```php
<?php
namespace App\Jobs;

// use Illuminate\Bus\Queueable;
// use Illuminate\Queue\InteractsWithQueue;
// use Illuminate\Contracts\Queue\ShouldQueue;
use App\Model\IterableApi;
use Log;
use Osiset\BasicShopifyAPI\BasicShopifyAPI;
use Osiset\BasicShopifyAPI\Options;
use Osiset\BasicShopifyAPI\Session;

class SendEditOrderMail extends Job
{
    // use InteractsWithQueue, Queueable;

    private $data;
    private $hmacHeader;
    private $api;

    /**
     * Create a new job instance.
     *
     * @return void
     */
    // public function __construct($requestData, $hmacHeader)
    public function __construct($data)
    {
        $this->data = $data->data;
    }

    public function handle()
    {
        $this->iterableObj = new IterableApi();

        $options = new Options();
        $options->setVersion(env('SHOPIFY_API_VER_DEV'));

        $this->api = new BasicShopifyAPI($options);

        $this->api->setSession(new Session(env('SHOPIFY_URL_DEV'),
env('SHOPIFY_API_PASSWORD_DEV')));

        //$verified = $this->verifyWebhook($this->data, $this->hmacHeader);
        $verified = true;

        if ($verified) {
            $orderData = json_decode($this->data);

            $orderId = $orderData->id;
            Log::info("Processing order - " . $orderId);

            $emailId = $orderData->email;
            $orderName = $orderData->name;

            $noteAttributes = $this->formatData($orderData->note_attributes);

            foreach ($orderData->line_items as $orderLineItem) {
```

```
                if (!isset($webhook[$orderLineItem->variant_id])) {

                    $webhook[$orderLineItem->variant_id]['product_title'] =
$orderLineItem->title;
                    $webhook[$orderLineItem->variant_id]['variant_title'] =
$orderLineItem->variant_title;
                    $webhook[$orderLineItem->variant_id]['quantity'] =
$orderLineItem->quantity;
                    $webhook[$orderLineItem->variant_id]['variant_id'] =
$orderLineItem->variant_id;
                    $webhook[$orderLineItem->variant_id]['price'] = $orderLineItem-
>price;
                    $webhook[$orderLineItem->variant_id]['product_id'] =
$orderLineItem->product_id;

                }
            }

            $res = $this->getOrderFulfillment($orderId);
            $count = count($res);
            $request = array();

            foreach ($res[$count - 1]['line_items'] as $lineItem) {

                if (!isset($currentOrder[$lineItem['variant_id']])) {

                    $variantName = isset($webhook[$lineItem['variant_id']]) ?
$webhook[$lineItem['variant_id']]['variant_title'] : '';

                    $productName = isset($webhook[$lineItem['variant_id']]) ?
