

# Microcontrollers

## BTEC: Set Assessment



**Written by:**

**Mohammed Ghalib Al-Hosni | 1615 | 7A**

### **Assessment aim**

The following assessment discusses and goes through the process of creating a client-based electrical system with the use of microcontrollers. This is done by utilizing prior knowledge on microcontroller programming and circuitry in order to create and enhance a client's lawnmower system while maintaining operator safety. As such, the assessment continues a full rundown of the technical specifications and client requirements from the system and while doing so also including possible enhancements to the original brief, while also including a testing plan for the stated requirements. With the assessment concluding with the microcontroller components, circuitry and programming. Additionally, a logbook stating everything that was worked on in each sessions is provided as a way of showing the thought process throughout the system's production process.

## **Contents table**

<b>1) Task planning and system design changes .....</b>	<b>2</b>
Time distribution and Gantt chart .....	2
Logbook.....	2
<b>2) Analysis of the brief .....</b>	<b>2</b>
Technical specifications .....	2
General technical specifications .....	2
Unexpected circumstances .....	3
Test planning.....	4
<b>3) System design.....</b>	<b>5</b>
Input and output components.....	5
Microcontroller Yourdon diagram .....	5
Component justification .....	5
Minor components .....	7
System circuitry.....	7
Circuit layout .....	7
Component connections.....	8
Program structure/Pseudocode .....	9
<b>4) System assembly and programming .....</b>	<b>12</b>
System code.....	12
Assembled circuit.....	20
<b>5) System testing and results analysis .....</b>	<b>21</b>
<b>6) Logbook .....</b>	<b>22</b>

## Task planning and system design changes

### Task 1.1

At the start of the task, create a short project time plan/Gantt chart and use it to monitor your progress throughout the rest of the task and make any adjustments as required.

### Time distribution

Activity #	Weeks																			
	Week 1				Week 2				Week 3				Week 4				Week 5			
Activity 1	N/A																			
Activity 2																				
Activity 3																				
Activity 4																				
Activity 5																				
Activity 6																				

### Task 1.2

During the other activities (2 to 5), you should also record in the Activity 1 section of your electronic task booklet: what you did in the session, details of any issues encountered in this session and solutions discovered, and action points for the next session.

**The stated task is placed at the end of the booklet. Said placement has been chosen in order to decrease clutter and be able to organize the booklet more efficiently.**

## Analysis of the brief

### Task 2.1

By interpreting the client brief into operational requirements, prepare a technical specification for a user-friendly system that can handle some unexpected events.



### General technical specifications

The overall functionality and utility of the system is to be able to prevent possible incidents when working with a lawnmower in terms of the rotating cutter as well as building a user-friendly system that is going to help the lawnmower operator in achieving their goal in an efficient and safe manner.

The methodology that is going to be utilized in achieving the stated task is to turn off the cutter whenever possible danger to the vehicle or operator is detected, that is the case since if something were to occur while the cutter maintains operation, a major risk and hazard would be placed upon the operator as vehicle or system failure could possibly lead to operator coming in contact with the blade therefore resulting in serious harm.

Said dangers could arise from a variety of different circumstances – some of which being rather unexpected – and as such the system should be able to handle a wide array of possibilities simultaneously without failing to ensure the safety of the operator. The hazards that are going to be worked on and maintained are going to be discussed in a subsequent section.

In addition to producing a safe system, it should also be user-friendly as assist the lawnmower operator by announcing changes in the system and warning them whenever possible danger could occur

## Unexpected circumstances to be countered

### The lawnmower tilting over

An issue that could be somewhat overlooked is that due to a lawnmower being a slow vehicle, the potential for it flipping over is high especially if it were operating on an angle even if said angle was not relatively steep. However, the danger that arises from the stated circumstance is not simply that the user/operator is going to fall over and injure themselves, but the major issue and risk that is upon the operator is the potential of the vehicle flipping in a way that makes it so the blade faces the operators, and as such if the cutter were to still be operating, major – and potentially life threatening – injuring are threatened.

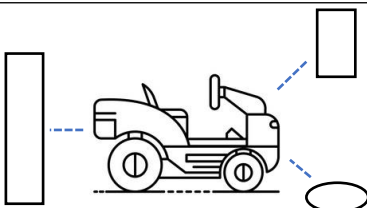
There are two different methods that can be used simultaneously in order to ensure that the severity of the stated issue is liquified. Said methods are the addition of a warning that lets the operator know whenever they have exceeded a certain driving angle, additionally, the motor operating the cutter would be programmed to cease operation if the stated driving angle were to be higher than what is deemed sage for a given period.



### Potential vehicle collisions

Due to the general working environment of a lawnmower, trees and potentially pitfalls would be in the general vicinity of the vehicle when operating for the majority of the time. Therefore, a factor that would have to be taken into consideration is the collision of the operator with objects such as low-hanging tree branches or the vehicle getting stuck within a pit or an unexpected hole in the ground. If the operator were to not notice said path interruption, a collision would occur and in severe cases, it could potentially lead to major injuries.

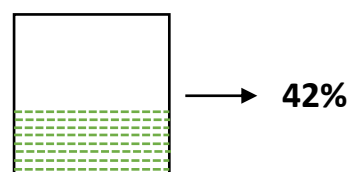
Since the stated issue is mostly dependent on human factor, no change in the system would be able to ensure the entire safety of the user from the hazard, for the exception of entirely turning off the vehicle which will be impractical and the sensor mechanism utilized could run into a variety of issues, therefore, the method that is going to be used is simply alarming the operator with the use of a buzzard as to let know of potential hazards in front of the vehicle. However, in the case the pit in front of the vehicle were to be very deep, the cutter will be stopped to ensure they blade does not break since there is a higher chance of it doing so if it hits the ground while it is rotating.



### Collection bag becoming full

The bag that is used for collecting that grass that is being lawned becoming full is not a relatively severe issue, however, it is one that has to be dealt with nonetheless, in that if the bag were to be full and the operator does not realize so, continuing operation could lead to several mechanical failures, an example of which could be the tearing of the collection bag, which in consequence lead to financial setbacks due to the repairs that would have to be made for the repairs of the vehicle

The main method of dealing with the issue is to use a sensor that will be used to receive information about the percentage of the bag. Using the stated method, the filled percentage could be displayed onto a screen for the operator, and as such the operator would be able to easily know whenever the collection bag would have to be emptied, this would also assist in the making the overall system more user friendly. Moreover, the blade could be made so that they stop operating whenever the bag is full as to ensure the stated problem does not occur in the case that the user does not realize or see that the bag is full from the provided information on the screen.



## Task 2.2

Prepare a test plan to check the functionality of the final solution against the technical specification and include some unexpected events.

