Review the assignment due date

FP-F24 IoT Venture Pitch

ESE5180: IoT Wireless, Security, & Scaling

1 GitHub URL

https://github.com/ese5180/fp-f24-iot-venture-pitch-smartbeehive

2 Team Selection

```
Team Name: SmartBee
```

Team Member Name	Email Address
Adithya Rajeev	adithyar@seas.upenn.edu
Tarush Sharma	tarushs@seas.upenn.edu

When will you have your 30-minute weekly team meeting?

Tuesday 3pm - 4pm

IoT-Enabled Beehive Monitoring System

The IoT-enabled beehive monitoring system is designed to provide real-time insights into the health and activity of beehives through a network of sensors and BLE connectivity. From a software standpoint, the system integrates a variety of sensors, including environmental monitors (temperature, humidity) and weight sensors, to gather data continuously.

Data collected from the beehives is transmitted via Bluetooth Low Energy (BLE) to a mobile app powered by the BlueSee platform, which facilitates data visualization and remote monitoring. The software features a user-friendly interface that allows beekeepers to view real-time metrics, receive alerts for unusual hive conditions, and track trends over time.

Additionally, the system implements cloud connectivity to store historical data and perform analytics, enabling proactive hive management and enhancing overall hive productivity.

Users of the IoT-Enabled Beehive Monitoring System

1. **Beekeepers**:
   The primary users of the system are beekeepers who manage beehives for honey production, pollination services, or conservation efforts. They utilize the system to monitor hive conditions, track bee activity, and receive alerts about any potential issues, such as temperature fluctuations or abnormal weight changes, which can indicate problems in the hive.

2. **Researchers**:
   Researchers studying bee behavior, health, and environmental impact can benefit from the data collected by the system. They can analyze trends over time, conduct experiments, and gain insights into the factors affecting bee populations, thereby contributing to conservation efforts and sustainable beekeeping practices.

3. **Hobbyists and Enthusiasts**:
   Individuals interested in beekeeping as a hobby can use the system to enhance their understanding of hive management. The data insights can help them make informed decisions to promote the health and productivity of their beehives.

This diverse user base highlights the versatility and potential impact of the IoT-enabled beehive monitoring system across various fields and interests.

3 Reading & Research

Nothing to submit for this section!

4 Concept Development

4.1 Target Market & Demographics

```
Who will be using your product?
```

- **Beekeepers of all levels**: Hobbyists, sideliners, and commercial beekeepers will all benefit from the real-time data and insights provided by the SmartBee.
- **Specifically**: Those interested in improving hive health, increasing honey production, and reducing the need for manual inspections. Those with limited time or who manage hives in remote locations will find the remote monitoring capabilities especially valuable.

```
Who will be purchasing your product?
```

- **Beekeepers directly**: This will likely be your primary customer base.
- **Beekeeping associations and clubs**: Group purchases or partnerships with these organizations could be a good way to reach more beekeepers.
- **Educational institutions and research organizations**: Universities and research institutions studying bees or beekeeping practices might be interested in using the system for research or teaching purposes.

```
Where in the world (or space!) would you deploy your product?
```

- **Anywhere with beekeeping activity**: The system can be deployed anywhere in the world where bees are kept, from backyard hives to large commercial apiaries.

- **Developing countries**: The system could be particularly valuable in developing countries where beekeeping is an important source of income and food security, but resources and expertise may be limited.

```
How large is the market you’re targeting, in US dollars?
```

- **Global honey market**: The global honey market was valued at approximately $9.21 billion in 2021 and is expected to grow to $12.35 billion by 2028. (Source: Fortune Business Insights)
- **Beekeeping equipment market**: The global beekeeping equipment market size was valued at $2.27 billion in 2021 and is expected to expand at a compound annual growth rate (CAGR) of 5.2% from 2022 to 2030. (Source: Grand View Research)
- **Precision beekeeping market**: While a relatively new segment, the precision beekeeping market (which includes hive monitoring systems) is expected to grow significantly in the coming years, driven by increasing adoption of technology in agriculture.

```
How much of that market do you expect to capture, in US dollars?
```

Since the majority beekeepers are Hobbyists, we plan on starting with hobbyists with less than 10 hives.

Target market: 80,000 hobbyist beekeepers in the US with under 10 hives.

Market penetration:

- Year 1: 1% (800 units sold)
- Year 2: 3% (2,400 units sold)
- Year 3: 5% (4,000 units sold)

System price: $250 per unit

Projected revenue:

- Year 1: $200,000
- Year 2: $600,000
- Year 3: $1,000,000

Assumptions:

- **Pricing**: The $250 price point is competitive and attractive to hobbyist beekeepers.
- **Production and distribution**: We have the capacity to manufacture and distribute the required number of units to meet demand.

```
What competitors are already in the space?
```

- **BeeHero**: Offers a platform that combines sensor data with AI to provide insights and recommendations for beekeepers.

- **ApisProtect**: Provides hive monitoring sensors and software focused on early problem detection.
- **SolutionBee**: Offers a hive monitoring system with a focus on Varroa mite detection and management.
- **OSBeehives**: Open-source platform for building DIY hive monitoring systems.

4.2 Security Requirements Specification

Overview

The Smart Hive Monitoring System will incorporate security measures to protect user data, maintain system integrity, and prevent unauthorized access. These measures will address potential threats such as data breaches, unauthorized access to hive controls, and malicious attacks on the system.

Definitions, Abbreviations

- **MCU**: Microcontroller Unit
- **AES**: Advanced Encryption Standard
- **API**: Application Programming Interface
- **HTTPS**: Hypertext Transfer Protocol Secure

Functionality

SEC 03 Messages from tampered systems shall be ignored

- Implement firmware integrity checks to detect any unauthorized modifications to the MCU.
- Use digital signatures to verify the authenticity of messages from the hive.

SEC 04 All key information shall be stored in a secure hardware region

- Store sensitive data (e.g., encryption keys, user credentials) in a secure hardware enclave within the MCU.
- Utilize secure boot mechanisms to prevent unauthorized code execution during startup.

Additional Security Considerations

- Firmware updates: Provide secure and reliable firmware updates to patch security vulnerabilities and improve system functionality.
- User access control: Implement role-based access control to restrict user access to specific functions and data within the system.
- Data backup and recovery: Implement secure data backup and recovery mechanisms to protect against data loss.

4.3 Hardware Requirements Specification

Overview

The peripheral devices utilize a low-power microcontroller, sensors, and a BLE transceiver to collect and transmit hive data. The central device (laptop or smartphone) provides a user interface for data visualization and analysis.

Definitions, Abbreviations

- **BLE**: Bluetooth Low Energy
- **ADC**: Analog-to-Digital Converter
- **IR**: Infrared
- **µA**: Microamperes
- **mA**: Milliamperes

Functionality

- **HRS 01 – Peripheral devices shall be based on a Nordic Semiconductor microcontroller with BLE capabilities.**
  - The microcontroller shall have a current consumption of no more than 30 mA during data acquisition and transmission.
- **HRS 02 – A temperature and humidity sensor shall be integrated into each peripheral device.**
  - The sensor shall have an accuracy of ±0.5 °C for temperature and ±2% relative humidity.
  - The sensor shall have a resolution of at least 0.1 °C for temperature and 0.1% for relative humidity.
  - The sensor shall have a current consumption of no more than 5 µA in normal operation mode.
- **HRS 03 – Each peripheral device shall include an ADC and load cell for weight measurement.**
  - The ADC shall have a resolution of at least 16 bits.
  - The load cell shall have an accuracy of ±0.5% of the measured weight.
  - The combined current consumption of the ADC and load cell shall not exceed 1 mA in normal operation mode.
  - **HRS 03a - A battery shall be included to power the load cell.**
    - The battery shall have a capacity sufficient to power the load cell for at least one year of continuous operation.
    - The battery shall be rechargeable or replaceable.
- **HRS 04 – An IR sensor shall be used to track bee movement at the hive entrance.**
  - The sensor shall be capable of detecting bees passing through a defined area with a minimum accuracy of 90%.
  - The sensor shall have a current consumption of no more than 50 mA during active sensing.
- **HRS 05 – The central device shall be a laptop or smartphone with BLE connectivity.**

4.4 Software Requirements Specification

Overview

The Smart Hive Monitoring System software manages data acquisition from various sensors, processes the data, and transmits it wirelessly to the beekeeper's device. The software also provides a user interface for data visualization, analysis, and alert management.

Users

- **Beekeepers**: Beekeepers of all levels (hobbyist, sideliner, commercial) will use the software to monitor hive conditions, track trends, receive alerts, and make informed decisions about hive management. For this prototype, we will be focusing on hobbyists.
- **System administrators**: Administrators will have access to manage user accounts, system settings, and potentially perform data analysis and reporting at a higher lev.

Functionality

- **SRS 01** – The environmental sensors (temperature, humidity) will be measured with 12-bit depth every 60 seconds +/-5 seconds.

- **SRS 02** – The weight sensor will report the hive weight with a resolution of 0.1 kg every 5 minutes +/-10 seconds.

- **SRS 03** – The system will transmit collected data via Bluetooth Low Energy (BLE) to the mobile app every 30 seconds +/-3 seconds.

- **SRS 04** – The mobile app will update the user interface to display real-time metrics, including temperature, humidity, and hive weight.

- **SRS 05** – The app will send notifications to the user for abnormal conditions, such as temperature exceeding 35°C or weight loss greater than 2 kg within 24 hours.

