SmartPark: IoT-Driven Automatic Parking Solution

DESIGN DOCUMENT

Team Number: sddec24-17
Client/Advisor: Md Maruf Ahamed

Team Members/Roles:
William Clemmons - Project and Software Lead
Mubassir Serneabat Sudipto - QA Lead and Software Engineer
Ethan Haberer - Software Engineer
Zachary Sears - Hardware Lead
Kennedey Reiling - Hardware Engineer
Brian Witherspoon - Hardware Engineer

Team Email: sddec24-17@iastate.edu
Team Website: https://sddec24-17.sd.ece.iastate.edu/

Revised: 4/27/2024

Executive Summary

Development Standards & Practices Used

Practices:

- Hardware
  - Circuit Design
  - Rapid Prototyping
  - Embedded Systems
- Software
  - Source Control
  - Testing
- Project
  - Agile
  - Kanban

Engineering Standards:

All Standards are described in Section 2.2 of this Design Document.

- Hardware
  - IEEE 802
- Software
  - IEEE 3156-2023
- Project
  - Standard 14-5A-5 C

Summary of Requirements

- Software
  - The user interface does not impede on user from driving.
  - The app is intuitive for users.
- Hardware
  - Sensors should be able to detect when a car has pulled in or left a parking spot.
  - Withstand being outside for extended periods of time.
  - Reliably send data to the server and keep it up to date.
- Server
  - A Firebase server setup for simple and secure data communication with the app.

    - An Advanced reservation system to minimize user interaction with the phone while driving, based on insights from user experience design.
    - A robust and scalable infrastructure to handle the growing volume of users and environmental challenges, as informed by technical and quality assessments.

Applicable Courses from Iowa State University Curriculum

- SE
    - SE 3190 - User Interfaces
    - COMS 3090 - Software Development Practices
    - CPR E 2880 - Embedded Systems
- CPRE
    - CPRE 3810 - Computer Organization and Assembly Level Programming
    - CPR E 2880 - Embedded Systems
- EE
    - EE 2850 - Problem Solving Methods
    - EE 2300 - Electronic Circuits and Systems
- CYBE
    - CYB E 2310: Cyber Security Concepts and Tools
    - CYB E 2340 - Legal, Professional, and Ethical Issues in Cyber Systems

New Skills/Knowledge acquired that was not taught in courses

- Software
    - Figuring out the best programming language for a specific application.
    - Setting up a server from scratch
- Hardware
    - Arduino programming
    - Online basic Arduino/hardware simulation
    - Rapid prototyping and iteration

Table of Contents

List of figures/tables/symbols/definitions

1 Introduction

1.1 Problem Statement

Currently, parking on campus can be tricky. Finding parking is always arduous because you must consider multiple things. This includes, parking that is staff only, if the lot you want to park in is full, and how long it could take to find a spot. This project aims to eliminate these issues by streamlining the parking experience. Our team will create a detection-based system to monitor parking spots and update an app that students, teachers, or whoever may need to park on campus and allow them to view and reserve available parking to eliminate confusion.

1.2 Intended Users

The intended users for our project are students, faculty, and visitors on campus looking for parking spaces. Students often face the challenge of finding a parking spot before class starts. Although faculty have reserved lots, they still need help finding the most ideal space. Additionally, visitors to the campus who are attending events or meetings also need help finding parking spaces. By developing a detection-based system that monitors parking spots and updates an app, our project aims to address the parking issues these user groups face. This will allow them to view and reserve the most convenient and available parking spots, streamlining the campus parking experience and eliminating confusion.

2 Requirements, Constraints, And Standards

2.1 REQUIREMENTS & CONSTRAINTS

We have divided our requirements into two categories: Functional Requirements and Non-Functional Requirements. Functional Requirements include goals that have definitive results. Non-functional requirements describe conditions that do not list specific numerical values but are areas of focus.

Functional Requirements:

Our functional requirements include live sensor data, a mobile app, a way to communicate with non-app users, and a way to accept payments. Our sensors must send live, up-to-date information about parking spaces. Each sensor must be able to tell if there is a car in a parking space and the sensor information must be sent to our server to ensure the user gets reliable parking data. A mobile application must be user-friendly and allow users to reserve parking spots for a given lot. The app must also direct the user either to their reserved spot or an open spot, depending on what they were looking for. Another requirement we must meet is a way to communicate with users without the app by directing the user through a parking lot. Our final functional requirement is a unique way to accept payment from the user within the app and through a kiosk.

Non-Functional Requirements:

Our non-functional requirements include creating a system with low latency to keep the status of parking spots accurate. Additionally, the reliability of our application, server, and hardware are pertinent to ensure a user-friendly system. Furthermore, our server and application need to be online and working consistently. To hold the trust of our clients, our server and application must encrypt and protect user information while taking secure payments. Our application must be presentable and navigable to gain popularity and keep customers.

2.2 ENGINEERING STANDARDS

IEEE 3156-2023

Standard Ruling: A standard for privacy-preserving computation integrated platforms is needed to meet the evolving requirements of multi-sourced data computing and sharing. Requirements of privacy computation integrated platforms, including the reference architecture, the functional requirements, the performance requirements, and the security requirements of privacy-preserving computation integrated platforms, are provided by this standard.

Justification: Our project will require us to take user payments through our app. We must implement security systems that protect those users from potential attempts to steal information.

IEEE 802

Standard Ruling: This report includes use cases and communication requirements for wired and wireless bridged networks. Dense use of wireless devices with differentiated QoS requirements and operation in a factory environment are considered. Gap analysis from existing IEEE 802 standards and necessary technology enhancement are also covered in the context of time-sensitive networks for the future.

Justification: To communicate with our server, we will use a wireless bridged network for our hardware. To abide by this standard, we must operate our hardware in compliance with the code.

Standard 14-5A-5 C: Parking, Stacking Space Size, And Drive Dimensions

1. The minimum size of a standard off-street parking space is nine feet by eighteen feet (9' x 18'), exclusive of aisle width.
2. The minimum size of a compact off-street parking space is eight feet by fifteen feet (8' x 15'), exclusive of aisle width.

Justification: While this standard is for Iowa City specifically, this applies more generally to parking lots nationwide. These two subsections of this standard apply to our project by giving us dimensions to work within. Therefore, any hardware we deploy to the lot should not infringe on these dimensions.

3 Project Plan

3.1 Project Management/Tracking Procedures

Our team has decided to adopt the agile management style. As a group, we have decided it is best to meet twice a week to work on parts of our project while also meeting with the client/advisor once a week. While we meet often, we do not have a structured set of tasks each time we meet and just tackle whatever we feel is most pressing that day. This works best for us because, between the hardware and software, many unexpected issues could arise that, if we were on a stricter timeline, would cause significant shifts in focus that would delay the project completion time.

To track our progress, we are using a software called Trello. This website allows us to organize our tasks into different categories based on the completion of the task. Trello allows specific team members to review

and complete the assigned tasks as needed. This makes it easy for our team to see who should be doing what and who to contact if there is an issue. Additionally, we are organizing our files in Google Drive to stay orderly.

3.2 Task Decomposition

As an agile team, we decompose tasks as small as possible; for example, when considering the overall task of researching information about the hardware, we broke it down into researching boards, sensors, how to power the boards, etc.

