TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1
Introduction

A bridge over a navigable waterway must allow boats and ships to cross its path, usually by being tall enough to allow them to sail underneath it. Sometimes itis impractical to build a bridge high enough; for example, it may rise too steeply or block the view of an important landmark. In such cases, the bridge can be designed so it can be easily moved out of the way for vessels that are too large to sail under it. The type of movable bridge that most people think of as a draw bridge is like those that spanned medieval castle moats [1].

Because a regular high bridge isn’t always an option, and that’s where engineers came to rescue by innovating Bascule bridges, a mobile bridge equipped with a counterweight that keeps the span, or leaf, balanced throughout its upward swing to allow room for boat traffic. It might have a single leaf or two.

However, bascule bridges aren’t perfect, and they still face some problems and defaults that needs to be enhanced and solved.

In this project we will address the problems and defaults in bascule bridges, define them, and try to solve or minimize them using automation to create a robust reliable system that doesn’t need much human interference so that we can reduce human errors as well.

1.1 Project Background

A lot of cities around the world enjoy the privilege of having one river or more passing through them, people actually have always looked for water sources specially rivers, and established cities on the banks of those beautiful rivers. The oldest and most famous civilizations in human history were built around rivers, like ancient Egyptian civilization, Iraqi, Chinese, and Indian civilizations were all the result of people gathering around rivers. Rivers have given people a second way of transporting through using boats to move from one place to another, however with the advancement of civilization, boats became bigger, and roads became heavier and more demanding in order to accommodate for cars and heavy vehicles and the old simple bridges that used to work are now blocking boat traffic in the river. Therefore, people came up with a solutions to keep the flow of both types of traffic and transportation The regular solution was to build higher and more robust bridges which was the implemented solution in most cases, but it wasn't a viable option in many places due to some problems with building high bridges. For example a high enough bridge that allows big boats and ships to pass also requires a long runway slope for cars and vehicles to ascend and descend at reasonable slope degree which requires a lot of space on the land and streets, in old touristic cities like Amsterdam, that means demolishing historic building and streets to make room for the bridge, especially if the river width is narrow and the water level is close to the land level. Some water canals connect the world trade routes for huge ships and oil tankers.

And that's why bascule bridges are important and still in use, but bascule bridges have their own issues as well, if we think about bascule bridges problem we find the following :

1. one obvious issue is safety. The risk of hitting and sinking a ship that is stuck under the bridge due to a technical problem or so, is very probable. Same story with land vehicles that has broken down on the bridge and can't clear the bridge while it starts to move.
2. One other issue is organizing the switch between Boat traffic and car traffic and determining priorities to prevent traffic jams and to keep a smooth flow

that saves time and costs. And to identify this problem probably we will need to collect more data from commuter to accurately reach the right solution.

In the following chapter we are going to draw a clearer image of the problems and describe them in detail to persue with analysis and full identification.

1.2 Description of the Problem.

To be able to truly identify the problem in hand we need to describe it precisely and analyse it to its sub-elements in detail.

Firstly, the description of the main situation here is that we have 2 perpendicular directions of traffic that are crossing each other, and we can't raise one of them to a higher plane, so we have to switch turns between them, and that problem was solved by using a bascule bridge.

Secondly, the sub-elements of the problem:

Using a bascule bridge has raised some issues that can be broken down to the following elements:

1- The moving bridges raised the risk of hitting stuck (broken-down) boats underneath it as well as the risk of moving up while there are still vehicles that haven't cleared the bridge.

2- How to organise the switching between car traffic and boat traffic and determining priorities to prevent traffic jams and to keep a smooth; keeping in mind that boats move slower than land vehicles usually, and to get a clear picture about this problem we are going to use a survey to collect data from commuters about the nature of traffic related to cross bridges.

3- What are the other issues that we don't know about? we also are going to ask commuters about other issues that they usually face.

We have used a survey to collect data from the commuters by reaching out to unimap students who have used a bascule bridge before and reaching out to Terengganu communities on facebook groups since they they have “Terengganu Drawbridge” and probably most of them have used it.

The Questions that we asked the commuters and the results:

1- 1) Have you used a moveable bridge before?

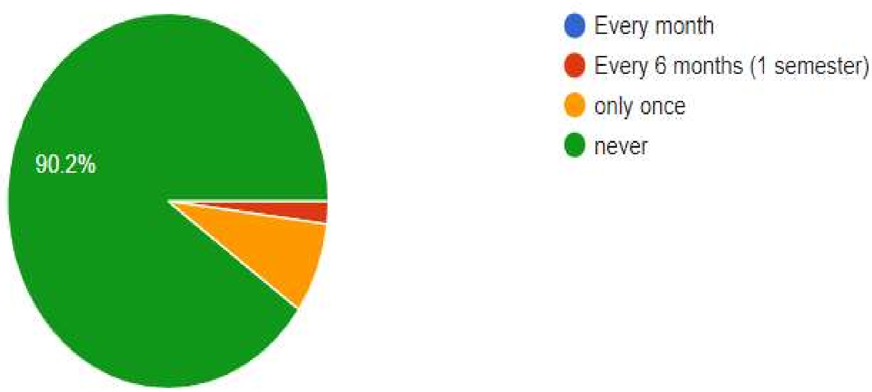

Figure 1.1 Have you used a moveable bridge before?

2- Do you have a general idea of how a moveable bridge work?

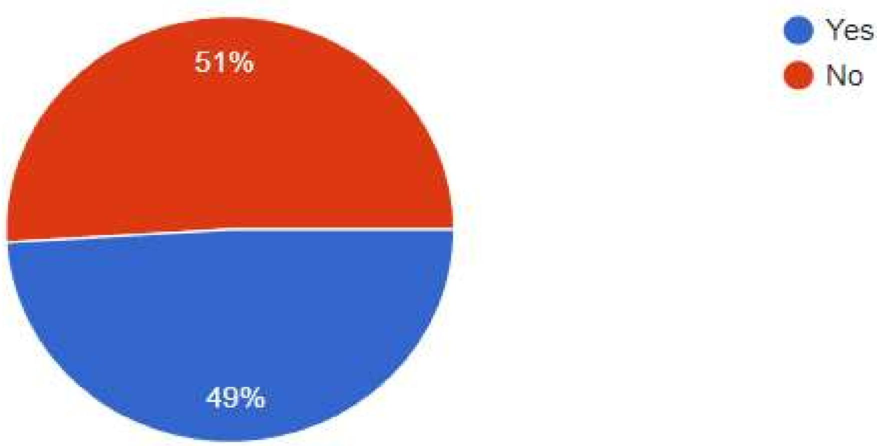

Figure 1.2 Do you have a general idea of how a moveable bridge work?

3- Based on your observations and opinions: which would have more traffic jam usually, boats or cars (land traffic)?

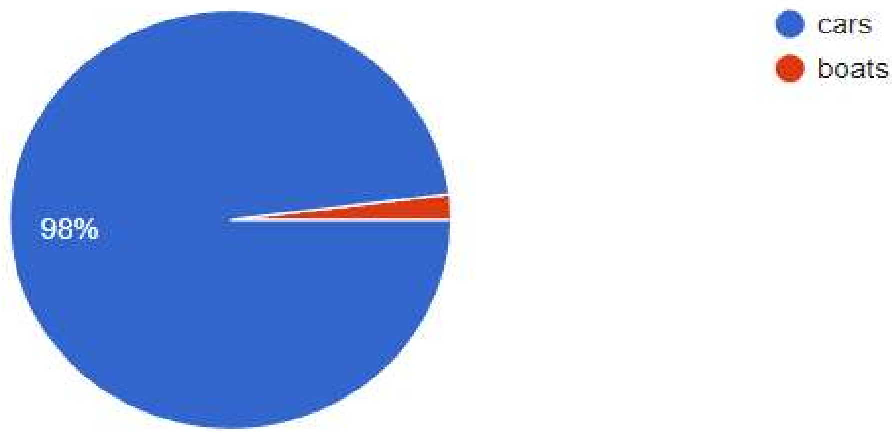

Figure 1.3 Based on your observations and opinions: which would have more traffic jam usually, boats or cars (land traffic)?

4- Have you ever been stuck on a movable bridge entrance waiting for it to be laid down or lifted up?

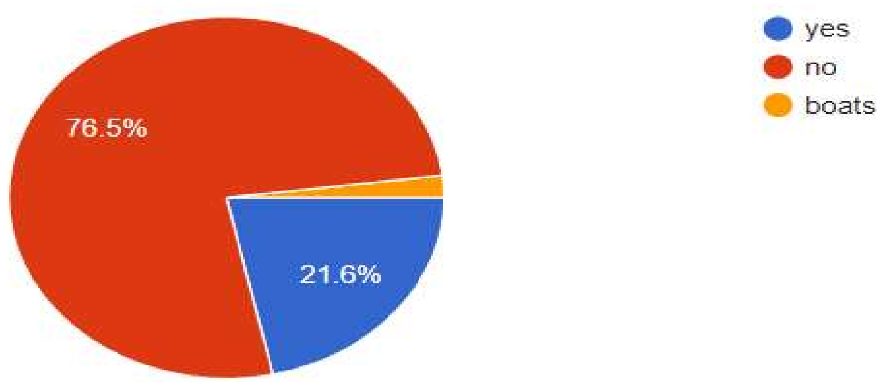

Figure 1.4 Have you ever been stuck on a movable bridge entrance waiting for it to be laid down or lifted up?

In the following section we are going to analyse the data that we have collected and fully identify the problems and address them.

1.2.1 Identification of the problem...

After describing and analysing the situation, and based on the data that we have collected from commuters, we conclude that commuters complain about traffic and that they have specified that car traffic usually have more traffic jams. In addition to the description of the safety problem, and since the the survey sample didn't report any other issues when they were asked about it in the open answer question.
we can narrow down the problems to two main categories:

1.2.1.1 Traffic Jam:

From the data we conclude that commuters complain about congestion and that they have specified that car traffic usually have more traffic jams. We are going to address this problem, offer some solutions, maybe allow longer time for the bridge to be laid down for cars to pass. We are going to ask the commuters which solution they think is more fitting and how to organize the switch between land traffic and water traffic as well. We will take their feed back into consideration for the solution.

1.2.1.2 Safety:

Based on the previous section (problem description), we can conclude that the safety concerns are very obvious, and the issues are clear and can be specified into the following two issues:

a. Safety of cars:

If a car breaks down on the bridge and can't clear it before it starts moving up, the damage will be severe also as it will start moving without control and will probably roll and crash or fall in the water.

b. Safety of ships:

If a ship gets stuck under the boat for any technical reason while the bridge is moving down, the damage to the ship will be severe and might lead to sinking it.

c. Human error:

There is also the problem of human errors that are presented in disrupting the synergy between managing the tolls and the movement of the bridge. For example: the car toll may get opened in the wrong time while the bridge is moving up or doesn't get closed in time before the start of the bridge movement. Same with the boat toll.

1.2.2 Proposed solution for the problem

In this section we will propose a solution that solves the problems stated in the previous sections. We will address every problem one by one and provide solutions for each one.

1.2.2.1 Traffic Congestion:

We offered two approaches to solve this problem and we took commuters opinions about both:

a) Use a fixed schedule with specific timing for cars to cross the bridge, and specific timing for boats as well, so that people would know what to expect and plan ahead.
b) Keeping the bridge open for land traffic all the time and pull it up only when there is a boat.

We asked the survey sample and 66.7% of them chose the first option.

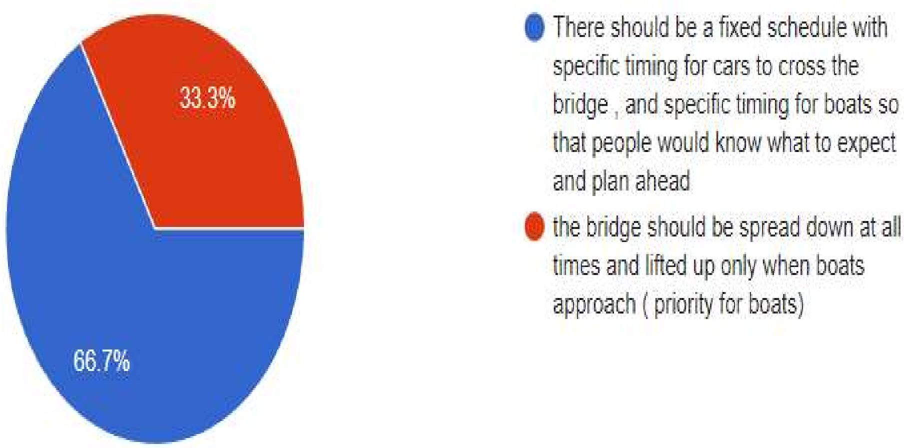

Figure 1.5 Proposed solution for the problem

So our solution is to design an automation system for the bridge to control the opening and closing of the bridge based on a fixed schedule with specific timing for cars to cross the bridge, and specific timing for boats so that people would know the schedule in advance and plan ahead. The initial passing time for cars will be double the time for boats to ease th traffic jam which will be around 20 mins and extra 10 minutes for clearing the route so that the bridge can be moved up or down safely. However, the system will be remotely connected to a wireless interface which will eneble the management to reset and modify the timings and the whole schedule based on needs.

