Smart Restaurants

Project Report

Submitted in fulfillment of the requirements of

ECE 18-748

By

Team - 3

Lavina Chandwani
lchandwa@andrew.cmu.edu

Shubhankar Pawade
spawade@andrew.cmu.edu

Under the supervision of:

Prof. Ragunathan Rajkumar

&

Shunsuke Aoki

May 2019

Abstract

In today’s digital world, we find ourselves getting our jobs done in the blink of an eye. Conventional strategies are long gone and everything and anything can be automated. Smart Restaurants is a concept where a traditional restaurant system working is replaced by interactive user interfaces. This project is an attempt to develop a prototype of such a system where today’s tech savvy customers get gadgets to serve their orders instead of waiters. From light settings on the table to paying for the order, everything is controlled and accessible to the customer.

We have implemented lighting system capable of turning itself on/off by monitoring the output of Passive Infrared sensor which detects the presence of customer near the table. Along with this, the customer can control the light color by choosing convenient light settings. There is also a temperature control system that utilizes DHT22 sensor to current temperature value around the table and the customer can feed in a value of his interest to switch on the fan/heater. Next, we have built an online ordering app using Android Studio to remove the conventional way of taking orders through pen and paper. To record customer’s order requests, we have used Firebase (real-time database). To automate payments, we used Smart Card payments. A Smart Card is an initiative of our restaurant to make our customers go cashless. It has a RFID tag and can be swiped on the centralized card reader. Payments are wirelessly recorded on the database. All these user-friendly systems are to make sure our customers put in least efforts to make interaction with the waiters. This improves efficiency and accuracy for restaurants by saving time and reduces human errors.

Contents

1. Motivation

When we think of a restaurant, we imagine not only having a great food but also having a great time, enjoying the services. Now we can’t really ensure the food quality in our prototype, but we wish to provide a unique dining experience to our customers with the support of technology. This is our primary motivation for doing this project. In its 2017 industry report a private company called TOAST surveyed customers of around 900 restaurants in USA. They stated that 73% of the restaurant-goers think technology improves their dining experience and not only that, 95% restaurateurs think that technology makes their restaurant management more efficient. It is a win-win situation for both the staff and the customer. Equipping our restaurant with the latest digital age features makes it attractive and appealing to an average customer and may help increase revenue in long term considering the food is just awesome.

Another thought that crossed our mind was automation to reduce human dependency. Now every customer likes to feel welcomed in a restaurant. Considering involvement of human emotions, the restaurant service plays an immense role in its success. Eliminating staff by technology may seem a little infeasible in situations where customers love to interact with the bartender while ordering drinks or when they want to customize food items according to their preferences. This is when we came up the idea of providing a virtual connection between the customer and staff through our Android app. Now when one knows to handle everything from the app so why interact. This eliminates the need waiters to wait on the customers & take care of their every order. Also, sometimes miscommunication between the customers and the restaurant staff may lead to incorrect orders and unnecessary delays. Our app deals with this crisis since every order request is stored on a real-time database. This will be convenient for customers and will make the restaurant management system very efficient. Along with that, all payments are also recorded by the database.

The things that we could imagine in our smart dining area is that you do not have call the waiter to assign you a different table if the ac near your table is freezing you, or that if the lighting is too bright you can change its color by one switch. You will not have to wait for the staff to come over to your table again if you forgot to order your favorite drink the first time. You can customize the food to suit your tastes and that too by not telling the waiter again and again what you want. The restaurant staff will never mess up your order because everything is recorded on the restaurant’s database. You do not have to worry about carrying cash every time you dine out. You can enjoy your meal and you can have a great time playing with technology. There are a lot of things involved in making a restaurant success, and we wish to achieve all this in our project.

2. Project Goals & Requirements

Our primary goal of this project is to build a working prototype of a “Smart Restaurant” that focuses on the following features:

- Smart Lighting
- Smart Temperature Control
- Online App-based Ordering
- Automated Delivery
- Smart Card Payments

Following is the list of components required in this project, classified by the feature name. We have also mentioned the online links for each component here.

Feature	Component	Link
Smart Lighting	PIR sensor HC-SR501	https://www.amazon.com/Adafruit-LK-918O-SANV-FBACA-PIR-Motion-Sensor/dp/B00JOZTAC6
	RGB LED	https://www.jameco.com/z/RGB-LED-CC-5-WC-4-Pin-5mm-Water-Clear-RGB-LED-Common-Cathode_2228957.html?CID=GOOG&gclid=CjwKCAiAnsnjBRB6EiwATkM1XpdJc8vgavfjM3fCfGYN90V1NyiQOTlKgMLVFN78rSgmbADpqzetChoCaH0QAvD_BwE
	SPDT switches	https://www.amazon.com/LIYUDL-Position-Vertical-Switch-SS12D00G3/dp/B075TYP6C5/ref=sr_1_15?brr=1&qid=1557253415&rd=1&s=industrial&sr=1-15
Smart Temperature Control	DC fan (12V, 0.08A)	https://www.amazon.com/MakerFocus-Printer-40X40X10mm-Appliances-Replacement/dp/B07CH6YC32/ref=sr_1_12_sspa?ie=UTF8&qid=1551070047&sr=8-12-spons&keywords=silent+dc+fan&psc=1
	Electric Heating Pad (9V)	https://www.adafruit.com/product/1481
	DHT22 temperature sensor (-40 °C – 125 °C)	https://www.adafruit.com/product/385
	3x4 keypad	https://www.adafruit.com/product/419
	16x2 I2C LCD screen	https://www.amazon.com/JANSANE-Arduino-Display-Interface-Raspberry/dp/B07D83DY17/ref=sr_1_3?keywords=JANSANE&qid=1557253456&s=gateway&sr=8-3
	9V batteries	https://www.amazon.com/AmazonBasics-Everyday-Alkaline-Batteries-8-Pack/dp/B00MH4QM1S?ref_=Oct_BSellerC_15745581_2&pf_rd_p=0f84f1c4-895a-58ad-9522-659718ad1142&pf_rd_s=merchandised-search-

		6&pf_rd_t=101&pf_rd_i=15745581&pf_rd_m=ATVPDKIKX0DER&pf_rd_r=4BMVA6YJ9WPWFMVYT7G0&th=1
	Battery Connectors	https://www.amazon.com/Pangda-Connection-Battery-Connector-Buckles/dp/B072Z91Y9H/ref=sr_1_1?crid=1WI1ZW80REFI&keywords=batteries+connectors&qid=1551066703&s=electronics&sprefix=batteries+conn%2Celectronics%2C138&sr=1-1
	10Ω resistor	https://www.amazon.com/Elegoo-Values-Resistor-Assortment-Compliant/dp/B072BL2VX1/ref=sr_1_1_sspa?ie=UTF8&qid=1551067463&sr=8-1-spons&keywords=resistor+kit&psc=1
	IRF530 N-type mosfets	https://www.amazon.com/Brands-IRF530-Transistor-N-Channel-MOSFET/dp/B00B888ZP0/ref=sr_1_1?ie=UTF8&qid=1551071858&sr=8-1&keywords=irf530
Automated Delivery	PVC pipes	
	SPDT switch	https://www.amazon.com/LIYUDL-Position-Vertical-Switch-SS12D00G3/dp/B075TYP6C5/ref=sr_1_15?brr=1&qid=1557253415&rd=1&s=industrial&sr=1-15
	DC motor	https://www.adafruit.com/product/3777
	Motor Mount	https://www.adafruit.com/product/3768
Smart Card Payments	ESP8266 WiFi module	https://www.amazon.com/gp/product/B010N1SPRK/ref=ppx_yo_dt_b_asin_title_o07_s00?ie=UTF8&psc=1
	RFID card reader rc-522 With RFID tags	https://www.amazon.com/Arduino-Kits-RFID-RC522-Sensor-Module/dp/B07KS17ZLF/ref=sr_1_6?keywords=rc522&qid=1557254177&s=toys-and-games&sr=1-6

We used Android Studio to build an app for placing orders. We also came across MIT app development tool wherein you do not need any programming, and it is a very interactive tool to build an app. But we wanted our app to be very user friendly and engaging, and the app develop tool builds a very basic button press app and was very different in terms of integration with the database. So, we went with programming our entire UI (Java Programming) with Android Studio. Also, we integrated our app with a real-time database provided by Google called Firebase. The database uses key-value pairs to store information, and it is very monotonous to scroll through it every day for every order request. To make it better, we developed an app for the chef/restaurant staff as well.

