finally commercial/professional participants. Sharing opportunists already own the asset and use the opportunity to make extra income [7] [8].

“Choudary and Parker use the Uber example to explain why platforms are so powerful: they eliminate gatekeepers, unlock new supply and demand and create community feedback loops. Uber, for example, performs a matching service that serves as a virtuous cycle. More demand is met by more opportunistic drivers, which increases geographic coverage, which leads to faster pickups, which encourages more customers to join the platform and more people to sign up as drivers. Driver downtime is lowered and so are prices, which leads to more scale.” [8] [9].

2.2. Mobile Application Development Frameworks

Although many standard development frameworks for an application exist such as Waterfall, Scrum or Agile, Vithani et al. argues that there is a need for a new unique mobile application development lifecycle mode. They reason that because mobile application development involves complex functionality and services like telephony services, location-based services and different connectivity modes, a specific framework should be used to develop mobile applications [10].

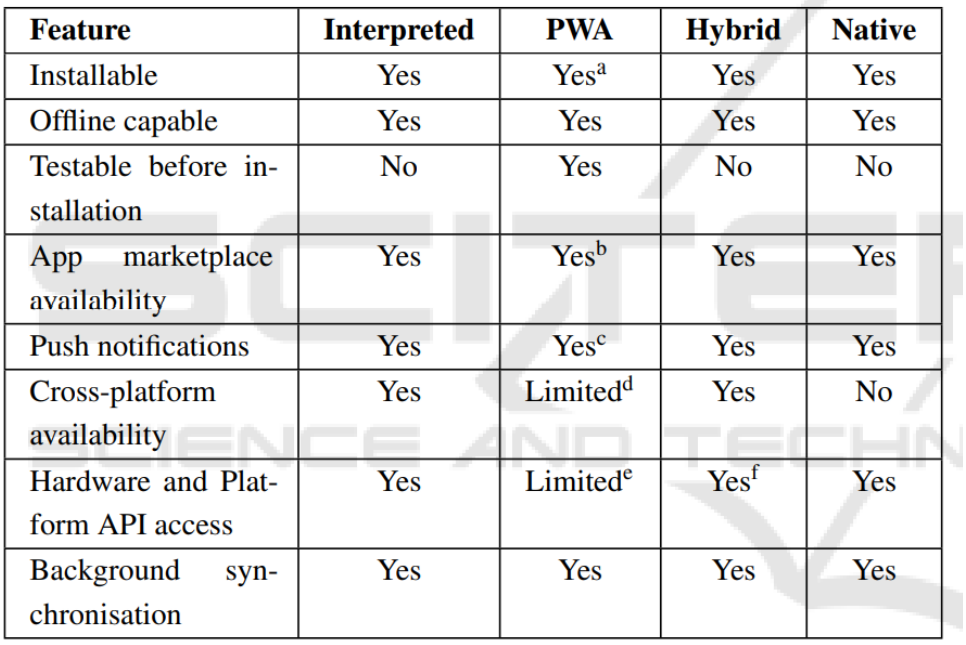
They identify some key differences between mobile and desktop application development. Mobile devices have a shorter lifespan [10]. Mobile applications have more complex functionality involving telephone, camera, GPS. There is a difference in physical interfaces. Desktops use keyboards, mice, touch screens, and other external devices while mobile devices only allow touch panel and keyboard.

Mobile devices also have smaller screens, so they need to use a layered UI and the functionality design also needs to be optimised for the smaller screen [10]. There is also a need to optimise usage of battery and memory in phones.

Therefore, they proposed a new framework called the MADLC Process. This process consists of the following phases: identification, design, development, prototyping, testing, deployment, and finally maintenance.

2.3. Technologies to Implement Mobile Applications

Bjorn-Hansen et al. investigated the features and performances between different cross-platform development frameworks (Fig. 2.) [11]. Among the cross-platform approaches, hybrid applications lagged in both launch time and time to render. The interpreted application had a considerably larger size compared to the other applications, and PWAs performed the best in both size and launch time (Fig. 3.).

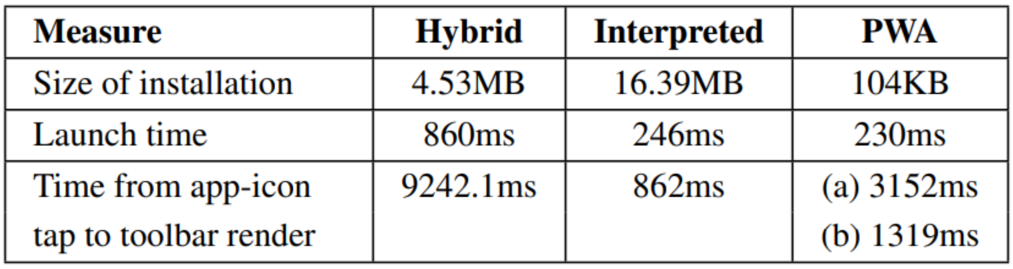


Feature	Interpreted	PWA	Hybrid	Native
Installable	Yes	Yes[a]	Yes	Yes
Offline capable	Yes	Yes	Yes	Yes
Testable before installation	No	Yes	No	No
App marketplace availability	Yes	Yes[b]	Yes	Yes
Push notifications	Yes	Yes[c]	Yes	Yes
Cross-platform availability	Yes	Limited[d]	Yes	No
Hardware and Platform API access	Yes	Limited[e]	Yes[f]	Yes
Background synchronisation	Yes	Yes	Yes	Yes

Fig. 2. Feature-comparison of cross-platform approaches *[11]*

Traditionally, web, native, and mobile platform code have been non-interoperable, so there has been no reusability of code [11]. Cross platform development has been popular alternative to reduce resources and need for specialized mobile developers. Reduces dev effort and faster time to market.

