iValet - Parking Lot Management System Using Computer Vision

ECE4871 Senior Design Project

iValet
Project Advisor, Dr Patricio Vela

Yunchu Feng, CmpE, yfeng336@gatech.edu
Wei Xiong Toh, EE, wtoh7@gatech.edu
Faiza Yousuf, CmpE, fyousuf6@gatech.edu
Kelin Yu, EE, kyu85@gatech.edu

Submitted

November 22, 2021

Table of Contents

Executive Summary

The iValet intelligent parking lot management system automatically directs drivers to the nearest vacant parking spot upon entering a crowded parking lot. The system consists of a camera and a user interface. The camera is used to provide a live video feed of the entire parking lot while the web application shows end users the directions to the empty parking spots based on their current location in real time.

When drivers enter the parking lot, they will need to scan a QR code that leads them to the landing page for the iValet web application. Once they select the type of vehicle they are driving, the system identifies the nearest vacant parking spot for their vehicle type from the video feed using an image segmentation model. The system will also obtain the driver’s location from the web application. iValet uses this information, together with the coordinates of the empty lot, to direct drivers to the vacant lot via a path planning algorithm. The route that the driver has to take will be displayed on the web application.

The ease at which drivers are able to find parking spaces for their vehicles can significantly affect their mood and overall experience of the subsequent activity they are taking part in. The use of iValet will help minimize some of the frustration, anger and oftentimes avoidable waiting that arises from being unable to locate a vacant spot in a crowded parking lot. The iValet system will cost an estimated $468.99.

1. Introduction

The iValet Parking Lot Management System is a novel and affordable parking management system that automatically directs drivers to the next available parking spot for their type of vehicle. The system consists of a camera system and a web application. The team requests $468.99 to fund the prototype of iValet.

1.1 Objective

The objective of iValet is to provide drivers with a seamless experience when searching for empty parking spaces in a crowded parking lot. A camera mounted in a position that captures the aerial view of the parking lot will provide the system with a live video feed. Individual frames from the video feed will be processed by the empty lot detection algorithm on an NVIDIA Jetson Nano and identifies the coordinates of suitable empty lots for the drivers' type of vehicle. A path finding algorithm will subsequently use these coordinates, along with the location information provided by the users' mobile phone, to calculate the shortest path from the users' current location to the aforementioned parking spot. This information will then be displayed on the web application for the drivers. Figure 1 shows the block diagram of the iValet system.

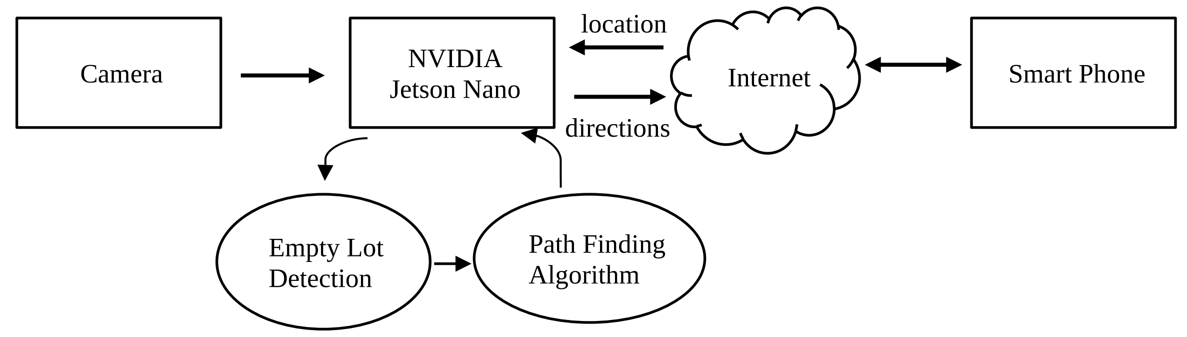


Figure 1. System overview block diagram of iValet

1.2 Motivation

According to parking data from 100,000 locations across more than 8,000 cities, INRIX, a transportation analytics firm, found that on average, drivers in the US, UK and Germany spend 17, 41 and 44 hours per year respectively searching for parking [1]. This amounts to an estimated $345 per driver in terms of wasted time, fuel and emissions in the US [1]. Products that are available in the market such as PlacePod [2], an Internet-of-Things (IoT) enabled magnetic sensor that detects the presence of cars in each available parking spot, primarily require relatively higher set up costs because of the large number of sensors that needs to be purchased for a large parking lot vis-à-vis the more cost-efficient prototype that the team aims to develop. The time and manpower needed to set up the prototype will also be significantly lower than that of traditional IoT devices since it uses a single camera to detect parked cars within in a wide area. iValet provides owners of large parking spaces, such as supermarkets or sporting venues, a cheap solution to reducing the time drivers have to spend circling their parking lots, which ultimately leads to better customer experience, reduced fuel costs for drivers and lower carbon emissions.

1.3 Background

While commercial hardware-based IoT solutions can be used to tackle the parking problem stated above, there are no software-based alternatives that provide an end-to-end service that identifies empty parking spaces based on the type of vehicle and provides drivers with the directions to those empty lots in real time. iValet relies on two key algorithms to work successfully.

For the iValet system to accurately identify the parking spaces in a particular parking lot, the image segmentation model needs to be trained on images of the parking spaces from that specific parking lot. Image segmentation algorithms work by classifying each pixel of a given image with a pixel-wise mask. Different labels in the mask correspond to a different region of interest. In the case of a parking lot, pixels can be classified into (1) empty lot, (2) occupied lot, (3) path for cars. Image segmentation models learn to classify each pixel by training on labeled images that are representative of the images used in deployment and iteratively refine their classification criteria. This training process usually takes a significant amount of time before the algorithms can perform well. However, to speed up the training process such that the iValet system is agnostic to the orientations of the parking spaces, layouts and weather conditions, transfer learning can be used. Transfer learning involves using a pre-trained model that performs relatively well on a particular dataset and training it on a new dataset. This method significantly reduces training time because the model only needs to learn the difference in orientation and not learn to distinguish empty lots from occupied lots again.

The next key component to the iValet system is the pathfinding algorithm. With the coordinates of the vacant lot and the driver's location, the system calculates the shortest path between the two coordinates subject to certain constraints using the pathfinding algorithm. An example of a constraint that might be imposed in a parking lot is the direction of traffic. Pathfinding algorithms generally work by discretizing the given space into a grid and performs a grid-based search over the entire space based on a certain heuristic measure to search for the shortest path between the two points. After calculating the shortest path, iValet will then relay this information to the driver via the web application.

2.1 Project Description, Customer Requirements, and Goals

The iValet managing system will consist of 3 components: the camera detecting system, path planning algorithm, and user interface. The camera detecting system will consist of several cameras, a small NVIDIA computer, and computer vision algorithm. Cameras will take some photos of the parking lots, and then the computer will process those photos with our deep learning models. With the camera detecting system, we can get the empty slots of parking lots. The computer will also work with our path planning algorithm. With the empty slots getting from the camera detecting system, our algorithm will find out the path to the best slot. The user interface can be used to show the empty slots and closest path, which can help.

2.2 Customer & Engineering Requirements

The requirements can be separated into several parts: cameras, time limits, accuracy, and cost. Also, working together with some general requirements, QFD table Figure 2 denotes our requirements.

Cameras:

Heights: Because of the heights of the building, it needs to be no more than 15 meters.

Maximum area: The largest parking lot to detect should not be greater than 10000 m^2.

Frame Rate: 24 frames/s is enough for us to work with real-time detecting.

Time limits:

We need to provide empty slots and paths to users as soon as possible, so we need time. limits for our algorithm. It cannot be more than 300 seconds.

Accuracy:

Users need to access an accurate map. Hence, we need to let our accuracy be greater than 90 percent.

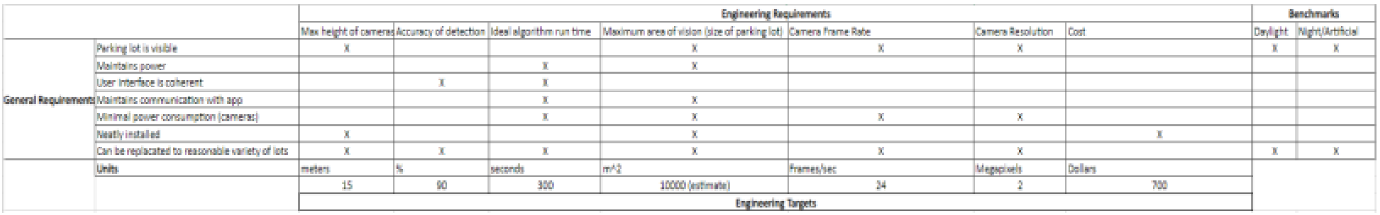
Cost:

We do not want the total cost of this project to be more than 700 dollars. Currently the system costs $468.99.

Prices:

Purchase prices for vendors and involvement of stakeholders would be dependent on the theoretical success of the product.

QFD Table of our requirements:



		Engineering Requirements							Benchmarks	
		Max height of cameras	Accuracy of detection	Ideal algorithm run time	Maximum area of vision (size of parking lot)	Camera Frame Rate	Camera Resolution	Cost	Daylight	Night/Artificial
General Requirements	Parking lot is visible	X			X	X	X		X	X
	Maintains power			X	X					
	User Interface is coherent		X	X						
	Maintains communication with app			X	X					
	Minimal power consumption (cameras)			X	X	X	X			
	Neatly installed	X			X			X		
	Can be replacated to reasonable variety of lots	X	X	X	X	X	X		X	X
	Units	meters	%	seconds	m^2	frames/sec	Megapixels	Dollars		
		15	90	300	10000 (estimate)	24	2	700		
		Engineering Targets								

Figure 2. QFD Table, see full image in Appendix A.

