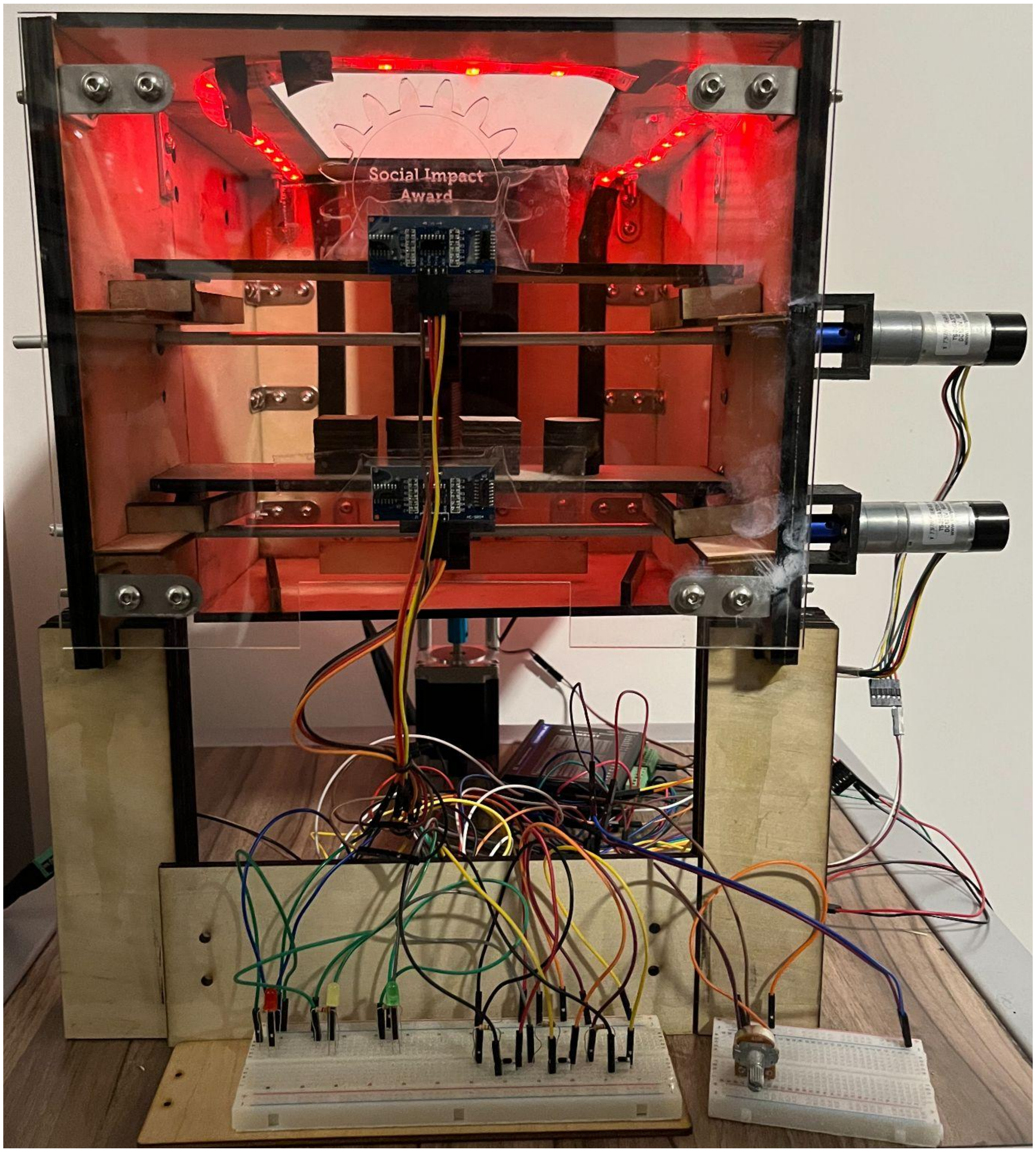
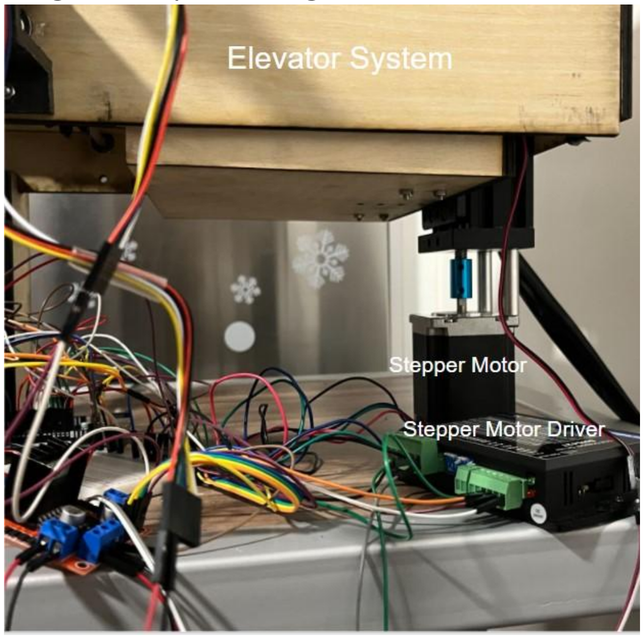
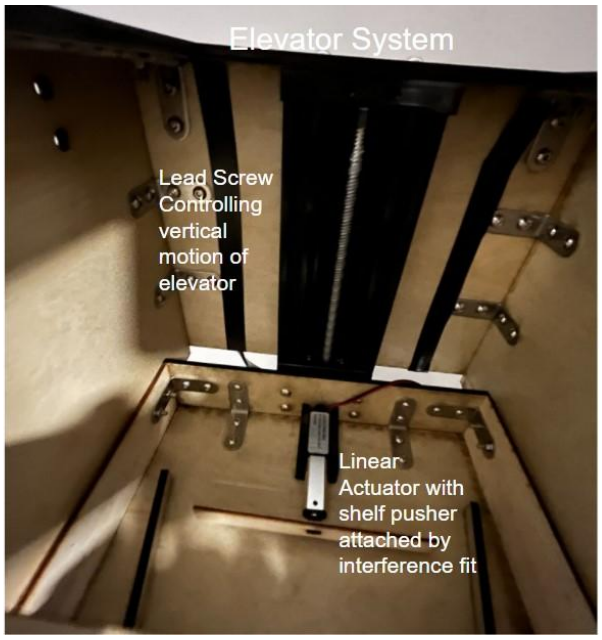
Assistive Cabinet

