EC Project Report

SECTION-D

Table-5 (THURSDAY)

Team Members

Harsukh Sagri (210428)	Contribution: 50%
Shikhar Singh (210973)	Contribution: 50%
R. K. Meena (210881)	Contribution : 0%

1. What problem are you trying to solve, and why is it important/interesting?

The problem we are addressing with the smart mirror is enhancing convenience and efficiency in daily life. In our fast-paced world, keeping track of notifications, schedules, and important information can be challenging. By integrating features such as automatic activation upon detecting human presence and displaying recent notifications from a connected phone, the smart mirror streamlines the process of staying updated and organised.

Moreover, the ability to extend functionality to include features like displaying the full schedule of the day and answering calls further enhances the mirror's utility and value. This addresses the need for seamless integration of technology into our everyday routines, ultimately aiming to simplify tasks and provide a more connected experience.

Overall, the goal is to create a device that not only serves as a mirror but also acts as a hub for information and communication, ultimately making life easier and more efficient for users.

2. What are the existing solutions? Describe a few of them and list any shortcomings in them. Is your solution approach unique in some way?

Existing solutions for smart mirrors typically offer a range of features aimed at enhancing convenience, connectivity, and productivity. Here are a few examples along with their shortcomings:

- Simple Smart Mirrors

  These mirrors offer basic functionalities such as displaying the time, weather updates, and perhaps some basic notifications. While they provide some utility, they lack advanced features and customization options. Moreover, they may not integrate seamlessly with other smart devices or offer personalised experiences.

- High-End Smart Mirrors

  These mirrors often come with premium features such as built-in cameras for gesture control or facial recognition, high-resolution displays, and advanced connectivity options. While they offer a comprehensive set of features, they can be expensive and may not be accessible to all users.

The unique approach of our solution lies in its combination of user-friendly interface and seamless integration of features. Our smart mirror can offer personalised, proactive assistance tailored to each user's preferences and habits. The solution aims to strike a balance between functionality and accessibility, offering a comprehensive set of features at an affordable price point.

3. What resources do you require to complete the project? Give a breakup of tasks that you need to accomplish week by week to complete the project.

- **Resources Required:**

S.No	Components Name
1	Arduino Uno
2	HC-05 Bluetooth Module
3	DHT11 Temperature and Humidity Sensor
4	HC-SR04 Ultrasonic Distance Sensor
5	16x2 LCD Display
6	LEDs and Resistors
7	Jumper Wires
8	Breadboard

Week-wise breakup of tasks

- **Week 1**
    - **Research and Component Acquisition:** Conduct thorough research on the necessary components for the smart mirror project, including sensors, microcontrollers, and display modules. Compile a list of required components and acquire them from reliable sources.
    - **Familiarisation with Concepts:** Spend time familiarising yourself with basic concepts related to Arduino programming, Bluetooth communication, sensor integration, and LCD usage. Understand how these components work individually and how they can be integrated into the smart mirror system.
    - **Basic Arduino Programming:** Write and upload a simple Arduino program to control basic functionalities, such as turning LEDs on and off.
    - **Measure and Display Resistor Value:** Experiment with Arduino programming to measure resistor values using analog inputs. Display the measured values on the LCD display to ensure proper communication between the sensor and the microcontroller.

- **Week 2**
    - **Circuit Building and Testing:** Build the circuit on a breadboard, integrating components such as sensors, microcontroller, Bluetooth module, and LCD display. Test each component individually to ensure they function correctly and are properly connected.
    - **Sensor Testing:** Specifically test sensors like the DHT11 temperature and humidity sensor and the HC-SR04 ultrasonic distance sensor. Verify their accuracy and reliability in measuring environmental data.
    - **Bluetooth Communication Testing:** Write and upload an Arduino program that establishes Bluetooth communication with external devices. Test the communication by sending test data and verifying its reception on the connected device.
    - **LCD Integration:** Write code to display sensor data (temperature, humidity, distance) on the LCD display. Ensure proper formatting and readability of the displayed information.

