

# ParkSmart

Group name: Khukuri

Members: Anish Timila, Topa Timilsena, Sulav Poudyal

## Table of Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
1.1	Overview	4
1.2	Purpose	4
1.3	System Analysis and Description	4
<b>2</b>	<b>Hardware Overview</b>	<b>5</b>
2.1	Raspberry Pi 3	5
2.2	Ultrasonic sensor	6
<b>3</b>	<b>Software Requirement Specification</b>	<b>6</b>
3.1	Functional Requirement	6
3.1.1	System Features	6
3.1.2	Potential System Features	7
3.2	Non- Functional Requirement	7
3.2.1	Performance Requirement	7
3.2.2	Safety Requirement	7
3.2.3	Maintenance and Support Requirements	7
<b>4</b>	<b>Design Overview</b>	<b>7</b>
4.1	Use Case Diagrams	7
4.1.1.	Find Parking	7
4.1.2.	Safe Ride	8
4.1.3.	Night Shuttle	9
4.2	Sequence Diagrams	9
4.2.1.	Find Parking for Students	9
4.2.2.	Night Shuttle	10
4.2.3.	Safe Ride	11
4.3	Class Diagram	12
<b>5</b>	<b>Implementation</b>	<b>13</b>
5.1	Connecting Raspberry Pi With Ultrasonic Sensor	13
5.2	Setting up MySQL on Raspberry Pi	14
5.3	Creating MySQL database	15
5.4	Python Code to Manipulate Sensor Data and Update Database	15
5.5	Setting up phpMyAdmin	16
<b>6</b>	<b>User Interface</b>	<b>18</b>
<b>7</b>	<b>Functionalities</b>	<b>19</b>
7.1	Completed	19
7.2	Ongoing	19
7.3	Future Implementation	19
<b>8</b>	<b>Implementation Issues</b>	<b>19</b>
8.1	Installing MySQL on Raspberry Pi	19

8.2	PIR Sensor Detection	19
8.3	Ultrasonic Sensor Alignment	20
8.4	Displaying JSON Data On Android Application	20
<b>9</b>	<b>Contribution</b>	<b>20</b>
<b>10</b>	<b>Interview Summary</b>	<b>20</b>
<b>11</b>	<b>References</b>	<b>24</b>

# 1 Introduction

## 1.1 Overview

Our product ParkSmart allows students and visitors to find parking in around Texas Tech University. The ParkSmart application provides information to commuter students, faculty, visitors, and residence students about their designated parking lots.

ParkSmart is an android application that helps user to find empty parking places around Texas Tech University using Raspberry PI with the help of ultrasonic sensors. For students and visitors, the system will display the parking spaces available on each rows of a particular parking lot. The system will also give direction to the parking lots using GPS and google maps. Apart from the parking spaces, the system will also provide useful information on SafeRide and NiteShuttle.

## 1.2 Purpose

This document provides the detailed design description of the functionalities in the ParkSmart. It will give the overall detailed view of our project illustrating with the basic hardware and software requirement specification, design view concepts, implementation, and the functionalities. The intended audience of this document is Texas Tech Parking holders and college town people as well.

## 1.3 System Analysis and Decomposition

The system can be decomposed into hardware and software part.

On the hardware side, the system is using a RaspberryPi 3 along with motion sensor. The system also has necessary power supply and wiring equipments. The main purpose of the hardware part of the system is to gather the data using motion sensor. The gathered data is stored in a database.

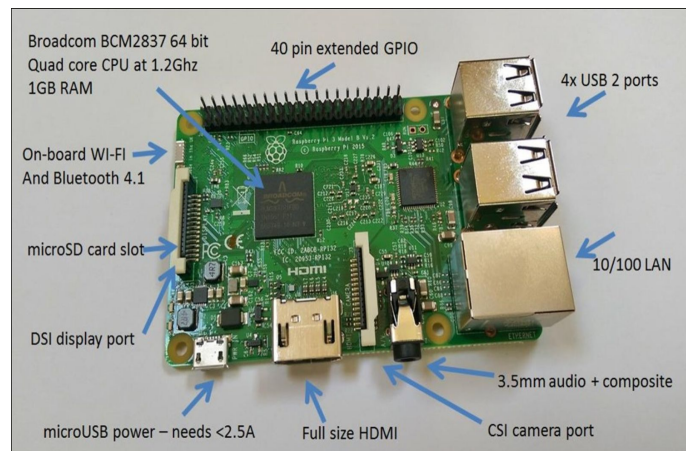
Software side of the system deals with android application, phpMyAdmin, MySQL database, and Python script. The main mechanism is once the Ultrasonic sensor retrieves the data, it is stored in MySQL database. phpMyAdmin handles the MySQL database and sends JSON data to the endpoint which is retrieved by Android App and displayed to user.



## 2 Hardware Overview

### 2.1 Raspberry Pi 3

A Raspberry Pi is a credit card-sized computer which is plug with TV or monitor and a uses mouse and keyboard. The main goal is to create a low-cost device that would improve programming skills and hardware understanding at the school level. Raspberry Model B is the third generation raspberry pi which is 10X faster than first generation. In addition, on-board Wi-Fi and bluetooth connectivity makes this model more ideal solution for connectivity design. 1GB RAM, 64 bit CPU and 40 pin extended GPIO (General Purpose Input/Output pins) helps to work on bigger and real world project.



Pi Model B/B+			
3V3 Power	1	2	5V Power
GPIO2 SDA1 I2C	3	4	5V Power
GPIO3 SCL1 I2C	5	6	Ground
GPIO4	7	8	GPIO14 UART0_TXD
Ground	9	10	GPIO15 UART0_RXD
GPIO17	11	12	GPIO18 PCM_CLK
GPIO27	13	14	Ground
GPIO22	15	16	GPIO23
3V3 Power	17	18	GPIO24
GPIO10 SPI0_MOSI	19	20	Ground
GPIO9 SPI0_MISO	21	22	GPIO25
GPIO11 SPI0_SCLK	23	24	GPIO8 SPI0_CE0_N
Ground	25	26	GPIO7 SPI0_CE1_N
ID_SD I2C ID EEPROM	27	28	ID_SC I2C ID EEPROM
GPIO5	29	30	Ground
GPIO6	31	32	GPIO12
GPIO13	33	34	Ground
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
Ground	39	40	GPIO21
Pi Model B+			

www.raspberrypi-spy.co.uk

Raspberry Pi 3 has 40 pinouts. All the models have different configuration of the pins. There are 14 pins and 26 pin headers/GPIO in Raspberry Pi 3. There are total of 8 ground pins as shown in the figure. Pin 1, 2, 4 and 17 are for the power. The devices connected to the pins shall be assembled to the pin voltage to make it work properly. The Pin 1 and Pin 17 provides 3.3V whereas Pin 2 and 4 both provide 5V. The 5V draws current directly from microUSB so is able to use the leftover of the power after the board uses its required amount.

