
Document Title: Software Requirements Specification for “ShopMaster_360”

- **Document Status:** Draft v1.0
- **Authors:** Telar Karan – As a Product Manager.

Revision History

Name	Date	Reason for Changes	Version

---

1. Introduction

1.1 Purpose

This Software Requirements Specification (SRS) describes the requirements for **ShopMaster_360**, an e-commerce web application intended to expand Rambhai’s physical store into an online platform. It covers all essential functionalities including user registration, catalog browsing, shopping cart, checkout, and order management. This SRS serves as a guide for developers, testers, and stakeholders to understand what the system must deliver and how it should behave.

1.2 Document Conventions

- **Priority Labels:** Requirements in this document may be labeled **High (H)**, **Medium (M)**, or **Low (L)** priority.
- **Requirements Identification:** Each functional requirement is uniquely identified as `REQ-<Number>`.

1.3 Intended Audience and Reading Suggestions

- **Developers**: To understand the functional, performance, and security requirements.
- **Project Managers**: For planning, task allocation, scheduling, and scope management.
- **Marketing Staff & Stakeholders**: To comprehend business justification and high-level product features.
- **Testers**: To design test cases aligning with outlined functionalities and acceptance criteria.
- **Documentation Writers**: To prepare user manuals and support guides.

Suggested Reading Path:

1. **Introduction (Section 1)** – to gain an overview of the product scope.
2. **Overall Description (Section 2)** – for high-level context, features, and constraints.
3. **System Features (Section 3)** – for detailed functional requirements.
4. **External Interface & Nonfunctional Requirements (Sections 4 & 5)** – for interface details and quality attributes.

1.4 Project Scope

ShopMaster_360 is an online e-commerce platform focused on:

- Expanding a local store's reach to broader markets.
- Providing a user-friendly and scalable solution.
- Enabling customers to view products, place orders, and securely complete payments. Key goals include improving sales, customer satisfaction, and operational efficiency through streamlined order management.

1.5 References

- **Business Requirements Specification (BRS)**: Business Requirements for ShopMaster_360, Version 1.0.
- **Payment Gateway Documentation**: Stripe, Paytm, and GPay developer guides.
- **Security Standards**: OWASP documentation (https://owasp.org/www-project-top-ten/).

---

2. Overall Description

2.1 Product Perspective

ShopMaster_360 is a **new, self-contained** web application, designed for modular deployment in a cloud environment. The application interfaces with:

- **Payment Gateways**: Stripe, Paytm, and GPay.
- **User Notification Services**: Email (SMTP) and SMS for order updates.

- **Database**: A relational or NoSQL database (TBD) for user and order records.

A simplified architectural diagram (TBD) would depict:

- Front-end user interface (browser/mobile)
- Middleware (business logic, shopping cart, order tracking)
- Database layer
- External payment and notification services

2.2 Product Features

1. **User Registration & Profile Management**
2. **Catalog Browsing & Advanced Search**
3. **Shopping Cart & Checkout**
4. **Order Management & Tracking**
5. **Notifications via Email/SMS**

2.3 User Types and Characteristics

1. **Guests**:
    - Can browse products but must register or log in to complete purchases.
2. **Registered Customers**:
    - Have personal profiles, can track orders, and manage saved addresses/payment options.
3. **Store Admins**:
    - Manage product catalog, prices, and promotions.
    - Oversee order fulfillment, cancellations, and refunds.

2.4 Operating Environment

- **Platform**: Web browsers (Chrome, Firefox, Safari, Edge), mobile-responsive.
- **Server OS**: Linux-based environment (e.g., Ubuntu, AWS Linux) or Windows Server if required.
- **Database**: (TBD).
- **Cloud/Hosting**: Custom Linux-based server or AWS, Azure, or a similar cloud service provider in future.

