RAFIK HARIRI UNIVERSITY

AUTOMATED COFFEE MACHINE

Done by

AHMAD ZOUHAIR HAMMOUDEH

AHMAD ACHAJI

ZIAD ZOK

Submitted to

DR. AHMAD KOUBEISSI

MECHREF-LEBANON

April 2024

Ahmad Zouhair Hammoudeh (2021-0482)

Ahmad Achaji (2021-0313)

Ziad Zok (2021-0281)

ABSTRACT

This project presents the design and implementation of a fully automated coffee machine capable of producing both hot and cold coffee beverages, with options for adding coffee mate and sugar according to user preference. The machine offers versatility in beverage customization, catering to varying tastes and preferences. Through a combination of hardware and software components, and a user-friendly interface, simplifying navigation and customization, users and enjoy freshly brewed coffee tailored to their liking with ease and convenience.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1 INTRODUCTION

1.1 Background Information

Coffee consumption is a prevalent part of daily routines worldwide, with preferences ranging from hot to cold beverages and varying levels of sweetness. Traditional coffee-making methods may not always provide the level of customization and convenience desired by consumers. Hence, there is a growing demand for automated coffee machines that offer versatility and ease of use. Additionally, the popularity of coffee mate and sugar as additives further underscores the need for a machine capable of accommodating these preferences. No two coffee lovers are alike, and automated coffee machines are embracing this diversity by offering a wide range of customization options. From adjusting the strength and temperature of your coffee to selecting specific ingredients and water level

Figure 1-1 – A Modern Automated coffee machine

1.2 Literature Review

Automated coffee machines have become increasingly popular in recent years, driven by advancements in technology and a growing demand for convenience and customization in coffee consumption. This literature review provides an overview of existing research and developments in the field of automated coffee machines, focusing on key themes such as functionality, user experience, and technological innovation.

1.2.1 Faema E61

- Overview:

  The E61 brew group is a legend in the industry, introducing hydraulic pre-infusion with a thermal balancing system. FAEMA's adjustable thermal balance system allows you to regulate the water temperature and to adapt each group to different blends of espresso coffee. As a testament to the quality of the true FAEMA E61 commercial espresso machine, even today, there are many original (that are from 1961). [1]

Figure 1-2 Faema E61 Automatic espresso machine

- Specification:

- Width (in) 21.9 Depth (in) 21.3 Height with tall feet (in) 26.5
- Weight (lb.) 112.
- Power at 208-240V~ 60Hz (W) 3000-3900
- 1 Steam wand
- Boiler capacity (liters) 6
- Hot water economizer YES
- Price: $9,400.00 (1 Group)

1.2.2 Gaggia Syncrony Digital

- Overview:

The reason for its success is simple: it's a straightforward, easy-to-use machine that has the right combination of features and produces great espresso. the Syncrony will grind whole beans, tamp the grounds, and brew delectable espresso. But this espresso machine also gives the user control over just about every part of that process. The Syncrony's brain extends far

beyond the ability to program brewing buttons. With its digital display, this espresso machine communicates with you, telling you in plain English when the Syncrony is ready to brew, ready to steam, or when just about any maintenance procedure needs to be completed. [2]

Figure 1-3: Gaggia Syncrony Digital

- Specification:

- Approximate dimensions: 22.4 cm (length) × 35.7 cm (height) × 43.5 cm (depth).
- The machine weighs approximately 8.5 kg.
- Grinding and Brewing
- Programmable Brewing Buttons:
- Pannarello Wand
- Digital Display alerts, and more.
- Allows brewing with pre-ground coffee.
- Hot Water Dispenser
- Price: $1,050.00

1.2.3 Mr. Coffee One-Touch Coffeehouse

This Mr. Coffee machine sits at a comfortable intersection where ease of use, automation, and affordability. It can extract a flavorful espresso from almost any beans and grind, and its milk reservoir will automatically mix a cappuccino or latte for you. [3]

Figure 1-4: Mr. Coffee One-Touch CoffeeHouse

- Specification:

Approximate dimensions: 8.86"D x 11.22"W x 12.6"H

Weight: 10.37 Pounds

Wattage: 1040

Material: Stainless Steel

Price: $199.99

1.3 Project Motivation

The motivation behind our project stems from a desire to provide coffee enthusiasts with a customizable and convenient experience. Traditional coffee machines often offer limited options for customization, leaving users with few choices beyond standard coffee blends. To address these limitations, we have developed an innovative coffee machine with three containers specifically designed for powder ingredients. These containers offer users the freedom to fill them with a variety of powdered substances, ranging from milk powder to flavorings and sweeteners. By allowing users to customize their coffee with their preferred ingredients, our machine empowers individuals to create personalized beverages tailored to their unique tastes and preferences. Users no longer need to measure and add powdered ingredients manually; instead, they can simply select their desired options from the machine's interface, and the ingredients will be dispensed automatically.

1.4 Project Objective

The objective of our project is to design and implement an automated coffee machine that offers versatility and convenience in brewing both hot and cold coffee beverages. This machine will be equipped with three containers specifically designed for powder ingredients, allowing users to customize their coffee with a variety of options, from milk powder to flavorings and sweeteners. The key objectives of the project include:

Developing a user-friendly interface that simplifies navigation and customization options for brewing coffee.

Integrating three containers for powder ingredients into the design, ensuring easy refilling and dispensing of user-selected ingredients.

Implementing features such as LED indicators to monitor ingredient levels and coffee readiness, enhancing user convenience and experience.

Including safety cup detection technology to ensure safe and accurate dispensing of brewed coffee into cups of varying sizes.

Introducing a robotic arm mechanism to mix the powder ingredients with water, ensuring thorough blending and optimal flavor extraction.

Integrating a load cell into the system to measure the quantity of each ingredient dropped, ensuring precise and accurate dispensing of ingredients for each cup of coffee.

1.5 Design Constraints

Technical Constraints:

In designing our Automated Coffee Machine, we've set clear design constraints to ensure technical strength, affordability, and environmental responsibility. These constraints cover key aspects like size, cost, and energy efficiency, ensuring the machine performs well, remains accessible, and minimizes its environmental footprint.

1. Weight Limitation: The Coffee Machine Weight depends on whether its full of ingredients or empty ranging from (4 kilograms empty to 6-7 full) (2.275 liters of water and 500g of ingredients)

2. Dimension Restriction: (40cm length, 40cm wide, 35cm height) as the average kitchen top depth is ranges from 60 to 88cm which is more than enough for our machine.

3. Energy Efficiency: A Switch mode power supply using a toroidal transformer which ensure high efficiency lower energy consumption and reducing environmental impact.

Non-Technical Constraints:

1. Economic Considerations: from an economic point of view our coffee machine should not cost more than 300$ (including parts, 3d printing, wood…).

2. Environmental Sustainability: To align with environmental considerations, the filament used in 3D printing the coffee machine components must be recyclable. This constraint aims to minimize the ecological footprint.

CHAPTER 2 DESIGN CONCEPTS

2.1 Design Iteration 1:

We utilized Fusion 360 for all design aspects to ensure seamless integration, precise engineering, and efficient prototyping. This approach facilitated collaborative design iterations, and 3d printing. Throughout the development process, we conducted two design iterations to refine and enhance the functionality and aesthetics of the coffee machine.

Exterior:

- Starting with the exterior design, the coffee machine box is made from NDF wood 12mm thick, 40cm width, 40cm length, and 35 cm height. The box consists of 3 parts, 10cm for the cup / mixing area, 20cm middle part containing the ingredients and boiler, and 10 cm in the back for the microcontroller, power supply and control circuit.

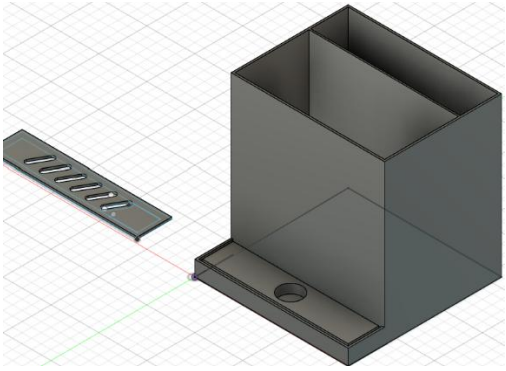


Figure 2-1: 3d model of the exterior

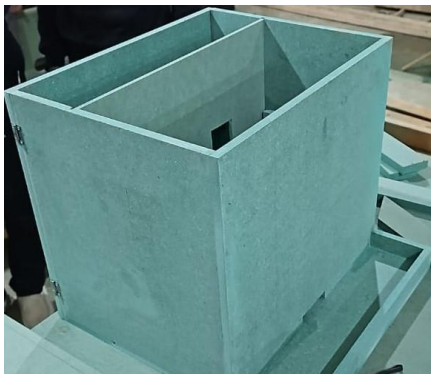


Figure 2-2: in real life wood exterior

- Ingredients Dropper:

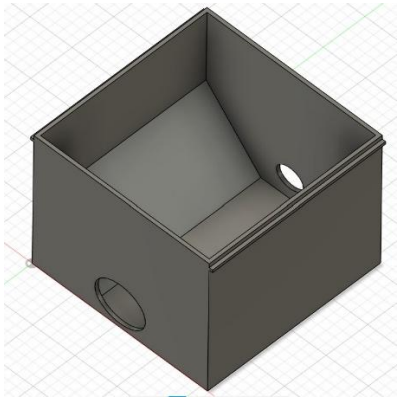


Figure 2-3: Powder dropper

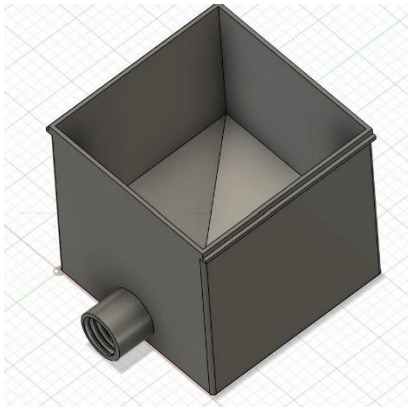


Figure 2-4: Water Tank

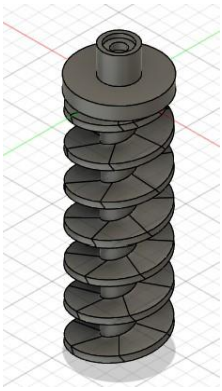


Figure 2-5: Auger

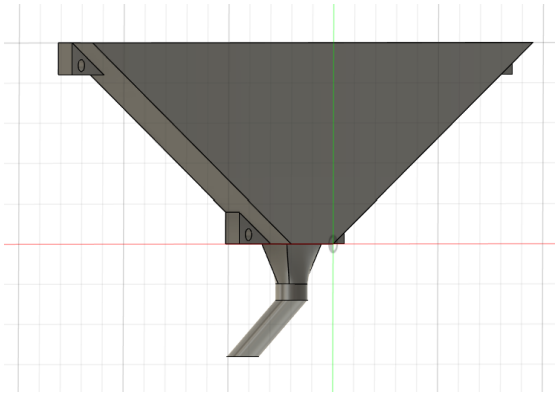


Figure 2-6: into the cup Funnel

(Figure 2-7) represent the first design of the powder ingredient container, the auger in (Figure 1-9) is used to push the powder from the container into the funnel which then leads the powder into the cup for next opertaions.

2.2 Final Iteration:

We faced multiple problems in iteration one design, starting with the “Powder dropper” which has some major flaws in the consistency of dropping the powder. The powder could sometimes get under the auger and increaser the friction and sometimes block it from turning thus stopping the dropping process.

