Dwight Look College of
ENGINEERING
TEXAS A&M UNIVERSITY

ECEN 404 Final Presentation
Team 4: Automatic Solar Lighting System

Team Members: Atahan Bakanyildiz, Romi Gilat, Cedar Maxwell, Nick Miller
TA: Fahrettin Ay
Sponsor: Dr. Wohnyeok Jang

What is the Automatic Solar Lighting System?

Problem:

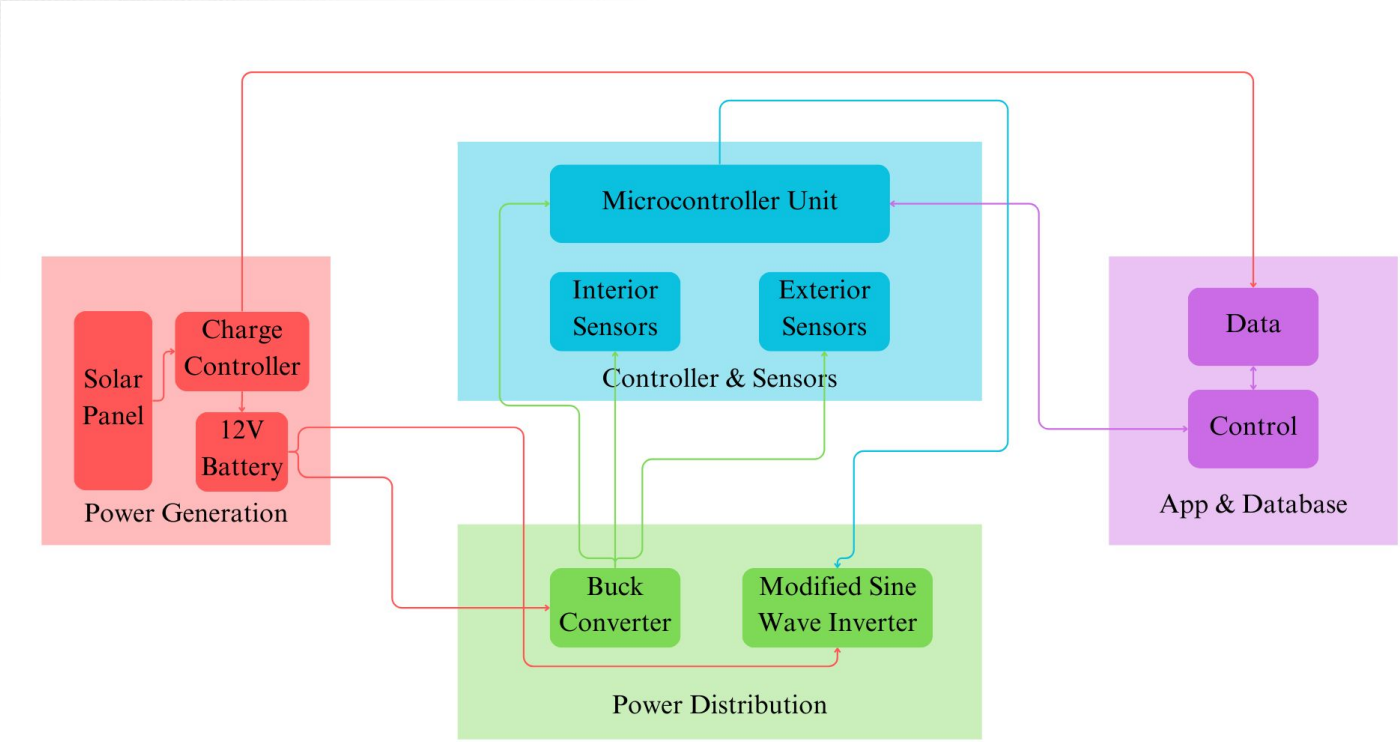
- Increasing grid demand and reliability issues make power outages more frequent.
- Integrating solar energy into homes ensures a dependable and independent power source.
- Solar power provides backup energy during outages, reduces electricity costs, and supports sustainability.

Solution:

- Solar panels provide up to 3 days of reliable power during emergencies.
- Motion-activated lighting for the foyer and patio ensures security and efficiency.
- Remote control via an app allows convenient access and management of the system.
- Enhances energy independence, cost savings, and sustainability for homeowners.

System Visual

Project/Subsystem Overview



Nick: Power Distribution

Romi: Controller & Sensors

Atahan: Power Generation

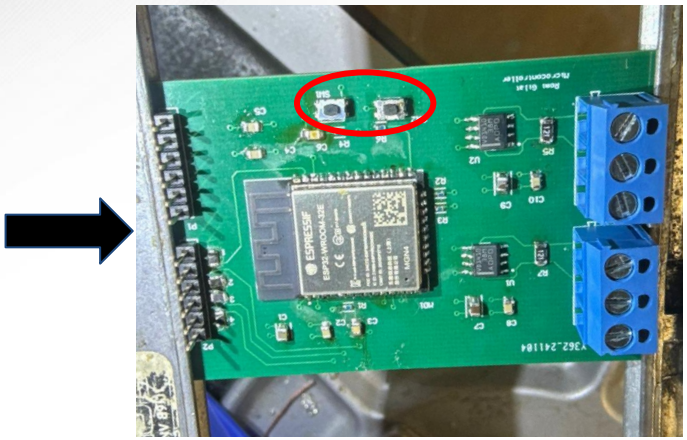
Cedar: App & Database

Microcontroller and Sensors

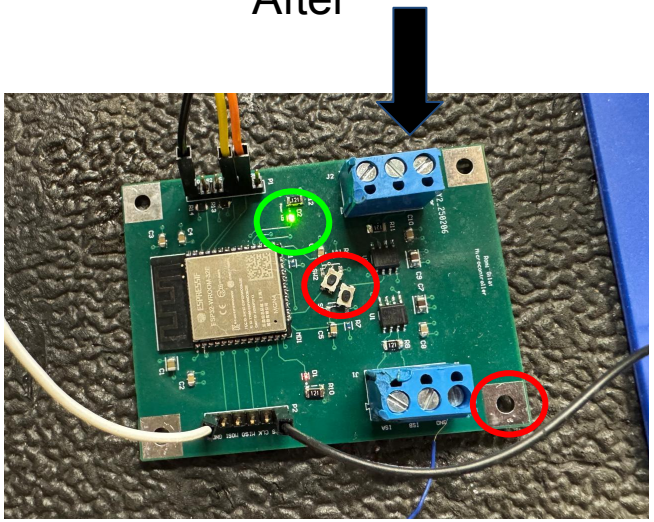
Component	Initial Design	Final Design
Microcontroller	- I2C, SPI and UART ports - wrong button footprint - no screw holes or LED	- Connects to app, power monitor, and sensors - Transmits battery data and sensors activation wireless to the app - functionally very similar to the initial design, with minor hardware corrections
Sensors	- no MCLR pinhole - no LED - no screw holes	- addition of screw holes - triggers at 100% hit rate - 2 hour warm up time
PCB run at 3V3 with 0.450 mA when drawing full power, which includes transmitting data to app, blinking LEDs, and communicating between devices		