2.3 Goals

Users:

Vendors such as retailers, restaurants, sport stadiums, etc., with large parking lots would benefit from an efficient parking system.

Functionality:

Our project can detect empty slots, find the best path to it, and show that information on the UI.

The accuracy of our detection system should achieve 90%.

The path and empty slots will show in the UI in several seconds.

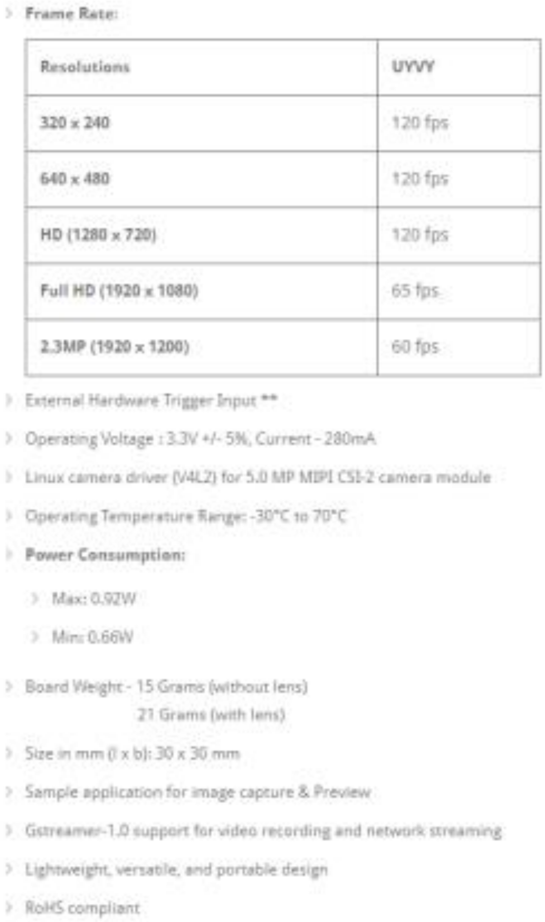
3 Technical Specifications

3.1 Camera

For the camera option, we selected 3 different Nvidia cameras to operate with the Nvidia Jetson Nano processor.

1. e-CAM24_CUNX - Color Global shutter Camera [3]:

Specifications:



- **Frame Rate:**

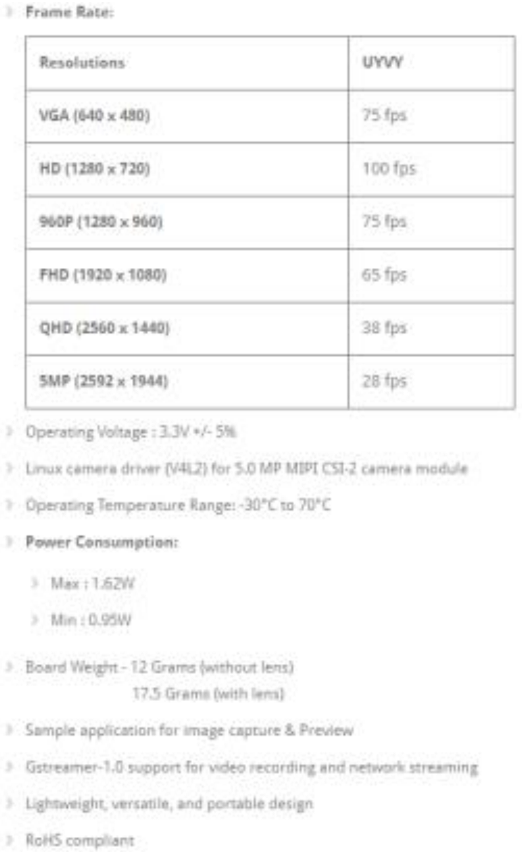
Resolutions	UYVY
320 x 240	120 fps
640 x 480	120 fps
HD (1280 x 720)	120 fps
Full HD (1920 x 1080)	65 fps
2.3MP (1920 x 1200)	60 fps

- External Hardware Trigger Input **
- Operating Voltage : 3.3V +/- 5%, Current - 280mA
- Linux camera driver (V4L2) for 5.0 MP MIPI CSI-2 camera module
- Operating Temperature Range: -30°C to 70°C
- **Power Consumption:**
  - Max: 0.92W
  - Min: 0.66W
- Board Weight - 15 Grams (without lens)
  21 Grams (with lens)
- Size in mm (l x b): 30 x 30 mm
- Sample application for image capture & Preview
- Gstreamer-1.0 support for video recording and network streaming
- Lightweight, versatile, and portable design
- RoHS compliant

Figure 3. e-CAM24_CUNX specifications

2. e-CAM50_CUNX - 5.0 MP NVIDIA® Jetson Xavier™ NX/NVIDIA® Jetson Nano™ Camera [4]

Specifications:



- **Frame Rate:**

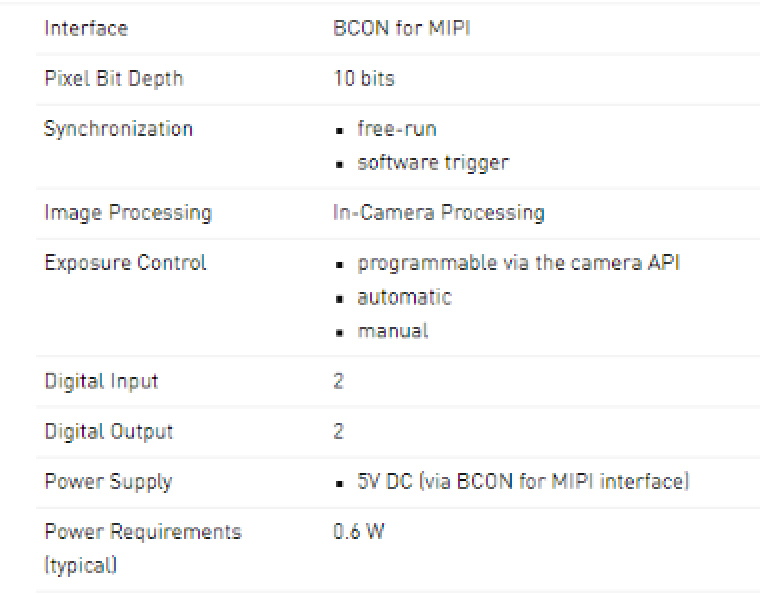
Resolutions	UYVY
VGA (640 x 480)	75 fps
HD (1280 x 720)	100 fps
960P (1280 x 960)	75 fps
FHD (1920 x 1080)	65 fps
QHD (2560 x 1440)	38 fps
5MP (2592 x 1944)	28 fps

- Operating Voltage : 3.3V +/- 5%
- Linux camera driver (V4L2) for 5.0 MP MIPI CSI-2 camera module
- Operating Temperature Range: -30°C to 70°C
- **Power Consumption:**
  - Max : 1.62W
  - Min : 0.95W
- Board Weight - 12 Grams (without lens)
  17.5 Grams (with lens)
- Sample application for image capture & Preview
- Gstreamer-1.0 support for video recording and network streaming
- Lightweight, versatile, and portable design
- RoHS compliant

Figure 4. e-CAM50_CUNX specifications

3. 13 MP, 30 fps Color Camera Rolling Shutter [5]

Specifications:

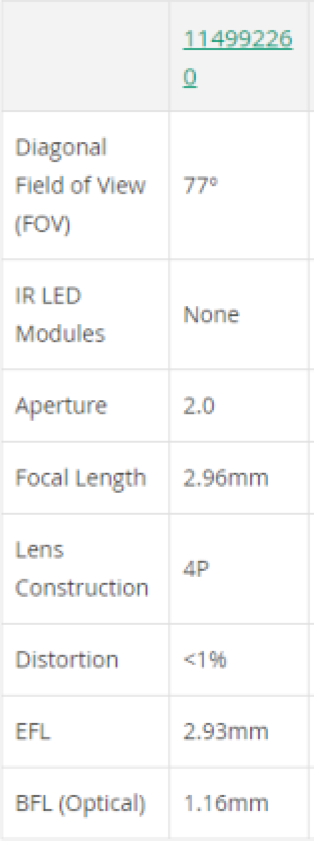


Interface	BCON for MIPI
Pixel Bit Depth	10 bits
Synchronization	▪ free-run ▪ software trigger
Image Processing	In-Camera Processing
Exposure Control	▪ programmable via the camera API ▪ automatic ▪ manual
Digital Input	2
Digital Output	2
Power Supply	▪ 5V DC (via BCON for MIPI interface)
Power Requirements (typical)	0.6 W

Figure 5. 13 MP, 30 fps Color Camera Rolling Shutter

3.2 Nvidia Jetson Nano Developer Kit [6]:

Specifications:

