MINI-PROJECT: WASHING MACHINE

GROUP 2

MCTA 3203

SEMESTER 1 2024/2025

MECHATRONICS SYSTEM INTEGRATION

DATE OF SUBMISSION: 16/1/2024

	NAME	**MATRIC NUMBER**
1.	AKMAL ZARIF BIN NAJIB	2211971
2.	AMIR FARHAN BIN GHAFFAR	2115617
3.	AMIRAH HUDA BINTI JAMALULAIL ASRI	2210776
4.	AMIRUL AIZAD BIN AMIR	2211263
5.	ANAS BIN BADRULZAMAN	2219945

TABLE OF CONTENTS

ABSTRACT

This project demonstrates the integration of key components learned throughout the course to develop a functional washing machine prototype. The system combines Arduino and WeMos D1 microcontrollers for IoT capabilities to create a seamless blend of hardware control and wireless communication, respectively. The prototype's physical operations, such as the opening/closing of the lid by the servo motor and the rotating part of the washing by the DC motor are managed by the Arduino microcontroller. Simultaneously, WeMos D1 provides real-time updates on the washing duration and status to the user via Wi-Fi communication. The successful implementation of this prototype highlights the effectiveness of integrating microcontrollers and IoT technology to develop a functional automated system.

INTRODUCTION

In modern households, washing machines are without a doubt one of the most vital appliances you can own. Washing clothes is such a laborious and time-consuming task without a washing machine. These appliances ensure thorough cleaning whilst significantly reducing the time and effort required for laundry. The importance of washing machines lies not only in their functionality but also in their ability to simplify and streamline daily routines. Top loaders, front loaders, semi and fully automatic washing machines are some of the different types of washing machines. Despite their widespread use in modern households, many traditional washing machines fail to address key limitations such as efficient resource management and real-time user feedback. In addition to that, they do not provide a way to keep users engaged during the waiting period of the washing process. These limitations reduce the user experience and create opportunities for enhancing functionality and convenience.

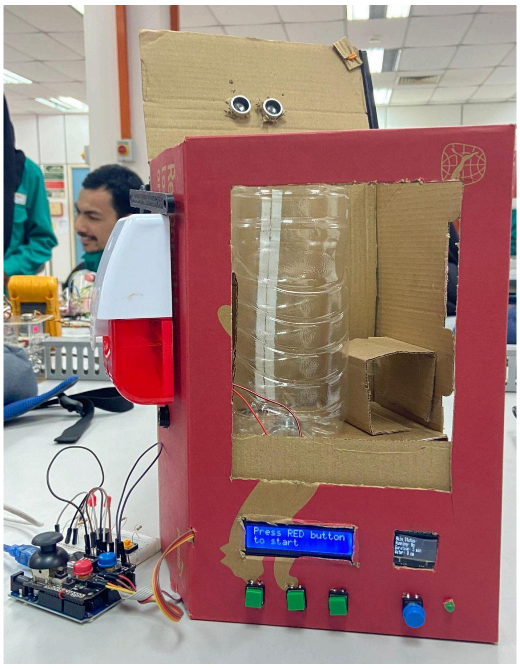
The objective of this project is to design and develop a washing machine prototype that incorporates all of the components learnt throughout the course. Our washing machine prototype addresses these limitations by integrating smart technologies and user-centric features. The system includes an ultrasonic sensor to detect the load size and adjust water levels accordingly. On the other hand, a Pixy Camera is used to identify between white shirts and coloured shirts for garment-specific sorting. The user receives real-time updates on their phone based on which coloured clothing is detected as well as feedback on the status of the washing process. This was done with the integration of WeMos D1 and Raspberry Pi microcontrollers. Additionally, instead of controlling the system from their phones, users can also adjust the settings of their washing by the buttons and LCD screen at the front of the device. Lastly, the display screen can also be used as a mini-gam for users to interact during the waiting period.