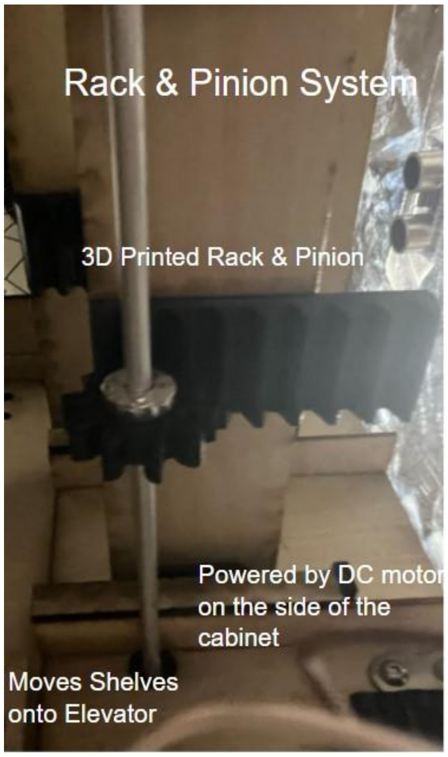
Monet Garrett, Mohamed Mohamed, Kevin Rubio
Group 17

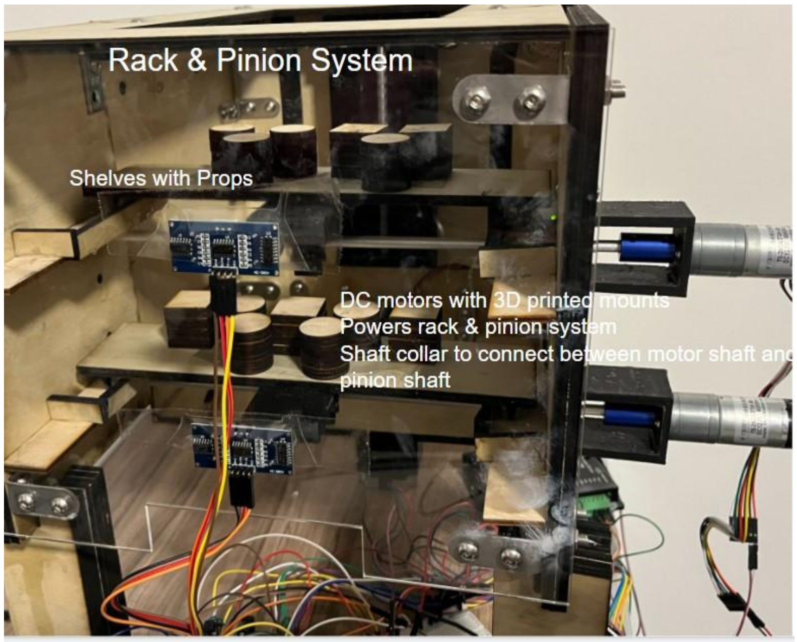


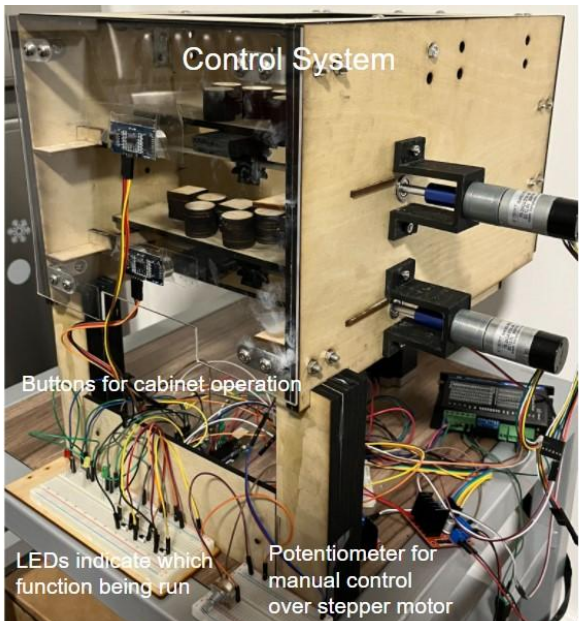
Opportunity
When it comes to cabinets, some shelves are out of reach. Although there are stepping stools, it is an inconvenience to take them wherever you need or find them difficult to use. We developed a device that assists in bringing objects down from higher shelves for people who are unable to reach high shelves. With the implementation of LEDs to maintain an aesthetically pleasing look, the cabinet uses a rack and pinion system to reach desired shelves. There are currently no similar products available that easily lower shelves with an aesthetic appearance.
High Level Strategy
The operation of the cabinet requires users to press one of the three buttons attached to a breadboard. As one button is pressed, its corresponding LED light will turn on to help indicate the user of their choice. Pressing the first button(State 0) returns a missing shelf. To return the shelf, the ultrasonic sensor checks which shelf is missing. After the readings are processed, the elevator moves up to desired shelf height. The elevator consists of a lead-screw linear guide powered by a stepper motor for vertical motion and a holder for where the shelves can be safely transported. The elevator is placed slightly higher than the shelf level initially. On the holder, there is also a linear actuator that pushes each shelf back into contact with the rack and pinion system. Once the linear actuator is fully pushed out, the elevator lowers slightly to align the shelf rack to the pinion. Powered by a DC motor, the rack and pinion system pushes the shelf back into its idle position.
Pressing the second button(State 1) results in the cabinet getting the second shelf. In this case, the elevator will move up to the first shelf height. The rack and pinion system will then activate to move the shelf onto the holder attached to the elevator. Pressing the third button(State 2) will get the second shelf in a similar manner as above.