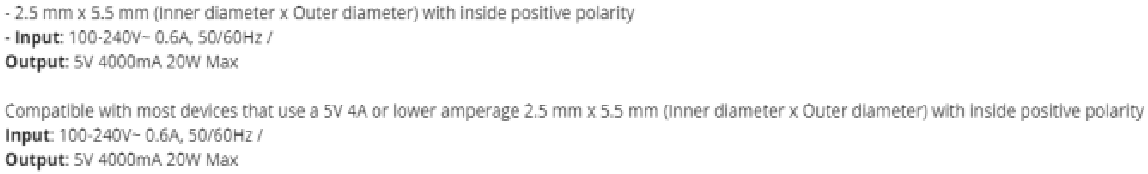


	114992260
Diagonal Field of View (FOV)	77°
IR LED Modules	None
Aperture	2.0
Focal Length	2.96mm
Lens Construction	4P
Distortion	<1%
EFL	2.93mm
BFL (Optical)	1.16mm

Figure 6. Nvidia Jetson Nano Developer Kit

3.3 Power supply [7]:

Specifications:



- 2.5 mm x 5.5 mm (Inner diameter x Outer diameter) with inside positive polarity
- **Input**: 100-240V~ 0.6A, 50/60Hz /
Output: 5V 4000mA 20W Max

Compatible with most devices that use a 5V 4A or lower amperage 2.5 mm x 5.5 mm (Inner diameter x Outer diameter) with inside positive polarity
Input: 100-240V~ 0.6A, 50/60Hz /
Output: 5V 4000mA 20W Max

Figure 7. Power supply

4 Design Approach and Details

4.1 Design Concept Ideation, Constraints, Alternatives, and Tradeoffs

The main components of the project are the camera system, path finding algorithm and the user interface. iValet's design will need to take both the individual cars and their positions in the parking lots and the location of each user's phone as inputs to the algorithm. At least one camera is needed to capture the locations of vehicles in the lot, and a lightweight but relatively fast processor to run the algorithm. The interface will run on either a cross-platform mobile app or an easily accessible webpage.

4.1.1 The Algorithm

The algorithm has three major problems to tackle. Processing the images of the lot, mapping which spaces are available, mapping which are taken and if any filled slots are attached to a user and deploying pathfinding methods to help users find a spot.

To determine which spots are filled, the images from the mounted camera need to be processed. This will be done using OpenCV, an open-sourced library for image processing and machine learning. The library has hundreds of algorithms involving computer vision and machine learning that will be used to read the images of the lot and determine which spots remain. OpenCV was chosen as it's an industry standard for computer vision projects. Also, it supports multiple operating systems (Windows, Linux, Android, and Mac OS) as well as interfaces for C++, MATLAB, Python, and Java, maximizing flexibility and changes throughout the design process [8].

To ensure the park mapping is scalable for any sized lot, a Voronoi diagram will be generated. In cases such as this design, Voronoi diagrams are used to divide the lot into segments, or "Voronoi polygons"[9].

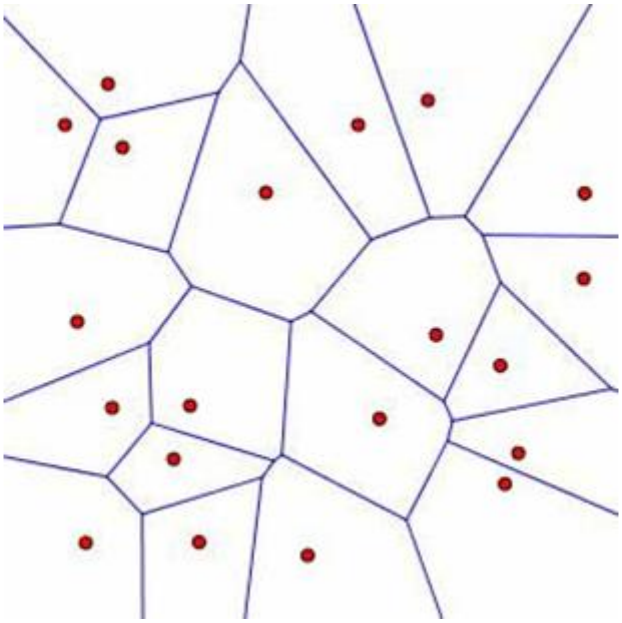


Figure . Sample Voronoi diagram [10]

Different regions, such as in the example Figure 5 above, would serve as parts of a map showing the locations closest to the nearest parking slot. Which will be used to optimize the calculations for the destination of a current user.

Using input from the generated Voronoi diagram, the pathfinding algorithm will then direct drivers to the nearest parking space. The path to that space will be calculated using an A* Algorithm, as it's most valuable when finding the optimal route between two points. A* typically is deployed when find paths to one specific location [11].

4.1.2 The User Interface

Users will be able to access a web page (or alternatively a mobile app) that allows them to verify they are at the correct venue, and select a button to start the navigation process. Ideally, this navigation screen will include a map (similar to Google Maps) that directs them to open spaces. Once they are parked, that spot will be flagged so the processor can direct the user back to their car once they leave the location (again on a map similar to Google Maps).

4.1.3 Design Factors to Consider

a. Global – It's unlikely a semester long project will account for variability across parking lots around the world (which would involve design changes for which side of the road drivers would use and any changes in parking lots).

b. Cultural - The intended solution has little cultural relevance. Especially since similar systems already exist [12].

c. Economic – If this system were to deploy as a marketable product, economic impact would be dependent on the owners of the lot who would deploy this system. Retailers or venues owners would be the ones to determine if the system is useful to them, and that decision will be dependent on their business practices and spending.

d. Environmental - The design is intended to leave any obstructions (like trees, plants, or wildlife ecosystems) undisturbed. Testing sites will be parking lots with spaces that the camera(s) can view without obstacles. Therefore, there should not be any effect on the environment.

e. Sustainability – The intended design is meant to be portable for easy installation. Meaning the system must be physically durable enough to survive maintenance and installation.

f. Manufacturability – The small scale of the prototype will be easy to manufacture. Costs per unit are estimated in Section 7.

g. Ethical - The only ethical concern is that the camera(s) may capture pedestrians. Nothing will be done with that data in the computer vision algorithm. Therefore, ethical concerns are negligible.

h. Health and Safety – Installation of the camera(s) and processor will not be a health or safety issue. The devices do not have a high enough voltage to cause a hazard or any similar harm.

i. Social – There are no likely social implications due to iValet.

j. Political – It is unlikely there would be any political impact caused by iValet.

4. Preliminary Concept Selection and Justification

Chosen components include three options for the camera and one option for the processor to run all software components. **Table 1** is the Decision Matrix for those known components.

4.2.1 Decision Matrix

The decision matrix (**Table 1**) has a Weight that totals to 1.00 (or 100%). The possible rating for each known device is from 0 to 5.

Table 1		**Weight**	**E-CAM50 _CUNX**	**E-CAM24 _CUNX**	**Basler daA4200**	**NVIDIA Jetson Nano**
Criteria	Cost	33.354	5*0.16667	2*0.16667	4*0.16667	5*0.16667
	Camera Frame Rate/Processor Speed	24.99	5*0.2499	5*0.2499	5*0.2499	5*0.2499
	Weight	8.333	5*0.08333	5*0.08333	5*0.08333	5*0.08333
	Size	16.677	5*0.16667	5*0.16667	5*0.16667	5*0.16667
	Durability	8.333	5*0.08333	5*0.08333	5*0.08333	5*0.08333
	Processing Time	24.99	5*0.2499	5*0.2499	5*0.2499	5*0.2499
Total		1.00	5.83335	4.83273	5.49981	5.83335

4.2.2 Current Critical Path Items in Design Process

1. Initial Design phase – choose future testing sites and finalize the scope of the project

2. Initial Software Design - Develop algorithms for computer vision and pathfinding.

3. Initial Hardware Design – Complete concurrently with, or earlier than, the Initial Software Design Phase to ensure the algorithm will have reliable data from the camera(s).

4. Train Algorithms – Train the algorithm using the data set found on Kaggle [13].

5. Test Algorithms on Model Cars – To avoid redesigning simple bugs while testing in a real parking lot, the initial test will be done on a printed map of a parking lot and model cars.

6. Redesign - According to the results of the initial test on the model car parking lot, redesigns to the hardware and software will be made.

7. Test with Real World Lot – When the system is reasonably functional, test will be run inside a real parking lot to verify the level of the project's success (with additional testing and debugging time permitting).

4.2.3 The Camera (Specs listed in Technical Specifications)

In order for images to provide the best results in the algorithm (to track for colors of vehicles and different lighting conditions), the camera must be able to photograph in color. In addition, for the design to be easily implemented in multiple lots, the camera must also be small enough to be portable. This tradeoff is unlikely to affect the design, as the size of the camera mainly changes the frame rate of the device. Because the algorithm is using still images, the frame rate can be low. The current cameras that are being considered (e-CAM50_CUNX, e-CAM24_CUNX, and the Basler daA4200 shown in **Table 1**) all have a frame rate of 30 frames per second [3, 4, 5].

4.2.4 The Processor (Specs listed in Technical Specifications)

Like the camera, the processor needs to be compact enough to be portable, but powerful enough to communicate with the camera(s), run both the algorithm (which includes the computer vision model, pathfinding, and communication with the UI). The current choice is the NVIDIA Jetson Nano Developer

Kit-B01 (listed in **Table 1**), as it’s designed to run modern AI processes and frameworks. It also supports the Linux OS, which is ideal when using OpenCV [6].

4.2.5 GUI

The user interface will be comprised of three pages, depicted in Figure 6. The home page will have a drop-down list to verify the appearance of the driver’s car and the “Find a Parking Spot” button will be used to navigate to the second page, where drivers will be given directions to the closest parking spot. Once they arrive, they can select the “Will you be parked in this spot?” button to help the system verify a car has been parked in that slot. Back on the first page, users can press the “Find where you parked” button to navigate to the third page, which aids them in finding where they have parked their vehicle. The “Scan QR code” will be used to verify the driver has access to the data of the pathfinding algorithm.

