Software Requirements Specification for Park'd: Your Go-To Parking Guide

Team #29, caPstOneGroup
Albert Zhou
Almen Ng
David Yao
Gary Gong
Jonathan Yapeter
Kabishan Suvendran

April 5, 2023

Contents

List of Tables

List of Figures

Table 1: Revision History

Date	Version	Notes
Oct 3, 2022	Albert, Almen, David, Gary, Jonathan, Kabishan	Revision 0
Nov 2, 2022	Almen, David	Adding Security Requirements outlined in the Hazard Analysis
Apr 1, 2022	Albert, Almen, David, Gary, Jonathan, Kabishan	Revision 1

Modifications to the Volere[1] Template

- Added the section Traceability Matrix for Functional and Non-Functional Requirements
- Added Reflection to the Appendix
- Added the section Priority Requirements

1 Project Drivers

This section discusses the purpose behind the project, Park'd, and identifies the stakeholders of project.

1.1 The Purpose of the Project

The following provides insight on the problem to be solved and the goals of the project.

1.1.1 The User Business or Background of the Project Effort

The project, Park'd, aims to provide a solution to a bothersome daily chore, finding parking spaces, by prompting drivers of open parking spaces and navigating the drivers to them. This project rose from the difficulty to find parking spaces in **McMaster** University as there are multiple parking lots scattered within and outside of the main campus and there is no way of knowing which lots are full upon arrival. Users of Park'd will be able to view all the available parking spaces at a parking lot, select a parking space, and be directed to the parking space in real time based on the shortest path.

1.1.2 Goals of the Project

The main goal for Park'd is to reduce the amount of time drivers takes to find a parking space. This becomes advantageous as drivers would be able to spend their time on more meaningful things and lessen the frustration in doing such a tedious task.

1.2 The Stakeholders

For this project, there are a variety of different stakeholders.

1.2.1 The Client

The client of this project is the instructor of SFWRENG 4G06A/B, Dr. Spencer Smith, and the teaching assistant (TA) of caPstOneGroup, Christopher William Schankula. As the clients, they will provide instructions on what deliverables need to be completed, offer assistance wherever possible, and evaluate the degree to which the project meets the requirements outlined in the **SRS**.

1.2.2 The Customers

The customers of this project are the individuals who are interested in looking for an open parking space in Ontario. The project targets mainly licensed Ontario drivers of all ages, however, it is available for the general public to access and use.

1.2.3 Other Stakeholders

The developers of Park'd, caPstOneGroup, are considered stakeholders of the project as their knowledge and skills necessary for the development of the project and they are interested in the success of the project. In addition, parking lot owners and managers are also stakeholders as they need to give permission to setup cameras or sensors on their property and would want customers to spend less time in parking lots when they can be shopping, and reduce the amount of collisions, traffic, and arguments that may occur.

1.2.4 The Hands-On Users of the Product

The hands-on users of the product can be defined into two groups, Ontario licensed drivers and the parking lot owners/managers. Firstly, Ontario licensed drivers would be the main users of the application as they are the ones actively using the application to find empty parking spaces and being navigated to them. The drivers must be capable of navigating to the website, comprehend directions written and spoken in English, and know the basic operations of a computer and mobile device, like selecting and dragging. As well, they must recognize Ontario road signs and symbols and realize that the application is to be considered as a guide and the drivers must still abide to all laws of the road, including being hands free on the road and staying within the speed limits. The next group of people are parking lot owners and managers. They would be the ones responsible for installing cameras in their parking lots and maintain the parking layout. Parking lot owners/managers must be able to navigate to the website, understand English instructions, and know the basic operations of a computer including but not limited to selecting and dragging.

2 Project Constraints

Outlined in this section are the restrictions on the project.

2.1 Mandated Constraints

The following are the mandated constraints of Park'd.

2.1.1 Solution Design Constraints

Description:
The system shall provide real-time parking lot information to all users fairly with no considerable lag
Rationale:
The system shall provide real-time parking lot information so that the user can get the most updated information and decide where to park their vehicles. Situations like a spot is being taken by other user but not yet reflected in our software is not allowed.
Fit Criterion: The updated information on parking lot should be reflected in the GUI and approved by testers in the parking lot.

Description:
The system shall operates on a Windows and Linux based machine, the web page should display and function with browsers with V8 engine.
Rationale:
These two OS are the most common OS in the market
Fit Criterion: The system should run on both system with no errors

2.1.2 Implementation Environment of the Current System

- ~~Back-end service will be implemented using Python 3.7 with Flask framework.~~
- ~~Machine learning model will be trained and tested using Python 3.7 with the Tensorflow and OpenCV libaries.~~
- ~~Front-end web page will be implemented using HTML,CSS and JavaScript~~.
- ~~The database will be hosted on Firebase.~~ the database and SQLAlchemy ORM will be used for accessing data for our application.
- ~~The ML model will be deployed on our own server.~~ cloud service.

- The system should be accessible from a browser on a desktop or mobile device.
- The system should be used by individuals at their computer or in a vehicle.

2.1.3 Partner or Collaborative Applications

- Parking lot surveillance system.
- Camera related hardware.

2.1.4 Off-the-Shelf Software

- N/A

2.1.5 Anticipated Workplace Environment

- Parking lot
- On the way where people commute

2.1.6 Schedule Constraints

- Proof of concept demonstration: November 20
- Design document revision 0: January 18
- V&V report revision 0: March 8

2.1.7 Budget Constraints

- N/A

2.1.8 Enterprise Constraints

- N/A

2.2 Naming Conventions and Terminology

Table 2: Table of Naming Conventions and Terminology

Term	Definition
SRS	A document that describes what the software will do and it's requirements
GPS	Global Positioning System is a system that finds your location using satellites.
RNN	Residual neural network is a artificial neural network best suited for image recognition task
Driver	A person using the application that has a drivers' license
Admin	A parking lot administrator using the application to manage parking lots
Python	A high level programming language that will be used to train and test our Machine Learning models and our backend
ML	Machine Learning
McMaster	A university located in Hamilton, Ontario
BE	Business event, a specific activity that must be supported
FR	Functional requirement, a specification of behaviours the system must have

2.3 Relevant Facts and Assumptions

In this section is important background information that was taken into account for the requirements specification.

2.3.1 Facts

- Parking spaces are denoted by white or yellow lines.
- Parking space dimension requirements vary by municipality. In Hamilton, spaces must measure at least 2.6 by 5.5 metres, as laid out in their zoning by-laws.
- Video surveillance footage should only be used for the purpose that surveillance is being undertaken, or for purposes that are permitted by law.
- Ontario's distracted driving laws can be found here. They do not permit the operation of a mobile device unless pulled over out of traffic. If viewing the device while driving, the device must be securely mounted to the dashboard.

