Software Requirements Specification (SRS)

Pappad Shop Management System

Document Version: 1.0
Date: January 27, 2026
Prepared By: Development Team
Project Name: Web-Based Pappad Shop Management System

Table of Contents

---

1. Introduction

1.1 Purpose

This Software Requirements Specification (SRS) document provides a complete description of the Web-Based Pappad Shop Management System. It describes the functional and non-functional requirements for the system that will be developed. This document is intended for:

- Development team members
- Project stakeholders
- System testers
- System administrators
- End users (Admin and Customers)

1.2 Scope

The Pappad Shop Management System is a web-based application designed to streamline and automate the order management process for a pappad retail business. The system consists of two separate applications:

Admin Application:

- Product management (add, edit, delete pappad items with descriptions and images)
- Inventory management and stock tracking
- Order management and processing
- Customer management
- Sales reports and analytics
- Notification management

Customer Application:

- Browse pappad products without login
- User registration and authentication for placing orders
- Shopping cart functionality
- Online order placement with payment
- Order tracking
- Order history
- Customer profile management
- Notification system

Key Benefits:

- Eliminates miscommunication from phone-based orders
- Provides customers with order review and confirmation capability
- Automates billing and inventory management
- Reduces manual work and human errors
- Enables secure data storage and efficient order processing

1.3 Definitions, Acronyms, and Abbreviations

Term	Definition
SRS	Software Requirements Specification
UI	User Interface
UX	User Experience
API	Application Programming Interface
CRUD	Create, Read, Update, Delete
SMS	Short Message Service
UPI	Unified Payments Interface
MongoDB	NoSQL Database Management System
JWT	JSON Web Token
HTTPS	Hypertext Transfer Protocol Secure
CSS	Cascading Style Sheets
HTML	HyperText Markup Language
JS	JavaScript

1.4 References

- IEEE Std 830-1998 - IEEE Recommended Practice for Software Requirements Specifications
- Project Synopsis - Pappad Shop Management System
- MongoDB Documentation
- Java Servlets/Spring Boot Documentation
- Bootstrap Framework Documentation

1.5 Overview

This document is organized into nine main sections:

- Section 1 provides an introduction to the document
- Section 2 gives an overall description of the system
- Section 3 details specific functional and non-functional requirements
- Section 4 describes external interface requirements
- Section 5 outlines the system architecture
- Section 6 specifies database requirements
- Section 7 covers security requirements
- Section 8 defines quality attributes
- Section 9 includes appendices with additional information

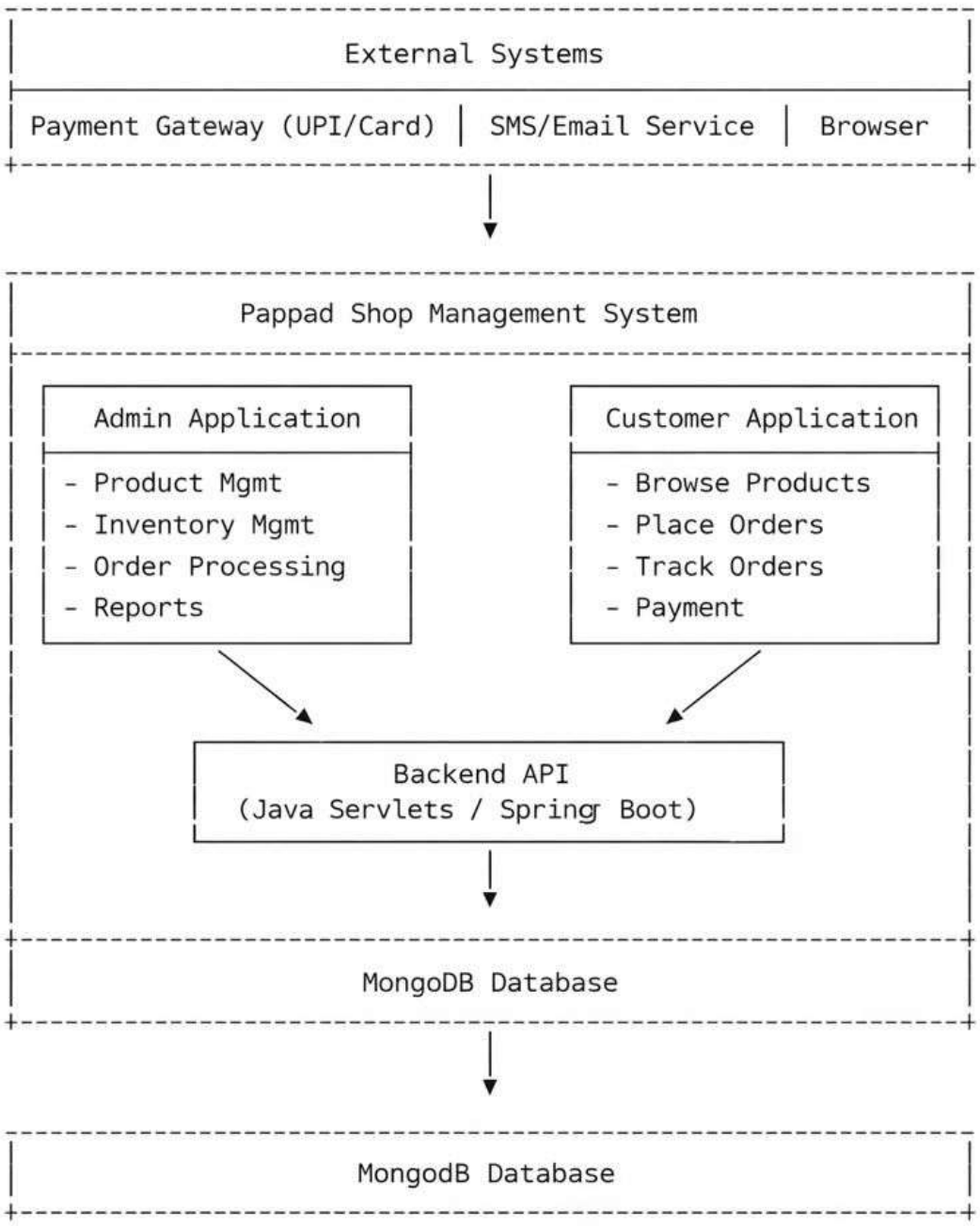
2. Overall Description

2.1 Product Perspective

The Pappad Shop Management System is a new, self-contained web-based application that replaces the existing manual and phone-based ordering system. The system consists of two distinct web applications:

1. **Admin Web Application** - For shop management
2. **Customer Web Application** - For customer ordering

System Context Diagram:



2.2 Product Functions

Major functions include:

1. **User Management**
    - Customer registration and authentication
    - Admin authentication
    - Profile management
    - Role-based access control
2. **Product Management**
    - Add/Edit/Delete pappad products
    - Manage product categories
    - Upload product images
    - Manage product descriptions and prices
    - Display common pappad items initially
3. **Inventory Management**
    - Track stock levels
    - Automatic stock reduction on order placement
    - Low stock alerts
    - Stock replenishment management
4. **Order Management**
    - Browse products without login
    - Add products to cart (requires login)
    - Place orders with delivery address
    - Online payment integration
    - Order summary generation
    - Order status tracking

- Order history access

5. **Notification System**
    - Order confirmation notifications (Customer & Admin)
    - New product notifications for regular customers
    - Low stock alerts (Admin)
    - Order status updates
6. **Reporting and Analytics**
    - Daily sales reports
    - Monthly revenue reports
    - Product-wise sales analysis
    - District-wise order reports
    - Customer reports
    - Export reports (PDF, Excel)
    - Dashboard with analytics and graphs

2.3 User Characteristics

1. Administrator

- **Technical Expertise:** Moderate computer literacy
- **Domain Knowledge:** Complete understanding of pappad business
- **Responsibilities:**
    - Manage products and inventory
    - Process and track orders
    - Generate reports
    - Manage customer information
    - System configuration
- **Frequency of Use:** Daily, multiple times

2. Customers

- **Technical Expertise:** Basic to moderate computer literacy
- **Age Group:** 18-65 years
- **User Types:**
    - Individual retail buyers
    - Bulk buyers (restaurants, stores)
    - Regular customers
    - First-time customers
- **Responsibilities:**
    - Browse products
    - Place orders
    - Make payments
    - Track orders
- **Frequency of Use:** Varies (weekly to monthly)

2.4 Constraints

1. **Technical Constraints:**
    - Must use Java (Servlets/Spring Boot) for backend
    - Must use MongoDB as the database
    - Must be deployed on Apache Tomcat
    - Must support all major web browsers
2. **Business Constraints:**
    - System must handle up to 5,000 users
    - Product catalog limited to 500 items
    - Daily order volume approximately 50 orders
    - Orders cannot be cancelled or modified after placement

3. **Regulatory Constraints:**
    - Must comply with data protection regulations
    - Secure payment processing required
4. **Design Constraints:**
    - Must be responsive (mobile, tablet, desktop)
    - Must work across different browsers
    - Separate applications for admin and customer
    - No login required for browsing products

2.5 Assumptions and Dependencies

Assumptions:

1. Users have access to internet-enabled devices
2. Users have basic computer/smartphone literacy
3. Payment gateway services will be available 99% of the time
4. SMS/Email services will be reliably available
5. Shop has manual logistics arrangement for delivery

Dependencies:

1. Third-party payment gateway integration
2. SMS/Email service provider APIs
3. Internet connectivity
4. Web hosting/server availability
5. MongoDB database service availability
6. Apache Tomcat server

---

3. System Features and Requirements

3.1 Functional Requirements

3.1.1 User Management Module

FR-UM-001: Customer Registration

- **Priority:** High
- **Description:** System shall allow new customers to register with required details
- **Input:** Name, Email, Phone Number, Password, Address
- **Process:**
  - Validate email format and uniqueness
  - Validate phone number format (10 digits)
  - Encrypt password
  - Store customer data in database
- **Output:** Registration confirmation and automatic login
- **Validation Rules:**
  - Email must be unique and valid format
  - Phone number must be 10 digits
  - Password minimum 8 characters with at least one uppercase, lowercase, and number
  - All fields mandatory

FR-UM-002: Customer Login

- **Priority:** High
- **Description:** Registered customers must login to place orders
- **Input:** Email/Phone and Password
- **Process:**
  - Validate credentials against database

    - Generate session token (JWT)
    - Redirect to homepage or cart (if items in cart before login)
- **Output:** Successful login with session creation
- **Error Handling:** Display appropriate error for invalid credentials

FR-UM-003: Admin Login

- **Priority:** High
- **Description:** Admin must authenticate to access admin panel
- **Input:** Admin Username/Email and Password
- **Process:**
    - Validate admin credentials
    - Verify admin role
    - Generate secure session token
    - Redirect to admin dashboard
- **Output:** Successful login to admin application
- **Security:** Implement account lockout after 5 failed attempts

FR-UM-004: Customer Profile Management

- **Priority:** Medium
- **Description:** Customers can view and edit their profile information
- **Input:** Updated profile fields
- **Process:**
    - Display current profile information
    - Allow editing of Name, Phone, Email, Addresses
    - Validate changes
    - Update database
- **Output:** Profile updated confirmation

- **Constraints:** Email and phone uniqueness must be maintained

FR-UM-005: Password Reset

- **Priority:** Medium
- **Description:** Users can reset forgotten passwords
- **Input:** Email/Phone number
- **Process:**
    - Verify user exists
    - Send OTP via SMS/Email
    - Validate OTP
    - Allow new password entry
    - Update password in database
- **Output:** Password reset confirmation

FR-UM-006: Multiple Address Management

- **Priority:** Medium
- **Description:** Customers can save multiple delivery addresses
- **Input:** Address details (House/Flat, Street, District, State, PIN)
- **Process:**
    - Add new address
    - Edit existing address
    - Delete address
    - Mark default address
- **Output:** Address saved/updated confirmation
- **Constraint:** Minimum one address required

3.1.2 Product Management Module (Admin)

FR-PM-001: Add Product

- **Priority:** High
- **Description:** Admin can add new pappad products to catalog
- **Input:**
  - Product Name
  - Category
  - Description
  - Price
  - Unit (kg, packet, piece)
  - Stock Quantity
  - Product Images (multiple)
  - SKU/Product Code
- **Process:**
  - Validate all required fields
  - Upload and store images
  - Generate unique product ID
  - Save product to database
  - Update inventory
- **Output:** Product added confirmation with product ID
- **Validation Rules:**
  - Product name required (max 100 characters)
  - Price must be positive number
  - At least one image required (max 5 images)
  - Image formats: JPG, PNG (max 2MB each)

- Stock quantity must be non-negative integer

FR-PM-002: Edit Product

- **Priority:** High
- **Description:** Admin can modify existing product details
- **Input:** Product ID and updated fields
- **Process:**
  - Retrieve product details
  - Allow editing of all fields
  - Update images if new ones uploaded
  - Validate changes
  - Update database
- **Output:** Product updated confirmation
- **Constraint:** Cannot edit product if active orders exist (warning only)