To fix these two major problems we designed a new container design consisting of 3 major parts:

First part is the tube that contains the auger and where the powder goes from the container into the main funnel. The powder enters from opening one and get pushed by the same auger in (Figure 2-5). As shown in (figure 2-7) below.

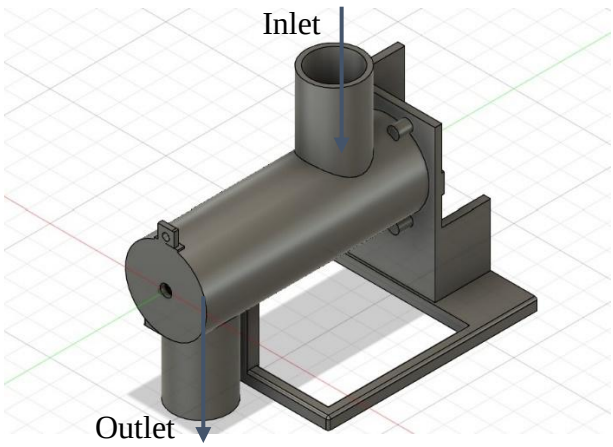


Figure 2-7: Powder Dropper

The desired powder is placed inside a funnel that sits on the inlet of the powder dropper this container has a cover that can be opened and closed regularly, additionally an opening in the lid is made for an ultra-sonic sensor. As shown in figure (2-8), and figure (2-9).

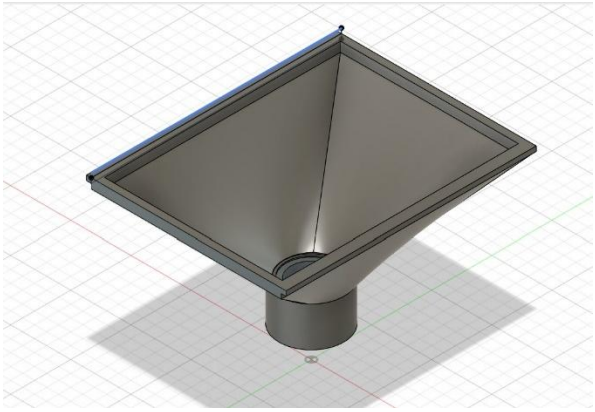


Figure 2-8: Powder Container/funnel

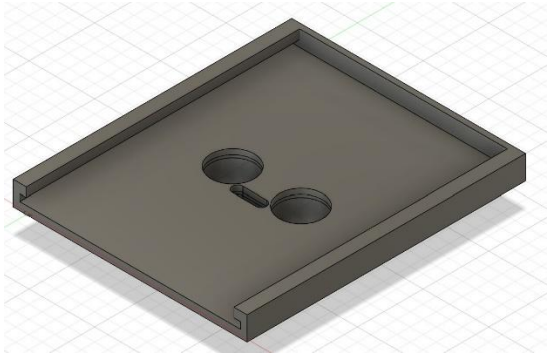


Figure 2-9: Container/funnel lid

This design ensures a smoother dropping process while maintaining the ability to monitor the powder level during the process. The container now works as a funnel and a container at the same time.

In design iteration 2 we also changed the design of the into the cup Funnel or the “main funnel”. We made it wider, thicker, and with a longer and wider nozzle to ensure no powder gets stuck and stops the operation which can lead to flowing problems, this part was designed to be printed by “ABS” filament instead of “PLA”.

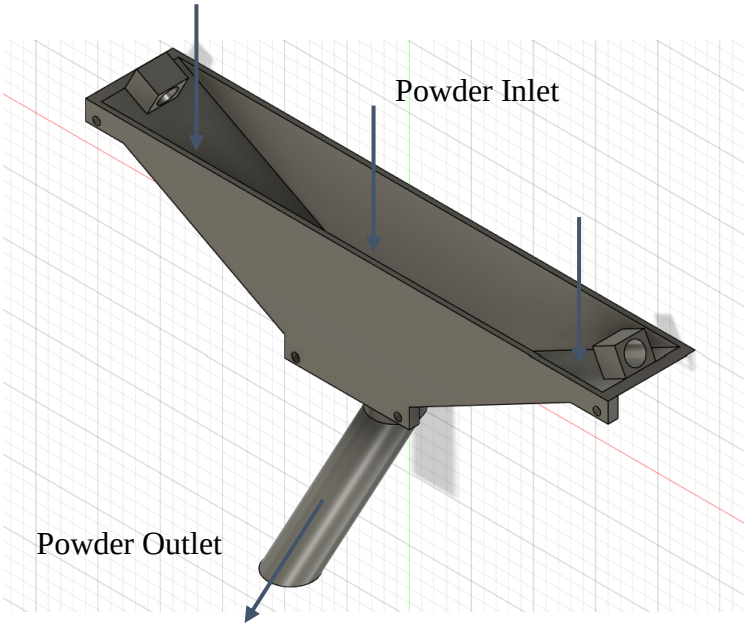


Figure 2-10: Main Funnel

To measure precisely how much ingredient is dropped and to ensure we have a closed loop control we used a load cell to measure the weight of the cup and the ingredients dropped, we needed a design to hold the load cell while still maintaining its functionality of measuring. We end up with our own design inspired by an already existing design for a 20kg load cell, with slight modifications and lesser infill percentage. It consists of 2 parts: part 1 is the base shown in (Figure 2-11) and the top where the cup is placed shown in (Figure 2-12).

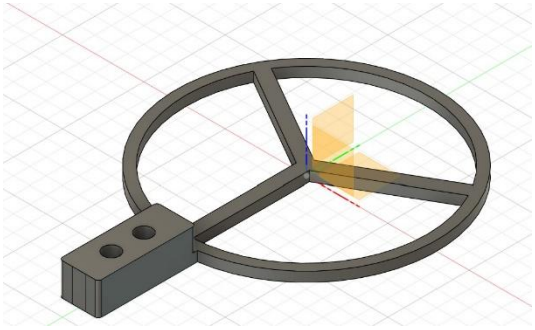


Figure 2-11: Load Cell Base

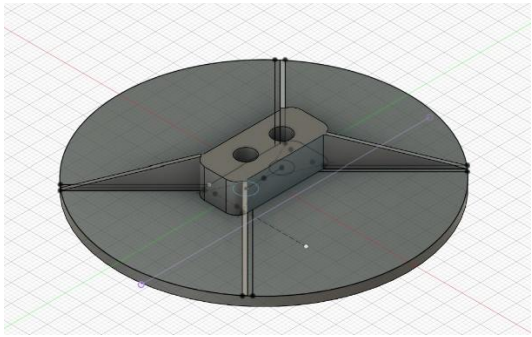


Figure 2-12: Load Cell Cup Platform

The user interface design is made to hold 1 16 by 2 lcd display, 5 led indicators and 4 push buttons for easy to reach interface and monitoring indicators as shown in the (Figure 2-13) below:

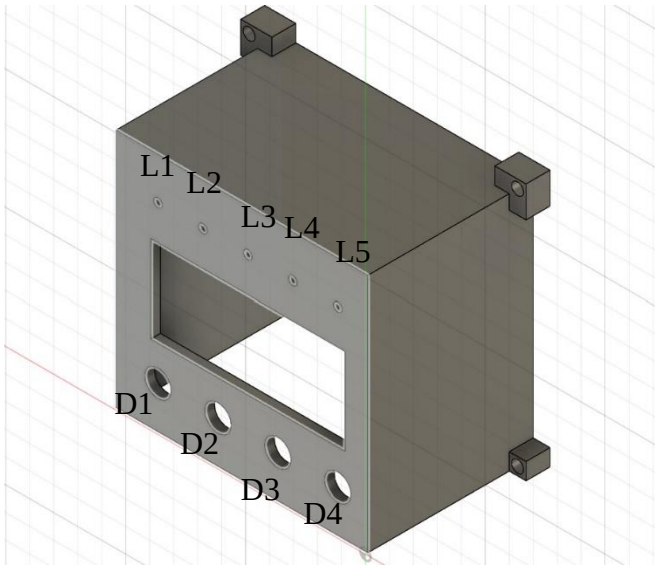


Figure 2-13: Interface

The last part of our design is the Mixing arm we designed a simple yet effective robotic arm with 2 DOFs, which holds a DC Motor, the motor's shaft is connected to a long rod for mixing. The arm parts are shown in the figures below:

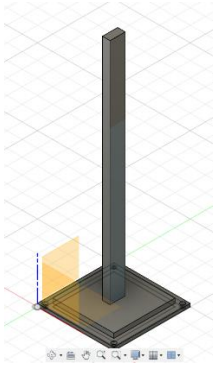


Figure 2-14: Main stand

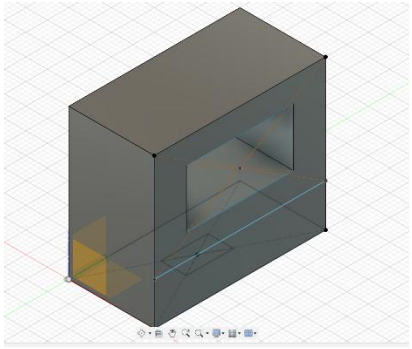


Figure 2-15: First servo holder

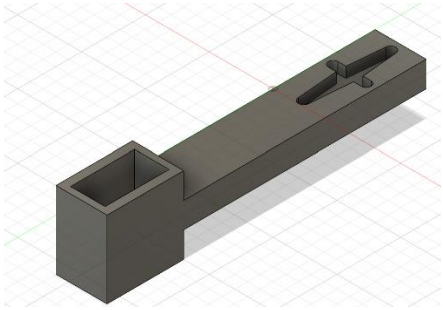


Figure 2-16: second servo holder

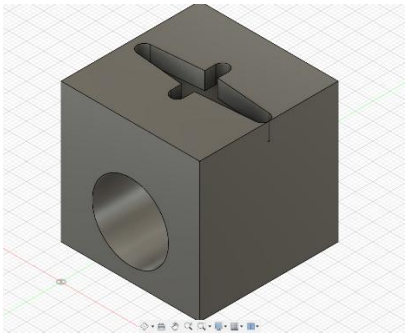


Figure 2-17: Dc motor holder

CHAPTER 3 Final Design Validation

3.1 Technical Validation of Final Prototype (process)

Chapter 3 presents the technical validation process of the final prototype, this chapter delves into various aspects of the prototype, including mode of operation, safety measures, and testing procedures. Through testing, we aim to evaluate the functionality, efficiency, and reliability of the final prototype, ensuring that it meets the desired performance standards. Each subsection provides detailed insights into the design, implementation, and validation of key components, offering a comprehensive overview of the technical validation process.

3.1.1 Electrical System Schematic

This system manages the user interface, managing both the selection of ingredients and the monitoring of ingredient levels. It allows users to easily choose their coffee preferences and check if any ingredients are running low. It also simulates the cup detecting infra-red sensor, along with 2 additional sensors: a load cell to measure the cup weight and a temperature sensor that measures the water temperature inside the boiler.

