BAR WATCH

By

George Oliver

Joshua Levy

Pranith Bottu

Final Report for ECE 445, Senior Design, Spring 2021

TA: Xihang Wu

05 May 2021

Project No. 52

Abstract

The purpose of Bar Watch is to provide the bar patron and staff a straightforward way to see their blood alcohol concentration (BAC) and be able to dispense drinks in a contact-free manner so long as the patron isn't too intoxicated. The system consists of three components: a self-serve drink dispenser, a watch band with three LEDs signifying the patrons BAC range, and a companion app for the bar staff. The ability for the patron to easily see their BAC range on the watch band, dispense drinks, and the ability for bar staff to monitor the intoxication levels of the patrons was successfully implemented. Implementing the ability for the dispenser to automatically update the amount of alcohol the patron consumed proved to be troublesome due to issues with the companion app. In this report, we outline the development process of the hardware and software behind Bar Watch's three components.

1 Introduction

1.1 Purpose

Responsible drinking has been a concern for a long time. In 2020, despite COVID restrictions, drunk drivers caused an average of 29 deaths per day and 28% of all traffic deaths [1]. In 2016, over one million drivers received DUIs and that is just 1% of the self-reported cases [1]. As a result, 43 of the 50 states have adopted the Dram Shop Law, which can hold bars liable for drunk driving accidents [1]. Aside from DUIs, bars have also been a major concern for new COVID-19 cases in the country due to their larger foot traffic. Thanks to the Dram Shop Law, patrons can now hold bars liable and file Dram Shop Cases for reasons ranging from recklessness to intentional conduct [2]. Therefore, it is in the best interest of the bars and the public to find a way to actively track alcohol consumption while providing a COVID-safe experience.

Our goal with Bar Watch is to facilitate safe drinking and also offer a more COVID-safe experience. It is a reusable accessory that bars can use to monitor their patrons' alcohol consumption and purchases. This device is given to customers upon entry and is able to interact with different dispensers and track the user's purchases. This has the added benefit of contact-free delivery and reduced waiting times for consumers. Each device is connected to a central database through a wireless communication module. The database contains the different prices and alcohol information, and tracks the user's consumption. If patrons reach a dangerous level of alcohol consumed as determined by their physical features and time between drinks, the bar is alerted to offer ride services if necessary. When a customer leaves, the device is returned and sanitized for use again. This endeavor would serve as a big first step toward responsible drinking and making the experience COVID-safe.

1.2 Background

There have been efforts to track alcohol consumption and automate dispensers, but none have sought to combine the two into an ergonomic, multipurpose wristband. With COVID still a threat, our solution's features have the potential to provide more value.

With regards to dispensers, Smartender is the closest product that mirrors this notion of automated dispensing [3]. Taproom bars also offer cards that patrons can use at dispensers without contacting a bartender. However, what they do not have is a BAC-tracking system to notify both the bar and patron about their alcohol consumption. Such features exist separately in downloadable apps that do not properly hold individuals accountable. By notifying bar employees, our product gives them the tools to ensure those under the influence are properly guided to potential ride services or otherwise taken care of appropriately. In addition, a patron's information is added to the wristband and database until he or she exits the bar. In the event of any altercation or emergency on premise, the necessary information is ready without delay.

To put it more succinctly, our product seeks to unify two different products (automated dispensing and BAC-tracking) into one solution, while incorporating a payment feature to further reduce human interaction. Furthermore, we intend to add a safety feature that allows for patrons to use the wristbands as a call for help. A single pushable button quietly alerts the bar employee of a person's distress. Our goal in designing Bar Watch is to make drinking at bars a safer interaction and take into consideration the COVID threat present today.

1.3 Visual Aid

The illustrations seen in Figures 1 and 2 illustrate the prototypes we built for the wristwatch and dispenser. Figure 3 shows what the entire Bar Watch system looks like and the blue lines indicate the communication channels that exist between the different components. We have two-way communication from our Android App to and from our MySQL database, wristwatch, and dispenser.

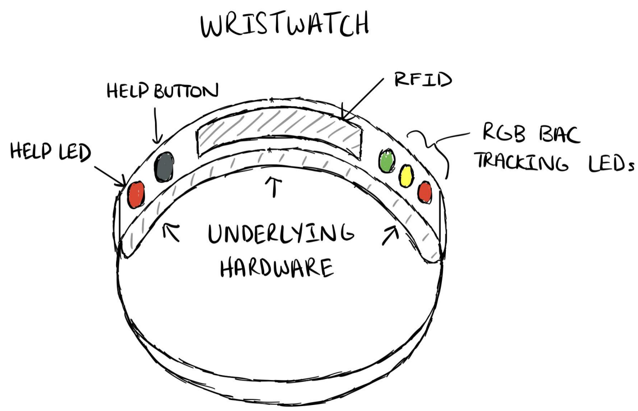


Figure 1: Wristwatch Physical Design

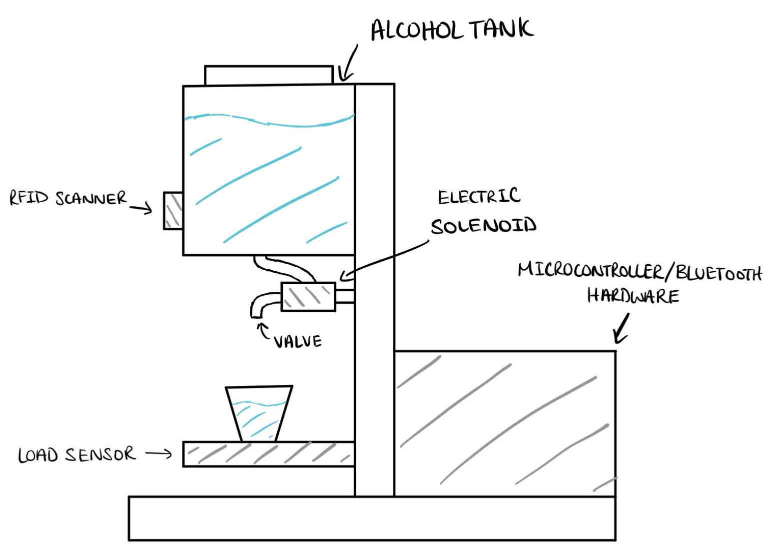


Figure 2: Dispenser Physical Design

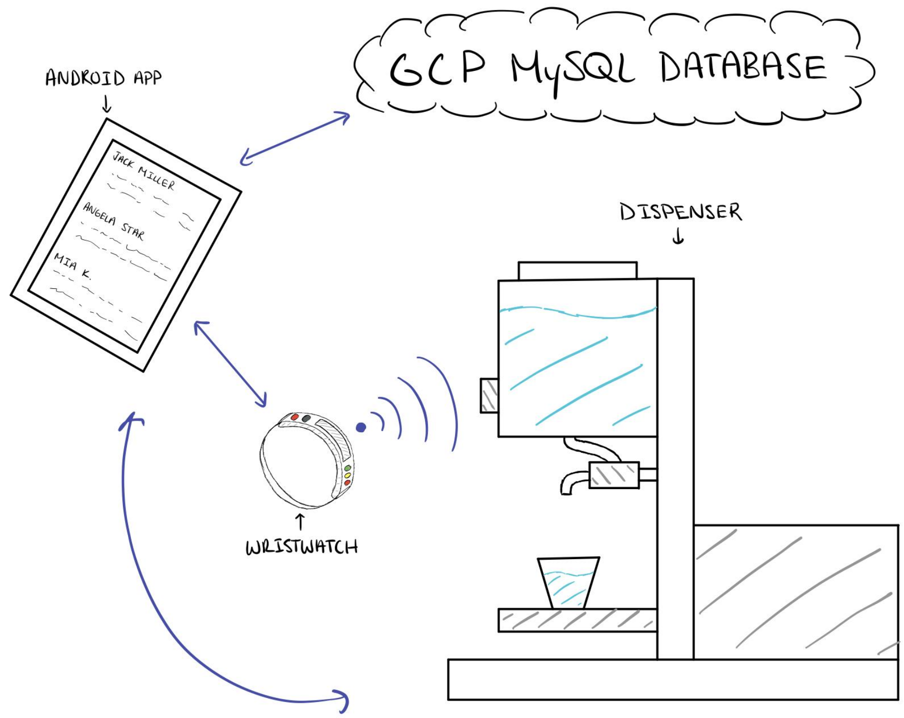


Figure 3: Bar Watch System Design

1.4 High-Level Requirements

- Measure volume of alcohol poured within 1-5% of actual volume and up to 16 fluid ounces poured to enable Blood Alcohol Content/Concentration (BAC) estimation, based on customer biometrics and volume of alcohol.

