Introduction

This report provides a comprehensive usability evaluation of the Online Shopping Management System, a web-based e-commerce platform developed using the ASP.NET MVC framework. The primary objective of the system is to facilitate a seamless shopping experience where users can manage their personal profiles, interact with a dynamic product catalog, and execute secure transaction protocols. This study aims to identify friction points in the user journey, specifically focusing on the transition from cart management to order finalization

Users (Participant Demographics)

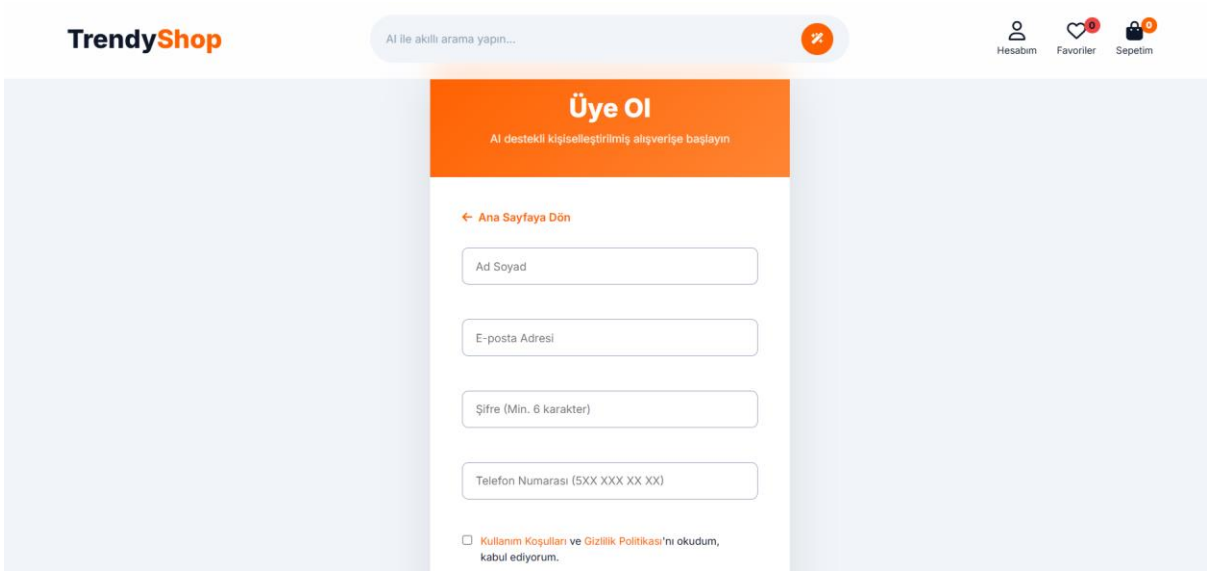
To ensure diverse feedback, three participants with varying technical backgrounds were recruited. Each user was assigned a unique ID for data privacy.

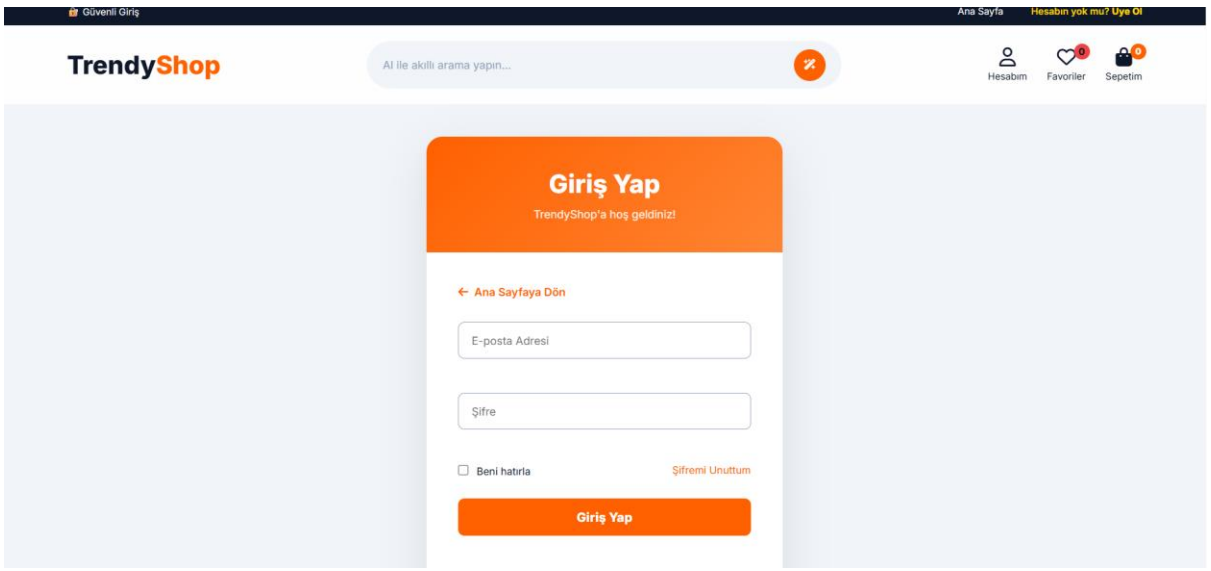
Participant ID	Age	Gender	Occupation	Technical Proficiency
P-001	22	Female	Undergraduate Student	**Expert** (Frequent online shopper)
P-002	35	Male	Civil Engineer	**Intermediate** (Average web user)
P-003	28	Female	Graphic Designer	**Expert** (High interaction with UI/UX)