2.3.2 Assumptions

- Drivers have an elementary proficiency in English
- Drivers have their licenses and know road terminology
- Drivers abide to all road rules and traffic laws
- The cameras will be at a high enough location to get a good view over the parking lot.
- All reserved parking are denoted by the same signs and symbols.
- Admins have access to a mouse for annotating their parking lots.

3 Functional Requirements

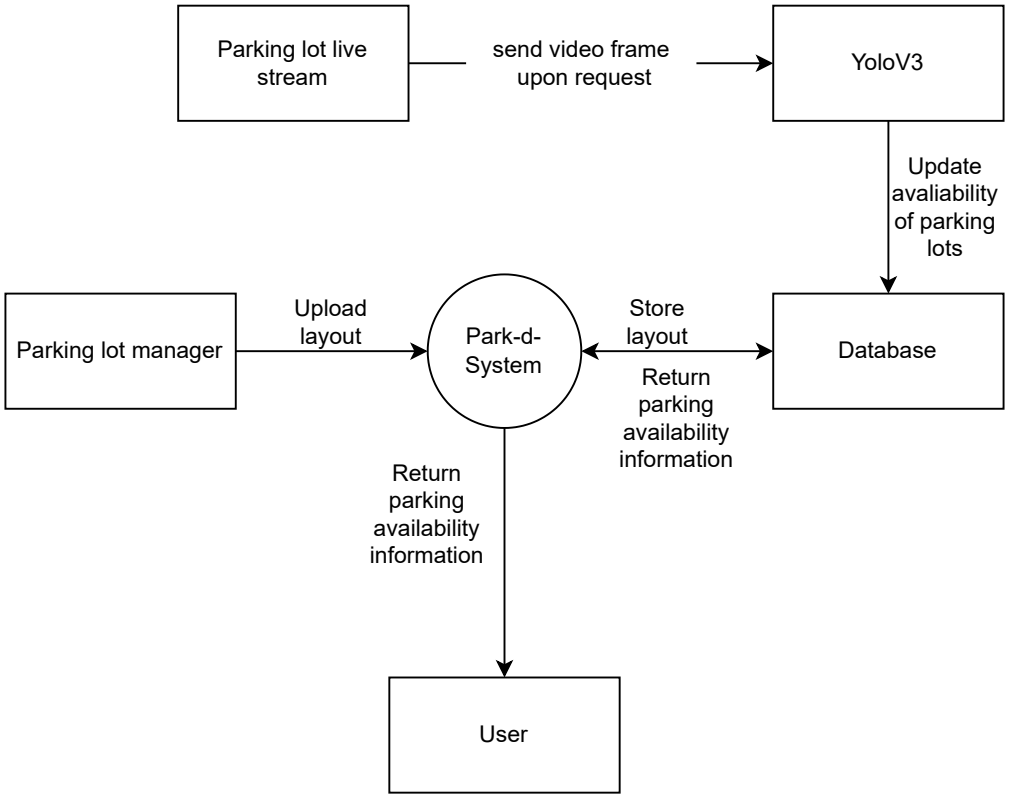
In this section are the functional requirements for Park'd, and related information including the context diagram, work partitioning, and use cases.

3.1 The Scope of the Work and the Product

The scope encompasses the range of functionality and usage scenarios the product is designed to cover.

3.1.1 The Context of the Work

Figure 1: Context diagram



3.1.2 Work Partitioning

Table 3: Work Partitioning Events

Event Number	Event Name	Input	Output
1	System start	User data	Parking lot map
2	Find an open spot	Parking lot data	Available spot list
3	Filter spots	Filter specification	Available spot list
4	Valid spot selection	Spot selection	Spot traversal directions
5	Invalid spot selection	Spot selection	Invalid selection
6	Spot traversal	None	Available spot list
7	Traversal complete	Spot reached	System feedback
8	Settings change	Settings list	None
9	Traversal termination	Termination selection	Parking lot map
10	Lot status	Parking lot data	Parking lot stats
11	Layout edit	Parking lot Layout	None

Table 4: Work Partitioning Summaries

Event Number	Summary
1	Starting the system. Location data is needed to determine which parking lot data to use and where the user is in the parking lot.
2	Use cameras in the parking lot and an AI to recognize open/special spots. List these spots to the user.
3	Use special spots such as accessible and reserved types.
4	Choose a spot in the list of available spots and get directions to it.
5	Choose a spot not on the list of available spots or was filtered out. Inform the user to make an appropriate selection.
6	The user follows the directions to the spot. If the status of the spot changes, the user will be prompted to change spots.
7	The user reaches the spot. The system data is updated and the user is asked for feedback.
8	Adjust system settings like sound, and vehicle details
9	Quit traversing to the current spot and return to the map.
10	View/create/modify parking lots.
11	Collect statistical data about the parking lot from cameras over time. The owner can use this data to make business decisions.
12	Allows the owner to add/remove/modify spots in the layout.

3.1.3 Individual Product Use Cases

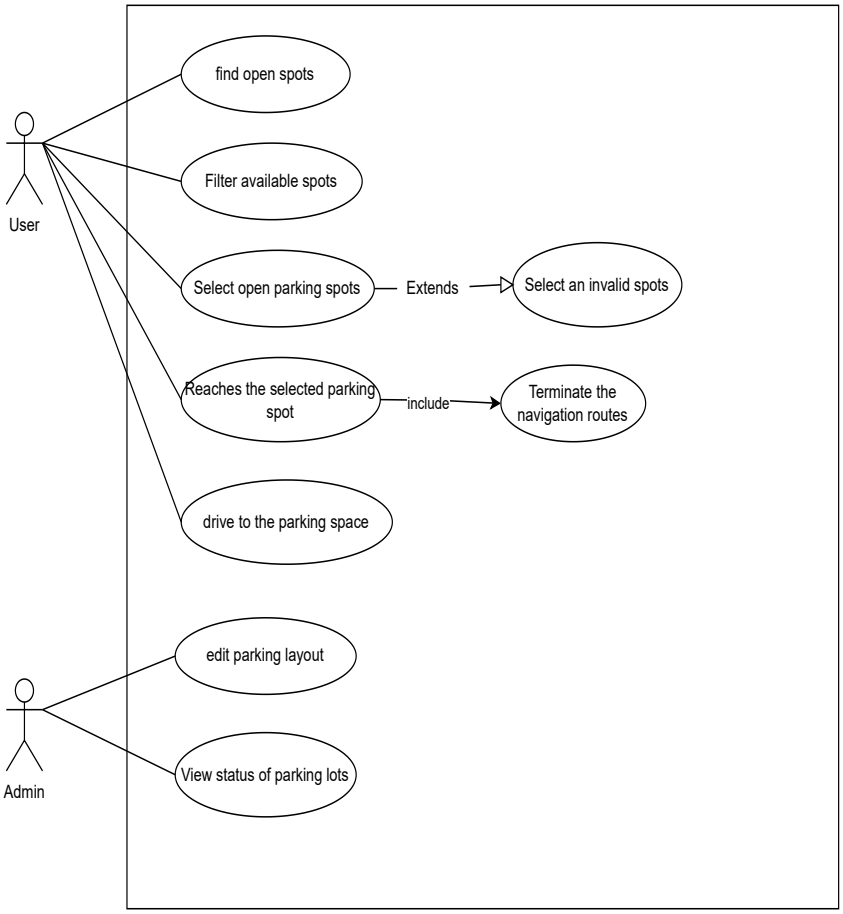


Figure 2: Use case diagram that displays the main functionalities of the application.

