The Trendy Vendy - Final Report

By: Elliot Hong, Christopher Luo, Aidan Li, Colby Winters

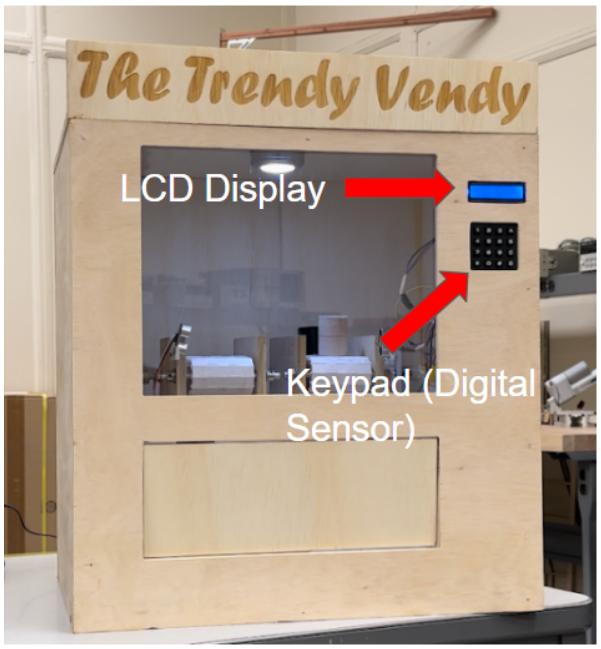
Opportunity:

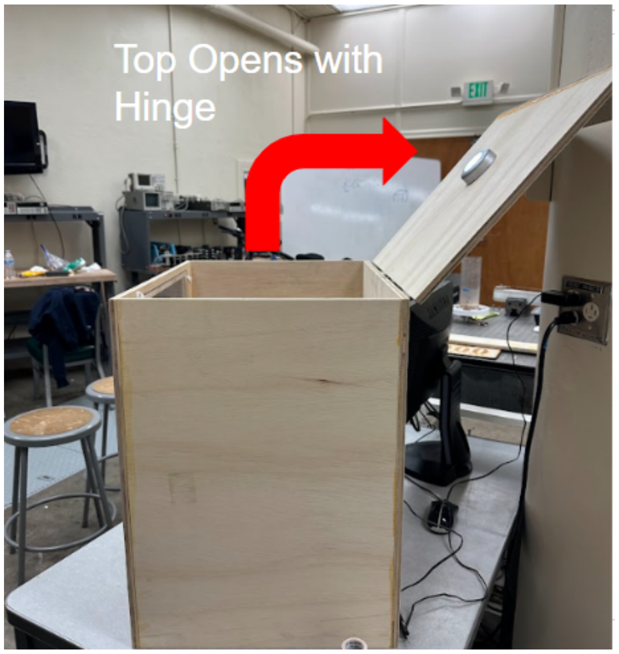
The opportunity our project sought to tackle was the improvement of vending machines to create a better overall experience for its users (owners and customers). It accomplished this by changing/adding a few parts compared to traditional vending machines. One improvement was changing the traditional coil in vending machines to a conveyor belt system allowing the owner to fit more items in the machine at once. Another improvement was adding an item damage mitigation cushion to minimize damage to items after they are dispensed. Finally, we added an item detection mechanism (ultrasonic sensor) to ensure that a user will always receive one item, instead of zero or multiple items that commonly occurs with traditional coils.

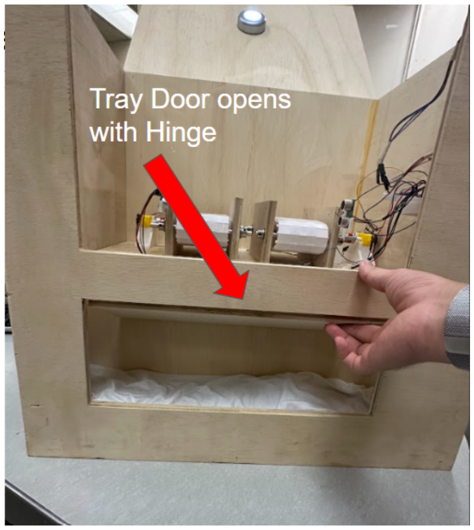
High Level Strategy:

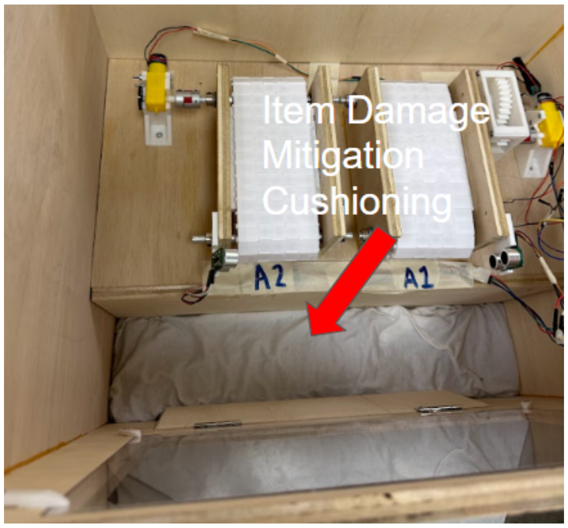
The customer will enter their desired item number on the keypad, which will turn the corresponding conveyor belt on and continue till an item falls and is detected by the item detection mechanism (ultrasonic sensor). The item then falls into the item damage mitigation tray, and the customer then grabs their item. We initially desired to have an automatic voice detection system to which the user could speak to the machine, and a system to track selling data, but these were not implemented due to time constraints. The conveyor belts were desired to move items with a mass of two soda cans (355 g), and based on our calculations we believed we could achieve this, but were initially unsuccessful. The conveyor belt without the gear box was only able to move items with less mass ~150 g. After adding a gearbox to the second belt, we were able to achieve the necessary torque on the belt to move the heavier items. In terms of overall functionality, the sensor integrated with the belt system achieved the desired ability to stop the belt after only dispensing one of any type of item consistently.

