Prepared by: Preston Ott, Younes Younes, Joshua Swanner, Joshua Moya, Luis Santillan
Instructor: Johnathan Votion Ph.D & August Allo

Product Requirements And Specifications Document (PRSD) OYSMS DigiPark

Approvals

Title	Print	Signature	Date
OYSMS Digi Park	Younes Younes	Younes Younes	1/25/2023
OYSMS Digi Park	Joshua Swanner		1/25/2023
OYSMS Digi Park	Luis Santillan		1/25/2023
OYSMS Digi Park	Preston Ott		1/25/2023
OYSMS Digi Park	Joshua Moya		1/25/2023

Revision History

Revision	Summary of Change	Originator
X0	New Document	UTSA-ECE
X1	Clarification	UTSA-ECE
X2	Revisions to accommodate the scope of the product	UTSA-ECE

1. Introduction

This document contains the system requirements for OYSMS DigiPark. These requirements have been derived from several sources, including regulations from the University Parking and Traffic Rules & Regulations at UTSA and competing products as a guideline for our project.

1.1 Purpose of This Document

This document is intended to guide development of OYSMS DigiPark. It will go through several stages during the course of the project:

1. **Draft:** The first version, or draft version, is compiled after requirements have been discovered, recorded, classified, and prioritized.
2. **Proposed:** The draft document is then proposed as a potential requirements specification for the project. The proposed document should be reviewed by several parties, who may comment on any requirements and any priorities, either to agree, to disagree, or to identify missing requirements. Readers include end-users, developers, university faculty, course instructor, and any other stakeholders.
3. **Validated:** Once the various stakeholders have agreed to the requirements in the document, it is considered validated.
4. **Approved:** The validated document is accepted as an appropriate statement of requirements for the project. The developers then use the requirements document as a guide to implementation and to check the progress of the project as it develops.

1.3 Scope of the Product

The problem of high traffic parking lots or finding an open parking space is one of the most significant issues faced by the drivers in a parking lot. Various solutions have been proposed by different companies. Our proposed solution is a mobile application based on the development of a pre-mapped parking lot and a scanner that detects free spots in a parking lot. The mobile application is highly efficient and offers distinct features such as real-time updates and statistics of the free parking spaces along with the closest free spots based on the user's location.

1.4 Case for the Product (Need)

A problem that has been an issue for some time now is the parking around campus. Finding a parking spot on campus can be tough because there are so many people trying to go to class, and with a limited amount of spots it is hard to know if other parking lots have open parking spots. This will help many people find parking on campus with little to no struggles. Approximately, parking costs 73 billion dollars a year and Americans spend 17 hours per year searching for a parking spot. All businesses and buildings that have parking lots will benefit from this solution because customers avoid shops due to inadequate parking space. All people who own or use a motor vehicle will benefit from this solution through fuel and time savings.

2. General Description

The problem we are going to solve is finding a parking space in a crowded parking lot, which was brought to our attention by our groupmate Preston on the first day of school at UTSA. In a large parking lot, it can be difficult to navigate and locate an open space and if an open space is obscured, you have no way of telling where it is. We want to solve this unawareness of all the open parking spaces in a parking lot so a driver can locate an open space obscured from view or an open space that could be thousands of feet away. The way we will accomplish this is by placing several devices in a parking lot to detect which parking spots are open/taken. We will have an app that accompanies this which the user will use to find an open parking spot.

2.1 Product Perspective

The idea for OYSMS DigiPark came from Preston Ott on the first day of school when the realization came to him that the parking situation at UTSA was a problem. Digipark serves to offer an efficient solution to people's problems when finding a parking spot in a parking lot. The project is being developed by our OYSMS group of 5 for senior design II at UTSA and is currently being funded by Mr. Chad Webster. On completion, anyone with a phone and a car will benefit from the project when searching for a parking spot.

2.2 Product Functions

There are two parts to the project, A device that goes into the parking lots and an app for smartphones. The device will detect which parking spots in a designated parking lot are taken or empty. The app is the main function that the users will be interacting with. The app will include a straightforward map of the parking lot, provide real time updates of new open parking spots within a parking lot, give the percentage of open spots available in a lot, give data on the availability of parking in a given parking lot throughout the day, portray the user's location within the parking lot, and highlight the nearest available parking spots.

2.3 User Characteristics

The expected demographic for the product is anyone with a car and a smartphone. The reason they would use the product is if they are having trouble looking for a parking spot in a busy parking lot. The app would help them in this search or help them decide to park in a low traffic parking lot. Problems may include navigation of the app if the user is not well versed in using their smartphone and applications on it.

2.4 General Constraints

The main constraint of the application are the guidelines placed on by the app stores and the capabilities of the programming language being used to develop the app. For the device there are rules and standards given by the FCC when transmitting radio frequencies. Another constraint would be communicating the devices' information to the server running our application.

2.5 Assumptions and Dependencies

An assumption involving the device is the power supply, as of now the device will be battery powered and will need maintenance when switching batteries, unless an indefinite power supply were to be provided. Another assumption of the device is the distance from the furthest parking lot device and the server will not exceed 1km as the RF transmitter will be approximately limited to that range. It is assumed that UTSA can help us in setting up the device in the parking lot to assist with testing and upon product completion.