All the pins with GPIO are pins that can be used to determine the input data from the sensor devices. GPIO's can be easily implemented in the Python. Python has a library that can be imported for Raspberry Pi for the GPIO to set up input and output channels:

Code:

```
import RPi.GPIO as GPIO          //Importing library
GPIO.setup(7, GPIO.IN)           //GPIO input channel
GPIO.setup(8, GPIO.OUT)          //GPIO output channel
```

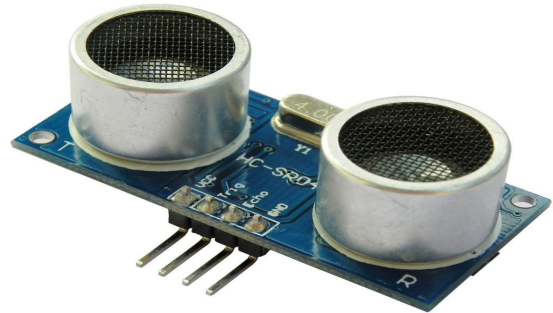
It is required to focus properly on setting up GPIO pin connections.

## 2.2 Ultrasonic sensor

Ultrasonic sensor consists of transmitters, receivers and control circuit. High frequency ultrasonic sound is sent from the transmitter. The ultrasonic sound reflects and returns back, which is then detected by receiver. The control circuit uses the detection made by receiver to calculate the time taken for the sound to return back. This time can be used, along with speed of sound to determine the distance of reflected object.

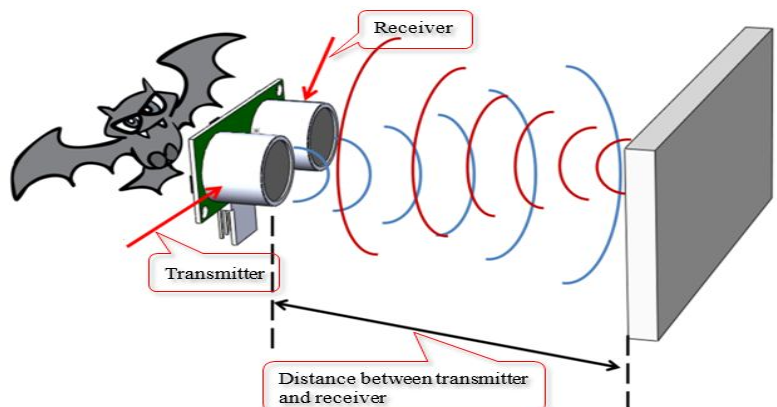
The working mechanism of the Ultrasonic Sensor is quite simple. In the Ultrasonic sensor there are two points which are Transmitter and Receiver.

Transmitter transmits the sound wave. The wave strikes the object after certain distance then it is bounced back to the device where receiver receives the signal. Calculating the time taken and with the help of speed of sound, the distance to the object can be easily determined using simple Physics.



**Electrical parameters**

Electrical Parameters	HC-SR04 Ultrasonic Module
Operating Voltage	DC-5V
Operating Current	15mA
Operating Frequency	40KHZ
Farthest Range	4m
Nearest Range	2cm
Measuring Angle	15 Degree



## 3 Software Requirement Specification

### 3.1 Functional Requirements

#### 3.1.1 System Features

- The application shall give option to choose parking lot base on user : Student, Visitor, and Faculty.
- The application shall navigate to user's selected parking lot.
- The application shall provide the number of parking spot availability based on user's selection by lot name.
- The application shall provide information of Night Shuttle.
- The application shall provide information of Safe Ride.

### **3.1.2 Potential System Features**

- The ParkSmart system shall provide information of event's parking.
- The ParkSmart system shall provide information of bus route.

## **3.2 Non-functional Requirements**

### **3.2.1 Performance Requirements**

- ParkSmart system shall update parking spot availability in every 10 sec.
- ParkSmart system shall take less than 5 sec to response.

### **3.2.2 Safety Requirements**

- The system shall dissipate heat produced by the components of the system to prevent overheating.
- The system shall update the parking details with zero interaction while driving to avoid accidents.

### **3.2.3 Maintenance and Support Requirements**

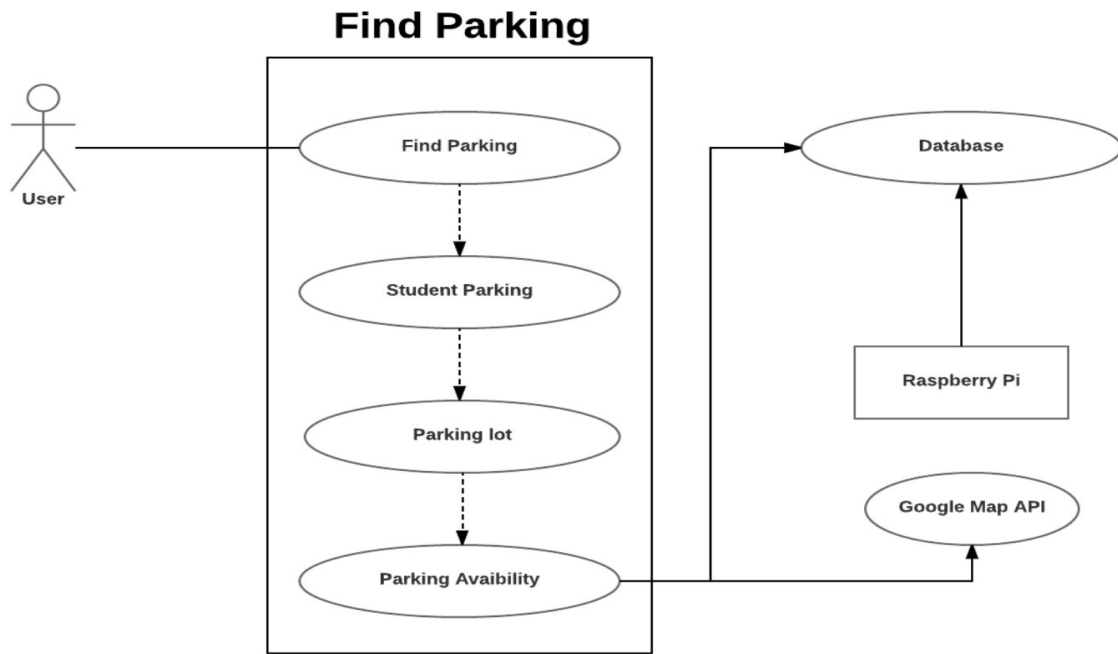
- The system shall have at least android version of 4.1 or above.
- The battery on ParkSmart hardware system shall be easily replaceable without the need of disassembling the product.
- The hardware system shall be easily replaceable if necessary.