We are using Arduino UNO microcontroller for each of our features which was provided to us from the course faculty.

The link for breadboard and connectors is also mentioned below:
https://www.amazon.com/Standard-Jumper-Solderless-Prototype-Breadboard/dp/B07H7V1X7Y/ref=sr_1_11_sspa?keywords=breadboard&qid=1552513856&s=gateway&sr=8-11-spons&psc=1

3. Key Use Cases

Through this project we target dining places like:

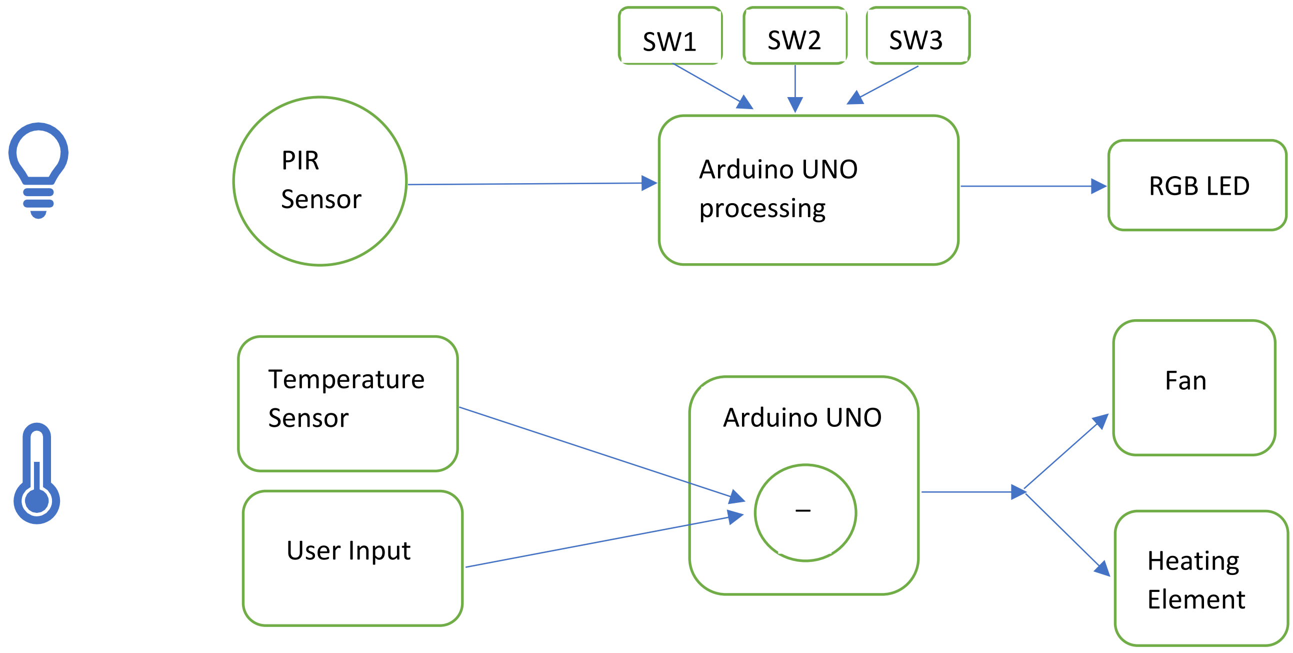
- Restaurants and hotel dine-in places.
- Movie theatre

The individual features of our restaurant could be used in many places.

1. Smart lighting and temperature control could be used in buildings, conference rooms, classrooms etc.
2. Nowadays every big chain of restaurants has their own mobile apps that enable customers to order from anywhere over internet from the nearest restaurant outlet.
3. Automated delivery is not entirely a new thing, but it remains a field worth exploring. In some countries like China and Japan, a rotating platform is used to deliver food to the customer's table.
4. Payments are automated like everywhere. Credit/Debit facility is available at every outlet. Now some outlets have come up with the idea of a card that is outlet specific. You cannot use it elsewhere.

4. System Design

Our system design can be summarized in the diagram below.



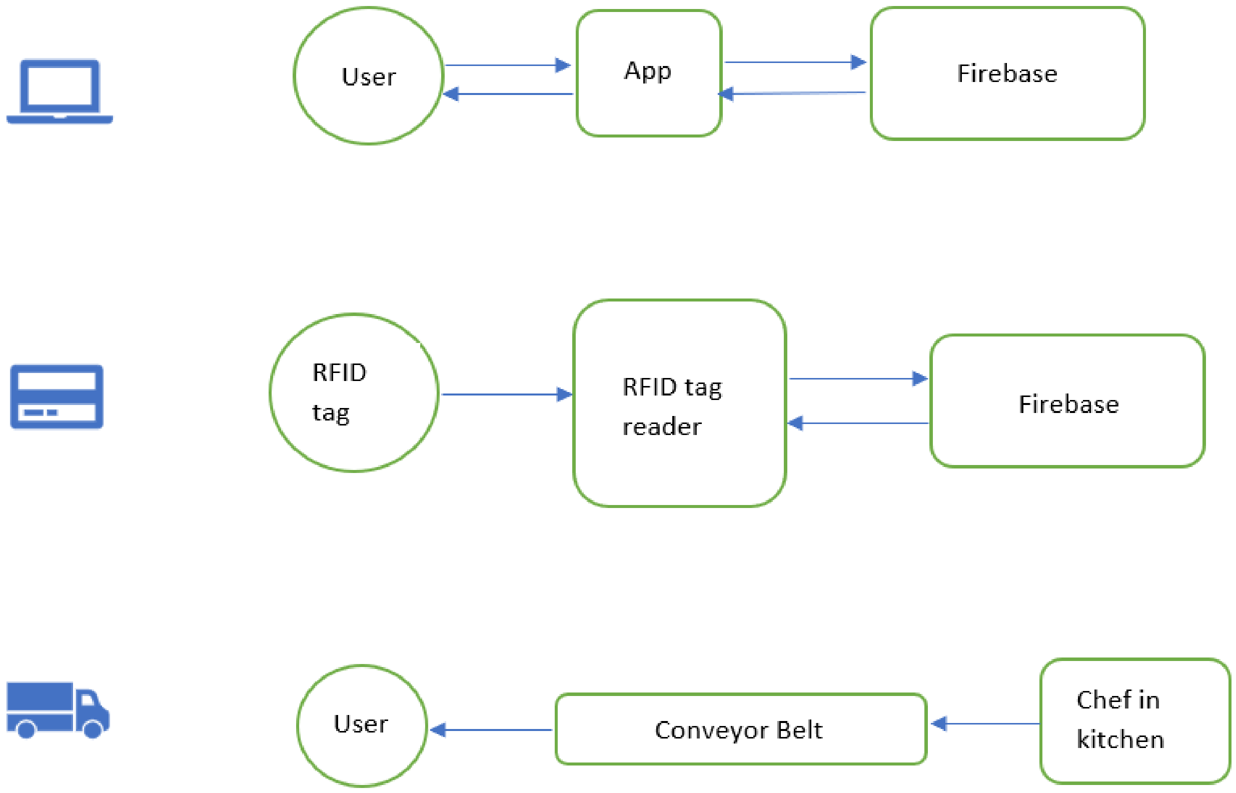


Fig. 4.1 Design of individual features of Smart Restaurant

5. System Implementation

Keeping the project scope in mind, we are following a bottom up approach in our project. We first designed and developed individual features and then integrated everything for our final demonstration.
Here is the system design for each individual feature we planned on implementing:

5.1 Smart Lighting

The aim of creating such a model is to harvest energy by turning on the lights only when it detects human presence near it. This is an economic solution to today's increasing energy needs. Along with this, the brightness of these lights is a customer controllable feature.

The PIR sensor HC-SR501 range has the following specifications. There is a delay of a few seconds before lights turn off when the PIR detects nothing.

a. Operating voltage - 5V
b. Sensor detection angle range - 100°
c. Distance range - 3m

We used SPDT on/off switches to choose color settings of the RGB LEDs. There are three settings in our prototype, and we have given here analogWrite values for the RGB pins of the LEDs. It turns out Red and Green at full intensity doesn't give Yellow color. It actually turns more green (GREEN has lower forward voltage i.e. 2V than RED having forward voltage of 3.2V so RED requires more voltage to reach its full intensity), so instead of giving digitalWrite (HIGH/LOW), we use analogWrite to control the intensity of green color.

a. Normal Setting – Yellow Color (R(255), G(50), B(0)). ------------------DEFAULT setting
b. Business Setting – White Color (R(255), G(100), B(50)).
c. Romantic Setting – Magenta Color (R(255), G(0), B(100)).

If any of the two settings mentioned above are simultaneously ON, we change the light color to its default setting.

Fig. 5.1.1 shows the virtual circuit design of Smart Lighting. We have used tinkerpad for the circuit building.

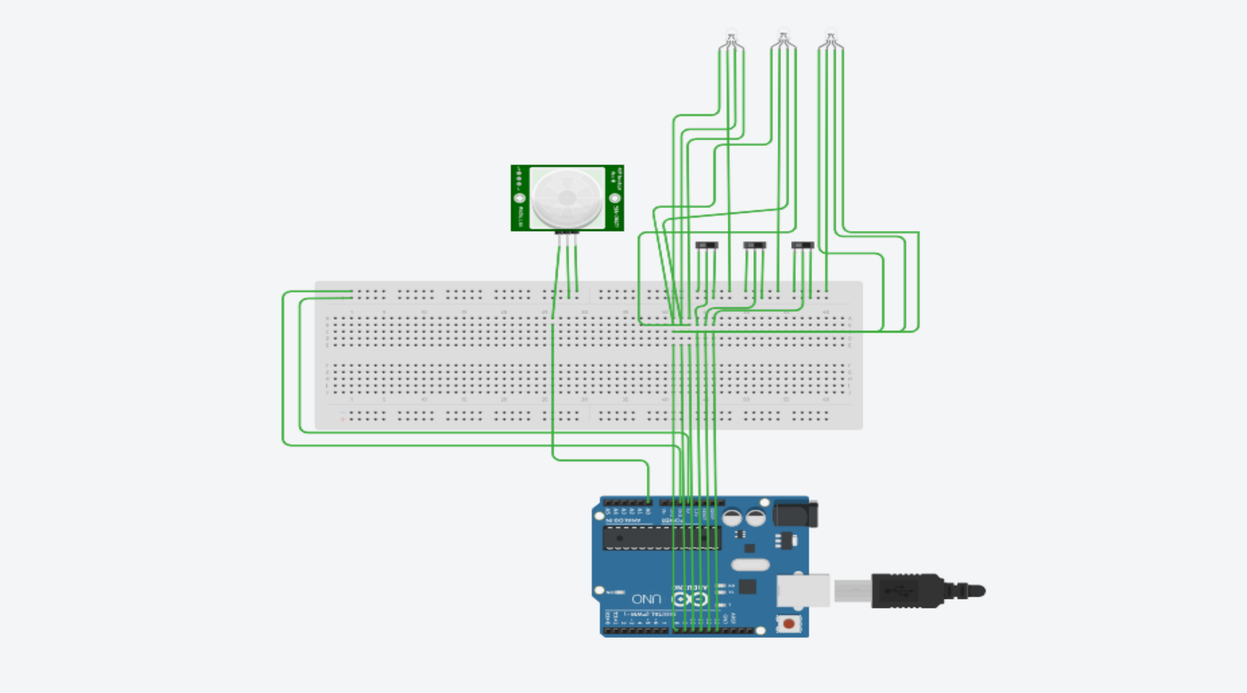


Fig. 5.1.1 Smart Lighting System: PIR senses the presence of the customer near the table. Arduino UNO records the output of PIR and monitors SPDT switches' state to control the color settings of the RGB LEDs (Normal- Yellow, Business- White, Romantic- Magenta).

5.2 Smart Temperature Control

The temperature could be controlled by a customer near his table. This was achieved by a temperature sensing system with a closed loop response to control the speed of a fan near a customer's table. In real scenarios this can be extended to control the temperature of an air conditioning system. Fig. 5.2.1 explains shows a tinkerpad schematic of our circuit and, also mentions what each component does.

Some important points to mention here:

1. We have used 9V batteries to supply enough voltage to fan and heating element as the Arduino output of 5V is not enough to drive them.

2. We have used IRF530 mosfet to provide enough drain current to drive the motor and the fan. Since mosfet's threshold voltage is 2V it can be easily driven by Arduino's 5V output.

3. We used I2C LCD displays as they have less number of pins as compared to normal LCDs. This leads to efficient usage of Arduino pins and we integrated 2 systems on 1 board, Smart Temperature Control and Automated Delivery (explained later). Since multiple I2C devices can make use of single Arduino board, we found out the I2C address of the LCD before using it in our system.