The hybrid frameworks Ionic and PhoneGap structure components using HTML and CSS [11]. The interpreted framework React Native uses native interface components. Another interpreted framework Xamarin uses cross-



Measure	Hybrid	Interpreted	PWA
Size of installation	4.53MB	16.39MB	104KB
Launch time	860ms	246ms	230ms
Time from app-icon tap to toolbar render	9242.1ms	862ms	(a) 3152ms (b) 1319ms

Fig. 3. Measurement-comparison of cross-platform approaches *[11]*

compilation to compile C# into native binaries supported on each platform, so the apps are not dependant on interpreters or web views.

Progressive Web Apps (PWAs) were proposed by Google Web Fundamentals group to bridge gap between mobile and web by introducing features such as offline support, background synchronisation, and home-screen installation [11]. These new functionalities are supported by service workers. However, from their literature review they found there was little academic research on PWAs and for service workers. Currently PWAs are not able to access all hardware and platform features. E.g. calendar and contact list, however new APIs are becoming available to achieve these functionalities.

Charland and Leroux discuss the advantages and disadvantages of native and web code. Native code is usually compiled, “which is faster than interpreted languages such as JavaScript” [12]. This means that the application “paint[s] pixels directly on a screen through proprietary APIs and abstractions for common user-interface elements and controls” [12]. User interfaces are created in webviews utilising HTML and CSS. This leads to a lower performance overall than native code.

However, the benefit of web code is that it is compatible on multiple operating systems, which means that expensive developer time is not repeated, and the code is easier to maintain as there is only one code base. Furthermore, it is often faster and easier to write code for the web.

2.4. Mobile Map Technology

Maps can impact user cognitive load. Small displays and limited interaction capabilities often make mobile map-based systems difficult to design and frustrating to use [13]. To make a more usable map, reduce clutter and assist users in finding information they need. Also make relevant information easy to see via contrast to improve usability. Mobile maps need to deliver real time content and support real time interaction. GPS can deliver map content relevant to users personal and situational context. Map interaction: zooming, panning, point of interest selection

Church et al suggest that map and text-based interfaces should be used in different contexts [14]. For their research they implemented two local community Q/A mobile applications with the same functionality but with different user interfaces, one map-based and one text-based. They found that users “produced more queries through the map interface”, but “retrieved content more often” and “answered queries more often” through the text interface.

Although they discovered that participants 47% of participants preferred the map-based application and 41% preferred the text-based application (with 12% expressing no preference), the researchers concluded that a hybrid solution is ideal. “In this way, the speed of navigation that a text-based interface can offer can be coupled with the overview and sense of place that a map can provide” [14]. This was not just an idea that was discussed by the researchers, but some participants also suggested a hybrid model. An example quote from a user is: “I like to start with the map to get a overview and move to the text version afterwards” [14].

2.5. Usability of Mobile Applications

Harrison et al. suggested a new usability model for mobile applications, as two widely used frameworks: Nielson and ISO were “derived from traditional desktop applications” [15]. Issues that the standard frameworks do not address are mobile context, connectivity, small screen size, different display resolution, limited processing capability and power, and data entry methods.

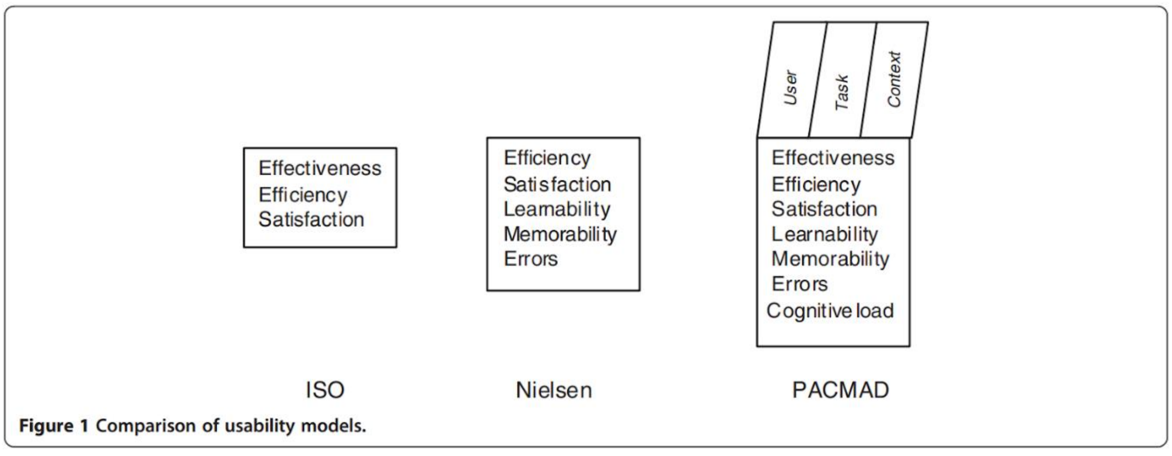


Fig.4. PACMAD usability model *[15]*

Their new model PACMAD (People At the Centre of Mobile Application Development) “aims to address some of the shortcomings of existing usability models when applied to mobile applications” [15]. The PACMAD model combines the Nielson and ISO models into one framework to evaluate usability (Fig. 4.). This model has three factors of usability: user, task, context of use. The model evaluates usability with respect to seven attributes: effectiveness, efficiency, satisfaction, learnability, memorability, errors, and cognitive load.

2.6. Existing Applications

Existing solutions for rental parking in New Zealand are Parkable, Sharedspace, and Anyspace. Sharedspace focuses on event, office and parking spaces, however the parking is rented on a monthly basis. Anyspace is like Sharedspace, offering office, storage and parking spaces rented on a weekly basis. Parkable is focused on short term rental parking spaces (hourly or daily), with some support for long term options.

Therefore, Parkable is the most similar existing application to this project. It also focuses on a mobile interface, with the desktop website not containing functionality except for pointing to the mobile application. "Parkable is an excellent example of a home-grown app-based disruptive transaction platform" [8].

3. Discussions

From the literature review, it was found that there is a lack of research on the impacts and implications of a shared economy version of parking. Much research has been conducted on the successes of Uber and Airbnb, however a dominant parking application has not fully disrupted the parking industry yet, which explains the lack of discussion in this area.