## **4 Design Overview**

### **4.1 Use Case Diagrams**

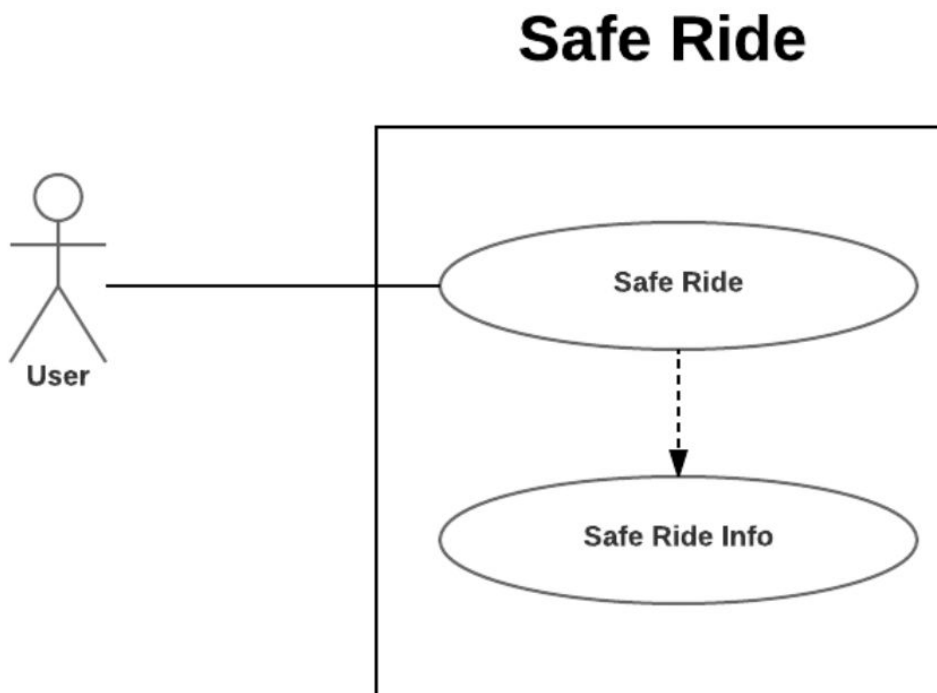
#### **4.1.1 Find Parking**

Actors:	Users
Precondition:	The user has downloaded the application.
Postcondition:	The application will give the direction and number of available parking spots at selected student parking lot.



#### 4.1.2 Safe Ride

Actors: User  
 Precondition: The user has downloaded the application.  
 Postcondition: The application will display information about Texas Tech University's 'Safe Ride' service.





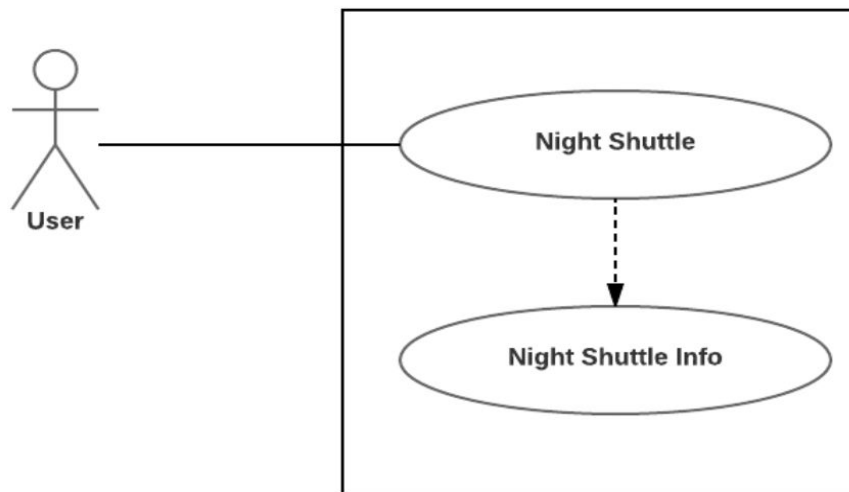
### 4.1.3 Night Shuttle

Actors: User

Precondition: The user has downloaded the application.

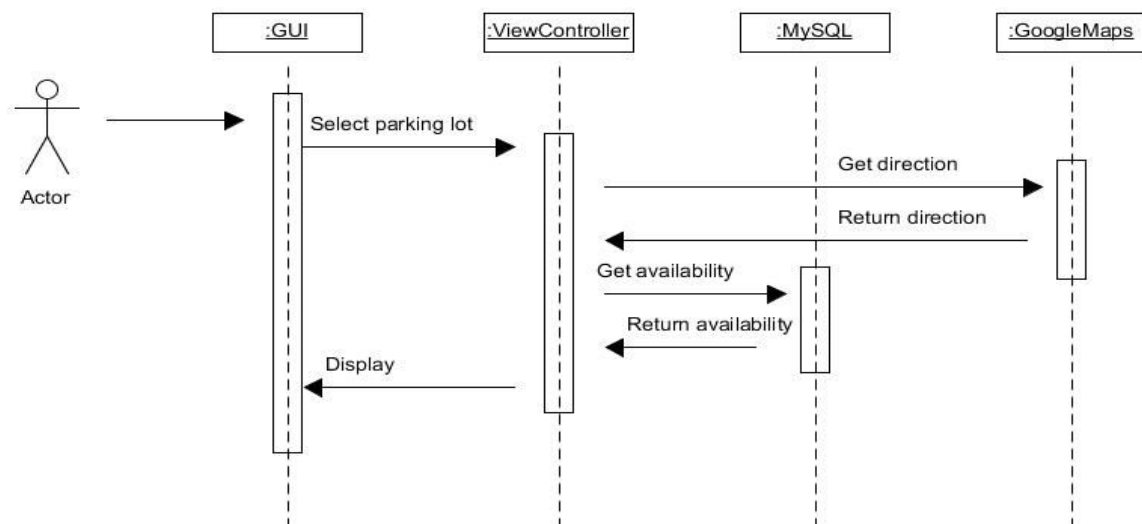
Postcondition: The application will display information about Texas Tech University's 'Night Shuttle' service.

