FINAL DOCUMENTATION
AutomationExercise Website

Project Group: G1_DEPI
Project Domain: Software Testing
Submitted To: DEPI/AMIT

TEAM MEMBERS:

1. **Belal Mohammed Taher - Tester**
2. **Mohammed Abedel Hamed - Tester**
3. **Mohammed abo El Yazeed - Tester**
4. **Ahmed Salah Qatar - Tester**
5. **Marian Ghaly - Tester**

Submission Date: 11/16/2025

TABLE OF CONTENTS

1.0 TEAM ROLES & RESPONSIBILITIES

MARIAN GHALY - Tester

- **API Testing: Login API validation & authentication flows**
- **Manual Testing: Cart operations, checkout process, payment validation**
- **Database Testing: Referential integrity & cross-table relationship validation**

AHMED SALAH QATAR - Tester

- **API Testing: User Account CRUD operations**
- **Manual Testing: Contact Us forms & user interface validation**
- **Database Testing: Settings configuration & module management**

MOHAMMED ABEDEL HAMED - Tester

- **API Testing: Product listing & catalog management**
- **Manual Testing: Product search, filtering & browsing functionality**
- **Database Testing: Order management & transaction data integrity**

BELAL MOHAMMED TAHER - Tester

- **API Testing: Brand listing & management operations**
- **Manual Testing: Home page navigation & user experience**
- **Database Testing: Customer data & admin user security**

MOHAMMED ABO EL YAZEED - Tester

- **API Testing: Search functionality & product discovery**
- **Manual Testing: User authentication & account management**
- **Database Testing: Products & categories data integrity**

2.0 EXECUTIVE SUMMARY

This comprehensive testing project involved systematic validation of the AutomationExercise e-commerce platform across UI, API, and Database layers. Our team executed 140+ manual test cases, validated 15+ API endpoints, and performed extensive database integrity testing using SQL queries on OpenCart tables.

Key database testing revealed critical data integrity issues including orphaned product-category relationships, incomplete cascading deletions, and rounding inconsistencies in financial calculations. Manual testing identified 27 defects with security vulnerabilities in payment processing being the most critical findings.

Key Achievements:

- **140+ manual test cases executed with 95% requirements coverage**
- **27 defects identified including 4 critical security vulnerabilities**
- **Comprehensive database testing across 15+ OpenCart tables**

• 47 orphaned database records identified and documented
• Multi-layered testing strategy (UI, API, Database) implemented

3.0 PROJECT OVERVIEW

The AutomationExercise website is a full-featured e-commerce platform built on OpenCart. Project objective was to deliver a robust testing framework validating front-end user experience, API functionality, and back-end data integrity while establishing automated regression tests.

In-Scope Features:
• UI Layer: User registration, login, product browsing, cart operations, checkout
• API Layer: REST endpoints for authentication, user management, products, orders
• Database Layer: OpenCart table validation (oc_product, oc_order, oc_customer, etc.)
• Data Integrity: Referential integrity, orphan record detection, business logic validation

Database Testing Scope:

• 15+ core OpenCart tables validated through direct SQL queries
• Referential integrity testing across product-category-order-customer relationships
• Orphan record detection and data consistency validation
• Business logic verification at database level

4.0 TEST STRATEGY & PLAN

TEST OBJECTIVES:

• Verify UI flows operate correctly for positive and negative scenarios
• Validate API endpoints with correct status codes and response payloads
• Ensure database integrity through direct SQL validation queries
• Identify orphan records and data consistency issues
• Develop automated tests for regression coverage

DATABASE TESTING APPROACH:

- **Direct SQL query execution in MySQL Workbench for data validation**
- **Referential integrity verification through JOIN operations**
- **Orphan record detection using LEFT JOIN with NULL checks**
- **Business logic validation comparing UI actions with database changes**
- **Data consistency testing across related tables**

5.0 TEAM CONTRIBUTIONS & TEST ARTIFACTS

5.1 Marian Ghaly - Authentication & Checkout Specialist

API Testing:

- **Validated Login API responses for valid/invalid credentials**
- **Verified status codes (200, 401, 403) and error message accuracy**

Manual Testing:

- **Executed 54 cart and checkout test cases**
- **Identified 8 critical security defects in payment processing**

• Verified complete order workflow from cart to confirmation

Database Testing:
• Ran integrity query: SELECT ptd.product_id FROM oc_product_to_category ptd LEFT JOIN oc_product p ON ptd.product_id = p.product_id WHERE p.product_id IS NULL
• Verified customer-address relationships across oc_customer and oc_address
• Tested user deletion cascade behavior in admin tables
• Finding: Database constraints effectively prevent orphan records, but application-level deletion logic has gaps

5.2 Ahmed Salah Qatar - User Management & Configuration
API Testing:
• Tested User Account API CRUD operations (Create, Read, Update, Delete)
• Validated user profile updates and data persistence

Manual Testing:
• Executed 17 Contact Us form test cases

• Identified form validation issues and submission problems

Database Testing:
• Modified store currency from USD to EUR, verified value field in oc_setting
• Enabled Banner Module, confirmed status changed from 0 to 1 in oc_extension
• Configured Featured Products module, validated position settings in oc_module
• Finding: Disabled modules retain configuration data in oc_setting rather than being removed

5.3 Mohammed Abedel Hamed - Order Management & Products
API Testing:
• Validated Product Listing API GET/POST operations
• Tested product search and filtering endpoints

Manual Testing:
• Executed 24 product page test cases
• Identified search and filter functionality issues

Database Testing:

• Placed order #1052, verified complete trail in oc_order, oc_order_product, oc_order_total
• Modified order quantity from 2 to 3 units, confirmed updates in oc_order_product
• Cancelled order #1048, validated status change in oc_order_history
• Finding: Order modifications don't always trigger oc_order_total recalculations for custom discounts

5.4 Belal Mohammed Taher - Customer Data & Security

API Testing:
• Tested Brand Listing API GET/PUT operations
• Validated brand management functionality

Manual Testing:
• Executed 48 home page navigation test cases
• Identified category and brand filter display issues

Database Testing:
• Registered customer 'test_user_15', validated customer_id generation in oc_customer
• Added shipping address, verified address_id linkage in oc_address

• Created admin user 'qa_tester', confirmed permissions in oc_user_group
• Finding: Customer deletion removes oc_customer record but oc_address records remain, creating potential data orphans

5.5 Mohammed abo El Yazeed - Authentication & Catalog

API Testing:
• Validated Search Product API functionality
• Tested product discovery and search endpoints

Manual Testing:
• Executed 25+ signup/login test cases
• Identified password validation and email format issues

Database Testing:
• Created product 'Winter Jacket', verified auto-increment product_id in oc_product
• Assigned product to categories, confirmed relationships in oc_product_to_category
• Deleted a category, found 12 orphaned records in oc_product_to_category

• Finding: Category deletion doesn't automatically remove product-category relationships, requires manual cleanup

6.0 DATABASE TESTING RESULTS

DATABASE TESTING METHODOLOGY:
Our database testing approach involved direct SQL query execution in MySQL Workbench against the OpenCart database. We focused on data integrity validation, referential integrity verification, orphan record detection, and business logic consistency across 15+ core tables.

KEY DATABASE FINDINGS:
• 47 orphaned product-category relationships identified in oc_product_to_category
• Customer deletion leaves address records in oc_address table (data orphans)
• Order total calculations show $0.01 discrepancies due to rounding methods
• Category deletion doesn't cascade to product-category relationships
• Disabled modules retain configuration data in oc_setting table

DATABASE TEST COVERAGE:

- **Tables Validated: 15+ OpenCart core tables**
- **SQL Queries Executed: 25+ validation queries**
- **Data Relationships Tested: 8+ key relationships**
- **Orphan Records Identified: 47 across multiple tables**

7.0 MANUAL TESTING RESULTS

MANUAL TESTING EXECUTION SUMMARY:

Our team executed comprehensive manual testing across all functional areas of the AutomationExercise website, focusing on user workflows, UI validation, and integration testing between frontend and backend systems.