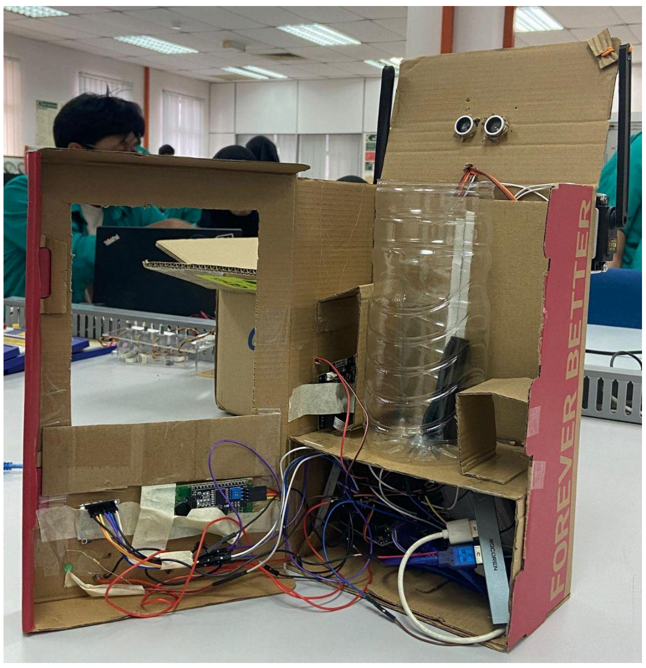
MATERIALS AND EQUIPMENT

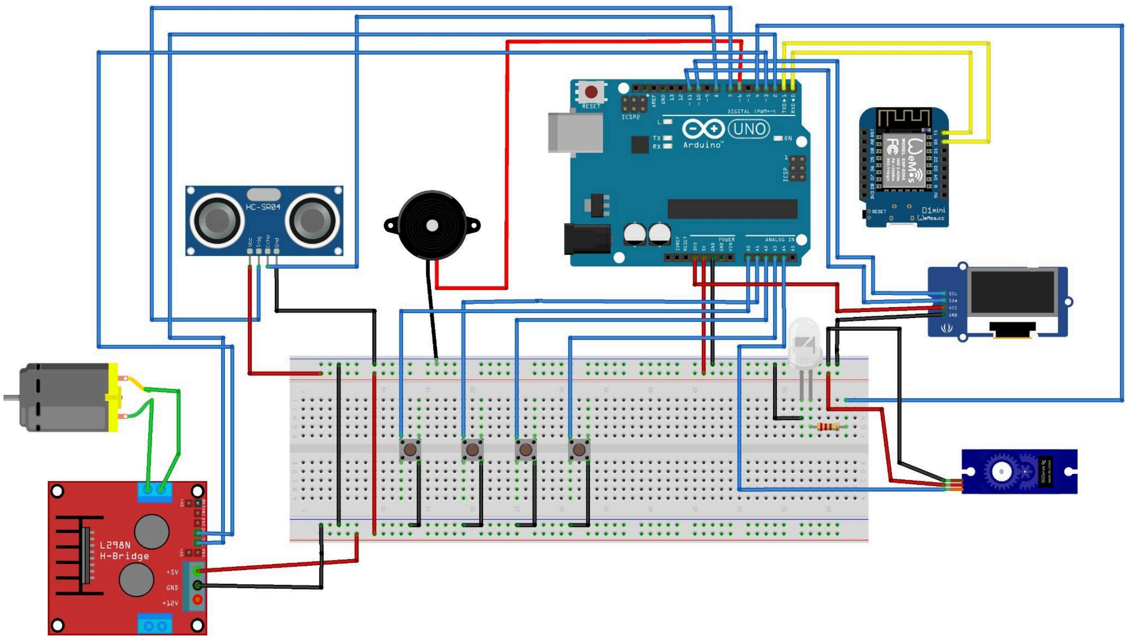
- Arduino Uno
- DFRobot Input Shield
- Buzzer
- LEDs
- Buttons
- Breadboard, Jumper wires and resistors
- Pixy Camera
- LCDs
- Wemos D1 Mini
- Raspberry Pi
- Battery
- Ultrasonic sensor

- Servo motor (MG945)
- DC motor
- Motor driver
- Bottle
- Cardboard box

EXPERIMENTAL SETUP







METHODOLOGY

The washing machine system was designed to integrate IoT functionalities, dividing tasks between the Arduino Uno for sensor interfacing and data processing and the Wemos D1 Mini for wireless communication. This architecture ensures seamless interaction between hardware components and remote monitoring capabilities.

1. System Design

Key components were chosen based on their functionality and compatibility:

- Pixy Camera: For white shirt recognition.
- Ultrasonic Sensor: For water level measurement.
- DC Motor: For mechanical operations such as rotation.
- Servo Motor: For the top-loading of clothes into the machine.

2. Hardware Setup

- **Assembly:** The hardware components were assembled systematically:
  - The **Arduino Uno** was connected to sensors, LEDs, and the buzzer using the DFRobot Input Shield and jumper wires.
  - The **Wemos D1 Mini** was integrated for Wi-Fi-enabled communication.
  - The **Motor Driver** was used to control the DC motor.
- **Power Management:** A power bank was utilized to provide a stable power supply to the system, ensuring portability and reliability.

3. Software Development

- **Arduino Programming:**
  - Code was developed to interface with the ultrasonic sensor, Pixy camera, push buttons, and serial monitor via RX and TX communication
  - Logic for controlling the DC motor, servo motor and buzzer through the button was implemented.

- **Wemos D1 Mini Programming:**
  - Wireless communication was enabled using Wi-Fi protocols, allowing remote data transmission and control.