Microcontroller

Before



After

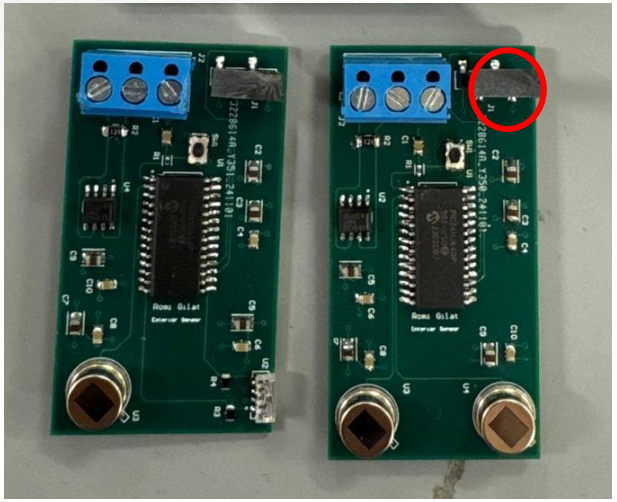


Key differences:

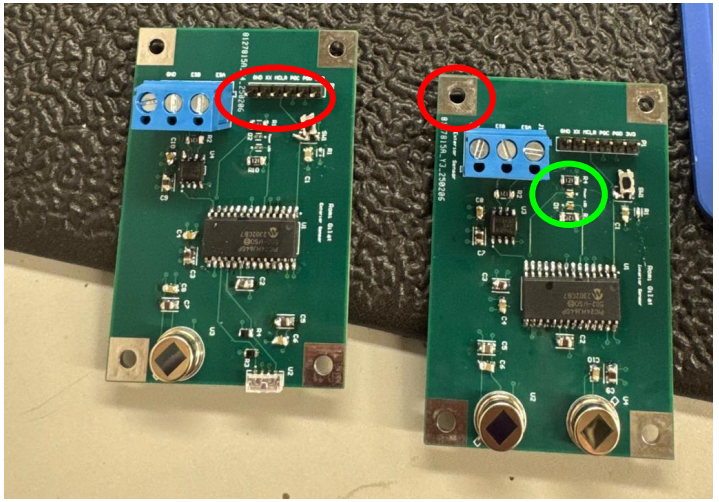
1. Added the addition of LED to show activity (ie, power, sending data)
2. Changing button footprint
3. Addition of screw holes
4. Additional power ports
5. Changed layout to accommodate better wiring design

Sensors

Before



After

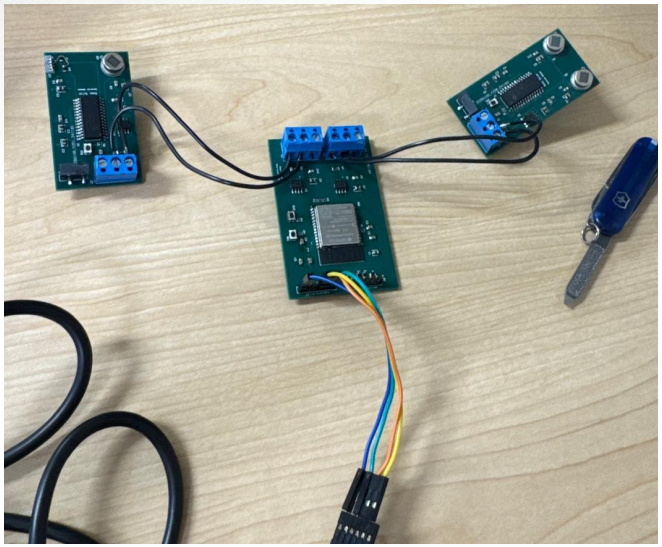


Key differences:

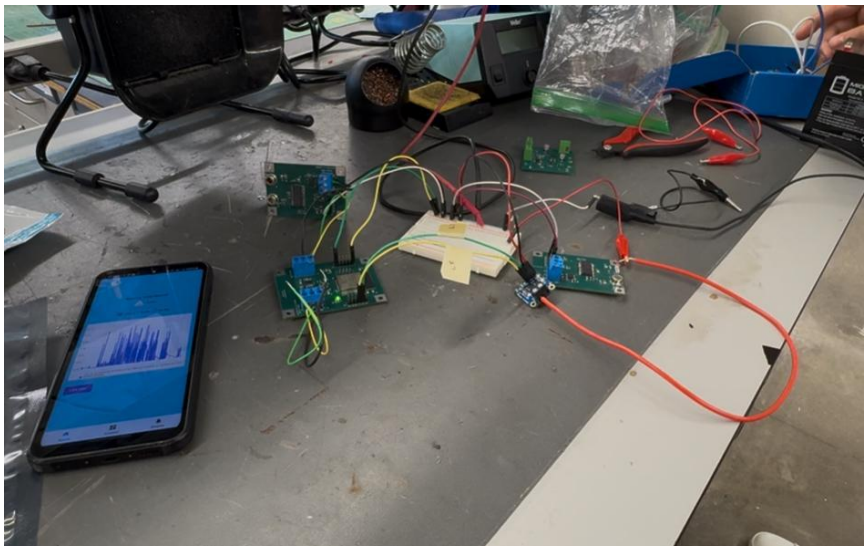
1. Added the addition of LED to show activity Changed layout to accommodate better wiring design (3 -> 6 pins)
2. Addition MCLR pin
3. Additional screw holes
4. Size of PCB is identical