1.2.2.2 Safety:

i. Safety of cars:

a) We are going to use an automated toll system and will integrate safety measures in it, by adding extra time for clearing the bridge before it is pulled up so the procedures will be like this:
The entrance and exit toll gates open → cars pass for 20 mins → entrance toll gate close → 10 more mins for cars to clear the bridge → exit toll close and bridge is pulled up

b) In addition we have added IR sensors at the entrance of the bridge and the exit so that we can count how many cars have entered and how many have exited. If the number of the cars that left is less that the number of the cars that entered the bridge, means that the bridge is not clear and it will not be pulled up until it is cleared and the all cars are counted at the exit IR.

ii. Safety of boats:

a) We are going to use the same concept that we have use for cars but instead of using a metal rode, we will use a safety net attached to a server motor from one of its corners to be pulled up above the water when the bridge is down and slacked away when the bridge is up so that the net will fall to the bottom of the river and boats can pass again.

b) In addition we have added one ultrasonic sensor below the bridge to sense if there is a boat under the bridge, the bridge won't be laid down until there is no object in the ultrasonic range, which is the is same range as the length of the bridge.

iii. Human errors:

Using this automated system will minimize human interference to minimum and will result in less human errors regarding the synergy managing the traffic, the tolls, and the movement of the bridge.

1.3 Project Objectives

The project is aimed to

a. To organize the switch between car traffic and boat traffic by developing an automated system that sets a fixed schedule for each route
b. To develop automated safety measures that can be implemented in any bascule bridge, for example : The Terengganu Drawbridge.
c. To simulate the whole system by building a prototype and testing it using 3d printing , esp32, IR sensors and ultrasonic sensors.

1.4 Project Scope

The main target of this project is to build a prototype of an automation system that can be implemented on any bascule bridge. The toll gates for cars can be easily implemented on any bridge. Same for the sensors, however we might need sensors with stronger capabilities and longer ranges to be implemented for longer and wider bridges. As for the safety nets that work as toll gates for boats, they can be implemented on small and short bridges only as the bigger bridges will require very big and heavy nets which will be impractical.

Chapter 2
Methodology

2.1 Introduction

This chapter provided a detailed overview of the technique used to complete this entire project. The report's introduction and other crucial details were taken from the previous chapter. In addition, this chapter will include a flowchart, a block diagram, the design and specification of the project, and, finally, a list of components and an explanation of each one that will be necessary for the project to be completed successfully.

2.2 Project Design

2.2.1 Block Diagram

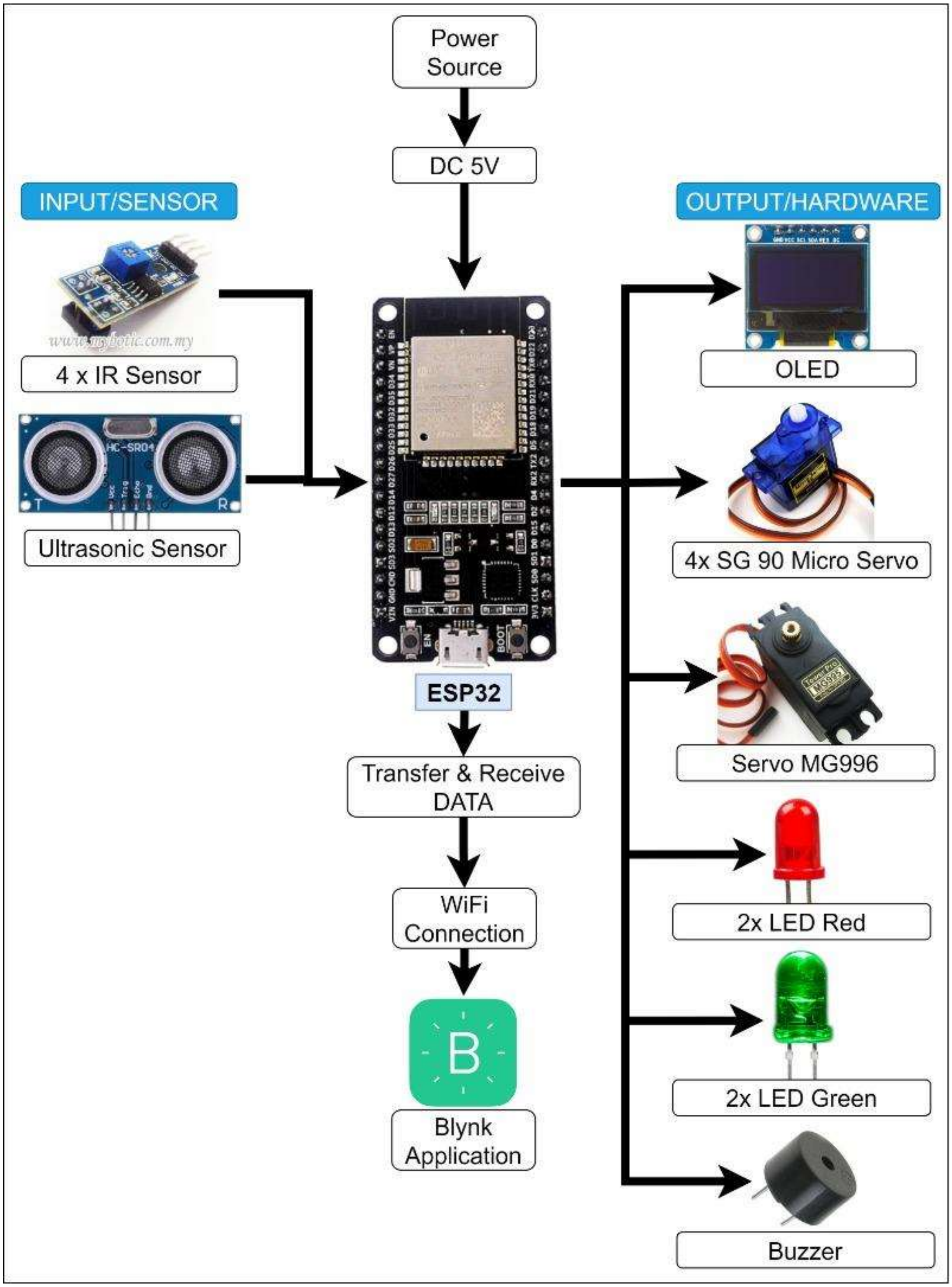

Figure 2.1 Block diagram of the overall project

From figure 2.1 which is block diagram for the project, ESP32 microcontroller will be used in this project. ESP32 microcontroller is microcontroller that built in WiFi module. . The data such as setting time in Blynk application, monitor the car along the bridge and all notification when the bridge is open or close, these all data need to transmit through WiFi connection, so that is why the ESP32 is used in this project. Based on the block diagram, the project used 5V of power source. Furthermore, this project able to operate with power source from power bank. Other than that, this project used 4 IR sensors in order to count the vehicle in and out of the bridge. The bridge will remain open for other vehicle users such as car, motorcycle, etc if there were still had vehicle on the bridge, so the IR sensor works for that purpose before it is open for ship to across the bridge. Next, this project also used ultrasonic sensor in order to the distance between ship and bridge. This bridge will not open to other vehicle users to use the bridge as long as the ship does not across the bridge, so this ultrasonic sensor will detect the presence of the ship when it is near to the bridge. Later, this project used OLED in order to display counted car across bridge and the distance of the ship when it is near with the bridge. SG 90 Micro Servo also used in the project and we used 4 of it. The purpose of using SG 90 Micro Servo is to open and close the toll gate as well as open and close the bridge. 2 of SG 90 Micro Servo will be used for open and close the toll gate and the remaining of 2 SG 90 Micro Servo will be used for the purpose to open and close the bridge. For the Servo MG996, it is used in order to pull up and down the net. For the LED, we used 2 red LED and 2 green LED. The purpose of using red LED is to give information for vehicle users to stop and not to enter the bridge. For the green LED, it is used to give information for vehicle users that the bridge is safely to across it. Last but not least, buzzer also used in our project. The buzzer operates in order to give warning for vehicle users in several situations such as when the toll gate is open or close, when the bridge is open or close for ship and more.

2.2.2 Flowchart

There are two flowcharts involved in this project. First is the progress flowchart and second is the project process flowchart.

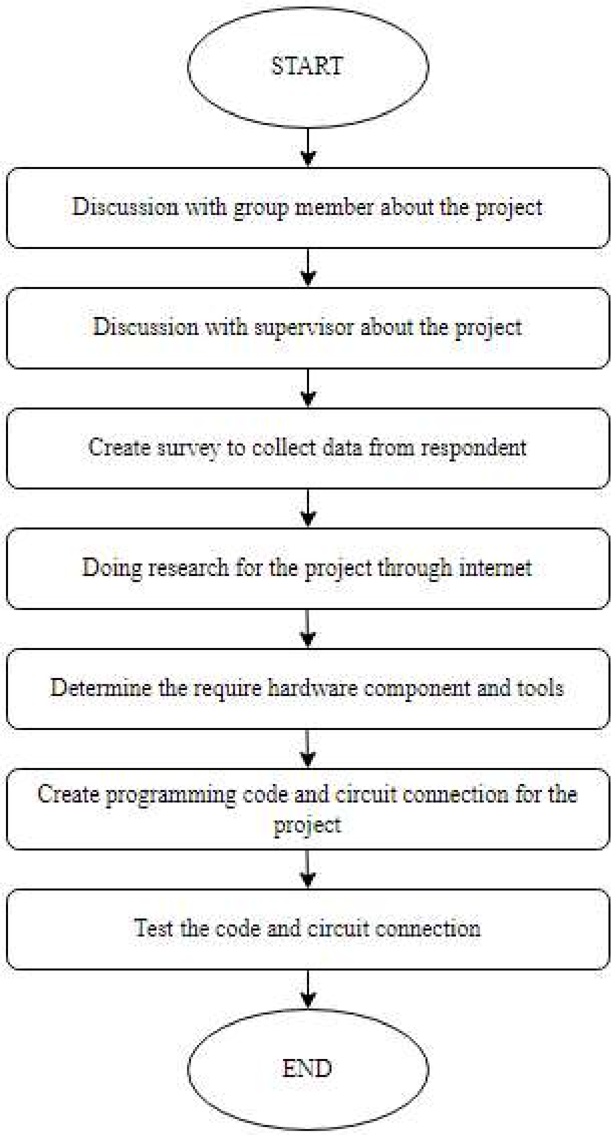

Figure 2.2 Progress Flowchart

The Figure 2.2 shows the progress flowchart of our project. Our project, which is an integrated design project, requires 14 weeks to accomplish (IDP). According to the flow chart above, we initially began having group conversations with a team member. The title is under discussion because it corresponds to the topic of our project, which is transportation. After we approach and validate our project title, we go to the next step, which is a project discussion with our supervisor, Dr. Ong Siok Lan. During our conversation with our supervisor, she provided numerous suggestions and ideas for making our effort a success. Following our supervisor's acceptance of our ideas, we conducted a poll using a Google Form to collect data from respondents. Additionally, we conduct internet research about our title. After that, hardware parts and tools have been chosen. Then, selecting the hardware and tools that will be employed in our project, Using the ESP32 software, we continue to write programming code and connect circuits on the breadboard. Last but not least, the system functions properly with the ESP32 software codes.