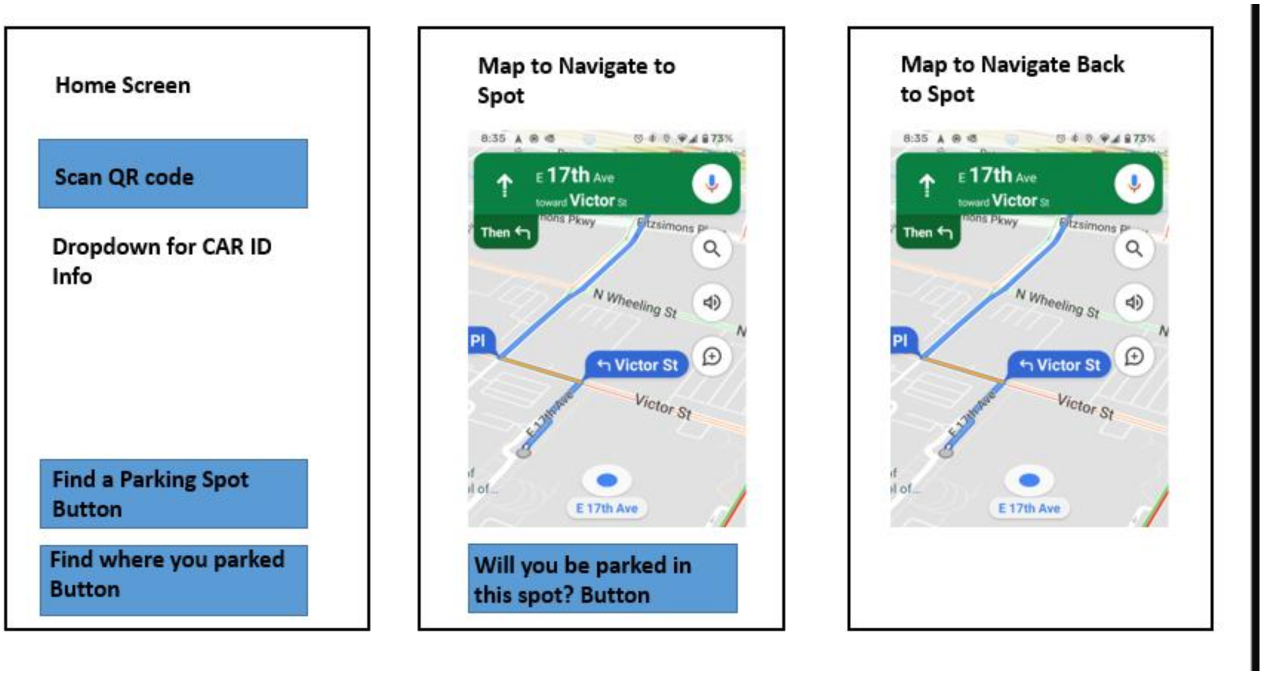


Figure 6. UI Sample. A Google maps image is used to represent a navigation map [14].

4.2.6 Unknowns and Contingency Plans

1. The platform for the UI remains undecided between a cross-platform mobile application (which would likely be built using Flutter or React Native) or an interactive web page.
2. Parking lots for the testing phase have yet to be finalized. That decision will be finalized at the start of the semester.

3. In case the algorithm takes too long to develop, the user interface may need to be omitted from the design scope.

4.3 Engineering Analyses and Experiment

When it comes to the testing phase of the iValet product our team decided to split it into two different testing scales. Right after the prototyping is complete with a working camera and Nvidia took kit board the team will set up small-scale testing in the CRC parking lot. The CRC parking lot can be viewed with a perfect angle in the CRC building, as students of GT, the team member, can conduct this testing safely in the CRC Figure 7. The ensure the experiments meet the specification and the goal of iValet, only a single camera will be used. The reason for limiting hardware during the testing phase is to lower the cost and differ from products already on the market. After results and data are gathered from the small-scale testing, the iValet team will redesign and retrain the CNN model and use the newly made iValet for large-scale testing. The location for large-scale testing is to test a market-ready product with multiple cameras, an LCD screen at the entrance of the parking lot, and a working pathfinding system for guidance. The location of this testing phase is still undecided due to permission and safety, though there are few candidates. One potential parking spot for testing is in The Brady apartment, where one of the members lives, with proper camera angle and permission. Another testing location is the Parking Lot Visitor 2 within Georgia Tech, which also has permission and access to a nearby building for camera installation are available. After the large-scale testing, the team will do a final redesign and training for a market-ready product. If time is permitted, more than one parking lot will be used for testing since iValet wishes to work with every single parking lot as an automotive guidance system.

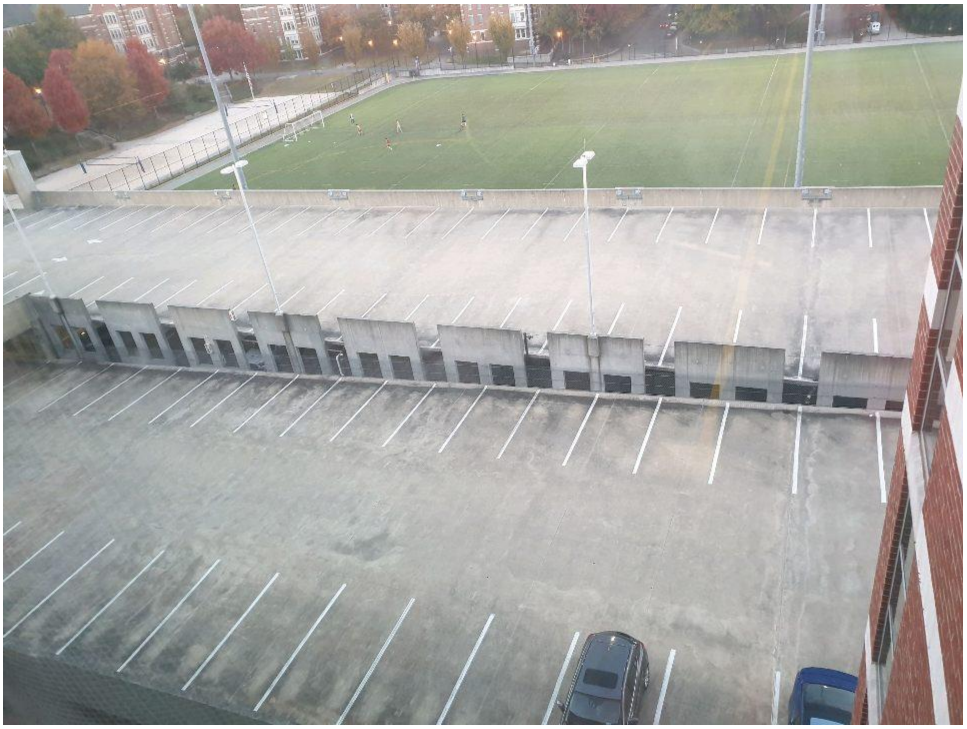


Figure 7. CRC parking lot

4.4 Codes and Standards

4.4.1 Standards

- Wireless communication between the processor, cameras, and end user devices – Use of HTTP and TCP will be critical for safely transferring data between different devices. [15]
- Web applications (for potential UI) for end users – Website applications are typically designed using HTML, CSS, and JavaScript. [16]
- ISO/IEC /IEEE 26351-2015 – Systems and software engineering – Content management for product lifecycle, user and service management documentation - 26351-2015 is an international standard which describes the requirements of any content or data used within a product's software, life cycle, service management system documentation, etc. It specifies the practices regarding content creation, publication and archiving, which will be useful for managing the various publications of the project [17].
- ISO/IEC/IEEE 24748-2-2018 Systems and software engineering — Life cycle management — Part 2: Guidelines for the application of ISO/ IEC/IEEE 15288 (System life cycle processes) The

24748-2-2018 standard discusses the processes needed for using a system-based approach to manage projects. It also highlights the purpose and benefits of applying system-based engineering to solve problems. The approaches described in this standard can be implemented in the group's final project to develop a more holistic solution for end users [18].

4.4.2 Codes

1. Video Recording Laws – Ga. Code 16-11-62(2) - According to Ga. Code 16-11-62(2), it is a crime to implement hidden cameras, or any similar device, to observe someone in a private setting. Therefore, the camera should be as clear a view as possible, and user's (both individuals and those who might install the system) should agree to terms to use the iValet service [19].

5. Project Demonstration

To validate the project, it can be separated into three steps and three design components. The three steps are training and testing with data sets, testing with car models, and testing with real parking lots. The three components are camera detection, path planning algorithm and user interface.

Training and Testing with Data Sets:

For this step, the computer vision algorithm will be tested. After implementing the computer vision algorithm, we need to train and test with a parking lot data set. A data set we plan to use was found on Kaggle [13]. This data set contains more than 700,000 pictures of different parking lots under different weather and lighting conditions. Since this data set is large, we plan to set the proportion of training, validation, and testing to be 8:1:1. Then, we can get the accuracy of testing part, and begin to optimize and change different algorithm until it reaches 90% accuracy.

Testing with car models:

For this step, we can use it to test our computer vision algorithm, path planning algorithm, and user interface. After showing that our computer vision algorithm works well with our data set, we can begin to work with car models or toys. The reason for us to use the car models is that it can be more efficient, and a variety of lots can be tested. Also, it does not have many obstructions like trees with car models. In this phase, we will use the camera(s) to take photos on our car models on a printed parking lot, and then we can check whether those empty slots and best path showing on our user interface correctly or not. Once we find that those parts work well, we can move to the next phase.