$webhook[$lineItem['variant_id']]['product_title'] : '';

                    $currentOrder[$lineItem['variant_id']]['variant_id'] =
$lineItem['variant_id'];
                    $currentOrder[$lineItem['variant_id']]['quantity'] =
$lineItem['quantity'];
                    $currentOrder[$lineItem['variant_id']]['variant_title'] =
$variantName;
                    $currentOrder[$lineItem['variant_id']]['product_title'] =
$productName;

                    $temp = [];
                    $temp['name'] = "Final : " . $variantName;
                    $temp['value'] = $lineItem['quantity'];
                    $request[] = $temp;
                }
            }

            foreach ($res[0]['line_items'] as $Item) {

                if (!isset($initialOrder[$Item['variant_id']])) {

                    $initialOrder[$Item['variant_id']]['variant_id'] =
$Item['variant_id'];
                    $initialOrder[$Item['variant_id']]['quantity'] =
$Item['quantity'];
                }
            }
```

```
            $lastUpdate = $this->formatLastUpdate($currentOrder);
            $isRefund = false;

            if (count($initialOrder) !== count($currentOrder)) {
                // need to send mail
                // if remove product from order

                if (empty($noteAttributes['noteAttributes'])) {
                    Log::info('need to send mail');
                    $this->sendEditMail($webhook, $initialOrder, $currentOrder,
$orderData);
                    $this->updateNoteAttributes($orderData,
$noteAttributes['fullNoteAttributes'], $lastUpdate);
                } else {


                    $lastUpdateFormat = $this->arrangeLastUpdate($lastUpdate);

                    $sendMail = false;
                    foreach ($noteAttributes['noteAttributes'] as $name => $value)
{

                        if(!in_array($name, array_keys($lastUpdateFormat))){
                            $sendMail = true;
                            break;
                        }

                        if (isset($lastUpdateFormat[$name]) && $value !==
$lastUpdateFormat[$name]) {
                            $sendMail = true;
                            break;
                        }


                    }

                    if ($sendMail) {
                        // send mail

                        Log::info('send mail');
                        $this->updateNoteAttributes($orderData,
$noteAttributes['fullNoteAttributes'], $lastUpdate);
                        $this->sendEditMail($webhook, $initialOrder, $currentOrder,
$orderData);
                        die('send mail => some value changes');

                    }
                }
            } else {
                // check condition  and send mail

                Log::info('check condition  and send mail');
                $sendMail = false;

                if (empty($noteAttributes['noteAttributes'])) {

                    foreach ($initialOrder as $variantId => $value) {
                        if ($value['quantity'] != $lastUpdate[$variantId]
['quantity']) {
```

```
                            $sendMail = true;
                            break;
                        }
                    }

                    $refundData = $orderData->refunds;
                    if(!empty($refundData)){
                        $refundLineItems = $orderData->refunds[0]-
>refund_line_items;

                        foreach ($initialOrder as $variantId => $value) {

                            foreach($refundLineItems as $refundDetails){
                                if ($refundDetails->line_item->variant_id ==
$variantId &&
                                    $refundDetails->line_item->quantity ==
$value['quantity']) {
                                    $isRefund = true;
                                    $sendMail = true;
                                    break;
                                }
                            }

                        }
                    }
                } else {


                    foreach ($lastUpdate as $variantId => $valueDetails) {

                        $prodName = $valueDetails['productTitle'];
                        $prodQty = $valueDetails['quantity'];

                        if (isset($noteAttributes['noteAttributes'][$prodName]) &&
$noteAttributes['noteAttributes'][$prodName] !== $prodQty) {

                            $sendMail = true;
                            break;
                        }
                    }
                }

            }

            if ($sendMail) {
                // send mail

                Log::info('send mail');
                $this->updateNoteAttributes($orderData,
$noteAttributes['fullNoteAttributes'], $lastUpdate);

                if($isRefund == true) {
                    $currentOrder = [];
                }