2.6 Objectives:

1.) Test each part, and assemble (scanner and transmitter system) circuit and develop program for (scanner and transmitter system)

1. Find or Develop code or circuit test to test each component individually
2. Build subsystems of circuit, develop test for subsystems and verify subsystems work
3. Assemble complete (scanner and transmitter system), write code for test, test complete circuit can take images and rotate camera to desired positions.
4. Develop Final program to control (scanner and transmitter system) and test it.

2.) Develop program for server scanner communication

5. Determine python libraries and resources for server (scanner and transmitter) system communication program and create software flow chart for program.
6. Develop a basic server (scanner and transmitter system) communication program to transmit and receive basic strings of data and test it.
7. Improve server (scanner and transmitter system) communication program to transmit and receive formatted messages.
8. Improve server program to process images, remove noise, reformat image to specific dimensions, and compress image.

3.) Develop Machine Learning Model for car in parking space detection (Convolutional Neural Network most likely)

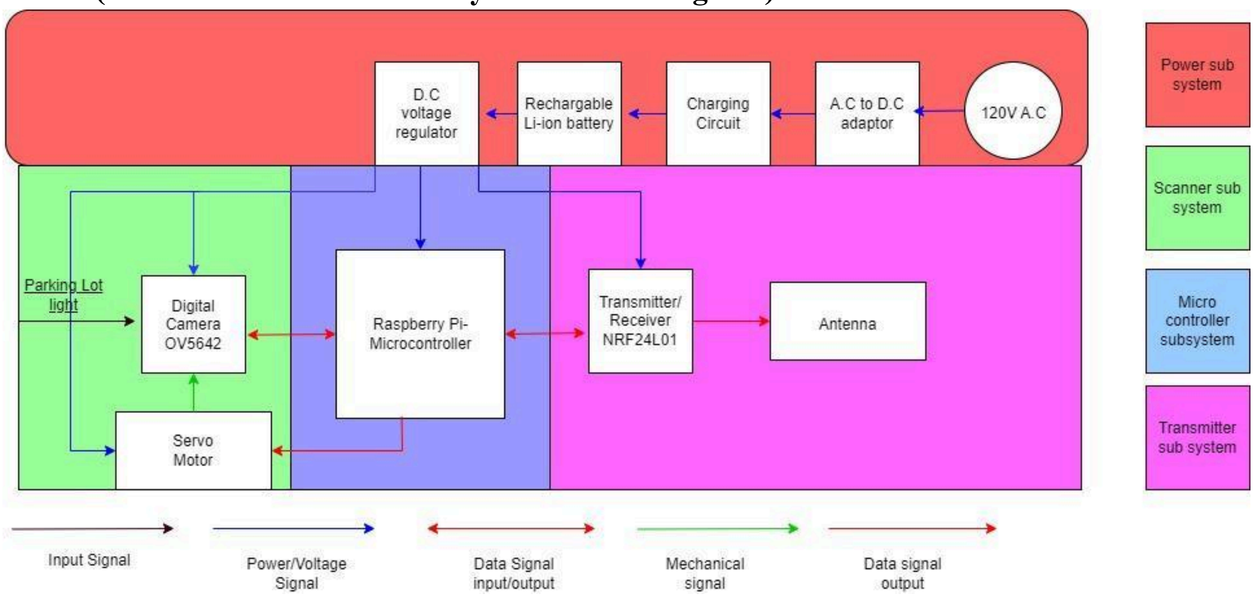
1. Determine type of Machine Learning Model and python libraries/resources for model
2. Find or create labeled data to train Machine Learning models with.
3. Write the program for Machine Learning model for training and testing.
4. Train Machine Learning Model and test it on labeled data.

