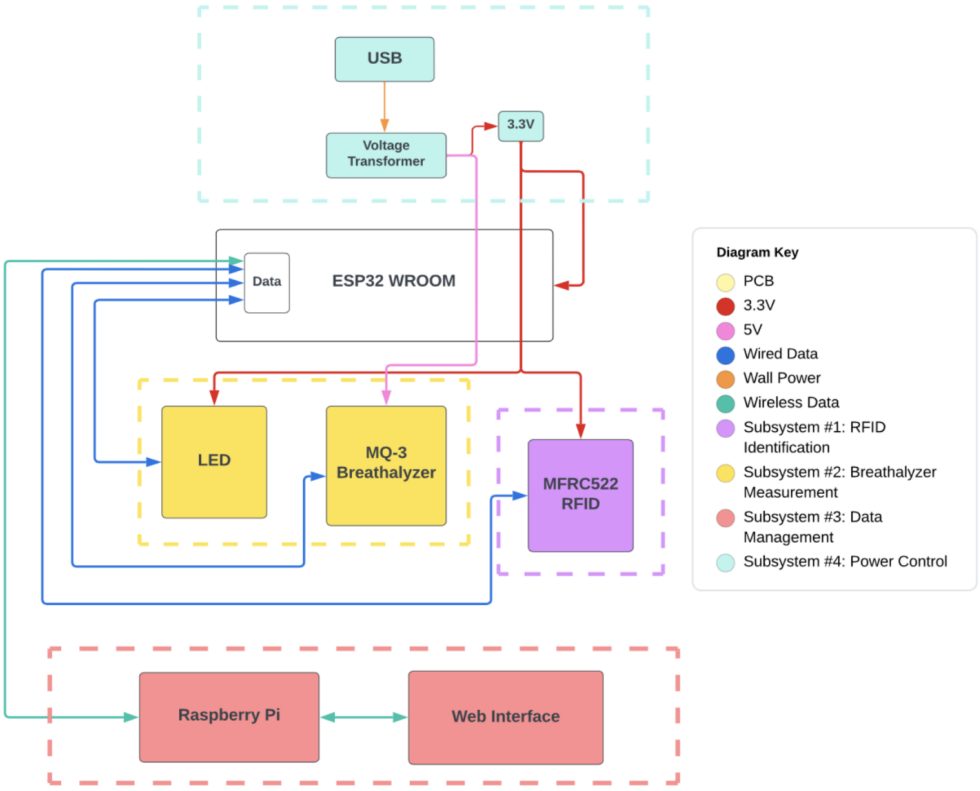
Akash Patel Lab Notebook

ECE 445

Date: January 29

For the TipsyTracker Project Proposal, our group worked together closely to develop the block diagram that included the RFID subsystem, Breathalyzer measurement subsystem, and Data management subsystem. We had multiple brainstorming sessions where we shared our ideas, discussed the feasibility of different approaches, and evaluated the pros and cons of each subsystem.

We all shared our expertise and worked together to ensure that the different subsystems complemented each other and functioned effectively in the overall system. Each team member had a specific role to play, and we ensured that everyone had a clear understanding of their responsibilities.



Snapshot of block diagram

Date: February 1

Our group identified the problem of irresponsible drinking, which is a common issue, especially among university students, that leads to significant negative consequences, including deaths caused by excessive alcohol consumption. Our solution, TipsyTracker, is a comprehensive system that employs RFID-enabled wristbands/cards, breathalyzer devices, a Raspberry Pi server, and an ESP32 microcontroller to measure the blood alcohol content (BAC) levels of partygoers/patrons and encourage responsible drinking habits.

TipsyTracker notifies guests about their BAC levels and sends reminders to them to check their levels, as well as alerts the host if a guest's BAC level exceeds a predetermined limit.

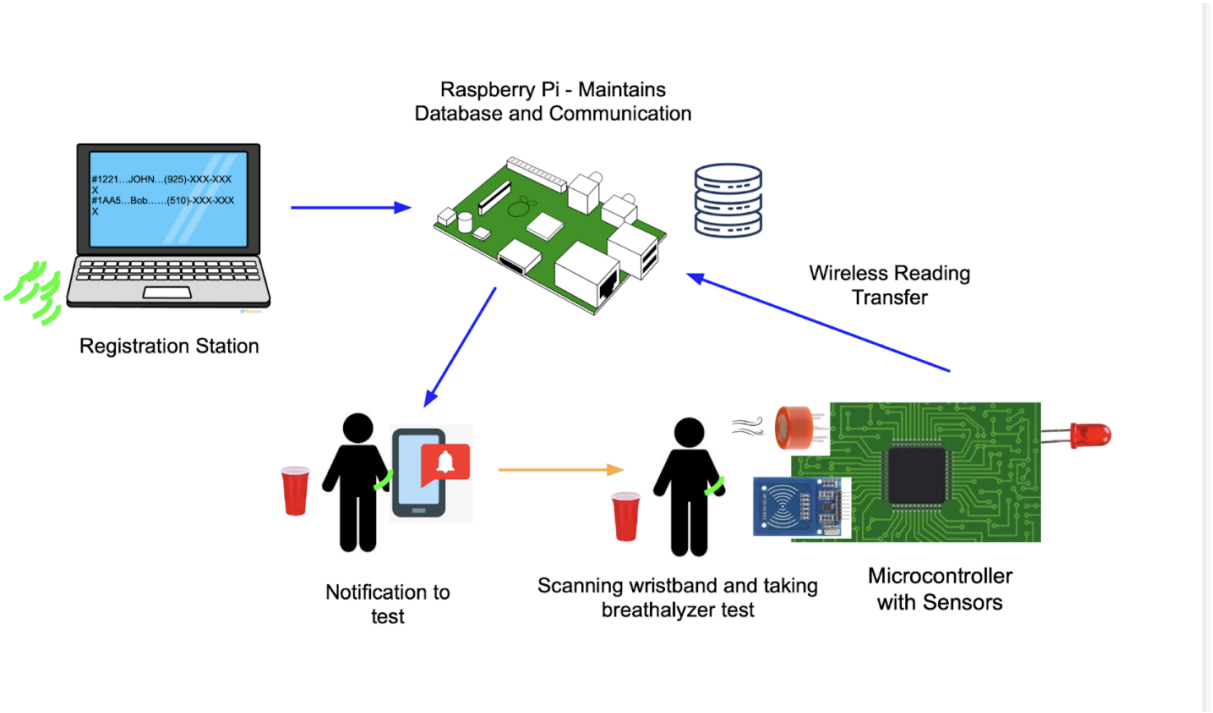
I specifically worked on identifying two subsystems. The Breathalyzer Measurement Subsystem, which involves the MQ-3 sensor, is responsible for measuring the BAC levels of partygoers/patrons by using a breathalyzer device. It is connected to the ESP32 microcontroller and communicates with the RFID identification subsystem to ensure that the test results are associated with the correct partygoer/patron. The Data Management Subsystem, powered by the Raspberry Pi server, is responsible for handling the communication between the device and the registration station and sending notifications to partygoers/patrons and the host. It hosts the necessary software and databases, handles data storage, analysis, and management, and sends reminders to partygoers/patrons at set intervals to test their BAC levels and notifies the host if necessary.

By working together as a group and collaborating effectively, we were able to develop a comprehensive solution that addresses the issue of excessive alcohol consumption and encourages responsible drinking habits. The TipsyTracker system is an effective solution to promote awareness and responsibility among partygoers and patrons.