3.2 Functional Requirements

BE1. The driver wishes to start the system.

FR1. The system must prompt the driver to turn on the driver's location data.
Rationale: Location data is required to navigate the driver from their current position to their desired parking space in the parking lot.

FR2. The system must present the driver with a map of their surroundings, if they have allowed location access.
Rationale: Upon start-up, the system presenting the map of the driver's surroundings will help the driver orient themselves in their current location.

FR3. The system shall disable the navigation component if the driver doesn't provide permission.

Rationale: Not all drivers wish to provide permission or use the navigation component.

BE2. The driver wishes to find an open parking spot.

FR4. The system shall use parking lot camera footage and sensors to survey the parking lot.
Rationale: To determine available spaces in the parking lot, camera footage of the parking lot and sensor information need to be collected for the parking spaces.

FR5. The system shall allow admins to mark special parking spots.
Formal Specification:

$$\forall SP | SP \in P \bullet Detect(SP) = true$$

Where SP: special spot, P: parking spot
Rationale: The system needs to know about these spots so that they are only selected by those who need them.those who need them.

FR6. The system shall allow admins to mark the physical location of parking spots.
of parking spots. **Rationale:** Drivers will be directed to their spot, so their destination has to be determined by the camera location.

FR7. The system must display all available parking lots and allow the driver to provide the parking lot that they wish to navigate to.
Formal Specification:
Available spot =

$$\forall S | S \in A \bullet x < x1 \wedge x < x3 \wedge x > x2 \wedge x > x4 \wedge y < y1 \wedge y < y3 \wedge y > y2 \wedge y > y4 = true$$

Where T = (x,y) be a pair of coordinates indicates the mid point of the detected vehicle.
S = [(x1,y1), (x2,y2), (x3,y3), (x4,y4)] be an array of 4 tuples with each tuple represent the coordinates of a parking space annotation from top left corner to bottom right corner in counter-clock wise direction
And A be a set of S

FR8. The system must display the available parking spaces of the specified destination parking lot.

BE3. The driver wishes to filter the available parking spots.

FR9. The system shall allow the driver to select between using available normal, accessibility and reserved parking spaces, with confirmation of authorization for the latter two.
Rationale: The driver should be able to view and select between all possible types of parking spaces if they are authorized. For example, a driver with a wheelchair would need to see accessible parking spaces, whereas, an able-bodied driver would only need to use normal parking spaces to avoid parking in an accessible or reserved parking space.

FR10. The system by default must only allow selections for normal, non-reserved parking spaces, if authorization is not confirmed.
Rationale: Most drivers will not be authorized to use accessible parking spaces or reserved parking spaces, such as those for public transit vehicles. Without any other input, the system should allow only the spaces that the most drivers can use.

BE4. The driver wishes to select an open parking spot.

FR11. The system shall allow the driver to select an available parking space.
Rationale: The driver must be allowed to select an available parking space to receive instructions to park at the desired parking space.

FR12. The system shall display driving directions to the driver upon selecting an available parking space.
Rationale: The driver should be informed on how to reach the parking space they selected from their current position.

FR13. The system shall provide a default recommendation if the driver does not wish or is not able to select a specific parking space.
Rationale: Providing the input to select a specific parking space may be difficult on small displays because of the relative size difference between a parking lot and a space.
Will be implemented in a future revision.

FR14. The system shall display navigation directions to the default recommended parking space if the driver makes no space selection within $DEFAULT\_DELAY$ seconds.
Will be implemented in a future revision.

BE5. The driver wishes to select an invalid parking spot.

FR15. The system shall not allow the driver to select an invalid spot such as spots that are already occupied.
Rationale: The driver should not be able to select an invalid spot.

FR16. The system shall notify the driver if they attempt to select a reserved or disabled parking space, if they have confirmed authorization for those spaces.
Rationale: The driver should be informed that they cannot park in a spot that requires other authorization.

BE6. The driver wishes to drive to the parking space.

FR17. The system shall provide real time directions to the driver to the selected parking space.

FR18. The system shall recommend a new available parking space to the driver upon the initial selected parking space being taken.
Will be implemented in a future revision.

FR19. The system shall provide the option to safely select a new parking space upon the initial selected parking space being taken.
Will be implemented in a future revision.

FR20. The system shall terminate the route and inform the driver if the chosen space has been taken before they arrive.
Rationale: The driver must choose a new space when theirs is taken.

FR21. The system shall terminate the route if the driver doesn't arrive within a time estimate plus $TIMEOUT\_TOLERANCE$ seconds.
Formal Specification:

$$\texttt{T} \geq \texttt{TimeOutTolerance} \Rightarrow Ter$$

$$\texttt{T} \leq \texttt{TimeOutTolerance} \Rightarrow Wor$$

Where TimeOutTolerance: the maximum time that system do not terminate when user's hasn't arrive to the Destination
Where Ter is the termination state
Where Wor is the working state
Where T is the time has elapsed after user start navigation
Rationale: The driver may take too long to reach the space or not go at all.

FR22. The system shall track the driver's location within $DISTANCE\_TOLERANCE$ meters.
Rationale: Location data isn't perfect so there must be a tolerance.

FR23. The system shall provide a time estimate to the selected space within $DRIVING\_ESTIMATE\_$ seconds.
Rationale: Drivers should take driving time in to consideration when finding a spot.

BE7. The driver reaches the selected parking spot.

FR24. The system shall prompt the driver that they have reached the parking spot.

FR25. The system shall change the status of the parking space from available to unavailable when the driver enters the parking space.
Rationale: Using the driver's location within the parking lot, the status of the parking space should be updated, such that future drivers do not pick the same parking space.

FR26. The system shall non-intrusively prompt the driver for feedback regarding their experience after parking.
Rationale: Driver feedback will be important to improve our system's experience, as well as to alert us of any issues they have encountered.
Will be implemented in a future revision.

BE8. The driver wishes to change the settings of the application.

FR27. The system shall provide a means of viewing and modifying the settings.
Will be implemented in a future revision.

FR28. The system's settings menu shall include volume and sound adjustment, unit identification, vehicle details, and notification preferences.
Will be implemented in a future revision.

BE9. The driver wishes to terminate the current navigation route.

FR29. The system shall provide a means to terminate the current navigation.