Component testing		
Test name	Purpose of test	Expected result
Motor's functionality	To check if the motor is working properly	For the motor to work at 7000 RPM on maximum power
LCD's functionality	To check the LCD screen's brightness and that it is working properly	For the LCD to have a brightness that is more than enough to be clearly visible at night
Ultrasonic sensor's functionality	To check if the ultrasonic sensor is working properly and at appropriate distance	For the ultrasonic sensors to work appropriately and for their wave to return successfully at 150cm
Tilt sensor's functionality	To check if the tilt sensor is working properly and at an angle of approximately 30 degrees	For the tilt sensor to change signals at 28 to 32 degrees
Buzzer functionality	To check if the buzzer is working and emits if it a high enough volume to overwhelm the motor's sound	For the buzzer to emit a volume at approximately ...decibels

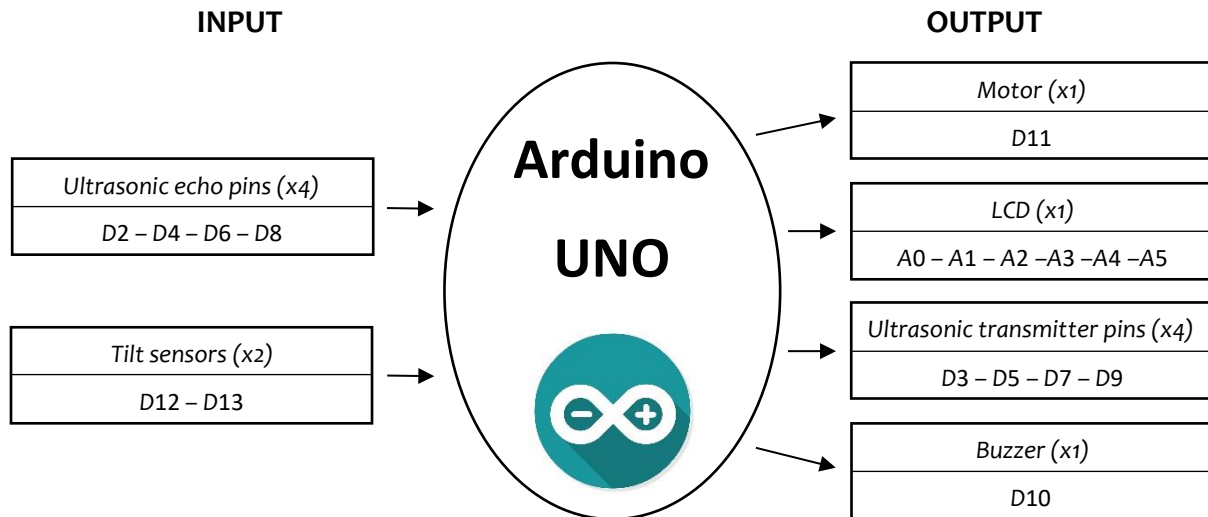
Logical testing		
Test name	Purpose of test	Expected result
Motor turnoff	Make sure that whenever a major hazard is detected that the motor would turn off	For the motor to turn off even if multiple actions are occurring simultaneously through the system.
Tilt warning and motor turnoff	Ensure that the motor is going to turn off whenever the vehicle has been at over 30 degrees for over 3 seconds	For the timer to remain counting 3 seconds no matter what the current iteration and to consistently turning off the motor if said issue is detected
Collision detection	Test if the ultrasonic sensors detect and relay information to the user	For a buzzer to warn the user with possible collision whenever the system deems the distance between the hazard and the vehicle dangerous
Accurate percentage display	Make sure that the user is always informed of the percentage that the collection bag is currently at.	For the system to provide the user with an easy-to-read percentage on the LCD screen of how full the collection bag currently is
Deep pit motor turnoff	To make sure that the ultrasonic responsible for checking pits is able to turn off the motor	For the motor to turn off whenever the stated ultrasonic sensor detects a hole in the ground that is over or is 50 centimetres deep
Full collection bag	Ensure that the motor is turned off whenever the collection bag is full	For the system to turn off the motor whenever the collection bag ultrasonic sensor turns detects that the bag is 98% full
Non-stuttering LCD	Utilize a method for clearing and adding to the LCD that makes it so the screen does not stutter	For the screen to remain stable and easily readable no matter the number of operations currently being ran that utilize the LCD screen, which would ensure a more user-friendly system

## System design

### Task 3.1

Selection and justification of suitable input and output devices.

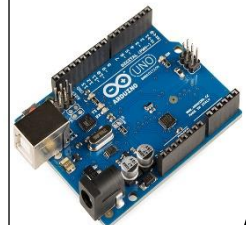
#### Microcontroller Yourdon diagram



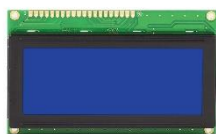
#### Component justification

##### Arduino UNO

The Arduino UNO is the most crucial component of the entire system as it is going to be the microcontroller of choice. The reason said microcontroller was chosen is due to it providing a relatively easy user experience in both software and hardware sides, with it possessing 20 pins, 14 of which being digital, while the rest being analog, which would be adequate amount for controlling all the required components. The microcontroller utilizes a special programming language known as the “Arduino” language which is used to control the stated pins. Arduino is based on the C programming language, a language that I have some prior experience in, which aided in choosing the Arduino UNO as the microcontroller to be used for the system.



##### 16x4 LCD

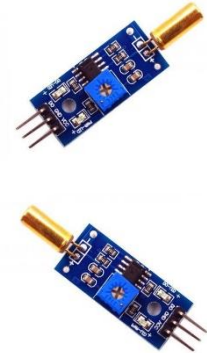


The LCD would be placed as the main source of information for operator, it will be placed on the dashboard of the vehicle as to face the operator and make it easier for them to read while driving the vehicle. The information that is going to be displayed on the LCD is going to be the following: One the first row, whether or not the cutter is on will be displayed. The second line will contain the collection bag's filling percentage. And finally, the third line will show if the operator has exceeded the maximum safe operating gradient. Using this, it would allow for an enhanced user experience as it would provide the necessary information that the operator might require without them having to manually check.

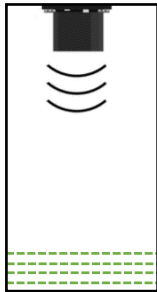


### Tilt sensor

A tilt sensor is a digital device that simply provides a HIGH or LOW signal depending on whether the sensor itself has been tilted a certain amount. However, as stated, the device is digital, meaning it is unable to provide specific information on the angle that it is currently at. However, on the component itself is a potentiometer that can be adjusted accordingly depending on how sensitive the component is required to be, in that by making it extremely sensitive, it would provide a HIGH signal at a relatively low angle, and vice versa if it were to be made insensitive. As such, initially said potentiometer would have to be adjusted to an angle of 30 degrees, which is what was specified earlier at the unexpected circumstances section. The amount of tilt sensors that would be utilized is two, one indicating the right side's gradient while the other indicates the left side. It should also be noted that there is the potential of external object accidentally turning sensor's potentiometer, and as such, to prevent the stated issue, a cover would be placed on the potentiometer as to block any objects from hitting it.