- When a customer uses the wristwatch at the dispenser or presses the "Help" button, the purchase/status is reflected in the database under the customer's entry within 1 second.

- Provide visual feedback via a 5 second long LED indication after purchase to customers and customer summary to bar staff within 5 seconds of request.

2 Design

2.1 Dispenser

2.1.1 Dispenser Design Procedure

There are a few different possibilities on how to design the dispenser. Overall, the dispenser must accomplish the following goals: scan and recognize the correct RFID associated with each wristwatch, dispense liquid into the user's cup, determine the amount of liquid poured into the user's cup, provide visual feedback to the user, and send data to the tablet app.

The most important part chosen for the dispenser is the choice of microcontroller that can easily communicate with all the circuit components and accomplish these goals that are provided. Our choice was to use the ATMega328P microcontroller, which is the same microcontroller that is used in many different boards such as Arduino Uno and Sparkfun RedBoard. This was chosen due to ease of setting up the microcontroller for the circuit. Once a specific bootloader was installed on the chip, it could easily interface with the Arduino IDE and program the desired program onto the chip. This also allowed for easier debugging as the breadboarded circuit could just be run by an Arduino Uno during testing.

This microcontroller choice also determined the choice of the other parts for the dispenser. A few parts such as the RFID RC522 and the Bluetooth HC05 chips were chosen due to their ease of interaction with the ATMega328P. These parts only require a few of the ATMega328P pins, and have prewritten libraries that allow us to successfully integrate all components of the dispenser together. The RFID RC522 came with two different IDs as well, which made it possible to test that the sensor could read and differentiate between two different IDs. The HC05 chip was used to verify that the correct data was read, as it could send data to a remote serial monitor that did not need to be connected to this portion of the circuit. It also is able to interface with the HC05 in the wristwatch as well as Bluetooth connection in the Android app.

The initial thinking on how to measure the volume of water poured into the cup was to use a flowmeter. A flowmeter is a device that can connect to the tubing and measure liquid flow as it dispenses. In theory, this device would most likely be used in a future design of the dispenser. There were issues with using this part due to the machine shop not having as much experience with setting up this part in conjunction with the solenoid needed to control the water flow. Instead, a load cell device is placed underneath where the cup is set before pouring a drink. This load cell will then zero the scale and measure the mass of the liquid poured. Using the density of the liquid and it's per fluid ounce alcohol content, this mass is converted to an alcohol level and then used to calculate the BAC of the user. Although this method is slower due to having more data to convert, it is the more reliable setup for the first time prototype of the dispenser.

2.1.2 Dispenser Design Details

Continuing in further detail about the load cell portion of the dispenser, the load cell was chosen as the 5 kilogram load cell and HX711 Analog to Digital Converter. This combination worked well with the design because it could easily measure the maximum amount of liquid poured, and the HX711 came embedded on a board that could directly interface with the ATMega328P microcontroller. From the load cell datasheet [4], the formula to calculate the measured force of the load cell is:

$$Measured\ Force = A * Measured\ \frac{mV}{V} + B\ (offset)\ [4]$$

A is a constant that is equal to the $\frac{Capacity}{Rated\ Output}$. The rated output of the load cell is $1.0 \pm 0.15\frac{mV}{V}$, and the capacity is 5 kg. Additionally, each load cell needs to be calibrated before use, which is where B, the offset is introduced. Because this variable can be calibrated, it will not be considered in the risk analysis. Instead, a range of which forces can be measured is shown in Table 2.1.2.

	Max Rated Output ($\frac{mV}{V}$)	Max Measured Force (kg)
Low end	0.85	4.25
Average	1.00	5.00
High end	1.15	5.75

Table 1: Load Cell range

For testing purposes, water will be used, which can be converted from kg to fluid ounces using the following formula in the worst case scenario.

$$4.25\ kg\ *\ 1000\ \frac{g}{kg}\ *\ 1\frac{cm^3}{g} * 29.5735\frac{fl\ oz}{cm^3} = 125687\ fluid\ oz$$

As can be seen, the load cell can easily tell weights up to 16 fluid ounces. To save on costs, a 1 kg sensor will be able to reach that maximum weight as well. Because this is a bar environment, it may be better to go with the 5 kg sensor if it is more tolerant to people pushing down on it. However, to determine if it is precise enough, more analysis must be done.

The outputs from the load cell will be connected to an HX711 chip, which will be the interface from the load cell to the microcontroller. It will take the analog differential voltage output of the load cell and convert it to a digital 24 bit number that corresponds to the mass measured. This chip has a small typical input noise range of 50 to 90 nanovolts. [5] Additionally, based on the load cell input, the circuit will be measuring the input differential between the excitation and the output voltages, which are changed by the amount of force on the load cell itself. Under typical 5V supply which will be used in this project, the full-scale differential input covers up to 20 mV in the worst-case scenario [5], which is well above the differential output of the load-cell circuit.

Next, these components and their wirings will be explained and the purpose behind each part. A full schematic and PCB layout of the circuit is included in **Appendix A**. **Figure 4** shows the electronics behind the mechanical portion of the dispenser

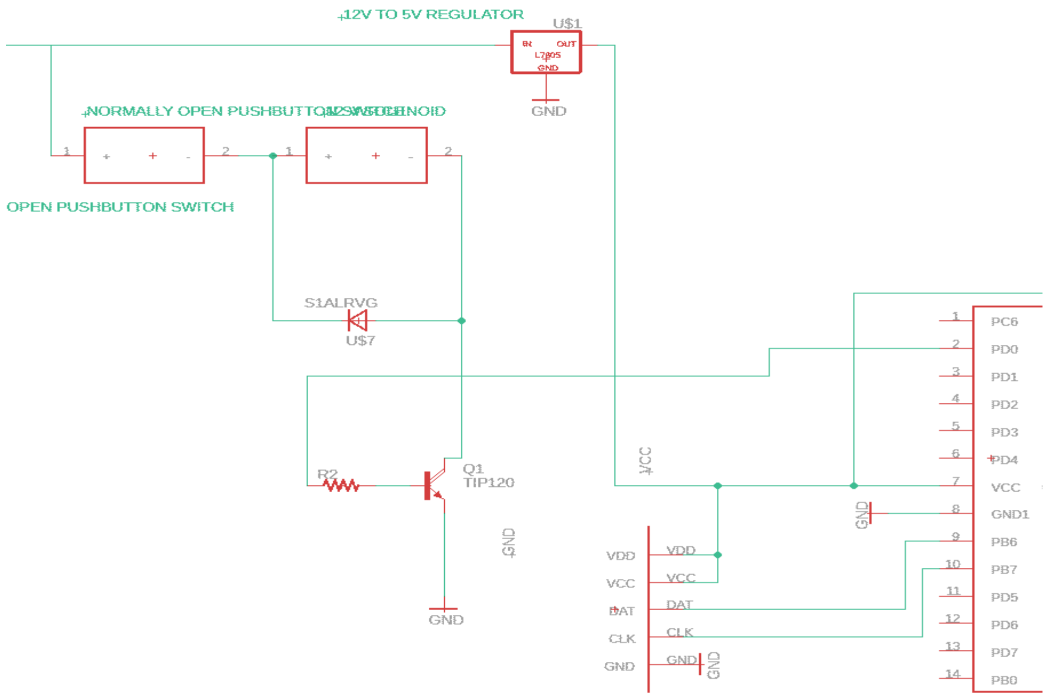


Figure 4: Mechanical Portion of Dispenser

To ensure that each drink poured is associated with the correct RFID card, there was a need to lockout the circuit until an ID is scanned. This is also needed to prevent customers from receiving free drinks from the dispenser whenever they press the dispense button. One engineering solution to this issue is to use a transistor as an electronic switch, which can complete the path to ground when driven into saturation [6]. Otherwise, the transistor is in cutoff, and the solenoid will not open even when the pushbutton is depressed. The transistor is also able to protect the lower voltage components from getting a large current surge and burning out. There is a flyback diode in parallel with the solenoid to protect it from sudden surges in current. Finally, the HX711 ADC is included to allow constant load cell readings to and from the design. This can constantly read the weight on the load cell if needed; however, it is better design to only take readings when a customer is using it. The microcontroller is able to zero the scale after the customer places their cup on the load cell. Then, the customer gets a set time limit to pour their drink. In proper use, the customer must wait until the LED indicates that it is alright for them to remove their drink. This will be after it takes its final measurement and sends it to the Android app via HC05 Bluetooth connection.

