ECE 445 Smart Inventory System

By
Krish Naik Aparaj (krishn2)
Rohan Harpalani (rohanhh2)
Rushil Duggal (rduggal2)

Final Report for ECE 445, Senior Design, Fall 2023
TA: Sanjana Pingali

7 December 2023
Project No. 10

Abstract

This project established a streamlined, automated system for students to borrow and return technical components in the ECE 445 laboratory. The system allows easy access through iCard RFID authorization, securely stores components in a locked container, enables efficient inventory management through a QR code detection system, and maintains a backend record of transactions for use by course staff. This project succeeded in achieving each of its high level requirements and is fully functional for implementation.

Contents

1. Introduction

1.1 Purpose

The management of course components in ECE 445 currently relies on a tedious system that places strain on teaching assistants and introduces logistical inefficiencies. The process involves manual handling of borrowing and returning items, leading to errors, misplaced items, and added workload for TAs. This report proposes a solution centered around an automated camera-inventory system to streamline the management of borrowed components.

The existing system for borrowing and returning components involves students physically visiting the inventory, manually selecting required components, and TAs recording each transaction through a Google Form. This manual process can lead to overlooked returns, complicating TA responsibilities and potentially requiring additional efforts to retrieve missing items. It also forces students to wait for TA office hours to be in session, further slowing the design and build processes.

Our solution involves a streamlined and automated process which begins with components tagged with QR codes and securely stored in a locked container. Students will access these components by scanning their iCard to unlock the container. Within the container, students can select needed parts, hold them up to the camera, and identify them based on QR-code. The camera will then log transaction data into a backend system. Returning components follows a similar process with students rescanning their iCard to unlock the container and initiating the return by rescanning the components.

The automated camera-inventory system showcases increased efficiency by automating the borrowing and return processes while also providing increased security, accountability, and reduced workloads for TAs. The implementation of this system offers a promising solution to the current inefficiencies in managing course components in ECE 445.

1.2 Functionality

Our system had 3 high level requirements, which we set out to achieve over the course of the semester:

1. *The system should unlock the container in less than 7 seconds from when a student first taps their iCard on the RFID reader.*

This requirement ensures a swift authorization process for students while maintaining component security and integrity in our system as a whole. By tapping their iCard on the RFID reader, students initiate this process, whereby the system decodes their corresponding UID, verifies that the student is authorized in the database, and enables the relay, unlocking the container.

2. *The system should be able to scan and recognize each component's QR code in less than 6 seconds of first appearing in the frame.*

This requirement ensures a seamless checkout and return process for users of our system, by allowing automated tracking and entry of components into the database. In these 6 seconds, the camera must focus on the QR code, decode it, and send it to the rest of the system.

3. *The system should correctly update the database in less than 6 seconds when a student borrows or returns a component after receiving its unique identifier from the QR code.*

This requirement allows for prompt inventory management and backend logging when students borrow or return components. Within 6 seconds, the system must send the component to the database, alongside the student that checked it out or returned it. This functionality is integral to ensuring accurate and up-to-date records of borrowed items, minimizing errors, and simplifying the tracking process for TAs. It aligns with the project's goal of automating processes and reducing the workload on TAs by handling transaction logging seamlessly.

Each of these high level requirements was satisfied in our finalized design, indicative of a successful project.

1.3 Subsystem Overview

Our system employed four primary subsystems: a power supply, control, user interface, and database subsystem. The power supply subsystem's AC-DC adapter provides a 5V input to the system, which is then stepped down to 3.3V via a linear regulator circuit, providing both voltage levels throughout our system. The control system contains the microcontroller, relay, and solenoid lock, used specifically in securing the components. This system interfaces with the user interface subsystem, containing the RFID reader, LCD display, and camera, as well as the database subsystem, maintaining a record of all transactions. The system's high level design can be seen in Figure 1.