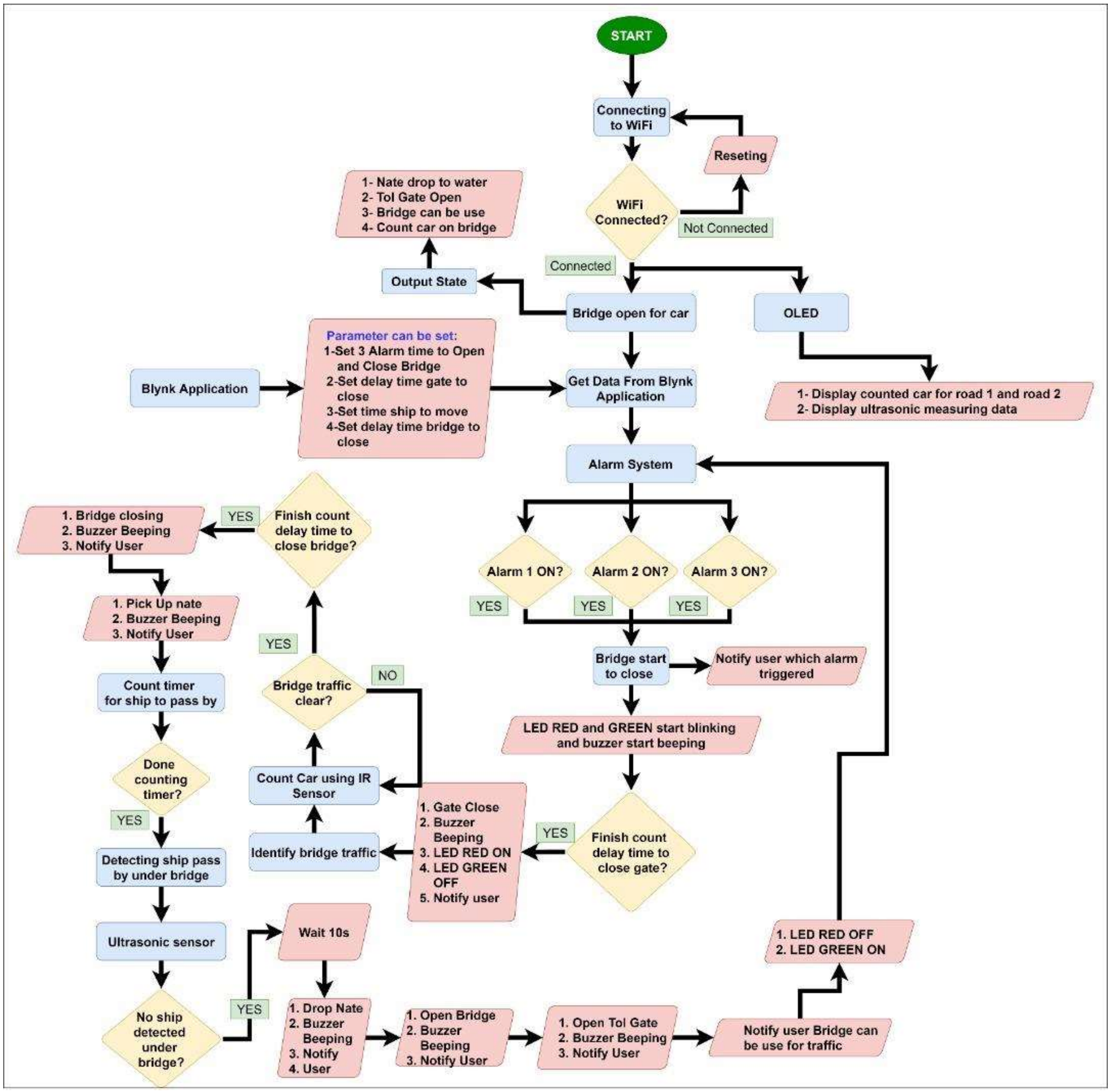

Figure 2.3 Process Flowchart

Based on the figure 2.3 above is the process flowchart for this project, it is started with connecting with WiFi. If the project loss connection internet or does not connect with WiFi connection, it will reset and start connecting again until it connected with internet connection (WiFi). After that, the bridge will open for car. When the bridge is open, automatically the output state of the project operate with the net drop to water, toll gate open, bridge can be used and count car on bridge. Then, OLED also will display the output where it display counted car for road 1 and road 2 as well as display ultrasonic measuring data where ultrasonic sensor will measure the distance between the bridge and ship. After the bridge open, the system will get data from

Blynk application where the data that we set through Blynk application which are 3 alarm time to open and close bridge, delay time gate to close, time ship to move and delay time bridge to close. There 4 parameters can be set through Blynk application. After the process of get data from Blynk application, it will go through to alarm system. In our project, we can set 3 alarm in Blynk application which are Alarm 1, Alarm 2 and Alarm 3. When one of the alarm that we set triggered, the bridge start to close and then LED red and green start blinking and buzzer start beeping. After that, the delay time to close gate will count until finish and automatically the gate will close followed by buzzer beeping. When the gate is close, LED red will be on while LED green will be off as well as it will notify users that they would not able to across the bridge. The IR sensor will operate in order to count the car and identify bridge traffic. When the bridge traffic is clear, the bridge will close with delay time that we set in Blynk application until it finish count. If the bridge traffic is not clear, the process will remain with IR sensor to count the car on the bridge. When the bridge is close, the buzzer will automatically beeping in order to notify users. Then, the Servo MG996 works in order to pick up net and the buzzer will automatically beeping. The process will go through with count timer for ship to pass by. If the counted timer was done, the ultrasonic sensor will detect ship passing by under bridge. If the ship is still had under bridge, the timer will remain count to pass by. If there is no ship detected under bridge, the next step will start within 10 seconds (delay time). The net will drop by the Servo MG996 and as usual the buzzer beeping to notify users. Next, the bridge will open again for traffic use. Before the traffic can use the bridge, the buzzer beeping first when the bridge open. The toll gate will be opening as well as the buzzer beeping to notify users that bridge can be used. After that, the vehicle will able to across the bridge when the LED green is on while LED red is off. The system will begin with alarm system and the process will repeat as usual.

2.2.3 Schematic Circuit Design

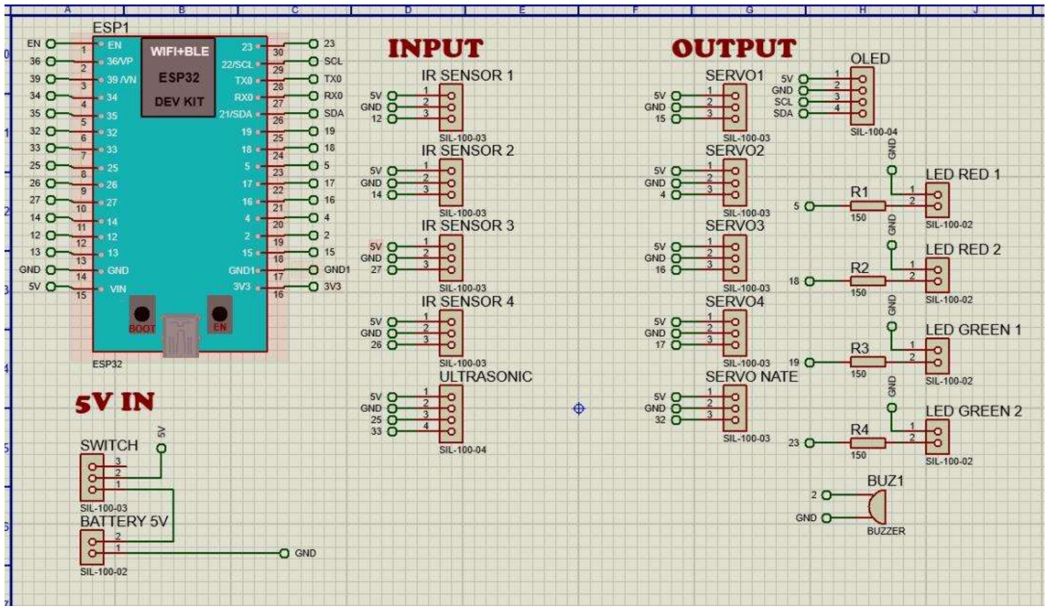

Figure 2.4 Schematic circuit

Figure 2.4 show schematic circuit of embedded system for our project automated bascule bridge system, for the connection connectivity can be refer to Table 2.1 and Table 2.2. This schematic circuit had been designed by using Proteus 8 professional software, Proteus 8 professional is a software can be used for designing circuit schematic, PCB layout, code and stimulate the schematic circuit (Proteus 8 Professional - a useful tool for electronic engineers).

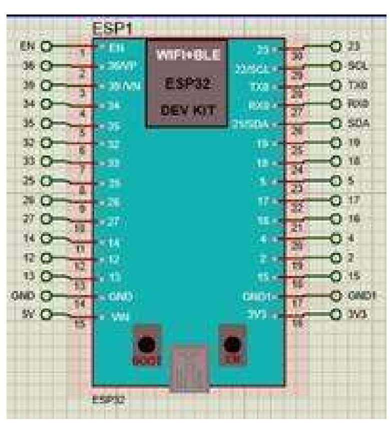

Figure 2.5 ESP32 GPIO Pins

Table 2.1 Embedded System Connection to ESP32 for Input/Sensor Device

NO.	INPUT	CONNECTION TO ESP32
1	IR SENSOR 1	5V
		GND
		12
2	IR SENSOR 2	5V
		GND
		14
3	IR SENSOR 3	5V
		GND
		27
4	IR SENSOR 4	5V
		GND
		26
5	ULTRASONIC SENSOR	5V
		GND
		25
		33

Table 2.2 Embedded System Connection to ESP32 for Output/Hardware

NO.	OUTPUT	CONNECTION TO ESP32
1	SERVO 01	5V
		GND
		15
2	SERVO 02	5V
		GND
		4
3	SERVO 03	5V
		GND
		16
4	SERVO 04	5V
		GND
		17
5	SERVO NET	5V
		GND
		32
6	OLED	5V
		GND
		SCL
		SDA
7	LED RED 1	GND
		5
8	LED RED 2	GND
		18
9	LED GREEN 1	GND
		19
10	LED GREEN 2	GND
		23
11	BUZZER	2
		GND

2.2.4 3-Dimensional Design

We created our prototype project based on the schematic diagram below. Next, we used Autodesk invertor software in order to create the schematic diagram. Before we proceed with the size of our prototype project, we already estimate the measurement accurately in order to put the components such as ESP32, IR sensor, ultrasonic sensor, servo SG60, LED and more to make sure our project works successfully.

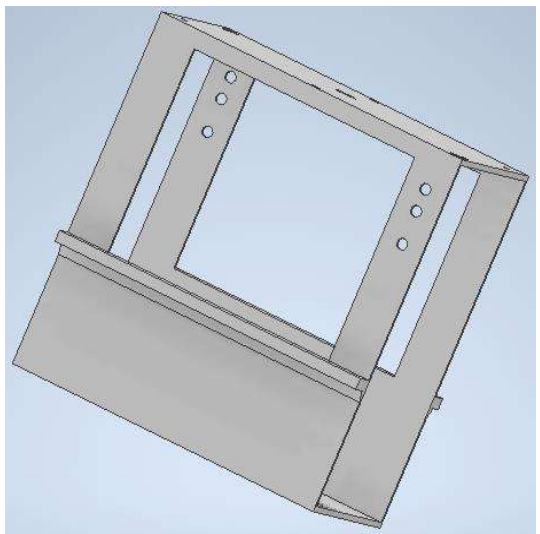

Figure 2.6 Prototype Design (front view)

Figure 2.7 Prototype Design (side view)

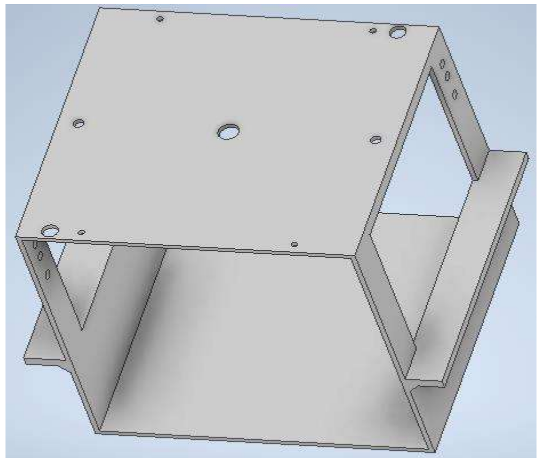

Figure 2.8 Prototype Design (on top view)

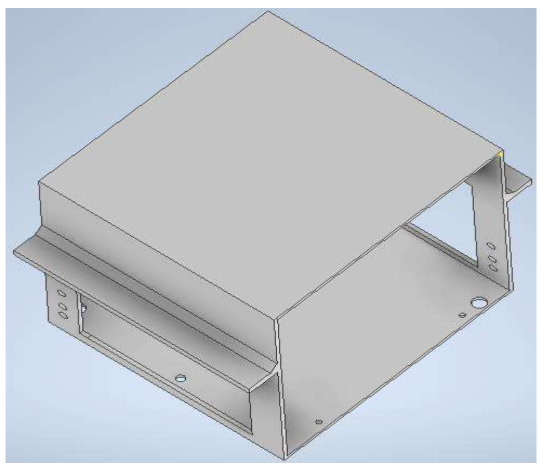

Figure 2.9 Prototype Design (bottom view)

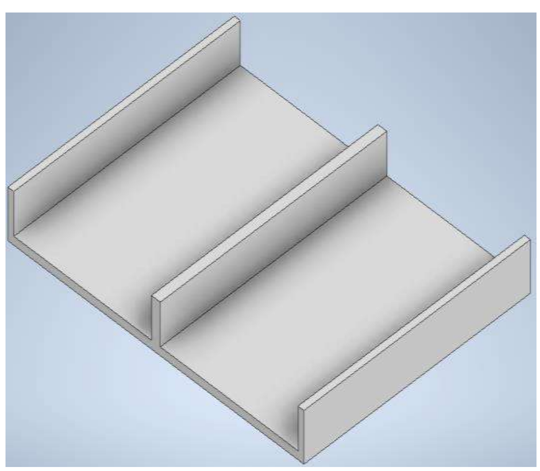

Figure 2.10 Bridge leave

Figure 2.11 Toll Gate

3D Printing

Ender 3D printer was utilised in the development of the 3D printing. Prototype Body, Bridge Leaves, and Toll Gates were all made using a 3D printer. We used plastic (PLA) material in the Ender 3D printer to b uild our prototype, which took 18 hours to print out completely.