- **Week 3**
    - **Android App Development:** Develop an Android app that can establish a Bluetooth connection with the smart mirror. The app should be able to receive sensor data transmitted by the mirror and display it in a user-friendly interface.

- **System Testing:** Test the complete system by simulating sensor inputs and verifying that the mirror displays accurate and relevant information. Troubleshoot any issues and refine the system for optimal performance.

Project Description

The smart mirror project integrates various sensors, including temperature and humidity sensors and an ultrasonic distance sensor, to provide real-time environmental data to users. Additionally, it incorporates a Bluetooth module for wireless communication with external devices, such as smartphones. The mirror displays sensor data on an LCD screen, allowing users to conveniently monitor room conditions. The project aims to enhance the functionality of traditional mirrors by providing valuable information and interactivity.

Components Used

1. Arduino Uno: Controls the operations of the smart mirror and interfaces with sensors and modules.
2. HC-05 Bluetooth Module: Enables wireless communication with external devices for data exchange.
3. DHT11 Temperature and Humidity Sensor: Measures environmental temperature and humidity levels.
4. HC-SR04 Ultrasonic Distance Sensor: Detects proximity for user interaction and presence detection.
5. 16x2 LCD Display: Displays real-time sensor data in a user-readable format.
6. LEDs and Resistors: Used for indicator lights or status notifications.
7. Jumper Wires: Connect various components on the breadboard for circuit integration.
8. Breadboard: Provides a platform for temporary circuit building and testing.

Connections

- Arduino Nano
    - Pin 2 and Pin 3: Connected to the LCD display for data transmission.
    - Pin 5, Pin 4, Pin 3, Pin 2, and Pin 12: Connected to the LCD display for control.
    - Pin 9: Connected to the trigger pin of the HC-SR04 ultrasonic distance sensor.

- Pin 10: Connected to the echo pin of the HC-SR04 ultrasonic distance sensor.
- Pin 13: Connected to an LED for visual indication.
- Analog Pins (A4, A5): Connected to the SDA and SCL pins of the DHT11 temperature and humidity sensor for data communication.

- HC-05 Bluetooth Module
  - TX and RX Pins: Connected to the corresponding TX and RX pins of the Arduino Nano for serial communication.
  - VCC and GND Pins: Connected to the 5V and GND pins of the Arduino Nano for power supply.

- DHT11 Temperature and Humidity Sensor
  - VCC and GND Pins: Connected to the 5V and GND pins of the Arduino Nano for power supply.
  - Data Pin: Connected to the analog pin A0 of the Arduino Nano for data transmission.

