Sprint 2 Code Review #1

Overview

- Review number：#1
- Review date: 30/4
- Review name: Sprint 2 ends code review
- Review tool used: GitHub pull request open ai code review actions
- Participants: Haoyang Zheng Yue Zhang Naixin Xu Junye Zhou Yinkai Chai Haitian Li

Goal

Purpose of review: Ensure code quality and make sure new code integrates with existing farmbot webapp and runs without issue. Ensure that button functions and image display functions are implemented correctly

Scope of review:

1. panel_header.tsx
   Check whether the thermal tab style and text are added correctly and meet the requirements
2. internal_urls.ts, nav_links.tsx, route_config.tsx
   Check whether the thermal tab is successfully bound to the link inside farmbot
3. thermal.tsx, thermal.css
   Check whether button and image display functions are implemented correctly and efficiently, and are suitable for existing webapps

Review process

• The code awaiting review is submitted on the branch "feature: display, request, download"
• The code writer initiates a pull request and requests to be merged into the main branch
• The reviewer will check the pull request on github and compare the changed content of the file with the new content

•Run the code to verify that the function is implemented correctly
• After all reviewers determine that the code meets the purpose of code review, the code will be merged into the master branch

Review results

• Total number of issues discovered: 1
• Severity: Moderate.
• Problem description: thermal.tsx currently displays thermal imaging images through the URL of the img tag without requesting it from within. This will cause the web server to be unable to obtain detailed data of the image.

Feedback from group

The overall code style is good, but there is some redundancy in the hook code of the component. These functions will be replaced in the next sprint. Currently, it cannot be implemented due to content security policy issues (CSP), so we can only retain this issue and temporarily merge the code for the purpose of displaying the function. Code review after the security policy issue is resolved will not allow this issue to exist.

Feedback from AI

ChatGPT review for src/Farmbot-Web-App-staging/frontend/thermal/thermal.tsx:
Here's the evaluation based on the provided checklist:

1. **Documentation Defects:**
   - **Naming:**

- The variable names like `isModalOpen`, `setModalOpen`, `imageSrc`, `setImageSrc`, `isFetching`, `setIsFetching` are clear and descriptive.
- The function names like `openModal`, `closeModal`, `toggleFetching`, and `handleDownload` are appropriately named and describe their functionality well.

- **Comment:**
  - Comments are minimal and generally describe what the code is intended to do, such as fetching images and avoiding cache issues with unique URLs. However, more detailed comments explaining why certain decisions were made (e.g., why `refreshInterval` is set to 3000) could enhance understanding.

2. **Visual Representation Defects:**
   - **Bracket Usage:**
     - Brackets are correctly used and placed.
   - **Indentation:**
     - There is a minor indentation inconsistency in the second `useEffect` block which affects readability.
   - **Long Line:**
     - The lines constructing `newUrl` are slightly long but still within an acceptable range for readability.
3. **Structure Defects:**
   - **Dead Code:**
     - No dead code is present.
   - **Duplication:**
     - The `fetchImage` function is duplicated in both `useEffect` hooks. This could be refactored into a single function defined outside the hooks to avoid duplication.
4. **New Functionality:**
   - **Use Standard Method:**
     - The code uses standard React and JavaScript methods appropriately.
5. **Resource Defects:**
   - **Variable Initialization:**
     - All variables are correctly initialized using `useState`.
   - **Memory Management:**
     - Proper cleanup is performed in `useEffect` by clearing intervals, which is good for memory management.
6. **Check Defects:**
   - **Check User Input:**
     - The code does not explicitly handle user input validation, especially considering URL parameters in image fetching. There should be checks to ensure that the inputs do not lead to security vulnerabilities such as URL injection.
7. **Interface Defects:**
   - **Parameter:**
     - Parameters and props are used correctly without any apparent issues.
8. **Logic Defects:**
   - **Compute:**
     - The logic to fetch images and manage UI state appears correct.
   - **Performance:**
     - Using timestamps to avoid caching is a simple and effective solution. However, continuously fetching images at a fixed interval could be optimized based on actual user needs or by allowing dynamic setting of the interval.

Summary of Recommended Changes:

1. Refactor to eliminate duplicate `fetchImage` function definitions.
2. Improve comments to include reasons behind specific implementation choices.
3. Address minor indentation inconsistencies for better readability.

4. Consider adding user input validation for URL parameters to enhance security.
5. Explore more efficient ways to manage image fetching, potentially allowing users to control the refresh interval.