- **Integration with IoT:**
  - The Blynk software in smartphones was configured to interact with the Telegram bot using a webhook widget for real-time monitoring and control.
- **Raspberry Pi:**
  - Raspberry Pi is used as a local Blynk server to receive the data from the other microcontrollers (In this project we used Wemos D1) and transmit the data to the Blynk app on smartphones.

4. Functional Modules

- **Sensor Calibration**:
    - The ultrasonic sensor was calibrated to measure water levels accurately.
    - The Pixy camera was configured to recognize and detect white shirts.
- **Motor Control**:
    - Programming logic was developed to control the duration of the DC motor using the motor driver.
- **User Interface**:
    - LCDs were used to display system status and user prompts. (mode, time reminding and mini-game)
    - Buttons and LEDs were configured to allow user interaction and system feedback.
    - A buzzer provided auditory alerts for important events.

5. Testing and Troubleshooting

- **Unit Testing**:
    - Each component, such as the ultrasonic sensor and Pixy camera, was tested individually to verify functionality.
    - The DC motor was tested for correct operation.
- **System Integration Testing**:
    - All components were integrated, and the system was tested to ensure smooth operation of the washing machine.
- **IoT Functionality Testing**:
    - Communication between the Wemos D1 Mini and a smartphone or Telegram bot was verified for data transmission and remote control.
- **Energy Efficiency Tests**:
    - The system's power consumption was measured to ensure energy-efficient operation.

6. Iteration and Optimization

- Challenges encountered during the development process, such as communication delays or sensor inaccuracies, were identified and resolved through iterative testing and modifications.
- Based on testing results, design improvements were implemented to enhance system reliability and performance.

PROCEDURE

Hardware Setup

1. **Assemble the Components**:
   - Connect **Wemos D1 Mini**, **Arduino UNO**, and **Raspberry Pi**.
   - Attach peripheral components: **LEDs, buzzer, servo motor, ultrasonic sensor, push buttons, and DC motor**.
2. **Power Configuration**:
   - Ensure all components receive appropriate voltage from a power source.
   - Connect **Wemos D1 Mini** and **Raspberry Pi** to Wi-Fi.
3. **Wiring and Circuit Connection**:
   - Connect **Wemos D1 Mini** to **Arduino UNO via Serial Communication (TX/RX)**.
   - Interface **ultrasonic sensor** with Arduino to measure water levels.
   - Attach a **servo motor** to control the washing machine lid.
   - Connect the **DC motor** to rotate the washing drum.
   - Wire **push buttons** for manual control inputs.
   - Set up **LED indicators and buzzers** for status notifications.

Software Setup

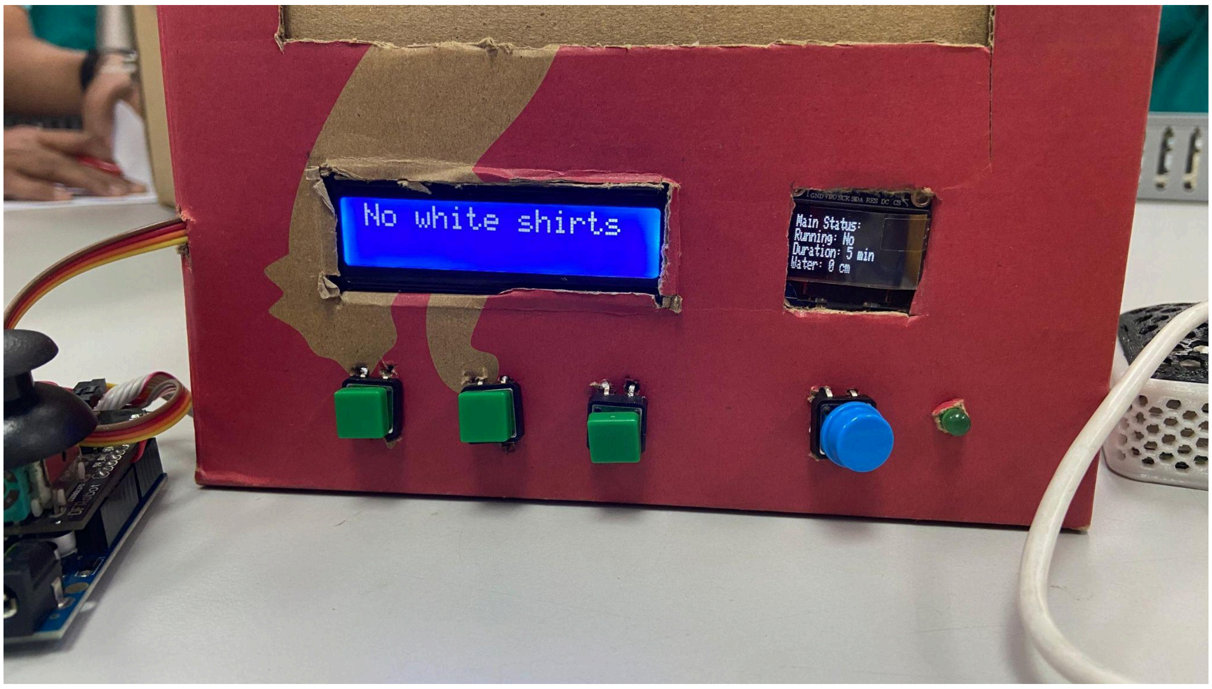
4. **Programming Wemos D1 Mini (Wi-Fi & Blynk Communication)**:
   - Develop a script to communicate with **Blynk Cloud** and send/receive washing machine data.
   - Use **Webhook** in Blynk to integrate with **Telegram Bot** for user notifications.
   - Send control signals to **Arduino UNO** based on virtual buttons in the Blynk app.
5. **Programming Arduino UNO (Machine Control)**:
   - Read inputs from **push buttons** and **ultrasonic sensors**.
   - Control outputs such as **LEDs, buzzers, servo motors, and DC motors** based on user commands.
   - Process serial data from Wemos D1 Mini to execute washing machine operations.
6. **Configuring Raspberry Pi as a Local Blynk Server**:
   - Install and set up the **Blynk Server** on Raspberry Pi.
   - Configure Wemos D1 Mini to communicate with the local Blynk server.
   - Ensure the **Blynk app** on the mobile device is linked to the server.