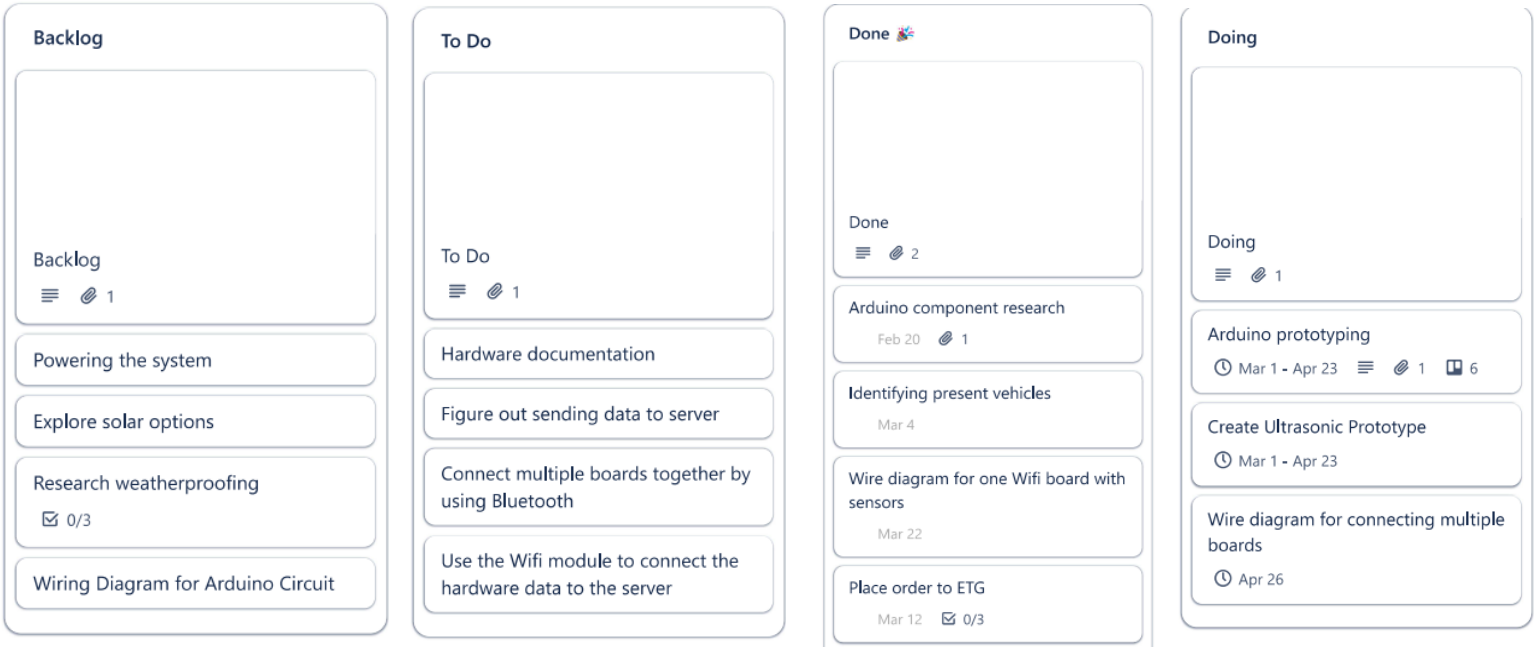

Figure 3.2.1: Decomposition of hardware team

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Hardware Team

One of the main upcoming objectives for the hardware team is to have a prototype by the end of the semester. We have broken this down into more achievable tasks to accomplish this goal. The first is to get the ultrasonic sensor working with Arduino. The success of this task will be measured by having LEDs connected to the Arduino that will light up if a vehicle is within 30cm of the ultrasonic sensors. The next task was to link multiple sensors to one board and get the system functioning for all the sensors. This is easily measured by getting the system to work with a standard test. Our goal at the end of this project is to know when a car has left a parking spot within 10 seconds of leaving. In other words, we want our program to have live data with a buffer of 30 seconds.

App Development Team

The app team has many milestones to accomplish to succeed in creating a usable application. For example, making the first functional prototype is the main upcoming objective for the app development team. This

involves having a functional live application that displays our app requirements. We will first have to finalize our UI design by breaking this task into subtasks. This will be accomplished after discussing which design suits our needs. After choosing a UI design, we will program with React Native. Our prototype will have multiple pages, so the creation of each page could be broken down into a subtask. This extrapolates our main idea of prototyping, as each page will require its prototyping phase.

3.4 Project Timeline/Schedule

TASKS	START DATE	DUE DATE	DURATION	STATUS
Phase 1. Research	01/16	02/25	41	Complete
Sensor Options	01/16	02/25	41	Complete
Programming Language	01/16	02/26	42	Complete
Server Type	01/16	02/26	42	Complete
Phase 2. Design and Development	01/30	02/25	2584	In Progress
App Prototype	01/30	05/01	93	In Progress
Hardware Prototype	01/30	05/01	93	In Progress
Phase 3. Integragtion	08/16	10/15	61	Needs Review
Connect App to Server	08/16	10/15	61	Needs Review
Connect Hardware to Server	08/16	10/15	61	Needs Review
Phase 4. Implementation	10/16	12/10	56	Needs Review
Testing in Parking Lot	10/16	12/10	56	Needs Review

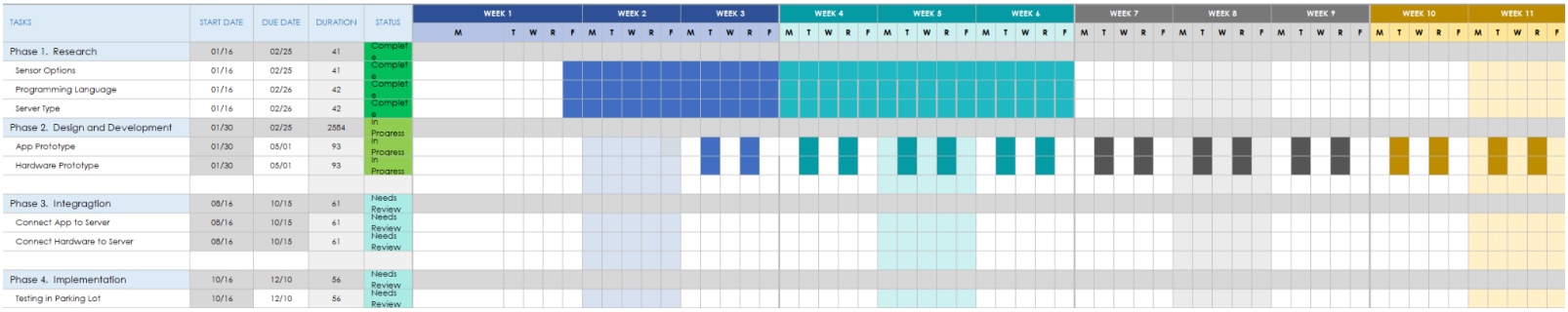

Figure 3.4.1: Projected timeline of project

The image above shows our team's Gantt chart. This is not one of our primary resources in scheduling. Instead, we use our team's Trello board, as mentioned earlier.

For each sprint, we focus on 1-2 tasks each group can work on throughout the week. Once the week is over, we discuss what task we got done, the issues we encountered, and how those issues will impact the following week. Once the problems have been discussed and potential solutions brought forth. We talk about what we should get done in the following week. Coming up with a weekly schedule allows us to address new problems quickly and continually improve our design as we become more knowledgeable about our project.