All functions are controlled by the different button inputs and each state shines an LED color to help the user understand which function the code is running.
Integrated Physical Design

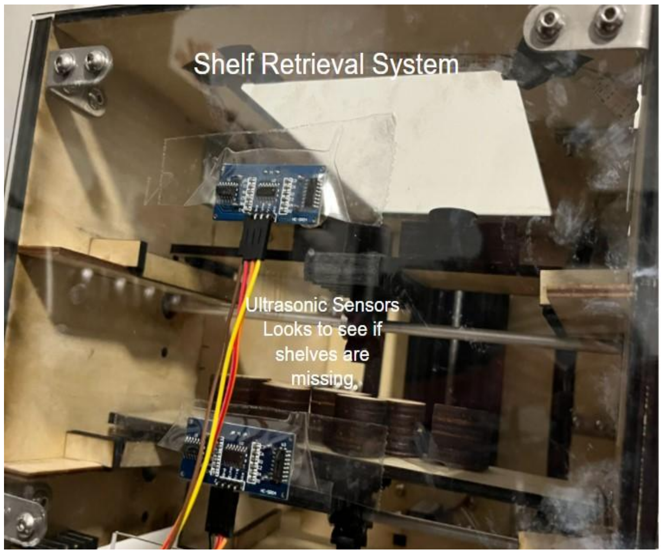



Elevator System
Lead Screw Controlling vertical motion of elevator
Linear Actuator with shelf pusher attached by interference fit


Rack & Pinion System
3D Printed Rack & Pinion
Powered by DC motor on the side of the cabinet
Moves Shelves onto Elevator


Rack & Pinion System
Shelves with Props
DC motors with 3D printed mounts
Powers rack & pinion system
Shaft collar to connect between motor shaft and
pinion shaft


Control System
Buttons for cabinet operation
LEDs indicate which function being run
Potentiometer for manual control over stepper motor



Function Critical Decisions

<u>Rack and Pinion Shelf Transfer System</u>

For our design, we were figuring out how to transfer the shelf platform to the elevator mechanism. Various mechanisms were considered for transferring the shelf platform, including lead screws, motorized shelves, and linear actuators on both ends. Each had their own respective issues either with cost, complexity, weight, or a combination of those characteristics. We decided to utilize a combination of mechanisms. For the stationary shelves, we used a rack and pinion mechanism for shelf transfer. A rack is attached to each shelf. Each rack is in contact with a shaft with a pinion attached, which is driven by a motor. The pinion rotating locks with the teeth on the rack, translating the shelves forward. To reduce friction, wheels are attached to the ends of the shelf.

- Calculations here:
- Our motor specifications are as follows:
    - 12 V input
    - No load speed of 1 revolution per second
    - Rated max torque is around 0.275 N*m

With the given pinion radius of 11.9 mm, we found the horizontal force taking the horizontal component of the force generated from the motor, which is around $(0.275/0.0119) * cos(20) = 21.68\,N$. This is the maximum force that the motor can emit to the rack. We knew that with wheels the shelves can be assumed to be frictionless. But we had to take into account the items on the shelves as well. We treat the items on the shelf as a single body.