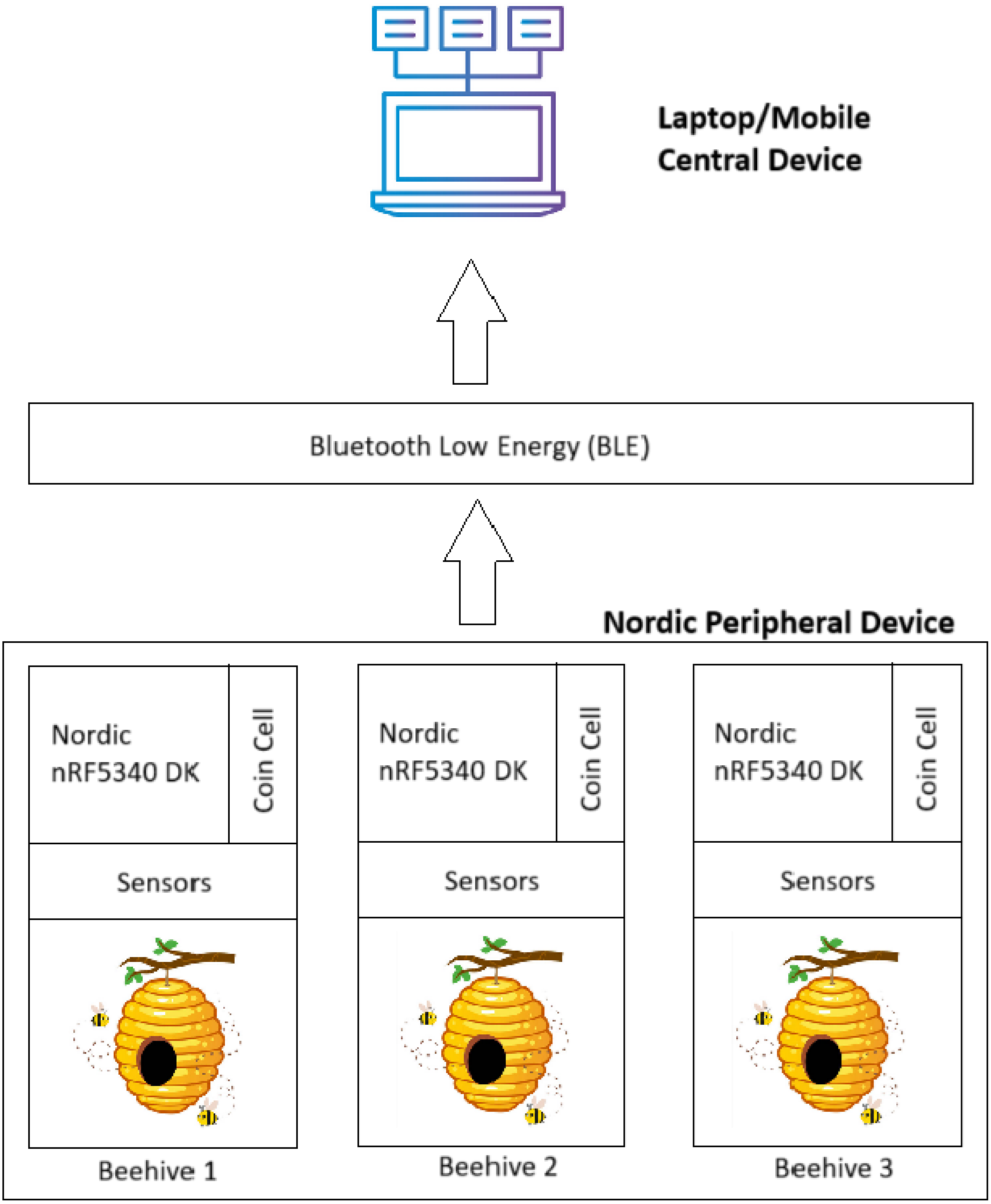
- **SRS 06** – The system will log historical data to the cloud for analysis, storing at least 30 days of data for each metric.

- **SRS 07** – The app will provide visualizations of historical data trends for temperature, humidity, and hive weight over time.

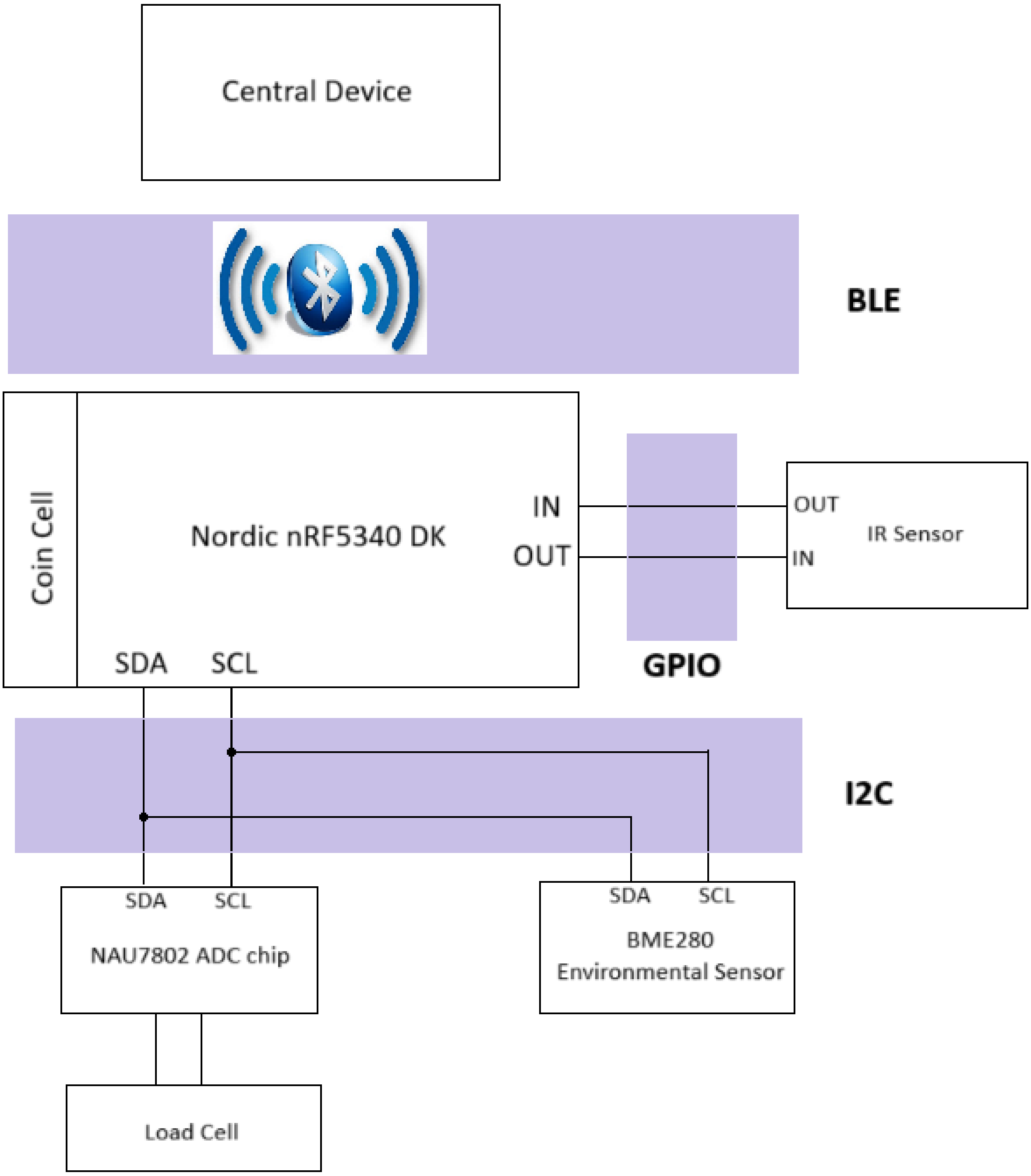
- **SRS 08** – The system will include user authentication to ensure secure access to data and app features.

4.5 System Level Diagrams

1. Device block diagram.



2. Communication diagram.



4.6 Budgeting

1. Create a rough power budget for your end device.

Component	Operating Voltage Range	Current Consumption
Nordic nRF5340 DK	1.7V - 3.6V	Up to 22 mA (TX mode)
BME280	1.8V - 3.6V	~2.8 µA (normal mode)
NAU7802	2.7V - 5.5V	~500 µA (normal mode)
Adafruit IR Remote Transceiver (IR RX/TX)	3.3V - 5V	~1 mA (receiving), up to 60 mA (transmitting)

Component	Operating Voltage Range	Current Consumption
Adafruit Load Cell - 5kg	2.7V - 5.5V	~1mA (while measuring, negligible when idle)

Total Estimated Current Consumption

Maximum Current (assuming all components active and transmitting):

Component	Current Consumption (mA)
Nordic nRF5340 DK (TX mode)	22
BME280 (normal mode)	0.0028
NAU7802 (normal mode)	0.5
Adafruit Load Cell - 5kg	1
IR Transceiver (TX mode)	60
Total:	**≈ 83.5 mA**

Battery Selection for Hardware Power Requirements

Based on the estimated power consumption of the system, we require a power source capable of reliably handling peak currents while also providing sufficient energy capacity for extended operation. Given the peak current draw of approximately 82.5 mA, a lithium-ion battery was determined to be the optimal choice due to its high energy density, stable voltage output, and ability to deliver higher currents compared to coin cell batteries.

Battery Specifications

1. **Current Rating**:
   The system's maximum current demand is 82.5 mA, primarily due to components such as the Nordic nRF5340 DK (in transmit mode) and the Adafruit IR Transceiver when actively transmitting. Therefore, a lithium-ion battery with a continuous discharge rating of at least **100 mA** is recommended. This rating ensures a safe margin above peak usage, accommodating any minor variations in current draw and preventing potential voltage drops or thermal issues.

2. **Capacity**:
   For sufficient operational time between charges, a battery with a capacity of **500-1000 mAh** is selected. This capacity range provides the following estimated runtimes based on the continuous peak current draw:

   - **500 mAh**: Approximate runtime of 6 hours (82.5 mA draw).
   - **1000 mAh**: Approximate runtime of 12 hours (82.5 mA draw).

3. **Voltage**:
   The selected lithium-ion battery should have a nominal voltage of **3.7V**, aligning well with the operating voltage requirements of all components in the system.