FR30. The system shall present the driver in their current location on the map upon termination of the route.

BE10. The parking lot owner/manager wishes to view the current status of their parking lot.

FR31. The system must provide a separate administrative view for parking lot owners/managers.

FR32. The system must provide a means for individual parking lot owners/managers to enter their own administrative console.

FR33. The system's administrative console shall provide a means for the parking lot owner/manager to view the availability of all parking spaces in their lot.

BE11. The parking lot owner/manager or driver wishes to view the analytics of their desired parking lot.

FR34. The system shall display live analytics on how many spaces are available/unavailable and categorize them in terms of their type (standard, accessible, reserved).

FR35. The system shall display twenty-four hours of averaged historical parking lot occupancy information.

BE12. The parking lot owner/manager wishes to edit their parking lot layout.

FR36. The system's administrative console shall include a means for individual parking lot owners/managers to edit their parking lot layout.

FR37. The system's administrative console shall include editing features such as redefining parking spaces and marking spaces as reserved.

4 Non-functional Requirements

This section specifies the non-functional requirements for Park'd.

4.1 Look and Feel Requirements

4.1.1 Appearance Requirements

LF1. The user interface must only consist of information immediately relevant to the driver.

Fit Criterion: For every element of the interface, all **Park'd** developers agree on its necessity.

4.1.2 Style Requirements

LF2. All user interface elements must conform to the park-d branding guidelines
Fit Criterion: All the user interface elements presented in our website should have consistent use of the organization's logo, font, and yellow black color scheme.

4.2 Usability and Humanity Requirements

4.2.1 Ease of Use Requirements

UH1. Operations from the working screen take no more than MAX_TAPS taps to complete.

UH2. For each function feature in the software, novice users are able to identify and complete each of them with no more than $MAX\_COMPLETE\_TIME$ minute

4.2.2 Personalization and Internationalization Requirements

N/A

4.2.3 Learning Requirements

UH3. The system's basic parking functionality should be useable with no prior instruction.
Fit Criterion: 80% of testers navigate to a designated parking spot without any prior instruction.

4.2.4 Understandability and Politeness Requirements

UH4. Any symbols must be understandable to the driver.
Fit Criterion: All symbols used adhere to the standard set forth by the Ontario Ministry of Transportation.

4.2.5 Accessibility Requirements

N/A

4.3 Performance Requirements

4.3.1 Speed and Latency Requirements

PE1. The system must maintain a minimum of $MIN\_FRAMERATE$ during normal operation.

4.3.2 Safety-Critical Requirements

PE2. The system must never direct driver user off of the road, for example, onto the sidewalk.
Fit Criterion: Computer constructed maps of the parking lot will be inspected by the developers to ensure no deviations from the marked surfaces.

PE3. The system must never direct the user to park in a reserved area such as a fire lane or across a driveway.
Fit Criterion: Computer constructed maps of the parking lot will be inspected by the developers to ensure only marked parking spots are recognized.

4.3.3 Precision or Accuracy Requirements

N/A

4.3.4 Reliability and Availability Requirements

PE4. The system shall be responsive, that is, it should respond to the user's command within $DEFAULT\_DELAY/4$ seconds.
Fit Criterion: We require for the maximum delay to be $DEFAULT\_DELAY/4$ because, based on user validation testing, users began to voice their frustration after $DEFAULT\_DELAY/4$ seconds.

4.3.5 Robustness or Fault-Tolerance Requirements

PE5. In the event of lost connection to the system back-end, the most recent recommended parking spot will remain on the display.

4.3.6 Capacity Requirements

PE6. The system back-end must support $CONCURRENT\_USERS$ using the system at the same time.

4.3.7 Scalability or Extensibility Requirements

PE7. The system shall provide administrators the ability to add any number of parking areas ($PARKING\_LOTS$ and above) without having to remove previous parking areas.

4.3.8 Longevity Requirements

N/A

4.4 Operational and Environmental Requirements

4.4.1 Expected Physical Environment

OE1. The system shall be operated by a driver in a vehicle.
Fit Criterion: The system is designed for a vehicle app or a device to travel with.

4.4.2 Requirements for Interfacing with Adjacent Systems

N/A

4.4.3 Productization Requirements

N/A

4.4.4 Release Requirements

OE2. The system shall have a new release at least once a month as GitHub commits.
Fit Criterion: Check GitHub commit log for update frequency.

4.5 Maintainability and Support Requirements

4.5.1 Maintenance Requirements

MA1. Critical issues will be addressed immediately with a target resolution time. Issues will be tracked and prioritized based on severity and impact on system functionality.
Fit Criterion: The software issues and bugs should be resolved in less than 24 hours after errors or bugs being located. A dedicated team member will be responsible for identifying, prioritizing, and addressing maintenance issues and these issues will be recorded with their severity in a internal document.

4.5.2 Supportability Requirements

MA2. The system shall provide drivers with instructions and contacts in the interface.
Fit Criterion: At least $SUPPORTABILITY\_SATISFACTION$ of surveyed drivers are satisfied with their support.

4.5.3 Adaptability Requirements

N/A

4.6 Security Requirements

4.6.1 Access Requirements

SR1. The system's parking lot data shall be accessible only to the team and to the property owner(s).
Fit Criterion: The data is password protected.

SR2. Only the parking lot owner(s) shall have the option to edit the parking space layout
Fit Criterion: Upon logging in as a standard user, there should not be a button to edit the layout of any parking area. For an administrator, there should only be a button to edit the layout of the parking lot that they own.

SR3. Only the parking space manager(s) of a parking lot are allowed to have access to the administrative console for their parking lot
Fit Criterion: The administrative console of a parking lot can only edit and view analytics of the parking lot

4.6.2 Integrity Requirements

SR4. The system shall prevent inaccurate data from being stored.
Fit Criterion: Stress test the system with accurate and inaccurate data and measure the data's accuracy.
Will be implemented in a future revision.

SR5. Unsaved parking layout information should be stored locally if the information cannot be uploaded to the server
Will be implemented in a future revision.

SR6. Unsaved parking layout information should attempt to upload to the server every $ATTEMPT\_UPLOAD\_TIME$ seconds
Will be implemented in a future revision.

SR7. Parking layouts will be automatically backed up daily
Will be implemented in a future revision.

SR8. No parking space should be stored in a different format in the database from other parking spaces

SR9. The system should only allow a parking spot to have $MAXIMUM\_SPECIAL\_PROPERTY$ special property
Fit Criterion: A parking space is either labeled as accessibility, electric vehicle, or reserved

SR10. Parking lot managers must be prompted when there is a failed attempt to add a parking spot to the database

SR11. Parking lot owners should be able to prompt the upload of their parking lot layout to the database and server

SR12. Correct paths should be stored to all parking spaces

SR13. Users are informed when an error occurs when the system is determining the navigation path

4.6.3 Privacy Requirements

SR14. The system shall ask for permission to use the driver's location data.
Fit Criterion: The system has a driver location agreement form.

