**A PROJECT REPORT ON**

**Automated Chemical Mixer**

SUBMITTED BY

**Ghananshu Gajendra Desale**

UNDER THE GUIDANCE OF

**Prof. Kapil Chaurasiya:**

****

**Department of Robotics & Mechatronics**

**INSTITUTE FOR DESIGN OF ELECTRICAL**

**MEASURING INSTRUMENTS**

SWATANTRAVEER TATYA TOPE MARG, CHUNABHATTI, SION, MUMBAI-400022

**CERTIFICATE**

This to certify that

**Ghananshu Gajendra Desale**

has undertaken this project **“Automated Chemical Mixer”** under our guidance as prescribed by the IDEMI, as the curriculum for the Diploma in Robotics & Mechatronics for the academic year 2024-2025.

**Project Guide: HOD**

**Prof. Kapil Chaurasiya Robotics & Mechatronics**

**Examiner**

**Date -\_\_\_\_\_\_\_\_\_\_\_**

**DECLARATION**

We declare that this written submission represents our ideas in our own words and where others’ ideas and words have been included. We have adequately cited and referenced the original sources. We also declare that we adhered to all principles of academic honesty and integrity and have not misinterpreted or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the institute and can also evoke penal action from then sources which have thus not been properly cited or from whom proper permission has not been taken when required.

**Ghananshu Desale**

**Acknowledgement**

It is our privilege to express our sincerest regard to our project guide **Prof. Kapil Chaurasiya,** who gave us the golden opportunity to do this wonderful project on the topic Automated Chemical Mixerfor his valuable inputs, guidance and encouragement, whole-hearted cooperation and constructive criticism throughout the duration of our project. We deeply express our sincere thanks to our HOD of Robotics & Mechatronics for encouraging and allowing our department premises for the partial fulfilment of the requirements. We also extend our deepest gratitude to all the esteemed faculty members and dedicated workers who credited their invaluable expertise and support to this project. Last but not the least we express our sincere thanks to all our friends and our parents who have patiently extended all sorts of help for accomplishing this undertaking.

**Project Details**

**Project Name: Automated Chemical Mixer**

**Group Members:**

|  |  |  |
| --- | --- | --- |
| **Sr. no** | **Name** | **Enrolment no.** |
| 1 | Vinayak Renukdas | 2111011 |
| 2 | Aditya Bodhi | 2211011 |
| 3 | Ghananshu Desale | 2212025 |

**Branch: Robotics & Mechatronics**

**Year: Third (Final)**

**Semester: Sixth**

**College Name: Institute for Design of Electrical Measurement Instruments**

**Assigned Faculties: Prof. Kapil Chaurasiya**

**Project Duration: January 2025 – June 2025**

**Table of content**

|  |  |  |
| --- | --- | --- |
| **Sr. no** | **Content** | **Page no.** |
| 1 | Table of content | 6 |
| 3 | Aim of the project | 7 |
| 4 | Key factors of consideration | 8 |
| 5 | Process of making | 9 |
| 6 | Block diagram | 11 |
| 7 | Electronics components | 12 |
| 7.1 | ESP 32 devkit V1 | 12 |
| 7.2 | L298N motor driver module | 15 |
| 7.3 | 16 x 2 LCD module | 17 |
| 7.4 | I2C module | 19 |
| 7.5 | 4 x 4 matrix keypad module | 21 |
| 7.6 | GPIO expander module | 23 |
| 7.7 | Relay module | 25 |
| 7.8 | R385 pump | 27 |
| 7.9 | Geared motor | 29 |
| 8 | Schematic | 31 |
| 9 | PCB design and fabrication | 32 |
| 9.1 | PCB specification | 32 |
| 9.2 | PCB layout | 33 |
| 9.3 | Manufactured PCB | 34 |
| 9.4 | Assembled PCB | 35 |
| 10 | Program | 36 |
| 11 | Mixer structure | 45 |
| 11.1 | Structure diagram | 46 |
| 11.2 | Assembled structure | 47 |
| 12 | Mixing container | 48 |
| 12.1 | Container diagram | 49 |
| 13 | Testing and calibration | 50 |
| 14 | Project working | 51 |
| 15 | Conclusion | 52 |
| 16 | References | 53 |
| 17 | Notes | 54 |

**Aim of the Project**

Our project’s primary goal was to advance our technical skills and knowledge while contributing to societal welfare. This project is budget friendly as well as beginner friendly from the perspective of someone who wants to learn about robotics and mechatronics. The emphasis was on expanding our knowledge and skills in mechanical, electronic and software engineering aspects.

**Initiation of the Idea**

The idea for this project was conceived with the intent to develop a practical yet innovative solution to simplify chemical mixing process for farmers. Observing the increasing demand for automation and efficiency in the agricultural industry, we aimed to design a versatile chemical mixer that integrates basic mechatronics principles. The concept evolved from the desire to create a functional prototype that not only showcases technical feasibility but also serves as an educational tool for aspiring engineers. The focus was on using readily available and budget-friendly components to ensure accessibility and replicability for beginners in the field.

**Key Factors of Consideration**

**Control System:** A robust control system was implemented to manage the operation of all components. It ensured precise coordination and smooth functionality.

**Programming:** The mixer was programmed with logical algorithms for pump control, user input, live display and mixing. The code was designed to be efficient, modular, and easy to debug.

**Power Supply Management**: Efficient power management was prioritized to maintain consistent performance and ensure that all components operated reliably within their power limit.

**Mechanical Design:** The mixer's mechanical design was optimized for sturdiness, portability and durability.

**Modular Integration**: The system was built with modularity, enabling easy replacement or upgrading of individual components without affecting overall functionality.

**Process of Making**

**Conceptualization:** The project began with brainstorming ideas for automated chemical mixer, focusing on its functionality to pour two different liquids with precise measurements that will be input by the user and the solution will be mixed inside a mixing container for amount of time input by user.

**Design:** In the design phase, we created a detailed block diagram of the mixer’s control system. This diagram helped in understanding how each component would interact with others, providing a clear roadmap for the system’s functionality. Based on the design, we could visualize the necessary features and functions, which guided the selection of specific components.

**Component Selection:** Once the control system was mapped out, we moved to selecting the appropriate components. We carefully chose pumps, motor, the microcontroller, and other electronic components based on their compatibility and functionality. The goal was to ensure seamless communication between all parts, balancing performance with budget.

**PCB Design:** components We designed a printed circuit board for our project using the Easy EDA software where all the important components will be placed, this PCB will serve as the control panel of our project.

