Desk Reservation System

ECE 445 Final Report - Spring 2020

Team 37: Pragya Aneja, Ninad Godbole, Varad Khandelwal

TA: Shaoyu Meng

Abstract

Students face many issues when it comes to finding an available spot to study consistently for a period of time which hinders their ability to study effectively. This was the problem that the original solution tried to solve with a desk reservation system which consists of a phone application, central server and a desk module. Our solution builds upon this but differentiates itself in the way communication occurs by making use of what is called a bluetooth scatternet. This network involves a master unit connected to several slave units which in turn connect to another level of slave units.

Table of Contents

1. Motivation

1.1 Problem Statement

As a student, libraries are a fundamental part of college life. It is therefore vital that the student is able to find a place to sit to be able to study effectively. However, due to overcrowding it can become very difficult to find a spot, and even more so if one has a group study session. Additionally, as different students have vastly different timetables, coming early into the library to save a spot is also hard. Furthermore, it can regularly happen that even though there are some desks available, it takes a lot of time to scan the entire library to find it. For example, in Grainger Engineering Library there are 5 floors and to find a desk to sit at can take a lot of time which then reduces the student's time to study. This problem only exacerbates during exam time, where finding alone, distraction free time becomes increasingly more of a necessity. As can be seen, there are a variety of issues that are involved when it comes to a student being able to reliably find a spot to study in a place like the Grainger Engineering Library.

1.2 Solution

1.2.1 Objective

Hence, our main objective and goal for this project is to reduce the inefficiency involved in a student trying to find a spot to study. Our solution consists of designing a desk reservation system where one can reserve a desk in advance at a specific time for upto a few hours. This system will increase space efficiency for a library, allow the student to more conveniently find a place to study and allow the student to make more prudent use of his/her time. The biggest aspect which differentiates the original and current solution is the cost which we managed to greatly reduce by changing the structure of our subsystems for the system.

1.2.2 Background

Today, reservation systems are used in a plethora of settings and contexts and have been for a while. One example of this is in the hospitality industry, i.e. booking rooms for a hotel. By allowing customers to use an online reservation system to choose the type of room and duration of stay, the customers have a more streamlined experience [1]. Another example of this is setting up an appointment to get a haircut. Scheduling the haircut beforehand increases efficiency by making it so that there isn't overcrowding and large wait times in the hair salon [2]. The last example worth discussing is that of reserving study rooms for libraries for universities, which is probably the most related to our project. Similar to the first example regarding the hospitality industry, giving the students the option to reserve study rooms improves their experience.

All these examples are inspiration for the desk reservation system project, which incorporate aspects from the examples mentioned above like improved efficiency through reduced overcrowding and waiting times as well as a better overall experience for the student. The concept of reserving a specific desk in a library isn't that popular most probably because of the cost required due to each desk requiring some sort of communication subsystem. Therefore, our project is trying to look into creating this system while keeping the cost as low as possible.

1.3 High-Level Requirements

1. Within a span of 15 seconds, the student must be able to reserve a particular desk from a desk map corresponding to the chosen time slot and receive confirmation of the reservation with a unique 6 digit reservation ID. The student can also cancel this reservation until 15 minutes of the reservation time.
2. The student must be able to input the 6 digit reservation ID into the master unit via keypad, which should then map to the corresponding desk and turn on the status LED on that desk red within 5 seconds.
3. After 15 minutes, if the student does not confirm the reservation, the central system must add the UIN to a database which it keeps track of. The student must be penalized after the third non-confirmation by removing his/her right to reserve a desk at peak hours.

1.4 Visual Aid

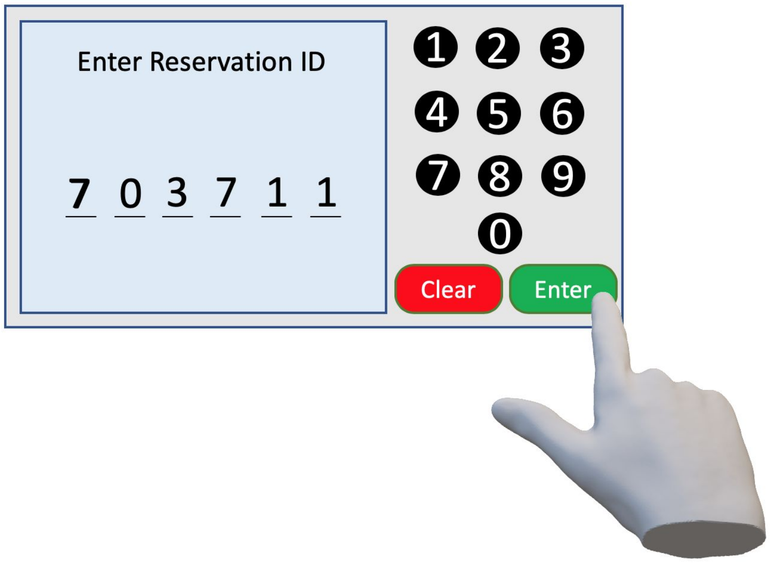


Figure 1: Student entering their reservation ID into the master unit

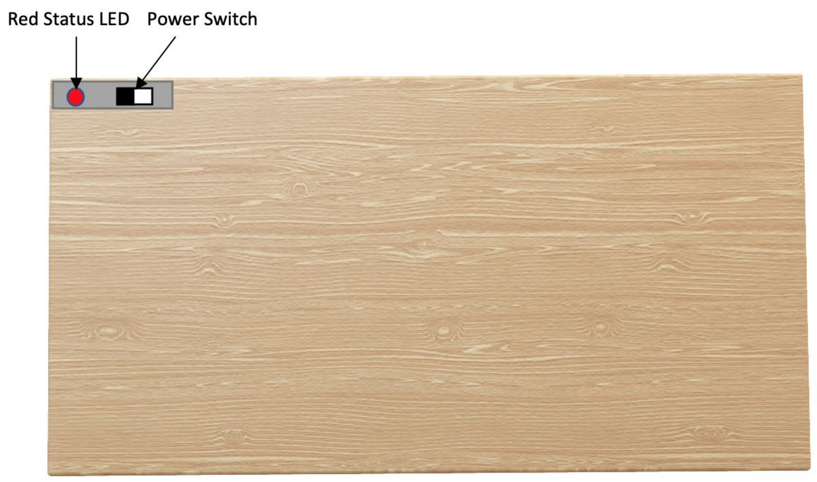


Figure 2: Slave unit attached to desk

1.5 Block Diagrams

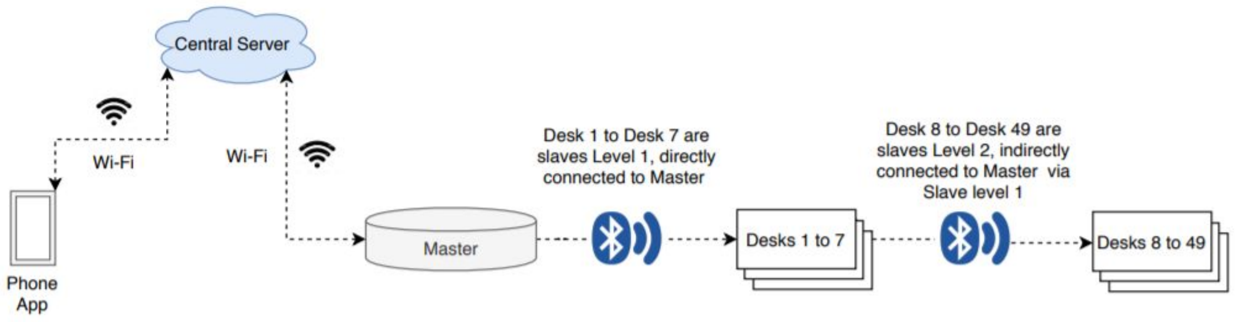

Central Server
Wi-Fi
Wi-Fi
Phone App
Master
Desk 1 to Desk 7 are slaves Level 1, directly connected to Master
Desks 1 to 7
Desk 8 to Desk 49 are slaves Level 2, indirectly connected to Master via Slave level 1
Desks 8 to 49