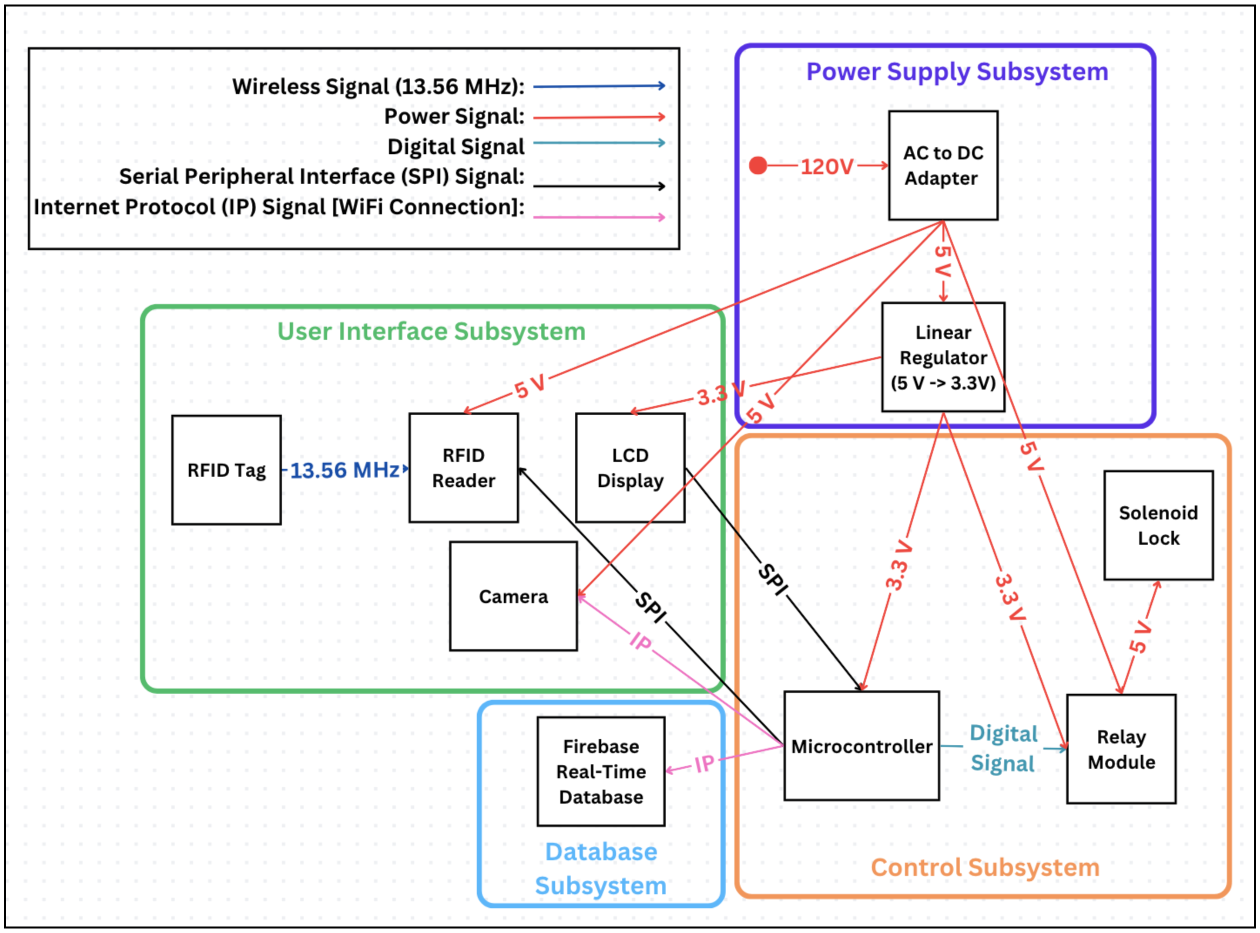


Figure 1: High level system block diagram

2. Design

2.1 Control Subsystem

The control subsystem consists of an ESP32-S3-WROOM-1 microcontroller, a relay board, and an electric solenoid lock. Each of these components are crucial to the functionality of the systems control and ensuring proper communication between different subsystems as well. The control subsystem first begins working when a student initiates access to the container by scanning their iCard on the RFID reader to borrow a component. The ESP32 microcontroller then extracts the UID from the iCard and then it verifies whether this UID is in the Google Firebase. If the UID is present, then the microcontroller sends an enable signal to the relay to unlock the solenoid. The relay-solenoid connection is initially set to normally-open (NO) [13], which indicates that the circuit is open and the solenoid is locked. When the microcontroller sends an enable signal, the electromagnet turns on and closes the circuit, allowing current to flow through and unlock the solenoid. The ESP32 maintains a record of every access to the container and records this transaction into the database. Similarly, this process occurs for those returning components and we ensure that the student is authorized before providing access to the components.

2.2 Power Supply Subsystem

The power subsystem is essential for powering our entire PCB and our peripherals. At first, this subsystem utilized a 120V to 12V AC to DC adapter to supply 12V to our PCB. As part of our design, we wanted to receive 5V and 3.3V to power different components on our PCB. To be able to supply both 5V and 3.3V, we initially utilized two buck converters to step down the 12V as needed. However, after soldering our respective components on and powering our board with 12V, we had unusual voltage readings on the board and realized our board overheated.

We realized a couple design issues with our initial PCB such as lack of ground stitching, possible noise from microcontroller placement and track size. To counter these issues and simplify the power input to our board and different components, we chose to use a 120V to 5V AC to DC adapter instead so that we can directly supply 5V to our board and only need to step 5V down to 3.3V. To step down to 3.3V, we utilized a 5V to 3.3V linear regulator instead of a buck converter and verified the thermal and current conditions before choosing to do so. We also modified the positioning of the components on the board to reduce noise and added ground stitching to make the PCB less error prone. This design decision served us well as we successfully supplied the necessary power to system components and ensured the functionality of the system.

