

Smart Liquid Dispenser Project Report

Presented by
Nishantkumar V Patel

Abstract

- As the technology has been developing since many years and it remains welfare upto better level in people's life. Many kind of liquid faucets and dispensers for instance, sensor based water taps developed and gradually with time few of small technological changes like design, efficiency, etc have been applied onto it.
- However, until now it can be seen that such dispensers and faucets stay beneficial upto some level. They can not resolve the water scarcity problems completely. On the other hand, my designed dispenser can play the peculiar role by providing a good amount of technological change to ramp-up the improvements in this sector.
- ***Key Words:*** ***Water Industry, Fluid Mechanics, Arduino UNO Microcontroller, Servo-motor, Infrared Sensor, Keypad.***

INDEX

- 1.) Introduction**
- 2.) Problem Description**
- 3.) Advantages of Proposed Design**
- 4.) Prototype Diagram**
- 5.) Working Function**
 - 5.1) Connection to Normal Dispenser Assembly
- 6.) Prototype Images**
- 7.) Components used in Prototype**
- 8.) Arduino (C/ C++) Code**
- 9.) Operation Instructions**
- 10.) Conclusion**

1. Introduction

- Since few decades, world has been facing the crisis for water. This proposed dispenser can contribute by its vital performance. By implementation of this device one can save the water on his/ her daily basis needs. It has special ability to dispense the water or any liquid as per your desirations and necessities. The water; people use to drink, to wash clothes, to wash utensils, to fill up the containers or tank on daily basis in the households can be usually found out its wastages. Not only this device can be used in the realm of domestics but can be proved helping factor at industrial level also.
- Here, the report presents device's working mechanism, its component details, bit amount of fluid mechanics and C/ C++ codes which were used for the microcontroller to make device smart enough from other dispenser and faucets which are already available in market.

2. Problem Description

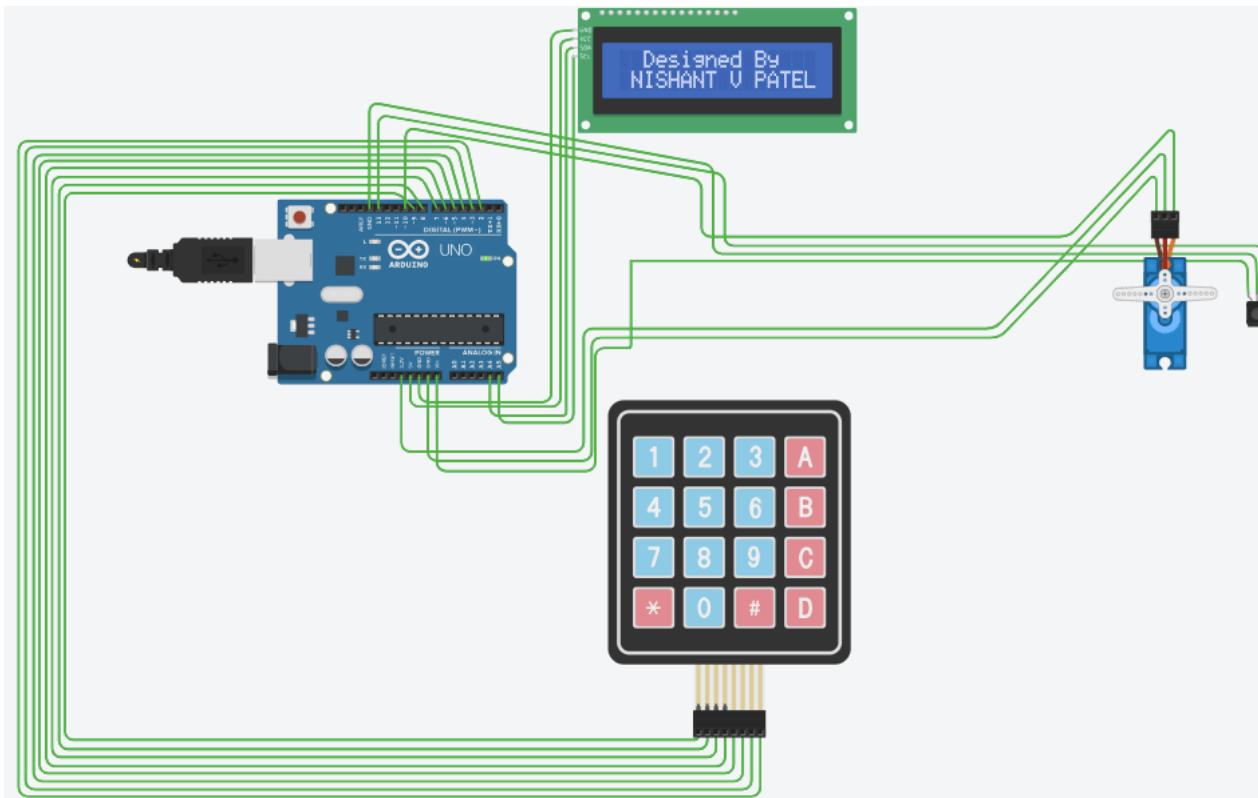
- Nowadays, conventional or traditional water taps are most commonly used at households, industry nearly everywhere in the world. Unfortunately, the sensored dispenser has its own limitations. When a person lay off his/ her hands in front of water tap then sensor detects the object and allows tap to flow the water through it. The second type of water tap is sensor based which is also immensely used specifically in coporate offices, headquarters.
- Thus, these 2 types can be seen mostly anywhere. But, this two kind of taps have its own limitations.
- The conventional water tap can not be turned off automatically once the targeted/ desired amount of water gets filled-up into water container. (the container for which one had turned on the tap)
- In additon, sensor based tap can not discharge the water/ liquid upto certain discharge rate or the person who wants tap to discharge the water/ liquid at his/ her desiration or need. As result, more water get discharged and ultimately go to wastage.
- Therefore, in order to eliminate such disabilities my Smart Liquid Dispenser enables the users to get the liquid as per his comfort and accordance.

