Modularized Electronic Locker Project Individual Progress Report

Team 61

1. Introduction

For our Modularized Electronic Locker, we have two hardware modules - control module and locker module and three software modules - cloud data platform, touchscreen GUI design and embedded programming for ESP32. My responsibility covers all of the modules. So far, I have designed the schematics for both of the hardware modules, built the MySQL server on and started the touchscreen GUI design. I have also done research into the FreeRTOS that the ESP32 will be using. The work left for me to do is to work with the Machine Shop to finish the physical prototype of the locker system, finish the GUI on Raspberry PI 4, and develop and debug the program on both RPI4 and ESP32 to support communication over RS485.

2. Design

2.1. Hardware

2.1.1. PCB

I designed the schematics for the PCB and have taken our original PCB design to a teaching assistant for improvements before sending it to the manufacturer. Both of the modules share the same PCB board. Different components will be soldered on the board when used in different modules. The PCB board has two converters which step down voltage from 12V to 5V for RPI4 and 3.3V for ESP32. The 12V-to-5V converter (Figure 1) can sustain current up to 3.1A to support the power rating (5V, 3.1A) of RPI4. I have used a trace width calculator (Figure 6) to ensure the thermal performance of our PCB.

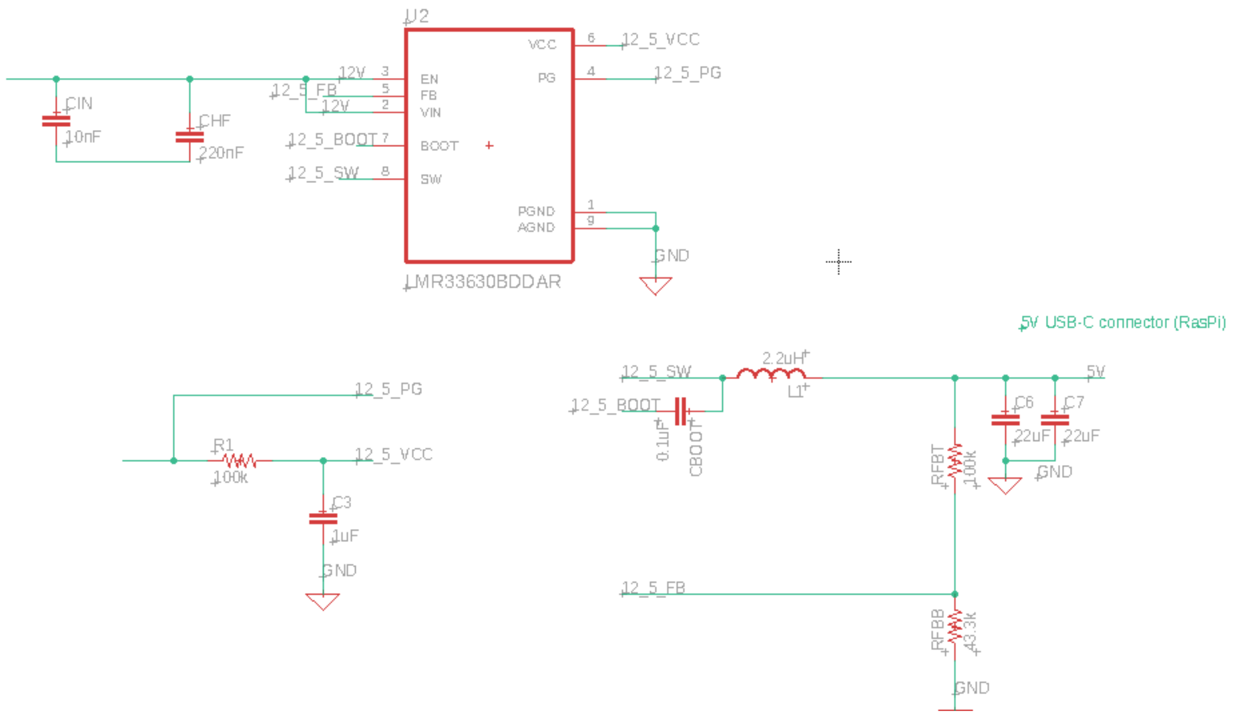


Figure 1. V-to-5V Converter

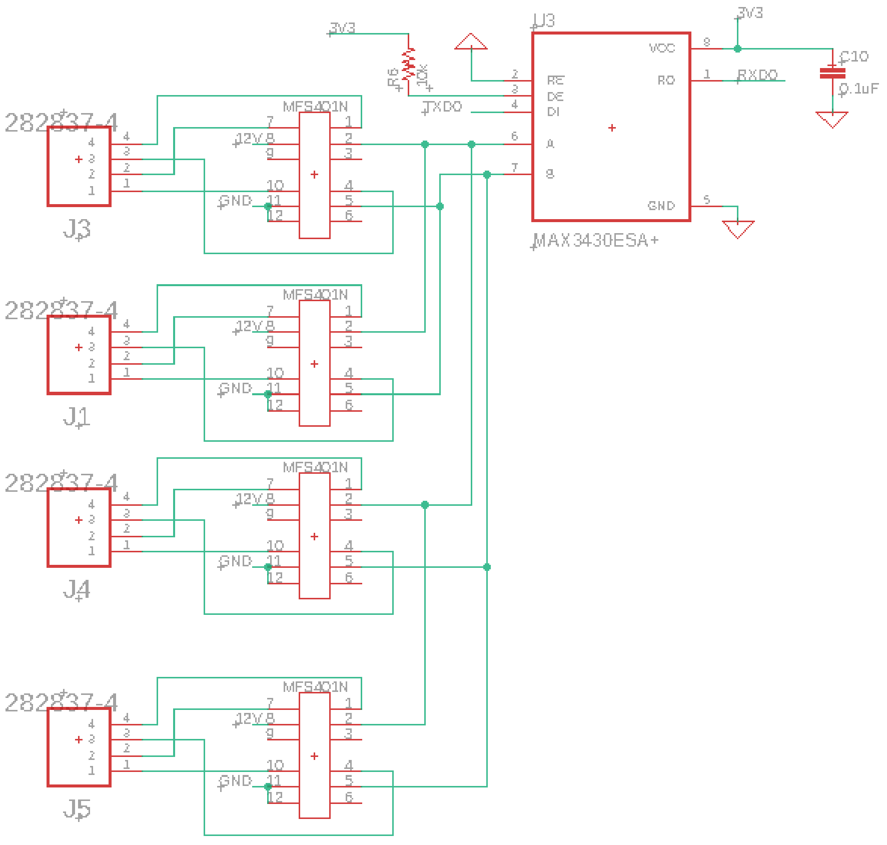


Figure 2. RS-485 Transceiver and Locker 4-Pin Interface

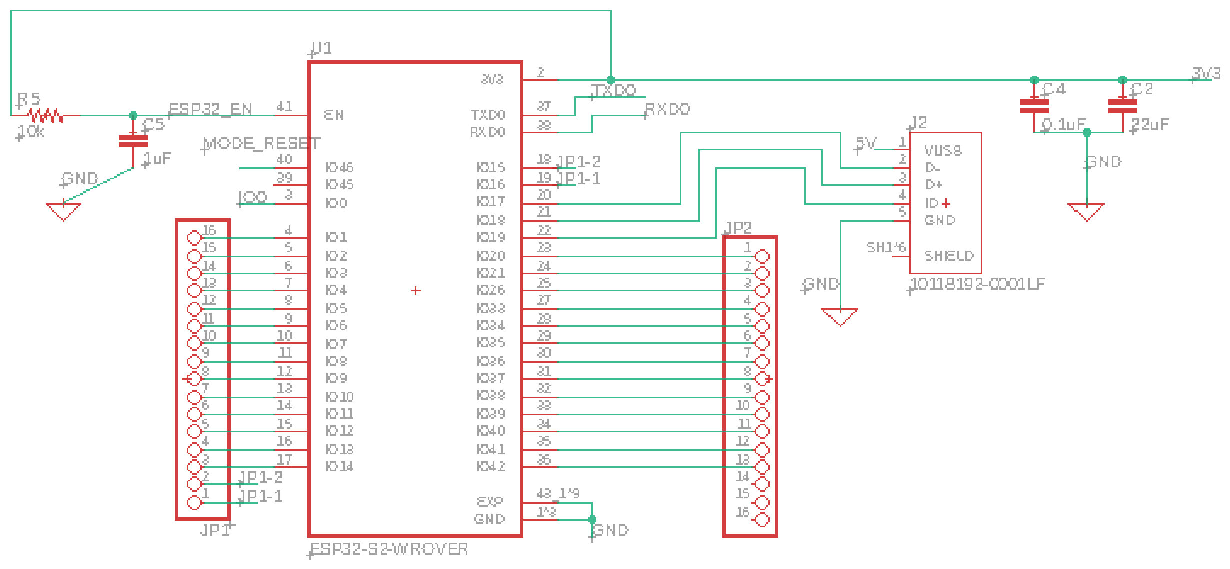


Figure 3. ESP32 and MicroUSB Connector

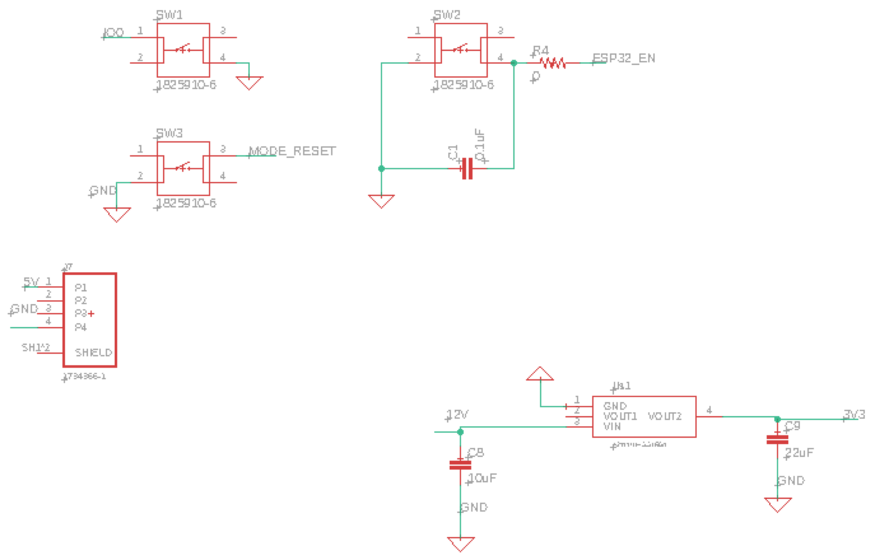


Figure 4. Reset Button, Enable Button, Programmable Button and 12V-to-3.3V Converter