Testing with real parking lots:

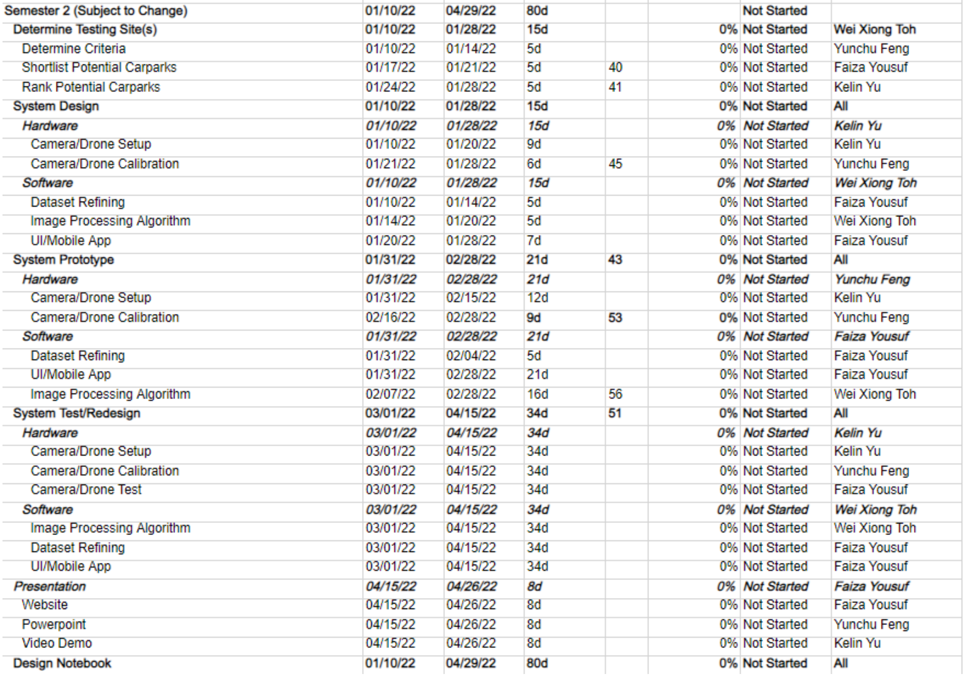
For this final step, we can use it to test our computer vision algorithm, path planning algorithm, and user interface in a real-world setting.

After testing our project in the car models parking lot, we can begin to work on it in the real parking lots. For this step, we need to find out some ideal parking lots and ideal shooting locations. Next, we will verify the functionality of the system in a real lot.

6 Schedule, Tasks, and Milestones:

The following GANTT, Figure 8, chart and PERT, Figure 9, diagram can clearly show our tasks, schedule, and milestones.

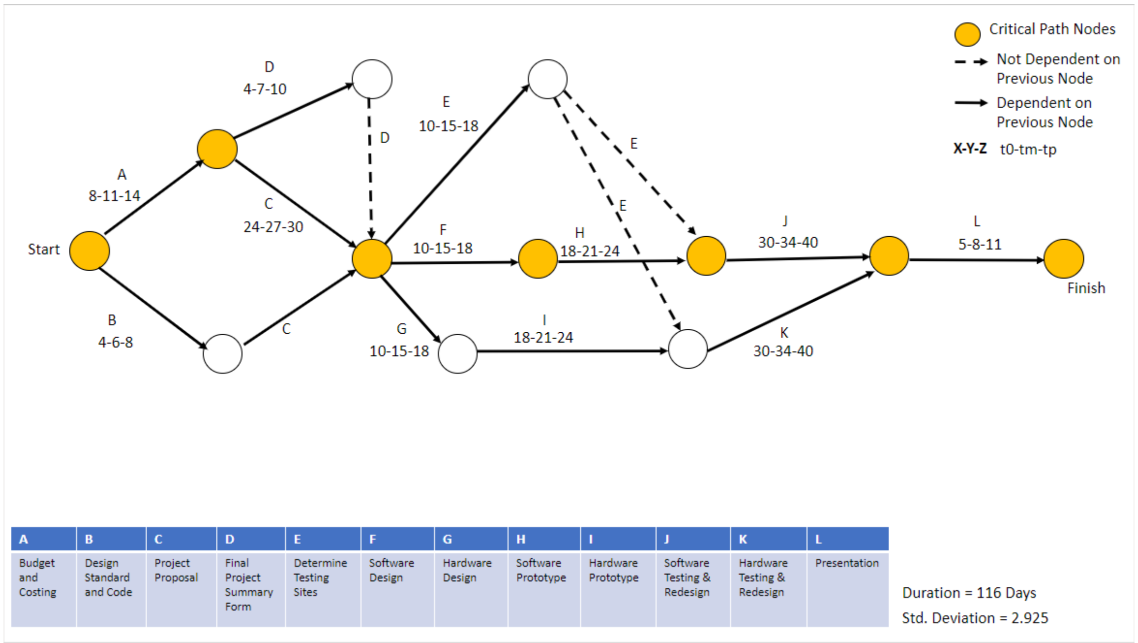
For the Gantt shows our schedule and tasks clearly. It also can show the duration and task leader. For the task leaders, we analyze our ability and allocate those tasks to appropriate students.



Semester 2 (Subject to Change)	**01/10/22**	**04/29/22**	**80d**			**Not Started**	
Determine Testing Site(s)	**01/10/22**	**01/28/22**	**15d**		**0%**	**Not Started**	**Wei Xiong Toh**
Determine Criteria	01/10/22	01/14/22	5d		0%	Not Started	Yunchu Feng
Shortlist Potential Carparks	01/17/22	01/21/22	5d	40	0%	Not Started	Faiza Yousuf
Rank Potential Carparks	01/24/22	01/28/22	5d	41	0%	Not Started	Kelin Yu
System Design	**01/10/22**	**01/28/22**	**15d**		**0%**	**Not Started**	**All**
Hardware	***01/10/22***	***01/28/22***	***15d***		***0%***	***Not Started***	***Kelin Yu***
Camera/Drone Setup	01/10/22	01/20/22	9d		0%	Not Started	Kelin Yu
Camera/Drone Calibration	01/21/22	01/28/22	6d	45	0%	Not Started	Yunchu Feng
Software	***01/10/22***	***01/28/22***	***15d***		***0%***	***Not Started***	***Wei Xiong Toh***
Dataset Refining	01/10/22	01/14/22	5d		0%	Not Started	Faiza Yousuf
Image Processing Algorithm	01/14/22	01/20/22	5d		0%	Not Started	Wei Xiong Toh
UI/Mobile App	01/20/22	01/28/22	7d		0%	Not Started	Faiza Yousuf
System Prototype	**01/31/22**	**02/28/22**	**21d**	**43**	**0%**	**Not Started**	**All**
Hardware	***01/31/22***	***02/28/22***	***21d***		***0%***	***Not Started***	***Yunchu Feng***
Camera/Drone Setup	01/31/22	02/15/22	12d		0%	Not Started	Kelin Yu
Camera/Drone Calibration	02/16/22	02/28/22	**9d**	**53**	**0%**	Not Started	Yunchu Feng
Software	***01/31/22***	***02/28/22***	***21d***		***0%***	***Not Started***	***Faiza Yousuf***
Dataset Refining	01/31/22	02/04/22	5d		0%	Not Started	Faiza Yousuf
UI/Mobile App	01/31/22	02/28/22	21d		0%	Not Started	Faiza Yousuf
Image Processing Algorithm	02/07/22	02/28/22	16d	56	0%	Not Started	Wei Xiong Toh
System Test/Redesign	**03/01/22**	**04/15/22**	**34d**	**51**	**0%**	**Not Started**	**All**
Hardware	***03/01/22***	***04/15/22***	***34d***		***0%***	***Not Started***	***Kelin Yu***
Camera/Drone Setup	03/01/22	04/15/22	34d		0%	Not Started	Kelin Yu
Camera/Drone Calibration	03/01/22	04/15/22	34d		0%	Not Started	Yunchu Feng
Camera/Drone Test	03/01/22	04/15/22	34d		0%	Not Started	Faiza Yousuf
Software	***03/01/22***	***04/15/22***	***34d***		***0%***	***Not Started***	***Wei Xiong Toh***
Image Processing Algorithm	03/01/22	04/15/22	34d		0%	Not Started	Wei Xiong Toh
Dataset Refining	03/01/22	04/15/22	34d		0%	Not Started	Faiza Yousuf
UI/Mobile App	03/01/22	04/15/22	34d		0%	Not Started	Faiza Yousuf
Presentation	***04/15/22***	***04/26/22***	***8d***		***0%***	***Not Started***	***Faiza Yousuf***
Website	04/15/22	04/26/22	8d		0%	Not Started	Faiza Yousuf
Powerpoint	04/15/22	04/26/22	8d		0%	Not Started	Yunchu Feng
Video Demo	04/15/22	04/26/22	8d		0%	Not Started	Kelin Yu
Design Notebook	**01/10/22**	**04/29/22**	**80d**		**0%**	**Not Started**	**All**

Figure 8. Semester 2 GANTT Chart

For the PERT diagram, Figure 6, it can show our tasks, our estimate duration, and longest duration. Also, we made several plans with different tasks to achieve various goals, which can ensure that we can successfully finish them on time.