Date: February 3

During the development of the TipsyTracker project proposal, I assisted the group by editing and refining the document to ensure that it was clear, concise, and well-organized. I worked closely with the team members to ensure that the document was coherent and that the language used was appropriate for the intended audience.

In addition to editing the proposal, I also drew a visual aid to help illustrate the system architecture and explain how the different subsystems interacted with each other. The visual aid helped to clarify the system design and made it easier for team members to understand the overall concept of the TipsyTracker system.

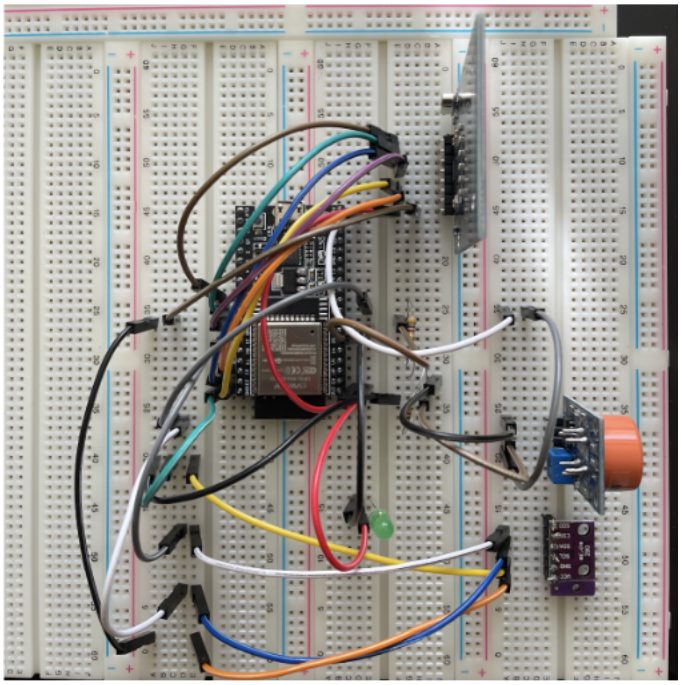


Snapshot of visual aid

Date: February 7

On February 7th, the TipsyTracker development team made significant progress by successfully breadboarding the entire system design. This was a crucial step in the development process as it enabled us to test the functionality of the different subsystems and identify any potential issues that needed to be addressed before moving forward with the manufacturing process.

Breadboarding the entire system also allowed the team to ensure that all components were properly connected and that the system was functioning as expected. By leveraging the expertise of each team member, we were able to address any issues that arose during testing and ensure that the system was operating as intended.



Breadboard of our project

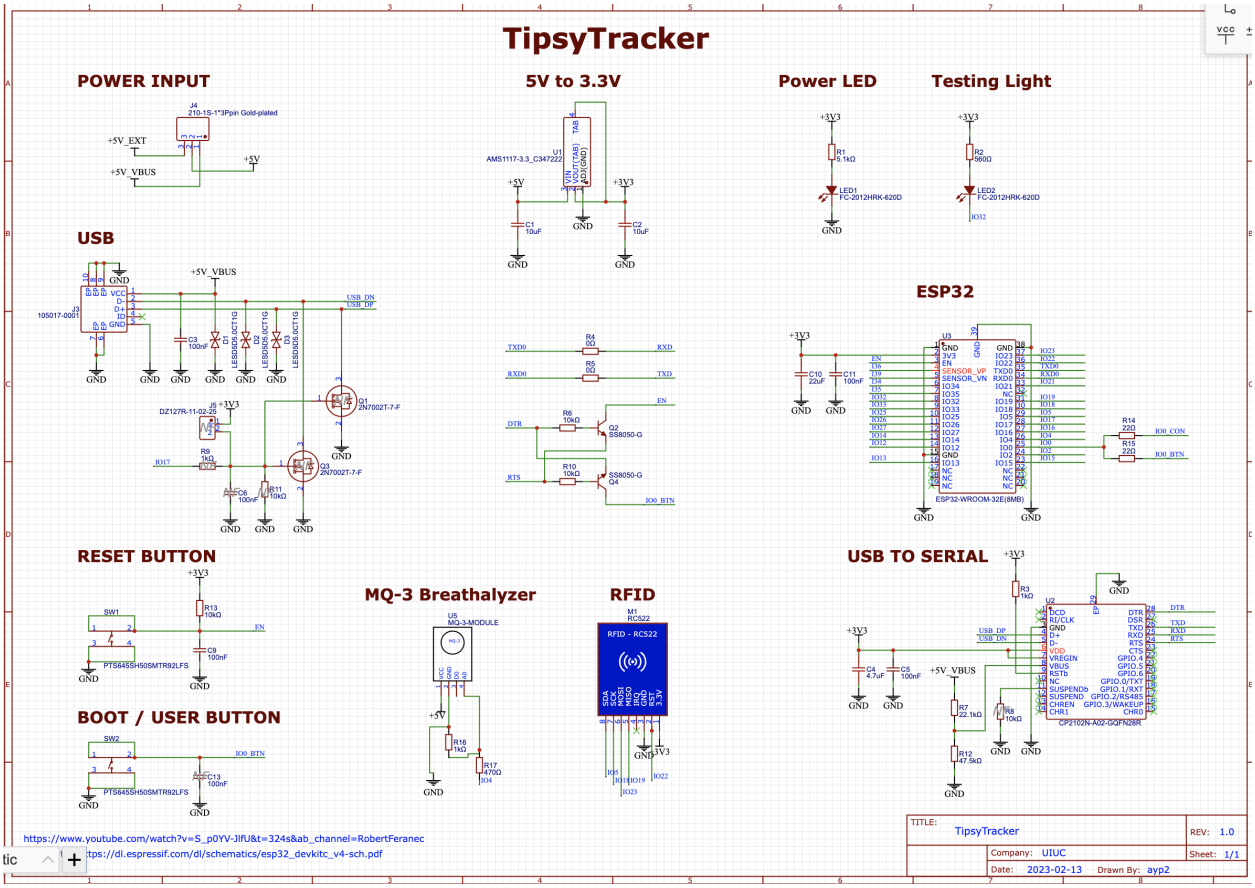
Date: February 13

I am pleased to report that during the development of the TipsyTracker system, I was able to make significant progress on the Printed Circuit Board (PCB) schematic. Specifically, I was able to complete the first schematic for the PCB, which included important components such as the power input, 5V to 3.3V regulator, USB-to-UART converter, MQ-3 sensor, RFID sensors, and other key parts.

The power input section of the schematic was critical to ensure that the device was able to operate properly and that there were no issues with power delivery. The 5V to 3.3V regulator was important to provide the correct voltage for the components, and the USB-to-UART converter enabled the device to communicate with the Raspberry Pi server.

The MQ-3 sensor and RFID sensors were also critical components that allowed the system to measure BAC levels and identify partygoers/patrons accurately. Other parts, such as capacitors and resistors, were included to ensure proper signal conditioning and noise reduction.

Overall, the completion of the first schematic for the PCB was a significant milestone in the development of the TipsyTracker system. By leveraging my skills in PCB design and component selection, I was able to ensure that the system was well-designed and capable of meeting the requirements of the project.