## Night Shuttle



## 4.2. Sequence Diagrams

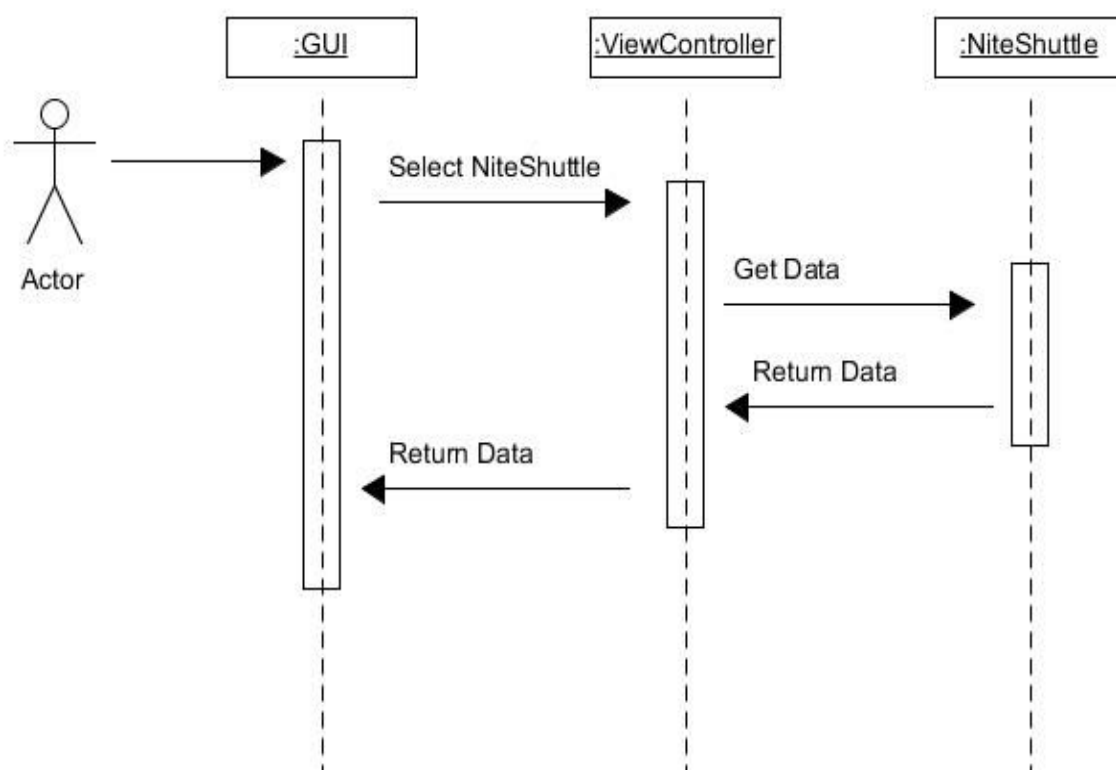
### 4.2.1. Find Parking for Students



This sequence diagram describes the main functionality of ParkSmart application, find parking. The main functionality of ParkSmart application is to display the number of available parking space. The objects in the diagram include an interface, controller, mysql database and google maps api.

When the user selects a parking lot, the application gets the direction to the parking lot. After getting the direction, the application gets the number of available parking spots from MySQL database. Then the application displays the number of available parking and direction of the parking lot to the user.

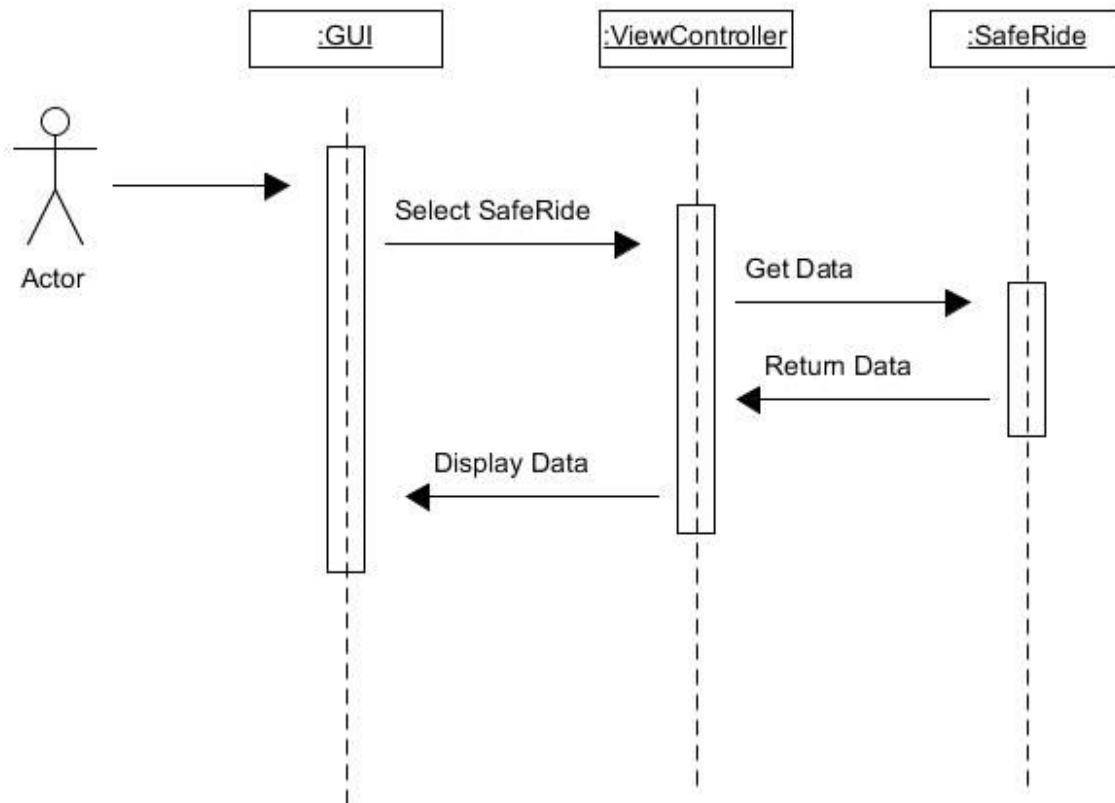
#### 4.2.2. Nite Shuttle



This sequence diagram describes another functionality of the application. It displays the information about 'Nite Shuttle' service of Texas Tech University. The objects in this diagram are interface, controller and nite shuttle data.

When the user selects 'Nite Ride' from main menu, the controller gets the information about 'Nite Ride'. After receiving the information, the controller displays it to the user.