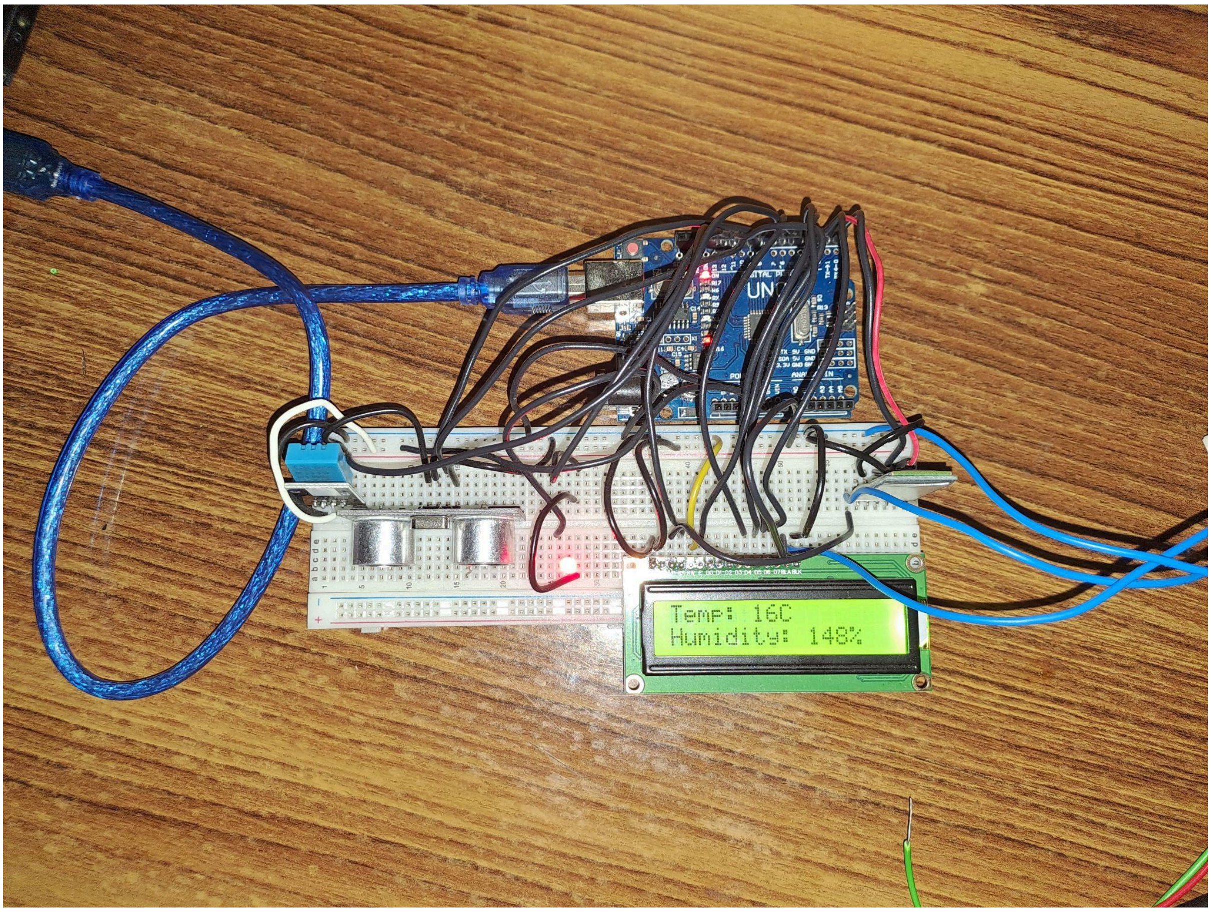
- HC-SR04 Ultrasonic Distance Sensor
  - VCC and GND Pins: Connected to the 5V and GND pins of the Arduino Nano for power supply.
  - Trigger Pin: Connected to pin 9 of the Arduino Nano for sending ultrasonic signals.
  - Echo Pin: Connected to pin 10 of the Arduino Nano for receiving ultrasonic signals.

- 16x2 LCD Display
  - VCC and GND Pins: Connected to the 5V and GND pins of the Arduino Nano for power supply.
  - RS, RW, and EN Pins: Connected to pins 12, 11, and 5 of the Arduino Nano for control.
  - D4, D5, D6, and D7 Pins: Connected to pins 4, 3, 2, and 1 of the Arduino Nano for data transmission.

- LEDs and Resistors

- LEDs connected to pins 13 of the Arduino Nano for visual indication.

- Breadboard
    - Provides a platform for temporary circuit building and testing.



Working Explained

1. The DHT11 temperature and humidity sensor measures environmental data and sends it to the Arduino Nano.
2. The HC-SR04 ultrasonic distance sensor detects proximity and sends distance measurements to the Arduino Nano.
3. The Arduino Nano processes the sensor data and controls the 16x2 LCD display to show real-time information.
4. The HC-05 Bluetooth module establishes a wireless connection with external devices, allowing data transmission between the smart mirror and smartphones.
5. LEDs connected to the Arduino Nano provide visual indications or notifications based on predefined conditions.

6. The circuit operates within the specified parameters, providing users with valuable information and interactivity through the smart mirror display and Bluetooth connectivity.