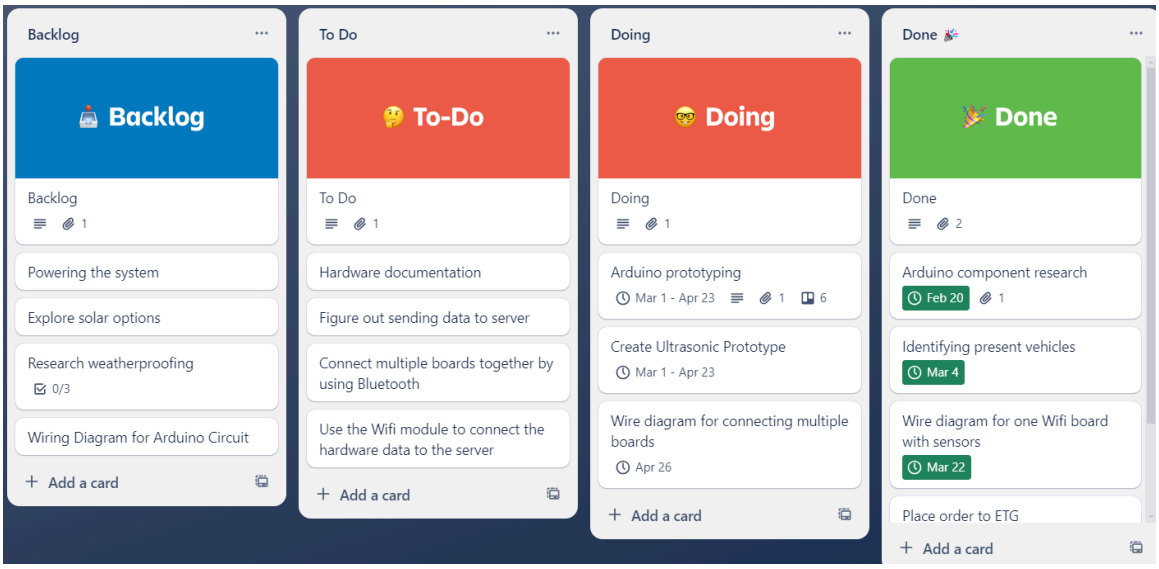

Figure 3.4.2: Hardware team Trello board

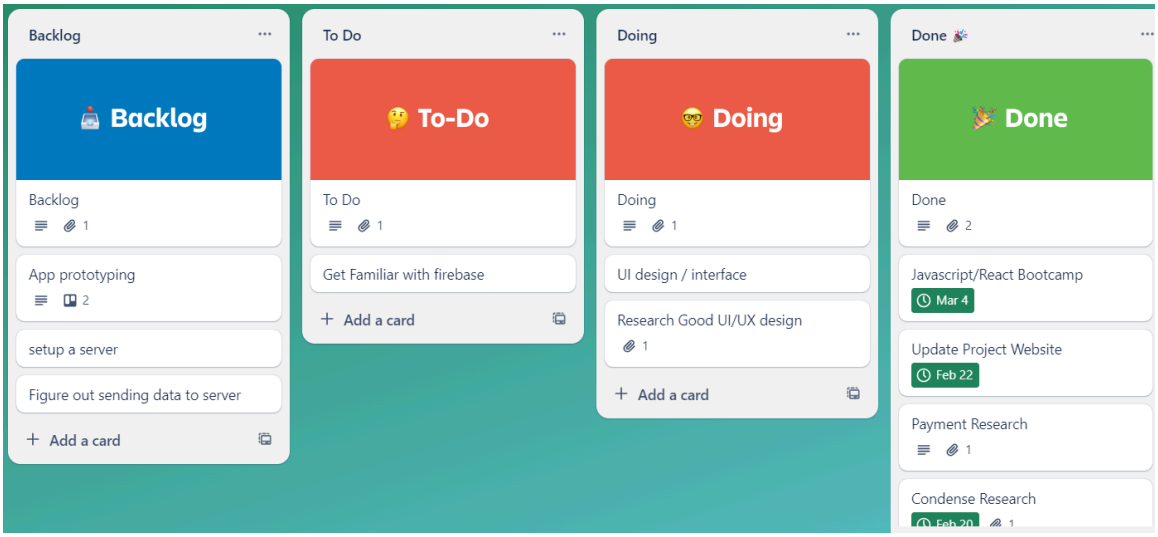

Figure 3.4.3: Software team Trello board

3.5 Risks And Risk Management/Mitigation

We have included a few scenarios in the table below to show examples of the risks attached to our tasks. Overall, our risks are insignificant but not negligible. To mitigate these risks, we will become aware of all possible risks by documenting them before each task is started. This will decrease the probability of these risks occurring, saving us many hours and potential money as it is less likely for equipment to become damaged. If we encounter a hazardous task, we will reevaluate the task to lower the risk factor. For our project, the main risk is losing time to ventures that do not end up contributing to the end goal of our project.

Table 3.5.1: Tasks, Risks and Likelihood Probability

Task	Risks	Probability
Learn the basics of React Native	No risks.	N/A
First app prototype	Spending many hours on an idea that does not satisfy our needs.	20%
Create hardware prototype	Similar to our other prototypes, we could lose many hours if the prototyping is unsuccessful.	35%
Testing hardware	We risk damaging the equipment to test the hardware in a real-world application.	10%

3.6 Personnel Effort Requirements

Table 3.6.1: Task and Time Decomposition

Task	Hours Required
Design UI	10
First App Prototype	45
Arduino Prototype	40
Test Bandwidth Capacity of Prototype	10
Hardware to Server	20
Documentation	15

3.7 Other Resource Requirements

The main resource for completing our project is hours. With the volume of our school work, it is difficult to allot time for this project. Additionally, it will be necessary for us to acquire an Apple Developers subscription to publish our application to the App Store.

4 Design

4.1 Design Context

4.1.1 Broader Context

In a broader context, our design is meant for visitors, students, and staff alike. Many people who have been to the ISU campus have expressed that parking is an unsatisfactory experience as it is often a confusing and stressful one. With all of these people in mind, it is important that we design this project with these users' safety and satisfaction in mind. In *Table 4.1.1*, our team has listed the areas we are prioritizing to keep the end users well being protected.

Table 4.1.1: Broader Context

Area	**Description**	**Examples**
Public health, safety, and welfare	This project affects the general well-being of the users of our application via the inherent risks of using an application in a vehicle, i.e. being distracted looking at a phone while driving to navigate an area. This is especially pertinent in a pedestrian-heavy area such as a parking lot. However, with our planned navigation system, we intend to mitigate these risks by parking efficiency while the application is being used. New possible hazards are also introduced by deploying sensors to a	We have been cognizant of our ultrasonic sensors and have verified that they will be harmless to humans and animals. Having a navigation feature, our application will require a phone to be looked at while driving. We will have a disclaimer on the app to mitigate this.

	parking lot. These sensors themselves are harmless to the public, however, the supporting hardware may increase the chances of single-vehicle collisions if not heeded adequately by drivers.	
Global, cultural, and social	Our project is aimed at employees of Iowa State, students, and visitors. It allows these groups to park more easily and more conveniently. If used correctly, all users will be able to park quicker.	Our reservation service will eliminate the need to drive in circles to find a parking spot.
Environmental	Our solution will increase power drawn from the relevant power plants in the area and could contribute to the detrimental environmental effects of electronics components mining and manufacturing.	While the power requirements of our hardware systems will be low individually, deploying this system to an entire parking lot may draw a significant amount of power. The software systems will require processing time on phones and a dedicated server, which must be kept cool, further increasing this solution's power draw. The resources required for our hardware have to be mined and manufactured into the components necessary for deployment.
Economic	Our system's automation could lower the need for monitoring certain lots. Additionally, our system, if added to multiple parking lots, could offer a new job of maintenance and upkeep of the entire system. The university will be liable to perform any maintenance or pay for it if necessary.	The economic impact of our solution will require a payment from users per parking session. An installation and labor fee from the university or relevant parties to install and maintain the hardware we plan to deploy. The university's parking division will have to spend less on refueling their vehicles since our application will provide them with relevant data regarding over-time parking and no-payment parking.

4.1.2 Prior Work/Solutions

Throughout the planning process, we encountered 3 companies trying to solve a similar issue: parking can be stressful and overwhelming. The first company is ParkMobile, the second is Parkingapp.com, and the third is SpotHero.

ParkMobile makes it easy for users to pay for a parking spot; it also allows users to search for selective parking spots, whether that is to be close to a certain location or to be near an electric car charger (Lister, 2020). ParkMobile is unique in these ways. One advantage of ParkMobile is that they are supported by all platforms making it easy to use. Some things that differentiate ParkMobile from our solution are the ability to guide users to open parking spots and our live data on available spots (Zuckerman, 2019).

Parkingapp.com was very simple and basic. There were not any unique features of their app. Although they did not have any uniqueness to them, they did have some advantages. They are compatible with

Parking.com and Parking Passport. Disadvantages for this company are their lack of information and the tiny community. When researching this company, our team took this as a takeaway of what not to do and what we wanted to include for our users(*Find ways to pay for parking*).

SpotHero allows users to see real-time parking availability and prices based on location. They are achieving something similar to our solution. One advantage of their product is their allowance of user reservations and payments. Although this sounds like a very sound and well-organized solution, it has disadvantages: the accuracy and reliability of their parking lot availability are not always correct, and their service fees increase the overall cost of parking (Frequently asked questions).