3. Advantages of Proposed Design

- The proposed design allows the user to offer the amount of water or any other liquid as per his/ her neccessity.
- It can get automatically turned off itself when the targeted amount of water gets filled up in container as Smart Liquid Dispenser also offers the Auto-stop feature.
- Moreover, it has configurations of the conventional and sensor based mode also as used in today's date.
- This device occupied not so much of space.

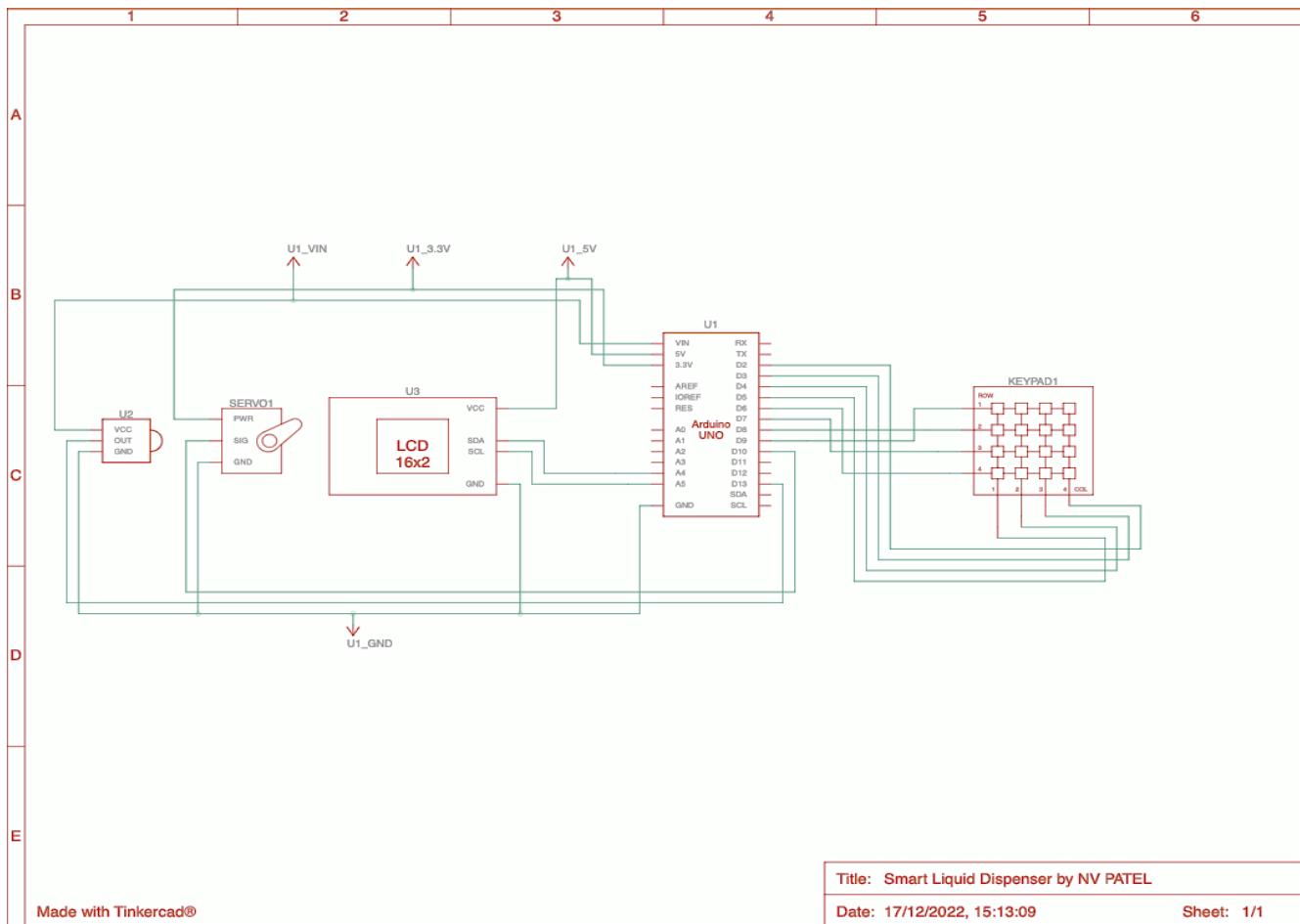
4. Prototype Diagrams

4.1 Component Diagram:



- This photo was taken from Tinker Card®.
- The Simulation with code have been done on Autodesk Tinker Card® online software.
- The photo diagram contains the Matrix Keypad, Arduino UNO, Micro Servo-motor, Infrared Sensor, and LCD Display.

4.2 Circuit/ Wire Diagram:



- This photo was also taken from TinkerCrad®. This diagram depicts the wiring connections among these components.
- Denotations used:
 - PWR= power pin
 - SIG/ OUT= signal pin
 - GND= ground pin
 - VCC= voltage pin
 - A0-A5= analog pins
 - D2-D13= digital pins
 - ROW 1-4 & COL 1-4= keypad pins