### Collection bag ultrasonic sensor



The stated issue of the collection bag being full would be solved with the use of a single ultrasonic sensor. The way that this is going to work is that the stated sensor would be placed at the top of the collection bag in an orientation that makes it so it is looking downwards towards the bottom of the bag, this way, the sensor will be able to detect the filling percentage of the bag as it would be able to detect the distance from the top of the bag where it is placed to the nearest surface below it, which would be either be the bottom of the bag or the grass within. After the distance has been calculated with the use of the sensor, it would be turned into a percentage, which will be the ratio of the current height and the total height of the bag, which will be given a hypothetical height of 150 centimeters. Thenceforth, the percentage would be relayed to the user as information via an LCD screen. Finally, the sensor itself would be covered to avoid the possibility of grass being stuck on the sensor's transmitter.

### Collision detection ultrasonic sensors

Three different ultrasonic sensors would be utilized for the prevention of unexpected collisions, two of which will be placed in front of the vehicle as to sense for potentially low-hanging branches or objects as well to sense for potential pits that the vehicle might fall into, both of which will be placed on top of each other and at an angle of around 45 degrees, with the one on top facing upwards and the one on the bottom facing downwards. Additionally, a third ultrasonic sensor would be placed behind the lawnmower at not major angle, which would be used sense for any objects that might be behind vehicle, as this would assist in sensing for objects as the vehicle is reversing.



### Buzzer



The buzzer – as stated earlier – is going to be used simultaneously with the 3 collision detection ultrasonic sensors, in that whenever said ultrasonic sensors detect an obstacle such as low-hanging tree branch or a pit in front of the lawnmower or whenever there is an object behind the lawnmower, the buzzer is going to emit a sound as to alert the vehicle operator. Additionally, an important point that should be taken into account is the placement of the buzzer and where it is going to be located on a real model of the lawnmower, in that the most appropriate location for it to be at would be within the dashboard as to ensure that the volume it emits would be heard by the operator even with the loud sound being caused by the cutter blade, since the buzzer would be used for two crucial tasks, one of which being emitting sound to alert the user that the collection bag is almost full, in addition to that the buzzer would also emit a noise whenever one of the collisions detection ultrasonic sensors have detected something at a distance which is deemed to be dangerous, which would be of major importance to be heard by the operator.

### DC motor

A single DC motor would be utilized in operating the grass cutter, the specific rate would have a maximum of approximately 3200 RPM, however, in order to avoid the potential malfunctioning and damages to the motor and increase its overall lifespan, around 75% of the maximum speed at around 2400 RPM would be utilized. In addition to the stated reasons, another useful advantage of decreasing the overall speed would be that if the motor were to necessitate being turned off, the shutting down process would be easier as less time would have to be taken in order to decrease the RPM to stopping point at 2400 RPM in comparison to if it were 3200 RPM.



### Minor components

#### Resistor

A resistor is going to be used for controlling the provided voltage going to certain components as to ensure the maximum voltage is not exceeded, with the ones that are going to require a resistor being the DC motor as well as the LCD, with the DC motor requiring it for its own operation while the LCD requires a resistor to control its brightness levels.



#### Transistor

A transistor – specifically of NPN type – is going to act as the power source for the DC motor, with the collector being connected to ground, the base to an analog pin, and the emitter to the Vcc pin of the DC motor itself. The configuration in which the transistor is going to be placed will make it so it simply acts as an electrical switch for the motor.



#### Diode

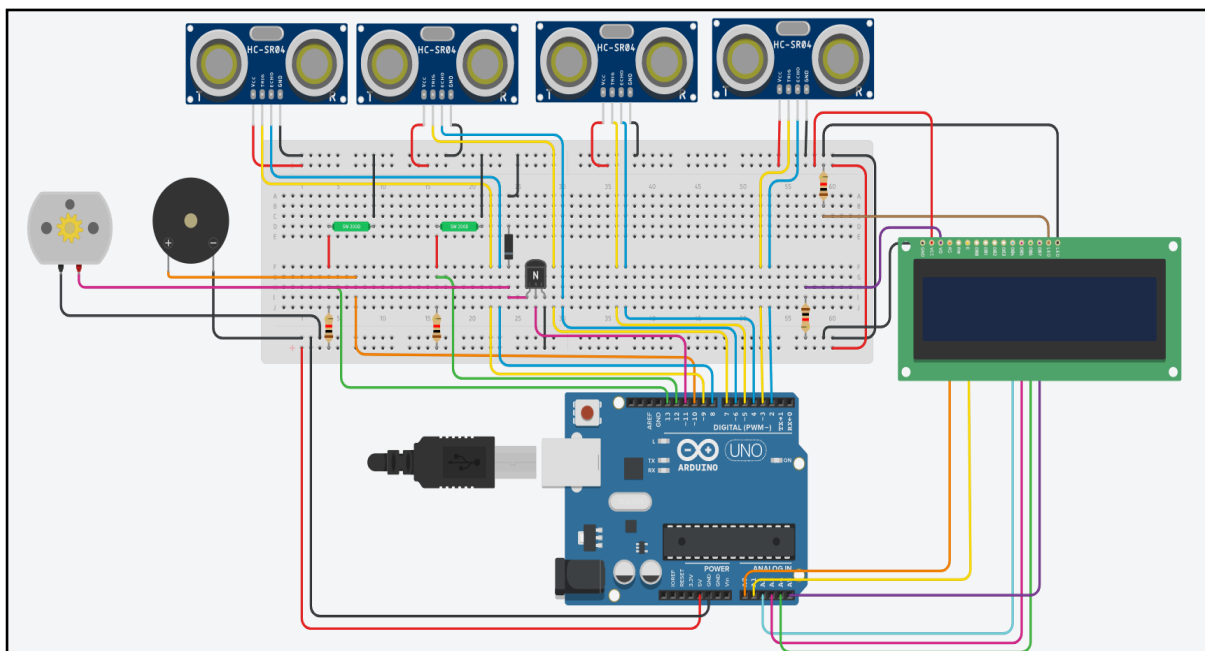
A single diode would be utilized and connected to the DC motor's ground pin in reverse bias. Which would consequently prevent the possibility of current flowing back to the Vcc pin of the DC motor since if it does so, potential errors with motor direction could occur which would cause the motor to rotate in the opposite direction to that required.



### Task 3.2

Prepare a description of the system's design covering input and output devices and microcontroller connections.

### Circuit layout





## Component connections

### LCD

When it comes to the LCD's connection to the Arduino, the first two, labeled as RS and E are connected to A0 and A1, with RS being the pin that lets the LCD know whether the data being received is to write something onto the board or to perform a command such as setting the cursor or clearing, while E represents the enable pin, a pin that determines whether the LED is going to perform that data being sent to it or not. In addition to the two stated pins, 4 more are connected directly to the microcontroller, with them being pins D4 to D7 in the LCD, said pins are responsible for receiving data from the Arduino board, with each one acting as one bit, and although the remaining Data pins (D0 to D3) are not utilized, they are used in order to increase the number of data that can be sent at once and therefore decreasing the operation complexity when the microcontroller is transmitting data to the LCD, however, due to the lack of space on the Arduino board, only four are used. In terms of the screen itself, it is mainly powered by pins V<sub>0</sub>, and the two LED pins, with the LED pins powering the screen by connecting to ground and 5 volts, while V<sub>0</sub> controls the screen's contrast, with the pin being connected to a resistor going to 5 volts. Finally, the remaining pins are simply the ones that power the LCD as a whole, which are simply the ground and Vcc pins.