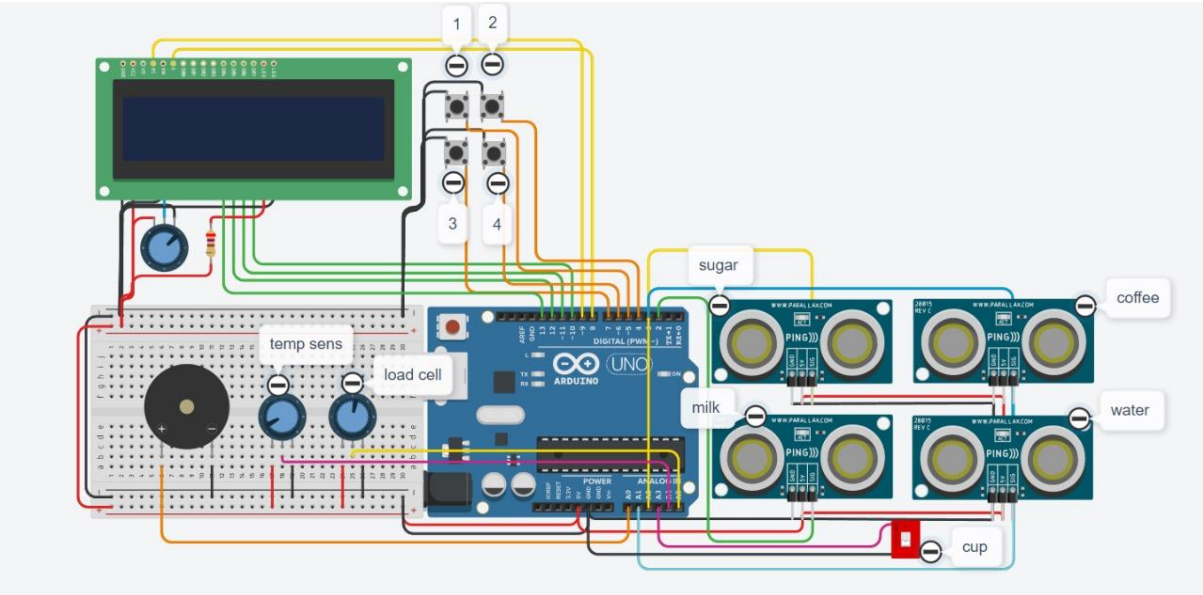


Figure 2-18: Tinker Cad Simulation

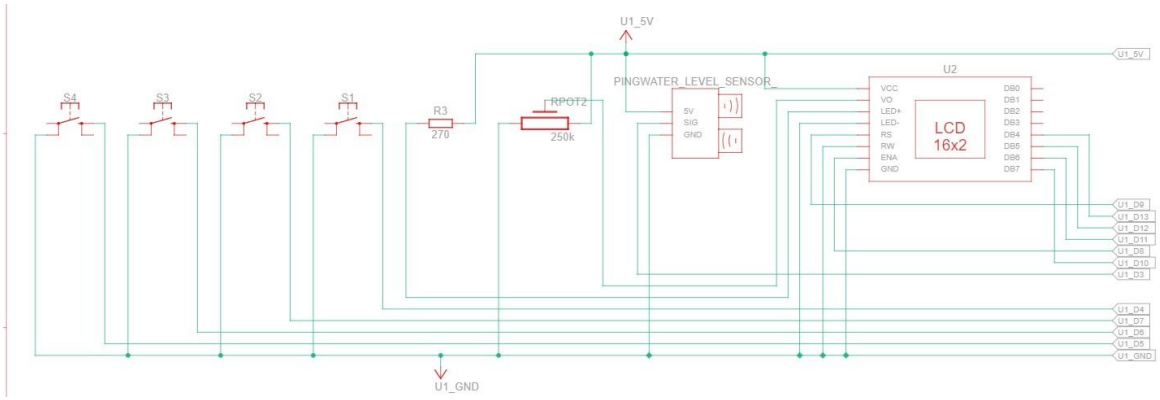


Figure 2-19: Schematic view 1

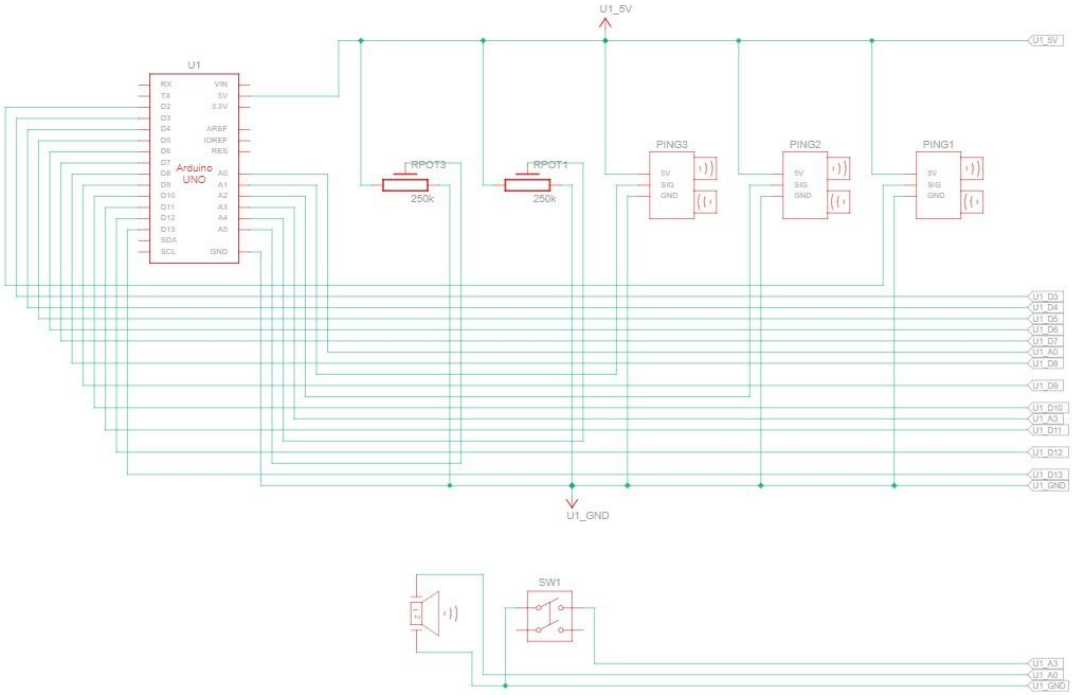


Figure 2-20: Schematic view 2

The second subsystem in our project is the control circuit. the Arduino Mega microcontroller, which acts as the brain, processing inputs from a diverse array of sensors and peripherals and generating corresponding outputs to control the machine's functions.

The control circuit encompasses a variety of sensors, including four ultrasonic sensors responsible for monitoring ingredient levels, a temperature sensor in the boiler to regulate brewing temperature, and an infrared cup detector to detect the presence of cups. Two servo motors control the robotic arm's movement, precisely positioning it for ingredient dispensing and mixing. Four relays are utilized to interface with DC motors for ingredient dispensing and

to control the boiler's heating element. The LCD screen and four buttons provide a user-friendly interface for interaction.

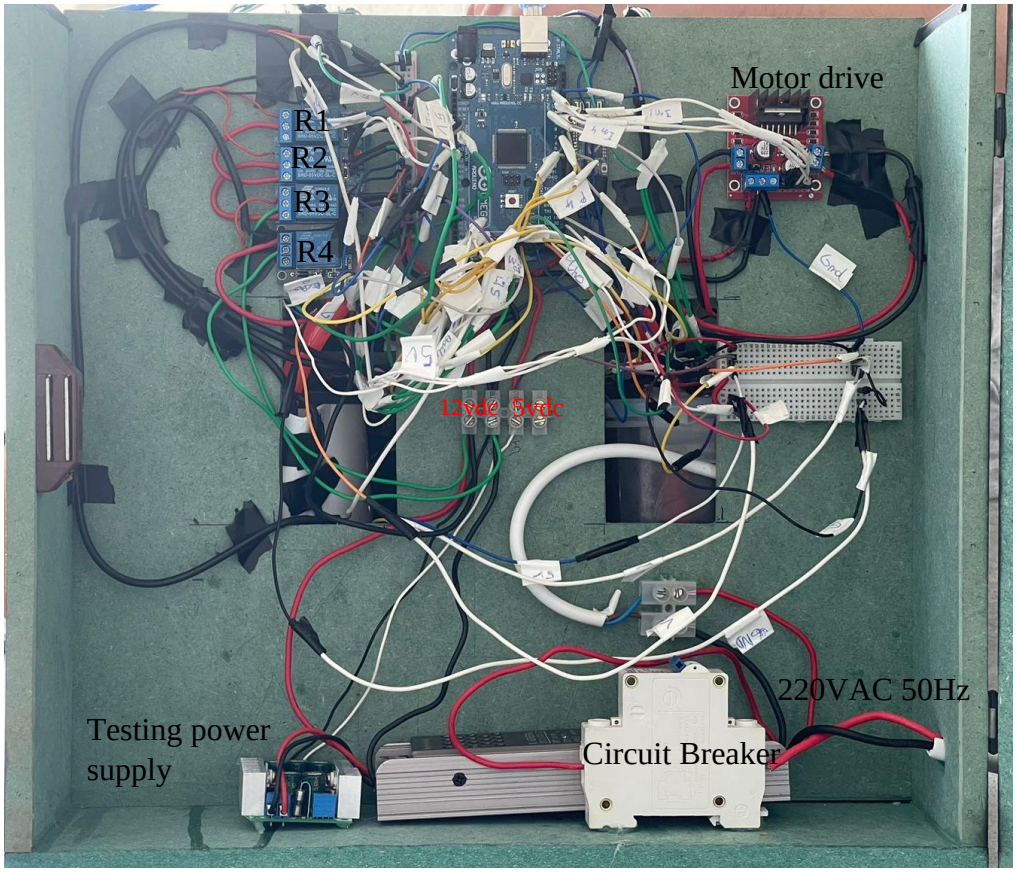
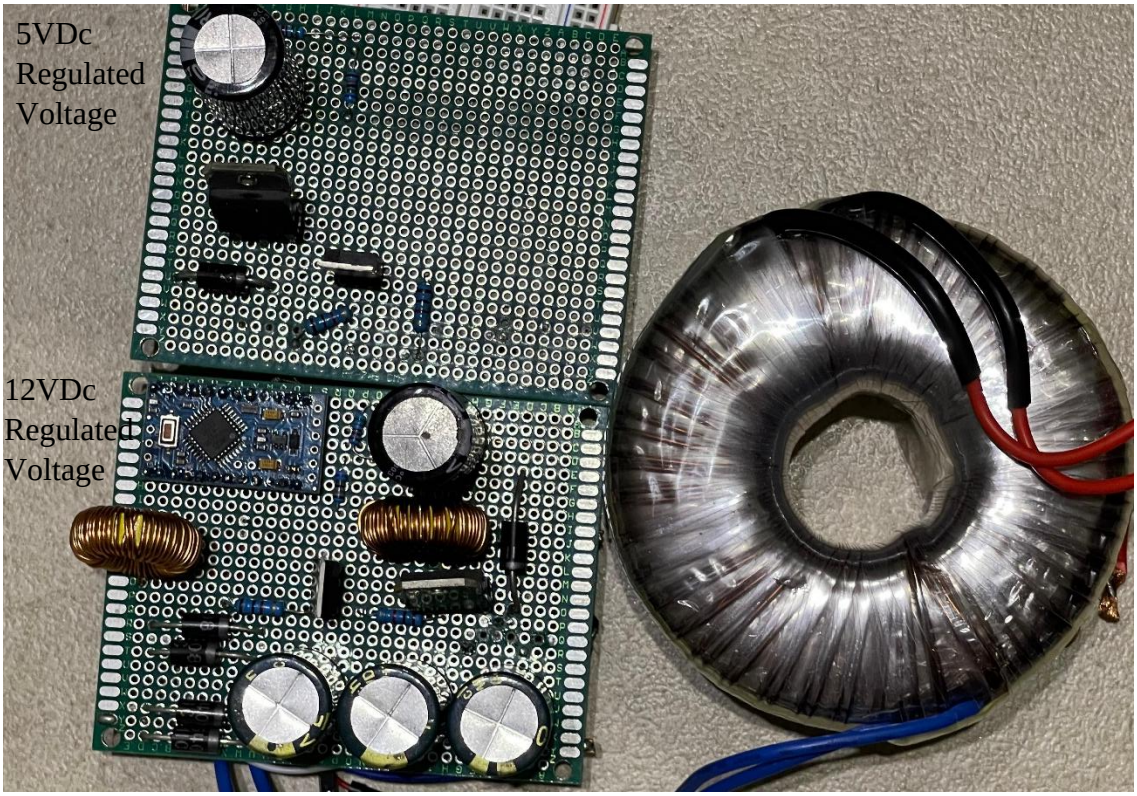


Figure 2-21: Circuit implementation

The Third subsystem in our project is the power supply capable of transforming 220VAC into 12VDC Regulated and 5VDC regulated using a toroidal transformer, full waver rectifier, and a buck booster controlled by a PWM signal from an Arduino pro mini to regulate the voltage according to the feedback from the output of the power supply. The simulation below done on proteus and provided by the instructor shows the circuit, with changed values of components according to the needed output (5-6 Amps, 12v, 5v dc regulated) along with the circuit implementation.



