

Digitizing the Restaurant with Network-Enabled Smart Tables

Group 3

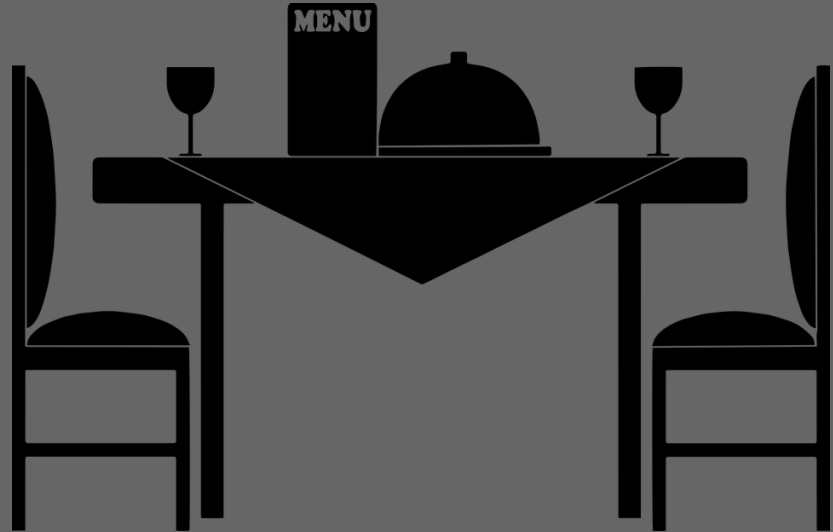
Andrew Chen (andrew6)

Eric Ong (eong3)

Can Zhou (czhou34)

Challenges in the Restaurant Industry

- **The Archaic Restaurant System**
- **Pandemic Regulations**
 - Restricted Dining Spaces
 - Limited Person-to-Person contact in Restaurant

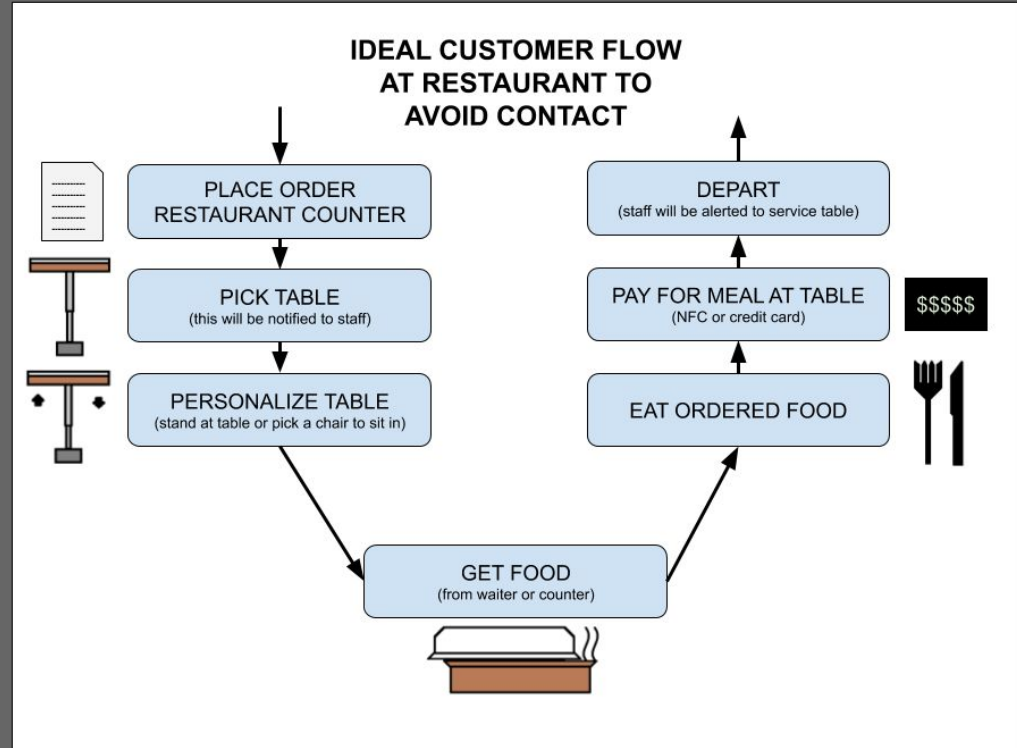


Our Solution

Modular seating device

Restaurant seating management
system

Contactless payment system built
into dining tables





Power & Control
Units

Weight Sensors

Raspberry Pi
Zero

NFC Reader

Bill Display

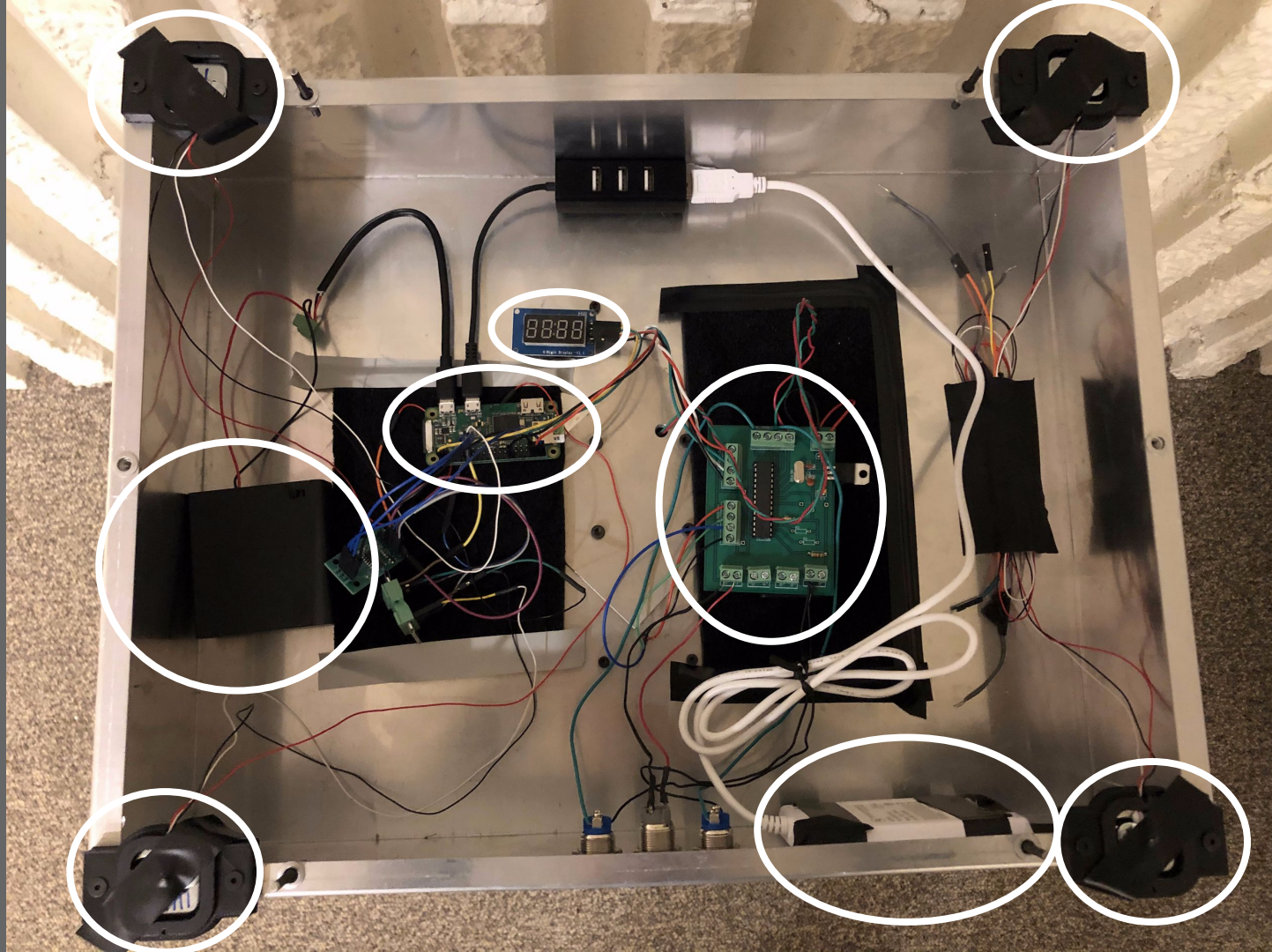


Table Design

Table - Physical Design

Wireless communication and payment modules are contained within the table.

Height adjustment driven by a motor attached to a lead screw, which rotates the load of the table itself.

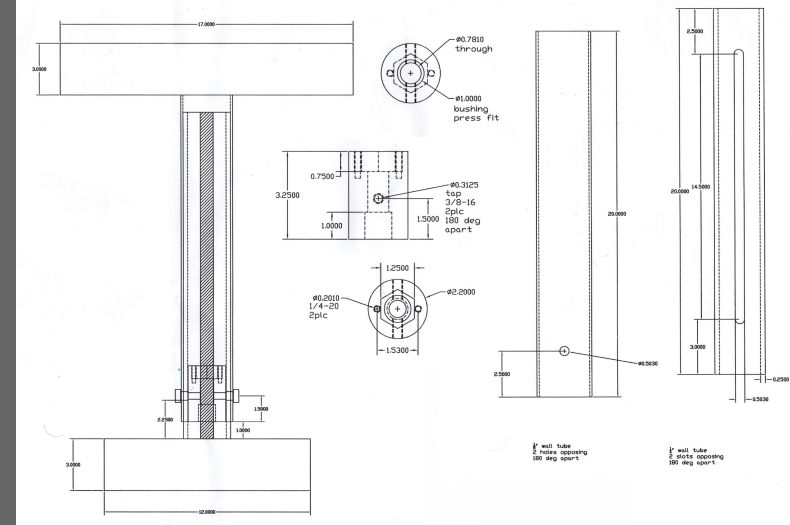


Table - Component Choices

L7805CV Linear Regulators -- More stable than batteries



L7805CV



HC-SR04

HC-SR04 Ultrasonic Distance Sensors --
Familiar, fast, and long range

ATMEGA328P-PU Microcontrollers --
Programmable with Arduino IDE, both digital and analog inputs, PWM output (for motor)



ATMEGA328P-PU

Table - Height Adjusting Algorithm

Used two HC-SR04 ultrasonic sensors, one for table height, the other for customer height detection

The prototype moved according to the customer height:

- $<7''$ or $>19''$: Up
- $>7''$ or $<9''$: Standby
- $>9''$ and $<19''$: Down



Table Height Adjustment R & V

Requirements:

1. **Sensors:** The ultrasonic sensors must be able to take 10 samples a second to determine the distance to any customer seating under it, and use a median to eliminate outliers
2. **Speed:** The height adjustment must work at a rate of an inch vertically around every 20 seconds
3. **Safety:** The height adjustment system must be able to work manually and must have safeties implemented, such that automatic adjustment is disabled

Verification:

1. We print the values read by the ultrasonic sensors into the serial output of our microcontroller, and confirm 10 values every second with a median
2. We set the table to run automatically and use a tape measure to confirm the height adjustment rate
3. We toggle the manual safety mode of the table while the table is adjusting automatically, and confirm that manual controls work

Table Height Adjustment R & V (1 of 3)

Requirements:

The ultrasonic sensors must be able to take 10 samples a second to determine the distance to any customer seating under it, and use a median to eliminate outliers

Verification:

We print the values read by the ultrasonic sensors into the serial output of our microcontroller, and confirm 10 values every second with a median

Using the Arduino IDE and connecting to a computer:

- **Print the ordered readings every second**
- **Print the acquired median every second**

Table Height Adjustment R & V (2 of 3)

Requirements:

The height adjustment must work at a rate of an inch vertically around every 20 seconds

Verification:

We set the table to run automatically and use a tape measure to confirm the height adjustment rate

- **Set table to manual mode**
- **Hold tape measurer parallel with shaft of the table**
- **Time the descent of the table and verify an inch takes between 10 and 20 seconds**