2.5 Design and Implementation Constraints

- **Limited Development Time & Budget**: The scope of initial features must be strictly managed.
- **Technology Stack**: Must integrate with Stripe, Paytm, GPay as payment gateways.
- **Security**: Must comply with PCI DSS if credit card payments are handled and OWASP for web standards.
- **Scalability**: Must support cloud deployments with potential load balancing.

2.6 User Documentation

- **User Guide**: An online help manual explaining how to browse, buy, and manage orders.
- **Admin Manual**: Covers product listings, order workflow, and refunds.
- **API Documentation**: For any future integrations with third-party systems.

2.7 Assumptions and Dependencies

- **Testing**: Initial performance testing will be done locally with a mock load.
- **Load Balancing**: Assumes readiness for cloud scaling or deployment on additional servers.
- **Payment Gateway Availability**: Reliance on Stripe, Paytm, or GPay to be fully operational.
- **Network Connectivity**: End users must have stable internet to use the service effectively.

---

3. System Features

This section describes the main functionalities of **ShopMaster_360**.

3.1 System Feature 1: User Registration & Profile Management

3.1.1 Description and Priority

Description: The system allows new users to register using an email address or social logins (Google, Facebook, etc.). Registered users can manage personal details, view order history, and update payment/shipping information.
Priority: Email registration - High, Social logins - Low

3.1.2 Stimulus/Response Sequences

1. **User clicks "Sign Up"** → System displays registration form.
2. **User fills details and submits** → System validates data and, if valid, creates an account.
3. **User logs in** → System fetches user profile and displays personalized dashboard.

3.1.3 Functional Requirements

- **REQ-1**: The system **shall** provide a registration form requiring at least an email, password, and name.
- **REQ-2**: The system **shall** allow social media login (Google/Facebook).
- **REQ-3**: The system **shall** enable users to edit and update their personal details (name, address, phone, etc.).

- **REQ-4**: The system **shall** allow password reset via email.

3.2 System Feature 2: Catalog Browsing & Search

3.2.1 Description and Priority

Description: Users can browse products by category, apply filters (price, brand, etc.), and perform advanced searches using keywords or product attributes.
Priority: High

3.2.2 Stimulus/Response Sequences

1. **User navigates to the home page** → System shows featured or recommended products.
2. **User applies filters or enters search query** → System returns a filtered list of products.
3. **User selects a product** → System displays product details (price, description, images).

3.2.3 Functional Requirements

- **REQ-5**: The system **shall** provide category-based navigation and filtering (e.g., price range, brand).
- **REQ-6**: The system **shall** allow keyword search for product names, descriptions, or tags.
- **REQ-7**: The system **shall** display product details, including images, description, and pricing.

3.3 System Feature 3: Shopping Cart & Checkout

3.3.1 Description and Priority

Description: Registered users can add products to a cart, view cart contents, and proceed with checkout using multiple payment options (cards, UPI, etc.).
Priority: High

3.3.2 Stimulus/Response Sequences

1. **User clicks "Add to Cart"** → System updates cart with product information and quantity.
2. **User proceeds to checkout** → System collects shipping info, payment method, and order summary.
3. **User confirms payment** → System processes payment and returns success/failure status.

3.3.3 Functional Requirements

- **REQ-8**: The system **shall** store items in the user's cart with quantity and price.
- **REQ-9**: The system **shall** support multiple payment methods (Credit Cards, UPI, Stripe, Paytm, GPay).
- **REQ-10**: The system **shall** validate shipping information before processing payment.
- **REQ-11**: The system **shall** confirm the order and provide an order confirmation number upon successful payment.

3.4 System Feature 4: Order Management

3.4.1 Description and Priority

Description: The system must track order status (pending, shipped, delivered, cancelled) and send real-time updates via email/SMS. Users and administrators should manage cancellations and refunds.
Priority: High

3.4.2 Stimulus/Response Sequences

1. **User views orders in profile** → System displays the status of all orders.
2. **Admin updates order status** → System notifies the user via email/SMS.
3. **User requests cancellation** → System validates if the order can be canceled and processes accordingly.