5. Improve Machine Learning model until classification accuracy threshold is reached.

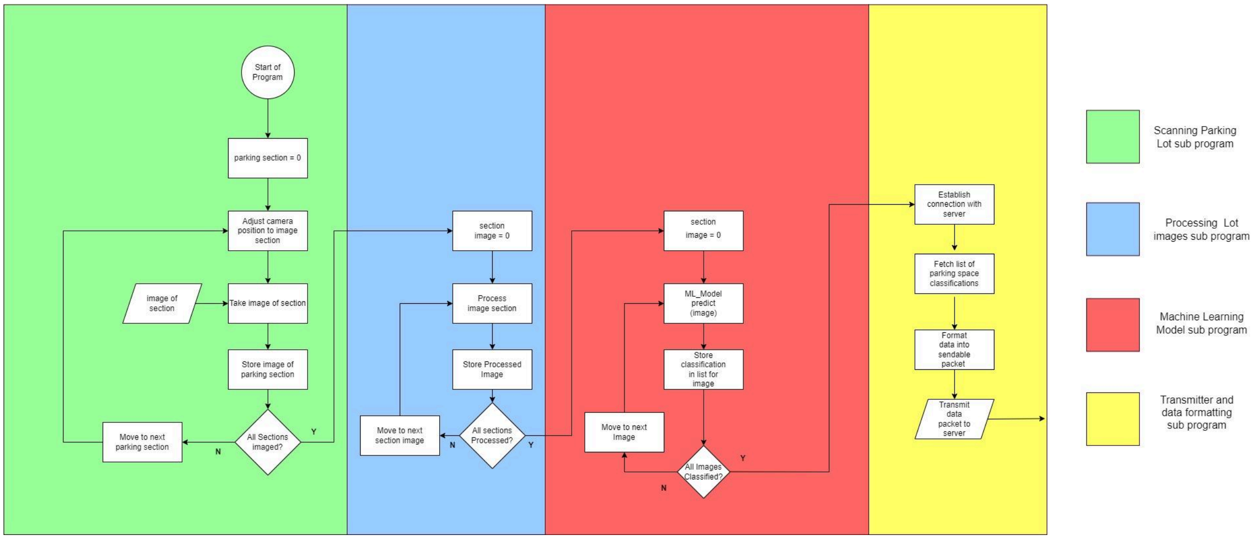
4.) Develop parking lot Mobile app

1. Determine all libraries/ modules needed to develop app and create detailed software flow chart for the app.
2. Write program for app for testing purposes only to run on PC or laptop. Basic program should include basic GUI and server client communication. Test app with (scanner and transmitter system) and server.
3. Improve app develop user-based GUI, Parking lot map from received data, improved menu. Test app on a mobile device such as iPhone or android phone.
4. Deploy the app into a store and test the deployed app.
5. Improve the app by adding navigation feature for mobile device using its GPS module and statistics on parking lot data feature and test new features.

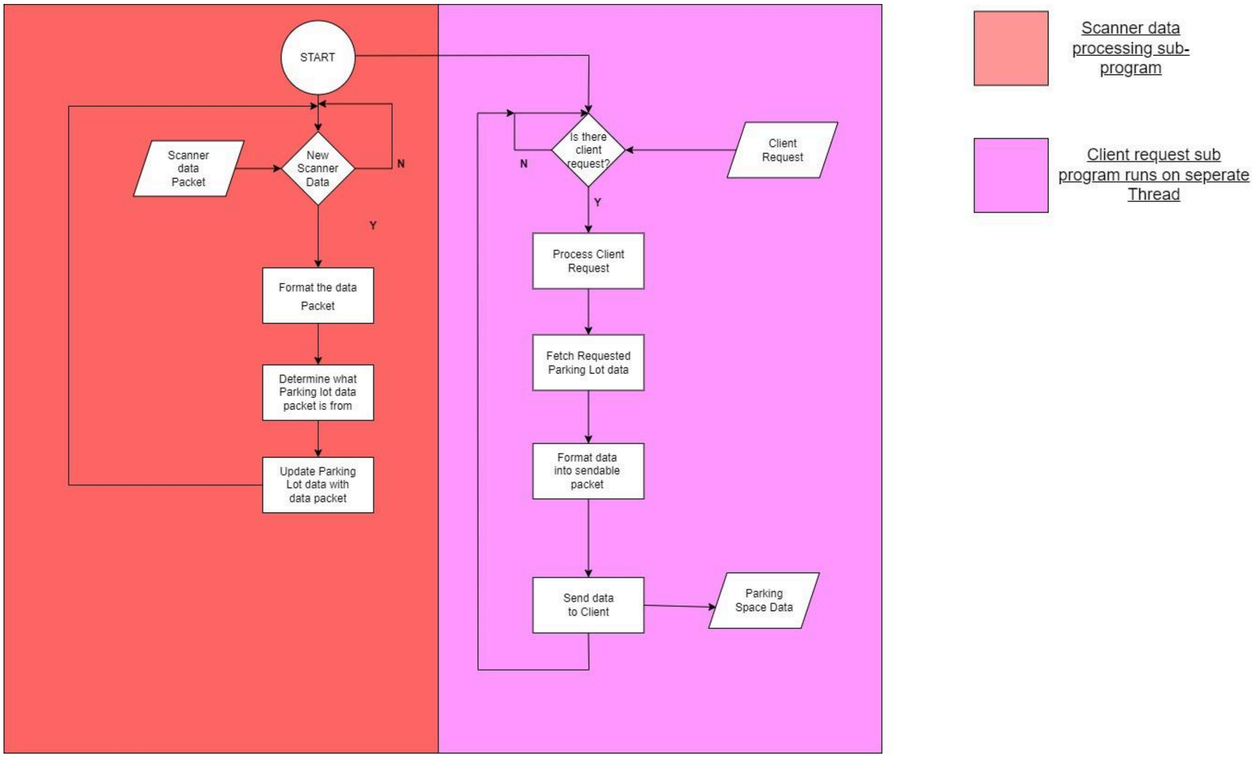
(Scanner and Transmitter system Block Diagram)