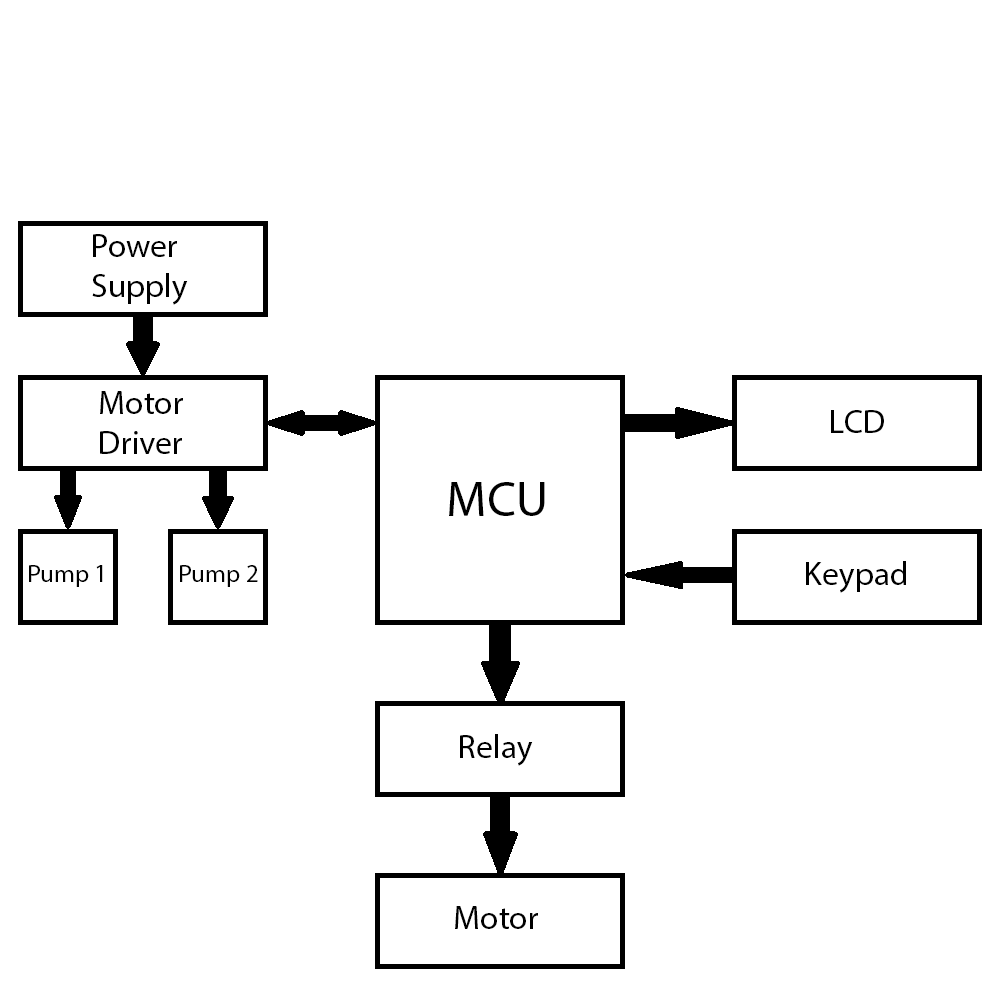
**Assembly:** With all the components ready, we began assembling the mixer. This step involves placing all the components like the ESP32, for LCD for volume and time display and keypad for user input, creating control panel.

**Programming**: After the assembly, we moved on to the programming phase. We first wrote and tested individual programs for each component, such as pump control, LCD display, keypad input etc. Then we integrated two or more components, for example LCD display with keypad input.

**Testing & Troubleshooting:** With the system assembled and programmed, we conducted extensive testing. Each component was tested separately to ensure functionality, and then the entire system was tested as a whole. We observed how the components interacted, verifying that the system behaved as expected. Any issues, such as design flaws, loose connections, or programming errors, were promptly identified and resolved through troubleshooting.

**Documentation:** Throughout the process, we maintained thorough documentation. This included detailed notes on the design, assembly steps, component selection, wiring diagrams, and programming. Additionally, a comprehensive user manual was created to guide future users in building, using, and maintaining the robot.

**Block Diagram**



**Electronic Components**

**ESP32 Devkit V1**

The ESP32 Devkit V1 is a development board that consists of the ESP32 WROOM 32 Module. The ESP 32 WROOM 32 is a powerful, generic Wi-Fi + BT + BLE MCU module that targets a wide variety of applications, ranging from low-power sensor networks to the most demanding tasks, such as voice encoding, music streaming and MP3 decoding. At the core of this module is the ESP32-D0WDQ6 chip\*. The chip embedded is designed to be scalable and adaptive. There are two CPU cores that can be individually controlled, and the clock frequency is adjustable from 80 MHz to 240 MHz. The user may also power off the CPU and make use of the low-power co-processor to constantly monitor the peripherals for changes or crossing of thresholds. ESP32 integrates a rich set of peripherals, ranging from capacitive touch sensors, Hall sensors, low-noise sense amplifiers, SD card interface, Ethernet, high speed SPI, UART, I2S and I2C.

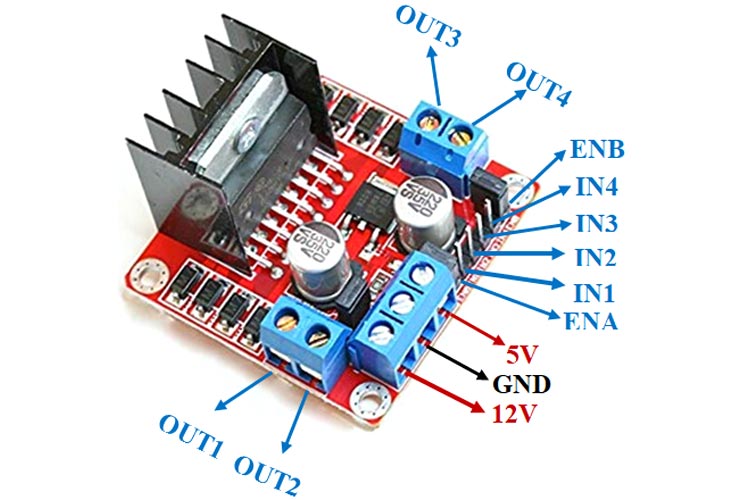
In this project, The ESP32 serves as the brain of the system. It controls the operation of the L298N motor driver, the LCD display, the keypad and the relay module. **(1)**

**Specifications**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Items** | **Details** |
| Wi-Fi | RF certifications | FCC/CETELEC/KCC/SRRC/NCC |
| Protocols | 802.11 b/g/n/e/i (802.11n up to 150 Mbps) |
| A-MPDU and A-MSDU aggregation and 0.4 µs guard interval support |
| Frequency range | 2.4 ~ 2.5 GHz |
| Bluetooth | Protocols | Bluetooth v4.2 BR/EDR and BLE specification |
| Radio | NZIF receiver with -97 dBm sensitivity |
| Class-1, class-2 and class-3 transmitter |
| AFH |
| Audio | CVSD and SBC |
| Hardware | Module interface | SD card, UART, SPI, SDIO, I2C, LED PWM, Motor PWM, I2S, IR |
| GPIO, capacitive touch sensor, ADC, DAC, LNA preamplifier |
| On-chip sensor | Hall sensor, temperature sensor |
| On-board clock | 40 MHz crystal |
| Operating voltage/Power supply | 2.7 ~ 3.6V |
| Operating current | Average: 80 mA |
| Minimum current delivered by power supply | 500 mA |
| Operating temperature range | -40°C ~ +85°C |
| Ambient temperature range | Normal temperature |
| Package size | 18±0.2 mm x 25.5±0.2 mm x 3.1±0.15 mm |
| Software | Wi-Fi mode | Station/SoftAP/SoftAP+Station/P2P |
| Wi-Fi Security | WPA/WPA2/WPA2-Enterprise/WPS |
| Encryption | AES/RSA/ECC/SHA |
| Firmware upgrade | UART Download / OTA (download and write firmware via network or host) |
| Software development | Supports Cloud Server Development / SDK for custom firmware development |
| Network protocols | IPv4, IPv6, SSL, TCP/UDP/HTTP/FTP/MQTT |
| User configuration | AT instruction set, cloud server, Android/iOS app |

