**Day 5 - Testing, Error Handling, and Backend Integration Refinement**

# Prepared By: Zeeshan Shah

# Date: 22 January 2025

**Table of Contents**

1. Introduction
2. Objectives
3. Key Areas of Focus
4. Testing Strategy
5. Test Results and Analysis
6. Error Handling Implementation
7. Documentation Summary
8. Lessons Learned
9. Conclusion

# Introduction

This document summarizes the testing, error handling, and backend integration refinement processes conducted for a fully functional Next.js e-commerce website powered by Sanity CMS. This document outlines the strategies, results, and improvements to ensure the system meets professional functionality, performance, and reliability standards.

# Objectives

* + Validate all core functionalities such as product listings, dynamic routings, cart operations, and checkout workflows.
  + Implement robust error handling to ensure user-friendly messaging for system failures.
  + Optimize the website for performance metrics like speed and responsiveness.
  + Test for cross-browser compatibility and device responsiveness.
  + Ensure data security and input validation throughout the platform.

# Key Areas of Focus

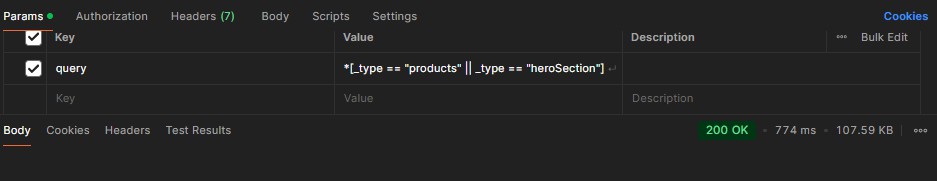
1. **Functional Testing**: Verifying critical features including product displaying, search, wishlist, and checkout.
2. **Error Handling**: Adding user-friendly messages for network errors and API failures.
3. **Performance Optimization**: Using tools like Lighthouse, Postman to identify and address bottlenecks.
4. **Cross-Browser and Device Testing**: Ensuring consistent rendering and functionality on major browsers and devices.
5. **Security Testing**: Validating input fields and securing sensitive API keys.

# Testing Strategy

## Functional Testing

* + **Tools Used**: Postman for API validation.

### Actions:

* + - Verified product listings and detail page functionalities.
    - Simulated user actions such as adding products to the cart and updating quantities.
    - Ensured dynamic routing works seamlessly for individual product pages.

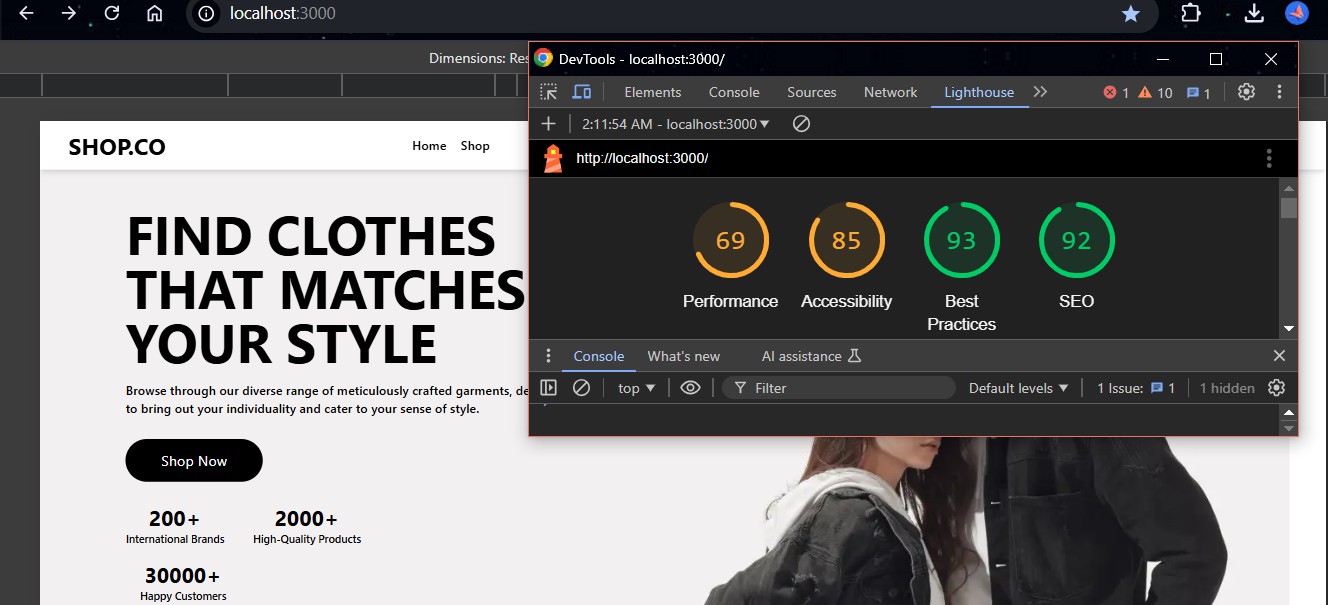
## Error Handling Testing

* + Tested API error handling by disconnecting services and observing fallback behaviors.
  + Verified that descriptive error messages are displayed to users during failures.

## Performance Testing

* + **Tools Used**: Lighthouse

### Actions:

* + - Analyzed page load times.
    - Compressed images to improve load times.

## Cross-Browser and Device Testing

* + **Tools Used**: manual testing on physical devices.

### Actions:

* + - Tested the website on Chrome, Firefox, Safari, and Edge.
    - Verified responsive design on mobile, tablet, and desktop devices.

# Test Results and Analysis

Attached .csv file

# Error Handling Implementation

* + Added try-catch blocks around all API calls to catch and log errors.
  + Displayed fallback UI messages, when the API fails.
  + Ensured the system logs detailed error information for debugging purposes.

Example Code Snippet:

# Documentation Summary

### Files Submitted:

* + - CSV test report with detailed results.
    - Performance reports

### Improvements Documented:

* + - Enhanced error messages and fallback UI.

### Screenshots Included:

* + - Performance metrics.
    - API error handling tests.

# Lessons Learned

* + Proactive error handling improves user trust and experience.
  + Regular performance audits are critical to maintaining a high-quality application.
  + Cross-browser testing ensures consistent user experience across platforms.

# Conclusion

By rigorously testing and optimizing the Next.js e-commerce website integrated with Sanity CMS, the project achieved its goals of functionality, performance, and user experience. The documented efforts and results ensure readiness for real-world deployment.