WatechPark: SMART Parking Capstone Project

Group Member Names: Elias Sabbagh, George Alexandris, Vikas Sharma

Department of Applied Technology, Computer Engineering Technology, Humber College Institute Of Technology & Advanced Learning

INTRODUCTION

Proposal:

Many busy parking lots are often plagued with congestion, with drivers competing to find a spot by cruising around and locating the right parking space. This is inefficient, time consuming where productivity is lost for consumers and businesses. The system we will be developing will address payment for parking, capacity management and location finding following an IoT approach using hardware and software. This project is focused on solving these issues by connecting consumers to parking lot owners and providing parking services by using a more convenient, simpler method to retrieve parking lot data seamlessly.

The main objective of this undertaking is to provide a more efficient and reliable platform to aid with parking scenarios. In particular, for the purpose of the consumer demographic who's in the market for an alternative parking lot management system. Our focus was to develop a platform, that would be the gateway to support consumers with finding the best parking space during any time, any place or anywhere in the world.

Idea:

Through the development of this product, we wanted to reach as many demographics and be able to provide an inexpensive and reliable platform where parking lot information can be retrieved at a glance. The idea of this project came up when the group realized that we can develop an easier way to find parking spots, by connecting all the spots to a SMART parking application.

We offer users with the ability to use a SMART parking mobile application to be able to add/manage cars, view parking lot data, make on-the go reservations for parking passes, accessible via an online database to send/receive information in real-time, all built-in with a simple and effective interface.

Background:

In the industry today, there have many occurrences where parking in general has become a hassle for city residents and parking lot owners. Due to this reason, it can lead to congestion in major traffic centric cities, with drivers competing to find a spot. This can be time-consuming, inefficient where productivity is lost for consumers and businesses. This project is focused on helping reduce the impact of this cause, by developing a system that will address payment for parking, capacity management, real-time information gathering to keep consumers up to date with their daily occurrences.

REQUIRED RESOURCES/TOOLS

Parts:

- VCNL4010 Proximity Sensor
- IR Break-Beam Sensor
- PCA9685 Servo Controller + 2 Micro-Servo Motors(entry/exit gate control)
- Raspberry Pi 4 Model B

Tools/Materials :

The hardware/software tools utilized include the following:

- Wire cutters, soldering iron, solder material, helping hand, pin headers, breadboard(PCB testing purposes), parts tool kit
- **Development Tools:** Android Studio, Fritzing , CorelDraw X7, Inkscape,

Facilities/People:

The prototype lab in Humber College is the main source of providing the services to etch the PCB board during its final stages of production as well as provide laser-cutting services initially planned. The electronics lab facility also allowed our team to work on the project, for both hardware/software purposes. Adjustments were made to the project through active involvement by our industrial collaborator from Parking, Mike Wrona.

BILL OF MATERIALS

Total Project Cost: $261.79

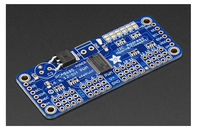
Product Name	Quantity	Unit Cost	Cost
Raspberry Pi 4b (4GB) Kit with power Supply and SD Card	1	$134.99	$134.99
VCNL4010 Proximity Sensor	4	$9.95	$39.80
IR Break Beam Sensor	2	$1.95	$3.90
PCA9685 PWM\Servo Controller	1	$19.84	$19.84
RGB LED (Pack of 10)	1	$5.95	$5.95
Power Adapter for external power	1	$12.98	$12.98
USB Camera	1	$15.00	$15.00
Micro Servo Motors	2	$5.95	$11.90
Jumper Wires	1	$2.59	$2.59
Resistor Kit	1	$14.89	$14.89
Total			$261.79

BOM – Final Project Budget

ELECTRONICS/PCB

In order to build this project in an efficient and clean manner, we made sure that most hardware sensors/effectors used in this project, utilize the I2C protocol, as it allowed for much easier testing and debugging for each individual sensor, and helped speed up and optimize the PCB design and creation process. For the proximity measurements to detect when a car is occupying a spot, we are using the VCNL4010 proximity sensor, which supports variable power input "3.3V – 5V" and capable of sensing objects up to 200mm away, as for our Gates servo and LED handling, we went with the PCA9685, which is a 16-channel servo/PWM controller, also supporting variable power input with an extra dedicated power bus for powering up the servos/LEDs. Our last sensor is an IR break beam sensor acting as an entry and exit detector, and is the only non I2C and analog sensor in our project, requiring it's own dedicated GPIO pins for data while sharing the same power input variability as the rest of the sensors.