Recommended Battery

For this setup, a 3.7V, 1000 mAh lithium-ion battery with a 100 mA discharge rating is recommended. This battery provides ample capacity for extended operation while meeting the peak current needs of the system, thereby ensuring stable performance throughout use.

2. Estimate the hardware costs in your product.

Component	Quantity	Unit Price (USD)	Total Cost (USD)
Nordic nRF5340 DK	1	$49.00	$49.00
BME280	1	$3.95	$3.95
NAU7802	1	$8.50	$8.50
Adafruit IR Remote Transceiver (IR RX/TX)	1	$5.95	$5.95
Lithium-Ion Battery (3.7V, 1000 mAh)	1	$12.00	$12.00
Adafruit Load Cell - 5kg	1	$3.95	$3.95
Total Estimated Cost			**$83.35**

3. Estimate the software costs in your product.

Cost Estimate for BLE Data Display Using BlueSee

To display data from the system via BLE, the **BlueSee** platform was selected for its ease of use, customizable dashboards, and reliable data communication over Bluetooth Low Energy (BLE). BlueSee provides a subscription-based service that supports data visualization and device management, ideal for showcasing sensor data and system metrics.

BlueSee Platform Subscription Cost

1. **Subscription Type**: Monthly
   BlueSee offers a **monthly subscription** plan with full access to its BLE connectivity and data visualization features, ideal for development and demonstration purposes.

2. **Cost per Month**: $15 (estimated)
   The monthly fee for BlueSee is estimated at **$15 per month**. This includes access to essential features such as customizable dashboards, data logging, and BLE data connectivity.

3. **Annual Estimate**:
   For extended use over a 12-month period, the estimated cost for BlueSee would be:
   - **12 months x $15 = $180**

Total Cost Estimate for BlueSee

Platform	Subscription Duration	Monthly Cost (USD)	Total Cost (USD)
BlueSee	12 months	$15	$180

Using BlueSee as the BLE platform for data display is expected to cost approximately **$180 per year**. This approach offers a flexible and cost-effective solution to showcase live data from the system without the need for a custom-built application, providing both a streamlined interface and high ease of integration for BLE data visualization.

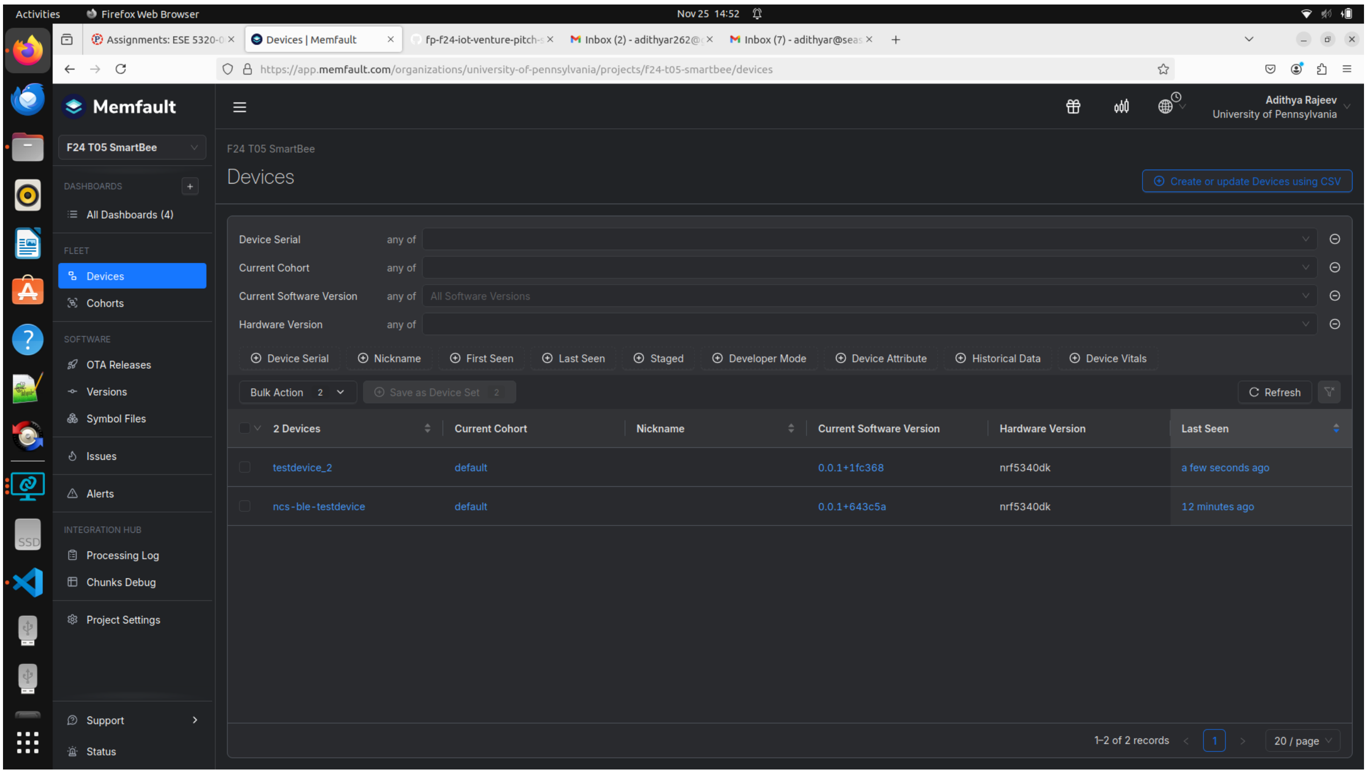
5 In-Class Pitch

Presented. Slide link - SmartBee Presentation

Fleet Management

Images/Videos can be found in the Fleet_Management_Media folder.
It can also be found at - link

Multiple Devices



https://app.memfault.com/organizations/university-of-pennsylvania/projects/f24-t05-smartbee/devices

F24 T05 SmartBee

Devices

Device Serial any of

Current Cohort any of

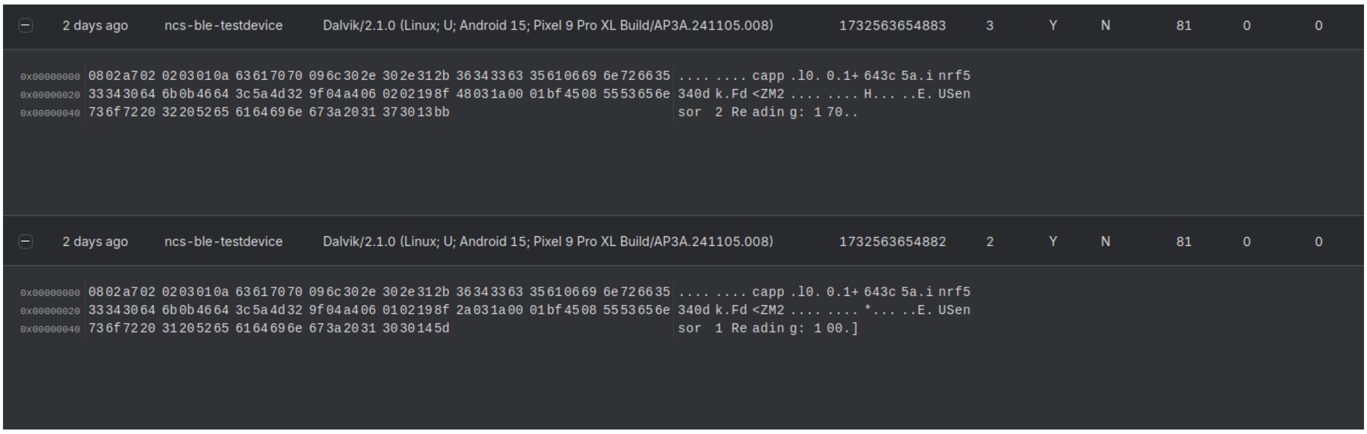
Current Software Version any of All Software Versions

Hardware Version any of

2 Devices	Current Cohort	Nickname	Current Software Version	Hardware Version	Last Seen
testdevice_2	default		0.0.1+1fc368	nrf5340dk	a few seconds ago
ncs-ble-testdevice	default		0.0.1+643c5a	nrf5340dk	12 minutes ago

