***Project*: *Amazon.eg Web Application***

***Test Team*: *Ahmed Ismail***
***Ahmed Saad Fayed***
***Ahmed Mohamed Abd El-Fattah***
***Mohmed Mahmoud Ismail***

# 1. Introduction

## 1.1 Purpose:

- To outline the testing strategy and approach for the Amazon.eg web application.

## 1.2 Scope:

- **Functional testing** of the Amazon.eg web application

### ❖ User Account Management

☑ This module allows users to register, log in, and manage their profiles.

### ✓ Account Creation

☑ Users can create an account by providing:

- ✶ Full Name
- ✶ Email Address (must be unique)
- ✶ Password (must meet complexity requirements)
- ✶ Optional: Phone Number, Address

☑ System Validations:

- ✶ Email format validation
- ✶ Password strength enforcement
- ✶ Duplicate email check
- ✶ CAPTCHA verification to prevent bot registrations.

### ✓ Login/Logout

☑ Users can log in using:

- ✶ Email + Password combination
- ✶ login (Google, Facebook) [if implemented]

☑ System Validations:

✶ Correct email and password verification

✶ Account lockout after multiple failed attempts

✶ Secure session management (cookies, JWT, session timeout)

### ✓ Profile Management

☑ Users can update:

✶ Name, Phone, Address

✶ Email (requires re-verification)

✶ Password (requires current password for security)

---

## ❖ Product Browsing and Search

This module enables users to explore available products.

### ✓ Product Catalog

☑ Features:

✶ Categorized display of products

✶ Pagination for large product listings

✶ Sorting options (Price, Popularity, New Arrivals)

### ✓ Product Search

☑ Search Capabilities:

✶ Keyword search

✶ Filters (Category, Price Range, Brand, Ratings)

✶ Auto-suggestions for search queries

### ✓ Product Details

☑ Users can view:

✶ Product name, price, images, and description

✶ Stock availability

✶ Customer reviews and ratings

---

## ❖ Shopping Cart

✶ This module allows users to manage selected items before purchasing.

### ✓ Add to Cart

☑ Users can add a product to their shopping cart.

☑ System Validations:

✶ Check stock availability before adding.

✶ Prevent adding duplicate items (or update quantity)

### ✓ Remove from Cart

☑ Users can remove unwanted items from the cart.

### ✓ Update Quantity

☑ Users can change the quantity of items in the cart.

☑ System Validations:

✶ Cannot exceed stock availability.

✶ Minimum quantity should be 1.

### ✓ View Cart

☑ Users can see:

✶ List of added products

✶ Subtotal, discounts, and total amount

✶ Estimated shipping cost.

---

## ❖ Checkout Process

✶ Ensures a smooth purchasing experience for users.

### ✓ Shipping Address

☑ Users can:

✶ Enter a new shipping address

✶ Select a saved address

### ✓ Payment Method

☑ Available Payment Options:

✶ Credit/Debit Cards

✶ PayPal

✶ Cash on Delivery (optional)

☑ System Validations:

- ✶ Secure card transactions with 3D Secure authentication

✓ Order Confirmation

☑ Before finalizing, users can:

- ✶ Review order summary
- ✶ Apply discount codes or coupons

✓ Order Tracking

☑ Users can track their order status:

- ✶ Pending → Processing → Shipped → Delivered.

---

## ❖ Order History & Tracking

- ✶ Users can view past purchases.

✓ Order History

☑ Features:

- ✶ List of past orders with details
- ✶ Invoice download option.

---

- ✶ **Contacting support** as a shopper
- ✶ **Answering a support inquiry** as an admin
- ✶ **As a shopper:** shipment tracking
- ✶ **As an admin**: fulfilling an order.
- ✶ **Reviews:** making a review or browsing the reviews as a shopper
- ✶ **Moderating reviews** as an admin
- ✶ **As an admin: validating in-stock/out-of-stock.**

- **Non-Functional testing** of the Amazon.eg web application

## ❖ Performance Requirements

- ✶ These define how efficiently the system should handle requests and process data.

☑ System Response Time:

- ✶ Pages should load within 2 seconds under normal conditions.
- ✶ Checkout process should complete within 5 seconds after payment submission.

☑ Concurrent Users Handling:

- ✶ System should support at least 1000 concurrent users.
- ✶ Database should handle 100,000+ product listings efficiently.

## ❖ Scalability:

- ✶ System should scale to accommodate increasing users and products.
- ✶ Support cloud-based auto-scaling when traffic spikes occur.

