QA Developer Assessment - Grading Rubric & Evaluation Guide

For Evaluators

This is a quick 45-minute technical skills assessment.

Overall Scoring Structure

Module	Time	Weight	Max Score	Passing Score
1. Programming Skills	20 min	50%	10	6
2. React Testing	15 min	50%	10	6
3. C Bonus	5 min	Bonus	+5	N/A
Total	**40 min**	**100%**	**20**	**12**

Score Interpretation

- **16-20 (80-100%)**: Excellent - Strong technical skills
- **12-15 (60-79%)**: Good - Solid foundation, hire
- **8-11 (40-59%)**: Fair - Consider with reservations
- **< 8 (< 40%)**: Not recommended

Module 1: Programming Skills (Max: 10 points)

Exercise 1: Test Plan Design (35% of module)

Score 5 (Expert):

- Complete test plan with all required sections
- 20+ well-organized test cases
- Clear traceability to requirements
- Comprehensive risk analysis
- Automation strategy defined
- Non-functional testing considerations
- Test data strategy with examples

Score 4 (Advanced):

- All sections present
- 15+ test cases covering major scenarios
- Good organization and prioritization
- Risk-based approach evident
- Cross-platform considerations

Score 3 (Intermediate):

- Basic test plan structure
- 10-15 test cases
- Main scenarios covered
- Some edge cases identified
- Clear scope definition

Score 2 (Beginner):

- Minimal structure
- 5-10 basic test cases
- Mostly happy path
- Limited organization

Score 1 (Novice):

- Incomplete or very basic list
- < 5 test cases
- No clear methodology

Exercise 2: Automation Strategy (30% of module)

Correct Answers Guide:

1. Login Flow: **Automated (A)** - stable, frequent, critical
2. Visual Design: **Manual (M)** - one-time, subjective
3. Payment Processing: **Both (B)** - automate core flow, manual exploratory
4. PDF Export: **Both (B)** - automate structure, manual visual check
5. Data Migration: **Automated (A)** - one-time but critical with large data
6. Exploratory Testing: **Manual (M)** - by definition
7. API Response Time: **Automated (A)** - continuous monitoring needed
8. Email Rendering: **Both (B)** - automate key clients, manual spot checks
9. Critical Bug Regression: **Both (B)** - automate for regression suite
10. Search Functionality: **Automated (A)** - stable, frequent, testable
11. Load Test: **Automated (A)** - requires tools
12. Onboarding Tutorial: **Automated (A)** - stable, critical UX, rare changes

Score 5: 12/12 correct with excellent justifications
Score 4: 9-11 correct with solid reasoning
Score 3: 6-8 correct with basic justification
Score 2: 3-5 correct, weak reasoning
Score 1: < 3 correct

Exercise 3: Root Cause Analysis (35% of module)

Bug #1: Silent Data Loss

- **Root Cause**: Cache updated but database write not confirmed (async queue pattern)

- **Key Evidence**: No UPDATE query in DB logs, only SELECTs

Bug #2: Intermittent Login Failures

- **Root Cause**: Database query timeout due to missing index + connection pool exhaustion
- **Key Evidence**: Slow query logs, 98/100 connections, table scan on 2.5M records

Bug #3: Duplicate Notifications

- **Root Cause**: Multiple device tokens per user (old tokens not cleaned up)
- **Key Evidence**: 3 tokens for one user, includes inactive tokens

Score 5: All 3 root causes correct + prevention + monitoring recommendations
Score 4: All 3 root causes correct + good prevention strategies
Score 3: 2/3 root causes correct with decent analysis
Score 2: 1/3 correct or superficial analysis
Score 1: No correct root causes identified

Module 2: React Testing (Max: 10 points)

Evaluation Criteria

Jest Testing (40%):

- **Score 5**: 20+ tests, comprehensive coverage, mocks, async handling
- **Score 4**: 15+ tests, good coverage, some advanced patterns
- **Score 3**: 10+ tests, covers main functions, basic patterns
- **Score 2**: 5+ tests, mostly happy paths
- **Score 1**: < 5 tests or poorly written

PyTest Testing (40%):

- **Score 5**: 30+ tests, fixtures, parametrize, mocking, markers
- **Score 4**: 20+ tests, uses fixtures and mocks appropriately
- **Score 3**: 10+ tests, basic fixtures, some mocking
- **Score 2**: 5+ tests, minimal fixtures
- **Score 1**: < 5 tests or no fixtures/mocking