- $Force\ of\ pinion = masses * acceleration\ from\ motor$
- $Masses = Mass\ of\ Shelf + Mass\ of\ Items$
- We do summation of forces along the x and y directions
- $\Sigma F_x = Mass\ of\ Items * acceleration\ from\ motor = Friction\ Force$
- $\Sigma F_y = 0 = N - Mass\ of\ Items * 9.18\,m/s^2$->$N = Mass\ of\ Items * 9.18\,m/s^2$
- For maintaining static friction, the inequality
  $Friction\ Force <= \mu * N = \mu * Mass\ of\ Items * 9.18\,m/s^2$->
  $Friction\ Force/Mass\ of\ Items <= \mu * 9.18\,m/s^2$
- We can take the acceleration and relate the pinion force with static friction as
  $a = Force\ of\ Pinion/masses = Friction\ Force/Mass\ of\ Items$
- We can set this equal to the friction force inequality as $21.68\,N <=$
  $\mu * 9.18\,m/s^2 * masses = 0.8 * 9.18$ *masses
- As long as the masses theoretically exceed 2.95 kg, the items will not slip.

Change from Arduino Uno to Mega

We selected to use buttons for our digital input interface to allow for selection of shelves. Although we had buttons for two shelves, we also included another button for the shelf return feature. We had to have at least 3 interrupt pins in a microcontroller to incorporate these buttons. We had first selected to use an Arduino Uno, since our stepper motor driver needed 5 volt inputs to function. But the Uno only had 2 interrupt pins available. So we instead used the Arduino Mega 2560 Rev3 microcontroller. This controller had 6 interrupt pins, which were more than enough for our desired digital interface. We also had a variety of pins to incorporate our various mechanical subsystems, motors, and actuators.

Stepper Motor selection, position calibration, and overheating

We also used a stepper motor to drive our linear guide. The linear guide consisted of a lead screw with an 8mm lead. It is attached to a carriage with a threaded hole for the lead screw and wheels to make contact with the c-beam rail it is housed. For our motor selection, we used a Nema 23 motor with the following specifications: With the 1600 step selection and the time increment between each step being 100 microseconds, the estimated torque based on the 2000 step specification sheet would be around 1.3 N*m. The weight of the elevator with a shelf added on is around 6 N. The following formulas were used to calculate the rising and lowering torque needed to move the carriage.

- Rising Torque = $(F * diameter/2).* (lead + \pi * \mu * diameter)/(\pi * diameter - \mu * lead)$
- Lowering Torque = $(F.* diameter/2).* (\pi * \mu * diameter - lead)/(\pi * diameter + \mu * lead)$

μ represents the coefficient of static friction of aluminum. With both a diameter and a lead of 8mm, a static coefficient of 0.3 for aluminum to aluminum contact, the rising torque is 0.0164 N*m while the lowering torque would be 4.0113 * 10^-4 N*m. Our stepper motor is sufficient in driving the lead screw with the suggested weight, with potential to lift heavier objects.
The lead of the lead screw shows how much it has traveled with one rotation of the motor. The positioning of the carriage is based on the step settings of the driver (1600 steps per revolution), and to determine the height of each shelf position it is calculated as follows:

- $Steps\ per\ Revolution = 1600$
- $Lead = Height\ per\ Revolution = 8\text{mm}$
- Relating the number of steps to the lead, we get 200 steps per mm.

To find the number of steps needed to embed into our programming, we multiply this rate by the desired height we want from our measurements, and calibrate from there. The heights we needed were at 59.5 mm for shelf 1 and 147.5 mm for shelf 2. The steps needed to displace for these are 11900 and 29500, respectively.