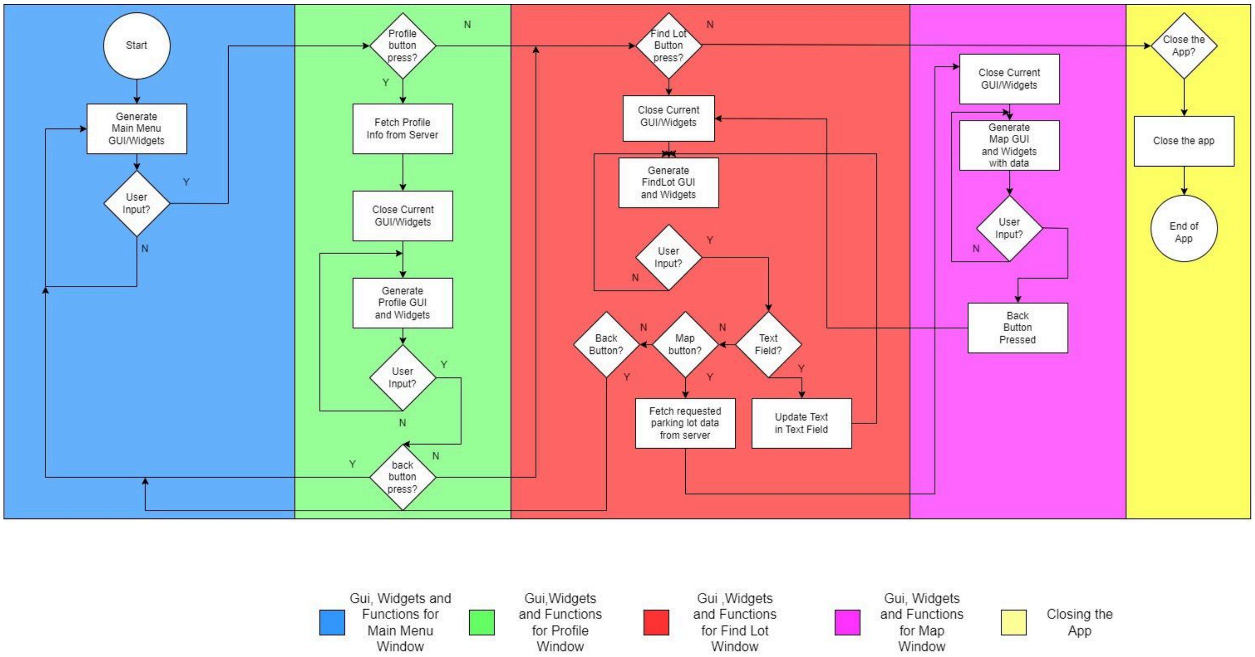
(Scanner and Transmitter and ML Model Program Software Flow Chart)


Start of Program
parking section = 0
Adjust camera position to image section
image of section
Take image of section
Store image of parking section
All Sections imaged?
Y
N
Move to next parking section
section image = 0
Process image section
Store Processed Image
All sections Processed?
N
Y
Move to next section image
section image = 0
ML_Model predict (image)
Store classification in list for image
All Images Classified?
Y
N
Move to next Image
Establish connection with server
Fetch list of parking space classifications
Format data into sendable packet
Transmit data packet to server
Scanning Parking Lot sub program
Processing Lot images sub program
Machine Learning Model sub program
Transmitter and data formatting sub program

(Software Flow Chart of Server and (Scanner and Transmitter System) Communication Program and Client Communication Program)


START
Scanner data Packet
New Scanner Data
N
Y
Format the data Packet
Determine what Parking lot data packet is from
Update Parking Lot data with data packet
Is there client request?
N
Y
Client Request
Process Client Request
Fetch Requested Parking Lot data
Format data into sendable packet
Send data to Client
Parking Space Data
Scanner data processing sub-program
Client request sub program runs on seperate Thread

(Software Flow chart for Parking app Program)


Start
Generate Main Menu GUI/Widgets
User Input?
Y
N
Profile button press?
N
Y
Fetch Profile Info from Server
Close Current GUI/Widgets
Generate Profile GUI and Widgets
User Input?
Y
N
back button press?
Y
N
Find Lot Button press?
N
Close Current GUI/Widgets
Generate FindLot GUI and Widgets
User Input?
Y
N
Back Button?
N
Y
Map button?
N
Y
Text Field?
Y
Fetch requested parking lot data from server
Update Text in Text Field
Close Current GUI/Widgets
Generate Map GUI and Widgets with data
User Input?
N
Back Button Pressed
Close the App?
Close the app
End of App
Gui, Widgets and Functions for Main Menu Window
Gui,Widgets and Functions for Profile Window
Gui ,Widgets and Functions for Find Lot Window
Gui, Widgets and Functions for Map Window
Closing the App

●

2.7 Plan

High Level Task Description:

- Create a working autonomous scanner system that can be implemented in a parking lot.
- Create a working transmitter/server system to communicate with the autonomous scanner system.
- Create a Machine Learning Model for classifying parking spaces as empty or unoccupied.
- Create a mobile app to display information on parking lot utilizing scanner system and transmitter/server system.

Principles and Theories to be used:

- Wireless communication theory
- Machine Learning theory
- Wired Communication
- Image Processing
- Robotics
- Mobile app development and design
- Optics

Complications:

- Wireless transmission of Strings of data.How to Handle noise in transmission.
- Finding/creating labeled data to train Machine Learning Model.
- Client server communication for mobile app.

Design approach/methodology:

- Developing Circuit Schematics before Implementation.
- Developing fundamental prototype programs for testing feasibility and proof of concept.

- Develop subsystems of circuit and test them independently.

Implementation of Project:

- Modular design of scanner circuit.
- Modular design of server app.
- Modular design of mobile app.
- Independent development of machine learning model.
- Server and Scanner Communication is Separate from Client Server Communication.