### Ultrasonic sensors

The ultrasonic sensor generally consists of four pins, two of which are used to power the component and are simply the Vcc and Ground pins. In addition to those, a Transmitter and Echo pin can be seen, with the function of the transmitter pin being that it sends the sound wave, while the Echo pin is responsible for providing a signal to the microcontroller whenever it detects that the wave has returned, as such, the Transmitter pin is going to be placed in an output pin while the Echo is going to be placed in an input pin, each ultrasonic sensor's transmitter pin is connected to an odd number starting from 3 and going to 9 while the echo pins are connected to even numbers starting from 2 and going to 8, with all of them being digital pins.

### Buzzer

The buzzer's connections simply consist of one pin being connected to ground while the other is connected to 5 volts, with the controlling digital pin being placed in between the 5 volts wire and the positive pin of the buzzer, with said digital pin that is utilized for the buzzer being digital pin 10.

### Tilt sensor

The connection utilized for the tilt sensors are similar to that of the buzzer's, in that one pin is connected to ground while the other goes to 5 volts, with the microcontroller's pin being in between the 5 volt's wire and the component's pin. However, unlike the buzzer, the tilt sensor is non-polarized, meaning that depending on which side the 5 volts runs through, the component could either provide a HIGH or LOW signal by default. However, the connection seen in the provided circuit layout, the tilt sensor's is in the configuration where it provides LOW when under 30 degrees. In terms of the microcontroller pins that the sensors use up, the right tilt sensor is connected to pin digital pin 12, while the left one is connected to digital pin 13.

### DC motor

The DC motor is a nonpolarized component indicating that the direction of the motor's rotation is dependent on which pin the main power source passes through. Overall, the component possesses two pins, one being connected to ground, while the other being connected and controlled by the microcontroller. However, for optimal performance and to ensure the proper amount of power is going to the motor, the microcontroller's signal initially has to go through a resistor and following that an NPN transistor, meaning the signal itself initially passes through the resistor, after which it would go through the base pin, while collector goes to ground, and the emitter pin acts as the output of the transistor. From there, the emitter sends the signal to the motor, but before doing so it passes a diode in reverse bias which is connected to ground, although this might seem useless since no signal actually passes through the diode, it assists in ensuring no signal travels from ground and back to the motor's ground pin, since if it were to do so, the motor's rotating direction would change and would consequently cause major issues to both the system as well as the operator's safety. From there, the signal simply reaches the motor and powers it. The pin utilized for the motor is digital pin 11, a pin that is also able to emit a PWM signal, which permits the control of the DC motor's speed, and as stated earlier, it is more appropriate for it to be kept at 75% rather than consistently operating at maximum power.

### Task 3.3

Create a plan for the program structure detailing key system operations.



#### Flowchart or Pseudocode

*In terms of overall visual appeal and ease of use from the user's end, a flowchart is the rather more preferable as is far easier to comprehend, however, that is the case when the overall system is small and can concluded in a relatively small quantity of operations as a flowchart would make the code easier to visualize, however, if the code were to be considerably large, which is the case in code to be written, Pseudocode would be far superior as it would provide information in sufficient detail for the developer to be able to follow step by step, rather than visualize the code at once and write it on their own, which would be appropriate for larger systems. As such, **Pseudocode would be utilized.***

1. Start program
2. Import LiquidCrystal library
3. Create a LiquidCrystal class variable and define it as "LCD"
4. Declare constant variable int CBEcho = 2
5. Declare constant variable int CBTrans = 3
6. Declare constant variable int ForwardUpEcho = 4
7. Declare constant variable int ForwardUpTrans = 5
8. Declare constant variable int ForwardDownEcho = 6
9. Declare constant variable int ForwardDownTrans = 7
10. Declare constant variable int BackEcho = 8
11. Declare constant variable int BackTrans = 9
12. Declare constant variable int Buzzer = 10
13. Declare constant variable int Motor = 11
14. Declare constant variable int TiltRight = 12
15. Declare constant variable int TiltLeft = 13
16. Declare bool MotorState, BuzzerState, CBool, PitBol, TiltBool, TiltTimerState all as true
17. Declare int ThreeSeconds = 3000
18. Make a method for ultrasonic sensors called "GetDistance", takes int TransPin and int EchoPin as values and returns int
  - 18.1 Activate the transmitter pin to HIGH
  - 18.2 Delay for 10 microseconds
  - 18.3 Deactivate the transmitter pin to LOW
  - 18.4 Declare int Duration = the duration that the EchoPin is HIGH
  - 18.5 Declare int Distance = Duration multiply by 0.034 all divided by 2
  - 18.6 Return the variable Distance

19. Create a method called "Display LCD" that takes int PinNum and int DistanceOrState which returns nothing
  - 19.1 If PinNum = Motor pin's value
    - 19.1.1 Set the LCD to the first row
    - 19.1.2 If DistanceOrState = 1
      - 19.1.2.1 Display "Motor: On" on the LCD
    - 19.1.3 If DistanceOrState is not equal to 1
      - 19.1.3.1 Display "Motor: Off" on the LCD
  - 19.2 If PinNum = Collection bag's transmitter pin
    - 19.2.1 Set the cursor to the LCD's second row
    - 19.2.2 Display "Bag: %"
    - 19.2.3 Display DistanceOrState one whitespace away from Bag:
    - 19.2.4 If DistanceOrState is a two-digit number
      - 19.2.4.1 Increase the number of whitespaces between "Bag:" and "%" by one
  - 19.3 If PinNum = Hanging objects detection ultrasonic sensor's transmitter pin number
    - 19.3.1 Set the LCD's cursor to the third row
    - 19.3.2 Display "Object above " on the LCD
  - 19.4 If PinNum = Pit detection ultrasonic sensor's transmitter pin number
    - 19.4.1 Set the LCD's cursor to the third row
    - 19.4.1 Display "Pit ahead " on the LCD
  - 19.5 If PinNum = Reverse collision detection ultrasonic sensor's transmitter pin
    - 19.4.1 Set the LCD's cursor to the third row
    - 19.4.1 Display "Object behind" on the LCD
  - 19.6 If PinNum = the left TiltSensor's pin number
    - 19.6.1 Set the LCD's cursor to the fourth row
    - 19.6.2 Display "Steep gradient" on the LCD
20. Declare int CBPercent = (100 – GetDistance)/1.5, send CBTrans and CBEcho as parameters
21. Declare int ForwardUpDistance = GetDistance, send ForwardUpTrans and ForwardUpEcho as parameters
22. Declare int ForwardDownDistance = GetDistance, send ForwardDownTrans and ForwardUpEcho as parameters
22. Declare int BackDistance = GetDistance, send BackTrans and BackEcho as parameters
23. Declare int TiltRightState = value of the right tilt sensor
24. Delcare int TiltLeftState = value of the left tilt sensor
25. Call DisplayLCD and send CBTrans and CBPercent as values
26. If MotorState is true
  - 26.1 Turn on the DC motor at 75% power

