|  |  |  |
| --- | --- | --- |
| RAZDA Co. | | |
| **Filename: [product\_management.py]** | | |
| **Summary***:*  This file is a central component of the Razda Market backend, responsible for the comprehensive management of products in the system. It supports CRUD (Create, Read, Update, Delete) operations on products, ensuring administrators can manage the product catalog effectively. Additionally, it provides reporting and search functionalities, allowing for inventory adjustments and sales data generation. The product\_management.py file contributes significantly to Razda Market by ensuring real-time and efficient control over products, prices, and stock levels, which are essential for maintaining a high-quality customer experience. | | |
| ***Processes*** | | |
| * **Product Addition** | * **Allows the addition of new products to the database, including attributes like name, description, price, inventory count, category, and images.** | |
| * **Product Editing** | * **Enables updates to product details, such as price adjustments, category changes, inventory levels, and image updates, keeping the catalog current.** | |
| * **Product Removal** | * **Facilitates the deletion of products from the database when no longer available, ensuring obsolete items don’t appear in the catalog.** | |
| * **Product Listing** | * **Lists all available products in a structured format, displaying essential information for each product to provide administrators a complete view.** | |
| * **Product Search** | * **Searches for products based on keywords, making it easier for admins to locate specific products quickly.** | |
| * **Bulk Price Update** | * **Updates prices across a specified category by a percentage, allowing for efficient sales and promotions.** | |
| * **Bulk Inventory Update** | * **Adjusts inventory for products below a certain threshold, ensuring products with low stock can be replenished automatically.** | |
| * **Sales Report Generation** | * **Generates sales reports within a specified date range, providing valuable insights into product performance and revenue generation.** | |
| **Files it Gets Information From:** | | **Files it Sends too:** |
| * **Environment Variables** (.env): Contains database credentials (DB\_HOST, DB\_USER, DB\_PASSWORD, DB\_NAME). | | * **Logging File**: Logs all actions, warnings, and errors to product\_management.log for audit and debugging purposes. |
| * **Database (MySQL)**: Interacts with products, product\_categories, product\_images, and order\_items tables for product data management and reporting. | | * - **Console Output**: Provides feedback on actions, such as successful product additions, updates, listings, and report generation. |
|  | |  |
|  | |  |
|  | |  |
| **Expected input into file:** | | **Expected output from file:** |
| * **Environment Variables**: For database connection details (DB\_HOST, DB\_USER, DB\_PASSWORD, DB\_NAME). * ‘’’python * DB\_HOST, DB\_USER, DB\_PASSWORD, DB\_NAME * ‘’’ * User Input(CLI): CLI arguments for each product management action, such as adding, editing, or removing products. | | * **Logging Output**: Logs major actions and errors, such as new product creation, bulk updates, and sales report generation. * ‘’’python * logging.info("Added new product 'item\_name'") * ‘’’ * **Database Changes**: Executes SQL commands to modify tables in the database for product, inventory, and price updates. |
| **Things that need to be taking place:** | | |
| |  | | --- | | **● Database Connection: Establishes secure and reliable connection to MySQL for consistent access to product and category data.** |  |  | | --- | | **● Product Validation: Ensures each product has valid category assignments by verifying category existence in the database, preventing invalid entries.** |  |  | | --- | | **● Image Management: Adds and updates product images in a dedicated table, allowing multiple images per product for enhanced product presentation.** |  |  | | --- | | **● Input Sanitization: Cleans inputs to prevent SQL injection, XSS attacks, or other security vulnerabilities, preserving data integrity.** |  |  | | --- | | **● Logging and Auditing: Logs all actions, errors, and warnings, providing a traceable history of product-related actions for audit and troubleshooting.** |  |  | | --- | | **● Bulk Operations: Supports efficient price and inventory adjustments, essential for managing promotions and replenishing stock on a large scale.** |  |  | | --- | | **● Reporting Capabilities: Generates sales reports based on historical data, giving insights into product performance and helping guide inventory decisions.** |  |  | | --- | | **● Error Handling: Catches and logs database errors, preventing system crashes and alerting administrators to potential issues with detailed messages.** |  |  | | --- | |  | | | |
| Edit log (update each time you make changes to doc or file). | | |
| - Oliver Smith (Razda Admin) Nov 8, 2024: Established core product management functions, configured logging, and connected to the Razda Market database for real-time product updates. | | |
| Contribution to the Entire Project:  The **product\_management.py** file is a foundational part of Razda Market’s backend. It enables administrators to efficiently manage the product catalog and ensure real-time control over the inventory, prices, and product availability. By handling everything from single product edits to large-scale updates and sales reports, this file supports the platform’s dynamic nature and provides a seamless user experience for both administrators and end customers.  This functionality is essential to the Razda Market ecosystem because it integrates with other systems (such as **orders**, **alerts**, and **analytics** modules), allowing the platform to automatically adjust inventory based on sales, trigger alerts for low stock, and generate insights that influence restocking and pricing strategies. | | |