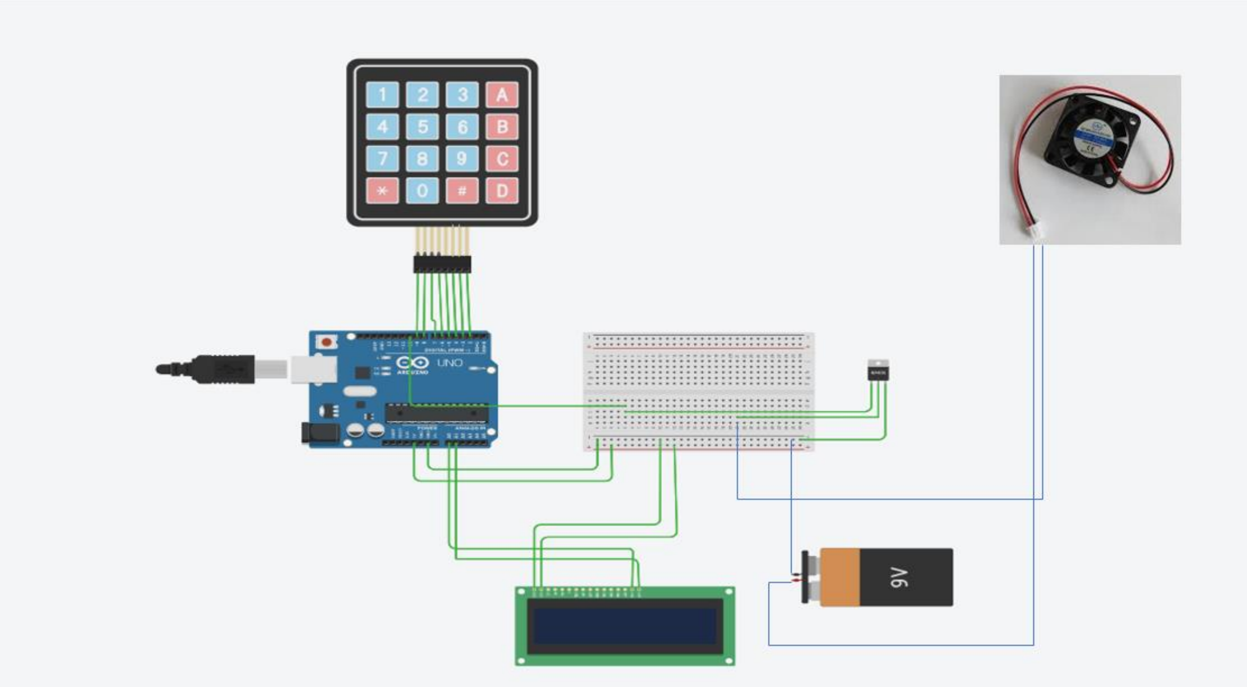


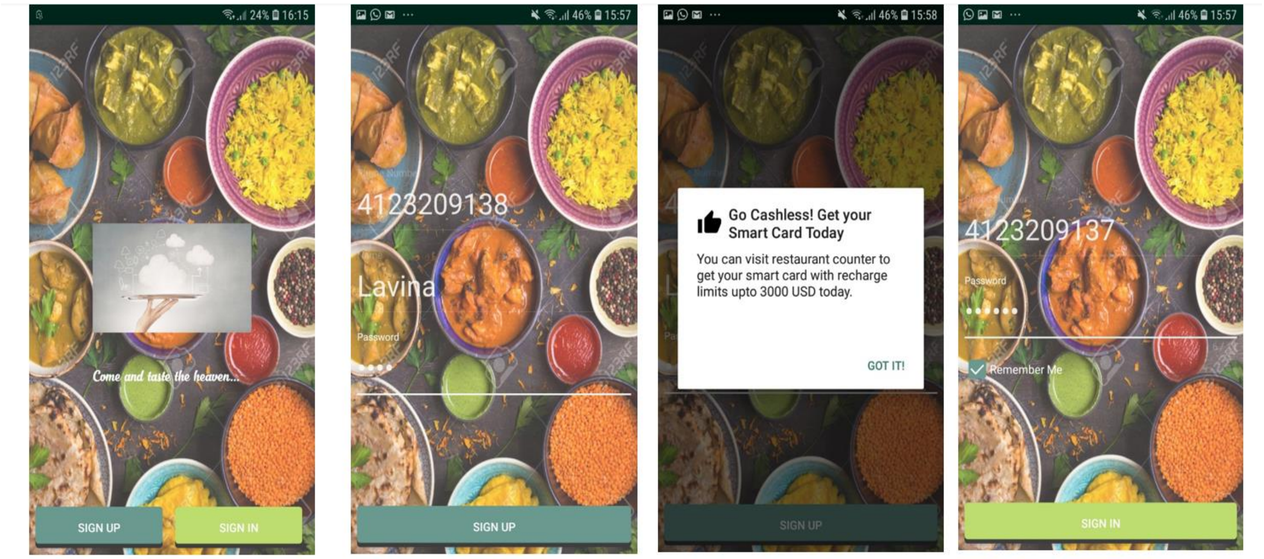
Fig. 5.2.1 Smart Temperature Control System: Temperature sensor DHT22 gives the value of current temperature to Arduino. Adafruit 3x4 Keypad takes temperature value as input from the customer. Arduino LCD – Displays the customer entered temperature value. Arduino UNO – Decides what element to turn on fan/heating pad depending on the difference of two temperature values (current temperature and customer entered value). **Here we have not shown heating element and the keypad shown is 4x4. We have used 3x4 keypad instead. Also, we have combined two 9V batteries in serial with a 10Ω resistor to supply enough voltage to our DC fan which operates at 12V.

5.3 Online App-based Ordering

This is the most tiring step in this prototype as it has a lot to deal with. We created an Android app for the restaurant to display restaurant menu, place order, review order details and make payments for each order.

The customer places his order through this app by selecting appropriate menu items, once ordering is complete the chef in the kitchen is able to see what the customer has ordered, its quantity, payment status and from which table the order was placed on another app that was built exclusively for restaurant staff. If the app is not accessible by the chef, he can see the orders on the database. We have used Firebase to store everything. The chef then sends real time notifications on the order status to the customer's app.

Following figures display the different pages on our app. Our app is compatible with Android 6.0.0 version and above.

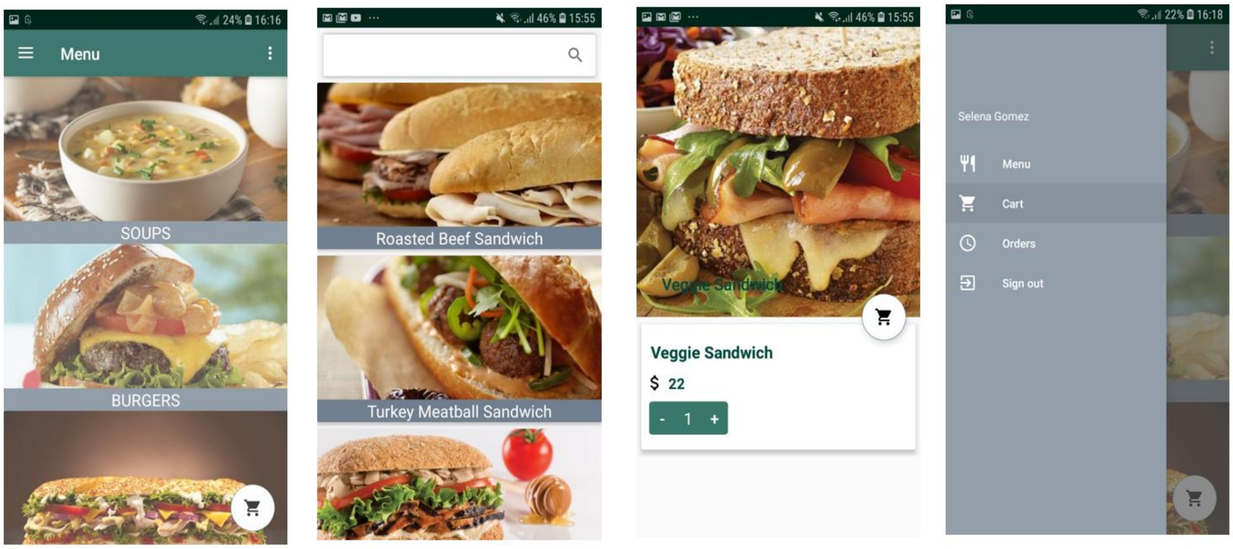


Go Cashless! Get your Smart Card Today

You can visit restaurant counter to get your smart card with recharge limits upto 3000 USD today.