3.4.3 Functional Requirements

- **REQ-12**: The system **shall** track each order with a unique Order ID and maintain status updates.
- **REQ-13**: The system **shall** send email and/or SMS notifications for order placement, shipping, delivery, or cancellation.
- **REQ-14**: The system **shall** allow authorized users (admins) to initiate refunds.
- **REQ-15**: The system **shall** allow users to view their order history and status in their profile.

---

4. External Interface Requirements

4.1 User Interfaces

- **Web UI**: Responsive pages for registration, product browsing, cart management, and order tracking.

- **Standard Layouts**: Adherence to typical e-commerce design for intuitive user experience (header navigation, product grid, checkout flow).
- **Error Messages**: Clear prompts when errors occur (e.g., payment failure, invalid form inputs).

4.2 Hardware Interfaces

No direct hardware interfaces beyond standard hosting infrastructure. The solution must be deployable on a server meeting minimum system requirements (CPU, RAM, disk) as specified by the chosen cloud provider.

4.3 Software Interfaces

- **Payment Gateways**: Integration APIs for Stripe, Paytm, GPay.
- **Email Services**: SMTP server or 3rd-party email API (e.g., SendGrid).
- **SMS Gateway** (optional/third-party): For sending order updates via text messages.
- **Database**: MySQL/PostgreSQL or other chosen DB for user accounts, orders, and product data.

4.4 Communications Interfaces

- **HTTP/HTTPS**: All client-server communications.
- **Secure Sockets (SSL/TLS)**: Required for PCI compliance during checkout and user login.
- **RESTful APIs**: For back-end microservices or external integrations.

---

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- The system **should** handle at least 1000 concurrent users on a minimal hosting configuration.
- Page load times **should** be under 2 seconds under normal conditions.
- Checkout process **must** respond in under 5 seconds during typical loads.
- Sufficient and Scalable Database storage, Expecting 50K users in First year.

5.2 Safety Requirements

- Not applicable beyond standard e-commerce safety measures (no direct hardware hazards).
- Any issues causing downtime or data inaccuracies **must** trigger immediate alerts to the admin.

5.3 Security Requirements

- **Secure Authentication**: Passwords stored using salted hashing.
- **PCI & OWASP Compliance**: Required if storing card data (if not, redirect to gateway-hosted payment page).
- **Encryption**: All sensitive data (user personal info, payment details) in transit over HTTPS.
- **Role-Based Access**: Admin functions must be accessible only to authenticated administrators.

5.4 Software Quality Attributes

- **Reliability**: Aim for 99.9% uptime; system monitoring for quick issue resolution.
- **Usability**: Intuitive interface design, minimal clicks for checkout.
- **Maintainability**: Modular code architecture, documented APIs for easy updates.
- **Scalability**: Cloud-based deployment with load balancing to handle peak traffic.

---

6. Other Requirements

- **Internationalization (Optional)**: If needed, support multiple languages/currencies in the future.
- **Legal/Compliance**: Must conform to local e-commerce regulations.
- **Data Backup**: Daily backups of the database and transaction logs.

Appendix A: Glossary

- **PCI DSS**: Payment Card Industry Data Security Standard.
- **UPI**: Unified Payments Interface (common in certain regions).
- **SKU**: Stock Keeping Unit, used for product identification.
- **SMS**: Short Message Service.

Appendix B: Analysis Models

(Optional – Include diagrams such as use case, data flow, or entity relationship here if needed.)

Appendix C: Issues List

- **TBD: Payment Gateway Fee Structures** – Detailed cost considerations for Stripe/Paytm/GPay.
- **TBD: Final Database Choice** – Decision between MySQL, PostgreSQL, or MongoDB pending.
- **TBD: Final Hosting Environment** – AWS, Azure, or another provider.

End of Software Requirements Specification for ShopMaster_360