Code Quality (20%):

- Follows testing best practices (AAA pattern, DRY, clear names)
- Proper use of framework features
- Tests are maintainable and readable

Overall Module Score

Average the Jest and PyTest scores, then adjust for code quality:

- Add 1 point for exceptional code quality
- Subtract 1 point for poor code organization

Module 3: C Bonus (Max: 5 bonus points)

Exercise 1: Bug Hunt (30% of module)

Intentional Bugs (9 total):

1. localStorage key typo ('todo' vs 'todos')
2. Missing useEffect dependency array
3. No validation for empty/whitespace todos
4. Date.now() ID collision risk
5. Edit allows empty todos
6. completeAll doesn't toggle
7. Active filter logic is backwards
8. AddTodo doesn't trim whitespace
9. Edit doesn't prevent default on Enter (minor)

Score 5: Finds 7-9 bugs with excellent documentation
Score 4: Finds 5-6 bugs with good repro steps
Score 3: Finds 3-4 bugs with basic documentation
Score 2: Finds 1-2 bugs
Score 1: Finds 0-1 bugs

Exercise 2: Component Testing (50% of module)

Minimum Expected Tests:

- TodoItem: 8+ tests (render, toggle, edit, delete, keyboard events)
- TodoList: 3+ tests (render, empty state, props)
- AddTodo: 5+ tests (input, submit, clear, validation)
- TodoFilters: 3+ tests (render buttons, active state, click handlers)
- BulkActions: 2+ tests (render, conditional display)

Score 5: 25+ comprehensive tests, excellent RTL practices
Score 4: 20+ tests, good coverage, proper queries
Score 3: 15+ tests, covers main functionality
Score 2: 10+ tests, basic coverage
Score 1: < 10 tests

Exercise 3: Debugging (20% of module)

Score 5: Fixes all tests with explanations
Score 4: Fixes 4-5 tests
Score 3: Fixes 3 tests
Score 2: Fixes 1-2 tests
Score 1: Fixes 0-1 tests

Criteria

Score 5: Comprehensive tests (30+), all bugs fixed, memory analysis complete
Score 4: Good test coverage (20+), most bugs fixed
Score 3: Basic tests (10+), some bugs found
Score 2: Minimal effort
Score 1: Incomplete

Candidate Profile Interpretation

Score 16-20: Senior/Expert Level

- Ready for complex QA automation projects
- Can lead testing initiatives
- Strong technical foundation
- Mentor potential

Score 12-15: Mid-Level

- Solid QA automation skills
- Can work independently on most tasks
- Good hire for QA developer role
- Growth potential

Score 8-11: Junior/Entry-Level

- Basic understanding but needs development
- Better suited for junior role
- Requires mentorship
- Consider for training program

Score < 8: Not Recommended

- Significant skill gaps
- May not be ready for QA developer role
- Consider other positions

Red Flags

Watch for these concerning patterns:

- Copy-pasted code without understanding
- No edge case testing
- Poor command-line skills for a QA role
- Unable to debug failing tests
- No documentation or unclear explanations
- Gives up easily (many incomplete sections)

Green Flags

Positive indicators:

- Goes beyond requirements
- Documents assumptions clearly
- Shows debugging process
- Writes maintainable test code
- Demonstrates curiosity
- Completes bonus module
- Clean, organized submissions

Time Guidelines

Expected completion times by skill level:

Senior (35-40 minutes):

- Completes both modules efficiently
- High quality documentation
- May complete bonus

Mid (40-45 minutes):

- Completes required modules
- Good quality work
- May attempt bonus

Junior (45-50 minutes):

- Struggles with some modules
- Basic completion
- Unlikely to attempt bonus

Red Flag: > 50 minutes with minimal completion

Final Recommendation Matrix

Score	Time	Recommendation
16-20	35-40min	**Strong Hire** - Senior level
12-15	35-40min	**Hire** - Mid level
12-15	40-45min	**Hire** - Junior/Mid level
8-11	35-40min	**Maybe** - Needs interview
8-11	> 45min	**Maybe** - Junior only
< 8	Any	**No Hire**

Evaluator: _____
Date: _____
Candidate: _____
Overall Score: _____ / 20 (+ ___ bonus)
Recommendation: _____