FR-PM-003: Delete Product

- **Priority:** Medium
- **Description:** Admin can remove products from catalog
- **Input:** Product ID
- **Process:**
  - Check for dependencies (active orders)
  - Confirm deletion
  - Soft delete (mark as inactive) or hard delete
  - Update database
- **Output:** Product deleted confirmation

- **Constraint:** Products with active orders should be soft-deleted (hidden but data retained)

FR-PM-004: View Product List

- **Priority:** High
- **Description:** Admin can view all products with filters and search
- **Input:** Optional filters (category, stock status, price range)
- **Process:**
    - Retrieve products from database
    - Apply filters if specified
    - Sort options (name, price, stock, date added)
    - Pagination (20 products per page)
- **Output:** Paginated product list with details
- **Features:** Search by name, filter by category, stock status

FR-PM-005: Manage Product Categories

- **Priority:** Medium
- **Description:** Admin can create and manage product categories
- **Input:** Category Name, Description, Display Order
- **Process:**
    - Add new category
    - Edit category
    - Delete category (if no products assigned)
    - Reorder categories
- **Output:** Category management confirmation
- **Default Categories:** Plain Pappad, Masala Pappad, Special Varieties

FR-PM-006: Bulk Product Upload

- **Priority:** Low
- **Description:** Admin can upload multiple products via CSV/Excel
- **Input:** CSV/Excel file with product data
- **Process:**
  - Validate file format
  - Parse data
  - Validate each product entry
  - Import valid products
  - Generate error report for invalid entries
- **Output:** Import summary with success/failure counts

3.1.3 Inventory Management Module

FR-IM-001: Stock Tracking

- **Priority:** High
- **Description:** System automatically tracks stock levels
- **Input:** Product ID and quantity change
- **Process:**
  - Update stock on order placement (reduce)
  - Update stock on stock addition (increase)
  - Maintain stock transaction history
  - Calculate available stock
- **Output:** Updated stock quantity
- **Real-time:** Stock updates reflected immediately

FR-IM-002: Automatic Stock Reduction

- **Priority:** High
- **Description:** Stock automatically reduces when order is placed
- **Input:** Order details with product quantities
- **Process:**
  - Validate sufficient stock available
  - Lock stock for order
  - Reduce stock quantity on order confirmation
  - Update inventory database
- **Output:** Updated inventory levels
- **Error Handling:** Prevent order if insufficient stock

FR-IM-003: Low Stock Alerts

- **Priority:** High
- **Description:** System alerts admin when stock falls below threshold
- **Input:** Minimum stock threshold per product (configurable)
- **Process:**
  - Monitor stock levels continuously
  - Compare against threshold
  - Generate alert notification
  - Display in admin dashboard
  - Send email/SMS to admin
- **Output:** Low stock alert notification
- **Threshold:** Default 10 units (configurable per product)

FR-IM-004: Stock Replenishment

- **Priority:** Medium
- **Description:** Admin can add stock for products
- **Input:** Product ID, Quantity to Add, Purchase Date, Supplier Info (optional)
- **Process:**
  - Validate product exists
  - Add quantity to current stock
  - Record transaction in stock history
  - Update database
- **Output:** Stock replenishment confirmation
- **History:** Maintain log of all stock additions

FR-IM-005: Stock History Report

- **Priority:** Medium
- **Description:** View complete stock transaction history
- **Input:** Date range, Product filter (optional)
- **Process:**
  - Retrieve stock transactions
  - Display additions and reductions
  - Show running balance
  - Calculate total changes
- **Output:** Stock history report
- **Export:** Available in PDF/Excel format

FR-IM-006: Inventory Dashboard

- **Priority:** Medium
- **Description:** Admin dashboard showing inventory overview
- **Display:**
  - Total products
  - Low stock products count
  - Out of stock products
  - Total inventory value
  - Recent stock movements
- **Output:** Visual dashboard with charts and statistics

3.1.4 Product Browsing Module (Customer)

FR-PB-001: View Product Catalog

- **Priority:** High
- **Description:** Customers can browse products without login
- **Input:** None (default view) or category filter
- **Process:**
  - Retrieve active products from database
  - Display with images, name, price, brief description
  - Group by categories
  - Show stock availability status
- **Output:** Product catalog grid/list view
- **Default Display:** Show common pappad items initially
- **Pagination:** 12-20 products per page

FR-PB-002: Product Search

- **Priority:** High
- **Description:** Search products by name or keywords
- **Input:** Search query string
- **Process:**
  - Search in product name and description
  - Return matching products
  - Highlight search terms
  - Show relevant results first
- **Output:** Filtered product list
- **Features:** Auto-suggest, fuzzy matching

FR-PB-003: Filter Products

- **Priority:** Medium
- **Description:** Filter products by various criteria
- **Input:** Filter options
- **Available Filters:**
  - Category
  - Price range
  - Availability (In Stock / Out of Stock)
  - Sort by (Price: Low to High, High to Low, Name A-Z, Newest)
- **Output:** Filtered product list
- **Reset:** Option to clear all filters

FR-PB-004: View Product Details

- **Priority:** High
- **Description:** View complete product information

- **Input:** Product ID (click on product)
- **Display:**
    - Product images (gallery with zoom)
    - Product name and SKU
    - Detailed description
    - Price per unit
    - Unit type (kg/packet/piece)
    - Stock availability
    - Customer reviews (future enhancement)
    - Related products
- **Output:** Detailed product page
- **Actions:** Add to Cart, Buy Now buttons

3.1.5 Shopping Cart Module

FR-SC-001: Add to Cart

- **Priority:** High
- **Description:** Customers can add products to cart (login required)
- **Input:** Product ID, Quantity
- **Process:**
    - Check if user is logged in (redirect to login if not)
    - Validate stock availability
    - Check if product already in cart (update quantity)
    - Add/update cart item
    - Store in session/database
- **Output:** Product added to cart confirmation
- **Feedback:** Cart icon badge shows item count

FR-SC-002: View Cart

- **Priority:** High
- **Description:** Display all items in shopping cart
- **Display:**
  - Product image thumbnail
  - Product name
  - Unit price
  - Quantity selector
  - Item subtotal
  - Remove item option
  - Continue shopping link
  - Proceed to checkout button
  - Cart summary (subtotal, estimated delivery, total)
- **Output:** Cart page with all items
- **Update:** Real-time total calculation

FR-SC-003: Update Cart Quantity

- **Priority:** High
- **Description:** Modify product quantity in cart
- **Input:** New quantity value
- **Process:**
  - Validate stock availability for new quantity
  - Update cart item quantity
  - Recalculate totals
  - Update display
- **Output:** Updated cart with new totals