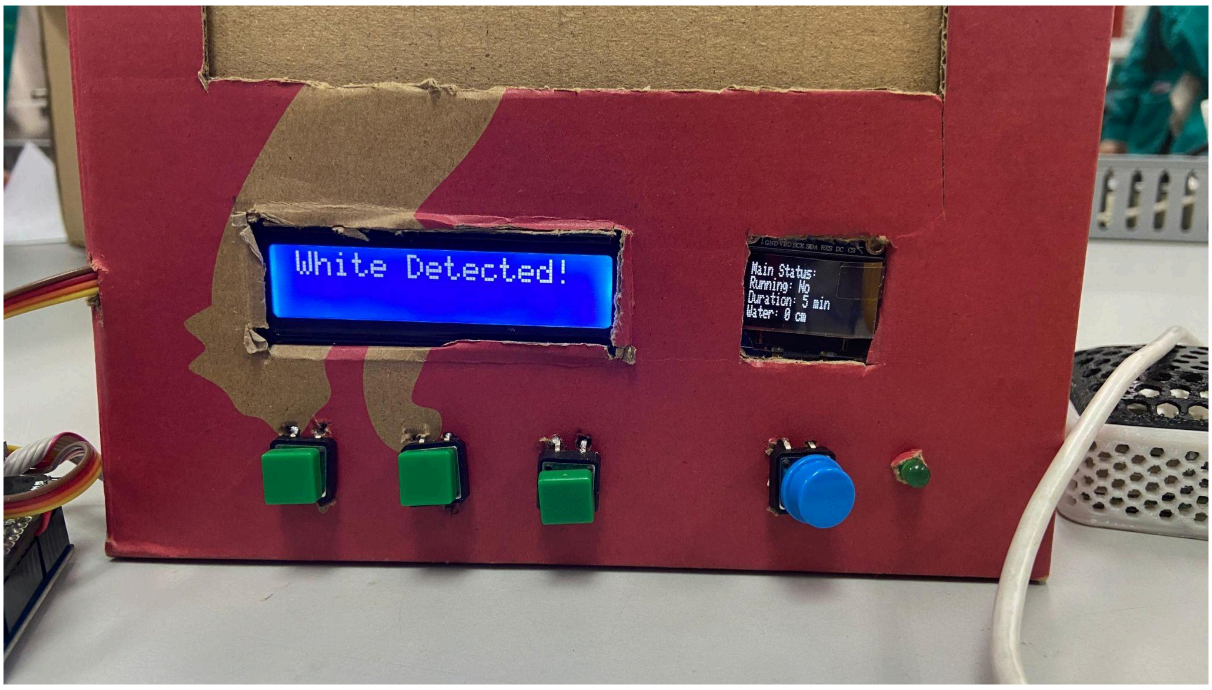
Testing and Implementation

7. **Testing the Washing Machine System**:
   - Run test cases for each component (**LED, buzzer, motor, ultrasonic sensor, buttons**).
   - Verify **water level detection** using the ultrasonic sensor.

- Check **motor operation** for drum rotation.
    - Ensure the **servo motor opens/closes the door correctly**.
8. **Testing Wi-Fi and Blynk Communication**:
    - Send commands from the **Blynk app** and validate Arduino UNO responses.
    - Confirm that **notifications are sent to Telegram** when washing is complete.
9. **Final Adjustments and Optimization**:
    - Adjust timing, thresholds, and response delays for smooth operation.
    - Perform **multiple washing cycles** to ensure reliability.
    - Verify **real-time monitoring and remote control through Blynk**.