2.2.1 Linear Regulator Tolerance Analysis

To ensure our chosen linear regulator could safely be used on the PCB, we conducted a tolerance analysis using parameters found in the datasheet & the expected current draw of our components.

We first totaled up the worst case current draw of each of the components requiring 3.3V using the following equation:

$$I_{out} = (I_{ESP32}) + (I_{LCD}) + (I_{Relay}) = 355\,mA + 30\,mA + 12\,mA = 397\,mA$$

We then used the parameters in Table 1 to calculate the linear regulator's junction temperature:

Variable	Value
T_{max} (maximum rated temperature)	150 °C
I_{out}	397 mA
V_{in}	5 V
V_{out}	3.3 V
Θ_{ja} (thermal resistance, junction to ambient)	125 °C/W
Θ_{jc} (thermal resistance, junction to case)	15 °C/W
T_a (ambient temperature)	38 °C

Table 1: Linear regulator parameters

We then used the following formula with the parameters shown in Table 1:

$$T_j = (I_{out}) * (V_{in} - V_{out}) * (\Theta_{jc} + \Theta_{ja}) + (T_a)$$
$$T_j = (0.397\,A) * (5\,V - 3.3\,V) * (125\,°C/W + 15\,°C/W) + (38\,°C)$$
$$T_j = 132.49\,°C < T_{max}$$

Based on the above analysis, our chosen linear regulator would output the necessary voltage without overheating, given that the estimated temperature ($T_j = 132.49\,°C$) is less than the maximum operating junction temperature (T_{max} = 150 °C).

2.3 User Interface Subsystem

The user interface subsystem comprises the primary functionality of our system, containing the RFID reader used to authorize users, the LCD to display real-time updates, and the camera to decode QR codes on components. This subsystem also made up the bulk of the software flow, where the user interaction process is replicated in the code.

This process works as follows: first, a user taps their iCard on the RFID reader. The software then maps that card's UID to the user, and checks via the database whether that user is authorized to access the course components. If authorized, the LCD prints a message to welcome the user and notifies the user that the box is open after the ESP32 sends a signal to the relay to unlock the container [3]. The system then spins until the user scans a component, decodes the QR code, determines automatically if the component is being checked out or returned, and updates the database accordingly. The relay then disables the solenoid, locking the container, and shows a message on the LCD stating that the component has been either checked out or returned.

2.3.1 Polling- vs Interrupt- Based Input

One of the most important design decisions involved the core operating system-like functionality while the system waits for an RFID tag. At this point, we had to decide between implementing polling- or interrupt- based IO. Using our knowledge of operating systems from ECE 391, we decided on polling-based IO for its simplicity and predictability. Since the system wasn't doing any computation before the user interface sequence was initiated, there would be no process to be interrupted by the RFID tag [2], making an interrupt-based IO superfluous and overly complicated. Additionally, since polling occurs at predefined intervals, it provides a predictable and consistent approach to checking the RFID reader's status, which is very beneficial given our real-time system [9].

2.3.2 Frictionless UX

Another important design decision involved designing software to create a frictionless experience for the user. Rather than requiring students to manually input whether they are checking out or returning components (i.e. specifying the type of transaction), our database & software design automatically checks whether the component is in the inventory or has already been checked out, indicating whether the transaction is a borrow or a return, another advantage of having both checkouts and inventory lists. This allows students to seamlessly tap their iCard and scan the component's QR code, and the system will automatically update the database as a checkout or as a return. In this software design, we also included protections for various edge cases, including a student trying to checkout or return a component that has already been checked out to another user. Since this system absolves the need for a TA to be present at every step of the checkout process, we wanted to ensure that the appropriate checks were still put into place.

2.3.3 Peer-to-Peer Microcontroller Communication

Our project relied on two ESP32s: one in the control subsystem and one to control the camera module. Since the camera ESP32 decoded QR codes on components and the primary ESP32 used those decoded names to borrow or check out components, we needed a method of peer-to-peer communication between the ESPs.

First, we explored using a hardware connection to directly connect the two. Though this would be the most reliable, this ultimately proved to be far too complex to implement, since it involved creating a fully customized SPI interface and additional ports. Next, we looked into peer-to-peer networking via the ESP NOW protocol, which essentially creates a low-level WiFi connection between the transponder and the receiver [14]. However, after multiple iterations of testing, we found this process to be highly unreliable, with the control ESP receiving sent data roughly only one out of every 10 times. Finally, we decided to use the existing Firebase database to facilitate this transfer of data between ESPs. To do so, we wrote a component name from the camera ESP in the database and read it from the main ESP. This proved to be highly efficient and made use of our existing database design, making it the optimal solution for this issue.