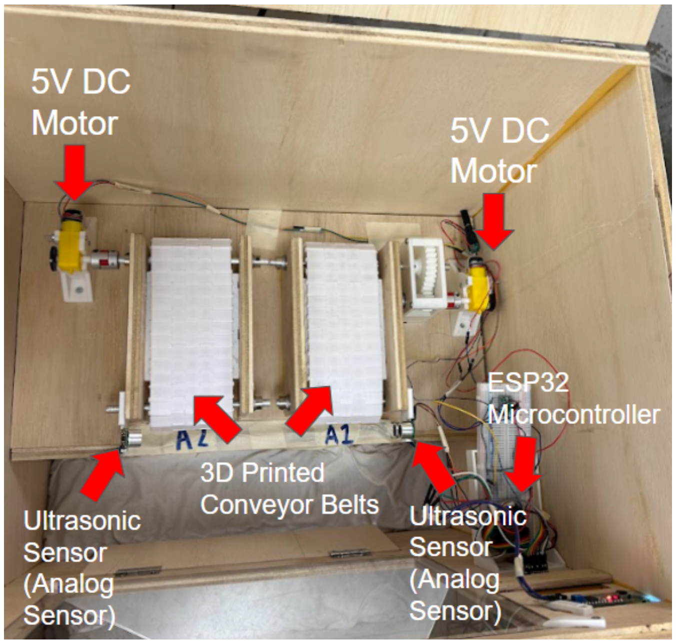
Photo of Device with Labels:

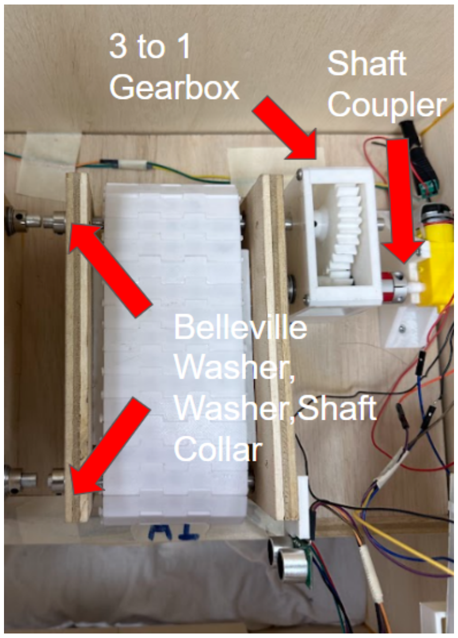












Critical Design Decisions:

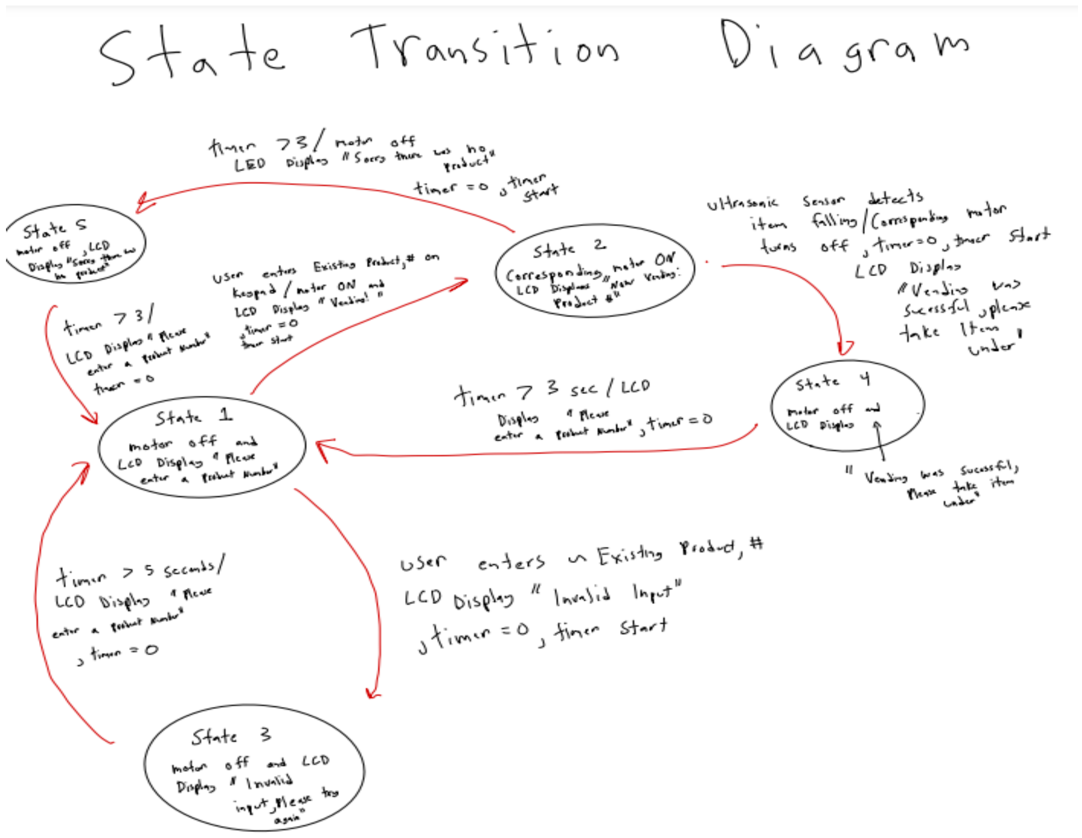
Our function critical design decision involved selecting the correct motor, belt size, and gear ratio in order to achieve the desired dispensing of items up to the weight of two soda cans. We decided to begin with the motor, and selected the one that came with our kits, as it would save money and we believed it could provide the right amount or torque to spin the conveyor belt. We decided that we wanted to fit about two soda cans, so we sized the length of the belt such that it would fit two cans, with a little extra room. The following are the calculations performed to ensure it would be strong enough.
Max Torque of Motor = 0.60 Nm (From Motor Spec Sheet)
We do not want to exceed 60%, so max operating torque = 0.60 * .6 = .36 Nm
Mass of each soda can = 0.355 kg
2 Soda cans = 0.355 * 2 = 0.710 kg

Weight of 2 Cans = 9.8 m/s^2 * 0.710 kg = 6.96 N
Formula for Required Torque, T = (F* r)/(n)
Efficiency (n) assumed to be 0.4
Radius of gear = 17 mm = .017 m
T = (6.96 * .017)/.4 = .30 Nm, which meets 0.36 Nm Requirement.

In reality the motor was unable to move the belt with two soda cans, so the efficiency must be lower than expected. After adding a 3 to 1 gearbox for one of the belts, max torque output allowed is .36 * 3 = 1.08 Nm, which is sufficient for an efficiency as low as 0.11.

For the bearings, the loads are equal to the required preload to tension the conveyor belt.
Output Torque Assumed = 0.30 Nm
Gear Radius = 17 mm = .017 m
Bearing Force = T/R, = .30/.017 = 17.6 N
Static Load Rating of Bearing = 474 N, meaning we have a SF (Safety Factor) of 27