4.6.4 Audit Requirements

N/A

4.6.5 Immunity Requirements

N/A

4.7 Cultural and Political Requirements

4.7.1 Cultural Requirements

N/A

4.7.2 Political Requirements

N/A

4.8 Legal Requirements

4.8.1 Compliance Requirements

LR1. The system shall follow laws relating to device usage while driving.
Fit Criterion: The laws relating to device usage while driving should be referenced and checked for compliance.

4.8.2 Standards Requirements

N/A

4.9 Health and Safety Requirements

HS1. The system shall not include flashing graphics or disturbing imagery.
Fit Criterion: Drivers surveyed should not find the system distracting or disturbing.

5 Priority Requirements

High priority requirements are identified here, as well as requirements that are unlikely to change.

5.1 Requirements for the Proof of Concept

November 7: The following requirements will need to be met at least one week before the proof of concept demonstrations begin, from November 14-25. The proof of concept plan can be found within the Development Plan. This is to allow time for debugging and polish. These requirements are fundamental to the basic function of the system.

- BE2FR4: The system must have access to useable parking area footage.
- BE2FR6: The system must be able to process footage to construct a valid parking space.
- BE4FR13: The system will be demonstrated with a single parking space.
- BE7FR24: The system must recognize and relay to the driver when they have arrived to the space.
- BE7FR25: The system must mark the space as full once a vehicle has entered it.
- LF1: The user interface should not need to be simplified over time.
- UH3: With only basic functionality, minimal instruction should be needed.
- UH4: We should not introduce any symbols that would be unfamiliar to the driver.
- OE1: The system will be developed for drivers.
- SR1: The video feed should never be accessible from the front-end, because it is processed by the back-end.

5.2 Requirements for Full Revisions

January 31: All other requirements set out in this document are to be met two weeks before the Revision 0 demonstration, which will occur between February 6-17. The additional margin is because extensive debugging and testing will likely be needed at this stage. At this time, the team will also begin to consider the addition of further requirements to meet our stretch goals.

March 13: Revision 1 will likely include the addition of some stretch goals, for which requirements will be added to this document. Those requirements will need to be fulfilled at least one week prior to the Revision 1 demonstrations, on March 20-31.

5.3 Requirements Unlikely to Change

~~Non-functional requirements relating to the user experience are unlikely to change because they do not rely on any specific software or hardware limitations. Instead, they specify how we will make some design decisions on the front-end to benefit drivers and administrators.~~

~~Requirements relating to the law such as parking in disabled parking spaces are unlikely to change because these laws are well established and unlikely to change. It would be irresponsible to release an application that recommends breaking the law and risking a ticket or a tow.~~

All requirements except for FR20, FR26, AND MA1 are unlikely to change.
FR20: The system might be changed to automatically redirect the user to another spot if it is close enough to the original spot selected.
FR26: Rather than prompting the user for feedback, the system may include another method for users voluntarily offer feedback as it may become annoying for the users to get a prompt after every session.
MA1: As the system scales and expands in the future, the maintenance procedure may have to change to support more features and users.

6 Traceability Between Functional and Non-Functional Requirements

Traceability between functional and non-functional requirements is shown by a series of traceability matrices mapping functional requirements to non-functional requirements.

6.1 Traceability Between Test Cases and Requirements

Table 5: Traceability Matrix for Functional and Non-Functional Requirements - Part 1

		Non-Functional Requirements													
		LF1	LF2	UH1	UH2	UH3	UH4	PE1	PE2	PE3	PE4	PE5	PE6	OE1	OE2
Functional Requirements	FR1	X	X	X	X		X	X	X	X		X		X	
	FR2	X	X	X	X		X	X	X	X				X	
	FR3								X	X					
	FR4										X				
	FR5								X	X	X				
	FR6	X	X	X	X	X	X	X						X	
	FR7	X	X		X	X	X	X	X	X	X				
	FR8	X	X	X	X	X	X	X						X	
	FR9					X		X							
	FR10					X		X	X	X	X			X	
	FR11	X	X		X	X	X	X	X	X	X			X	
	FR12							X	X	X	X			X	
	FR13					X		X	X	X	X			X	
	FR14	X	X		X		X	X	X	X				X	
	FR15	X	X		X		X	X	X	X				X	
	FR16					X		X	X	X	X	X		X	
	FR17					X		X	X	X				X	
	FR18	X	X		X	X	X	X	X	X	X			X	
	FR19	X	X		X	X	X	X	X	X				X	
	FR20							X							
	FR21	X	X		X	X	X	X						X	
	FR22	X	X		X		X	X						X	
	FR23	X	X		X		X	X						X	
	FR24					X		X						X	
	FR25	X	X		X	X	X	X						X	
	FR26							X							
	FR27														
	FR28							X							
	FR29														
	FR30							X							
	FR31							X							
	FR32		X	X	X			X						X	
	FR33			X	X			X	X	X					
	FR34		X					X							
	FR35			X	X				X			X		X	
	FR36							X	X	X				X	
	FR37			X	X			X						X	

Table 6: Traceability Matrix for Functional and Non-Functional Requirements - Part 2

		Non-Functional Requirements													
		MA1	MA2	SR1	SR2	SR3	SR4	SR5	SR6	SR7	SR8	SR9	SR10	SR11	SR12
Functional Requirements	FR1														
	FR2		X												
	FR3			X			X				X				X
	FR4	X		X			X				X	X			
	FR5		X	X											X
	FR6	X													
	FR7														
	FR8	X													
	FR9	X		X								X			
	FR10	X													
	FR11	X													X
	FR12	X													
	FR13	X													X
	FR14						X								
	FR15						X								
	FR16	X													X
	FR17		X												
	FR18		X												
	FR19	X													
	FR20						X								
	FR21														
	FR22			X											
	FR23			X			X								
	FR24			X											
	FR25														
	FR26			X		X	X								
	FR27			X		X	X								
	FR28					X	X								
	FR29					X									
	FR30			X	X	X	X	X	X	X			X	X	
	FR31				X	X	X	X	X	X			X	X	
	FR32		X					X				X			
	FR33	X			X								X		
	FR34					X						X		X	
	FR35			X									X		
	FR36		X		X		X				X				
	FR37	X			X								X		

Table 7: Traceability Matrix for Functional and Non-Functional Requirements - Part 3

		Non-Functional Requirements			
		SR13	SR14	LR1	HS1
Functional Requirements	FR1		X		
	FR2		X		X
	FR3				
	FR4				
	FR5				
	FR6		X	X	
	FR7				X
	FR8				
	FR9				
	FR10			X	
	FR11	X		X	X
	FR12			X	X
	FR13	X		X	X
	FR14				
	FR15				
	FR16	X	X		
	FR17				
	FR18				
	FR19				
	FR20				
	FR21			X	X
	FR22		X		
	FR23		X	X	X
	FR24			X	
	FR25		X	X	X
	FR26		X		X
	FR27			X	
	FR28		X		X
	FR29		X		
	FR30		X		
	FR31		X		
	FR32			X	
	FR33	X			
	FR34		X	X	
	FR35			X	X
	FR36	X	X		
	FR37			X	

7 Project Issues

Some threats or potential issues that may arise during development are listed here.