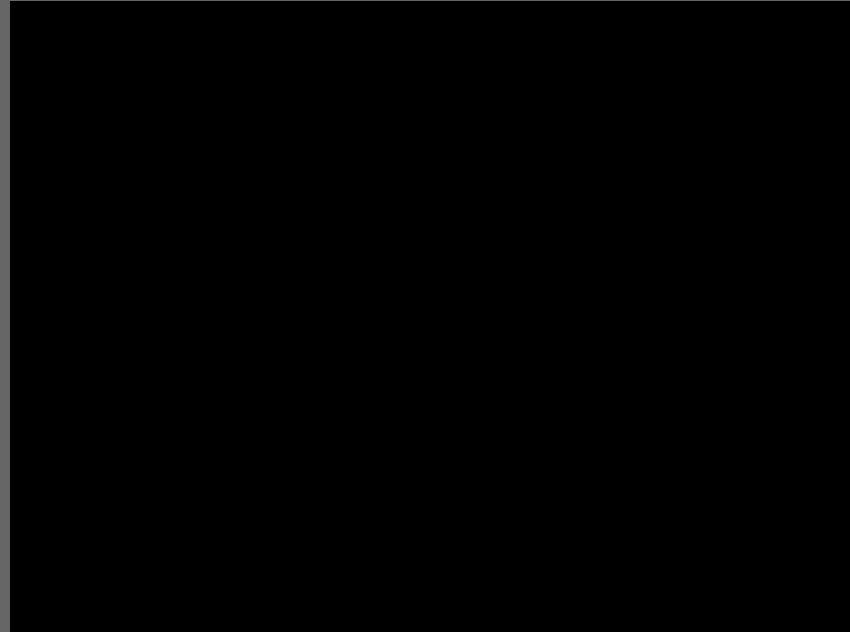


Table Height Adjustment R & V (3 of 3)

Requirements:

The height adjustment system must be able to work manually and must have safeties implemented, such that automatic adjustment is disabled

Verification:

We toggle the manual safety mode of the table while the table is adjusting automatically, and confirm that manual controls work

Using the Arduino IDE and connecting to a computer:

- **Print polling from automatic collection**
- **Enable the manual safety mode**
- **Print and visually confirm operation**

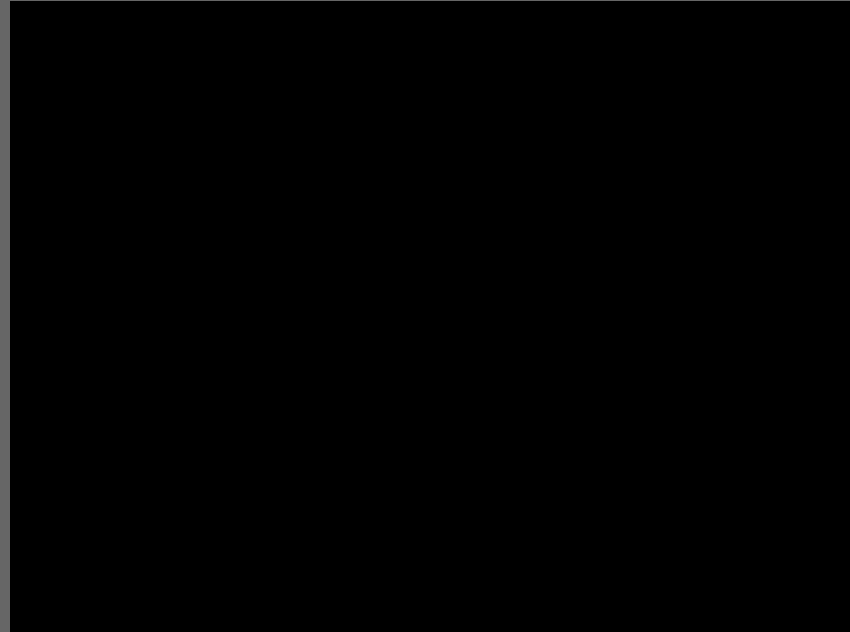


Table Height Adjustment Challenges

Problem: Digital pins 0 and 1 were dedicated to serial input/output -- limited functionality!

Unreliable input/output to microcontroller!

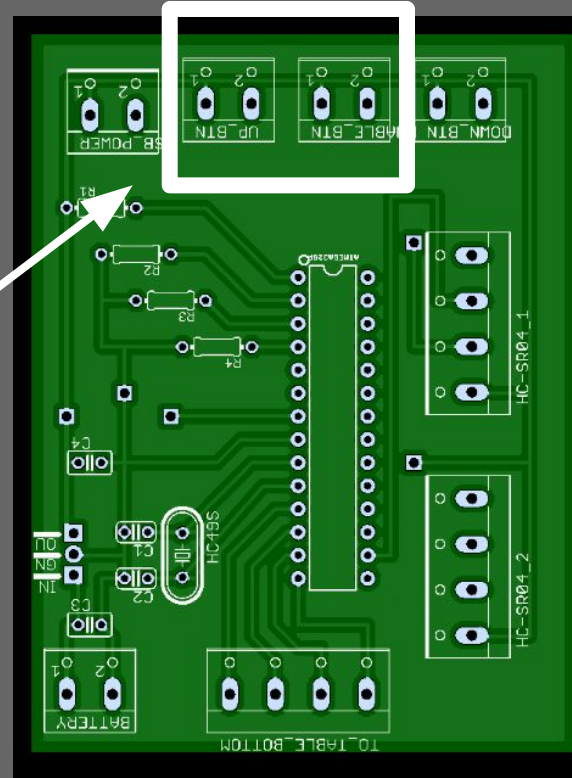


Table Height Adjustment Challenges

Solution: We had two vacant pins due to physical design constraints, and these were the same type of pins!

Available input pins!

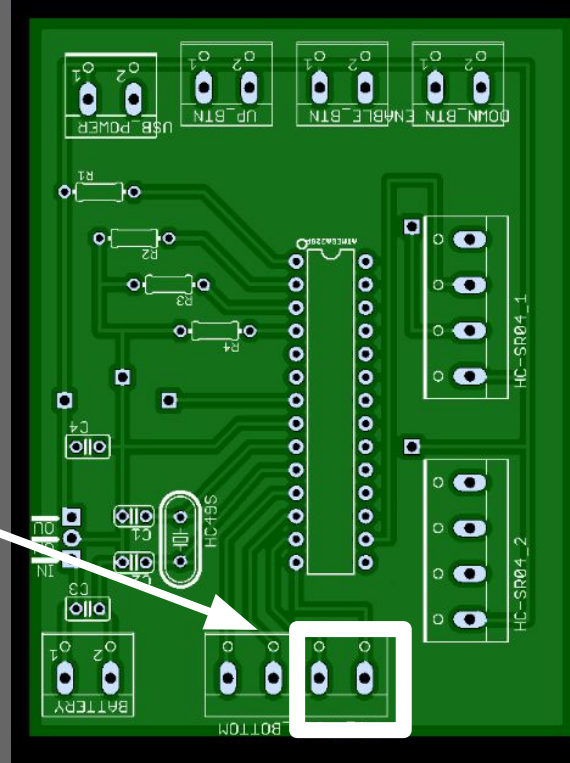


Table Height Adjustment Challenges

New problem: No way to power pins connected here, nor any feasible way to pull down these signals!

Incompatible!

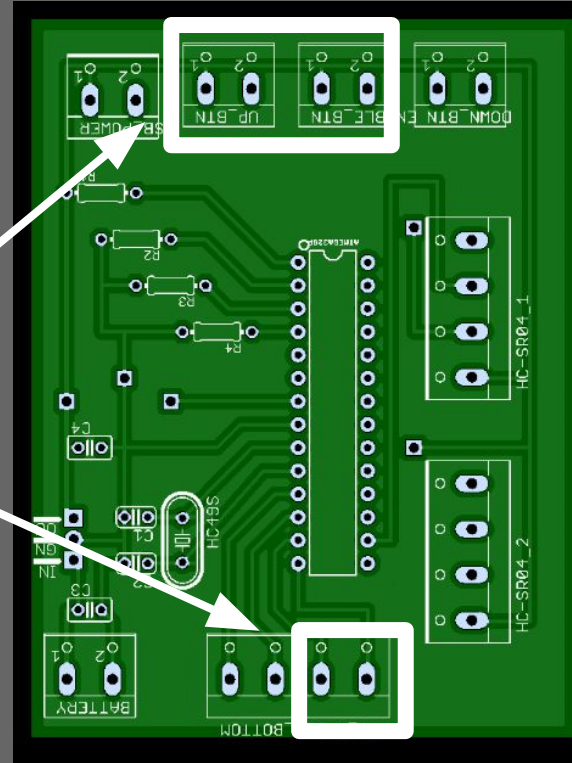
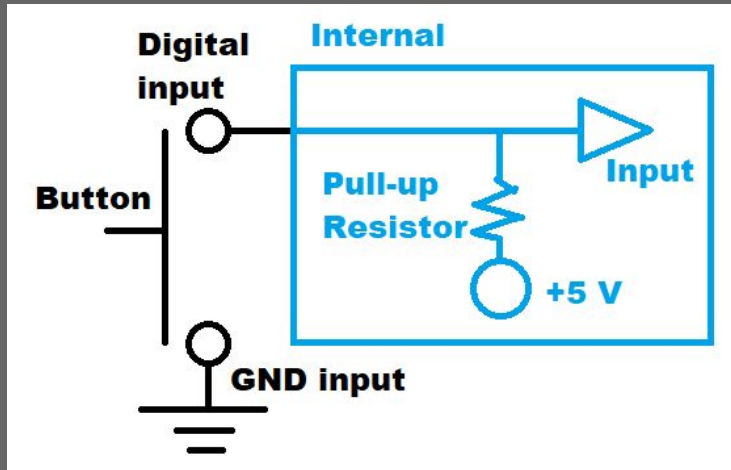


Table Height Adjustment Challenges

Solution: Use the pull-up resistors in each of these digital pins!



Have pull-up resistors

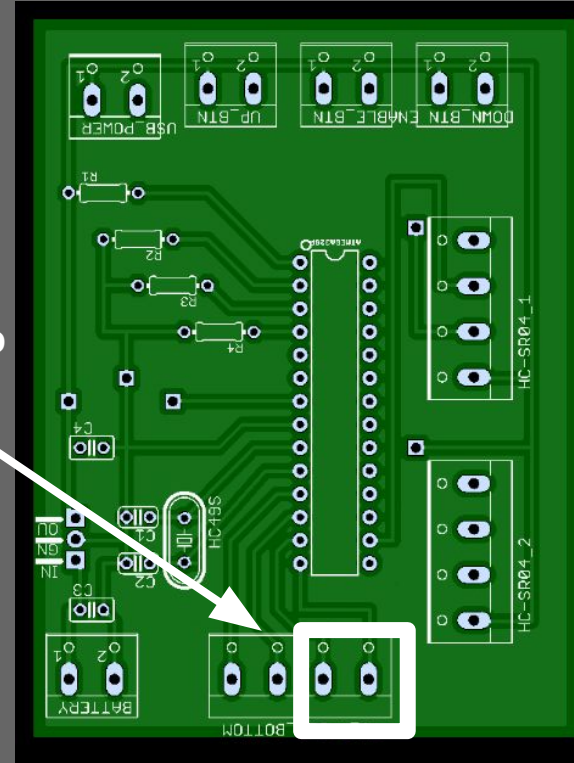


Table Height Adjustment Challenges

We retained all PCB functionality!

Bonus: Freed pins let us directly program the microcontroller via the PCB during testing!