PCB schematic

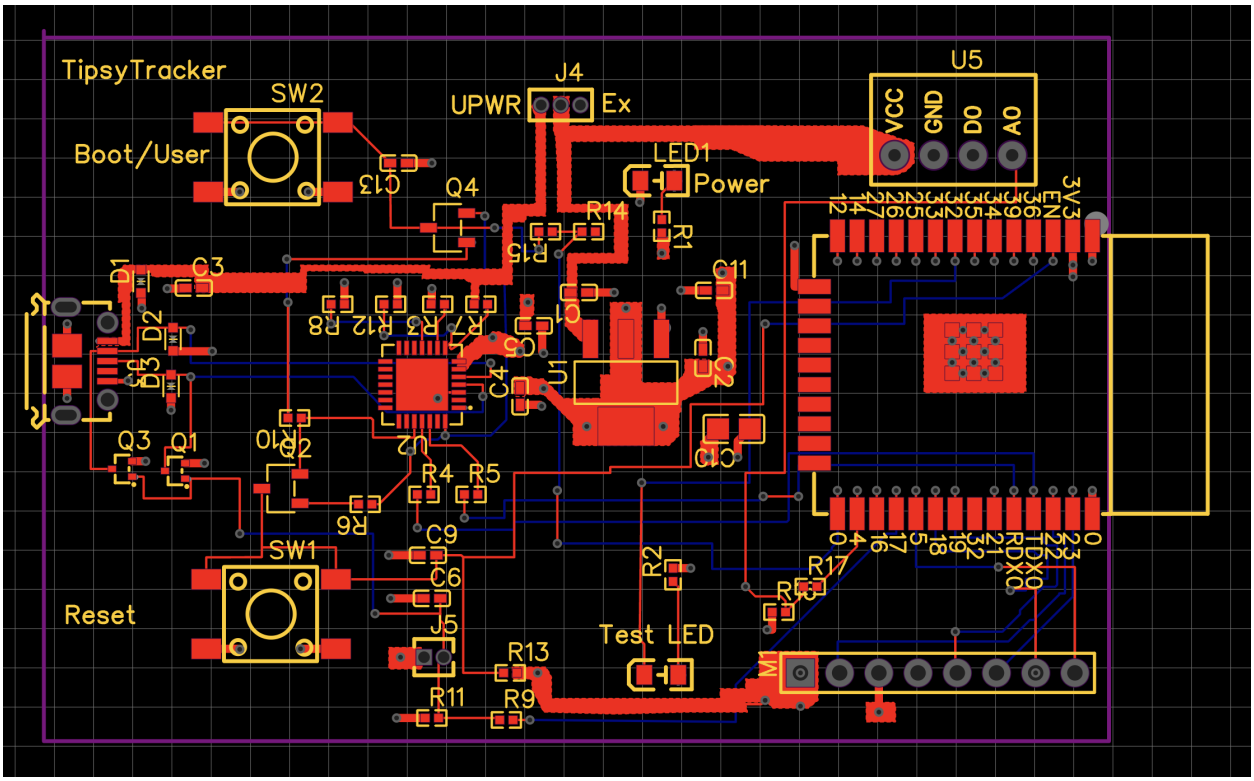
Date: February 15

I was able to make significant progress on the Printed Circuit Board (PCB) modeling and connections. Specifically, I was able to complete the modeling and line drawing of the PCB, which was a critical step in the development of the device.

The modeling and connections of the PCB were important because they enabled the team to visualize the physical layout of the device and ensure that all components fit correctly. I was able to create a detailed model of the PCB and ensure that all components were properly placed.

I helped to identify any potential issues or conflicts with the design, allowing the team to make any necessary adjustments before moving forward with the manufacturing process. This ensured that the device would function properly and meet the requirements of the project.

Overall, the completion of the first version of the PCB was a significant milestone in the development of the TipsyTracker system.



Original PCB connections

Date: February 20

On February 20th, we started a discussion on the design document for the TipsyTracker system. This was an important step in the development process as it allowed us to ensure that all team members were on the same page regarding the system design and implementation.

During the discussion, we reviewed the current system design and identified any areas that needed clarification or further development. We also discussed the implementation plan and identified any potential issues that could arise during the manufacturing process. Through this discussion, we were able to identify areas of improvement and ensure that the system design was well-documented and understood by all team members.

Date: February 21

We worked on refining the design document for the system. Specifically, we focused on adding more information about the requirements and verifications for all subsystems in the system, and I worked on adding a new power subsystem to the design.

I took the lead on adding detailed information about the functionality of each subsystem and the tests that would be used to verify that the subsystem was operating as intended (R and V).

By adding this additional information to the design document, we were able to ensure that all team members had a clear understanding of the requirements for each subsystem and the expected performance of the system as a whole. This allowed us to identify any potential issues or areas of improvement and make the necessary adjustments to ensure that the system would meet the requirements of the project.

Date: February 24

I made significant progress in the development of the TipsyTracker system by ordering all the components needed for the PCB. This was a critical step in the manufacturing process, as it ensured that we had all the necessary parts to build the device.

To ensure that all the components were of high quality and met the requirements of the project, I carefully researched and selected each part. This involved evaluating different vendors and comparing prices to ensure that we were getting the best value for our money.

Once I had selected the components, I placed the order and worked closely with the team to ensure that the parts were delivered on time and in good condition.

Date: March 2

I successfully wrote the Arduino code for the MQ-3 and RFID sensors. This was a crucial step in the development process as it allowed us to test the functionality of the sensors and ensure that they were operating as intended.

The Arduino code that I wrote included logic for reading data from the MQ-3 and RFID sensors and processing the data for use in the TipsyTracker system. Additionally, I incorporated functionality for sending POST requests to a simple webserver that I wrote in Python. This feature enabled the system to send data to the webserver for storage and analysis, and allowed us to monitor the performance of the sensors in real-time.

The successful implementation of the Arduino code for the MQ-3 and RFID sensors, with the added functionality of sending post requests to a webserver, was a significant accomplishment in the TipsyTracker development process. My contribution to this milestone helped to move the project forward, and we are now one step closer to achieving our goal of creating an effective and reliable system for monitoring alcohol consumption.

Compilation of arduino code

Test Case	RFID Tag	Bac Level	Expected Outcome	Actual Outcome
1	b449a889	0.03	Data received and stored	Data received and stored
2	c23a9111	0.04	Data received and stored	Data received and stored
...	...	...	...	...
300	j34k1501	0.1	Data received and stored	Data received and stored

Test Cases demonstraitng successful sending of data from ESP32 to the Raspberry Pi