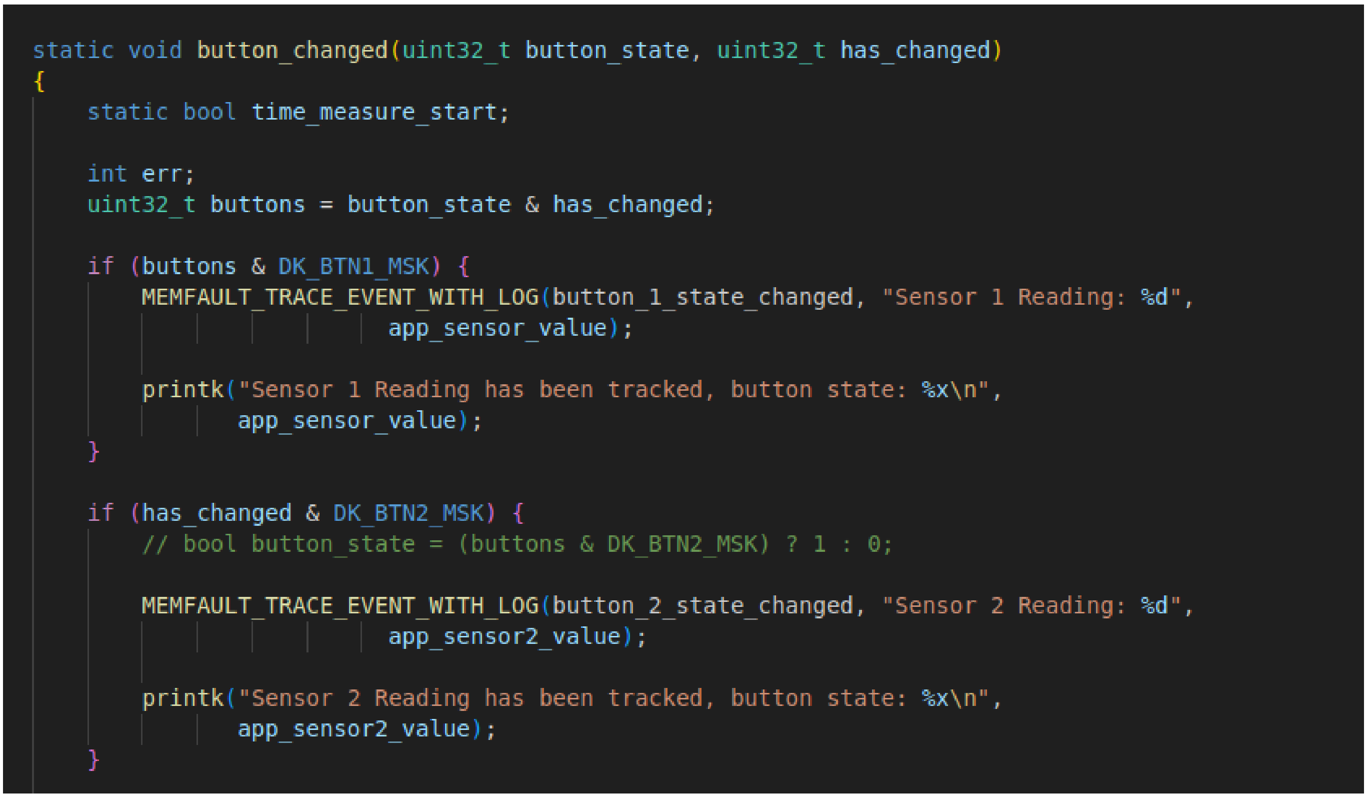
Device Data Feature

Memfault Chunk



```
2 days ago    ncs-ble-testdevice    Dalvik/2.1.0 (Linux; U; Android 15; Pixel 9 Pro XL Build/AP3A.241105.008)    1732563654883    3    Y    N    81    0    0

0x00000000 0802a702 0203010a 63617070 096c302e 302e312b 36343363 35610669 6e726635 .... .... capp .l0. 0.1+ 643c 5a.i nrf5
0x00000020 33343064 6b0b4664 3c5a4d32 9f04a406 0202198f 48031a00 01bf4508 5553656e 340d k.Fd <ZM2 .... .... H... ..E. USen
0x00000040 736f7220 32205265 6164696e 673a2031 373013bb                            sor  2 Re adin g: 1 70..

2 days ago    ncs-ble-testdevice    Dalvik/2.1.0 (Linux; U; Android 15; Pixel 9 Pro XL Build/AP3A.241105.008)    1732563654882    2    Y    N    81    0    0

0x00000000 0802a702 0203010a 63617070 096c302e 302e312b 36343363 35610669 6e726635 .... .... capp .l0. 0.1+ 643c 5a.i nrf5
0x00000020 33343064 6b0b4664 3c5a4d32 9f04a406 0102198f 2a031a00 01bf4508 5553656e 340d k.Fd <ZM2 .... .... *... ..E. USen
0x00000040 736f7220 31205265 6164696e 673a2031 3030145d                            sor  1 Re adin g: 1 00.]
```

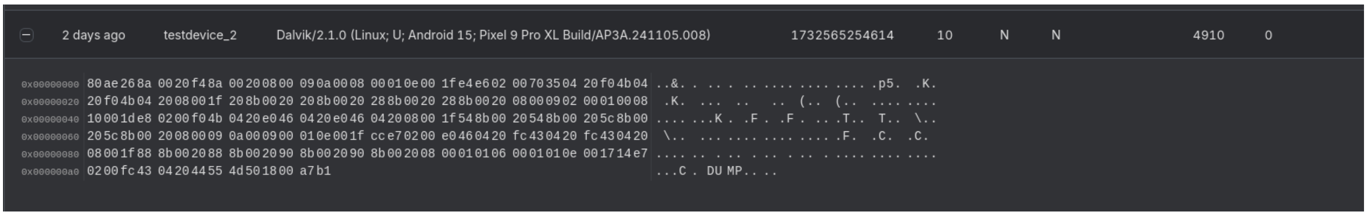
Code



```c
static void button_changed(uint32_t button_state, uint32_t has_changed)
{
    static bool time_measure_start;

    int err;
    uint32_t buttons = button_state & has_changed;

    if (buttons & DK_BTN1_MSK) {
        MEMFAULT_TRACE_EVENT_WITH_LOG(button_1_state_changed, "Sensor 1 Reading: %d",
                                      app_sensor_value);

        printk("Sensor 1 Reading has been tracked, button state: %x\n",
               app_sensor_value);
    }

    if (has_changed & DK_BTN2_MSK) {
        // bool button_state = (buttons & DK_BTN2_MSK) ? 1 : 0;

        MEMFAULT_TRACE_EVENT_WITH_LOG(button_2_state_changed, "Sensor 2 Reading: %d",
                                      app_sensor2_value);

        printk("Sensor 2 Reading has been tracked, button state: %x\n",
               app_sensor2_value);
    }
```

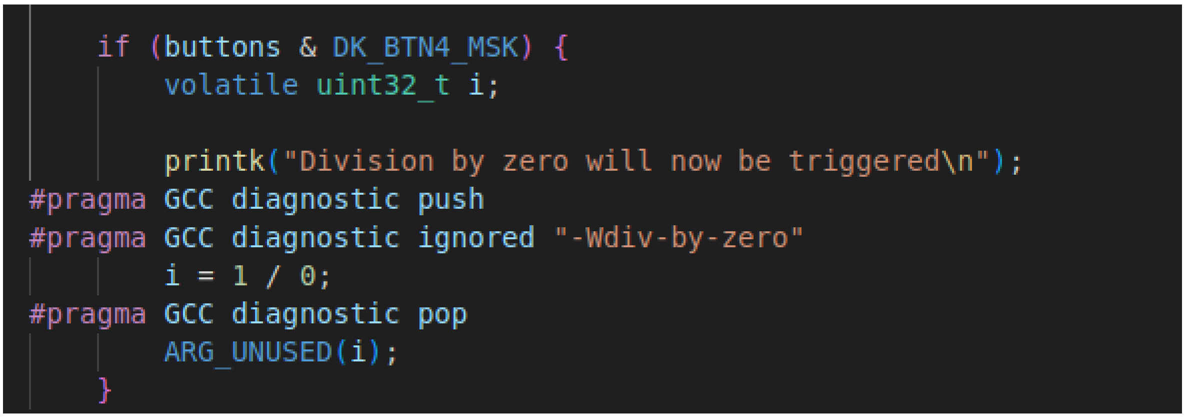
Core Dump Feature

Memfault Chunk



```
2 days ago    testdevice_2    Dalvik/2.1.0 (Linux; U; Android 15; Pixel 9 Pro XL Build/AP3A.241105.008)    1732565254614    10    N    N    4910    0

0x00000000 80ae268a 0020f48a 00200800 090a0008 00010e00 1fe4e602 00703504 20f04b04 ..&. . ... . .. .... .... ..... .p5.  .K.
0x00000020 20f04b04 2008001f 208b0020 208b0020 288b0020 288b0020 08000902 00010008  .K.  ...  ..   ..  (..  (..  .... ....
0x00000040 10001de8 0200f04b 0420e046 0420e046 04200800 1f548b00 20548b00 205c8b00 .... ...K . .F . .F . ... .T..  T..  \..
0x00000060 205c8b00 20080009 0a000900 010e001f cce70200 e0460420 fc430420 fc430420  \..  ... .... .... ..... .F.  .C.  .C.
0x00000080 08001f88 8b002088 8b002090 8b002090 8b002008 00010106 0001010e 001714e7 .... .. . .. . .. . .. . .... .... ....
0x000000a0 0200fc43 04204455 4d501800 a7b1                                          ...C . DU MP.. ..
```

Code



```
    if (buttons & DK_BTN4_MSK) {
        volatile uint32_t i;

        printk("Division by zero will now be triggered\n");
#pragma GCC diagnostic push
#pragma GCC diagnostic ignored "-Wdiv-by-zero"
        i = 1 / 0;
#pragma GCC diagnostic pop
        ARG_UNUSED(i);
    }
```

Video Demo

Demo Video

Final Report (Website)

1. Changes Throughout the Project

Target Market & Demographics

Initially, the target market was broadly defined to include hobbyists, sideline beekeepers, and commercial beekeepers. Over time, the focus narrowed to primarily hobbyist beekeepers with fewer than 10 hives. This shift was likely driven by the realization that hobbyists represent a more accessible entry point for market penetration, given their smaller-scale operations and interest in affordable, user-friendly solutions.

Security, Hardware, & Software Requirements

The project evolved to include more robust security measures such as firmware integrity checks, secure storage of sensitive data in hardware enclaves, and secure boot mechanisms. Hardware requirements were refined to ensure low power consumption and compatibility with BLE connectivity. Software requirements expanded to include real-time data visualization and cloud storage for historical data analysis.

Product Function & Components

The product's core functionality remained consistent—monitoring hive conditions using sensors for temperature, humidity, weight, and bee movement. However, refinements were made to improve sensor accuracy and reduce power consumption. The inclusion of a mobile app for BLE data display became a central feature.

Power & Cost Budgeting

The power budget was optimized by selecting low-power components like the Nordic nRF5340 DK microcontroller and BME280 sensor. The estimated hardware cost per unit was finalized at $83.35, while software costs (e.g., BlueSee subscription) were estimated at $15/month or $180/year.