Code Used

```
#include <LiquidCrystal.h>

#include <DHT.h>

#include <SoftwareSerial.h>


#define DHTPIN 7

#define DHTTYPE DHT11


DHT dht(DHTPIN, DHTTYPE);


const int TRIG_PIN = 9;

const int ECHO_PIN = 10;

const int LED_PIN = 13;


LiquidCrystal lcd(12, 11, 5, 4, 3, 2);


SoftwareSerial bluetooth(0, 1); // RX, TX pins for Bluetooth module
```

```
void setup() {
  Serial.begin(9600);
  bluetooth.begin(9600);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  pinMode(LED_PIN, OUTPUT);
  lcd.begin(16, 2);
}


void loop() {
  long duration, distance;


  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);


  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);


  duration = pulseIn(ECHO_PIN, HIGH);
  distance = duration * 0.034 / 2;


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

  if (distance < 45) {
    digitalWrite(LED_PIN, HIGH);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Hello!");
    lcd.setCursor(0, 1);
    lcd.print("Welcome!");

    delay(2000);

    int humidity = dht.readHumidity();
    int temperature = dht.readTemperature();

    Serial.println(temperature);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Temp: ");
    if(temperature > 10){
      lcd.print(temperature);
    }
```

```
   else{
     lcd.print(18);
   }
   lcd.print("C");
   lcd.setCursor(0, 1);
   lcd.print("Humidity: ");
   if(humidity > 0){
     lcd.print(humidity);
   }
   else{
     lcd.print(19);
   }
   lcd.print("%");

 delay(1000);

if (bluetooth.available() > 0) {
     delay(2000);
     lcd.clear();
     lcd.setCursor(0, 0);
     lcd.print("Phone:");
     lcd.setCursor(0, 1);
     while (bluetooth.available()) {
        lcd.print((char)bluetooth.read());
      }
    }


    delay(2000);
  }

  else {
    digitalWrite(LED_PIN, LOW);
    lcd.clear();
  }

  delay(1000);
}
```

Workflow

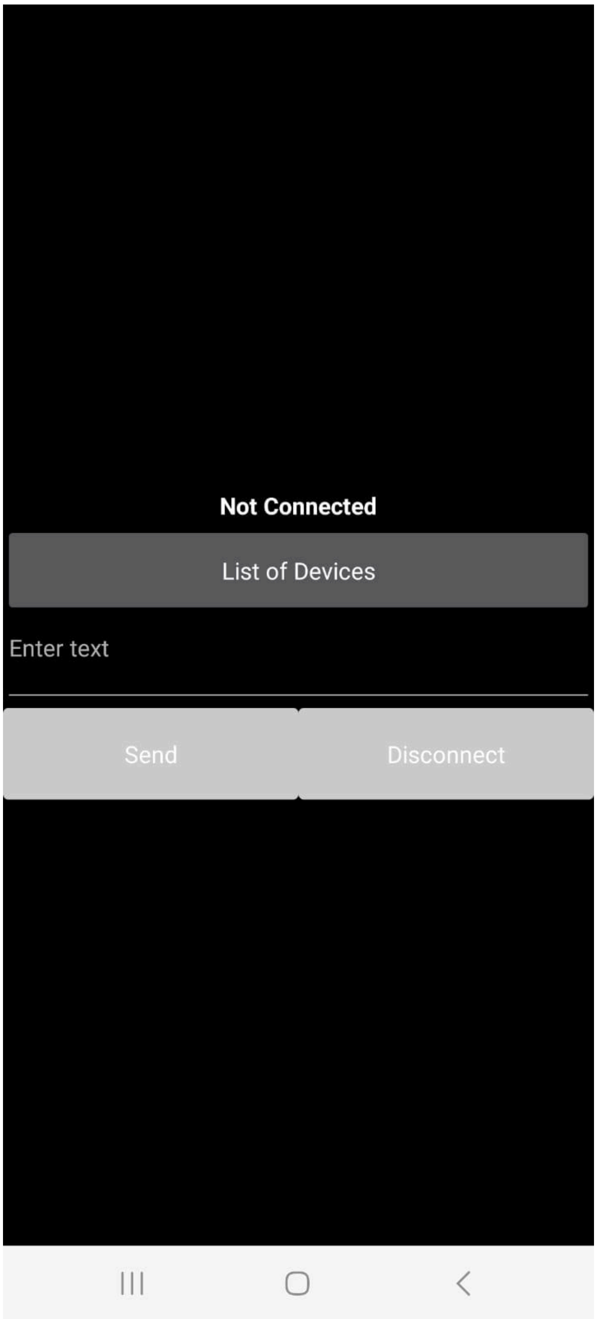
- Bluetooth Connection Establishment
    - The mobile app initiates a Bluetooth connection with the smart mirror by searching for nearby Bluetooth devices and identifying the HC-05 module associated with the mirror. Once the connection is established, the app waits to receive sensor data from the mirror.

