




**Md Abdullah Al Mahmud Pias | QA Engineer**

 [abdullahpias09@gmail.com](mailto:abdullahpias09@gmail.com) |  [almahmudpias.netlify.app](http://almahmudpias.netlify.app) |  [github.com/almahmudpias](https://github.com/almahmudpias)

Dhaka, Bangladesh

Version & Date: 0.1 - 25-10-2025

## Project Name E-Commerce Web Platform – QA Validation Project

### 1.Introduction:

The E-Commerce Web Platform is a full-stack application that allows users to browse products, add them to the cart, and complete purchases. This test plan defines the strategy, scope, and deliverables for manual and automated testing to ensure the platform functions correctly and meets user expectations.

### 2.Objective

- Validate core functionality (login, product browsing, cart, checkout).
- Identify and document functional, UI, and API defects.
- Perform regression, smoke, and integration testing.
- Demonstrate professional QA process including bug reporting, test documentation, and automation.

### 3.Test Items

The testing scope includes:

- Frontend components: React-based UI pages (Login, Product Listing, Cart, Checkout, User Account).
- Backend APIs: Node.js REST endpoints (products, orders, users).
- Database: MySQL/MongoDB (product and order management).
- Integration between frontend and backend.
- Cross-browser and responsive behavior (desktop Chrome, mobile view simulation).

### 4. Features to be Tested

Feature	Test Type	Description
Login/Register	Functional	Validate valid/invalid credentials, password reset.
Product Search	Functional/UI	Validate search results, filters, sorting, and responsive display.
Cart Management	Functional/UI	Add/remove items, quantity update, total price calculation.
Checkout	Functional/API	Validate order submission, payment simulation, API responses.

User Account	Functional/UI	Profile update, order history display.
API Endpoints	API/Integration	Validate GET, POST, PUT, DELETE for products and orders.
UI Responsiveness	UI/Compatibility	Verify layout, alignment, and usability across screen sizes.

## 4. Features Not to be Tested

- Third-party payment gateway integration (sandbox only).
- Email notifications (mocked).
- Admin panel features (out of scope).

## 5. Approach

### Testing Methodology:

- Manual Testing: Functional, UI, regression, smoke tests.
- Automation Testing: Selenium with Python for core workflows (Login → Search → Cart → Checkout).
- API Testing: Postman to validate REST endpoints.
- Bug Tracking: Jira / Excel sheet for defect reporting with severity and screenshots.

### Testing Levels:

- Unit: Limited verification of backend API responses.
- Integration: Verify data flow between frontend and backend.
- System: Full end-to-end user flow testing.
- Regression: Re-run automated and manual tests after fixes.

## 6. Test Environment

- **OS:** Windows 10
- **Browser:** Chrome (latest stable), Firefox (optional)
- **Backend:** Node.js (local/deployed)
- **Database:** MySQL/MongoDB
- **Tools:** Selenium, pytest, Postman, Jira, Chrome DevTools, Excel
- **CI/CD:** GitHub Actions (optional for automation test runs)

## 7. Entry Criteria

- Application deployed locally or on staging environment.
- Test accounts created (valid/invalid users).
- Required tools installed and configured (Selenium, Postman, ChromeDriver).
- Test data prepared for products, cart, and checkout scenarios.

## 8. Exit Criteria

- All critical and major defects logged and verified.

- Smoke test flows (login, search, add to cart, checkout) pass successfully.
- Test reports (manual & automation) completed and saved.
- QA Summary Report prepared with defect statistics and recommendations.

## 9. Test Deliverables

- Test Plan Document (this document).
- Test Cases (Excel/Markdown, 40–50 test cases covering functional, UI, API, and regression).
- Bug Reports (Jira/Excel with screenshots).
- Automation Scripts (Selenium Python scripts for 8–12 core flows).
- Test Execution Reports (HTML / PDF).
- QA Summary Report (PDF with charts and recommendations).

## 10. Test Schedule

Phase	Tasks	Duration
Planing	Create Test Plan & Test Cases	1–2 days
Execution	Manual Test Execution + Bug Logging	3–4 days
Automotion	Develop Selenium Tests & Run	2-3 days
Reporting	Prepare QA Summary & Reports	1 days

## 11. Responsibilities

Role	Name	Responsibilities
QA Engineer	Md Abdullah Al Mahmud Pias	Create test plan, write test cases, execute tests, log bugs, automate scripts, prepare reports
Developer	Md Abdullah Al Mahmud Pias	Fix reported defects (simulated or collaborator)

## 12.Risk & Mitigation

Risk	Mitigation
Backend API unstable	Use Postman to validate responses separately
Test environment differences	Test on local staging, document environment
Incomplete test data	Prepare sample users/products before testing

Approval By  
Md Abdullah Al Mahmud Pias