Figure 3: System Overview

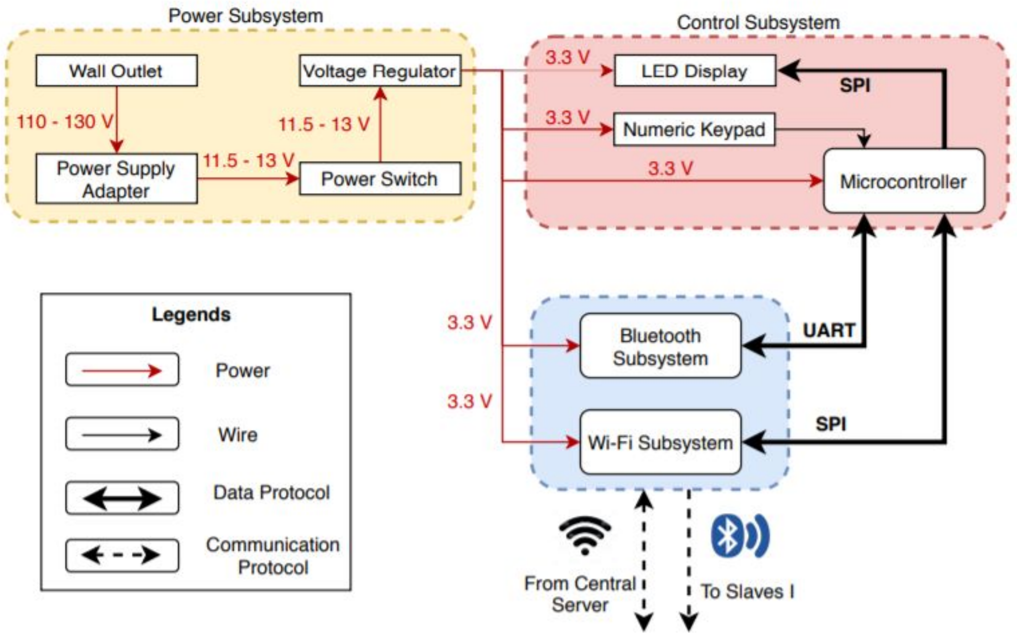


Figure 4: Master Unit

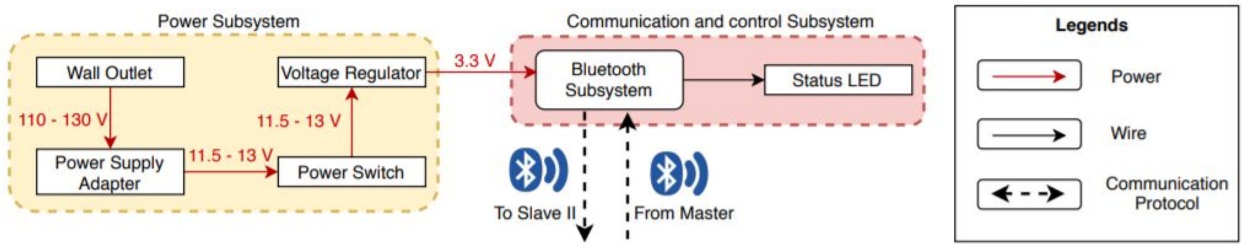


Figure 5: Slave Level 1 Unit

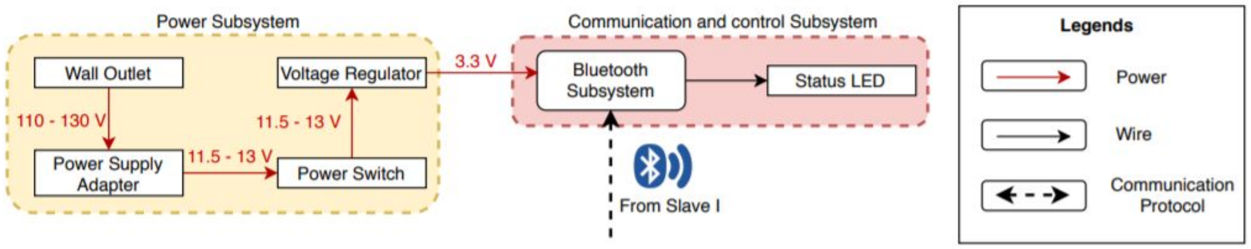


Figure 6: Slave Level 2 Unit

2. Implementation

2.1 Phone application

The desk reservation application is a simple phone application which can be run on both ios and android platform. It is the starting point for user interface journey, and hence we believe an extremely critical component to our project. The phone application works without communicating with the central server and hence there are no API calls implemented in it. It aids us in fulfilling the 1st higher level requirement through:

1. Allowing users to see seat maps (only for 18 seats due to implementation constraint) based on the provided time.
2. Stores reservation details for later reference and provides UI for cancellation. (backend for cancellation not implemented)

2.1.1 Implementation of User Interface

We leveraged a web application called Appgyver to design the UI and basic interface for the application [3]. The first page is the login page, where the user can either log in via typing his UIN and password or the user could sign up. Once the user is done signing up, he/she is directed back to the login page.