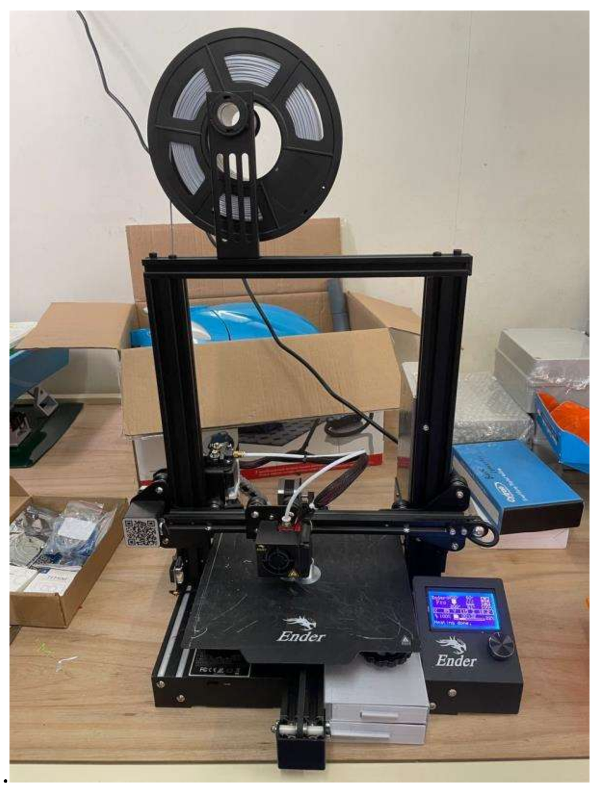

Figure 2.12 3D printer Ender

Figure 2.13 Prototype Body

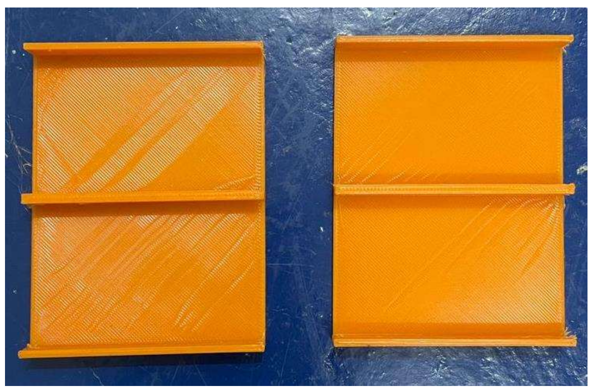

Figure 2.14 Bridge leaves

Figure 2.15 Toll gates

1. Component and Description

To build circuits suitable for the project, we use a few components. The components include an ESP32 Arduino, an IR sensor, two servos (SG60 and MG995R), a USB Micro B cable, an OLED, a buzzer, an LED, a male and female header, wire, and a PCB board.

2. ESP32

ESP32 is a line of compact, quick microcontroller boards similar to Arduino, but with built-in Bluetooth and Wi-Fi capabilities. A low-cost, low-power System on Chip is called ESP32 (SoC). It has dual-mode Bluetooth and integrated Wi-Fi. Its microprocessor is a Tensilica Xtensa LX6.

Figure 2.16 ESP32

3. Ultrasonic Sensor

An ultrasonic sensor is an electronic device that employs ultrasonic sound waves to detect the distance between two objects and converts the reflected sound into an electrical signal. The speed of audible sound is greater than the speed of ultrasonic waves (i.e. the sound that humans can hear). The transmitter (which generates sound using piezoelectric crystals) and the receiver are the two major parts of an ultrasonic sensor (which encounters the sound after it has travelled to and from the target).

Figure 2.17 Ultrasonic Sensor

4. IR Sensor

An electrical device that monitors and detects infrared radiation in its environment is called an infrared (IR) sensor. An infrared sensor (also known as an IR sensor) is an optoelectronic component that is radiation-sensitive and has a spectral sensitivity in the infrared wavelength range of 780 nm to 50 m. Motion detectors, which are used in building services to turn on lights or in alarm systems to detect unwanted visitors, increasingly frequently incorporate IR sensors.

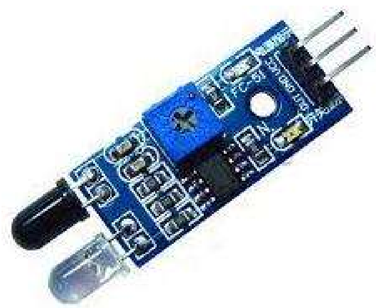

Figure 2.18 IR sensor

5. Servo SG60

High output power while being small and light. Smaller versions of the conventional types, servos may rotate about 180 degrees (90 in each direction). These servos can be controlled by any servo code, hardware, or library. Good for novices who don't want to create a motor controller with feedback and gear box because it can fit in compact spaces. Hardware and three horns (arms) are included.

Figure 2.19 Servo SG60

6. **Servo MG995R**

The servo motor MG995 is well-known for its effectiveness and affordability. The motor is utilised in a variety of applications, the most common of which include robotics and drones. According to the pin diagram, the MG995 has three terminals. First, Signal pin (Orange pin) is the PWM signal which states the axis position is given through this pin. Second, VCC (Red Pin. This pin receives a positive power source for the servo motor. The last is Ground (Brown pin). This pin is attached to the power supply's or the circuit's ground.

Figure 2.20 Servo MG995R

7. **USB Micro B cable**

The Universal Serial Bus (USB) is an industry standard that specifies cables, connections, and protocols for interfacing (connecting) computers, peripherals, and other computers, as well as for connection, communication, and power supply. To standardise the connection of peripherals to personal computers for communication and power supply, USB was developed. It is now widely used on a variety of devices and has largely superseded interfaces like serial ports and parallel ports. Keyboards and mouse for computers, video cameras, printers, portable media players, mobile (portable) digital telephones, disc drives, and network adapters are a few examples of peripherals that can be linked via USB. As cables for charging portable electronics, USB connectors are progressively taking the place of other varieties.

Figure 2.21 USB Micro B cable

8. OLED

A light-emitting diode (LED) with an organic compound film as the emissive electroluminescent layer generates light in response to an electric current is referred to as an organic light-emitting diode (OLED or organic LED), also known as an organic electroluminescent (organic EL) diode. This organic layer is sandwiched between two electrodes, at least one of which is transparent. OLEDs are utilised to make digital displays in gadgets like televisions, computer monitors, and portable gaming systems like smartphones.

Figure 2.22 OLED

9. Buzzer

A beeper or buzzer is an example of an auditory signalling device that can be either mechanical, electromechanical, or piezoelectric. The signal is converted from audio to sound as its primary function. It is often powered by DC voltage and used in timers, alarm clocks, printers, computers, and other electronic devices. It can produce a variety of sounds, including alarm, music, bell, and siren, according on the varied designs.

Figure 2.23 Buzzer

10. LED

In electronics, an LED, or complete light-emitting diode, is a semiconductor device that, when charged by an electric current, generates visible or infrared light. Visible LEDs are utilised as indication lamps in several electronic gadgets, as brake and rear-window lights in cars, and as alphanumeric displays or even full-color posters on billboards and signs. Autofocus cameras, television remote controls, and fiber-optic telecommunication systems all use infrared LEDs as light sources.

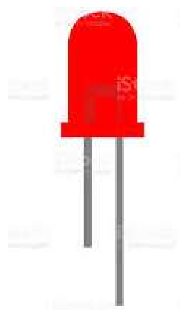

Figure 2.24 LED

11. Female Header

A female connector is one that is attached to a wire, cable, or piece of hardware and has one or more recessed holes with electrical terminals within. It is designed so that a male connector with exposed wires may be placed snuggly into it to create a secure electrical and physical connection. A jack, outlet, or receptacle are other names for female connectors. This sort of connector can be identified by the fact that the electrical conductors are not directly exposed when it is detached or removed; as a result, they are less likely to come into touch unintentionally with outside items or conductors.

Figure 2.25 Female Header

12. Male Header

A male pin header is made up of one or more rows of metal pins that are moulded into a plastic base and spaced often at 2.54 mm (0.1 in), though there are many other options available. Male pin headers are inexpensive because they are straightforward.

Figure 2.26 Male Header

13. Wire

A flexible metal strand is called wire. The most common way to make wire is to draw the metal through a hole in a draw plate or die. When expressed in terms of a gauge number, wire gauges are available in a variety of common sizes. Mechanical loads are carried by wires, frequently in the form of wire rope. A "wire" can refer to an electrical cable in the context of electricity and telecommunications signals. This type of cable may have a "solid core" made up of a single wire or multiple strands woven together in stranded or braided forms.

Figure 2.27 Wire

14. PCB Board

A printed circuit board (PCB) or printed wiring board (PWB) is a laminated sandwich structure made up of conducting and insulating layers. PCBs provide two complementing tasks. The initial step is to solder electronic components into certain positions on the outer layers. The second is the regulated provision of dependable electrical connections (and likewise dependable open circuits) between the component's terminals, which is also known as PCB design. Each of the conductive layers is created with an artistic pattern of conductors that creates electrical connections on that layer. These conductors resemble wires on a flat surface.

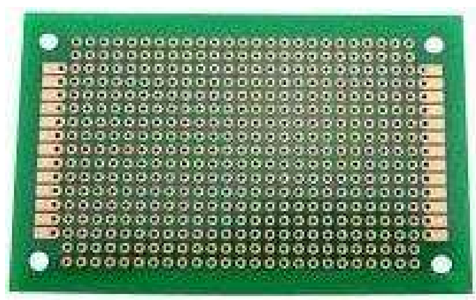

Figure 2.28 PCB Board

2.2.5 3D-prototype design

Based on the figure below, which depicts our project's 3D design, we used Autodesk invertor software to generate the 3D design. We design in 3D with the purpose of printing it out using a 3D printer.