- **Constraints:** Quantity must be positive integer, within stock limit

FR-SC-004: Remove from Cart

- **Priority:** High
- **Description:** Remove product from cart
- **Input:** Product ID to remove
- **Process:**
    - Confirm removal (optional)
    - Delete cart item
    - Recalculate totals
    - Update display
- **Output:** Item removed confirmation

FR-SC-005: Cart Persistence

- **Priority:** Medium
- **Description:** Save cart for logged-in users across sessions
- **Process:**
    - Store cart in database for logged-in users
    - Store cart in browser storage for guests
    - Merge guest cart with user cart on login
    - Auto-clear cart after 30 days of inactivity
- **Output:** Cart restored on login

FR-SC-006: Clear Cart

- **Priority:** Low
- **Description:** Remove all items from cart
- **Input:** Clear cart action
- **Process:**

    - Confirm action
    - Remove all cart items
    - Reset cart totals
- **Output:** Empty cart state

3.1.6 Checkout and Order Module

FR-CO-001: Checkout Process

- **Priority:** High
- **Description:** Complete order placement workflow
- **Steps:**
    1. Review cart items
    2. Select/add delivery address
    3. Review order summary
    4. Proceed to payment
    5. Payment confirmation
    6. Order placement
- **Input:** Cart items, delivery address, payment details
- **Process:**
    - Validate cart (stock availability)
    - Confirm delivery address
    - Calculate final total
    - Process payment
    - Create order
    - Reduce inventory
    - Generate order summary
    - Send notifications

- **Output:** Order confirmation page with order ID
- **No Minimum Order:** Orders of any value accepted

FR-CO-002: Delivery Address Selection

- **Priority:** High
- **Description:** Select or add delivery address during checkout
- **Input:** Saved address selection or new address entry
- **Process:**
    - Display saved addresses
    - Allow selection of default or other address
    - Allow adding new address on-the-fly
    - Validate address fields
    - Save new address to profile (optional)
- **Output:** Confirmed delivery address for order
- **Required Fields:** Name, Phone, House/Flat, Street, District, State, PIN Code

FR-CO-003: Order Summary Generation

- **Priority:** High
- **Description:** Generate order summary (bill)
- **Display:**
    - Order ID and Date
    - Customer details
    - Delivery address
    - Product details (name, quantity, price, subtotal)
    - Subtotal amount
    - Delivery charges (if applicable)

    - Total amount
    - Payment method
    - Order status
- **Output:** Order summary displayed and stored
- **Format:** HTML view with print option
- **Note:** No GST breakdown required

FR-CO-004: Payment Processing

- **Priority:** High
- **Description:** Process online payment for order
- **Input:** Payment method selection and details
- **Payment Methods:**
    - UPI (Google Pay, PhonePe, Paytm, etc.)
    - Debit/Credit Card
    - Net Banking
- **Process:**
    - Integrate with payment gateway
    - Redirect to payment page
    - Process payment securely
    - Receive payment status callback
    - Update order status based on payment result
- **Output:** Payment success/failure status
- **Security:** Use HTTPS, PCI-DSS compliant gateway
- **Error Handling:** Handle payment failures gracefully

FR-CO-005: Place Order

- **Priority:** High
- **Description:** Finalize order after successful payment
- **Input:** Payment confirmation
- **Process:**
    - Generate unique order ID
    - Save order to database
    - Reduce inventory for ordered items
    - Clear shopping cart
    - Generate order summary
    - Send confirmation notifications (customer & admin)
    - Update order status to "Order Placed"
- **Output:** Order confirmation with order ID and summary
- **Restrictions:**
    - No order cancellation allowed
    - No order modification allowed

3.1.7 Order Tracking and History Module

FR-OT-001: Track Order Status

- **Priority:** High
- **Description:** Customers can track their order status
- **Input:** Order ID or automatic display from order history
- **Order Statuses:**
    1. Order Placed
    2. Processing
    3. Shipped

4. Out for Delivery
5. Delivered

- **Display:**
  - Current status with timestamp
  - Status timeline/progress bar
  - Estimated delivery date
  - Order details
  - Delivery address
- **Output:** Order tracking page with status
- **Real-time:** Status updates reflected immediately

FR-OT-002: Order History

- **Priority:** High
- **Description:** View all past orders
- **Input:** Customer login credentials
- **Display:**
  - Order ID
  - Order date
  - Total amount
  - Order status
  - View details button
  - Reorder option
- **Process:**
  - Retrieve orders from database for logged-in customer
  - Sort by date (newest first)
  - Pagination for long lists

- **Output:** Order history list
- **Filter:** By date range, status

FR-OT-003: View Order Details

- **Priority:** High
- **Description:** View complete details of a specific order
- **Input:** Order ID
- **Display:**
    - Order summary
    - Product details
    - Delivery address
    - Payment details
    - Order status with timeline
    - Download/Print order summary option
- **Output:** Complete order details page
- **Access Control:** Only customer who placed order can view

FR-OT-004: Reorder

- **Priority:** Medium
- **Description:** Quickly reorder items from past order
- **Input:** Order ID (from order history)
- **Process:**
    - Retrieve products from selected order
    - Check current stock availability
    - Add available items to cart
    - Notify about unavailable items
    - Redirect to cart

- **Output:** Cart populated with order items
- **Note:** Prices may have changed since original order

3.1.8 Order Management Module (Admin)

FR-OM-001: View All Orders

- **Priority:** High
- **Description:** Admin can view all customer orders
- **Input:** Optional filters (status, date range, customer)
- **Display:**
  - Order ID
  - Customer name and phone
  - Order date
  - Total amount
  - Payment status
  - Order status
  - Action buttons (View Details, Update Status)
- **Process:**
  - Retrieve orders from database
  - Apply filters if specified
  - Sort and paginate
- **Output:** Orders list with pagination
- **Default View:** Show pending/processing orders first

FR-OM-002: View Order Details (Admin)

- **Priority:** High
- **Description:** Admin can view complete order details
- **Input:** Order ID
- **Display:**
  - All order information
  - Customer details
  - Product details
  - Delivery address
  - Payment information
  - Status history
  - Print order summary
- **Output:** Detailed order view
- **Actions:** Update status, contact customer, print

FR-OM-003: Update Order Status

- **Priority:** High
- **Description:** Admin can update order status
- **Input:** Order ID, New Status
- **Process:**
  - Validate status transition (logical flow)
  - Update order status in database
  - Record timestamp
  - Send notification to customer
- **Output:** Status updated confirmation
- **Notification:** Customer notified via SMS/Email

