Homepage & Popup Introduction (index.php)

This script serves as the main landing page of the e-commerce platform, combining backend data retrieval with a dynamic frontend introduction. It begins by including the configuration file and fetching all products from the products table using a prepared statement. The page then displays a styled popup modal on load, presenting an overview of the site's key features. The layout includes modular templates for the header, main content, and footer to maintain consistent structure. A help link is also provided to guide users in navigating the page. This script sets the stage for a dynamic and user-friendly shopping experience while emphasizing the system's functionality from the very first interaction.

Header Template with SEO & Sort Menu (header.php or embedded head section)

This script sets up a dynamic and SEO-optimized <head> section and header navigation bar for GPU Paradise. It initializes default meta tags (title, description, keywords) for search engines, supports dynamic theming via a cookie-controlled stylesheet switch, and includes responsive layout styling using Google Fonts and Material Symbols. The header grid features a company logo, a comprehensive navigation menu with icons, and a sort menu dropdown. The sort control is powered by JavaScript and allows users to sort products by name or price. The header also conditionally displays links like "My Answers" and "Logout" if the user is logged in, ensuring a personalized experience across the site.

Product Listing with Dynamic Sorting (index.php main section)

This script retrieves and displays a list of products from the database with dynamic sorting capabilities based on a GET parameter. Supported sort options include alphabetical order, price (ascending), price (descending), and the default (most recently added first). It uses a flexible SQL ORDER BY clause and securely fetches all product data using prepared statements. Each product is rendered with its image, name, price, and a link to its detailed page, providing users with a clean and responsive shopping experience that adapts to their preferred viewing order.

Product Details (product_details.php)

This script dynamically displays detailed information for a single product based on the id parameter passed via the URL. It begins by validating the product ID and querying the database to fetch the corresponding product record. If the product does not exist or the ID is invalid, an error message is shown. Upon successful retrieval, the product's image, name, description, and price are displayed in a styled layout. A purchase option dropdown allows users to choose between "New," "Open Box," or "Refurbished" versions of the product. Based on the selected option, relevant action buttons appear dynamically, directing users to the appropriate cart or purchase pages. The page also includes visual trust indicators (e.g., Secure Checkout, Fast Shipping) and integrates a customer review section. Reviews are retrieved from the database, joined with member usernames, and displayed with star ratings and comments. This script serves as a central component for the product browsing experience, providing a clean, interactive, and informative interface for prospective buyers.

Shopping Cart Management (cart.php)

This script provides a complete interface for managing the shopping cart functionality of the e-commerce platform. It initializes the cart in the session if it doesn't already exist and supports three main actions: adding a product to the cart via a GET request, removing items via a GET parameter, and updating item quantities via a POST request. When a product is added, it's retrieved from the database and either appended to the cart or its quantity is incremented if it already exists. The cart is displayed in a structured HTML table, with individual rows for each product, showing name, price, quantity, total cost, and actions to update or remove the item. A confirmation message appears when a product is successfully added. Styling is applied for a clean and responsive layout, and a prominent "Buy" button links users to the checkout process via order_items.php. This script is central to the user's purchasing journey, enabling a smooth and interactive cart management experience.

Open Box Purchase Flow (openbox_purchase.php)

This script manages the purchase process for open box products and is accessible only to logged-in members. It begins with a session check, redirecting unauthorized users back to the login page with an alert. The product is retrieved from the database using the id provided via a GET parameter, and its open box price is calculated at 10% off the original. When the form is submitted via POST, the script validates the input fields (name, address) and ensures the user acknowledges a disclaimer about final sale conditions. If valid, the purchase is recorded in the orders table with an openbox status, and a confirmation email is sent to the user containing full order details. The email is sent either using the session-stored email or retrieved from the members table. The interface includes a styled form with responsive design, error handling, and dynamic messaging upon success or failure. This script provides a secure, user-friendly interface for discounted open box purchases, ensuring accountability through database logging and email confirmation.

Refurbished Product Purchase (refurbished_purchase.php)

This script facilitates the direct purchase of refurbished products by authenticated members. It begins with a session check to confirm the user is logged in, redirecting unauthorized visitors to the login page. The script retrieves product details using a GET parameter and calculates a refurbished price at 15% off the original. Upon form submission, it validates the user's name, shipping address, and confirmation of a mandatory disclaimer outlining the final sale nature of refurbished items. If valid, the order is inserted into the orders table with a refurbished status. A confirmation email is then sent to the user's email (retrieved from session or database), summarizing the purchase details and reiterating the no-return policy. The frontend includes a styled form with clear error and success messaging, along with a prominent disclaimer. Refurbished products bypass the standard cart flow due to their limited availability, making this script a specialized purchase endpoint tailored for one-off, discounted sales.