RESULTS





DISCUSSION

Hardware and Communication Integration

The architecture is designed to partition tasks effectively between the Arduino Uno and Wemos D1

- **Arduino Uno as the Core Processor**:
  The Arduino Uno is responsible for connecting with sensors and performing basic data processing. For example, it manages data from the ultrasonic sensor and Pixy camera, performing basic functions like motor operation and buzzer triggering. The DFRobot Input Shield streamlines connections, promoting an organized hardware arrangement.
- **Wemos D1 Mini for IoT**:
  The Wemos D1 Mini serves as the communication link, utilizing its Wi-Fi feature to facilitate real-time monitoring and control through a Telegram bot. This separation of functions reduces the processing burden on the Arduino and consolidates IoT-related tasks within the Wemos.
- **Motor Control via Motor Driver**:
  The motor driver connects to the Arduino Uno, obtaining signals to manage the DC motor. This guarantees accurate mechanical functions, such as the timing of the rotation cycle, which is crucial for both washing and drying.

2. POWER MANAGEMENT

The system's power needs are addressed using a portable power bank, which provides a reliable power source. This approach supports the goal of portability without sacrificing functionality.

3. SOFTWARE AND IOT INTEGRATION

- **Arduino Programming:**
  The programming logic incorporates multiple components into a unified control system. It ensures that sensors, motors, and output devices (LCD, buzzer) function cohesively.
- **IoT via Wemos D1 Mini**:
  The Wemos D1 Mini's programming supports Wi-Fi protocols, enabling remote interactions. For example, users can receive alerts or send commands through a Telegram bot, linking the physical system to a smartphone or other IoT platforms.
- **User Interface**:
  The incorporation of LCDs, LEDs, buttons, and a buzzer provides an engaging experience for the user. These elements offer immediate feedback and control, including mode selection or adjustments to cycle time.

4. TESTING AND OPTIMIZATION

- **Iterative Testing:**
  By systematically testing components (unit testing) and their integration (system testing), potential issues such as communication delays or sensor inaccuracies were identified early and resolved.
- **Energy Efficiency:**
  Through power consumption measurement and usage pattern optimization, the design achieves a balance between functionality and energy efficiency, a vital element for IoT-enabled devices.

