

TEST SCENARIO TS-03: ERROR HANDLING & CONFIGURATION

Project Name: Library Inventory Manager

Document ID: TS-03

Module: Reliability, Validation & Configuration

Author: Dominik Hoch

Date: 2.1.2026

1. TEST OBJECTIVE

To verify that the application is robust and reacts reasonably to invalid inputs, missing files, and configuration errors, without crashing (preventing "Internal Server Error" pages).

2. PREREQUISITES

- The application is running.
 - Access to the `config/db_config.json` file.
-

3. TEST STEPS EXECUTION

STEP 1: Input Validation (Invalid Type)

- **Action:** Navigate to "Add Book".
 - **Title:** Error Test
 - **Price:** Expensive (Type text instead of a number).
 - Click "Save Book".
- **Expected Result:** The application **does not crash**. A red warning message is displayed: *"Error: Price must be a number"*. The book is NOT saved to the database.

STEP 2: Import Validation (Missing File)

- **Action:** Navigate to "Import". Click the "Upload" button *without* selecting any file.
- **Expected Result:** A warning message *"No file selected"* is displayed.

STEP 3: Import Validation (Invalid File)

- **Action:** Create a dummy text file named `bad.txt`. Try to upload this file in the Import section.

- **Expected Result:** The application detects the issue (either file extension or JSON parsing error) and displays a "Import failed" error message.

STEP 4: Configuration Error (Database Connection)

- **Action:**
 1. Stop the application (CTRL+C in terminal).
 2. Open `config/db_config.json`.
 3. Change the "password" to an incorrect value (e.g., "WRONG_PASS").
 4. Save the file and start the application again (`python app.py`).
 5. Open the web browser and reload the page.
- **Expected Result:** The application does not show a raw Python stack trace. Instead, it displays a user-friendly error message indicating a **Database Connection Error** or **Access Denied**.

STEP 5: Recovery (Restoring Configuration)

- **Action:**
 1. Stop the application.
 2. Correct the password in `config/db_config.json`.
 3. Start the application.
 4. Reload the page.
- **Expected Result:** The application loads the Dashboard correctly again.

4. NOTES FOR TESTER

- In Step 4, the application is expected to handle the `mysql.connector.Error` exception gracefully.
- This scenario proves that the system configuration is externalized (not hardcoded) and the system is resilient to connection failures.

5. TEST RESULT

(To be filled by the Tester)

Status: [] **PASS** (Application handled errors gracefully) [] **FAIL** (Application crashed or showed raw code errors)

Tester

Name: _____ **Date:** _____