Final Design

Before



After



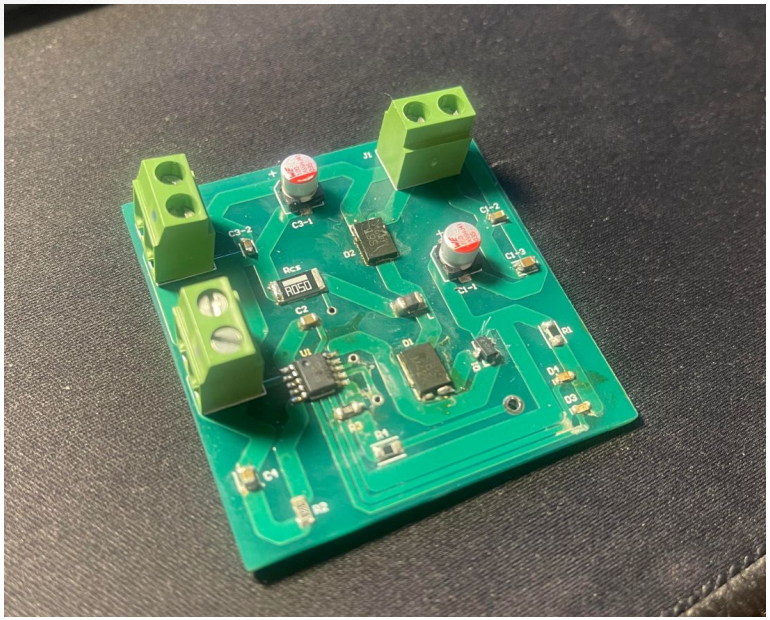
*Final design would not have breadboard, it is only used for connection to power supply.

Conclusions

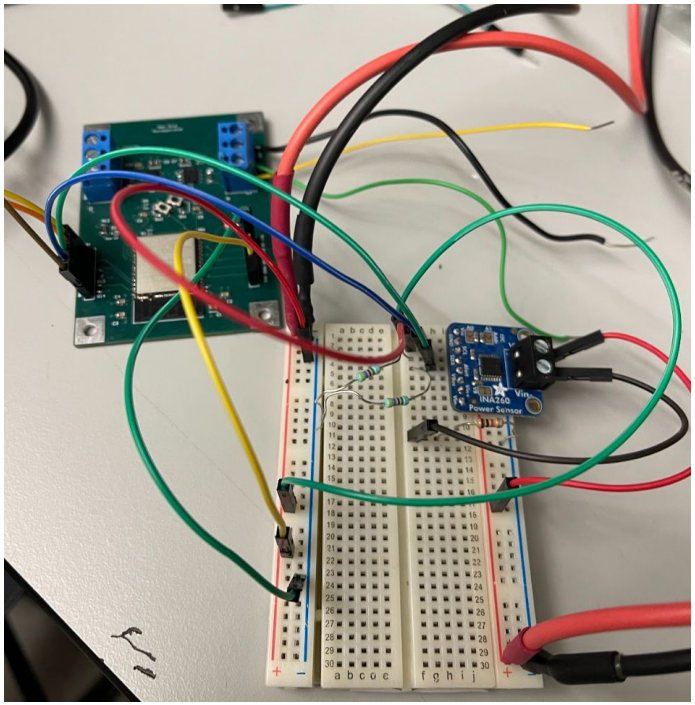
- Changes from 403 documentation:
  - Reorganization of PCB to allow better wiring for final system.
  - Addition of LED.
  - Addition of pin connections.

- Current status:
  - Successful integration with App and power distribution.
  - Sensors have 100% hit rate, with pre programed cooldown to prevent over triggering & power from being wasted.

