|  |  |  |
| --- | --- | --- |
| RAZDA Co. | | |
| **Filename: [user\_login.py]** | | |
| **Summary***:*  The user\_login.py script handles user login within Razda Market. It validates input, authenticates user credentials against stored database information, and applies rate limiting to secure against excessive login attempts. Key features include environment-based configuration, password hashing verification, logging, and error handling for a secure and user-friendly login experience. | | |
| ***Processes*** | | |
| * **Enviroment Setup and imports** | **Loads environment variables from .env, securing sensitive information such as database credentials. Imports essential modules for managing user input, database interaction, and security (e.g., mysql.connector, bcrypt, logging).** | |
| * **Database connection pooling** | **Configures a MySQL connection pool, enabling efficient database access while handling concurrent login requests. If a pool cannot be established, logs the error and terminates the program to prevent further errors.** | |
| * **Input Validation Functions** | **Functions validate and sanitize user inputs. is\_valid\_email checks email format, is\_valid\_password enforces password strength, and sanitize\_input removes potentially malicious characters, enhancing security and data integrity** | |
| * **User Authentication** | **Authenticates users by checking if the provided credentials match a database entry. Fetches user data by username/email, verifies the hashed password with bcrypt, and logs outcomes, such as successful logins or failed attempts, for traceability.** | |
| **Files it Gets Information From:** | | **Files it Sends too:** |
| * **.env file**: Contains database credentials and log file configurations. Environment variables are securely loaded, preventing exposure of sensitive information. | | * **Log file (app.log)**: Logs login attempts, including successful authentications, warnings for invalid logins, and database connection errors. Records details with timestamps, helping troubleshoot and monitor login activity. |
| **Expected input into file:** | | **Expected output from file:** |
| * **Username/Email and Password**: User-provided login information. **Username or Email** is sanitized, and **Password** is checked for format compliance before being used in authentication. | | * **Authentication Result**: If successful, authenticates the user and prints a welcome message. If unsuccessful, provides feedback on issues like incorrect passwords or invalid accounts, and logs events. |
| **Things that need to be taking place:** | | |
| |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  | | --- | | **1. Environment Variable Loading: Confirm all required environment variables are in .env, such as DB\_HOST, DB\_USER, DB\_PASSWORD, DB\_NAME, and LOG\_FILE. Ensure .env is stored securely to protect sensitive credentials.** |  |  | | --- | | **2. Logging Verification: Ensure app.log accurately captures all login attempts, including valid and invalid inputs, rate-limited attempts, and database connection issues, providing a full audit trail for login activity.** |  |  | | --- | | **3. Connection Pool Functionality: Verify that razda\_pool connection pooling works as expected, efficiently handling concurrent connections without causing delays or consuming excessive resources.** |  |  | | --- | | **4. Password Security: Confirm that bcrypt successfully hashes passwords and validates them against stored hashes, preventing unauthorized access through password verification.** |  |  | | --- | | **5. Rate Limiting: Test the RateLimiter class under typical and excessive login attempts to ensure accurate limiting of requests and prevention of brute-force login attempts.** |  |  | | --- | | **6. Error Handling: Check that error-handling messages provide users with feedback on issues like invalid credentials and excessive attempts, without revealing sensitive information.** |  |  | | --- | | **7. Template Integration (if applicable): Ensure that this script integrates seamlessly with any frontend templates used in the login process, correctly redirecting users upon successful authentication.** |  |  | | --- | | **8. Session Management: If this login script is integrated within a session-based application, ensure that session information (e.g., user ID) is securely stored and cleared as appropriate.** |  |  | | --- | | **9. Security Testing: Conduct security tests to ensure SQL injection protection, especially around sanitize\_input and database queries, preventing malicious user input from compromising the system.** |  |  | | --- | | **10. User Feedback and Accessibility: Confirm that error messages are accessible, guiding users through troubleshooting failed login attempts without exposing system vulnerabilities.** | | | | |
| Edit log (update each time you make changes to doc or file). | | |
| * Oliver Smith (Razda Admin) Nov 8, 2024: | | |