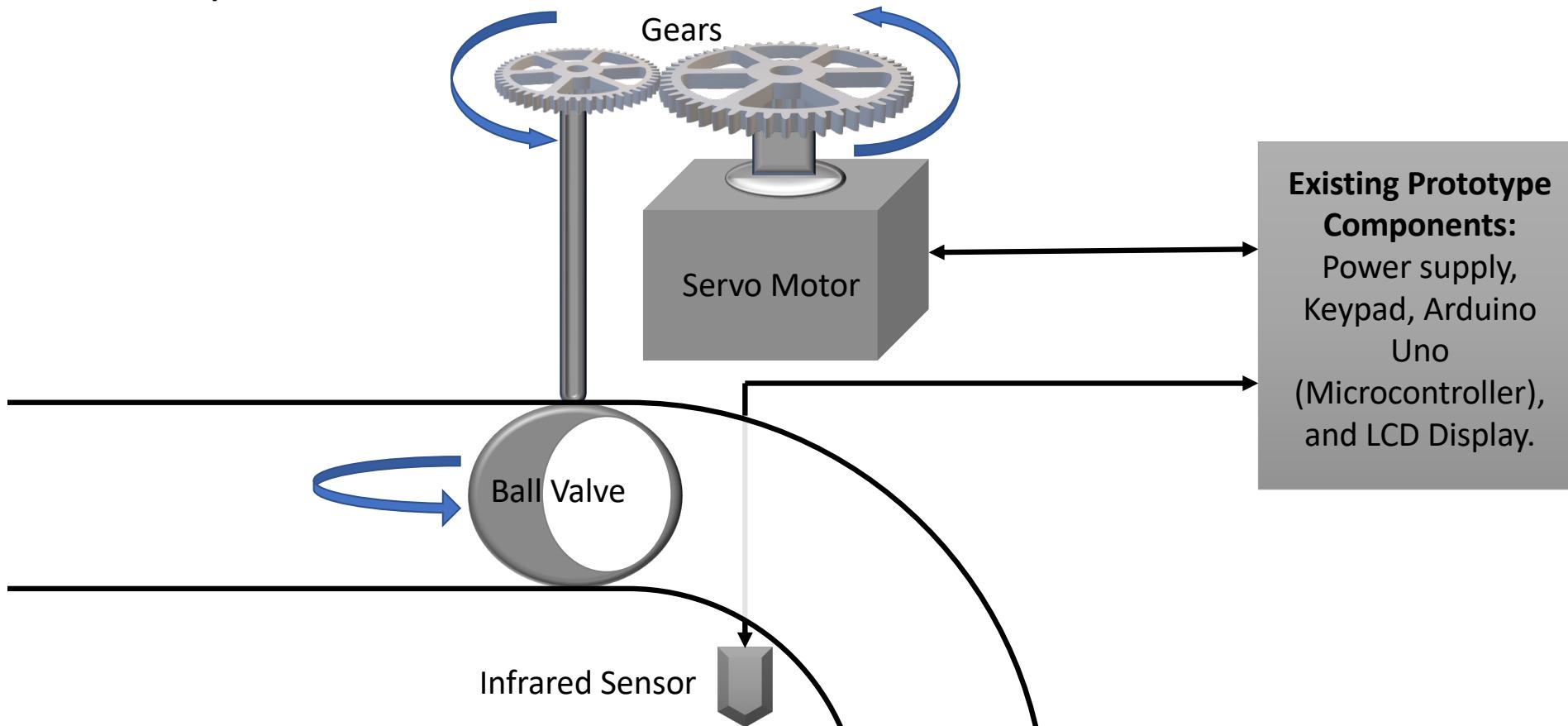
5. Working Function

- The working function is very simple.
- Keypad and IR Sensor are input device to microcontroller whereas LCD display and Servo-motor are output device for this particular prototype system.
- The Microcontroller acts as a median between input & output devices. This microcontroller Arduino takes in the different particular set of commands from keypad (which is ultimately given by user) and from IR sensor (when the user hand is being detected by sensor).
- Then According to the code written for this device, Arduino acts upon.
- Arduino then sends the signals to Servo-motor and LCD display.
- According to the signals received, these output devices will act and outputs the result which user had exactly expected from the device.

- The servo motor connected to ball valve and as the servo rotate the ball valve rotate itself accordingly then allows/ restricts the liquid to flow/ not to flow through itself during opening and closing phases.
- The servo motor is connected to ball valve in 2 efficient ways.
- First way is gear connection and second one is only-shaft/ direct connection.
- The next 2 pictorial representations will clear more of how actually this device works.