- 26.2 Call DisplayLCD and send Motor and 1 as parameters
- 27. If MotorState is false
  - 27.1 Turn off the DC motor
  - 27.2 Call DisplayLCD and send Motor and 0 as parameters
- 28. If CBPercent bigger than or equal to 98
  - 28.1 Make CBbool false
  - 28.2 If BuzzerState is true
    - 28.2.1 turn on the buzzer
    - 28.2.2 Delay one second
    - 28.2.3 Turn off the buzzer
    - 28.2.4 Make BuzzerState false
- 29. If CBPercent is smaller than 98
  - 29.1 Make CBbool true
- 30. If ForwardDownDistance is bigger than 20 or ForwardUpDisatance is smaller than 50 or BackDistance is smaller than 15
  - 30.1. Turn on the buzzer
  - 30.2 If ForwardDownDistance is smaller than 20
    - 30.2.1 Call DisplayLCD and send ForwardDownTrans and 0 as parameters
    - 30.2.2 If ForwardDownDistance is bigger than 50
      - 30.2.2.1 Make PitBool false
    - 30.2.3 IF ForwardDownDistance is smaller than 50
      - 30.2.3.1 Make PitBool true
  - 30.3 If ForwardUpDistance is smaller than 50
    - 30.3.1 Call DisplayLCD and send ForwardUpTrans and 0 as parameters
  - 30.4 If BackDistance is smaller than 15
    - 30.4.1 Call DisplayLCD and send BackTrand and 0 as parameters
- 31. If ForwardDownDistance is smaller than 20 or ForwardUpDisatance is bigger than 50 or BackDistance is bigger than 15
  - 31.1 Turn off the buzzer
  - 31.2 make PitBool false
  - 31.3 Clear everything from the second row of the LCD
- 32. If either tilt sensor is over 30 degrees
  - 32.1 Call DisplayLCD and send TiltLeft and 0 as parameters
  - 32.2 If TiltTimerState is true
    - 32.2.1 Increase ThreeSeconds by the time the MCU has been running for
    - 32.2.2 Make TiltTimerState false
  - 32.3 If the current time of the MCU is bigger than ThreeSeconds

32.3.1 Make TiltBool false

33. If neither tilt sensor is over 30 degrees

33.1 Make TiltBool true

33.2 Make TiltTimerState true

33.3 Make ThreeSeconds equal to 3000

33.4 Clear the fourth row of the LCD

34. If BuzzerState is true and CBPercent is smaller than 98

34.1 Make BuzzerState true

35. If any of CBbool, PitBool, or TiltBool is false

35.1 Make MotorState false

36. If all of CBool, PitBool, and TiltBool are true

36.1 Make MotorState true

37. Loop back to line 20 until power is cut off

38. End

## System assembly and programming

### Task 4.1 & 4.3

*Produce the software program and annotate the code while refining the system so that it operates as expected and can handle some unexpected circumstances*

//1 Global

//1.1 Importing required libraries

#include <LiquidCrystal.h>

LiquidCrystal LCD(A0, A1, A2, A3, A4, A5);

//1.2 Defining macros for pins - makes code easier to read

#define CBEcho 2

#define CBTrans 3

#define ForwardUpEcho 4

#define ForwardUpTrans 5

#define ForwardDownEcho 6

#define ForwardDownTrans 7

#define BackEcho 8

#define BackTrans 9

#define Buzzer 10

```

#define Motor 11
#define TiltRight 12
#define TiltLeft 13

//1.3 Adding required variables to be used
bool MotorState = true;
bool BuzzerState = true;
bool CBbool = true;
bool PitBool = true;
bool TiltBool = true;
bool TiltTimerState = true;
int ThreeSeconds = 3000;
//1 End of global

//2 Setup
void setup()
{
    //2.1 Defining input and output pins
    for(int i = 2; i < 10; i++) //For loop to set pins required for the ultrasonic sensors
    {
        if(i % 2 == 0) //If a variable modulo 2 gives results in 0, then the variable is an even number
        {
            pinMode(i, INPUT); //Even numbers used for the echo pins
        }
        else
        {
            pinMode(i, OUTPUT); //Odd numbers used for the transmitter pins
        }
    }

    //Extra pins required
    pinMode(Buzzer, OUTPUT);
    pinMode(Motor, OUTPUT);
    pinMode(TiltRight, INPUT);

```



```
pinMode(TiltLeft, INPUT);
```

```
//2.2 Defining the LCD screen's size and displaying LCD rows that would be on the entire time.
```

```
LCD.begin(16, 4);
```

```
DisplayLCD(Motor, 1);
```

```
DisplayLCD(CBTrans, 0);
```

```
}
```

```
//2 End of Setup
```

```
//3 Loop
```

```
void loop()
```

```
{
```

```
//3.1 Calling variables that would have to be read each loop
```

```
int CBPercent = 100 - GetDistance(CBTrans, CBEcho)/1.5; //Equation to get the collection bag's filling percentage
```

```
int ForwardUpDistance = GetDistance(ForwardUpTrans, ForwardUpEcho); //Ultrasonic sensor distance for hanging objects
```

```
int ForwardDownDistance = GetDistance(ForwardDownTrans, ForwardDownEcho); //Ultrasonic sensor distance for pits
```

```
int BackDistance = GetDistance(BackTrans, BackEcho); //Ultrasonic sensor distance for objects behind the vehicle
```

```
int TiltRightState = digitalRead(TiltRight); //Reading right tilt sensor's state
```

```
int TiltLeftState = digitalRead(TiltLeft); //Reading left tilt sensor's state
```

```
//3.2 Displaying the percentage each loop
```

```
DisplayLCD(CBTrans, CBPercent);
```

```
//3.3 Checking if the variable motor state is true or not
```

```
if(MotorState)
```

```
{
```

```
    analogWrite(Motor, 191); //Turn on the motor
```

```
    DisplayLCD(Motor, 1); //Display that the motor is on by sending one as the second parameter
```

```
}
```

```
else //If MotorState is false
```

```
{
```

```

analogWrite(Motor, 0); //Turn off the motor

DisplayLCD(Motor, 0); //Display that the motor is off by sending 0 as the second parameter
}

//3.4 Checking the Collection bag percentage
if(CBPercent >= 98) //If the collection bag is more than 98% full
{
    CBbool = false; //Make the boolean CBool true (To turn off the motor)
    if(BuzzerState) //If the boolean BuzzerState is true
    {
        digitalWrite(Buzzer, HIGH); //Turn on the buzzer for a second and a half
        delay(1000); //Since delay here is only 1 second, it should not majorily disrupt the subsequent code.
        digitalWrite(Buzzer, LOW);
        BuzzerState = false;
    }
}
else //If the collection bag's percentage is less than 98%
{
    CBbool = true; //To return CBool to false so it does not stay at true if the collection bag's percentage got to 98% once
}

//3.5 Check if any of the collision sensors are on
if(ForwardUpDistance <= 50 || ForwardDownDistance >= 20 || BackDistance <= 15)
{
    digitalWrite(Buzzer, HIGH); //Turn the buzzer on

    //3.5.1 Pit sensor
    if(ForwardDownDistance >= 20) // Check if the pit is less than or is 20 centimeters deep
    {
        DisplayLCD(ForwardDownTrans, 0); //Display a warning message for the pit
        if(ForwardDownDistance >= 50) //Nested if statement to check if the pit is more than 50 centimetres deep
        {
            PitBool = false; //Make the boolean PitBool true (to turn off the motor)