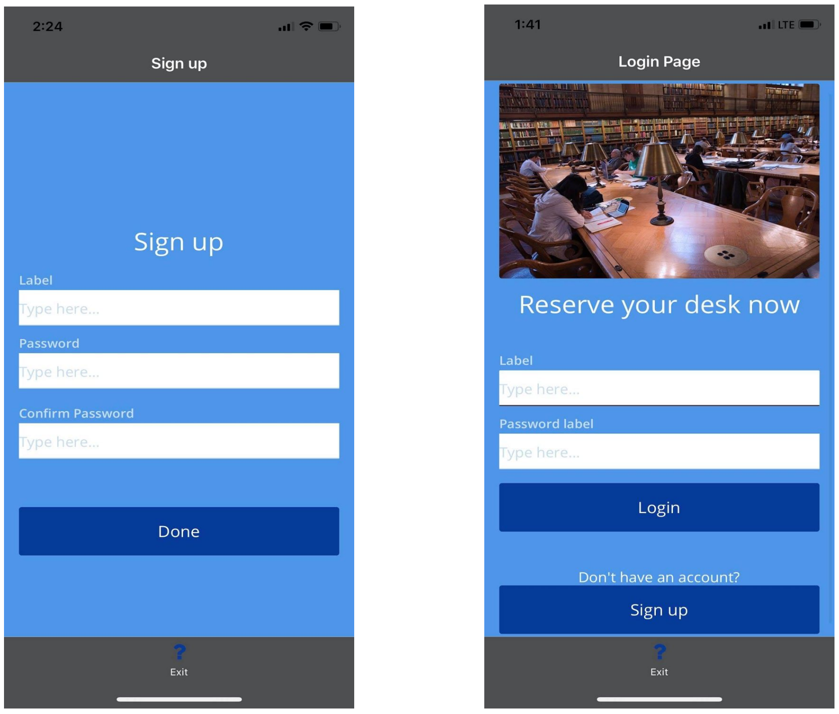
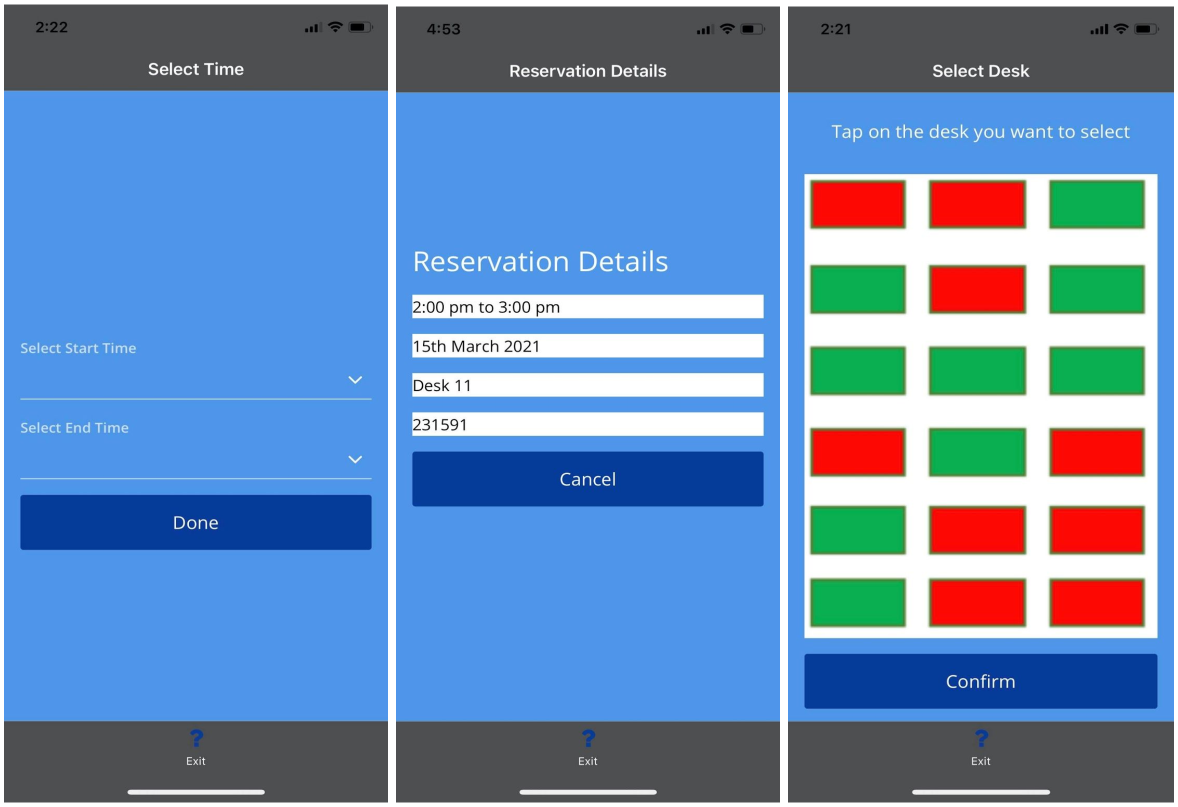


Figure 7: (a) On the left is the sign up page (b) On the right is the login page.

Once the user has entered the application, he/she needs to enter the start & end time of the reservation. Here, the user is automatically directed to a new page to select the desks. In our application, desks are arranged in 3 columns but in a real world use case they would be arranged based on the library blueprint and would even include an option to select different floors. Once, the user taps the desk and confirms, his reservation is saved. He/she can open the application at the time of need to check for reservation details.



Reservation Details

2:00 pm to 3:00 pm

15th March 2021

Desk 11

231591

Figure 8: (a) On the left is the select time page (b) On the right is the confirm seat page (c) In the center is the reservation page with 6 digit reservation code.

2.2: Bluetooth Module

Another critical component of our project was the Bluetooth module as it communicates with slave modules to turn LED lights on. Even before implementation of scatternet, the bluetooth module in the master unit needs to receive the information about the desk units. This information will be provided by the microcontroller and be transferred over the UART protocol. Hence, an understanding of UART protocol is essential in meeting the requirements for the Bluetooth module.

2.2.1 Role in overall project

It critically aids us in fulfilling the 2nd higher level requirement through:

1. Sending data at sufficient speed from microcontroller to bluetooth such that turning on the status LED on that desk red only takes 5 seconds.

It also aids in fulfilling requirement 2 & 3 of Bluetooth communication module (Refer to **6.2.1 - Appendix A**) via:

1. To be able to send packets at the speed of 2Kbps, we need to send 2 kB in 1 second. However, there is latency due to intra-master module communication. So, the implementation of UART protocol is critical such that it consumes less than 10% of the time.
2. Due to pin limitations on the master unit, we ought to use UART protocol as it uses only 2 pins.

2.2.2 Basics of UART

Universal asynchronous receiver/transmitter is a transfer protocol which works without a clock using baud rate. Baud rate is the frequency (in bps) corresponding to the time (in seconds) required to transmit one bit of digital data [4]. Our system works at a 9600 baud rate and in UART protocol data is sent serially. Both the devices have an Rx & Tx pin, where Rx stands for receiver and Tx for transmitter. The pins are cross-connected and the line maintained between them is high at normal state. A normal UART package contains 1 start bit, 5 to 9 data bits, an optional parity bit, and 1 or 2 stop bits.

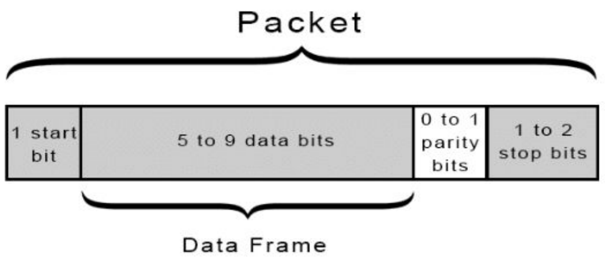


Figure 9: UART package [5]

To start the communication, the line is pulled low and the Rx pin detects it and starts recording the message. Using the parity bit we can check for odd or even parity (needs to be pre-decided) and to end the line pull high for 2 bits.

2.2.3 Transmitting Data

We will simulate sending a signal to desk 4 to turn on the LED. We will assume that desk 1-3 won’t get there LEDs turned on. Additionally, the bluetooth to bluetooth communication is assumed to work. In the design, we are broadcasting information about all desks to each slave at level I, however, here we only show UART signals for upto 4 desks only.

The microcontroller will need to send information about all desks and their LED status. Assume 49 desks, we will need 6 bits for desk ID and 1 bit for LED status (1 would signal LED to turn on). We will start the transmission by sending information about desk 1 and then numerically the next desk and will use even parity.

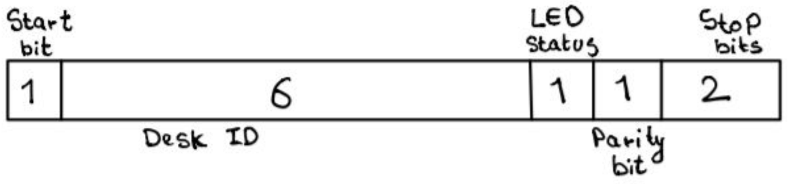


Figure 10: Sample UART package for us [5]

Signals for desk 1-3 will have the LED status bit turned to low, however, desk 4 it will be high. We also send the start bit. Each package will contain information about 1 desk.