A	B	C	D	E	F	G	H	I	J	K	L
Budget and Costing	Design Standard and Code	Project Proposal	Final Project Summary Form	Determine Testing Sites	Software Design	Hardware Design	Software Prototype	Hardware Prototype	Software Testing & Redesign	Hardware Testing & Redesign	Presentation

Duration = 116 Days
Std. Deviation = 2.925

Figure 9. Working PERT chart

7. Marketing and Cost Analysis

7.1 Marketing Analysis

The target market consists of venue owners with garages and parking lots in busy areas wanting to enhance the parking experience of their customers. This parking guidance system is not a new concept, with more than a few companies just in the Metro Atlanta area. Hub Parking, for example, will install hardware in every single parking lot and intersection to guide the driver. With sensors and LCD screens almost everywhere in the garage, the setup will cost around $10,000, and 20,000 for larger parking lots [12]. With iValet, only a few cameras are needed which will save the cost to around $1000 - $3000, with some just needing a single camera and central control unit for the entire lot. An example of a low-cost parking lot budget is listed in **Table 2**.

Table 2 Components Cost for Prototype			
Current Parts List with Cost and Citations			
Part	***Part Type***	***Cost***	***Citation***
Full HD Color Global Shutter Camera (e-CAM24_CUNX)	Camera	$149	[3]
5.0 MP Camera (e-CAM50_CUNX)	Camera	$99	[4]
13MP, 30fps Color Camera Rolling Shutter	Camera	$112	[5]
NVIDIA Jetson Nano Developer Kit	Computation/Processing	$99	[6]
Power Supply	Power	$9.99	[7]
Total Cost		$468.99	

7.2 Cost Analysis

Four engineers will complete the design and development of iValet. The total labor hours for the iValet and Labor cost are calculated in **Table 3**. Data for hourly payment is the average entry cost found online [20]. The table has been separated into the meeting, data, model training, software, and hardware portion with the total cost coming to around $4,962 for the labor cost.

Table 3. Estimated Hours per Teammate						
estimates are from https://www.salary.com/ [20]						
Yunchu:	Meetings	Data	Model Training	Testing	Path Finding	Total Hour
Hour:	8	6	10	10	2	36
Salary/Hour:	$30	$35	$45	$30	$35	
Total Cost:	$240	$210	$450	$300	$70	$1,270
Faiza	Meetings	Data	Hardware	Testing	Path Finding	
Hour:	8	6	8	10	2	34
Salary/Hour:	$30	$35	$32	$30	$35	
Total Cost:	$240	$210	$256	$300	$70	$1,076
Wei Xiong	Meetings	Data	Model Training	Testing	Path Finding	

Hour:	8	3	14	10	6	41
Salary/Hour:	$30	$35	$45	$30	$35	
Total Cost:	$240	$105	$630	$300	$210	$1,485
Colin	Meetings	Data	Hardware	Testing	Path Finding	
Hour:	8	1	13	10	4	36
Salary/Hour:	$30	$35	$32	$30	$35	
Total Cost:	$240	$35	$416	$300	$140	$1,131

The statistics over five years of production and service of iValet are given in with the following assumptions. First, with each year the business will expand, meaning the number of sales will increase. For example, the first increase will be five more sales in the third year then ten more in the fourth. Revenue will also be increased from $11,724 to $16,414. Second, after each year, it is assumed iValet will run into certain design issues and maintenance issues with each redesign, costing around $15,000 due to engineering and part usage. Third, all parts are kept at a stable current market price with no volatility. These sales numbers are gathered online from some similar businesses like parking Guidance System, LLC, and Hub Parking. Twenty units in the first year is below average for most auto-guidance system services. Also, most guidance systems services are subscription-based, meaning their customers will remain high each year.

Table 4. Reoccurring Costs										
	Number/ Cost /Salary per yr	Year 1		Year 2		Year 3		Year 4		Year 5
Sales Volume (units)	20		20		25		35		45	
Unit Price	$468.99		$468.99		$468.99		$468.99		$468.99	
Sale Revenue		$9,379.80		$9,379.80		$11,724.75		$16,414.65		$21,104.55
Non-re Cost	$20	$400	$20	$400	$20	$500	$20	$700	$20	$900
1. Research and Dev (based on industry estimate)										
Redesign		$15,000		$15,000		$15,000		$15,000		$0

Engr Change Order		$10,000		$10,000		$10,000		$10,000		$0
2.Production										
Total Parts	$468.99	$9,379.80		$9,379.80		$11,724.75		$16,414.65		$21,104.55
Full HD Color Global shutter Came	$149.00	$2,980		$2,980.		$3,725		$5,215		$6,705
5.0 MP Came	$99.00	$1,980		$1,980		$2,475		$3,465		$4,455
13 MP, 30 fps Color Camera Rolling Shutter	$112.00	$2,240		$2,240		$2,800		$3,920		$5,040
NVIDIA Jetson Nano Developer Kit	$99	$1,980		$1,980		$2,475		$3,465		$4,455
Power Supply	$9.99	$199.80		$199.80		$249.75		$349.65		$449.55

Profits are shown in **Table 5.** Profits are calculated based on the number of sales each year, minus the cost of that year. The cost consists of marketing, packaging, support and maintenance, and distribution. Even though the total profit is very similar, we are increasing the sales each year. This is because more parts and labor are included. iValet is looking to cut even in 5 years, and if subscription-based maintenance is deployed the profit will be even higher.

Table 5	Number /Cost/Salary per yr	Year 1	Year 2	Year 3	Year 4	Year 5
3. Packaging	$10,000	$200,000	$200,000	$250,000	$350,000	$450,000
4. Marketing	$15,000	$300,000	$300,000	$375,000	$525,000	$675,000
5. Sales	$20,000	$400,000	$400,000	$500,000	$700,000	$900,000
6. Distribution	$20,000	$400,000	$400,000	$500,000	700,000	$900,000
7. Support	$15,000	$300,000	$300,000	$375,000	$525,000	$675,000
Total Cost/Year		$43,760	$43,760	$48,450	$57,829	$42,209

Overhead Total	150	$65,639	$65,639	$2,465,639	$86,744	$63,314
Adjusted Cost		$109,399	$109,399	$2,514,089	$144,573	$105,523
Cost/Unit		$468.99	$468.99	$468.99	$468.99	$468.99
Total Profit/Yr		$21,880	$21,880	$2,417,190	$28,915	$21,105

8 Current Status

All parts, components, and major design features have been selected. The team is split into multiple groups each will research their own topics, including CNN, path-finding, camera information, and Nvidia tool kit board. All team member owns a computer and can use other recourses provided by the Georgia Tech ECE department. The team still needs to order the parts and run tests with them, based on the result and the compatibility the plan might change by a minor amount. After the testing is finished the team will deploy the prototype to a real large-scale parking lot.

9 Leadership Roles

The leadership roles are listed as follows in **Table 6**.

Table 6		
Role	**Description**	**Team Member**
Design Lead	In charge of keeping the team to the design schedule and tracking what deliverables were met.	Yunchu Feng
Software Lead	Leads and directs tasks related to the computer vision, pathfinding, and user interface.	Kelin Yu (& Faiza Yousuf as needed)
Hardware Lead	Leads tasks related to setting up camera and processor in testing environments.	Wei Xiong Toh
Testing Lead	In charge delegating and tracking all testing processes.	Wei Xiong Toh
Documentation	Keeps track of all documentation including notes, deliverables, etc.	Yunchu Feng
Webmaster	Will head design and content of the project website.	Faiza Yousuf
Expo Coordinator	Ensures the project is presentable at the expo (necessary PowerPoints, video footage, monitors, posters, etc.)	Kelin Yu

10 References

[1] Inrix. 2021. *Searching for Parking Costs Americans $73 Billion a Year*. [online] Available at: <https://inrix.com/press-releases/parking-pain-us> [Accessed 22 November 2021].

[2] PNI Sensor. 2021. *PlacePod Smart Parking Sensor*. [online] Available at: <https://www.pnicorp.com/placepod/#:~:text=PlacePod%20is%20an%20IoT%2Denabled,location%20of%20available%20parking%20spaces.> [Accessed 22 November 2021].

[3] E-consystems.com. 2021. *Global Shutter Camera for Jetson Xavier NX/TX2 NX/Nano*. [online] Available at: <https://www.e-consystems.com/nvidia-cameras/jetson-xavier-nx-cameras/full-hd-xavier-nx-global-shutter-camera.asp> [Accessed 22 November 2021].

[4] e-consystems. 2021. *e-CAM50_CUNX - 5.0 MP NVIDIA® Jetson Xavier™ NX/NVIDIA® Jetson Nano™ Camera*. [online] Available at: <https://www.e-consystems.com/nvidia-cameras/jetson-nx-cameras/5mp-mipi-camera-board.asp> [Accessed 22 November 2021].

[5] AG, B., 2021. *Basler dart daA4200-30mci (S-Mount) - Area Scan Camera*. [online] Basler AG. Available at: <https://www.baslerweb.com/en/products/cameras/area-scan-cameras/dart/daa4200-30mci-s-mount/> [Accessed 22 November 2021].

[6] Relshop.seeedstudio.com. 2021. *NVIDIA Jetson Nano Developer Kit-B01*. [online] Available at: <https://relshop.seeedstudio.com/NVIDIA-Jetson-Nano-Development-Kit-B01-p-4437.html> [Accessed 22 November 2021].