Based on the other solutions out there, we have decided that our solution must have advantages for each company to accomplish the best solution on the market by letting users reserve spots from the comfort of their homes, guiding the users to their reserved spots, and allowing users to search for available parking with their specific needs. Our final design will follow this pros and cons list in Table 4.

Table 4.1.2: Pros and Cons List of our solution

Pros	Cons
Tension free operation	It relies on the accessibility of technology
Easy and secure way to pay	No hard prevention of inappropriate or illegal parking
Ability to reserve a parking spot	Reckless drivers could damage technology
Live data on parking lot availability	Handling app users and non app users
Guides users to parking spaces	Hardware maintenance

4.1.3 Technical Complexity

Hardware

On the hardware side, our team is looking into using WiFi signals to send data between Arduino boards and a server. This will be done via the NINA W102 chip on the Arduino Nano 33 IoT board, which gives the board WiFi and Bluetooth capability. This chip also allows the board to be used as an internet access point. This WiFi functionality is accessed through the Arduino library system. Finally, our hardware team will review the documentation and circuitry to minimize the system's power consumption while maximizing its sustainability and ease of maintenance.

Software

The software portion of our design involves multiple levels of software development. These include mobile application development, server communication, and managing a database. The application is the vessel for users to communicate with our server. The data needed by the server will be stored in our database. The server is important as it is responsible for receiving the live data from our hardware. The user interface for the application will be simple, but this is a benefit of our design as it will ensure all users successfully operate our system. All three portions of the software team are important to communicate with the user and hardware.

4.2 Design Exploration

4.2.1 Design Decisions

Several critical design decisions were pivotal in developing our innovative solution to shaping the final product. Selecting the appropriate sensor for vehicle detection was a foundational step, ensuring our system could accurately and reliably identify vehicles in various conditions. Recognizing the diversity of our user base, we also strategized how to assist users not utilizing our app, aiming for inclusivity and accessibility in our services. The reservation system presented another complex challenge, prompting us to devise a user-friendly, efficient method for managing bookings. Finally, determining the most convenient times and locations to make payments was crucial as we sought to streamline the user experience and enhance satisfaction. Each of these decisions was made carefully considering their impact on functionality, user experience, and the overall success of our project.

4.2.2 Ideation

When choosing sensors to detect vehicles, we went through a few iterations. First we started with infrared sensors because our client wanted to use those if possible. However, after some feasibility testing, we learned we would need an emitter and receiver for each spot. Not to mention that unskilled drivers would likely hit the placement of the emitter and the receiver. Then, the team went through a brainstorming phase where we discussed possible ideas for a solution. Cameras, LiDAR, and Ultrasonic sensors were brought up during the brainstorming. We even discussed a new technology allowing the parking spot to detect when weight is present.

4.2.3 Decision-Making and Trade-Off

Regarding decision-making, we do most of ours through open discussion. When considering a decision, we get together, talk about it, and share our opinions and ideas. If necessary, we document our concerns and research to address them if we don’t know. Since we are in the early stages of our projects we want to give concerns the time they need so that we can address issues early. Once we have decided, we write another document explaining what we learned and why we made that decision.

4.3 Proposed Design

4.3.1 Overview

For our smart parking lot system to operate, ultrasonic sensors pointed at each parking space will detect the state of the parking space or whether a space is occupied. This information will be sent to our server utilizing an Arduino board connected to the internet through the NINA chip onboard. This information will then be updated in a server database. This information will be accessible by the mobile application. Our users will interact with our system via a mobile application. This app will allow users to search for a parking lot near their destination, reserve a space, and pay for parking. The user can simply pay for parking without participating in the reservation process. The application will direct the user to the lot with Google Maps. Once the user parks in their reserved parking spot, the application will transition into the payment page, where the user will securely enter their payment information. If the user bypasses the reservation system, they will park in the lot and be instructed to take note of the parking space number where they parked. They can enter this information into the app and pay for parking. If a driver does not have the application downloaded, they can scan a QR code located in the lot to access a “one-time use” version of the application where they can pay.

4.3.2 Detailed Design and Visual(s)

Overall Design:

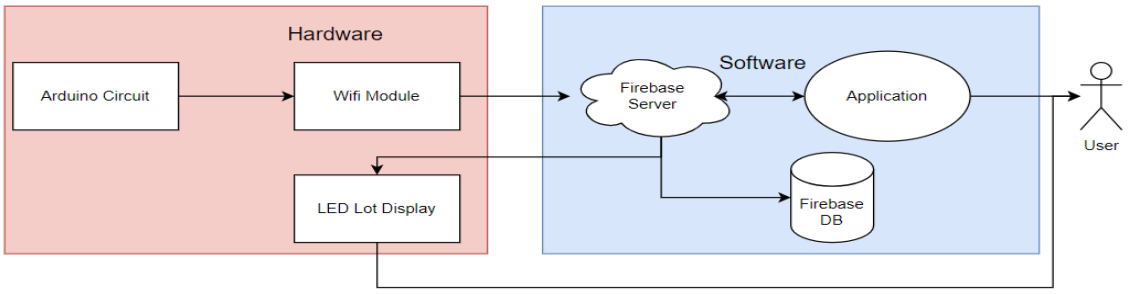

Figure 4.3.1: Overall Design Flowchart

Our system is broken into two main components: hardware and software. The hardware component of the system takes the multiple Arduino circuits and uses the wifi module to output our hardware information to our server. The software component of our system uses the server loaded with the sensor information and sends it to the application. The application is a way for users to check and reserve parking spaces. The server is also used to send information back to the LED display. This display will represent the parking lot availability, which allows users without the application to use our system.

Software:

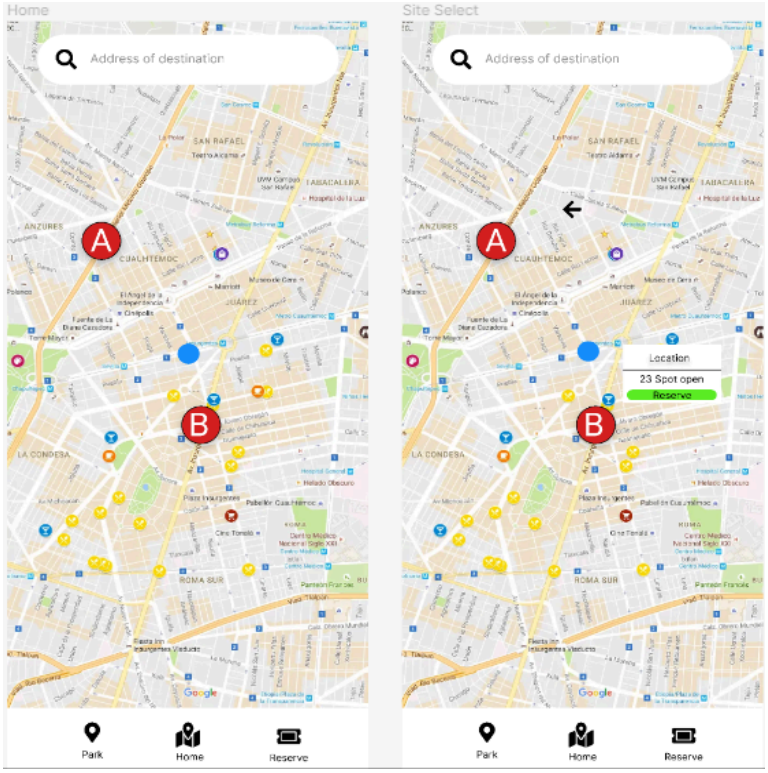

Figure 4.3.2: Application User Interface

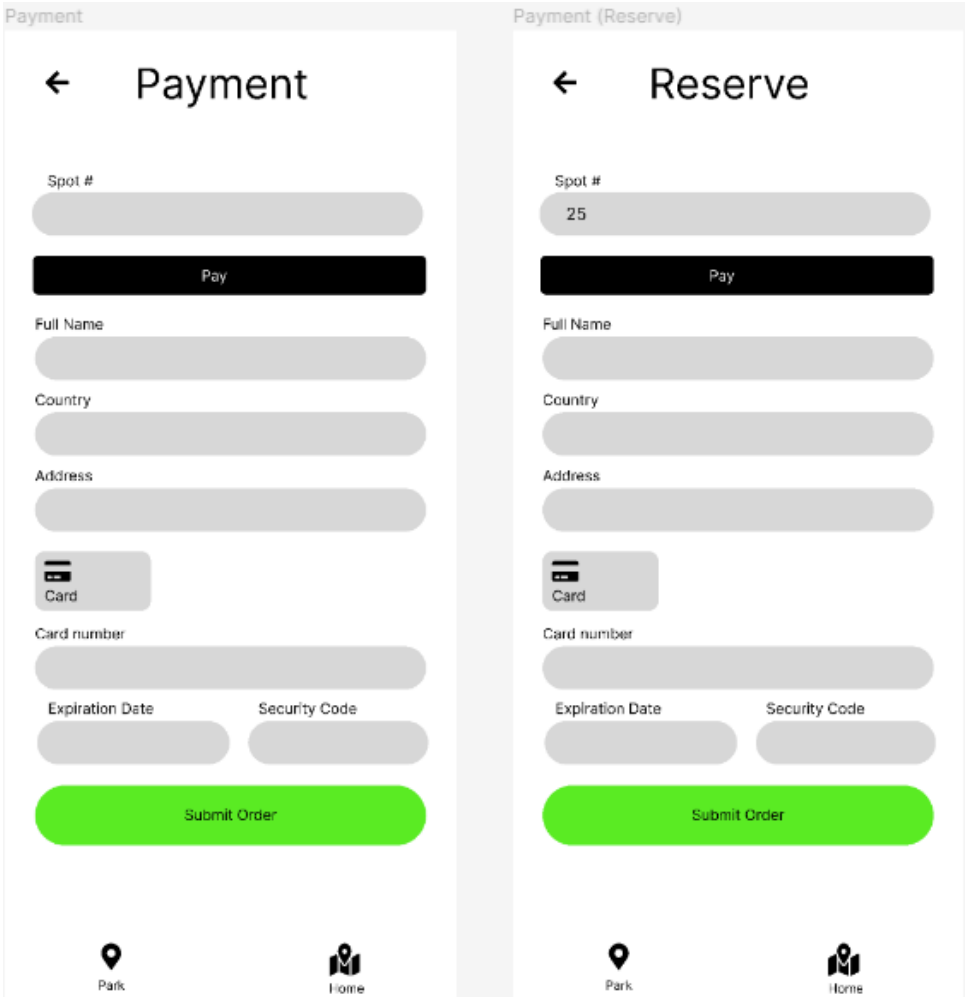

Figure 4.3.3: Application User Interface Continued

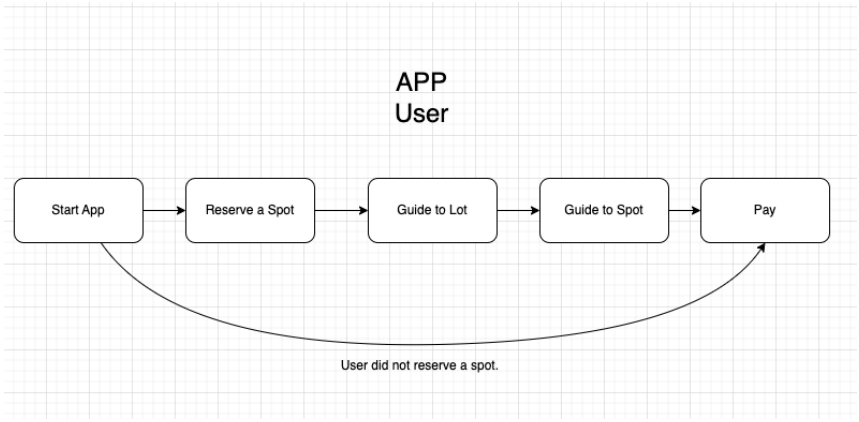

Figure 4.3.4: Application Flowchart

Above is the flowchart for our application. The user interface of our application will consist of two different pathways: Reserve and pay.

Reserve:

The user will open the application and choose the reserve tab on the navigation bar on the screen's bottom. The user will then enter their destination in a search box. Our software will search the web for parking lots affiliated with our program in a 2-mile radius of the user's destination. The resulting parking lots will be shown on screen. Each parking lot will have a respective box showing the address, distance from the destination, and number of available parking spots. The user will choose their desired lot to continue to the next step. All of the map functionality will be completed utilizing Google Maps.

After choosing a lot, our application will signal the server to pull an available lot number. This number will be stored in a new variable to ensure actual reservations. Next, the application will direct the user to the parking space with map software upon arrival which the map software with a certain proximity to the parking space will sense. Once parked, the application will continue to the payment function.

Pay:

For the users who arrived at their reserved spot and those who did not participate in a reservation but are still using the lot, the payment process will begin by selecting the app's payment button or scanning a QR code in the parking lot. The payment page will have prompts for spot number, credit card information, and amount of parking time. The users who reserve a spot will have their spot number automatically entered into the spot number prompt. In contrast, the other users will have to take note of their space number, which will be located on a sign in front of the space, and enter it into the app manually. After the necessary information is entered, the app will create a receipt, which will be downloadable for the user. This is the final step of the payment process.

Hardware:

On the hardware side, our team has set out to create an Arduino-based detection system that will be used to sense whether cars are parked or not.

Physical Parking Lot Setup:

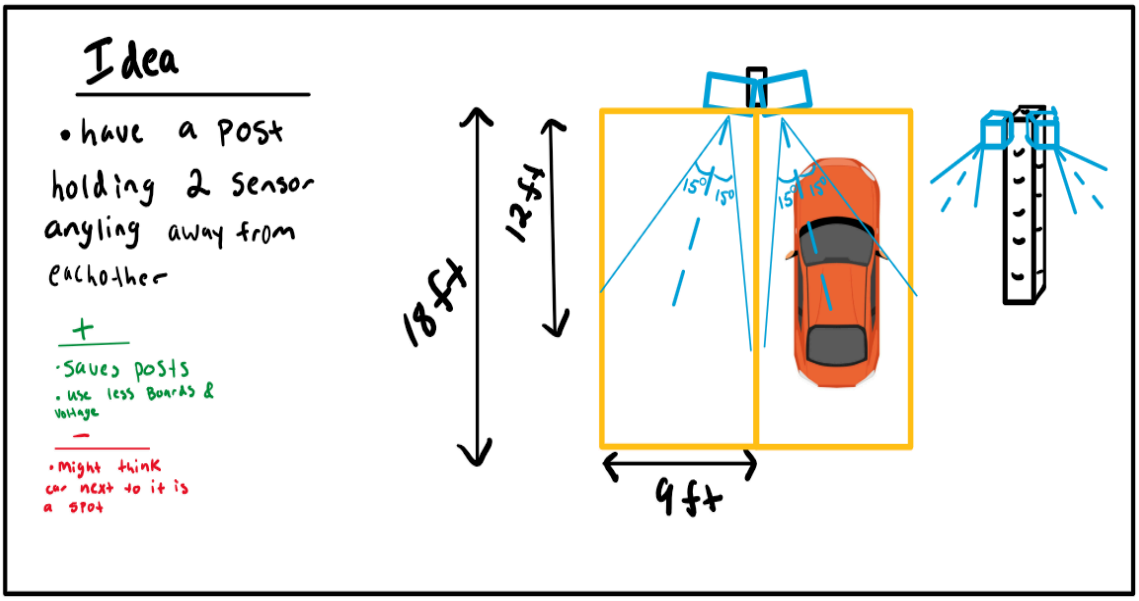

Figure 4.3.5: Sketch of Physical Sensor Representation

1 Arduino UNO per post with ultrasonic sensors connected to it. Each sensor will point toward an individual parking spot from an elevated position on the light post.

These sensors will be pointed down at an estimated angle of 30°, as represented by *Figure 4.3.5.* This setup allows for minimal posts to be inserted into the parking lot.