The other portion of the design includes the RFID RC522 scanner (not pictured), HC05 Bluetooth module, RGB LED, and 12 to 5V voltage regulator (not pictured) in **Figure 5.**

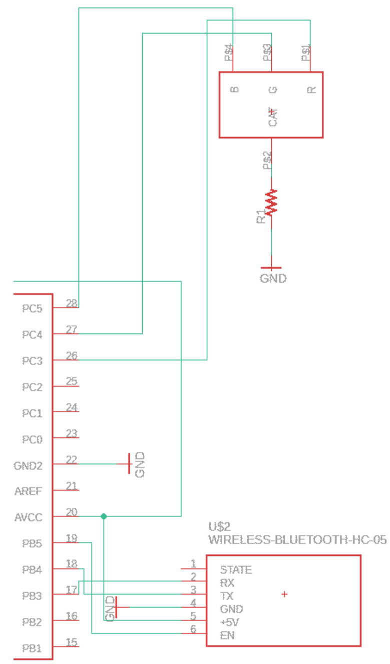


Figure 5: Wireless Communication and User Feedback Portion of Dispenser

These are all the lower voltage components besides the HX711 ADC. As stated above, the RFID scanner is able to scan and differentiate between different user IDs. When scanned, the user will receive visual feedback from the RGB LED changing from red to green. At this time, the microcontroller will store the RFID associated with the tag scanned. The RGB LED will remain green for 30 seconds as the user is able to pour what they want into their cup. After 30 seconds, the microcontroller will request the current weight on the load cell, which it then sends, along with the RFID, over Bluetooth via the HC05 chip. The HC05 chip will only be used to send signals in this design. At this time, the LED will briefly change to a blue color to indicate it is sending a signal, and afterwards flash red for five seconds to indicate to the customer they can take their drink and leave.

This was the initial design which would definitely be changed for the next prototype. Due to inexperience with working with microcontrollers, this layout was not optimized correctly for the design requirements. In the schematic, there are two different power levels, 12V which is required to be connected to the solenoid for proper operation, and 5V which is required for most everything else. However, where this PCB will fail is that the HC05 component must have 3.3V signal to it's TX and RX pins for proper operation. In testing with the breadboard, the circuit will not send proper information if receiving a 5V signal from the microcontroller. Additionally, the RFID scanner was not included in this portion of the PCB, although it is a necessary component in our final breadboarded design. Finally, the ATMega328P pins chosen were not ideal for each component. The two pins designated to interface with the HX711 ADC are typically reserved as crystal pins, although they can also be used as digital input/output pins. To actually program these as I/O pins using the Arduino IDE is quite difficult and should only be done if there are no additional pins available, which is not the case in this design. Furthermore, the RFID library and interface assumes that specific pins on the ATMega328P will be used, which in this design are not available. This necessitated a long search to find an older library that can actually interface with the ATMega328P through other pins.

2.2 Watch Band

2.2.1 Watch Band Design Procedure

The main goal of the watch band is to provide visual feedback to the patron about the level of intoxication that they are at. With that in mind, there were two possible approaches that were considered: a watch with a 1.5'' LCD display that shows the users BAC and the amount of drinks they purchased and a watch with a red, yellow, and green LED. The decision was made to go with the watch that just shows status using three LEDs because when a person is drunk, their ability to think clearly is clouded, so having a simple LED array with red, yellow, and green would make their current status easily understood. In addition to ease of use, this allows the watch to have greater battery life because the LCD display consumed 0.125 watts whereas the LED lights we considered would consume 0.06 watts leading to greater battery life. Our watch design also features an RFID tag that the user taps against the dispenser's RFID reader in order to dispense drinks as long as they're not in the red status.

The next major design consideration with the watch band was which technology to use to get it to communicate with the database and retrieve the user's BAC. Two approaches considered were to use a Wi-Fi module and get the device connected to the SQL database directly and the second approach would be to use a bluetooth module and create a high-level app that will relay the information to the database and also relay information to the watch. The Wi-Fi module approach would require the watch to be programmed with the bar's Wi-Fi information, which would be tedious and restrict scalability. The bluetooth module approach would easily allow the bar staff to use their existing smartphone or tablet to just connect to the watch via bluetooth and have everything working, so this was the approach taken.