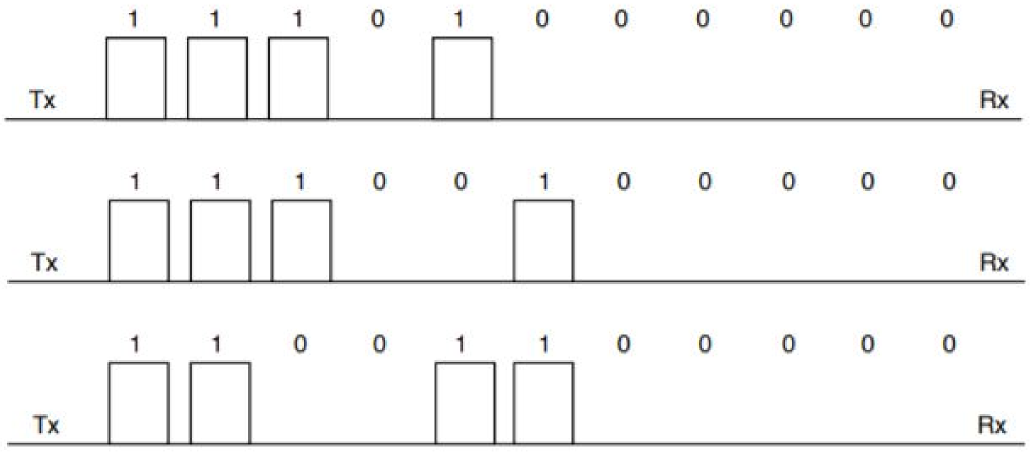


Figure 11: (a) At top a package for Desk 1 representing 0-000001-0-1-11 (b) In middle a package for desk 2 representing 0-000010-0-1-11 (c) At bottom a package for desk 3 representing 0-000011-0-0-11 [5]

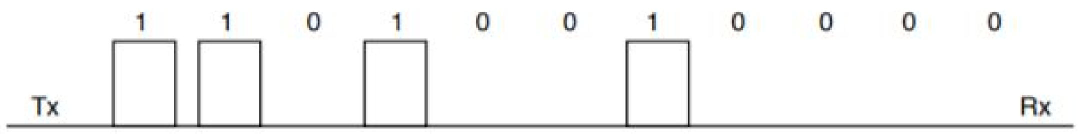


Figure 12: A package for desk 4 representing 0-000100-1-0-11 [5]

2.3 Wi-Fi

As the only channel for transferring information from master unit and to master unit, Wi-Fi forms a critical backbone of the project. As we didn't implement the hardware needed to test the Wi-Fi or the central server, we believe a mathematical analysis will provide better implementation constraints. Additionally, Wi-Fi aids us in fulfilling the 1st and 3rd higher level requirement via:

1. By giving users the freedom to remotely reserve a desk and not force him/her to come into the library to prebook.
2. By transferring data from the phone application to central server and from central server to master within 15 seconds. Additionally, by allowing cancellation requests to go through and help the central server keep track of late users.

2.3.1 Mathematical Analysis: Data overhead

As any communication protocol needs headers and footers to aid the data transfer, communication over Wi-Fi incurs severe overhead due to implementation of TCP protocol. A normal TCP package has an overhead of 16/40 bytes of IPv4/6 header and 16 bytes of TCP header [6]. Assume a churn rate of 30% (30% desk reservations are renewed every hour) and we have 1000 desks in grainger library.

$$Reservations\ renewed\ per\ Master\ unit\ tree\ =\ 30\%\ of\ 49\ =\ 15\ desks$$
$$Number\ of\ Master\ unit\ \ 21\ units$$
$$Reservations\ renewed\ in\ the\ entire\ library\ =\ 15 * 21\ =\ 315/hr$$
$$Packet\ size\ to\ per\ reservation\ =\ 32\ bytes$$
$$Amount\ data\ sent\ per\ hour\ over\ Wi-Fi\ =\ 315\ *\ 32\ bytes\ =\ 10,080\ bytes$$

So, every hour we send 10kB of data and we have 16 bytes of overhead on 128 bytes of data per TCP package.

$$Overhead\ size\ =\ 16/128 * 100\ = 12.5\%$$
$$Data\ sent\ in\ a\ year\ =\ 10Kb\ *\ 24\ *\ 365\ =\ 84.21\ Mb$$

Now we can calculate overhead due to Wi-Fi in a year:

$$Totaldata\ =\ Overhead\ +\ Data\ sent\ over\ an\ year$$
$$TotalData\ *\ (1\ -\ \ Overhead\ Size)\ =\ 84.21$$
$$So,\ TotalData\ =\ 96\ Mb$$
$$Overhead\ \ =\ TotalData\ *\ Overhead\ Size\ =\ 96\ *\ 12.5\%\ =\ 12\ Mb$$

So, even after a year of use with a high churn rate Wi-Fi overhead is relatively less and won't cost anything to accommodate.