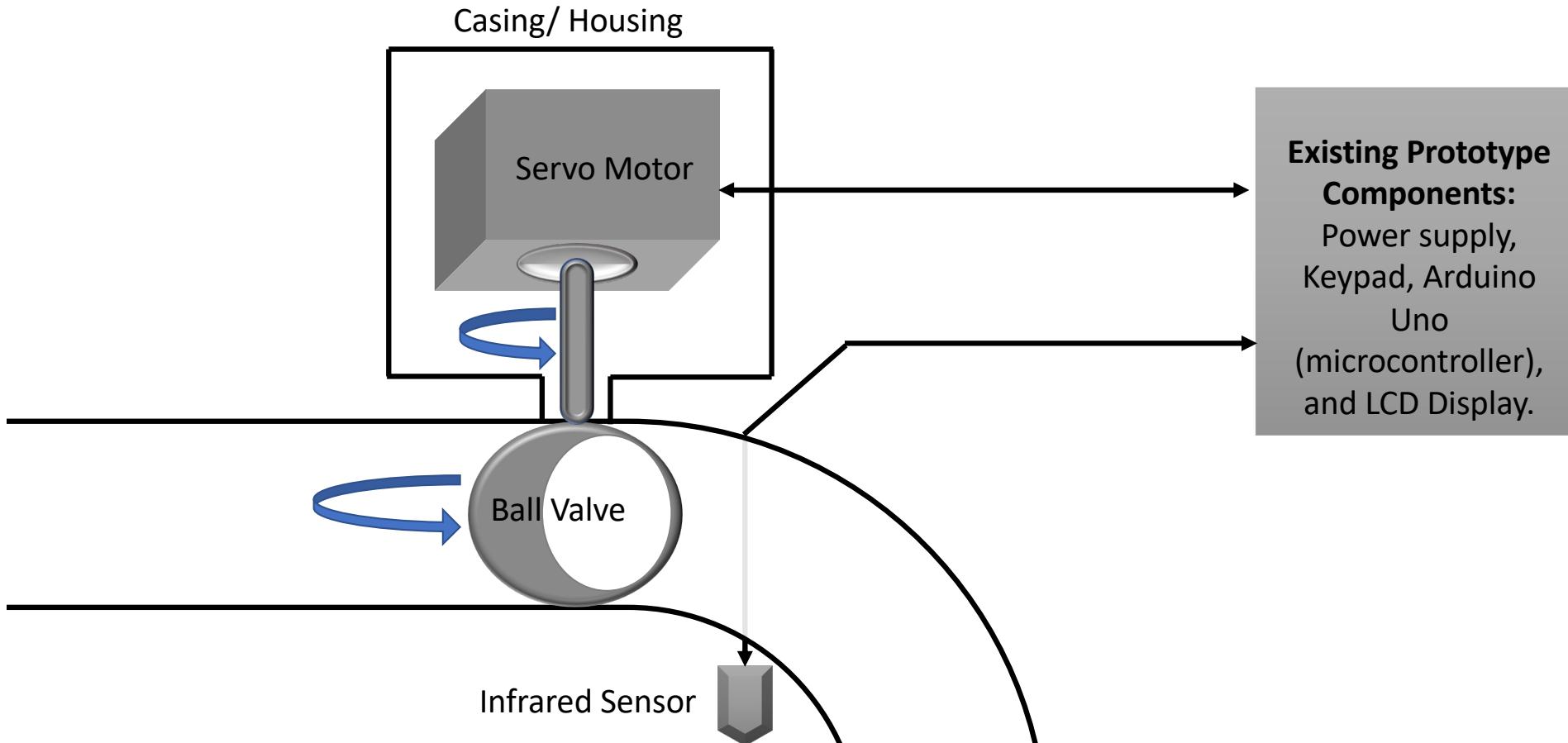
5.1 Connection of Proposed Assembly to Normal Dispenser

5.1.1) Gear Connection:

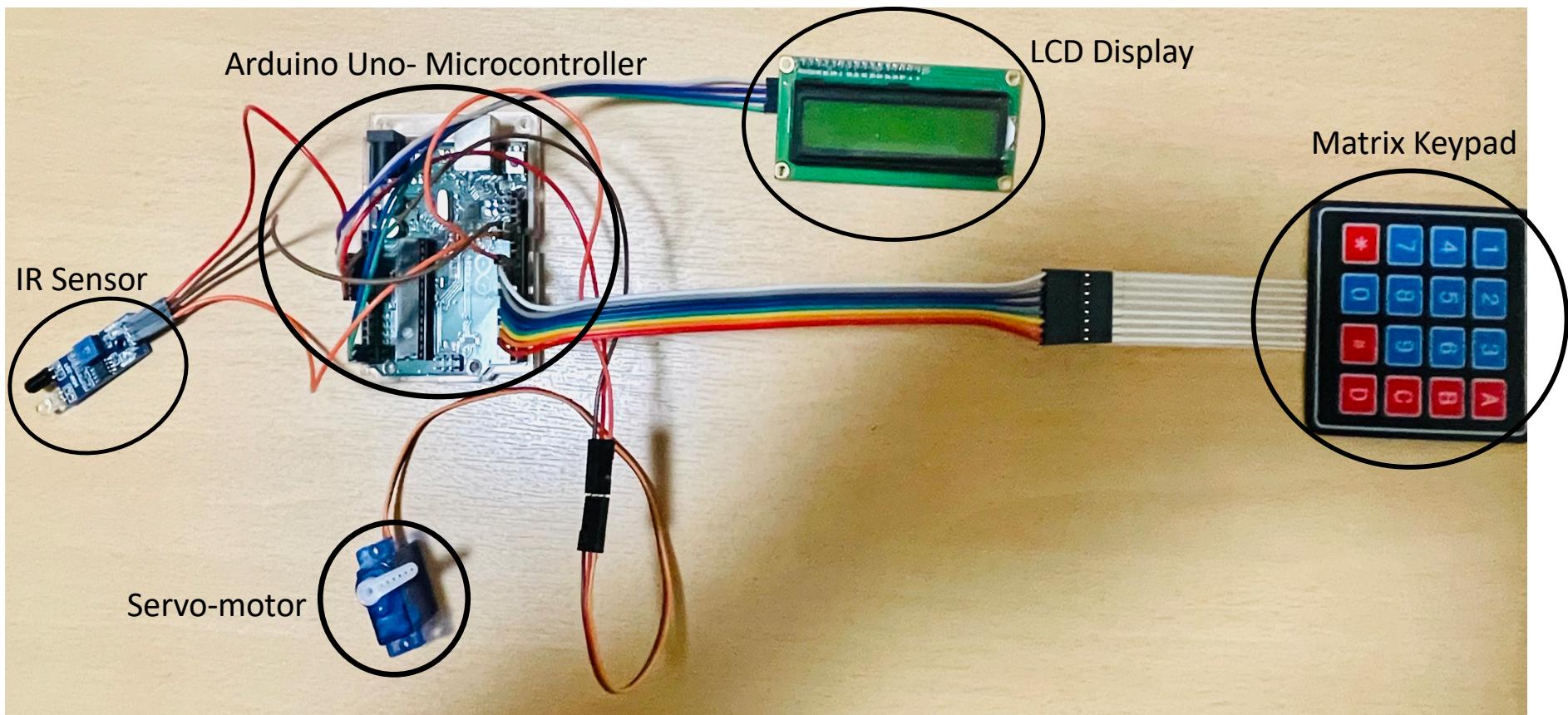


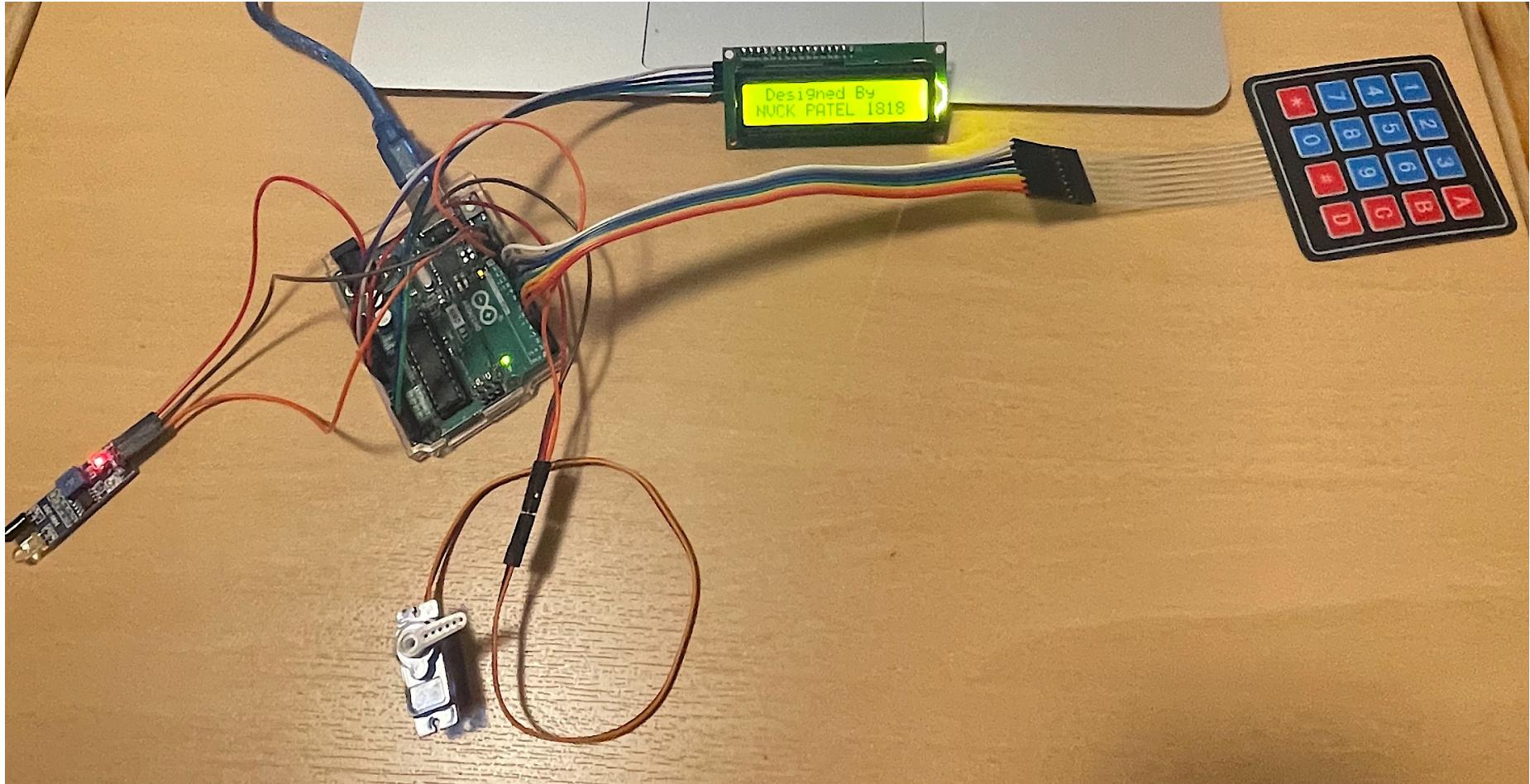
- In the gear connection, although the servo motor provides good enough torque for the ball valve to rotate. To stay at safe side increasing the torque output, the gear connection is best option to go with. In order to have higher torque, the driven gear should be of smaller diameter than driving gear.
- The servo motor degree rotation of 180^0 means the ball valve rotate to 90^0 as the gear ratio is 0.5. Therefore, the rotational frequency will be reduced by half. On the contrary, the torque will become doubled.