2.2.2 Watch Band Design Details

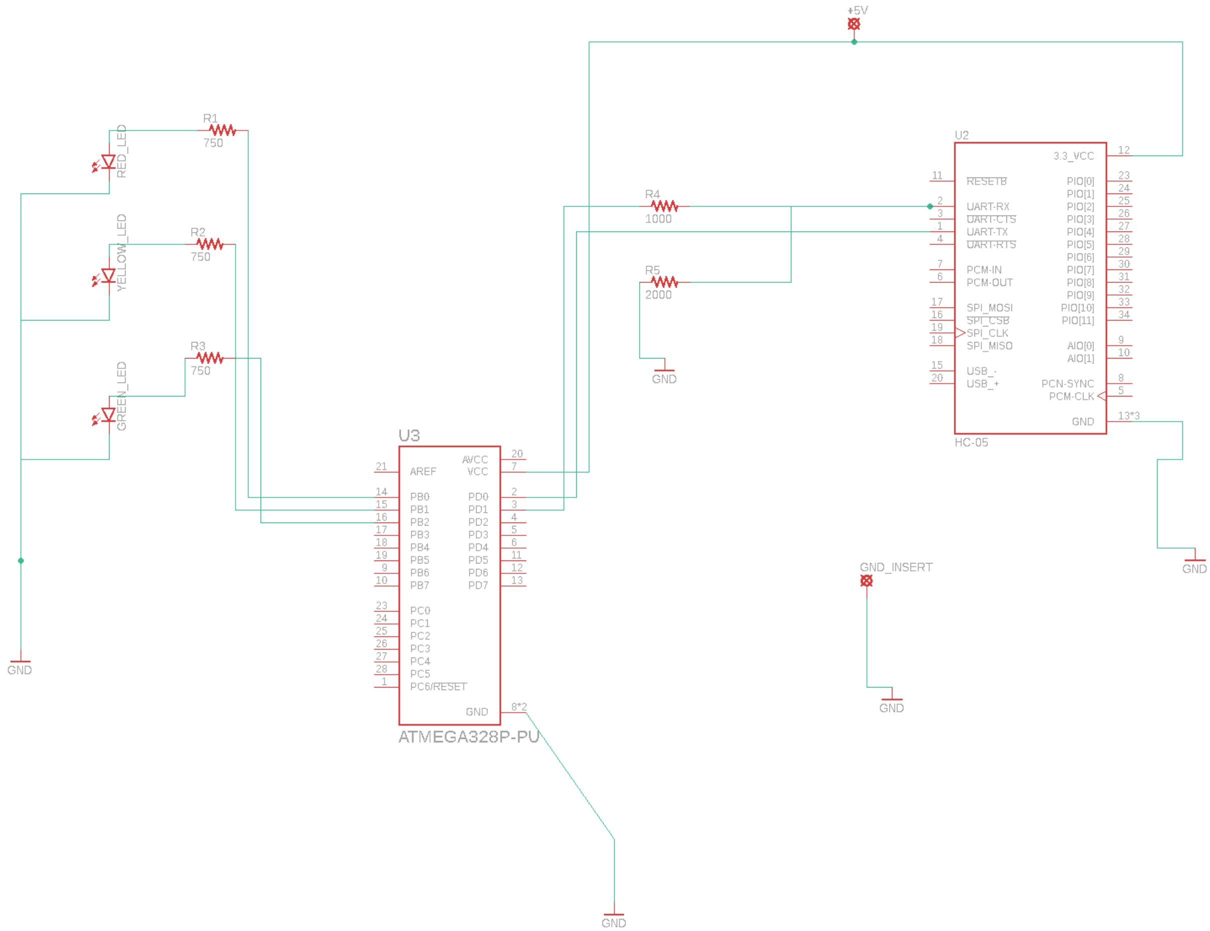


Figure 6: Circuit Schematic for Watch Band

The watch band needs to be able to last at least 7 hours, so the choice of battery had to be carefully considered. From the datasheets of our components, we found the maximum current draw of each component at 5V as shown in **Table 2**.

Component	Maximum Current Draw
ATMega328P with 8 MHz Internal Clock	5.5 mA
HC-05 Bluetooth Module	40 mA
3 LED Lights	60 mA
Total	105.5 mA

Table 2: Component Maximum Current Draw

As we can see, the total current draw of the components we chose is 0.1055 amps. In order for us to have 7 hours of battery life, we need to have a battery that is at least 0.8 Ah, so we ended up using a 1 Ah battery.

We chose to use the ATMega328P as our microcontroller for the watch band because it allowed us to upload code onto it using the Arduino development environment using Arduino development boards that we already had. The Arduino development board is also based on the ATMega328P, so it offered a very similar I/O pins to our development boards which allowed us to test the feasibility of our circuit on the development board before flashing onto the microcontroller. The ATMega328P also has a programmable serial Universal Synchronous/Asynchronous Receiver Transmitter (USART) with an on-chip oscillator, so this would allow us to interface with a bluetooth module.

We chose the HC-05 bluetooth module because it communicates using the USART standard at a 9600 baud rate, so it would communicate perfectly with our ATMega328P. Another advantage that this module has is that there was a surface-mount version of it available, which we needed in order to minimize the size of our watch.

The design of the circuit for our watch band is simple. We receive a stable 5 volts and 1 amp from the battery charger/booster module and this 5 volts powers the ATMega328P and the HC-05 bluetooth module. We then connect the three status LEDs to the ATMega328P digital output pins and use a resistor in series with the LED to limit the current through the LED and prevent burnout. The HC-05 bluetooth module has a USART receive pin which we connect to the USART transmit pin of the ATMega328P so that serial data from the ATMega328P can be given to the bluetooth module and thus gives the ability of the watchband to send information to the companion application. Since the HC-05 USART receive pin operates on 3.3V logic and the ATMega328P USART transmit pin transmits based on 5V logic, a voltage divider had to be done using a 1,000 and 2,000 ohm resistor in order to drop the voltage from 5V to 3.3V. The HC-05 USART transmit pin is then connected to the ATMega328P receive pin so that whatever is received from the companion app can be transmitted to the ATMega328P.

The logic that is on the ATMega328P is simple as well. The companion app sends out an integer every 5 minutes regarding the BAC level of the patron to the watch. Receiving a ‘1’ corresponds to being in the green status, which means that the patron’s BAC is at or below 0.08%. Receiving a ‘2’ corresponds to being in yellow status, which means that the patron’s BAC is above 0.08% and at or below a 0.1%. Receiving a ‘3’ corresponds to being in red status, which means that the patron’s BAC is above a 0.1%. The ATMega328P simply receives the patron’s BAC level and lights up the corresponding LED.

2.3 Bar Watch App

2.3.1 App Design Details

The purpose of the companion app is to receive data from the dispenser about the amount of drinks a specific patron has consumed and keep track of it and keep the patrons BAC level up-to-date on their watch. The patrons BAC is calculated by using the Widmark equation [7] which states:

$$\% BAC = (Alcohol\ Consumed * 5.14 / Weight * (.73\ for\ men\ or\ .66\ for\ women)) - 0.015 * Hours\ since\ drinking$$

This information is collected when the user is being handed their watch and inputted into the database on the initialized user screen as shown in **Figure 7.**

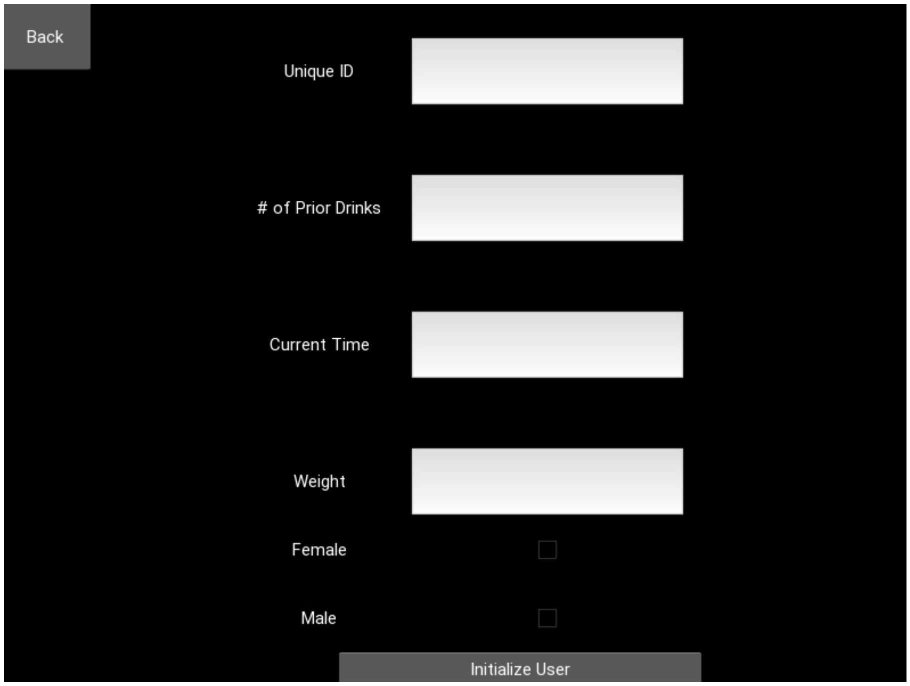


Figure 7: Initialize User Screen

When the information is entered in the following fields, the BAC is calculated and all the information is used in a query to the database. The person's watch also receives their first update. The companion app also allows the bar staff to manually change any field that they want in the event that information is wrong as seen in **Figure 8**.

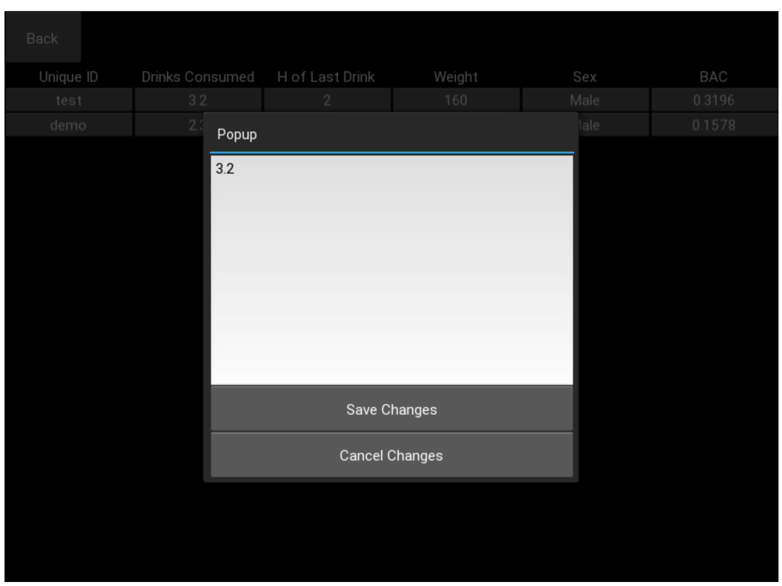


Figure 8: Manual User Editing

when a user's information is manually edited, the app automatically gets the unique ID of the person in question and generates an update query to the database.