```

```

    }

    else //If the distance to the pit is less than 50 centimetres
    {
        PitBool = true; //Make sure the motor is on
    }
}

//3.5.2 Hanging object sensor
if(ForwardUpDistance <= 50) //Check if the hanging object is less than or is 50 centimeters away
{
    DisplayLCD(ForwardUpTrans, 0); //Display a warning message for the hanging object
}

//3.5.3 Reversing collision sensor
if(BackDistance <= 15) //Check if the object behind is less than 15 or is centimeters away
{
    DisplayLCD(BackTrans, 0);
}
}

else //If none of the collision sensors detect anything in the required margin
{
    digitalWrite(Buzzer, LOW); //Make sure the buzzer is off

    PitBool = true; //A second line where PitBool is return to false in case it does not earlier
    LCD.setCursor(0, 2); //Set the LCD cursor to the third row
    LCD.print("      "); //Clear the third row of any messages by adding whitespaces
}

//3.6 Check the tilt sensors
if(TiltLeftState == LOW || TiltRightState == LOW) //If either tilt sensor has tilted over 30 degrees
{
    DisplayLCD(TiltLeft, 0); //Display that the gradient is too steep

    if(TiltTimerState) //Check if the TiltTimerState boolean is true
    {

```

```

ThreeSeconds = ThreeSeconds + millis(); //Add the current running time of the arduino in milliseconds by 3000

TiltTimerState = false; //Change TiltTimerStaet to false to make sure that ThreeSeconds does not increase next iteration
}

if(millis() >= ThreeSeconds)

//Check if the current running time (a constantly increasing number) is bigger than or equal to the variable ThreeSeconds
(A constant variable since TiltTimerStaet has now become false), and as such, 3 seconds will be counted without disrupting
the subsequent lines of code

{
    TiltBool = false; //Make the boolean TiltBool true (To turn off the motor)
}
}

else //If neither tilt sensor has tilted over 30 degrees
{
    TiltBool = true; //Make sure to return TiltBool to false since it if not it will remain true
    TiltTimerState = true; //Make sure TiltTimerState is back to true so that it works the following iteration if it became false
    ThreeSeconds = 3000; //Return ThreeSeconds back to 3000 so that it does not remain 3000 + millis()the next iteration
    LCD.setCursor(0, 3); //Set the LCD cursor to the fourth row
    LCD.print("      "); //Clear the fourth row by adding whitespaces
}

//3.7 Return BuzzerState

if(!BuzzerState && CBPercent < 98) //If the collection bag went under 98% and BuzzerState was false
{
    BuzzerState = true; //BuzzerState returns to true so that it sounds again next time the collection bag reaches 98%
}

//3.8 Checking if all motor running requirements are met

if(!CBbool || !PitBool || !TiltBool) //If any of the major unexpected circumstances were alerted
{
    MotorState = false; //Change the boolean MotorState to false (To turn off the motor)
}

else //If no major hazard risking the user
{

```

```

    MotorState = true; //If neither of the state issues were detected, then keep the motor running
}
}

//3 End of loop


//4 Distance method

int GetDistance(int TransPin, int EchoPin) //Ask for two parameters, the Transmitter pin and Echo pin numbers for the
required ultrasonic sensor
{
    digitalWrite(TransPin, HIGH); //Turn on the sensor's transmitter pin to send a wave from the transmitter pin
    delayMicroseconds(10); //Keep the wave going for 10 microseconds
    digitalWrite(TransPin, LOW); //Turn off the sensor's transmitter

    int Duration = pulseIn(EchoPin, HIGH); //Time the duration Echo pin is at HIGH at and place it in variable "Duration"
    int Distance = (Duration * 0.034)/2;

    //Multiply the duration by the speed of sound in centimeters per microseconds to get the Distance (s = v/t), then divide the
    answer by two since the wave takes double the time to go to the detected object and back to the sensor. Place the
    calculated distance in an integer variable "Distance"

    return Distance; //Return the Distance whenever the method is called
}


//5 LCD method

void DisplayLCD(int PinNum, int DistanceOrState) //Ask for components pin to know what message to display
{
    switch(PinNum) //Check the PinNnum parameter
    {

        //5.1 Displaying if the motor is on or off
        case Motor: //If PinNum is equal to the Motor pin
            LCD.setCursor(0, 0); //Set the cursor to the first row
            if(DistanceOrState == 1) //If the second parameter was 1
            {
                LCD.print("Motor: On "); //Display that the Motor is on
            }
    }
}

```

```

else //If the second parameter is 0
{
    LCD.print("Motor: Off"); //Display that the Motor is off
}

break; //leave the case

//5.2 Displaying the bag's filling percentage
case CBTrans: //If PinNum is equal to the collection bag ultrasonic sensor's transmitter
{
    LCD.setCursor(0, 1); //Set the LCD cursor to the second row
    if(DistanceOrState <=9) //If the percentage was a one digit number
    {
        LCD.print("Bag: %"); //Put two whitespaces after "Bag:" (To ensure the percentage sign does not get overwritten)
    }
    else if(DistanceOrState >=10) //If the percentage was a two digit number
    {
        LCD.print("Bag: %"); //Put three whitespaces after "Bag:" (To ensure the percentage sign does not get overwritten)
    }
    else //If the percentage was a three digit number
    {
        LCD.print("Bag: %"); //Place four whitespaces after "Bag:" (To ensure the percentage sign does not get overwritten)
    }
    LCD.setCursor(5, 1);
    LCD.print(DistanceOrState);
}

break; //leave the case

//5.3 Displaying hanging object collision warning
case ForwardUpTrans: //If PinNum is equal to the hanging object sensor's transistor
{
    LCD.setCursor(0, 12);
    LCD.print("Object above "); //Display that there is an object above the vehicle
    break; //leave the case
}

```



```
//5.4 Displaying pit warning
```

```
case ForwardDownTrans: //If PinNum is equal to the pit sensor's transistor
```

```
LCD.setCursor(0, 2);
```

```
LCD.print("Pit ahead "); //Display that there is a pit ahead
```

```
break; //leave the case
```

```
//5.5 Displaying reversing collision warning
```

```
case BackTrans: //If PinNum is equal to the reverse sensor's transistor pin
```

```
LCD.setCursor(0, 2);
```

```
LCD.print("Object behind"); //Display that something is behind the vehicle on the LCD
```

```
break; //leave the case
```

```
//5.6 Display if the maximum safe operating gradient is exceeded
```

```
case TiltLeft: //If PinNum is equal to the left tilt sensor's pin (only one of the Tilt sensors is needed for the method)
```

```
LCD.setCursor(0, 3); //Set the LCD cursor to the fourth (final) row
```

```
LCD.print("Steep gradient!"); //Display that the gradient is steep
```

```
break; //leave the case
```