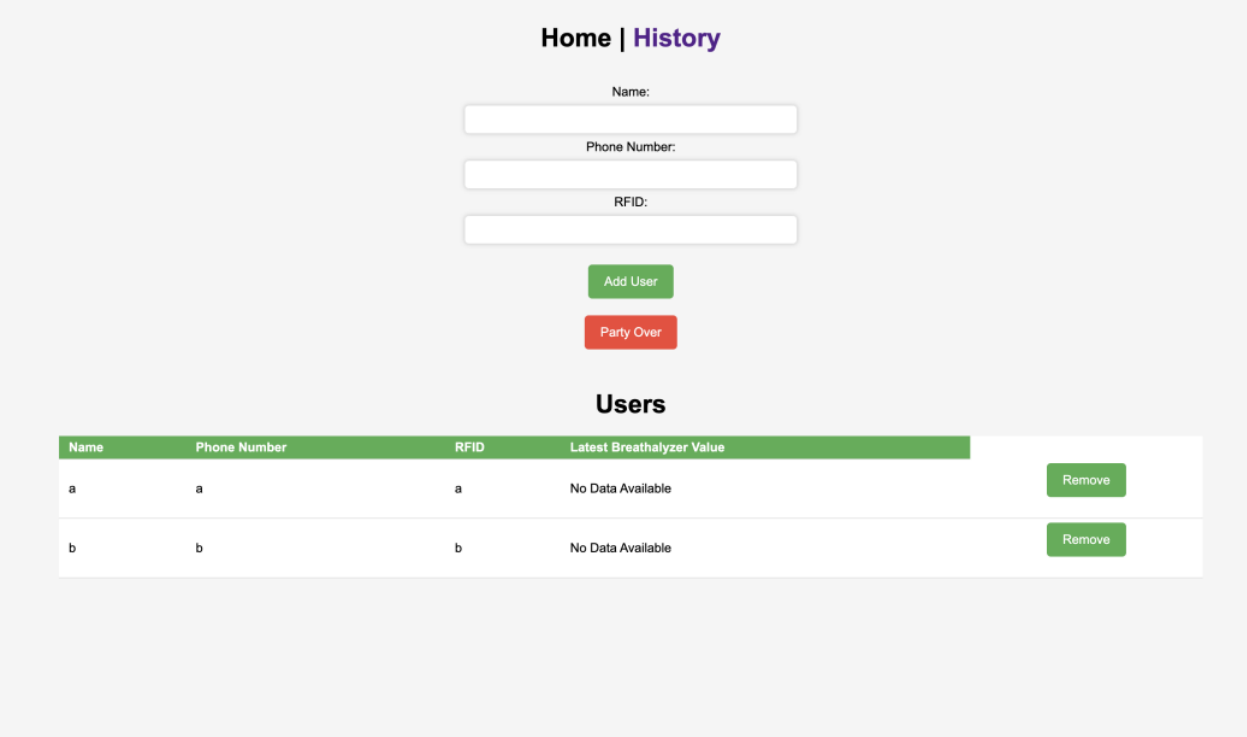
Date: March 5

I revamped the Python code to incorporate a web portal that allows for adding entries, viewing user history, and sending text messages about the BAC levels using Twilio. This was a crucial step in the development process as it allowed us to test the functionality of the web portal and ensure that it was operating as intended.

The revamped Python code includes a user-friendly web interface that allows hosts to add new entries and monitor the BAC levels of party-goers in real-time. The code also includes Twilio integration, allowing the system to send text messages to party-goers about their BAC levels and reminding them to check their levels at regular intervals.

Additionally, the code includes functionality for storing user history (during the party) to help hosts identify trends in alcohol consumption and make informed decisions about future events.

The successful implementation of the revamped Python code was a significant accomplishment in the TipsyTracker development process. By incorporating a user-friendly web portal and Twilio integration, the system is now more effective and easier to use than ever before.



Version 1 of Web Portal

Date: March 6

We discussed the plan for soldering the PCB, as well as went over my Arduino and Python code. This was an important step in the development process as it allowed us to prepare for the manufacturing phase and ensure that the code was properly reviewed and tested.

We also organized all of the components that were ordered for the PCB, ensuring that they were properly labeled and stored for easy access during the manufacturing process. This helped to streamline the process of assembling the PCB and reduce the risk of errors during manufacturing.

Additionally, we discussed the plan for soldering the PCB, identifying potential issues and developing a plan to mitigate these risks.

Date: March 7

My teammates attempted to solder the PCB, which proved to be quite challenging due to the incredibly small size of the surface-mount resistors. Despite our best efforts, they found themslves unable to solder them accurately and securely, which led us to the realization that we needed to reevaluate our approach.

After some deliberation, I decided I would go ahead andredesign the PCB using through-hole resistors and capacitors instead of surface-mount components. This change would make the soldering process significantly more manageable for our team, as through-hole components are easier to handle and solder in place. Furthermore, this adjustment would reduce the likelihood of errors and improve the overall reliability of the final product.

In the coming days, I will work on finalizing the new PCB design and ordering the necessary through-hole components. I hope that this decision will streamline the manufacturing process and ultimately lead to a more successful outcome for our project.

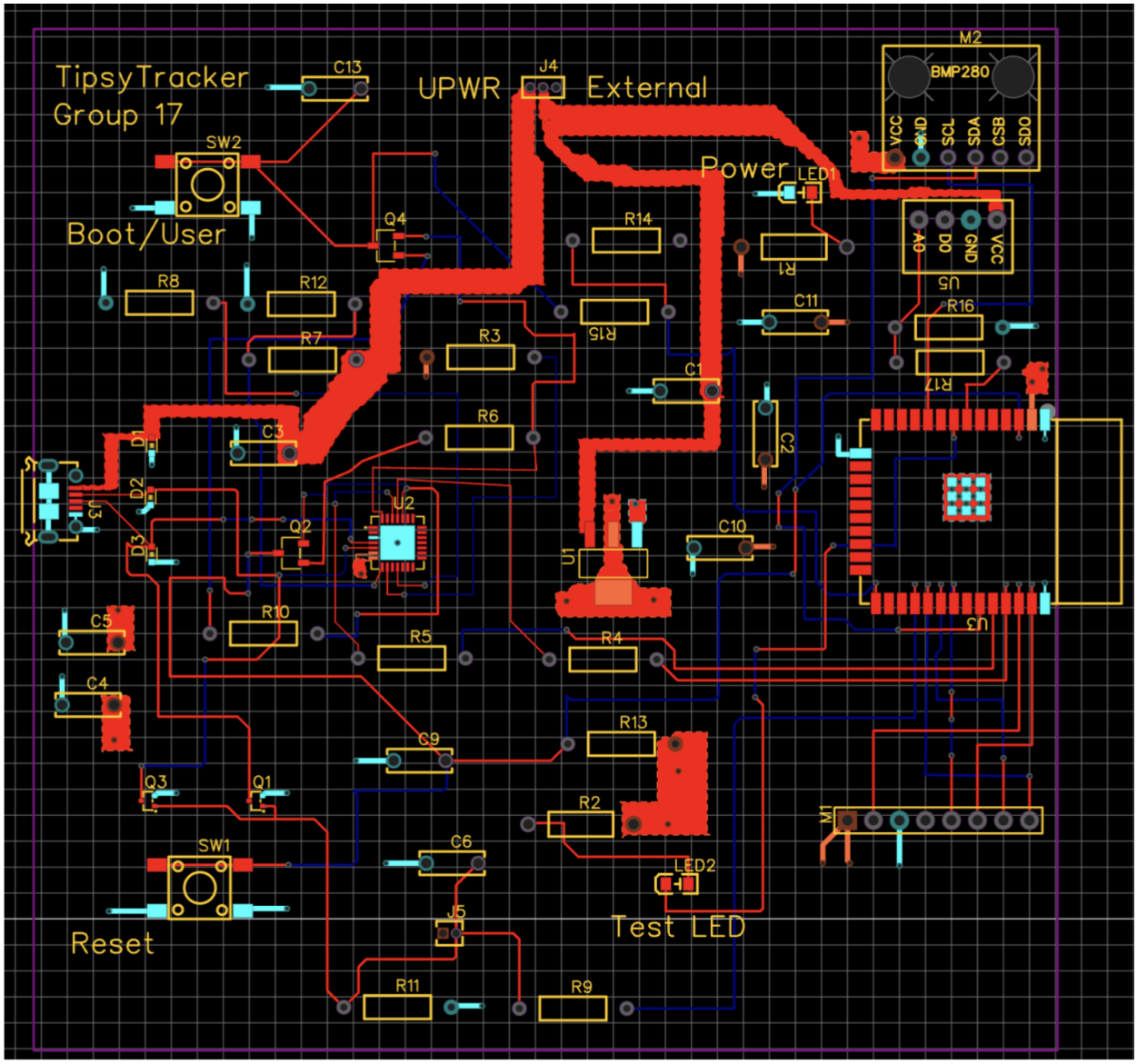
Date: March 9

Today marked a significant milestone in our project, as I successfully completed the redesign of our PCB to incorporate through-hole resistors and capacitors. This decision was made to address the challenges we faced while attempting to solder the surface-mount components, which proved to be too small for our team's skill level.