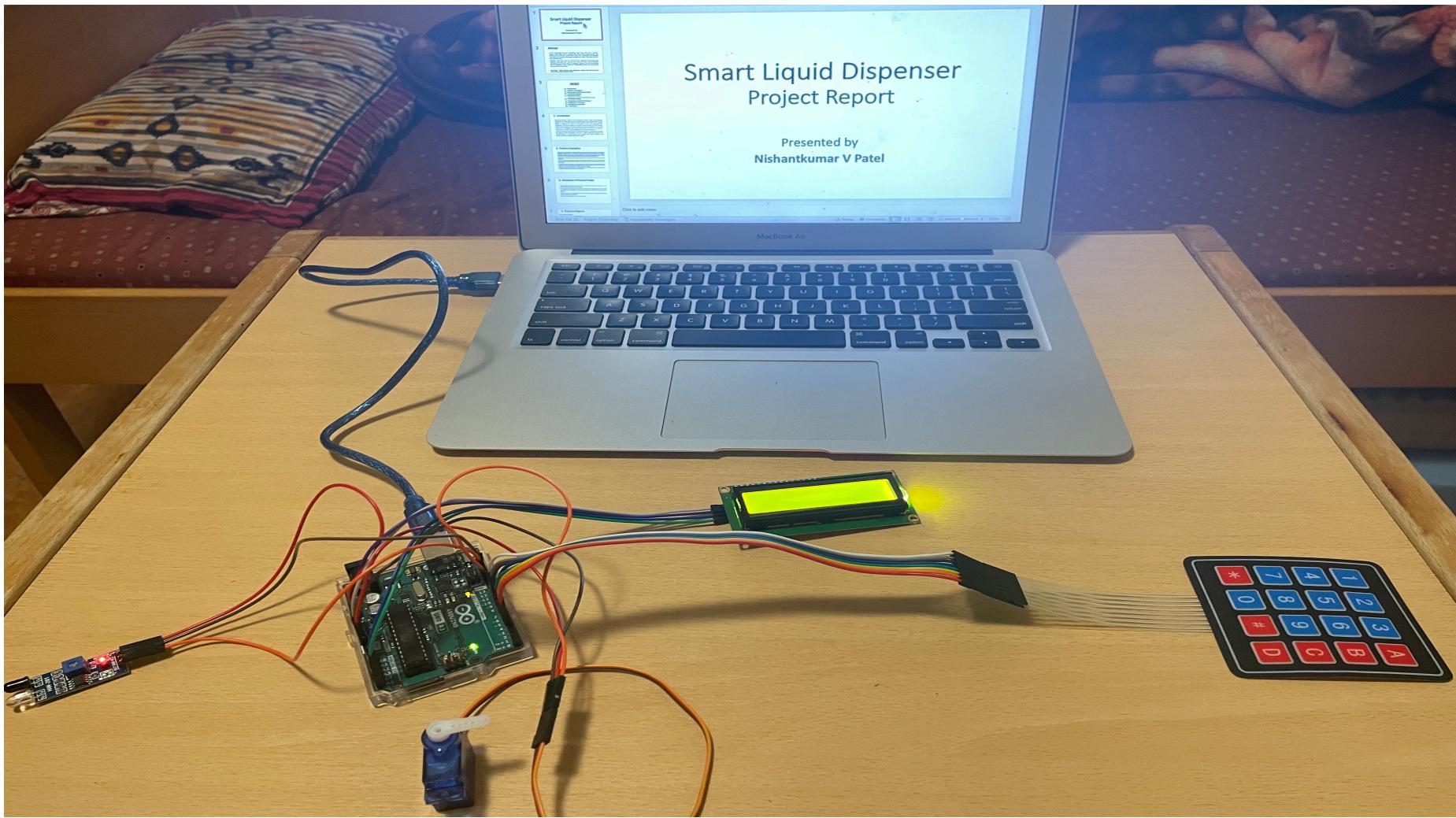
5.1.2) Only-shaft/ Direct Connection:



6. Prototype Images







7. Components for Final Design

- Note: The components which actually will be used for the final assembly are listed and described. Here, no components are mentioned which were used for the existing assembly.
- **Servo Motor:** Standard Servo motor will be used to provide the torque required for the ball valve to open and close.
- **Gears:** Two gears (driving & driven) of different diameters are used.
- **IR Sensor:** Infrared sensor module is used to detect the user hands for closing and opening the valve.
- **Jumper Wires:** These wires are used to connect all the components with each other for sending, receiving the signals and providing power to motor, lcd and sensor.

- **Microcontroller:** Arduino Uno R3 or Arduino Nano can be used as a microcontroller.
- **Keypad:** 4×3 matrix keypad is utilized to send the inputs to the microcontroller.
- **LCD Display:** 16×2 LCD display with I2C module is used to display the information and updatations.

8. Arduino (C/ C++) Code

- The Arduino code has been created where basically the C/ C++ languages are used.
- Different IoT Liabraries are included in the code for dealing with its functions.
- Plenty of If_else if_else control or conditional statements, for loops for servo motor degree rotations, keypad functions, and LCD display's functions are used.
- The comments inside this program has been written out, which persuade and justify the different kind of logics/ methods used and applied to this code.

Arduino IDE C:C++ Code.ino

/Users/jamesbond/Desktop/Arduino IDE C:C++

Code/Arduino IDE C:C++ Code.ino

```
2 |     The Code and Logic used behind this project is independently designed by
3 |     |     |     | ***** NISHANTKUMAR V PATEL *****
4 | */
5 |
6 #include <Servo.h> // Different Libraries Included.
7 #include <Keypad.h>
8 #include <LiquidCrystal_I2C.h>
9 #include <Wire.h>
10
11 int current; // Variable Declarations.
12 int previous;
13 String inputString;
14 long inputInt;
15 String binput;
16 const float flowrate = 25;
17
18 void ServoMotor_Rotation_FUNCTION(); // Function Prototypes Declarations.
19 void Discrete_ServoMotor_Rotation_FUNCTION();
20 void IR_Sensor_FUNCTION();
21 void Device_Information_FUNCTION();
22 void User_Manual_FUNCTION();
23
24 const int ROW_NUM = 4;
25 const int COLUMN_NUM = 4;
26 char keys[ROW_NUM][COLUMN_NUM] = // different key declarations
27 {
28     { '1', '2', '3', 'A' },
29     { '4', '5', '6', 'B' }, // it is 4x3 matrix keypad Array.
30     { '7', '8', '9', 'C' },
31     { '*', '0', '#', 'D' }
32 };
33 byte row_pins[ROW_NUM] = { 9, 8, 7, 6 }; // to setup the pin numbers.
34 byte col_pins[COLUMN_NUM] = { 5, 4, 3, 2 };
35
36 Keypad keypad = Keypad(makeKeymap(keys), row_pins, col_pins, ROW_NUM, COLUMN_NUM); // create the Keypad Object.
37 LiquidCrystal_I2C lcd(0x27, 16, 2); // create the LCD Display Object with I2C chip address with 16 characters and 2 rows or lines.
38 Servo s1;
```