**L298N Motor Driver Module**

This dual bridge bi-directional motor driver, is based on the very popular L298 Dual H-Bridge Motor Driver Integrated Circuit. The circuit will allow you to easily and independently control two pumps of up to 2A each in both directions. It is ideal for robotic applications and well suited for connection to a microcontroller requiring just a couple of control lines per motor. It can also be interfaced with simple manual switches, TTL logic gates, relays, etc. This board equipped with power LED indicators, on-board +5V regulator and protection diodes. **(2)**

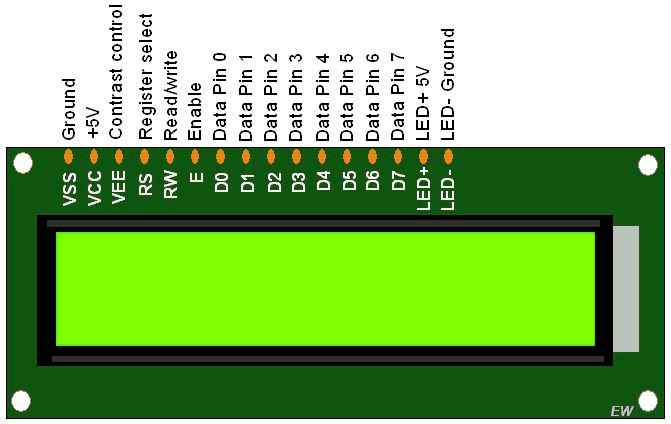


**Specifications**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Details** | |
| Input voltage | 3.2V - 40V DC | |
| Driver | L298N Dual H Bridge DC Motor Driver | |
| Peak current | 2A | |
| Operating current range | 0 – 36mA | |
| Control signal input voltage range | Low | High |
| -0.3V ≤ Vin ≤ 1.5V. | 2.3V ≤ Vin ≤ Vss. |
| Enable signal input voltage range | Low | High |
| -0.3 ≤ Vin ≤ 1.5V | 2.3V ≤ Vin ≤ Vss |
| Max power consumption | 20W | |
| Storage temperature | -25 ℃ ~ +130 ℃ | |
| Dimensions | 44.00 x 44.00 mm | |

**16x2 LCD Module**

A 16x2 LCD (Liquid Crystal Display) is a common alphanumeric display module. Here, “16” refers to the number of characters in a line and “2” refers to the number of character lines. There for the 16x2 module can display total of thirty-two characters, sixteen characters per line. It is paired with an I2C module to reduce the number of pins required on the ESP32 development board. In this project the LCD will be used to display things like the volume selection menu, the mixing time menu, the mixing time countdown etc. (**3)**

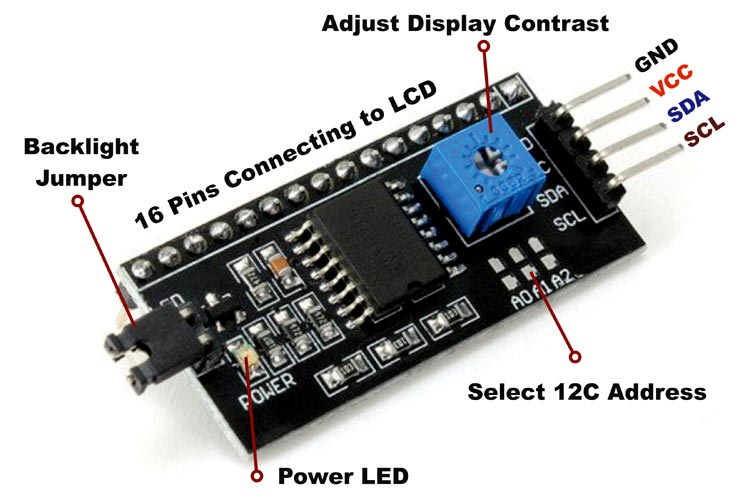


**Specifications**

|  |  |
| --- | --- |
| **Feature** | **Details** |
| Display format | 16 x 2 characters |
| Input voltage | 5V DC |
| Duty cycle | 1/16 |
| Viewing area | 66.0 x 16.0 mm |
| Dot size | 0.55 x 0.65 mm |
| Dot pitch | 0.60 x 0.70 mm |
| Character size | 2.95 x 5.95 mm |
| Dimensions | 80.00 x 36.00 mm |

**I2C Module**

The I2C (Inter-Integrated Circuit) module used with a 16x2 LCD display is a small add-on board that enables communication between a microcontroller (like an Arduino, ESP 32, or Raspberry Pi) and the LCD using the I2C communication protocol, which significantly reduces the number of required GPIO pins. In this project, we have connected the LCD module to the I2C module and the I2C module is connected to ESP 32 to control the display of the LCD module. The I2C module consists of a potentiometer to adjust the contrast of the LCD’s display, so that an external potentiometer is not required. **(4)**



**Specifications**

|  |  |
| --- | --- |
| **Feature** | **Details** |
| Chip | PCF8574 |
| I2C protocol | Compatible with standard I2C (100kHz) and Fast I2C (400kHz) |
| Operating voltage | 5V DC |
| Dimensions | 42.00 x 19.00 mm |

**4x4 Matrix Keypad Module**

The 4x4 matrix keypad module is an input device that consists of 16 push buttons arranged in 4 rows and 4 columns, forming a grid. This keypad allows the user to interact with the system and give input. In this project we have used this keypad module to interact with the mixer, selecting the volume of the liquid that needs to be dispensed and setting the duration for mixing. They keypad is paired with a GPIO expander that helps in reducing the number of pins required on the ESP 32 development board. **(5)**



**Specifications**

|  |  |
| --- | --- |
| **Feature** | **Details** |
| Number of keys | 16 |
| Key layout | 0 – 9, A – D, \*, # |
| Key type | Tactile push button |
| Switch type | Normally open |
| Operating voltage | 3.3 V |
| Operating current | < 1 mA per key press |
| Dimensions | 65.00 x 69.00 mm |

**GPIO Expander Module**

The GPIO expander is an I2C based device that reduces the number of pins being used. It incorporates the PCF8574 IC. It is paired with the 4 x 4 matrix keypad module to reduce the number of pins being used on the ESP 32 from eight pins to four pins. **(6)**

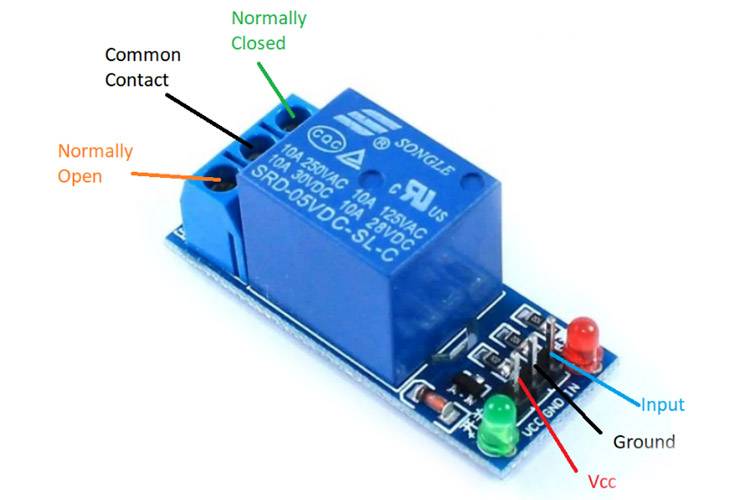


**Specifications**

|  |  |
| --- | --- |
| **Feature** | **Details** |
| Chip | PCF8574 |
| I2C protocol | Compatible with standard I2C (100kHz) and Fast I2C (400kHz) |
| Operating voltage | 5V DC |
| Dimensions | 42.00 x 19.00 mm |