The new PCB design features well-organized through-hole components, strategically placed to optimize both the ease of assembly and the overall performance of the final product. The resistors and capacitors are now more accessible, allowing for a more straightforward soldering process and reducing the likelihood of errors during assembly.

I carefully calculated the required values for the resistors and capacitors, ensuring that they met the specifications of our project, and selected components with suitable power ratings and tolerances. To further streamline the assembly process, I also made sure to label each component on the PCB layout, making it easy for the team to identify and place them correctly.

Once the design was finalized, I exported the Gerber files and shared them with the team for review. We will soon order the updated PCB and through-hole components, confident that this new design will lead to a smoother manufacturing process and ultimately a more reliable end product.



New PCB connections

Date: March 10

We took an essential step forward in our project by ordering the newly redesigned PCB with through-hole resistors and capacitors. After reviewing the updated Gerber files and ensuring that all necessary adjustments had been made to accommodate the through-hole components, we felt confident in proceeding with the order. Overall, our team is eager to receive the new PCBs and begin the next phase of our project. We are optimistic that the redesign will make a positive impact on the ease of assembly and the performance of the final product.

Date: March 14

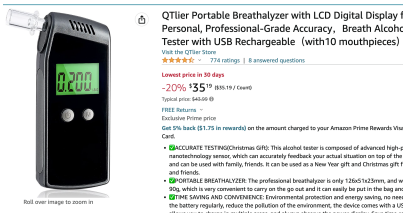
Today, our team held a productive meeting to discuss the Design Document and address any potential limitations in our project. Alongside our TA, we examined various aspects of our design and explored ways to improve the overall functionality and performance of the final product.

During the discussion, we came up with the idea of incorporating a pressure sensor into our design in order to determine if a user is actually breathing into the breathalyzer. We discussed different ways of incporating it, and my team decided to conduct numerous tests in the future to determine the capability.

Date: March 16

Today, in our pursuit to improve the accuracy and reliability of our MQ-3 breathalyzer sensor, I placed an order for a commercial breathalyzer from Amazon. The primary motivation behind this purchase is to create a correlation between Blood Alcohol Content (BAC) readings from the commercial breathalyzer and the output of our MQ-3 sensor.

Upon receiving the commercial breathalyzer, we plan to conduct a series of tests and comparisons between the two devices. This will enable us to better understand the relationship between the BAC readings and the MQ-3 output values, allowing us to calibrate our sensor more accurately.



Breathalyzer ordered

Date: March 20

Party 1: Training Data

Today, we organized a gathering with 10 participants to help us test our MQ-3 breathalyzer system and gather valuable training data. We provided both our MQ-3 sensor and the commercial breathalyzer we had previously ordered from Amazon for comparison purposes.

Each participant was asked to use both devices and record the values outputted by the MQ-3 sensor and the actual BAC reading from the commercial breathalyzer. This data will be instrumental in creating a linear correlation between the two sets of readings, which will allow us to calibrate our MQ-3 sensor more accurately.

Once we have analyzed the data and established the correlation, we plan to test the effectiveness of our calibration method during a secondary testing party. This follow-up event will provide us with the opportunity to evaluate the performance of our MQ-3 breathalyzer system under real-world conditions and assess the accuracy of our calibration process.

Arduino Value	True BAC Value
10	0
245	0.061
365	0.097
488	0.12
306	0.09
347	0.103

Truncated image of our training data.

Date: March 21

Today, our team set out to determine the most effective way to ensure that someone is breathing into the MQ-3 breathalyzer sensor with the correct amount of pressure. This is crucial for obtaining accurate readings and maintaining the reliability of our system. We explored and tested four potential methods to achieve this goal, but unfortunately, each one failed to produce the desired results:

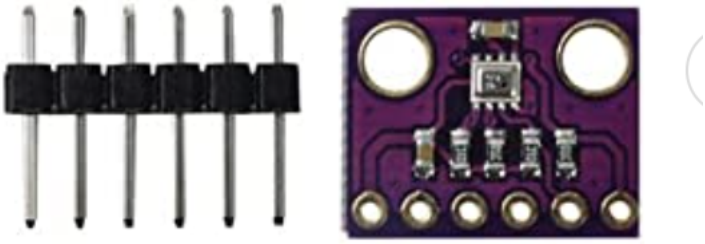
Placing a cloth in front of the breathalyzer: We attempted to use a cloth as a barrier to ensure that the user breathes directly into the sensor. However, this method did not provide consistent results, and we could not differentiate blowing pressure.

Creating an enclosure to measure pressure differences: We designed an enclosure around the breathalyzer sensor to monitor pressure changes within the confined space. Unfortunately, this approach proved to be unreliable as the senor was not sensitive enough to such changes.

Placing a paper over the pressure sensor: We tried covering the pressure sensor with a piece of paper, which was supposed to respond to changes in pressure as the user breathes into the breathalyzer. This method was inconsistent, as the paper's movement was difficult to measure accurately and could be influenced by other factors such as humidity.

Analyzing MQ-3 value changes to detect breathing: Our final approach involved examining the raw data generated by the MQ-3 sensor to identify patterns or fluctuations that would indicate whether someone is breathing into the breathalyzer correctly. However, the sensor's response time and variability made it challenging to differentiate between proper breathing and other sources of interference.

As a result, the team has decided to move forward with the project without incorporating the pressure sensor.



Pressure sensor we are using

Date: March 21

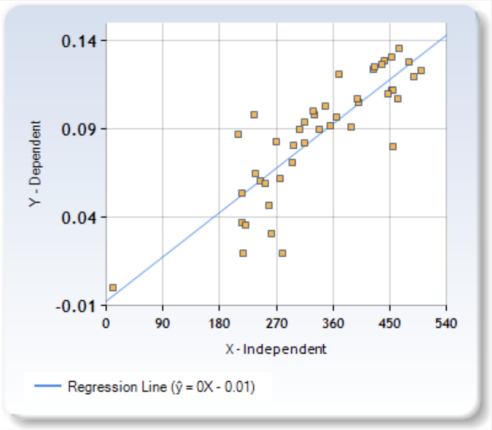
Today, our team also focused on calculating the linear relationship between the MQ-3 sensor output values and the commercial breathalyzer BAC readings, using the training data we gathered during the initial testing party. This relationship is crucial for calibrating our MQ-3 breathalyzer system and ensuring its accuracy and reliability.

To calculate the linear relationship, we decided to use an online Linear Regression Calculator provided by SocialStatistics, a user-friendly tool that streamlines the process of finding the best-fitting linear equation to represent the relationship between two sets of values.

We started by organizing the data in a spreadsheet, listing the output values of the MQ-3 sensor alongside the corresponding BAC readings from the commercial breathalyzer. We then inputted the data into the SocialStatistics Linear Regression Calculator, following the instructions provided on the website.

The calculator performed the regression analysis for us, determining the best-fitting linear equation that accurately represented the relationship between the MQ-3 sensor values and the commercial breathalyzer BAC readings. The resulting equation provided us with the slope and intercept values necessary to calculate BAC estimates based on the output of our MQ-3 sensor.

With the linear relationship now calculated using the SocialStatistics Linear Regression Calculator, we are well-equipped to move forward with the calibration of our MQ-3 breathalyzer system. We will soon test its effectiveness during a secondary testing party, which will allow us to assess the accuracy and reliability of the linear relationship we've established and make any necessary adjustments to our system.