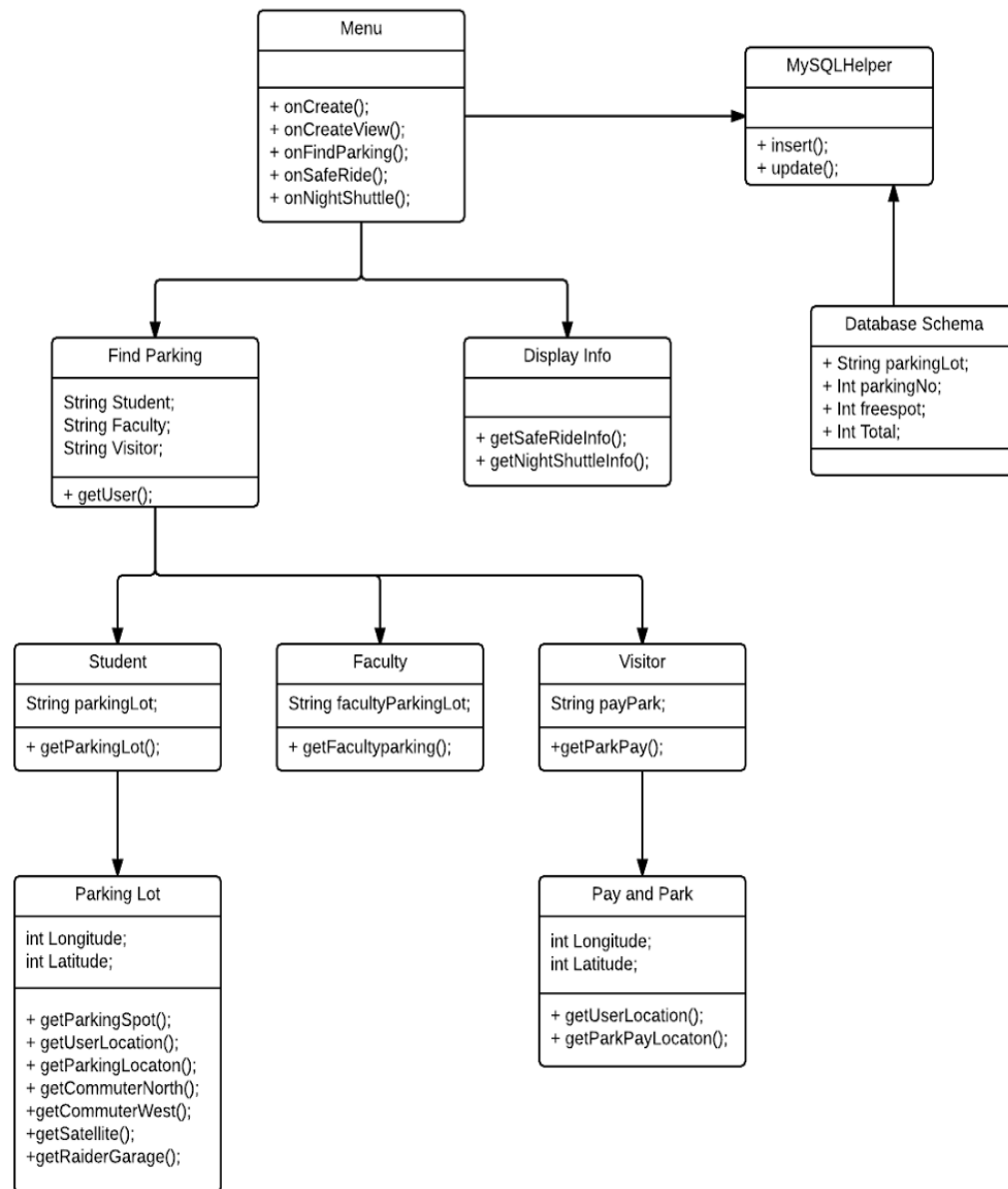
#### 4.2.3. Safe Ride



The above sequence diagram describes the functionality of 'Safe Ride' option. This provides the information about 'Safe Ride' service of Texas Tech University. The objects involved are interface, controller and data source.

When the user selects the 'Safe Ride' option from the main menu, the controller gets the information from the source. After getting the information, the controller displays the information to the user.

### 4.3 Class Diagram



ParkSmart Application has the main class that displays menu to the user on FindParking, SafeRide Info, and NiteShuttle Info. The main menu connects with the MySQL database through the helper forming the database schema of parking lot, parking lot number, freespot, and total no of parking lots. The main menu divides into sub classes find parking and display info. Find parking further sub divides into Student, Faculty and Visitor. Student class has parking lot class that has the attributes of longitude and latitude. Student class also has the methods to getParkingSpot, and getUserLocation from all the parking lots i.e. Commuter North, Commuter West, Satellite, and Raiders Garage.

Similarly, Visitor class subdivides into Pay and Park class which has the attributes of longitude and latitude of the pay and park locations. It also has methods to getUserLocation and getParkPayLocation.

## **5 Implementation**

To give directions to different parking lots, the system will use google maps api. The system will also use the GPS from the user's smartphone to find their current location. The location of the parking lots on Texas Tech University will be pinned in the system which will allow the users to select and get directions to any of the parking lots.

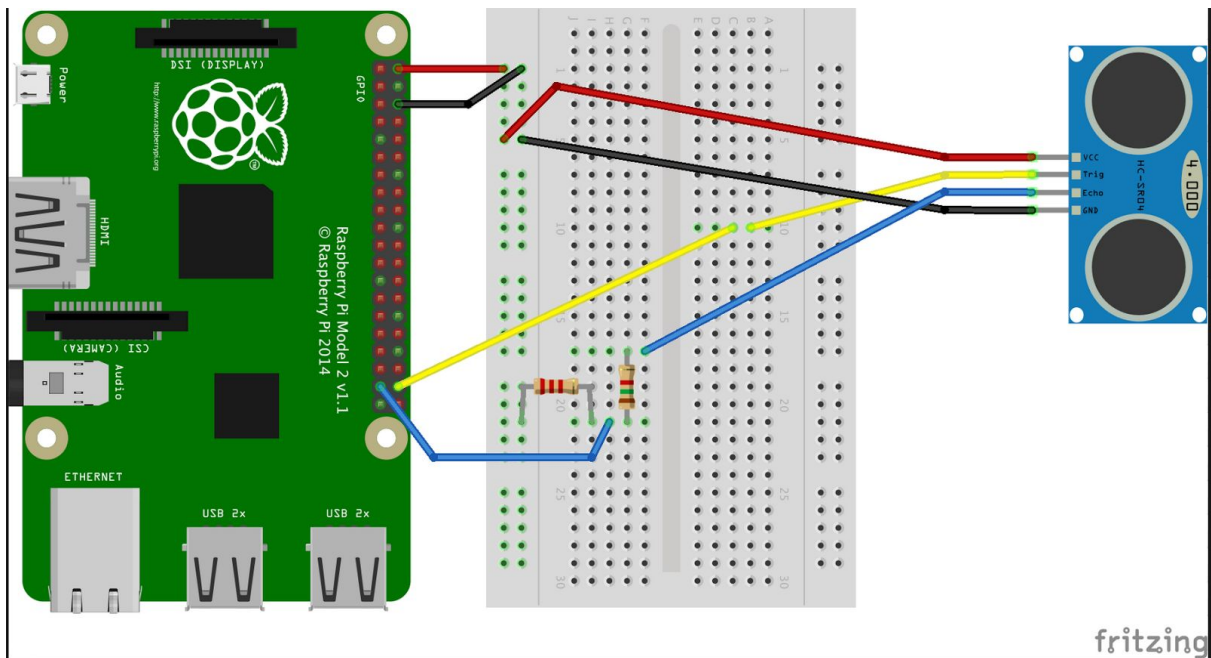
To identify the available number of parking spots, the system will use Raspberry Pi 3 and motion sensor. Motion sensor will detect if a car is parked and send the data through Raspberry Pi to the MySql database. The data will be processed in the Raspberry Pi and will be stored MySQL database.