**Relay Module**

Relay is an electromechanical device that uses electric current to open or close the contacts of a switch. The single-channel relay module is much more than just a plain relay, it comprises of components that make switching and connection easier and act as indicators to show if the module is powered and if the relay is active or not. In this project, the relay is used to control the operation of the mixing motors. **(7)**



**Specifications**

|  |  |
| --- | --- |
| **Feature** | **Details** |
| Supply voltage | 3.75 V – 6 V |
| Quiescent current | 2 mA |
| Active relay current | 70 mA |
| Maximum contact voltage | 250 V AC or 30 V DC |
| Maximum current | 10 A |
| Dimensions | 46.0 x 17.5 mm |

**R385 Pump**

The R385 pump is water pump powered by a 12V DC motor. The motor is connected to a diaphragm via a cam, as the cam rotates it causes the diaphragm to move up and down or in and out to pump water. The pumping mechanism is housed inside a plastic casing. In this project, we have used two R385 pumps to pump the liquid from the source container to the mixing container. The operation of the pumps is controlled by the L298N motor driver. **(8)**

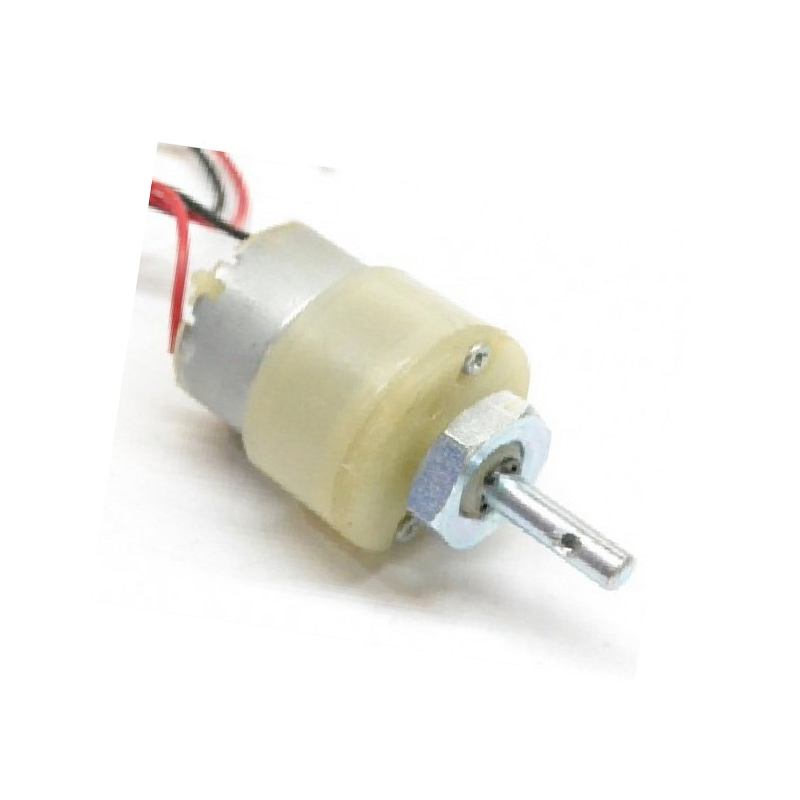


**Specifications**

|  |  |  |
| --- | --- | --- |
| **Category** | **Specifications** | |
| Rated voltage | 9V to 12V DC | |
| Load current | 0.7A | |
| Max current | 2A | |
| Power rating | 4W - 7W | |
| Flow | 1.5 - 2 L/min | |
| Max lift | 3m | |
| Max suction | 2m | |
| Max water temperature | 70 °C | |
| Input/output tube diameter | Outer | Inner |
| 8.5 mm | 6 mm |
| Dimensions | Motor | Pump |
| 52.00 x 39.00 mm | 42.50 x 46.00 mm |

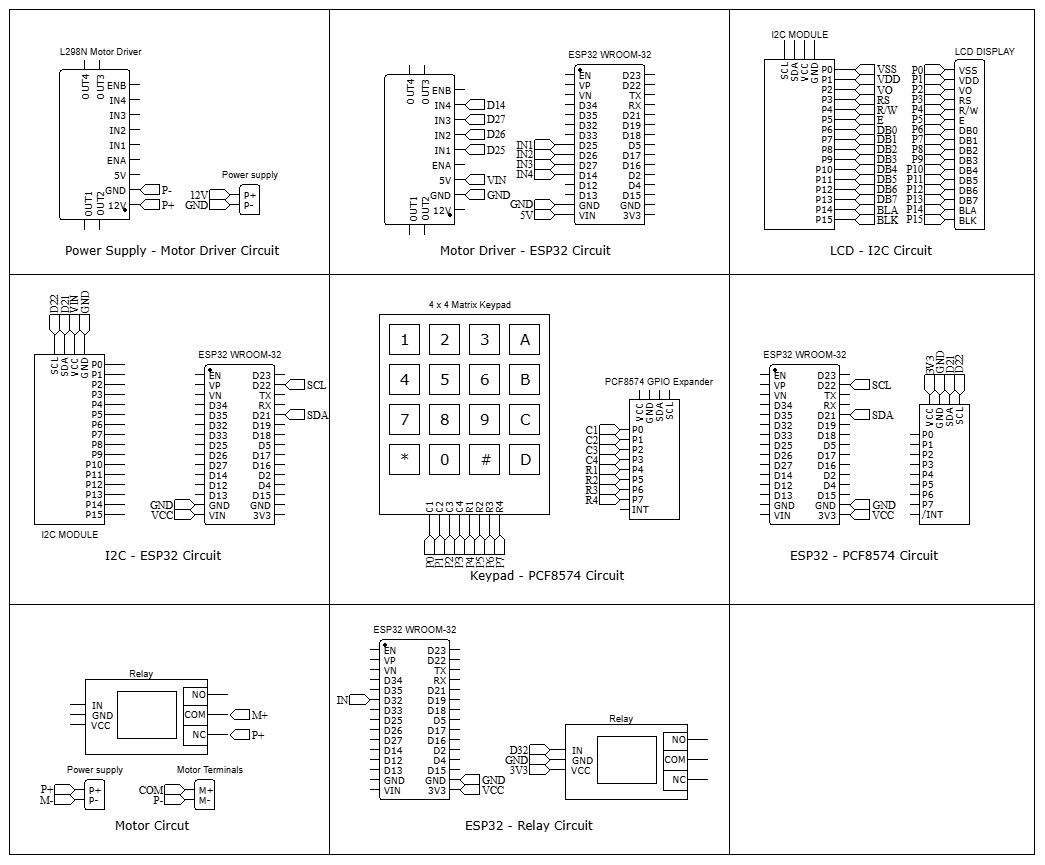
**Geared Motor**

The motor that we used in this project is for rotating the stirrer for mixing is a 100 RPM geared DC motor, it has a voltage rating of 12 V and it spins at a moderate speed and provides enough torque for stirring more viscous chemicals. The shaft of the motor has a socket wrench attachment that fits perfectly onto the stirrer of the mixing container which is an m6 bolt. The motor is turned on and off with the help of the relay. **(9)**