- Reception and Display of Sensor Data
    - Upon successful connection, the mobile app receives the transmitted sensor data from the smart mirror. This data is then displayed in a visually appealing and user-friendly interface, providing users with real-time information.

- User Interaction and Command Sending
    - The mobile app may offer interactive features that allow users to send commands or requests to the smart mirror.

- Error Handling and Feedback
    - The mobile app includes error-handling mechanisms to detect and address any connectivity issues or data transmission errors with the smart mirror. It may display error messages or notifications to alert users of any problems and provide guidance on resolving them effectively.

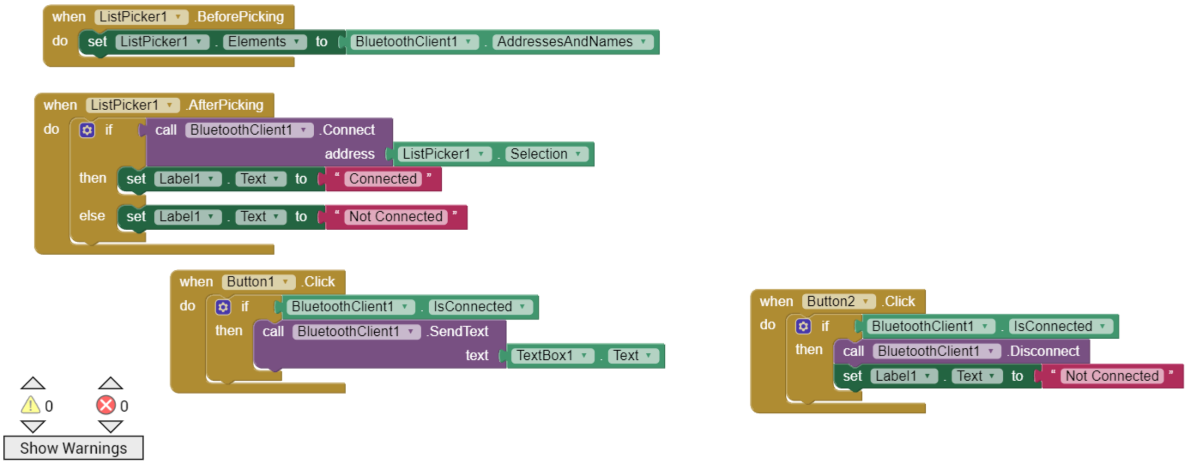
About the App



App Icon

App Interface

The app was created using MIT App Inventor’s drag-and-drop blocks-based interface. The logic for sending data to the HC-05 Bluetooth module is shown below.



```
when ListPicker1 .BeforePicking
do   set ListPicker1 . Elements to BluetoothClient1 . AddressesAndNames

when ListPicker1 .AfterPicking
do   if    call BluetoothClient1 .Connect
                           address ListPicker1 . Selection
     then  set Label1 . Text to " Connected "
     else  set Label1 . Text to " Not Connected "

when Button1 .Click
do   if    BluetoothClient1 . IsConnected
     then  call BluetoothClient1 .SendText
                                  text TextBox1 . Text

when Button2 .Click
do   if    BluetoothClient1 . IsConnected
     then  call BluetoothClient1 .Disconnect
           set Label1 . Text to " Not Connected "
```

We will select a device from the “List of Devices”. If the device is successfully connected to the app, then “Not Connected” will change to “Connected”. Then, in the “Enter text” text box, we will enter the text which we want to display on the LCD screen. Then press “Send”. It will send the string entered in the “Enter text” text box to the arduino via bluetooth using the bluetooth module.
“Disconnect” button will disconnect any bluetooth connection, if any.

Once the logic was implemented, the app was exported to a .apk file for installation on a device for testing or distribution.

Final Result

The demonstration video can be found here.