A similar feedback loop to Choudary and Parker's example of Uber can be theorised for a rental parking application. More demand is met by more opportunistic parking space owners, which increases geographic coverage of parking spaces, leading to closer parking spaces to destinations, encouraging more customers and owners to join the platform.

Min et al. examined the features of Uber and found "adding the unique features into the Uber mobile application, such as providing more accurate GPS location information on the map" and more specific search terms and refinement options "would help to enhance users' perceived advantages of using the Uber mobile application" [Min2018]. The rental parking application should provide similar functionality around GPS location accuracy and search refinement.

Features that they found to increase observability were "information about the requested service such as estimated time of arrival, cost for the trip" and "transaction history" [16].

The MADLC development process proposed by Vithani et al. does not appear unique. The phases of the process are very similar to those of the Waterfall process. The justification of a new process for mobile application development did have some valid points but was lacking and not clear in others. For example, "complex functionality" is not unique to mobile applications, and many desktop applications are now also utilising GPS, camera, and voice sensors. Furthermore, the preliminary results were weak, the article claims the process helped developers to plan and execute more effectively, but with no reasoning or evidence [10].

An important measure of usability is how much bandwidth and data an application uses [13]. To measure these attributes, the average mobile download/upload speeds in NZ should be used to benchmark the application.

Unlike other parking applications, we will experiment with a time slicing pricing system, where instead of being charged each hour, the user will be charged every 15 minutes instead. This adds value to the user by providing more perceived value as they are not charged as much for time they do not use. This is also a pricing advantage compared to traditional parking systems.

Another area where our application can differentiate from existing applications such as Parkable is by not relying solely on a map-based interface. Instead, this should be combined with text-based components. Church et al. concluded that hybrid-based applications are ideal as they combine advantages from both systems in terms of ease of use and aesthetics [14].

4. Statement of Research Intent

The intent of this research project is to produce a minimum viable product (MVP) of a rental parking system for the New Zealand Market. This MVP should support a mobile user interface to allow users to rent a parking space in New Zealand.

By exploring the implementation of such a product, we hope to explore whether problems in the parking industry can be solved using a shared economy proposition. These problems are sparse parking space, high cruising time, excess parking at residential properties due to parking mandate by council and decreasing parking space per capita in the Auckland CBD [17].

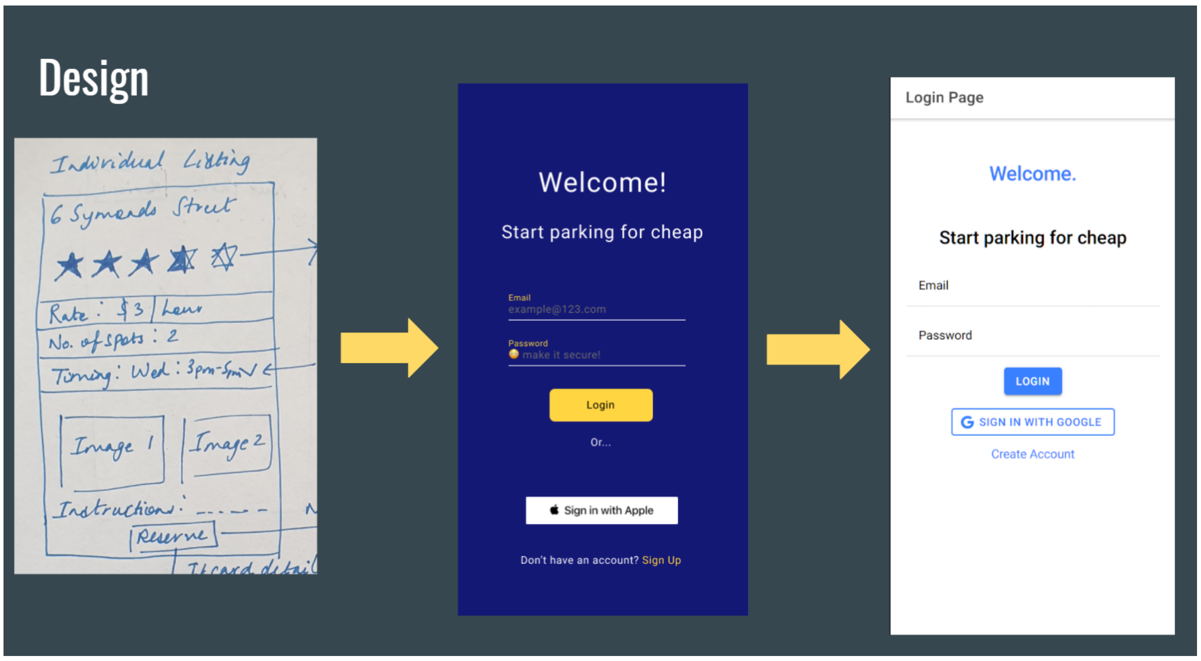
There are also technical areas of research which will be analysed. The mobile application framework (interpreted, hybrid, native, or PWA) will be determined as well as the larger tech stack to fit user requirements that are discovered. Target demographics will also be determined to be used for generating specific usability requirements.

The MVP will be produced as an example to demonstrate these learned concepts. To determine the usability and usefulness of the application, usability heuristics will be analysed, and it will be evaluated via user testing with target demographics.

There can also be an investigation of the benefits and disadvantages of time slice pricing, where the granularity for parking time is reduced from one hour. Measurements of demand and user preference can be used to determine the optimum time slice.

5. Design

We developed initial designs for Airpark on paper and whiteboard, and transitioned into hifi prototypes on figma. After some casual discussion with some friends, we found that the default styling of our framework was very appealing so we decided to use that and focus on functionality instead of theming. The default theming of the application is expanded on in Section X.X (Ionic).