The app does a few handy features in the background via multithreading. The patron's watch has 3 LEDs: green corresponding to at or below 0.08%, yellow corresponding to above 0.08% and below or at 0.1%, and red being above 0.1%. To keep their watch up-to-date, every five minutes, a thread spawns in the background and queries the patron's BAC in the database and converts it to a '1' corresponding to the green status, a '2' corresponding to the yellow status, or a '3' corresponding to the red status and the app then sends that number to the watch band via bluetooth so that the watch band can update it's LED. From the Widmark equation, we can also see that the person's BAC drops by 0.015% every hour, so in the same threads that spawns every five minutes, the app generates a query that decreases the BAC of every patron in the database by 0.00125%.

The last multithreaded task that the app was supposed to do was have a thread that runs in the background and is spawned when the app is opened that constantly checks the bluetooth stream from the drink dispenser to see if any patron has just recently bought a drink and if they did, the app would update the database with the amount of alcohol that patron purchased and an updated BAC. However, we couldn't get this thread to work. This is likely because of hardware limitations of the tablet we tested with. The CPU on the tablet we tested only had two cores, so it could only execute two threads at a time, which prevented us from having this third thread that constantly checks the dispenser bluetooth stream.

3 Design Verification

To verify the design a variety of different tests were conducted to ensure that the product is working as intended. One of the first major tests to ensure proper communication between the dispenser and product was to send the RFID of a scanned tag via Bluetooth to another device and examine it via Bluetooth Module. **Figure 9** shows this connection on the actual tablet device and the RFID tag that was sent.

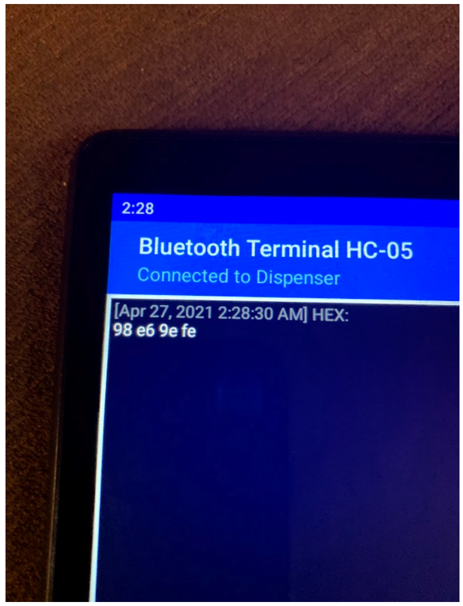


Figure 9: RFID Tag on the Tablet

This simple test can verify a few different parts functionalities. First, it ensures that the design can properly differentiate between two different RFIDs, which is necessary for the database to properly update each customer that is using the dispenser. Second, it proves functionality of the HC05 component and that it can be used furthermore to send load cell data to the database to properly calculate and update BAC levels.

This test was further improved upon to show that the updated database can be updated within 1 second, which was one of the high-level requirements of the project. By adding an LED color to indicate that the HC05 is in a transmitting state, the amount of time it takes to transmit the data can be measured. In our testing, this was done via turning the RGB LED of the dispenser circuit blue. We were unable to actually measure the length of time it took for it to transmit because the LED would blink quite quickly, but it was well within the 1 second timeframe that was desired.

Another crucial design requirement was the setup of the user database and bar employees being able to interact with it. **Figure 10** shows the finished application database that the bar employees can add, modify, and delete users in the database. To test this, several users were added to the database, along with different entries in each category. Then, the user of the app was able to test out all the required features and ensure that they properly worked.

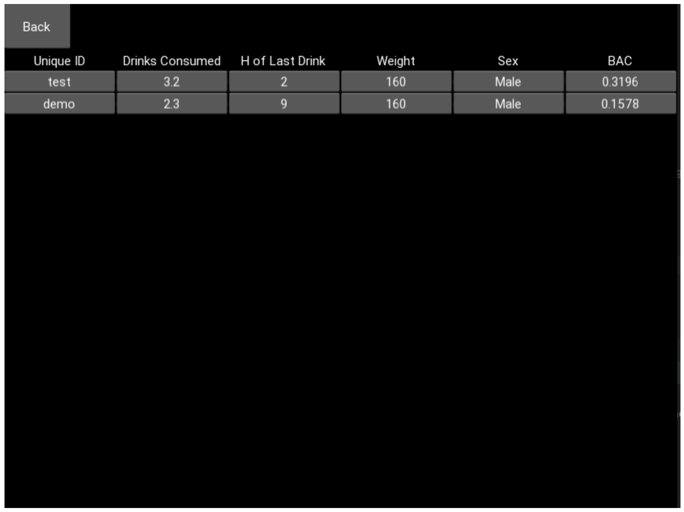


Figure 10: Database with Current Customers

One important requirement that was unable to be verified was the functionality of the load cell and HX711 chip. During our testing process, the HX711 chip had something burn out on it, which prevented any sort of verification of any part of the project related to the weight. To modify this and ensure that the device can still have some sort of functionality, the user is no longer able to determine how long of a pour they receive in their drink. Instead, when the dispenser is activated by scanning an RFID device, it will dispense for a set amount of time (15 seconds). Through frequent measurements, we found that the device would almost always dispense around 114 grams of water, which is close to ⅓ of a standard beer. Before the load cell device burned out, we did compare the mass it measured to the mass that was measured by a standard kitchen scale. Generally, the load cell was within 1 to 3 grams of the actual scale, but we do not have any documentation of this to provide.

The other design requirement that was not able to be verified was any functionality of the help button on the user's watch. Due to time constraints and other important aspects of the project being more pertinent, the help button was not implemented in the final design. In the future, we would like to include these features, and would use a similar test as the dispenser Bluetooth verification test to verify all design choices.

4 Costs

4.1 Labor

The average annual starting salary for Illinois Electrical Engineering graduates is $79,714 [8]. Assuming that the engineer works 40 hours a week, for 52 weeks a year, the average hourly salary is $38.32. From the date of approval, February 11th, to the last date of the semester for this course, May 6th, there are 12 weeks. Because there are three engineers, the total cost for labor this semester is:

$$12\, weeks \ * \ 40 \frac{hours}{week} * 38.32 \frac{\$}{hour} * 3 \ = \ \$55,180.80$$

4.2 Parts

Price of all these parts are based on the prototype. **Table 3** shows the parts and their costs.

Part	Amount	Price per part	Total Price
HC-05 Bluetooth Module	2	$7.99	$15.98
Lithium-Ion Battery	1	$7.29	$7.29
Charging Module	1	$15.95	$15.95
Load Cell and HX711	1	$12.99	$12.99
ATmega328P	2	$4.46	$8.92
RFID Reader and Chip	1	$6.99	$6.99
12V 0.5 inch Solenoid	1	$25.99	$25.99
Dispenser Wall to 12V DC Converter	1	$8.95	$8.95
Normally Open Push Button Switch	1	$4.94	$4.94
L7805 5V Voltage Regulator	1	$0.30	$0.30
RGB LED	5	$0.90	$4.50
TIP120 Darlington Transistor	1	$0.64	$0.64
Total			$113.44

Table 3: Parts Price and List

5 Conclusion

5.1 Accomplishments

We prototyped our system with the functionality to accurately track Blood Alcohol Concentration (BAC) and update the database, wristwatch, and dispenser with information according to a patron's actions within a simulated bar environment. Under normal operating conditions, the wristwatch's RGB LEDs were properly modified based on the volume of alcohol consumed. Furthermore, we managed to update our database with the amount of alcohol dispensed and its price during every patron's interaction with our dispenser. Like we planned, the RFID tag and scanner were used to distinguish different patrons and ensure only valid IDs were allowed. Lastly, we successfully managed to install LEDs on the dispenser to provide patrons visual feedback to show that their actions were registered.