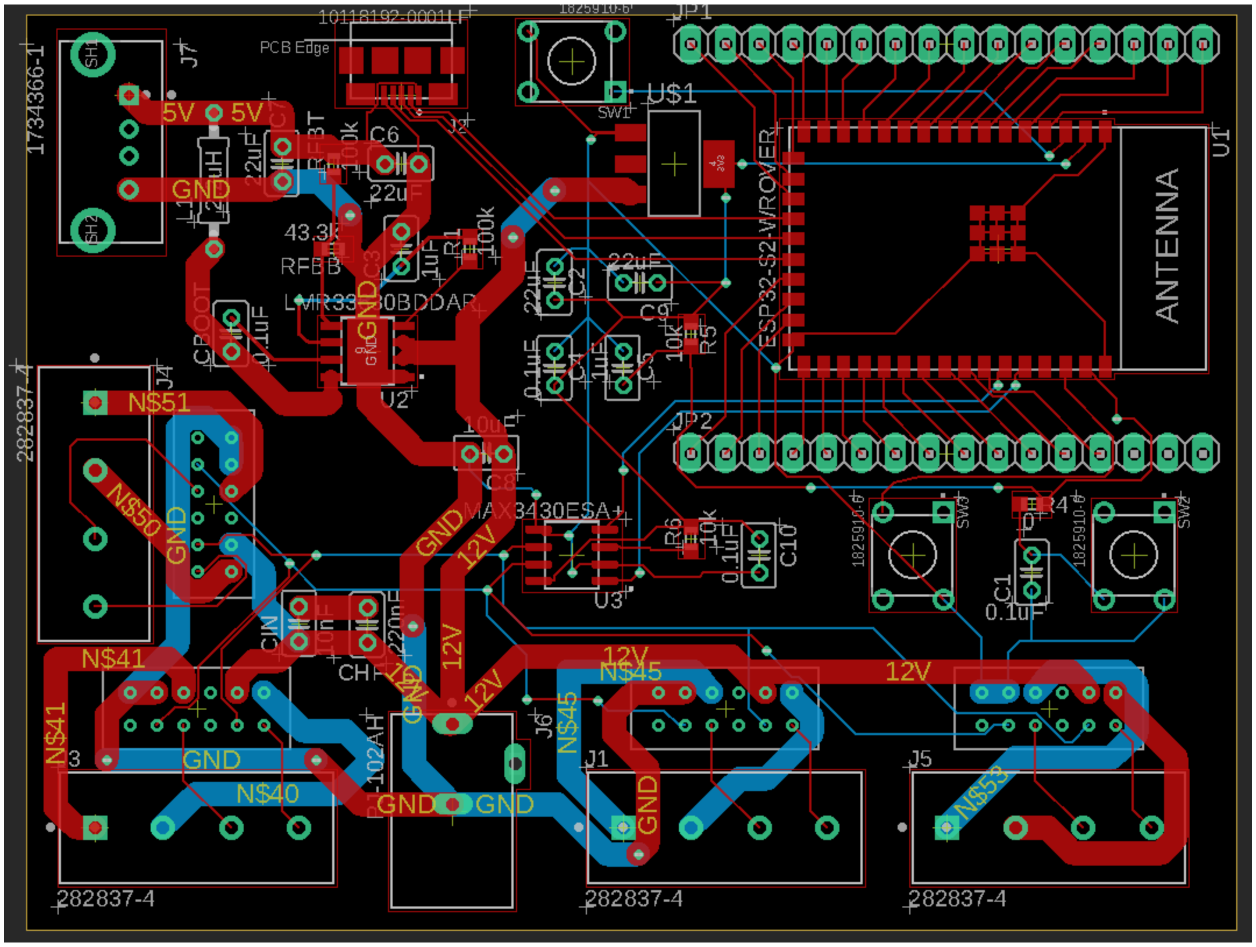
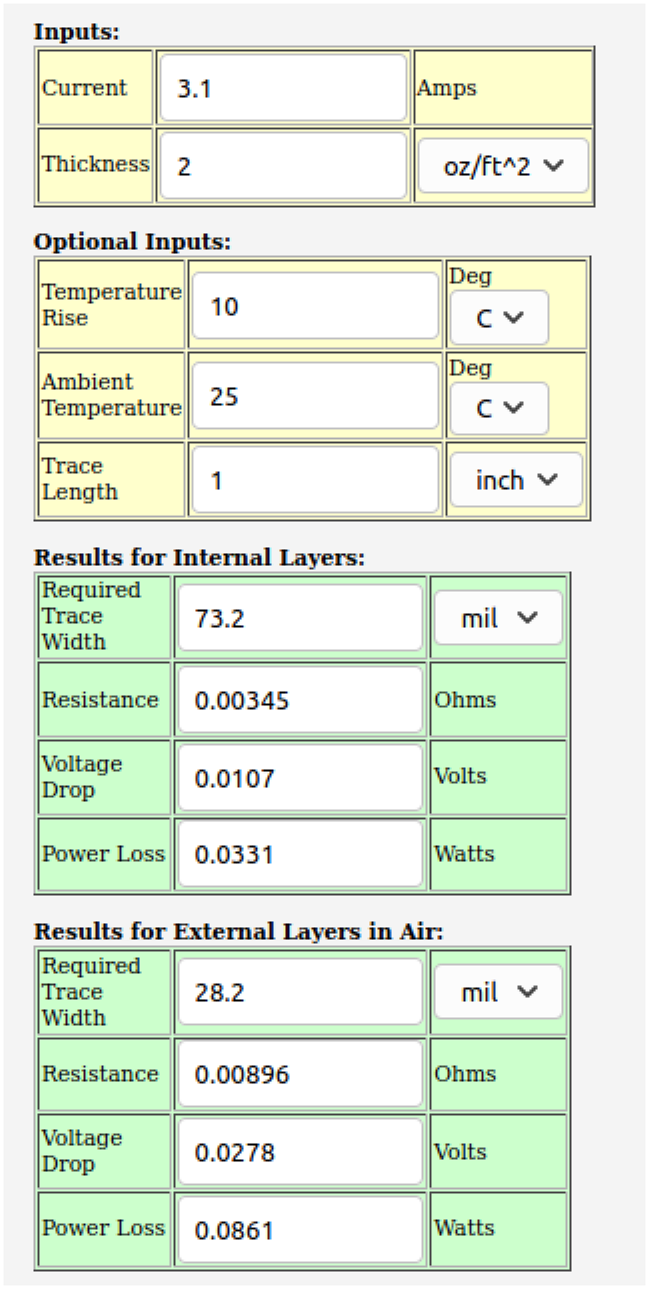


Figure 5. Final PCB Design



Inputs:

Current	3.1	Amps
Thickness	2	oz/ft^2

Optional Inputs:

Temperature Rise	10	Deg C
Ambient Temperature	25	Deg C
Trace Length	1	inch

Results for Internal Layers:

Required Trace Width	73.2	mil
Resistance	0.00345	Ohms
Voltage Drop	0.0107	Volts
Power Loss	0.0331	Watts

Results for External Layers in Air:

Required Trace Width	28.2	mil
Resistance	0.00896	Ohms
Voltage Drop	0.0278	Volts
Power Loss	0.0861	Watts

Figure 6. PCB Trace Width Calculator on www.4pcb.com

2.1.2. Locker Module

The locker module is based on ESP32 which will be running FreeRTOS. We chose EPS32 because it is cheap but powerful as a microcontroller. We can also benefit from its large community by learning from numerous tutorials and various open-source firmware for this chip. We chose ESP32-S2 so that we can use the USB to flash program into the microcontroller.