2. Successful Aspects of the Project

- **Market Alignment:** Successfully identified a specific target demographic (hobbyist beekeepers), aligning the product with their needs.
- **Technical Implementation:** Achieved integration of multiple sensors with BLE connectivity for real-time monitoring.
- **Cost Efficiency:** Maintained a competitive hardware cost of $83.35 per unit while delivering essential features.
- **User-Friendly Design:** Developed an intuitive mobile app interface for data visualization and alerts.

These successes were driven by clear problem identification (beehive monitoring challenges) and effective use of available resources.

3. Challenges Faced

- **Time Constraints:** Limited time may have restricted deeper testing or refinement of certain components.
- **Wireless Protocol Limitations:** BLE was chosen for its low power consumption but may have limited range compared to alternatives like Wi-Fi or LoRaWAN.
- **Market Validation:** Limited feedback from actual target users (hobbyist beekeepers) may have hindered validation of product-market fit.

These issues arose due to the finite resources and scope of the project.

4. Improved Development Approach

- **Early Focus on Specifics:** Define the target audience, hardware components, and functionalities in more detail early in the project. This would help to streamline the development process and avoid unnecessary changes later on.
- **Prototyping and Testing:** Allocate more time for prototyping and testing, especially for critical aspects like BLE range and sensor accuracy. This would help to identify and address potential issues early on.
- **Iterative Development:** Adopt an iterative development approach, starting with a minimum viable product (MVP) and gradually adding features based on user feedback and testing.

5. System Design Considerations

```
a. Wireless Communication Protocol
```

BLE is a suitable choice for hobbyist beekeepers due to its low power consumption, ease of use, and cost-effectiveness. However, range limitations need to be addressed through testing and potential mitigation strategies. While BLE was effective for low power consumption, exploring options like LoRaWAN could improve range and scalability for remote hive locations.

```
b. Sensors or Actuators
```

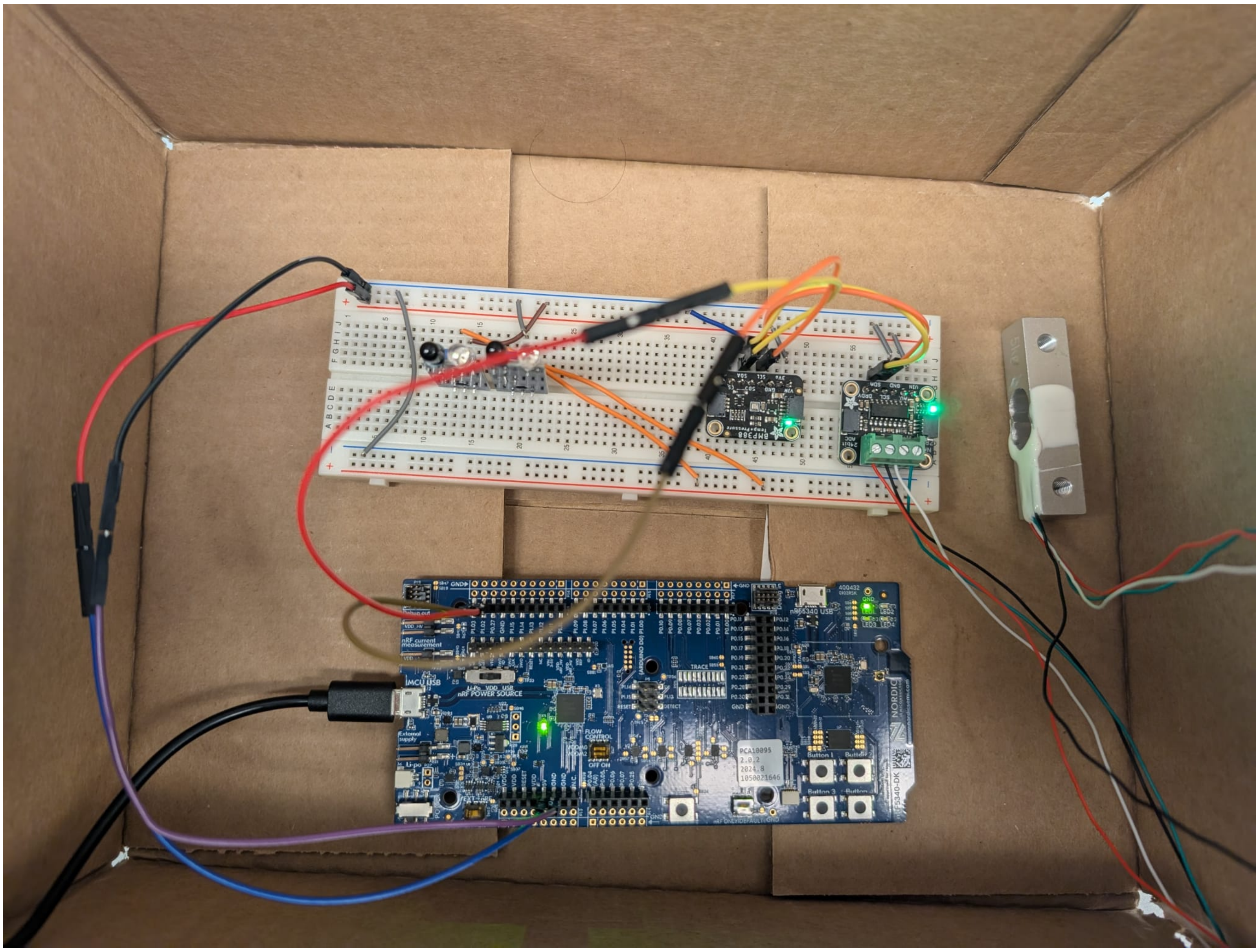
The selected sensors (temperature, humidity, weight, bee counter) are relevant for hobbyist beekeepers. Additional sensors (e.g., acoustic sensors) could be considered for future iterations based on user needs and feedback.

```
c. Target Market Feedback
```

Focusing on hobbyist beekeepers was a good decision, but further market research and user feedback would help to validate the product's features and ensure they meet the specific needs of this target audience.