2.4: Overall Schematic

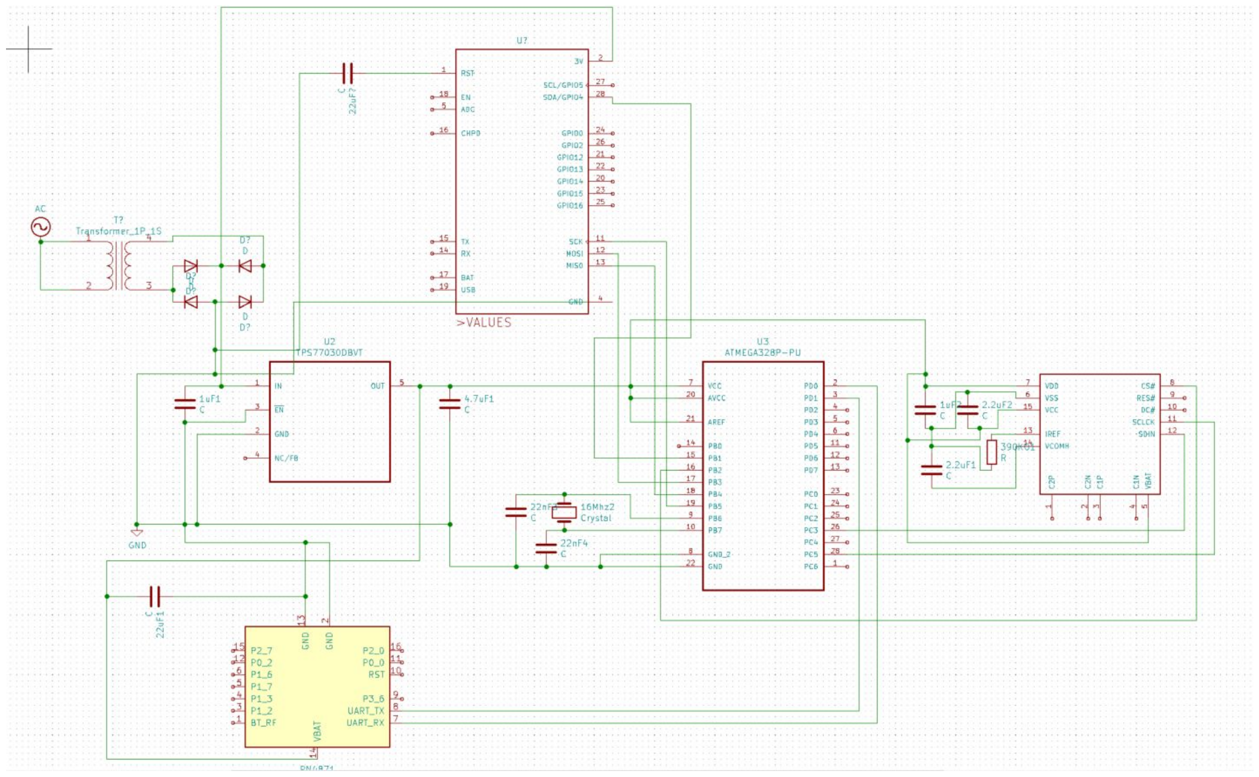


Figure 13: Master Unit Schematic

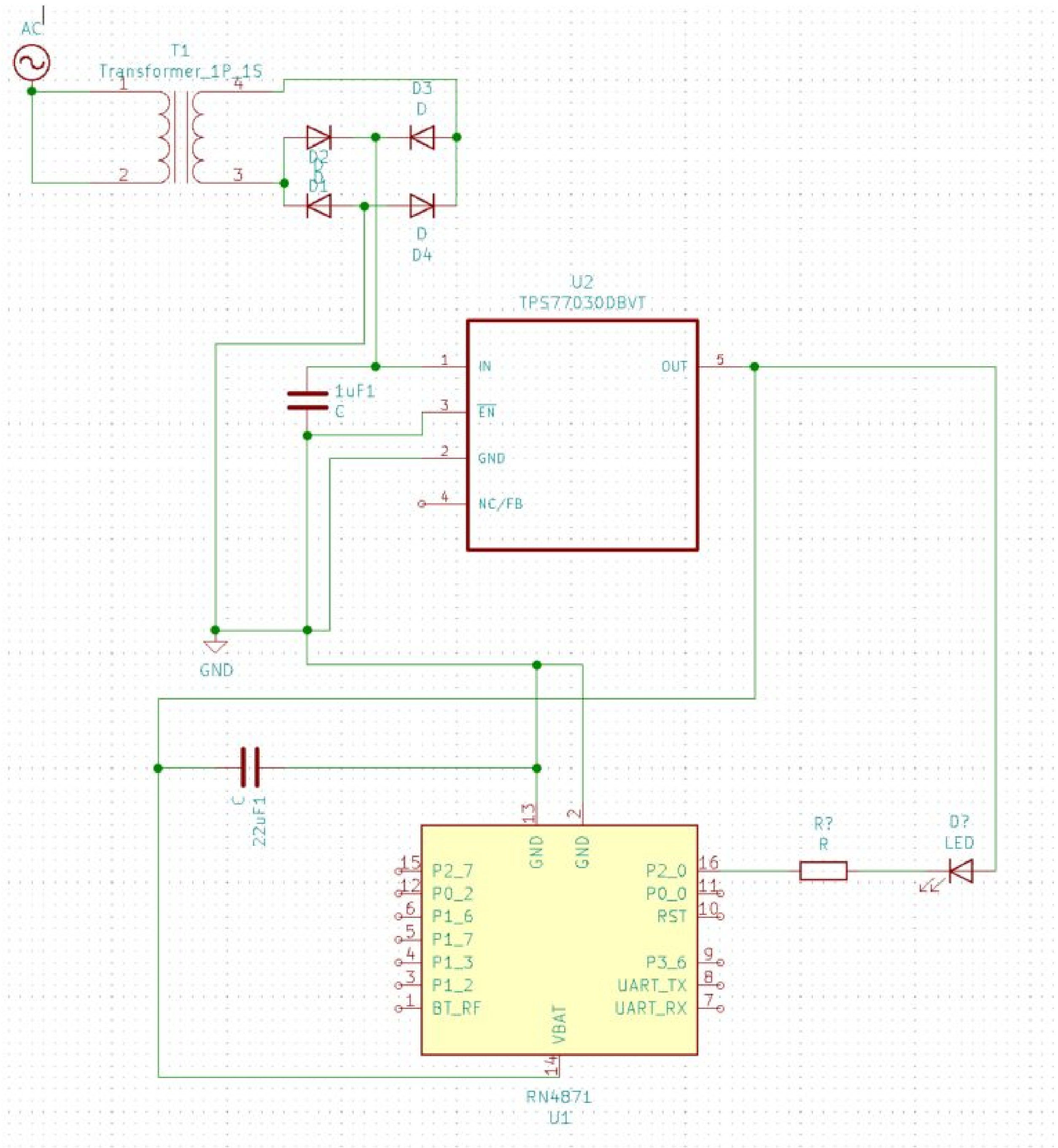


Figure 14: Slave Unit Schematic

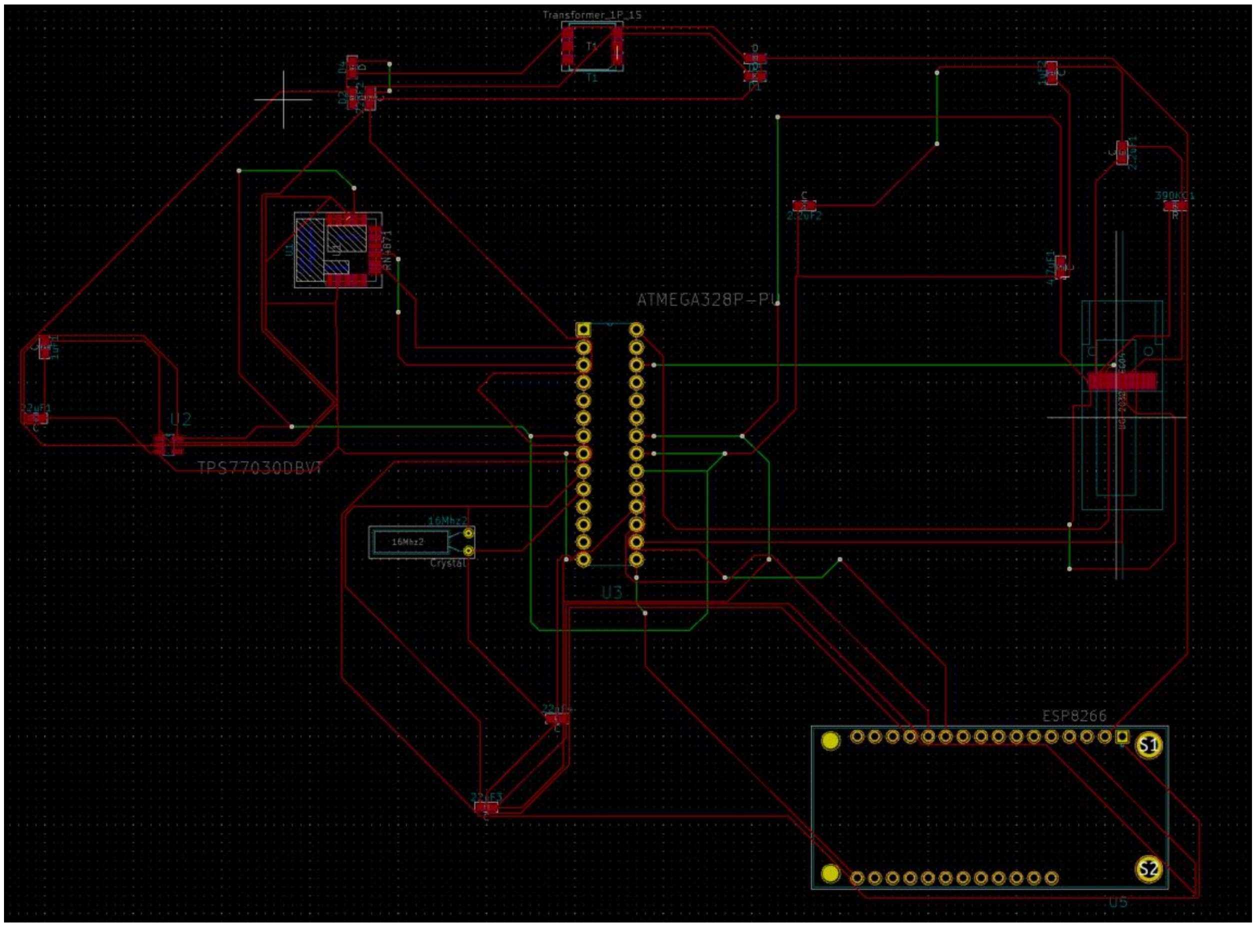


Figure 15: PCB Layout

3. Conclusion

3.1 Implementation Summary

Although we were not able to access the hardware components required to build the master and slave units as part of our desk reservation project, the software components form a critical component of the project and could be accomplished to an extent. One of the key components of our reservation system is the user interface in the form of a phone app, which we were able to successfully implement. Taking inspiration from other reservation apps and trying to improve on their fallacies, we attempted to create a clean user interface to create a smooth user experience. The phone app allows users to see seat maps based on provided time and stores their reservation details for later reference (the user would need the unique reservation ID to enter into the numeric keypad at the master unit for the initiation of the reservation). It also allows the user to cancel their reservation, although currently the cancellation would not be successful due to incomplete backend implementation. The phone app currently stores all of the reservations made by users through the app on a local database. But to implement our desk reservation system to scale, a powerful central server is also key in order to support large data volume, assuming thousands of students would be reserving desks in university libraries. The phone app would then be communicating with the central server using GET and POST requests in order to manipulate the central database.

