Table 1: Revision History

Date	**Developer(s)**	**Change**
April 5, 2023	Albert, Almen, David, Gary, Jonathan, Kabishan	Revision 1

Reflection Report on Park'd

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

In our busy car-centric cities, finding a place to park can either be straightforward or a complete chore. How often do you actually enter a parking lot and immediately find a spot to your liking? Or do you often find yourself driving up and down, looking left and right for a good spot? What if there was a way to save you the trouble and show you an open spot directly?

Introducing Park'd, an application that aims to provide a solution to a bothersome daily chore by clearly showing open spaces to drivers, and providing directions to reach them. Park'd allows users to view every available parking space at participating parking lots in real time, while providing live navigation directions to the space of their choice.

1 Project Overview

The objective of Park'd is to leverage machine vision to determine the occupancy status of spots in parking lots, with minimal additional hardware required. All that is required for the model to function is a live stream of an overhead camera view of the parking lot, and annotations for parking spot locations in the camera view. The requirements for this project were focused on key user interactions, as well as the relevant distracted driving laws for the province of Ontario. Users needed to be able to select from all participating parking lots, select available parking spots, and navigate to them with location access. Administrators needed to be able to create new parking lots and mark spot locations on the camera view and the user-facing map view.

2 Key Accomplishments

- The project is able to detect parking spots based on real time video feed and the result is quite accurate.

- The entire project is hosted online which means everyone is able to access it.
- We consistently upheld our quality standards for delivering documentation and code, which resulted in a splendid outcome.
- Team members all had well-defined roles and areas of expertise. This made delegating tasks much simpler.
- Presentations were well-structured, engaging, and went exactly as planned.
- We followed human computer interface design principles while designing our application which makes our product look very professional, aesthetically pleasing, as well as promote usability of the project to our end users.
- We all challenged ourselves by taking on tasks in areas we were unfamiliar with which promoted personal development in those areas

3 Key Problem Areas

- The machine learning model is hardware-demanding which could've been more optimized if we retrain the fully-connected layers in the YOLOV3 architecture
- We migrated our project from React to vanilla JavaScript after the proof of concept demonstration because React introduced unnecessary complication for features that we were not making use of. This led to us needing to re-implement some functionality.
- Organizing realistic deadlines for six people and holding members accountable for them.
- With all members having a full course load, it was rather hard to manage our time for this course with many deliverables and other courses.
- There were many technologies and tools that could have helped us with managing our time/tasks and making some of our tasks easier, such as online time management tools with paid plans and paid assets. We decided against spending using paid technologies and tools which added more to our plates and made tracking tasks a little disorganized

4 What Would you Do Differently Next Time

- More meetings should be held regarding the development to keep everyone in the right track, and internal deadlines should be set to ensure work is done in time.

- Introducing redundancy into our system by adding more inputs besides video footage. For example, using ultrasonic sensor information. We can accomplish this by using a MVC pattern, where we can add a model for different input types.
- Be more reserved in our planning stages and expect features to be complicated than anticipated.
- Determine what functionality is shared between different parts of the application and implement it in one file so that important changes are reflected everywhere at once.
- Have an even more formalized meeting minutes and being more consistent with tracking our issues. It was rather difficult at times to keep track of what one was responsible of.