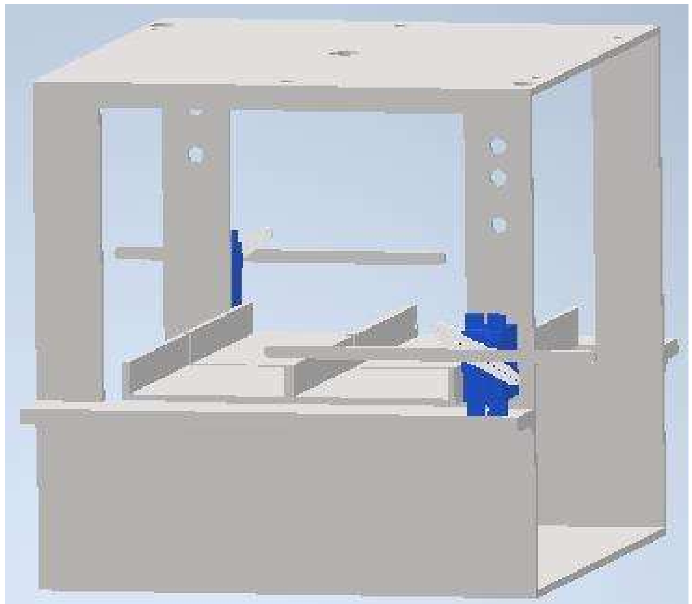

Figure 2.29 Front view

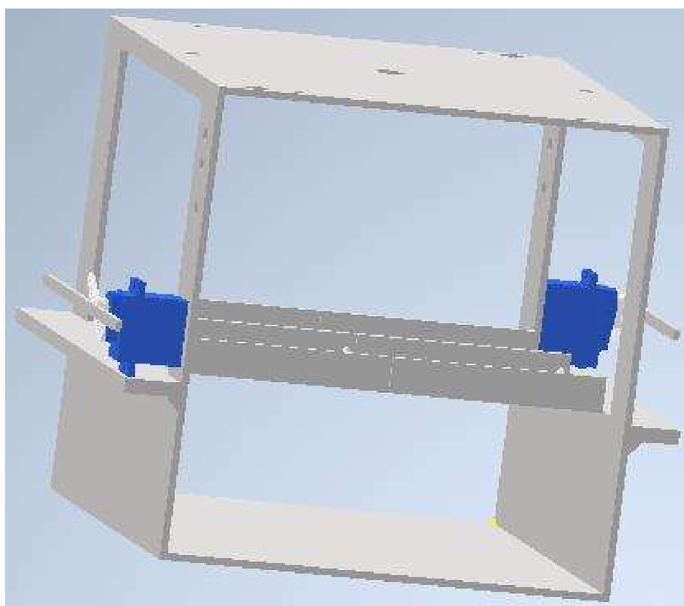

Figure 2.30 Side view

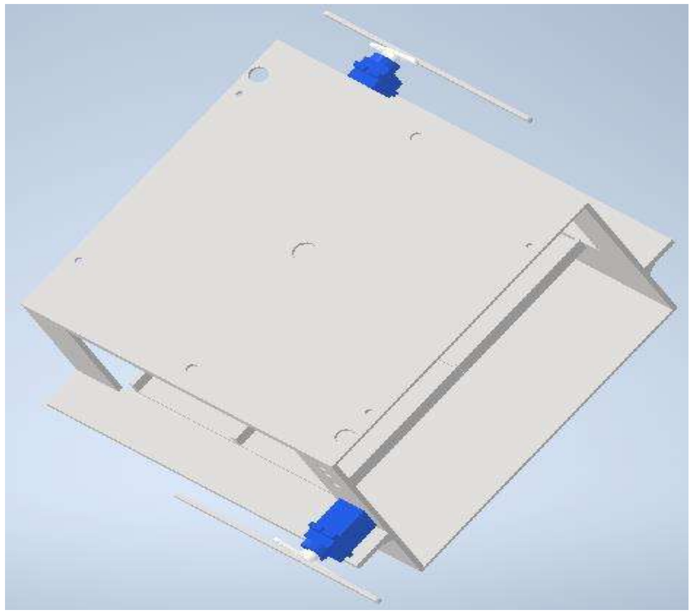

Figure 2.31 On top view

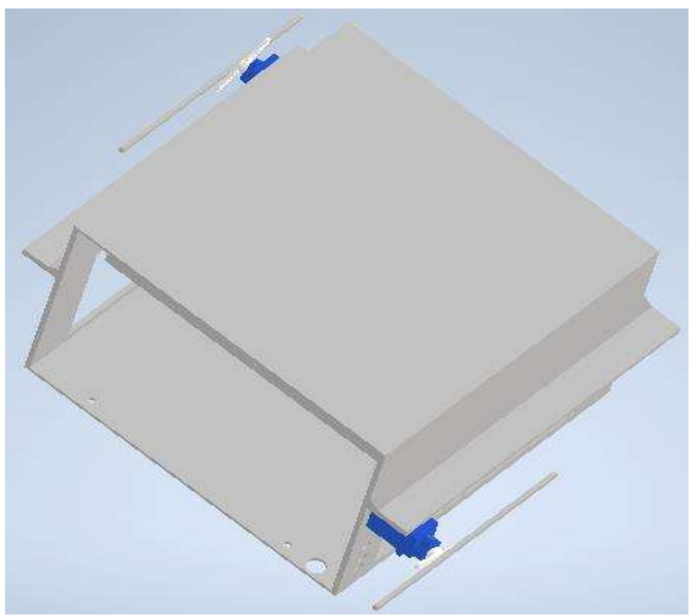

Figure 2.32 Bottom view

2.3 Conclusion

To summarise, we use a variety of components, including software and applications, to execute this project. The software is used to write and run the code required for this project, and the component is used to complete the circuit so that all of the key components may be used for the purpose of this project. In this project, the code is used to establish a connection with the Blynks programme, which is the application we use.

Chapter 3
Results and discussion

3.1 Introduction

This chapter will explain the construction of a result of our project prototype which is "Automatic Bascule Bridge System (moveable bridge system)". The prototype was put to the test to see if it worked as expected. To construct a circuit with ESP32, IR sensor, servo SG60, servo MG995R, Ultrasonic sensor, OLED display, buzzer and LED for signal was quite challenging. Along the way, there are several unforeseen challenges that created some good learning opportunities. This chapter will explain all about the results achieved and discuss the problems facing.

3.2 Sub-topics

In the experiments conducted, there are several steps to obtain the result. Firstly, it needs to design the schematic circuit, PCB layout design, PCB etching and then model testing.

3.2.1 PCB layout design

For the circuit board, firstly we design the schematic by using proteus software. Then, we will use the virtual circuit to design our PCB layout. For this part, we need to make sure there is no double layered for the blue lines to avoid error in etching later. If there has double layered, just double-clicking the left button on the mouse, then re-arrange it manually. After complete tracking the PCB layout, click the 3D visualization to see our final circuit, we can view all the angles, components, board without the components placed, front and back view.

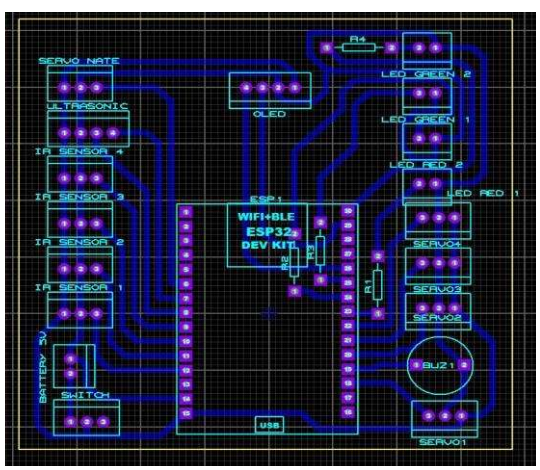

Figure 3.1 PCB Layout of circuit board

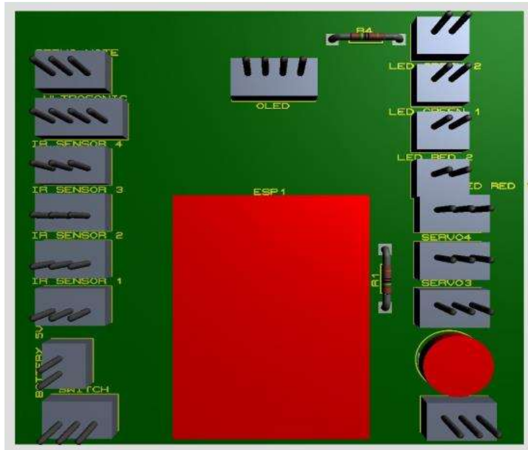

Figure 3.2 PCB 3D visualization with components of circuit board

3.2.2 PCB Etching

After the PCB 3D visualization, we need to print the copper layer and place it on the copper PCB board. Then, apply heat to the printed paper by an iron or any heat source. Next, place the board into the water and remove the paper, the carbon print on the board will remain. After that, place the board in ferric chloride liquid, the copper will react with the ferric chloride whereas the copper that have no carbon layer will be dissolved. Lastly, clean the board with sand to remove any excess carbon layer[2].

Figure 1.3 Bland copper PCB board

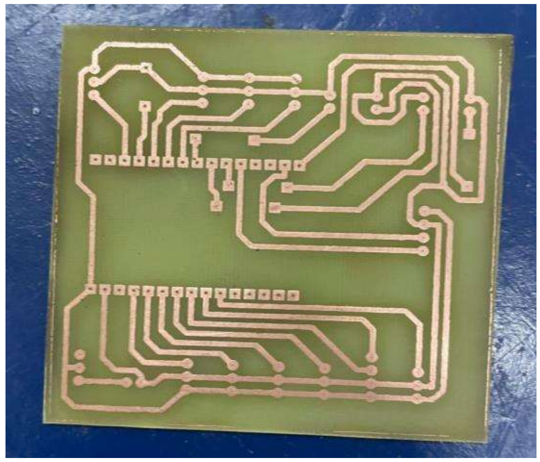

Figure 3.4 Complete circuit on the copper PCB board

3.2.3 Model testing

The ESP32 is used as our microcontroller to conduct our coding that is interfaced from Arduino IDE and is connected to Blynk IoT mobile application that is installed in our smartphone from the Play Store. Firstly, we need to initialize the project coding along with the Blynk IoT application in the Arduino IDE. Then, setup the OLED screen width and height for declaring the display. Then, initialize ultrasonic for distance limit, initialize SERVO pins for gate, bridge, and net, and initialize out pins for buzzer, LED red and green and initialize IR sensor pins to detect the exit and in cars on the road G1 and G2. After that, we need to define the steps of the bridge system for all the components that use in our prototype. Since our bridge system for vehicles and boats/ships is operating according to schedule, the time for bridge opening and closing need to be set at the beginning in Blynk IoT application. Thus, the results are shown in Experiment result Case 1.0, Case 1.1, Case 2.0 And Case 2.1.

3.2.4 Experimental results

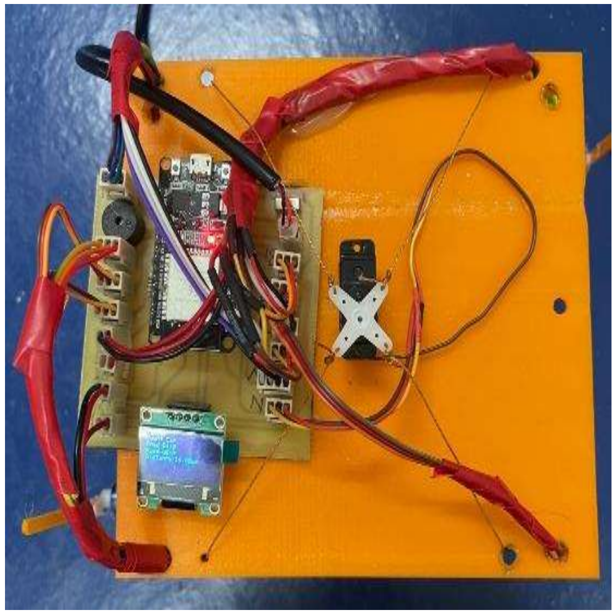

Figure 3.5 The circuit of project prototype

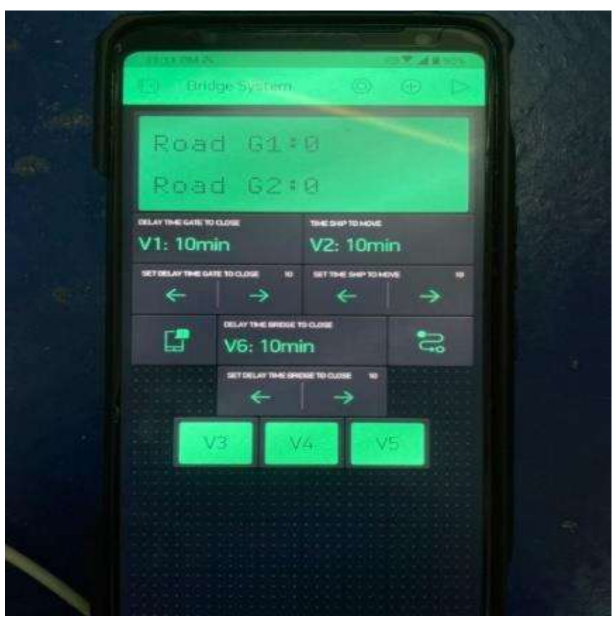

Figure 3.6 Blynk IoT application

3.2.5 Experiment result Case 1.0: When the boats/ships can pass through

During the schedule for bridge opening and when the bridge is clear, it is safe for the bridge to open. The signal from LED (GREEN and RED) will be blinking five-time, buzzer is ON to give a warning for toll gate to close, and the signal turn RED at both side which is at road G1 and road G2. OLED will display "Road G1: 0 and Road G2: 0" which explain that there is no vehicle on the road, while on the Blynk IoT application it will be stated "GATE WILL NOW CLOSE, Gate now closing", at the same time the toll gates will close. After the vehicle on the bridge clear, then the bridge open, which mean the road is closed. The safety net will pull up, buzzer OFF, the boats/ships can pass through under the bridge for 10 second.

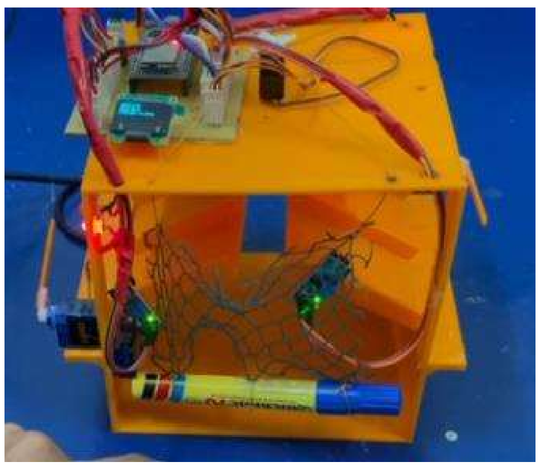

Figure 3.7 Tool gate close and bridge open

Figure 3.8 Blynk display "GATE WILL NOW CLOSE, Gate now closing"

3.2.6 Experiment result Case 1.1: When there is vehicle on the bridge

From Case 1.0, if there is any vehicle on the bridge that is detected by the IR sensor, the bridge is not yet ready to be open. The OLED will display “Road G1: n and Road G2: n” where “n” is the count of the cars on the bridge and the buzzer ON to give a warning. The system will wait until the bridge is clear. After it is clear, it will wait about 10 second and will back to operating state from Case 1.0 where the boats/ships can pass through under the bridge normally.

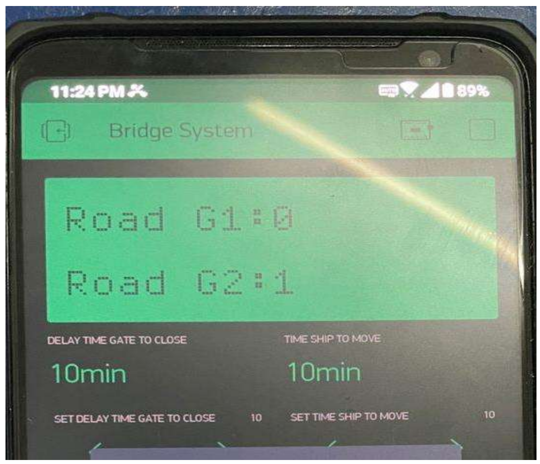

Figure 3.9 Blynk display "Road G1:0 and Road G2:1”

3.2.7 Experiment result Case 2.0: When the car can pass through

During the schedule for bridge closing and when the bridge is clear, it is safe for the bridge to close. The signal from LED turn RED, buzzer is ON to give a warning for safety net to close, while on the OLED will display “Road G1: 0 and Road G2: 0” which explain that there is no vehicle on the road and Blynk IoT application it will be stated “Bridge now closing and Net Being pull”, at the same time the safety net will pull down. When the boats/ships under the bridge is clear, then the bridge close. The toll gates will open, buzzer OFF, signal turn GREEN at both side of road G1 and road G2, the vehicle can pass on the bridge for 10 second.