2.3.4 Multi-Core Camera

The ESP32-CAM module is responsible for decoding the QR-codes and we want this task to occur in real-time. To decode the QR-codes, we utilize FreeRTOS but given that we use an RTOS, we are susceptible to schedulers delaying the tasks that we want to execute. To counter this, we looked into offloading our tasks onto a separate core, free of externally scheduled tasks. The ESP32 has an Xtensa LX6 Processor which contains two cores and when creating tasks, they are normally executed on core 1 by default [18]. To ensure real-time decoding of QR codes without interrupts and waiting for other scheduled tasks, we offloaded the QR-code decoding computation to core 0.

2.4 Database Subsystem

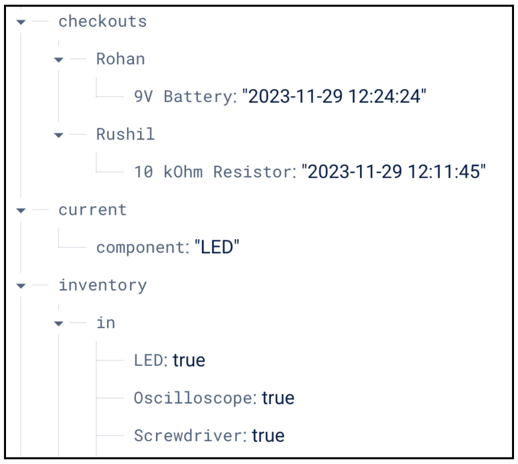
Our database subsystem is used to maintain a record of all transactions occurring within the system. When a student borrows or returns a component, the database updates such that the student's name, component, and timestamp are all logged in the system. Additionally, course staff can maintain the index of authorized students, ensuring component security and inventory integrity.

Employing a no-SQL, serverless database was an important design consideration. As we did not want to maintain the infrastructure for a database ourselves, we decided to investigate one of the many serverless hosting options available. Additionally, to make the database easily accessible to a variety of course staff, we also required a no-SQL database. Rather than forcing TAs and professors to make complex SQL queries to update the database or check the inventory log, a no-SQL database allows easy read and write access via the web console and GUI.

There were a few database options that met these requirements, including MongoDB Atlas, AWS Redshift, and Google Firebase. We ultimately landed on Google Firebase due its generous free tier, well-documented example code, and schemaless format [16]. Rather than being organized into rows and tables, Firebase's data is stored in documents, and ordered into collections. This allowed a lot more freedom in designing the database and the main software for our project, since documents can store multiple different data types.

The internal structure of the database maintained 4 primary collections of documents:

1. Checkouts: Lists all parts checked out, separated by user
2. Inventory: Lists all parts in system, divided by whether they are in the container or outside of it
3. Users: Lists all users, including whether they are authorized or not
4. Current: Lists the current component scanned as a means of sending data from the camera to the main control system (this decision further detailed in section 2.1)



```
checkouts
    Rohan
        9V Battery: "2023-11-29 12:24:24"
    Rushil
        10 kOhm Resistor: "2023-11-29 12:11:45"
current
    component: "LED"
inventory
    in
        LED: true
        Oscilloscope: true
        Screwdriver: true
```

Figure 2: Database organization

2.4.1 Firebase Lookup Runtime Optimization

When storing these components in Firebase, we used a Firebase paradigm suggested in their documentation to invert the data as keys and set the values to either True, None, or some other “don’t care” (X) value [15]. This paradigm greatly simplifies the process of checking whether a key is present; rather than searching through the entire collection to determine if a component is available for checkout (which would necessitate linear runtime in the worst case), we can instead just try to access the value at /inventory/in/$component (constant runtime). If this access operation returns a value, regardless of its value, we know that the component is present in the list and can be checked out. This paradigm becomes even more powerful in storing additional data — when writing the checkout timestamp in the database, that string can be stored as the value in the { component : timestamp } key : value pair, further condensing the amount of information required in the database [17].

2.4.2 Database Management Python Script

As part of this project's final deliverable, we also created a simple Python script to allow ECE 445 course staff to easily modify the database records. For example, if a student manually returns a component to the TA, rather than having to go into the system and edit the database themselves to check it back in, they could simply run the provided script, seamlessly updating the database. Additionally, course staff can use this to manually adjust authorizations, add or remove components from the master inventory, and maintain control of the overarching system.