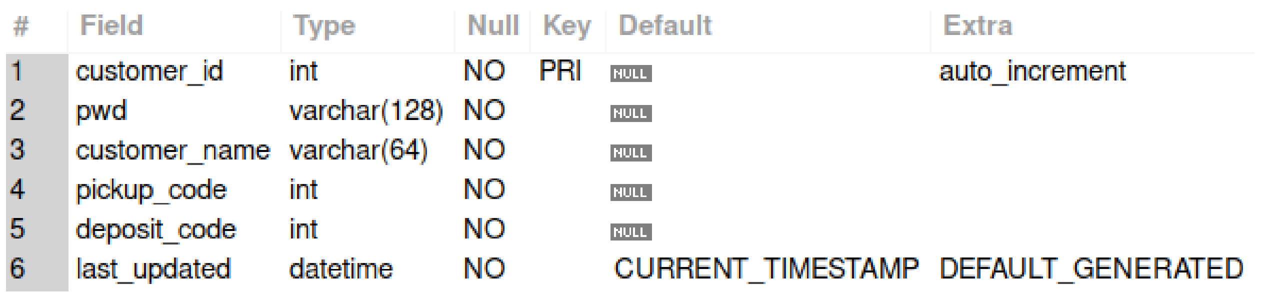
2.1.3. Control Module

The control module is based on Raspberry Pi 4 which is a ARM system running a Linux system. The reason we pick RPI4 rather than ESP32 is for the computation power. It syncs the user data with the cloud server, processes input from users through touchscreen and keypad, calculates all pixels for the touchscreen, takes pictures whenever the passive infrared sensor is triggered, and sends unlock commands to locker modules. ESP32 has a maximum frequency of 240 MHz and cannot handle this much data. (Espressif System) In contrast, RPI 4 can work at 1.5 GHz. (Raspberry Pi (Trading) Ltd.)

2.2. Software

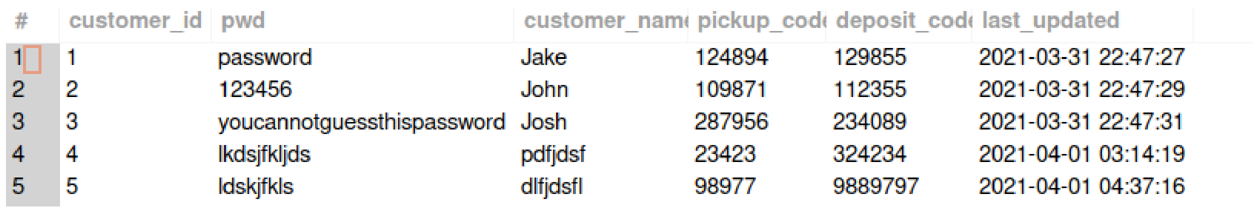
2.2.1. Cloud Service

There will be two software services running outside the lockers. They are website frontend service and the MySQL backend service. I set up the MySQL server in my home and exposed the port 3306 for Joshua's website frontend to connect. I have shown the schema in Figure 7 and the example data in Figure 8.



#	Field	Type	Null	Key	Default	Extra
1	customer_id	int	NO	PRI	NULL	auto_increment
2	pwd	varchar(128)	NO		NULL	
3	customer_name	varchar(64)	NO		NULL	
4	pickup_code	int	NO		NULL	
5	deposit_code	int	NO		NULL	
6	last_updated	datetime	NO		CURRENT_TIMESTAMP	DEFAULT_GENERATED

Figure 7. MyMQL Schema for Locker Database



#	customer_id	pwd	customer_nam	pickup_cod	deposit_cod	last_updated
1	1	password	Jake	124894	129855	2021-03-31 22:47:27
2	2	123456	John	109871	112355	2021-03-31 22:47:29
3	3	youcannotguessthispassword	Josh	287956	234089	2021-03-31 22:47:31
4	4	lkdsjfkljds	pdfjdsf	23423	324234	2021-04-01 03:14:19
5	5	ldskjfkls	dlfjdsfl	98977	9889797	2021-04-01 04:37:16

Figure 8. Example Content in MySQL Database

2.2.2. Graphic User Interface with Touchscreen

There are a few libraries to create GUI on an embedded system. I decided to go with LVGL since one of our TAs is familiar with it. LVGL can be implemented on microcontrollers with no OS and Linux with FrameBuffer.