TEST CASE DISTRIBUTION:

- **Authentication & User Management: 25+ test cases *(Mohammed abo El Yazeed)***
- **Home Page & Navigation: 48 test cases *(Belal Mohammed Taher)***
- **Product Catalog & Search: 24 test cases (*Mohammed Abedel Hamed)***

• Cart & Checkout: 54 test cases ***(Marian Ghaly)***
• Contact Forms & UI: 17 test cases ***(Ahmed Salah Qatar)***

KEY MANUAL TESTING FINDINGS:
• Payment system accepts invalid card numbers and expiration dates
• Input validation missing for email formats and password strength
• Category and brand filters display incorrect products
• Form submissions work with missing required fields
• Security vulnerabilities in session management and data handling

8.0 DEFECT ANALYSIS & QUALITY METRICS

Total defects identified: 27 across all testing phases

Defect Severity Distribution:
• Critical: 4 defects (Security vulnerabilities in payment processing)

• High: 8 defects (Data integrity issues, functional failures)
• Medium: 10 defects (UI/UX issues, validation problems)
• Low: 5 defects (Cosmetic issues, minor enhancements)

Quality Metrics:
• Test Case Pass Rate: 87%
• Defect Detection Rate: 4.2 defects per test cycle
• Requirements Coverage: 95%
• Test Execution Completion: 100%

9.0 TOOLS & TECHNOLOGIES STACK

Testing Tools:
• Selenium WebDriver - Web automation testing
• Postman - API testing and validation
• MySQL Workbench - Database testing and SQL queries
• JUnit/TestNG - Test framework and assertions
• Maven - Build automation and dependency management

Programming Languages:
• Java - Primary automation language
• SQL - Database querying and validation

• JavaScript - Frontend validation scripts

Project Management:
• JIRA - Defect tracking and project management
• Git - Version control and collaboration

10.0 TESTING FRAMEWORK ARCHITECTURE

Our testing framework follows a hybrid approach combining keyword-driven and data-driven methodologies with a modular architecture that supports both UI and API testing layers.

Framework Components:
• Test Base Layer - Configuration and setup management
• Page Object Model - UI element abstraction and encapsulation
• API Client Layer - REST endpoint management and validation
• Database Utility - SQL query execution and data validation
• Reporting Module - Test results and metrics generation

Key Features:
• Cross-browser compatibility testing support
• Parallel test execution capability
• Data-driven testing with Excel/JSON inputs
• Comprehensive logging and screenshot capture

11.0 CONCLUSION & LESSONS LEARNED

The comprehensive testing of the AutomationExercise website successfully identified critical issues across UI, API, and database layers. The project demonstrated the importance of multi-layered testing approach in ensuring software quality and reliability.

Key Successes:
• Successfully executed 140+ test cases with 95% requirements coverage
• Identified and documented 27 defects including critical security issues
• Established robust database testing methodology using SQL queries
• Developed reusable automation framework for future regression testing

Lessons Learned:

- **Database testing is crucial for identifying data integrity issues not visible in UI**
- **Security testing should be integrated throughout the testing lifecycle**
- **Team collaboration and knowledge sharing significantly improve test coverage**
- **Comprehensive documentation ensures test reproducibility and maintenance**

12.0 APPENDICES

Appendix A: Test Case TemplatesStandard test case templates used for manual and automated testing including test steps, expected results, and actual results.

Appendix B: SQL Query Repository
Collection of all SQL queries used for database testing including orphan record detection, data integrity validation, and business logic verification.

Appendix C: Defect Reports

Detailed defect reports including steps to reproduce, severity, priority, and resolution status for all 27 identified defects.

Appendix D: API Test Collections

Postman collection exports containing all API test scenarios, request payloads, and response validations.

ACKNOWLEDGMENTS

We would like to express our sincere appreciation to our instructors and DEPI/AMIT for their guidance and support throughout this testing project. Their expertise and mentorship have been invaluable in helping us develop our software testing skills and complete this comprehensive documentation successfully.