3. Design Verification

3.1 Control Subsystem

Our first requirement for the control subsystem was to map a UID taken from a student iCard to a student in the database within 3.5 seconds. We wanted this to apply to both authorized and unauthorized students. Therefore, to test this, we took two authorized iCars and one unauthorized iCard and measured the time taken between scanning the iCard to the LCD being populated with a message stating whether the user was authorized or not authorized. The results of the test can be seen in Table 2.

iCard	Time taken for message displayed (sec)
Authorized	2.65
Authorized	2.89
Not Authorized	2.74

Table 2: Control Subsystem Requirement 1 Results

The next two requirements for the control subsystem was to ensure that the system only unlocks the container if the student is authorized. To test this logic within the microcontroller, we set up a test point which was connected to the output of GPIO 20 on the microcontroller. The microcontroller outputs 3.3V [3] through GPIO 20 to the relay to unlock the solenoid given that an authorized user scans their iCard. Therefore, to test this, we took one authorized iCard and one unauthorized iCard, scanned the cards one at a time, and with each card scanned, measured the voltage at the test point with an oscilloscope. We performed this test twice to verify the voltages. If the voltage was 3.3V, we are expecting that an authorized user tapped their iCard and vice versa. The results of the test can be seen in Table 3.

iCard	Measured Test Point Voltage (V)
Authorized	3.242
Not Authorized	-0.008
Authorized	3.283
Not Authorized	-0.006

Table 3: Control Subsystem Requirement 2 and 3 Results

These requirements and verifications can be seen in further depth in Appendix A.

3.2 Power Supply Subsystem

Our main requirement for the power supply subsystem was to ensure that the circuit that we designed for dropping 5V to 3.3V functions as expected. To test this, we included a test point to validate a 5V input into our linear regulator circuit and also included another test point to validate a 3.3V output from our linear regulator circuit. We then plugged in the barrel jack and used a multimeter, with one end on the test point and another end on ground, to validate the voltages that we were getting. We performed this test three times and the results can be seen in Table 4.

Trial	Measured V_{In} (V)	Measured V_{Out} (V)
1	4.834	3.2935
2	4.983	3.2874
3	4.949	3.2966

Table 4: Power Supply Subsystem Requirement Results

These requirements and verifications can be seen in further depth in Appendix A.

3.3 User Interface Subsystem

We had two primary requirements for the user interface subsystem. First, we wanted to ensure that the camera could detect and decode QR codes on components within 4 seconds. To test this, we held the camera above each component and waited for it to decode the QR code, upload the component's name to the database, and display it on the LCD, timing this entire process. The results of this test can be seen in Table 5.

Trial	QR Decoding Duration (sec)
1	3.54
2	2.21
3	3.78

Table 5: User Interface Subsystem Requirement Results

Our second requirement was for the LCD to display a message stating that the container has been unlocked after the user is authorized. To test this, we tapped the iCard of an authorized user on the RFID reader and waited for the solenoid to unlock the container. Once this happened, we verified that the LCD showed a message stating that the container was unlocked and that a student could now choose a component.

These requirements and verifications can be seen in further depth in Appendix A.

3.4 Database Subsystem

Our database subsystem had 3 primary requirements. First, we wanted to ensure that the database could correctly update to reflect component checkout, including the name of the student, the component checked out, and the timestamp of the transaction. To test this, we modeled the entire user interface interaction, starting with an authorized user tapping their iCard, opening the container, and scanning a QR code. We then visually inspected the database via the web console GUI, verifying that the student name, component type, and timestamp matched that of our experiment.

Second, we wanted to ensure that the database could correctly update to reflect component return. The same steps as above were used to test this, except we returned a component that was already checked out. Again, we inspected the database to verify that the component name was no longer checked out to the corresponding student.

Finally, we wanted to verify that the system would not allow a student to check out or return a component already checked out to another student. We verified this by first checking out a component using one authorized iCard, and then attempting to return that same component using another iCard.

These requirements and verifications can be seen in further depth in Appendix A.

4. Cost and Schedule

4.1 Cost

Our team's labor costs can be calculated as follows: $50 (hourly wage) x 15 (hours per week of work) x 11 (weeks of work) x 2.5 (scaling cost) x 3 (3 people in the group) = $61,875 (total labor costs)

An abbreviated breakdown of our team's costs can be seen in Table 6.. For a detailed breakdown of all parts ordered, refer to Appendix B.

Description	**Cost**
PCB Parts	$137.22
3D Printing	$10
Physical Design	$25

Table 6: Overall team costs

4.2 Schedule

An abbreviated version of our schedule can be seen in Table 7 below. For an expanded version, refer to Appendix B.

Week	**Overarching Task**
Sept. 24 - Sept. 30	Order components and start designing subsystem schematics
Oct. 1 - Oct. 7	Finalize first PCB version and design document
Oct. 8 - Oct. 14	Finalize physical design and place first round PCBWay order
Oct. 15 - Oct. 21	Begin programming ESP32 and testing modules
Oct. 22 - Oct. 28	Make second round PCBWay order
Oct. 29 - Nov. 4	Begin assembly and initial subsystem testing
Nov. 5 - Nov. 11	Finalize assembly and test overall functionality
Nov. 12 - Nov. 18	Fix any bugs from mock demonstration
Nov. 26 - Dec. 2	Finalize final demonstration

Table 7: Schedule

5. Conclusion

5.1 Accomplishments

Our project was successful in meeting all of its high level requirements and achieving full finalized functionality. This system presents a streamlined, efficient, automated process for managing inventory in the ECE 445 lab, both reducing labor on the part of the course staff and making it easier for students to obtain needed parts for their projects.