5.1. Lofi Prototyping

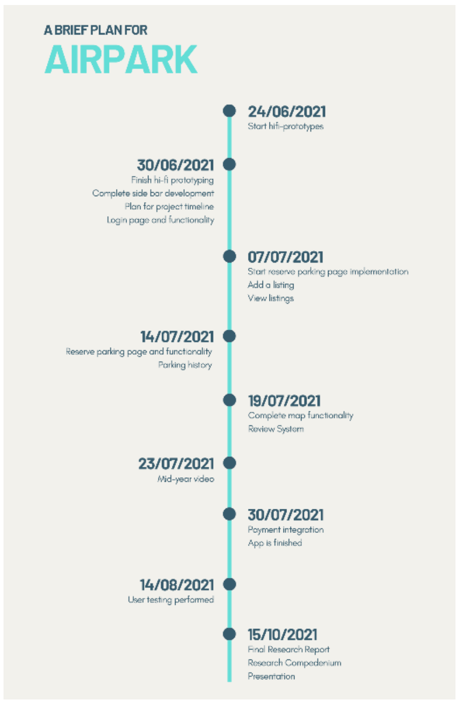
When developing lofi prototypes, we focused on functionality and responsiveness. Starting with lower fidelity designs allowed us to change our designs easily and discuss alternatives. An example of change made in the lofi stage was the add listing page being separated into separate pages and having a custom button for time availability which allowed for more fine detail tuning of parking times.

5.2. Hifi Prototyping

Hifi designs were modelled on Figma [1 in the top ref to Figma]. In this stage, we created components for our application, such as buttons, inputs and typography. These were used to construct example pages for AirPark, which were the sidebar, login page, and add listing page.

6. Implementation

The implementation of the project occurred over a period of approximately eighteen weeks. In late July, an initial scope of the project was established, with deadlines for functionalities (Fig X.).

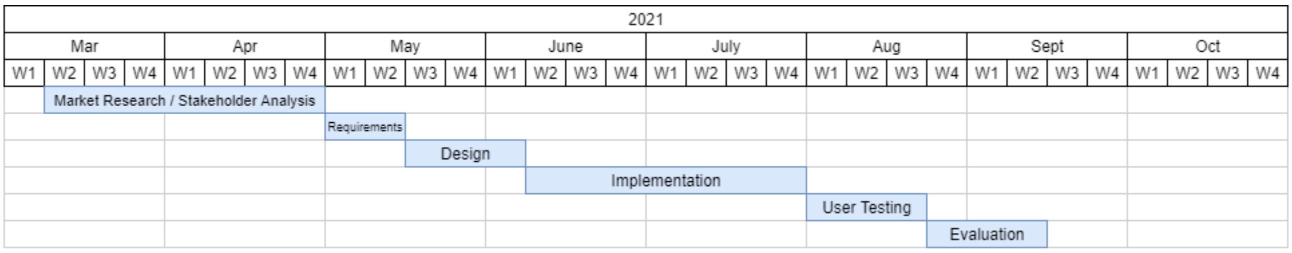


6.1. Scope

We defined the scope of our project in mid-July, around when we started the implementation of the project. We have completed all our major goals, which were hifi prototyping, login functionality, adding a listing, viewing listings, parking functionality, parking history, map functionality, and user testing. However, we did not complete the review system and payment integration for the project due to time constraints.

6.2. Development timeline

Overall, the project is expected to take 25 weeks in total, ending in the second week of September. The outline of the project timeline (Fig. X) is a rough idea of the steps needed and an estimate of each stage. It should be noted that each stage may overlap in the actual timeline. The stages that were identified as necessary for this project were: market research/stakeholder analysis, requirements, design, implementation, user testing and evaluation.



2021																															
Mar				Apr				May				June				July				Aug				Sept				Oct			
W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4
	Market Research / Stakeholder Analysis	🟦	🟦	🟦	🟦	🟦	🟦																								
								Requirements	🟦																						
										Design	🟦	🟦																			
													Implementation	🟦	🟦	🟦	🟦	🟦	🟦												
																				User Testing	🟦	🟦									
																							Evaluation	🟦	🟦						

Start of implementation: June 13th

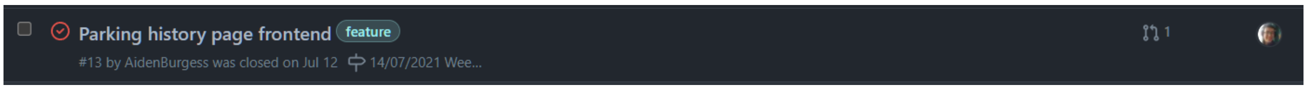
End of implementation: October 5th

Around 17 weeks of implementation

6.3. Development Process

Our development process was determined at the start of the implementation process. First, we create a GitHub issues, specifying the task to be completed, the type of task, and the deadline it should be completed by (Fig X.). The tasks to be completed and their deadlines were derived from our planned timeline (Fig X.). Next, a team member assigns an issue to themselves. They complete the feature on a new branch, and create a pull request. This pull request needs to be reviewed by the project partner before continuing this process. The reviewer checks that the code compiles, that the task is completed, and that the code style and quality is consistent with the existing code. Often, there are discussions in this stage of changes to increase usability. After these suggestions are resolved, the branch is merged into the main branch. These changes were automatically deployed on our website via Netlify.

The benefits of using this process are many. First, it is very easy to follow. It is very clear what you are expected to do and how you are expected to do it. Second, it allows the team members to work independently. They are given clear tasks to complete, and they know they will be held accountable for meeting deadlines. Third, it is very effective at minimizing the time it takes to complete a task. The team members can work on their tasks without any distractions.



Parking history page frontend feature

#13 by AidenBurgess was closed on Jul 12 14/07/2021 Wee...

1

6.4. Technologies

Neither of us was familiar with Ionic and firebase, and we decided to learn new technologies through this project. Typescript provided added type security, which helped reduce errors.