Order Placement & Confirmation (order_items.php)

This script processes and confirms product orders for logged-in members. It begins by checking if the user has confirmed their shipping location (Ontario or not), which determines the delivery time—1 day for Ontario residents, 3 days otherwise. If the user is not authenticated or their cart is empty, they are redirected appropriately. The total order cost is calculated from the items in the session-based cart, and a new record is inserted into the orders table with the user ID, order total, status (pending), and calculated delivery days. Each item in the cart is then recorded in the order_items table with its associated quantity and unit price. Once processed, the cart is cleared and an order confirmation page is rendered. This page displays a thank-you message, order ID, total cost, and estimated delivery date, along with navigation options to continue shopping or leave a product review. The script ensures a smooth and secure checkout experience, while also handling dynamic delivery logic based on location.

Order History Viewer (order_history.php)

This script displays a logged-in member's full order history in a clean, tabular format. It begins by verifying that the user is authenticated before retrieving their member_id and querying all related orders from the database. The query joins standard orders (with multiple items) from the order_items table and one-off purchases like refurbished items using product_id directly. For each order, it retrieves the product name, order date, status, and expected delivery date (calculated dynamically using the stored delivery_days). The frontend visually distinguishes pending and delivered orders based on current date and delivery expectation. Each order includes a "Leave Review" button linking to a review submission page. If no orders are found, a message is shown instead. The layout includes a styled table for clarity, and a help link is provided at the bottom for user guidance. This page offers members a centralized, accessible record of their past purchases, reinforcing transparency and ease of engagement post-purchase.

Product Review Submission (reviews.php)

This script allows authenticated members to submit reviews for products they've purchased. It begins by verifying that the user is logged in, then determines the product_id either directly via GET/POST or by resolving an order_id from the order_items table. On form submission (POST), the script validates the star rating (ensuring it's between 1 and 5) and saves the review in the reviews table along with the user's comment and timestamp. The interface includes a responsive star rating widget built with JavaScript for real-time visual feedback, a comment textarea, and clear success or error messaging. The product ID is stored in a hidden input to maintain context. With a clean, mobile-friendly layout, this page provides an intuitive way for customers to leave meaningful feedback on their purchases, enhancing trust and engagement across the platform.

Membership Portal & Account Dashboard (membership.php)

This script serves as the central hub for both prospective and logged-in members, dynamically adjusting its content based on authentication status. When a member is signed in, they are greeted with a logout option and exclusive access to premium features such as the **5080 Series video showcase**, a **Custom GPU Quote Builder**, and a **Financing Quote Calculator**—each offering tailored experiences and real-time quotes. The page also includes a dedicated **RMA (Return Merchandise Authorization)** section that guides users through the return process with step-by-step instructions. If a user is not logged in, the interface displays a styled login form alongside a persuasive membership benefits section that outlines perks like discounts, faster checkout, and personalized recommendations. Session-based alerts notify users of login errors. Clean layout, responsive design, and segmented content ensure both new and returning users experience a personalized, informative, and engaging membership experience.

Member Login Authentication (login_member.php)

This script securely handles member login requests submitted via a POST form. It begins by validating that both the username and password fields are filled. It then queries the members table to find a matching user and verifies the submitted password using password_verify(). If the credentials are valid and the account is marked as "enabled," the script logs the login event into the login_history table with the user's IP address, and stores essential member information in session variables (username, id, and login status). If the account is disabled or authentication fails, an appropriate error message is stored in the session and the user is redirected back to the membership.php page. With strong validation, status checks, and activity logging, this script ensures a secure, trackable, and user-friendly login process for members.

GPU Quote Customizer (gpu_quote_customizer.php)

This interactive script allows users to build a personalized GPU quote by selecting a base GPU model and optional add-ons. Upon loading, it queries the database for available products and populates a dropdown with GPU names, images, and prices. The interface then presents a series of checkboxes for popular add-ons such as extended warranties, professional installation, thermal upgrades, and shipping enhancements—all with associated costs. Real-time JavaScript updates the total estimated price based on user selections and displays it in a summary section. A call-to-action button labeled "Email Me

My Quote" initiates the final step (handled via external JavaScript) to generate and send the full quote. The form is designed for clarity and responsiveness, offering a seamless user experience that balances flexibility, customization, and visual simplicity. This page acts as a modern pre-sales tool, enhancing customer engagement by offering transparent, tailored pricing options.

Financing Quote Calculator (financing_quote.php)

This script provides an interactive interface that allows users to calculate estimated monthly payments for GPU purchases using financing. Upon page load, it retrieves a list of available GPU products from the database and displays them in a dropdown menu. Users can then select a financing term (12, 24, or 36 months) and enter a custom down payment percentage. With the help of JavaScript, the page dynamically calculates and displays the estimated monthly payment based on the selected GPU price, down payment, and term length. A button labeled “Email Me My Quote” is provided to allow users to receive a personalized financing breakdown via email (handled by external JavaScript). The layout is clean, responsive, and styled for clarity and ease of use. This calculator enhances the purchasing experience by offering flexible budgeting options and helping users plan their GPU investment in a transparent, user-friendly way.