Our system was able to authorize users and unlock the container storing components, scan and decode component QR codes, and update the database with the required information within the time limits originally specified in our high level requirements. Additionally, our system worked on a PCB, without the need for any external modules or power supplies, making this project a success.

5.2 Uncertainties

While testing our printed circuit board, our original power supply design caused overheating issues and abnormal voltage readings. Specifically, our power supply subsystem was set to receive a 12V input and then drop it down to both 5V and 3.3V. However, we encountered an issue where one of our buck converter circuits unexpectedly outputted approximately 24V, which caused our printed circuit board to overheat. Due to the time constraints we faced for meeting our project deliverables, we decided to redesign our power supply subsystem. Subsequently, we managed to program the ESP32 microcontroller on the printed circuit board without using the original 12V power supply. Hence, for our new design, we reduced our input voltage from 12V to 5V, and then replaced the 12V to 3.3V buck converters with a 5V to 3.3V linear regulator. As a result, our final design consisted of two PCBs as one handled the power supply subsystem, and the other managed the ESP32 microcontroller and other functionalities.

5.3 Future Work and Alternatives

In considering potential enhancements for our project, a few future improvements can be identified to improve the functionality and user experience of our system. First, developing a mount for the ESP32 camera would eliminate the need for manual scanning by students and allow for faster and more precise component recognition.

Another opportunity for improvement involves expanding user input interfaces to enable multiple checkout or return transactions. Implementing additional interfaces would allow students to initiate multiple transactions in a single session, improving user efficiency and simplicity.

Another improvement would be developing a dedicated web interface as an indirection of the Firebase database, allowing ease of usability by ECE 445 course staff. This interface could offer a comprehensive overview of inventory, allowing staff to easily monitor borrowing activities, track component availability, and generate reports as needed. Integrating user-friendly visualization tools and search functionalities within the web interface could further optimize data retrieval and analysis for course staff.

5.4 Ethical Considerations

One relevant ethical concern for this system would be regarding user privacy. Since users are expected to stand in front of a camera to scan their item, there may be concerns about storing their biometric/facial information in our system as well, which users may understandably be uncomfortable with. The ACM Code of Ethics specifically raises this concern in their principle to respect the privacy of users, making it all the more important to heed. To mitigate this concern, we could make it clear to users that no visual information is stored - the camera is only used to read the QR code.

Another ethical concern is to avoid discrimination. As detailed in the ACM Code of Ethics, this is especially important given that our project may lead to technology-driven discrimination in terms of monitoring the supply cabinet [8]. This project should not be used to prejudice or discriminate against any groups. To mitigate this, any violations found using our system should be thoroughly investigated by a user as well, to avoid any wrongful accusations.

6. References

[1] BQ24616 Jeita guideline compatible stand-alone synchronous switched ..., https://www.digikey.com/htmldatasheets/production/742277/0/0/1/bq24616evm.pdf (accessed Sep. 28, 2023).

[2] "Different types of RFID systems," Different, https://www.impinj.com/products/technology/how-can-rfid-systems-be-categorized (accessed Sep. 28, 2023).

[3] "Esp32 datasheet by Espressif Systems," DigiKey, https://www.digikey.com.mx/en/htmldatasheets/production/3025158/0/0/1/3320 (accessed Sep. 28, 2023).

[4] "Factors affecting PC monitor responsiveness," Factors Affecting PC Monitor Responsiveness | PC Monitors, https://pcmonitors.info/articles/factors-affecting-pc-monitor-responsiveness/ (accessed Sep. 28, 2023).

[5] Farnell, https://www.farnell.com/datasheets/2712232.pdf (accessed Sep. 28, 2023).

[6] J. Czerniak, A. Gacek, and P. Szopa, "Analysis of Power Bank quality criteria that are important from the consumer's point of View," MDPI, https://www.mdpi.com/1996-1073/14/18/5747 (accessed Sep. 28, 2023).

[7] Prashanth Jayaram, "Backup and restore (or recovery) strategies for SQL server database," SQL Shack - articles about database auditing, server performance, data recovery, and more, https://www.sqlshack.com/backup-and-restore-or-recovery-strategies-for-sql-server-database/ (accessed Sep. 28, 2023).

[8] "The code affirms an obligation of computing professionals to use their skills for the benefit of society.," Code of Ethics, https://www.acm.org/code-of-ethics (accessed Sep. 28, 2023).