Sensor Arduino Circuit:

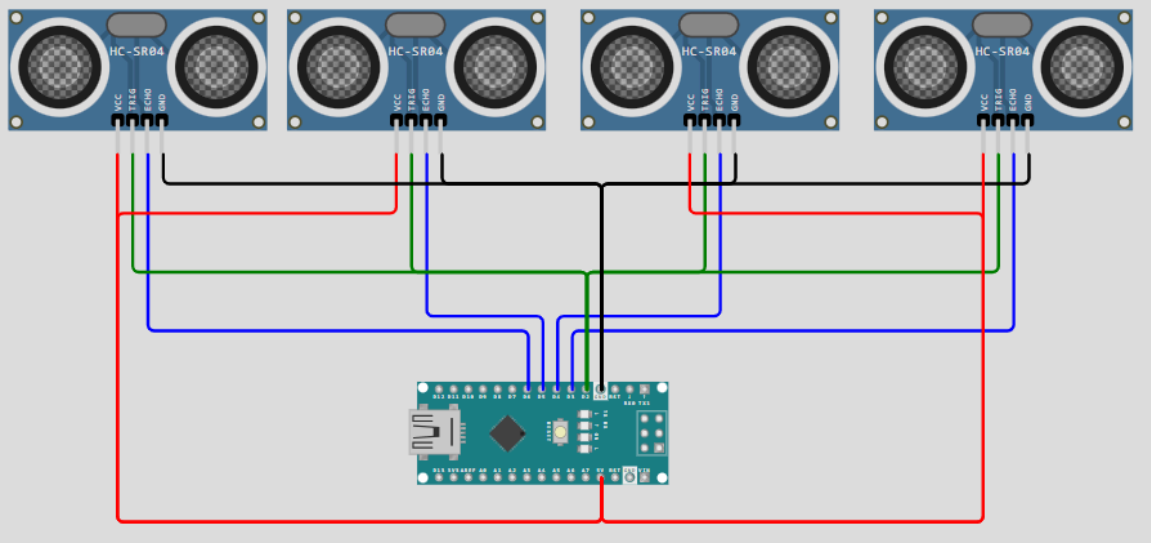

Figure 4.3.6: Circuit for Sensors

Each UNO board will have four 5V (Connected to red wire represented in *Figure 4.3.6*) ultrasonic sensors connected to it. The sensors will all be connected to the same trigger, so they fire an ultrasonic ping simultaneously (Green wire represented in *Figure 4.3.6*).

While the trigger signal is on the same wire, the echo signals are received on their own pins so the board can differentiate them. (Blue wire represented in *Figure 4.3.6*).

4.3.3 Functionality

Users can enter our lot regardless of whether they have the app or not. However, they will have different experiences. Our application will direct the users who reserve a spot to their spot, and they will pay upon arrival. For the users who do not reserve a place, they will pay after parking by using the app and entering their spot number which will be posted near their parking space. All necessary data will be taken from ultrasonic sensors, sent to our server, and stored in a database.

4.3.4 Areas of Concern and Development

Our design incorporates all of our user needs and should leave them satisfied if completed as intended. Knowing the impact on users without being finished or without prototyping is challenging. This will be a future focus of the team to come back to. We are concerned with the usability of our application and its friendliness for our users. We are trying to balance the amount of information and detail with simplicity that will be easy to navigate. We plan on holding a usability study when our first app prototype is completed to address this concern.

Our hardware team is focused on a few pending issues. The primary problem is the connectivity of our hardware devices. To update the information about the status of each parking space, our boards must be able to connect to Wi-FI. Using the Nano board eliminates the WiFi issue, but our team is unsure about how

much bandwidth each board will take up and if it will cause any issues with them all uploading data at the same time. Similarly, we are worried about the database's capability of taking in all the information of each board and if there's a possibility of overloading the server.

Weatherproofing is another issue that our hardware team faces. In the variable weather of Iowa, including rain, snow, ice, and intense heat, our sensors and boards need to survive these conditions. We have been researching for weeks how to accomplish this. Some solutions include creating a case for our sensors and purchasing weatherproofed Arduinos.

For our server, we must carefully choose the correct server for the data type we are working with. Failure to do so can drastically decrease the app's performance and the parking lot. Since we will be working with sensor data, having a database that can handle the constant data stream would be best. However, at this time, we are unsure if Firebase will be able to work for this application.

Finally, for the software, we must ensure that our application is intuitive and safe enough to be used while driving, even in a busy parking lot. We understand that using a phone while driving can be a safety risk. Therefore, we need to minimize the user input while driving and use an acceptable interface for moving applications. At the moment, we are still determining what the best standard of design is for a driving application, so more research is needed.

4.4 Technology Considerations

Describing the distinct technologies we used in our design by highlighting the strengths, weaknesses, and trade-offs made in technology. Also, discussing possible solutions and design alternatives below:

Table 4.4.1: Sensor Options

Sensor	Pros	Cons
Infrared	• Preferred by the client • Cheap	• Requires two sensors per parking space • Positioning sensors out of the way of users would be tricky • Sensitive to the color of the car
LiDAR	• Can detect everything in an area	• Expensive • Not entirely appropriate for our use case
Ultrasonic	• Cheap • Only need one sensor per parking space • Familiar to the team	• Little control over where the signal goes
Pavement	• Discrete	• Super expensive • Will have to renovate the whole lot to implement
Cameras	• Could work over multiple spaces	• Power intensive • Difficult to implement • Possibly expensive

Our final decision was to use ultrasonic sensors because the team is most familiar with them and relatively low cost, and their ability to function in a variety of conditions that other options may fail in.

4.5 Design Analysis

On the hardware side, we have created a base prototype for the detection system. This system takes an Arduino Nano and connects four ultrasonic sensors to it. We have tested this prototype, and it is working well so far. For the future, we are looking into beginning prototyping for the planned implementation of Arduino Wifi boards to connect to our server. We will start looking into weatherproofing options for the system because it is mainly outdoors. The system must be encased in something waterproof so it does not break down due to water damage.

We have just begun designing the application prototype for the software side in Figma. We reached this after discussing with the client what the user experience should be and doing a whiteboard to create a baseline for our application's appearance. We have also started experimenting with Firebase and React Native for backend and frontend development, respectively.

5 Testing

5.1 Unit Testing

We conduct hardware testing on a feature level to ensure consistent functionality in varying conditions. This includes testing the detection of objects with ultrasonic sensors and data transmission to the server. On the software side, we conduct component-level testing to prevent unintended behavior and degradation of our React components. For this, we will be using the Jest and React Testing Library. Jest will be used to test business logic, while React Testing Library will test that the user interface appears as intended.

5.2 Interface Testing

Our design incorporates two primary interfaces: a parking availability tracking hardware and a user-facing application that displays parking lot status. We will thoroughly test the hardware's sensors, communications, and reliability to ensure that we meet our requirements. For the software, we will evaluate its user interface, performance, and safety. The user interface must be visually appealing and intuitive for most users. To verify this, we plan to conduct user testing on our initial app prototype. Additionally, the software must perform well; slow data transfer times could create more issues than it solves. To measure performance, we will monitor the time for new server data to appear in the app. We will also test how the software functions under heavy user traffic.

5.3 Integration Testing

The two major integrations are the connection from the hardware to the server and from the server to the application. We need events picked up by the hardware to flow through the pipeline in real time. By that, we mean once a car has completely pulled out of a parking spot, it should appear in the app as open. Our best way of testing this is to simulate the real experience in our senior design lab. We will connect the hardware and app to the server and simulate a car pulling into a spot. Our goal will be to upload the hardware detection information to the server, and then the server sending that information to the app, and it being updated within 20 seconds.

5.4 System Testing

The final step of testing would be field system testing. In this test, we would set everything up like a client's lot and test various interactions and edge cases. If any issue occurs, we need to troubleshoot it on-site, which is less than ideal. We only want to do system testing once all our unit and integration tests pass.

5.5 Regression Testing

As we plan our hardware implementation, we explore ways to identify and prevent hardware malfunctions. While our unit tests will cover much of our software regression testing, we are also working on a comprehensive test suite to detect potential regressions as we develop our code. As the hardware is responsible for capturing crucial data and the app is our users' main point of interaction, these components must remain stable and reliable.

