Digitizing the Restaurant Experience

Design Document
Team #36
Aman Kishore (amank2@illinois.edu)
Jun Woo Seo (jseo35@illinois.edu)
Patrick Moore (ptmoore2@illinois.edu)

UIUC ECE445

TA: Sophie Liu (yiqiaol3)

Table of Contents

I. Introduction ……..…….. 2
A. Background ...…..…….... 2
B. Overview ……..... 2
C. Visual Aid ……... 3
D. High-Level Requirements ……. 4
II. Design ……..…… 5
A. Block Diagram ……... 5
B. Physical Design …….. 6
C. Schematics …….. 7
D. Subsystems ……..... 8
E. Risk Analysis …… 12
F. Contingency Plan ……..……... 12
III. Cost and Schedule ……..…….. 13
A. Cost Analysis ……..…….. 13
B. Schedule ……..…….. 14
IV. Safety & Ethics ……..……... 15
V. References ……..……... 16

I. INTRODUCTION

A. Background

Throughout the Covid-19 pandemic, many businesses have been experiencing huge losses. Restaurants have been one of the businesses hit the hardest, and this is since most restaurants can only function when people can come in and sit down. Over 60%[1] restaurants that closed during the pandemic are closed for good; the restaurants that are still open may face a similar fate. The main problem that these restaurants are facing is that customers do not want to dine-in, as that would mean interacting with servers. Currently, servers have a lot of unnecessary contact with customers. The majority of restaurants have limited seating with most operating with takeout only and while others operate with minimal outdoor seating. As the weather gets colder, it will further reduce the amount of seating at restaurants. The huge problem that restaurants are facing is that most customers feel that it is too risky to dine inside.

Currently, the restaurant industry is unable to provide a safe environment indoors as there is too much interaction between servers and the customers. Many industry solutions offer contactless service, but no products exist to provide a contactless dine-in experience. For example, many third-party services, like Uber Eats and GrubHub, offer contactless delivery. The most similar service would be Rooam, but that is only a solution for contactless payment. The issue is, especially with fine dining, tables need to get filled. There are currently no effective solutions on the market that offer contactless dine-in. The lack of contactless dine-in options costs restaurants magnitudes in lost revenue and safety. Solving this challenge would allow restaurants to survive through the winter and generate enough revenue to sustain themselves through the pandemic. .

B. Overview

Our goal is to reduce the amount of time a server spends with each customer. We want our final solution to remove as much unnecessary contact between customers and servers as possible to create a safe dine-in experience. We have designed a system that will aid servers in determining if a table is clean and available for the next patron. Additionally, the system will limit the amount of time that each customer needs to interact with the server. This is achieved through an on table unit where the customers can order food and request drinks. This solution will limit the amount of interaction that a server has with each customer. The on-table units encompass an RFID scanner, a series of labeled buttons, and a WiFi module to communicate requests to the servers. A 3D-printable case will house all the components; this allows for low-cost production while maintaining a sleek look.

Our solution would be a cost-effective way to make indoor dining safer and limit any unwanted interactions. The system we will be building includes a table module, through which a customer can make requests, and an application where servers can see active requests. The on-table unit will also indicate when a staff member has cleaned the table. This module will communicate with a React app using a Flask API to populate a SQL database with all of these customers' requests. It would give smaller restaurants the ability to offer customers a novel way to order and keep everyone safer during the pandemic.