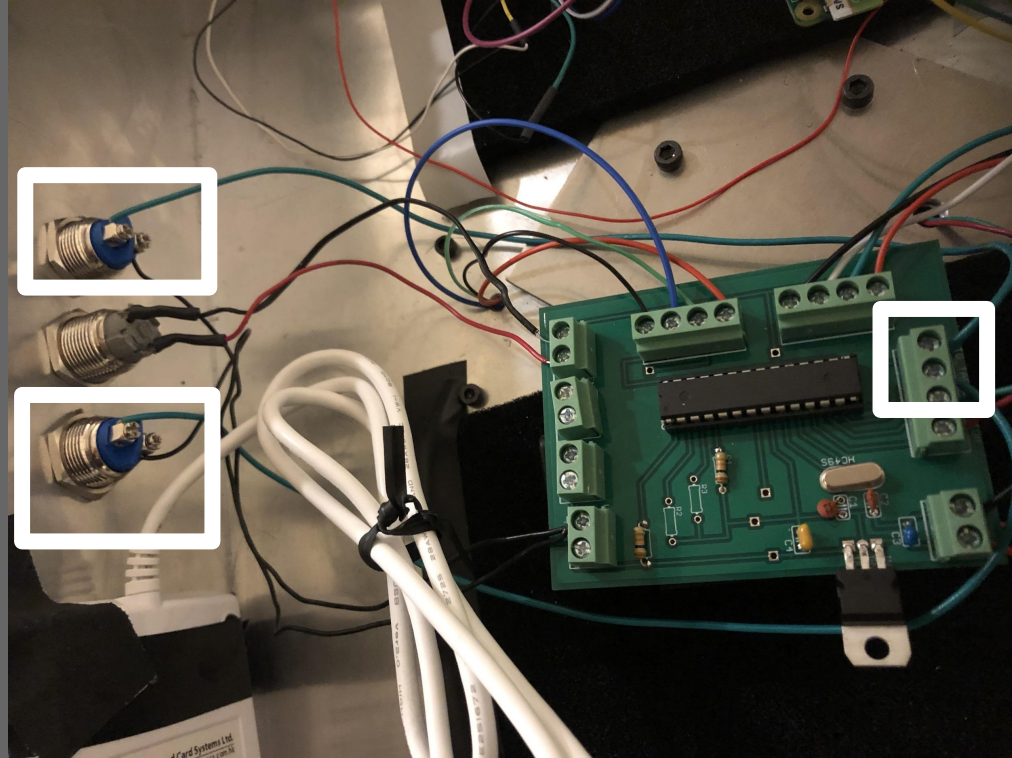


Table - Customer-Customized Seating

A system with adjustable height means every table can serve as:

- A standard height dining table
- A table that accommodates those with seating accommodations
- A bar height table
- A standing table

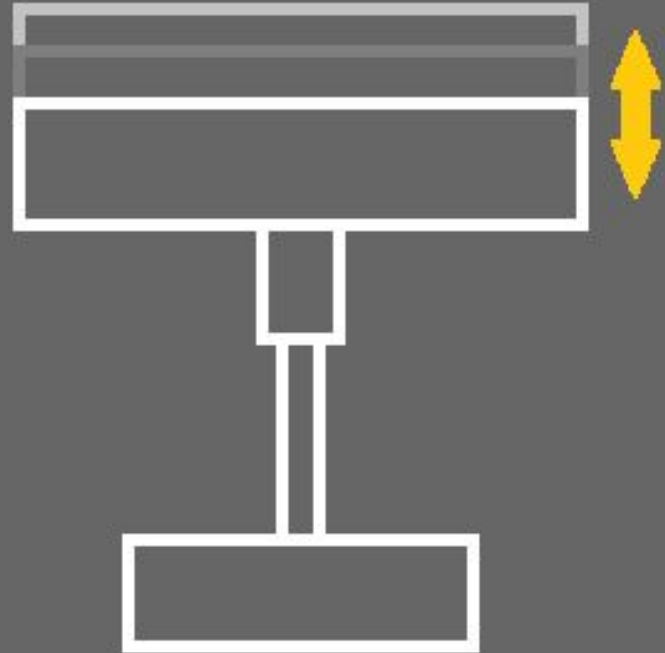


Table Network

Table Network - Communications

- **TCP/IP** for
Table-to-Dashboard
- **Bluetooth** for
Chair-to-Table

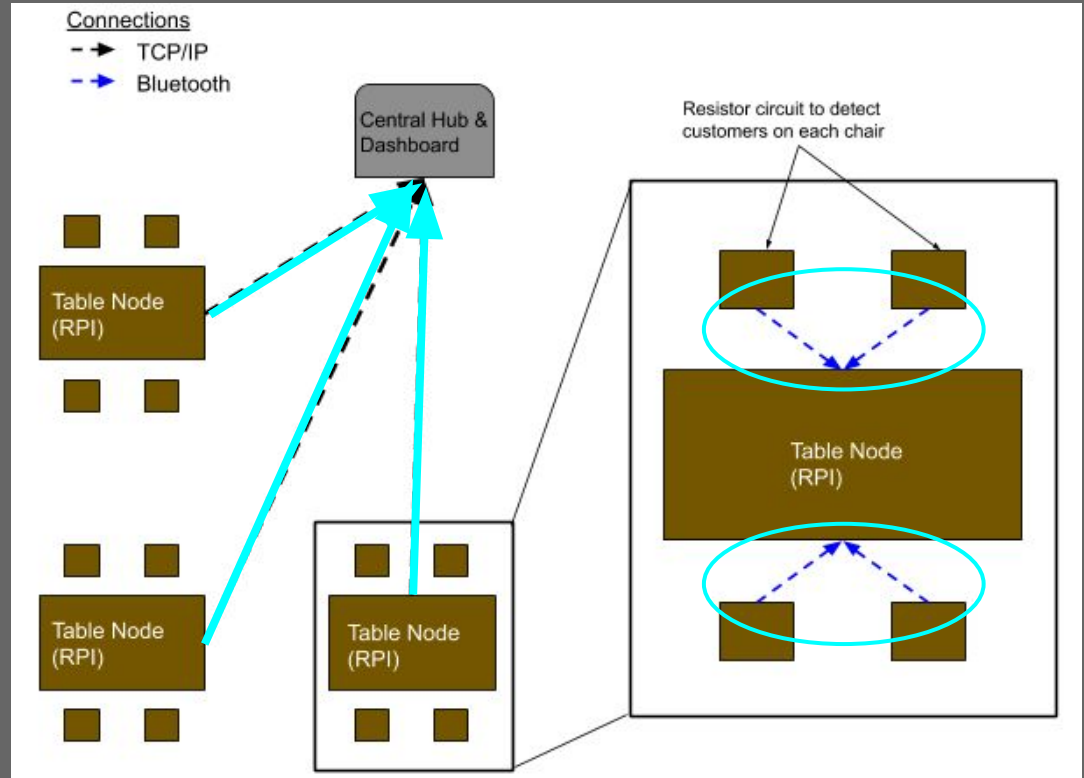


Table Network - Customer Presence

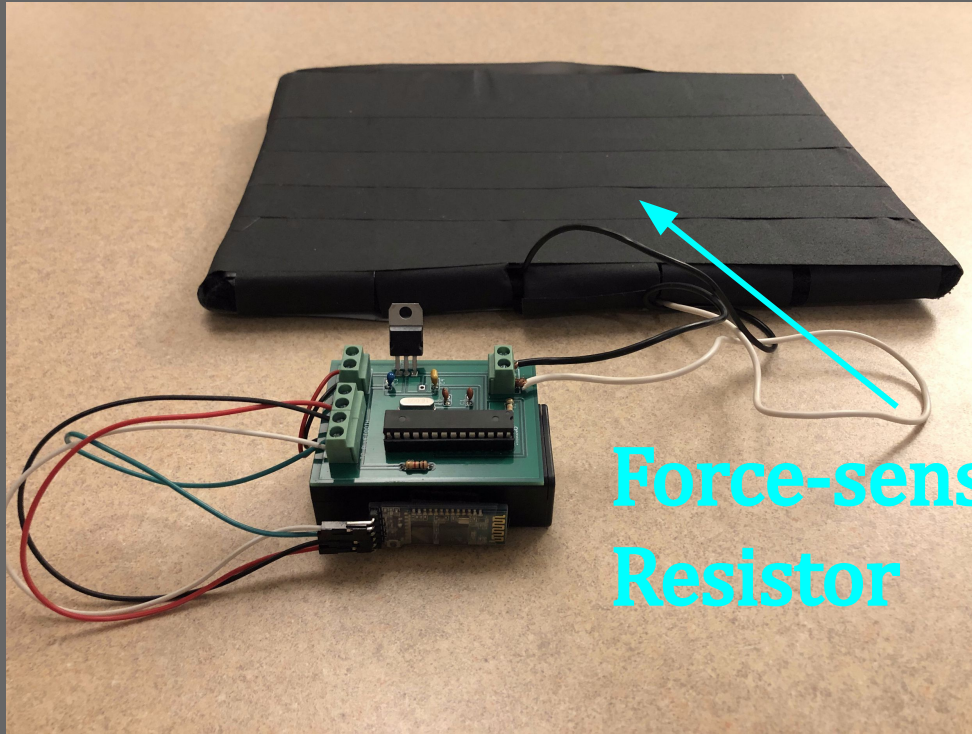
How do we know when someone is sitting at the table?

Chair Force Sensing Resistor

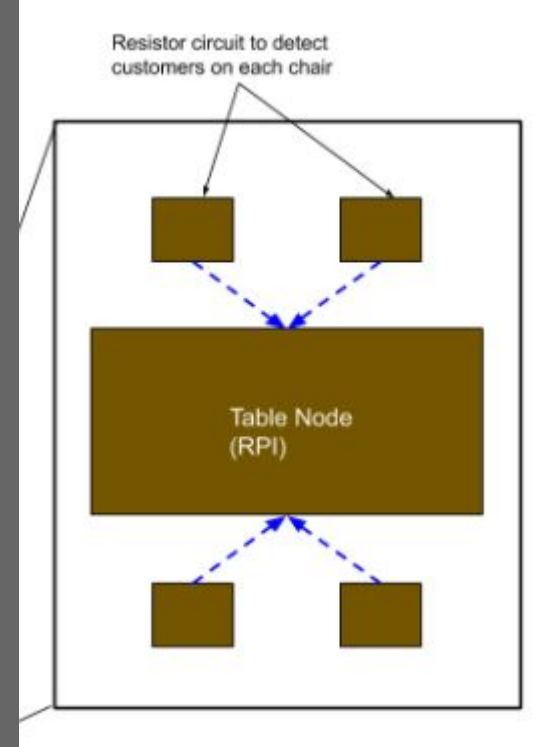
How do we know if the tabletop is still dirty?

Tabletop Load Cells

Table Network - Detecting Customers



Force-sensing
Resistor



Collected FSR Values

Weight on the FSR (in lb)	Resistance Measured by Voltmeter (in Ω)	Weight on the FSR (in lb)	Resistance Measured by Voltmeter (in Ω)
0	40 to 60 k Ω	31	82
17	330	32	80
18	130	33	92
19	130	34	71
20	200	35	150
21	155	36	81
22	134	37	70
23	142	38	62
24	136	39	66
25	132	40	45
26	140	41	43
27	109	42	42
28	108	43	44
29	90	44	40
30	400	45	49