We will be building an android application to provide an interface for our system. The android application will be used as an interface to give direction and display the number of available parking spots. To display the information about available parking spots, the android application will connect to the MySQL database and get the information.

### **5.1 Connecting Raspberry Pi Model B with Ultrasonic Sensor**

Ultrasonic sensor has four pins: ground, echo, trigger, and power supply(Vcc). The module is powered by connecting Vcc to GPIO pin 1. The ground is connected to GPIO pin 6, which is the designated pin for ground. Trigger is connected to GPIO pin 23 . Input signal is sent to trigger, which triggers the sensor to send an ultrasonic pulse. The pulse is reflected back and is detected by the receiver. Then the time between the trigger and returned pulse is measured by the sensor, after which the sensor sends a 5V signal on the echo pin. The echo pin of the sensor is connected to GPIO pin 24.

Since the sensor sends 5V signal on the echo pin, resistors have to be used to limit the power sent to raspberry pi. Two resistors of value 1K Ohm and 2k Ohm are used to limit the power going into the raspberry pi. Breadboard and jumper cables are used to complete the connection.



## 5.2 Setting up MySQL database in Raspberry Pi

MySQL can be installed in raspberry using the following commands. Internet connection is required for these commands to work.

```
sudo apt-get install mysql-server python-mysqldb #Installing modules mysql-server and python-mysqldb
python-mysqldb
sudo apt-get install default-libmysqlclient-dev #Installing module default-libmysqlclient-dev

$ mysql -u root -p #Starting MySQL as root user
Enter password:
maria> CREATE DATABASE ParkSmart #Creating a database ParkSmart
maria> USE ParkSmart #Selecting database ParkSmart

#Creating a user for the database named 'monitor'
maria> CREATE USER 'monitor'@'localhost' IDENTIFIED BY 'password';

#Granting all access to the newly created user 'monitor'
maria> GRANT ALL PRIVILEGES ON ParkSmart.* TO 'monitor'@'localhost'

#Reloads the grant table so that recent changes are implemented
maria> FLUSH PRIVILEGES;
maria> quit
```

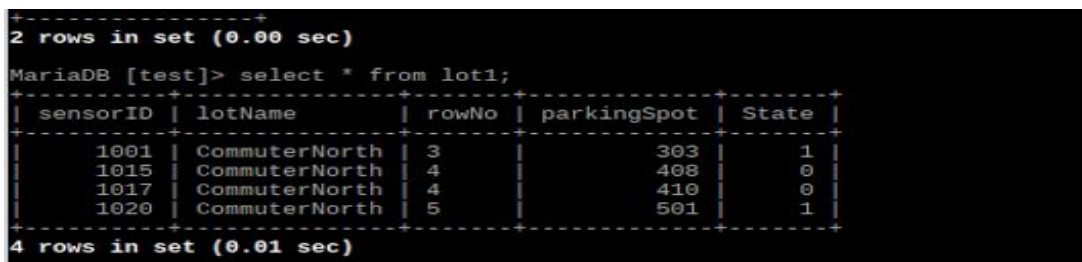
### 5.3 Creating MySQL database

After the hardware has been set up, we need to install MySQL on raspberry pi. For this project, we store the sensor data on the database and manipulate the data to show the available number parking spots. MySQL can be set up in raspberry by:

```
sudo apt-get install default-libmysqlclient-dev
sudo apt-get install mysql-server python-mysqldb

$ mysql -u root -p
Enter password:
mysql> CREATE DATABASE ParkSmart
mysql> USE ParkSmart
mysql> CREATE USER 'monitor'@'localhost' IDENTIFIED BY 'password';
mysql> GRANT ALL PRIVILEGES ON ParkSmart.* TO 'monitor'@'localhost'
mysql> FLUSH PRIVILEGES;
mysql> quit
```

Here, we set up a SQL database called ParkSmart and create a user called 'monitor', who has full access to the database ParkSmart. Then we set up a table for a parking lot. A sample table is given below:



```
2 rows in set (0.00 sec)

MariaDB [test]> select * from lot1;
+-----+-----+-----+-----+-----+
| sensorID | lotName      | rowNo | parkingSpot | State |
+-----+-----+-----+-----+-----+
| 1001     | CommuterNorth | 3     | 303         | 1     |
| 1015     | CommuterNorth | 4     | 408         | 0     |
| 1017     | CommuterNorth | 4     | 410         | 0     |
| 1020     | CommuterNorth | 5     | 501         | 1     |
+-----+-----+-----+-----+-----+
4 rows in set (0.01 sec)
```

We have five columns in the table, sensorID, lotName, rowNo, parkingSpot, and state. Each sensor is associated with a sensor id as we have assumed each parking spot in a given parking lot, has a ultrasonic sensor. The lotName, rowNo and parkingSpot are all given information and these three information will be specific to each sensor id. The last column, state, is updated based on the sensor data. If a car is parked, state is changed to 1, and if there is no car parked, state is changed to 0. Then by running a SQL query to count all the rows, where state is 0, we can find the number of free parking spots.

### 5.4 Python Code to Manipulate Sensor Data and Update Database

Following is the screenshot of the code that we used to get the sensor data from the ultrasonic sensor and then use that data to update the database. The code is executed in the raspberry pi.