Sign In/Sign Up options:
If you choose Sign Up, you'll have to create an account and enter your credentials. For Sign Up phone number should be unique. After successful Sign Up a new user is created on the Firebase database with your credentials. Also, you'll see a popup asking you to request for your restaurant smart card.
If you choose Sign In, you'll enter your phone number and password to sign in. You can click on Remember Me so that you do not have fill in your credentials again and again when you reopen your app.

Fig. 5.3.1 Sign In/ Sign Up page



Veggie Sandwich

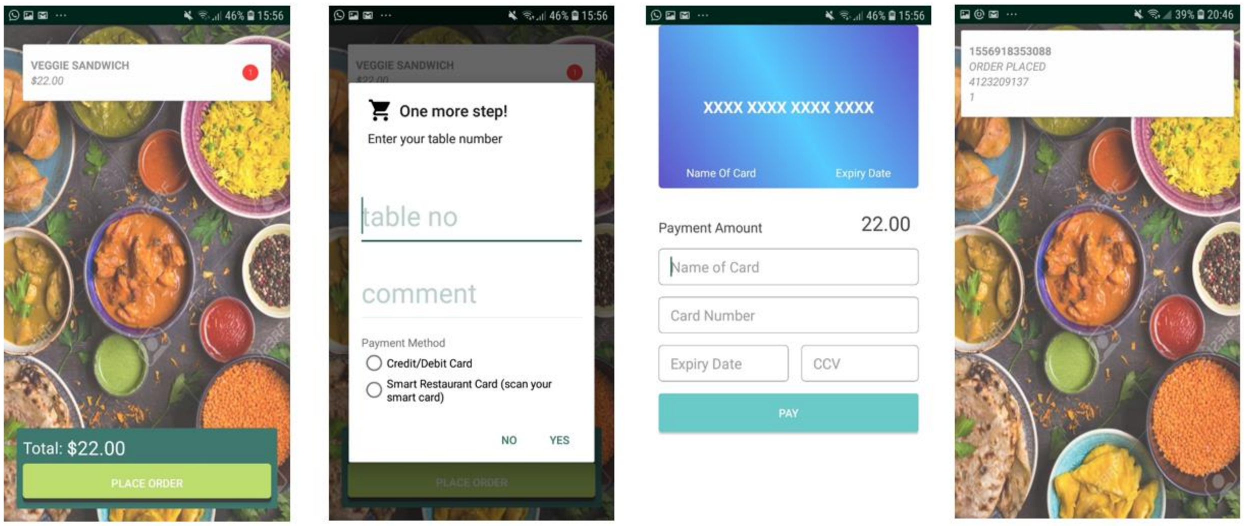
$ 22

- 1 +

Menu:

Our menu provides you with variety of food items. When you choose sandwiches sections, a page with all the available sandwiches shows up. There is a search option in this page and you can use it to search your favorite sandwich. If you type chicken, all sandwiches with chicken phrase appear in your search. You can tap on any sandwich to open up the page where you can add the sandwich to your cart (cart button) and also decide the quantity (quantity +/- button). All your added items appear in your cart. You can view items in your cart by clicking cart.

Fig. 5.3.2 Menu options and adding food items to your cart.



VEGGIE SANDWICH
$22.00

Total: $22.00

One more step!

Enter your table number

table no

comment

Payment Method

○ Credit/Debit Card

○ Smart Restaurant Card (scan your smart card)

Payment Amount 22.00

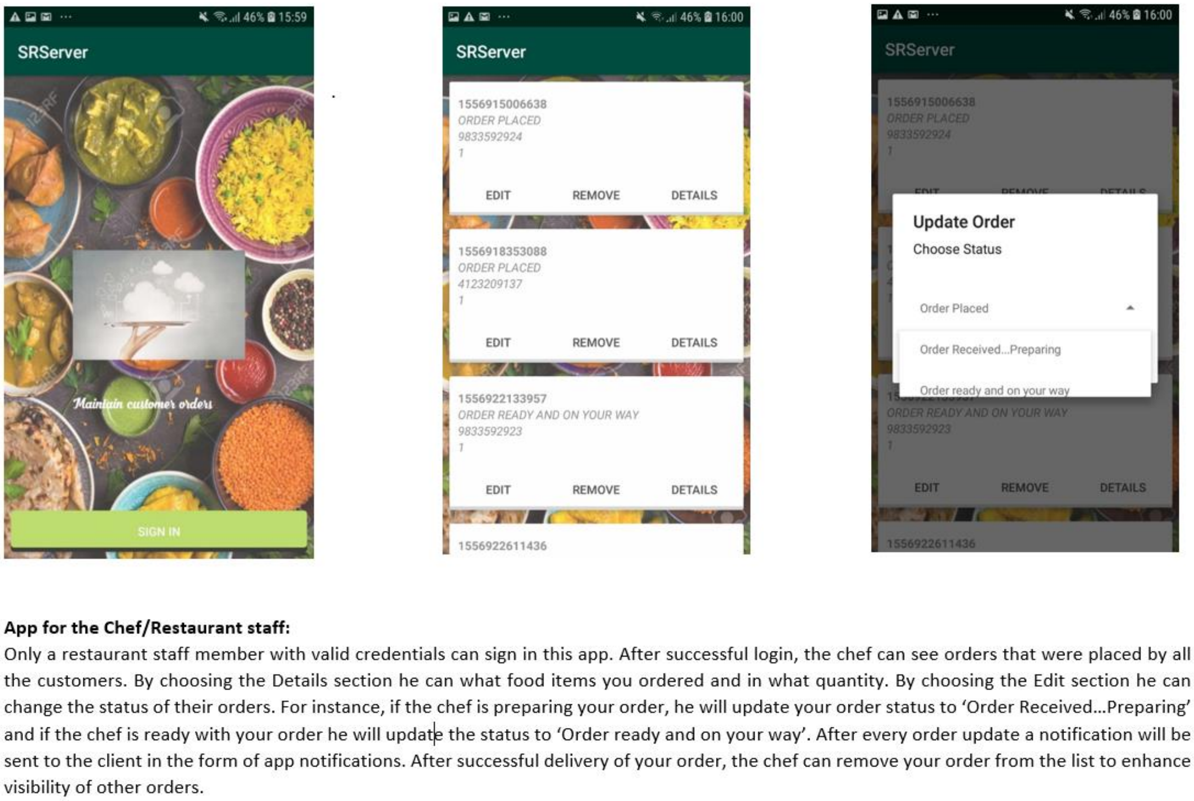
1556918353088
ORDER PLACED
4123209137
1

Cart and Place Order:

You can view what food items are present in your cart, see total price (without tax) and place order. When you click on Place Order, you will see a pop up to fill in your table no, comments to customize your order (like pizza with extra cheese) and select a payment option. If you choose credit/debit card you will be prompted to a page to fill in your card details and pay. If you choose Smart Card option to pay, the app will trigger a timer in its background and check for the status (scanned/unscanned) of your smart card after timer expires. Your job is to you're your card with the centralised card reader system available in the restaurant. After successful payment, you can view your orders and order status.

Fig. 5.3.3 View cart, place order and pay.

We have also developed an app for the chef.