2.2.3. Embedded Programming in ESP32

Although we do not need FreeRTOS to deal with RS485 communication, FreeRTOS provides a kernel, a scheduler and many more features to ensure expandability and modularity if we add more sensors to the module. (FreeRTOS) (The PCB is equipped with connectors to expose all unused pins of ESP32 for any add-on components.)

3. Finished Verification

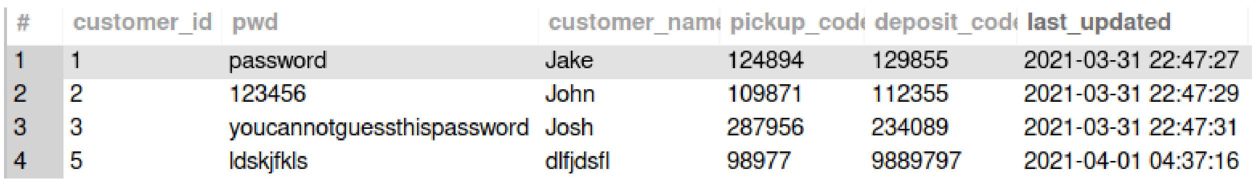
3.1. MySQL Database

There are 4 basic operations in the MySQL database: check, update, insert, delete. Figure 8 has shown that we can perform a check operation. In the following Figure 9 - Figure 11, we can see the results after a deletion, an insertion and an update.

```
SELECT * FROM locker; // check
DELETE FROM locker WHERE customer_id = 4;
SELECT * FROM locker;
INSERT INTO locker(pwd, customer_name, pickup_code,
deposit_code)
VALUES ("thisisapassWord", "John Doe", 918764, 128937);
```

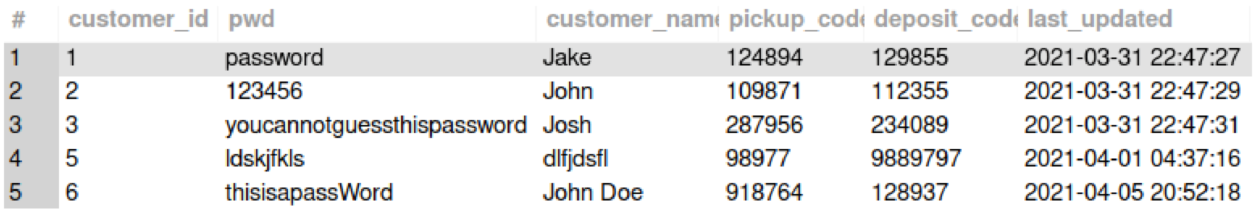
```sql
SELECT * FROM locker;
UPDATE locker
SET pickup_code = 999999, deposit_code = 222222
WHERE customer_id = 6;
SELECT * FROM locker;
```

Table 1. SQL Code to Perform Database Verification



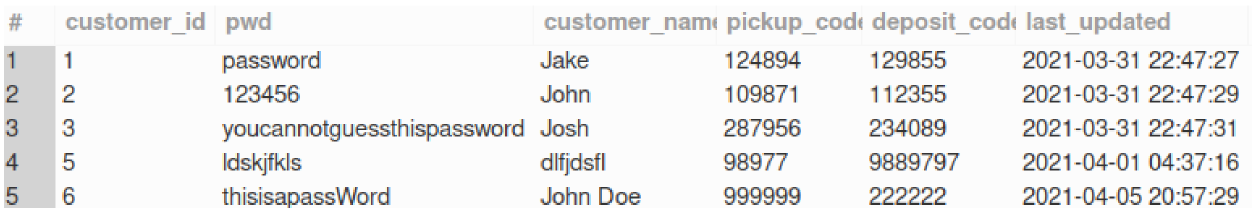
#	customer_id	pwd	customer_nam	pickup_cod	deposit_cod	last_updated
1	1	password	Jake	124894	129855	2021-03-31 22:47:27
2	2	123456	John	109871	112355	2021-03-31 22:47:29
3	3	youcannotguessthispassword	Josh	287956	234089	2021-03-31 22:47:31
4	5	ldskjfkls	dlfjdsfl	98977	9889797	2021-04-01 04:37:16