Device Images

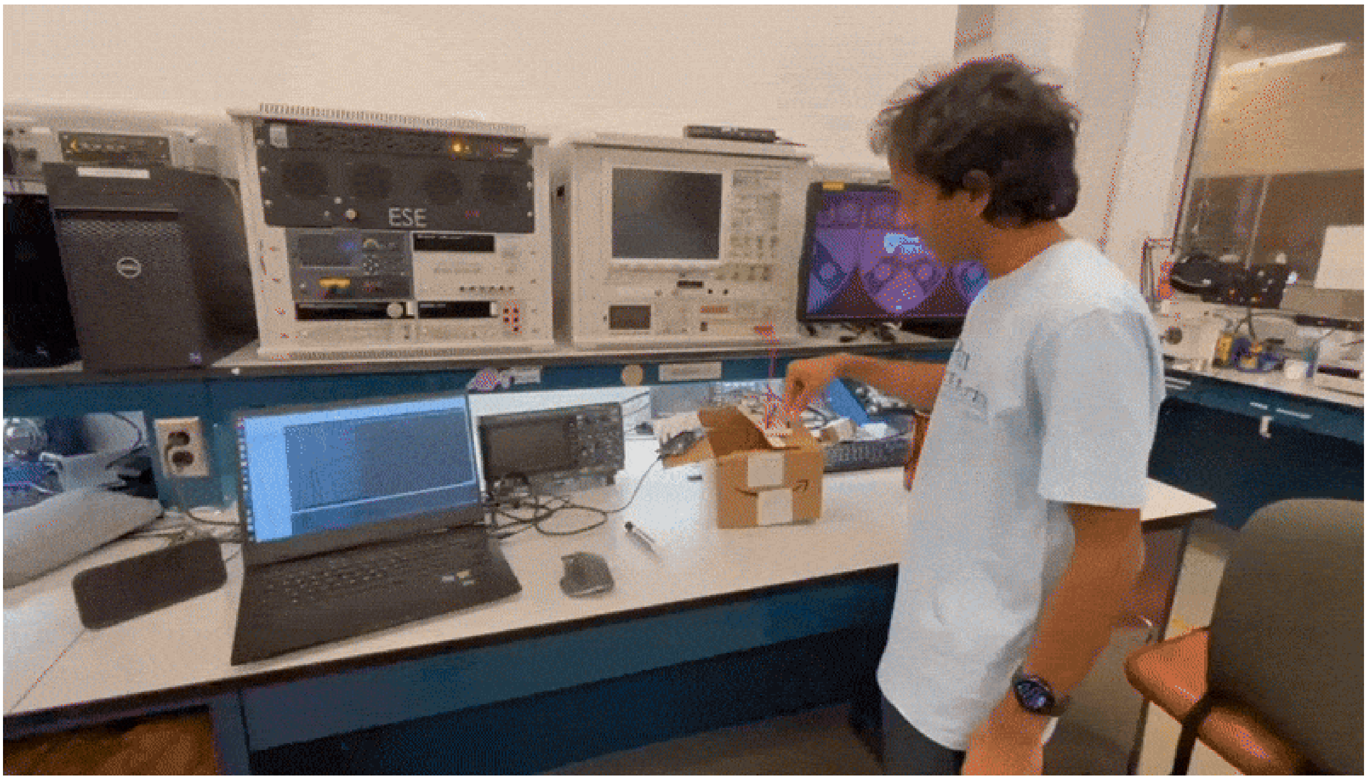
MVP Device



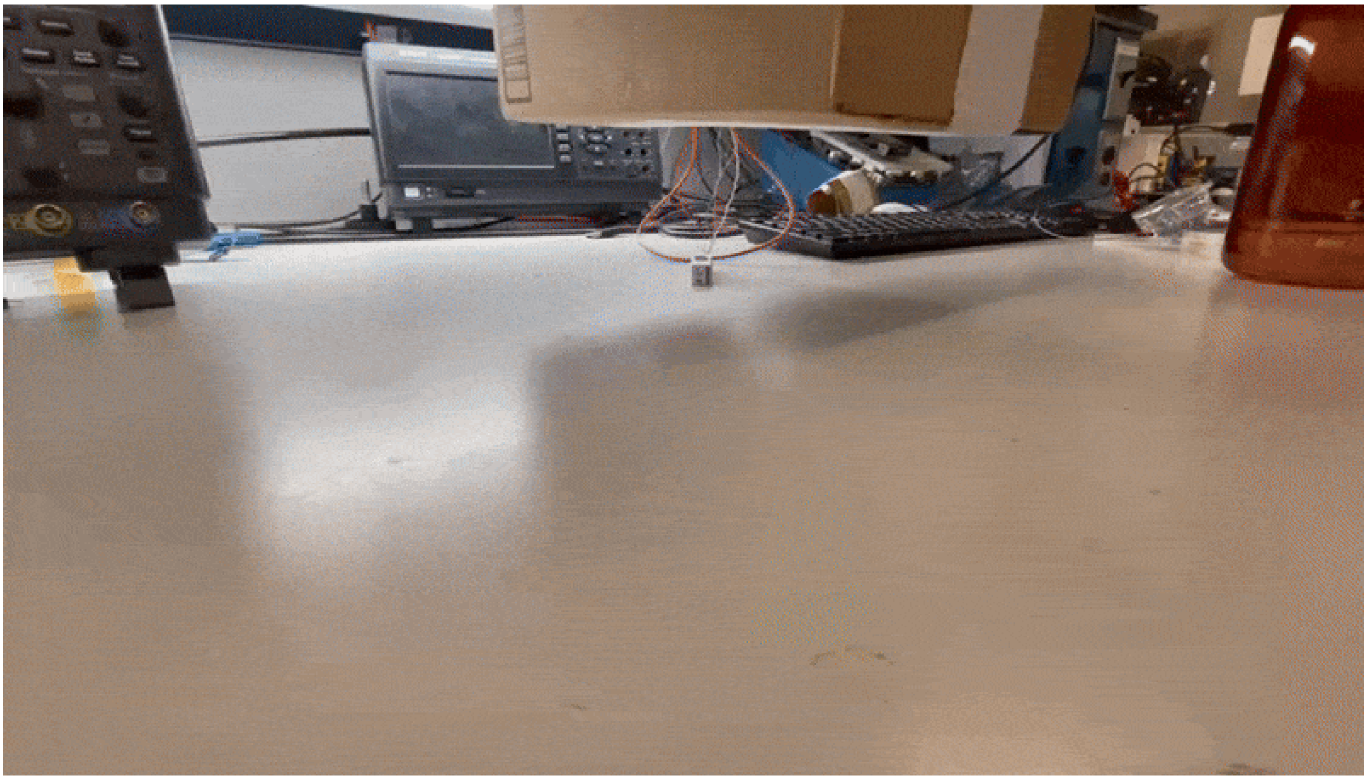
Video demo

```
- Your Core Product Function
```