Arduino IDE C:C++ Code.ino

...

```
39
40
41 void setup() {
42     lcd.init(); // to initialize the LCD.
43     lcd.backlight();
44     pinMode(13, INPUT); // input pin for reading IR Sensor signals.
45     s1.attach(10); // pin for sending the signals to servo-motor.
46     inputString.reserve(9);
47 }
48
49
50
51 void loop() // --Main Function--           // to run Infinitely....
52 {
53
54 {
55     IR_Sensor_Mode_FUNCTION();
56 }
57
58 char key = keypad.getKey(); // pressed key accepted as input for keypad.
59 if (key) // if any key is pressed then condition becomes true and enter inside the below conditional statements.
60 {
61     if (key >= '0' && key <= '9') {
62         inputString += key; // storing the multiple key input in a single string variable.
63         lcd.setCursor(1, 0);
64         lcd.print("Your Input is");
65         lcd.setCursor(1, 1);
66         lcd.print(inputString);
67         lcd.setCursor(12, 1);
68         lcd.print("(ml)");
69         delay(100);
70         inputInt = inputString.toInt(); // converting from String to Integer data type.
71     }
72
73     else if (key == '#') // to execute the multiple key input.
74     {
75         ServoMotor_Rotation_FUNCTION();
76     }
77
78     else if (key == '*') // to clear the multiple key input.
79     {
80         inputString = "";
81         inputInt = 0;
82         binput = "";
83     }
}
```

Arduino IDE C:C++ Code.ino

...

```
84
85     else if (key == 'A') // if A is pressed then servo-motor starts to rotate to Open the Dispenser.
86     {
87         lcd.clear();
88         lcd.setCursor(5, 0);
89         lcd.print("Opened");
90         for (int i = 0; i <= 102; i += 1) // to initialize the loop with conditions.
91         {
92             s1.write(i); // to rotate the servo motor.
93             delay(10);
94         }
95     }
96
97     else if (key == 'B') // if B is pressed then servo-motor starts to rotate discretely to Open the Dispenser partially.
98     {
99         binput += key; // store the multiple keys in a single variable- binput.
100        Discrete_ServoMotor_Rotation_FUNCTION();
101    }
102
103    else if (key == 'C') // if C is pressed then servo-motor starts to rotate back to Close the Dispenser.
104    {
105        lcd.clear(); // to clear the LCD display
106        lcd.setCursor(5, 0);
107        lcd.print("Closed");
108        for (int i = 102; i >= 0; i -= 1) {
109            s1.write(i);
110            delay(10);
111        }
112        delay(1800);
113        lcd.clear();
114    }
115
116    else if (key == 'D') {
117        Device_Information_FUNCTION();
118        delay(1500); // delay(milliseconds)
119        User_Manual_FUNCTION(); // Every Function Prototypes are called back in main infinitely running void loop().
120    }
121 }
122 }
123
124
125
126 /* Function Prototyping of Different Functions for Modes.
127 or Function Prototypes; which is called back in main code structure or void loop() */
128
```

Arduino IDE C/C++ Code.ino

...

```
129 void ServoMotor_Rotation_FUNCTION()
130 {
131 {
132     flowrate; // Flowrate of Actual Liquid Dispenser. Flowrate is in (ml/second) unit.
133     float time = inputInt / flowrate; //Fluid mechanics calculations to get the time in seconds.
134     float seconds = time * 1000;
135
136     if (inputInt > 0 && inputInt != 0) {
137         lcd.clear();
138         lcd.setCursor(3, 0);
139         lcd.print("Started!");
140         for (int i = 0; i <= 102; i += 1) // for loop with initial and ending degree angles for servo-motor.
141         {
142             s1.write(i);
143             delay(10);
144         }
145         lcd.clear();
146         lcd.setCursor(3, 0);
147         lcd.print("Running...");
148
149     {
150         delay(seconds - 400); //give the delay of amount of seconds which is derived from the fluid mechanics calculations.
151     }
152
153     for (int i = 102; i >= 0; i -= 1) //it tells to servo-motor to rotate up to how much of angles and with speed increments
154     {
155         s1.write(i);
156         delay(10);
157     }
158     lcd.clear();
159     lcd.setCursor(3, 0);
160     lcd.print("Completed!");
161     delay(2000);
162     lcd.clear();
163 }
164 }
165
166
167 void Discrete_ServoMotor_Rotation_FUNCTION()
168 {
169 {
170     lcd.clear();
171     lcd.setCursor(0, 0); //setting up the location of the cursor in lcd display.
172     lcd.print("Partial Openings");
```

Arduino IDE C:C++ Code.ino

...

```
173 if (binput == "B") {
174     lcd.setCursor(5, 1);
175     lcd.print("@ 1X");
176     for (int i = 0; i <= 20; i += 1) // if B key is pressed one time, this loop will execute.
177     {
178         s1.write(i);
179         delay(10);
180     }
181 } else if (binput == "BB") {
182     lcd.setCursor(5, 1);
183     lcd.print("@ 2X");
184     for (int i = 20; i <= 40; i += 1) // if B key is pressed consecutively two times, this loop will run.
185     {
186         s1.write(i);
187         delay(10);
188     }
189 } else if (binput == "BBB") {
190     lcd.setCursor(5, 1);
191     lcd.print("@ 3X");
192     for (int i = 40; i <= 60; i += 1) {
193         s1.write(i);
194         delay(10);
195     }
196 } else if (binput == "BBBB") {
197     lcd.setCursor(5, 1);
198     lcd.print("@ 4X");
199     for (int i = 60; i <= 80; i += 1) {
200         s1.write(i);
201         delay(10);
202     }
203 } else if (binput == "BBBBB") // if B key is pressed consecutively five times, below loop will run.
204 {
205     lcd.setCursor(5, 1);
206     lcd.print("@ 5X");
207     for (int i = 80; i <= 102; i += 1) {
208         s1.write(i);
209         delay(10);
210     }
211 }
212 }

213
214
215 void IR_Sensor_Mode_FUNCTION() {
216
217     current = digitalRead(13); // 0 means, if object is detected. 1 means, if object is not detected.
```