                $this->sendEditMail($webhook, $initialOrder, $currentOrder,
$orderData);

                die('send mail => some value changes');
```

```
        } else {
            Log::info('send mail => no need');
            die('send mail => no need');
        }

        die;
    }
}

function formatLastUpdate($currentOrder){

    $returnArray = array();
    if(!empty($currentOrder)){
        foreach($currentOrder as $orderID => $orderDetails) {

            $returnArray[$orderDetails['variant_id']] = array(
                "quantity" => $orderDetails['quantity'],
                "variantId" => $orderDetails['variant_id'],
                "productTitle" => $orderDetails['product_title']
            );
        }
    }

    return $returnArray;
}

function arrangeLastUpdate($lastUpdate){


    $returnData = [];

    if(!empty($lastUpdate)) {
        foreach($lastUpdate as $key => $value) {
            $prodName = $value['productTitle'];
            $prodQty = $value['quantity'];
            $returnData[$prodName] = $prodQty;
        }
    }

    return $returnData;
}

public function formatData($noteAttributes)
{
    $returnArray = [];
    $fullNoteAttributes = [];
    if (!empty($noteAttributes)) {
        foreach ($noteAttributes as $attributes) {
            $name = $attributes->name;
            $value = $attributes->value;

            if (strpos($attributes->name, "Final : ") !== false) {
                $name = str_replace("Final : ", "", $attributes->name);
                $returnArray[$name] = $value;
            }
            $fullNoteAttributes[$name] = $value;
        }
    }
```

```
    return [
        "noteAttributes" => $returnArray,
        "fullNoteAttributes" => $fullNoteAttributes,
    ];
}

public function updateNoteAttributes($data, $noteAttributes, $updatedData) {

    $orderId = $data->id;
    $orderTag = $data->tags;
    $orderNoteAttributesObject = $data->note_attributes;
    $orderNoteAttributes = [];
    if(!empty($orderNoteAttributesObject)){
        foreach($orderNoteAttributesObject as $object){
            $temp['name'] = $object->name;
            $temp['value'] = $object->value;
            $orderNoteAttributes[] = $temp;
        }
    }

    $noteRequest = [];

    // update note attributes
    if(!empty($updatedData)) {
        foreach($updatedData as $variantId => $variantIdDetails) {

            $variantQty = $variantIdDetails['quantity'];
            $name = $variantIdDetails['productTitle'];

            if(!empty($orderNoteAttributes)) {

                foreach($orderNoteAttributes as $key => $noteValues) {

                    if(!is_array($noteValues)) {
                        $noteValues = (array)$noteValues;
                    }

                    $noteName = $noteValues['name'];
                    $noteQty  = $noteValues['value'];

                    // $noteName = $noteValues->name;
                    // $noteQty  = $noteValues->value;

                    $checkName = str_replace("Final : ", "", $noteName);

                    if(strpos($noteName, $name) !== false) {
                        $noteQty = $variantQty;

                        $orderNoteAttributes[$key]['value'] = $noteQty;
                        // $orderNoteAttributes[$key]->value = $noteQty;
                    }

                }
            }

            if(!array_key_exists($name, $noteAttributes)){
                $temp['name'] = "Final : ".$name;
                $temp['value'] = $variantQty;
```

```
                    $orderNoteAttributes[] = $temp;
                }
            }
        }

        $newAttributes = (array)$orderNoteAttributes;

        $request['order']['id'] = $orderId;
        $request['order']['note_attributes'] = $newAttributes;

        $result = $this->api->rest(
            'PUT',
            '/admin/api/'.env('SHOPIFY_API_VER_DEV').'/orders/'.$orderId.'.json',
            $request
        );

        if (isset($result->errors) && $result->errors == true) {

            echo "error";
            return null;
        } else {

            $updatedOrder = $result['body']->container['order'];
        }
    }

    public function getOrderFulfillment($orderId)
    {

        $result = $this->api->rest(
            'GET',
            '/admin/api/' . env('SHOPIFY_API_VER_DEV') . '/orders/' . $orderId . '/
fulfillment_orders.json',
            null
        );

        if (isset($result->errors) && $result->errors == true) {
            return null;
        } else {

            $fulfillmentDetails = $result['body']->container['fulfillment_orders'];
        }

        return $fulfillmentDetails;
    }

    public function getImageUrl($productId)
    {

        $imageUrl = '';
        if (!empty($productId)) {
            $result = $this->api->rest(
                'GET',
                '/admin/api/' . env('SHOPIFY_API_VER_DEV') . '/products/' .