Circuit Diagram

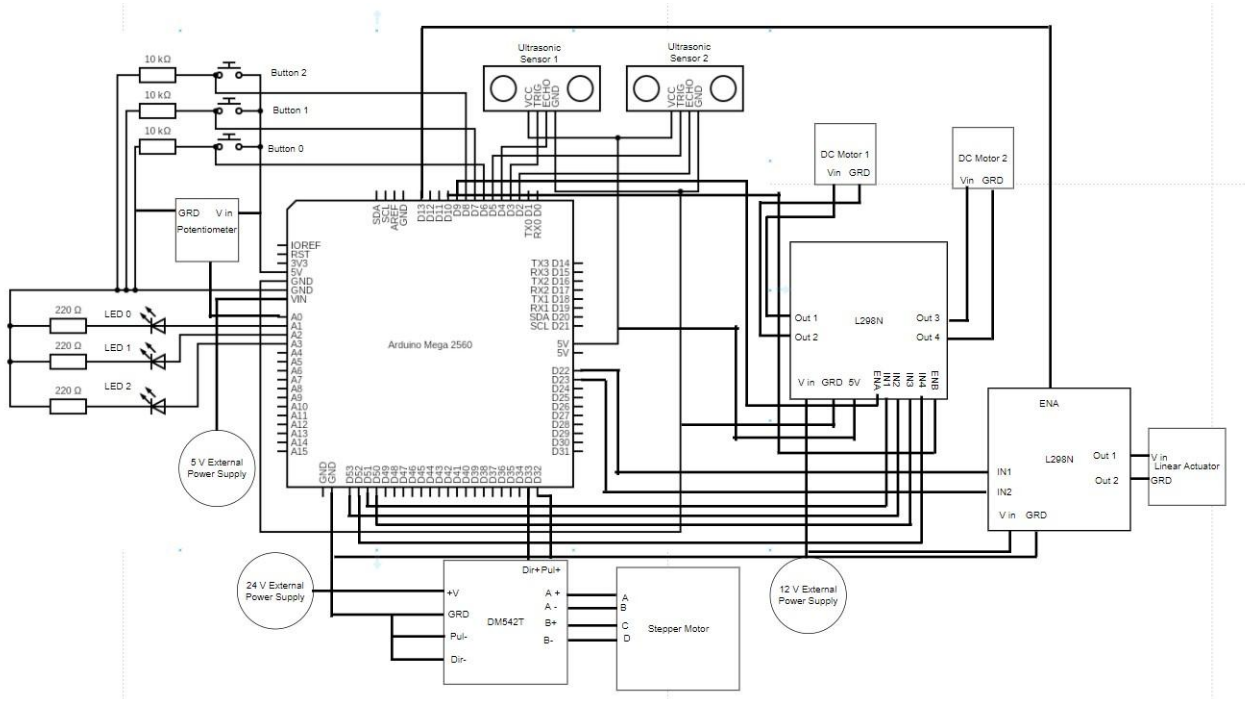


Diagram: https://drive.google.com/file/d/1b_py12VaO3QC3ksgTHKnClD3XzoUimln/view?usp=sharing

State Diagram

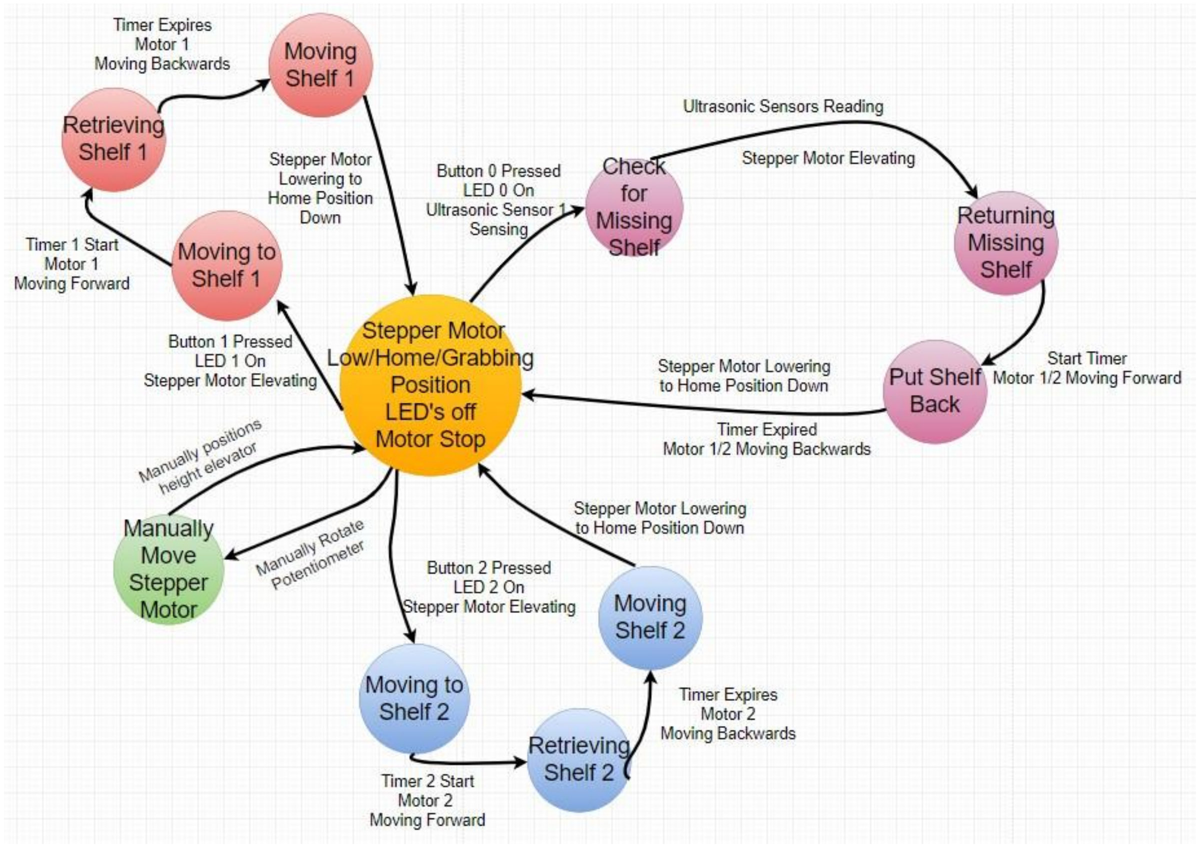


Diagram:https://drive.google.com/file/d/1dfpa8nPIaJiHONG0OkDBvyUDC_aAwZQS/view?usp=sharing

Reflection

We wished we had developed an easier/smoother transition between shelves and lowering mechanism since it led to laborious calibration tests. However, we had a good division of tasks among our members, allowing us to work faster and more efficiently.