VCNL4010

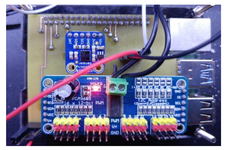


PCA9685 Servo/PWM controller

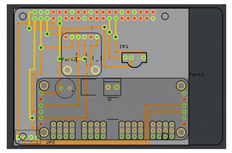
IR Break Beam Sensor

Our design and testing procedure started with each individual sensor being connected to the Raspberry Pi through a breadboard first, which in our case was the VCNL4010, occupying the 3.3v, SDA, SCL and ground pins for it to function. Once the VCNL4010's connections and testing was complete, we repeated the same testing scheme with the PCA9685 Servo controller using the same mentioned pins, and for the IR Break Beam, a simple logic test was sufficient. Finally, and once everything was confirmed to be working individually, we connected the sensors together and concluded the testing by confirming functionality and safe power load distribution.

The PCB design was created to be compact enough to not interfere with the Raspberry Pi's internals and have all the sensors present on it for direct debugging. A total of 10 VIA holes were used to route power, ground and the I2C dedicated buses to all sensors while also accounting for future expandability options, the design was thoroughly tested in simulations prior to production and after production to ensure safe operation for both the PCB and the Raspberry Pi.



Assembled and soldered PCB



PCB Design in Fritzing

FIRMWARE

The Python firmware was designed coded with clear logic in mind, as we needed the parking lot to be independently reactive to each sensor, and for that we implemented the multithreading library and divided the code into separated functions that are run by individual threads that only share access to specific parts of the database.

Here's how the firmware logic flows:

- At the start, all needed variables are declared and a connection to the database using the Pyrbase library is established by calling an external file that returns a database object, this method was used to hide the database credentials and improve security during and after development
- The next stage to be to activate each thread in the following order, "adminControl", "vcnlFunc" and finally "irbFunc"

- adminControl: monitors if the admin status has been activated in the database and calls on the entGate function to open the entry gate when needed

- vcnlFunc: establishes a connection to the VCNL4010, takes measurements every 5 seconds and updates the database accordingly

- irbFunc: handles the logic reported by the IR Beam and starts the process of car verification by calling the licCheck function which does license plate comparisons and then the entGate function and extGate functions which handle instructing the designated servos to open and close the gates

The firmware extensively documented and equipped with error handling and reporting logic to allow for easier debugging and future modifications

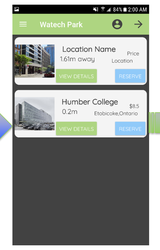
MOBILE APPLICATION

Development Environment: Android Studio (JAVA) – API 21 and above

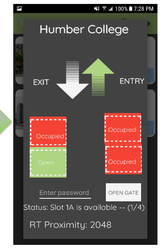
The mobile application works alongside an online database structure through the Firebase database and on-site devices which include the VCNL4010 Proximity sensor, IR Break Beam Sensor, and the 2 servo motors running alongside the PCA9685 servo controller.

The application follows a login and authentication structure and is designed with various screens in mind to support our consumer application. This includes options to add a car, manage added cars, view real-time parking lot data and provide status updates/changes through the supporting sensors/effectors and the data sent and retrieved by the online database. Other features include consumer abilities to reserve a spot in a parking lot, select a parking pass, payment services and ability to view order history/transactions. As well as access settings, customize language preferences and other in-app options.

Login Screen



Home Screen

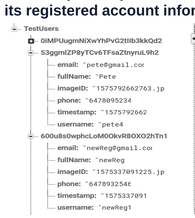


'View Details' Screen

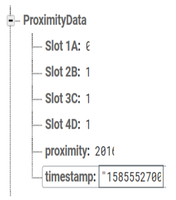
DATABASE CONFIGURATION

The online database is configured based on essential criteria needed to access the mobile application, hardware and vice-versa. The main source of delegating data is done through the Google Firebase database.
There are five main data structures used in the project, along with four sub-siding tables used for the purpose of the mobile application, and other intended functionality of our parking application. SQL scripts are used with Firebase/Pyrebase, to gain access to the Firebase API using the API key for both the mobile application and parking lot prototype.

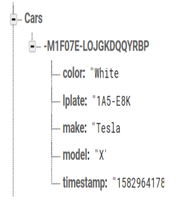
The 'TestUsers' data structure stores registration details specific to the user. The UID (user ID) acts as the primary key identifying each existing user and its registered account information.