Arduino IDE C:C++ Code.ino

...

```
218 if (current == 0)          // if something is detected (human hands for washing) then servo-motor rotate to Open the Dispenser.
219 {
220     if (previous == 1) {
221         for (int i = 0; i <= 102; i += 1) {
222             s1.write(i);
223             delay(7);
224         }
225         delay(1000);
226     }
227 }
228
229 else if (current == 1) // if something is not detected then servo motor rotate back (reverse) to close the dispenser.
230 {
231     if (previous == 0) {
232         for (int i = 102; i >= 0; i -= 1) // for (Initialization; Condition; Updation)
233         {
234             s1.write(i);
235             delay(7);
236         }
237     }
238 }
239 previous = current; // assign the value of current into previous variable for next iteration.
240 delay(50);
241 }
242
243
244 void Device_Information_FUNCTION() // this function is made up of entirely upon LCD codes.
245 {
246     lcd.clear();
247     lcd.setCursor(2, 0);
248     lcd.print("Hello World");
249     delay(1500);
250     lcd.clear();
251
252     lcd.setCursor(4, 0);
253     lcd.print("I AM");
254     delay(1000);
255     lcd.clear();
256     lcd.setCursor(5, 0);
257     lcd.print("SMART");
258     lcd.setCursor(0, 1);
259     lcd.print("LIQUID DISPENSER");
260     delay(3500);
261     lcd.clear();
262 }
```

Arduino IDE C/C++ Code.ino

...

```
263     lcd.setCursor(1, 0);
264     lcd.print("Designed By");
265     delay(500);
266     lcd.setCursor(0, 1);
267     lcd.print("NVCK PATEL 1818");
268     delay(3500);
269     lcd.clear();
270
271     lcd.setCursor(2, 0);
272     lcd.print("Save Water!");
273     delay(2000);
274     lcd.clear();
275 }
276
277 void User_Manual_FUNCTION() // this function also is made up of entirely upon LCD codes.
278
279 ~ {
280     lcd.clear();
281     lcd.setCursor(3, 0);
282     lcd.print("Welcome to");
283     lcd.setCursor(2, 1);
284     lcd.print("User Manual");
285     delay(3000);
286     lcd.clear();
287
288     lcd.setCursor(4, 0);
289     lcd.print("Mode-1");
290     lcd.setCursor(5, 1);
291     lcd.print("Manual");
292     delay(3000);
293     lcd.clear();
294     lcd.setCursor(0, 0);
295     lcd.print("Press A to Open");
296     lcd.setCursor(0, 1);
297     lcd.print("Press C to Close");
298     delay(3500);
299     lcd.clear();
300
301     lcd.setCursor(4, 0);
302     lcd.print("Mode-2");
303     lcd.setCursor(0, 1);
304     lcd.print("Partial Openings");
305     delay(3500);
```

Arduino IDE C:C++ Code.ino

...

```
307     lcd.clear();
308     lcd.setCursor(0, 0);
309     lcd.print("Press *");
310     lcd.setCursor(0, 1);
311     lcd.print("to start Mode-2");
312     delay(3500);
313     lcd.clear();
314     lcd.setCursor(0, 0);
315     lcd.print("Press B to set");
316     lcd.setCursor(0, 1);
317     lcd.print("openings upto 5X");
318     delay(3500);
319     lcd.clear();
320
321     lcd.setCursor(4, 0);
322     lcd.print("Mode-3");
323     lcd.setCursor(3, 1);
324     lcd.print("Auto-Stop");
325     delay(3500);
326     lcd.clear();
327     lcd.setCursor(0, 0);
328     lcd.print("Press 0 to 9 keys");
329     lcd.setCursor(0, 1);
330     lcd.print("as Input in ml");
331     delay(3500);
332     lcd.clear();
333     lcd.setCursor(0, 0);
334     lcd.print("Press *");
335     lcd.setCursor(2, 1);
336     lcd.print("to clear Input");
337     delay(3500);
338     lcd.clear();
339     lcd.setCursor(0, 0);
340     lcd.print("Press #");
341     lcd.setCursor(2, 1);
342     lcd.print("to start Input");
343     delay(3500);
344     lcd.clear();
345
346     lcd.setCursor(4, 0);
347     lcd.print("Mode-4");
348     lcd.setCursor(0, 1);
349     lcd.print("Sensor/ Default");
350     delay(3500);
351     lcd.clear();
352 }
```

353

9. Operation Instructions

- The different keypad buttons stands for different mode
- For ‘Manual Mode’, to turn on and turn off the dispenser Press A and Press C respectively.
- Press * to start the ‘Partial Openings Mode’ then press B button to open the dispenser discretely up to 5 times. Means, when user press this specific button B one time valve opens at 18^0 , then again by pressing the same button valve further open 18^0 , means upto now dispenser opens at 36^0 in total. So, one can partially open in such manner upto 5 times. (0^0-18^0 , 18^0-36^0 , 36^0-54^0 , 54^0-72^0 , 72^0-90^0)
- Press D button for User Manual and Device Information.
- The ‘Sensor Mode’ is default therefore no keypad key is needed to be pressed for activating, as this mode is always ON and whenever the hands are detected it will start to implement.

- Press * to start the Auto-Stop Mode.
- Here one needs to press the numeric values. So, after pressing * press numeric keys 0 to 9 as the liquid quantity [only in Mili-Liter (ml)] user wants.
- Then simply press # to start device or to start the dispensing process.
- In between if user wants to change the quantity of liquid, then press * to clear/ erase the previous entered input value.

10. Conclusion

- In conclusion, it is clearly seen that the proposed assembly can help and establish the better framework for discharges the required and specific amount of liquid whatever user wants.
- As a consequence, it can be proved more efficient as compared to other dispenser.
- Auto-stop and Partial-openings, these two features bring the advantages in effectiveness of device's performance.
- Therefore, Smart Liquid Dispenser has more things to offer than traditional taps and faucets. Its like all in one agenda.