Power Generation Engineering Design Accomplishments

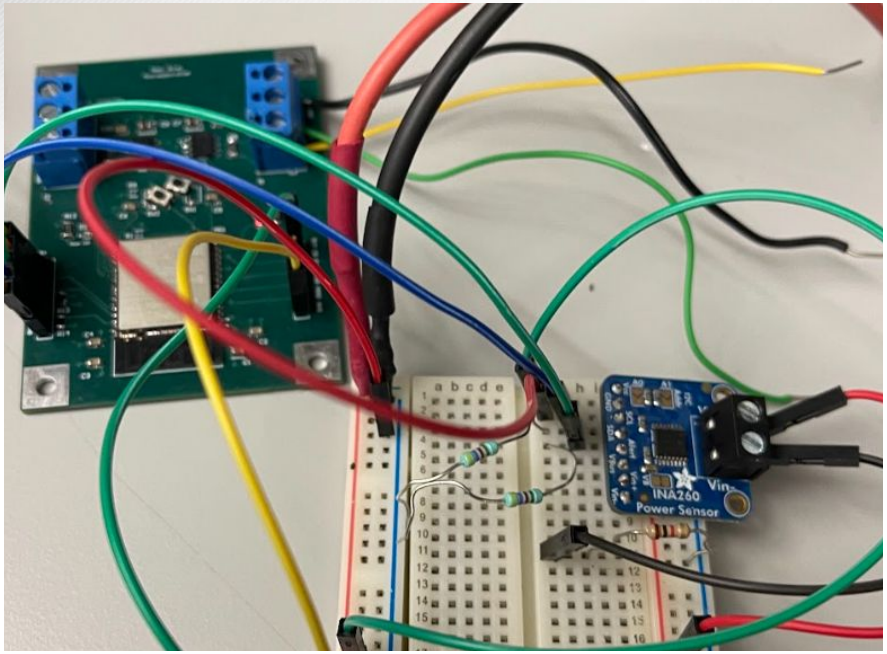


MPPT Solar Charge Controller



INA260 Power Monitor

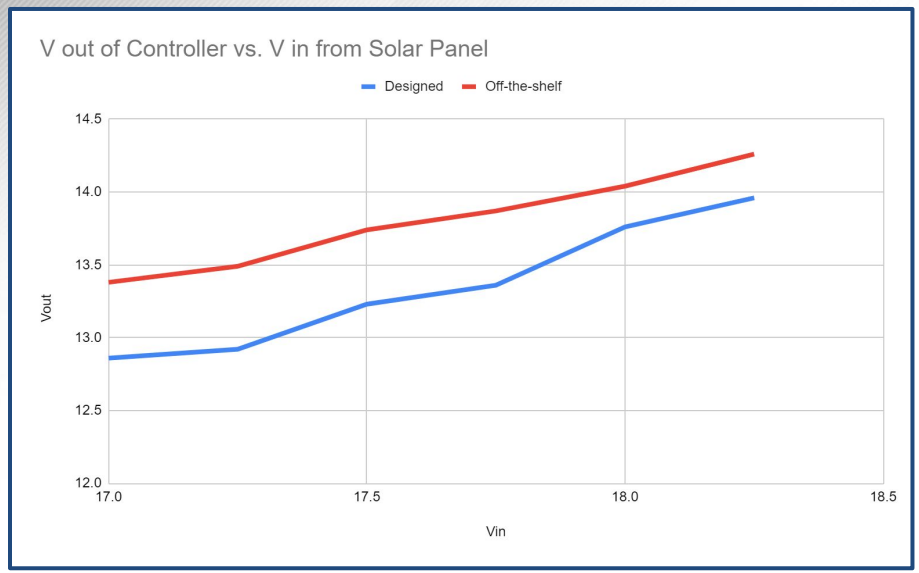
Power Monitor

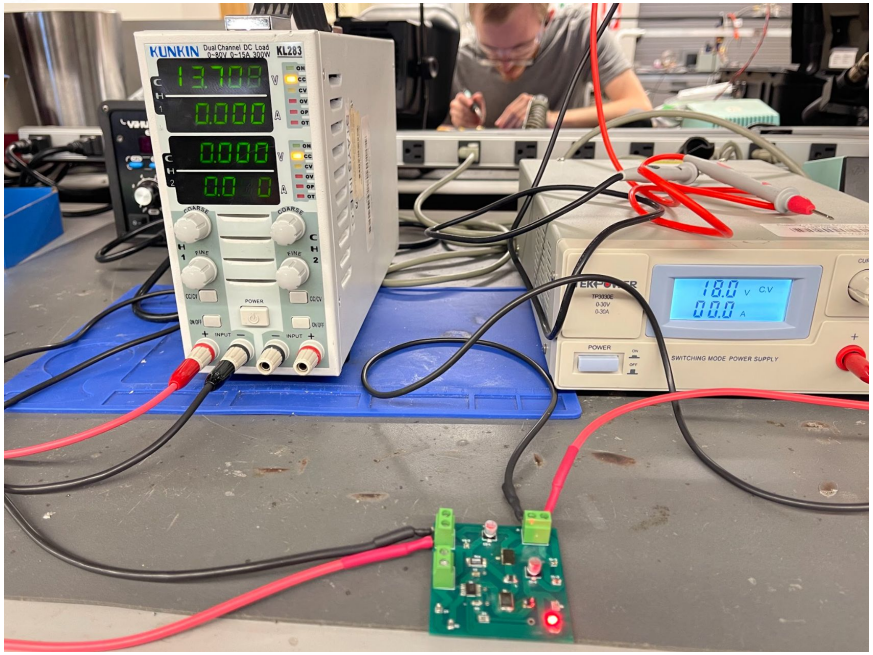



```
I (5354) ina260: Current: 0.000 A
I (5354) ina260: Voltage: 0.000 V
I (5354) ina260: Power:   0.000 W
I (10354) ina260: Current: 0.000 A
I (10354) ina260: Voltage: 0.998 V
I (10354) ina260: Power:   0.000 W
I (15354) ina260: Current: 0.000 A
I (15354) ina260: Voltage: 1.995 V
I (15354) ina260: Power:   0.000 W
I (20354) ina260: Current: 0.002 A
I (20354) ina260: Voltage: 1.994 V
I (20354) ina260: Power:   0.000 W
```

INA260 - Reads new data instantly
Communicates with MCU, doesn’t interfere with wifi and database

Charge Controller





Testing/Validation

- Power supply to test values with off the shelf version
- 3.56% discrepancy from off the shelf
- Overcharge protection is working (red LED)
- MPPT working (5 A in to controller, 6 A out)
- Charges/attempts charging with load (E-load representing Power Distribution subsystem)

Charge Controller

Testing Example

- 30 Minutes at 4:30 PM
- .02 V charged
- 1.5% - 2% charged

Charge Controller

Charge Controller		
9:30 AM- 10:00 AM		
Start Voltage (V)	End Voltage (V)	Charged (%)
12.8	12.81	0.7692307692
12.58	12.62	3.076923077
12.65	12.69	3.076923077
12.68	12.72	3.076923077
12.77	12.8	2.307692308
12:00 PM - 12:30 PM		
Start Voltage (V)	End Voltage (V)	Charged (%)
12.77	12.84	5.384615385
12.82	12.88	4.615384615
12.69	12.76	5.384615385
12.73	12.79	4.615384615
12.79	12.86	5.384615385
2:00 PM - 2:30 PM		
Start Voltage (V)	End Voltage (V)	Charged (%)
12.69	12.74	3.846153846
12.76	12.8	3.076923077
12.83	12.86	2.307692308
12.78	12.83	3.846153846
12.79	12.83	3.076923077
4:30 PM - 5:00 PM		
Start Voltage (V)	End Voltage (V)	Charged (%)
12.83	12.85	1.538461538
12.8	12.82	1.538461538
12.76	12.79	2.307692308
12.81	12.83	1.538461538
12.77	12.81	3.076923077

Testing/Validation:

- Tested 5 times at 4 different times
- Expected results of charge for each time

Power Generation Conclusions

- Changes from 403 documentation:
  - Power monitor is INA260 chip.
  - Charge controller design and chip changed from CONOPS/FSR to include MPPT.

- Current status:
  - Housing for subsystems still designing and 3D printing.

Dwight Look College of ENGINEERING
TEXAS A&M UNIVERSITY

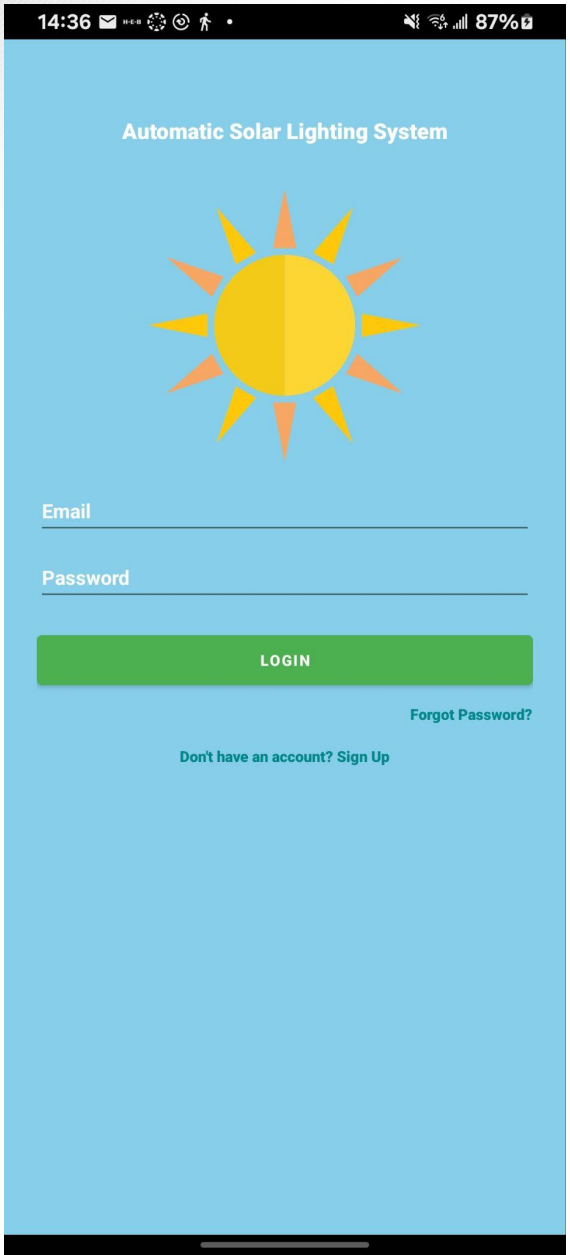
App & Database Engineering Design Accomplishments