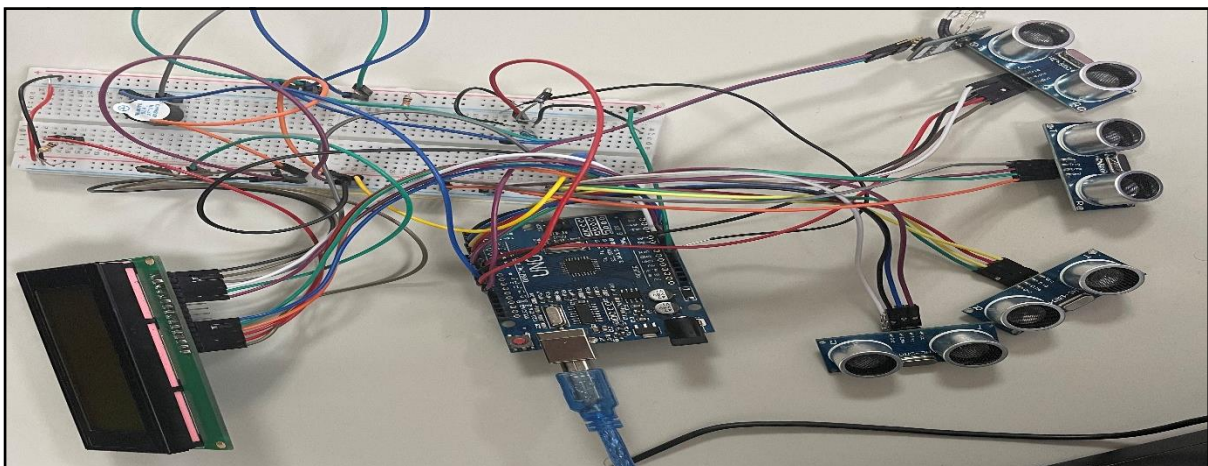
```
}
```

```
}
```

```
//5 End of LCD method
```

## Task 4.2


Assemble the hardware (if required)










## System testing and results analysis

### Task 5.1, 5.2, and 5.3

Test the system using the test plan from (Activity 2) and include some unexpected circumstances. Record the test plan in the template provided and analyze the test results and evaluate the system for conformance against the client brief.

Component testing			
Test name	Actual results	Test condition	Solution
All components	All the component provided worked as expected, with all output components working at optimal capacity in terms of the DC motor, LCD screen, and buzzer		N/A

Logical testing			
Test name	Actual results	Test condition	Solution
Motor turnoff and displaying	The motor turned off as required which was done by only tuning it on when all safety measures are met while also displaying if the motor is on or off.		N/A
Tilt warning and motor turnoff	Whenever the tilt sensor was over 30 degrees, it was able to alarm the user as required, but the methodology initially used to turn off the motor after three seconds by using the pulseIn() method did not work as expected		The fix was to use an if statement that increases a variable by the current run time using millis(), then fixing the variable at the calculated value, after which comparing the current run time with the variable, and turning off the motor if it was higher
Collision detection	The methodology utilized for detected collision worked as expected without any issues by alerting the user using a buzzer and also displaying the type of collision on the LCD screen.		N/A
Accurate percentage display	Displaying the percentage worked, however, using $100 - \text{Distance}/150 * 100$ , the value received was always 100 and not the actual percentage		The solution to the stated issue was rather simple, and was done by changing the expression to $\text{Distance}/1.5$
Deep pit motor turnoff	This testing process went as expected and was able to immediately turn off the motor whenever the pit ahead was deemed to be too deep (At 50 cm deep)		N/A
Full collection bag	Testing for whenever the collection bag was full went as expected, with the motor turning off as required when the collection bag reaches 98% capacity		N/A
Non-stuttering LCD	The initial method used for removing lines from the LCD screen was to clear the entire screen, which caused text to be removed and re-type again, and consequently causing a lot of stuttering		The way this issue was resolved was by removing unwanted text by replacing it with whitespaces rather than clearing the entire screen.

# Logbook

## Initial Task Plan

Initially, after the expected time distribution has been set, the plan would move on to the subsequent tasks, however, it should also be stated that the overall plan would not be done the order of the given activities as some of them would require work seen in a further task.

1. The first task that is going to be worked on is analysing the given client brief, after which, an overall brief of the system and its functionality is going to be written
2. After which, the possible unexpected circumstances that could occur during operation would be discussed as well as providing solutions for said circumstances while maintaining and prioritizing operator safety throughout the entire process.
3. After the possible circumstances and their solutions have been stated, analysis of the solution is going to be done in order to deduce the possible components that are going to be utilized, after which the chosen components and their functionality within the system is going to be discussed.
4. Draw the Yourdon diagram for the microcontroller connection to represent which components are going to be input and which are going to be output.
5. State the testing plan that is going to be utilized in activity, which is going to be split into two sections, component testing to make sure that all the components work as required and also logical testing, which makes sure that the system is working as required
6. Draw the circuit showing all the component and their connections using a CAD software
7. Talk about the connections and what each one represents in each of the given components
8. Write the system's code and annotate it, which will contain conditions for the tilt sensors going over 30 degrees, collection bag ultrasonic sensor which monitors how full the collection bag is, 3 ultrasonic sensors to alert the user in the case of possible collision. With all of which following the client brief and requirements.
9. Write the Pseudocode for the system, which will be done after writing the initial code as it would be easier to talk about and edit
10. Build the required circuit in real life
11. Test the code with the built circuit to make sure it is in accordance with the client's brief
12. Record a video talking about the built system and the methodology utilized for programming

Date:	30/ 3 / 2022	First session
<b>What have I done this session:</b>		
<p>The main tasks that were worked on during the initial session were simply about planning. The first thing that was done was working on setting the expected time and session usage for each activity/task, which was done by creating and filling a Gantt chart. In addition to that, general planning of the required tasks was written down in order to have a set plan for the following sessions.</p> <p>The stated tasks constitute of all that was required in activity one. After that, activity two was worked by initially stating the purpose of the system and the goals it is aiming to achieve while also writing a general brief on the technicality of the system.</p>		
<b>Issues encountered in this session and solutions with justification:</b>		
No issues were ran into as no technical work was started during the first session.		
<b>Action points for the next session:</b>		

1. Adding onto the technical specifications of the system, which will be done the following way:
  - 1.1 State the main issues that the program is going to solve while explaining how it going to do so
  - 1.2 Talk about potential unexpected circumstances and how the system will ensure they are dealt with
  - 1.3 Talk about the components that are going to be utilized and their individual roles in the system
2. Setting the testing plan that is going to be followed throughout activity 5