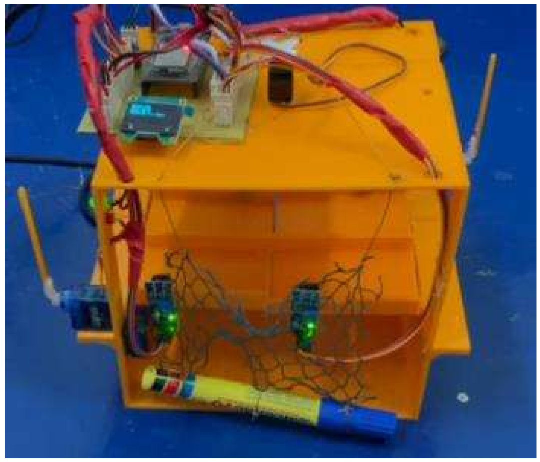

Figure 3.10 Bridge close and tool gates open

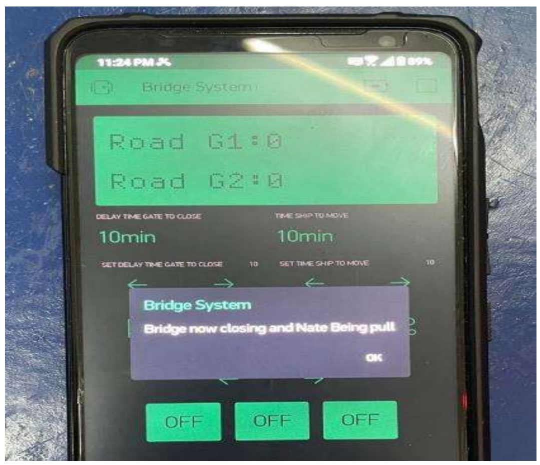

Figure 3.11 Blynk display "Bridge now closing and Nate Being pull"

3.2.8 Experiment result Case 2.1: When there are boats/ships under the bridge

From the Case 2.0, if there are ships under the bridge, the bridge is not yet ready to close. The bridge will not close until it is clear/empty. After there are no boats/ships under the bridge, it will wait about 10 second, the Blynk IoT application it will be stated “BRIDGE CAN BE USE NOW” and the bridge will back to operating state from Case 2.0 where the boats/ships can pass through under the bridge normally.

Figure 3.12 Blynk display "BRIDGE CAN BE USE NOW"

3.3 Conclusion

Our prototype uses IR sensor to detect and count the car on the bridge. If there is car on the bridge, it will wait 10 seconds to make sure the bridge is clear. Then, the servo SG60 is used to open and close the tool gate for the car to pass through on the bridge and the LED will display the red or green colour for the car to move or not. While Ultrasonic sensor detect and measure the distance about 14cm for the ships to pass through under the bridge. If there are ships under the bridge, it will also wait 10 seconds to make sure the bridge is clear. Then, the servo MG955R is used to pull-up and drop down the safety net. The buzzer is uses to give a warning to the driver for safety. The output of every movement will be display on the OLED display and Blynk application. All the data for sensor, servo motor, buzzer and led was conduct using ESP32. From experiment result Case 1.0, Case 1.1, Case 2.0 And Case 2.1, it is shown that our prototype is working properly and satisfied with our project objectives.

Chapter 4
Project Design Consideration

4.1 Introduction

From every project that has been designed, it will bring an impact to the society and environment. This chapter will be explaining in depth of the effect of this project to the society and environment. As professional engineers there are also responsibilities to the social, cultural, global, and environmental that need to be taken care of when working on a project. The professional and ethical responsibilities and commitment to the community also need to highly consider before launching the project to the society and environment.

There are positive and negative effects that occurred from this project usage [3].

Positive effects:

1. Passage of vessels are not required to request to open the bridge, the bridge will open automatically according to the schedule.
2. The bridge owners and tenders do not need to operate the bridge manually.
3. Signal and alarm system for bridge opening and closing.
4. Safety precaution system for both vehicles and ships that utilize the bridge.

Negative effects:

1. When the schedule for ships to pass through the bridge start, vehicles on the road have to wait behind the toll gate when the bridge is opened until it is closed again.

2. When the schedule for vehicles to pass through the bridge start, those ships that haven't passes before the bridge is closed have to wait behind the safety net until it is opened again.

4.2 Health and Safety

4.2.1 Safe working practices

It is not always possible to eliminate all potential risks to maintain a safe workplace for this project. Everyone needs to work carefully and with extreme caution. To reduce the risk of injury or death from any venue, there must be safety practices. The safety practices can be applied not only for the users, but also to the project. For this project, the safety practices are such as below [4].

1. Under-Bridge Protection.

   Construction machinery, tampering, weather events, and natural calamities can all harm bridge utility wires. These dangers can result in a loss of utility service, high repair expenses, injury, or even death. This issue can be prevented by installing caging units under the bridge to protect vulnerable utility lines under bridges.
2. High-Visibility Safety System.

   Signal system that will be used to give information about bridge status and alarm system to give warning to the bridge users.
3. Proper grounding.

   The risk of electrocution can be decreased and undesired voltage can be eliminated with proper grounding. Never remove the metallic ground pin since it is the source of errant voltage back to the ground.

4.2.2 Risk assessment

Table 4.1: Risk assessment 1

Risk 1	Vehicle stuck on the bridge during bridge opening time. • The risk of during bridge opening for ships to pass under the bridge but there are vehicles that still passing on the bridge.
Action	Installation of toll gates and sensor. • IR sensors work to count and check whether there are still vehicles that still remain on the bridge. Bridge will only open when the bridge is cleared. • Toll gates at the end of each side of the bridge where vehicles need to wait behind the toll gate during bridge opening.
Likelihood	3
Severity	4
Risk rating	12

Table 4.2: Risk assessment 2

Risk 2	Ship stuck under bridge during bridge closing time. • The risk of during bridge closing for vehicles to pass on the bridge but there are ships that still passing under the bridge.
Action	Installation of safety net and sensor. • Ultrasonic sensor detect whether there still have boats/ships that remain under the bridge. Bridge will only close when it is cleared. • Safety nets at the front and back side under the bridge where ships and boats need to wait behind the toll gate during bridge closing.
Likelihood	3
Severity	4
Risk rating	12

Table 4.3: Risk assessment table

Severity \ Likelihood	Rare (1)	Unlikely (2)	Possible (3)	Likely (4)	Rare (5)
Catastrophic (5)	5	10	15	20	25
Major (4)	4	8	12	16	20
Moderate (3)	3	6	9	12	15
Minor (2)	2	4	6	8	10
Very minor (1)	1	2	3	4	5

4.2.3 Project risk

There are project risks that need to be analyse when working on this project. Project risk analysis is crucial since it aids in identification of a project's shortcomings, strengths, and potential possibilities [5]. The dominant project risks are as below:

1. External hazard risk.

   The external hazard risk that can happen to this project is wind load. The wind load on a bascule bridge is rather high, especially when it is open. In contrast to the situation where there is no wind load, the machinery used to manage bascule bridges must be extremely sturdy and durable.

2. Operational risk.

   The stability of the bridge implements the crucial operation and core processes as production or procurement. The shear locks that located at each leaf of the bridge, typically at the end of the channel, are likely to suffer from wearing because of debris that contaminate the lubrication, and high traffic. The proper seating of the leaves on their live load shoe, the alignment of the leaves, and the fitting of each leaf's end by its locks all contribute to the stability of a double leaf bascule bridge. Any wear on the aligning components or damages to these components caused by temperature differences may result in the bridge not seating properly.

4.3 Cultural and Benefit to Society

This project is not against any community rules, norms, and values, it also has promoted good values and ethics where it brings positive impact to the society and community.

With the further development of our society, our demands on infrastructure will inevitably continue to increase. Take transportation as an example. As more vehicles ride on the road every day, the need for elastic structural systems to cope with the load increases. Without it, it is not easy to deal with traffic jams and traffic jams that can be very time consuming. This automatic bascule bridges system has help on both regulate and minimize traffic congestion.

Traffic: As already mentioned, bridges are crucial for controlling traffic because they are very good at diverting traffic to other parts of the city and easing the burden on the roads during peak hours. They must not be taken lightly because failing to do so would result in traffic jams and congestion that would completely waste your time [6].

The connection: In general, bridges make it easier to travel a shorter distance between two distant points by connecting them in a straight line. Smaller bridges barely notice this effect, but it is much more prominent over long distances and even aids in connecting islands and the mainland [6].

4.4 Environment and Sustainability: Sustainable Development Goal 11: Sustainable Cities and Communities

Figure 4.1 Sustainable Development Goal 11: Sustainable Cities and Communities

This project will support with UN's 17 Sustainable Development Goal number 11 which is Sustainable Cities and Communities, to make cities and human settlements inclusive, safe, resilient, and sustainable.

The world's population keep rising from day to days. For cities transportation to be sustainable and to satisfy everyone, there are some aspects that need to be improved. For examples are modern, intelligent urban design that generates affordable, secure, and resilient communities with green and culturally inspiring living conditions for all of us to survive and prosper [7].

Water transportation is the most cost-effective method in carrying people and large products across long distance. To ensure that the traffic for water transportation and land transportation will be upgraded to more advance technology, automatic transportation systems can be applied.

Based on Goal 11 Target: By 2030, provide access to safe, affordable, accessible and sustainable transport systems for all, improving road safety, notably by expanding public transport, with special attention to the needs of those in vulnerable situations, women, children, persons with disabilities and older persons [8].

This project will help to connect between land transportation and water transportation for the communities and at the same time will help to reduce the traffic jams and also the accidents that might occur on the bridge.

4.5 Commercialization Potential

This project is a technology that can be used for any bridge construction in the world. The target marketplaces are for bridge contractors and water and land transportation industry.

Table 4.4: Comparison survey between your product with available product in the market.

Projects / Aspects	Automatic Bascule Bridge System	Bascule Bridge
	Project Prototype	Available product in the market
Cost	RM 1000200000	RM 1000000000
Extra features	1. Sensors: • Vehicles and ships detection 2. LED: • GO and STOP signal 3. Toll gates: • Safety system for mainland 4. Safety net: • Safety system for waterway	1. Toll gates: • Safety system for mainland
Advantage	• Prevention of accidents. • Traffic jams control. • The constant monitoring of the structure's activities and changes provided by the bridge, enabling more efficient and secure operation and exploitation of the works.	• Whether fully or partially opened, most of bascule superstructure bridge is out of vessel reach during collision, so it would not suffer considerable damage. • Double leaf bascule bridge offers the broadest spaces for vessels compared with other types of movable bridges.
Disadvantage	• Require electrical construction	• Manual monitoring system

4.5.1 Project Costing

The project costings needed for this project are including the prototype cost until the market-ready price. There are material cost, operational cost, and total cost Material cost for this prototype is for the components and hardware part that are used to build the prototype.

Table 4.5: Material cost

Description	Quantity	Rate (RM)	Amount (RM)
ESP32 x1	1	29.00	29.00
IR sensor x4	4	3.00	12.00
Servo SG60 x5	5	8.00	40.00
USB Micro B Cable x1	1	5.00	5.00
OLED x1	1	25.00	25.00
Buzzer x1	1	1.50	1.50
LED x4	4	0.20	0.80
Female Header x1	1	2.00	2.00
Male Header x1	1	3.00	3.00
Wire x1	1	3.00	3.00
PCB Board x1	1	11.30	11.30
3D printing part 265g	1	26.40	26.40
Door Engsel	2	1.00	2.00
Ultrasonic Sensor	1	10.00	10.00
Nate	1	6.00	6.00
Rope	1	3.00	3.00
		Total	**RM 180.00**

The operational cost for this project prototype is not included because we are using existing tools and equipment that are contributed by the group members. to be precise, the operational methods in this project are for the prototype model 3D printing, and PCB board etching.

Overall, the total cost and also commercial price for this project is RM180.00 from the material cost.

Chapter 5
Conclusion and Future Work

5.1 Conclusion

We Have managed to develop an automation system that can organize the switch between water traffic and car traffic to prevent traffic Jams and to keep a smooth flow of traffic in both routes to saves time and costs with minimum human interaction, and at the same time implement safety measures in the system by synchronizing traffic flow with bridge movement, and adding extra safety measures by making sure there are no vehivles on the bridge or boats under the bridge while it is moving by using IR sensors and ultrasonic sensor. We have built a prototype and tested the system and it works flawlessly.