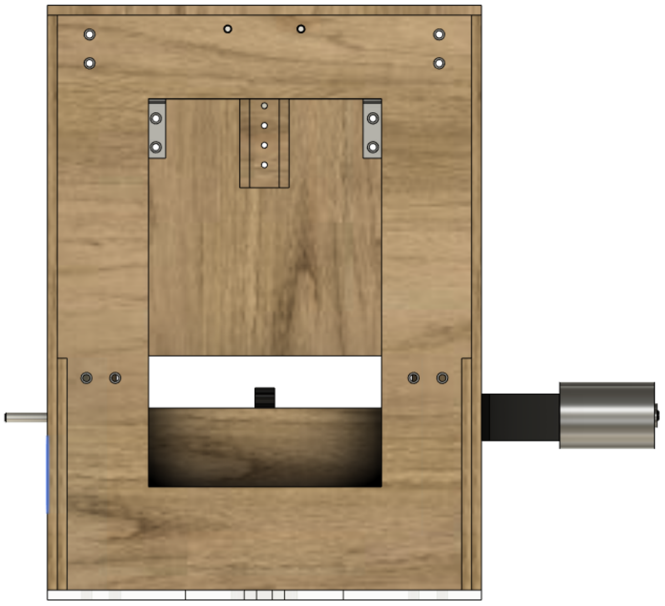
BOM

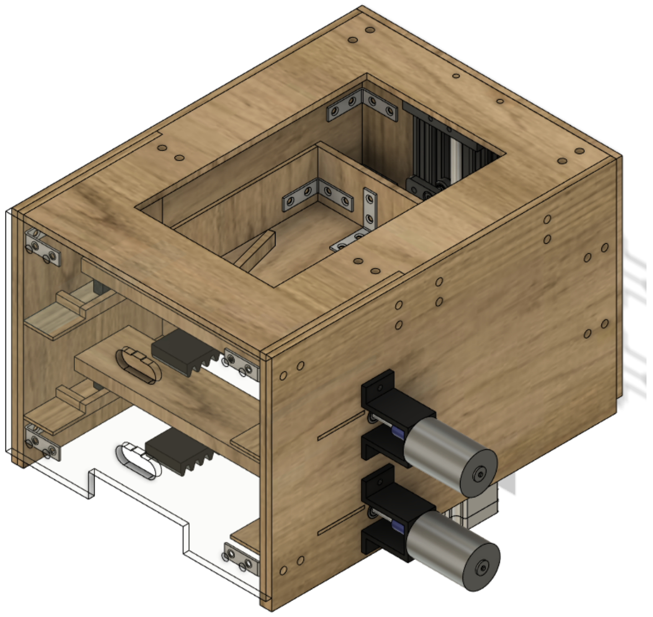
Item #	QTY	Item Name	Supplier Name	Cost per Unit	Total Price	Comments
Reference number	-	name of part	Name of supplier (may be easier to group by supplier)	If manufactured in house, give estimated cost of the maximum material you will need		Any comments you need to let us know. If you are manufacturing in house, please give *maximum* dimensions of the part
1	1	DC Motor Linear Actuator	Jacobs Material Store	26.97	26.97	

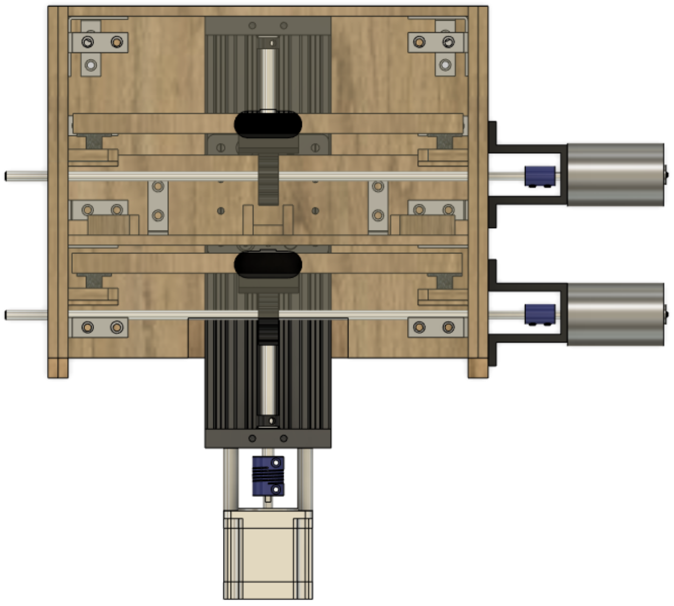
2	3	Plywood - 1/4" x 24" x 48"	Jacobs Material Store	13.32	39.96	Will be laser cut at Jacobs
3	2	Plywood - 1/4" x 12" x 24"	Jacobs Material Store	1.67	3.34	Will be cut at Jacobs
4	1	Acrylic - 1/8 x 12' x 24'	Invention Labs Material Store	9	9	Will be cut at Jacobs
5	1	Wires	Amazon	6	6	Pack of various lengthed wires
6	1	Arduino Mega	Amazon	34.94	34.94	
7	3	LEDs	Microkit	0	0	Have material through club
8	3	Buttons	Microkit	0	0	
9	2	Ultrasonic Sensor	Microkit	0	0	
10	1	PLA - 100m	Jacobs Material Store	0	0	3D printing at Jacobs Hall
11	1	Pack of Various Resistors	Member already owned	0	0	Mainly used 10K and 220 ohm resistors
12	2	12V DC Gear Motor	Amazon	17.88	35.76	