Calculation Summary

Sum of X = 14938
Sum of Y = 3.846
Mean X = 339.5
Mean Y = 0.0874
Sum of squares (SS_X) = 462341
Sum of products (SP) = 129.098

Regression Equation = $\hat{y} = bX + a$

$b = SP/SS_X$ = 129.1/462341 = 0.00028

$a = M_Y - bM_X$ = 0.09 - (0*339.5) = -0.00739

$\hat{y} = 0.00028X - 0.00739$

Output from linear regression calculator

Date: March 22

We hosted a second testing party to evaluate the effectiveness of the linear relationship we developed for calibrating our MQ-3 breathalyzer system. Our goal was to assess the accuracy and reliability of our calibrated system by comparing its BAC estimates with the readings from the commercial breathalyzer.

With a group of participants, we conducted a series of tests during which each individual used both our MQ-3 breathalyzer system and the commercial breathalyzer. For the MQ-3 sensor, we took the output values and applied the linear relationship we previously calculated using the SocialStatistics Linear Regression Calculator. This allowed us to obtain estimated BAC levels based on the MQ-3 sensor's output.

Next, we compared the estimated BAC levels from our MQ-3 breathalyzer system with the actual BAC readings from the commercial breathalyzer. This side-by-side comparison provided us with valuable insights into the performance and accuracy of our calibrated system.

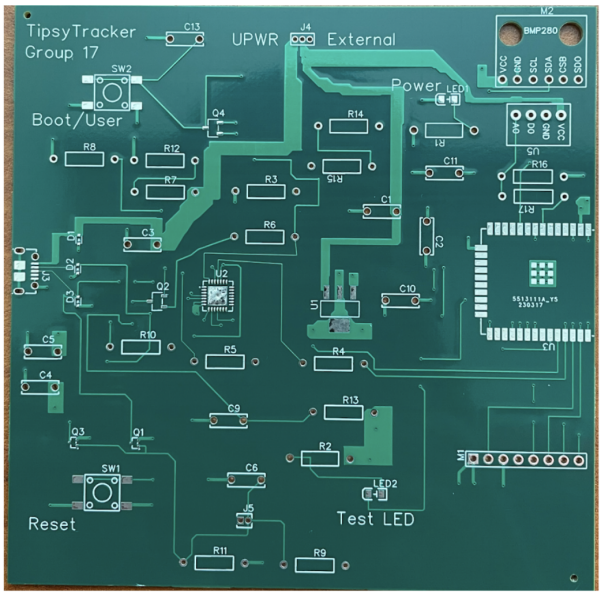
Arduino Value	Model BAC	True BAC
297	0.076	0.16
327	0.084	0.144
205	0.05	0.01
339	0.088	0.058
140	0.032	0.072
171	0.04	0.03
157	0.037	0.008
126	0.028	0.018
277	0.07	0.14

Truncated image of our testing data

Date: March 23

Today marks an exciting milestone for our project, as the PCB we ordered has finally arrived. With all the components in hand, we are now ready to proceed with the assembly and soldering process, bringing us one step closer to completing our breathalyzer system.

To streamline the soldering process and make it as efficient as possible, I took the initiative to carefully separate and organize the resistors and capacitors. By sorting them according to their values and clearly labeling each group, I made it easy for my partners to quickly identify and locate the correct components during the soldering process. This organization will not only save time but also minimize the risk of errors that may result from using incorrect components.



New, printed PCB

Date: March 28

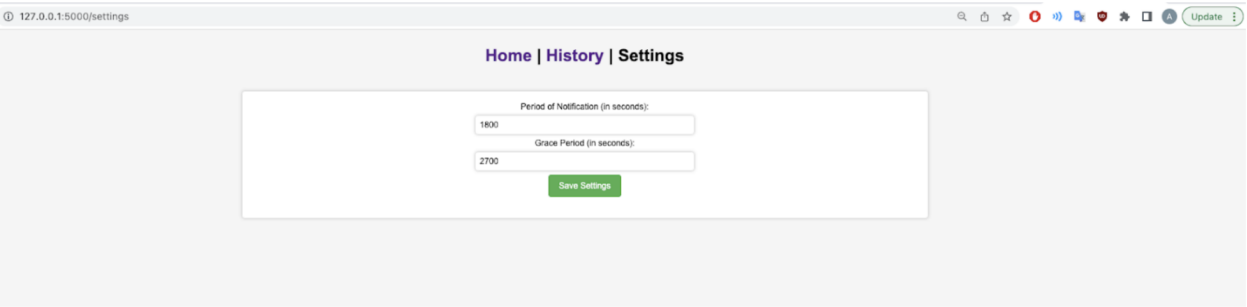
Today, our team had a productive meeting with our TA to discuss the challenges and best practices associated with soldering small SMD (Surface Mount Device) components onto the PCB. As these components are significantly smaller and more delicate than their through-hole counterparts, they require a higher level of precision and expertise during the soldering process.

Date: April 1

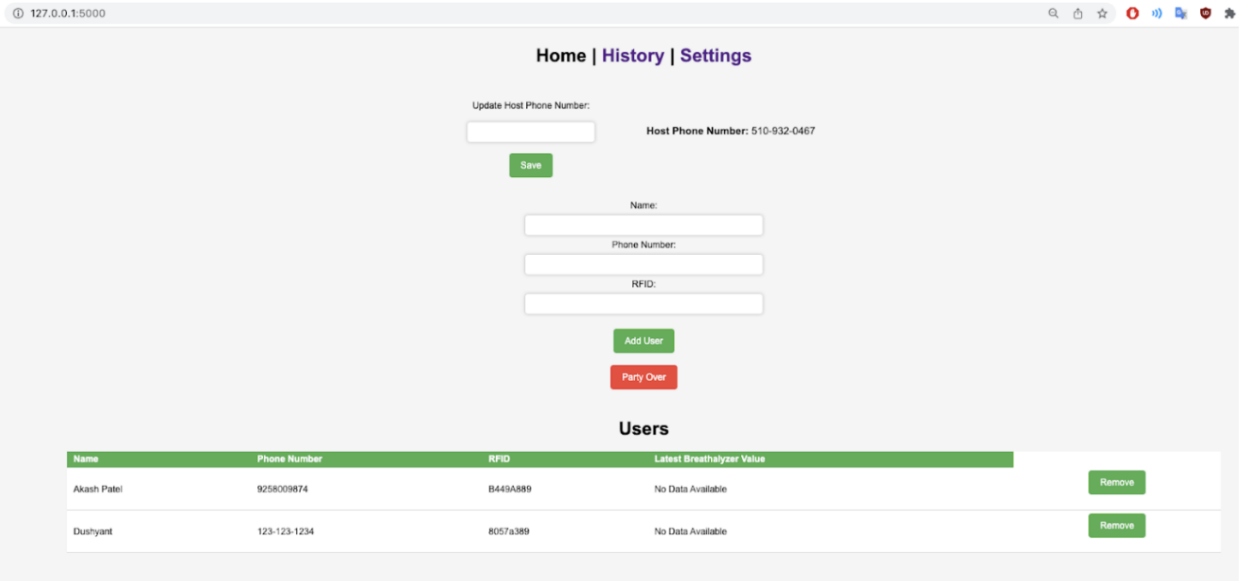
I made significant progress in developing the software component of our breathalyzer system by incorporating the linear relationship we established earlier into the Python code. This integration allows our MQ-3 sensor-based breathalyzer system to provide accurate BAC estimates, which are crucial for its overall reliability and functionality.