Another very critical component of our project is the bluetooth module located in all the master units. Apart from the obvious importance of the implementation of the Bluetooth scatternet which would allow the master units to communicate with the desks i.e the slave units, the communication between the bluetooth module and control subsystem is extremely important. When a reservation is initiated by a user, in order to inform the slave units of the initiation and turn on the status LED on the reserved desk, the microcontroller must send the associated desk ID with a status bit =1 to the bluetooth module. Additionally, at the expiration of a reservation realized by tallying of current time with timings of all active reservations, the microcontroller must send the desk ID with a status bit = 0 to the bluetooth module. Infact, although this communication seems trivial, it is crucial for the scatternet implementation as it allows the bluetooth module to have access to the information packet to be broadcasted. In our implementation, this communication is facilitated via the UART transfer protocol. UARTs use just two pins while SPI devices require four, and given the fact that we had to use the SPI protocol for communication with the WiFi module and LED display, using two pins for this communication was a constraint to be fulfilled. The low-energy bluetooth module we planned to use for our scatternet implementation was compatible with UART, this simple transport protocol results in less software overhead without compromising on data throughput. It also results in a more cost-effective hardware solution, which is one of the key value propositions of our solution.

Lastly, the WiFi module also forms a very critical component of our implementation as it aids communication between our hardware components i.e master and slave units, and the central server. Transmission of a data packet across a wireless network such as WiFi includes additional information, or overhead due to the implementation of TCP protocol. Accommodating this data overhead over long periods of time could lead to extra cost incurred, and due to the fact that cost-effectiveness is of utmost important to our solution, we calculated data overhead due to transmission across WiFi over a year as part of our mathematical analysis.

3.2 Unknowns, Uncertainties and Testing

This project has several subsystems as part of the design and can be broken down into many different parts. Perhaps the best way to break down the desk reservation system is in terms of software and hardware.

Looking at the software aspect, this corresponds to the phone application through which the student can handle all of his or her reservations. This involves being able to reserve a desk, being able to cancel the reservation, view reservation history as well as the amount of penalties that have been incurred over the duration of the usage of the desk reservation system. Along with this is the central server which is key in order to support the large data volume, assuming thousands of students would be reserving desks in university libraries. This central server consists of various databases to keep track of all reservations and penalties. The phone app and master unit would then be communicating with the server using GET and POST requests in order to manipulate the central database and obtain information regarding reservations. From the previous section, it can be seen that we were able to visually depict our phone application's user interface and this was one of the things we implemented as it was feasible in the given conditions. The central server is part of the software aspect but it was not possible for us to implement this mainly due to the fact that we would have to purchase a subscription to a server powerful enough to support the data volume and the manipulation requests.

The obvious and much bigger aspect of this project is the hardware. The whole desk reservation system involves multiple master units and slave units with each of these units consisting of hardware subsystems like power, communication and control. It can be seen that circuit schematics have been made for both of these units as well as a PCB layout, but no further progress can be made without access to the PCB and soldering equipment in order to begin testing. Therefore whether it be the power subsystem which consists of a power switch, power adaptor and switch, the control subsystem which consists of the microcontroller or the communication subsystem which consists of the bluetooth scatternet, the physical implementation of these is much more difficult due to inaccessibility to the parts and a lab for testing.

If we were able to use the laboratory and other resources, then we would be able to proceed building the individual subsystems for the master and slave units. Having made some preliminary schematics for this, as well as a PCB layout, we would be able to place orders for the PCB and the hardware components. Following this, we would break down the schematics into smaller parts which will also allow us to implement unit testing on each hardware component, especially the purchased bluetooth and WiFi modules each part more easily. The first part that we will start to work on is most likely the bluetooth scatternet as this is the most important part of our project and what everything is built upon, in addition to being complicated to implement even from a software perspective. We would start working on this by creating a smaller version of the scatternet and testing it to gain knowledge of all the constraints, and from there add more layers to it. By working from the ground up and constantly unit testing various subsystems as we construct them, we would be able to effectively implement our desk reservation system design as described in our design document.

3.3 Safety and Ethics

The IEEE code of ethics which are made up of 10 guidelines as well as the ACM code of ethics provided a good framework for us when we consider the ethical aspects with regards to our project. From this IEEE and ACM ethics framework, there are a couple of ethical issues that are worth discussing.

A few ethical issues for our project correspond to one of the ACM code of ethics which says “Be fair and take action not to discriminate” [7]. The ethical issues here correspond to some of the parts or subsystems which could potentially be faulty, leading some students to not be able to use the reservation system properly which would be unfair to them. One example of this is if the status LED on the desk does not light up red as it has stopped working. This could confuse the student as to the status of the desk and worsen the experience. Another example is if the numeric keypad stops working properly. This would become very problematic as this is what the student uses to confirm the reservation ID and so if it doesn’t work the student won’t get access to the desk. There are other parts that if they stop working will create an unfair experience for some students who went through the correct process of reserving a desk through this system. We will therefore strive to overcome these issues by making sure that the parts are high quality and tested thoroughly before placed in use.

Another ethical issue that should be brought up is students potentially taking advantage of the system which relates to one of the ACM code of ethics which says “Be honest and trustworthy” [7]. An example of this is if a student decides to reserve a desk without really having the intention of using the desk but just as a backup and then does not show up to the reservation, causing someone else to miss out. An additional ethical issue which is extremely important involves the data security of the students who are using the phone application as well as their UIN when reserving the desks, which corresponds to one of the ACM code of ethics which says “Respect privacy” [7]. We will therefore make it a priority to make sure all of our systems keep the data of the students secure.

Safety is the other aspect of this project that we have to be aware of which also relates to the first IEEE code of ethics which says “to hold paramount the safety, health, and welfare of the public” and one of the ACM code of ethics which says “Avoid harm” [8][9]. The biggest safety issue with regards to this project involves our usage of the lithium ion battery. As usage of these batteries increases, they begin to overheat and in a low likelihood scenario, can catch on fire putting a student in a lot of harm if he or she is near the battery [9]. We therefore have to be aware of the safety hazard of the lithium ion battery in our design.

3.4 Project Improvements

When it comes to project improvements there are several that come to mind that we would start to look into given a year’s worth of time.

One of the improvements that can be applied for this project given the current context is the concept of social distancing when assigning desks to students trying to reserve them. Due to the novel coronavirus, social distancing has become a very important paradigm in combating the spread. Therefore something that we could potentially do to adapt our project as a further expansion and improvement with a year's worth of time is designing algorithms that assign desks to students while abiding by the guidelines of social distancing. Since social distancing requires a minimum of six feet between two people we will also have to take into account how this will physically affect the placement of the desks in the library. However, with this improvement of assigning desks based on social distancing, this system becomes a lot more useful as it can now be used in different scenarios and not necessarily completely stop working as a result of crises.