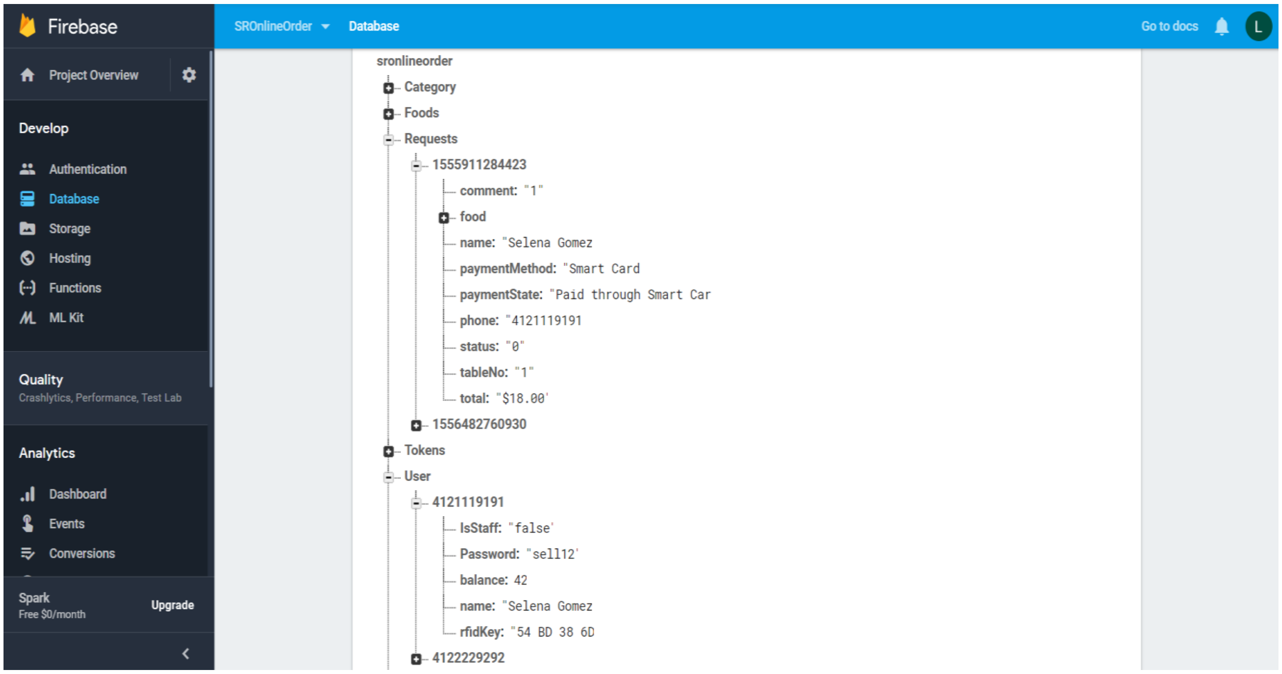
App for the Chef/Restaurant staff:

Only a restaurant staff member with valid credentials can sign in this app. After successful login, the chef can see orders that were placed by all the customers. By choosing the Details section he can what food items you ordered and in what quantity. By choosing the Edit section he can change the status of their orders. For instance, if the chef is preparing your order, he will update your order status to 'Order Received…Preparing' and if the chef is ready with your order he will update the status to 'Order ready and on your way'. After every order update a notification will be sent to the client in the form of app notifications. After successful delivery of your order, the chef can remove your order from the list to enhance visibility of other orders.

Fig. 5.3.4 App for the chef.

If for some reason the app is not accessible to the restaurant staff, they can control all requests on the database. We have integrated our app with Firebase- a real time database provided by Google. It runs on Google Cloud so we have no worries of losing our data. We have used following features of Firebase:

a. Messaging – to send order update notifications to our customers.
b. Realtime Database – to keep track of orders and payments.
c. Storage – to store information about customer accounts (login credentials), and their Smart Card balance.
d. Content – to display images of food items in the menu. We never stored any image on the Cloud. All images were picked up from Google Images and we provided the link of these images in database. Whenever one opens the app, all food images are grabbed from that link and displayed on the app.



```
sronlineorder
 ⊞- Category
 ⊞- Foods
 ⊟- Requests
     ⊟- 1555911284423
         ├ comment: "1"
         ⊞- food
         ├ name: "Selena Gomez
         ├ paymentMethod: "Smart Card
         ├ paymentState: "Paid through Smart Car
         ├ phone: "4121119191
         ├ status: "0"
         ├ tableNo: "1"
         └ total: "$18.00'
     ⊞- 1556482760930
 ⊞- Tokens
 ⊟- User
     ⊟- 4121119191
         ├ IsStaff: "false'
         ├ Password: "sell12'
         ├ balance: 42
         ├ name: "Selena Gomez
         └ rfidKey: "54 BD 38 6D
     ⊞- 4122229292
```

Fig. 5.3.5 Firebase: This is a glimpse of our database. All information is stored as key-value pair, for ex. User is the key, 4121119191 (customer's phone no.) is the child (one of the users) and Password is the value for that child. In the Requests section you can view all orders and their details.

5.4 Automated Delivery

For the delivery system we built a conveyor belt from cardboard, pvc pipes, chopsticks, rubber band and DC motor. The belt carries customer's order and delivers it to his table. The movement of the belt depends on SPDT on/off switch state which is in turn chef controlled.

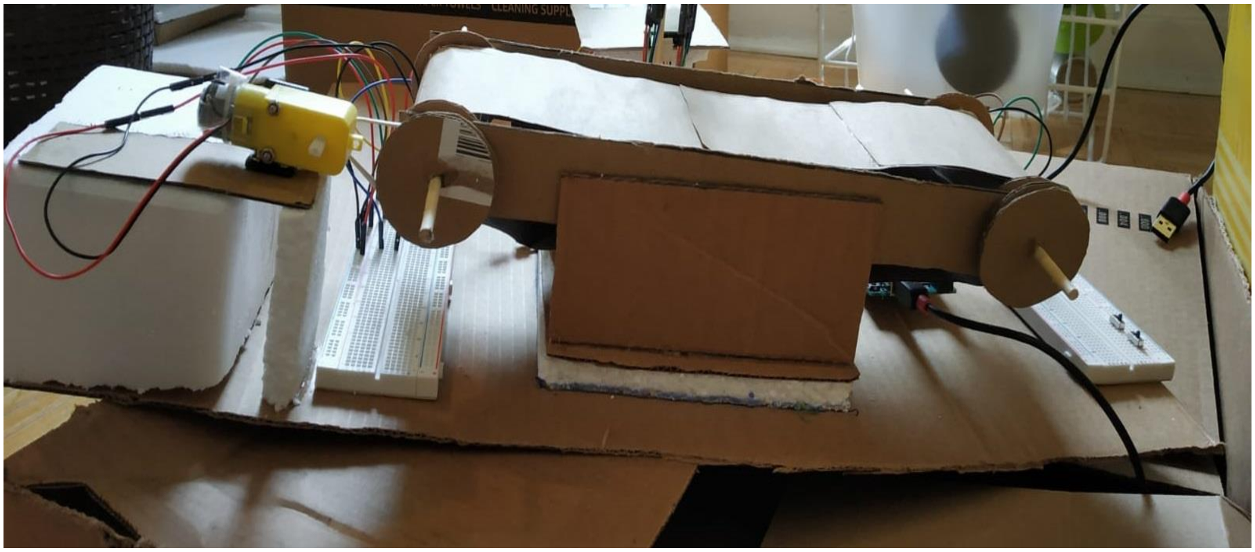


Fig. 5.4.1 Conveyor belt.