C. Visual Aid

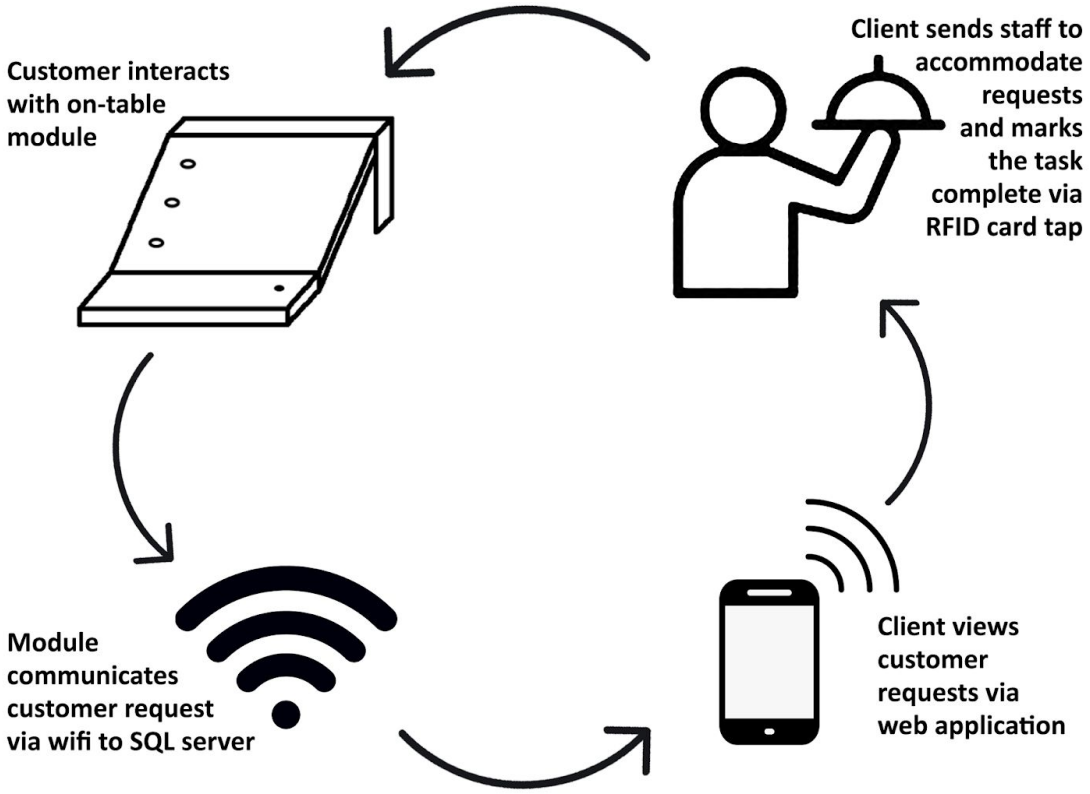

Fig 1. Visual Aid for this project

D. High-Level Requirements

- The first requirement is an in-unit RFID reader that will interact with a unique RFID Tag held by the staff with an accuracy of 90% +/- 10% when the server scans the tag. This will indicate if a table is sanitized and ready for the next customer or if the table is unavailable.
- The next requirement is building a screen that will allow the customer to communicate with the restaurant for 3 tasks which include requesting water, requesting food, and paying the bill.
- The final requirement is to design and produce a React Native application and a Flask API to store information from each module's screen and store up to 20 unique tables in a SQL database (one entry per table). The latency between the React Native application and the table should be under 30 seconds.