FR-OM-004: Search Orders

- **Priority:** Medium
- **Description:** Search orders by various criteria
- **Input:** Search criteria
- **Search By:**
  - Order ID
  - Customer name/phone
  - Date range
  - Product name
  - Order status
- **Output:** Matching orders list

FR-OM-005: Order Processing Dashboard

- **Priority:** High
- **Description:** Admin dashboard for order management
- **Display:**
  - New orders count (requires attention)
  - Orders by status (pie chart)
  - Today's order count and revenue
  - Pending orders list
  - Recent orders
- **Output:** Visual dashboard
- **Refresh:** Auto-refresh every 5 minutes

FR-OM-006: Bulk Status Update

- **Priority:** Low
- **Description:** Update status for multiple orders at once

- **Input:** Multiple order IDs, new status
- **Process:**
    - Select multiple orders
    - Apply same status to all
    - Validate each transition
    - Update database
    - Send notifications
- **Output:** Bulk update confirmation

3.1.9 Customer Management Module (Admin)

FR-CM-001: View Customer List

- **Priority:** High
- **Description:** Admin can view all registered customers
- **Display:**
    - Customer ID
    - Name
    - Email
    - Phone
    - Registration date
    - Total orders
    - Total purchase value
    - Last order date
    - Status (Active/Inactive)
- **Process:**
    - Retrieve customer data
    - Calculate statistics

    - Sort and filter options
    - Pagination
- **Output:** Customer list with details

FR-CM-002: View Customer Details

- **Priority:** High
- **Description:** View complete customer profile and history
- **Input:** Customer ID
- **Display:**
    - Personal information
    - Contact details
    - Saved addresses
    - Order history
    - Total orders and revenue
    - Purchase patterns
- **Output:** Detailed customer profile
- **Privacy:** Passwords not visible

FR-CM-003: Search Customers

- **Priority:** Medium
- **Description:** Search customers by various criteria
- **Input:** Search query
- **Search By:**
    - Name
    - Email
    - Phone number

    - Customer ID
- **Output:** Matching customers list

FR-CM-004: Customer Segmentation

- **Priority:** Medium
- **Description:** Categorize customers based on purchase behavior
- **Categories:**
    - Regular customers (ordered 5+ times)
    - New customers (first order within 30 days)
    - Inactive customers (no order in 90+ days)
    - High-value customers (total purchase > threshold)
- **Output:** Segmented customer lists
- **Use Case:** Targeted notifications for new products

FR-CM-005: Export Customer Data

- **Priority:** Low
- **Description:** Export customer list for analysis
- **Input:** Filter criteria (optional)
- **Process:**
    - Generate customer report
    - Format data
    - Export to Excel/CSV
- **Output:** Downloadable customer data file
- **Privacy:** Exclude sensitive data (passwords)

3.1.10 Notification System

FR-NS-001: Customer Order Confirmation

- **Priority:** High
- **Description:** Send confirmation when order is placed
- **Trigger:** Order placement successful
- **Recipients:** Customer (Email & SMS)
- **Content:**
  - Order ID
  - Order summary (items, total)
  - Delivery address
  - Estimated delivery date
  - Track order link
- **Timing:** Immediate (within 1 minute)

FR-NS-002: Admin New Order Notification

- **Priority:** High
- **Description:** Notify admin when new order is received
- **Trigger:** New order placed
- **Recipients:** Admin (Email & SMS)
- **Content:**
  - Order ID
  - Customer name and contact
  - Order value
  - Quick view link to admin panel
- **Timing:** Immediate (within 1 minute)

FR-NS-003: Order Status Update Notification

- **Priority:** High
- **Description:** Notify customer on order status change
- **Trigger:** Admin updates order status
- **Recipients:** Customer (Email & SMS)
- **Content:**
  - Order ID
  - Updated status
  - Expected next step
  - Track order link
- **Timing:** Immediate on status change

FR-NS-004: New Product Notification

- **Priority:** Medium
- **Description:** Notify regular customers when new products added
- **Trigger:** Admin adds new product
- **Recipients:** Regular customers (5+ orders)
- **Content:**
  - New product name and image
  - Brief description
  - Price
  - View product link
- **Timing:** Within 24 hours (batch notification)
- **Frequency:** Max once per week per customer

FR-NS-005: Low Stock Alert (Admin)

- **Priority:** High
- **Description:** Alert admin when product stock is low
- **Trigger:** Stock falls below threshold
- **Recipients:** Admin (Email & Dashboard notification)
- **Content:**
  - Product name
  - Current stock level
  - Threshold level
  - Restock action link
- **Timing:** Immediate when threshold crossed

FR-NS-006: Notification Preferences

- **Priority:** Low
- **Description:** Customers can manage notification preferences
- **Options:**
  - Email notifications (On/Off)
  - SMS notifications (On/Off)
  - New product notifications (On/Off)
  - Order updates (Always on, cannot disable)
- **Output:** Preferences saved

FR-NS-007: Notification History

- **Priority:** Low
- **Description:** View history of sent notifications
- **Display:**
  - Date and time

    - Recipient
    - Notification type
    - Status (Sent/Failed)
- **Access:** Admin only
- **Output:** Notification log

3.1.11 Reporting and Analytics Module

FR-RA-001: Daily Sales Report

- **Priority:** High
- **Description:** Generate daily sales summary
- **Input:** Date (default: today)
- **Display:**
    - Total orders
    - Total revenue
    - Products sold (quantity)
    - Average order value
    - Payment method breakdown
    - Hourly sales trend
- **Output:** Daily sales report
- **Export:** PDF, Excel

FR-RA-002: Monthly Revenue Report

- **Priority:** High
- **Description:** Generate monthly revenue analysis
- **Input:** Month and Year
- **Display:**
    - Total revenue

    - Total orders
    - Day-wise revenue graph
    - Top-selling products
    - Revenue by category
    - Growth comparison with previous month
- **Output:** Monthly revenue report
- **Export:** PDF, Excel

FR-RA-003: Product-wise Sales Report

- **Priority:** High
- **Description:** Analyze sales by product
- **Input:** Date range, Product filter (optional)
- **Display:**
    - Product name
    - Units sold
    - Revenue generated
    - Percentage of total sales
    - Rank by sales volume
    - Sales trend graph
- **Output:** Product sales analysis report
- **Sort:** By revenue, quantity, or product name
- **Export:** PDF, Excel

FR-RA-004: District-wise Order Report

- **Priority:** High
- **Description:** Analyze orders by delivery district
- **Input:** Date range

- **Display:**
  - District name
  - Number of orders
  - Total revenue
  - Average order value
  - Top products per district
  - Geographic distribution map (optional)
- **Output:** District-wise analysis report
- **Use Case:** Identify high-demand areas
- **Export:** PDF, Excel