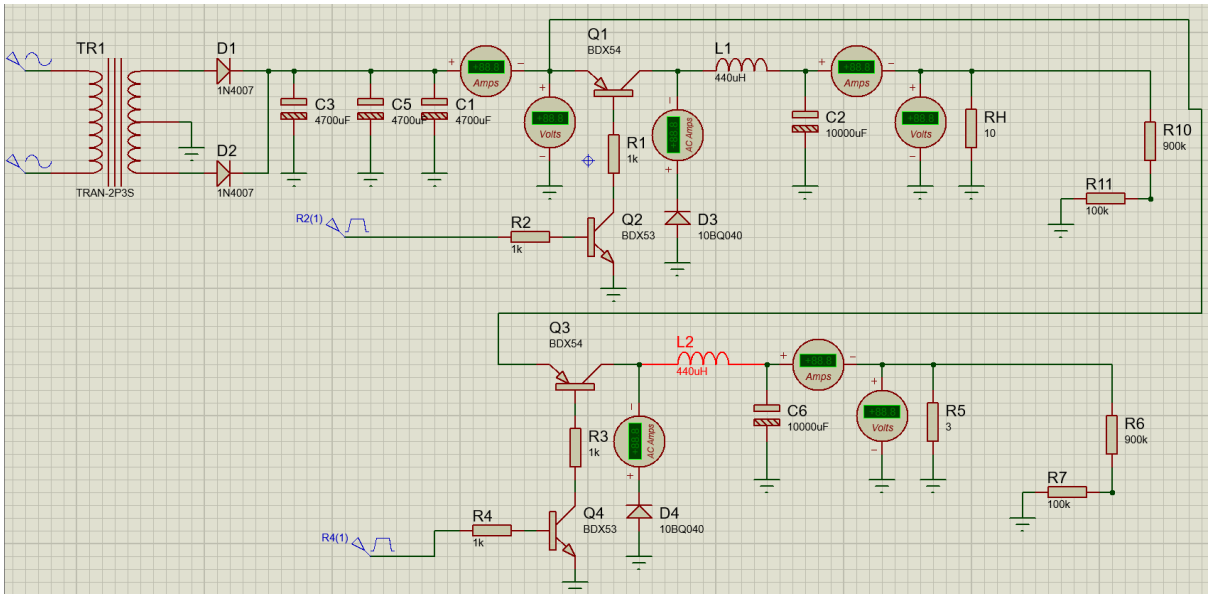


Figure 2-22: Proteus Power Supply simulation and implementation

3.1.2 Mode of Operation:

The mode of operation of the coffee machine involves several key steps:

-User Interaction: The operation begins when the user interacts with the machine's interface to select their desired coffee beverage and customize it according to their preferences. This may include specifying parameters such as beverage type (e.g., cold, hot), strength, size, and additional ingredients (e.g., Coffee mate, sugar).

-Ingredient Dispensing: Once the user has made their selection, the machine initiates the ingredient dispensing process. Depending on the chosen beverage, the machine dispenses the appropriate quantities of coffee, water, coffee mate, and other additives from their respective containers.

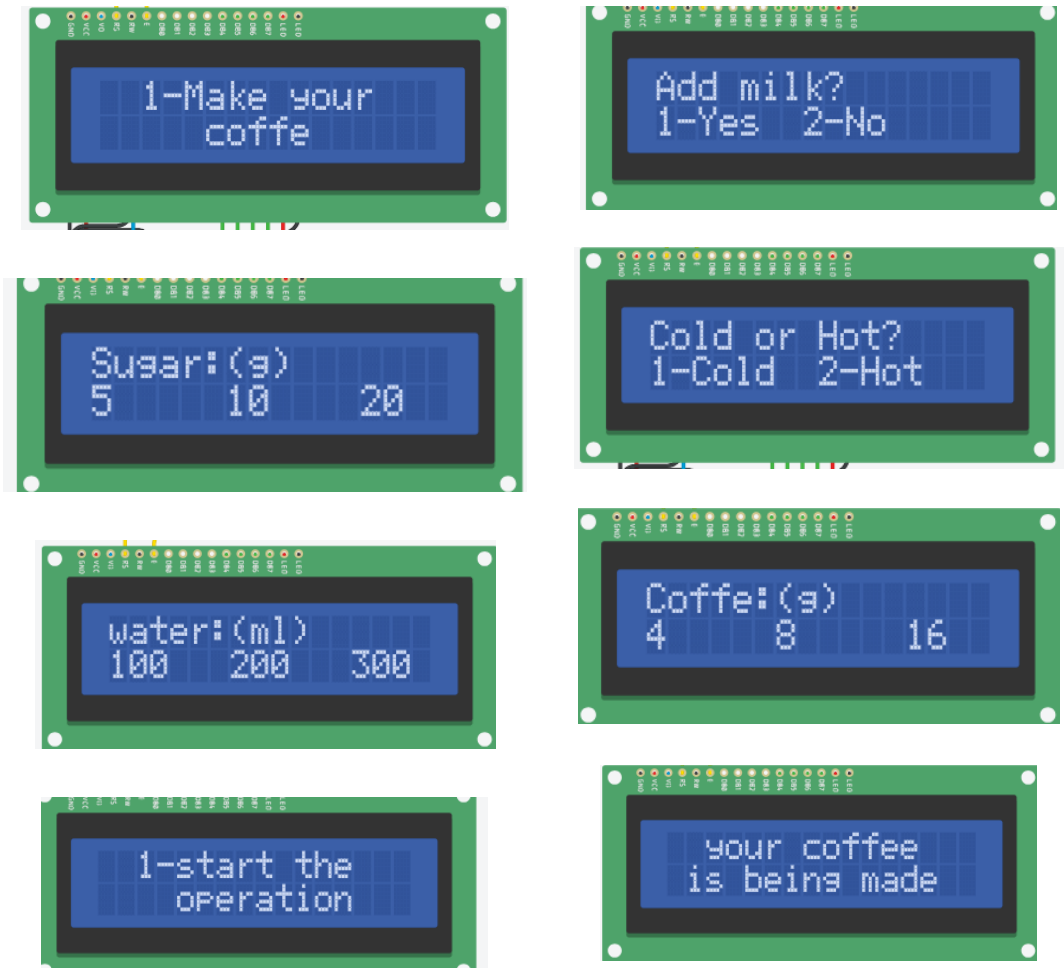
-Brewing Process: After all ingredients have been dispensed, the machine begins adding water to the cup (hot or cold). This involves taking cold water from the container into the boiler through a pump heating it to the optimal temperature and passing it into the cup. All of the ingredient’s weight are monitored by the load cell to ensure perfect quantity of each ingredient is dropped.

-Mixing and Preparation: The machine utilizes its robotic arm to mix the dispensed ingredients thoroughly, ensuring a well-blended and consistent beverage.

-Alerts and Notifications: Throughout the operation, the machine continuously monitors ingredient levels using sensors. If any ingredient is running low, the machine alerts the user through the interface, and stops until the ingredient is refilled and then it continues from where it stopped.

3.1.3 Working sequence:

1- The coffee ordering process in our machine is initiated by the user pressing the designated button "1" to commence the coffee-making sequence. Subsequently, the user is prompted to select various customization options, starting with the choice to add milk. Following this selection, the user then specifies whether they prefer their beverage to be served cold or hot and proceeds to input the desired quantity of sugar. The user is then prompted to input the quantity of water and coffee desired. Upon confirming their selections by pressing "1" again, the order is finalized.



Figures 2-22, 23, 24, 25, 26, 27, 28,29: User Interface

2- Throughout the entire ordering process, the machine employs infrared sensors to continuously monitor the presence of the cup, ensuring operational safety. Additionally, ultrasonic sensors are utilized to monitor ingredient levels, promptly alerting the user in the event of any ingredient falling below a predefined threshold. In such instances, the user is notified to replenish the ingredient accordingly.

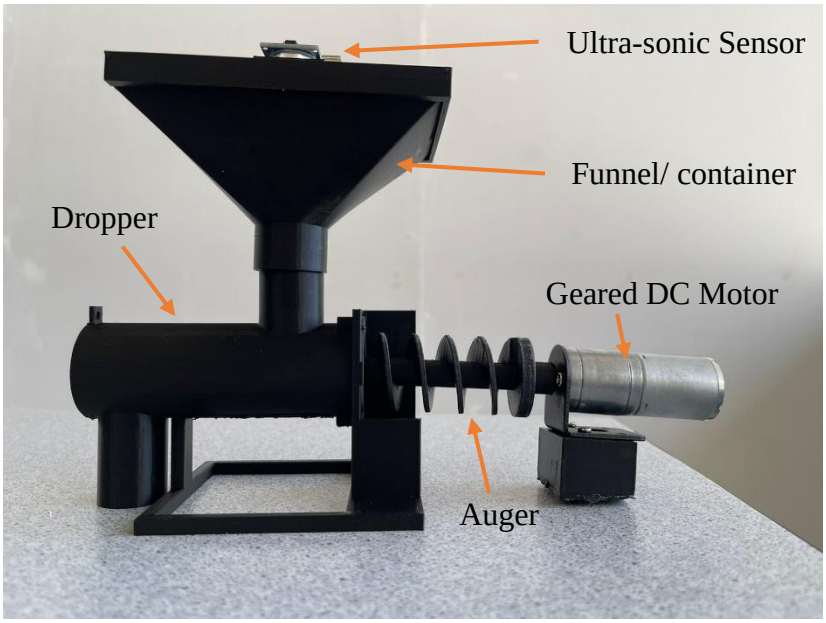


Figure 2-30: Material dropper Mechanism

3- Upon confirmation of the order and verification that all ingredients are within acceptable levels, the coffee-making process commences. The machine systematically dispenses powdered ingredients into the cup, meticulously ensuring the precise quantity of each ingredient using a load cell sensor. Once all powders are dispensed, the machine proceeds to either dispense cold water directly into the cup or pumps water into the boiler for heating. The heated water is then pumped into the cup using a dedicated pump mechanism.

