|  |  |  |
| --- | --- | --- |
| RAZDA Co. | | |
| **Filename: [error\_handler.py]** | | |
| **Summary***:*  This file manages error handling, retry logic, and alerting for critical issues. It provides mechanisms for logging errors, applying retry logic with exponential backoff, sending email alerts to admins, and gracefully degrading with fallback functions. It integrates with Flask routes to automatically handle errors and return appropriate responses. | | |
| ***Processes*** | | |
| * **Error Logging and Caching** | **log\_error logs error messages with context, including extra information. It increments an error count and adds error details to ERROR\_LOG\_CACHE.** | |
| * **Email Alert System** | **The send\_alert function sends email alerts when errors exceed the threshold. Alerts use SMTP configuration from environment variables and include spam protection to limit frequency.** | |
| * **Retry Mechanism** | **handle\_with\_retry wraps function calls in a retry mechanism, retrying a specified number of times with exponential backoff if an error occurs.** | |
| * **Graceful Degradation with Fallbacks** | **graceful\_degradation is a decorator that wraps functions with a retry mechanism and fallback function, executed if retries fail.** | |
| * **Fetching and Resetting Error Logs** | **The fetch\_recent\_errors function retrieves recent errors from cache, and reset\_error\_count clears the cache and resets the error count.** | |
| * **Flask Error Handling Decorator** | **The handle\_errors decorator wraps Flask routes, logging any errors that occur and returning a JSON response with an error status.** | |
| **Files it Gets Information From:** | | **Files it Sends too:** |
| * **Environment Variables**: Loads SMTP settings, retry parameters, and error threshold | * **Logs Directory**: Logs errors to error\_handler.log, recording critical issues and admin notifications. | |
| * **Flask Blueprint**: Uses Blueprint for routing error log access and reset functions in Flask | * **Email Alerts**: Sends email alerts to administrators if error frequency exceeds threshold. | |
| * **Alert Modules (optional)**: Can be integrated with alerting modules to escalate certain errors. |  | |
| **Expected input into file:** | | **Expected output from file:** |
| * **Function Calls**: Wrapped functions receive retry logic and error handling through this file. * **SMTP Server**: Receives configuration details to enable email alerts. * **API Request Parameters**: HTTP request parameters to retrieve recent error logs and reset error count. | | * **Error Logs**: Logs for critical errors, retries, and alert messages stored in error\_handler.log. * **Alerts**: Sends email alerts for critical errors to the admin email configured in environment variables. * **JSON Responses**: API endpoints return JSON responses with error logs or success messages |
| **Things that need to be taking place:** | | |
| |  | | --- | |  | | **- Retry and Fallback Consistency: Ensure all error-prone functions that could impact functionality are wrapped with retries and fallbacks for reliability.** | |  |  | | --- | | **- Error Frequency Monitoring: Monitor error frequency and adjust ALERT\_THRESHOLD and retry parameters as needed to balance performance and alerting efficiency.** |  |  | | --- | | **- SMTP Server Reliability: Confirm SMTP configuration for alerting to prevent missed notifications during critical incidents.** |  |  | | --- | | **- Error Cache Management: Regularly manage or prune ERROR\_LOG\_CACHE to avoid excessive memory usage, especially if error frequency is high.** |  |  | | --- | | **- Testing and Monitoring: Regularly test error handling, alerting, and retry functions, including manual tests and integration with a testing suite if available.** | | | |
| Edit log (update each time you make changes to doc or file). | | |
| * Oliver Smith (Razda Admin) Nov 8, 2024: | | |