RMA Ticket Submission (rma.php)

This script enables logged-in members to initiate a Return Merchandise Authorization (RMA) request for previously purchased products. It begins by verifying user authentication and retrieving the member’s order history from the database. The interface presents a list of eligible order items in a styled table with checkbox selectors. Once an item is selected, additional input fields become visible, allowing the user to specify the issue type, describe the problem, and optionally upload a supporting document. Upon form submission, the script processes the selected item IDs, captures form input, handles file uploads securely, and inserts the RMA request into the rma_requests table. Success or error feedback is displayed accordingly. The form supports multiple item selection and includes client-side JavaScript for dynamic field visibility. This script offers a streamlined and user-friendly experience for members needing post-purchase support, while ensuring backend consistency and traceability for support staff.

Frequently Asked Questions – RTX 5080 (faq.php)

This static FAQ page provides customers with clear, detailed answers to the most common questions surrounding the RTX 5080 GPU. Organized in a clean, card-based layout, the content covers technical specifications, architecture (Blackwell), gaming and productivity performance, pricing, availability, compatibility, power requirements, cooling, and unique features. It also includes comparisons with competing GPUs like the RTX 4080, 4090, and AMD alternatives, emphasizing the RTX 5080's value in terms of performance, energy efficiency, and DLSS 4 support. Each FAQ section is styled for readability and structured with headings and concise responses. At the bottom, a call-to-action invites users to contact support for further questions. This page is a valuable educational and support resource that helps potential buyers make informed decisions while reducing customer service inquiries.

Customer Inquiry Submission (contact.php)

This script enables logged-in members to submit support or product-related questions via a contact form. After confirming authentication, it presents a simple interface where users can enter a subject and their question. On form submission, the inputs are validated to ensure both fields are filled. Valid entries are inserted into the questions table along with the member's ID and a timestamp. A success or error message is displayed based on the result of the database operation. The form is cleanly styled and centered for accessibility, with a confirmation message shown upon successful submission. An instructional link is also included to guide users. This script offers a straightforward and secure way for members to reach out to support, while also providing backend traceability and easy ticket tracking.

User RMA & Q&A Dashboard (my_answers.php)

This script provides a personalized dashboard for logged-in members to review their **RMA (Return Merchandise Authorization)** tickets and **submitted questions**. After confirming authentication, the script queries the database for all RMA requests and Q&A records associated with the member's ID. RMA tickets are displayed in a responsive table that shows ticket details, order item IDs, issue descriptions, optional attached documents, submission timestamps, and admin responses if available. Questions and answers are presented in a card-style layout, showing the original subject, member-submitted question, creation date, and any admin replies. If no entries exist, helpful placeholder messages are displayed. A help link is provided for additional guidance. This dual-purpose page serves as a centralized self-service portal, enhancing transparency, support accessibility, and user engagement post-purchase.

Product Stock Visualization (stock.php)

This script provides a dynamic, chart-based overview of product inventory levels using js. It queries the database to join products with stock (defaulting to zero if no stock is recorded), cleans up product names by removing common substrings (e.g., "video card", "GeForce", "GDDR7"), and formats the data into labels and quantities. The frontend renders a responsive bar chart that switches between horizontal and vertical orientation depending on screen size. The chart dynamically adjusts axes, spacing, and label rotation for optimal readability across devices. This tool gives administrators and stakeholders a quick, intuitive snapshot of current stock levels in real time, supporting better inventory oversight and restocking decisions. A help link is included for user guidance.

Regional In-Store Pickup Availability Map (map.php)

This interactive page visually displays in-store pickup availability across Canadian provinces using an image map of Canada. Users can click on different regions to view whether local pickup is supported in that area, with real-time updates shown in a styled display box. JavaScript dynamically updates the message based on predefined regional availability and applies conditional styling (green for available, red for unavailable). The layout is responsive and includes the Image Map Resizer library to maintain map functionality on all screen sizes. Each <area> tag is tied to a corresponding region, making the map fully clickable and intuitive. This tool helps customers quickly understand regional service availability while providing a visually engaging experience and support resource for logistical planning.

Secure Member Logout with History Tracking (logout.php)

This logout script ensures a secure and complete sign-out process for logged-in members. If a user session is active, it logs the logout event into the login_history table along with the member ID and IP address to maintain accurate authentication tracking. The script then clears all session data using session_unset() and session_destroy() to securely terminate the session. Finally, it redirects the user back to the membership/login page to complete the logout flow.