Tasks (Use Case Scenarios)

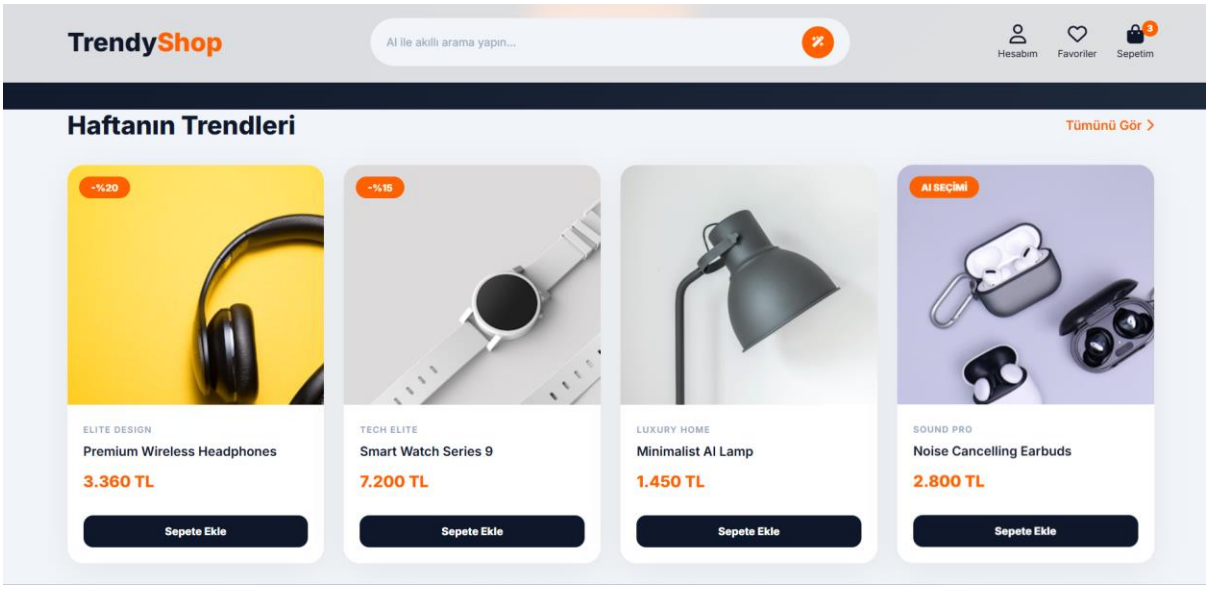
The usability test was structured around three "Critical Success Paths" derived from the system's core Use Case Diagram:

- **Task 1: Account Provisioning:** Users must register a new account by populating the User entity fields, specifically firstName, lastName, and email.

