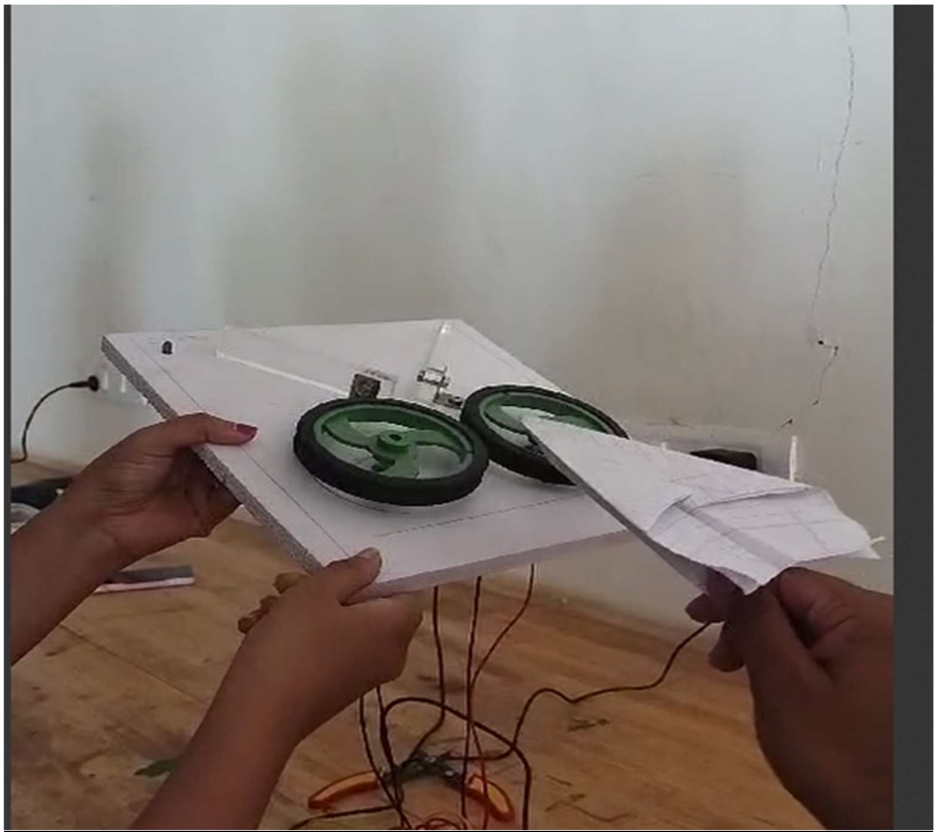
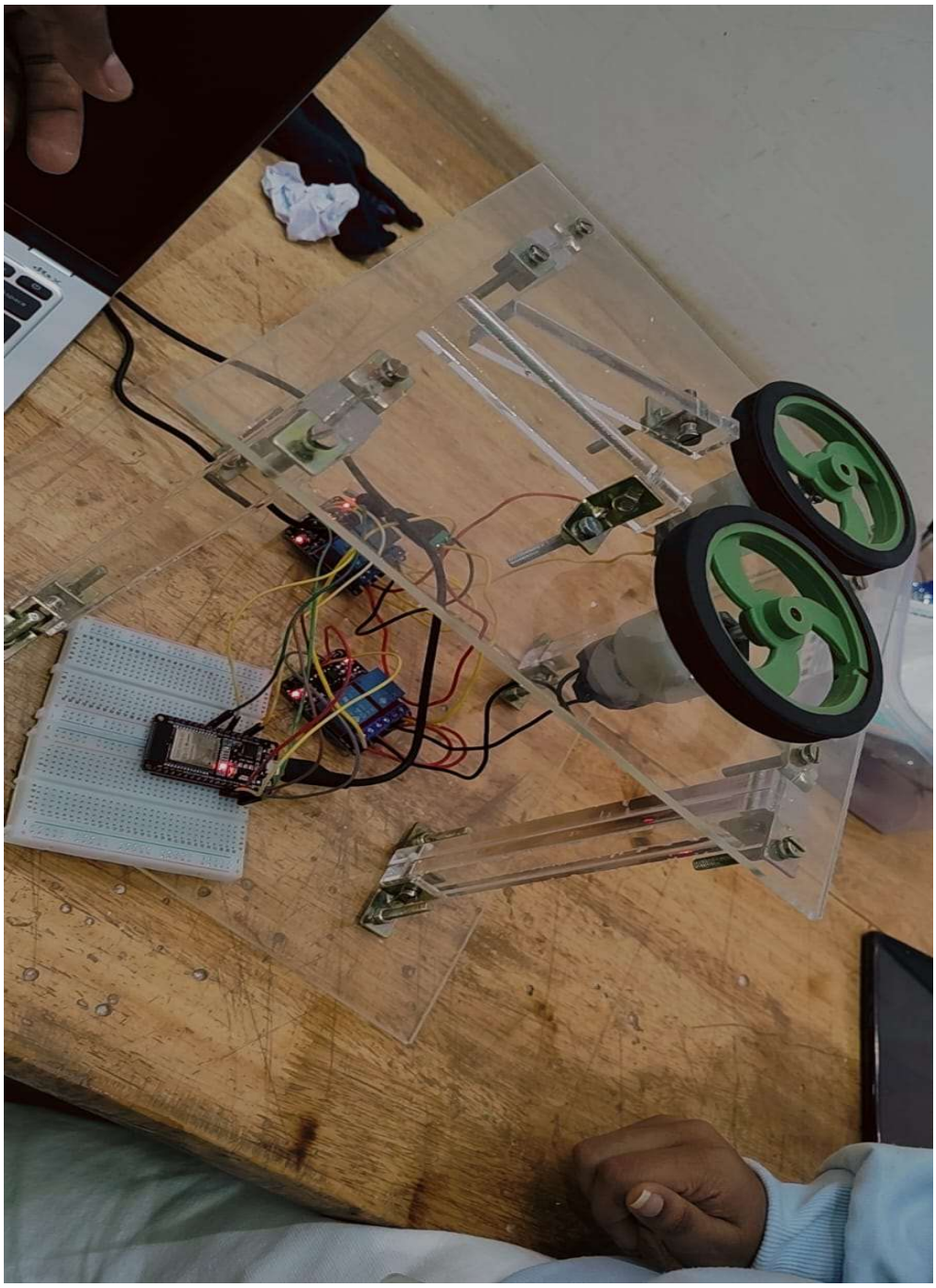
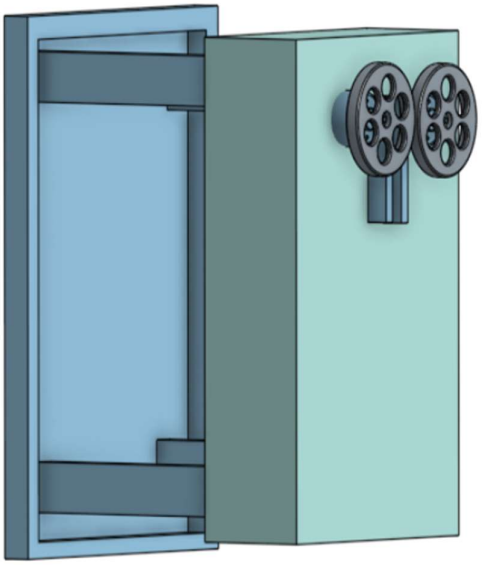
LAUNCHING PAPER PLANE