Force Sensing Resistor (FSR) Force vs Resistance

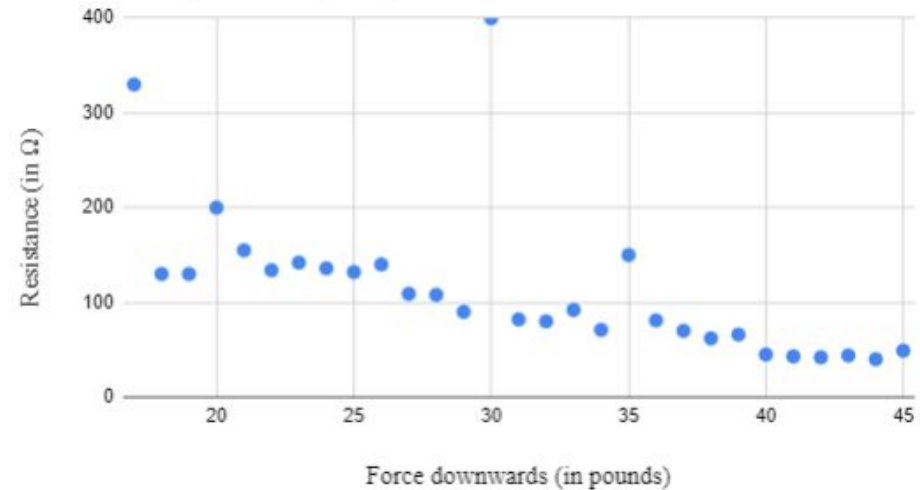
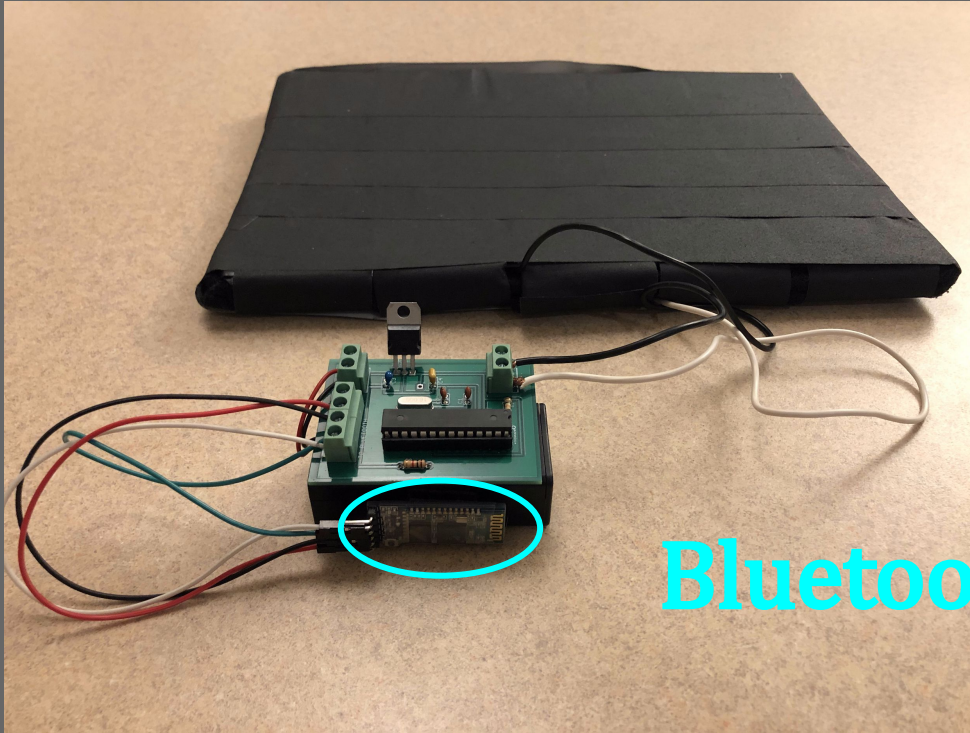


Table Network - Detecting Customers



Bluetooth

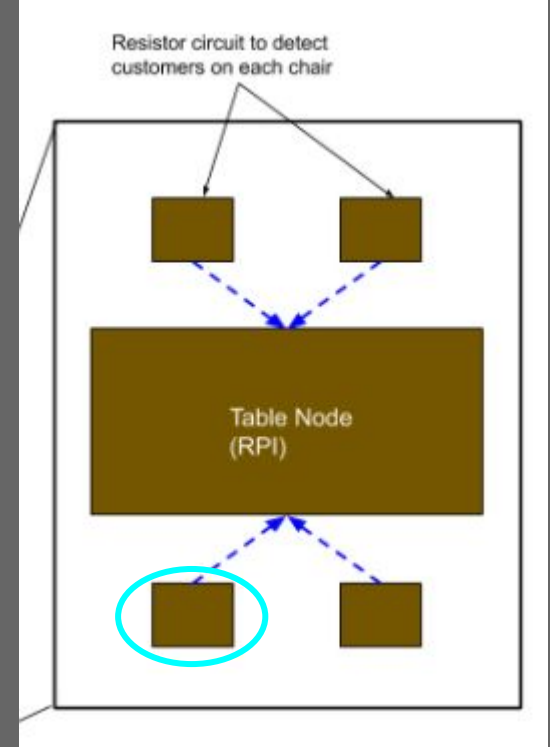


Table Network - Detecting Objects on the Tabletop

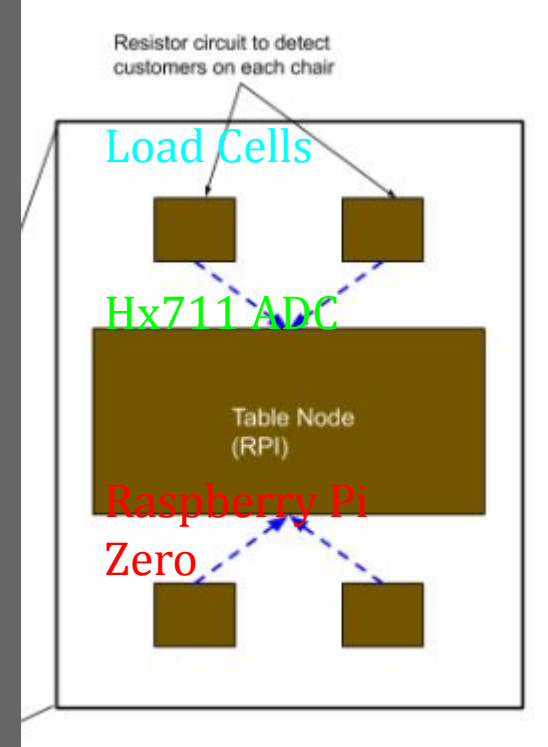
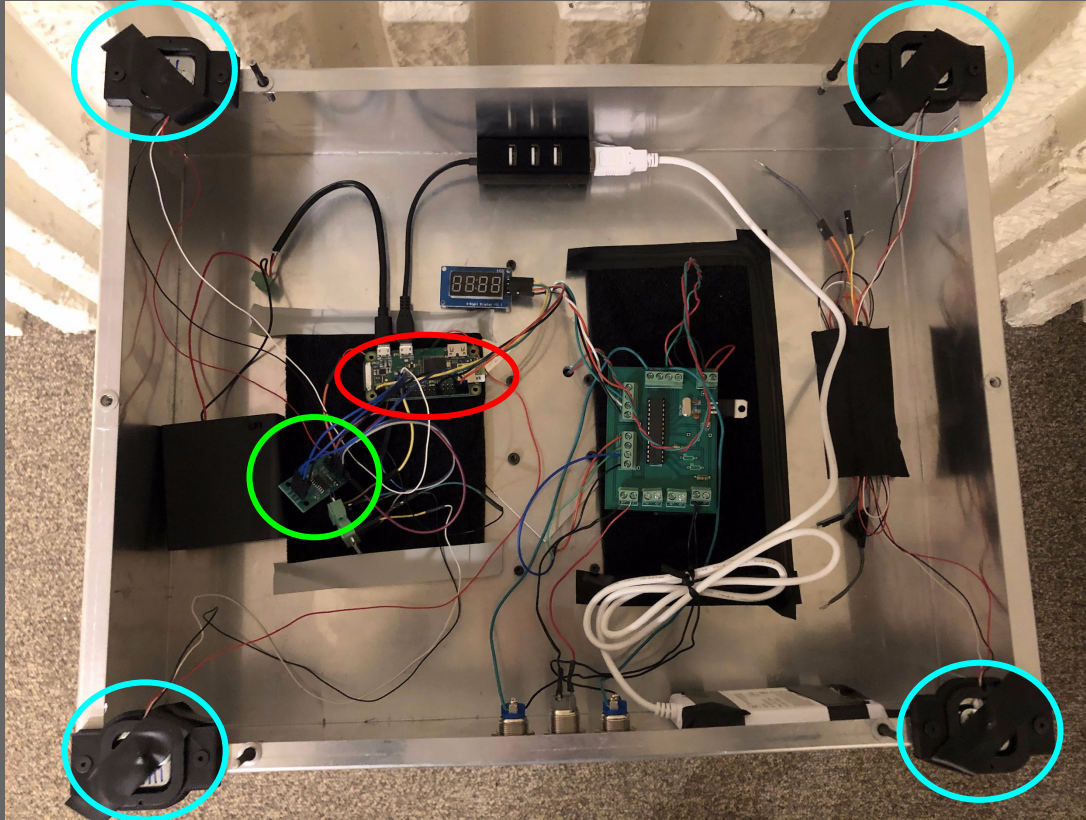
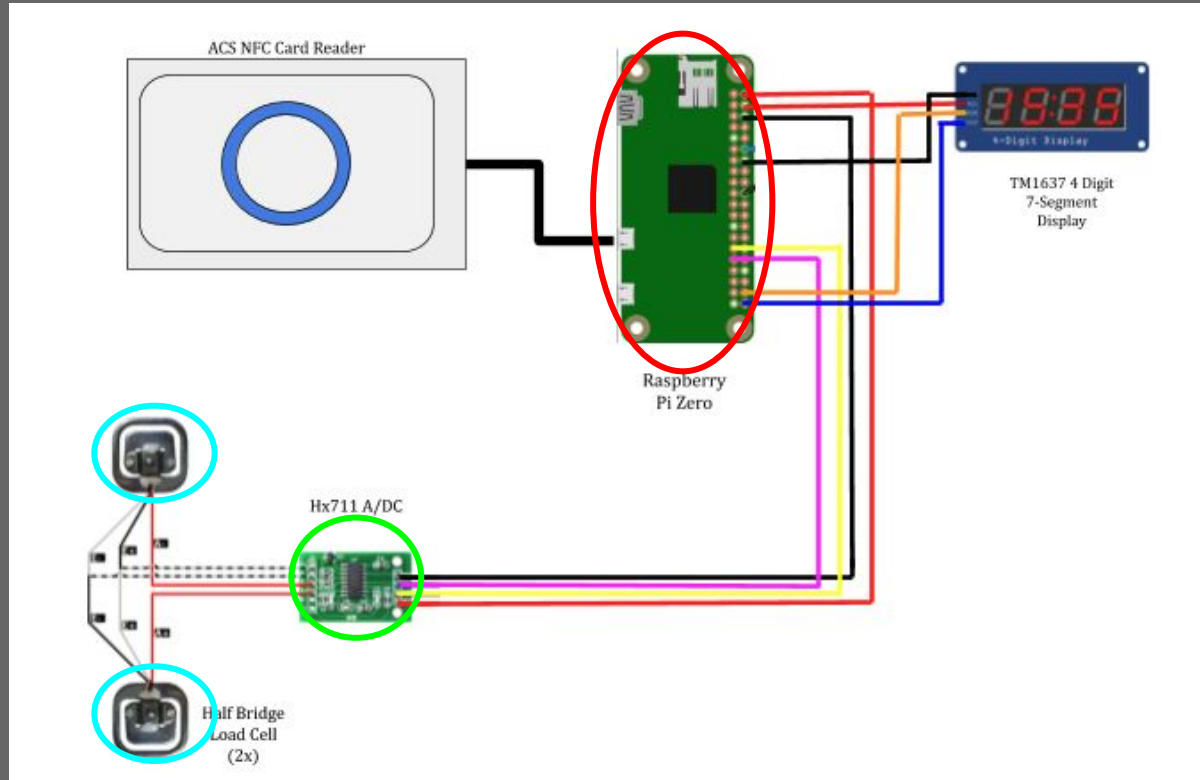