'TestUsers' Table



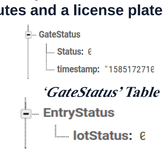
'ProximityData' Table



'Cars' Table

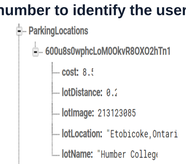
The 'ProximityData' structure stores raw proximity values sent from the VCNL4010 hardware device, and is retrieved by the mobile application to display the real-time proximity levels of the lot. 'Cars' table stores each registered vehicle attributes and a license plate number to identify the user.

'Orders' Table



'GateStatus' Table

'EntryStatus' Table



'ParkingLocations' Table

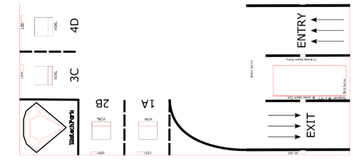
'Orders' table stores the payment processing details of the user using a OID(foreign key) to identify the order. The 'GateStatus' table stores IR Break Beam entry/exit status and a timestamp to indicate the exact time an action is performed. The 'EntryStatus' table stores a value of 0 each time the lot is full to track the overall status. The 'ParkingLocations' table stores parking lot details under the specific UID of the current user.

The 'AdminControl' table stores the status of the gate and sends a value of 1 or 0 to indicate an opening or closing of the barrier.

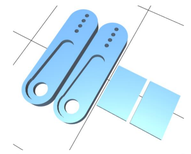
ENCLOSURE/PROTOTYPE

The enclosure would have been set to have a parking lot prototype along with the Raspberry Pi/PCB attached to the center of the entrance and exit. This design would have been used to house the Raspberry Pi platform/PCB components, serving as solid protection from any outside harm and ensuring the safety of the project assembled. The following visuals below, showcase the end design of the SMART parking lot prototype and the final enclosure to hold the Raspberry Pi/PCB unit .

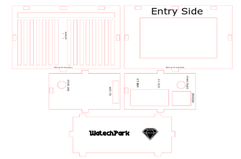
Parking Lot Prototype Design(Inkscape)



As shown above, between the entrance and exit there is available space for the Raspberry Pi and the sensors attached. The sensors/effectors would go through under the parking lot and connect back to the space allotted. The IR Break-Beam sensors are connected to the entry and exit. The VCNL4010 proximity sensor is located at Slot 1A, which would be connected through a wall and attached facing the parking spot to detect the presence of a car approaching the spot. The camera would have been attached to a bar over the entry to scan the license plate on top of the car. The 3D printed barrier would have been attached to the sides of the servo motors horns, designed using the Inkscape software along with the case as shown below. The enclosure would clip onto the SMART parking lot model and hold the Raspberry Pi/PCB in place.



3D Printed SG90 Servo Horn Extender(barrier)



Final Enclosure Design(Inkscape)

CONCLUSIONS: (Next Steps)

The WatechPark IoT capstone project was developed to address the consumer demographic by determining a alternative method in regards to payment for parking, capacity management, and real-time information gathering. During the course of fifteen weeks, we have worked extensively on designing, developing, and integrating hardware/software components in preparation of unit testing, mass production testing phases, and overall refinement. Therefore, through the combined effort of all team members, we have successfully achieved our proposed objective by creating an alternative parking lot management system, to assist with everyday consumer occurrences and parking situations.

Future steps may include, a more compact PCB design to accompany the use of the two servo motors. Thus, reducing the overall footprint size of the project by a more considerable amount and allowing further room for improvement in terms of hardware use/capabilities. Other possible additions, may include adding support for each available parking spot on our prototype model. This would require the use of a I2C multiplexer to allow individual VCNL4010 proximity sensors to be positioned at each parking space, rather than the current single VCNL4010 device. Additionally, to appeal to the mass market a web application may be developed to allow a wide range of public users the ability to view advanced metrics/live parking lot data. The application would mirror the mobile application and allow the consumer to interact through a global reach. Thus, in return gaining further popularity to contend as a major alternative SMART parking IoT platform from a investors standpoint.

REFERENCES

Park Indigo Canada Inc. (2019). Indigo. Retrieved from https://ca.parkindigo.com/en

ParkWhiz. (2019). Find and Book Parking Anywhere. Retrieved from https://www.bestparking.com/

EasyPark. (2016). Retrieved from EasyPark Mobile Parking App: https://www.easypark.ca/products-services/mobile-parking-app

Thingiverse. (2019). SG90 Servo Horn Extender. Retrieved from https://www.thingiverse.com/thing:3814678

Adafruit Industries. (2020). Adafruit. Retrieved from https://learn.adafruit.com/