Automation Exercise

Software Requirements Specifications

## Prepared By

Omar Saber

# 1. Introduction

## 1.1 Purpose

This document outlines the functional and non-functional requirements for the Automation Exercise platform. It serves as a reference for developers, QA engineers, and stakeholders to ensure alignment with project objectives.

## 1.2 Scope

Automation Exercise is an e-commerce simulation platform designed for QA engineers and developers to practice automation testing and API testing. It provides predefined test cases and an interactive environment for learning.

## 1.3 Intended Audience and Usage

This document is intended for:  
- Developers – To implement the system based on defined requirements.  
- QA Engineers – To design test cases and validate system functionality.  
- Project Managers – To track development progress and ensure project goals are met.

## 1.4 Definitions and Acronyms

- SRS – Software Requirements Specification  
- QA – Quality Assurance  
- API – Application Programming Interface

# 2. Overall Description

## 2.1 Product Perspective

The system is a standalone web application that provides a simulated e-commerce platform for automation testing practice. It is designed as an educational tool without integration with real-world payment gateways.

## 2.2 Product Features

- User Registration & Authentication – Secure user sign-up and login functionality.  
- Product Catalog – Display of products with detailed information.  
- Shopping Cart & Checkout – Users can add products and complete simulated orders.  
- API Testing Playground – Allows users to test predefined API endpoints.  
- Test Case Repository – Provides pre-written manual and automated test cases.

# 3. Functional Requirements

The system must provide the following core functionalities:  
- Secure user authentication with password encryption.  
- Product browsing, search, and filtering features.  
- Shopping cart management and simulated order placement.  
- API testing with real-time request-response interaction.  
- Access to predefined test cases for automation practice.

# 4. Non-Functional Requirements

Performance:  
- System response time should be under 2 seconds under normal load.  
- The platform should handle up to 1,000 concurrent users efficiently.  
  
Security:  
- User data must be encrypted using AES-256.  
- Protection against SQL injection, XSS, and CSRF attacks.  
  
Usability:  
- The UI should be intuitive and responsive across all devices.  
  
Compatibility:  
- The system should be compatible with modern browsers (Chrome, Firefox, Safari, Edge).

# 5. Use Cases

## Use Case 1: User Registration

Actors: New User  
  
Description: A user registers an account on the platform.  
  
Steps:  
1. The user clicks on 'Sign Up'.  
2. The system prompts for email and password.  
3. The user submits the details.  
4. The system validates the input and creates an account.  
5. The user receives a confirmation message.

## Use Case 2: Adding a Product to the Cart

Actors: Registered User  
  
Description: A user adds a product to their shopping cart.  
  
Steps:  
1. The user browses the product catalog.  
2. The user clicks the 'Add to Cart' button.  
3. The system updates the cart and displays the updated total.

## Use Case 3: API Testing

Actors: QA Engineer  
  
Description: A QA engineer tests an API endpoint.  
  
Steps:  
1. The user selects an API endpoint from the API Playground.  
2. The user enters request parameters and sends the request.  
3. The system returns a response with status codes and data.

# 6. Deliverables

- SRS Document (this document).  
- Wireframes & UI Prototypes (if applicable).  
- Test Cases & Automation Scripts (for API testing and UI testing).

# 7. Future Enhancements

- Integration with real-world payment gateways.  
- AI-powered test script generation based on user behavior.  
- Multi-language support for a global audience.

# 8. Conclusion

This document provides a structured SRS for Automation Exercise, ensuring a clear understanding of system requirements. It serves as a blueprint for developers, testers, and stakeholders to align expectations and development goals.