$productId . '.json',
                null
            );

            if (isset($result->errors) && $result->errors == true) {
```

```
                return null;
            } else {

                $imageDetails = $result['body']->container['product']['images'];
                $imageUrl = isset($imageDetails[0]['src']) ? $imageDetails[0]
['src'] : "";
            }

            return $imageUrl;
        }
    }
    public function setLineItemDetails($webhook, $initialOrder, $currentOrder,
$orderData)
    {

        $lineItemContent = '';
        foreach ($currentOrder as $variantId => $variantDetails) {

            $productName = '';
            $variantName = '';
            $oldQty = '';
            $newQty = '';
            $imageUrl = $this->getImageUrl($webhook[$variantDetails['variant_id']]
['product_id']);

            if (isset($initialOrder[$variantId])) {
                $productName = $variantDetails['product_title'];
                $variantName = $variantDetails['variant_title'];

                $productQty = $variantDetails['quantity'];
                $oldQty = $initialOrder[$variantId]['quantity'];
                $newQty = $currentOrder[$variantId]['quantity'];
                $price = isset($webhook[$variantId]) ? $webhook[$variantId]
['price'] : '0.00';
                $total = $price * $productQty;

                $lineItemContent .= '<tr class="order-list__item" style="width:
100%;">
                <td class="order-list__item__cell"
                style="font-family: -apple-system, BlinkMacSystemFont, &quot;Segoe
UI&quot;, &quot;Roboto&quot;, &quot;Oxygen&quot;, &quot;Ubuntu&quot;,
&quot;Cantarell&quot;, &quot;Fira Sans&quot;, &quot;Droid Sans&quot;,
&quot;Helvetica Neue&quot;, sans-serif; padding-bottom: 15px;">
                <table style="border-spacing: 0; border-collapse: collapse;">
                <tbody>
                <tr>
                <td
                style="font-family: -apple-system, BlinkMacSystemFont, &quot;Segoe
UI&quot;, &quot;Roboto&quot;, &quot;Oxygen&quot;, &quot;Ubuntu&quot;,
&quot;Cantarell&quot;, &quot;Fira Sans&quot;, &quot;Droid Sans&quot;,
&quot;Helvetica Neue&quot;, sans-serif;">
                <img align="left" alt="" class="order-list__product-image"
height="60"
                src="' . $imageUrl . '"
                style="margin-right: 15px; border-radius: 8px; border: 1px solid
#e5e5e5;" width="60" />
                </td>
                <td class="order-list__product-description-cell"
                style="font-family: -apple-system, BlinkMacSystemFont, &quot;Segoe
```

```
UI&quot;, &quot;Roboto&quot;, &quot;Oxygen&quot;, &quot;Ubuntu&quot;,
&quot;Cantarell&quot;, &quot;Fira Sans&quot;, &quot;Droid Sans&quot;,
&quot;Helvetica Neue&quot;, sans-serif; width: 100%;">
                <span class="order-list__item-title"
                style="font-size: 16px; font-weight: 600; line-height: 1.4; color:
#555;">' . $productName . ' × ' . $newQty . '</span>
                <br />
                <span class="order-list__item-variant" style="font-size: 14px;
color: #999;">' . $variantName . '</span>
                </td>
                <td class="order-list__price-cell"
                style="font-family: -apple-system, BlinkMacSystemFont, &quot;Segoe
UI&quot;, &quot;Roboto&quot;, &quot;Oxygen&quot;, &quot;Ubuntu&quot;,
&quot;Cantarell&quot;, &quot;Fira Sans&quot;, &quot;Droid Sans&quot;,
&quot;Helvetica Neue&quot;, sans-serif; white-space: nowrap;">
                <p align="right" class="order-list__item-price"
                style="color: #555; line-height: 150%; font-size: 16px; font-
weight: 600; margin: 0 0 0 15px;">