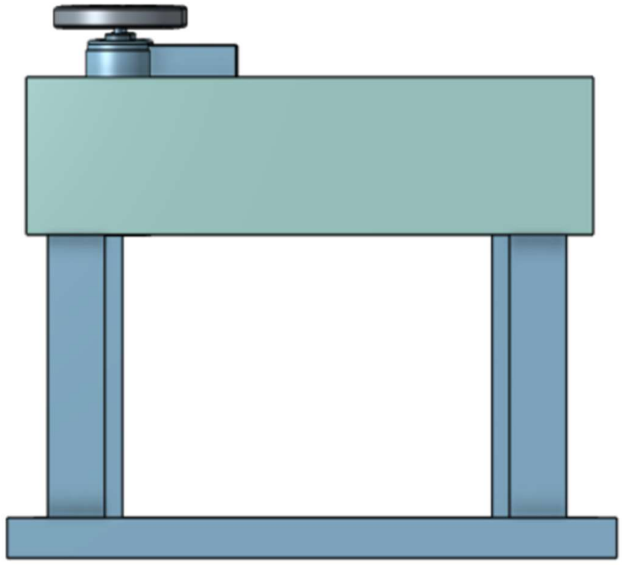




(changed to acrylic)



(3d ON ONSHAPE)



Summary

This circuit is designed to control two 12V geared motors and two LEDs (one red, one green) using an ESP32 microcontroller. The ESP32 reads distance measurements from an ultrasonic sensor and controls the motors and LEDs based on the distance detected. The motors are interfaced through a two-channel 5V relay, which allows for on/off control. The LEDs are used as indicators, with the green LED indicating an object is detected within a certain range and the red LED indicating no object is detected or it

is out of range. The circuit is powered by a 12V power supply, which also powers the motors, while the ESP32 and other low-voltage components are powered through voltage regulation.

Component List

● Microcontroller ESP32:

A microcontroller with Wi-Fi and Bluetooth capabilities, featuring multiple GPIO pins for interfacing with sensors, actuators, and other peripherals.

● Actuators 12V Geared Motors (x2):

Motors that convert electrical energy into mechanical motion, geared for torque over speed.

● Two Channel Relay 5V:

An electromechanical switch that allows the ESP32 to control higher voltage components such as the 12V motors.

Indicators :

LED: Two Pin (red): A red light-emitting diode used as an indicator.

LED: Two Pin (green): A green light-emitting diode used as an indicator.

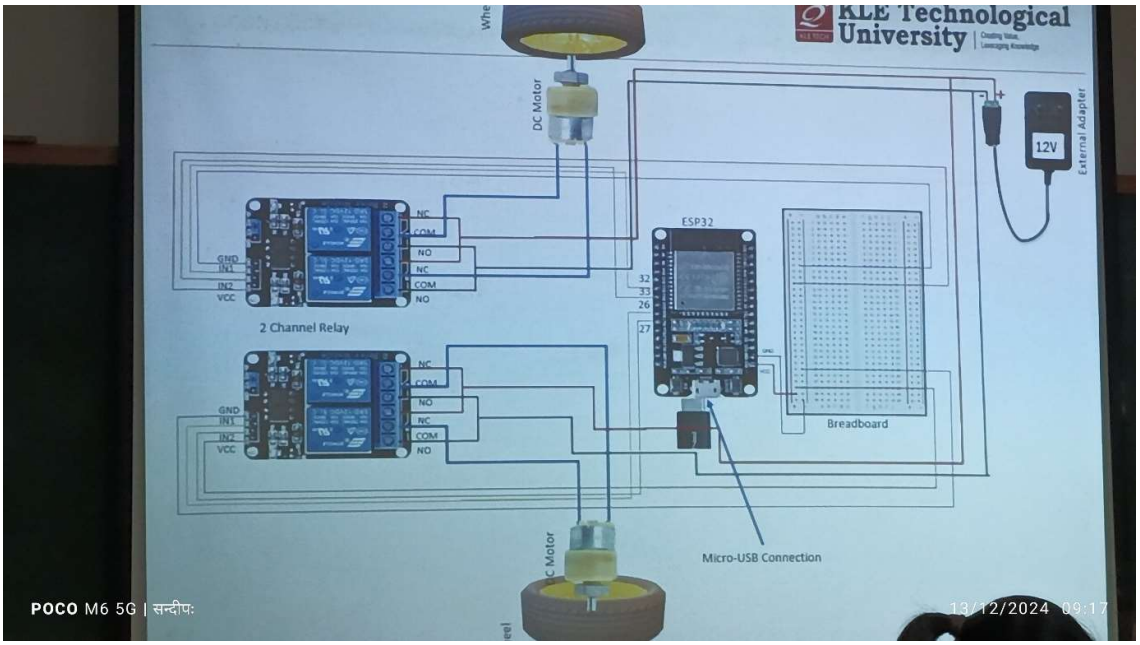
● 12V Power Supply:

Provides the necessary power to the motors and is also used to derive power for the relay and ESP32 through voltage regulation.

● Resistors (200 Ohms):

Used to limit current to the LEDs to prevent damage.

Circuit Documentation:



DOCUMENTED CODE:

```
#include <BluetoothSerial.h>

// Motor control pins (Relay pins)

#define motor1_forward 25

#define motor1_backward 26

#define motor2_forward 4

#define motor2_backward 5

#define red 12

#define green 13

BluetoothSerial SerialBT;

void setup() {

  // Start the serial communication

  Serial.begin(115200);

  SerialBT.begin("ESP32_Sandeep");

 // Set motor control pins as outputs
```

```
  pinMode(motor1_forward, OUTPUT);
  pinMode(motor1_backward, OUTPUT);
  pinMode(motor2_forward, OUTPUT);
  pinMode(motor2_backward, OUTPUT);
  pinMode(red, OUTPUT);
  pinMode(green, OUTPUT);
}
void loop() {
// Nothing needs to be done in the loop as the server handles everything
if (SerialBT.available()) {
   char incomingChar = SerialBT.read();
   Serial.println(incomingChar);
switch (incomingChar) {
     case '1':
       digitalWrite(motor1_forward, HIGH);
       digitalWrite(motor1_backward, LOW);
       digitalWrite(motor2_forward, LOW);
       digitalWrite(motor2_backward, HIGH);
       digitalWrite(green, HIGH);
       digitalWrite(red, LOW);
       break;
case '2':
       digitalWrite(motor1_forward, LOW);
       digitalWrite(motor1_backward, LOW);
       digitalWrite(motor2_forward, LOW);
       digitalWrite(motor2_backward,LOW);
       digitalWrite(red, HIGH);
       digitalWrite(green, LOW);
       break;
}
delay(5);
```

```
  }
}
```

WORKING:

Explanation

1. Include Libraries:

```
#include <BluetoothSerial.h>
```

   - This library is necessary to use Bluetooth communication on the ESP32.

2. Define Motor and LED Control Pins:

```
#define motor1_forward 25
#define motor1_backward 26
#define motor2_forward 4
#define motor2_backward 5
#define red 12
#define green 13
```

   - These lines define the GPIO pins on the ESP32 used to control the motors and LEDs.

3. Initialize Bluetooth Serial:

```
BluetoothSerial SerialBT;
```

   - Creates an instance of the BluetoothSerial class to handle Bluetooth communication.

4. Setup Function:

```
void setup() {
  Serial.begin(115200);
  SerialBT.begin("ESP32_Sandeep");
  pinMode(motor1_forward, OUTPUT);
  pinMode(motor1_backward, OUTPUT);
  pinMode(motor2_forward, OUTPUT);
  pinMode(motor2_backward, OUTPUT);
```

```
  pinMode(red, OUTPUT);
  pinMode(green, OUTPUT);
}
```

- Initializes the serial communication and Bluetooth communication.
- Sets the motor and LED control pins as output pins.