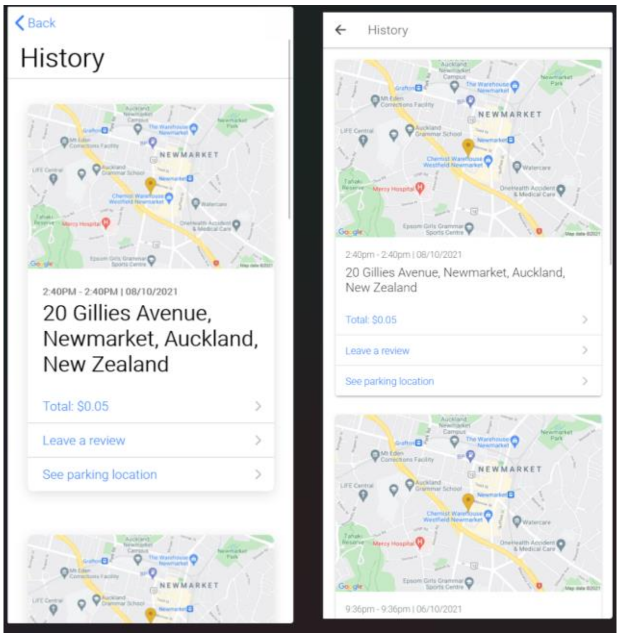
Ionic made it possible for the application to be compatible with and rendered differently on different platforms platforms including web, Android and iOS without having to write the code multiple times. Originally, we strongly considered Flutter to develop our application, however this option was eliminated because the map component was not compatible with desktop applications.

Firebase cloud functions were used as part of the backend. Using cloud functions meant that we could easily deploy the backend API using firebase's hosting feature without the need for a separate hosting service. Firebase is optimised to be used for mobile app development, and its integrated user authentication made it an appealing choice for the backend.

6.4.1. Ionic and Capacitor

Ionic[1] is a framework for building a application which supports many platforms. It provides a library of components to create an application which changes its design based on the platform. For example in Fig X. the differences between iOS and Material Design can be seen. The header text is larger and a different font on iOS, and the card has a different shadow as well.

Capacitor[2] complements Ionic by cross compiling the Ionic application onto different platforms. The application can be compiled to support Web, Android, iOS, PWA, and Electron.


Back
History
2:40PM - 2:40PM | 08/10/2021
20 Gillies Avenue, Newmarket, Auckland, New Zealand
Total: $0.05
Leave a review
See parking location
History
2:40pm - 2:40pm | 08/10/2021
20 Gillies Avenue, Newmarket, Auckland, New Zealand
Total: $0.05
Leave a review
See parking location
9:36pm - 9:36pm | 06/10/2021

6.4.2. React[3] and Typescript

Ionic is framework agnostic, it is compatible with many other frameworks. Plain JavaScript, React, Angular and Vue were possible choices for this project. Traditionally, Ionic used to be used with Angular, however the team was more familiar with React, so we chose React as our framework.

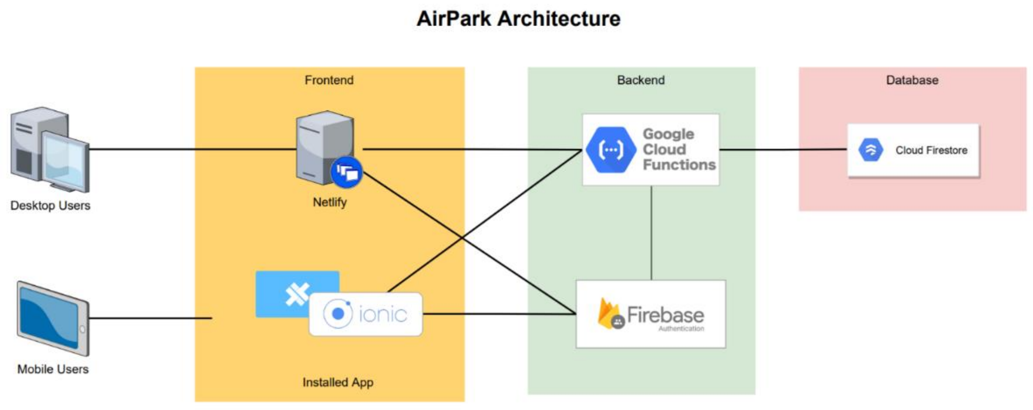
Why did we choose to use Typescript?

6.4.3. Firebase

Cloud functions

Authentication

Firestore Storage



6.5. Time Slice Pricing

Time slice pricing is when the granularity of the parking time is reduced, instead of charging by the hour, the granularity is reduced to a few minutes. Time slice pricing is not commonly used in the parking industry because the need to price the parking by the hour is important to manage the parking spaces. With time slice pricing, it is more likely that the spaces will be under-priced, which means there will be more demand for spaces. This can be seen in the parking closest to the doors of businesses where the time slice is lower than the cost of parking in the CBD.

Requirements doc (market research stuff), project implementation, features decided based on requirements, design, architecture frontend api database, user testing, changes incorporated, tools technologies, process (agile) (peer programming, sprints)

7. User Testing Methodology

Our user tests had two major parts. The first was scenario driven exploration of the application. In this section, the interviewer gave open ended prompts to the user. This allowed us to discover bugs, and specific comments about pages and components. The second part was the user feedback form, where we gather specific feedback and impressions about the application.

7.1. User Testing Session

The prompts were open ended, designed to allow the user to achieve a desired outcome, rather than follow a set of instructions. This allowed us to observe variations in how the users responded to different situations and what their expectations were compared to other users. If needed, we would provide a nudge to the user of how to perform the action. These sessions were recorded with the consent of the user, allowing us to replay certain moments to analyse the user's reaction.

Unfortunately, due to the pandemic, all the user testing sessions bar one were held online. Participants were asked to have their cameras on so that we could see their reactions to the scenarios. Another downside was that we could not have the users use an actual mobile application. Therefore, we asked users to use Chrome Developer Tools to simulate a mobile device in their browser.

There were four workflows in the user testing session. Users were asked to: creating a new listing as an owner, booking a parking spot, analysing the history of their previous park, and exploring UI differences across iOS compared with Android.

7.2. User Feedback Form

After completing the user testing session, the user was asked to fill out the feedback form. This was done immediately after the testing session to ensure that the experience was recent in their minds. The feedback form contained both open-ended questions such as 'What, if anything, surprised you about the experience?' and Likert scale prompts like 'I found it difficult to find information about my parking location or booking'.

The Likert scale questions allow for numerical analysis of the results and we used some standardised questions to allow for comparison to similar systems and future and current results. The open-ended questions were used to understand how respondents felt about Airpark's concept, usability, and aesthetics.