The project has been successfully carried out within the time stipulated this semester. As presented in Chapter 4, all objectives have been achieved. Objective 1 has been achieved on week 6 , whereas Objective 2 has been carried out on week 7 after tweeking the code and adding the safety features. Finally, objective 3 is achieved by building the prototype using 3d printing and fully testing the system and after fixing the bugs, all objectives have been thoroughly tested to make sure that they are working as intended.

5.2 Future Work

As stated in chapter 1, the project scope is to be be implemented on any bascule bridge. The toll gates for cars can be easily implemented on any bridge. Same for the sensors, however we might need sensors with stronger capabilities and longer ranges to be implemented for longer and wider bridge. The project limitations are that the safety nets that work as toll gates for boats, they can be implemented on small and short bridges only as the bigger bridges will require very big and heavy nets. So, the project can be further improved by stronger sensors that can cover wider ranges and using nets of lighter material or using another innovative way to implement toll gates for boats.

REFERENCES

[1] Singh, S. P., Anand, R. K., Vinay, V., Srivastava, S., Kumar, S., & Shaktawat, P. A. (2017). Automation of drawbridge model using by PLC. IARJSET, 4(5), 63–70. https://doi.org/10.17148/iarjset.2017.4513

[2] James, Zad, & Yonas. (2021, September 29). Designing your own PCB with proteus: Easy tutorial for beginners. Electronics Lovers ~ Technology We Love. Retrieved July 1, 2022, from https://www.electronicslovers.com/2018/09/design-your-own-pcb-by-using-proteus.html

[3] I.Berger, D. Healy and M. Tilley. (2015). Movable Span Bridge Study. Volume 2: Bascule and Swing Span Bridges.

[4] Bridge Master. (2016). 10 Essential Safety Tools for Bridge Workers. https://bridgemastersinc.com/10-safety-tools-bridge-workers/

[5] Indeed Editorial Team. (2022). 10 Common Project Risks (Plus the Steps To Solve Them). https://www.indeed.com/career-advice/career-development/project-risks#:~:text=Why%20is%20it%20important%20to,or%20after%20completing%20a%20project.

[6] Alan Behrens. (2018). Positive and Negative Impact of Bridges. https://positivenegativeimpact.com/bridges

[7] Null, (null). Goal 11:Sustainable cities and communities. The Global Goals. https://www.globalgoals.org/goals/11-sustainable-cities-and communities/

[8] Null, (null). Goal 11: Sustainable cities and communities. https://www.jointsdgfund.org/sustainable-development-goals/goal-11-sustainable-cities-and-communities

APPENDIX A

GANTT CHART

Activities	Week													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
IDP Briefing														
Meeting with supervisor for title discussion and research related to the project.														
Project design														
•Circuit component														
•PCB circuit														
•Programme code														
•Prototype design														
Operational work														
•Programme code interface														
•Prototype frame 3D printing														
•PCB etching														
•Circuit assembly														
•Prototype assembly														
Troubleshoot														