4- Upon completion of the brewing process, a robotic arm, precisely controlled by two servo motors and equipped with a mixing motor, delicately mixes the beverage to ensure uniform consistency. Subsequently, the robotic arm returns to its original position, signaling the completion of the preparation process. A buzzer emits an audible signal to indicate that the drink is ready for consumption. Additionally, the buzzer serves as a safety mechanism to alert the user if the cup is not properly positioned. A LED indicator illuminates to signify the readiness of the coffee, while red LEDs indicate the status of ingredient levels, ensuring a user-friendly and intuitive experience. Overall, the mode of operation of the coffee machine is designed to be intuitive, efficient, and user-friendly, providing a seamless coffee-making experience from start to finish. The following flow chart illustrate the working process.

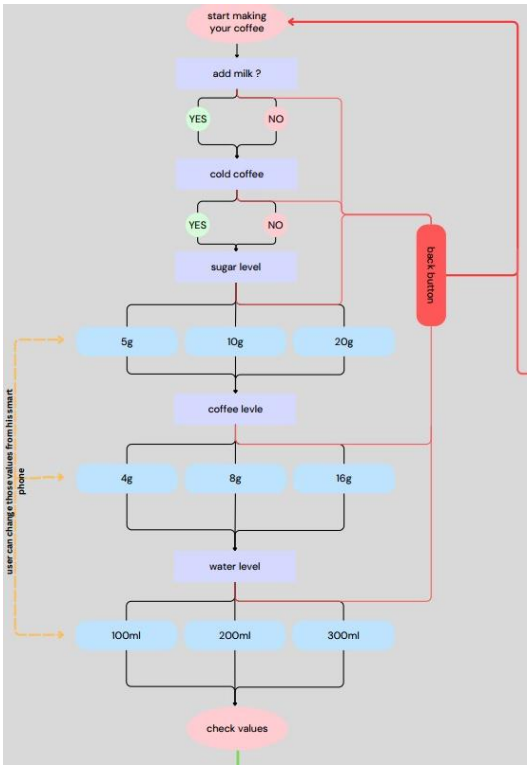


Figure 2-31: Flow chart part 1

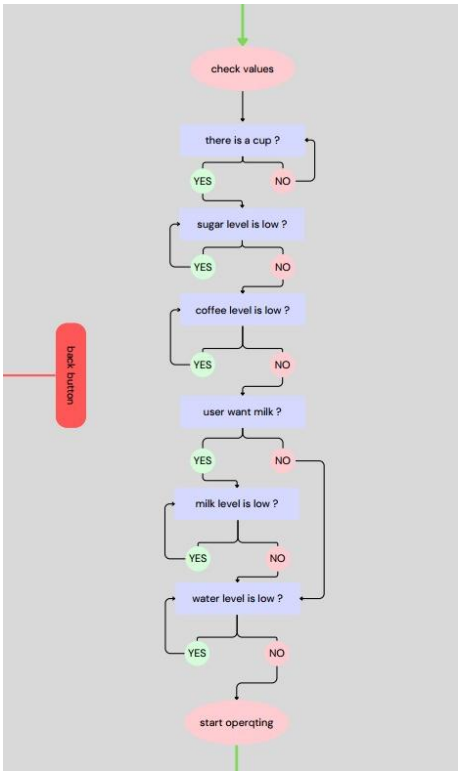


Figure 2-32: Flow chart part 2

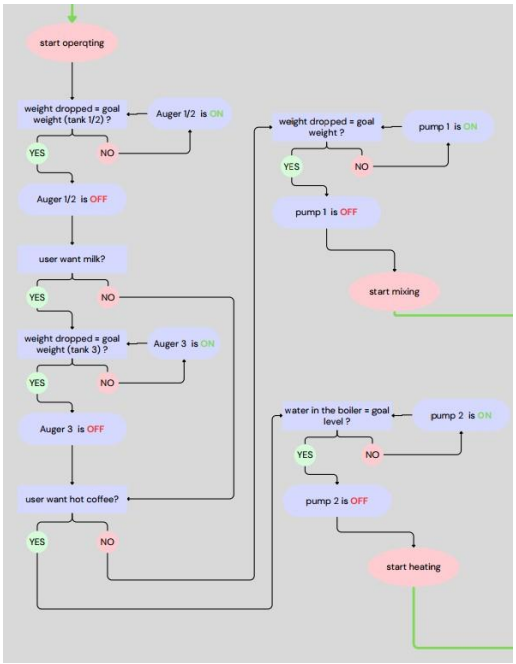


Figure 2-33: Flow chart part 3

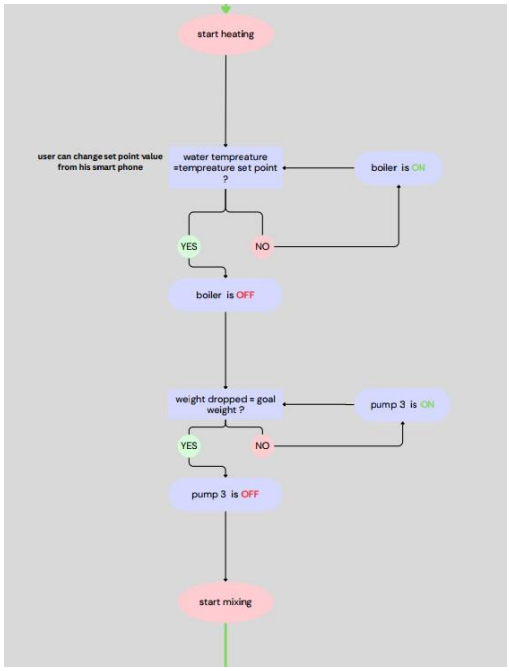


Figure 2-34: Flow chart part 4

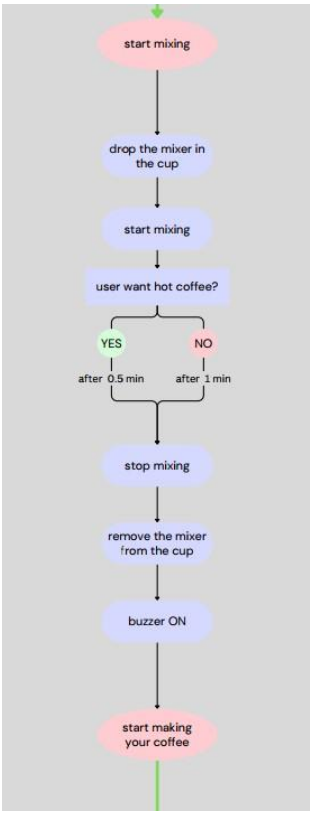


Figure 2-35: Flow chart part 5

The Following Picture shows the front of the machine including the mixer, user interface, nozzle, cup, and load cell.

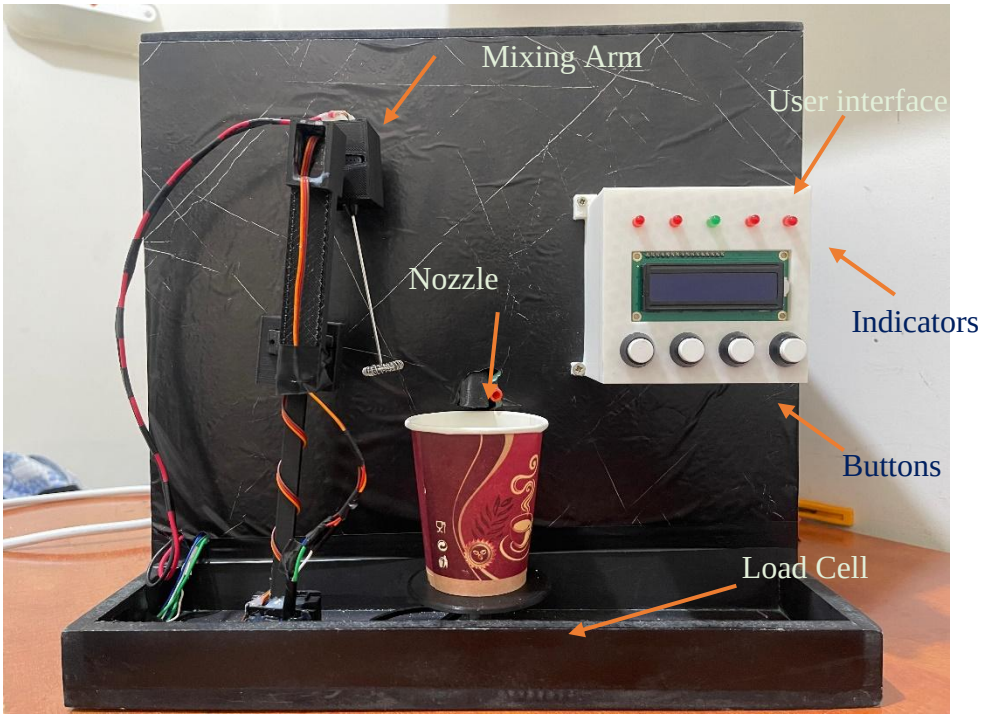


Figure 2-36: Front of The Machine

3.1.5 Safety:

- Cup Detector: In our safety measures, we prioritize user protection and equipment integrity. The Infra-red sensor ensures no liquid nor product drops when a cup is not detected. , the IR sensor acts as a guardian, ensuring that your coffee machine dispenses into the waiting cup, without any unintended drops

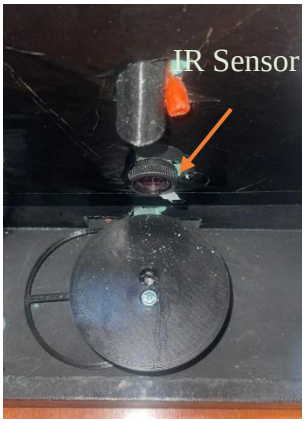


Figure 2-37: IR cup detector

- This circuit breaker serves as a safeguard against electrical overloads or faults that may occur during operation. In the event of an electrical fault, such as a short circuit or excessive current flow, the circuit breaker automatically interrupts the electrical circuit, preventing potential damage to the machine and minimizing the risk of electrical hazards.

- the pumps are connected using relays in a way that ensures no 2 relays work together for more safety and ensure that ingredients don’t mix in the weighting process.

Figure 2-38: Relays connection

3.1.6 Mobile Application:

The mobile application serves as a versatile tool for users to personalize their coffee-making experience according to their preferences. Through the application, users have the flexibility to adjust the quantities and values of each ingredient and level to tailor their beverages to their exact specifications. Whether it's adjusting the strength of the coffee, the amount of milk or sugar, or even customizing the temperature or brewing time.