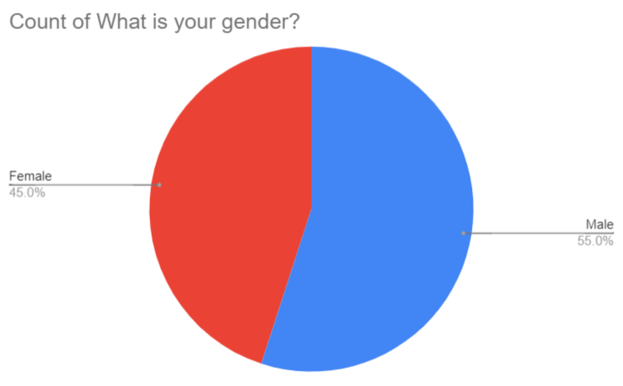
8. User Testing Findings

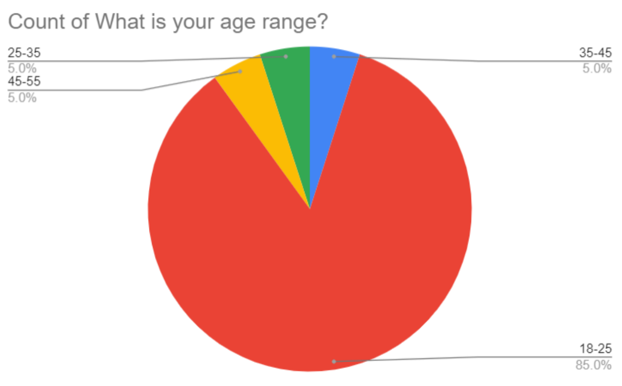
The testing took place over the course of seven weeks. Due to time constraints with the project, the user testing phase and the fixes and improvements phase overlapped. Three questions were added to the user feedback form after user testing had begun, so those questions have less data.

8.1. Participant Demographics

Our participants were mostly peers from the University of Auckland. Therefore, the ages of our respondents were quite homogenous with 85% between 18-25 years old (Fig X.). The gender distribution of our participants was slightly male skewed with 55% of our respondents being male and 45% identifying as female.

Fourteen respondents answered questions about whether they drove and whether they owned a domestic parking space which they would be willing to rent out for short durations. 50% (7) of respondents drove, and four respondents owned a parking space, although one was unwilling to rent out that parking space. Out of the seven respondents that drove, five of them strongly agreed that the concept of a rental parking platform appealed to them.





8.2. Overall Impressions

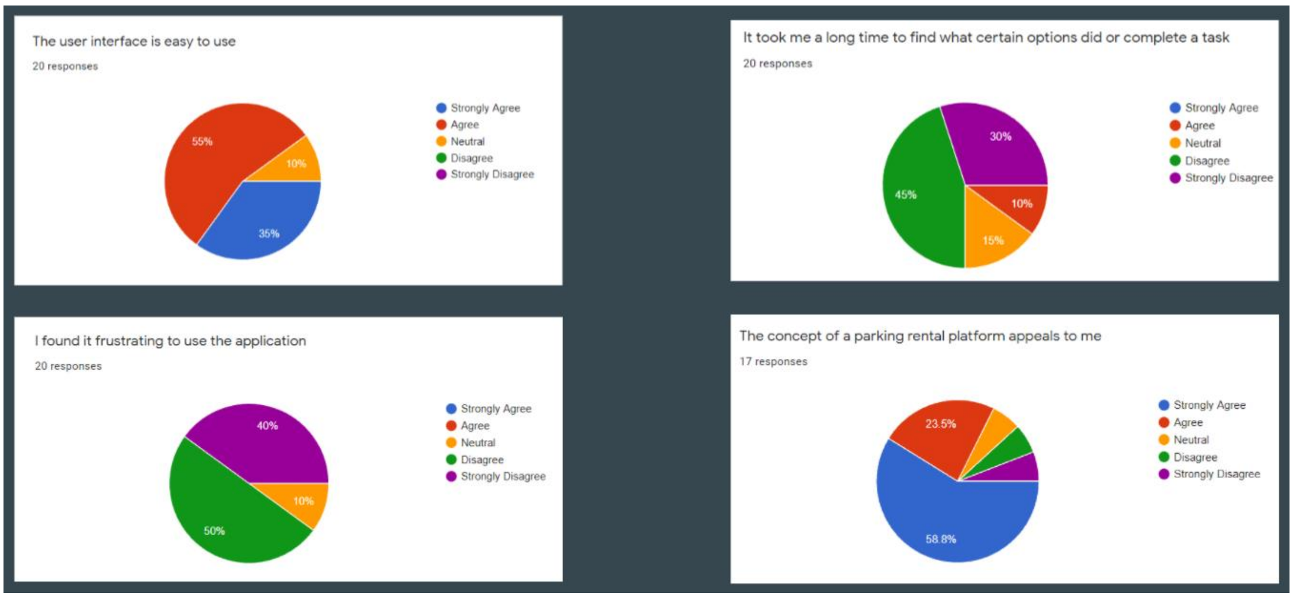
90% of the participants rated the user interface positively when asked about ease of use. Similarly, 90% disagreed or strongly disagreed that the application was frustrating to use. In both questions, no user found that the application was difficult to use. However, this is not to say that users found all aspects of the application easy to use. Indeed, there were some scenario specific usability issues which are discussed in section 8.3. (**Fix this in final to actual section #)**. The respondents also suggested improvements to the overall application usability. **Discuss some overall usability improvements**

Rephrase all the following

In terms of the overall applications usability, 90% of users found the UI easy to use, with no users finding the application to be frustrating. Additionally, 90% of users were able to identify and use the application options almost instantly, and 82% of respondents thought that the concept of the platform appealed to them.

When highlighting the positive aspects of the application, most respondents commented on the appeal of the platform and not having to worry about finding parking in specific areas, while also being able to generate additional income. Users found the application easy to navigate and use and the aesthetic professional and simple. Some users were specifically impressed by the map component implementation.