                $' . $total . '
                </p>
                </td>
                </tr>
                </tbody>
                </table>
                </td>
                </tr>';

            } else {

                $productName = $variantDetails['product_title'];
                $variantName = $variantDetails['variant_title'];
                $productQty = $variantDetails['quantity'];
                $price = isset($webhook[$variantId]) ? $webhook[$variantId]
['price'] : '0.00';
                $total = $price * $productQty;

                $lineItemContent .= '<tr class="order-list__item" style="width:
100%;">
                <td class="order-list__item__cell"
                style="font-family: -apple-system, BlinkMacSystemFont, &quot;Segoe
UI&quot;, &quot;Roboto&quot;, &quot;Oxygen&quot;, &quot;Ubuntu&quot;,
&quot;Cantarell&quot;, &quot;Fira Sans&quot;, &quot;Droid Sans&quot;,
&quot;Helvetica Neue&quot;, sans-serif; padding-bottom: 15px;">
                <table style="border-spacing: 0; border-collapse: collapse;">
                <tbody>
                <tr>
                <td
                style="font-family: -apple-system, BlinkMacSystemFont, &quot;Segoe
UI&quot;, &quot;Roboto&quot;, &quot;Oxygen&quot;, &quot;Ubuntu&quot;,
&quot;Cantarell&quot;, &quot;Fira Sans&quot;, &quot;Droid Sans&quot;,
&quot;Helvetica Neue&quot;, sans-serif;">
                <img align="left" alt="" class="order-list__product-image"
height="60"

src="https://cdn.shopify.com/s/files/1/0066/7569/3639/products/NM2844L600VITD3GUMMI
ESfront_compact_cropped.jpg?v=1588875151"
                style="margin-right: 15px; border-radius: 8px; border: 1px solid
#e5e5e5;" width="60" />
                </td>
```

```
                <td class="order-list__product-description-cell"
                style="font-family: -apple-system, BlinkMacSystemFont, &quot;Segoe
UI&quot;, &quot;Roboto&quot;, &quot;Oxygen&quot;, &quot;Ubuntu&quot;,
&quot;Cantarell&quot;, &quot;Fira Sans&quot;, &quot;Droid Sans&quot;,
&quot;Helvetica Neue&quot;, sans-serif; width: 100%;">
                <span class="order-list__item-title"
                style="font-size: 16px; font-weight: 600; line-height: 1.4; color:
#555;">' . $productName . ' × ' . $productQty . '</span>
                <br />
                <span class="order-list__item-variant" style="font-size: 14px;
color: #999;">' . $variantName . '</span>
                </td>
                <td class="order-list__price-cell"
                style="font-family: -apple-system, BlinkMacSystemFont, &quot;Segoe
UI&quot;, &quot;Roboto&quot;, &quot;Oxygen&quot;, &quot;Ubuntu&quot;,
&quot;Cantarell&quot;, &quot;Fira Sans&quot;, &quot;Droid Sans&quot;,
&quot;Helvetica Neue&quot;, sans-serif; white-space: nowrap;">
                <p align="right" class="order-list__item-price"
                style="color: #555; line-height: 150%; font-size: 16px; font-
weight: 600; margin: 0 0 0 15px;">
                $' . $total . '
                </p>
                </td>
                </tr>
                </tbody>
                </table>
                </td>
                </tr>';
            }
        }

        return $lineItemContent;
    }

    public function setRemovedLineItemsDetails($webhook, $initialOrder,
$currentOrder)
    {
        $removedOrderSet = false;
        $removedOrders = '<table class="row section" style="width: 100%; border-
spacing: 0; border-collapse: collapse; border-top-width: 1px; border-top-color:
#e5e5e5; border-top-style: solid;">
        <tbody>
          <tr>
            <td class="section__cell" style="font-family: -apple-system,