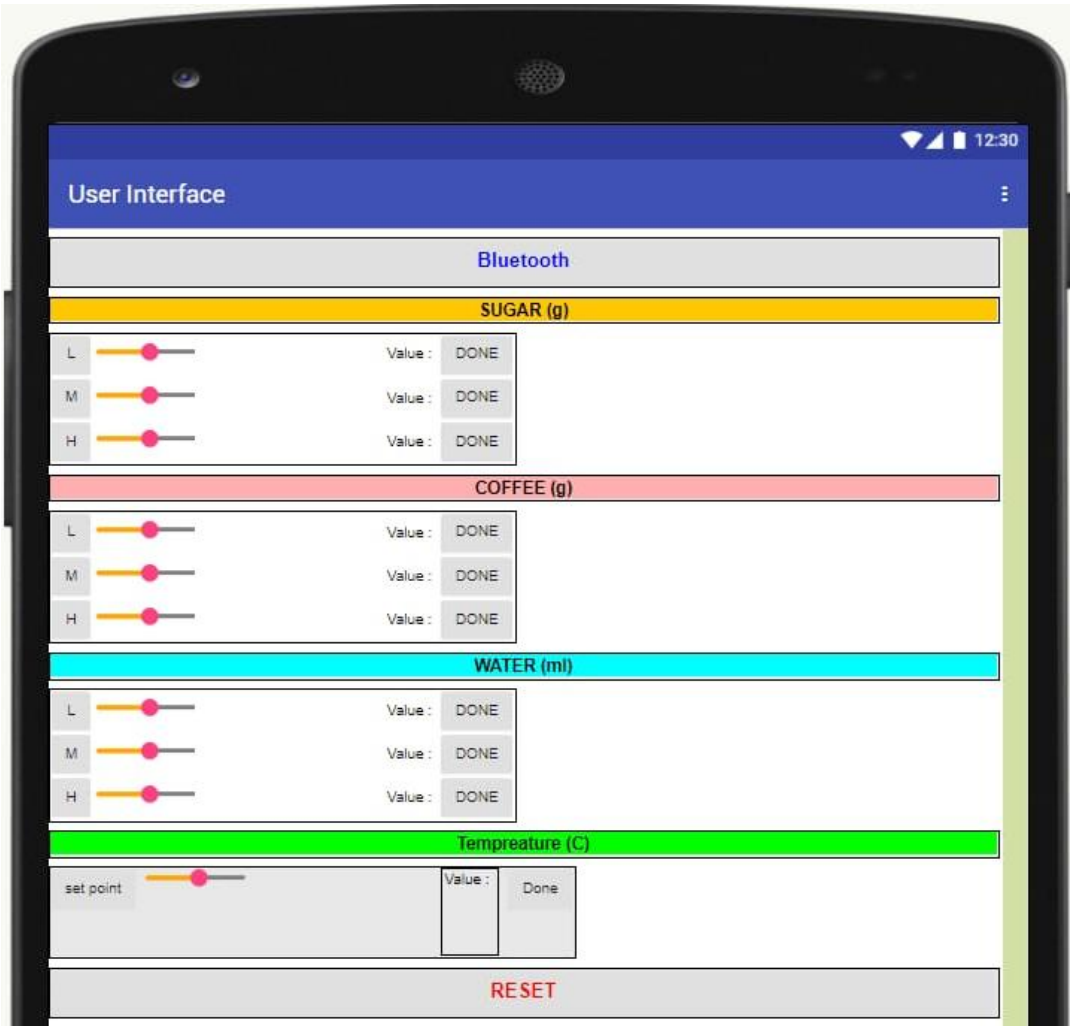


Figure 2-39: Mobile Application Interface

3.2 Compliance of Final Design with Set Constraints

- The development of our Automated Coffee Machine was guided by a set of stringent design constraints aimed at achieving technical excellence, economic feasibility, and environmental responsibility. As we progressed from conceptualization to prototype development, we

meticulously evaluated our adherence to these parameters. Below, we provide an assessment of how the final design aligns with the defined constraints.

- Weight Limitation: The final prototype of our coffee machine weighs in at 6 kilograms, which falls within the initial weight limit set at 4 kilograms when empty to 6-7 kilograms when fully loaded. This weight range ensures ease of handling and transportation while accommodating the necessary components for brewing.

- Dimension Restriction: Our coffee machine design successfully complies with the constraint of compact dimensions, measuring 40cm in length, 40cm in width, and 35cm in height. These dimensions allow the machine to fit comfortably on average kitchen countertops while maximizing space efficiency.

- Energy Efficiency: A key success of our project lies in achieving energy efficiency through the incorporation of a switch-mode power supply with a toroidal transformer. This design choice ensures high efficiency, lower energy consumption, and reduced environmental impact during operation.

- Economic Considerations: While economic constraints posed a challenge, our coffee machine remained within the target budget of $300. Final costs amounted to $290, demonstrating our commitment to affordability without compromising on quality or functionality.

- Environmental Sustainability: Our project excelled in meeting the environmental sustainability constraint by utilizing recyclable filament in the 3D printing process. This eco-conscious approach minimizes the ecological footprint of the manufacturing process and promotes sustainable practices in material usage.

3.3 Applicable Codes and Standards

1- IEC 60335-1: General Requirements: IEC 60335-1 deals with the safety of electrical appliances for household and similar purposes. It covers appliances with a rated voltage of: Not more than 250 V for single-phase appliances. IEC 60335-1 ensures that household appliances are safe for use by all individuals. [6]

2- IEC 60204-1: Safety of Machinery - Electrical Equipment of Machines During the implementation of electrical systems, this standard ensures that all wiring, components, and assemblies adhere to safety requirements. Protocols for proper insulation, grounding, and protection against overcurrent and short circuits are established to safeguard against electrical hazards. [5]

3- ISO 13849-1: Safety of Machinery - Safety-Related Parts of Control Systems. This standard is crucial for designing and integrating the safety-related parts of the control system used in the ping pong trainer, ensuring the system functions correctly in response to inputs, including fault conditions. [4]

REFERENCES LIST

[1] Faema E61 Legend. (n.d.). Retrieved from Absolute Espresso website: https://www.absoluteespresso.com/products/faema-e61-legend?variant=32271047884894.

[2] Great Northern Coffee Roasting Co. (n.d.). Retrieved from https://www.greatnortherncoffee.com/em_sync.html.

[3] Wired. (n.d.). The Best Latte and Cappuccino Machines of 2022. Retrieved from https://www.wired.com/story/best-latte-and-cappuccino-machines/.

[4] ISO. (2015). Safety of machinery - Safety-related parts of control systems - Part 1: General principles for design (ISO 13849-1:2015). International Organization for Standardization. https://www.iso.org/obp/ui/#iso:std:iso:13849:-1:ed-4:v1:en

[5] IEC. (2016). Safety of machinery - Electrical equipment of machines - Part 1: General requirements (6th ed.). International Electrotechnical Commission. https://webstore.iec.ch/preview/info_iec60204-1%7Bed6.0.RLV%7Den.pdf

[6] IEC 60335-1: the Standard for the Safety of Household and Similar Electrical Appliances | UL Solutions. (n.d.). UL Solutions. https://www.ul.com/services/iec-60335-1-standard-safety-household-and-similar-electrical appliances

APPENDIX A – BILL OF MATERIALS

The following table shows all the components purchased by our team used while implementing the project, their quantities, and prices.

Components	Price
Arduino MEGA 2560 R3 + USB Cable	25
Stepper Motor 5V 4-Phase + ULN2003 Driver	9
HC-SR04 Ultrasonic Sensor 4pin	6
HC 05 HC05 6pin Bluetooth Module	5
Ultrasonic Sensor Kit | Range Detection 2.5M | Waterproof | AJ-SR04M	6
Water Pump | 5.5 ~ 12V | DC-1020	16.5
CNC Spare Parts | Spindle | Water Pipe | 8mm | Per Meter	0.9*10
Weight Load Cell for HX711 Sensor 1Kg	3
LCD 1602 IIC/I2C Blue Backlight	6
Active Buzzer Module	0.4
Servo Motor MG90s 180 Degree	5*2
1 Channel 5V Relay Module With LED	3*4
DS18B20 Waterproof Temperature Sensor	2
wood - 3d printing	120
diodes 1n5408	0.6*4
capacitors 4700uf 35v	1.8*4
capacitors 10000uf 25v	3.5*2
inductors 350uh 5A	1.5*2
diodes schotkey 1n5821	1.3*2
npn transistor TIP41C	0.7*2
pnp transistor TIP36C	1.8*2
Transformer Toroidal AC | 220V - 12V | 100VA	25
Arduino Pro Mini Wavgat ATmega168 16Mhz	6
Serial Module |Mini USB B - TTL Converter | FTDI	2