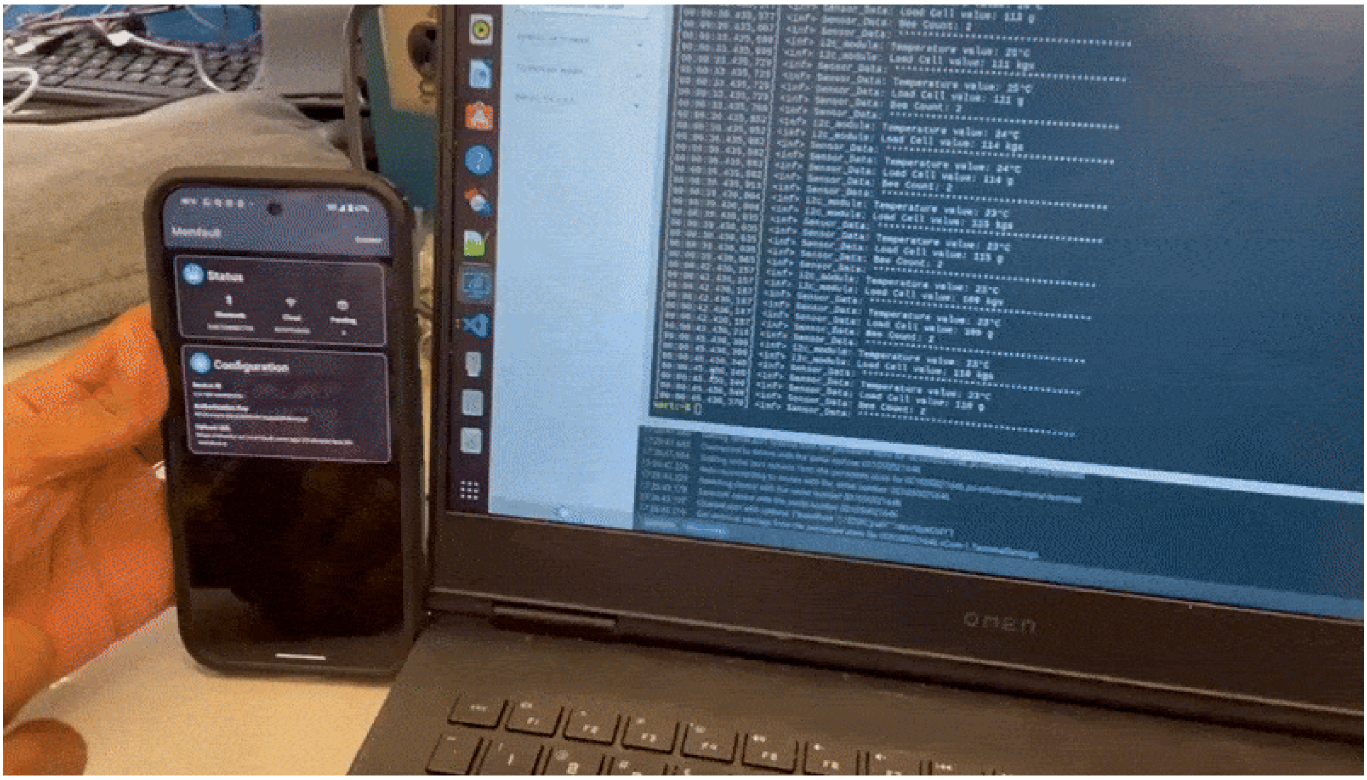
Sensor Data



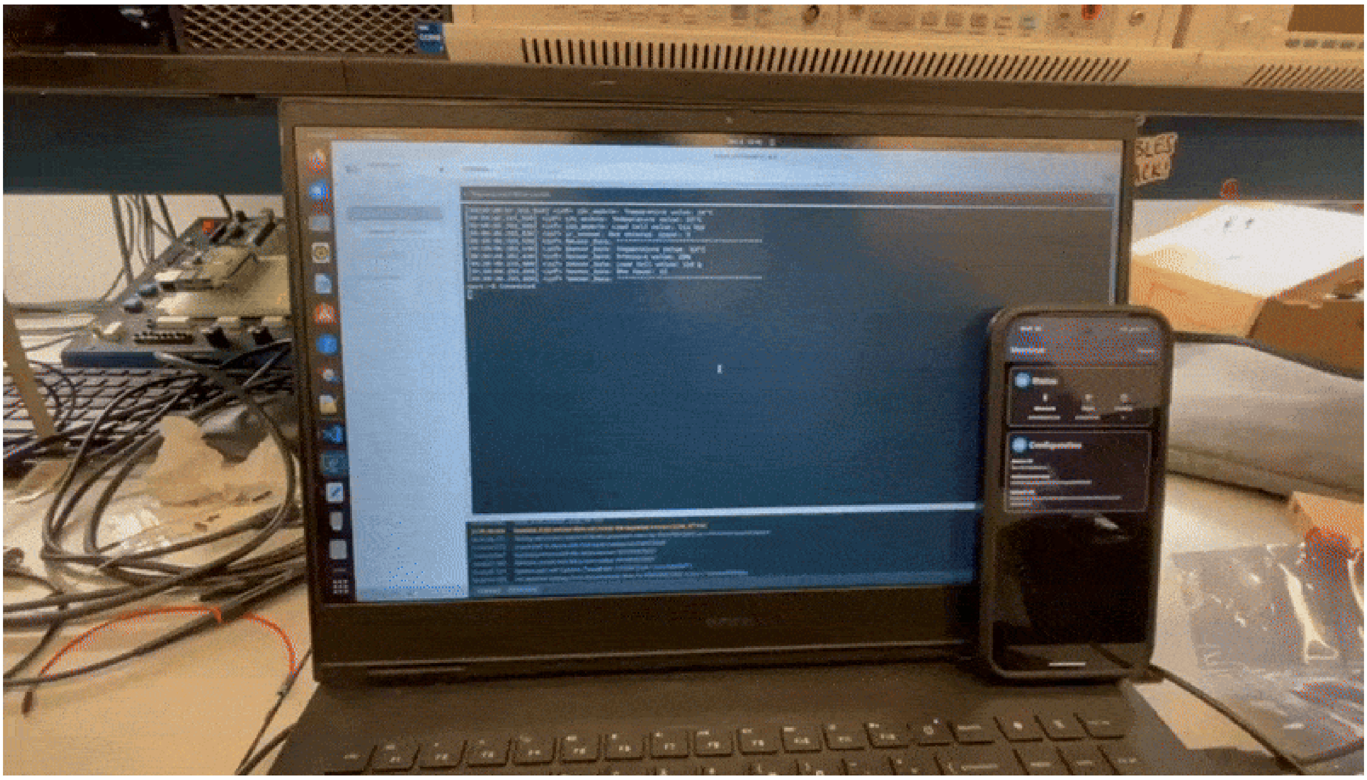
Loadcell



BLE Security

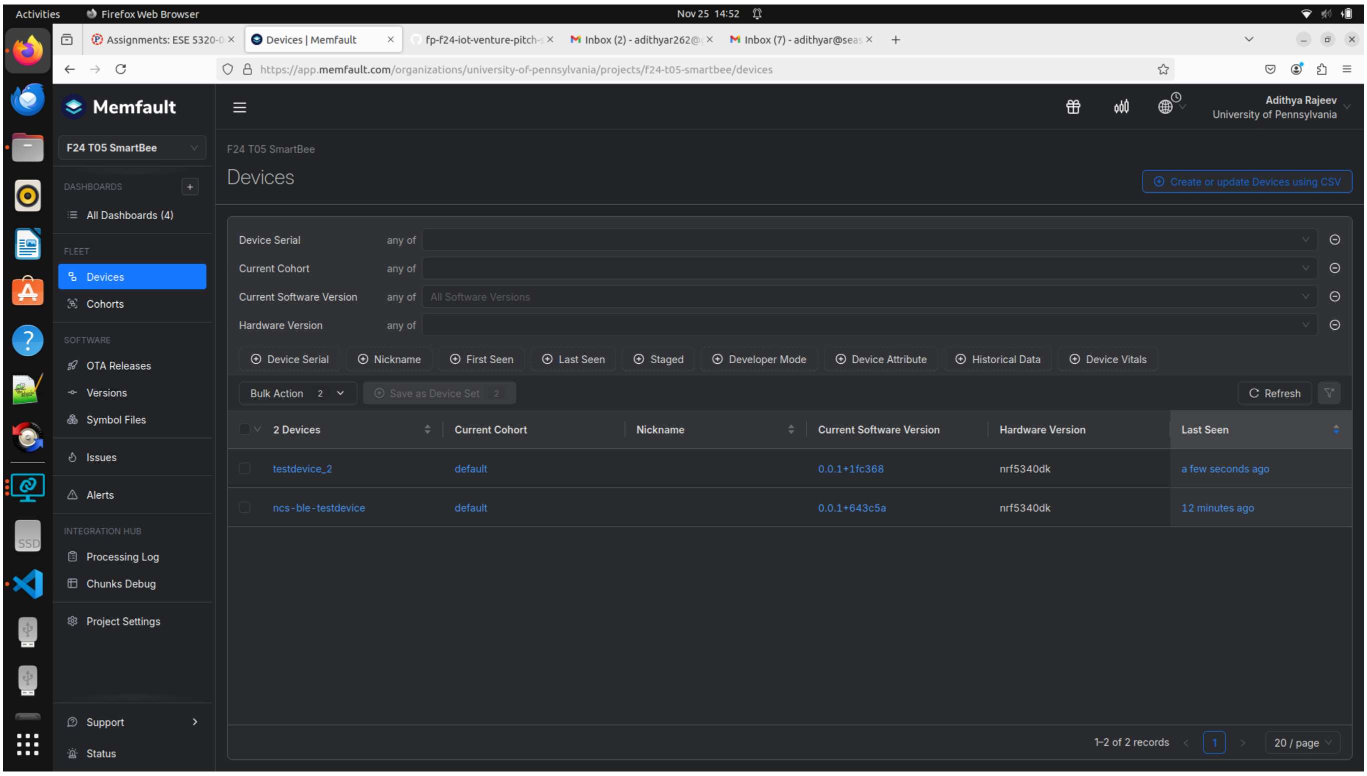


Memfault



```
- Memfault integration features
```

Multiple Devices



https://app.memfault.com/organizations/university-of-pennsylvania/projects/f24-t05-smartbee/devices

F24 T05 SmartBee

Devices

Device Serial any of

Current Cohort any of

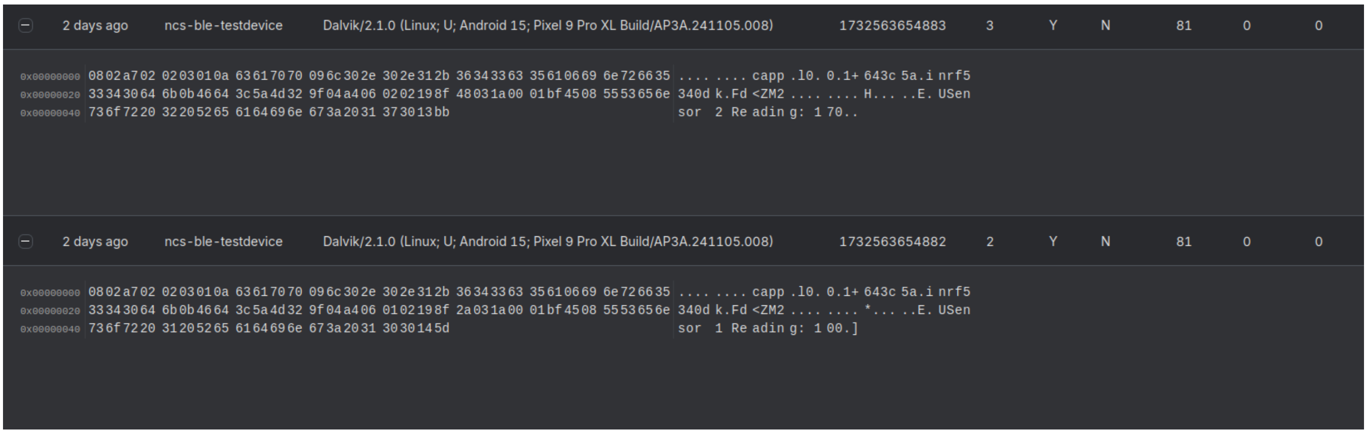
Current Software Version any of All Software Versions

Hardware Version any of

2 Devices	Current Cohort	Nickname	Current Software Version	Hardware Version	Last Seen
testdevice_2	default		0.0.1+1fc368	nrf5340dk	a few seconds ago
ncs-ble-testdevice	default		0.0.1+643c5a	nrf5340dk	12 minutes ago

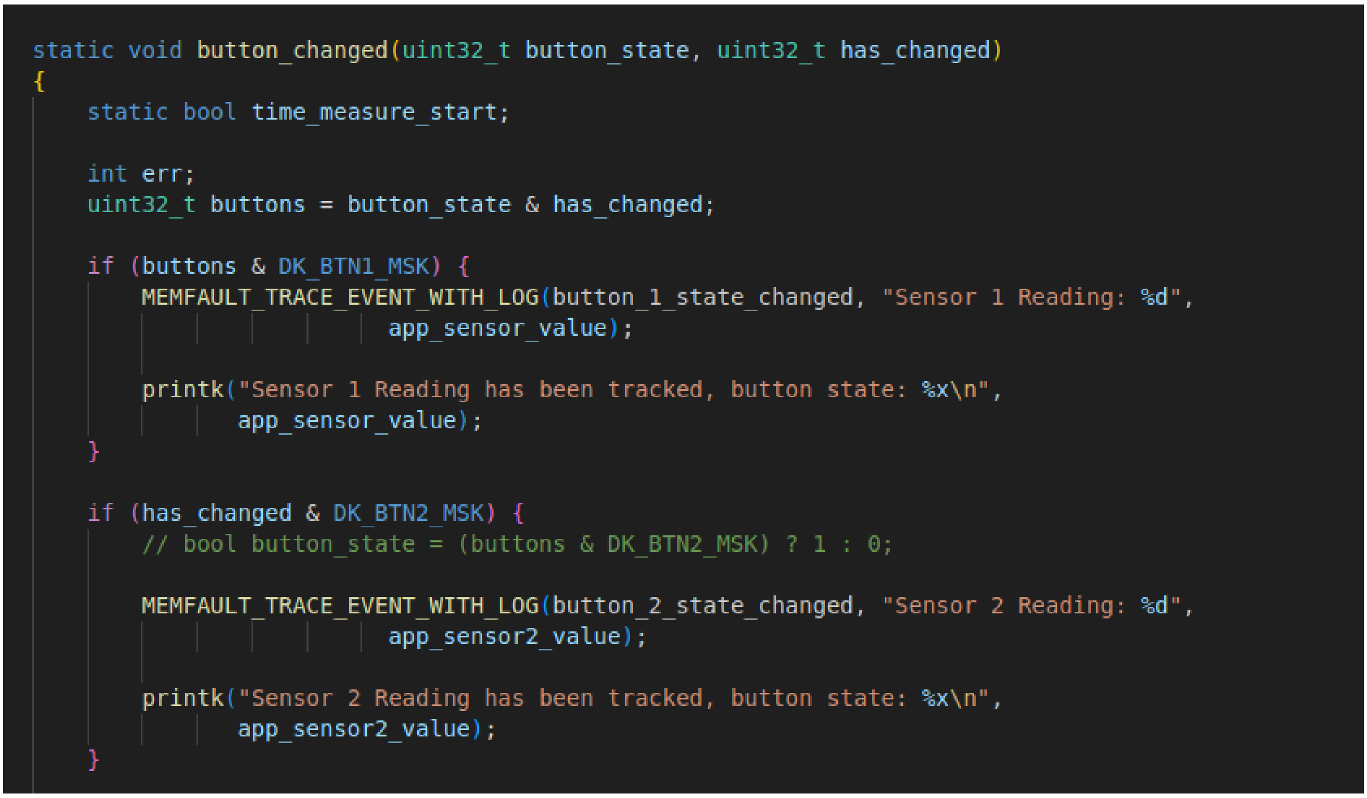
Device Data Feature

Memfault Chunk



```
2 days ago   ncs-ble-testdevice   Dalvik/2.1.0 (Linux; U; Android 15; Pixel 9 Pro XL Build/AP3A.241105.008)   1732563654883   3   Y   N   81   0   0

0x00000000 0802a702 0203010a 63617070 096c302e 302e312b 36343363 35610669 6e726635 .... .... capp .l0. 0.1+ 643c 5a.i nrf5
0x00000020 33343064 6b0b4664 3c5a4d32 9f04a406 0202198f 48031a00 01bf4508 5553656e 340d k.Fd <ZM2 .... .... H... ..E. USen
0x00000040 736f7220 32205265 6164696e 673a2031 373013bb                            sor  2 Re adin g: 1 70..

2 days ago   ncs-ble-testdevice   Dalvik/2.1.0 (Linux; U; Android 15; Pixel 9 Pro XL Build/AP3A.241105.008)   1732563654882   2   Y   N   81   0   0

0x00000000 0802a702 0203010a 63617070 096c302e 302e312b 36343363 35610669 6e726635 .... .... capp .l0. 0.1+ 643c 5a.i nrf5
0x00000020 33343064 6b0b4664 3c5a4d32 9f04a406 0102198f 2a031a00 01bf4508 5553656e 340d k.Fd <ZM2 .... .... *... ..E. USen
0x00000040 736f7220 31205265 6164696e 673a2031 3030145d                            sor  1 Re adin g: 1 00.]
```