After applying the linear relationship to the MQ-3 sensor output values within the Python code, I proceeded to add a new feature to enhance the user experience of our system. I incorporated the ability for the host to input their phone number, which will enable the breathalyzer system to send relevant information and alerts directly to the host's mobile device. This feature ensures that the host stays informed and can take appropriate action when needed, contributing to the overall safety and effectiveness of our system.

In addition to the phone number input feature, I also developed a settings page for our breathalyzer system. This page allows the host to easily access and adjust various settings, such as calibration parameters, alert thresholds, and notification preferences. The settings page provides a convenient and user-friendly interface, making it simple for the host to customize the system according to their specific requirements and preferences.



Settings page



New home page

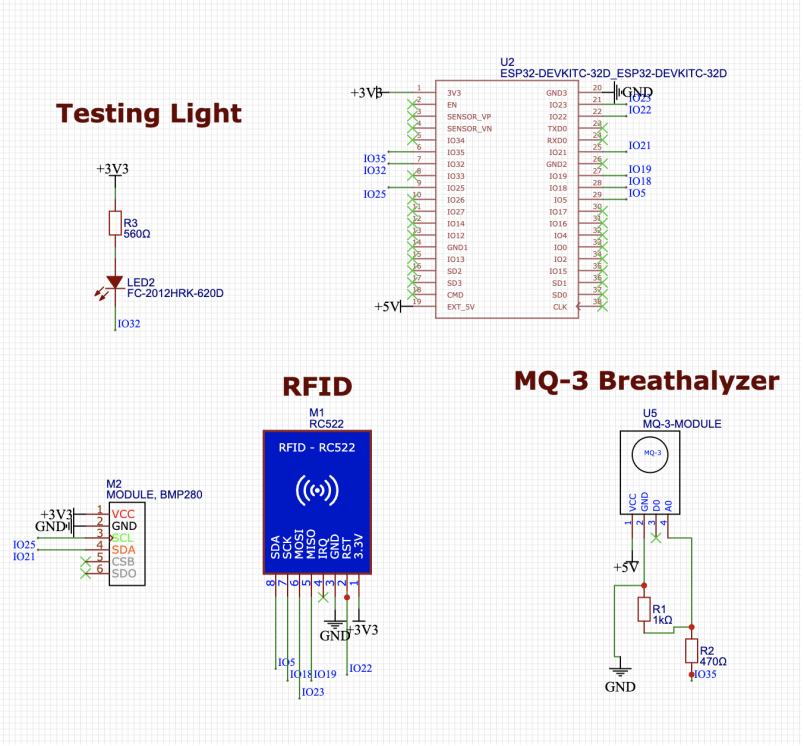
Date: April 2

Today, I focused on adapting our breathalyzer system's Python code to accommodate shorter timestamps, specifically for an upcoming demo presentation. This modification is crucial to ensure that our system performs efficiently and effectively during the demonstration, allowing us to showcase its full potential to the audience.

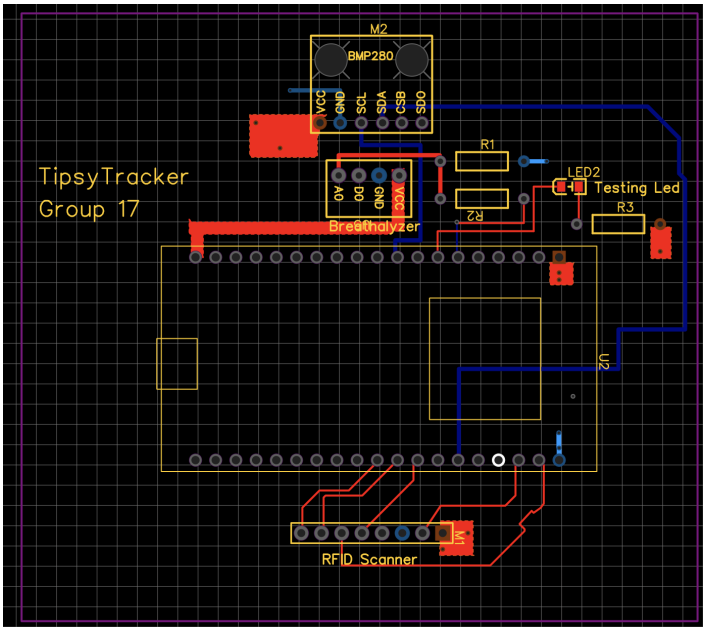
To achieve this, I carefully reviewed the existing code, identifying the sections that controlled the time intervals and the data collection process. I then changed them accordingly.

Date: April 4

After a recent team discussion, we concluded that it would be prudent to create a backup PCB that utilizes the development board, ensuring that we have a reliable fallback option in case our primary breathalyzer system encounters any issues. With the team's agreement, I took on the responsibility of designing the schematic and layout for the backup PCB that integrates the development board. This involved carefully considering the circuitry, component placement, and routing, ensuring that the design was both functional and efficient. I also made sure to include all the necessary connections and interfaces for the development board, allowing for seamless integration with the breathalyzer system's software and sensor components.



Schematic for dev board PCB



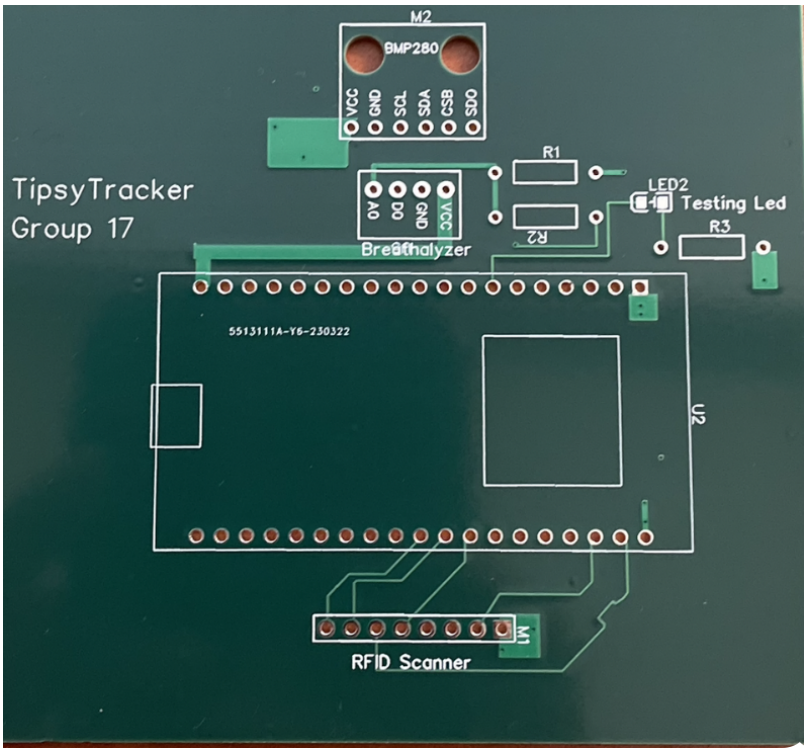
Layout for dev board PCB

Date: April 8

Today we received the dev board PCB, which we designed as a backup solution for our system. Having this backup PCB in hand provides us with a sense of security and reassurance, knowing that we have a reliable fallback option in case we encounter any issues with our primary system.

Upon inspecting the dev board PCB, we were pleased to find that the fabrication process had been completed accurately and without any visible flaws. The board's layout, component footprints, and routing all appeared to be in line with our design, giving us confidence in the quality and functionality of the backup solution.