Table Network - Detecting Objects on the Tabletop



Load Cells

Hx711 ADC

Raspberry Pi
Zero

Table Network - Detecting Objects on the Tabletop

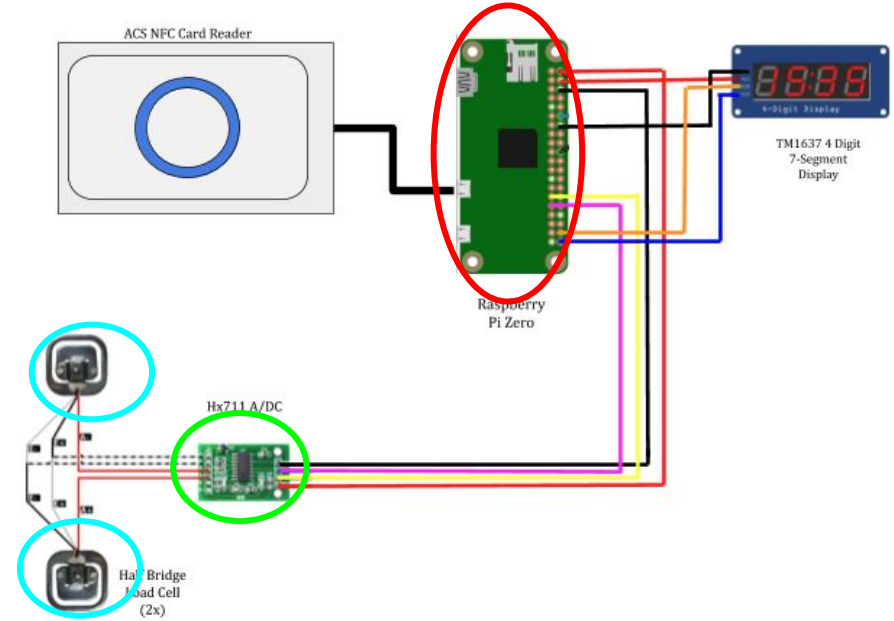
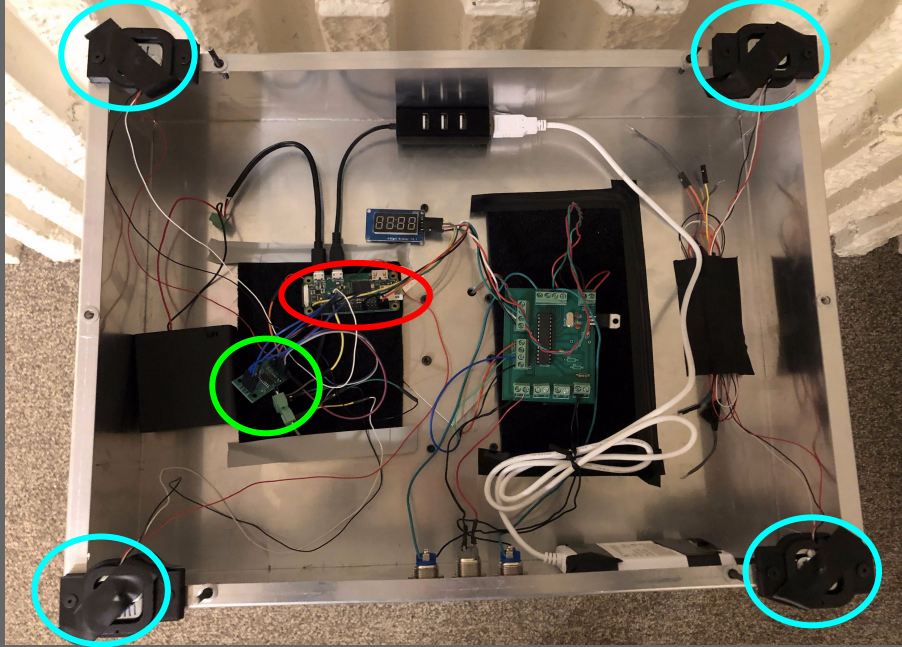


Table Network - Talking to the Dashboard

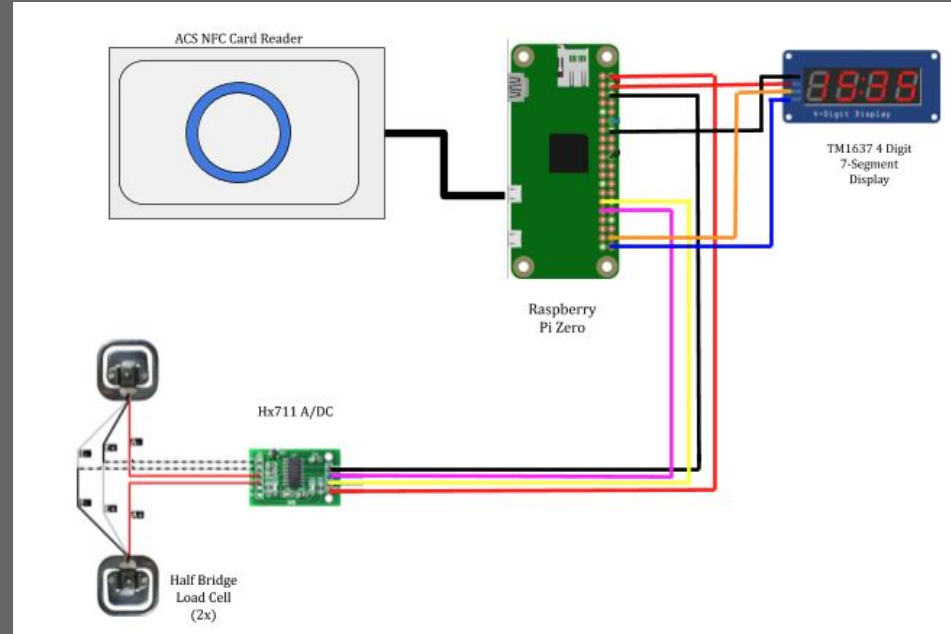
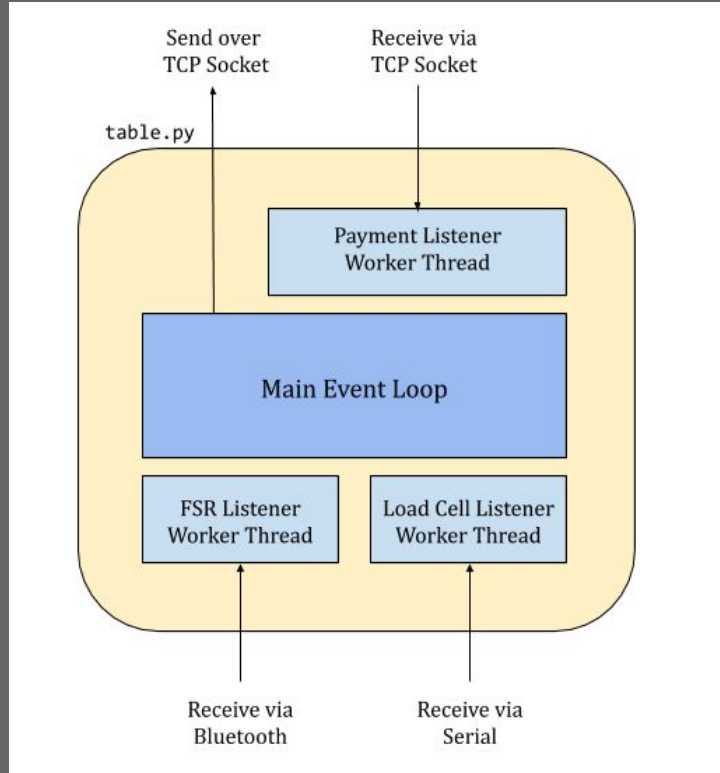
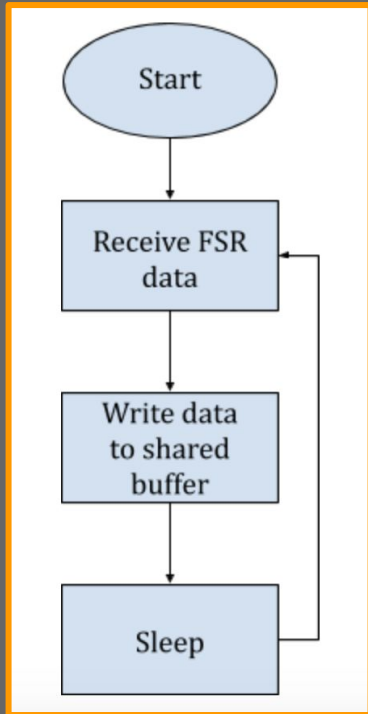


Table Network - Talking to the Dashboard

FSR Listener
Thread



Load Cell Listener
Thread

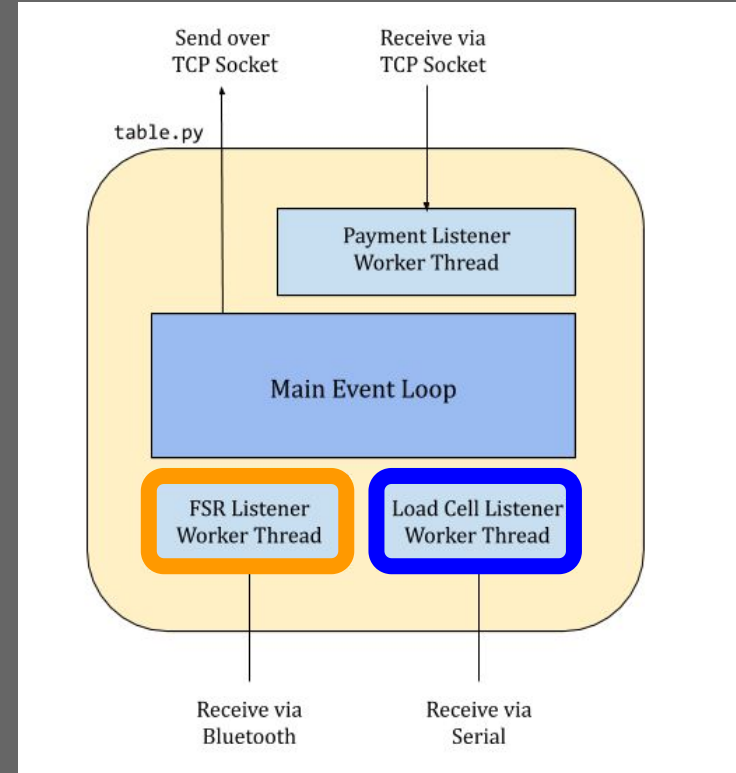
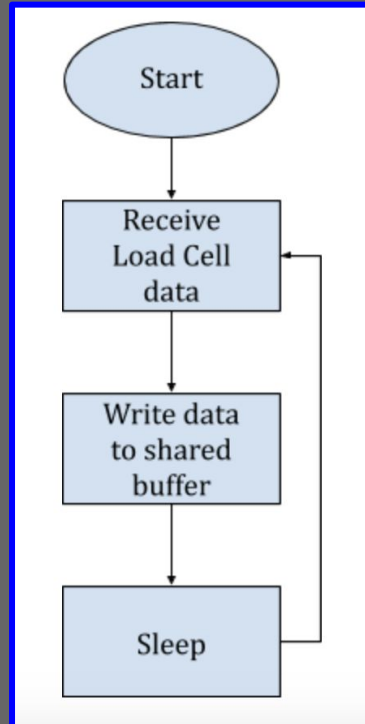


Table Network - Talking to the Dashboard

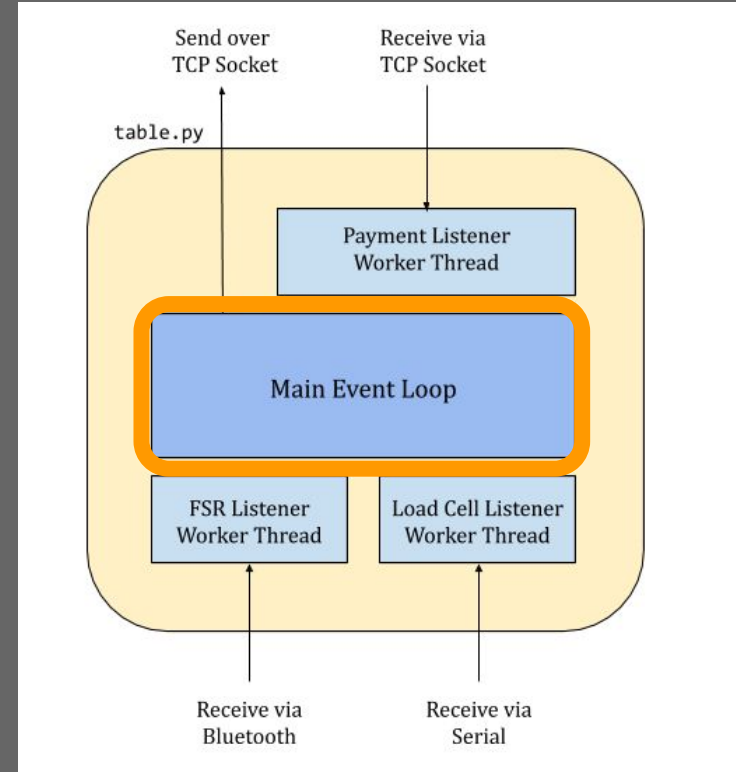
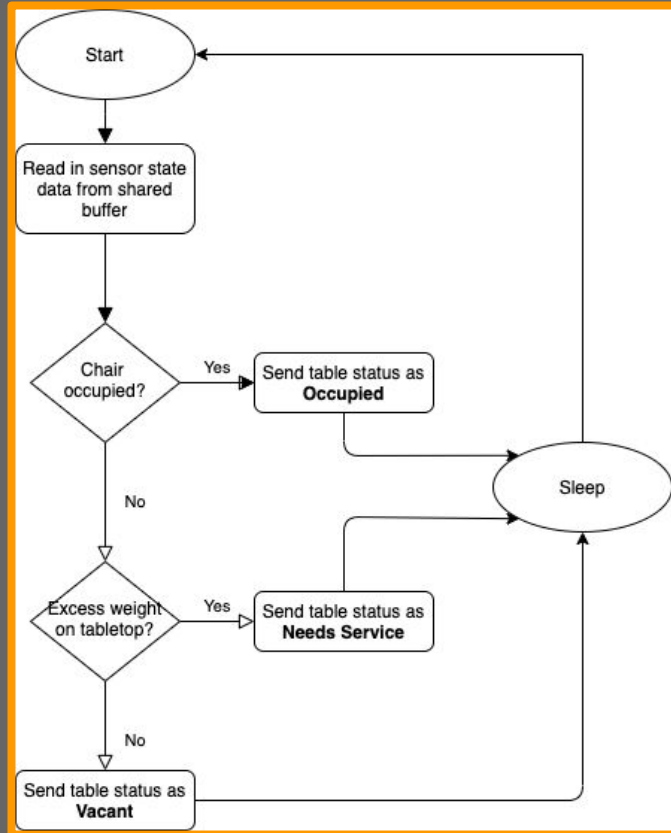


Table Network - Requirements & Verification (1 of 3)

Requirements:

Data from FSR sent via Bluetooth can be read at an approximate rate of one message every five minutes.