☑ Load Testing:

- ✶ Conduct stress testing to simulate peak traffic (e.g., Black Friday sales).

## ❖ Security Requirements

- ✶ **Ensures the protection of user data, transactions, and system integrity.**

☑ Authentication & Authorization:

- ✶ Enforce strong password policies (8+ characters, mix of uppercase, lowercase, numbers, symbols).
- ✶ Implement **multi-factor authentication** (MFA) for high-risk actions.
- ✶ Use **role-based access control (**RBAC) for admin vs. user privileges.

☑ Data Protection:

- ✶ Encrypt sensitive data (passwords, payment details) using AES-256.
- ✶ Use HTTPS with TLS 1.2+ to encrypt all network communications.
- ✶ Secure cookies to prevent cross-site scripting (XSS) attacks.

☑ Payment Security:

- ✶ Comply with PCI-DSS standards for secure payment transactions.
- ✶ Implement tokenization to avoid storing credit card details.

☑ Session Management:

- ✶ Auto-log out inactive users after 15 minutes of inactivity.
- ✶ Use secure session tokens (JWT) to prevent session hijacking.

☑ Vulnerability Protection:

- ✶ Perform regular penetration testing to detect security flaws.
- ✶ Implement firewall and intrusion detection systems (IDS/IPS).

## ❖ Usability Requirements

- ✶ Ensures a smooth and intuitive user experience.

☑ User-Friendly Interface:

- ✶ Provide clean, simple navigation with a responsive UI.
- ✶ Use standardized UI elements (buttons, dropdowns, forms).
- ✶ Offer dark mode/light mode for better accessibility.

☑ Mobile Responsiveness:

- ✶ Ensure the system is fully functional on mobile, tablet, and desktop.
- ✶ Support all major screen sizes (from 360px width to large desktops).

☑ Accessibility Compliance:

- ✶ Follow WCAG 2.1 guidelines for users with disabilities.
- ✶ Implement keyboard navigation and screen reader support.

☑ Error Handling & Feedback:

- ✶ Display clear error messages (e.g., "Invalid email format" instead of generic "Error!").
- ✶ Provide real-time form validation (e.g., invalid email format warnings).

## ❖ Reliability & Availability

- ✶ Ensures system stability and uptime.

☑ Uptime Guarantee:

- ✶ System must have **99.9% availability** with minimal downtime.

- ✶ Implement automatic failover to a backup server in case of crashes.

☑ Database Reliability:

- ✶ Use data replication to prevent single points of failure.
- ✶ Implement daily automated backups to prevent data loss.

☑ Error Recovery & Logging:

- ✶ Implement **real-time monitoring** for error detection.
- ✶ Store **error logs** for debugging and troubleshooting.

## ❖ Compatibility Requirements

- ✶ Ensures the system works across different devices and platforms.

☑ Browser Compatibility:

- ✶ Support **Chrome, Firefox, Edge, Safari** (latest 3 versions).
- ✶ Ensure compatibility with **JavaScript** disabled scenarios.

☑ Operating System Compatibility:

- ✶ Support Windows, macOS, Linux, Android, and iOS.
- ✶ Ensure proper rendering in **light and dark mode settings**.

☑ API Compatibility:

- ✶ Expose RESTful APIs for integration with third-party services (e.g., Payment Gateway, Shipping API).
- ✶ Ensure **APIs return standardized HTTP status** codes (200, 400, 500).

## ❖ Maintainability & Scalability

- ✶ Ensures the system is easy to update and expand.

☑ Code Maintainability:

- ✶ Follow **modular architecture** to allow easy updates.
- ✶ Use consistent **coding standards** (e.g., naming conventions, documentation).

☑ Logging & Monitoring:

- ✶ Use centralized logging to track system errors.
- ✶ Implement real-time alerts for system failures.

☑ Future Scalability:

- ✶ Ensure the system can handle a **10x increase** in users/products.
- ✶ Support **microservices-based architecture** for flexibility.

## 2. Test Strategy

### 2.1 Testing Approach:

- ✶ **The Black Box Testing Strategy for Amazon.eg covers.**
- ✶ **functional testing (user authentication, search, checkout, payments, order management)**
- ✶ **non-functional testing (performance, security, usability, compatibility, scalability, accessibility, and reliability) to ensure a seamless, secure, and high-performing e-commerce experience.**

### 2.2 Test Environment:

- ✶ **Chrome Version 132.0.6834.160 (Official Build) (64-bit)**
- ✶ **Windows 10, 11**
- ✶ **IOS**