**Specifications**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Details** | |
| Voltage rating | 12 V DC | |
| No load current | 800 mA | |
| Output speed | 100 RPM | |
| Base motor speed | 18000 RPM | |
| Shaft dimension | Length | Diameter |
| 22.00 mm | 6.00 mm |
| Gearbox dimensions | Length | Diameter |
| 21.50 mm | 37.00 mm |
| Base motor dimensions | Length | Diameter |
| 20.00 mm | 32.00 mm |

**Schematic**

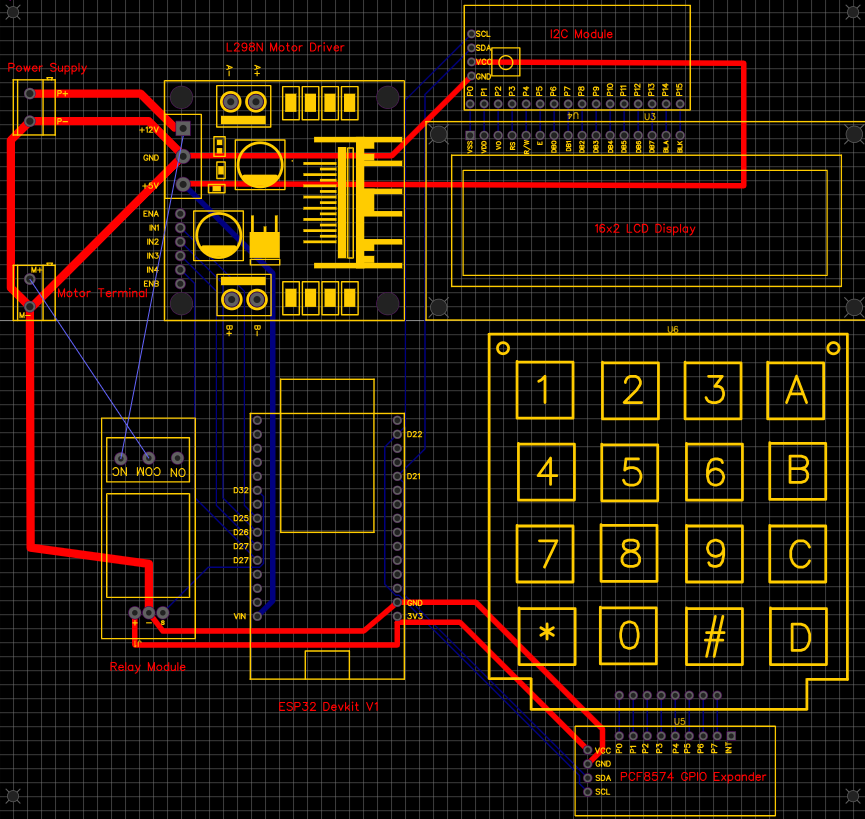
**PCB Design & Fabrications**

With the schematic diagram given above we designed a PCB using the EasyEDA software. Then we sent the Gerber file to a local PCB manufacturer for PCB fabrication. We designed the PCB layout to resemble a control panel, containing all the important components in this circuit but focusing on the lcd and keypad specifically. The other components that it consists of are ESP 32 devkit V1, L298N motor driver and the relay module.

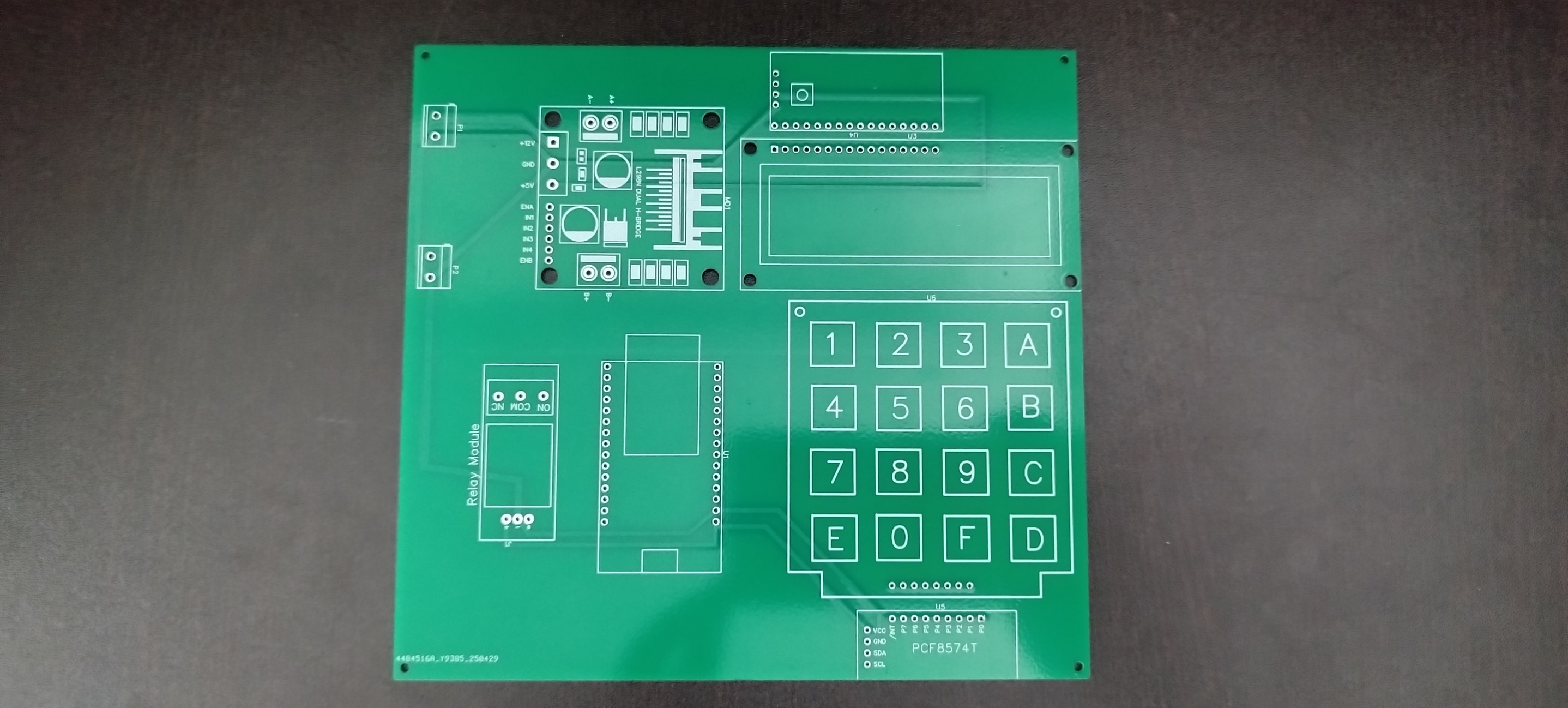
**PCB Specifications**

|  |  |
| --- | --- |
| **Category** | **Specification** |
| Base material | FR-4 |
| Number of layers | 2 |
| Thickness | 1.6 mm |
| Dimensions | 157.48 x 147.32 mm |
| Colour | Green |
| Surface finish | HASL |
| Track width | 1.500 mm, 1.000 mm, 0.254 mm |