Verification:

1. Pair Bluetooth device with Raspberry Pi
2. Send and read in bytes across the devices.

Result:

Requirement met but modified:
Instead of reading once per five minutes we instead read once every 3 seconds.

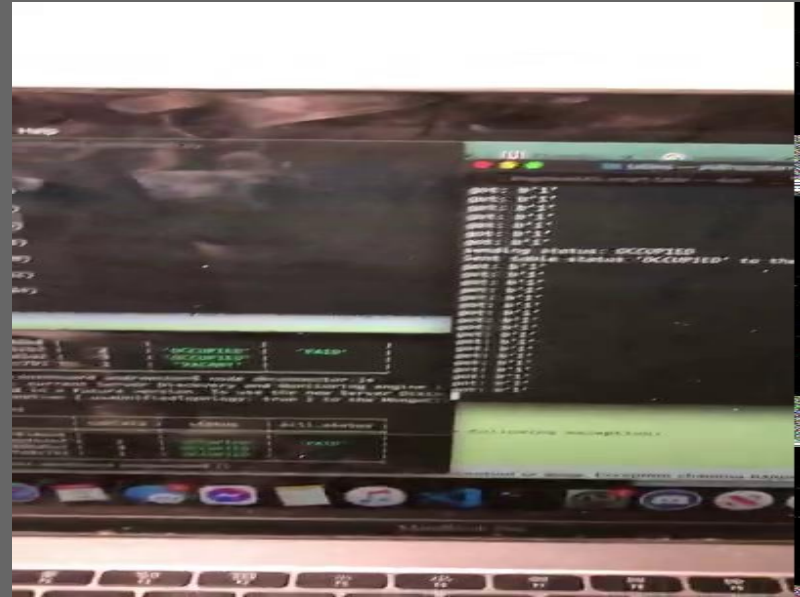


Table Network - Requirements & Verification (2 of 3)

Requirements:

Load Cell can detect when there is an excess amount of weight on the tabletop.

Verification:

1. Wire up the load cells to the ADC and Raspberry Pi
2. Run the Python software to read in values
3. View the output while placing various weights

Result:

Requirement met.



Table Network - Requirements & Verification (3 of 3)

Requirements:

All table nodes will be able to send information to a single central server node via sockets over a network.

Verification:

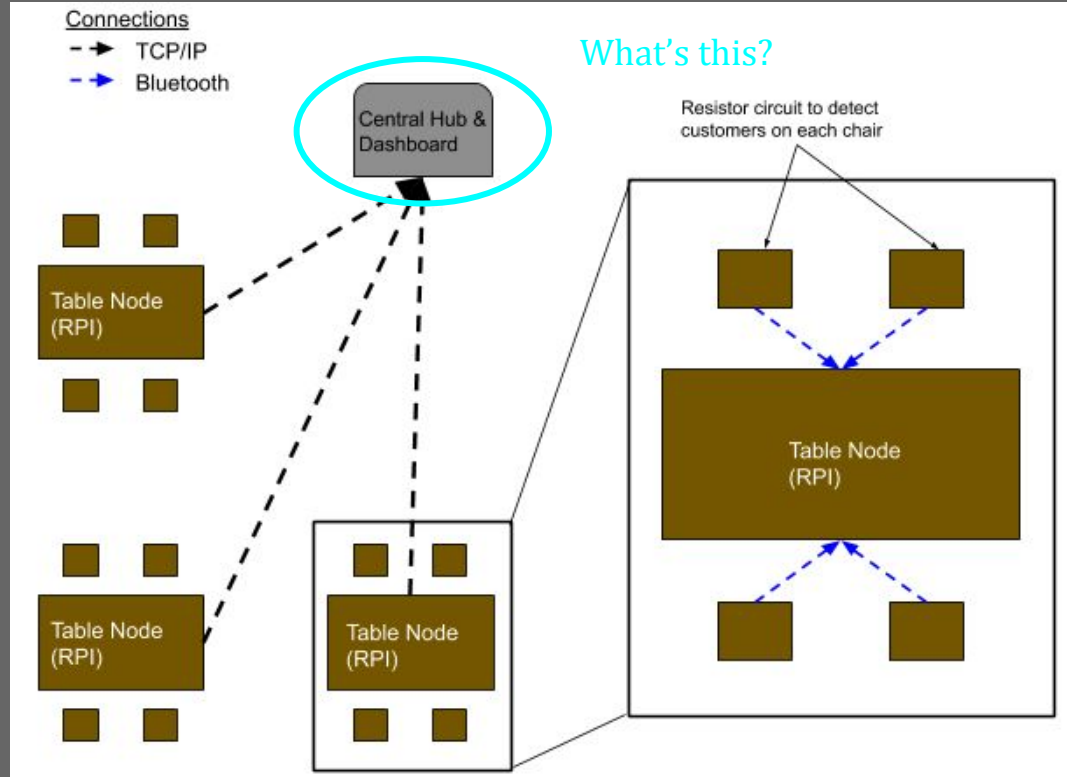
1. Connect devices to be on the same network
2. Ping one device from the other to verify a connection.
3. Run server and client software on each respective device and send data over a socket.

Result:

Requirement met.



Table Network - Talking to the Dashboard



Dashboard

Dashboard

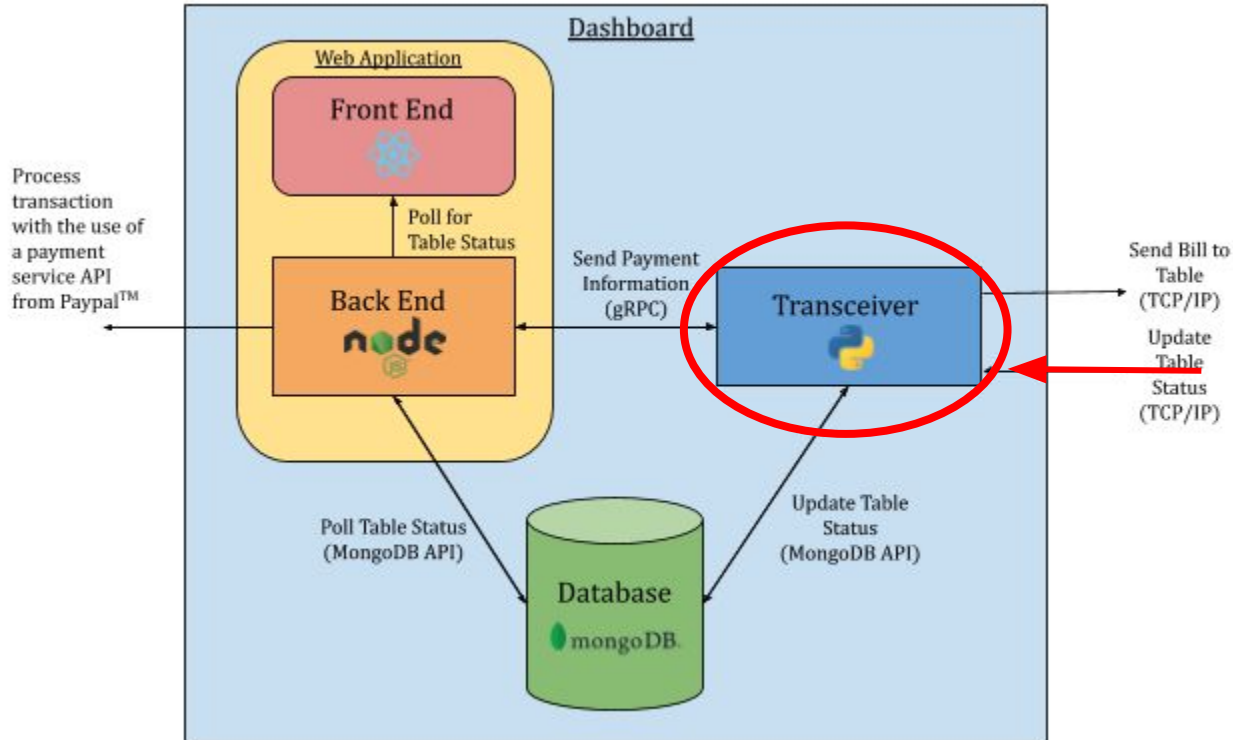
Talks to all the tables.

Displays the status of all tables in real time.

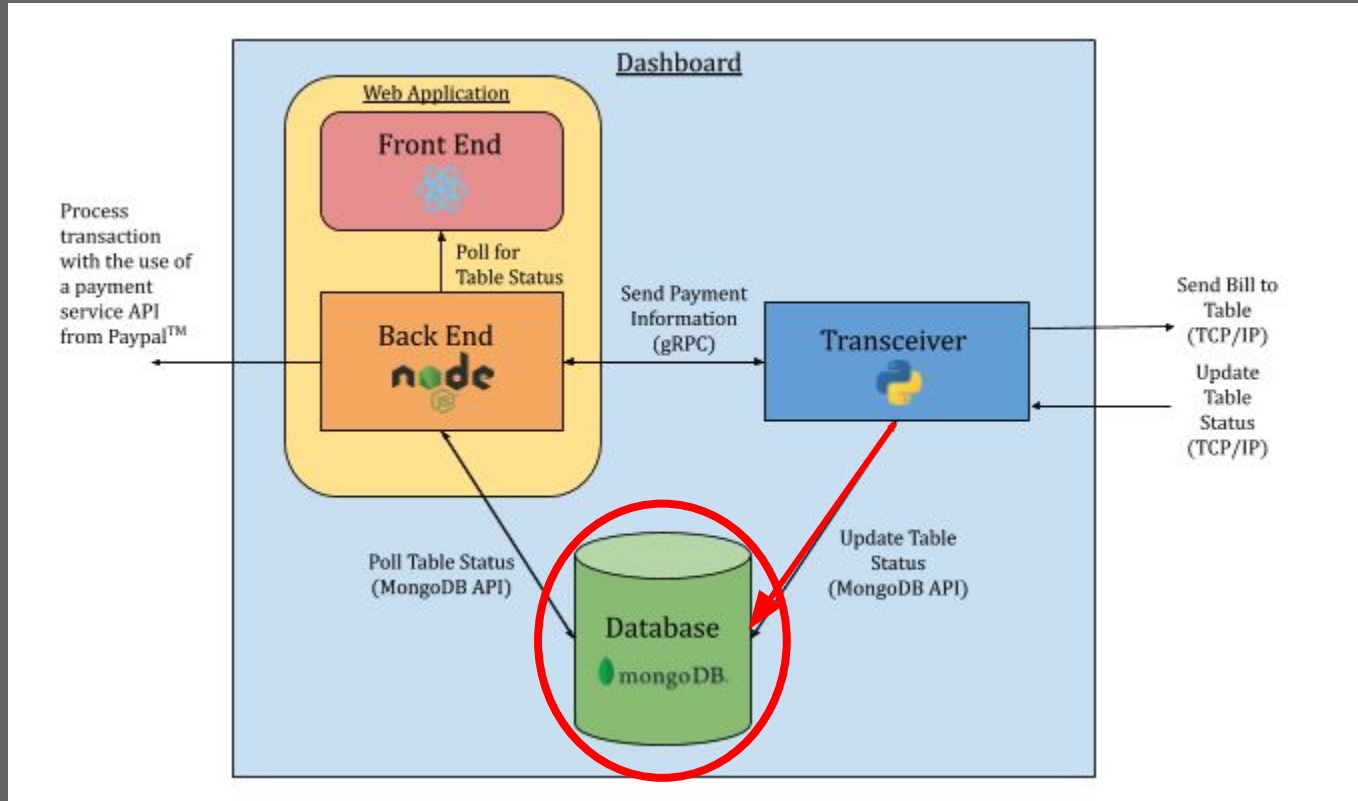
Issues bills.