<b>Date:</b>	<b>4 / 4 / 2021</b>	<b>Second session</b>
<b>What have I done this session:</b>		
<p>As stated in the previous session's log, the main thing that was going to be done during the second sessions is talking about the potential unexpected circumstances as well as talking about how they were going to be dealt with, which was done during the session.</p> <p>However, talking about the components was also intended, although due to time circumstances that was not completed</p>		
<b>Issues encountered in this session and solutions with justification:</b>		
<p>Time issues led to the incompletion of a task that was intended to be done. The reason to which was underestimating the time it was going to take to talk about the unexpected circumstances</p>		
<b>Action points for the next session:</b>		
<ol style="list-style-type: none"> <li>1. During the third session, the only task that would be done is continuing activity 2</li> <li>2. Talk about the major components to be used. (DC motor, LCD, Tilt sensor, Buzzer, Ultrasonic sensors)</li> </ol>		

<b>Date:</b>	<b>6 / 4 / 2021</b>	<b>Third session</b>
<b>What have I done this session:</b>		
<p>During the third session, the main task that was tackled was simply working on activity 2, which was to simply continue working on explaining the technical specification of the system by specifying how each major component is going to be used and how it could be placed in a real-life application.</p>		
<b>Issues encountered in this session and solutions with justification:</b>		
<p>No issues were encountered during this session.</p>		
<b>Action points for the next session:</b>		
<ol style="list-style-type: none"> <li>1. Work on explaining the minor components of the system (Transistor, Diode, Resistor)</li> <li>2. Edit the layout of the booklet and organise it</li> </ol>		

<b>Date:</b>	<b>11 / 4 / 2021</b>	<b>Fourth session</b>
<b>What have I done this session:</b>		
<p>As stated earlier, explanation of the minor component were to be done during the fourth session while also organizing the booklet. However, said tasks did not take a long time to be done, and therefore work on the Yourdon diagram on task 3 was started while additionally working on circuit layout, which was not completed but almost is.</p>		
<b>Issues encountered in this session and solutions with justification:</b>		
<p>No issues were encountered during this session.</p>		
<b>Action points for the next session:</b>		
<ol style="list-style-type: none"> <li>1. Continue working on the circuit layout           <ol style="list-style-type: none"> <li>1.1 Add the remaining component on the circuit layout (Buzzer, DC motor)</li> </ol> </li> <li>2. Start working on the system code</li> </ol>		

<b>Date:</b>	<b>13/ 4/ 2021</b>	<b>Fifth session</b>
<b>What have I done this session:</b>		
The circuit layout was completed, and the code was worked by adding all the required methods, with them being the DisplayLCD and GetDistance methods. While also completing the If statements and conditions required for when the collection bag is at 98% and if the collisions sensors detected an object at a distance deemed to be dangerous.		
<b>Issues encountered in this session and solutions with justification:</b>		
The methodology initially utilized for the LCD method made it so the LCD stutters a lot whenever a one of the conditions were called, this was due to the LCD.clear() method occurring too often, this issue was fixed by making it so whenever it was required for a row to be cleared, what would happen is that whitespaces would be added in place of the words that were initially on said row.		
<b>Action points for the next session:</b>		
<ol style="list-style-type: none"> <li>1. Finish writing the code and annotate it <ol style="list-style-type: none"> <li>1.1 Work on the conditions if either tilt sensor were to be over 30 degrees.</li> <li>1.2 Refine the code and make sure everything works as expected</li> <li>1.3 Annotate the code</li> </ol> </li> </ol>		

<b>Date:</b>	<b>18/4 / 2021</b>	<b>Sixth session</b>
<b>What have I done this session:</b>		
Initially, the circuit connections and wiring was talked about, which did not take a long amount of time, as such, immediately work on the code was started, which involved working on the final two functions that were stated and also annotating the code. After this has been done, a bit of time was remaining and in it the Pseudocode was worked on.		
<b>Issues encountered in this session and solutions with justification:</b>		
The main issue that was encountered during coding was that the methodology initially utilized for timing the required 3 seconds whenever the tilt sensor is HIHG did not work efficiently, and therefore it had to be changed, which was done by utilizing if statement that function on a boolean, and in it the if statement an integer variable equal to 3000 is increased by the method millis(), from then, the Boolean becomes false so that the if statement is not true next iteration, from there the current running time is compared to the variable increased by millis(), and if it is higher, then the motor will be turned off.		
<b>Action points for the next session:</b>		
<ol style="list-style-type: none"> <li>1. Colour code the entire system's code</li> <li>2. Complete the Pseudocode</li> <li>3. Build the circuit</li> <li>4. Organize the booklet and add images</li> </ol>		

<b>Date:</b>	<b>20/ 4 / 2021</b>	<b>Seventh session</b>
<b>What have I done this session:</b>		
At first, the booklet was organized by simply adding the required layout for some of the pages, after which, the Pseudocode was written, which did not take a lot of time since it was already started the previous session. From there, images were also added to places where they were required. Finally, work on building the actual circuit was started but not finished.		
<b>Issues encountered in this session and solutions with justification:</b>		
No issues were encountered during this session.		
<b>Action points for the next session:</b>		
<ol style="list-style-type: none"> <li>1. Finish building the circuit</li> <li>2. Organize the booklet</li> </ol>		

<b>Date:</b>	<b>25 / 4 / 2021</b>	<b>Eighth session</b>
<b>What have I done this session:</b>		
At the start of the eighth session, the first thing that was done was adding minor finishing pages to the booklet such as the title page, contents table, and report aim. After which, work on the circuit was continued, during which, the quantity of jumpers wires was insufficient, meaning cutting and unsheathing wires had to be done, which took a majority of the time since the amount required was significant. Nonetheless, after that was done, every component of the circuit was placed, and the only thing that was remaining was recording the video		
<b>Issues encountered in this session and solutions with justification:</b>		
An issue that was ran into during this session was the fact that the quantity of available jumpers wires was insufficient, which meant that manual cutting and unsheathing of the wires had to be done, said issue was simply overcome by working as fast as possible in order to get the circuit done during the session		
<b>Action points for the next session:</b>		
1. Recording the system explanation video		

<b>Date:</b>	<b>28 / 4 / 2021</b>	<b>Nineth session</b>
<b>What have I done this session:</b>		
During the last session, no other work on the booklet or circuit had to be done as said tasks were entirely finished, as such, the only thing remaining was to record the video explaining the circuit. Said video took a couple of attempts to take perfectly due to noise related issues, however, other than that the video was completed and recorded as intended and in the required time range. From there, both the video and booklet were submitted to the subject supervisor		
<b>Issues encountered in this session and solutions with justification:</b>		
There were some noise related issues that disrupted the recording, nonetheless, said noise was simply avoided by switching the room in which recording was taking place		
<b>Action points for the next session:</b>		
N/A		

*“Hardware becomes a piece of culture that anyone can build upon, like a poem or a song.”*

**-Massimo Banzi**  
**Founder of the Arduino project**