Most of the negative aspects highlighted by the users was fixed as a result of their feedback during the application iteration process. This included fixing input fields from not extending beyond the page, changing the terminologies used in the application to be consistent, making the timing consistently use the 12 hour format, and adding a search bar for the map component. Other suggestions included being able to upload multiple images at the same time and integrating a payment system, which were not within the scope of the project and will be included in the future work for the application.



The difference the user fixes made can be measured by comparing the average recommendation score of the first half of users as opposed to the second half. The average score was 4 for the first half of users, and 4.3 for the last half users. This result is not significant as measured by a t-test with a confidence interval of 95% as there were not enough user tests performed.

What did we find out that was surprising, how did we fix these issues?

When asked “how do you know you are currently parking at a location” we found some users tried to go back to the listing page to see if they were parking there, others were unsure about the homepage, so we made it clearer on both pages.

Users found surprising that the time slices were by the minute:

“The way the app calculated the amount to the minute.”

“The app pro rated the price for the actual time used.”

“The fact that it does not charge you hourly amazes me! :)”

Rephrase following paragraph

Majority of the user’s found the various application features easy to use and intuitive. 100% of users found the signup process straightforward, Only 25% of users found the process to add a parking location challenging and all the users were able to start and end their parking booking successfully. Additionally, 80% of users were able to find information about parking locations and bookings and 95% of users agreed that they were able to find their parking history and identify the buttons to seek help.

8.3. Scenario Impressions

8.4. Cross-platform Impressions

8.5. User Testing Fixes

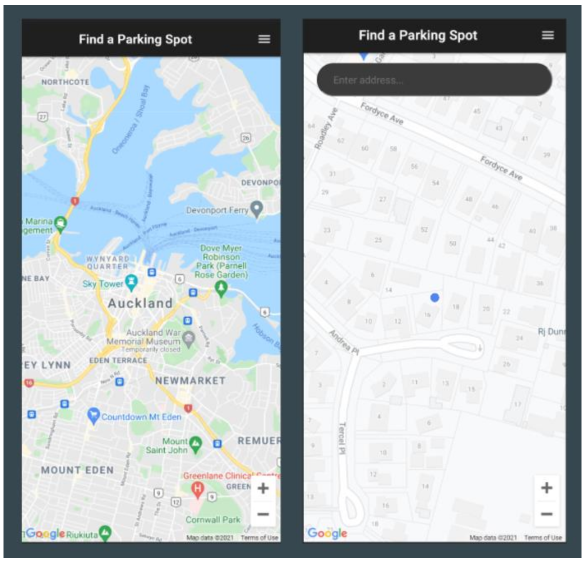
There were many bugs which were found throughout the course of user testing. There were many minor bugs such as a back button not being present on a particular page, or the map component having a slight scrollbar on the homepage. These fixes were relatively easy to implement.

There were also many usability changes which were made to the application. We found that many assumptions that we made while making the application were incorrect and discovered much variability in how different screens were viewed. In this section, we analyse some of these changes.

8.5.1. Address search bar

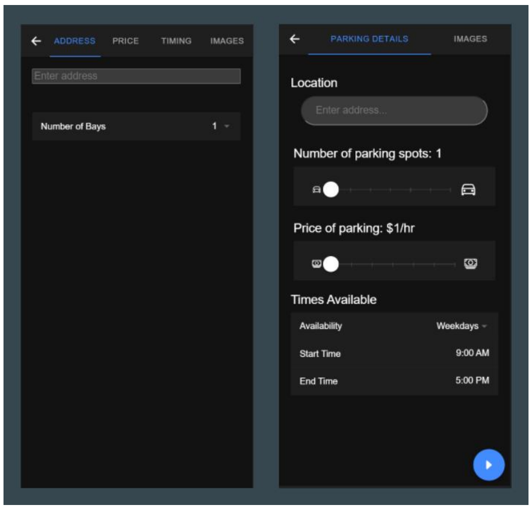
The most common issue we found was that users found it “frustrating when I can’t just type an address out”. Participants wanted to “[find] an exact address on the map”. Additionally, some users commented that they found were “not great at reading maps”. It was clear both in terms of usability and specificity that the map was inadequate.

We developed a search bar for the application which allowed users to easily navigate to a specific address, including local landmarks (Fig X.). This functionality was possible using the Google Places API. From user tests conducted after this feature was introduced, most users used this search bar when navigating to a specific address. The current location of the user was also added, and the zoom level of the map was tweaked to allow users to understand where they were in relation to the greater area. The search results were also restricted to addresses and locations within a 100km radius of the users current location to improve the relevancy of these results.



8.5.2. *Add listing page*

The original add listing page was separated into multiple pages, with the user needing to swipe between the pages or click the headers in the top bar to navigate between sections. Participants commented that “it was a bit weird to swipe through when filling out the listing details” and “creating the parking space could be tied up a little, put all of the fields on one page". Therefore, the address, pricing, and timing details were added onto the same page (Fig.X.). The number of bays field and pricing field were changed to a slider input for greater mobile compatibility.



Another pain point for users was the custom availability toggle button on the timing page. The users "didn't realise what the custom field did in the create listing page until" they clicked it. This was partly a terminology issue (section 8.5.4.) but also a visual issue where the option was disconnected from the main dropdown. Therefore, we added this functionality in the dropdown menu under 'custom days'. Additionally, we changed the default time for availability to 9:00am – 5:00pm to reduce confusion from the original 00:00 – 00:00 which indicated the day had no availability.

Many users also asked about the images page, wondering whether the images were optional, and how many images they could add, so there was clarification text added to the images tab. After this change we noticed that there was no confusion about that functionality.

8.5.3. Currently parking status

Under what did you find challenging: "How to end parking booking"

Suggestion: "Being able to end your parking session in history (potentially, not sure how useful since you can end it on the main page)."

"Difficult to figure out how to end my parking booking"

Jiaru thought parking in progress was just in general and not referencing that the current user was parked at that location.

8.5.4. Terminology changes

One of the most surprising discoveries were the perceptions of our terminology that we used throughout the application.