5.2 Future Work

Future design iterations would involve improving our PCB design, finishing our implementation of the "help" button, and controlling multiple dispensers.

With regards to PCB design, our inexperience with advanced Arduino programming led to a faulty dispenser PCB. Specifically, we did not realize that two of the pins were not I/O ports like we believed. As a result, it was not possible to pass in the data needed to the database. There was another hiccup when we realized the RFID needed a direct connection to the PCB that we could not safely provide. If given more time, we would have rearranged the I/O pin layout to be more accommodating of the data transmission pipeline. With that time, we would have rearranged the PCB's components to hold the RFID. Lastly, we could switch out the wristwatch PCB for a smaller, more form-fitting PCB, for a better aesthetic experience for patrons.

The "Help" button was not fully implemented due to time constraints. Our database supports its use-case and has threads that could take on that data transmission line. The hardware, however, was lacking. Therefore, a key future step would be to update the wristwatch PCB to hold a spare red LED that it would turn on when prompted by button press or database command.

The final future step would be to control multiple dispensers. Our goal with this prototype was to show this was doable with one dispenser. In an actual bar, there would be multiple dispensers corresponding to different drinks. Having this implemented would simulate an exact bar environment.

5.3 Ethical Considerations

There are two ethical concerns we had when developing our project. The first concern was having the BAC-tracking LEDs on the watches light up. This gives rise to the situation where a patron may be taken advantage of due to their LED making it evident to others that he or she is intoxicated. The implementation of the "help" button would resolve this issue since the patron would be able to call for assistance in this event. The "help" button can also be used to alert staff to any other dangerous situation that may be occurring in the bar.

The other ethical concern we had was the usage of Personal Identifiable Information (PII) data in our database. This was resolved by only taking a patron's height and weight since that was all we required in order to track BAC levels. In the future, with more security, we might want to explore the tradeoffs of having more personal information such as card details. This could be a kind of collateral in the event the bar needs to hold a patron accountable. For our use case, we just used height and weight to avoid any conflicts with PII data.

6 References

[1] "How COVID-19 has Impacted DUI Rates and Arrests in 2020?," *Breatheeasyins.com*, 03-Sep-2020. [Online]. Available: https://www.breatheeasyins.com/how-has-covid-19-impacted-dui-rates/. [Accessed: 14-Feb-2021].

[2] Nolo, "Dram shop laws: Can I sue a bar after an alcohol-related accident?," *Nolo.com*, 10-Jul-2014. [Online]. Available: https://www.nolo.com/legal-encyclopedia/dram-shop-laws-can-i-sue-bar-after-alcohol-related-accident.html. [Accessed: 15-Feb-2021].

[3] "Smart Bar USA," *Smartbarusa.com*. [Online]. Available: https://www.smartbarusa.com/. [Accessed: 17-Feb-2021].

[4] "3133- Micro Load Cell (0-5 kg) - CZL 635," 13-May-2011. [Online]. Available: https://www.robotshop.com/media/files/pdf/datasheet-3133.pdf. [Accessed: 04-Mar-2021].

[5] "24-Bit Analog-to-Digital Converter (ADC) for Weigh Scales," *Sparkfun*. [Online]. Available: https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/hx711_english.pdf. [Accessed: 04-Mar-2021].

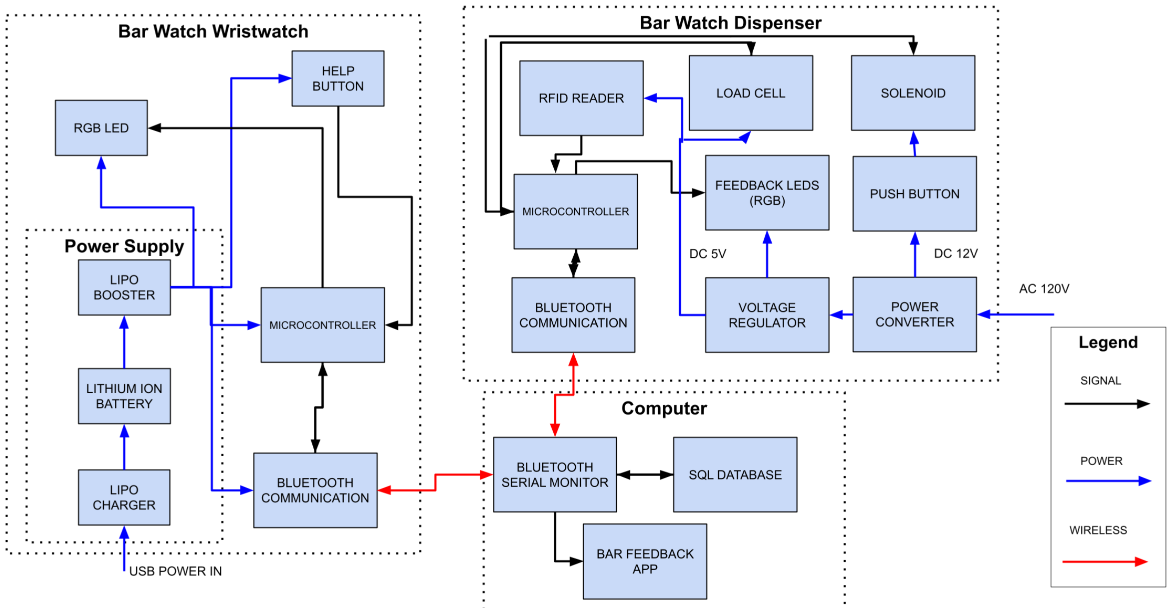
[6] Chris, "Controlling A Solenoid Valve With Arduino," BC Robotics, Jul-2015. [Online]. Available: https://bc-robotics.com/tutorials/controlling-a-solenoid-valve-with-arduino/. [Accessed: 06-May-2021].

[7] Gambonelaw, "The Widmark Formula and Calculating Your BAC Level," Gambone Law. [Online]. Available: https://gambonelaw.com/faqs/the-widmark-formula-and-calculating-your-bac-level/. [Accessed: 06-May-2021].

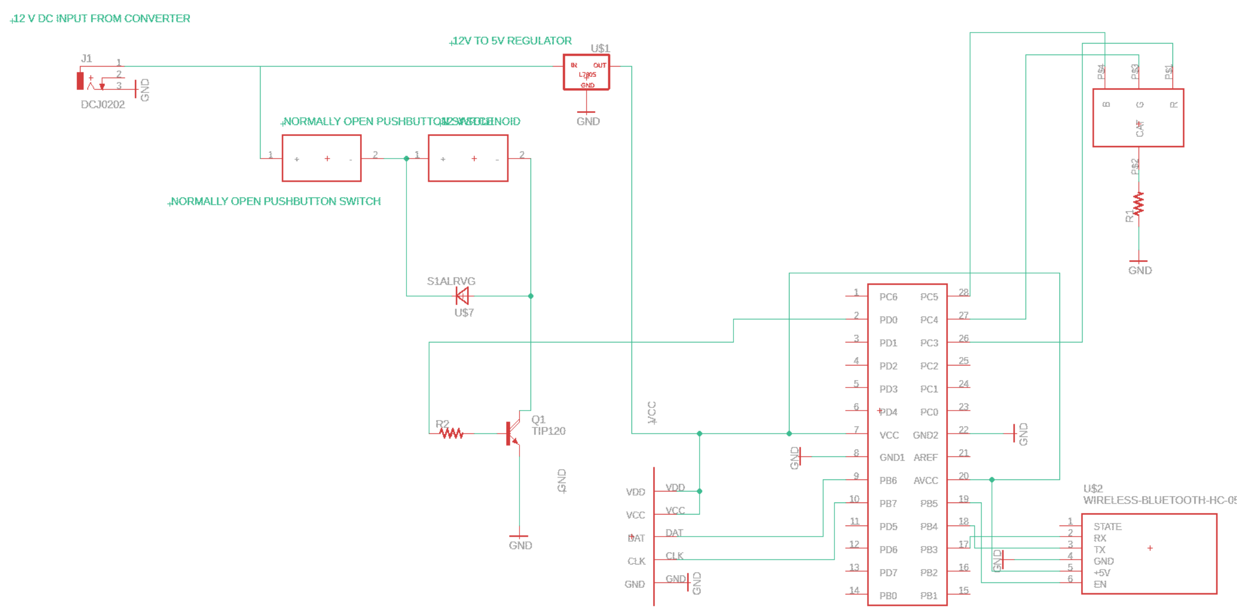
[8] Grainger Engineering Office of Marketing and Communications, "Salary Averages," Electrical & Computer Engineering | UIUC. [Online]. Available: https://ece.illinois.edu/admissions/why-ece/salary-averages. [Accessed: 02-Mar-2021].

