ISSUE 1: Loan rejection appears again for an already failed or rejected Loan

Test Scenario

system understanding:
Loan has terminal states (rejected and failed). Once reached, no further action should be allowed. Re-appearance indicates state machine violation or a duplicate trigger.

Positive scenarios:

Case1: Reject loan from Created to Rejected
Case 2: Reject loan from Under review to Rejected

Negative scenarios:

Case1:Attempt to reject a loan already in rejected list
Case2:Attempt to reject a loan already in failed list
Case3:API retry causing duplicate rejection
Case4:User refresh during rejection
Case5:Parallel rejection attempt by two users

Boundary / Edge Cases:

Case1:Network timeout after rejection
Case2:Partial rejection
Case3:Manual DB correction earlier

State-Based Scenarios:

Previous State	Action	Expected Outcome
Rejected	Reject again	Blocked
Failed	Reject again	Blocked

Disbursed	Reject	Blocked
Approved	Reject	Allowed

User Behavior Scenarios

Case1:click on “Reject”
Case2:Browser refresh after rejection
Case3:Same loan opened in two tabs
Case4:Backend retry from queue

2. Test Case Design:

Test Case: Prevent duplicate rejection of failed loan

Preconditions:
Loan status = failed
User has rejection permission

Steps:
Open loan details page
Click “Reject Loan”
Observe UI & API response
Refresh page

Expected Results:
UI disables reject action

API returns status code 409 No new rejection record created
Audit log unchanged

Post-Conditions:
Loan remains in failed status
No duplicate workflow triggered

3. Regression Strategy

Mandatory regression (Every Release):

Loan state transition matrix validation

Valid transitions:

CREATED → UNDER REVIEW
UNDER_REVIEW → APPROVED
UNDER REVIEW → REJECTED
APPROVED → DISBURSED

Invalid transitions (must be blocked):

REJECTED → REJECTED
FAILED → REJECTED
DISBURSED → REJECTED
DISBURSED → APPROVED

Verify:

Allowed transitions succeed
Disallowed transitions return error
State remains unchanged after invalid action

Terminal state immutability tests:

Once a loan reaches a terminal state, it must become read-only.

Terminal states:

REJECTED
FAILED
CLOSED

(No further business actions should modify the loan)

Scenario:

Loan status = REJECTED
User tries:
Reject again
Update loan amount
Trigger workflow

Expected behavior:

All actions blocked
UI disabled
API returns “Invalid operation on terminal state”

if not tested

Duplicate rejection records
Audit log corruption
Same loan processed multiple times

Retry & idempotency tests:

The system must handle duplicate requests safely, especially for:

Network retries:

Scenario:
User clicks “Reject”
Network times out

Same request is retried automatically

Correct behavior:

First request rejects loan
Second request is ignored
No duplicate rejection entry

UI double clicks:

Scenario:

User double-clicks “Reject” button

Correct behavior:

Button disabled after first click
Only one backend request processed

if not tested:

Duplicate audit logs
Multiple notifications sent
Downstream systems triggered twice

Automation:

API state transition tests
Duplicate request detection
Backend workflow idempotency

Manual:

UI double-click & refresh behavior
Parallel user scenarios

4. Validation & data integrity

UI status vs API response comparison
DB status vs workflow engine status
CRM vs Loan engine vs reporting consistency
Audit log immutability verification

5. Failure prevention thinking:

QA validates “happy rejection”
Terminal states assumed immutable
No retry or concurrency testing

Early signals:

Duplicate audit entries
Same loan in retry queues
Reappearing rejection events

QA–Dev Collaboration:
Explicit terminal state guards

Mandatory idempotency keys