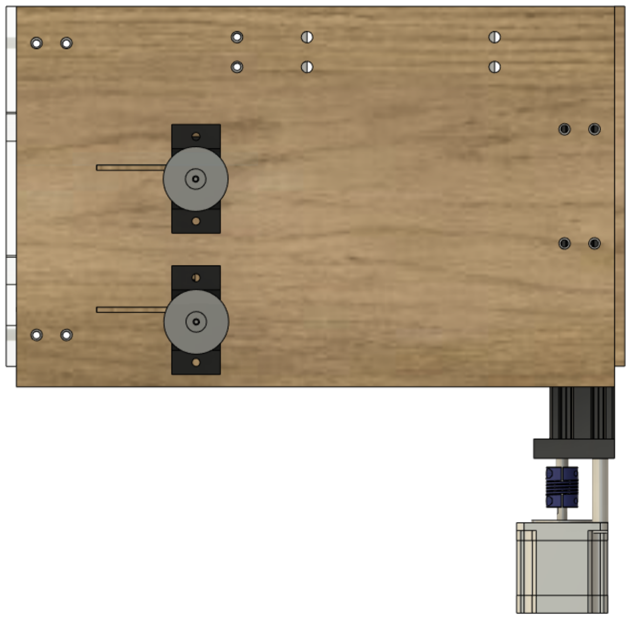
13	2	L298N DC Motor Drive Controller	Amazon	8.99	17.98	
14	1	CNC Stepper Motor Driver	Amazon	28.99	28.99	
15	1	Aluminum Profile Z axis Stepper Motor	Amazon	74.9	74.9	
16	1	Various Nuts and Screws	Amazon	17.89	17.89	Mainly used M3, M4, and M5 screws
17	1	12 V external Power Supply	Amazon	0	0	Came with Stepper Motor
18	1	5V power supply	Microkit	0	0	
19	1	24 V Power Supply	Amazon	0	0	Came with Stepper Motor
20	1	Plastic Castor Wheels	Amazon	7.99	7.99	