[9] "MFRC522 RFID Reader with Arduino Tutorial | Random Nerd Tutorials," *Random Nerd Tutorials*, Mar. 23, 2016. https://randomnerdtutorials.com/security-access-using-mfrc522-rfid-reader-with-arduino/

[10] R. S. Sanketh, "An Introduction to RFID," *AUTONOMOUS ROBOTICS*, May 26, 2020. https://medium.com/autonomous-robotics/an-introduction-to-rfid-dc6228767691#:~:text=Working (accessed Dec. 07, 2023).

[11] “I2C LCD with ESP32 on Arduino IDE - ESP8266 compatible | Random Nerd Tutorials,” Feb. 01, 2019. https://randomnerdtutorials.com/esp32-esp8266-i2c-lcd-arduino-ide/

[12] “[solved] WSL V3: A fatal error occurred: Failed to connect to ESP32-S3: Wrong boot mode detected (0x14),” *Heltec Automation Technical Community*, Nov. 28, 2022. http://community.heltec.cn/t/solved-wsl-v3-a-fatal-error-occurred-failed-to-connect-to-esp32-s3-wrong-boot-mode-detected-0x14/11962/6 (accessed Dec. 07, 2023).

[13] “In-Depth: Interface One Channel Relay Module with Arduino,” *Last Minute Engineers*, Nov. 07, 2019. https://lastminuteengineers.com/one-channel-relay-module-arduino-tutorial/

[14] D. Workshop, “ESP NOW - Peer to Peer ESP32 Communications,” *DroneBot Workshop*, Apr. 03, 2022. https://dronebotworkshop.com/esp-now/

[15] “Cloud Firestore Data model,” Firebase. https://firebase.google.com/docs/firestore/data-model

[16] “Firebase-ESP32/examples/Authentications/SignInAsUser/EmailPassword/EmailPassword.inoat master · mobizt/Firebase-ESP32,” GitHub. https://github.com/mobizt/Firebase-ESP32/blob/master/examples/Authentications/SignInAsUser/EmailPassword/EmailPassword.ino (accessed Dec. 07, 2023).

[17] “Installation & Setup in JavaScript | Firebase Realtime Database,” Firebase. https://firebase.google.com/docs/database/web/start

[18] “ESP Modules | Espressif Systems,” *www.espressif.com*. https://www.espressif.com/en/products/modules/esp-wroom-02/surf.aspx (accessed Dec. 07, 2023).

Appendix A: Requirements & Verification Tables

User Interface Subsystem

Requirements	**Verification**
Verify that the camera system can detect and decode QR codes in 4 seconds or less	● Hold the QR code for a component in front of the system camera ● Verify that the Arduino console identifies the QR code by printing a message out ● Verify that the Arduino console identifies the correct component has been scanned
Verify that the LCD can display a message stating that the box has been unlocked after tapping iCard	● Tap iCard on the RFID reader to unlock the solenoid lock ● Verify that the LCD displays a message stating that the box has been unlocked

Table 8: Requirements and Verification Table for User Interface Subsystem

Control Subsystem

Requirements	**Verification**
The UID obtained from the RFID scanner should be mapped to a student in less than 3.5 seconds	● Scan iCard on system RFID reader ● Verify that the LCD prints the correct student's name within 3.5 seconds
When an authorized user taps their iCard, the relay should enable, closing the circuit with the solenoid and unlocking the box	● Tap the iCard of an authorized student on the RFID reader ● Verify that the LCD indicates the box is opened ● Verify that a HIGH enable signal has been sent to the relay ● Verify that the solenoid unlocks and the box is accessible
When an unauthorized user taps their iCard, the relay should not enable, leaving the circuit open so that the solenoid and box remain locked	● Tap the iCard of an unauthorized student on the RFID reader ● Verify that the LCD indicates the user is unauthorized

	● Verify the enable signal remains LOW to the relay ● Verify that the solenoid does not unlock and that the box is inaccessible

Table 9: Requirements and Verification Table for Control Subsystem

Power Supply Subsystem

Requirements	**Verification**
The 5V from the AC/DC adapter should drop to 3.3V ± 0.5 through our linear regulator circuit	● Plug the AC/DC adapter to the barrel jack on the PCB to supply 5V to the PCB ● Use a multimeter to measure one test point at the output of the linear regulator circuit and one ground point and check that the voltage shown is 3.3V

Table 10: Requirements and Verification Table for Power Supply Subsystem

Database Subsystem

Requirements	**Verification**
The database should correctly update to reflect component *checkout*, including the name of the student, the component borrowed, and the time checked out.	● Scan the iCard of an authorized student on the RFID scanner and wait for the box to unlock ● Hold up component to the camera and wait for the LCD to indicate that the component was checked out ● Inspect database to ensure component record was successively checked out to student ● Verify that the component is checked out to the correct student ● Verify that the time matches the time of check out
The database should correctly update to reflect component *return*.	● Scan the iCard of an authorized student on the RFID scanner and wait for the box to unlock ● Hold up component to the camera and wait for the LCD to indicate that the component was returned

