|  |  |  |
| --- | --- | --- |
| RAZDA Co. | | |
| **Filename: [dynamic\_pricing\_manager.py]** | | |
| **Summary***:*  This file dynamically adjusts product prices based on demand, competitor pricing, inventory levels, and time-based factors. It utilizes real-time data from external APIs and internal modules to calculate optimal pricing, ensuring competitiveness while maintaining profitability. The module interfaces with other files like pricing\_calculator.py, inventory\_status.py, and SquareSpace integration files for a comprehensive pricing strategy. | | |
| ***Processes*** | | |
| * **Competitor Price Fetching** | **fetch\_competitor\_price retrieves competitor pricing data through API calls to maintain competitive prices. If the fetch fails, an alert is generated, and a default action is taken.** | |
| * **Demand-Based Pricing Adjustment** | **calculate\_dynamic\_price uses recent sales data to adjust prices based on demand, increasing for high demand and adjusting for lower sales volumes.** | |
| * **Competitor Based Pricing Strategy** | **Adjusts prices based on competitor data, following the configured strategy (e.g., match, undercut, or premium).** | |
| * **Inventory Based Pricing Adjustment** | **Adjusts prices based on stock levels: low stock increases price, while excess stock results in discounts to move inventory.** | |
| * **Cost Based Markup Application** | **Calculates a minimum price based on cost, ensuring the final price maintains a minimum profit margin. Applies the SAFETY\_PRICE\_FLOOR to prevent prices from falling too low.** | |
| * **Time-Based Pricing Adjustment** | **Adjusts prices during peak hours (e.g., 6 PM to 9 PM) to capture higher traffic and increase revenue.** | |
| * **Update Product Price** | **update\_product\_price pushes the final price to the external system (e.g., SquareSpace) and logs or alerts for any failures.** | |
| * **Monitor and Adjust Prices** | **Continuously monitors and adjusts prices for all products, recalculating every PRICE\_UPDATE\_FREQUENCY. Adjusts prices only if a significant change is detected.** | |
| * **Alert for Errors or Failures** | **Sends alerts for critical issues (e.g., pricing update failures, API fetch issues) to ensure prompt administrative response.** | |
| **Files it Gets Information From:** | | **Files it Sends too:** |
| * **Environment Variables**: Loads settings for price margins, adjustment frequency, and API configurations | * **Logs Directory**: Writes operational logs to dynamic\_pricing\_manager.log for monitoring and debugging. | |
| * **inventory\_status.py**: Fetches stock levels and recent sales data to inform demand and inventory-based adjustments | * **SquareSpace API**: Updates adjusted prices through squarespace\_sync, ensuring prices on the marketplace are current. | |
| * **pricing\_calculator.py**: Uses cost data to set minimum pricing, ensuring profitability. | * **Alert Automation**: Sends alerts on errors or critical pricing issues. | |
| * **Competitor API**: Fetches competitor prices for price matching or adjustments |  | |
| **Expected input into file:** | | **Expected output from file:** |
| * **Product Data**: Product IDs and base prices for dynamic calculation. * **Inventory Data**: Stock levels for demand and inventory-based adjustments. * **Competitor Prices**: Prices from competitors via external API. | | * **Adjusted Prices**: Final adjusted price for each product, updated in both the local system and external platforms (e.g., SquareSpace) * **Alerts**: Alerts sent for pricing failures, competitor fetch errors, and critical issues. * **Logs**: Logs of each adjustment step, including demand, competitor, and inventory changes. |
| **Things that need to be taking place:** | | |
| |  | | --- | | **- Competitor API Access: Ensure continuous access to competitor APIs, and define fallback behavior if unavailable.** |  |  | | --- | | **- Pricing Consistency: Maintain consistency between internal system prices and those displayed on external platforms (e.g., SquareSpace), minimizing delays in synchronization.** |  |  | | --- | | **- Scheduled Monitoring: Implement robust scheduling for monitor\_and\_adjust\_prices, ensuring it runs reliably every set interval to keep prices competitive.** |  |  | | --- | | **- Error Handling & Alerts: Robust error handling for API and sync failures, with real-time alerts to administrators to address issues quickly.** |  |  | | --- | | **- Minimum Margin Enforcement: Guarantee minimum profit margins by using safeguards, such as SAFETY\_PRICE\_FLOOR and cost-based minimum prices, preventing unsustainable pricing adjustments.** |  |  | | --- | |  | | | |
| Edit log (update each time you make changes to doc or file). | | |
| * Oliver Smith (Razda Admin) Nov 8, 2024: | | |