II. DESIGN

A. Block Diagram

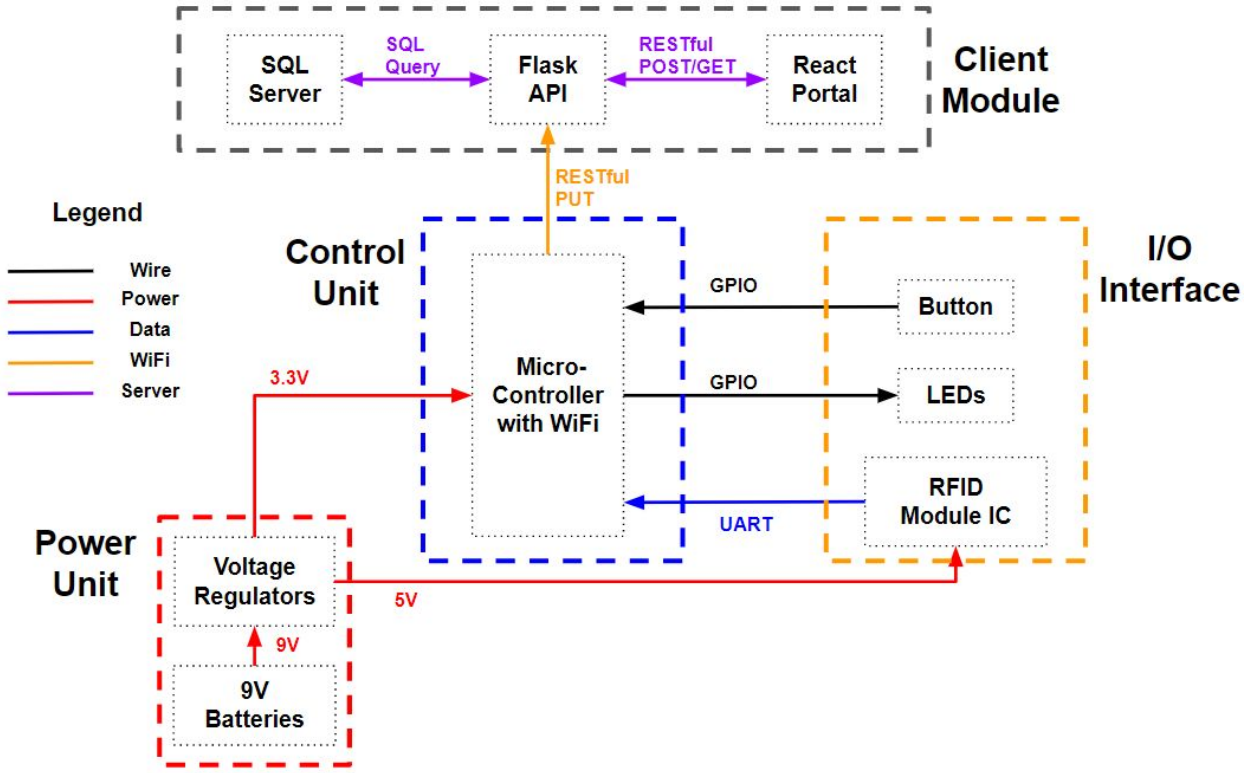

Fig 2. Block Diagram

B. Physical Design

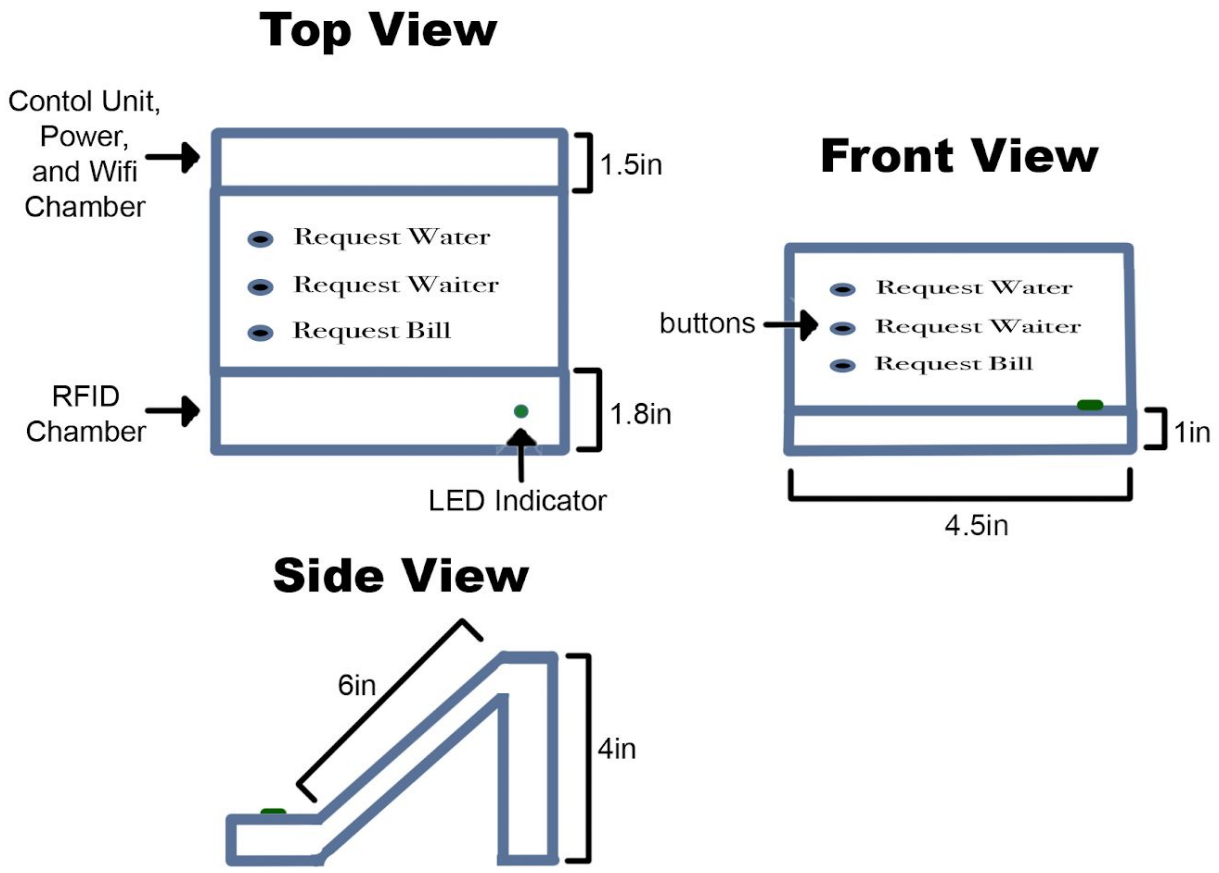

Fig 3. Physical Design