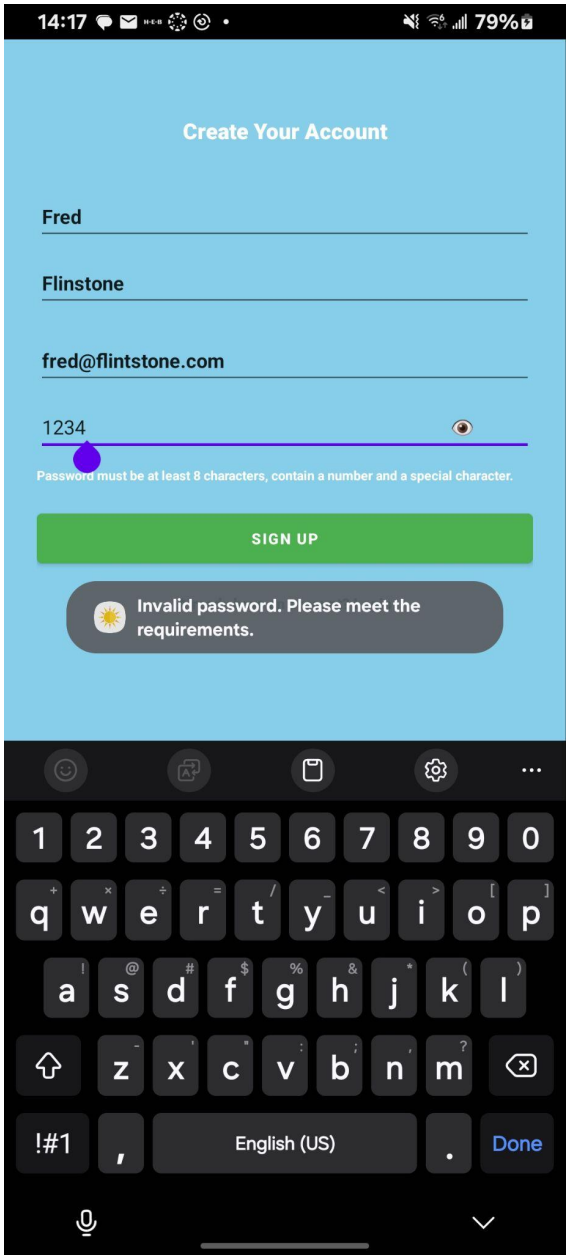
Main Screen:

- E-mail and Password login with Firebase
- Buttons to Forgot Password and Sign Up pages

Sign Up Screen:

- Name, E-mail, and Password sign up with Firebase, with enforced password requirements