5.5 Smart Card Payments

If the customer opts to pay through Smart Card (restaurant customized card for easy payments), he must scan his card at the centralized RFID card reader section. Each card has a unique RFID key and its value is initially set to ‘Unscanned’. When the card is scanned, its value is updated to ‘Scanned’ and the changes are reflected in the database. The app triggers a countdown timer (30 sec) in its background as soon as the customer select the Smart Card Payment option and checks for the status (scanned/unscanned) of his smart card after timer expires. If the app doesn’t detect a Scanned status within 30 sec it will again start the counter and wait for the customer to scan his card. If it detects the Scanned status through the database, it will place the order request on the database and deduct the order amount from the current balance of the customer’s smart card.

We have used the ESP8266 WiFi module to send the status of RFID tag to our database over WiFi/mobile hotspot. Digital payments help checkout real-quick without any hassles.

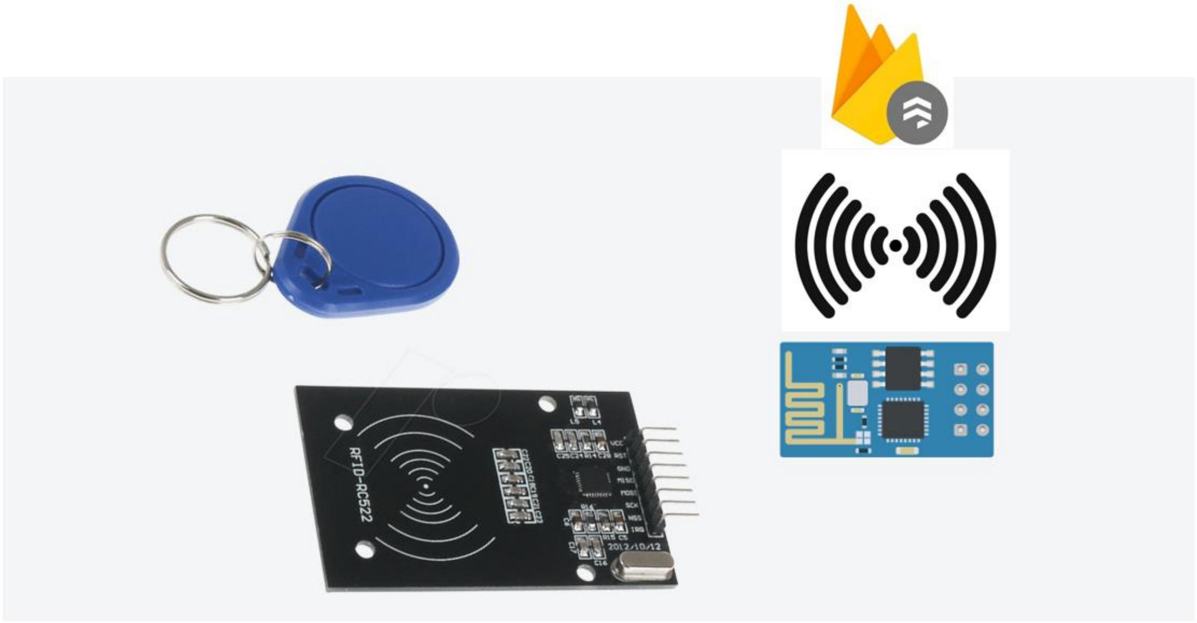


Fig. 5.5.1 Smart Card Payments.

6. System Status & Evaluation

After integrating all the features, this is what our system looks like:

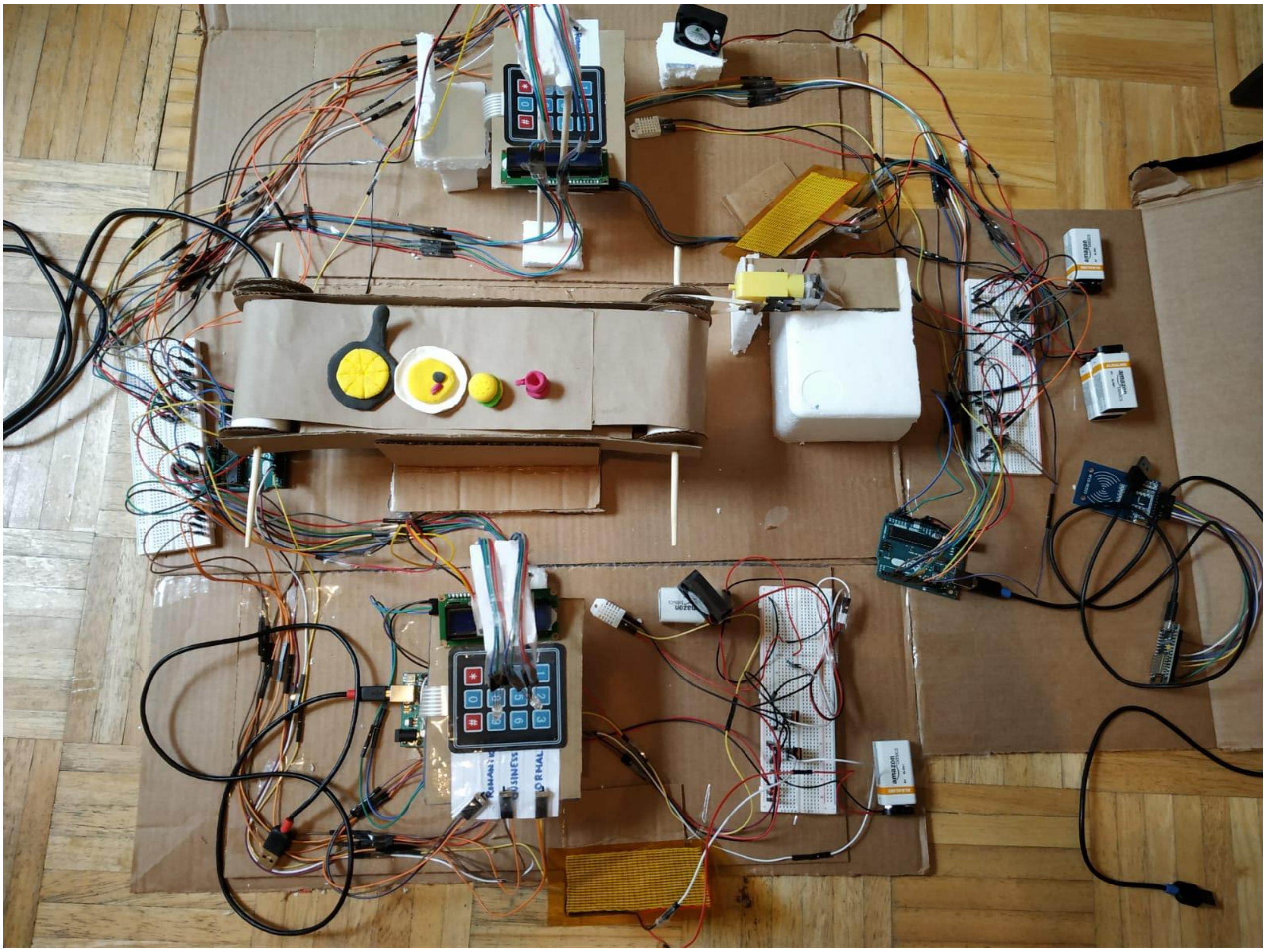


Fig. 6.1 Smart Restaurant Prototype.

Here is the status of each sub-component in our system.

