**Faculty of Engineering and Information Sciences**

**ECTE250 Deliverable Cover Sheet**

Team Name: Team - D

DELIVERABLE NUMBER AND TOPIC: 5 – Smart Parking System

Date Submitted: 29th May 2022

Team Members

1. Karthik Roshan Bharathraj – 7003821
2. Razveen Kashif – 7025014
3. Mohammad Safwan – 7056370
4. Mohammed Abuklal – 5902514

Team Member Submitting: Name: Karthik Roshan Bharathraj

ID No. 7003821 Signature:



Grade:

1. **Progress from previous deliverable**

Last deliverable’s (*deliverable 4*) main aim was to implement the circuit on breadboard. It was successfully implemented, and the desired output was achieved.

For this deliverable it was suggested by the instructor that:

The sensor circuit of the project must be implemented on the perfoboard.

Implementation of I2C module.

IOT must be implemented through means of blynk app.

The sensor circuit has been successfully implemented on the perfoboard and the desired output has been achieved. Since the LCD screen was working without any issues, I2C module was not implemented. Blynk app could not be implemented due to the few errors in the code implementation.

Below sections will explain the two targets that were set and how much of it is completed.

1. **Perfoboard**

With the breadboard prototype circuit constructed successfully from the last deliverable, the main outcome of this deliverable was to transfer the circuit onto a perfoboard, which is then attached to a prototype model. The model illustrated below is the prototype model constructed:

A picture containing text

Description automatically generated

Figure 1: Parking Model

The above model will be further modified to fulfill the proper functioning of the circuit, and to present a proper idea of how the system will work, by placing the relevant labels on places of the parking area, such as the entrance/exit.

However, to explain the model itself, borders are placed throughout the model, to replicate how an actual parking area looks as.

Upon further inspection, it is visible that a hole is cut, which will be used for the LDR to detect whether the car is at the entrance or not. Additionally, an entrance area is placed to allow the car to enter where the motor servo is positioned after it, to allow the car to enter. It is to note that the red block placed, is used to elevate the motor servo so that when it rotates, it can be done without touching the base of the model.

The Arduino which will be used to power up and upload the code to the sensors, is positioned as such where it can be easily supplied with a voltage source. For simplicity, the LCD is connected via a mini breadboard (also positioned at the entrance) to the Arduino UNO, which is used to display whether the slots are empty/full.

Since the model cannot function without the sensors, they had to be soldered onto a perfoboard and positioned at different areas of the model, such as the ultrasonic sensor and RGB LED placed the edge of the model, and as mentioned previously the LDR will be underneath the model. However, for the current deliverable the sensors weren’t positioned since the wires connected to the soldered components were delicate, thus for the purpose of the demonstration they were left untouched.

The soldered components, are as seen below:

**LED**

A picture containing red

Description automatically generated

Figure 2: LED: Top view

A picture containing indoor, wall, red

Description automatically generated

Figure 3: LED: Front view

**LDR**

A picture containing indoor

Description automatically generated

Figure 4: LDR: Top view

A close-up of a circuit board

Description automatically generated with low confidence

Figure 5: LDR: Back view

**Ultrasonic Sensor**

A picture containing text

Description automatically generated

Figure 6: Ultrasonic sensor: Top view

A close-up of a circuit board

Description automatically generated with medium confidence

Figure 7: Ultrasonic sensor: Back view

Additionally, the code used for the current deliverable was slightly modified, to not rotate the motor servo when the parking is full, thus not allowing any cars to enter the parking area. The modified code is as shown below:

A picture containing text

Description automatically generated

Figure 8: Modified Code

1. **BLYNK App**

The idea of using blynk app on our design is to display whatever on the lcd to the new version of blynk app, and this can be done in steps. It is important to follow the steps, as if one of these steps is not done, few errors and difficulties will appear.

* 1. Blynk Layout

For the project, the use of blynk app is just to display the LCD reading not to control things, and hence only the lcd screen have been used in the blynk layout. After that, the LCD pin has been set to V1 and the data type to string, from the advanced LCD settings on the blynk app.

**Graphical user interface, application

Description automatically generated**

Figure 9: Blynk Layout

* 1. Blynk Code

For the code, once the blynk layout is created, the app will generate a **'TEMPLETE\_ID'** and an **'AUTH\_TOKEN'** and these two are important to verify the layout that has been created and to be able to run the blynk code without errors.

The blynk code can be created from **'examples section > Blynk > Board\_USB\_Serial > Arduino\_Serial\_USB**' in arduino app. After that, the code will ask for the **'TEMPLETE\_ID'** and an **'AUTH\_TOKEN**'.

The LCD pin from the blynk app needs to be defined by writing **"widget LCD (V1)"**, as well as using the **'lcd.clear'** and **'lcd.print'** to clear and print anything on the LCD. In addition, one last thing needs to be written in the code to be able to change the blynk statues to online and send the reading from the lcd screen to the blynk layout which is **'Blynk.run()'**.

**Graphical user interface, text, application, email

Description automatically generated**

Figure 10: Blynk Code (1)

**Graphical user interface, text, application, email

Description automatically generated**

Figure 11: Blynk Code (2)

* 1. Output

After uploading the blynk code to the Arduino circuit, the written message from 'lcd.print()' line will be displayed on the LCD screen. The blynk app can run online by running the file inside the script file in the blynk library, and by doing this the reading from the lcd will be displayed on the blynk app.

**Graphical user interface

Description automatically generated**

Figure 12: Output on LCD

**Graphical user interface

Description automatically generated**

Figure 13: Output on Blynk app

The main code of the project has been combined with the blynk code to be able to make the blynk app display the message **"Slot 1: empty"** when the LCD displays the parking is empty, and **"Slot 1: full**" when the LCD displays the parking is full.

Unfortunately, the combined code is showing few errors for now, as a meeting with the instructor will be arranged soon to solve these errors and use the combined code for the project circuit with blynk app.