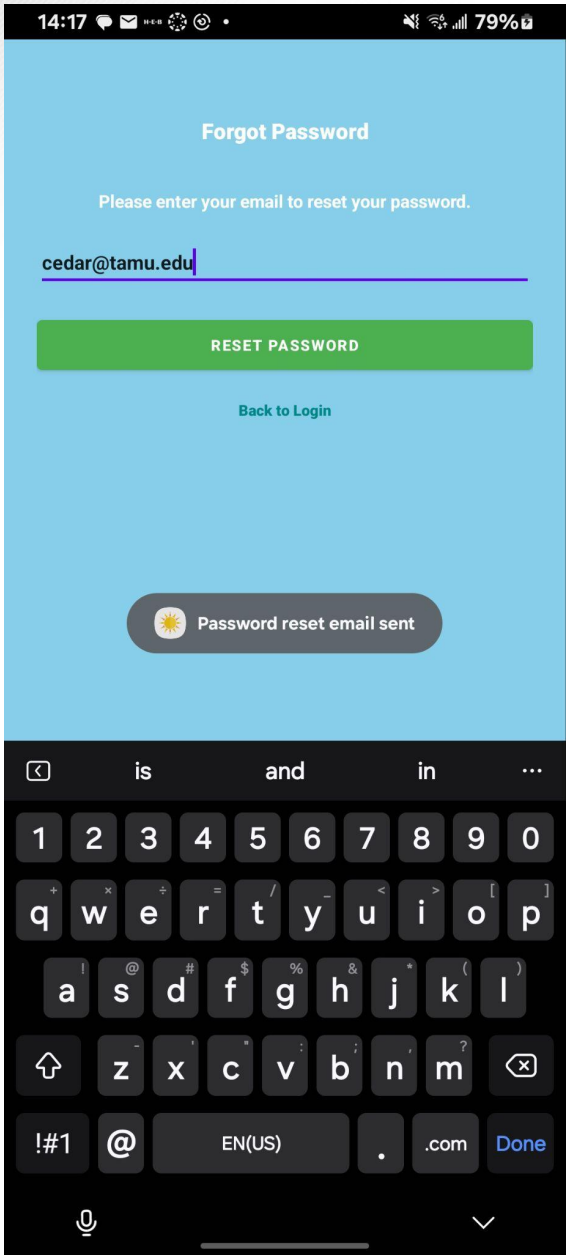


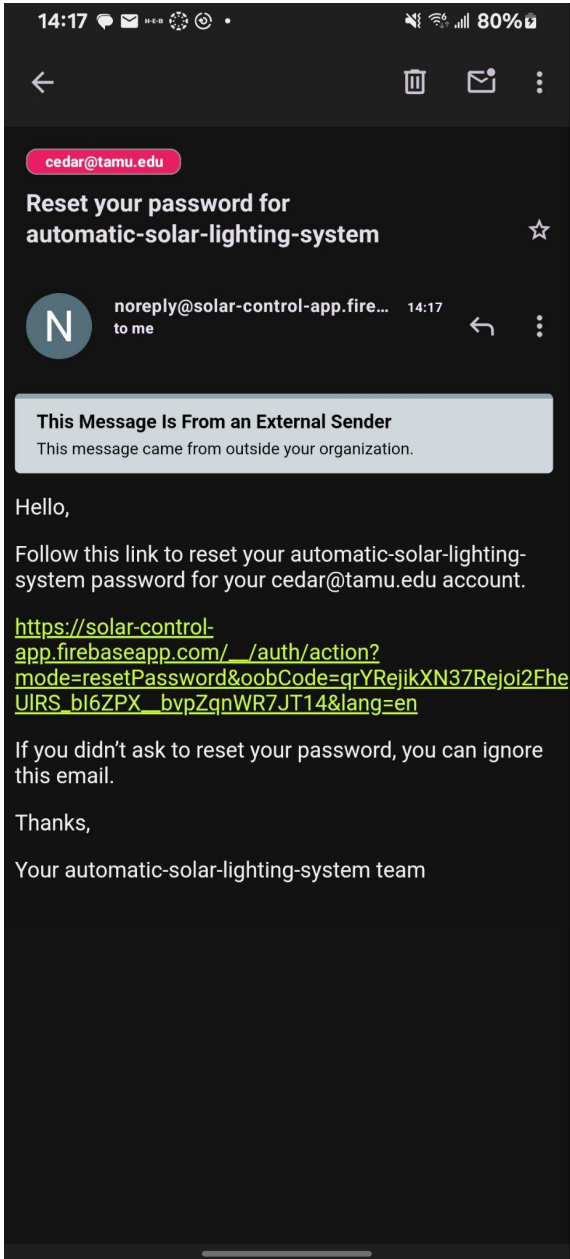


App & Database Con't.

Forgot Password Screen:

- Sends Password reset e-mail
- Checks against Firebase for invalid e-mail addresses





App & Database Con't.

Home Screen:

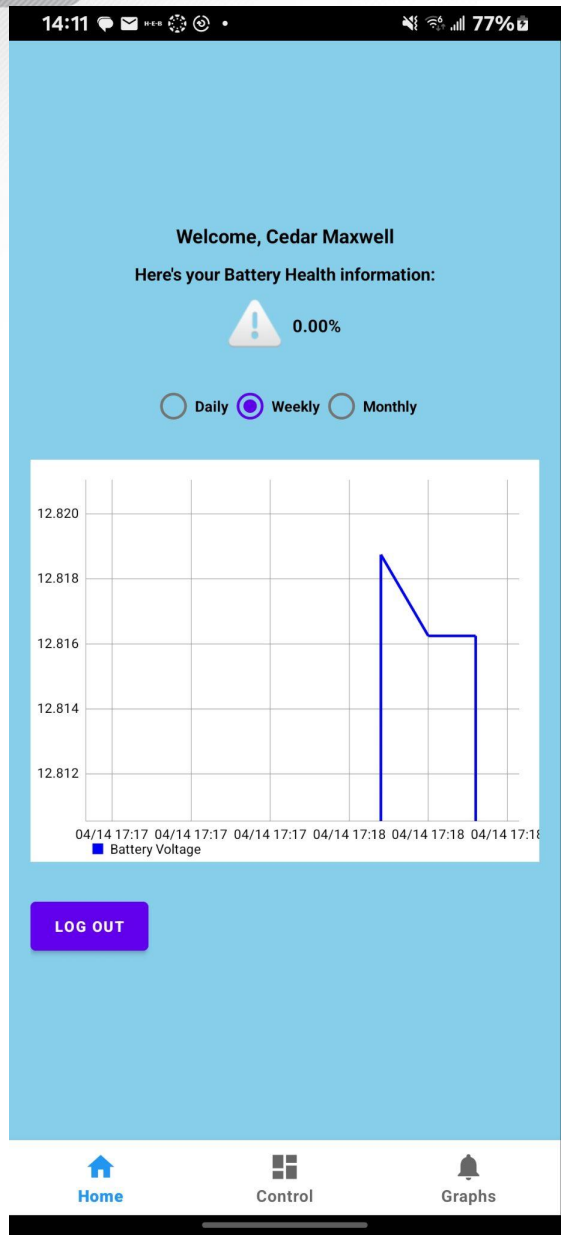
- Shows Daily, Weekly or Monthly graph data
- Shows battery % derived from the graph
- Updates in real time from Firebase

Control Screen:

- Allows user to toggle on/off switches, sent to Firebase and in turn to ESP32
- Responds to changes from Firebase

What Doesn't Work (Yet):

- Scheduling of On/Off



ESP32 console print out of Firebase data

```
I (14111) WIFI: Received Firestore Response: {
  "name": "projects/solar-control-app/databases/(default)/documents/users/SQSdfpXEHzSEW2WU7pkDBzWIA8i1",
  "fields": {
    "label1": {
      "stringValue": "Indoors"
    },
    "switch1": {
      "booleanValue": false
    },
    "firstName": {
      "stringValue": "Cedar"
    },
    "label2": {
      "stringValue": "Outdoors"
    },
    "switch2": {
      "booleanValue": true
    },
    "lastName": {
      "stringValue": "Maxwell"
    }
  },
  "createTime": "2025-02-20T00:18:21.463195Z",
  "updateTime": "2025-04-16T19:15:55.673833Z"
}

I (14161) JSON_PARSE: Parsed Values - switch1: false, switch2: true
I (15501) FIREBASE: Data Sent: {"1744832758": {"batteryVoltage": 13.34}}
I (15501) Realtime Database: Uploaded voltage: 13.34
```

App & Database Conclusion

- Current status:
  - Fully Functional:
    - User account system
    - Bidirectional communication between App and Firebase
    - Bidirectional communication between Firebase and ESP32
    - Graphing from Power Monitor
    - All data segregated by user
  - Still in progress:
    - On/Off scheduling

Power Distribution Engineering Design Accomplishments

Modified Sine Wave Inverter

Problem:

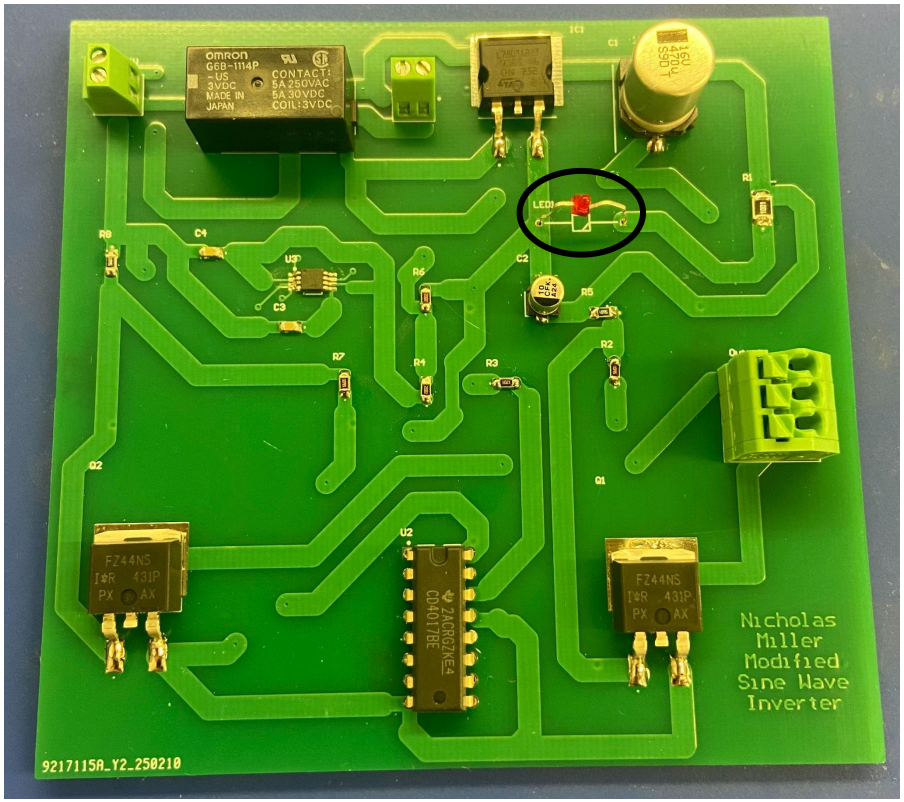
- LED indicator shorted the circuit upon connection.

Solution:

- Removed LED indicator from circuit.

Testing/Validation

- Handles 12V, 5aH battery as power source.
- Tested and validated with 40W, 60W, and 100W LED light bulbs. Each bulb lit when relay was switched, minor pulsing with 100W light bulb.



Power Distribution Engineering Design Accomplishments

Buck Converter

Problems:

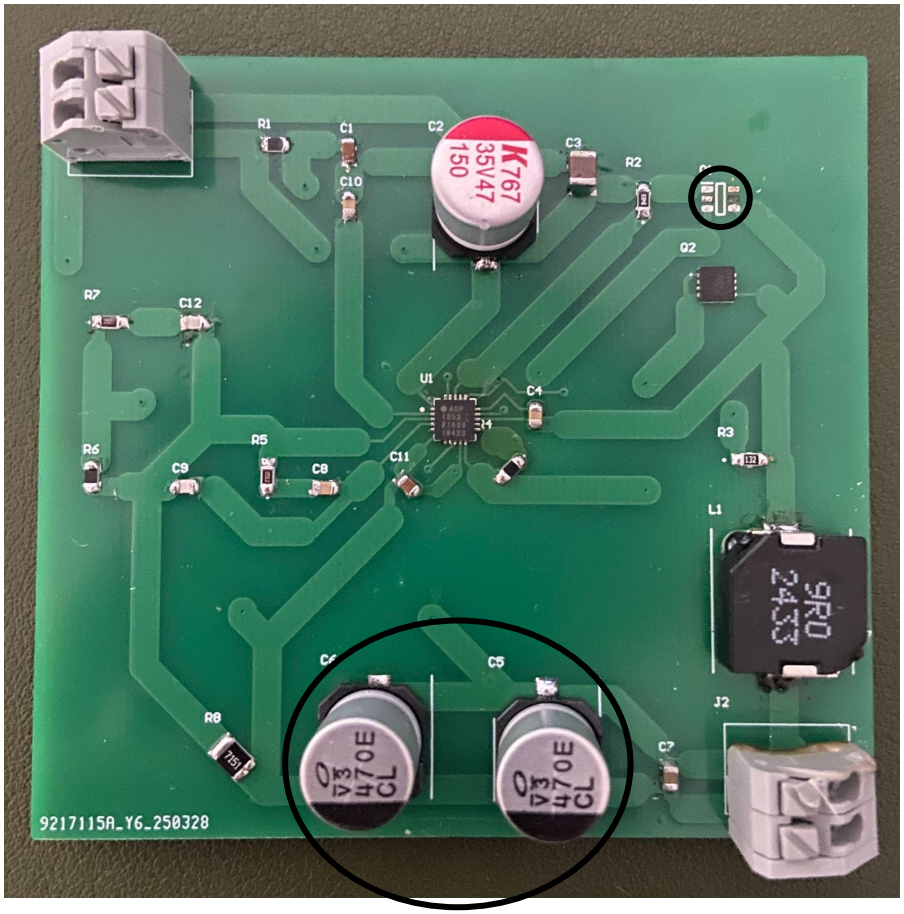
- High-side MOSFET shorted, causing battery voltage to flow directly through output.

Solution:

- Reduced output capacitance.

Testing/Validation:

- Operating range is between 150mA to 450mA, maximum output current at 500mA.