C. Schematics

1) RFID Reader schematic:

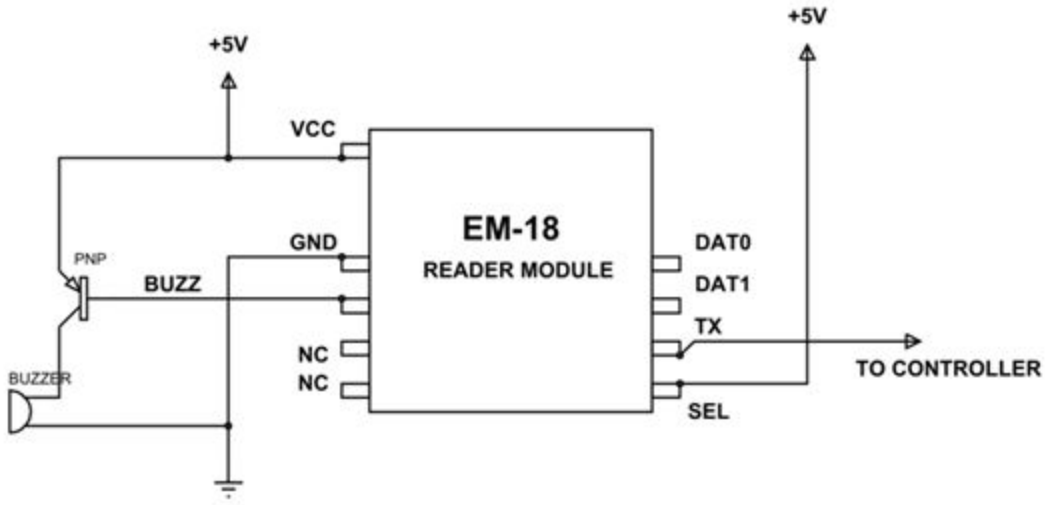

Fig 4. Schematic for RFID Reader[6]

2) Microcontroller with WiFi schematic:

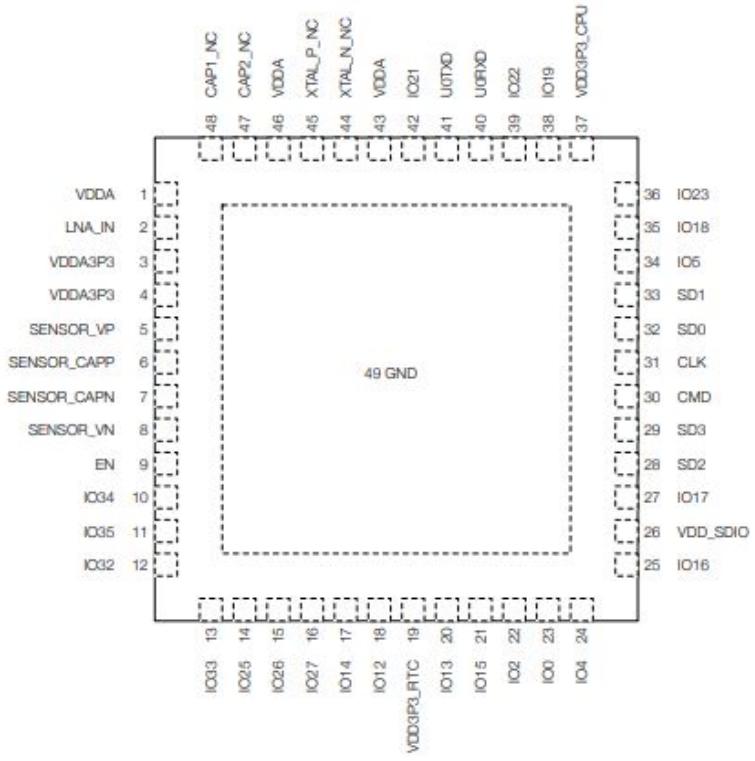

Fig 5. Schematic for Microcontroller with WiFi[7]

D. Subsystems

1) Power Module: A power supply is required to keep the RFID and microcontroller working. We will use an AC battery and two voltage regulators: a 3.3V regulator for the RFID module and a 5V regulator for the microcontroller.

a) Li-ion battery: The Li-ion battery will supply the power to the system. We will use a 9V Lithium battery with a capacity of 500mAh. Since the total current consumption for RFID and microcontroller is 130mA, the total capacity of the battery needs to be at least 1560mAH to run for 12 hours. Therefore, we need three 9V Lithium batteries.

Requirements	**Verification**
1. A lithium ion battery will need to output 9V +/- 5% 2. The battery can store at least 1560mAH of charge	1. Connect battery to a multimeter and ensure that the voltage is 9V +/- 5% 2. Discharge the battery by connecting it to a positive terminal for 12 hours at 130mA and use a multimeter to ensure that the voltage is still 9V +/- 5%

b) Voltage regulators: This integrated circuit supplies the required 9V to the system. This chip must be able to handle the peak input from the battery (9V) at the peak current draw (130mA). We will be using two voltage regulators. One voltage regulator that can provide 3.3V for the microcontroller and one voltage regulator that can provide 5V for the RFID reader. We are going to use LM1086 for the 3.3V voltage regulator and LM2940 for a 5V regulator.

Requirements	**Verification**
1. Provides 3.3V +/- 10% from a 9V source 2. Provides 5V +/- 10% from a 9V source	1. Connect a power supply to the voltage regulator with 9V. Measure the output voltage using an oscilloscope to see if it stays within 3.3V +/- 5% 2. Repeat test 1 for each of the required voltages in requirements 2

2) I/O Interface: Every unit will contain an RFID reader which will be an effective way to indicate when a table is ready or served. The RFID reader will act as a tool usable by the server to indicate when a customer's request has been completed as the servers will be equipped with RFID cards. Reading a proper RFID card will send a signal, indicating the presence of a card, which will clear the customer's request from the SQL server.

a) RFID Reader: The RFID reader, contained within the on-table unit, will be able to recognize the unique RFID card which is in the possession of wait-staff which, when read, will indicate the table has been helped/cleaned. When the reader recognizes a tag, it will trigger an LED indicator (which will be placed on the circuit) and send a signal to the SQL server to clear their request. When the table is occupied, there will be a queue for that table and once the table is cleaned and the RFID is scanned that queue will be cleared. The queue is created when the RFID receives a signal that the table is occupied. This information is passed to the SQL database and shows up on the web application. We are going to use EM18 for our RFID reader.