BlinkMacSystemFont, &quot;Segoe UI&quot;, &quot;Roboto&quot;, &quot;Oxygen&quot;,
&quot;Ubuntu&quot;, &quot;Cantarell&quot;, &quot;Fira Sans&quot;, &quot;Droid
Sans&quot;, &quot;Helvetica Neue&quot;, sans-serif; max-width: 500px; padding: 20px
0 0;">
              <center>
                <table class="container" style="width: 560px; text-align: left;
border-spacing: 0; border-collapse: collapse; max-width: 500px; margin: 0 auto;">
                  <tbody>
                    <tr>
                      <td style="font-family: -apple-system, BlinkMacSystemFont,
&quot;Segoe UI&quot;, &quot;Roboto&quot;, &quot;Oxygen&quot;, &quot;Ubuntu&quot;,
&quot;Cantarell&quot;, &quot;Fira Sans&quot;, &quot;Droid Sans&quot;,
&quot;Helvetica Neue&quot;, sans-serif;">
                        <h3 style="font-weight: normal; font-size: 20px; margin: 0
0 25px;">Removed Products
```

```
                    </h3>
                  </td>
                </tr>
              </tbody>
            </table>
            <table class="container" style="width: 560px; text-align: left;
border-spacing: 0; border-collapse: collapse; max-width: 500px; margin: 0 auto;">
            <tbody>
              <tr>
                <td style="font-family: -apple-system, BlinkMacSystemFont,
&amp;quot;Segoe UI&amp;quot;, &amp;quot;Roboto&amp;quot;, &amp;quot;Oxygen&amp;quot;, &amp;quot;Ubuntu&amp;quot;,
&amp;quot;Cantarell&amp;quot;, &amp;quot;Fira Sans&amp;quot;, &amp;quot;Droid Sans&amp;quot;,
&amp;quot;Helvetica Neue&amp;quot;, sans-serif;">
                  <table class="row" style="width: 100%; border-spacing: 0;
border-collapse: collapse;">
                    <tbody>';

        foreach ($initialOrder as $initial) {
            $productName = $webhook[$initial['variant_id']]['product_title'];
            $variantName = $webhook[$initial['variant_id']]['variant_title'];

            $productQty = $webhook[$initial['variant_id']]['quantity'];
            $price = isset($webhook[$initial['variant_id']]['price']) ?
$webhook[$initial['variant_id']]['price'] : '0.00';
            $total = "$" . $price * $productQty;
            $imageUrl = $this->getImageUrl($webhook[$initial['variant_id']]
['product_id']);

            if (!isset($currentOrder[$initial['variant_id']])) {
                $removedOrderSet = true;
                $removedOrders .= '
                <tr class="order-list__item" style="width: 100%;">
                    <td class="order-list__item__cell" style="font-family: -apple-
system, BlinkMacSystemFont, &amp;quot;Segoe UI&amp;quot;, &amp;quot;Roboto&amp;quot;,
&amp;quot;Oxygen&amp;quot;, &amp;quot;Ubuntu&amp;quot;, &amp;quot;Cantarell&amp;quot;, &amp;quot;Fira
Sans&amp;quot;, &amp;quot;Droid Sans&amp;quot;, &amp;quot;Helvetica Neue&amp;quot;, sans-serif;
padding-bottom: 15px;">
                        <table style="border-spacing: 0; border-collapse:
collapse;">
                            <tbody>
                                <tr>
                                    <td style="font-family: -apple-system,
BlinkMacSystemFont, &amp;quot;Segoe UI&amp;quot;, &amp;quot;Roboto&amp;quot;, &amp;quot;Oxygen&amp;quot;,
&amp;quot;Ubuntu&amp;quot;, &amp;quot;Cantarell&amp;quot;, &amp;quot;Fira Sans&amp;quot;, &amp;quot;Droid
Sans&amp;quot;, &amp;quot;Helvetica Neue&amp;quot;, sans-serif;">
                                        <div style="width: 62px;border-radius:
8px;overflow: hidden;     background-image:
url(https://cdn.shopify.com/s/files/1/0066/7569/3639/files/soldout.png?
v=1602823461); background-repeat: no-repeat; background-size: 100% 100%; margin-
right: 15px;"><img alt="" class="order-list__product-image" height="60" src="' .