[9] S. Al-Mutlaq, N. Seidle, and Alex, Load Cell Amplifier HX711 Breakout Hookup Guide. [Online]. Available: https://learn.sparkfun.com/tutorials/load-cell-amplifier-hx711-breakout-hookup-guide/all. [Accessed: 06-May-2021].

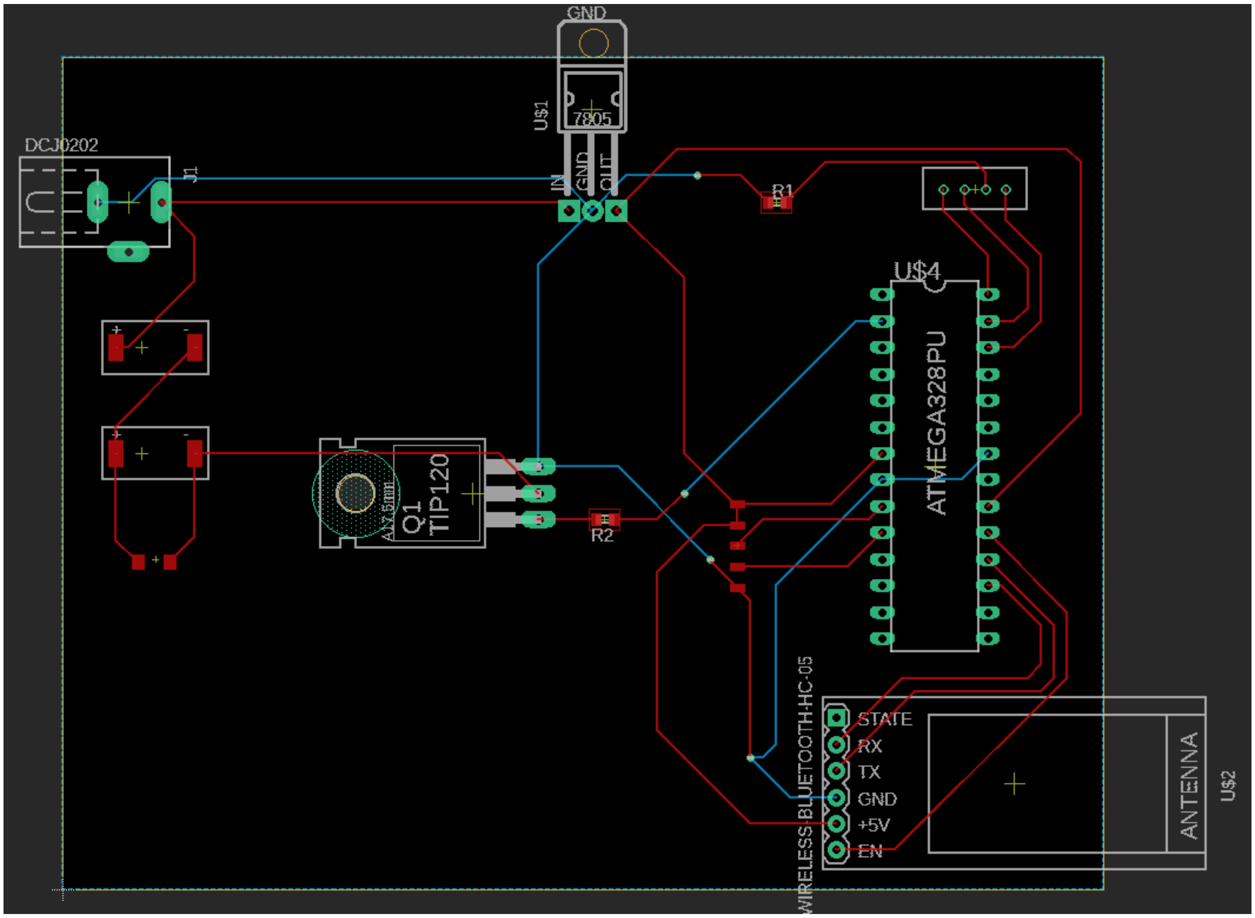
Appendix A Block Diagram



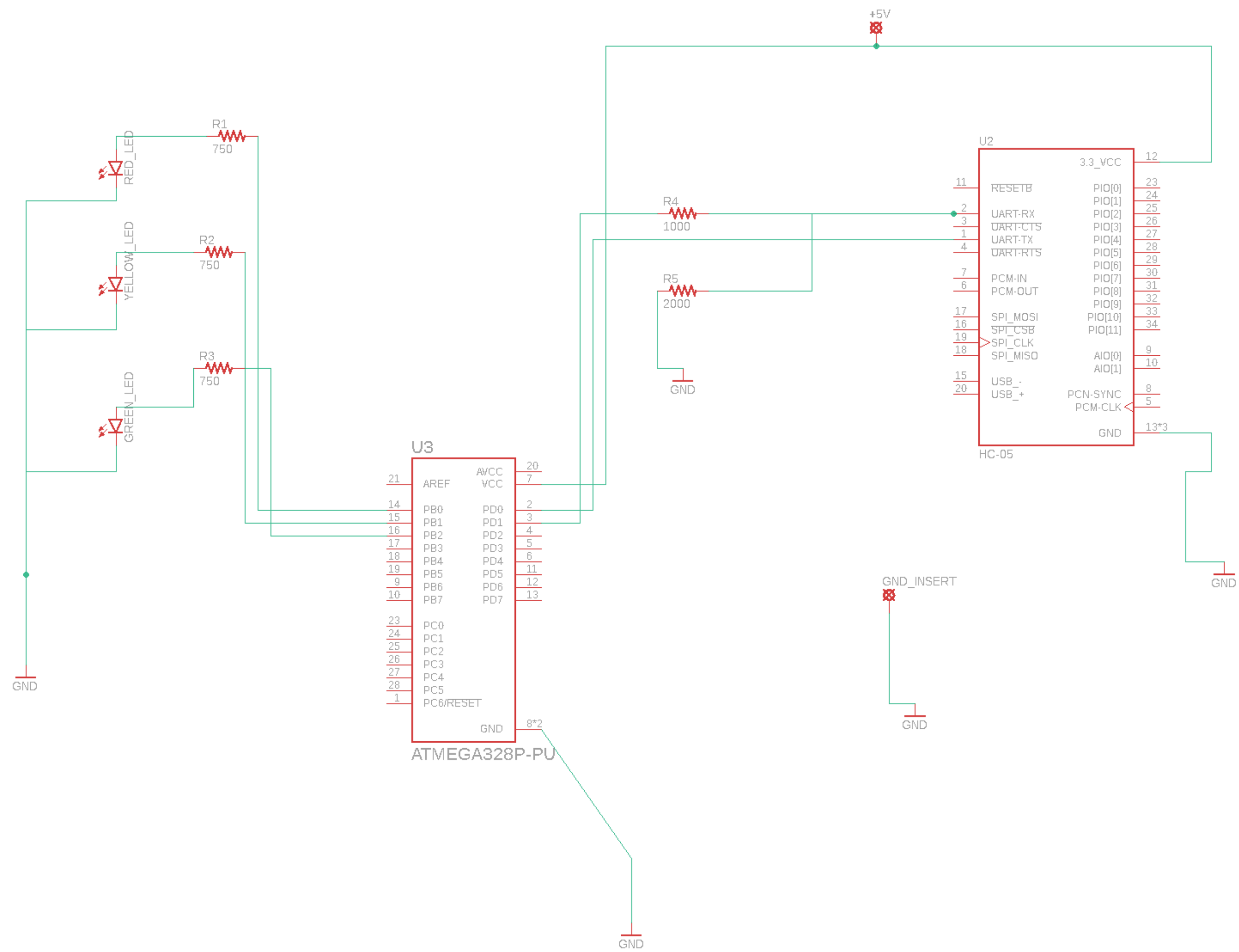
Appendix B Circuit Schematics and PCB layouts