7.1 Open Issues

1. There are many laws regarding the use of smartphones when operating a vehicle. Distracted driving laws in Ontario can be found here. We must make sure that our application complies with any laws that may apply as well as allow drivers to interact with our application in a way that puts safety and the law first. In Ontario, this means the driver cannot interact with their mobile device by touch unless safely stopped out of traffic. When viewing their device, it must be securely mounted. This means our app must not prompt the driver for inputs while driving, such that usage cannot continue unless that input is provided.

2. We plan to use the Global Positioning System location of the driver's phone to track them around the parking lot. This will help us guide them to an available parking spot. However, the accuracy of **GPS** may not be high enough to consistently track the driver's location around a parking lot.

7.2 Off-the-Shelf Solutions

1. ParKam App
   ParKam is an application that is very similar to what we are trying to develop. They also use cameras to detect parking spots and navigate users to available spots. Although they have not entered the North American market, they seem to be doing pretty well in other countries such as Australia, Brazil, Dubai, the Netherlands, and Israel where they are based. We also could not find any videos of the app working so we could not judge how well their solution works.

2. Spot Sensor
   From personal experience, some indoor or underground parking lots in Toronto have sensors above each parking spot. It uses some sort of ultrasonic sensor to detect if a car is occupying the spot. If the spot is empty, a light on the sensor turns green and if the spot is occupied, the light turns red. This solution is pretty good as you can quite easily spot the green light and see which area the empty spot is. However, there are several issues with this solution. One main problem is that it requires installing hardware at every parking spot. Furthermore, the light does not tell you exactly how to get to the parking spot.

7.3 New Problems

These are problems that may be caused by Park'd itself.

7.3.1 Effects on the Current Environment

The Park'd app should relieve traffic around parking lots as less people would be driving around a parking lot looking for empty spots. This should allow more cars to quickly get to a parking spot and less cars roaming the parking lot. This would also make the parking lots safer for pedestrian since less traffic would reduce the chance for accidents.

7.3.2 Effects on the Installed Systems

N/A

7.3.3 Potential User Problems

A major potential user problem would be users getting distracted when using the app in their car. Users getting distracted when using our application could cause accidents which would be unacceptable. We have to make sure that safety is our number one priority when developing the app. Users should be able to use the app hands-free and without distractions when they are operating a vehicle. As per previously stated Ontario law, there should be no need for continuous input while the user is driving.

7.3.4 Limitations in the Anticipated Implementation Environment That May Inhibit the New Product

N/A

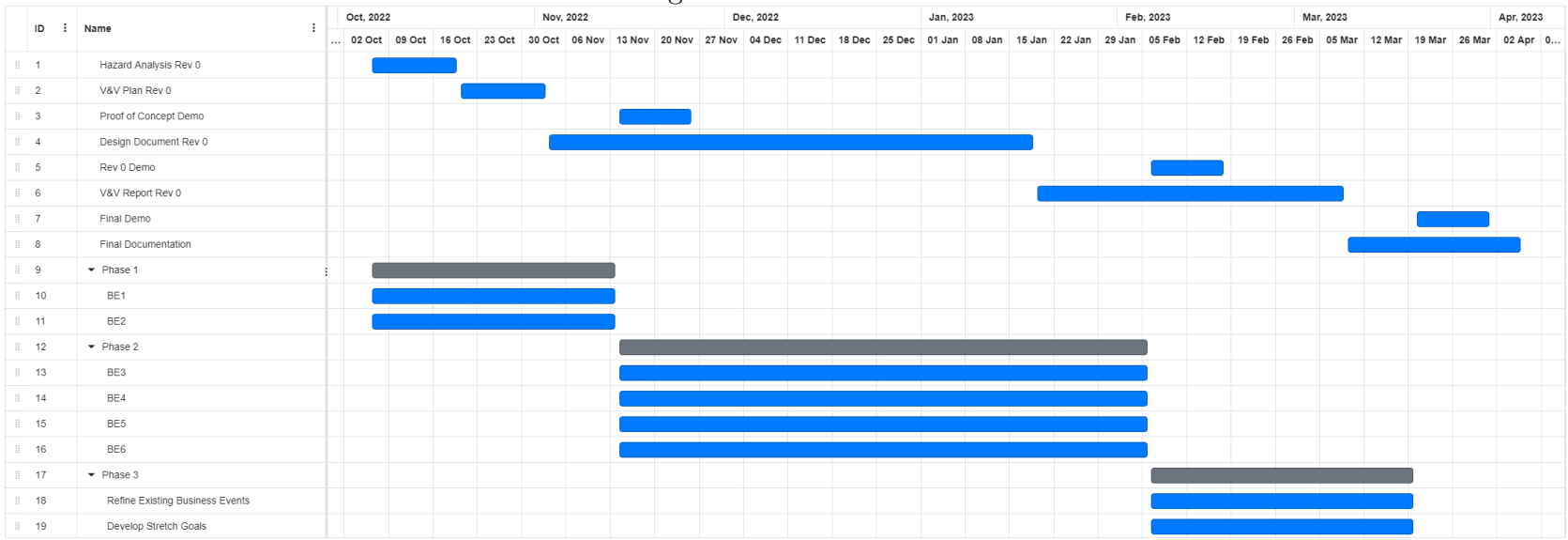
7.3.5 Follow-up Problems

N/A

7.4 Tasks

The different tasks and schedule to successfully develop our application is outlined in the Gantt chart provided.

Figure 3: Gantt Chart



7.5 Migration to the New Product

7.5.1 Requirements for Migration to the New Product

N/A

7.5.2 Data That Has to Be Modified or Translated for the New System

N/A

7.6 Risks

1. Inadequate time management is a key risk to the development of our app. While working on creating the app, our team is also busy working and studying for other courses throughout the year. It is important for the team to keep in mind that we have other responsibilities to take care of besides this project. We often set very big goals for ourselves and we need to keep each other in check to make sure we are being realistic with our goals and work. We only have two terms to complete this project, which may seem like a lot of time now but it will pass by quickly and we have to stay on top of our plans.

2. There are some technologies that are new for most or all of our team members. One of these technologies is machine vision and machine learning. The team will have to learn about these technologies as we develop our project. This could pose risks of obstructions and road blocks in our development which could take more time than usual to figure out as we are learning as we go.