TABLE 6 - BILL OF MATERIALS

Appendix B – Arduino Code

```
#include <LiquidCrystal_I2C.h>
```

```
#include <Stepper.h>
#include <Ultrasonic.h>
#include <Servo.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <HX711_ADC.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);  // A5 scl     A4 sda

Servo myservo;
int servo_pin = 2;

HX711_ADC LoadCell(35, 36);  // dt pin, sck pin

OneWire oneWire(37);
DallasTemperature sensors(&oneWire);

Ultrasonic UR_milk(4, 5);
Ultrasonic UR_sugar(6, 7);
Ultrasonic UR_water(8, 9);
Ultrasonic UR_cofee(10, 11);

const int stepsPerRevolution = 2038;
Stepper myStepper_sugar = Stepper(stepsPerRevolution, 14, 16, 15, 17);  //stepper
Stepper myStepper_coffee = Stepper(stepsPerRevolution, 18, 20, 19, 21);
Stepper myStepper_milk = Stepper(stepsPerRevolution, 22, 24, 23, 25);

int bt_1 = 3, bt_2 = 4, bt_3 = 5, bt_4 = 6;
int ir_s_cup = 12;  // ir sensors
int tank_toboiler_pump = 30;
```

```
int tank_tocup_pump_cold = 31;
int tank_tocup_pump_hot = 32;
int boiler = 13;
int mixer_motor = 33;


int mixer =9;
int mixer_GND = 10 ;


int load_cell = A0;
int tempreature_sens = A1;
int LED1 = 50, LED2 = 53, LED3 = 52, LED4 = 51, LED5 = 48;
bool TEMP_SETUP = 0;

bool TEMP_CHOOSING_SCREEN = 0;
bool temp_start_choosing_milk_or_not = 0, temp_start_choosing_milk_or_not_2 = 0;
bool temp_start_choosing_hot_or_cold = 0, temp_start_choosing_hot_or_cold_2 = 0;
bool temp_start_choosing_sugar_level = 0, temp_start_choosing_sugar_level_2 = 0;
bool temp_start_choosing_water_level = 0, temp_start_choosing_water_level_2 = 0;
bool temp_start_choosing_coffe_level = 0, temp_start_choosing_coffe_level_2 = 0;
int milk_level = -1, milk_selected_value = -1, hot_cold_selected_value = -1,
sugar_level_selection = -1, water_level_selection = -1, coffe_level_selection = -1;

bool TEMP_START_OPERATING = 0;

unsigned long delay_value_between_each_push = 200;  //ms

int L_sugar = 5, M_sugar = 10, H_sugar = 20;      //unit : g
int L_water = 100, M_water = 200, H_water = 300;  //unit : ml
int L_coffee = 4, M_coffee = 8, H_coffee = 16;    //unit : g
```

```
int coffee_tempreature_setpoint = 60;



bool TEMP_CHECKING_VALUES = 0;

bool temp_check_sugar = 0, temp_check_if_theire_isa_cup = 0, temp_check_cofee = 0, temp_check_milk = 0, temp_check_water = 0;


bool TEMP_START_OPERATING_BUTTON = 0;

bool temp_stop_bt_4_back_button = 0;


bool TEMP_DROPING_VALUES = 0;

bool temp_drop_sugar = 0, temp_drop_coffee = 0, temp_drop_milk = 0, temp_drop_water = 0;


bool temp_start_heating = 0;

bool temp_drop_hotwater_to_cup = 0;

bool temp_start_mixing = 0;

bool temp_removing_cup = 0;

void drop_the_required_ingredients(int& powder_level_selection, String name_of_powder, Stepper& stepper_name, bool& end_curent_temp, bool& next_temp);


int temp = 0;  //just to fill some spaces on a functions

bool temp_bool = 0;

bool limiting_repeating_same_screen_many_time = 0;

//for bluetooth connection:

char data_received = 0;

bool temp_changing_L_value = 0, temp_changing_M_value = 0, temp_changing_H_value = 0;

bool temp_reset_after_changing_values = 0;


void setup() {
```

```
sensors.begin();
LoadCell.begin();
LoadCell.start(1000);
LoadCell.setCalFactor(475);

pinMode(bt_1, INPUT_PULLUP);
pinMode(bt_2, INPUT_PULLUP);
pinMode(bt_3, INPUT_PULLUP);
pinMode(bt_4, INPUT_PULLUP);
pinMode(ir_s_cup, INPUT_PULLUP);
pinMode(tank_toboiler_pump, OUTPUT);
pinMode(tank_tocup_pump_cold, OUTPUT);
pinMode(tank_tocup_pump_hot, OUTPUT);
pinMode(boiler, OUTPUT);
pinMode(LED1, OUTPUT);
pinMode(LED2, OUTPUT);
pinMode(LED3, OUTPUT);
pinMode(LED4, OUTPUT);
pinMode(LED5, OUTPUT);
myservo.write(0);  //initial postition of my servo
pinMode(mixer_motor, OUTPUT);
pinMode(buzzer, OUTPUT);
Serial.begin(9600);
lcd.init();
lcd.init();
lcd.backlight();
lcd.noCursor();
lcd.noBlink();
lcd.setCursor(5, 0);
lcd.print("hello!");
```

```
  delay(500);
  lcd.clear();
}

void loop() {
  bool button_1_value = digitalRead(bt_1), button_2_value = digitalRead(bt_2),
button_3_value = digitalRead(bt_3), button_4_value = digitalRead(bt_4);



  if ((Serial.available() > 0) && temp_stop_bt_4_back_button == 0) {  //the user can change
the values of the ingredients if he is not in the operating screen
    recieving_data('L_sugar_availebel', 'M_sugar_availebel', 'H_sugar_availebel', L_sugar,
M_sugar, H_sugar);
    recieving_data('L_coffee_availebel', 'M_coffee_availebel', 'H_coffee_availebel', L_coffee,
M_coffee, H_coffee);
    recieving_data('L_water_availebel', 'M_water_availebel', 'H_water_availebel', L_water,
M_water, H_water);
    recieving_data('coffee_tempreature_availebell', 'nothing', 'nothing',
coffee_tempreature_setpoint, temp, temp);
  }


  if ((button_4_value == 0 || temp_reset_after_changing_values == 1) &&
temp_stop_bt_4_back_button == 0) {  // return to home main screen
    temp_reset_after_changing_values = 0;
    TEMP_SETUP = 0;
    TEMP_CHECKING_VALUES = 0;
    temp_start_choosing_milk_or_not = 0;
    temp_start_choosing_milk_or_not_2 = 0;
    temp_start_choosing_hot_or_cold = 0;
    temp_start_choosing_hot_or_cold_2 = 0;
    temp_start_choosing_sugar_level = 0;
    temp_start_choosing_sugar_level_2 = 0;
    temp_start_choosing_water_level = 0;
```

```
  temp_start_choosing_water_level_2 = 0;
  temp_start_choosing_coffe_level = 0;
  temp_start_choosing_coffe_level_2 = 0;
  milk_selected_value = -1;
  hot_cold_selected_value = -1;
  sugar_level_selection = -1;
  water_level_selection = -1;
  coffe_level_selection = -1;
  TEMP_CHECKING_VALUES = 0;
  temp_check_sugar = 0;
  temp_check_cofee = 0;
  temp_check_milk = 0;
  temp_check_water = 0;
  TEMP_START_OPERATING_BUTTON = 0;
  TEMP_START_OPERATING = 0;
  lcd.clear();
 }

 if (TEMP_SETUP == 0) {  // main screen
  lcd.clear();
  lcd.setCursor(2, 0);
  lcd.print("1-Make your ");
  lcd.setCursor(5, 1);
  lcd.print("coffe ");
  TEMP_SETUP = 1;
  temp_start_choosing_milk_or_not = 1;
  TEMP_CHOOSING_SCREEN = 1;
  delay(delay_value_between_each_push);
 }
```

```
  else if (TEMP_CHOOSING_SCREEN == 1) {  // choosing the values of your coffevoid choosing_value_screen(String first_line,String second_line,  int& L_level,int& M_level,int& H_level, int& end_curent_temp, int& next_temp, int 2or3_value_screen, int g0_ml1)


    if (temp_start_choosing_milk_or_not == 1 && button_1_value == 0) choosing_value_screen("Add milk?", "1-Yes  2-No", temp, temp, temp, temp_start_choosing_milk_or_not, temp_start_choosing_milk_or_not_2, 0, 0);

    else if (temp_start_choosing_milk_or_not_2 == 1) return_2option_value(milk_selected_value, temp_start_choosing_milk_or_not_2, temp_start_choosing_hot_or_cold);


    else if (temp_start_choosing_hot_or_cold == 1) choosing_value_screen("Cold or Hot?", "1-Cold  2-Hot", temp, temp, temp, temp_start_choosing_hot_or_cold, temp_start_choosing_hot_or_cold_2, 0, 0);

    else if (temp_start_choosing_hot_or_cold_2 == 1) return_2option_value(hot_cold_selected_value, temp_start_choosing_hot_or_cold_2, temp_start_choosing_sugar_level);


    else if (temp_start_choosing_sugar_level == 1) choosing_value_screen("Sugar:", "1tsp  2tsp  3tsp", L_sugar, M_sugar, H_sugar, temp_start_choosing_sugar_level, temp_start_choosing_sugar_level_2, 1, 0);

    else if (temp_start_choosing_sugar_level_2 == 1) return_3option_value(sugar_level_selection, temp_start_choosing_sugar_level_2, temp_start_choosing_water_level, L_sugar, M_sugar, H_sugar, temp_bool, temp_bool);


    else if (temp_start_choosing_water_level == 1) choosing_value_screen("water:", "1-L  2-M  3-H ", L_water, M_water, H_water, temp_start_choosing_water_level, temp_start_choosing_water_level_2, 1, 1);

    else if (temp_start_choosing_water_level_2 == 1) return_3option_value(water_level_selection, temp_start_choosing_water_level_2, temp_start_choosing_coffe_level, L_water, M_water, H_water, temp_bool, temp_bool);


    else if (temp_start_choosing_coffe_level == 1) choosing_value_screen("Coffe:", "1-L  2-M  3-H ", L_coffee, M_coffee, H_coffee, temp_start_choosing_coffe_level, temp_start_choosing_coffe_level_2, 1, 0);

    else if (temp_start_choosing_coffe_level_2 == 1) return_3option_value(coffe_level_selection, temp_start_choosing_coffe_level_2,
```

```
temp_check_sugar, L_coffee, M_coffee, H_coffee, TEMP_CHOOSING_SCREEN,
TEMP_CHECKING_VALUES);

  }


  else if (TEMP_CHECKING_VALUES == 1) {  // check if we have some low value of the
engredients


    if (temp_check_sugar == 1) check_values_of_engrediens_UR(UR_sugar, "sugar",
temp_check_sugar, temp_check_if_theire_isa_cup);  // i can let this be else if

    else if (temp_check_if_theire_isa_cup == 1) check_values_of_engrediens_IR(ir_s_cup,
"Cup", temp_check_if_theire_isa_cup, temp_check_cofee);

    else if (temp_check_cofee == 1) check_values_of_engrediens_UR(UR_cofee, "coffee",
temp_check_cofee, temp_check_milk);

    else if (milk_selected_value == 0 && temp_check_milk == 1)
check_values_of_engrediens_UR(UR_milk, "milk", temp_check_milk, temp_check_water);
// 0 mean yes i want milk

    else if (milk_selected_value == 1 && temp_check_milk == 1) {

      temp_check_water = 1;  //1 mean i dont want milk

      temp_check_milk = 0;

    } else if (temp_check_water == 1) check_value_of_water(temp_check_water,
TEMP_START_OPERATING_BUTTON, TEMP_CHECKING_VALUES);

  }


  else if (TEMP_START_OPERATING_BUTTON == 1) {  // pressing 1 to start operating
screen

    lcd.clear();

    lcd.setCursor(2, 0);

    lcd.print("1-start the ");

    lcd.setCursor(4, 1);

    lcd.print("operation");

    end_start_temp(TEMP_START_OPERATING_BUTTON,
TEMP_START_OPERATING);

  }
```

```
  else if (TEMP_START_OPERATING == 1 && button_1_value == 0) {  // stop the back button and start operating
    waiting_screen();
    TEMP_START_OPERATING = 0;
    temp_drop_sugar = 1;
    temp_stop_bt_4_back_button = 1;
    TEMP_DROPING_VALUES = 1;

Serial.println("_____ ");
    Serial.print("milk selected value: ");
    Serial.println(milk_selected_value);
    Serial.print("hot cold: ");
    Serial.println(hot_cold_selected_value);
    Serial.print("sugar selected value: ");
    Serial.println(sugar_level_selection);
    Serial.print("water selected value: ");
    Serial.println(water_level_selection);
    Serial.print("coffee selected value: ");
    Serial.println(coffe_level_selection);

Serial.println("_____ ");
  }


  else if (TEMP_DROPING_VALUES == 1) {
    if (temp_drop_sugar == 1) drop_the_required_ingredients(sugar_level_selection, "sugar", myStepper_sugar, temp_drop_sugar, temp_drop_coffee);
    else if (temp_drop_coffee == 1) drop_the_required_ingredients(coffe_level_selection, "coffee", myStepper_coffee, temp_drop_coffee, temp_drop_milk);
    // 0= want milk 1= cont want
    else if (milk_selected_value == 0 && temp_drop_milk == 1) {
```

```
    milk_level = water_level_selection / 15;  //15 can be cahnge, we need a linear relation between the water level and milk

    drop_the_required_ingredients(milk_level, "milk", myStepper_milk, temp_drop_milk, temp_drop_water);

  } else if (milk_selected_value == 1 && temp_drop_milk == 1) end_start_temp(temp_drop_milk, temp_drop_water);


  //hot water  =1 cold = 0

  else if (hot_cold_selected_value == 1 && temp_drop_water == 1) drop_the_required_water_ultrasonic(TEMP_DROPING_VALUES, temp_drop_water, temp_start_heating);

