**Test Design Review Checklist**

**1. Test Case Structure and Format**

**Test Case ID**: Each test case has a unique identifier for tracking and reference.

**Test Case Title**: The title is descriptive, concise, and indicates the purpose of the test.

**Test Case Description**: The description clearly states what is being tested and why.

**Pre-Conditions**: All pre-requisites or initial conditions are specified (e.g., user logged in, system state).

**Test Steps**: Each test step is clear, sequential, and easy to follow.

**Expected Results**: The expected result for each test step is well-defined, measurable, and accurate.

**Post-Conditions**: Defines the system state after test execution, if applicable.

**Test Data**: Test data is clearly identified and associated with the test case.

**2. Test Coverage and Traceability**

**Requirement Traceability**: Test cases are linked to specific requirements, user stories, or features to ensure full coverage.

**Test Case Completeness**: All functionality covered by the requirements is tested (positive, negative, boundary conditions).

**Boundary Conditions**: Test cases validate the boundary values (e.g., minimum, maximum, empty input).

**Negative Scenarios**: Test cases cover invalid inputs, error handling, and edge cases.

**Edge Cases**: Specific edge cases are tested to ensure robustness.

**Test Case Priority**: Test cases are prioritized based on business impact, risk, and criticality.

**3. Test Case Consistency and Clarity**

**Format Consistency**: The format and structure of test cases are consistent throughout (e.g., same fields, style).

**Naming Conventions**: Standardized naming conventions are followed for easy identification (e.g., Test Case ID format, test steps).

**Clarity and Simplicity**: Test case language is simple, clear, and unambiguous.

**Grammar and Spelling**: There are no grammar or spelling errors in the test case.

**Test Case Independence**: Test cases can be executed independently, without dependencies on other test cases unless explicitly required.

**4. Test Case Validity**

**Realistic Scenario**: The test case represents a real-world scenario, ensuring practical relevance.

**Correctness**: The test case accurately reflects the expected behaviour based on the application’s requirements and design.

**Reproducibility**: The test case can be executed multiple times with the same expected result.

**Test Data Accuracy**: The test data used in the test case is accurate, valid, and relevant to the scenario being tested.

**5. Test Execution Feasibility**

**Environment Setup**: Test cases specify any environment requirements (e.g., browser version, OS, database state).

**Test Tools**: The test case clearly identifies any tools, scripts, or automation frameworks required for execution.

**Automation Feasibility**: Test cases are assessable for automation, and automation suitability is documented if applicable.

**Execution Steps**: The steps are clear for execution, whether manual or automated.

**Error Handling**: Test cases specify how to handle failures or unexpected results during execution.

**Reusability**: Test cases are reusable in different environments or scenarios, where applicable.

**6. Test Case Clarity and Understand ability**

**Step-by-Step Detail**: Each test step is broken down with sufficient detail for understanding and execution.

**Expected Results**: The expected results are clearly stated for each test step and cover both success and failure conditions.

**Ambiguity**: Test cases are free from ambiguity or vague instructions that could lead to misinterpretation.

**Visual Aids**: Where necessary, screenshots, diagrams, or flowcharts are used to clarify complex test scenarios.

**7. Test Case Review and Collaboration**

**Stakeholder Review**: Test cases have been reviewed by relevant stakeholders (e.g., Business Analysts, Developers, Product Owners).

**Feedback Incorporation**: Reviewer comments and feedback are addressed and incorporated into the final test case.

**Peer Review**: Test cases have undergone peer review to identify issues early in the process.

**Approval**: Test cases have been approved by the Test Lead or other designated authority.

**8. Traceability and Dependencies**

**Linked to Requirements**: Every test case is traceable back to a specific requirement or user story to ensure completeness.

**Dependencies**: Any dependencies between test cases are documented (e.g., prerequisite tests).

**Test Case Versioning**: Test cases include version numbers to track changes and ensure consistency with the latest requirements.

**9. Special Considerations**

* **Compliance**: Test cases comply with organizational or project-specific guidelines and standards.
* **Risk-Based Testing**: High-risk areas are given higher priority in the test cases.
* **Non-Functional Testing**: Non-functional requirements like performance, security, usability, etc., are tested if applicable.
* **Complex Scenarios**: Complex test scenarios, like integration points, third-party interactions, etc., are covered.
* **Recovery and Rollback**: Any necessary test case steps for verifying system recovery, rollback, or error recovery are included.

**10. Documentation and Reporting**

* **Test Case Documentation**: All fields and sections in the test case template are fully filled out.
* **Clear Reporting**: Test case results will be easy to document, track, and report on during execution.
* **Traceability Matrix**: Test cases are included in a traceability matrix to ensure all requirements are covered.

**Additional Considerations**

**Test Data Coverage**: Ensure that all possible data combinations, edge cases, and negative inputs are covered.

**Test Case Prioritization**: High-priority test cases should be easily identifiable based on their impact on the system.

**Test Case Readability**: The test case should be easy to read and follow by anyone executing it, including new team members.

**Test Case Update**: Test cases are kept up to date with any changes in requirements, product features, or functionality.