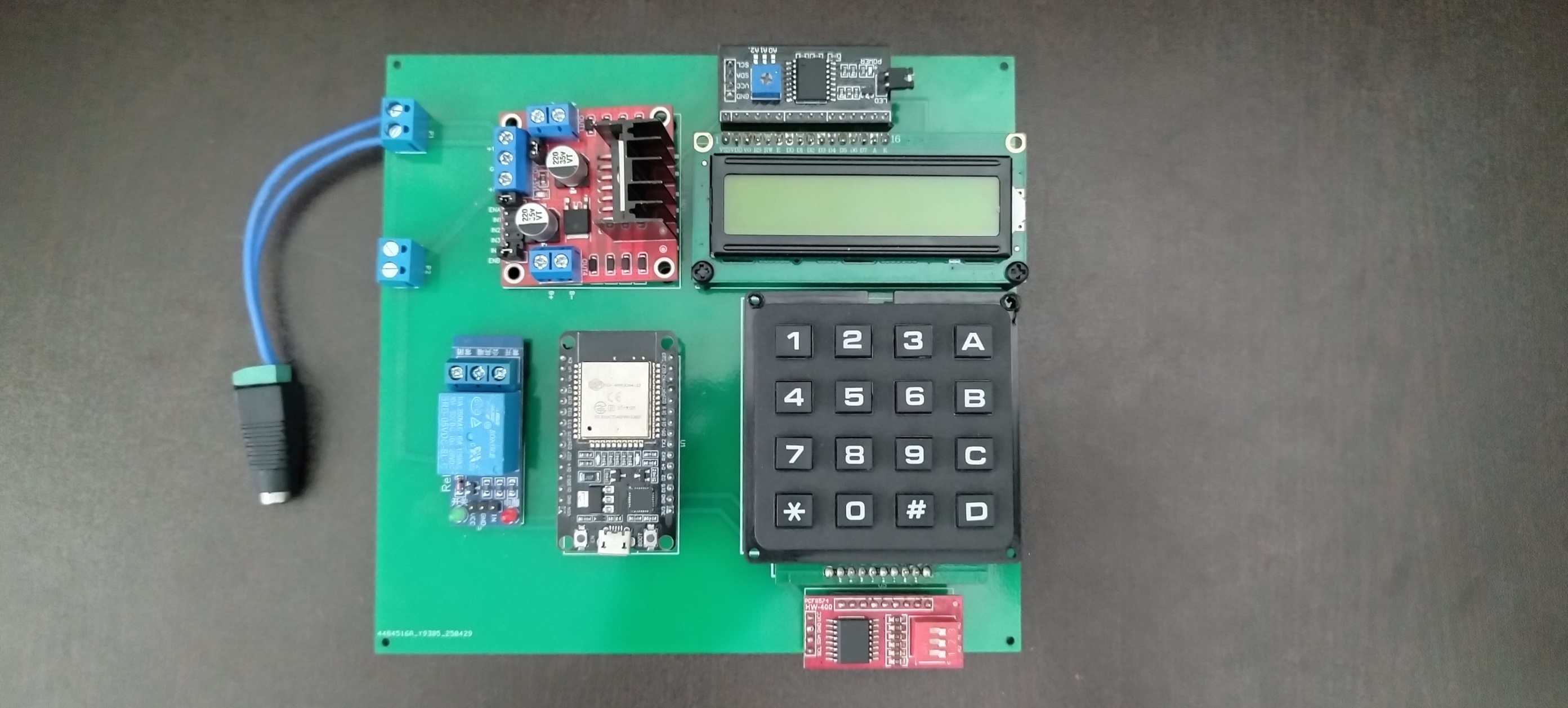
**PCB Layout**



**Manufactured PCB**



**Assembled PCB**



**Program**

#include <Wire.h>

#include <PCF8574.h>

#include <LiquidCrystal\_I2C.h>

// I2C Addresses

#define KEYPAD\_I2C\_ADDRESS 0x20

#define LCD\_I2C\_ADDRESS 0x27

// Pump control pins (L298N)

#define IN1 25

#define IN2 26

#define IN3 27

#define IN4 14

// Relay control pin (Active LOW)

#define RELAY\_PIN 32

PCF8574 pcf(KEYPAD\_I2C\_ADDRESS);

LiquidCrystal\_I2C lcd(LCD\_I2C\_ADDRESS, 16, 2);

char keys[4][4] = {

{'1','2','3','A'},

{'4','5','6','B'},

{'7','8','9','C'},

{'\*','0','#','D'}

};

uint8\_t rowPins[4] = {4, 5, 6, 7}; // PCF8574 row pins

uint8\_t colPins[4] = {0, 1, 2, 3}; // PCF8574 column pins

int containerA = 0, containerB = 0;

int mixMinutes = 0, mixSeconds = 0;

const int pumpTimes[] = {2500, 5000, 6950, 8600, 10400, 12365, 14420, 16480, 18300, 20500};

const int volumes[] = {50, 100, 150, 200, 250, 300, 350, 400, 450, 500};

void setup() {

Serial.begin(115200);

Wire.begin(21, 22);

pinMode(IN1, OUTPUT);

pinMode(IN2, OUTPUT);

pinMode(IN3, OUTPUT);

pinMode(IN4, OUTPUT);

pinMode(RELAY\_PIN, OUTPUT);

digitalWrite(IN1, LOW);

digitalWrite(IN2, LOW);

digitalWrite(IN3, LOW);

digitalWrite(IN4, LOW);

digitalWrite(RELAY\_PIN, HIGH); // Relay off (active LOW)

if (!pcf.begin()) {

Serial.println("PCF8574 not found!");

while (1);

}

lcd.init();

lcd.backlight();

for (int i = 0; i < 4; i++) pcf.write(rowPins[i], HIGH);

showWelcomeScreen();

}

void loop() {

while (true) {

containerA = selectVolume("Select Volume A");

containerB = selectVolume("Select Volume B");

dispenseLiquids();

getMixingTime();

startMixingCountdown();

}

}

void showWelcomeScreen() {

lcd.clear();

lcd.setCursor(3, 0);

lcd.print("WELCOME");

lcd.setCursor(0, 1);

lcd.print("Chemical Mixer");

delay(2000);

}

int selectVolume(String message) {

int index = 0;

lcd.clear();

lcd.setCursor(0, 0);

lcd.print(message);

while (true) {

lcd.setCursor(0, 1);

lcd.print(" ");

lcd.setCursor(0, 1);

lcd.print(volumes[index]);

lcd.print(" mL");

char key = getKey();

if (key == 'A') index = (index > 0) ? index - 1 : 9;

else if (key == 'B') index = (index < 9) ? index + 1 : 0;

else if (key == '\*') return volumes[index];

}

}

void dispenseLiquids() {

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Dispensing A...");

pumpLiquid(containerA, IN1, IN2);

delay(1000);