5. Loop Function:

```
void loop() {
  if (SerialBT.available()) {
    char incomingChar = SerialBT.read();
    Serial.println(incomingChar);
    switch (incomingChar) {
      case '1':
        digitalWrite(motor1_forward, HIGH);
        digitalWrite(motor1_backward, LOW);
        digitalWrite(motor2_forward, LOW);
        digitalWrite(motor2_backward, HIGH);
        digitalWrite(green, HIGH);
        digitalWrite(red, LOW);
        break;
      case '2':
        digitalWrite(motor1_forward, LOW);
        digitalWrite(motor1_backward, LOW);
        digitalWrite(motor2_forward, LOW);
        digitalWrite(motor2_backward, LOW);
        digitalWrite(red, HIGH);
        digitalWrite(green, LOW);
        break;
    }
    delay(5);
```

```
  }
}
```

- Continuously checks for available data on the Bluetooth serial connection.
- Reads an incoming character and prints it to the serial monitor.
- Depending on the character received:
  - '1': Activates the motors in a specific configuration and lights the green LED.
  - '2': Stops both motors and lights the red LED.
- Adds a small delay to debounce the signals.

Summary

- The code establishes a Bluetooth connection named "ESP32_Sandeep".
- It uses incoming Bluetooth commands to control the state of two motors and two LEDs.
- The motors can be turned on or off based on the received command ('1' or '2').

KLE Technological University | Creating Value Leveraging Knowledge

Centre For Engineering Education Research

PRAYOG SHISHIR 2025

Team No-K10

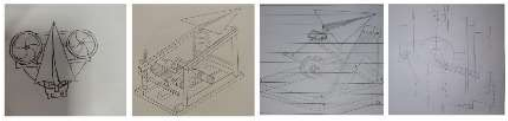
Roll No.	Name
1121	PRIYANKA K
1122	GAYATRI S
1123	PRATHEEK
1126	SANDEEP
1145	SONIKA S K

Title: PAPER PLANE LAUNCHING MACHINE

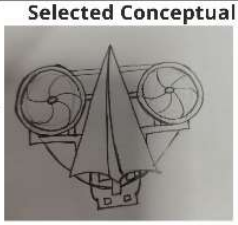
Problem Definition:

Design a portable paper plane launching machine with dimensions of 2ft x 2ft x 2ft, weighing 1.5kg, and costing approximately ₹3000. The machine should include features like launching alerts, on/off indicators, and mechanisms for error handling and plane stabilization. A high level of automation, using transparent acrylic sheets and ease of portability which should be aesthetically good looking.

Conceptual Designs



Selected Conceptual Design



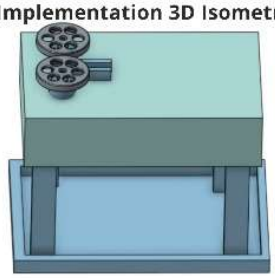
Mechanisms used in the Selected Design are:

1. Motorized Launch Mechanism
2. Control System using ESP32
3. Guiding Mechanism

Motor Sizing and Battery/Adapter Selection

Motor	Torque required	RPM	Power required	Selected adaptor
DC Motor	0.1	1000 rpm	7.2 W	12V-1A

Virtual Implementation 3D Isometric Design



Sprint -1

Acrylic Structural Frame

Bolt and Nut with Hinged Joints

The subsystem acts as a stable base structure for launching machine. The hinged joints provide stiffness, while the transparent panels ensure easy visibility of internal mechanisms during operation.

Sprint -2

Wheel-Based Motion Subsystem

DC Motors with Wheel Assembly

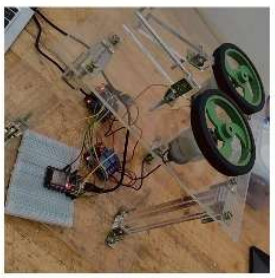
It enables controlled motion for the launching machine. The DC motors power the wheels to achieve smooth and precise movement, providing mobility

Sprint -3

Motion Control and Circuit Integration

ESP32-Based Motor Control

This integrates a microcontroller with the motorized wheels to enable programmable and precise motion control.



Final Prototype