Figure 9. MySQL Database: Result after a Deletion



#	customer_id	pwd	customer_nam	pickup_cod	deposit_cod	last_updated
1	1	password	Jake	124894	129855	2021-03-31 22:47:27
2	2	123456	John	109871	112355	2021-03-31 22:47:29
3	3	youcannotguessthispassword	Josh	287956	234089	2021-03-31 22:47:31
4	5	ldskjfkls	dlfjdsfl	98977	9889797	2021-04-01 04:37:16
5	6	thisisapassWord	John Doe	918764	128937	2021-04-05 20:52:18

Figure 10. MySQL Database: Result after an Insertion



#	customer_id	pwd	customer_nam	pickup_cod	deposit_cod	last_updated
1	1	password	Jake	124894	129855	2021-03-31 22:47:27
2	2	123456	John	109871	112355	2021-03-31 22:47:29
3	3	youcannotguessthispassword	Josh	287956	234089	2021-03-31 22:47:31
4	5	ldskjfkls	dlfjdsfl	98977	9889797	2021-04-01 04:37:16
5	6	thisisapassWord	John Doe	999999	222222	2021-04-05 20:57:29

Figure 11. MySQL Database: Result after an Update

3.2. LVGL Simulation

I have tested the LVGL in the simulator on my Ubuntu 20.04 desktop. I can use pop-up messages to prompt users to interact and use buttons to receive their feedback shown in Figure 12. Figure 13 shows that we can have dropdown and dark mode for more interactions with users.