	● Inspect database to ensure component record was successively returned to the main inventory
The system should not allow a student to borrow or return a component already checked out to another student.	● Scan the iCard of an authorized student on the RFID scanner and wait for the box to unlock ● Hold up a component to the camera and wait for the LCD to indicate that the component was checked out ● Inspect database to ensure component record was successively checked out to student ● Scan the iCard of an *another* authorized student on the RFID scanner and wait for the box to unlock ● Hold up that same component to the camera ● Verify that the LCD indicates the component cannot be checked out because it is checked out to another student already

Table 11: Requirements and Verification Table for Database Subsystem

Appendix B: Expanded Cost & Schedule Table

Expanded Cost Table

Description	Manufacturer	Manufacturer Product #	#	Unit Cost ($)
RF MOD CHIP	Espressif Systems	ESP32-WROOM-32 (16MB)	1	$4.50
LITH-ION 3.7V	Adafruit Industries	ICR18650 4400mAh 3.7V	1	$19.95
AC/DC ADAPTER	Tri-Mag, LLC	L606H-120	1	$5.82
IC BATT LI-ION	Microchip Technology	MCP73812T-420I/OT	1	$0.73
CONVERTER 12V	MEAN WELL USA	EPS-15-12	1	$9.92
REG LIN 3.3V	onsemi	NCP115ASN330T2G	1	$0.36
3V RELAY	Pi Supply	PIS-1267	1	$5.59
SOLENOID LOCK	Adafruit Industries	5065	1	$7.50
LED RGB	Kingbright	APHF1608LSEEQBDZGKC	1	$0.76
LCD MOD 32	Lumex Components	LCM-S01602DTR/M	1	$10.47
WI-FI CAMERA	M5STack Technology	U109	1	$10.95
ESP32-CAM	CANADUINO	ESP32-CAM WiFi BT BLE	1	$9.72
RFID READER	DLP Design Inc.	DLP-RFID2	1	$37.95
RES 1K OHM	Panasonic Electronic	ERJ-PA3J102V	10	$0.12
RES 10K OHM	Panasonic Electronic	ERJ-P06J103V	10	$0.13
RES 100K OHM	YAGEO	RV1206FR-07100KL	10	$0.31
CAP CER 22UF	Murata Electronics	GRM188R61A226ME15D	10	$0.22
CAP CER 1UF	Murata Electronics	GRM155R60J105KE19D	10	$0.10
CAP CER 10UF	Murata Electronics	GRM31CR71E106KA12L	10	$0.32
CAP CER 0.1UF	Samsung Mechanics	CL10B104KA8NNNC	10	$0.10

Table 12: Costs of Parts

Total Cost: **$137.22**

Expanded Schedule

Week	Task	Assigned Member
Sept. 24 - Sept. 30	Order Components	All
	Start Control Subsystem Schematic	Krish
	Start Power Supply Subsystem Schematic	Rushil
	Start User Interface Subsystem Schematic	Rohan
Oct. 1 - Oct. 7	Finish initial PCB design	All
	Design Review	All
	PCB Review	All
Oct. 8 - Oct. 14	Final Machine Shop revisions to build container after getting components	Rohan
	Make necessary revisions after PCB review	All
	First Round PCBway Order	Rushil
	Set up MySQL backend and link with Google Form	Krish
Oct. 15 - Oct. 21	Work on ESP32 programming and interfacing with backend	Krish
	Work on ESP32 and camera integration	Rushil
	Work on QR code identification with camera	Rohan
	Make PCB Revisions	All
	Second Round PCBway Order	Krish

Oct. 22 - Oct. 28	Work on integrating RFID reader and ESP32 for student identification	Krish + Rohan
	Work on integrating LCD and ESP32	Rushil
	Continue working on QR code identification and camera + ESP32 integration	Krish + Rushil
	Make PCB revisions	All
	Third Round PCBway Order	Rohan
Oct. 29 - Nov. 4	Start assembly	All
	Begin unit testing Control Subsystem	Krish
	Begin unit testing Power Supply Subsystem	Rohan + Rushil
Nov. 5 - Nov. 11	Finish assembly	All
	Begin unit testing User Interface Subsystem	Krish + Rohan
	Test functionality of subsystems working together	All
	Fix bugs	All
Nov. 12 - Nov. 18	Mock Demo	All
	Fix Bugs	All
Nov. 19 - Nov. 25	Fall Break	None
Nov. 26 - Dec. 2	Final Demo	All
Dec. 3 - Dec. 9	Final Presentation	All

Table 13: Detailed Weekly Schedule