One of the main components of feedback for us after our design review was in regards to our communication system. Since our whole project is based on a bluetooth scatternet where there are three levels consisting of the master unit, slave level 1 unit and slave level 2 unit, we were advised to not exceed the limits of the bluetooth capabilities and stay well within the constraints. Given three levels with 6 to 7 branches per depth, although it is still possible for the communication to occur, this is putting a lot of pressure on the bluetooth network and so we thought we can improve upon this in two different ways. The first way involves lowering the number of connections in each level and simply increasing the number of levels. This will allow for less strain on the bluetooth capabilities and allow for a similar or greater number of total connections. The second way involves using two bluetooth chips for each level. This essentially allows us to double the amount of connections for each level and we also solve the problem of straining the bluetooth capabilities by making use of a second chip. This improvement will go a long way in the design of the system and allowing it to work more smoothly and efficiently.

The last improvement involves splitting the master units into two levels in the bluetooth scatternet. This can become a potential way to make the master units more cost effective because by splitting the master units into two levels it reduces the total number of master units needed. Along with this comes the reduction in use of other resources like the LED display and numeric keypad which are expensive items attached onto the master unit. This improvement will go a long way in reducing our already low costs even lower and so will be useful to implement given enough time.

In general with a year of extra time the improvements we can do for the project will make it a much more efficient, low-cost and adaptive desk reservation system.

4. Progress made on First Project

4.1 Circuit Schematics

Refer to **6.2.1 - Appendix B**

4.2 PCB layouts

Refer to **6.2.2 Appendix B**

4.3 Software Implementations

The software aspect of our project revolved around creating a program to handle a virtual queue system that would be uploaded to the microcontroller of the company's receiver, such that:

a) When a student taps his/her badge on the receiver, he/she gets added to the queue. This involves the *enqueue* functionality of a queue.
b) When a student taps the badge again on the receiver again after interacting with the company's recruiter, he/she is removed from the queue. This indicates the requirement of the *dequeue* function.
c) When a student is on top of the queue and has not tapped again for 15 minutes, he/she is automatically removed from the queue through a timeout, as this indicates that he/she may have requested to be removed from the queue. This indicates the requirement of the *dequeue* function as well, preceded by an automatic timeout.
d) If the queue is already full, the student is not added to the queue.

We implemented the code for a basic virtual queue which can handle upto 256 students, as an array of a struct "VirtualQueueEntry". This struct contains the UID (unique identifier of the RFID tag on the badge), and a Timer, that would store the instant when a student reaches on top of the queue. From this instant, after 15 minutes if the student has already not been removed from the list, this automatic timeout would ensure the dequeuing. The *enqueue* function would be called at each tap of a badge, and if an entry with this UID already exists in the virtual queue, the associated entry will subsequently be removed.

Due to the unavailability of a physical microcontroller and RFID receiver, we were unable to test this program, and manipulation of virtual queue due to taps on the receiver by a badge. We simulated the code on an online PIC microcontroller simulator to test the list manipulation functions instead.

For Pseudocode, refer to **6.3 - Appendix C**

5. References

[1] “How do hotel reservation systems increase efficiency and profits?,” SiteMinder, 27-Aug-2018. [Online]. Available: https://www.siteminder.com/r/hotel-distribution/hotel-direct-bookings/hotel-reservation-system-increase-profits/. [Accessed: 03-Apr-2020].

[2] Square, “How to Optimize Salon Scheduling,” Square. [Online]. Available: https://squareup.com/us/en/townsquare/effective-salon-scheduling-strategy. [Accessed: 04-Apr-2020].

[3] “AppGyver,” AppGyver. [Online]. Available: https://www.appgyver.com/. [Accessed: 09-May-2020].

[4] “Back to Basics: The Universal Asynchronous Receiver/Transmitter (UART) - Technical Articles,” All About Circuits. [Online]. Available: https://www.allaboutcircuits.com/technical-articles/back-to-basics-the-universal-asynchronous-receiver-transmitter-uart/. [Accessed: 09-May-2020].

[5] “Basics of UART Communication,” Circuit Basics, 11-Apr-2017. [Online]. Available: https://www.circuitbasics.com/basics-uart-communication/. [Accessed: 09-May-2020].

[6] “What is better? Lots of small TCP packets, or one long one?,” Game Development Stack Exchange, 01-Mar-2015. [Online]. Available: https://gamedev.stackexchange.com/questions/96945/what-is-better-lots-of-small-tcp-packets-or-one-long-one. [Accessed: 18-Apr-2020].

[7] “The Code affirms an obligation of computing professionals to use their skills for the benefit of society.,” Code of Ethics. [Online]. Available: https://www.acm.org/code-of-ethics. [Accessed: 03-Apr-2020].

[8] “IEEE Code of Ethics,” IEEE. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 14-Feb-2020].

[9] BatteryGuy, “Home,” The BatteryGuy.com Knowledge Base. [Online]. Available: https://batteryguy.com/kb/knowledge-base/safety-issues-with-lithium-batteries/. [Accessed: 04-Apr-2020].

6. Appendix

6.1 Appendix A (Bluetooth Module RV Table)

Requirements	Verifications
Requirement 1**:** It should be able to connect to seven bluetooth devices simultaneously within a 10m range.	a. Send an information packet to a bluetooth device stationed 10m away. b. Ensure that the packet is received by bluetooth module through connection to a microcontroller.
Requirement 2: It must be able to communicate with other bluetooth devices at ~2kBps.	c. Send an information packet of 2kB to a bluetooth device stationed 10m away. d. Ensure that the packet is received by the device in 1 s by connecting to a microcontroller.
Requirement 3: Should be able to communicate over UART protocol with a microcontroller.	a. Connect the bluetooth module to the UART input port of the microcontroller. b. Using a terminal, send data via the UART bus and verify whether data is received by bluetooth module.