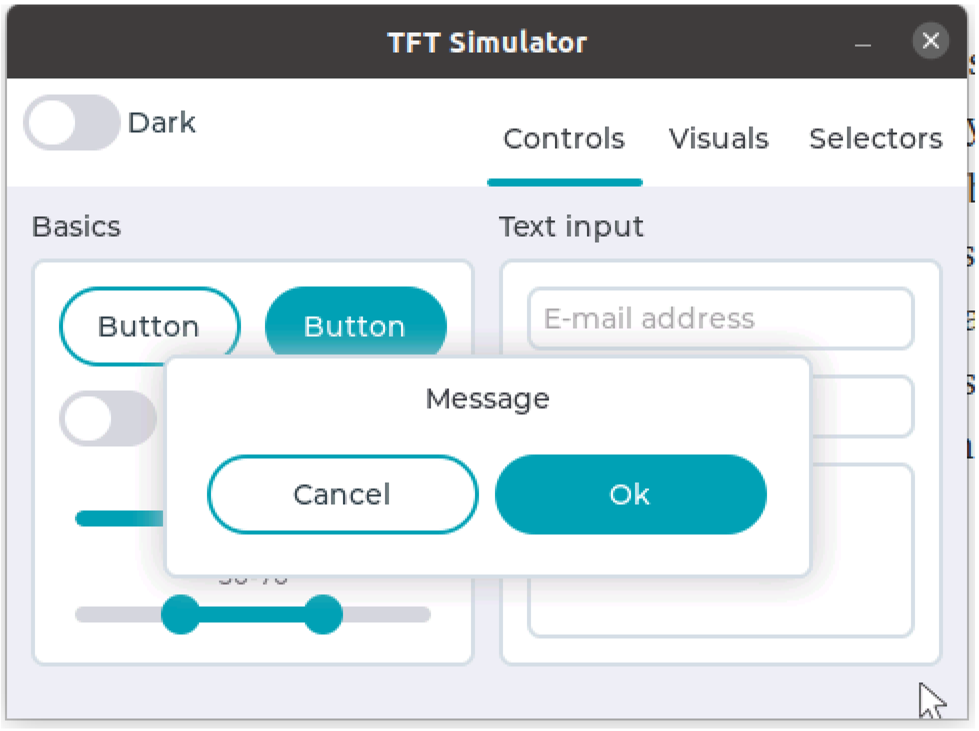


Figure 12. Simulation of LVGL GUI with Message and Button

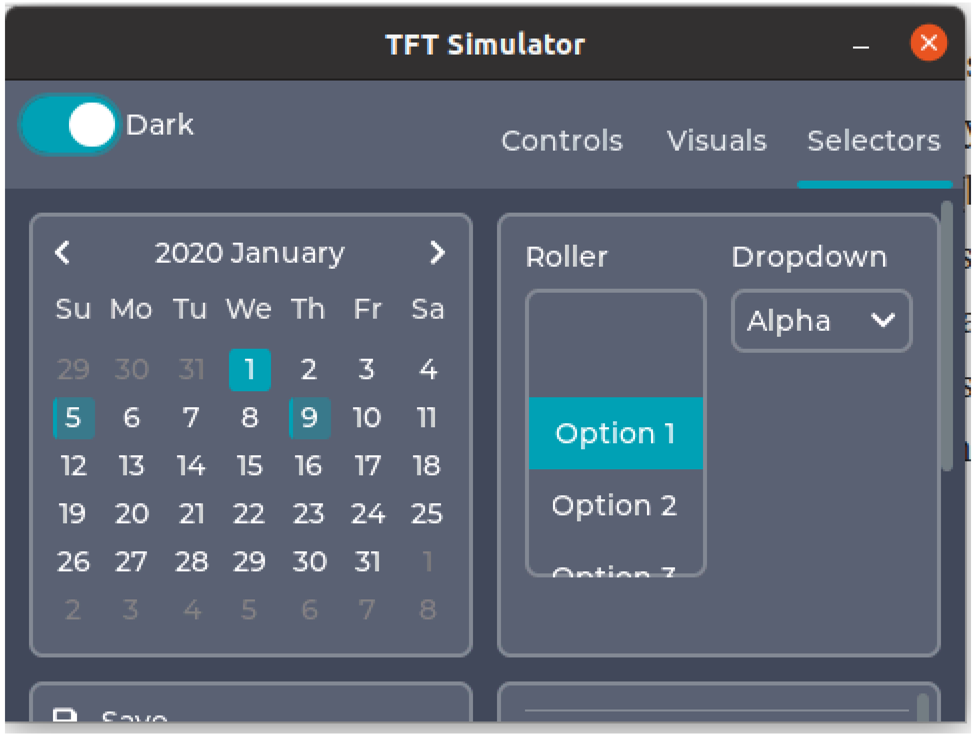


Figure 13. Simulation of LVGL GUI with Dropdown and Dark Mode

4. Verifications to Perform

4.1. LVGL GUI on RPI4 with Touchscreen

We will have to compile LVGL GUI program for ARM platform and calibrate the touchscreen on RPI4.

4.2. Communication on RS485

We should verify communication on RS485 by sending an unlock signal to different locker modules with different addresses and check if the matching locker unlocks.

4.3. Camera with Motion Detection

Camera should be able to capture images if the PIR sensor has detected a motion. We can verify it by waving a hand in front of the sensor and checking the log directory in the RPI4.

5. Conclusion

To sum up, I have made great progress on both the hardware side and the software side. To avoid electrocution mentioned in the Ethics section in the design review, I have already put jumpers in the PCB design to physically detach signal and power line from any interface that is exposed. Also, with the addition of the camera, we can record every personnel who had access with the locker and can provide it as evidence if the lockers are exploited for illegal activities. With these iterations in design, we are ruling out most of the ethics concerns. However, we welcome any comments or criticism to help me improve our Modularized Electronic Locker. I have attached a time table for better scheduling.

Week	**Josh**	**Jack**	**Jake**
4/4/21	Finish website layout	Assemble PCB and test PIR sensor	Finish LVGL GUI design and load them to RPI4

4/11/21	Debug GUI/Software, write code to activate camera with PIR trigger	Finish building physical locker with machine shop.	Calibrate touchscreen, implement communication on RS485
4/18/21	Ensure subsystems work together	Ensure subsystems work together	Ensure subsystems work together
4/25/21	Debugging/Stress Testing	Debugging/Stress Testing	Debugging/Stress Testing
5/3/21	Begin Report	Prepare Final Presentation	Prepare Final Presentation

Table 2. Timetable for Following Weeks

6. Citation

Bibliography

Espressif System. “ESP23-S2 Family Datasheet.” https://www.espressif.com/sites/default/files/documentation/esp32-s2_datasheet_en.pdf. Accessed 05 04 2021.

FreeRTOS. “FreeRTOS FAQ - What is This All About?” https://www.freertos.org/FAQWhat.html.

Raspberry Pi (Trading) Ltd. “Raspberry Pi 4 Model B Datasheet.” https://www.raspberrypi.org/documentation/hardware/raspberrypi/bcm2711/rpi_DATA_2711_1p0_preliminary.pdf. Accessed 05 04 2021.