

Secure Order Management System (SOMS): Functional Requirements Assessment

Kaya Nelson

Grand Canyon

University

CST-452 – Senior Capstone Project

Professor Michael Landreth

October 26, 2025

## Secure Order Management System (SOMS): Functional Requirements Assessment

### FR-001: User Registration

Description. The registration capability permits new users to create an account by providing a valid username (5–30 characters, alphanumeric or underscore) and a password (5–100 characters). Realtime validation reinforces input quality. Passwords are hashed with BCrypt before database storage to preserve confidentiality.

Status. Fully implemented.

Details. The register.html interface provides live input feedback, color cues, and actionable error messages. Duplicate usernames are detected and rejected. Successful registration redirects to the login view and persists in encrypted credentials to the database.

### FR-002: User Authentication

Description. Registered users authenticate with their credentials. Upon a successful login, the application establishes a secure session, enabling access to protected resources.

Status. Fully implemented.

Details. The login.html view validates username and password length and displays context specific error messages on failure. Valid logins initialize a secure token or session and redirect to shop.html.

### FR-003: Session Management

Description. Protected views require an active authenticated session; unauthenticated users are redirected to the login page.

Status. Fully implemented.

Details. shop.html, checkout.html, and confirmation.html verify the presence of a valid session before rendering. Middleware safeguards backend endpoints from unauthorized access.

#### FR-004: User Logout

Description. Authorized users may terminate their sessions at any time.

Status. Fully implemented.

Details. A logout control on shop.html prompts confirmation, clears session data, and returns the user to login.html, preventing reuse of stale sessions.

#### FR-005: Product Display

Description. Authenticated users can browse a catalog with product name, category, description, and price.

Status. Fully implemented.

Details. Products are retrieved via a secured GET /products request and presented in a responsive grid with hover based affordances.

#### FR-006: Shopping Cart Management

Description. Users can add items to a cart that persists for the session and updates in real time.

Status. Fully implemented.

Details. The POST /cart/add endpoint synchronizes server side state; the navigation counter and confirmation messages update immediately.

#### FR-007: Cart Viewing

Description. Users may inspect current cart contents, quantities, unit prices, and aggregate totals.

Status. Fully implemented.

Details. A tabular cart display recalculates subtotals and totals as the state changes, ensuring consistent client–server values.

#### FR-008: Cart Modification

Description. Quantities can be increased or decreased, and items can be removed entirely.

Status. Fully implemented.

Details. The application uses PUT /cart/update for quantity changes and DELETE /cart/remove for deletions; zero quantity entries are dropped automatically.

#### FR-009: Checkout Process

Description. Checkout captures shipping information and order summaries; the system generates a unique order identifier and records the transaction.

Status. Partially implemented.

Details. checkout.html enforces city selection and phone number formatting, then posts to POST /orders. The cart is cleared after confirmation. A simulated payment module was deferred to prioritize database integrity.

#### FR-010: Order History

Description. Users can later review completed orders with dates, items, and totals.

Status. Not implemented.

Details. Backend data structures support order archival; front-end presentation and API exposure will be delivered in a subsequent iteration.

#### FR-011: Navigation System

Description. Navigation remains consistent across authenticated views while preserving session state.

Status. Fully implemented.

Details. Menus are standardized across home.html, login.html, register.html, shop.html, checkout.html, and confirmation.html.

## FR-012: Data Persistence

Description. Credentials, carts, and orders are stored securely and reliably.

Status. Partially implemented.

Details. User and cart data persist with encryption and access controls. Order records are written at checkout; history retrieval awaits the FR-010 feature.

## Planned Improvements

Order History (FR-010). Implement orders.html and a GET /orders/user endpoint for peruser history with itemization.

Payment Simulation (FR-009). Add a mock payment form with field validation to approximate a realistic checkout.

Enhanced City Options. Expand the city list or allow validated free text entry. Search and Filtering. Integrate product search and category filters into shop.html. User

Profile Page. Provide password and profile updates via a secure dashboard.

Email Notifications. Send registration and order confirmations via an SMTP or mail API.

Performance Optimization. Improve database query efficiency; evaluate server managed session scaling.

Localization. Introduce multilingual support beginning with Spanish.