TEST SCENARIO TS-03: ERROR HANDLING & EDGE CASES

Project Name: AniTrack

Module: System Resilience & Validation

Author: Matyáš Prokop

Date: 5.1.2026

Test Objective

This scenario verifies the application's ability to handle incorrect configurations, database connectivity failures, invalid user inputs, and boundary conditions. The goal is to ensure the system provides user-friendly error messages and maintains data integrity under stress or improper use.

Prerequisites

Application is installed and environment is ready.

Physical access to `src/config/config.json` for modification.

Ability to start/stop the MySQL Server service.

Test Steps

1 Configuration & Connection Errors

- Invalid JSON: Edit `config.json` to break syntax (e.g., remove a comma). Verify that `python main.py` triggers: *"Invalid JSON in config file"*.
- Missing Config: Rename `config.json`. Verify the application launches using hardcoded default values.
- Authentication Failure: Set a `wrong_password` in config. Verify the browser displays: *"Database connection failed: Access denied"*.

- Service Interruption: Stop the MySQL service and attempt to navigate the site. Verify the error page: *"Cannot connect to MySQL server"*.

2 Input Validation (Forms)

- User Constraints: Attempt to save a user with an empty username, an invalid email format (e.g., `test@`), or a username shorter than 3 characters.
- Anime Constraints: Attempt to save anime with negative episode counts or a score outside the 0–10 range.
- Duplication: Attempt to register a username or email that already exists in the database.

3 CSV Import Failures

- File Validation: Attempt to import a `.txt` file instead of `.csv`.
- Content Integrity: Import a CSV with missing required columns or invalid data types (e.g., text in the `episodes_total` column).

4 Logic & Constraints

- Invalid Transfers: Attempt to transfer anime from a user who doesn't own it, or to a user who already has it in their watchlist.
- FK Violations: Attempt to delete a Genre that is currently assigned to one or more Anime. Verify the system blocks this to prevent orphaned records.

5 Edge Cases & 404s

- Special Characters: Create entries using quotes (`"Special"`), apostrophes (`user'test`), and Japanese characters (進撃の巨人).
- Boundary Values: Test scores exactly at `0` and `10`, and usernames exactly at 32 characters.
- Routing: Navigate to `http://localhost:5000/nonexistent`. Verify a custom 404 Page Not Found is displayed.

Success Criteria

#	Criterion	Expected Result	Status
1	**Config Resilience**	Syntax errors in JSON are caught during startup.	[ ]

2	**Connection Handling**	DB errors (wrong pass, server down) show user-friendly pages.	[ ]
3	**Field Validation**	Empty or malformed inputs are blocked with specific tooltips/labels.	[ ]
4	**Data Integrity**	Duplicate entries (Users/Genres) are prevented by the application.	[ ]
5	**CSV Robustness**	Invalid CSV files do not crash the server; records are skipped or errors shown.	[ ]
6	**Logic Safety**	Watchlist transfers only occur if both source and target criteria are met.	[ ]
7	**Clean Routing**	404 errors are handled by a dedicated view instead of a raw browser error.	[ ]
8	**UTF-8 Support**	Special and international characters are saved and rendered correctly.	[ ]

Test Data Summary

Category	Details
Configuration Files	config.json (syntax testing), config.json.bak (missing file testing)[1]
Invalid CSV Files	invalid_anime.csv (wrong headers), bad_data.csv (data type mismatch)
Validation Boundaries	Username: 3–32 chars, Score: 0.0–10.0, Rating: 0–10
Conflict Data	Existing User: validuser, Existing Genre: Action
Error Test URLs	http://localhost:5000/nonexistent (404 testing)[2222]
Database Constraints	Foreign Key (Genre-Anime link), Unique Constraints (Email/Username)[3333]