“Inconsistency between calling spots "listings" and "parking spots", I think it's better to use a consistent name throughout.”

“Meaning of 'Custom' was not clear. 'Custom days' would be better. I had not expected that owner info would be accessed through the billing part of history.”

“I was not entirely sure if the red "CLOSE" on the listings were a description of it being close to me, or whether it was a button.”

Jiaru thought close as in close to your location

Tianren didn’t understand what bays meant

Jiaru didn’t realise custom was part of availability

Jennifer was unsure what bays meant

Jennifer was unsure what close meant

9. Limitations

Our user testing study had many limitations. These were the result of the covid-19 pandemic, our choice of participants, and time constraints. Therefore, our testing conditions, accuracy and repeatability of this user study were impacted negatively.

The age range of participants was quite homogenous, with 85% being between in the 18-25 range. Younger people also tend to be better with technology, so the usability of our application may be overstated. Therefore, there needs to be a larger investigation into a more broad audience, as road users cover a broad range of ages, with most being over the age of 25.

Users were not using their own money, so it was not a real world scenario. User’s overall feeling towards the app could be different if they were paying for the service.

Respondents were also people that we knew, so they could be less likely to give us bad feedback on Airpark.

The study was done while we were making fixes, so the data is not all about the exact same version of the application. This was done as we had a limited time to perform the user studies, so we could not run another user study after a fixing phase.

Users did not experience the application on a mobile device, instead it was simulated in a browser. Therefore, some interactions were less usable. For example, we received complaints that the timing input was difficult to use as users had to scroll manually by holding down their mouse. The input's real usability was thus unable to be measured as it was designed for a mobile device with touch input.

Rephrase the following

Firstly, the entire user testing process had to be conducted on Zoom. This required participants to use a desktop web browser to access the deployed application, which doesn't accurately represent the user experience on a mobile device.

Additionally, the number of participants who tested the application was relatively small and the 20 person sample size doesn't represent the entire user demographic. 17 out of 20 participants were engineering students at the university of auckland, and were all in the age group of 18 to 25 years. Only 50% of respondents drove a car and only ~21% owned a parking space they wanted to rent out, which made it slightly challenging to accurately depict the market sentiment.

10. Conclusions

In conclusion, there appears to be much research on shared economy platforms such as Uber and Airbnb, but there is a lack of research with respect to rental parking. The main reason behind this could be that there is currently no large application which has disrupted the industry.

Therefore, there is an opportunity to introduce a new parking application utilising existing usability research, mobile map technology, and a new shorter time slice feature for pricing. This project aims to develop an MVP of such an application to investigate the impact on parking problems such as cruising time, prices, and per capita parking space.

Furthermore, technical aspects such as which mobile framework should be used. Usability and usefulness will be measured via user testing and PACMAD analysis.

Acknowledgements

The author would like to thank their project supervisor Andrew Meads for providing support and advice throughout the entire research project. They would also like to thank their project partner Sreeniketh Raghavan for providing ideas, research and support while working together on this project. Finally, they would like to thank the participants in the user testing sessions, for providing their time and valuable insight into improving Airpark.

11. References

[1] R. Belk, “You are what you can access: Sharing and collaborative consumption online,” *Journal of Business Research,* vol. 67, pp. 1595-1600, 2014.

[2] Z. Mao and J. Lyu, “Why travelers use Airbnb again?,” *International Journal of Contemporary Hospitality Management,* vol. 29, p. 2464–2482, 9 2017.

[3] D. C. Shoup, The high cost of free parking, Planners Press, American Planning Association, 2011.

[4] G. Pierce and D. Shoup, “Getting the Prices Right,” *Journal of the American Planning Association,* vol. 79, p. 67–81, 1 2013.

[5] P. B. Goodwin, “Traffic reduction,” in *Handbook of transport systems and traffic control*, Emerald Group Publishing Limited, 2001.

[6] C. Gibbs, D. Guttentag, U. Gretzel, L. Yao and J. Morton, “Use of dynamic pricing strategies by Airbnb hosts,” *International Journal of Contemporary Hospitality Management,* vol. 30, p. 2–20, 1 2018.

[7] R. Stock, “The share club,” *Dominion Post,* 11 2017.

[8] K. Jenkins, “Platforms in Aotearoa: our fast-growing sharing economy,” *Policy Quarterly,* vol. 14, 3 2018.

[9] S. P. Choudary and G. Parke, “How to build a successful platform business,” *INSEAD Entrepreneurship blog,* 6 2016.

[10] T. Vithani and A. Kumar, “Modeling the mobile application development lifecycle,” in *Proceedings of the International MultiConference of Engineers and Computer Scientists*, 2014.

[11] A. Biørn-Hansen, T. A. Majchrzak and T.-M. Grønli, “Progressive Web Apps: The Possible Web-native Unifier for Mobile Development,” in *Proceedings of the 13th International Conference on Web Information Systems and Technologies*, 2017.

[12] A. Charland and B. Leroux, “Mobile application development,” *Communications of the ACM,* vol. 54, p. 49–53, 5 2011.

[13] V. Setlur, C. Kuo and P. Mikelsons, “Towards designing better map interfaces for the mobile,” in *Proceedings of the 1st International Conference and Exhibition on Computing for Geospatial Research & Application - COM.Geo \textquotesingle10*, 2010.

[14] K. Church, J. Neumann, M. Cherubini and N. Oliver, “The "Map Trap"?,” in *Proceedings of the 19th international conference on World wide web - WWW \textquotesingle10*, 2010.

[15] R. Harrison, D. Flood and D. Duce, “Usability of mobile applications: literature review and rationale for a new usability model,” *Journal of Interaction Science,* vol. 1, p. 1, 2013.

[16] S. Min, K. K. F. So and M. Jeong, “Consumer adoption of the Uber mobile application: Insights from diffusion of innovation theory and technology acceptance model,” *Journal of Travel & Tourism Marketing,* vol. 36, p. 770–783, 9 2018.

[17] MRCagney, “The Economic Impacts of Parking Requirements in Auckland,” *Auckland Council,* 8 2013.