5.6 Acceptance Testing

Our acceptance test would mostly be handled through meetings with our client. Once we present our final demo, the client will have the final say on whether our design meets his vision. We have the requirements we came up with, but they can quickly become outdated if the client shifts thinking.

5.7 Security Testing

To avoid handling user payments, we use a third-party API from Stripe. Stripe has a good history of making payments easy for users and developers. However, we need to ensure that we are not storing any unnecessary information from the users. We should be able to detect these through our code reviews, but a more defined method may be necessary.

5.8 Results

Our unit and integration test will allow us to catch issues early, verify that we are ready for the next step, and assist in identifying system regression. When it comes to meeting requirements, we will continually communicate with the client to ensure their needs are met.

6 Implementation

The implementation phase of our project has yet to be solidified. We plan to set up a block of sensors in a parking lot on campus. Instead of applying sensors to each parking space, we will implement our system for a few spaces to prove its effectiveness. Our application will be uploaded to the Apple App Store and Google Play. The server and database have already been created and are currently online.

7 Professional Responsibility

In this project, our team's approach to professional responsibility is embedded within a well-defined team contract that outlines individual roles, communication protocols, decision-making policies, and quality control measures, ensuring that all members adhere to high standards of ethical conduct and professional practice. Our team's commitment to professionalism is evident through structured strategies for collaboration, inclusive participation, and the resolution of any inclusion issues, with designated responsibilities such as client interaction and project leadership reinforcing the importance of maintaining professional integrity throughout the project's lifecycle. By agreeing to a clear set of consequences for any breaches of this contract, our team underscores each member's serious commitment to the project's success and upholding the professionalism expected in the engineering domain.

7.1 Areas of Responsibility

We will discuss how the topics relate by applying the IEEE Code of Ethics to our broader context table (*Table 4.1.1*). The IEEE Code of Ethics stresses the importance of protecting the public by encouraging engineers to stand up for humanity against corruption. Global, cultural, and social topics are discussed concerning being truthful when releasing information to the public and treating all individuals with respect. The environment portion

Table 7.1.1: Areas of Responsibility

Area	IEEE	Differences from NSPE
Work Competence	An engineer should not work in an area outside of their expertise	Very similar
Financial Responsibility	Avoidance of conflicts of interest	NSPE is more focused on shareholders and clients
Communication honesty	Communicate truthfully to protect the population	NSPE is more generally pertaining to the company
Health, Safety, and Well-being	Protecting the public by encouraging engineers to stand up for humanity against corruption	NSPE has similar terms for this category
Property Ownership	Keep private information secure	Protect the property of all clients and privacy
Sustainability	Make it known if the environment is in danger	NSPE is more detailed in this category
Social Responsibility	Encourage coworkers and seek to improve public knowledge	Similar guidelines

7.2 Project Specific Professional Responsibility Areas

Table 7.2.1: Project Specific Responsibilities

Area	Is it relevant	Performance
Work Competence	It is irrelevant to our project because we must learn and work outside of what we currently know. However, we still do our best to create an effective solution.	N/A
Financial Responsibility	It is important to be responsible for the user's payment information and the money they	Medium

	send. We want to do our best to avoid fraud and embezzlement.	
Communication honesty	When reporting to the client, we always ensure that we are telling him the truth regardless of how it may affect his opinion of us.	High
Health, Safety, and Well-being	Since we are creating an application meant to be used while driving, we need to ensure that we are not inhabiting the users driving when using our app.	High
Property Ownership	We are doing our best to make our client and everyone involved in this project feel included and considered	Medium
Sustainability	We are constantly considering the environmental impacts of our design	High
Social Responsibility	We do our best to keep each other in check. If someone is doing something questionable, we can always talk it out.	High

7.3 Most Applicable Professional Responsibility Area

Safety is our most important concern because we want to ensure our users are safe using the app.

8 Closing Material

8.1 Discussion

Upon completion, our system will provide accurate information regarding the availability of a parking lot. Our application will allow users to reserve parking spots and pay for parking. If these criteria are met, we will consider our work successful.

8.2 Conclusion

Until now, the software team has created the first prototype for the application and set up the Firebase server. Additionally, we have researched various APIs, coding languages, and frameworks. The hardware team created the wiring diagram for four ultrasonic sensors with an Arduino board. The hardware team has researched multiple boards that will meet our requirements. Our goal is to implement our system into a portion of an existing parking lot on campus. This system will sense the state of a parking spot, occupied or free, upload this information to our database in live time, and allow users to reserve a parking spot online. The app will support payment online as well. To achieve these goals, we are implementing the agile

method. Specifically, we have decided to initially complete as much research as possible to prevent backtracking. Our main constraints have been finding time for the entire team to meet. Our initial designs were completed later than expected, and completing these earlier would be a main focus if we were to restart the project.

8.3 References

[1] IEEE SA, "IEEE Standards Association," *IEEE Standards Association*. https://standards.ieee.org/ieee/3156/10834/

[2] IEEE SA, "IEEE Standards Association," *IEEE Standards Association*. https://standards.ieee.org/ieee/White_Paper/10123/

[3] "14-5A-5: CONSTRUCTION AND DESIGN STANDARDS:," *American Legal Publishing*. https://codelibrary.amlegal.com/codes/iowacityia/latest/iowacity_ia/0-0-0-23897

[4] M. Lister, "Get ready for takeoff!: 5 advantages of using the Parkmobile app for Airport Parking," ParkMobile, https://parkmobile.io/blog/get-ready-for-takeoff-5-advantages-of-using-the-parkmobile-app-for-airport-parking/ (accessed Apr. 16, 2024).

[5] A. Zuckerman, "Parkmobile Review: Pricing, pros, cons & features," CompareCamp.com, https://comparecamp.com/parkmobile-review-pricing-pros-cons-features/ (accessed Apr. 16, 2024).

[6] "Find ways to pay for parking," ParkingApp.com | Find Ways to Pay for Parking, https://www.parkingapp.com/ (accessed Apr. 16, 2024).

[7] "Frequently asked questions," SpotHero, https://spothero.com/faq (accessed Apr. 16, 2024).

