|  |  |  |
| --- | --- | --- |
| RAZDA Co. | | |
| **Filename: [manage\_verticallist.html]** | | |
| **Summary***:*  The manage\_verticallists.html file provides a user interface for managing vertical lists within Razda Market. It includes sections for creating new lists and viewing existing ones. Each vertical list displays essential details such as List ID, Name, Description, and Created Date. Admins can add, view, and delete lists, facilitating effective organization and management of categorized items or lists. | | |
| ***Processes*** | | |
| * **Create New Vertical List** | **Provides a form for admins to enter details for a new vertical list, including name, description, and items (one item per line). Submitting the form sends a POST request.** | |
| * **View Existing Vertical Lists** | **Displays a table of existing vertical lists, including details like ID, name, description, and creation date, allowing admins to keep track of all created lists.** | |
| * **Delete Vertical List** | **Each list has a delete button that sends a POST request to remove the selected list. The delete option is styled as a form submission to follow best practices for destructive actions.** | |
| * **Confirmation for Deletion** | **Before deleting, an onsubmit confirmation prompt is shown to prevent accidental deletions, adding a layer of security for data management.** | |
| **Files it Gets Information From:** | | **Files it Sends too:** |
| * **CSS file (styles.css)** (if linked): Adds styling to the form, table, and buttons, ensuring a consistent and professional interface. | * **Create List Route (verticallist\_bp.manage\_verticallists)**: The form submission for creating a new vertical list sends a POST request to this route, which handles list creation in the backend. | |
| * **header.html** and **footer.html** (if included later): For consistency in layout and navigation. | * **Delete List Route (verticallist\_bp.delete\_verticallist)**: The delete button sends a POST request to remove a vertical list from the backend database, using the list ID to identify the target. | |
|  | * **View List Route (verticallist\_bp.view\_verticallist)**: The "View" link for each list navigates to a detailed view, allowing admins to see all items within the list. | |
| **Expected input into file:** | | **Expected output from file:** |
| * **verticallists**: A list of dictionaries or objects containing details for each vertical list, including fields like verticallist\_id, list\_name, description, and created\_at. This data is used to populate the table. * **Form Data**: User input from the "Create New Vertical List" form, such as list\_name, description, and items. | | * **Vertical List Management Interface**: Displays a form to create new lists and a table listing all current vertical lists with details and action buttons for viewing and deleting. * **List Management Actions**: Provides functionality for list creation and deletion, as well as navigation to detailed list views. |
| **Things that need to be taking place:** | | |
| |  | | --- | | **1. Data Loading: Ensure that verticallists is passed to the template, allowing the table to display all existing lists accurately.** |  |  | | --- | | **2. Create List Handling: Verify that the verticallist\_bp.manage\_verticallists route correctly processes POST requests, creating a new list based on the form inputs (e.g., list name, description, and items).** |  |  | | --- | | **3. Delete Functionality: Confirm that the delete\_verticallist route processes the POST request to delete a list, using the provided verticallist\_id to ensure accurate targeting.** |  |  | | --- | | **4. Confirmation for Deletion: Implement and test the confirmation dialog to prevent accidental deletions, adding a layer of safety.** |  |  | | --- | | **5. Error Handling: Provide error handling for cases where list creation or deletion encounters issues, with user-friendly messages guiding the admin on next steps.** |  |  | | --- | | **6. Data Persistence: Ensure all changes (creation and deletion) are properly saved in the backend and reflect immediately in the template upon refresh or navigation.** |  |  | | --- | | **7. Styling and Responsiveness: Confirm that the form and table are styled for readability and accessibility, with responsive adjustments for mobile viewing.** |  |  | | --- | | **8. Form Validation: Add server-side validation for form inputs to ensure data integrity (e.g., required list name, valid item entries), preventing invalid data from being submitted.** |  |  | | --- | |  | | | |
| Edit log (update each time you make changes to doc or file). | | |
| * Oliver Smith (Razda Admin) Nov 8, 2024: | | |