Features	Status
Smart Lighting	Working
Smart Temperature Control	Working
Online Ordering App	Working
Automated Delivery	Working
Smart Card Payments	Working

System Evaluation:

Although there are no perfect grounds on which we can measure our system performance with an ideal system (calculating error etc.), we have come up with some points on which we try to sell our project prototype:

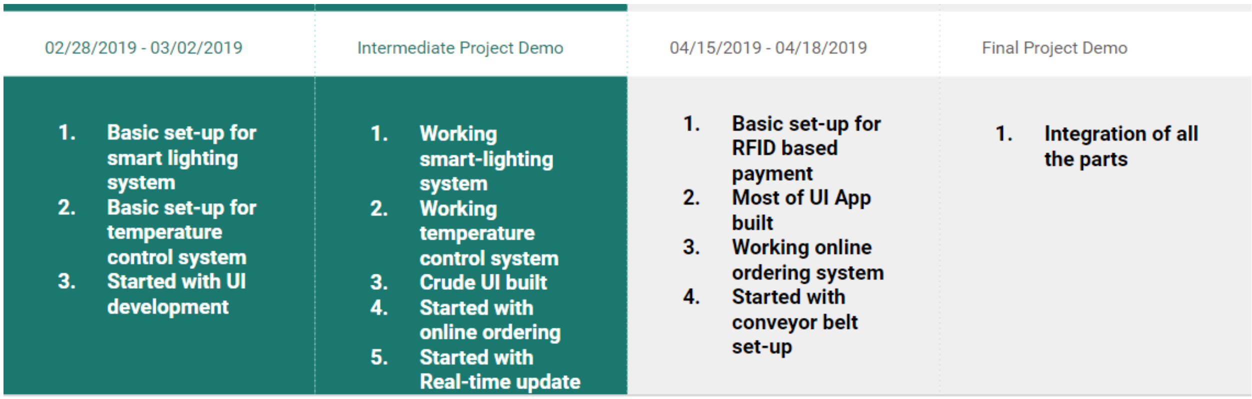
1. Scalability: Perfect scalable solution. We have demonstrated only two tables in our project. Each feature is independent of the other and is scalable to multiple tables in the restaurant. Database can handle data from multiple customers at once.

2. Power Consumption: Since we are operating in an indoor environment, we have wall socket power. So, power consumption is not a serious issue. We have tried lowering consumption in our Smart Lighting, Smart Temperature control and Automated Delivery features that operate only when there is customer involvement. In the Temperature control block, as soon as the current temperature reaches the temperature set by user, fan/heating element are switched off again to save power.

3. Robustness: The system is very robust. We are storing every information of our customer, from his login credentials to his order requests on the database running on Google Cloud. So, the chances of losing data is minimal.

7. Work Schedule

We did follow a strict schedule to get things done on time, especially the app development and integration with Firebase took a lot of time and effort. We finished the project within a time frame of 2 months and presented our final demo on May 3, 2019.



02/28/2019 - 03/02/2019	Intermediate Project Demo	04/15/2019 - 04/18/2019	Final Project Demo
1. Basic set-up for smart lighting system 2. Basic set-up for temperature control system 3. Started with UI development	1. Working smart-lighting system 2. Working temperature control system 3. Crude UI built 4. Started with online ordering 5. Started with Real-time update	1. Basic set-up for RFID based payment 2. Most of UI App built 3. Working online ordering system 4. Started with conveyor belt set-up	1. Integration of all the parts

For final demonstration we had 2 tables, each with its own light settings and temperature control. We had a conveyor belt in the middle of the two tables carrying food orders of the customers. The customer gets live updates of the order through the app. Here is our Demonstration Sequence:

1. Customer arrives at the restaurant
   a. PIR sensor will detect the motion and turn on the lights for that specific table.

   b. Customer can choose the light color through on/off switches.
   c. Customer can set a temperature value that will be compared with current temperature value and monitors the switching on/off the fan/heating element. present near each table.
2. Ordering food
   a. Customer will place the order through the app.
   b. The customer can pay through any one of the digital payment options available.
   c. Once, the order is placed, chef will be notified about the order through the chef's app.
3. Delivery of the food
   a. Once the food is ready, chef will update the order status through the chef's app which will send the notification to the customer's app.
   b. Chef will place the food on the conveyor belt, which can then be grabbed by the customer when it reaches near his table.

Complete animated demonstration video: https://www.youtube.com/watch?v=39aKXtcYLIc

8. Project Budget

The price of each component and its quantity are mentioned in the following table. The total expense is stated in the end.

*This is the total expense of our project. There are some components (in Red) that we planned to use but didn't use in our project, for ex. Motor Wheels for conveyor belt and Logic Gates.

Component Name	Quantity	Price/unit	Total price (USD)
Connectors + breadboard	1 (pack of 120 connectors +3 breadboards)	11.99	11.99
LCD screen	1 (pack of 2)	9.99	9.99
Switches	1 (pack of 10)	1.83	1.83
Motor Wheels	2	1.5	3
Connectors	3 (pack of 120)	5.99	17.97
DC motors	2	2.95	5.9
Keypad	2	3.95	7.9
PIR sensor	2	11.5	23
Battery Connectors	1 (pack of 8)	5.99	5.99
Batteries	1 (pack of 8)	9.99	9.99
RGB LED	15	0.39	5.85
Temperature sensor	2	9.95	19.9

Resistor	1 (pack of 525)	10.86	10.86
Mosfet	1 (pack of 10)	7.9	7.9
Logic Gates	1 (pack of 30 different IC)	21.99	21.99
DC fan	1 (pack of 4)	9.49	9.49
Motor mount	2	1.5	3
ESP8266 WiFi module	1 (pack of 2)	12.99	12.99
Heating Element	2	8.59	17.18
Bicycle tube	1	2.99	2.99

Total expense: 206.71 USD
Total used expense (excludes components we didn't use): 178.73 USD.

We were provided RFID tags, RFID tag reader module and Arduino UNO boards by the course faculty.

9. Responsibilities

Here is the work partitioning for our project:

Task	**Responsibility**
Lighting Control	Lavina
Temperature Control	Shubhankar
App development	Lavina and Shubhankar
Conveyor belt delivery system	Lavina and Shubhankar
Billing System with RFID	Shubhankar
Database Management	Lavina

10. Challenges Faced and Solutions

Our major challenges and their solutions are given below:

1. To build a synchronized and controlled conveyor belt: We were initially using Motor wheels to build our conveyor belt system. The problem was no material stayed intact on the wheels due to its very small surface area.
   Solution: Instead of using wheels we used PVC pipes and wrapped a cardboard belt around it. PVC pipes increased the surface area of the rotating part of the conveyor belt.

2. App Development: In the way of making a very interactive UI for the app, we faced several crashes and app failures in the testing phase.
   Solution: Online help and rigorous testing made our app very robust.

11. System Improvements for Future

We propose following improvements in our system:

1. Feedback from customers: Our app gives freedom to our customers to customize their orders but doesn't ask their feedback for the food quality. We can introduce that feature in our app.

2. Conveyor Belt: Our conveyor belt now is controlled manually by the chef through wired connection. We can make this wireless in future.

12. Conclusion

Smart Restaurants is a new concept with promising revenue generation for today's modern restaurants. We developed a prototype of a Smart Restaurant and presented a working demonstration in the final demo session.

13. References

1. https://www.75f.io/blog/topic/sustainable-restaurant
2. https://pos.toasttab.com/blog/2017-restaurant-technology-trends
3. https://www.mpja.com/download/31227sc.pdf