FR-RA-005: Customer Report

- **Priority:** High
- **Description:** Customer analytics and insights
- **Input:** Date range, Customer segment filter (optional)
- **Display:**
  - Total customers
  - New customers
  - Regular customers
  - Customer retention rate
  - Top customers by purchase value
  - Customer lifetime value
  - Purchase frequency distribution
- **Output:** Customer analytics report
- **Export:** PDF, Excel

FR-RA-006: Inventory Report

- **Priority:** Medium
- **Description:** Current inventory status report
- **Display:**
    - Product name
    - Current stock
    - Stock value
    - Low stock items highlighted
    - Out of stock items
    - Stock movement (last 30 days)
    - Inventory turnover ratio
- **Output:** Inventory status report
- **Export:** PDF, Excel

FR-RA-007: Admin Dashboard

- **Priority:** High
- **Description:** Comprehensive admin analytics dashboard
- **Display:**
    - Key metrics cards (total sales, orders, customers, revenue)
    - Today's statistics
    - Sales trend graph (7/30 days)
    - Top products chart
    - Recent orders table
    - Low stock alerts
    - Order status distribution
    - Revenue by category pie chart
- **Output:** Interactive dashboard

- **Refresh:** Auto-refresh every 5 minutes
- **Date Range:** Selectable (Today, 7 days, 30 days, Custom)

FR-RA-008: Export All Reports

- **Priority:** Medium
- **Description:** All reports exportable to PDF and Excel
- **Process:**
    - Generate report in selected format
    - Include charts and graphs
    - Maintain formatting
    - Add report header (title, date range, generated date)
- **Output:** Downloadable file
- **Formats:** PDF (for viewing/printing), Excel (for further analysis)

3.2 Non-Functional Requirements

3.2.1 Performance Requirements

NFR-PR-001: Response Time

- Page load time: < 3 seconds (normal network conditions)
- Search results: < 2 seconds
- Add to cart action: < 1 second
- Order placement: < 5 seconds
- Report generation: < 10 seconds
- API response time: < 500ms for 95% of requests

NFR-PR-002: Concurrent Users

- System must support up to 500 concurrent users
- Peak load capacity: 1000 concurrent users (with degraded performance acceptable)
- Database connections: Pool size minimum 50

NFR-PR-003: Scalability

- System architecture should support horizontal scaling
- Database should handle up to 5,000 users
- Support up to 500 products in catalog
- Handle 50-100 orders per day
- Growth capacity: 50% increase annually

NFR-PR-004: Database Performance

- Query execution time: < 100ms for simple queries
- Complex report queries: < 5 seconds
- Database indexing for frequently accessed fields
- Connection pooling for optimal resource usage

3.2.2 Security Requirements

NFR-SR-001: Authentication

- Secure password storage using bcrypt or similar hashing (minimum 10 rounds)
- JWT-based session management
- Token expiry: 24 hours for customers, 8 hours for admin
- Session timeout: 30 minutes of inactivity
- Account lockout: After 5 failed login attempts (15-minute lockout)

NFR-SR-002: Authorization

- Role-based access control (Admin, Customer)
- Admin actions restricted to authenticated admin users
- Customers can only access their own data
- API endpoints protected with authentication middleware

NFR-SR-003: Data Protection

- All passwords encrypted (hashed) before storage
- Sensitive data encrypted in database (payment details if stored)
- HTTPS/TLS encryption for all data transmission
- Payment processing via PCI-DSS compliant gateway
- No storage of complete card details

NFR-SR-004: Input Validation

- Server-side validation for all inputs
- Protection against SQL injection (use parameterized queries)
- Protection against XSS attacks (sanitize user inputs)
- CSRF tokens for state-changing operations
- File upload validation (type, size, content)

NFR-SR-005: Privacy

- Customer data accessible only by authorized personnel
- Passwords never displayed or transmitted in plain text
- Payment information handled only by payment gateway
- Comply with data protection regulations
- Secure deletion of customer data on request

NFR-SR-006: Audit Trail

- Log all admin actions (who, what, when)
- Log authentication attempts (success and failures)
- Log critical transactions (orders, payments, stock changes)
- Logs retained for minimum 1 year
- Access to logs restricted to admin only

3.2.3 Usability Requirements

NFR-UR-001: User Interface

- Clean, intuitive, and modern design
- Consistent navigation across all pages
- Clear visual hierarchy and typography
- Accessible color scheme (WCAG AA compliance)
- Clear error messages and guidance
- Minimal clicks to complete tasks (max 3-4 clicks for common actions)

NFR-UR-002: Responsiveness

- Fully responsive design (mobile-first approach)
- Support for devices:
    - Mobile phones (320px - 767px)
    - Tablets (768px - 1024px)
    - Desktops (1025px and above)
- Touch-friendly elements (minimum 44x44px touch targets)
- Consistent experience across all device sizes

NFR-UR-003: Browser Compatibility

- Support for modern browsers:
  - Google Chrome (latest 2 versions)
  - Mozilla Firefox (latest 2 versions)
  - Safari (latest 2 versions)
  - Microsoft Edge (latest 2 versions)
- Graceful degradation for older browsers
- No dependency on specific browser plugins

NFR-UR-004: Accessibility

- Keyboard navigation support
- Screen reader compatible (ARIA labels)
- Alt text for all images
- Proper heading hierarchy
- Sufficient color contrast
- Focus indicators visible

NFR-UR-005: Learning Curve

- New users should be able to browse and order within 5 minutes
- Admin training should take less than 2 hours
- Contextual help available where needed
- Clear labels and instructions

NFR-UR-006: Multilingual Support (Future)

- System architecture should support multiple languages
- Initial version in English
- Potential for Tamil language support

3.2.4 Reliability Requirements

NFR-RR-001: Availability

- System uptime: 99.5% (approximately 3.65 hours downtime per month)
- Planned maintenance: During off-peak hours (announced in advance)
- Unplanned downtime: < 1% annually

NFR-RR-002: Error Handling

- Graceful degradation on component failures
- User-friendly error messages (no technical jargon)
- Automatic error logging
- Fallback mechanisms for critical operations
- Retry logic for network failures

NFR-RR-003: Data Integrity

- Database transactions for critical operations (order placement)
- Backup mechanism for data recovery
- Referential integrity maintained
- Validation at multiple layers (client, server, database)

NFR-RR-004: Backup and Recovery

- Automated daily database backups
- Backup retention: 30 days minimum
- Recovery time objective (RTO): < 4 hours
- Recovery point objective (RPO): < 24 hours
- Regular backup restoration testing (monthly)