[7] Newegg.com. 2021. *SMAKN DC 5V/4A 20W Switching Power Supply Adapter 100-240 Ac(US) - Newegg.com*. [online] Available at: <https://www.newegg.com/p/16J-025P-01F82?item=9SIAY4BEF88655&source=region> [Accessed 22 November 2021].

[8] OpenCV. 2021. *About - OpenCV*. [online] Available at: <https://opencv.org/about/> [Accessed 22 November 2021].

[9] Mathworld.wolfram.com. 2021. *Voronoi Diagram -- from Wolfram MathWorld*. [online] Available at: <https://mathworld.wolfram.com/VoronoiDiagram.html> [Accessed 22 November 2021].

[10] American Mathematical Society. 2021. *Feature Column from the AMS*. [online] Available at: <http://www.ams.org/publicoutreach/feature-column/fcarc-voronoi> [Accessed 22 November 2021].

[11] Medium. 2021. *Pathfinding algorithms : the four Pillars.*. [online] Available at: <https://medium.com/@urna.hybesis/pathfinding-algorithms-the-four-pillars-1ebad85d4c6b> [Accessed 30 November 2021].

[12] Kenneth and P. Articles, “Parking guidance systems,” *PARKING GUIDANCE SYSTEMS, LLC.*, 02-Mar-2020. [Online]. Available: https://parkingguidancesystems.com/. [Accessed: 07-Oct-2021].

[13] Kaggle.com. 2021. *Parking Lot Dataset*. [online] Available at: <https://www.kaggle.com/blanderbuss/parking-lot-dataset> [Accessed 22 November 2021].

[14] Support.google.com. 2021. *Extra step required to exit navigation with Google Maps - Google Maps Community*. [online] Available at: <https://support.google.com/maps/thread/70106536/extra-step-required-to-exit-navigation-with-google-maps?hl=en> [Accessed 22 November 2021].

[15] Learning Center. 2021. *What is OSI Model | 7 Layers Explained | Imperva*. [online] Available at: <https://www.imperva.com/learn/application-security/osi-model/> [Accessed 22 November 2021].

[16] Hack Reactor, “What is JavaScript used for?,” Hack Reactor, 26-Aug-2021. [Online]. Available: https://www.hackreactor.com/blog/what-is-javascript-used-for. [Accessed: 22-Nov-2021].

[17] "ISO/IEC/IEEE International Standard for Systems and software engineering -- Content management for product life-cycle, user, and service management documentation," in ISO/IEC/IEEE 26531:2015 (E) , vol., no., pp.1-60, 15 May 2015, doi: 10.1109/IEEESTD.2015.7106441.

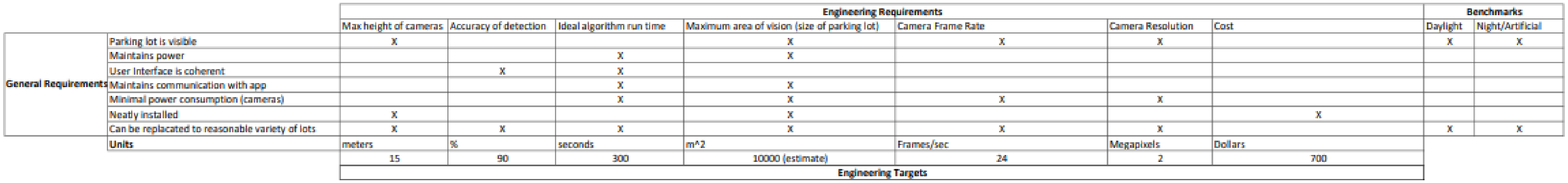
[18] “ISO/IEC/IEEE Draft International Standard - Systems and Software Engineering-- Life Cycle Management-- Part 2: Guidelines for the Application of ISO/IEC/IEEE 15288 (System Life Cycle Processes). IEEE, 2018, pp. 1–69.”

[19] Chanco Schiffer Law, LLC. 2021. Georgia’s Video Recording Laws. [online] Available at: [Accessed 22 October 2021].

[20] Salary.com, “Unlock the power of pay,” Salary.com, 01-Apr-2021. [Online]. Available: https://www.salary.com/. [Accessed: 22-Nov-2021].

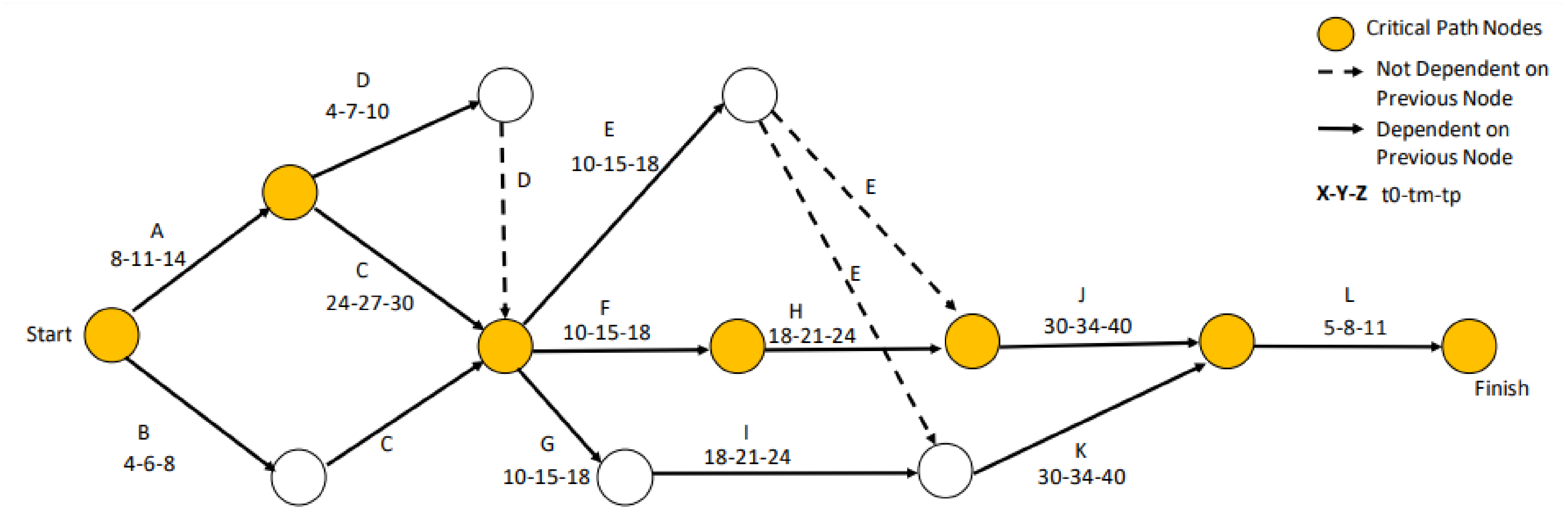
Appendix

QFD



		Engineering Requirements							Benchmarks	
		Max height of cameras	Accuracy of detection	Ideal algorithm run time	Maximum area of vision (size of parking lot)	Camera Frame Rate	Camera Resolution	Cost	Daylight	Night/Artificial
General Requirements	Parking lot is visible	X			X	X	X		X	X
	Maintains power			X	X					
	User Interface is coherent		X	X						
	Maintains communication with app			X	X					
	Minimal power consumption (cameras)			X	X	X	X			
	Neatly installed	X			X			X		
	Can be replacated to reasonable variety of lots	X	X	X	X	X	X		X	X
	Units	meters	%	seconds	m^2	Frames/sec	Megapixels	Dollars		
		15	90	300	10000 (estimate)	24	2	700		
		Engineering Targets								

PERT

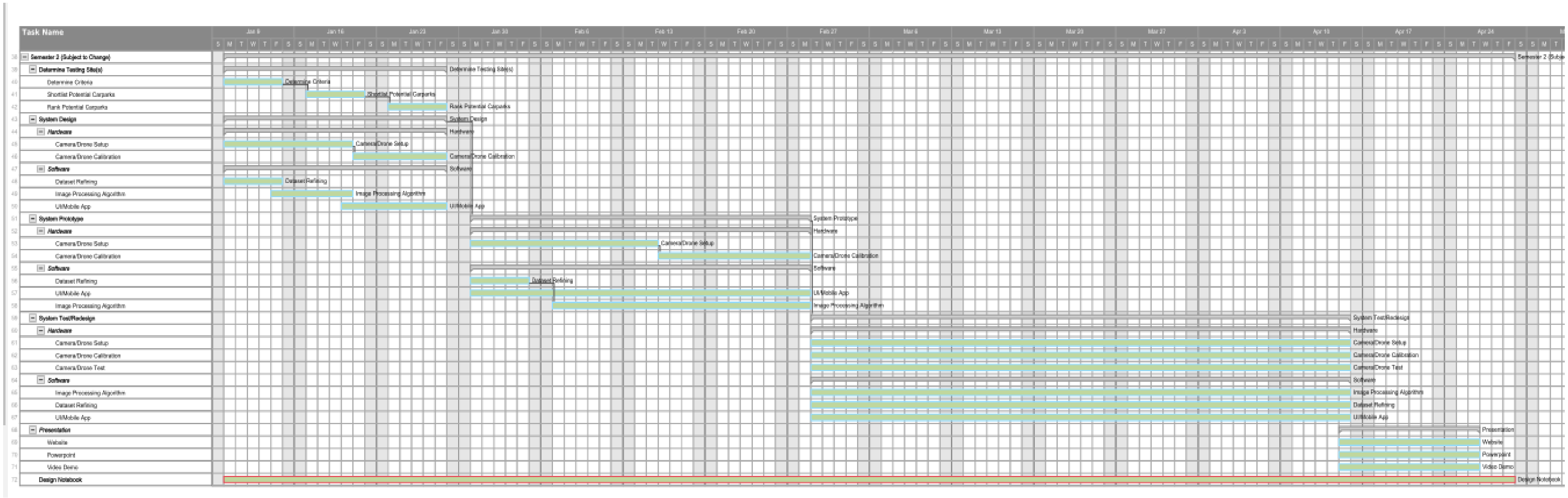


A	B	C	D	E	F	G	H	I	J	K	L
Budget and Costing	Design Standard and Code	Project Proposal	Final Project Summary Form	Determine Testing Sites	Software Design	Hardware Design	Software Prototype	Hardware Prototype	Software Testing & Redesign	Hardware Testing & Redesign	Presentation

