PHASE 5: SECURE CHECKOUT FLOW (DEADLINE: 20 APRIL 2025; **with bonus extension: 27 April, 2025**) (SUBTOTAL: 16')

This is a **tough** phase, yet the most critical phase to escalate the professional level of your website to the next level. (You'll likely be offered a job if you can demonstrate such a level of web programming skills) The implementation has already been outlined below (the requirements are not strict if you are using other languages/SDK). Be prepared to spend a substantial amount of time in debugging.

1. Sign up at https://developer.paypal.com/ and create test accounts: _____/ 1'
2. Enclose your shopping cart with a <form> element _____/ 3'
    - Use the Cart Upload Command of PayPal Website Payment Standard (cmd=_cart&upload=1)
    - Insert additional hidden fields that are required by PayPal (Read the first reference)
        - business, charset, currency_code, item_name_X, item_number_X, quantityX
        - invoice and custom
    - Create a checkout button that submits the form

    NOTE: The above if for NVP/SOAP method; If you are using API v2, your server can POST to /v2/checkout/orders with shopping cart items received from user's form submission.
3. When the checkout button is clicked (**Order validation**): _____/ 4'
    - Pass ONLY the *pid* and *quantity* of every individual product to your server using AJAX and cancel the default form submission
    - Server generates a digest that is composed of at least:
        - Currency
        - Merchant's email address
        - A random salt
        - The *pid* and *quantity* of each selected product (Is quantity positive number?)
        - The current price of each selected product gathered from DB
        - **The total price of all selected products**

    Hint: separate them with a delimiter before passing to a hash function
    - Server stores all the items to generate the digest into a **new database** table called *orders*
        - The user could be logged in or as "guest" to purchase, store username with order in DB
    - Pass the newly inserted **orderID (identifying the order)** and the generated digest back to the client by putting them into the hidden fields of invoice and custom respectively
    - Clear the shopping cart at the client-side
    - Submit the form now to PayPal using programmatic form submission
4. Setup an endpoint/webhook to get notified once a payment is completed
    - Validate the authenticity of data by verifying that it is indeed sent from PayPal _____/ 1'
        - Your endpoint is served over HTTPS
    - Check that the transaction has not been previously processed _____/ 1'
    - Regenerate a digest with the data provided by PayPal (same order and algorithm) _____/ 2'
    - Validate the digest against the one stored in the database table *orders* _____/ 2'
        - If validated, the integrity of the hashed fields is assured
        - Save the transaction and product list (pid, quantity and price) into DB

Debugging Hint: You can print out the parameters passed by PayPal to console for checking

5. After the buyer has finished paying with PayPal, auto redirect the buyer back to your shop _____/ 1'
6. Display the DB *orders* table in admin panel: product list, payment status…etc. _____/ 1'
7. Let members check what they have purchased in the most recent five orders. _____/ 4'
    - Show the order information in the member portal.

Note: You can use Stripe instead; to get a demo/sandbox account, just register without validation.
https://docs.stripe.com/sandboxes/dashboard/manage

There are two general approaches: using (REST) API or SDK provided by the payment gateway.
https://developer.paypal.com/studio/checkout/standard/integrate
https://github.com/paypal/PayPal-TypeScript-Server-SDK
https://developer.paypal.com/docs/api/orders/v2/

https://docs.stripe.com/get-started/development-environment
https://github.com/stripe/stripe-node (and example therein)
https://docs.stripe.com/payments/accept-a-payment?platform=web&ui=embedded-form