5. SOFTWARE

The code on the Arduino Uno board:

```
#include <SPI.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <Servo.h>

// OLED Display dimensions
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 32

// OLED Software SPI Pins
#define OLED_MOSI   9
#define OLED_CLK    10
#define OLED_DC     11
#define OLED_CS     12
#define OLED_RESET  13

// Initialize OLED display using software SPI
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, OLED_MOSI,
OLED_CLK, OLED_DC, OLED_RESET, OLED_CS);

// Pins for components
#define GREEN_LED_PIN 4
#define MOTOR_IN1_PIN 2
#define MOTOR_IN2_PIN 3
#define BUZZER_PIN 6
#define BUTTON_DECREASE A0
#define BUTTON_INCREASE A1
#define BUTTON_MODE A2
#define BUTTON_START_STOP A3
```

```
#define TRIG_PIN 7
#define ECHO_PIN 8
#define SERVO_PIN A4 // Servo motor for door control

// Servo object
Servo doorServo;

// Variables
int mode = 0; // 0: Main Status, 1: Duration, 2: Water Level, 3:
Auto/Manual Mode
bool running = false;
int duration = 10;      // Default washing duration in minutes
int waterLevel = 0;     // Measured water level in cm
int maxWaterLevel = 30; // Maximum water level in cm
bool autoMode = true;   // Auto mode enabled by default

unsigned long lastUpdateTime = 0;        // For live countdown
updates
unsigned long pressStartTime = 0;        // For long-press detection
unsigned long lastWaterLevelCheck = 0;   // For periodic water level
measurement

void setup() {
  // Pin Modes
  pinMode(GREEN_LED_PIN, OUTPUT);
  pinMode(MOTOR_IN1_PIN, OUTPUT);
  pinMode(MOTOR_IN2_PIN, OUTPUT);
  pinMode(BUZZER_PIN, OUTPUT);
  pinMode(BUTTON_DECREASE, INPUT_PULLUP);
  pinMode(BUTTON_INCREASE, INPUT_PULLUP);
  pinMode(BUTTON_MODE, INPUT_PULLUP);
  pinMode(BUTTON_START_STOP, INPUT_PULLUP);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);

  // Attach servo to pin
  doorServo.attach(SERVO_PIN);
  doorServo.write(0); // Initially set to open position

  // Initialize Serial
  Serial.begin(9600);

  // Initialize OLED
  if (!display.begin(SSD1306_SWITCHCAPVCC)) {
    Serial.println(F("OLED initialization failed"));
    for (;;); // Halt if initialization fails
  }

  // Initial OLED display
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(0, 0);
  display.println("Washing Machine Ready");
  display.display();

  // Startup buzzer indication
  digitalWrite(BUZZER_PIN, HIGH);
  delay(500);
  digitalWrite(BUZZER_PIN, LOW);
  delay(500);
}

void loop() {
  // Handle button presses
  handleButtons();

  // Run the washing machine if it's running
  if (running) {
    runWashingMachine();
  }

  // Periodically measure water level
  if (millis() - lastWaterLevelCheck >= 1000) { // Measure every
second
    lastWaterLevelCheck = millis();
    measureWaterLevel();
    if (!running) {
      updateAutoDuration(); // Update duration in Auto mode
      displayStatus();      // Update the OLED with live water level
    }

    // Trigger buzzer if water level exceeds maximum
    if (waterLevel > maxWaterLevel) {
      triggerBuzzer();
    }
  }
}

// Function to handle button presses
void handleButtons() {
  // Button: Change Mode
  if (digitalRead(BUTTON_MODE) == LOW) {
    mode = (mode + 1) % 4; // Cycle through 4 modes
    buttonFeedback();
    displayStatus();
    delay(200); // Debounce delay
  }

  // Button: Decrease Value
  if (digitalRead(BUTTON_DECREASE) == LOW) {
    if (mode == 3) { // Toggle Auto/Manual mode in Mode 3
      autoMode = false;
      buttonFeedback();
    } else if (!autoMode && mode == 1 && duration > 1) { // Decrease
duration in Manual mode
      duration--;
      buttonFeedback();
    }
    displayStatus();
    delay(200); // Debounce delay
  }

  // Button: Increase Value
  if (digitalRead(BUTTON_INCREASE) == LOW) {
    if (mode == 3) { // Toggle Auto/Manual mode in Mode 3
      autoMode = true;
      buttonFeedback();
    } else if (!autoMode && mode == 1) { // Increase duration in
Manual mode
      duration++;
      buttonFeedback();
    }
    displayStatus();
    delay(200); // Debounce delay
  }

  // Button: Start/Stop or Abort
  if (digitalRead(BUTTON_START_STOP) == LOW) {
    if (pressStartTime == 0) {
      pressStartTime = millis(); // Record the time when button is
first pressed
    }

    if (millis() - pressStartTime > 2000) { // Long-press detected
(2 seconds)
      abortProcess();
    }
  } else {
    // If button is released before 2 seconds, toggle Start/Stop
    if (pressStartTime > 0 && millis() - pressStartTime <= 2000) {
      running = !running; // Toggle running state
      if (running) {
        startWashingSequence();
      } else {
        stopWashingSequence();
      }
      displayStatus();
    }
    pressStartTime = 0; // Reset press timer
  }
}

// Function to start the washing sequence
void startWashingSequence() {
  buttonFeedback();
  digitalWrite(GREEN_LED_PIN, HIGH); // Turn on green LED

  // Indication before starting
  for (int i = 0; i < 5; i++) {
    digitalWrite(BUZZER_PIN, HIGH);
    delay(100);
    digitalWrite(BUZZER_PIN, LOW);
    delay(100);
  }

  // Close the door
  doorServo.write(90);
  delay(1000); // Wait for the door to close

  // Print start message
  Serial.println("Washing Machine: START");
  Serial.print("Duration Set: ");
  Serial.print(duration);
  Serial.println(" minutes");
}

// Function to stop the washing sequence
void stopWashingSequence() {
  stopMotor(); // Ensure motor stops
  digitalWrite(GREEN_LED_PIN, LOW); // Turn off green LED

  // Indication after finishing
  for (int i = 0; i < 5; i++) {
    digitalWrite(BUZZER_PIN, HIGH);
    delay(100);
    digitalWrite(BUZZER_PIN, LOW);
    delay(100);
  }

  // Open the door
  doorServo.write(0);
  delay(1000); // Wait for the door to open

  // Print finish message
  Serial.println("Washing Machine: FINISH");
}

// Function to update duration in Auto mode
void updateAutoDuration() {
  if (autoMode) {
    duration = map(waterLevel, 0, maxWaterLevel, 5, 20); // Map
water level to duration
    duration = max(1, duration); // Ensure the duration is at least
1 minute
  }
}

// Function to measure water level using ultrasonic sensor
void measureWaterLevel() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  long duration = pulseIn(ECHO_PIN, HIGH);
  waterLevel = maxWaterLevel - (duration * 0.034 / 2); // Convert
duration to cm
  waterLevel = max(0, min(waterLevel, maxWaterLevel)); // Clamp to
valid range
}

// Function to simulate running the washing machine
void runWashingMachine() {
  unsigned long currentMillis = millis();

  // Update OLED every second
  if (currentMillis - lastUpdateTime >= 1000) {
    lastUpdateTime = currentMillis;

    if (duration > 0) {
      duration--;
      displayLiveCountdown();

      // Print remaining time
      Serial.print("Time Left: ");
      Serial.print(duration);
      Serial.println(" minutes");
    } else {
      running = false;
      stopWashingSequence();
      displayStatus();
    }
  }

  // Simulate motor running
  if (running) {
    startMotor();
    digitalWrite(GREEN_LED_PIN, HIGH); // Turn on the green LED when
running
  } else {
    stopMotor();
    digitalWrite(GREEN_LED_PIN, LOW); // Turn off the green LED when
stopped
  }
}
```

```
// Function to start the motor
void startMotor() {
  digitalWrite(MOTOR_IN1_PIN, HIGH);
  digitalWrite(MOTOR_IN2_PIN, LOW); // Forward direction
}

// Function to stop the motor
void stopMotor() {
  digitalWrite(MOTOR_IN1_PIN, LOW);
  digitalWrite(MOTOR_IN2_PIN, LOW); // Stop motor
}

// Function to abort the washing process
void abortProcess() {
  running = false;
  duration = 0; // Reset duration
  digitalWrite(GREEN_LED_PIN, LOW);
  stopMotor(); // Stop the motor when aborted

  // Indication of abort
  for (int i = 0; i < 3; i++) {
    digitalWrite(BUZZER_PIN, HIGH);
    delay(300);
    digitalWrite(BUZZER_PIN, LOW);
    delay(300);
  }

  doorServo.write(0); // Open the door
  display.clearDisplay();
  display.setCursor(0, 0);
  display.println("Process Aborted");
  display.display();
  delay(2000); // Pause for 2 seconds to show the message
  displayStatus(); // Return to main status
}

// Function to trigger the buzzer for warnings
void triggerBuzzer() {
  digitalWrite(BUZZER_PIN, HIGH);
  delay(500);
  digitalWrite(BUZZER_PIN, LOW);
}
```

```
// Function for button feedback
void buttonFeedback() {
  digitalWrite(BUZZER_PIN, HIGH);
  delay(100);
  digitalWrite(BUZZER_PIN, LOW);
}

// Function to update OLED display for live countdown
void displayLiveCountdown() {
  display.clearDisplay();
  display.setCursor(0, 0);
  display.println("Running...");
  display.setCursor(0, 10);
  display.print("Time Left: ");
  display.print(duration);
  display.println(" min");
  display.setCursor(0, 20);
  display.print("Water Level: ");
  display.print(waterLevel);
  display.println(" cm");
  display.display();
}

// Function to update OLED display for general status
void displayStatus() {
  display.clearDisplay();
  display.setCursor(0, 0);

  if (mode == 0) {
    display.println("Main Status:");
    display.print("Running: ");
    display.println(running ? "Yes" : "No");
    display.print("Duration: ");
    display.print(duration);
    display.println(" min");
    display.print("Water: ");
    display.print(waterLevel);
    display.println(" cm");
    display.print("Mode: ");
    display.println(autoMode ? "Auto" : "Manual");
  } else if (mode == 1) {
    display.println("Set Duration:");
    display.print(duration);
```

```
      display.println(" min");
    } else if (mode == 2) {
      display.println("Set Water Level:");
      display.print(waterLevel);
      display.println(" cm");
    } else if (mode == 3) {
      display.println("Mode:");
      display.println(autoMode ? "Auto" : "Manual");
      display.println("Press Buttons to Toggle");
    }

    display.display();
  }
```