Duration = 116 Days

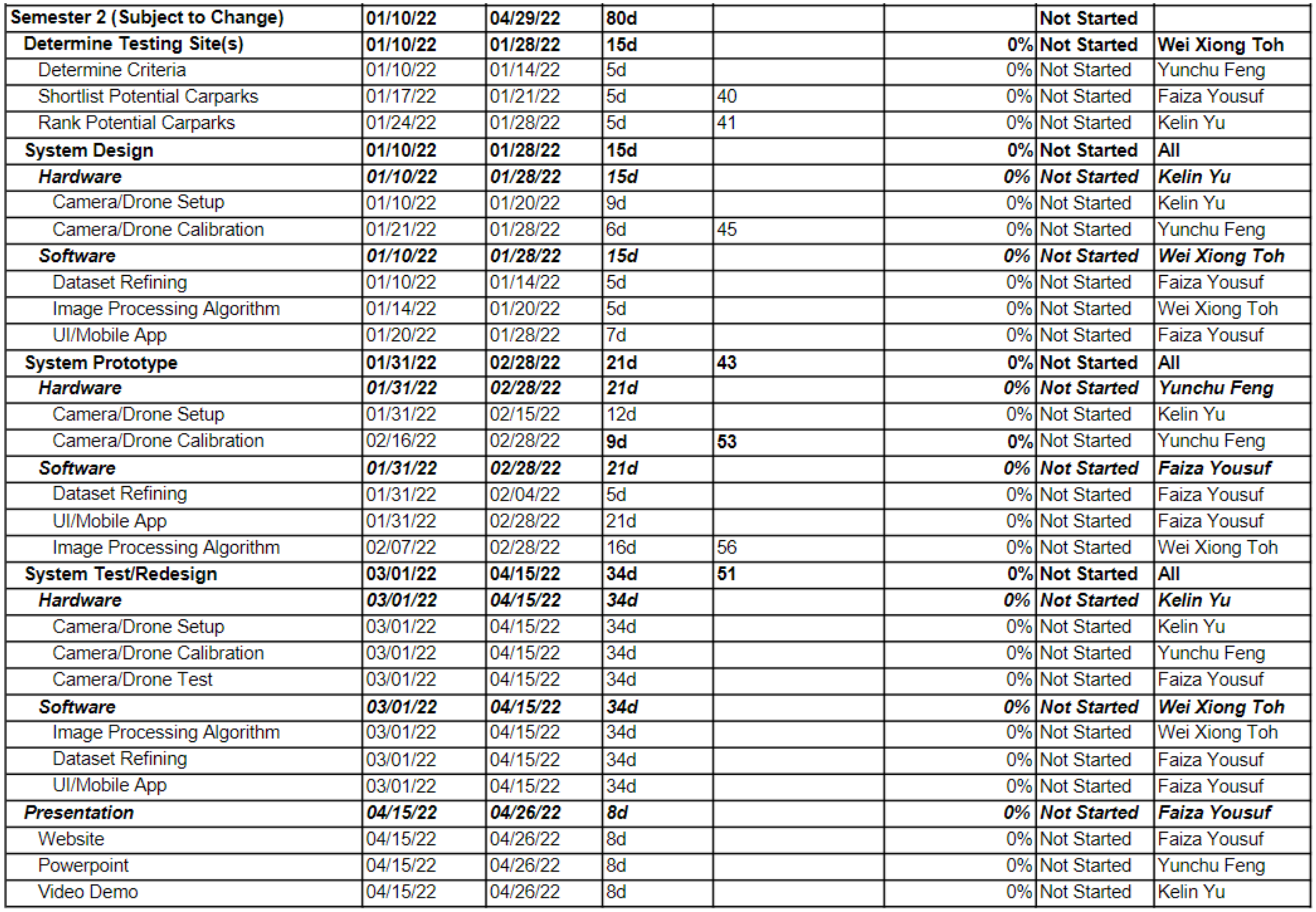
Std. Deviation = 2.925

GANTT



	Task Name
38	Semester 2 (Subject to Change)
39	Determine Testing Site(s)
40	Determine Criteria
41	Shortlist Potential Carparks
42	Rank Potential Carparks
43	System Design
44	*Hardware*
45	Camera/Drone Setup
46	Camera/Drone Calibration
47	*Software*
48	Dataset Refining
49	Image Processing Algorithm
50	UI/Mobile App
51	System Prototype
52	*Hardware*
53	Camera/Drone Setup
54	Camera/Drone Calibration
55	*Software*
56	Dataset Refining
57	UI/Mobile App
58	Image Processing Algorithm
59	System Test/Redesign
60	*Hardware*
61	Camera/Drone Setup
62	Camera/Drone Calibration
63	Camera/Drone Test
64	*Software*
65	Image Processing Algorithm
66	Dataset Refining
67	UI/Mobile App
68	*Presentation*
69	Website
70	Powerpoint
71	Video Demo
72	Design Notebook

Semester 2 Timeline



Semester 2 (Subject to Change)	**01/10/22**	**04/29/22**	**80d**			**Not Started**	
Determine Testing Site(s)	**01/10/22**	**01/28/22**	**15d**		**0%**	**Not Started**	**Wei Xiong Toh**
Determine Criteria	01/10/22	01/14/22	5d		0%	Not Started	Yunchu Feng
Shortlist Potential Carparks	01/17/22	01/21/22	5d	40	0%	Not Started	Faiza Yousuf
Rank Potential Carparks	01/24/22	01/28/22	5d	41	0%	Not Started	Kelin Yu
System Design	**01/10/22**	**01/28/22**	**15d**		**0%**	**Not Started**	**All**
Hardware	***01/10/22***	***01/28/22***	***15d***		***0%***	***Not Started***	***Kelin Yu***
Camera/Drone Setup	01/10/22	01/20/22	9d		0%	Not Started	Kelin Yu
Camera/Drone Calibration	01/21/22	01/28/22	6d	45	0%	Not Started	Yunchu Feng
Software	***01/10/22***	***01/28/22***	***15d***		***0%***	***Not Started***	***Wei Xiong Toh***
Dataset Refining	01/10/22	01/14/22	5d		0%	Not Started	Faiza Yousuf
Image Processing Algorithm	01/14/22	01/20/22	5d		0%	Not Started	Wei Xiong Toh
UI/Mobile App	01/20/22	01/28/22	7d		0%	Not Started	Faiza Yousuf
System Prototype	**01/31/22**	**02/28/22**	**21d**	**43**	**0%**	**Not Started**	**All**
Hardware	***01/31/22***	***02/28/22***	***21d***		***0%***	***Not Started***	***Yunchu Feng***
Camera/Drone Setup	01/31/22	02/15/22	12d		0%	Not Started	Kelin Yu
Camera/Drone Calibration	02/16/22	02/28/22	**9d**	**53**	**0%**	Not Started	Yunchu Feng
Software	***01/31/22***	***02/28/22***	***21d***		***0%***	***Not Started***	***Faiza Yousuf***
Dataset Refining	01/31/22	02/04/22	5d		0%	Not Started	Faiza Yousuf
UI/Mobile App	01/31/22	02/28/22	21d		0%	Not Started	Faiza Yousuf
Image Processing Algorithm	02/07/22	02/28/22	16d	56	0%	Not Started	Wei Xiong Toh
System Test/Redesign	**03/01/22**	**04/15/22**	**34d**	**51**	**0%**	**Not Started**	**All**
Hardware	***03/01/22***	***04/15/22***	***34d***		***0%***	***Not Started***	***Kelin Yu***
Camera/Drone Setup	03/01/22	04/15/22	34d		0%	Not Started	Kelin Yu
Camera/Drone Calibration	03/01/22	04/15/22	34d		0%	Not Started	Yunchu Feng
Camera/Drone Test	03/01/22	04/15/22	34d		0%	Not Started	Faiza Yousuf
Software	***03/01/22***	***04/15/22***	***34d***		***0%***	***Not Started***	***Wei Xiong Toh***
Image Processing Algorithm	03/01/22	04/15/22	34d		0%	Not Started	Wei Xiong Toh
Dataset Refining	03/01/22	04/15/22	34d		0%	Not Started	Faiza Yousuf
UI/Mobile App	03/01/22	04/15/22	34d		0%	Not Started	Faiza Yousuf
Presentation	***04/15/22***	***04/26/22***	***8d***		***0%***	***Not Started***	***Faiza Yousuf***
Website	04/15/22	04/26/22	8d		0%	Not Started	Faiza Yousuf
Powerpoint	04/15/22	04/26/22	8d		0%	Not Started	Yunchu Feng
Video Demo	04/15/22	04/26/22	8d		0%	Not Started	Kelin Yu