Runs on your computer.

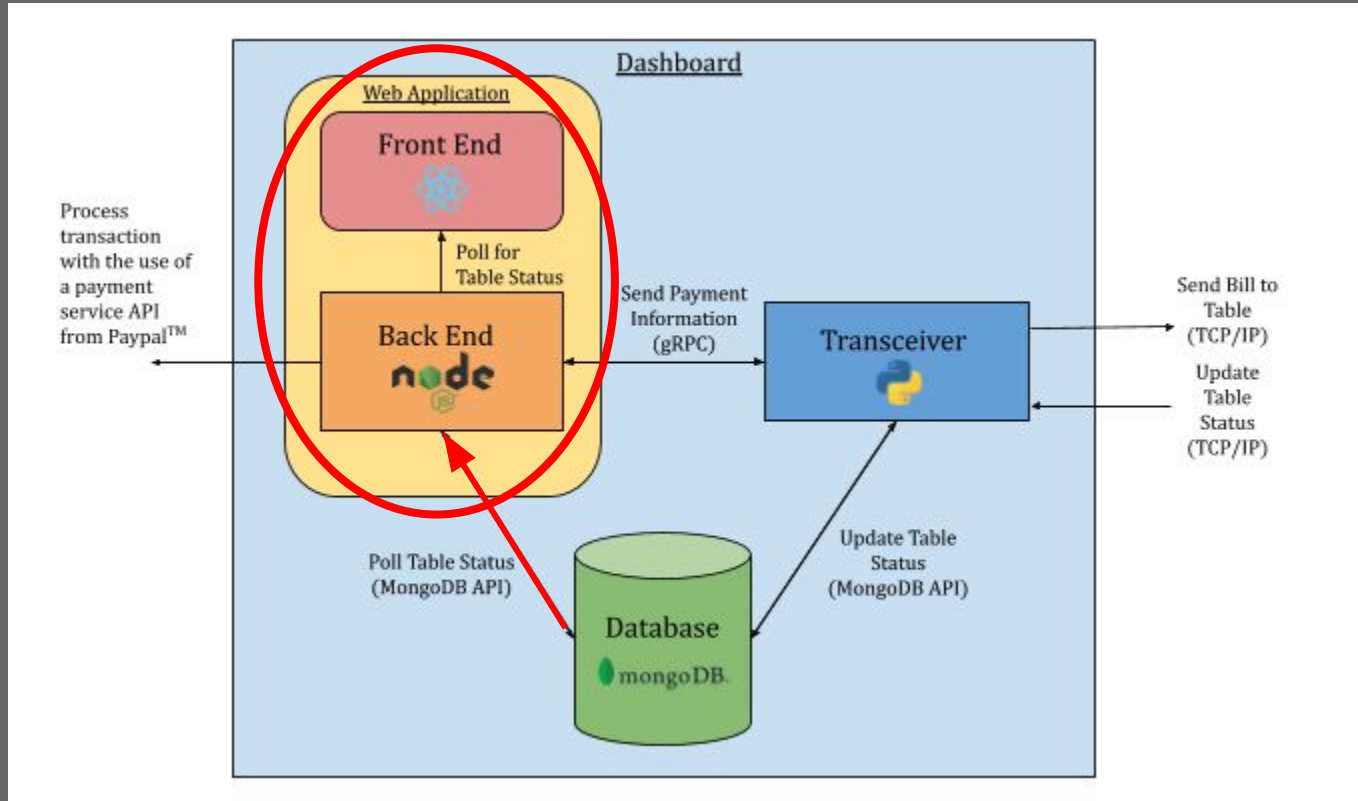
Dashboard Architecture



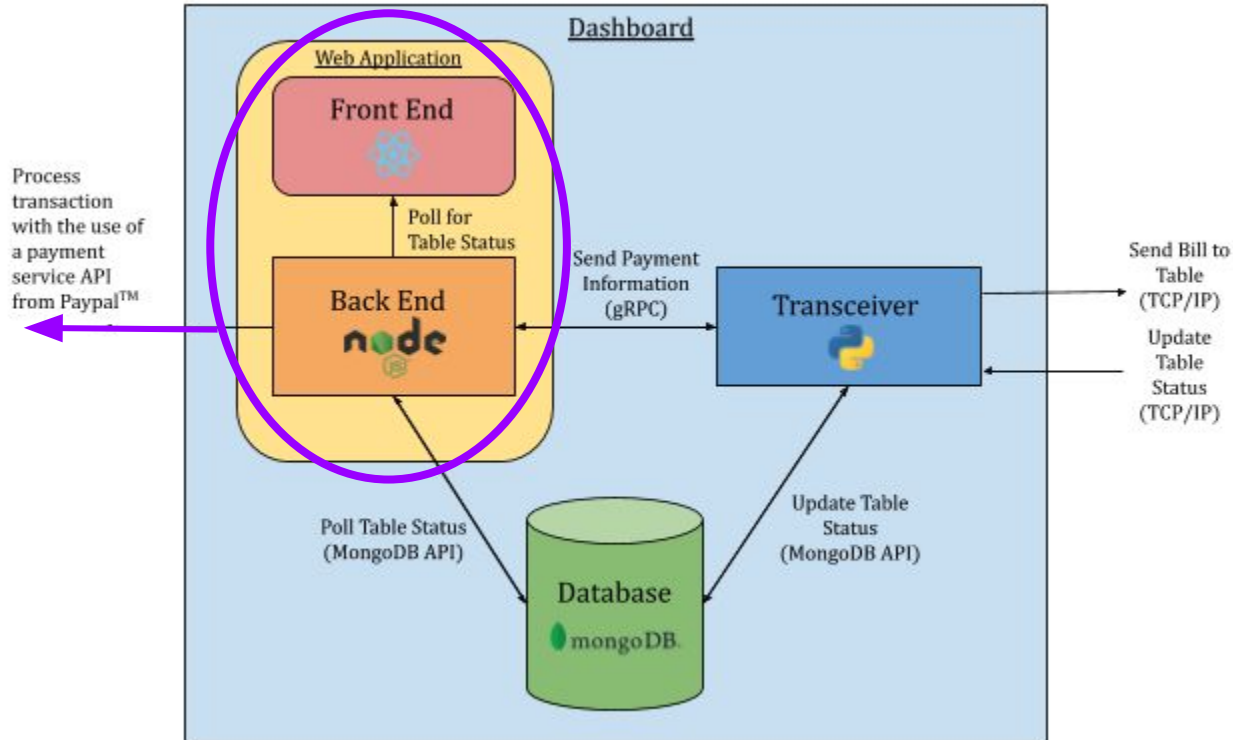
Dashboard Architecture



Dashboard Architecture

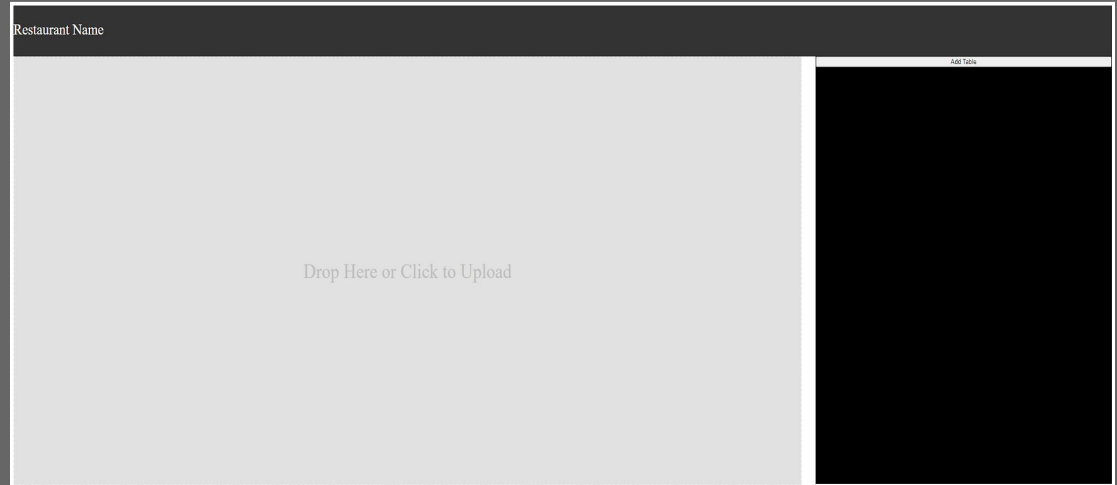


Dashboard Architecture



Web Application - Frontend

- Drag and drop or upload Floor Plan
- Click and drop tables onto Floor Plan
- Table changes color based on their status
- Send Bills to each Table



Web Application - Requirements & Verification (1 of 3)

Requirements:

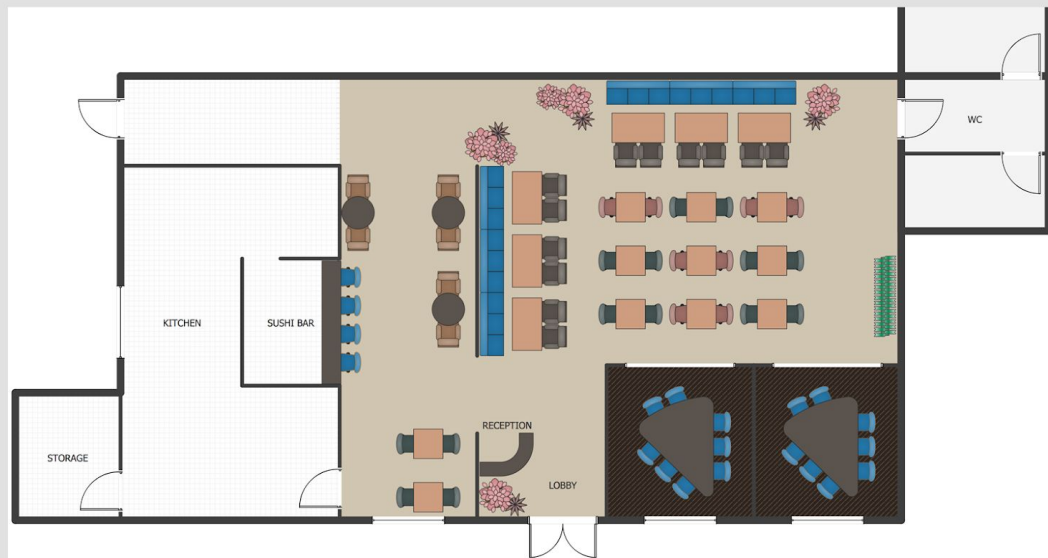
User will be able to upload a floor plan onto the dashboard

Verification:

Drag and drop or upload a floor plan in image format onto the dashboard

Web Application - Frontend

Restaurant Name



Add Table

Table 0

Web Application - Requirements & Verification (2 of 3)

Requirements:

User will be able to mark and drop a table onto the uploaded floor plan

Verification:

Click on the floor plan and drag a table block from the right side onto the floor plan and verify its on the floor plan

Web Application - Requirements & Verification (3 of 3)

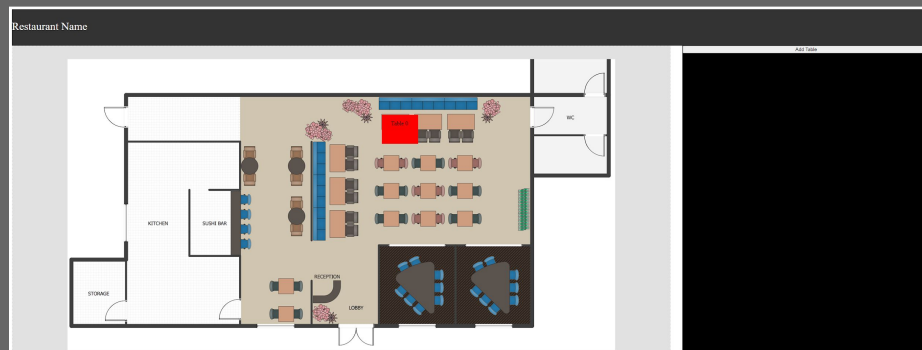
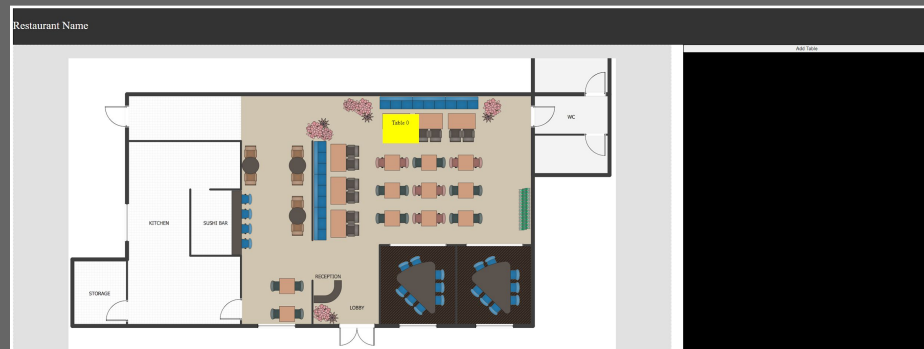
Requirements:

The dashboard will update the status represented by color of each table in realtime.

