Here’s a strategy to implement your abandoned cart detection and handling logic within a WooCommerce plugin using the WPPB.me boilerplate.

### 1. \*\*Goal Overview\*\*

You want to:

- Check all active carts in the `woocommerce\_sessions` table.

- Define abandoned carts based on your custom expiration time, not WooCommerce's default session expiration (e.g., 1 hour).

- Implement a cron job to check periodically (using a PHP cron, external to WordPress) to detect abandoned carts and perform some action on them (e.g., send notifications, clean up, etc.).

### 2. \*\*Key Components of the Strategy\*\*

1. \*\*Database Query to Access All Active Carts\*\*:

- WooCommerce stores active cart sessions in the `woocommerce\_sessions` table. You will need to query this table to get all carts and their session details.

- This will include `session\_expiry`, which you will use to calculate the "last action" time based on the session expiration interval, as discussed.

2. \*\*Custom Expiration Logic\*\*:

- For each cart, calculate the last action time by subtracting the WooCommerce session expiration interval (typically 48 hours) from `session\_expiry`.

- Compare the last action time with the current time to determine if the cart has been abandoned based on your custom threshold (e.g., 1 hour).

3. \*\*Actions for Abandoned Carts\*\*:

- Once a cart is detected as abandoned, you may want to:

- Send an email reminder to the user.

- Mark the cart as abandoned in the database for future reference.

- Log the abandoned cart information for admin review.

4. \*\*Cron Job for Periodic Checks\*\*:

- A \*\*PHP-based cron job\*\* (instead of WP-Cron) ensures that the process runs independently of site visits. This is particularly useful for abandoned cart checks since you want it to run at a regular interval (e.g., every hour or every day).

- You can use server-level cron (Linux `cron` jobs) that hits a specific PHP script within your plugin.

### 3. \*\*High-Level Implementation Strategy\*\*

#### Step 1: \*\*Query the WooCommerce Sessions Table\*\*

- Write a function that queries the `woocommerce\_sessions` table, fetching all the session data (e.g., `session\_key`, `session\_expiry`).

- This function will loop through each session and calculate the \*\*last action time\*\* based on the session expiration interval (default is 48 hours).

- Determine if the cart has been inactive for longer than your custom threshold (e.g., 1 hour). If so, consider it abandoned.

#### Step 2: \*\*Define the Cron Job (PHP-Based)\*\*

- Create a PHP file within your plugin that runs the abandoned cart check function. This file will be called periodically by a server-side cron job (e.g., every hour).

- The cron job on the server will execute this PHP script independently of WordPress visits, ensuring that abandoned carts are checked regularly even if the site is not frequently visited.

#### Step 3: \*\*Abandoned Cart Actions\*\*

- Once a cart is detected as abandoned, you can:

- \*\*Send notification emails\*\*: Send an email to the customer reminding them about their abandoned cart.

- \*\*Log abandoned carts\*\*: Store information about the abandoned carts for later review in a custom table.

- \*\*Optional Cleanup\*\*: If needed, remove abandoned carts from the `woocommerce\_sessions` table.

#### Step 4: \*\*Server Cron Job Setup\*\*

- You’ll need to set up a cron job on your server to call your plugin's PHP script at regular intervals.

- Example cron job command on a Linux server (to run every hour):

```bash

0 \* \* \* \* /usr/bin/php /path/to/your/plugin/cron-check-abandoned-carts.php

```

This will ensure the check runs independently of site traffic.

### 4. \*\*Detailed Workflow\*\*

1. \*\*Cron Trigger\*\*: Every hour (or your chosen interval), the server cron job triggers your plugin’s abandoned cart check script.

2. \*\*Query Active Carts\*\*: The script queries the `woocommerce\_sessions` table to get all active cart sessions.

3. \*\*Custom Abandonment Logic\*\*: For each session:

- Calculate the last user action time using `session\_expiry - session\_expiration\_interval`.

- If the time since the last action exceeds your custom abandonment threshold (e.g., 1 hour), mark the cart as abandoned.

4. \*\*Perform Actions on Abandoned Carts\*\*:

- Send emails to users reminding them of their abandoned carts.

- Log details of the abandoned cart for administrative purposes.

- Optionally clean up old, abandoned sessions.

### 5. \*\*Considerations and Enhancements\*\*

- \*\*Error Handling and Logging\*\*: Make sure the cron job and abandoned cart checks include proper error handling. Log the results of each run (e.g., how many carts were checked and how many were marked abandoned).

- \*\*Scalability\*\*: If your site has many carts, make sure the query and abandoned cart checks are efficient. You might need pagination or batch processing for large datasets.

- \*\*Notification System\*\*: You can enhance the abandoned cart process by sending follow-up emails over a series of days or implementing recovery incentives like discount codes.

### 6. \*\*Next Steps\*\*

- \*\*Cron Setup\*\*: Set up a cron job on your server to run the PHP script that checks for abandoned carts.

- \*\*Plugin Development\*\*: Implement the functions in your plugin to handle the querying, abandonment detection, and actions for abandoned carts.

- \*\*Testing\*\*: Ensure the cron job runs correctly and that abandoned carts are being handled according to your custom logic.

### Summary

- Use a PHP-based cron job to trigger a custom script that runs regularly.

- Query the WooCommerce sessions table to get all active carts.

- Apply custom logic to determine if a cart is abandoned based on a user-defined expiration period (e.g., 1 hour of inactivity).

- Perform actions on abandoned carts, such as sending reminders or logging the data.

This approach ensures that your plugin checks for abandoned carts independently of site traffic, with custom logic and actions specific to your needs.