$imageUrl . '" style="border-radius: 8px; border: 1px solid #e5e5e5; opacity: 0.5;"
width="60" />
                                        </div>
                                    </td>
                                    <td class="order-list__product-description-
cell" style="font-family: -apple-system, BlinkMacSystemFont, &amp;quot;Segoe UI&amp;quot;,
&amp;quot;Roboto&amp;quot;, &amp;quot;Oxygen&amp;quot;, &amp;quot;Ubuntu&amp;quot;, &amp;quot;Cantarell&amp;quot;,
&amp;quot;Fira Sans&amp;quot;, &amp;quot;Droid Sans&amp;quot;, &amp;quot;Helvetica Neue&amp;quot;, sans-
serif; width: 100%;"><span class="order-list__item-title" style="font-size: 16px;
```

```
font-weight: 600; line-height: 1.4; color: #555;"><del style="opacity:0.5">' .
$productName . ' × ' . $productQty . '</del></span>
                                        <br />
                                        <span class="order-list__item-variant"
style="font-size: 14px; color: #999;"><del style="opacity:0.5">' . $variantName .
'</del></span>
                                    </td>
                                    <td class="order-list__price-cell" style="font-
family: -apple-system, BlinkMacSystemFont, &quot;Segoe UI&quot;,
&quot;Roboto&quot;, &quot;Oxygen&quot;, &quot;Ubuntu&quot;, &quot;Cantarell&quot;,
&quot;Fira Sans&quot;, &quot;Droid Sans&quot;, &quot;Helvetica Neue&quot;, sans-
serif; white-space: nowrap;"><del class="order-list__item-original-price"
style="font-size: 14px; display: block; text-align: right; color: #999; opacity:
0.5;"></del>
                                        <p align="right" class="order-list__item-
price" style="color: #555; line-height: 150%; font-size: 16px; font-weight: 600;
margin: 0 0 0 15px;"><del style="opacity:0.5">' . $total . '</del>
                                        </p>
                                    </td>
                                </tr>
                            </tbody>
                        </table>
                    </td>
                </tr>';
            }
        }

        $removedOrders .= '
        </tbody>
            </table>
            </td>
        </tr>
        </tbody>
        </table>
        </center>
        </td>
        </tr>
        </tbody>
        </table>';

        return ($removedOrderSet == true) ? $removedOrders : '';
    }

    public function getRefundSubTotalTaxAmount($orderData){
        $refundTaxAmount = 0;
        $refundSubtotalAmount = 0;

        if(empty($orderData->refunds)) {
            return [
                "refundTaxAmount" => $refundTaxAmount,
                "refundSubtotalAmount" => $refundSubtotalAmount
            ];
        } else {

            if(isset($orderData->refunds[0]->refund_line_items)){
                foreach($orderData->refunds[0]->refund_line_items as $refundData) {
                    $refundTaxAmount += $refundData->total_tax;
                    $refundSubtotalAmount += $refundData->subtotal;
                }
```

```
            }
        }

        return [
            "refundTaxAmount" => $refundTaxAmount,
            "refundSubtotalAmount" => $refundSubtotalAmount
        ];
    }

    public function sendEditMail($webhook, $initialOrder, $currentOrder,
$orderData)
    {
        $orderName = $orderData->name;

        $refundDetails = $this->getRefundSubTotalTaxAmount($orderData);

        // $orderTotal = '$' . $orderData->total_price;
        $orderTotal = '$'.($orderData->total_price -
$refundDetails['refundSubtotalAmount']);

        // $subTotal = '$' . $orderData->subtotal_price;
        $subTotal = '$'.($orderData->subtotal_price -
$refundDetails['refundSubtotalAmount']);

        $address = $this->getAddressContent($orderData);

        //$totalTax = '$' . $orderData->total_tax;
        $totalTax = '$'.($orderData->total_tax -
$refundDetails['refundTaxAmount']);

        $currency = $orderData->currency;
        $orderStatusUrl = $orderData->order_status_url;
        $paymentName = isset($orderData->payment_gateway_names[0]) ? $orderData-
>payment_gateway_names[0] : '';