Validation and Verification:

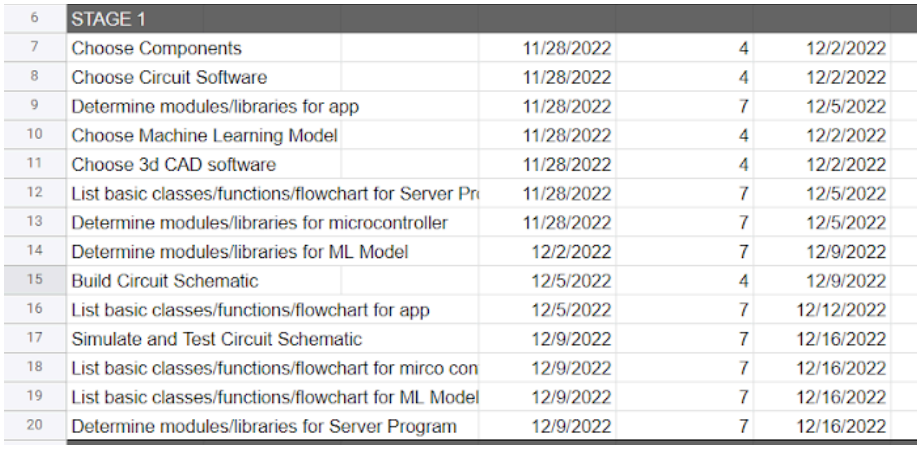
- Machine learning models will be validated with test data and will use cross validation for overfitting/underfitting avoidance (Multiple machine learning models will be trained).
- Classes and functions for programs independently tested with preset inputs and expected outputs.
- Individual circuit component testing with test programs.
- Individual circuit module test.
- Circuit modules integration testing.
- Wireless communication test between server and transmitter under different conditions.
- Early prototype testing for all major engineering requirements in a small parking lot.

2.8 Schedule Requirements

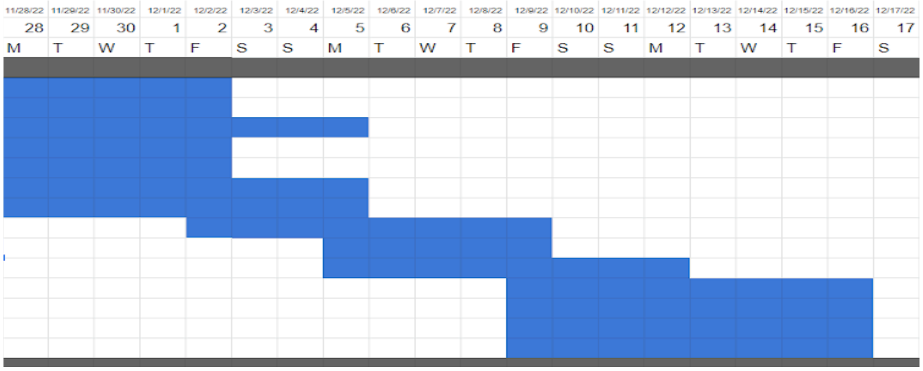
Overview:

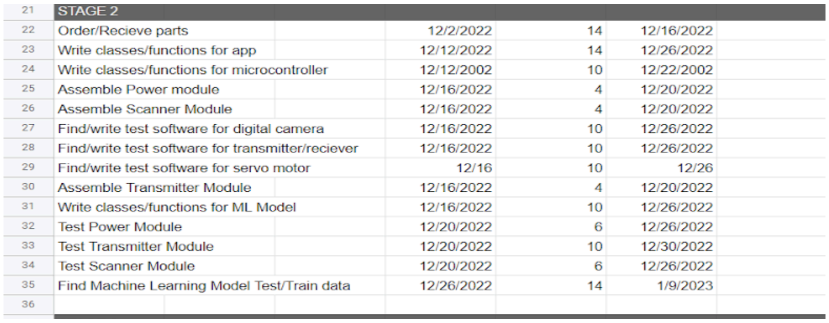
The schedule requirements are broken down into seven segments. Each segment depends on the other being completed. The first segment is the high-level design/ preparation/choice stage. This stage picks all software and hardware components and makes high level design decisions. This stage will take 2 weeks to complete from the end of fall semester. The second is testing individual components. This stage will develop and test individual circuit components and software components as well as assemble and test the different modules of circuit. This stage will take about 1 month to complete during winter break. The third stage is the early prototype stage. This consists of assembling our bare hardware and software modules to make a very basic prototype to test all major functionality of our project such as basic transmission of data, taking images of parking lot autonomously with scanner, Machine Learning model car positioning and more. This stage will take about 2 and a half weeks to complete during winter break. The fourth stage is testing early prototype stage. This stage consists of testing the early prototype and identifying all bugs in each module and debugging them and improving the server and scanner communication to transmit images. This stage will take about a month to complete during the start of the spring semester. The fifth stage is the improvement stage. This stage will improve on the basic prototype to include more functionality in mobile app and server program and transfer scanner/transmitter circuit into a PCB or perf board with a 3d printed case. These improvements will also be tested in this stage. This stage will take about a month to complete during the spring semester. The sixth stage is the testing and finalizing stage. This stage consists of testing and debugging the improved prototype in an actual parking lot. This stage will finalize the 3 programs for the project and deploy our app into an app store. This stage will take about 3 weeks to complete during spring semester. The seventh and last stage is the preparation stage. This stage consists of preparing project for tech symposium and final testing/debugging of the final prototype. This stage will take about 3 weeks to complete during spring semester.