PROJECT CODING

```
//===================== INTIALIZE BYLNK ========================
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>

// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "7SW2R9CtkU469aPDUKQ-KWh1T9TSh_ct";

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "KAIZOKU";
char pass[] = "nakpassword";

WidgetLCD lcd0(V0);

int Bridge_Alarm1_Triggered = 0,
    Bridge_Alarm2_Triggered = 0,
    Bridge_Alarm3_Triggered = 0,
    GATE_CLOSE_TIME_DELAY = 10,// In Minute
    BRIDGE_CLOSE_TIME = 10,//In Minute
    DELAY_AFTER_CAR_CNT = 10;//10 min

BLYNK_CONNECTED(){
  Blynk.syncVirtual(V1);
  Blynk.syncVirtual(V2);
  Blynk.syncVirtual(V3);
  Blynk.syncVirtual(V4);
  Blynk.syncVirtual(V5);
  Blynk.syncVirtual(V6);
}

BLYNK_WRITE(V1){
  GATE_CLOSE_TIME_DELAY = param.asInt();
}
BLYNK_WRITE(V2){
  BRIDGE_CLOSE_TIME = param.asInt();
```

```
}

BLYNK_WRITE(V3){
  Bridge_Alarm1_Triggered = param.asInt();
}
BLYNK_WRITE(V4){
  Bridge_Alarm2_Triggered = param.asInt();
}
BLYNK_WRITE(V5){
  Bridge_Alarm3_Triggered = param.asInt();
}
BLYNK_WRITE(V6){
  DELAY_AFTER_CAR_CNT = param.asInt();
}

int Notification_1 = 0,
    Notification_2 = 0;
int Alarm_Flag = 0;
//===================== OLED Setup ================
#include <Adafruit_GFX.h>        //OLED libraries
#include <Adafruit_SSD1306.h>
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
#define OLED_RESET    -1 // Reset pin # (or -1 if sharing Arduino reset pin)

#include <Wire.h>
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
//Declaring the display name (display)
//================= Initialize Ultrasonic =================
#define trigPin 25
#define echoPin 33

float duration;

float distance;
const int distance_limit = 5;

int State = 0;
//================= Initialize SERVO PINS ================
```

```
#include <ESP32_Servo.h>

Servo Gate1,
      Gate2,
      Bridge1,
      Bridge2,
      Nate;

#define Gate1_Pin 26
#define Gate2_Pin 16
#define Bridge1_Pin 12
#define Bridge2_Pin 17
#define Nate_Pin 32

int Nate_Drop = 180;
int Nate_Pickup = 0;

int Bridge_Close_STEP = 0,
    Bridge_Open_STEP = 90;

int Bridge1_Open = 90,
    Bridge1_Close = 45;

int Bridge2_Open = 90,
    Bridge2_Close = 45;

int Gate_Close_STEP = 180,
    Gate_Open_STEP = 90;

int Gate1_Open = 90,
    Gate1_Close = 180;

int Gate2_Open = 90,
    Gate2_Close = 180;

int Servo_Move_Delay = 50;
//================ Initialize OUT PINS ================
#define LED_RED_G1 18
#define LED_RED_G2 23
```

```
#define LED_GREEN_G1 5
#define LED_GREEN_G2 19
#define BUZZER 2
int CNT_STEP = 0;
int LED_BLINKING = 0,
   ENABLE_LED_BLINKING = 0;
//================= DELAY WITHOUT DELAY VARIABLES =================
unsigned long previousMillis = 0;
const long interval = 1000;
//================= Initialize IR SENSOR PINS =================
#define IR_G1_OUT 13
#define IR_G1_IN 27
#define IR_G2_OUT 14
#define IR_G2_IN 4

int IR_ROAD1_EXIT = 0,
   IR_ROAD1_IN= 0,
   IR_ROAD2_EXIT = 0,
   IR_ROAD2_IN = 0;

int Last_IR_ROAD1_EXIT = 0,
   Last_IR_ROAD1_IN= 0,
   Last_IR_ROAD2_EXIT = 0,
   Last_IR_ROAD2_IN = 0;

int CNT_CAR_G1 = 0,
   CNT_CAR_G2 = 0;
//===================== DEFINE STEP ============
enum _BRIDGE_SYSTEM
{
   BRIDGE_OPEN,
   BRIDGE_CLOSE
};
volatile _BRIDGE_SYSTEM BRIDGE_STATE = BRIDGE_OPEN;

enum _BRIDGE_CLOSE_SYSTEM
{
   IDLE_1,
   TRIGGER_BRIDGE_CLOSE,
```

```
  MAKE_SURE_NO_CAR,
  ALLOW_SHIP_PASSBY,
};
volatile _BRIDGE_CLOSE_SYSTEM BRIDGE_CLOSE_STATE = IDLE_1;

enum _BRIDGE_OPEN_SYSTEM
{
  IDLE_2,
  DROP_NATE,
  OPEN_BRIDGE,
  OPEN_GATE
};
volatile _BRIDGE_OPEN_SYSTEM BRIDGE_OPEN_STATE = IDLE_2;

int ENABLE_BRIDGE_CLOSE = 0;
int BRIDGE_CLOSE_NOW = 0;
int ENABLE_BRIDGE_OPEN = 0;

int Debouncer_1 = 0;
int Ship_waitime = 0;
//================ VOID SETUPS =================
void setup() {
 Serial.begin(9600);
 display.begin(SSD1306_SWITCHCAPVCC, 0x3C); //Start the OLED display
 display.display();

 // DEFINE SERVO PINS
 Gate1.attach(Gate1_Pin, 500, 2400);
 Gate2.attach(Gate2_Pin, 500, 2400);
 Bridge1.attach(Bridge1_Pin, 500, 2400);
 Bridge2.attach(Bridge2_Pin, 500, 2400);
 Nate.attach(Nate_Pin, 500, 2400);
 // DEFINE IR SENSOR AS INPUT
 pinMode(IR_G1_OUT,INPUT);
 pinMode(IR_G1_IN,INPUT);
 pinMode(IR_G2_OUT,INPUT);
 pinMode(IR_G2_IN,INPUT);
 // DEFINE ULTRASONIC PINS
 pinMode(trigPin,OUTPUT);    //sets trigpin as output to send ultrasound
```

```
pinMode(echoPin,INPUT);     //sets echopin as input to receive ultrasound
// DEFINE LED AND BUZZER AS OUTPUT
pinMode(LED_RED_G1,OUTPUT);
pinMode(LED_RED_G2,OUTPUT);
pinMode(LED_GREEN_G1,OUTPUT);
pinMode(LED_GREEN_G2,OUTPUT);
pinMode(BUZZER,OUTPUT);

digitalWrite(LED_RED_G1,LOW);
digitalWrite(LED_RED_G2,LOW);
digitalWrite(LED_GREEN_G1,HIGH);
digitalWrite(LED_GREEN_G2,HIGH);

digitalWrite(BUZZER,HIGH);
delay(150);
digitalWrite(BUZZER,LOW);

Gate1.write(Gate1_Open);
Gate2.write(Gate2_Open);
Bridge1.write(Bridge1_Open);
Bridge2.write(Bridge2_Open);
Nate.write(Nate_Drop);

display.clearDisplay();

display.setTextSize(2);
display.setTextColor(WHITE);
display.setCursor(20,1);
display.print("WIFI");
display.setTextSize(1);
display.setCursor(0,20);
display.print("Connecting to");
display.setCursor(0,30);
display.print(ssid);
display.display();

Serial.print("Connecting to ");
Serial.println(ssid);
```

```
WiFi.begin(ssid, pass);

long int StartTime=millis();
  while (WiFi.status() != WL_CONNECTED)
  {
    int i;
    i++;
    Serial.println(i);
    if ((StartTime+10000) < millis()) break;
  }
if (WiFi.status() != WL_CONNECTED)
{
    Serial.println("Reset");
    display.clearDisplay();

    display.setCursor(0,0);
    display.print("Reseting..");

    display.display();

    digitalWrite(BUZZER,HIGH);
    delay(500);
    digitalWrite(BUZZER,LOW);
    delay(500);
    ESP.restart();
  }else{
    Blynk.begin(auth, ssid, pass);

    Serial.println("WiFi connected");

    display.clearDisplay();

    display.setCursor(0,0);
    display.print("Wifi Connected");
    display.setCursor(0,10);
    display.print("with IP:");
    display.setCursor(0,20);
    display.print(WiFi.localIP());
```

```
      display.display();

      delay(1000);
      display.clearDisplay();
      lcd0.clear();


    }
}
//================= VOID LOOPS =================
void loop() {
  Blynk.run();
  IR_ROAD1_EXIT = digitalRead(IR_G1_OUT);
  IR_ROAD1_IN = digitalRead(IR_G1_IN);
  IR_ROAD2_EXIT = digitalRead(IR_G2_OUT);
  IR_ROAD2_IN = digitalRead(IR_G2_IN);

    unsigned long currentMillis = millis();

  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;
    CNT_STEP++;
    Ship_waitime++;
/*
    Serial.println();
    Serial.print("IR_ROAD1_EXIT:");
    Serial.print(IR_ROAD1_EXIT);
    Serial.print("  ");

    Serial.print("IR_ROAD1_IN:");
    Serial.print(IR_ROAD1_IN);
    Serial.print("  ");

    Serial.print("IR_ROAD2_EXIT:");
    Serial.print(IR_ROAD2_EXIT);
    Serial.print("  ");

    Serial.print("IR_ROAD2_IN:");
    Serial.print(IR_ROAD2_IN);
```

```
    Serial.print("  ");
*/
    display.clearDisplay();
    display.setCursor(0,0);
    display.print("Count Car");
    display.setCursor(0,10);
    display.print("Road G1:");
    display.print(CNT_CAR_G1);
    display.setCursor(0,20);
    display.print("Road G2:");
    display.print(CNT_CAR_G2);
    display.setCursor(0,30);
    display.print("Distance:");
    display.print(distance);
    display.print("cm");


    lcd0.print(0,0,"Road G1:"+String(CNT_CAR_G1));
    lcd0.print(0,1,"Road G2:"+String(CNT_CAR_G2));

    Ultrasonic_Data(); /*
    Serial.print("Distance:");
    Serial.print(distance);
    Serial.println("cm");*/

    LED_BLINKING_FUNCTION();
  }

  //===COUNT CAR====
  COUNT_CAR();
  Bridge_System_Overall();

  display.display();
}
 /*================ Read Obstacle =================*/
void Ultrasonic_Data(){
  digitalWrite(trigPin, LOW);   //sets trigpin to LOW so that it do not emits ultrasound

                                                    //  initially
```

```
  delayMicroseconds(2);

  digitalWrite(trigPin, HIGH);              //sets trigpin to HIGH state to emit ultrasound

  delayMicroseconds(10);

  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);                    //reads the travel time of the pulse

  distance = microsecondsToCentimeters(duration);    //function
}
/*================ Ultrasonic Equation Function =================*/
long microsecondsToCentimeters(long microseconds)
{

 return microseconds / 29/ 2;  /*Speed of sound is 340m/s. So it takes 29 microseconds to  travel 1
centimeter length of the path.All the working  is same as explained in calculation of inches*/

}
 /*================ Bridge_System_Overall =================*/
void Bridge_System_Overall(){
  switch (BRIDGE_STATE)
  {
  case BRIDGE_OPEN:

    if(Bridge_Alarm1_Triggered == HIGH ){
      if(ENABLE_BRIDGE_CLOSE == LOW){
        Blynk.notify("Alarm 1 Triggered Now");
        Blynk.virtualWrite(V3,0);
        Blynk.virtualWrite(V4,0);
        Blynk.virtualWrite(V5,0);
        Bridge_Alarm1_Triggered = LOW;
        Bridge_Alarm2_Triggered = LOW;
        Bridge_Alarm3_Triggered = LOW;
        //Alarm_Flag = 1;
        ENABLE_BRIDGE_CLOSE = HIGH;
        Serial.println("Alarm1 Triggered");
```

```
  }
}
if(Bridge_Alarm2_Triggered == HIGH ){
  if(ENABLE_BRIDGE_CLOSE == LOW){
    Blynk.notify("Alarm 2 Triggered Now");
    Blynk.virtualWrite(V3,0);
    Blynk.virtualWrite(V4,0);
    Blynk.virtualWrite(V5,0);
    Bridge_Alarm1_Triggered = LOW;
    Bridge_Alarm2_Triggered = LOW;
    Bridge_Alarm3_Triggered = LOW;
    //Alarm_Flag = 2;
    ENABLE_BRIDGE_CLOSE = HIGH;
    Serial.println("Alarm2 Triggered");

  }
}
if(Bridge_Alarm3_Triggered == HIGH ){
  if(ENABLE_BRIDGE_CLOSE == LOW ){
    Blynk.notify("Alarm 3 Triggered Now");
    Blynk.virtualWrite(V3,0);
    Blynk.virtualWrite(V4,0);
    Blynk.virtualWrite(V5,0);
    Bridge_Alarm1_Triggered = LOW;
    Bridge_Alarm2_Triggered = LOW;
    Bridge_Alarm3_Triggered = LOW;
    //Alarm_Flag = 0;
    ENABLE_BRIDGE_CLOSE = HIGH;
    Serial.println("Alarm3 Triggered");

  }
}

if(ENABLE_BRIDGE_CLOSE == HIGH && ENABLE_BRIDGE_OPEN == LOW){
  CNT_STEP = 0;
  Serial.println("Close Bridge S1");
  BRIDGE_CLOSE_STATE = TRIGGER_BRIDGE_CLOSE;
  BRIDGE_STATE = BRIDGE_CLOSE;
}
```

```
    if(ENABLE_BRIDGE_OPEN == HIGH){
      Bridge_Open_System_Overall();
    }

    break;

  case BRIDGE_CLOSE:
      Bridge_Close_System_Overall();
    break;

  default:
    break;
  }
}
/*================ LED_BLINKING_FUNCTION =================*/
void Bridge_Open_System_Overall(){
  switch (BRIDGE_OPEN_STATE)
  {
  case DROP_NATE:
    //Drop Nate Slowly
      if(Notification_2 == 0){
        Serial.println("Dropping nate now");
        Blynk.notify("Dropping nate now");
        Notification_2 = 1;
      }
      for( Nate_Drop = 0; Nate_Drop <= 180; Nate_Drop++){
        Nate.write(Nate_Drop);
        digitalWrite(BUZZER,HIGH);
        delay(Servo_Move_Delay);
        digitalWrite(BUZZER,LOW);
        delay(Servo_Move_Delay);
      }
    BRIDGE_OPEN_STATE = OPEN_BRIDGE;
    break;

    case OPEN_BRIDGE:
       //Open Bridge Slowly
      if(Notification_2 == 1){
```

```
      Serial.println("Opening bridge now");
      Blynk.notify("Opening bridge now");
      Notification_2 = 0;
    }
    for( Bridge_Open_STEP = Bridge2_Close; Bridge_Open_STEP <= Bridge2_Open;
Bridge_Open_STEP++){
      Bridge1.write(Bridge_Open_STEP);
      Bridge2.write(Bridge_Open_STEP);
      digitalWrite(BUZZER,HIGH);
      delay(Servo_Move_Delay);
      digitalWrite(BUZZER,LOW);
      delay(Servo_Move_Delay);
    }
    BRIDGE_OPEN_STATE = OPEN_GATE;
    break;

   case OPEN_GATE:
    //Open Gate Slowly
    if(Notification_2 == 0){
      Serial.println("Opening gate now");
      Blynk.notify("Opening gate now");
      Notification_2 = 1;
    }
    for( Gate_Open_STEP = 180; Gate_Open_STEP >= 90; Gate_Open_STEP--){
      Gate1.write(Gate_Open_STEP);
      Gate2.write(Gate_Open_STEP);
      digitalWrite(BUZZER,HIGH);
      delay(Servo_Move_Delay);
      digitalWrite(BUZZER,LOW);
      delay(Servo_Move_Delay);
      }
    ENABLE_LED_BLINKING = 0;
    digitalWrite(LED_RED_G1,LOW);
    digitalWrite(LED_RED_G2,LOW);
    digitalWrite(LED_GREEN_G1,HIGH);
    digitalWrite(LED_GREEN_G2,HIGH);
    BRIDGE_OPEN_STATE = IDLE_2;
    break;
```

```
  default:
    if(Notification_2 == 1){
      Blynk.notify("BRIDGE CAN BE USE NOW");
      ENABLE_BRIDGE_CLOSE = LOW;
      Notification_2 = 0;
      CNT_STEP = 0;
    }
    ENABLE_BRIDGE_OPEN = LOW;
    break;
  }
}
 /*================ Bridge_Close_System_Overall =================*/
void Bridge_Close_System_Overall(){
  switch (BRIDGE_CLOSE_STATE)
  {

    case TRIGGER_BRIDGE_CLOSE://CLOSE GATE FIRST
        ENABLE_LED_BLINKING = 1;
        if(CNT_STEP > GATE_CLOSE_TIME_DELAY){
          if(Notification_1 == 0){
            ENABLE_LED_BLINKING = 0;
            Serial.println("BRIDGE WILL NOW CLOSE\nGate Now Close");
            Blynk.notify("BRIDGE WILL NOW CLOSE\nGate now closing");
            digitalWrite(LED_RED_G1,HIGH);
            digitalWrite(LED_RED_G2,HIGH);
            digitalWrite(LED_GREEN_G1,LOW);
            digitalWrite(LED_GREEN_G2,LOW);
            Notification_1 = 1;
          }
          //Clossing Gate Slowly
          for( Gate_Close_STEP = 90; Gate_Close_STEP <= 180; Gate_Close_STEP++){
            Gate1.write(Gate_Close_STEP);
            Gate2.write(Gate_Close_STEP);
            digitalWrite(BUZZER,HIGH);
            delay(Servo_Move_Delay);
            digitalWrite(BUZZER,LOW);
            delay(Servo_Move_Delay);
            }
          BRIDGE_CLOSE_NOW = LOW;
```

```
      CNT_STEP = 0;
      BRIDGE_CLOSE_STATE = MAKE_SURE_NO_CAR;
     }
    break;

   case MAKE_SURE_NO_CAR://SCAN NO CAR & PULL NATE SLOWLY
     if(CNT_CAR_G1 == 0 && CNT_CAR_G2 == 0 && BRIDGE_CLOSE_NOW == LOW &&
CNT_STEP >= DELAY_AFTER_CAR_CNT){
      BRIDGE_CLOSE_NOW = HIGH;
     }
     if(CNT_CAR_G1 > 0){
      CNT_STEP = 0;
     }
     if(CNT_CAR_G2 > 0){
      CNT_STEP = 0;
     }

     if(BRIDGE_CLOSE_NOW == HIGH){
     //Clossing Bridge Slowly
      if(Notification_1 == 1){
       Serial.println("Bridge now closing and Nate Being pull");
       Blynk.notify("Bridge now closing and Nate Being pull");
       digitalWrite(LED_RED_G1,HIGH);
       digitalWrite(LED_RED_G2,HIGH);
       digitalWrite(LED_GREEN_G1,LOW);
       digitalWrite(LED_GREEN_G2,LOW);
       Notification_1 = 0;
      }
      for( Bridge_Close_STEP = Bridge2_Open; Bridge_Close_STEP >= Bridge2_Close;
Bridge_Close_STEP--){
       Bridge1.write(Bridge_Close_STEP);
       Bridge2.write(Bridge_Close_STEP);
       digitalWrite(BUZZER,HIGH);
       delay(Servo_Move_Delay);
       digitalWrite(BUZZER,LOW);
       delay(Servo_Move_Delay);
      }
     //Pull Nate Slowly
      for( Nate_Pickup = 180; Nate_Pickup >= 0; Nate_Pickup--){
```

```
        Nate.write(Nate_Pickup);
        digitalWrite(BUZZER,HIGH);
        delay(Servo_Move_Delay);
        digitalWrite(BUZZER,LOW);
        delay(Servo_Move_Delay);
       }
       CNT_STEP = 0;
       BRIDGE_CLOSE_STATE = ALLOW_SHIP_PASSBY;
      }
     break;

    case ALLOW_SHIP_PASSBY:// SHIP PASS BY TIMER
      //If time more than BRIDGE_CLOSE_TIME & distance >= 8 then bridge start to open
      if(CNT_STEP >= BRIDGE_CLOSE_TIME && distance >= 10){
       if(Ship_waitime > 10){//more than 10min
        ENABLE_BRIDGE_OPEN = HIGH;
        BRIDGE_OPEN_STATE = DROP_NATE;
        BRIDGE_STATE = BRIDGE_OPEN;
       }
      }else if(distance < 10){
       Ship_waitime = 0;
      }
     break;

    default:
     break;
  }
}
/*================ LED_BLINKING_FUNCTION =================*/
void LED_BLINKING_FUNCTION(){
  if(CNT_STEP < GATE_CLOSE_TIME_DELAY){
    if(ENABLE_LED_BLINKING == 1){
     if(LED_BLINKING == 0){
      LED_BLINKING = 1;
     }else{
      LED_BLINKING = 0;
     }
     digitalWrite(BUZZER,LED_BLINKING);
     digitalWrite(LED_RED_G1,LED_BLINKING);
```

```
      digitalWrite(LED_RED_G2,LED_BLINKING);
      digitalWrite(LED_GREEN_G1,LED_BLINKING);
      digitalWrite(LED_GREEN_G2,LED_BLINKING);
    }
  }else{
    if(ENABLE_LED_BLINKING == 1){
      digitalWrite(LED_RED_G1,HIGH);
      digitalWrite(LED_RED_G2,HIGH);
      digitalWrite(LED_GREEN_G1,LOW);
      digitalWrite(LED_GREEN_G2,LOW);
      ENABLE_LED_BLINKING = 0;
    }

  }
}
/*================ COUNT_CAR =================*/
void COUNT_CAR(){
  //ROAD G1
    //INCREMENT
    if(IR_ROAD1_IN != Last_IR_ROAD1_IN){
      if(IR_ROAD1_IN == HIGH){
        CNT_CAR_G1++;
      }
      delay(100);
      Last_IR_ROAD1_IN = IR_ROAD1_IN;
    }
    //DECREMENT
    if(IR_ROAD1_EXIT != Last_IR_ROAD1_EXIT){
      if(IR_ROAD1_EXIT == HIGH){
        CNT_CAR_G1--;
        if(CNT_CAR_G1 <= 0){
          CNT_CAR_G1 = 0;
        }
      }
      delay(100);
      Last_IR_ROAD1_EXIT = IR_ROAD1_EXIT;
    }
    //ROAD G2
    //INCREMENT
```

```
  if(IR_ROAD2_IN != Last_IR_ROAD2_IN){
    if(IR_ROAD2_IN == HIGH){
      CNT_CAR_G2++;
    }
    delay(100);
    Last_IR_ROAD2_IN = IR_ROAD2_IN;
  }
  //DECREMENT
  if(IR_ROAD2_EXIT != Last_IR_ROAD2_EXIT){
    if(IR_ROAD2_EXIT == HIGH){
      CNT_CAR_G2--;
      if(CNT_CAR_G2 <= 0){
        CNT_CAR_G2 = 0;
      }
    }
    delay(100);
    Last_IR_ROAD2_EXIT = IR_ROAD2_EXIT;
  }

}
```