7.7 Costs

1. Monetary Costs:
   Some of the technologies we plan to use, such as hosting services, have free versions or credit that are available for University students. Cameras would be useful to capture our own footage of scale models or real parking lots for training and testing purposes. This can be done for free with any smartphone or webcam any team member owns. We can also use a RaspberryPi and camera module to have a standalone system that can record footage and even send a live feed. A RaspberryPi and camera module would cost around $50 dollars.

2. Time Costs:
   Time costs are difficult to estimate as each requirement or feature will take different amounts of time to complete. Unforeseen issues may also affect the duration to develop certain features. A rough estimate based on our six business events would be 1.5 months, assuming an average of one week of development per business event to get our application to a state that can be considered a Minimum Viable Product. On top of this, we would still have to complete testing, documentation, and improving our existing features before moving on to stretch goals.

7.8 User Documentation and Training

This section specifies what users may need to do to prepare to use the application, and what resources will be made available.

7.8.1 User Documentation Requirements

The following is a list of documentation that will be available to the user:

1. User manual
   The User manual in the About page explains the user-facing features of the application and how to use it to find a parking spot. by the business events outlined in Section 3.2.

These documents will be updated and modified as the team sees fit throughout the development process.

7.8.2 Training Requirements

No specific training will be required for users to utilize our application. The contents in the user manual and installation manual should be sufficient to get users to start using our application.

7.9 Waiting Room

Refer to Section 3 of the Problem Statement document for Stretch Goals that the team plans to develop if time allows.

7.10 Ideas for Solutions

N/A

8 Appendix

Any additional information is located here.

8.1 Symbolic Parameters

The definition of the requirements will likely call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance.

$CONCURRENT\_USERS = 10$
$DEFAULT\_DELAY = 10$
$MIN\_FRAMERATE = 15$
$MAX\_TAPS = 2$
$MAX\_TIME_COMPLETE = 1$
$PARKING\_LOTS = 2$
$SUPPORTABILITY\_SATISFACTION = 80\%$
$ATTEMPT\_UPLOAD\_TIME = 30$
$MAXIMUM\_SPECIAL\_PROPERTY = 1$
$TIMEOUT\_TOLERANCE = 60$
$DISTANCE\_TOLERANCE = 5$
$DRIVING\_ESTIMATE\_TOLERANCE = 60$

8.2 Reflection

The SRS reflection details areas of improvement and learning for team members.

8.2.1 Presentation Skills

One of the skills that our team will collectively need to acquire is our presentation skills. Presenting in a professional setting is a nerve-wracking task, especially while being evaluated on our performance. For the team members that are pursuing the management program, this skill has been developed to a greater extent, but there is still room for improvement, since the audience for management projects vary vastly from the audience for the capstone project, as we will be presenting to teaching assistants, the professor, onlookers at the EXPO competition and even prospect students if chosen to present at May @ Mac.

Learning Approaches

1. Improvisational Skills: It is highly likely that something will go wrong or important content was missed during the presentation. At this moment, team members will have to go off the script to keep the show going and to cover all the material required.

2. Meditation Skills: When things do go wrong, remaining calm will allow you to avoid voicing incoherent and irrelevant ideas, and derailing the focus of other team members.

- Almen: I believe that the approach of mastering improvisational skills would be beneficial to me. I like keeping things very structured, and that includes what is being said during a presentation. It provides a sense of security for me knowing what is happening next. However, knowing the nature of in-person presentations, there are a lot of factors that can affect the overall structure including members forgetting their part, members are unable to make it to the presentation due to unforeseen circumstances, etc. On top of that, questions will be brought up that may not have been planned for and it is up to us to come up with an appropriate response to them. In the event that that happens, I will have to think on my toes which is normally something that I am unprepared for. This is why I chose working on improvisational skills as a way to work on my presentation skills.

- Albert: My history with class presentations have always been on the more theatrical side. Messing things up could be played for laughs and no one would be thinking lesser. But, with a formal presentation, it isn't an option. I should improve my skills through mediation, so that mess ups don't ruin the professional tone. Maybe it could be done so well that it's like it was all part of the plan.

- David: My priority would be improving my improvisational skills, because sometimes I have issues thinking on the spot about what I need to say next. Nerves is definitely an issue as well but my strategy of plowing through any slip-ups I make has worked well. The consequences of not knowing what to say or saying the wrong thing are more serious.

- Gary:I believe I can improve my presentation skills through mastering improvisational skills as it almost always happen in a presentation where you get interrupted by a question or some member forget their parts, and you have to improvise in the crunch time to make sure the presentation continues smoothly.

- Jonathan: I have always been a firm believer that the presentation is sometimes just as important as the product. Many people buy average products just from the presentation and how it is marketed. One common example are restaurants and cafes that serve average food but the plating is pretty or the ambiance is nice. In the context of our project, a solid and smooth presentation will give a better impression to our audience and in turn, they will have a better impression of the product itself. To better guarantee a solid presentation, strong improvisational skills are necessary to smooth out any bumps that will occur. Being stuck in a presentation because of some issue is the worst thing that can happen and having good improvisational skills can help alleviate that problem.

- Kabishan: I believe that I can improve my presentation skills by mastering meditation skills. In doing so, not only will I be an asset to the presentation, but I will also improve my ability to cope with stressful situations outside of the capstone project. I often find myself unable to think coherently or cohesively during stressful situations, so taking this approach will allow me to calm down and think more rationally.

8.2.2 Time Management

Another skill our team will need to collectively acquire is time management. With such a large scale project and course, there are many deliverables that need to be completed by a certain deadlines and a lot of work needed to be done. On top of the other assignments and midterms that we have with other courses, it is crucial to be able to manage our time wisely to ensure that we output something we are proud of and that is well thought out.

Learning Approaches

1. Start sooner rather than later. More often than not, the amount of work that needs to be done is often overlooked until we actuall y start working on it. In times where we may think that the task would be easy and start later would be the times where we would be rushing at the end to complete it the most. By starting early, we would know exactly what needs to be done, ask any questions that we have about topics we are unclear about, and have more time to complete the work for this project as well as others.

2. Determine our priorities and keeping track of them and their deadlines. With all the capstone deliverables and other course work that we have, it can be overwhelming to juggle between the different deadlines, unsure of which one to start or complete first. By defining our priorities, it can give us a direction as to what needs to be done and what goals need to be accomplished. Furthermore, using software like Google Calendar or Notion can help us stay on track with deadlines and also provide reminders on when upcoming due dates are.

- Almen: I believe the best approach would be to start sooner. With everyone's different course work and schedules, knowing exactly what we need to do will provide us ample time to accommodate for course work and unexpected work items.

- Albert: I've always been a start sooner and get small bits of work done and have time at the end to change all that work, but, with a group of 6, we may have to start before the previous deliverable is finished to fit in enough time for sync ups in everyone's schedule.