        $shippingName = isset($orderData->shipping_lines[0]->title) ? $orderData-
>shipping_lines[0]->title : '';
        $lineItemDetails = $this->setLineItemDetails($webhook, $initialOrder,
$currentOrder, $orderData);
        $removedDetails = $this->setRemovedLineItemsDetails($webhook,
$initialOrder, $currentOrder);
        $email = $orderData->email;


        $iterableMailRequest = array(

            "campaignId" => intval(env('ITERABLE_CAMPAIGN_ID', true)),
            "recipientEmail" => $email,
            "dataFields" => array(
                "orderName" => $orderName,
                "totalPrice" => $orderTotal,
                "shippingAddress" => $address['shippingAddress'],
                "billingAddress" => $address['billingAddress'],
                "subTotal" => $subTotal,
                "totalTax" => $totalTax,
                "currency" => $currency,
                "shippingMethod" => $shippingName,
                "paymentMethod" => $paymentName,
                "lineItemDetails" => $lineItemDetails,
                "removedLineItemDetails" => $removedDetails,
```

```
                "orderStatusUrl" => $orderStatusUrl,
            ),
        );

        $mailTriggerResponse = $this->iterableObj-
>sendMailFromIterable(json_encode($iterableMailRequest));

        if (strtolower(trim($mailTriggerResponse->code)) == 'success') {
            echo "mail send..... ";
            Log::info('mail send.....');
            $returnData = [
                "mailStatus" => true,
                "message" => "mail triggered successfully (" . $email . ")",
            ];

            return 'suceess';

        } else {
            echo "mail not send..... ";
            Log::info('mail  not send.....');
            Log::info(json_encode($mailTriggerResponse));
            $error = $mailTriggerResponse->code;

            if (empty($error)) {

                $returnData = [
                    'error' => 'failed to send mail(' . $email . ')',
                ];
            } else {

                $returnData = [
                    'error' => $error . ' (' . $email . ')',
                ];
            }
            return 'Error';
        }

    }

    public function getAddressContent($orderData)
    {

        $shippingAddress = isset($orderData->shipping_address) ? $orderData-
>shipping_address : '';
        $billingAddress = isset($orderData->billing_address) ? $orderData-
>billing_address : '';

        $shippingAddressContent = '';
        $billingAddressContent = '';

        if (!empty($shippingAddress)) {
            $shippingAddressContent .= '
            <p style="color:#777777;font-size:16px;margin:0;line-height:150%;">' .
            $shippingAddress->first_name . '  ' . $shippingAddress->last_name .
            '<br />' . $shippingAddress->company . '<br />' .
            $shippingAddress->address1 . '<br />' . $shippingAddress->province .
            '<br />' . $shippingAddress->country . '<br />' . $shippingAddress->zip
. '</p>';
        } else {
```

```
            $shippingAddressContent .= '
            <p style="color:#777777;font-size:16px;margin:0;line-height:150%;">' .
                '  <br />  <br /> <br /> <br /> <br /> </p>';
        }

        if (!empty($billingAddress)) {
            $billingAddressContent .= '
            <p style="color:#777777;font-size:16px;margin:0;line-height:150%;">' .
            $billingAddress->first_name . '  ' . $billingAddress->last_name .
            '<br />' . $billingAddress->company . '<br />' .
            $billingAddress->address1 . '<br />' . $billingAddress->province .
            '<br />' . $billingAddress->country . '<br />' . $billingAddress->zip .
'</p>';
        } else {
            $billingAddressContent .= '
            <p style="color:#777777;font-size:16px;margin:0;line-height:150%;">' .
                '  <br />  <br /> <br /> <br /> <br /> </p>';
        }

        return [
            "billingAddress" => $billingAddressContent,
            "shippingAddress" => $shippingAddressContent,
        ];

    }
}
```