Requirements	**Verification**
1. Determine the reader should be able to receive signals in the 125 - 134 KHz range as that is the requirement for the low frequency tags we will be designing.[5]	1. Using a frequency generator determine if the reader is able to pick up frequencies in the range of 125 -134 KHz
2. The RFID reader should only work with the RFID tag used by the restaurants and no other RFID enabled electronics	2. Verify that only the RFID reader only accepts one RFID tag (with the correct information) by testing multiple RFID tags (eg. school id)
3. Once the RFID reader scans and RFID tag, the SQL database should be updated to toggle the table's state (available or occupied)	3. Verify that the SQL database is updated correctly and that the proper table state is stored.
4. The RFID reader needs 5V +/- 10% and the minimum current consumption is 10mA.	4. Connect the voltage input of RFID to a multimeter and ensure that the voltage is 5V +/- 5% and the current is at least 10mA

b) RFID Tag: Each staff member will have an RFID tag (card). The tag will act as an unobtrusive and swift way to clear notifications from the on-table unit. The RFID reader will only accept this card's unique signal.

Requirements	**Verification**
1. The RFID Tag should emit at a frequency of 125 - 134 KHz (Low Frequency Tag)[4]	1. Using a frequency meter ensure that the frequency that the RFID Tag emits is in the range of 125 - 134 KHz +/- 5%
2. RFID Tag should communicate to the SQL database and update the table's status every time it is scanned.	2. Verify that the database entry corresponding to the table is modified each time the RFID is swiped

3. POS/Customer Interface: The Customer Interface's primary use is to communicate a customer's needs with the client. This is accomplished through a series of labeled buttons which, when pressed, will cause the control unit to send a signal which populates a server with the appropriate data which is monitored by the client via a web-portal. This will be used by the customer to send requests to the server and all of this information will be sent through the backend to the Client Module.

a) Client Module: The Client Module includes the server which stores the data, and the customer is trying to communicate with the establishment and a web-portal to view such data. The restaurant will use the portal to see which tables are making requests and will have the ability to clear requests through the portal. The data will be stored in SQL tables and organized to be displayed through the web-portal. The data on the SQL tables will be managed by a flask-based backend which will interpret API calls from the table units to determine their status and update the database accordingly. The Client portal will be a simple javascript React portal. Both the React portal and on-table units communicate data via our API calls. Most of all software will be hosted by Google Cloud Services and Heroku. By using an ID and password unique to the restaurant, the staff will be able to see the status of tables through this portal. Ideally, in the future, this project will expand to having a customer-facing application that will allow diners to accomplish much of this project's functionality through a mobile application (out of the scope of this project).

Requirements	**Verification**
1. The web application properly fetches data from the SQL	1. Check SQL tables to ensure that the application shows ALL data from a single

tables 2. The web app properly refreshes to display new/current data within 15 seconds when updating SQL tables. 3. Ensure that the web app is responsive on desktop and mobile. 4. Be able to update status and tables via the web platform	table (and that the table is different for every account). 2. Add new data via API call and ensure that the new data is viewable via the platform within 15 seconds of updating the SQL tables. 3. Use Chrome's dev tools to test multiple devices. 4. Check the SQL tables to ensure the web platform has made the proper changes

4. Control Unit: The Control Unit is the bridge between the I/O Interface and Client Module. It will interpret signals from the RFID reader and buttons and send data to Flask API through the WiFi.

a) Microcontroller with WiFi: We are going to use ESP32-pico-d4 for our microcontroller. This microcontroller contains a WiFi integrated circuit. The WiFi IC allows for the microcontroller to send data to the Flask API hosted on the Heroku. There are a series of API commands in the flash storage of the microcontroller for certain conditions. For example, when the button is pressed, the microcontroller receives the data via GPIO. Then, data is sent back to LEDs via GPIO, one of the three LEDs turns on (green), and an API call is made.