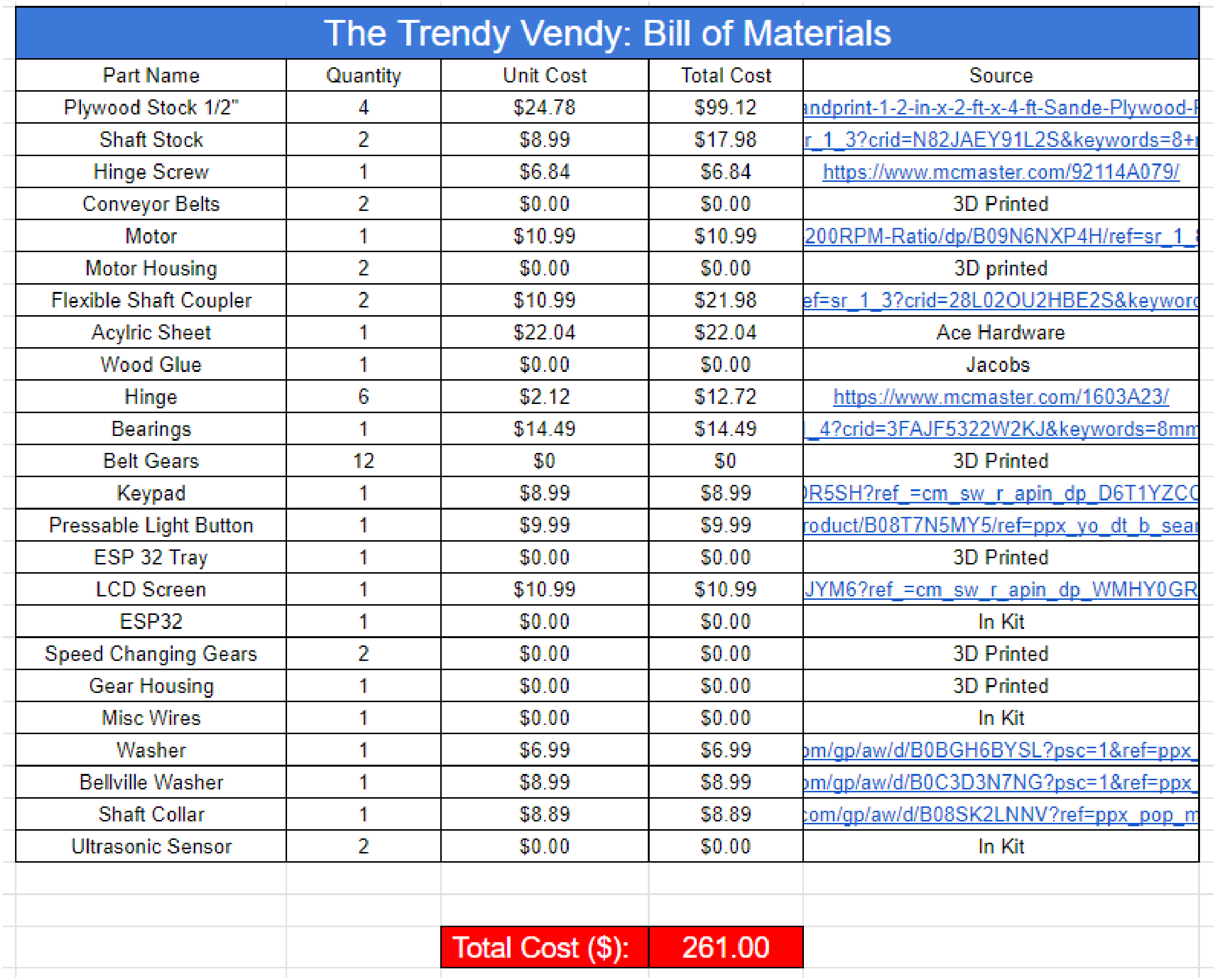
State Transition Diagram:


State Transition Diagram
timer > 3 / motor off LED Displays "Sorry there was no product" timer = 0, timer start
State 5 motor off, LCD Displays "Sorry there was no product"
User enters Existing Product # on keypad / motor ON and LCD Displays "Vending!" , timer = 0 timer start
State 2 Corresponding motor ON, LCD Displays "Now Vending: Product #"
ultrasonic sensor detects item falling / Corresponding motor turns off, timer = 0, timer start LCD Displays "Vending was successful, please take item under"
timer > 3 / LCD Displays "Please enter a Product Number" timer = 0
State 1 motor off and LCD Display "Please enter a Product Number"
timer > 3 sec / LCD Displays "Please enter a Product Number", timer = 0
State 4 motor off and LCD Displays
"Vending was successful, Please take item under"
timer > 5 seconds / LCD Displays "Please enter a Product Number", timer = 0
User enters un Existing Product, # LCD Display "Invalid Input" , timer = 0, timer start
State 3 motor off and LCD Displays "Invalid input, Please try again"

Reflection: Overall, the biggest strategy that led to our success was ensuring that the topic/project chosen was one that everyone in the group was passionate about. Since everyone was passionate about our project this led to everyone wanting to work on it and willing to go above and beyond to make our vision come true. One thing our group could have done differently was sticking to the time we set in the beginning to meet every week to ensure that everyone is on the same page and the progress that needs to be done every week gets finished.

Appendix

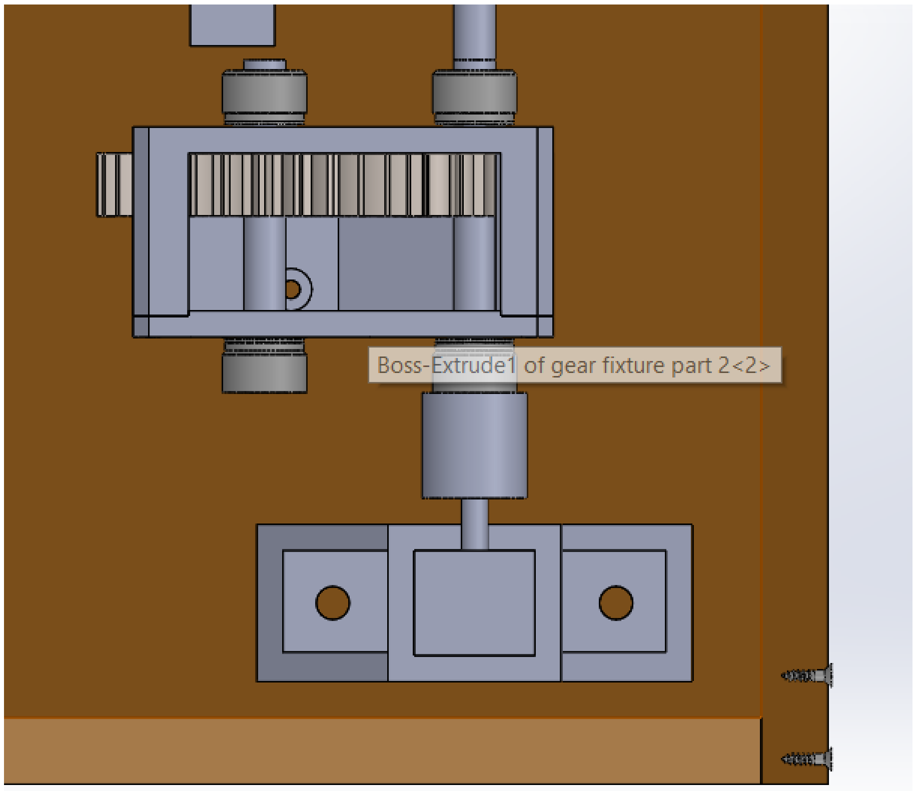
Bill of Materials:

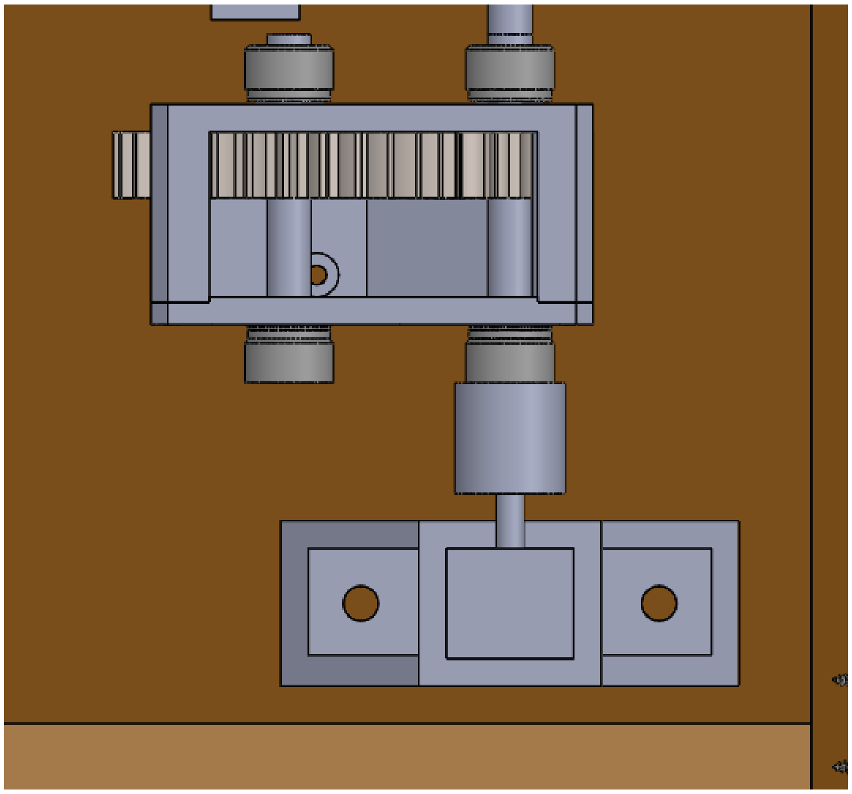


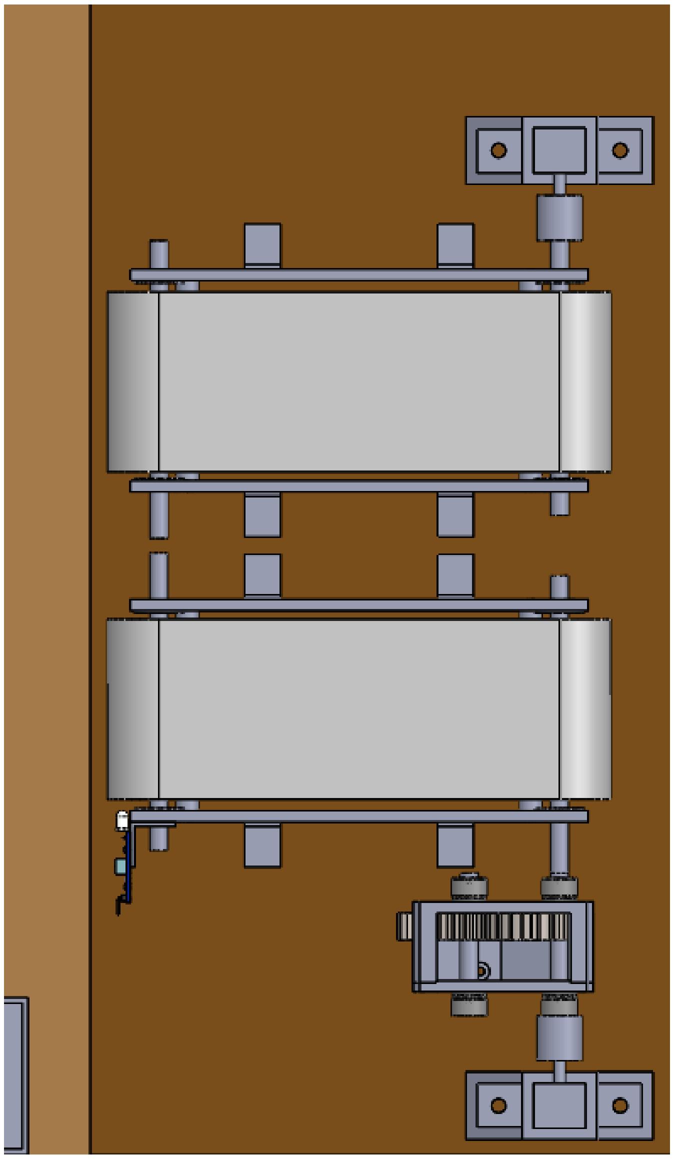
The Trendy Vendy: Bill of Materials				
Part Name	Quantity	Unit Cost	Total Cost	Source
Plywood Stock 1/2"	4	$24.78	$99.12	andprint-1-2-in-x-2-ft-x-4-ft-Sande-Plywood-F
Shaft Stock	2	$8.99	$17.98	r_1_3?crid=N82JAEY91L2S&keywords=8+
Hinge Screw	1	$6.84	$6.84	https://www.mcmaster.com/92114A079/
Conveyor Belts	2	$0.00	$0.00	3D Printed
Motor	1	$10.99	$10.99	200RPM-Ratio/dp/B09N6NXP4H/ref=sr_1_
Motor Housing	2	$0.00	$0.00	3D printed
Flexible Shaft Coupler	2	$10.99	$21.98	ef=sr_1_3?crid=28L02OU2HBE2S&keyword
Acylric Sheet	1	$22.04	$22.04	Ace Hardware
Wood Glue	1	$0.00	$0.00	Jacobs
Hinge	6	$2.12	$12.72	https://www.mcmaster.com/1603A23/
Bearings	1	$14.49	$14.49	_4?crid=3FAJF5322W2KJ&keywords=8mm
Belt Gears	12	$0	$0	3D Printed
Keypad	1	$8.99	$8.99	R5SH?ref_=cm_sw_r_apin_dp_D6T1YZC
Pressable Light Button	1	$9.99	$9.99	roduct/B08T7N5MY5/ref=ppx_yo_dt_b_sear
ESP 32 Tray	1	$0.00	$0.00	3D Printed
LCD Screen	1	$10.99	$10.99	JYM6?ref_=cm_sw_r_apin_dp_WMHY0GR
ESP32	1	$0.00	$0.00	In Kit
Speed Changing Gears	2	$0.00	$0.00	3D Printed
Gear Housing	1	$0.00	$0.00	3D Printed
Misc Wires	1	$0.00	$0.00	In Kit
Washer	1	$6.99	$6.99	om/gp/aw/d/B0BGH6BYSL?psc=1&ref=ppx_
Bellville Washer	1	$8.99	$8.99	om/gp/aw/d/B0C3D3N7NG?psc=1&ref=ppx_
Shaft Collar	1	$8.89	$8.89	com/gp/aw/d/B08SK2LNNV?ref=ppx_pop_m
Ultrasonic Sensor	2	$0.00	$0.00	In Kit
		Total Cost ($):	261.00	