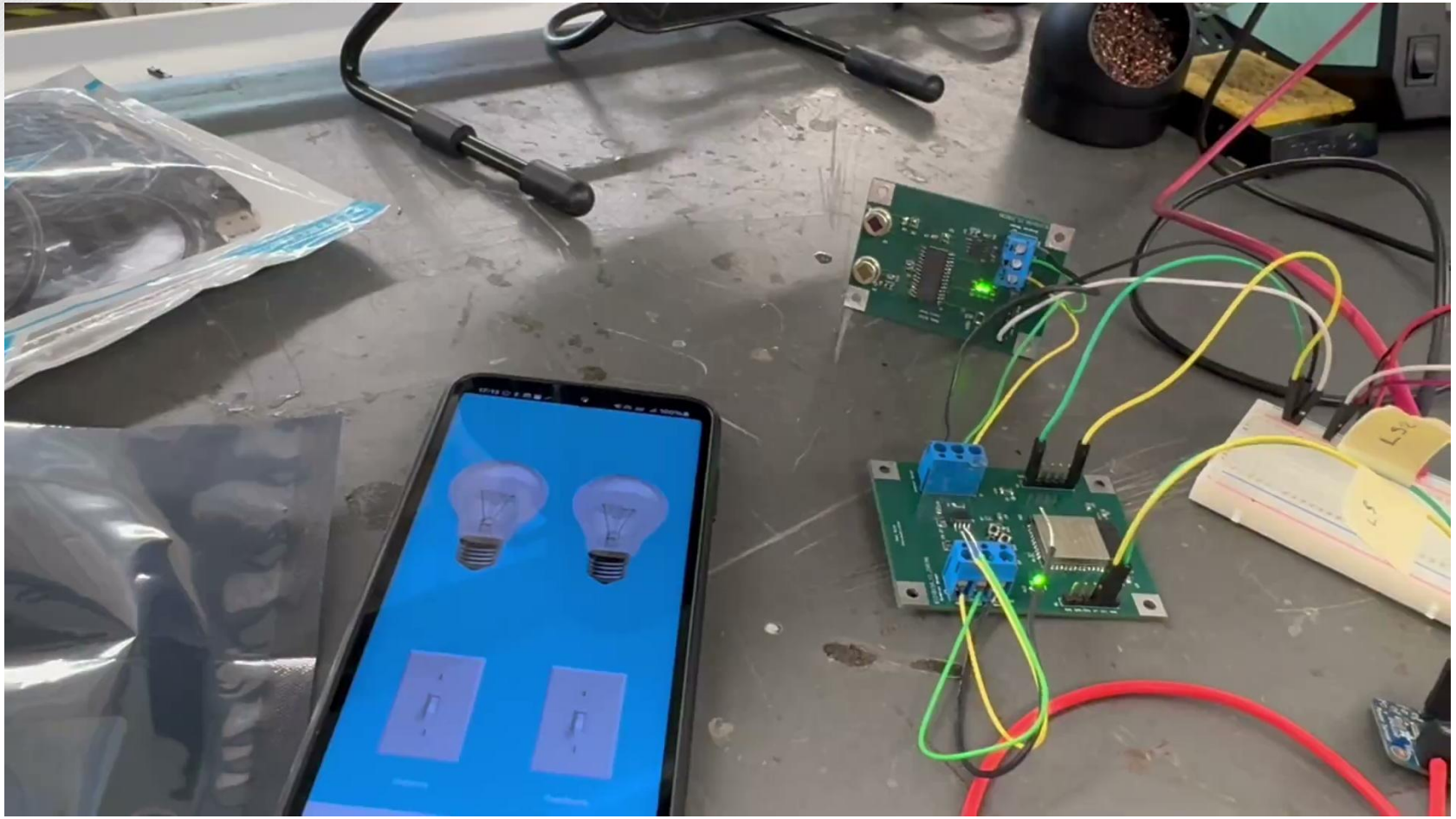


Power Distribution Conclusion

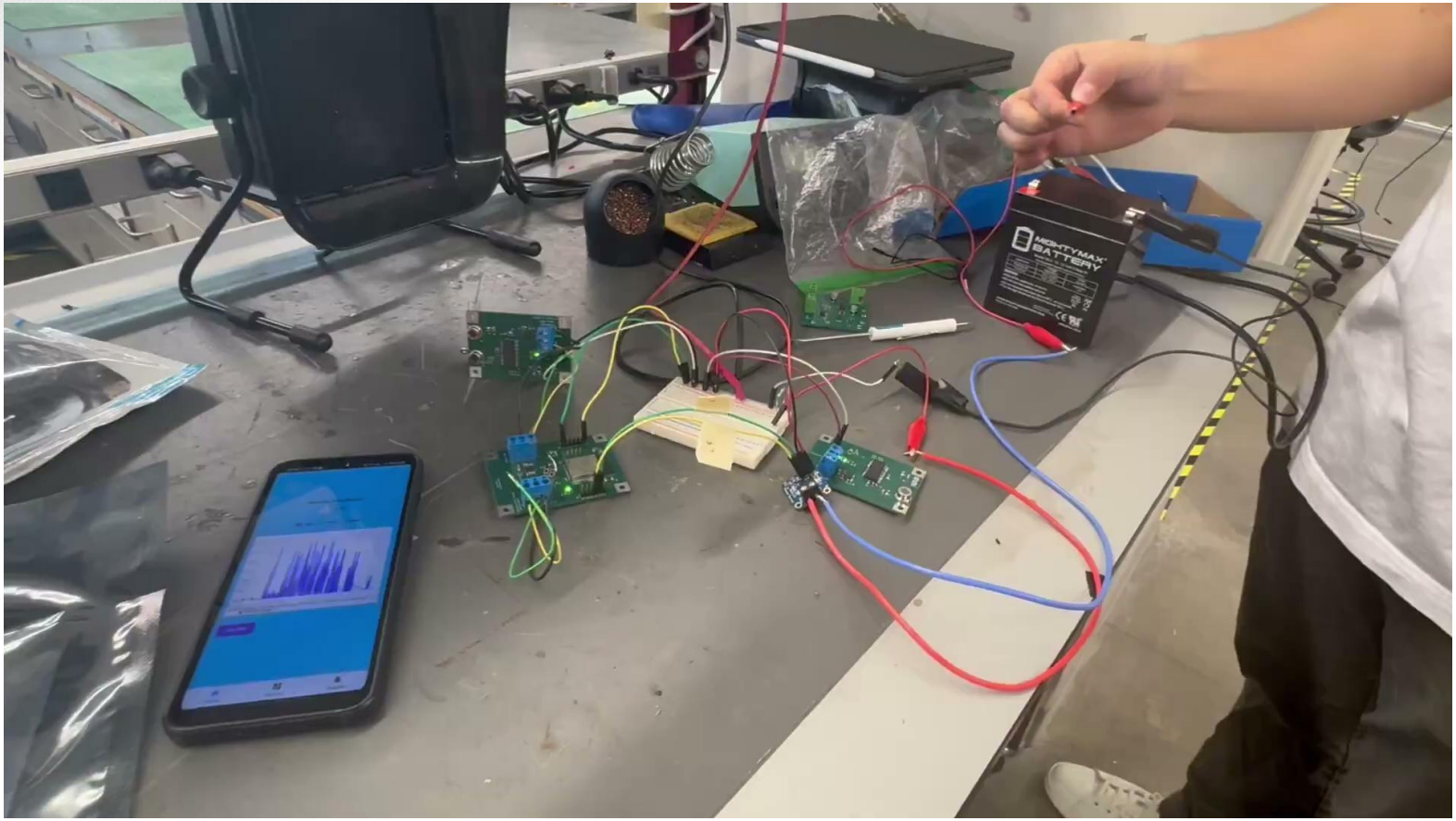
- Changes from 403 documentation:
  - SPI Relay Controller has been added to subsystem.
  - Buck Converter quantity reduced to one Buck Converter.

- Current status:
  - Minor modification to output capacitance to prevent high-side MOSFET from shorting in Buck Converter have been made.
  - Finalizing full integration with MCU & Sensors.
  - Finalizing code for SPI Relay Controller.

Integrated System Results: MCU & Sensors – App/Database Integration



Integrated System Results:
Power monitor – MCU – App/Database Integration



Team design Conclusion

What needs to be done before final demo:

- Integrate power distribution with rest of system
- Final programming:
  - How long light will be on
  - Cooldown time between devices
  - App override of sensor triggers
- Creating of housing structure

Thank you!