CAD Design Model









```
#include <AccelStepper.h>
const int motor1pin1 = 7;
const int motor1pin2 = 8;
const int ENA_pin = 4;
const int motor2pin1 = 9;
const int motor2pin2 = 10;
const int ENB_pin = 6;
const int actuatorpwm = 13;
const int actuatorpin1 = 22;
const int actuatorpin2 = 23;
const int BTN0 = 2;
const int BTN1 = 18;
const int BTN2 = 19;
const int LED0 = 11;
const int LED1 = 12;
const int LED2 = 51;
const int trigPin1 = 3;
const int echoPin1 = 5;
const int trigPin2 = 46;
const int echoPin2 = 47;
const int dirPin = 27;
const int stepPin = 26;
const int shelf1height1 = 12650;
const int shelf2height1 = 30250;
const int shelf1height2 = 11150;
const int shelf2height2 = 28750;
volatile int button = 3;
```

```
AccelStepper stepper(2, 26, 27);

int val = 0;
int previous = 0;
int long newval = 0;



long debouncing_time = 50; //Debouncing Time
in Milliseconds
volatile unsigned long last_millis;
long duration; // variable for the duration
of sound wave travel
int distance; // variable for the distance
measurement

void isr0() {
  if ((long)(millis() - last_millis) >=
debouncing_time) {
    button = 0;
    last_millis = millis();
  }
}

void isr1() {
  if ((long)(millis() - last_millis) >=
debouncing_time) {
    button = 1;
    last_millis = millis();
  }
}

void isr2() {
  if ((long)(millis() - last_millis) >=
debouncing_time) {
    button = 2;
    last_millis = millis();
  }
}

void setup() {
  Serial.begin(115200);
  // Motor Set Up:
  pinMode(motor1pin1, OUTPUT);
  pinMode(motor1pin2, OUTPUT);
  pinMode(ENA_pin, OUTPUT);
  pinMode(motor2pin1, OUTPUT);
  pinMode(motor2pin2, OUTPUT);
  pinMode(ENB_pin, OUTPUT);
  pinMode(actuatorpin1, OUTPUT);
  pinMode(actuatorpin2, OUTPUT);
  pinMode(actuatorpwm, OUTPUT);
  pinMode(dirPin, OUTPUT);
  digitalWrite(dirPin, LOW);
  pinMode(stepPin, OUTPUT);

  //Button Setup
  pinMode(BTN0, INPUT);

attachInterrupt(digitalPinToInterrupt(BTN0),
isr0, RISING);
  pinMode(BTN1, INPUT);

attachInterrupt(digitalPinToInterrupt(BTN1),
isr1, RISING);
  pinMode(BTN2, INPUT);

attachInterrupt(digitalPinToInterrupt(BTN2),
isr2, RISING);

  //LED Setup
  pinMode(LED0, OUTPUT);
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  digitalWrite(LED0, LOW);
  digitalWrite(LED1, LOW);
  digitalWrite(LED2, LOW);

  //Ultrasonic Sensor Setup
  pinMode(trigPin1, OUTPUT);
  pinMode(echoPin1, INPUT);
  pinMode(trigPin2, OUTPUT);
  pinMode(echoPin2, INPUT);

  //POT Control Set Up
  pinMode(A0, INPUT);
  stepper.setMaxSpeed(50000);
  stepper.setAcceleration(25000);
}

void manual() {
  val = analogRead(A0);
  if ((val > previous + 4) || (val <
previous - 4)) {
    int long newval = map(val, 0, 1024, 0,
130000);
    stepper.runToNewPosition(newval);
    previous = val;
  }
}

// Code to bring out shelf
void bringoutx(int shelf) {
  if (shelf == 1) {
    analogWrite(ENA_pin, 255);
    digitalWrite(motor1pin1, LOW);
    digitalWrite(motor1pin2, HIGH);
    Serial.println("First Motor");
    delay(2000);
    digitalWrite(motor1pin2, LOW);

  } else if (shelf == 2) {
    analogWrite(ENB_pin, 255);
    digitalWrite(motor2pin1, LOW);
    digitalWrite(motor2pin2, HIGH);
    Serial.println("Second Motor");
    delay(2000);
    digitalWrite(motor2pin2, LOW);

  }
}

//Code to put back shelf
void putbackx(int shelf) {
  analogWrite(actuatorpwm, 255);
  digitalWrite(actuatorpin1, HIGH);
  digitalWrite(actuatorpin2, LOW);
  delay(6000);
  if (shelf == 1) {
    elevatedown(1000);
    analogWrite(ENA_pin, 255);
    digitalWrite(motor1pin1, HIGH);
    digitalWrite(motor1pin2, LOW);
    Serial.println("First Motor");
    delay(900);
    digitalWrite(motor1pin1, LOW);
  } else if (shelf == 2) {
    elevatedown(1000);
    analogWrite(ENB_pin, 255);
    digitalWrite(motor2pin1, HIGH);
    digitalWrite(motor2pin2, LOW);
    Serial.println("Second Motor");
    delay(900);
    digitalWrite(motor2pin1, LOW);
  }
}

// code to lift shelf up
void elevateup(int shelf) {
  digitalWrite(dirPin, LOW); // Enables the
motor to move in a particular direction
  // Makes 200 pulses for making one full
cycle rotation
  for (int x = 0; x < shelf; x++) {
    digitalWrite(stepPin, HIGH);
    delayMicroseconds(50);
    digitalWrite(stepPin, LOW);
    delayMicroseconds(50);
  }
}
// code to lower shelf down
```

```
void elevatedown(int shelf) {
  digitalWrite(dirPin, HIGH); // Enables the
motor to move in a particular direction
  // Makes 200 pulses for making one full
cycle rotation
  for (int x = 0; x < shelf; x++) {
    digitalWrite(stepPin, HIGH);
    delayMicroseconds(50);
    digitalWrite(stepPin, LOW);
    delayMicroseconds(50);
    analogWrite(actuatorpwm, 255);
    digitalWrite(actuatorpin1, LOW);
    digitalWrite(actuatorpin2, HIGH);
  }
}
float ultrasonic(int shelf) {
  if (shelf == 1) {
    digitalWrite(trigPin1, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin1, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin1, LOW);
    duration = pulseIn(echoPin1, HIGH);
    distance = (duration * .0343) / 2;
    Serial.print("Distance: ");
    Serial.println(distance);
    delay(100);
    return distance;
  } else if (shelf == 2) {
    digitalWrite(trigPin2, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin2, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin2, LOW);
    duration = pulseIn(echoPin2, HIGH);
    distance = (duration * .0343) / 2;
    Serial.print("Distance: ");
    Serial.println(distance);
    delay(100);
    return distance;
  }
}

void loop() {
  switch (button) {
    case 0:
      digitalWrite(LED0, HIGH);
      if (ultrasonic(1) > 9 & ultrasonic(1)
< 100) {
        elevateup(shelf1height1);
        putbackx(1);
        elevatedown(shelf1height1 - 1000);

      } else if (ultrasonic(2) > 9 &
ultrasonic(2) < 100) {
          elevateup(shelf2height1);
          putbackx(2);
          elevatedown(shelf2height1 - 1000);
        }
        digitalWrite(LED0, LOW);
        button = 3;
        break;
      case 1:
        digitalWrite(LED1, HIGH);
        elevateup(shelf1height2);
        bringoutx(1);
        elevatedown(shelf1height2);
        digitalWrite(LED1, LOW);
        button = 3;
        break;
      case 2:
        digitalWrite(LED2, HIGH);
        elevateup(shelf2height2);
        bringoutx(2);
        elevatedown(shelf2height2);
        digitalWrite(LED2, LOW);
        button = 3;
        break;
      case 3:
        manual();
    }
```