3.2.5 Maintainability Requirements

NFR-MR-001: Code Quality

- Well-documented code (inline comments for complex logic)
- Follow coding standards and conventions
- Modular and reusable components
- Code review before deployment
- Unit test coverage: minimum 70% for critical modules

NFR-MR-002: Documentation

- Complete API documentation
- Database schema documentation
- User manuals (admin and customer)
- Installation and deployment guide
- Troubleshooting guide

NFR-MR-003: Logging

- Comprehensive logging framework
- Log levels: DEBUG, INFO, WARN, ERROR
- Application logs separate from server logs
- Log rotation to manage disk space
- Centralized log management

NFR-MR-004: Monitoring

- Server resource monitoring (CPU, memory, disk)
- Application performance monitoring
- Database performance monitoring
- Error rate monitoring
- Automated alerts for critical issues

3.2.6 Portability Requirements

NFR-PR-001: Platform Independence

- Run on any OS supporting Java (Windows, Linux, macOS)
- Web-based access (no client installation required)
- Database portable (MongoDB standard installation)

NFR-PR-002: Deployment

- Containerization support (Docker - optional)
- Standard Apache Tomcat deployment
- Environment-specific configuration files
- Minimal manual configuration required

---

4. External Interface Requirements

4.1 User Interfaces

4.1.1 Customer Application UI

Home Page

- Header: Logo, Search bar, Cart icon, Login/Profile
- Hero section with featured products/offers
- Product categories grid
- Featured products section
- Footer: Links, Contact info, Social media

Product Listing Page

- Sidebar: Category filter, Price filter, Availability filter
- Main area: Product grid (image, name, price, stock status)
- Sorting options
- Pagination controls

Product Detail Page

- Product image gallery (zoomable)
- Product name, SKU, price
- Detailed description
- Quantity selector
- Add to Cart and Buy Now buttons
- Stock availability indicator
- Related products section

Shopping Cart Page

- Cart items table (image, name, price, quantity, subtotal, remove)
- Quantity update controls
- Cart summary (subtotal, delivery, total)
- Continue shopping link
- Proceed to checkout button

Checkout Page

- Order summary (read-only cart items)
- Delivery address section (select existing or add new)
- Order total breakdown
- Proceed to payment button

Order Confirmation Page

- Order success message
- Order ID and date
- Order summary
- Estimated delivery date
- Download/Print order summary button

- Track order button
- Continue shopping button

My Account Pages

- Profile: View/Edit personal information
- Addresses: Manage delivery addresses
- Orders: Order history and tracking
- Notifications: Notification preferences

Order Tracking Page

- Order status timeline
- Current status details
- Order information
- Delivery address
- Contact support option

4.1.2 Admin Application UI

Admin Login Page

- Login form (username, password)
- Forgot password link
- Secure authentication indicators

Admin Dashboard

- Top metrics cards (Sales, Orders, Customers, Revenue)
- Sales trend graph
- Order status distribution chart
- Recent orders table
- Low stock alerts panel
- Quick action buttons

Product Management Pages

- Product list table with search, filter, pagination
- Add product form (all fields, image upload)
- Edit product form (pre-filled fields)
- Delete confirmation modal
- Category management section

Inventory Management Pages

- Stock overview table
- Add stock form
- Low stock alerts list
- Stock history report
- Inventory value summary

Order Management Pages

- Orders list with filters (status, date, customer)
- Order details view (all information)
- Update status dropdown
- Search orders functionality
- Print order summary button

Customer Management Pages

- Customer list table with search
- Customer details view (profile, order history)
- Customer segmentation tabs
- Export customer data button

Reports Section

- Report type selector (dropdown)

- Date range picker
- Filter options (varies by report type)
- Generate report button
- Report display area with charts/tables
- Export buttons (PDF, Excel)

Notification Management

- Notification history table
- Send notification form (for manual notifications)
- Notification settings (configure thresholds, templates)

4.2 Hardware Interfaces

Server Requirements:

- Processor: Minimum Dual-Core 2.0 GHz (Recommended: Quad-Core 2.5 GHz+)
- RAM: Minimum 4 GB (Recommended: 8 GB+)
- Storage: Minimum 50 GB SSD (Recommended: 100 GB SSD)
- Network: 100 Mbps internet connection

Client Requirements:

- Any device with web browser (smartphone, tablet, desktop)
- Minimum screen resolution: 320px width
- Touch screen support for mobile devices
- Internet connection: Minimum 2 Mbps

4.3 Software Interfaces

4.3.1 Backend Framework

- **Component:** Java Servlets / Spring Boot
- **Version:** Spring Boot 2.7+ or Java EE 8+

- **Purpose:** Application business logic, API endpoints, request handling
- **Data Exchange:** JSON format for API requests/responses

4.3.2 Database System

- **Component:** MongoDB
- **Version:** 6.0+
- **Purpose:** Data persistence (products, orders, customers, etc.)
- **Connection:** MongoDB Java Driver
- **Data Format:** BSON (JSON-like documents)

4.3.3 Web Server

- **Component:** Apache Tomcat
- **Version:** 9.0+
- **Purpose:** Deploy and run Java web application
- **Configuration:** Connection pooling, session management

4.3.4 Payment Gateway

- **Component:** Third-party payment gateway (Razorpay, Paytm, or similar)
- **Purpose:** Process online payments (UPI, Cards, Net Banking)
- **Integration:** REST API
- **Data Exchange:** JSON over HTTPS
- **Security:** PCI-DSS compliant, webhook for payment confirmation

4.3.5 SMS Service

- **Component:** SMS Gateway API (Twilio, MSG91, or similar)
- **Purpose:** Send SMS notifications to customers and admin
- **Integration:** REST API
- **Data Exchange:** JSON over HTTPS

- **Functionality:** OTP, Order confirmations, Status updates

4.3.6 Email Service

- **Component:** SMTP server or Email API (SendGrid, AWS SES, or similar)
- **Purpose:** Send email notifications
- **Integration:** SMTP protocol or REST API
- **Functionality:** Order confirmations, Notifications, Password reset

4.3.7 Cloud Storage (Optional)

- **Component:** AWS S3, Google Cloud Storage, or similar
- **Purpose:** Store product images and generated reports
- **Integration:** SDK or REST API
- **Alternative:** Local file system storage

4.4 Communication Interfaces

4.4.1 HTTP/HTTPS Protocol

- All client-server communication via HTTPS (TLS 1.2+)
- RESTful API architecture
- JSON data format for API requests and responses
- Standard HTTP methods (GET, POST, PUT, DELETE)
- Status codes for response indication