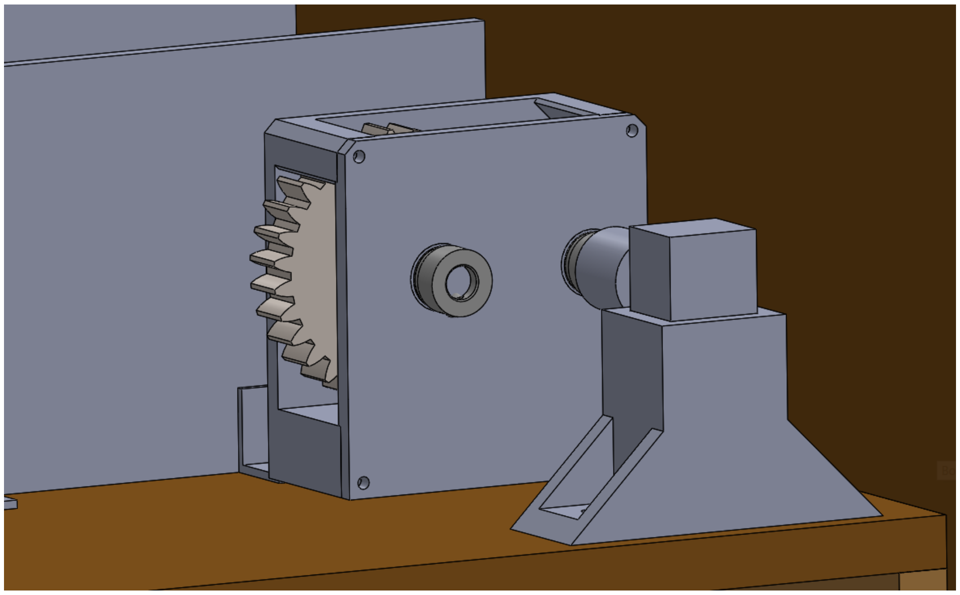
https://docs.google.com/spreadsheets/d/1o25A7YxaQ-j9CvxKn_9-_HlHdxigR9muAbKDfllPjHM/edit?usp=sharing

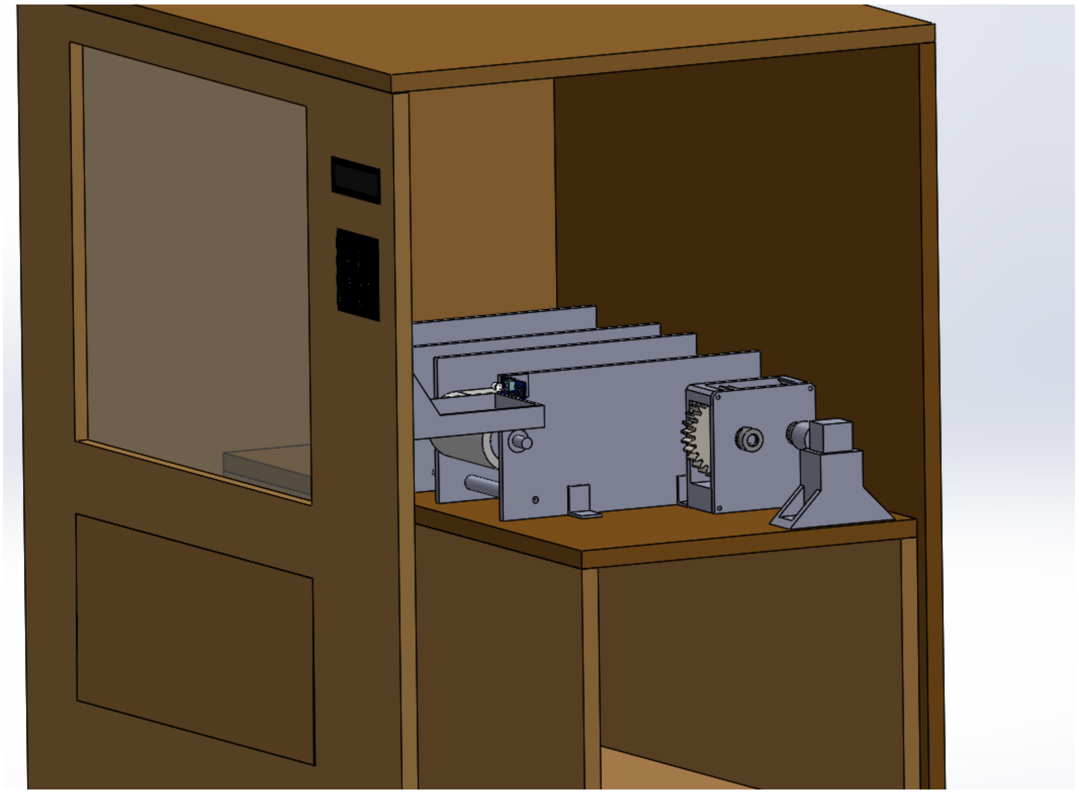
Images of CAD:



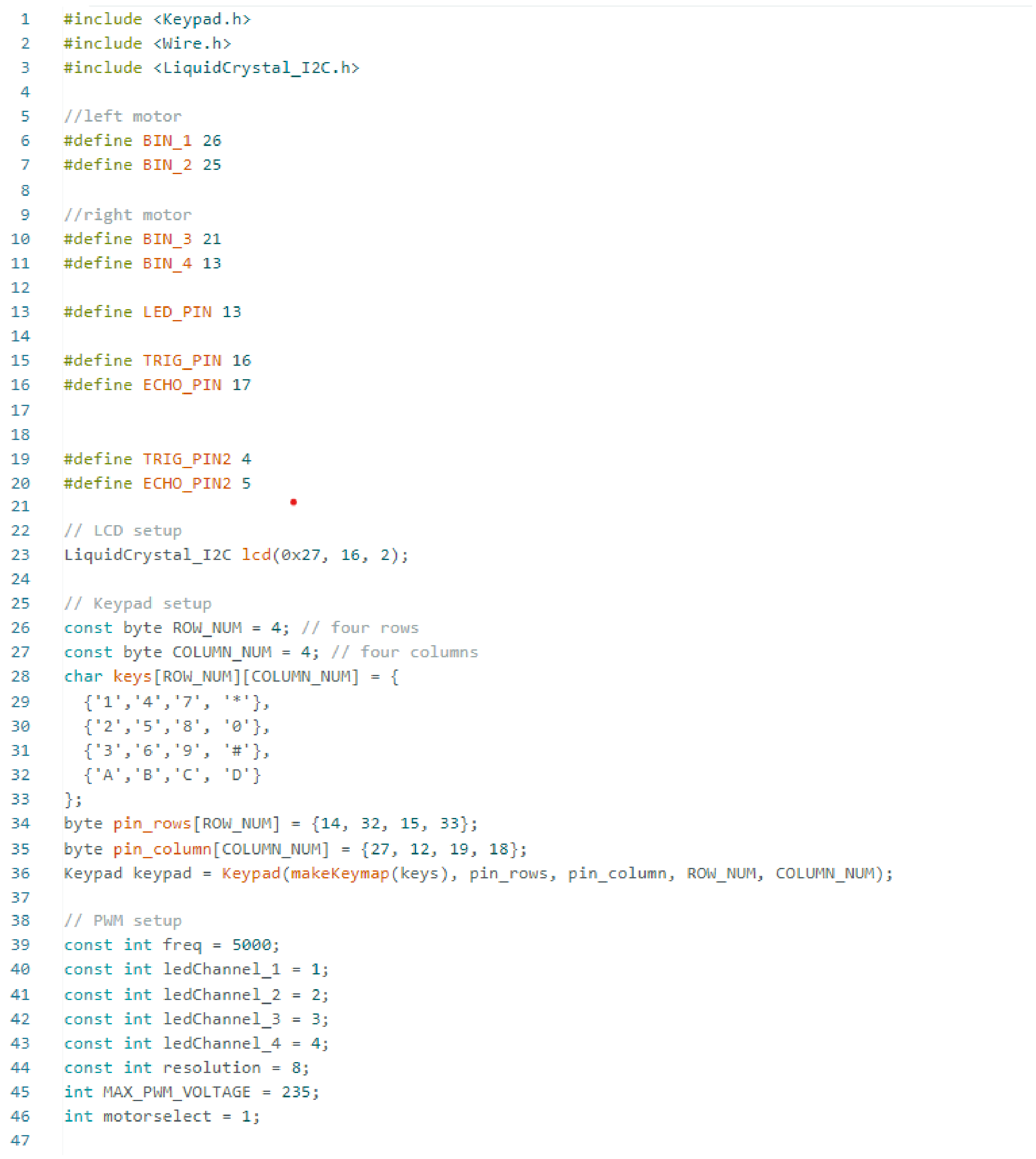






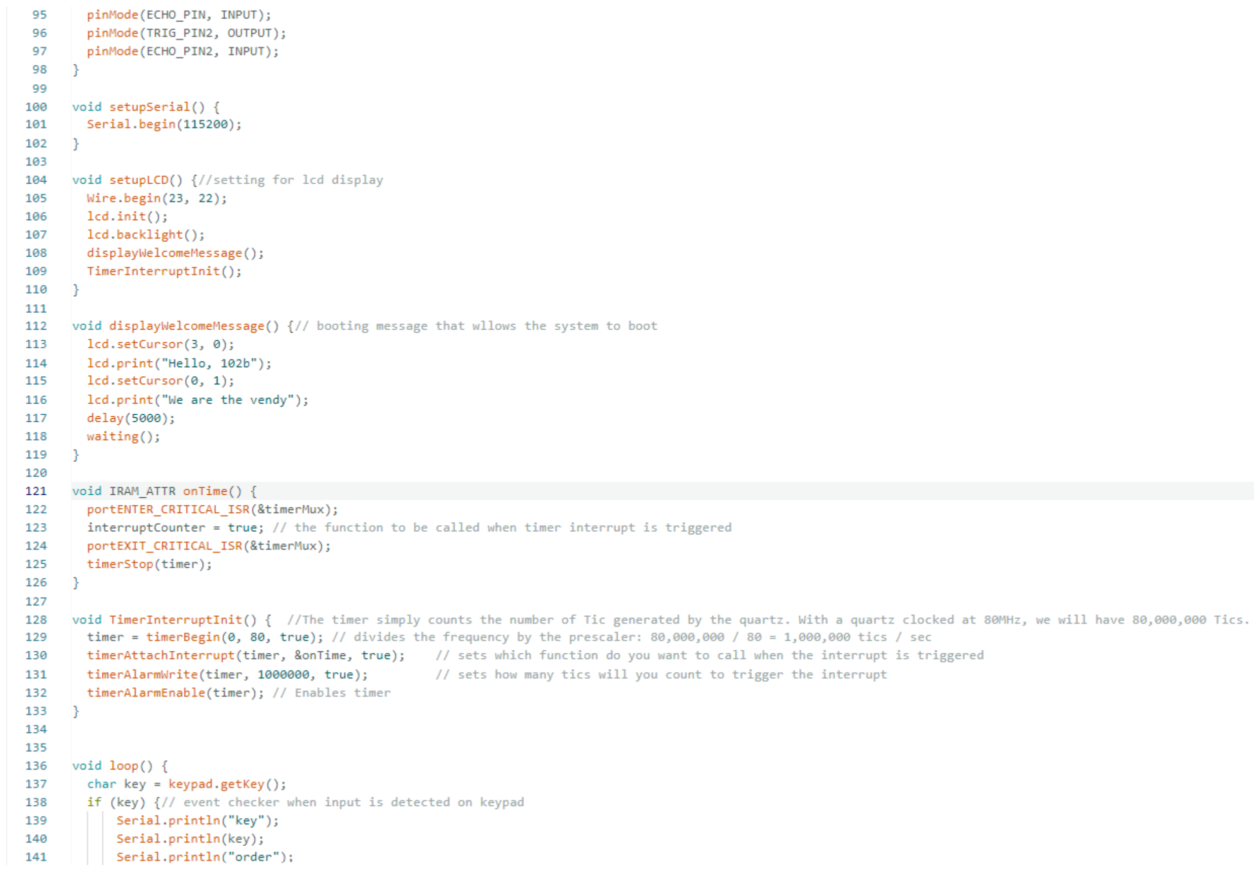


Screenshots of Code:

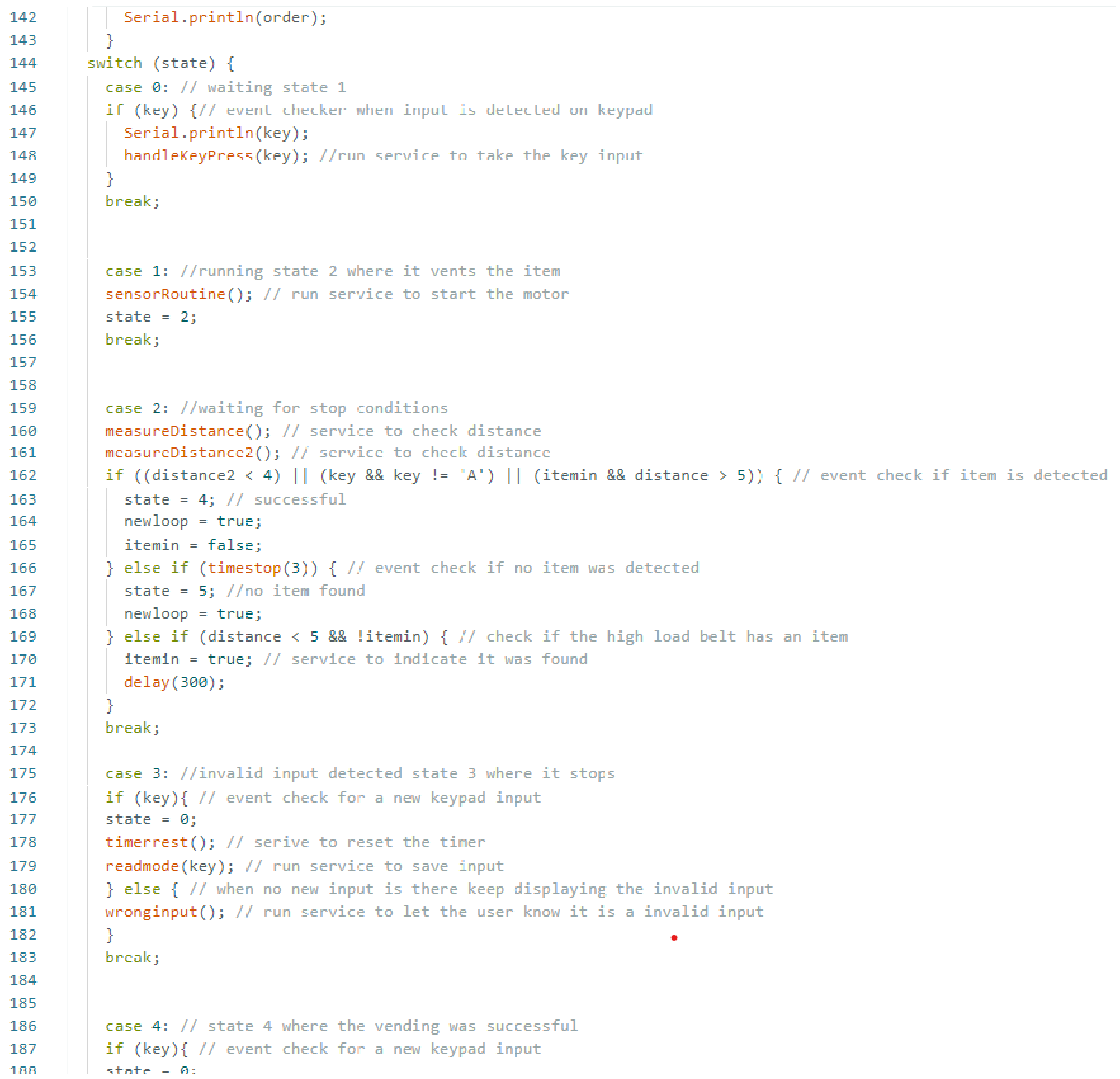


```
#include <Keypad.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

//left motor
#define BIN_1 26
#define BIN_2 25

//right motor
#define BIN_3 21
#define BIN_4 13

#define LED_PIN 13

#define TRIG_PIN 16
#define ECHO_PIN 17


#define TRIG_PIN2 4
#define ECHO_PIN2 5

// LCD setup
LiquidCrystal_I2C lcd(0x27, 16, 2);

// Keypad setup
const byte ROW_NUM = 4; // four rows
const byte COLUMN_NUM = 4; // four columns
char keys[ROW_NUM][COLUMN_NUM] = {
  {'1','4','7', '*'},
  {'2','5','8', '0'},
  {'3','6','9', '#'},
  {'A','B','C', 'D'}
};
byte pin_rows[ROW_NUM] = {14, 32, 15, 33};
byte pin_column[COLUMN_NUM] = {27, 12, 19, 18};
Keypad keypad = Keypad(makeKeymap(keys), pin_rows, pin_column, ROW_NUM, COLUMN_NUM);

// PWM setup
const int freq = 5000;
const int ledChannel_1 = 1;
const int ledChannel_2 = 2;
const int ledChannel_3 = 3;
const int ledChannel_4 = 4;
const int resolution = 8;
int MAX_PWM_VOLTAGE = 235;
int motorselect = 1;