First we import all the required libraries using import statements. Then we set the GPIO pins of the raspberry and also set the values to connect the SQL database. We have a function called "pingy" that gets the distance from the raspberry pi. If the distance returned by the sensor is less than 10 cm, it means that a car is parked in the lot, so we update the state in the database to 1. We use a variable called 'flag' to keep track of the state in the

code. Then when a car moves out of the parking spot, distance measured by the sensor will be greater than 10 cm and flag will be 1, so the state variable in the database is updated to 0.

```
#!/usr/bin/env python

import MySQLdb
import time
import MySQLdb
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
db = MySQLdb.connect("localhost", "root", "password", "test")
curs=db.cursor()
TRIG = 23
ECHO = 24
GPIO.setup(ECHO,GPIO.IN)
GPIO.setup(TRIG,GPIO.OUT)
end=0
start=0
flag1=0
def pingy():
    GPIO.output(TRIG, False)

    time.sleep(2)
    GPIO.output(TRIG, True)
    time.sleep(0.00001)
    GPIO.output(TRIG, False)
    while GPIO.input(ECHO)==0:
        pulse_start = time.time()
    while GPIO.input(ECHO)==1:
        pulse_end = time.time()
    pulse_duration = pulse_end - pulse_start
    distance = pulse_duration * 17150
    distance = round(distance, 2)
    print ("Distance:",distance,"cm")
    return distance
print ("Reading Distance \n")
try:
    while True:
        dist=pingy()
        print("Inside loop with distance",dist)
        if(dist < 10 and flag1==0):
            print("Car moving in")
            time.sleep(1)
            flag1=1
            try:
                curs.execute ("UPDATE NCOM set state=1 where SensorNo=1001;")
                db.commit()

            except:
                print ("Error updating database")
                db.rollback()
        elif (dist > 10 and flag1==1):
            print("Car moving out")
            time.sleep(1)
            flag1=0
            try:
                curs.execute ("UPDATE NCOM set state=0 where SensorNo=1001;")
                db.commit()
```

## 5.5 Setting up phpMyAdmin

After the database is updated, it is connected with phpMyAdmin server in order to be accessed remotely. phpMyAdmin is installed in the Raspberry Pi and the settings are configured with the required credential to access MySQL database. The database can be easily managed by logging in to the Raspberry Pi IP address with phpMyAdmin route. The "lot1" database consists of sensorID (Primary Key), lotName, rowNo, parkingSpot, and State. State determines the parking availability of a particular spot. State is either 1 or 0. 1 indicates that the parking is occupied whereas 0 indicates that the parking is free as shown in the figure below:



```
SELECT * FROM `lot1`
```

☐ Show all | Number of rows: 25 ▼ | Filter rows:

sensorID	lotName	rowNo	parkingSpot	State
1001	CommuterNorth	3	303	1
1015	CommuterNorth	4	408	0
1017	CommuterNorth	4	410	0
1020	CommuterNorth	5	501	0
1007	CommuterNorth	3	301	1
1010	CommuterNorth	3	303	0
1013	CommuterNorth	4	402	0

The following query is implemented in order to show free parking availability by Row of the parking lot. SQL Query to show free parking spots by Row:

```
SELECT `rowNo`,`lotName`, COUNT(*) AS FreeSpace FROM `lot1` WHERE `State`=0 GROUP BY `rowNo`
```

Result:

← → ↻ ⓘ Not secure | 192.168.0.17/phpmyadmin/tbl\_sql.php?db=test&table=lot1&sql\_query=SELECT+%60rowNo%60%2C%60lo

Apps ★ Bookmarks 8 Polymorphism in Java Blackboard Learn raiderlink Hire Red Raiders | Un Game Design CS4397 Li

**phpMyAdmin**

Recent Favorites

- New
- information\_schema
- mysql
- performance\_schema
- test
  - New
  - lot1
  - NCOM
  - people

Server: localhost:3306 » Database: test » Table: lot1

Browse Structure SQL Search Insert Export Import

✓ Showing rows 0 - 2 (3 total, Query took 0.0024 seconds.)

```
SELECT `rowNo`,`lotName`, COUNT(*) AS FreeSpace FROM `lot1` WHERE `State`=0 GROUP BY `rowNo`
```

☐ Profiling

☐ Show all | Number of rows: 25 ▼ | Filter rows:

+ Options

rowNo	lotName	FreeSpace
3	CommuterNorth	1
4	CommuterNorth	3
5	CommuterNorth	1

## 6 User Interface



Fig. 1

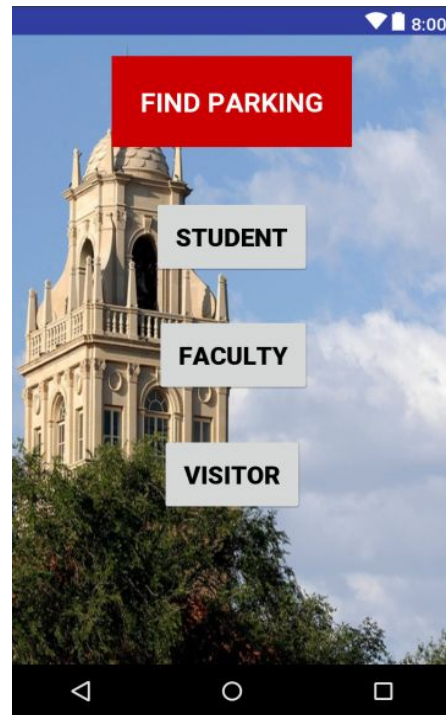


Fig. 2



Fig. 3

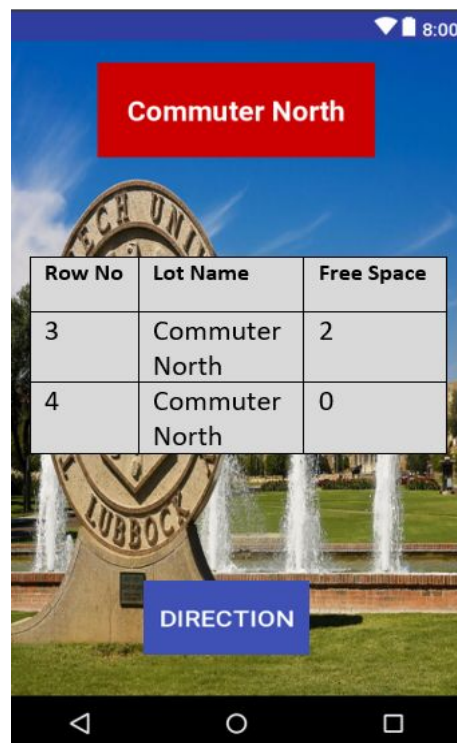


Fig. 4

Row No	Lot Name	Free Space
3	Commuter North	2
4	Commuter North	0

The above interfaces gives the basic ideas of our ParkSmart application. Fig. 1 is the main menu from where users can make his/her desired selection. Fig.2 displays the option of the Find Parking. Fig. 3 has component of the student's parking lots where user's have to select parking lot based on their parking permit. Finally, fig. 4 gives the information of the total number of parking availability, parking availability details of each row and navigation option as well.

## **7. Functionalities**

### **7.1 Completed: Hardware Part and Layout of App**

We completed the hardware part and simple layout of application in the project. Now, we are getting number of available parking of designated parking lot on a web server.

### **7.2 Ongoing: Connecting phpMyAdmin to Android App**

Currently, we are working on connecting the MySQL data to android app in which we can display the information of the parking details.

### **7.3 Future Implementation: GPS Navigation**

We consider the GPS functionalities in our project as the future implementation. In this functionality, GPS will help to navigates user to the designated parking lots around the Texas Tech University. For example, new visitor at Texas Tech university can use this app to find the pay and park station around the campus.