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Dispensing B...");

pumpLiquid(containerB, IN3, IN4);

delay(1000);

}

void pumpLiquid(int volume, int pin1, int pin2) {

int duration = 0;

for (int i = 0; i < 10; i++) {

if (volumes[i] == volume) {

duration = pumpTimes[i];

break;

}

}

if (duration > 0) {

int step = 5;

int steps = volume / step;

int stepTime = duration / steps;

digitalWrite(pin1, HIGH);

digitalWrite(pin2, LOW);

for (int disp = 5; disp <= volume; disp += step) {

lcd.setCursor(0, 1);

lcd.print("Dispensed: ");

lcd.print(disp);

lcd.print(" mL ");

delay(stepTime);

}

digitalWrite(pin1, LOW);

digitalWrite(pin2, LOW);

}

}

void getMixingTime() {

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Enter time (m):");

mixMinutes = getNumericInput();

if (mixMinutes > 30) {

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Limit is 30m");

delay(2000);

getMixingTime();

return;

}

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Enter time (s):");

mixSeconds = getNumericInput();

int totalSec = mixMinutes \* 60 + mixSeconds;

if (totalSec > 1800) {

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Limit is 30");

delay(2000);

getMixingTime();

return;

}

mixMinutes = totalSec / 60;

mixSeconds = totalSec % 60;

}

void startMixingCountdown() {

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Mixing");

digitalWrite(RELAY\_PIN, LOW); // Motor ON

for (int t = mixMinutes \* 60 + mixSeconds; t >= 0; t--) {

int min = t / 60;

int sec = t % 60;

lcd.setCursor(0, 1);

lcd.print("Time: ");

lcd.print(min);

lcd.print("m ");

lcd.print(sec);

lcd.print("s ");

delay(1000);

}

digitalWrite(RELAY\_PIN, HIGH); // Motor OFF

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Mixing Done");

delay(3000);

}

int getNumericInput() {

String input = "";

char key;

lcd.setCursor(0, 1);

lcd.print("> ");

while (true) {

key = getKey();

if (key >= '0' && key <= '9') {

input += key;

lcd.print(key);

} else if (key == '\*') {

if (input.length() == 0) continue;

return input.toInt();

}

}

}

char getKey() {

for (int row = 0; row < 4; row++) {

for (int i = 0; i < 4; i++) pcf.write(rowPins[i], (i == row) ? LOW : HIGH);

for (int col = 0; col < 4; col++) {

if (pcf.readButton(colPins[col]) == LOW) {

delay(200);

while (pcf.readButton(colPins[col]) == LOW);

return keys[row][col];

}

}

}

return 0;

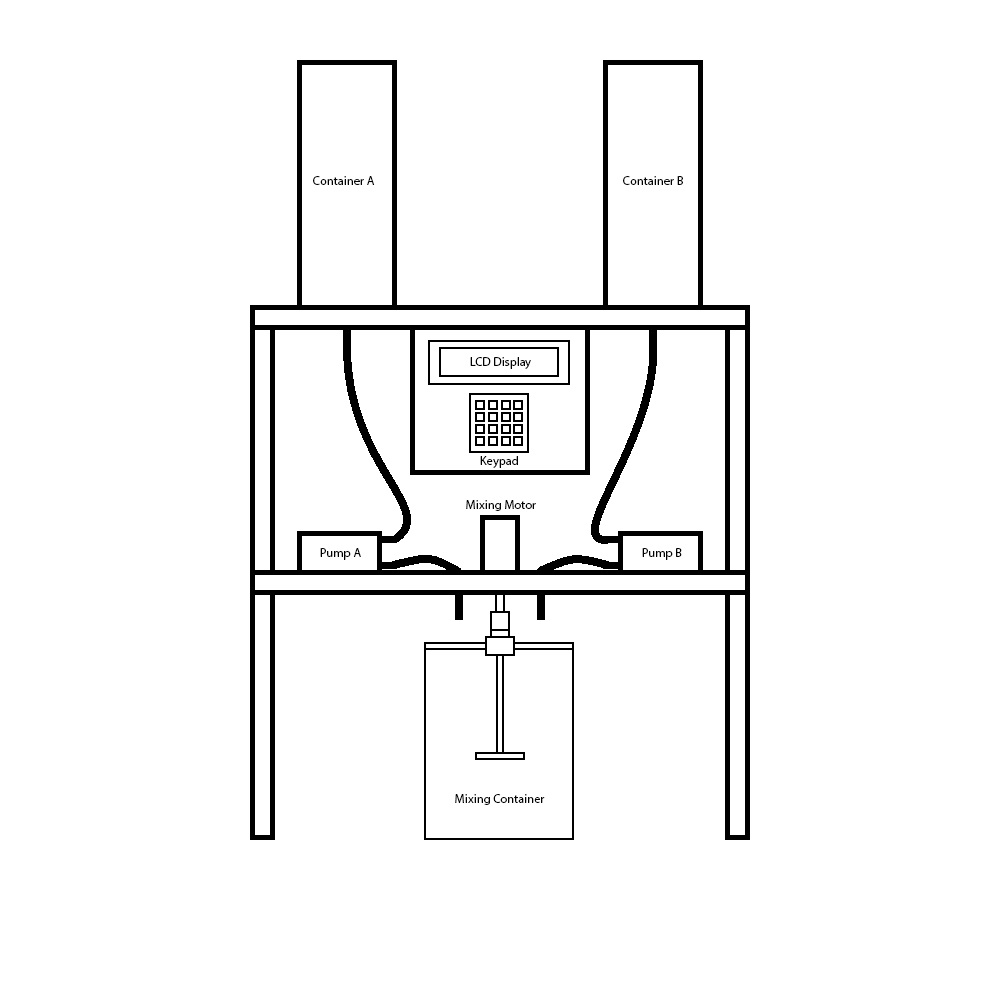
}

**Mixer Structure**

To construct the structure for the chemical mixer, we used a three-shelf plastic table. Two plastic bottles are mounted on the top shelf, serving as containers for two different chemicals. The second shelf houses both pumps and the mixing motor, with the motor shaft oriented downward. Silicone rubber tubes connect each container to its respective pump, enabling fluid transfer.

The PCB is mounted on an acrylic sheet fixed to the top shelf at an angle for better accessibility. All wiring for the pumps and motor is routed through the hollow legs of the table, ensuring a clean and organized setup. We used 26 AWG PVC-insulated wires for all electrical connections. A 12V DC jack is installed on the angled acrylic sheet, allowing the system to be powered by simply inserting a 12V adapter plug into the jack.