Requirements	**Verification**
1. Ensure SPI flash holds all the proper API commands 2. Ensure all sensors produce signals as expected 3. Ensure circuit recognizes all signals 4. The microcontroller needs 3.3V +/- 10% and the average	1. Reading the contents of flash will prove this to be true or false 2. Use oscilloscope to verify the sensors are not flawed 3. Use meter to verify that our controller attempts to send data to the Flask 4. Connect the voltage input of RFID to a multimeter and ensure that the voltage is

operating current is 80mA and the minimum current is 50mA	3.3V +/- 5%, the average current is 80mA, and the minimum current is 50mA

E. Risk Analysis

Getting this module to properly communicate with the restaurant through the screen heavily depends on the WiFi module. If the Wifi module does not have enough range, we will do the following modifications. One option is to reduce the range with which the WiFi module will have to communicate. The other option is that we will purchase or build a signal extender in order to improve the range. Additionally, if the WiFi module is unable to send any signals then the module will not be able to send any information. This would significantly impact our project and, as a workaround, we could directly pass the data from the module but this would take a lot of reworking of the components of our project.

There is also a risk that this RFID reader will be unable to communicate with the RFID tag that we create. If the RFID is unable to communicate with the RFID tag then one of the main functionalities of our project will not be able to work. In that case, we may have to simplify our design and purchase an NFC tag. Additionally, if the RFID sensor is unable to properly pick up signals then we may have to redesign the solution by implementing a High-Frequency Tag which is usually more reliable but more complex.

F. Contingency Plan

In case the school shuts down, we will not have to alter the power module of our design as that is fairly straightforward. In case the school shuts down, the client module will be unaffected. However, the wifi module may be difficult to complete. In this case, we will use this as a proof of concept, or we may try to purchase a WiFi module in order to have a completed design. For the RFID Module, we will have to purchase an RFID and tag in order to have a working design. If this is not possible, we will have to work with the parts that we have as a proof of concept.

III. COST AND SCHEDULE

A. Cost Analysis

1) Labor: On average an ECE Major from UIUC made $88,000 as a starting salary. This roughly translates to $42.31 per hour.

The main aspects of the project are as follows:
SQL Database Creation: 8 hours (Aman)
Flask API: 8 hours (Patrick)
Making the React Application: 25+ (approximately 40) hours (Aman & Patrick)
Designing the PCB: 10 hours (Junwoo & Aman)
Building the RFID: 5 hours (Junwoo)
Building the WiFi Module: 5 hours (Junwoo)
Wiring the Circuit: 4 hours (Junwoo & Patrick)
3D Design/Print Shell: 3 hours (Patrick)

The total amount of labor will be around 68 - 83 hours which (using the equation ($/hour) x 2.5 x hours to complete = TOTAL) roughly equates to about $7,192.70 - $8,779.33 cost of labor for creating this project.

2) Parts:

Description	Manufacturer	Part #	Quantity	Cost
LED	Rohm	SLR-56VR3F	3	$0.49
Button	WURTH ELEKTRONIK	430186070716	3	$0.50
RFID Module	RNDMFG	EM-18	1	$5.50
RFID Card	HID	208	1	$2.49
Microcontroller with WiFi	Espressif Systems	ESP32-pico-d4	1	$4.95
Voltage Regulator for 3.3V	Texas Instruments	LM1086IT-3.3/NOPB	1	$4.86
Voltage Regulator for 5V	Texas Instruments	LM2940CT-5.0/NOPB	1	$1.53

3D Printer Filament (for Printed Casing)	Snapmaker	34011	1	$8.00
9V Battery	Duracell	MN1604B1Z	2	$5.44

NOTE: The software will ultimately be hosted on GCS and Heroku which will incur a recurring charge. As it is far too early to estimate the traffic and size of the software components, we have neglected to include the hosting service as part of the initial creation costs.