## **8. Implementation Issues**

We encountered few issues while implementing this project. They are described below.

### **8.1 Installing MySQL on Raspberry Pi:**

While installing MySQL on raspberry pi, we got an error saying 'mysql\_config' file not found. We found that the error was common and had many solutions posted online. But none of those solutions worked for us. After searching for a few days, we found that installing 'default-libmysqlclient-dev' fixed the error. As its name suggests, this library should have been installed in the raspberry pi by default but it was not, that is why the other solutions weren't working for us.

### **8.2 PIR sensor detection:**

Initially, we planned on using PIR motion sensor for our project. It would detect the car coming in the parking spot and we would use that data to update the state in the database. It turned out that PIR motion sensor detects heat as well, so it kept detecting motion because of the heat fluctuation as well. So we switched to ultrasonic motion sensor to get more accurate data.

### **8.3 Ultrasonic Sensor Alignment**

To get accurate reading from the ultrasonic sensor, it has to be placed directly in front of the object. If it is placed at an angle, the sound waves get reflected at multiple places and we get inaccurate data. Due to this, we place the sensor at the top of the parking spot, faced downward, to get accurate data.

### **8.4 Displaying JSON Data to the Android App**

It was hard to display the JSON data retrieved from the MySQL database. No matter the fact that the JSON data was well received by Android App, it was hard to display it in the Android App using the display adapter.

## **9. Contributions**

### **Anish:**

- Creating general layout of the ParkSmart Application in Android Studio 3.0.
- Setup phpMyAdmin in Raspberry PI 3.
- Server-side scripting (php script) for MySQL database in Raspberry Pi 3.
- Creating Android script to retrieve and parse JSON from MySQL database in Raspberry Pi 3 and display in Android Application.
- Contributed to write reports for projects and prepare presentation slides.

### **Sulav:**

- Connected raspberry pi to the ultrasonic sensor using a breadboard.
- Setup MySQL database on the raspberry pi.
- Wrote python script to manipulate the sensor data.
- Wrote python script to update the database based on sensor data and to perform a SQL query to find number of free parking spaces.
- Helped to write the report for the projects and prepare presentation slides.

### **Topa:**

- Creating layout of the ParkSmart App in Android Studio.
- Connected Google map to android device and pin the different location.
- Prepare the reports and presentation slides for the project.

## **10. Interviews Summary**

The people we interviewed were from different background and majors. 40% of people we interviewed didn't have a parking permit and used bus or walked to campus. Of the people we interviewed, all those who park on campus have received a citation. They also said they have been late to class because they couldn't find parking spot during peak hours. They agreed that parking on campus during football games is confusing. 20% of the students we interviewed said that the TTU parking website is helpful. Most of the them were not aware about pay and park stations. The students who used bus or walked to get to campus were aware about the bus routes and night bus services compared to those who drove to campus. All of them said they would like an app to know number of available parking spots ahead of time.

Other additional feature which users mentioned were showing which lots are closed for a current day/event. Another user mentioned knowing the availability and picture of a map. One also mentioned add safe ride tracking, like the one in uber.

## **Interview 1**

### **Major: Petroleum Engineering**

1. Do you have a TTU parking permit?

= No

2. What transportation do you use to get to TTU?

= I walk

3. Have you had a parking citation?

= Yes

4. Have you been late to a class because you couldn't find a parking spot?

= No

5. Have you ever been confused as to where to park around TTU campus, especially during football games?

= No

6. How helpful is the information in the TTU parking website?

= I haven't looked at their information

7. Do you know where Pay and Park stations are located on campus?

= Yes

8. Are you aware about bus routes/safe ride/night shuttle services?

= Yes

9. Would you like to get information about parking space availability ahead of time?

= Yes

10. Do you think it would be better if we have a TTU parking app that gives information about number of available parking spots in a parking lot, and gives direction to parking Pay and Park stations?

= Yes

11. Name the parking features you would like to have on a TTU parking app.

= Safe ride driver locations (like UBE

## **Interview 2**

### **Major: Psychology**

1. Do you have a TTU parking permit?

= Yes

2. What transportation do you use to get to TTU?

= Car/Bus

3. Have you had a parking citation?

= Yes

4. Have you been late to a class because you couldn't find a parking spot?

= Yes

5. Have you ever been confused as to where to park around TTU campus, especially during football games?

= No

6. How helpful is the information in the TTU parking website?

= Sometimes it is outdated, I got a ticket because the maps weren't up to date

7. Do you know where Pay and Park stations are located on campus?

= No

8. Are you aware about bus routes/safe ride/night shuttle services?

= Yes

9. Would you like to get information about parking space availability ahead of time?

= Yes

10. Do you think it would be better if we have a TTU parking app that gives information about number of available parking spots in a parking lot, and gives direction to parking Pay and Park stations?

= Yes

11. Name the parking features you would like to have on a TTU parking app.

= Knowing availability and picture of a map

**Interview 3**  
**Major: Physics**

1. Do you have a TTU parking permit?  
= Yes
2. What transportation do you use to get to TTU?  
= Car
3. Have you had a parking citation?  
= Yes
4. Have you been late to a class because you couldn't find a parking spot?  
= Yes
5. Have you ever been confused as to where to park around TTU campus, especially during football games?  
= Yes
6. How helpful is the information in the TTU parking website?  
= Pretty helpful
7. Do you know where Pay and Park stations are located on campus?  
= No
8. Are you aware about bus routes/safe ride/night shuttle services?  
= No
9. Would you like to get information about parking space availability ahead of time?  
= Yes
10. Do you think it would be better if we have a TTU parking app that gives information about number of available parking spots in a parking lot, and gives direction to parking Pay and Park stations?  
= Yes
11. Name the parking features you would like to have on a TTU parking app.  
= Showing which lots are closed for a current day/event

## 11. References

1. Setting up MySQL server:  
<http://raspberrypiwebserver.com/sql-databases/using-mysql-on-a-raspberry-pi.html>
2. Connecting to MySQL via python:  
<https://www.jeremymorgan.com/tutorials/python-tutorials/how-to-connect-to-mysql-with-python/>
3. Connecting android application to raspberry pi:  
<http://www.instructables.com/id/Raspberry-Pi-Android-App-communication/>
4. Connecting ultrasonic sensor to raspberry pi:  
<https://www.modmypi.com/blog/hc-sr04-ultrasonic-range-sensor-on-the-raspberry-pi>
5. Devices and Figures:  
<https://www.google.com/>
6. Data Manipulation Concept for Ultrasonic Sensor:  
<https://circuitdigest.com/microcontroller-projects/arduino-ultrasonic-sensor-based-distance-measurement>