The code on Wemos D1 mini:

```
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

// Blynk credentials
char auth[] = "QcAwv9ue9gP1qj2fHQJ9p1EaiPNamaWF";  // Your Blynk auth
token
char ssid[] = "Wifehere";          // Your Wi-Fi name
char pass[] = "amfawhere1";         // Your Wi-Fi password

// Virtual Pins
#define BLYNK_LCD V1
String machineStatus = "";


void setup() {
    Serial.begin(9600); // Initialize hardware serial for communication
with Arduino

    // Connect to Wi-Fi and Blynk
    Blynk.begin(auth, ssid, pass, IPAddress(192, 168, 192, 5), 8080);
    WiFi.begin(ssid, pass);
    while (WiFi.status() != WL_CONNECTED) {
        delay(1000);
        Serial.println("Connecting to Wi-Fi...");
    }
    Serial.println("Connected to Wi-Fi");
}

void loop() {
  Blynk.run();

  // Read data from Arduino UNO
  if (Serial.available()) {
    machineStatus = Serial.readStringUntil('\n'); // Read data until a
newline
    updateBlynkLCD(machineStatus);
  }
}

// Function to update Blynk LCD widget
void updateBlynkLCD(String status) {
  Blynk.virtualWrite(BLYNK_LCD, status); // Send status to Blynk LCD on
V1
}
```

- **MODES**
  1. **Main Status:**
     - Displays whether the machine is running, current duration, water level, and mode (auto/manual).
  2. **Duration Setting:**
     - Adjust washing duration in manual mode.
  3. **Water Level Setting:**
     - Displays current water level measured by the ultrasonic sensor.
  4. **Mode Selection:**
     - Toggles between auto and manual modes.