8.4 Appendices

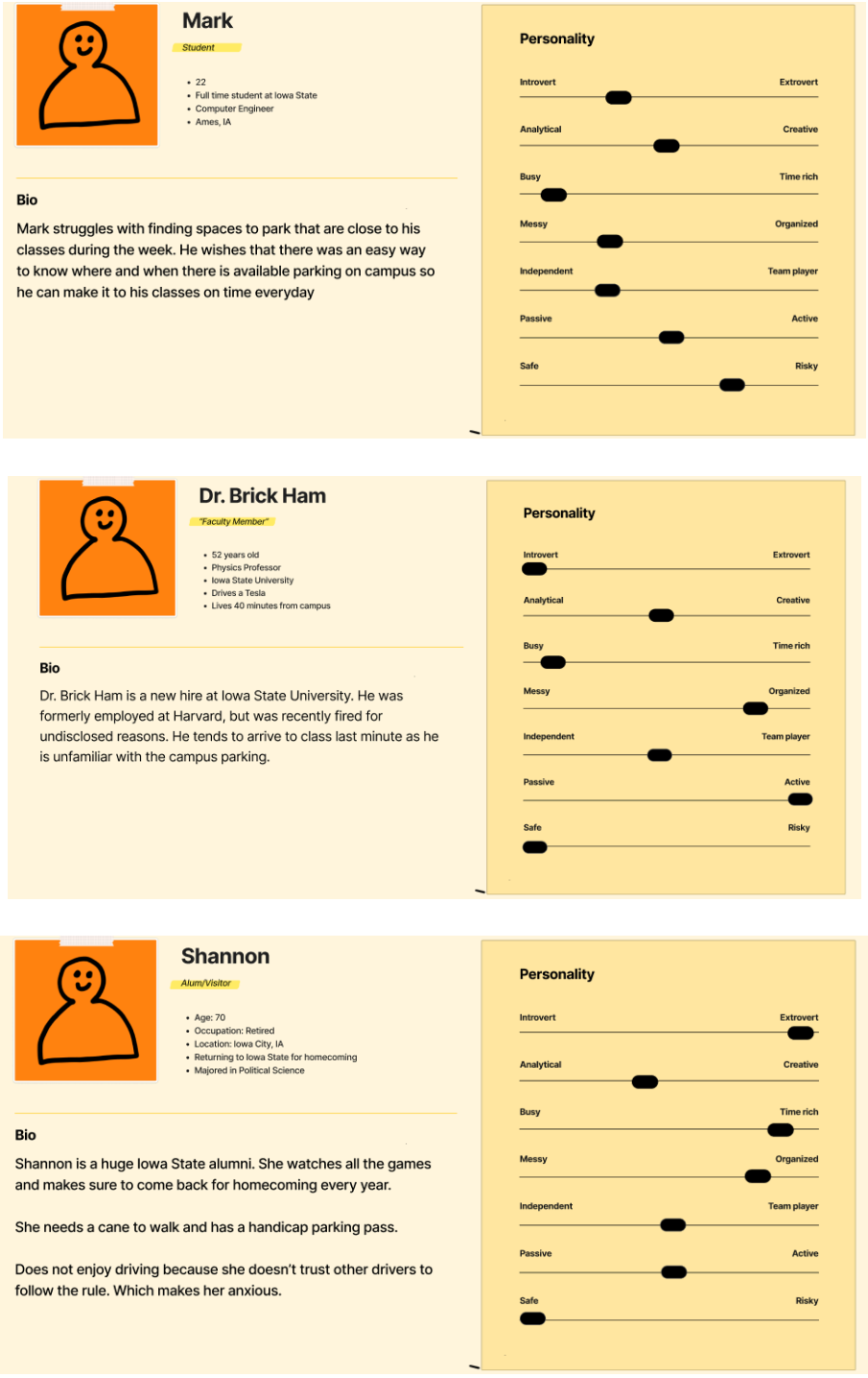

Figure 8.4.1: *User Profiles*

9 Team

9.1 Team Members

1) William Clemmons
2) Kennedey Reiling
3) Brian Witherspoon
4) Zachary Sears
5) Mubassir Serneabat Sudipto
6) Ethan Haberer

9.2 Required Skill Sets for Your Project

- Coding
- Circuitry
- Embedded Systems
- App Development
- Server Development

9.3 Skill Sets Covered by the Team

William Clemmons: Coding, App Development, Embedded Systems
Kennedey Reiling: Circuitry, Coding, Embedded Systems
Brian Witherspoon: Circuitry, Coding, Embedded Systems
Zachary Sears: Circuitry, App Development, Embedded Systems, Coding
Mubassir Serneabat Sudipto: Coding, Server Development, App development, Embedded Systems
Ethan Haberer: Circuitry, Coding, Embedded Systems

9.4 Project Management Style Adopted by the Team

Our team has adopted the project management style of Agile

9.5 Initial Project Management Roles

William Clemmons - Project and Software Lead
Mubassir S Sudipto - QA lead and Software Engineer
Ethan Haberer - Software Engineer
Zack Sears - Hardware Lead
Kennedey Reiling - Hardware Engineer
Brian Witherspoon - Hardware Engineer

9.6 Team Contract

Team Name: 17

Team Members:

1) William Clemmons
2) Kennedey Reiling
3) Brian Witherspoon
4) Zachary Sears
5) Mubassir Serneabat Sudipto
6) Ethan Haberer

Team Procedures

1. Day, time, and location (face-to-face or virtual) for regular team meetings:
    - **Tuesday from 6:00 PM to 7:00 PM, Thursday from 6:00 PM to 7:00 PM.**
2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):
    - **Face-to-face meetings are preferred, but online is acceptable.**

3. Decision-making policy (e.g., consensus, majority vote):
    - **Group conversation turning to majority vote.**
4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived):
    - **Shared decision based on group majority decision.**

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:
    - **All group members must come to planned meetings. If unable to attend, the group member must inform the team.**
2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:
    - **Each group member will be assigned a relevant task to their skill set as the project progresses. Each team member will be expected to complete the task by the timeline. The group will discuss and work around the issue if the deadline still needs to be met.**
3. Expected level of communication with other team members:
    - **Team members will be expected to communicate any adjustments made to the project to the team members that will be affected while also making sure other group members understand.**
4. Expected level of commitment to team decisions and tasks:
    - **Everyone will be expected to participate and complete their tasks. There will be open and honest communication.**

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction, individual component design, testing, etc.):

Role	Description	Assignee(s)
Project Lead	**Make sure that the team is organized and on the same page.**	**William Clemmons**
Client Interaction	**Responsible for reaching out to the client when needed.**	**Mubassir Serneabat Sudipto, Kennedey Reiling**
Quality Control	**Double-check designs and documents to make sure that they meet requirements.**	**Mubassir Serneabat Sudipto, Zachary Sears, Ethan Haberer**
Hardware Design	**Responsible for hardware aspects of the project.**	**Brian Witherspoon, Kennedey Reiling, Ethan Haberer, Zachary Sears**
Software Design	**Accountable for software aspects of the project.**	**William Clemmons, Mubassir Serneabat Sudipto**

2. Strategies for supporting and guiding the work of all team members:
    - **a. Constant and honest communication is to be kept when working. Updates are to be given in the updates chat of our team discord.**
      b. Any help needed should be reported in the help channel of our discord.
      c. Have a structured to-do list so each person knows what to do.

3. Strategies for recognizing the contributions of all team members:
    - **Have an updated page in our team chat to track what people have completed/contributed.**

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the team.

Name	Major	Most Relevant Skills
William Clemmons	**Software Engineering**	**Embedded Systems, Software, Presentations.**
Kennedey Reiling	**Electrical Engineering**	**Circuitry/Hardware design, Embedded Systems, Modeling, Client Relationship.**
Brian Witherspoon	**Electrical Engineering**	**Circuitry, Hardware design, Modeling, Documentation.**
Zachary Sears	**Computer Engineering**	**Embedded Systems, Hardware design, Programming, Documentation.**
Mubassir Serneabat Sudipto	**Cyber Security Engineering**	**System Security Essentials, Scripting, Debugging, Penetration Testing, Technical Documentation, Professional Presentations.**
Ethan Haberer	**Electrical Engineering**	**Circuit Design, Modeling, Technical Documentation, Programming.**

2. Strategies for encouraging and supporting contributions and ideas from all team members:
    - **a. Having democratic and non-judgmental discussions about design decisions when necessary.**
      b. Everyone will be encouraged to speak their opinions.
      c. Create a brain dump channel on Discord for when someone has an idea, they can put it in there for review later.
3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will a team member inform the team that the team environment obstructs their opportunity or ability to contribute?)
    - **a. Consult the team to gather group input on how to proceed.**
      b. Adjust how the team operates to include anyone who feels they need to be included.

Goal-Setting, Planning, and Execution

1. Team goals for this semester:
    - **a. Completion of the overall design of our project.**
      b. Gain a working-level knowledge of the components of the project.

2. Strategies for planning and assigning individual and teamwork:
    - **a. Have a timeline with specific deadlines and tasks that must be completed.**
      b. Delegate work based on skill set.
      c. Preplanning to understand what will be required and who will be responsible.
3. Strategies for keeping on task:
    - **Have a specific plan from the beginning of our project and continuously update it to fit the timeline.**

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?
    - **One a single offense, bring it up to the team either in the discord or in our next meeting.**
2. What will your team do if the infractions continue?
    - **After repeated offenses, the problem will be discussed with our advisor and professor if needed.**

****

a) *I participated in formulating the standards, roles, and procedures as stated in this contract.*
b) *I understand that I am obligated to abide by these terms and conditions.*
c) *I understand that if I do not abide by these terms and conditions, I will suffer the consequences as stated in this contract.*

1) William Clemmons DATE: 01/30/2024
2) Kennedey Reiling DATE: 01/30/2024
3) Brian Witherspoon DATE: 01/30/2024
4) Zachary Sears DATE: 01/30/2024
5) Mubassir Serneabat Sudipto DATE: 01/30/2024
6) Ethan Haberer DATE: 01/30/2024