B. Schedule

Week	Junwoo	Patrick	Aman
9/27	Design Review Order Parts	Design Review Order Parts	Design Review Order Parts
10/4	Design PCB, RFID, WiFi Order PCB	Work on React App	Design PCB Order PCB
10/11	Design PCB, RFID, WiFi	Work on Flask API Work on React App	Work on database
10/18	Implement the RFID, WiFi module	Work on React App Work on RFID and Wifi	Work on database Work on React App
10/25	Wire up the circuit	Wire up the circuit Work on React App	Connect API to database Work on React app
11/1	Combine Software & Hardware	Combine Software & Hardware	Combine Software & Hardware
11/8	Mock Demo	Mock Demo	Mock Demo
11/15	Final Demo	Final Demo	Final Demo
11/30	Final Paper/Presentation	Final Paper/Presentation	Final Paper/Presentation

IV. SAFETY & ETHICS

There are several safety hazards that we will need to handle when it comes to this project. In accordance with the IEEE Code of Ethics[2] I.1: "to hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices." The first safety hazard that we face is that all of the power needs to be properly grounded. If there is an exposed wire there is a chance of electrical. Additionally, we will need to make sure that our project has some level of waterproofing. If a customer accidentally spills water on our system then it could cause some electrical issues not to mention that it will be extremely unsafe for the consumer. Some safety considerations when designing this project are as follows. We need to ensure that when we are constructing each module that we are cautious of common electrical standards.

We will adhere to the IEEE Code of Ethics[2] I.1: "to hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices." In order to ensure that our project is ethically sound we will need to ensure that all of the user data that is inputted in our POS will be encrypted and not stored locally. Keeping consumer safety is paramount to safety in this digital age when personal information can be stolen and used for gain. We need to ensure that the technology is understandable and discloses exactly what information it needs from the user. Ethically it is important that while we design this system we ensure that how it works is easy to understand by both the customer and the server. That entails having clear descriptions as to what information the POS system requires and what information will be stored. This includes any payment information especially if the customer uses their credit card to pay. This information must be kept secure and confidential in order for our design to fully adhere to the IEEE code of ethics.

As we work with our design, we will adhere to the IEEE ethic code I.5[2], “to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, to be honest.” We will ensure this by taking all feedback constructively and ensure that the work we are achieving is up to the highest standards.

V. REFERENCES

[1] Croft, Jay. “Yikes! Yelp Says 60% of Restaurant Covid-19 Closures Are Permanent.” *CNN*, Cable News Network, 25 July 2020, Available: www.cnn.com/2020/07/25/business/restaurants-reopen-coronavirus-shutdown-trnd/index.html.

[2] “IEEE Code of Ethics.” *IEEE*, 1974, www.ieee.org/about/corporate/governance/p7-8.html.

[3] “LM1086 1.5-A Low Dropout Positive Voltage Regulators datasheet” Available: https://www.ti.com/lit/ds/symlink/lm1086.pdf?ts=1601607738923&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FLM1086

[4] Karygiannis, Tom, et al. *Guidelines for Securing Radio Frequency Identification (RFID) Systems*. Recommendations of the National Institute of Standards and Technology.

[5] “How to Select a Correct Tag – Frequency.” *RFID4U*, 2020, Available: rfid4u.com/rfid-frequency/.

[6] “EM18 RFID Reader Datasheet.pdf.” *components.com* Available: https://components101.com/sites/default/files/component_datasheet/EM18%20RFID%20Reader%20Datasheet.pdf

[7] “esp32-pico-d4_datasheet_en.pdf.” *espressif.com* Available: https://www.espressif.com/sites/default/files/documentation/esp32-pico-d4_datasheet_en.pdf