**Structure Diagram**



**Assembled Structure**

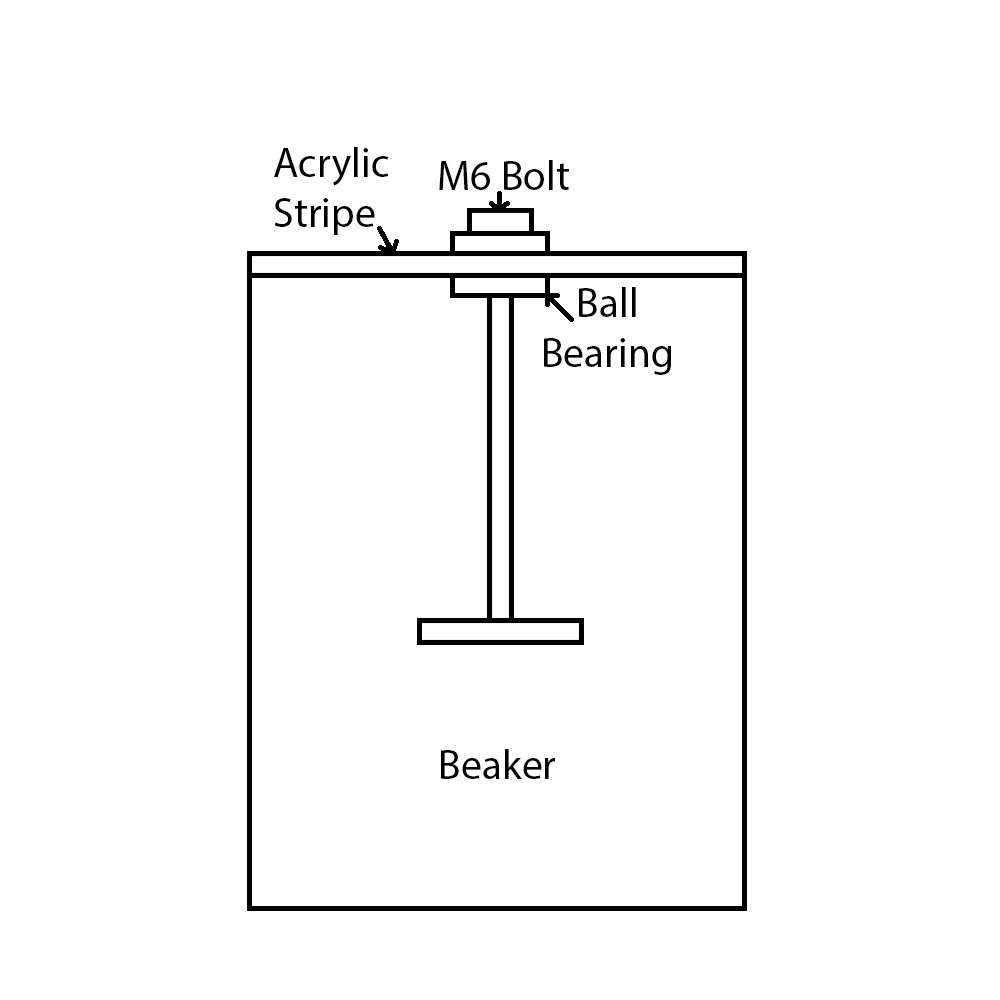


**Mixing Container**

We have designed a customized mixing container for the chemical mixer. It features an acrylic strip mounted on top of the container, with a ball bearing positioned at its center. A 4-inch M6 bolt is inserted vertically through the bearing and secured on the opposite side using a nut. A blade is attached to the lower end of the bolt, forming the stirring mechanism of the container.

A socket wrench is connected to the shaft of the mixing motor and is capable of limited vertical movement. To operate the mixing container, lift the socket wrench and position the container such that the bolt head aligns directly below it. Releasing the socket wrench allows it to automatically lock onto the bolt head. When the motor is activated, it rotates the bolt, effectively stirring the contents of the container.

**Container Diagram**



**Testing & Calibration**

To ensure accurate operation of the liquid dispensing mechanism, extensive testing and calibration of the pump motors were carried out. Since this project uses a time-based liquid dispensing method instead of flow sensors, it was essential to determine how long each pump needs to run to dispense specific volumes of liquid.

The motors connected to Pump 1 and Pump 2 were individually tested using a stopwatch and a measuring container. The aim was to establish a correlation between motor run-time (in milliseconds) and the volume of liquid dispensed (in milliliters). Each test was repeated 2–3 times to ensure consistency.

Testing was performed for all the values from 50 ml to 500 ml with a 50 ml difference and the follwing observations were made.

* The flow rate of water is not linear.
* The time value for each volume did not increase linearly.
* The observed time values are given in the table below.

|  |  |
| --- | --- |
| **Volume** | **Time** |
| 50 ml | 2500 ms |
| 100 ml | 5000 ms |
| 150 ml | 6950 ms |
| 200 ml | 8600 ms |
| 250 ml | 10400 ms |
| 300 ml | 12365 ms |
| 350 ml | 14420 ms |
| 400 ml | 16480 ms |
| 450 ml | 18300 ms |
| 500 ml | 20500 ms |

**Project working**

As the system powers up, the LCD module displays the message “Welcome Chemical Mixer” followed by the volume selection menu. The user is prompted to select the volume for Container A first. Using the “A” and “B” keys on the keypad, the user can navigate through the available volume options (e.g., 50 mL, 100 mL, 150 mL, etc.). Once the desired volume is highlighted, pressing the “\*” key confirms the selection. The system then moves to the volume selection for Container B, where the same procedure is followed.

After both volumes are selected, the system begins dispensing. Pump 1 (connected to Container A) is activated first, and the live volume being dispensed is displayed on the LCD in steps of 5 mL. After Pump 1 finishes, Pump 2 (for Container B) is activated and operates in the same way.

Once both liquids are dispensed, the LCD prompts the user to select the mixing time in minutes and seconds. If the entered time exceeds the 30-minute limit, an error message is displayed. After a valid time is set, the mixing motor is turned on, and the LCD shows a countdown timer alongside the message “Mixing”. When the countdown ends, the motor stops automatically, completing the mixing process.

**Conclusion**

The Chemical Mixer project successfully demonstrates an automated system designed for agricultural applications, where the accurate dispensing and mixing of liquids such as fertilizers, pesticides, or nutrients is essential but extreme precision is not critical. Using an ESP32 microcontroller, submersible pumps, keypad input, and an LCD display, the system provides an easy-to-use interface for selecting volumes and mixing times, making it practical for farmers and agricultural technicians.

By automating the mixing process, this system helps reduce manual labor and minimizes errors in liquid measurement, improving efficiency and consistency in field operations. The time-based dispensing approach is suitable for typical agricultural needs, offering a cost-effective and reliable solution. This project also lays the groundwork for future improvements like sensor feedback and remote monitoring, which could further enhance its usability in agriculture.

**References**

(1) ESP 32 devkit V1: <https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf>

(2) L298N motor driver module: <https://www.handsontec.com/dataspecs/L298N%20Motor%20Driver.pdf>

(3) 16 x 2 LCD module:

<https://www.vishay.com/docs/37484/lcd016n002bcfhet.pdf>

(4) I2C module

<https://www.handsontec.com/dataspecs/module/I2C_1602_LCD.pdf>

(5) 4 x 4 matrix keypad module:

<https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/761/3844_Web.pdf>

(6) GPIO expander module:

<https://www.ti.com/lit/ds/symlink/pcf8574.pdf>

(7) Relay module:

<https://components101.com/switches/5v-single-channel-relay-module-pinout-features-applications-working-datasheet>

(8) R385 Pump:

<https://5.imimg.com/data5/SELLER/Doc/2021/4/RH/JK/XD/1833510/r385-dc-9-12v-water-pump-maximum-lift-3-meters.pdf>

(9) Geared motor:

https://potentiallabs.com/cart/dc-motors-100rpm

**Notes**