  else if (hot_cold_selected_value == 0 && temp_drop_water == 1) drop_the_required_water_load_cell(tank_tocup_pump_cold, TEMP_DROPING_VALUES, temp_drop_water, temp_start_heating);

 }


 //hot water  =1 cold = 0

 else if (temp_start_heating == 1 && hot_cold_selected_value == 1)

  start_heating(temp_start_heating, temp_drop_hotwater_to_cup);

 else if (temp_start_heating == 1 && hot_cold_selected_value == 0) end_start_temp(temp_start_heating, temp_start_mixing);


 else if (temp_drop_hotwater_to_cup == 1) drop_the_required_water_load_cell(tank_tocup_pump_hot, temp_bool, temp_drop_hotwater_to_cup, temp_start_mixing);


 else if (temp_start_mixing == 1) start_mixing(temp_start_mixing, temp_removing_cup);


 else if (temp_removing_cup == 1) /// 0=no object  1=ther is a object/ {

  bool there_is_a_cup_y_n = digitalRead(ir_s_cup);

  if (there_is_a_cup_y_n == 0) {

   temp_removing_cup = 0;

   TEMP_SETUP = 0;                //back to main screen

   temp_stop_bt_4_back_button = 0;  // we can use back button again
```

```
    }
  }
}

void choosing_value_screen(String first_line, String second_line, int& L_level, int&
M_level, int& H_level, bool& end_curent_temp, bool& next_temp, int value_screen_option,
int g0_ml1) {
  String unit = "";
  if (g0_ml1 == 0) unit = "g";
  else if (g0_ml1 == 1) unit = "ml";

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print(first_line);

  if (value_screen_option == 0) {
    lcd.setCursor(0, 1);
    lcd.print(second_line);
  } else if (value_screen_option == 1) {
    lcd.print("(");
    lcd.print(unit);
    lcd.print(")");
    lcd.setCursor(0, 1);
    lcd.print(L_level);

    lcd.setCursor(6, 1);
    lcd.print(M_level);

    lcd.setCursor(12, 1);
    lcd.print(H_level);
  }
```

```
  end_start_temp(end_curent_temp, next_temp);
  delay(delay_value_between_each_push);
}
void return_2option_value(int& value, bool& end_curent_temp, bool& next_temp) {
  int button_1_value = digitalRead(bt_1), button_2_value = digitalRead(bt_2),
button_3_value = digitalRead(bt_3), button_4_value = digitalRead(bt_4);
  if (button_1_value == 0) {
    value = 0;
    end_start_temp(end_curent_temp, next_temp);
  } else if (button_2_value == 0) {
    value = 1;
    end_start_temp(end_curent_temp, next_temp);
  }
}
void return_3option_value(int& value, bool& end_curent_temp, bool& next_temp, int&
Low_value, int& Medium_value, int& Max_value, bool& end_overall_temp, bool&
start_next_overall_temp) {
  int button_1_value = digitalRead(bt_1), button_2_value = digitalRead(bt_2),
button_3_value = digitalRead(bt_3), button_4_value = digitalRead(bt_4);
  if (button_1_value == 0) {
    value = Low_value;
    end_start_temp(end_curent_temp, next_temp);
    end_start_temp(end_overall_temp, start_next_overall_temp);
  } else if (button_2_value == 0) {
    value = Medium_value;
    end_start_temp(end_curent_temp, next_temp);
    end_start_temp(end_overall_temp, start_next_overall_temp);
  } else if (button_3_value == 0) {
    value = Max_value;
    end_start_temp(end_curent_temp, next_temp);
    end_start_temp(end_overall_temp, start_next_overall_temp);
  }
```

```
}
void waiting_screen() {
  lcd.clear();
  lcd.setCursor(2, 0);
  lcd.print("your coffee");
  lcd.setCursor(1, 1);
  lcd.print("is being made");
}
void low_level_screen(String tank_type, int different_error) {
  lcd.clear();
  if (different_error == 0) {
    lcd.setCursor(3, 0);
    lcd.print("LOW ");
    lcd.print(tank_type);
    lcd.setCursor(1, 1);
    lcd.print("please refill ");
    delay(200);

  } else if (different_error == 1) {
    lcd.setCursor(6, 0);
    lcd.print(tank_type);
    lcd.setCursor(2, 1);
    lcd.print("not detected");
    delay(200);
  }
}
void check_values_of_engrediens_IR(int& sensor_pin_number, String name_of_powder,
bool& end_curent_temp, bool& next_temp) {
  int ir_object_check = digitalRead(sensor_pin_number);
  if (ir_object_check == 0) low_level_screen("cup", 1);  // 0=no object  1=ther is a object
  else end_start_temp(end_curent_temp, next_temp);
```

```
}
void check_values_of_engrediens_UR(Ultrasonic& ultrasonic, String name_of_powder,
bool& end_curent_temp, bool& next_temp) {
  int ultrasinoc_low_level_check = ultrasonic.read();
  if (ultrasinoc_low_level_check >= 10) low_level_screen(name_of_powder, 0);
  else end_start_temp(end_curent_temp, next_temp);
}


void drop_the_required_ingredients(int& powder_level_selection, String name_of_powder,
Stepper& stepper_name, bool& end_curent_temp, bool& next_temp) {
  LoadCell.update();
  float weight = LoadCell.getData();


  float reference_mass_of_load_cell = weight /* = calculate_mass_of_the_cup()*/;
  float new_mass_of_load_cell = reference_mass_of_load_cell;


  stepper_name.setSpeed(15);
  while (abs(reference_mass_of_load_cell - new_mass_of_load_cell) <
powder_level_selection * 0.8) {
    int ir_sens = digitalRead(ir_s_cup);
    if (ir_sens == 0) emergency_alarm();
    else {
      if (limiting_repeating_same_screen_many_time == 1) {
        waiting_screen();
        limiting_repeating_same_screen_many_time = 0;
      }
      Serial.print("droping ");
      Serial.println(name_of_powder);
      LoadCell.update();
      weight = LoadCell.getData();
      new_mass_of_load_cell = weight /* = calculate_mass_of_the_cup()*/;
```

```
      Serial.print("weight of the cup: ");
      Serial.println(weight);
    }
  }
  end_start_temp(end_curent_temp, next_temp);
}
void check_value_of_water(bool& end_curent_temp, bool& next_temp, bool&
end_overall_temp) {
  float water_volume = calculate_water_volume();
  if (water_volume <= (water_level_selection * 1.2)) low_level_screen("Water", 0);  //water
level : high medium low
  else {
    end_start_temp(end_curent_temp, next_temp);
    end_overall_temp = 0;
  }
}
float calculate_water_volume() {
  int hight = 17, long1 = 12, width = 8;
  float distance_of_empty_Space = UR_water.read();
  if (distance_of_empty_Space > 17) distance_of_empty_Space = 17;
  else if (distance_of_empty_Space > 30) distance_of_empty_Space = 0;
  float water_volume = (hight - distance_of_empty_Space) * long1 * width;  //ml
  if (water_volume < 0) water_volume = 0;
  return water_volume;
}
void drop_the_required_water_ultrasonic(bool& end_overall_temp, bool& end_curent_temp,
bool& next_temp) {
  float reference_volume = calculate_water_volume();
  float new_volume = reference_volume;
  while (abs(reference_volume - new_volume) < water_level_selection) {
    digitalWrite(tank_toboiler_pump, HIGH);
```

```
    new_volume = calculate_water_volume();
    Serial.println("droping COLD water to boiler ");
    Serial.println(new_volume);
  }
  digitalWrite(tank_toboiler_pump, LOW);
  end_start_temp(end_curent_temp, next_temp);
  end_overall_temp = 0;
}
void drop_the_required_water_load_cell(int& pump, bool& end_overall_temp, bool&
end_curent_temp, bool& next_temp) {
  LoadCell.update();
  float weight = LoadCell.getData();

  float reference_mass_of_load_cell = weight /* = calculate_mass_of_the_cup()*/;
  float new_mass_of_load_cell = reference_mass_of_load_cell;
  while (abs(reference_mass_of_load_cell - new_mass_of_load_cell) < water_level_selection
* 0.9) {  // 0.9 that to reduce error of falling time
    bool ir_sens = digitalRead(ir_s_cup);
    if (ir_sens == 0) emergency_alarm();
    else {
      if (limiting_repeating_same_screen_many_time == 1) {
        waiting_screen();
        limiting_repeating_same_screen_many_time = 0;
      }
      Serial.println("droping water to cup");
      digitalWrite(pump, HIGH);

      LoadCell.update();
      weight = LoadCell.getData();

      new_mass_of_load_cell = weight /* = calculate_mass_of_the_cup()*/;
```

```
      Serial.print("weight of the cup: ");
      Serial.println(weight);
    }
  }
  digitalWrite(pump, LOW);
  end_overall_temp = 0;
  end_start_temp(end_curent_temp, next_temp);
}
void recieving_data(char L_availebel_char, char M_availebel_char, char H_availebel_char,
int& L_tank_type, int& M_tank_type, int& H_tank_type) {
  data_received = Serial.read();
  if (data_received == L_availebel_char) temp_changing_L_value = 1;
  else if (data_received == M_availebel_char) temp_changing_M_value = 1;
  else if (data_received == H_availebel_char) temp_changing_H_value = 1;
  else if (data_received == 'reset_values') {
    L_sugar = 5;
    M_sugar = 10;
    H_sugar = 20;
    L_water = 100;
    M_water = 200;
    H_water = 300;
    L_coffee = 4;
    M_coffee = 8;
    H_coffee = 16;
    coffee_tempreature_setpoint = 60;
  } else if (data_received == 'back') {
    temp_changing_L_value = 0;
    temp_changing_M_value = 0;
    temp_changing_H_value = 0;
  } else if (temp_changing_L_value == 1) converting_and_saving_values(data_received,
L_tank_type, temp_changing_L_value);
```

```
  else if (temp_changing_M_value == 1) converting_and_saving_values(data_received,
M_tank_type, temp_changing_M_value);
  else if (temp_changing_H_value == 1) converting_and_saving_values(data_received,
H_tank_type, temp_changing_H_value);
}
void converting_and_saving_values(char& data_received, int& igredient_new_value, bool&
end_current_temp) {
  int convert_to_integer = data_received - '0';
  igredient_new_value = convert_to_integer;
  end_current_temp = 0;
  //send a message to confirm that data recieved sucsecfully
  lcd.clear();
  lcd.setCursor(1, 0);
  lcd.print("Change Done");
  temp_reset_after_changing_values = 1;
}
void end_start_temp(bool& end_curent_temp, bool& next_temp) {
  end_curent_temp = 0;
  next_temp = 1;
}
void start_heating(bool& end_curent_temp, bool& next_temp) {
  int water_tempreature = analogRead(tempreature_sens);
  sensors.requestTemperatures();
  water_tempreature = sensors.getTempCByIndex(0);
  Serial.println("start heating ");
  Serial.println(water_tempreature);
  if (water_tempreature < coffee_tempreature_setpoint) {
    digitalWrite(boiler, HIGH);
  } else {
    digitalWrite(boiler, LOW);
    end_start_temp(end_curent_temp, next_temp);
```

```
  }
}
void start_mixing(bool& end_curent_temp, bool& next_temp) {
  Serial.println("start mixing ");
  unsigned long delay_time = 0;
  if (hot_cold_selected_value == 1) delay_time = 5000;  //when its hot it will take 30 s
  else if (hot_cold_selected_value == 0) delay_time = 6000;
  //myservo.attach(29);
  myservo.write(60);               //servo on ;
  digitalWrite(mixer_motor, HIGH);  //dc on
  delay(delay_time);
  digitalWrite(mixer_motor, LOW);  //dc off
  myservo.write(0);               //servo back to initial position
  Done_screen();
  end_start_temp(end_curent_temp, next_temp);
}
void Done_screen() {
  //lcd print that cofffe is done
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("<Your coffee is>");
  lcd.setCursor(0, 1);
  lcd.print(">>>>>READY!<<<<<");
  for (int i = 0; i < 3; i++) {
    tone(buzzer, 380, 250);
    delay(300);
    tone(buzzer, 400, 250);
    delay(300);
    noTone(buzzer);  // Stop sound...
    delay(800);
```

```
  }
}
void emergency_alarm() {
  const int startFreq = 523;  // C5 is 523Hz
  low_level_screen("cup", 1);
  limiting_repeating_same_screen_many_time = 1;
  for (int freq = startFreq; freq <= (startFreq * 2); ++freq) {
    tone(buzzer, freq, 10);
    delay(3);
  }
}
```

Appendix C – Mobile Code