- **FUNCTIONAL HIGHLIGHT**
  1. **Washing Sequence:**
     - The washing machine starts with a beeping sequence and closes the door.
     - Simulated motor action and a countdown timer represent the washing process.
     - Stops automatically when the duration ends, beeping and opening the door.
  2. **Auto Mode:**
     - Automatically adjust washing duration based on water level using a linear mapping function.
  3. **Button Interaction:**
     - Buttons control mode switching, value adjustments, and start/stop functionality.
     - Long-pressing Button_Start_Stop aborts the process immediately.

4. **Water Level Monitoring:**
    - Periodically measures the water level.
    - Warns if water exceeds the maximum level by triggering the buzzer.
5. **OLED Updates:**
    - Displays relevant status and data for each mode.
    - Updates the countdown in real-time during operation.

CONCLUSION

The Mini Smart Washing Machine project successfully integrates IoT-based remote control and monitoring using Wemos D1 Mini, Arduino UNO, and Raspberry Pi. By utilizing Blynk and Telegram notifications, users can operate and receive updates from the washing machine remotely.

The system features:

- Automatic water level detection using an ultrasonic sensor.
- Secure door control with a servo motor.
- Real-time status updates via Blynk and Telegram.
- Manual or remote control through physical buttons and Blynk virtual buttons.
- Efficient washing process automation, reducing human intervention.

Through testing and implementation, the project demonstrates a functional, connected, and user-friendly smart washing machine. Future improvements could include enhanced automation, energy efficiency monitoring, and integration with AI for optimal washing cycles.

RECOMMENDATIONS

1. Enhance IoT Integration

- Explore the use of advanced IoT platforms for better data analytics and remote control capabilities.
- Incorporate real-time notifications and alerts for user convenience through a mobile app or web interface.
- Build a graphical user interface (GUI), to make it more attractive.

2. Expand Sensor Capabilities

- Add additional sensors, such as temperature or vibration sensors, to monitor the washing process more comprehensively.
- Implement water quality sensors to ensure optimal washing performance.
- Adding an RFID sensor to identify the user and store the usage in the cloud.
- Add a weight sensor (load cell) to predict the water level needed based on the weight.

3. Improve Energy Efficiency

- Integrate power optimization algorithms to minimize energy consumption during operation.
- Consider using solar panels or other renewable energy sources to power the system.

4. User Experience Enhancements

- Develop a more intuitive user interface with touchscreen controls or voice commands.

5. Prototype Scalability

- Design the system to accommodate larger loads or industrial-scale applications.
- Explore modular designs to make the system easier to upgrade and maintain.

6. Future Research and Development

- Investigate the integration of AI for predictive maintenance and smarter washing cycle recommendations.

ACKNOWLEDGEMENT

We would like to express our gratitude to Dr. Wahju Sediono, Dr. Ali Sophian, Dr. Zulkifli bin Zainal Abidin, my teaching assistant, and my peers for their invaluable help and support in finishing this project. Their advice, feedback, and experience have greatly influenced the level of quality and understanding of this work.

STUDENT'S DECLARATION

Certificate of Originality and Authenticity

This is to certify that we are responsible for the work submitted in this report, that **the original work** is our own except as specified in the references and acknowledgement, and that the original work contained herein has not been untaken or done by unspecified sources or persons.

We hereby certify that this report has **not been done by only one individual** and **all of us have contributed to the report**. The length of contribution to the reports by each individual is noted within this certificate.

We also hereby certify that we have **read** and **understand** the content of the total report and that no further improvement on the reports is needed from any of the individual contributors to the report.

We, therefore, agreed unanimously that this report shall be submitted for **marking** and this **final printed report** has been **verified by us.**

Signature: [illegible]	**Read**	/
Name: AKMAL ZARIF BIN NAJIB	**Understand**	/
Matric No: 2211971	**Agree**	/

Signature: [illegible]	**Read**	/
Name: AMIR FARHAN BIN GHAFFAR	**Understand**	/
Matric No: 2115617	**Agree**	/

Signature: [illegible]	**Read**	/
Name: AMIRAH HUDA BINTI JAMALULAIL ASRI	**Understand**	/
Matric No: 2210776	**Agree**	/

Signature: [illegible]	**Read**	/
Name: AMIRUL AIZAD BIN AMIR	**Understand**	/
Matric No: 2211263	**Agree**	/

Signature: [illegible]	**Read**	/
Name: ANAS BIN BADRULZAMAN	**Understand**	/
Matric No: 2219945	**Agree**	/