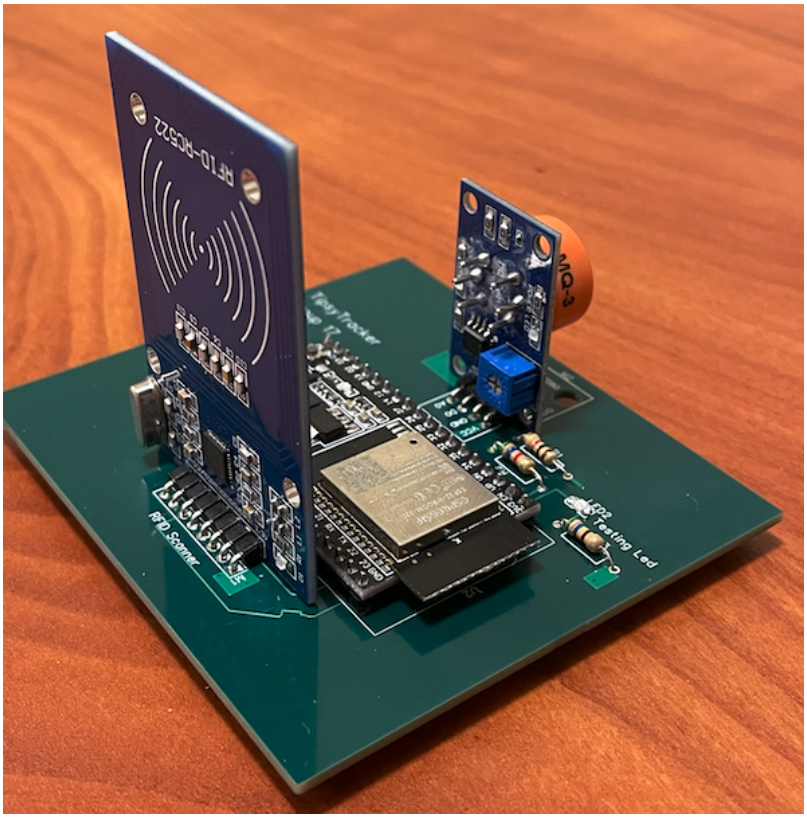
In the coming days, we will proceed with the soldering of the dev board PCB.



Printed dev board PCB

Date: April 10

Today, our team successfully soldered the dev board PCB, our backup solution for the breathalyzer system. We gathered, reviewed the schematic and layout, and soldered the PCB together. We then tested to make sure the PCB worked, which it did. We also began discussing the 3D-enclosure for the dev-board pcb.



Soldered dev board PCB

Date: April 15

Today, our team gathered to practice for the upcoming mock demo of our breathalyzer system. This rehearsal allowed us to refine our presentation skills, ensure a smooth delivery of information, and identify any areas for improvement before the actual demo. During the practice session, each team member took turns presenting their respective sections, focusing on conveying the key features and benefits of our TipsyTracker system in a clear and concise manner. We also made sure to address any potential questions or concerns that may arise during the demo.

After each presentation, we provided constructive feedback to one another, discussing ways to enhance the clarity, flow, and overall effectiveness of our delivery. This collaborative approach helped us fine-tune our presentation, boosting our confidence and ensuring that we are well-prepared for the mock demo.

Date: April 18

During our mock demo of the breathalyzer system at the ECEB, we encountered an unexpected challenge: the lack of access to data in the building. This limitation presented a potential obstacle for our presentation, as we needed to demonstrate the system's data communication and processing capabilities. We also found issues in our RFID sensor, leading us to re-order the part to resolder the system.

Date: April 19

I came up with a solution to address the data access issue we encountered during our mock demo at the ECEB. I proposed that the ESP32 could create a local area network (LAN) to facilitate communication

between itself, the Raspberry Pi, and the computer. This would allow us to demonstrate the breathalyzer system's data communication capabilities without relying on external internet access.

However, we acknowledged that this solution would not enable the Raspberry Pi to send notifications during the demo. To overcome this limitation, we decided to film a video before the demo, showcasing the communication system in action and demonstrating that the notification feature works as intended. This video will provide valuable evidence of our breathalyzer system's functionality and help compensate for any limitations during the live demo.

In addition to addressing the data access issue, we also received the new RFID component and proceeded to re-solder it onto our PCB. This update ensures that our breathalyzer system is fully equipped and prepared for the upcoming demo presentation.

LAN Networking Diagram

Date: April 22

Today, our team gathered for a meeting with the primary goal of filming all the notification features of our breathalyzer system. We wanted to create a comprehensive video that demonstrates the complete functionality of our communication system, including sending notifications to users and hosts, which will be shown during the demo presentation.

During the meeting, we made sure to set up the breathalyzer system and connect all the necessary components, ensuring that everything was working as intended. Each team member played a role in the filming process, from operating the breathalyzer system to recording the notifications on various devices.

We carefully filmed each notification feature, showcasing the seamless communication between the breathalyzer system, the Raspberry Pi, and the users' devices. This video will not only serve as evidence of the system's full capabilities but also provide a clear understanding of how the communication system works for the audience during the demo.

Date: April 23

I spent time editing a video that showcases the notification features of our TipsyTracker device. The objective was to create a visually engaging and informative presentation that clearly demonstrates how the device alerts partygoers and hosts about their blood alcohol content levels. Once the editing was complete, I shared the video with the team for their feedback and made necessary adjustments based on their input. The final product effectively showcases the TipsyTracker's notification system, highlighting its value in promoting responsible drinking and ensuring a safe and enjoyable party experience for all.

Video Thumbnail

In addition, our team gathered for one final practice demo of our TipsyTracker device. During the practice run, we simulated a live demo, setting up the device and walking through its operation. We also practiced handling potential questions or concerns from the audience, preparing ourselves for any inquiries that may arise during the actual presentation.

Date: April 26

Today marked a significant milestone for our team, as we conducted the final demo of our TipsyTracker device. It was a moment of excitement, as we showcased the fruits of our hard work and dedication. The demonstration went exceptionally well, with all subsystems functioning seamlessly together. Each team member confidently presented their respective components, highlighting the unique features and benefits of the TipsyTracker. We also provided a live demonstration of the device in action, accurately measuring BAC levels and successfully utilizing the RFID technology for guest identification.

Date April 27

Today, our team convened to work on our presentation in preparation for the upcoming mock presentation. This meeting was essential to ensure that we effectively communicated the key aspects of our TipsyTracker device and to fine-tune our delivery before presenting it in a more formal setting.

Date April 29

Today, our team made significant progress on our TipsyTracker project. After refining the presentation based on our practice sessions and team feedback, we turned our attention to the final design document, which will serve as a comprehensive record of our project's development journey.

References

[1] "Development boards | ESPRESSIF systems." [Online]. Available: https://www.espressif.com/en/products/devkits. [Accessed: 26-Mar-2023].

[2] "IEEE code of Ethics." [Online]. Available: https://www.ieee.org/content/dam/ieee-org/ieee/web/org/about/corporate/ieee-code-of-ethics.pdf.

[3] "Power supply ESP32 module - Espressif." [Online]. Available: https://dl.espressif.com/dl/schematics/esp32_devkitc_v4-sch.pdf. [Accessed: 26-Mar-2023].

[4] "RF Wireless World," ESP32 Arduino Interfacing with Gas sensor diagram, working, code. [Online]. Available: https://www.rfwireless-world.com/ApplicationNotes/ESP32-interfacing-with-Gas-sensor.html. [Accessed: 26-Mar-2023].

[5] Xukyo, “Using an RFID module with an ESP32 • aranacorp,” AranaCorp, 03-May-2021. [Online]. Available: https://www.aranacorp.com/en/using-an-rfid-module-with-an-esp32/. [Accessed: 26-Mar-2023].