Gantt Chart:

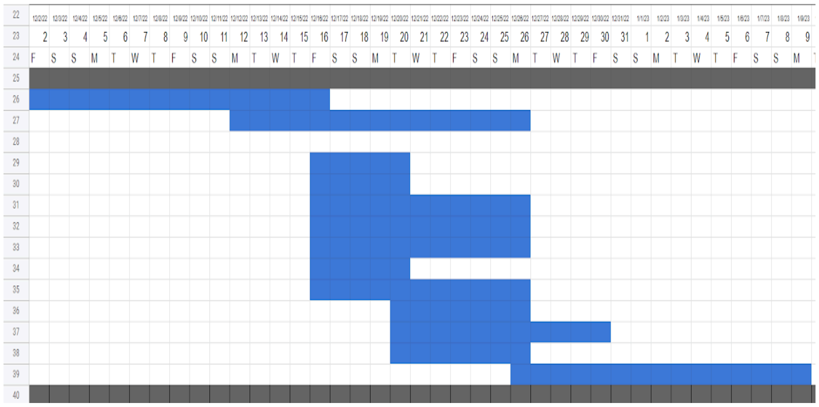


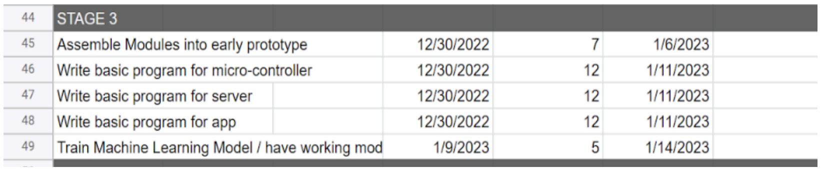
6	STAGE 1			
7	Choose Components	11/28/2022	4	12/2/2022
8	Choose Circuit Software	11/28/2022	4	12/2/2022
9	Determine modules/libraries for app	11/28/2022	7	12/5/2022
10	Choose Machine Learning Model	11/28/2022	4	12/2/2022
11	Choose 3d CAD software	11/28/2022	4	12/2/2022
12	List basic classes/functions/flowchart for Server Pr	11/28/2022	7	12/5/2022
13	Determine modules/libraries for microcontroller	11/28/2022	7	12/5/2022
14	Determine modules/libraries for ML Model	12/2/2022	7	12/9/2022
15	Build Circuit Schematic	12/5/2022	4	12/9/2022
16	List basic classes/functions/flowchart for app	12/5/2022	7	12/12/2022
17	Simulate and Test Circuit Schematic	12/9/2022	7	12/16/2022
18	List basic classes/functions/flowchart for mirco con	12/9/2022	7	12/16/2022
19	List basic classes/functions/flowchart for ML Model	12/9/2022	7	12/16/2022
20	Determine modules/libraries for Server Program	12/9/2022	7	12/16/2022



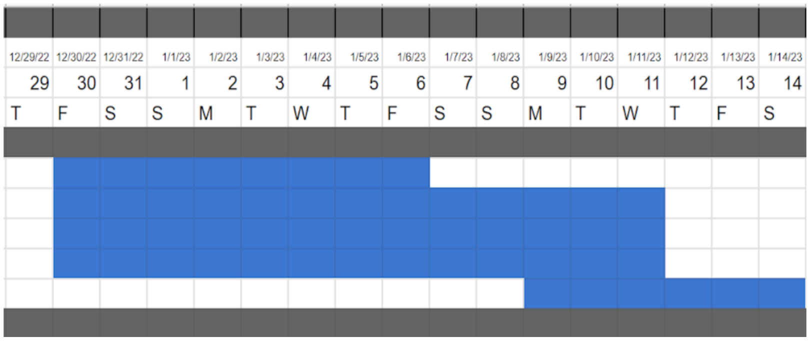


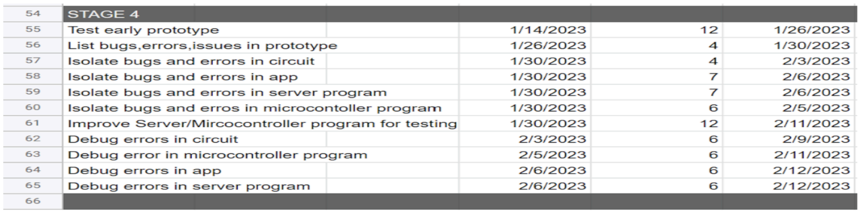
STAGE 2			
Order/Recieve parts	12/2/2022	14	12/16/2022
Write classes/functions for app	12/12/2022	14	12/26/2022
Write classes/functions for microcontroller	12/12/2002	10	12/22/2002
Assemble Power module	12/16/2022	4	12/20/2022
Assemble Scanner Module	12/16/2022	4	12/20/2022
Find/write test software for digital camera	12/16/2022	10	12/26/2022
Find/write test software for transmitter/reciever	12/16/2022	10	12/26/2022
Find/write test software for servo motor	12/16	10	12/26
Assemble Transmitter Module	12/16/2022	4	12/20/2022
Write classes/functions for ML Model	12/16/2022	10	12/26/2022
Test Power Module	12/20/2022	6	12/26/2022
Test Transmitter Module	12/20/2022	10	12/30/2022
Test Scanner Module	12/20/2022	6	12/26/2022
Find Machine Learning Model Test/Train data	12/26/2022	14	1/9/2023