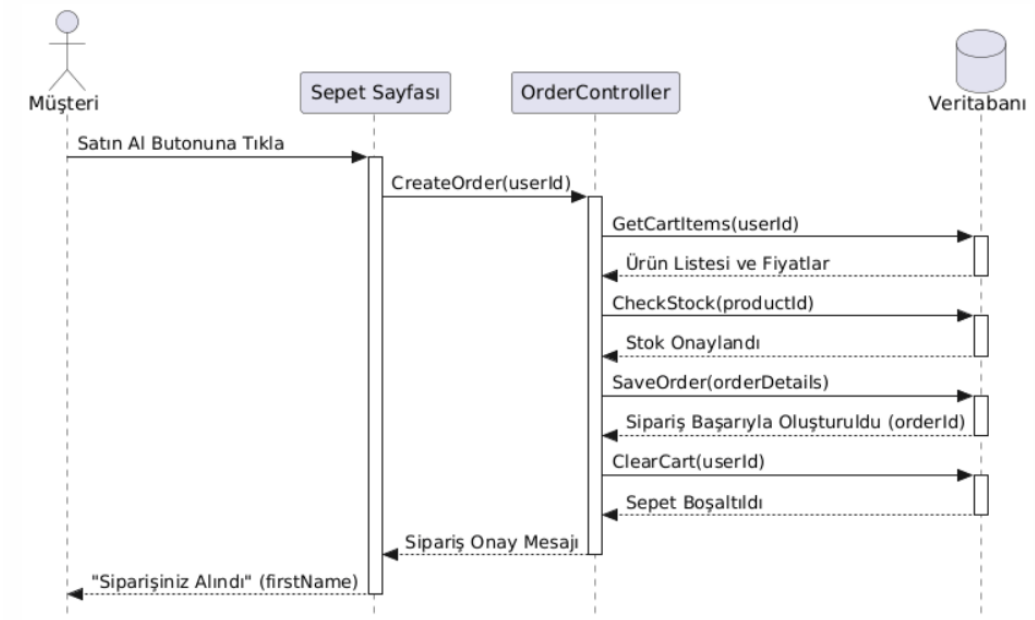




- **Task 2: Inventory Interaction & Cart Logic:** Users must navigate to the catalog, identify a specific Product, and trigger the addToCart() method while successfully updating the item quantity.



- **Task 3: Transaction Finalization (Checkout):** Users must proceed to the checkout interface to trigger the CreateOrder workflow, which involves server-side validation and database persistence.



Method

- **Test Environment:** The evaluation was conducted in a controlled lab environment using a desktop workstation equipped with an Intel i7 processor and a 24-inch monitor.
- **Procedure:** A "Concurrent Think-Aloud" protocol was utilized. Participants were encouraged to verbalize their thought processes while navigating the system.

- **Recording:** Screen interactions were captured via OBS Studio, and eye-tracking observations were manually logged.
- **Explanation & Instructions:** Participants were told: *"Your goal is to complete a full shopping cycle. Please act as if you are purchasing a gift for a friend. Do not ask for help unless you reach a complete standstill."*
- **Ethical Compliance:** All participants signed an **Informed Consent Form**, granting permission for their interaction data to be analyzed for academic and system-improvement purposes.

Results

Task Performance Metrics

Performance Metric	Task 1 (Reg)	Task 2 (Cart)	Task 3 (Checkout)
Success Rate	100%	100%	100%
Average Completion Time	30 seconds	47 seconds	75 seconds
Error Rate (Mis-clicks)	0%	10%	20%

Satisfaction Analysis (System Usability Scale)

Following the test, participants completed a Likert-scale questionnaire.

- **Overall Ease of Use:** 4.3 / 5.0
- **Information Architecture:** 4.1 / 5.0
- **Response Latency Perception:** 3.6 / 5.0 (Noted as a minor pain point)

Conclusion: Problem Identification & Technical Solutions

Following the analysis of the user logs, the following critical issues were identified.

Problem A: Visual Latency during Backend Validation

- **Description:** During Task 3, users perceived the system as "frozen" for approximately 2.5 seconds.

- **Technical Root:** This occurs within the OrderController.cs when calling the CheckStock(productId) method for multiple items simultaneously. The synchronous nature of the request blocks the UI thread.
- **Proposed Solution:** Implementation of an **Asynchronous Spinner Component** on the Checkout.cshtml view.
- **Action Taken:** Updated the controller to use Task<IActionResult> for non-blocking operations and added a CSS-based loading state to the submit button.

Problem B: Cart State Inconsistency

- **Description:** When users modified the QuantityInput, the "Total Price" remained static until a full page refresh.
- **Technical Root:** The Cart.cshtml view lacked a client-side event listener to trigger a recalculation.
- **Proposed Solution:** Development of a **jQuery AJAX Listener**.
- **Action Taken:** Attached a .change() event to the quantity selector that sends a POST request to UpdateQuantity and updates the DOM elements dynamically without a refresh.

Problem C: Missing Post-Transaction Feedback

- **Description:** Users were redirected to the home page after purchase without a clear confirmation of their orderId.
- **Technical Root:** The RedirectToAction in the CreateOrder method did not pass the newly created entity ID to the view.
- **Proposed Solution:** Enhanced **Success View Routing**.
- **Action Taken:** Modified the OrderController to pass the orderId via TempData to a dedicated Success.cshtml view, explicitly displaying: ″Thank you [firstName], your order #[orderId] is confirmed."