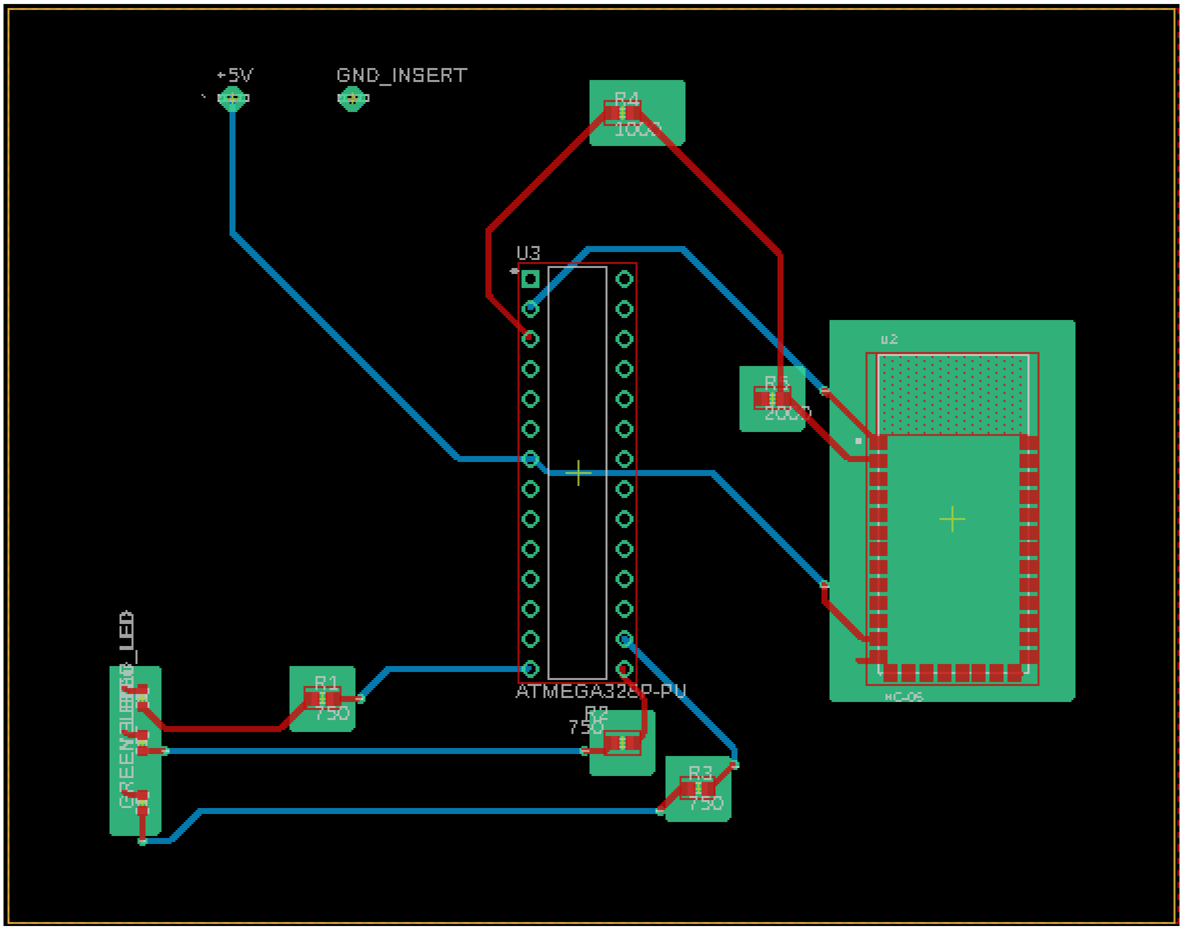
Dispenser Complete Schematic



Dispenser PCB



Wristwatch Complete Schematic



Wristwatch PCB

Appendix C Requirement and Verification Tables

1. LiPo Charger/Booster

Requirements	Verifications
1. Fully charge battery	a) Use a micro-usb charger to charge the battery until the LED indicates it's fully charged b) Check that the battery is outputting 3.7V
2. Reliably output 5V +/- 5% at 1 Amp +/- 5%	a) Measure the output voltage from the booster b) Ensure that it's outputting 5V with a current of 1A

2. Microcontroller

Requirements	Verifications
2. Receive and verify RFID data via the USART communication	a) Scan RFID tag b) Use a serial monitor to make sure that the correct UID of the RFID tag is being displayed
3. Send and receive information to the bluetooth module via USART communication.	a) Send a few test characters from a smartphone via bluetooth b) Use a serial monitor to make sure that the correct character is received by the

	bluetooth module

3. Load Cell & Load Cell Amplifier

Requirements	Verifications
1. Read weight with a range of error within +/- 0.0015kg	a) Calibrate our weight cell using a simple Arduino circuit b) Weigh an object of known weight c) Use the Arduino IDE to check if the reading from our load cell is within the range of error
2. Transmit the weight to the microcontroller via USART	a) Weigh an object of known weight b) Use a serial monitor to see if the weight is communicated successfully

4. Bluetooth Module

Requirements	Verifications
1. Send and receive information to the bluetooth module via USART communication	a) Send a few test characters from a smartphone via bluetooth b) Use a serial monitor to make sure that the correct character is received

5. Case

Requirements	Verifications
1. Passes IP26 environmental standards	a) Touch wristwatch and dispenser with bare hands b) Spray case with water for 30 sec c) Ensure parts function as intended

6. Data Transmission

Requirements	Verifications
1. Data is transmitted to computer from wristwatch/dispenser	a) Press ‘Help’ button on wristwatch and verify signal is received by computer b) Use wristwatch on dispenser c) Verify weight sensor data & wristwatch user information is received by computer
2. Data is transmitted to wristwatch from computer	a) Manually set user’s BAC status to ‘red’ and ensure wristwatch LED is ‘red’

7. Data Storage

Requirements	Verifications
1. Data is added to database	a) ‘Activate’ wristwatch and ensure user information is added to the database

2. Data is updated in database	a) Use wristwatch on dispenser and ensure user information is updated in database
3. Data is deleted from database	a) 'Deactivate' wristwatch and ensure user information is deleted from the database

8. Logic Processing

Requirements	Verifications
1. When wristwatch is scanned by dispenser, must update the user's information in database	a) Use wristwatch on dispenser and ensure user information is updated in database
2. When 'Help' button is pressed, must update the user's information in the database and flash LED on wristwatch 'red'	a) Press 'Help' button on the wristwatch and ensure database updates with this information b) Ensure LED on wristwatch flashes 'red'

9. RGB LEDs

Requirements	Verifications
1. RGB lights must change depending on user's BAC levels	a) Modify the user's database entry for amount of alcohol consumed and ensure all 3 RGB lights are functional based on corresponding BAC level

10. Help Button

Requirements	Verifications
1. Button must be pressed for at least 3 seconds	a) Accidental press of the 'Help' button does not activate the 'Help' subroutine
2. Wristwatch must flash red	a) Press 'Help' button on the wristwatch and ensure the LED light flashes 'red'
3. Database must be updated	a) Press 'Help' button on the wristwatch and ensure the user's database entry is updated
4. Computer must display an update	a) Press 'Help' button on the wristwatch and ensure the computer displays a notification for the bar staff