Verification:

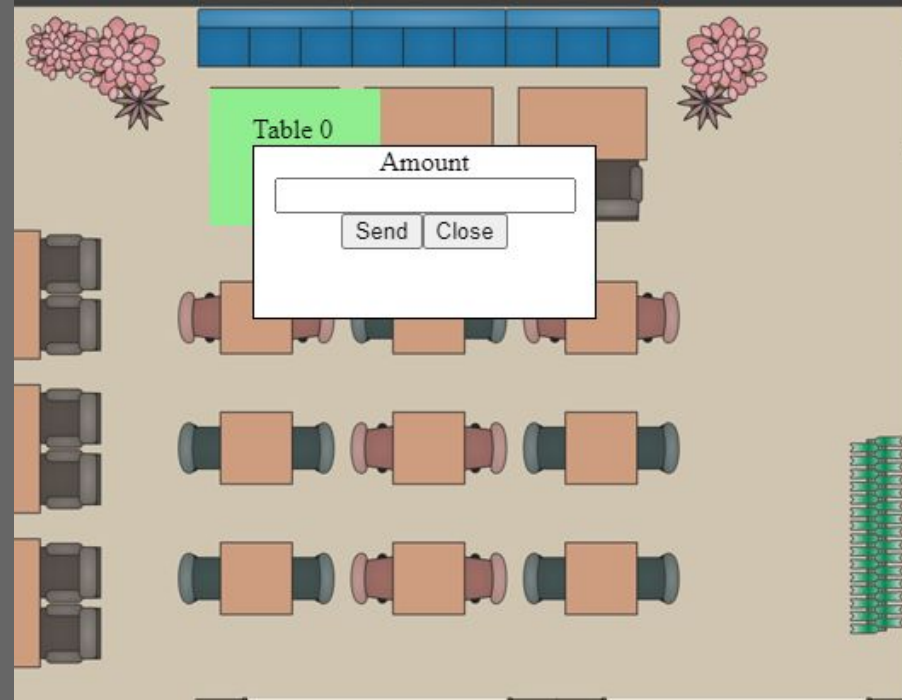
Change the status of the table by either pushing on the FSR or Table load cell to update the Database or manually updating the Database. Verify the color of the table changes according to the status.

Web Application - Frontend Cont.



Payment

- Send bill to table independent of each other.
- Payment process via Paypal
- Sends bill amount and payment status to table's hex display
- Waits for NFC contact



Payments Hardware - Requirements & Verification (1 of 2)

Requirements:

Seven-segment display will be able to output integer values.

Verification:

Wire up the seven-segment display and program the GPIOs to print something on it.



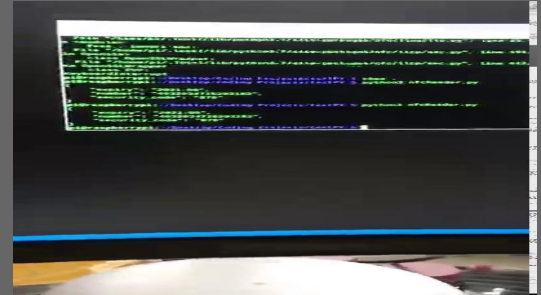
Payments Hardware - Requirements & Verification (2 of 2)

Requirements:

NFC Reader is able to wait for NFC cards and read the data in the card


Verification:


Connect the NFC Reader to a Raspberry Pi and run the reading program to print the data from the card when it is near the reader



Payment Cont.

John Doe's Test Store

Developer | Help |  John Doe

 Home Activity Pay & Get Paid Marketing For Growth Financing App Center

All

Search for transactions

Archive
Active

Transaction Type
All activity

Date
Past 30 days

Amount & Currency
All currencies

☐

Date

☐

6:28 AM

☐

Nov 18, 2020

☐

Nov 17, 2020

☐

Nov 17, 2020

☐

Nov 17, 2020

Payment from

8ML75777SF359080A@dcc2.paypal.com

Payment from

8ML75777SF359080A@dcc2.paypal.com

Payment from

8ML75777SF359080A@dcc2.paypal.com

Payment from

8ML75777SF359080A@dcc2.paypal.com

Payment from

8ML75777SF359080A@dcc2.paypal.com

Completed

Completed

Completed

Completed

Completed

\$12.34 USD

\$34.00 USD

\$1.23 USD

\$13.00 USD

\$2.50 USD

-\$0.66

-\$1.29

-\$0.34

-\$0.68

-\$0.37

\$11.68

\$32.71

\$0.89

\$12.32

\$2.13

\$5,428.99 USD

\$5,417.31 USD

\$5,384.60 USD

\$5,383.71 USD

\$5,371.39 USD

Print shipping label

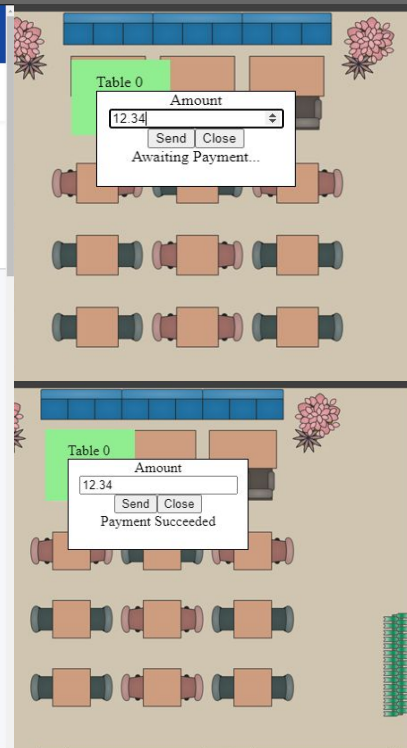
Print shipping label

Archive

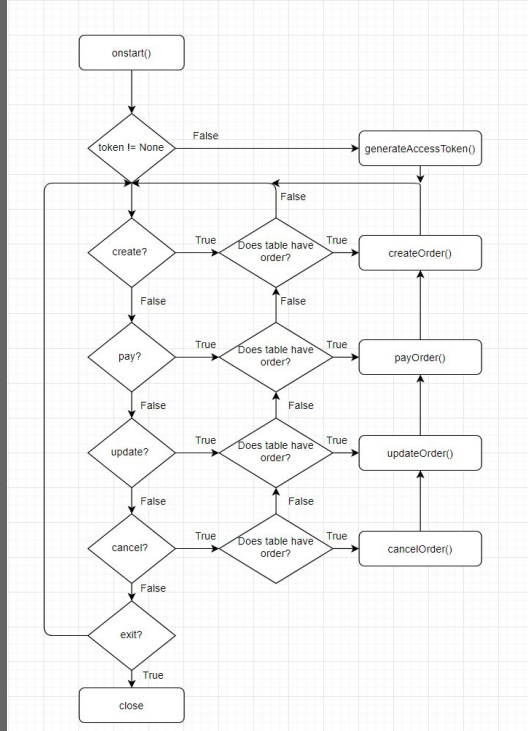
Print shipping label

Print shipping label

Download



Payment Flowchart and Paypal API

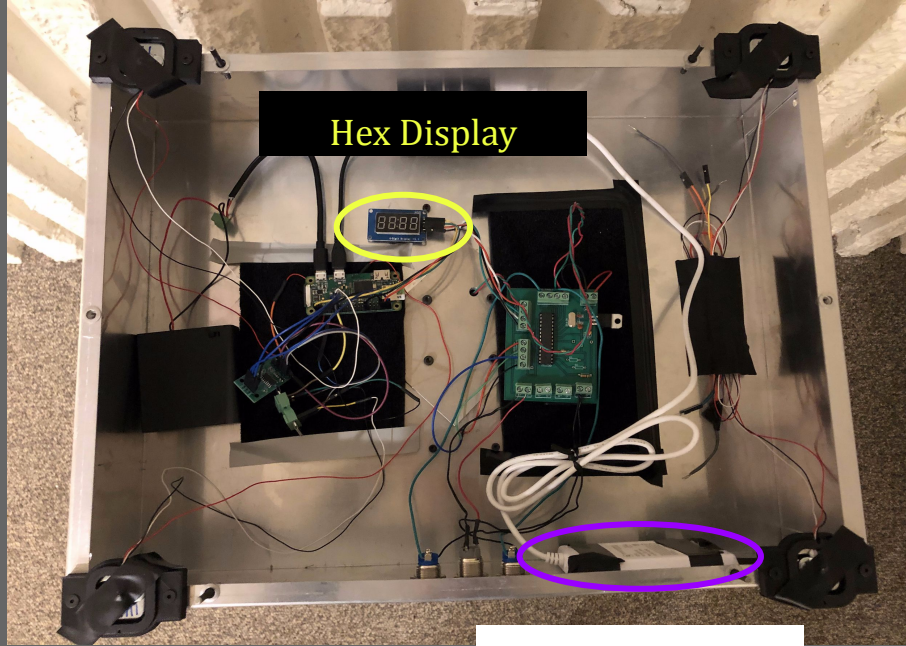


```
//input: string access token, promise order
//desc: request to pay for an Order given an order's id
//output: None
async function payOrder(access_token, order) {
  getPayment().then(data => {
    let body = {
      "payment_source": {
        "card": {
          "number": `${data.number}`,
          "expiry": `${data.expiry}`,
          "security_code": `${data.security_code}`
        }
      }
    }
  })

  let option = {
    uri: `https://${ENV.sandbox}${SCOPE.v2.checkout.createOrder}/${order.id}/capture`,
    method: 'POST',
    headers: {
      ContentType: "application/json",
      Authorization: `Bearer ${access_token}`
    },
    body: body,
    json: true
  }

  request.post(option).then((data) => {
    console.log(data)
  });
}
```


Payments Hardware



NFC Card Reader

Going Forward

- Explore different sensors for use with the tables in operation if desired
- Better integrate modules with dining room furniture
- Upgrade our payment system to have better NFC encryption

Ethics and Safety

Table could malfunction and injure the customer's legs

- IEEE Code of Ethics #9
- Manual height adjustment system
- Sensors use a median system to eliminate erroneous readings
- Movement is slow enough to notice and slow enough not to disrupt food on table

Sensitive data transfers between tables and dashboard

- IEEE Code of Ethics #1
- Kept on a private network
- In the future, can apply encryption such as AES, OTP, or 3DES
- In the future, a better NFC solution is needed for reading credit card information

Acknowledgements

Special thanks to:

Professor Jing Jiang

Our TA: Sophie Liu

Our sponsor: Neil Misak and his Restaurant Technology Associates

Jordan Spezia, Gregg Bennett, and the ECE Machine Shop Team