6.2 Appendix B (First Project)

6.2.1 Circuit Schematics

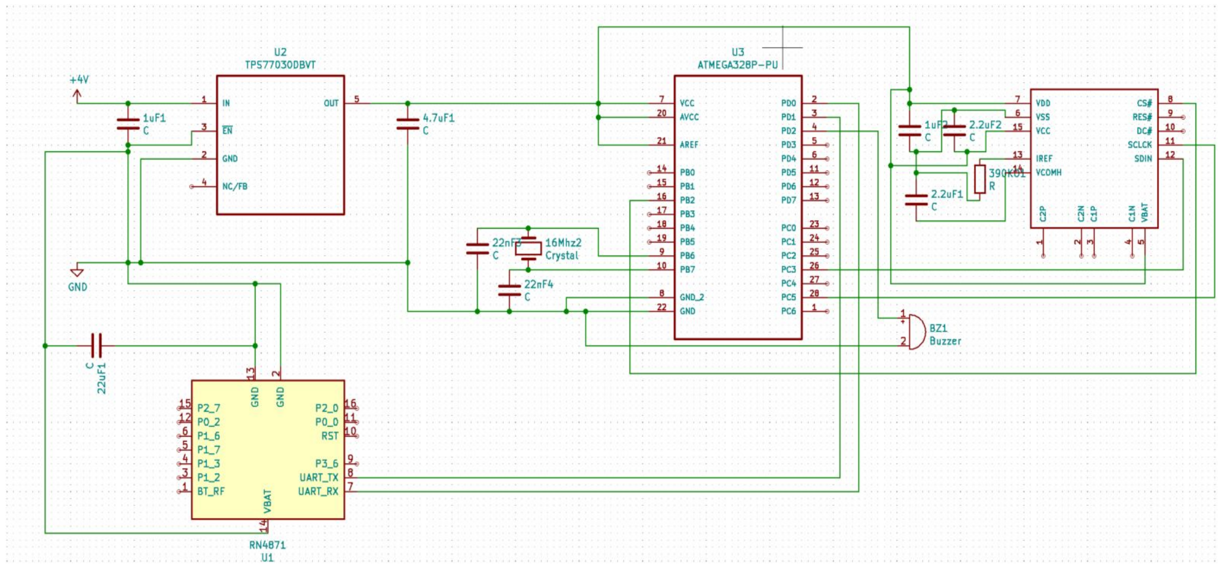


Figure 16: A complete schematic for the Electronic badge

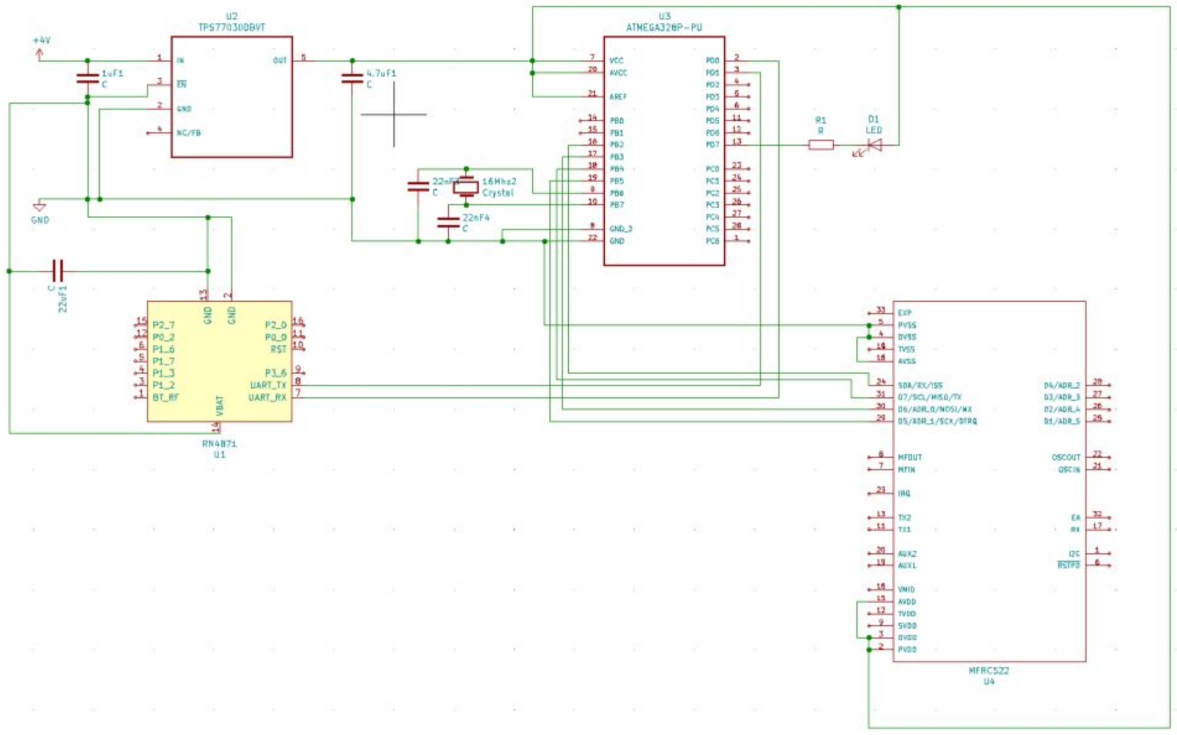


Figure 17: A complete schematic for the Electronic receiver

6.2.2 PCB Layout

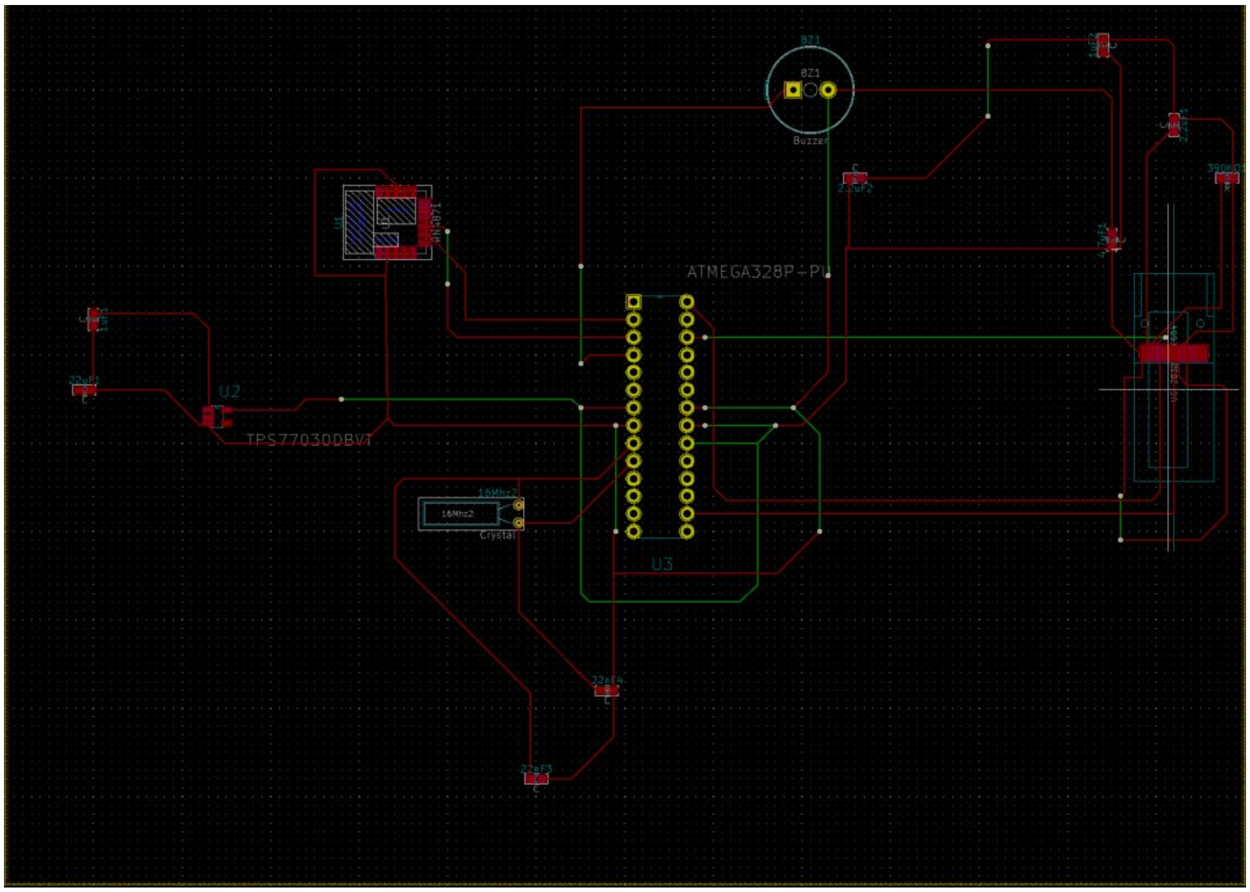


Figure 18: A PCB layout for Electronic receiver

6.3 Appendix C (Pseudocode)

```
typedef struct VirtualQueueEntry {
    int UID; # unique identifier of RFID tag
    int Timer;
}
struct VirtualQueue VirtualQueueEntry[256]

def enqueue(UID) {
    #The person spoke to the company and has tapped to be removed
    if ( UID in VirtualQueue)  {
        dequeue(UID)
    }
    else {
        NewEntry VirtualQueueEntry(UID, 0, 0)
        VirtualQueue.append(NewEntry)
    }
}
def dequeue(UID) {
    Entry <- getEntry in VirtualQueue.search(UID)
    VirtualQueue.pop(Entry)
}

def deleteTimeout(UID) {
    # The entry hasn't responded within 15 minutes and needs to be deleted
    for Entry in VirtualQueue:
    if (Entry.Timer > 900 seconds) {
        VirtualQueue.pop(Entry)
    }
}
```