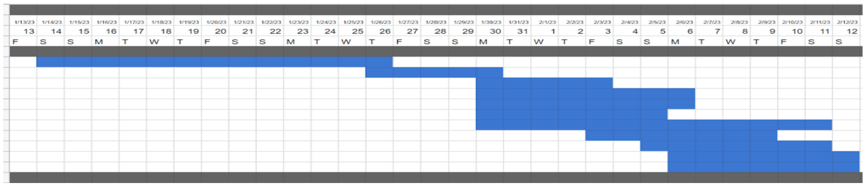


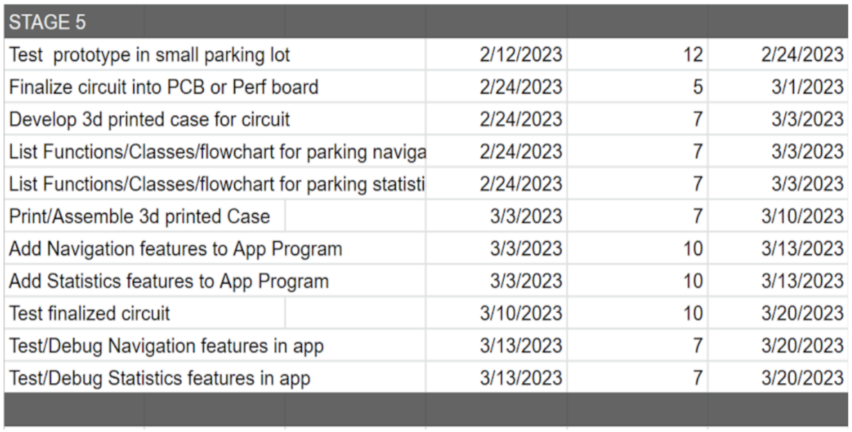
STAGE 3			
Assemble Modules into early prototype	12/30/2022	7	1/6/2023
Write basic program for micro-controller	12/30/2022	12	1/11/2023
Write basic program for server	12/30/2022	12	1/11/2023
Write basic program for app	12/30/2022	12	1/11/2023
Train Machine Learning Model / have working mod	1/9/2023	5	1/14/2023



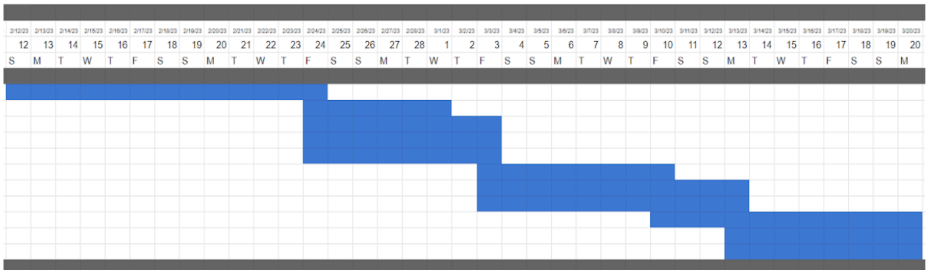


54	STAGE 4			
55	Test early prototype	1/14/2023	12	1/26/2023
56	List bugs,errors,issues in prototype	1/26/2023	4	1/30/2023
57	Isolate bugs and errors in circuit	1/30/2023	4	2/3/2023
58	Isolate bugs and errors in app	1/30/2023	7	2/6/2023
59	Isolate bugs and errors in server program	1/30/2023	7	2/6/2023
60	Isolate bugs and erros in microcontoller program	1/30/2023	6	2/5/2023
61	Improve Server/Mircocontroller program for testing	1/30/2023	12	2/11/2023
62	Debug errors in circuit	2/3/2023	6	2/9/2023
63	Debug error in microcontroller program	2/5/2023	6	2/11/2023
64	Debug errors in app	2/6/2023	6	2/12/2023
65	Debug errors in server program	2/6/2023	6	2/12/2023
66				





STAGE 5			
Test prototype in small parking lot	2/12/2023	12	2/24/2023
Finalize circuit into PCB or Perf board	2/24/2023	5	3/1/2023
Develop 3d printed case for circuit	2/24/2023	7	3/3/2023
List Functions/Classes/flowchart for parking naviga	2/24/2023	7	3/3/2023
List Functions/Classes/flowchart for parking statisti	2/24/2023	7	3/3/2023
Print/Assemble 3d printed Case	3/3/2023	7	3/10/2023
Add Navigation features to App Program	3/3/2023	10	3/13/2023
Add Statistics features to App Program	3/3/2023	10	3/13/2023
Test finalized circuit	3/10/2023	10	3/20/2023
Test/Debug Navigation features in app	3/13/2023	7	3/20/2023
Test/Debug Statistics features in app	3/13/2023	7	3/20/2023



3. Specific Requirements and Specifications

This section of the document lists specific requirements and specifications for OYSMS DigiPark. Requirements and specifications are divided into the following sections:

1. User requirements and specifications. These are requirements and specifications written from the point of view of end users, usually expressed in narrative form.
2. System requirements and specifications. These are detailed requirements and specifications describing the functions the system must be capable of doing.
3. Interfac specifications. These are requirements and specifications about the user interface, which may be expressed as a list, as a narrative, or as images of screen mock-ups.

3.1 User Requirements and Specifications

3.1.1 User interface
The user interface will be available from a mobile app that the user will be able to use on their cellular device. They will be able to look at the parking spaces available and the parking lots that are available and the ones that aren't available.

3.1.2 Ergonomics
This product will be in the air above the reach of anyone, so that it will be able to scan the designated area. This will make sure it is not in anyone's way.

3.1.3 Training or skills required
Technical understanding and experience of operating a smart device is required for the app. To be able to use the app you just must know how to use your phone. Being able to navigate through the app without no trouble. You also need to be able to drive and/or have the navigation skills to get to the designated spot.