- David: Starting sooner is best so I can avoid other deadlines. Scheduling ahead of time is unreliable because we can't predict how long each task will actually take. There will also be sufficient time to fix any issues that crop up.

- Gary: I would prefer to start sooner rather later so that we can find issues at early stage and keep ourself in the right track.

- Jonathan: I think that starting sooner would be the better approach as we would give ourselves more time to accomplish our goals as well as allowing larger margin of error for any problems or roadblocks that may come up. However, starting sooner is not always easy to do and procrastination is inevitable. In that sense, keeping track of our priorities and deadlines will be key to making sure that we complete the work that must be done before spending more time on aspects that can be considered a bonus.

- Kabishan: I believe I would benefit from pursuing both approaches in this regard, for starting sooner than later enables me to work through problems at my own pace rather than rushing through all of the work near the end. This would help me avoid silly mistakes and would give me ample time to think of new ways to approach problems. With regards to the second approach, it is best to prioritize my work in terms of deadlines or importance, which I can easily determine by using software like Outlook Calendar or GitHub Issues. This would also help me keep track of all of my deliverables for the capstone project.

8.2.3 Team Trust

A skill often ignored in student groups is team trust. We all know the pain of having someone not do their part and wait until the last minute to say something and have everyone else scramble to get it done. It's too late to think about what they should've done, so should build this skill now. One should trust others enough to say "I need help I can't get this done" and have understanding. We should aim to trust each other, not as students trying to do well, but as individuals of team caPstOneGroup making something great.

Learning Approaches

1. Collaborate on other courses.
2. Include team building exercises in meetings.

- Almen: I would prefer collaborating on other coursework with the team as we have a lot of assignments, it is difficult to find time to work on building that team trust outside working on other courses.
- Albert: I prefer collaborating on other courses as there's already too much work, and, with more work, someone is more likely to ask for help.
- David: Collaborative study is a fantastic way to build relationships and team chemistry. It can be far more effective than studying alone, and it is a good way to get an idea of what my teammate's strengths are.
- Gary: Collaborating on other courses would be suitable for me to build team trust since there is already too much work and if we can build team trust while working in other courses, that would be kill two birds with one stone.
- Jonathan: I think collaborating on other courses would be a great way to build team trust. Since we are all in the same program, we share many courses and can collaborate and help each other to do better in our courses. Also, if collaborate with each other on other courses, we could finish any work we have sooner and can allocate more time to develop our application.
- Kabishan: I believe that by pursuing team building exercises in meetings, I can improve this skill, for even a fun activity, such as playing board games, can reveal our personalities and allow us to better understand each other. By strengthening this facet of our relationship, we can also improve our trust in each other.

8.2.4 Front-end Development

Web frameworks will be unfamiliar to some members of our team, so all team members will need to familiarize themselves to make a good user interface. Even though not everyone will be working directly on the front-end, it will be helpful to have some knowledge while working on the back-end to help account for any information the front-end would need. Front-end requirements will help tailor back-end design decisions.

Learning approaches:

1. Use knowledge acquired from the web development group project in SFWRENG 4HC3, Human-Computer Interfaces.
2. Self-study using a online tutorials.

- Almen: With experience with front-end development, I believe that the best approach would be to self-study using online tutorials. This way, I can work at my own pace and refine what skills I would need to learn.
- Albert: As someone with no web development experience, the resources from SFWRENG 4HC3 make it seem like it's simple to do, but difficult to get right. More effort is needed getting it to look right than to work right.
- David: I have no front-end experience so both avenues will be necessary for me. 4HC3 requires a certain degree of self-study anyway because the web frameworks that will be needed are not part of the instructional plan.
- Gary: I had experience with front-end development three years ago and I have forgotten some of them, therefore the best approach for me is to review and self-study again.
- Jonathan: Since I already have some experience working with web frameworks and front-end development, I will try to learn and apply the knowledge from our SFWRENG 4HC3 course to apply good design practices that can help make our interfaces more intuitive and easy-to-use.
- Kabishan: I will use both approaches, where I will initially use the resources provided in the Human-Computer Interface course to get my bearings and then I will use a self-study online tutorial to seek additional materials. This will allow me to become a front-end developer that is more conscious about interface and interaction design.

8.2.5 Machine Learning

Most of the team does not have experience with developing and training Machine Learning models, which is a key aspect of our project. We have already designated Gary to lead our development in the Machine Learning aspects but it is important for every team member to have a basic understanding of how the models work. This way, we would be able to better gauge the progress of the project and set more realistic goals for ourselves.

Learning approaches:

1. Designate a time during some meetings for Gary to explain the basics of how to develop and train Machine Learning models.

2. Learn individually through online videos and tutorials.

- Almen: I believe that it would be best to have Gary, our machine learning expert, to designated time to teach us concepts regarding training and developing machine learning models. This is because machine learning can be a very daunting topic to learn and having someone who knows what is needed would be the best.

- Albert: Machine learning has always seemed intimidating and math heavy, so I hope Gary can break it down easier than finding info online.

- David: I hope that Gary will be a helpful mentor for all of us with his prior **ML** experience. On the side, individual study will also be important because of Gary's time constraints.

- Gary: I will be responsible for explaining the details for trainging and testing the **ML** model.

- Jonathan: I would personally do both approaches. The machine learning component is something that I want to focus on developing for this project and I would need to learn more in-depth on my own time before working with Gary to see how to put it all together in the context of our application.

- Kabishan: I find that I learn better when mentored by a domain expert, so I will be resorting to the option in which Gary will be explaining the basics of machine learning. Of course, for questions to which even Gary does not have the answers, I will be using the internet.

8.2.6 Project Deployment

Project deployment and hosting a web service on the cloud could be hard for some of the group members. There are configuration including domain names needs to be set up and building docker image pipeline. This is essential for the success of our project and proof of concept demo.

Learning approaches:

1. Designate a time during some meetings for Gary to explain the process and make sure everyone can deploy their code into production.

2. Learn individually through online videos and tutorials on Microsoft Azure help centre.

- Almen: For project deployment, I would try to learn through designated meetings conducted by Gary while supplementing the knowledge with online videos and tutorials as I believe it would be beneficial to learn in future endeavours.

- Albert: Like with machine learning, I hope Gary can lead a peasant such as myself to his level of understanding.
- David: I will use online videos and tutorials to brush up on my knowledge, while also attending any designated meetings to discuss the content.
- Gary: I will try to review the process individually through the Azure documentation because this is the most effective way.
- Jonathan: For project deployment, I would try to learn from videos and tutorials online as I believe it is an important skill to have for future projects and employment.
- Kabishan: Once again, I find that I learn better when being mentored by someone more experienced, as reading documentation can be tiresome and often irrelevant. As such, I shall consult Gary for learning the basics of project deployment and will seek alternate means to answer more advanced queries.

References

[1] James Robertson and Suzanne Robertson. *Volere Requirements Specification Template*. 16th ed. 2012.