```

```
//defines for ultrasonic
long duration;
int distance;
long duration2;
int distance2;
bool itemin = false;

//defines for timers
bool turn = false;
volatile bool interruptCounter = false;    // check timer interrupt
int totalInterrupts;   // counts the number of triggering of the alarm
hw_timer_t * timer = NULL;
portMUX_TYPE timerMux = portMUX_INITIALIZER_UNLOCKED;

//define input
String order = "";
int count = 0;
bool moving = false;
bool newloop = false;
bool looping = false;

//define state
int state = 0; // waiting state


void setup() {// set up the output and input pins
  setupLED();
  setupDistanceSensor();
  setupSerial();
  setupLCD();
}

void setupLED() { //setting for motor
  pinMode(LED_PIN, OUTPUT);
  digitalWrite(LED_PIN, LOW);
  ledcSetup(ledChannel_1, freq, resolution);
  ledcSetup(ledChannel_2, freq, resolution);
  ledcAttachPin(BIN_1, ledChannel_1);
  ledcAttachPin(BIN_2, ledChannel_2);
  ledcSetup(ledChannel_3, freq, resolution);
  ledcSetup(ledChannel_4, freq, resolution);
  ledcAttachPin(BIN_3, ledChannel_3);
  ledcAttachPin(BIN_4, ledChannel_4);
}

void setupDistanceSensor() {// setting for ultrasonic
  pinMode(TRIG_PIN, OUTPUT);
```



```
  pinMode(ECHO_PIN, INPUT);
  pinMode(TRIG_PIN2, OUTPUT);
  pinMode(ECHO_PIN2, INPUT);
}

void setupSerial() {
  Serial.begin(115200);
}

void setupLCD() {//setting for lcd display
  Wire.begin(23, 22);
  lcd.init();
  lcd.backlight();
  displayWelcomeMessage();
  TimerInterruptInit();
}

void displayWelcomeMessage() {// booting message that wllows the system to boot
  lcd.setCursor(3, 0);
  lcd.print("Hello, 102b");
  lcd.setCursor(0, 1);
  lcd.print("We are the vendy");
  delay(5000);
  waiting();
}

void IRAM_ATTR onTime() {
  portENTER_CRITICAL_ISR(&timerMux);
  interruptCounter = true; // the function to be called when timer interrupt is triggered
  portEXIT_CRITICAL_ISR(&timerMux);
  timerStop(timer);
}

void TimerInterruptInit() {  //The timer simply counts the number of Tic generated by the quartz. With a quartz clocked at 80MHz, we will have 80,000,000 Tics.
  timer = timerBegin(0, 80, true); // divides the frequency by the prescaler: 80,000,000 / 80 = 1,000,000 tics / sec
  timerAttachInterrupt(timer, &onTime, true);    // sets which function do you want to call when the interrupt is triggered
  timerAlarmWrite(timer, 1000000, true);         // sets how many tics will you count to trigger the interrupt
  timerAlarmEnable(timer); // Enables timer
}


void loop() {
  char key = keypad.getKey();
  if (key) {// event checker when input is detected on keypad
      Serial.println("key");
      Serial.println(key);
      Serial.println("order");
```



```
    Serial.println(order);
  }
switch (state) {
  case 0: // waiting state 1
  if (key) {// event checker when input is detected on keypad
    Serial.println(key);
    handleKeyPress(key); //run service to take the key input
  }
  break;


  case 1: //running state 2 where it vents the item
  sensorRoutine(); // run service to start the motor
  state = 2;
  break;


  case 2: //waiting for stop conditions
  measureDistance(); // service to check distance
  measureDistance2(); // service to check distance
  if ((distance2 < 4) || (key && key != 'A') || (itemin && distance > 5)) { // event check if item is detected
    state = 4; // successful
    newloop = true;
    itemin = false;
  } else if (timestop(3)) { // event check if no item was detected
    state = 5; //no item found
    newloop = true;
  } else if (distance < 5 && !itemin) { // check if the high load belt has an item
    itemin = true; // service to indicate it was found
    delay(300);
  }
  break;

  case 3: //invalid input detected state 3 where it stops
  if (key){ // event check for a new keypad input
  state = 0;
  timerrest(); // serive to reset the timer
  readmode(key); // run service to save input
  } else { // when no new input is there keep displaying the invalid input
  wronginput(); // run service to let the user know it is a invalid input
  }
  break;


  case 4: // state 4 where the vending was successful
  if (key){ // event check for a new keypad input
  state = 0;
```

```
    state = 0;
    timerrest();  //reset timer
    readmode(key); // run service to save input
    } else { //when no new input is there keep displaying the successful service
    successful(); // display the successful serive
    }
    break;


    case 5: // state 5 where no products was found
    if (newloop){ //event check to see if this is a new loop
      noitem(); //run the no time service
      newloop = false; // make it to a second loop
    } else if (key || timestop(5)) { //  when there is a new input or have a5 second pass it reset the state to state 1
      state = 0;
    }
    break;
  }
}

void readmode(char key){// reading mode that make the input into string
  if(count <2){
    order += key;
  } else {
    count = 0;
    order = key;
  }

  if(key == '*'){
    count = -1;
    order = "";
  }
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Curruent order: ");
  lcd.setCursor(14, 1);
  lcd.print(order);
  count++;
}

void handleKeyPress(char key) { //different input depending on key input
  if (key == '#') {
    if(order == "A1"){
    state = 1; // sensor mode
    motorselect = 1; // left motor
    } else if (order == "A2"){
    state = 1; // sensor mode
    motorselect = 2; // left motor
    } else {
    newloop = true;
    state = 3;
    }
  } else {
    readmode(key);
  }
}

void sensorRoutine() {// check for item falling
  if (moving == false){
    lcd.clear();
    lcd.setCursor(2, 0);
    lcd.print("Now Vending");
    lcd.setCursor(0, 1);
    lcd.print("Product  :");
    lcd.setCursor(14, 1);
    lcd.print(order);

  }
  moving = true;
  measureDistance();
  measureDistance2();
  delay(5);
  motoron(motorselect);
}

void measureDistance() {// look for distance on ultrasonic sensor
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  duration = pulseIn(ECHO_PIN, HIGH);
  distance = duration * 0.034 / 2;
  Serial.print("Distance: ");
  Serial.println(distance);
}

void measureDistance2() {// look for distance on right ultrasonic sensor
  digitalWrite(TRIG_PIN2, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN2, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN2, LOW);
  duration2 = pulseIn(ECHO_PIN2, HIGH);
  distance2 = duration2 * 0.034 / 2;
  Serial.print("Distance2: ");
  Serial.println(distance2);
}

bool timestop(int delays){
  if (interruptCounter) { // interruptCounter will be 'true' when timer interrupt is triggered
  timerStart(timer);
  Serial.println("on");
  portENTER_CRITICAL(&timerMux);
  totalInterrupts++;

  Serial.print("totalInterrupts");
  Serial.println(totalInterrupts);
  Serial.print("Timer interrupt is triggered in every ");
  Serial.print(timerAlarmReadSeconds(timer));
  Serial.println(" second(s)");
  interruptCounter = false; // reset interruptCounter flag to false
  portEXIT_CRITICAL(&timerMux);
  if ( totalInterrupts%delays == 0) {
        Serial.println("time up");
        return true;
      }
  }
  return false;
}


void stopAllActivities() {//stop all when break event is met
  Serial.println("Stop");
  digitalWrite(LED_PIN, LOW);
  ledcWrite(ledChannel_1, LOW);
  ledcWrite(ledChannel_2, LOW);
  ledcWrite(ledChannel_3, LOW);
  ledcWrite(ledChannel_4, LOW);
  moving = false;
  delay(500);
  waiting();
}

void waiting(){//display the waiting message
  lcd.clear();
  lcd.setCursor(2, 0);
  lcd.print("Please enter");
  lcd.setCursor(0, 1);
  lcd.print("a product number");
}
```

```

void successful(){
  if (newloop){ // event check for a new loop
  stopAllActivities(); //run service to stop all motors and display
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("The vending was");
  lcd.setCursor(2, 1);
  lcd.print("successful");
  timerrest();
  newloop = false;
  looping = true;
  } else if (looping && timestop(5)) { // when it is displaying after 5 second ask the person to take the item
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Please take the");
    lcd.setCursor(2, 1);
    lcd.print("item under");
    looping = false;
  } else if (timestop(11)) { // event check to see if 11 second passes
    state = 0; // serive the state to 1 waiting for a input
    Serial.print("reset");
    waiting();
    timerrest();
  }
}

void timerrest(){ // resets the timer
  totalInterrupts = 0;
}

void noitem(){ // no item service
      stopAllActivities(); // stop all motors
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Sorry there was");
      lcd.setCursor(3, 1);
      lcd.print("no product");
      state = 3; // make the state to 3
      newloop = true; // make the next loop into a new loop
}

void motoron(int which){ // turn on the motor depending on the input
  switch (which) {
  case 1:
  Serial.print("right");
  digitalWrite(LED_PIN, HIGH);
  ledcWrite(ledChannel_1, LOW);
  ledcWrite(ledChannel_3, MAX_PWM_VOLTAGE);
  break;

  case 2:
  digitalWrite(LED_PIN, HIGH);
  Serial.print("left");
  ledcWrite(ledChannel_4, 255);
  ledcWrite(ledChannel_2, LOW);
  break;
  }
}

void wronginput(){ // check for wrong input
  if(newloop == true){ //event check if the new loop is in
    lcd.clear();
    lcd.setCursor(1, 0);
    lcd.print("Invalid input");
    lcd.setCursor(0, 1);
    lcd.print("Please try again");
    count = 0;
    order = "";
    newloop = false;
  } else if (timestop(6)){//event check if 6 seocnd has passes after first loop has run
    state = 0; // run service back to state 1
    timerrest();
    waiting();
  }
}
```