3.2 System Requirements and Specifications

3.2.1 Physical Characteristics

− Raspberry pi 4		
	o Weight:	46 g
	o Dimensions:	85.6mm x 56.5mm
− Transmitter		
	o Weight:	15.4 g
	o Dimensions:	0.63x0.63x0.07 in
− Camera		
	o Weight:	34 x 24 mm
	o Dimensions:	20 g

3.2.2 Material Requirements

- 3-D Printed case

3.2.3 Electrical Requirements

- Raspberry pi 4:
  - Power: 5V DC, 2.5A
  - Operating System: Python
  - Operating Temperature: 0-50 degrees C

3.2.4 Abilities

- Take an image and processing that image

3.2.5 Limitations

- Not enough range to cover every parking lot simultaneously
- Wi-Fi signal not strong enough

3.2.6 Equipment or materials required to use the product

- Smart Phone
- Mobile App installed

3.2.7 Equipment interface requirements

- SD memory storage reading capabilities

3.2.8 Handling and storage requirements

- Needs to be up high in the air

3.2.9 Cleaning and Sterilization

- Cleaning the camera is going to be essential for us to get a clear picture of the parking lot

3.2.10 Product maintenance and serviceability

- Cleaning the camera to get a clearer picture
- Fixing the bugs of the app as we go along
- Wi-Fi needs to be available

3.2.11 Operating parameters

- OV5642
    - Power Supply: 3.3V - 5V
    - Pixel Size: 1.4µm x 1.4µm
    - Temperature: -10°C～+55°C
- Raspberry pi 4
    - Power Supply: 5V DC, 2.5A
    - Temperature: 0-50 degrees C
- RFM95
    - Power Supply: 3.3V
    - Temperature: -55°C～+115°C

3.2.12 Repeatability and reproducibility

- The project is going to be able to refresh and be able to repeat each process every 15 seconds. If the camera is going to be clean, then there should be no problems.

3.2.13 Reliability

- Depending on the weather conditions and other conditions, this product will be reliable on any day. Up to the user to refresh the app to see availability.

3.2.14 Mechanical safety features

- 3-D printed case

3.2.15 Electrical safety features

- Raspberry Pi 4
- Wires
- Transmitter
- Camera
- PCB

3.3 Interface Requirements and Specifications

Access and translation between the digital camera and transmitter should be monitored by our microcontroller. For the sake of consistent image upload to a database server which will translate said information into a readable form for our users. From this, our users will interact with the given parking information via an application that will take user input and guide them to their desired location. The device will be detached from the elevated location for maintenance if physical repairs are necessary and updates to the server can be done at a separate location.

3.4 Environmental Conditions
3.4.1 Temperature:
3.4.1.1 Operating: -20°C to 50°C
3.4.1.2 Storage: -25°C to 60°C
3.4.2 Humidity
3.4.2.1 Operating: 5-90% non-condensing
3.4.2.2 Storage:
- 5-90% non-condensing

3.4.3 Shipping, transportation vibration
- NA

3.4.4 Pressure and Altitude
- Maximum External Pressure: 1 atm
- Altitude: 5-10 meters

3.4.5 Electromagnetic Interference
- TBD

3.4.6 Electrostatic Discharge
- N/A

3.4.7 Impact Resistance
- N/A

3.5 Manufacturing

3.5.1 Cost
- Electronic Components:
- RFM95/96/97/98: $20-$30
- Raspberry Pi 4: $20-$40
- OV5642 Digital Camera $50-$100
- Other Materials:
- Weather Resistant Filament $20-$25per kg

3.5.2 Environmental requirements for production
- Source and Containment

3.5.3 Raw materials and suppliers
- Weather Resistant Filament for 3D Printing

3.5.4 Test methods, standards
- Weather Durability tests
- Software/Image tests
- Compatibility tests

3.6 Packaging

3.6.1 Packaging configurations
- Appropriately Sized Pelican Case
3.6.2 Packaging materials
- Case full of packing foam
3.6.3 Special shipment requirements
- Special protection for camera lens

3.7 Labeling

3.7.1 Detail intended use, warning, directions for use, cleaning, expiration date
- Intention – Parking Lot Monitoring and Analysis
- Warning – Fragile Components
- Directions – For Setup
3.7.2 Identify the target audience for labeling
- Client installations and security
3.7.3 Language requirements
- English

3.8 Regulatory
3.8.1 Clinical trials
- N/A
3.8.2 Submission type
- N/A
3.8.3 CE mark
- N/A
3.8.4 US and international standards
- RF Radio Frequency used for Image Transmission
- Pspice Simulation Software
- Graphical User Interface Design Standards
- Federal Communications Commission Standards for Communication Equipment
3.8.5 Patent issues
- None
3.8.6 Existing technology to avoid
- None

4. Appendices

N/A

5. Glossary

OYSMS: Team name

UTSA: University of Texas at San Antonio
FCC: Federal Communications Commission
RF: Radio Frequency
PC: Personal Computer
GUI: Graphical User Interface
GPS: Global Positioning System
Pspice: A design and simulation environment that helps evaluate functionality of analog circuits.
PCB: Printed Circuit Board

6. References

- University Parking and Traffic Rules & Regulations
- OmniVision OV5642 Datasheet
- Raspberry Pi 4 Model B Datasheet
- HopeRF RFM95/96/97/98(W) Datasheet

7. Index

N/A