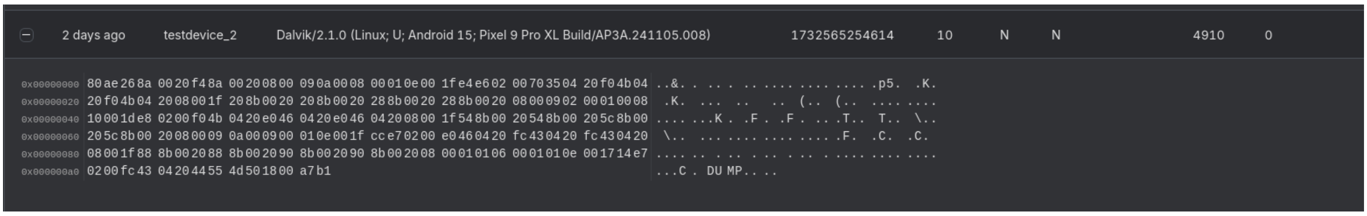
Code



```c
static void button_changed(uint32_t button_state, uint32_t has_changed)
{
    static bool time_measure_start;

    int err;
    uint32_t buttons = button_state & has_changed;

    if (buttons & DK_BTN1_MSK) {
        MEMFAULT_TRACE_EVENT_WITH_LOG(button_1_state_changed, "Sensor 1 Reading: %d",
                                      app_sensor_value);

        printk("Sensor 1 Reading has been tracked, button state: %x\n",
               app_sensor_value);
    }

    if (has_changed & DK_BTN2_MSK) {
        // bool button_state = (buttons & DK_BTN2_MSK) ? 1 : 0;

        MEMFAULT_TRACE_EVENT_WITH_LOG(button_2_state_changed, "Sensor 2 Reading: %d",
                                      app_sensor2_value);

        printk("Sensor 2 Reading has been tracked, button state: %x\n",
               app_sensor2_value);
    }
```

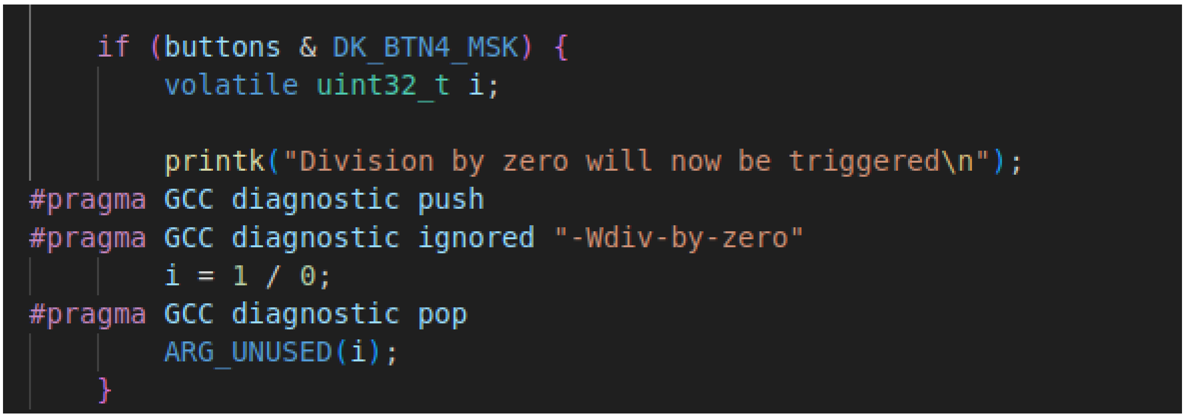
Core Dump Feature

Memfault Chunk



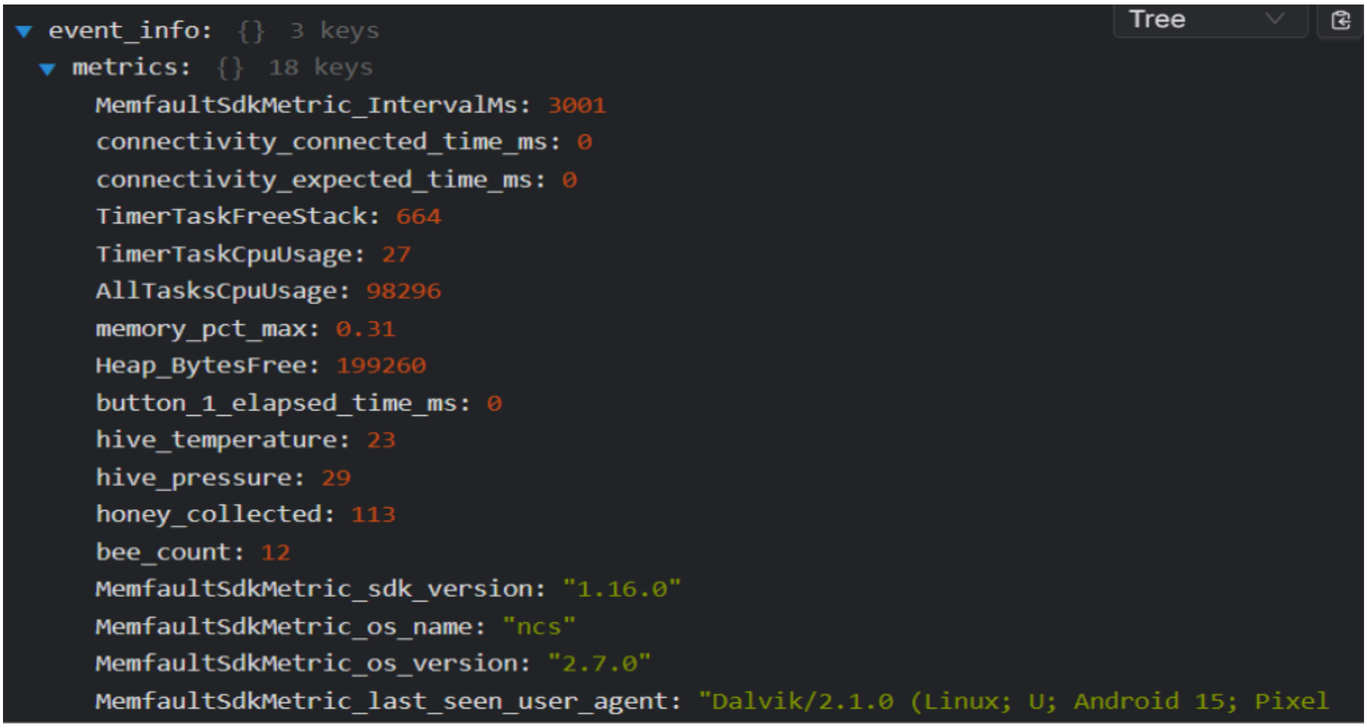
```
2 days ago   testdevice_2   Dalvik/2.1.0 (Linux; U; Android 15; Pixel 9 Pro XL Build/AP3A.241105.008)   1732565254614   10   N   N   4910   0

0x00000000 80ae268a 0020f48a 00200800 090a0008 00010e00 1fe4e602 00703504 20f04b04 ..&. . ... . .. .... .... .... .p5.  .K.
0x00000020 20f04b04 2008001f 208b0020 208b0020 288b0020 288b0020 08000902 00010008  .K.  ...  ..   ..  (..  (..  .... ....
0x00000040 10001de8 0200f04b 0420e046 0420e046 04200800 1f548b00 20548b00 205c8b00 .... ...K . .F . .F . ... .T..  T..  \..
0x00000060 205c8b00 20080009 0a000900 010e001f cce70200 e0460420 fc430420 fc430420  \..  ... .... .... .... .F.  .C.  .C.
0x00000080 08001f88 8b002088 8b002090 8b002090 8b002008 00010106 0001010e 001714e7 .... .. . .. . .. . .. . .... .... ....
0x000000a0 0200fc43 04204455 4d501800 a7b1                                        ...C . DU MP.. ..
```

Code



```c
    if (buttons & DK_BTN4_MSK) {
        volatile uint32_t i;

        printk("Division by zero will now be triggered\n");
#pragma GCC diagnostic push
#pragma GCC diagnostic ignored "-Wdiv-by-zero"
        i = 1 / 0;
#pragma GCC diagnostic pop
        ARG_UNUSED(i);
    }
```

Memfault Heartbeat Packet



```
▼ event_info: {} 3 keys
  ▼ metrics: {} 18 keys
      MemfaultSdkMetric_IntervalMs: 3001
      connectivity_connected_time_ms: 0
      connectivity_expected_time_ms: 0
      TimerTaskFreeStack: 664
      TimerTaskCpuUsage: 27
      AllTasksCpuUsage: 98296
      memory_pct_max: 0.31
      Heap_BytesFree: 199260
      button_1_elapsed_time_ms: 0
      hive_temperature: 23
      hive_pressure: 29
      honey_collected: 113
      bee_count: 12
      MemfaultSdkMetric_sdk_version: "1.16.0"
      MemfaultSdkMetric_os_name: "ncs"
      MemfaultSdkMetric_os_version: "2.7.0"
      MemfaultSdkMetric_last_seen_user_agent: "Dalvik/2.1.0 (Linux; U; Android 15; Pixel
```