4.4.2 API Endpoints Structure

Base URL: https://pappadshop.com/api/v1

Customer APIs:

- POST /auth/register - Customer registration

- POST /auth/login - Customer login

- POST /auth/forgot-password - Password reset request

- GET /products - Get product list
- GET /products/{id} - Get product details
- POST /cart - Add to cart
- GET /cart - Get cart items
- PUT /cart/{id} - Update cart item
- DELETE /cart/{id} - Remove from cart
- POST /orders - Place order
- GET /orders - Get order history
- GET /orders/{id} - Get order details
- GET /profile - Get customer profile
- PUT /profile - Update profile

Admin APIs:

- POST /admin/auth/login - Admin login
- GET /admin/products - Get all products
- POST /admin/products - Add product
- PUT /admin/products/{id} - Update product
- DELETE /admin/products/{id} - Delete product
- GET /admin/orders - Get all orders
- PUT /admin/orders/{id} - Update order status
- GET /admin/customers - Get customer list
- GET /admin/inventory - Get inventory details
- POST /admin/inventory/add - Add stock
- GET /admin/reports/daily - Daily sales report
- GET /admin/reports/monthly - Monthly revenue report

4.4.3 WebSocket (Optional)

- Real-time order notifications for admin
- Real-time stock updates
- Live dashboard refresh

4.4.4 Network Security

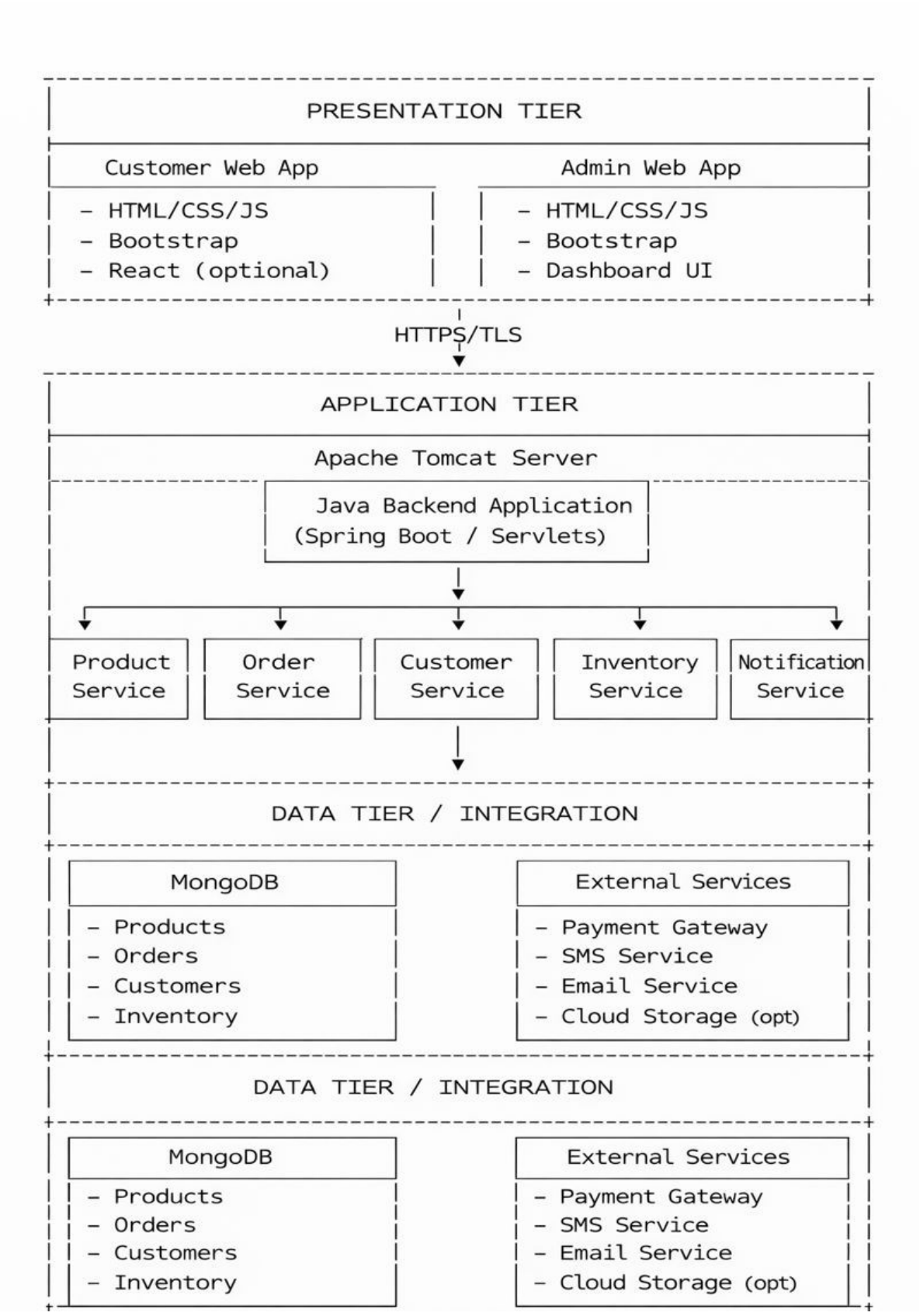
- All communication encrypted (HTTPS/TLS)
- API authentication via JWT tokens
- CORS policy for API access control
- Rate limiting to prevent abuse (100 requests/minute per IP)

---

5. System Architecture

5.1 Architecture Overview

The Pappad Shop Management System follows a **three-tier architecture**:



5.2 Component Description

Presentation Tier:

- Customer-facing web application (HTML, CSS, JavaScript, Bootstrap)
- Admin web application with dashboard interface
- Responsive design for all devices
- Client-side form validation
- AJAX calls to backend APIs

Application Tier:

- Java backend (Spring Boot or Servlets)
- RESTful API endpoints
- Business logic implementation
- Service layer architecture:
    - **Product Service:** Manage products, categories
    - **Order Service:** Order processing, order management
    - **Customer Service:** User authentication, profile management
    - **Inventory Service:** Stock tracking, inventory updates
    - **Payment Service:** Payment gateway integration
    - **Notification Service:** SMS/Email notifications
- Authentication and authorization middleware
- Error handling and logging

Data Tier:

- MongoDB database for data persistence
- Collections for Products, Orders, Customers, Inventory, etc.
- Indexes for optimized queries
- Data validation at database level

External Integrations:

- Payment Gateway API integration
- SMS Gateway API integration
- Email Service API integration
- Optional cloud storage for images

5.3 Design Patterns

1. MVC (Model-View-Controller)

- Model: Data models (Product, Order, Customer)
- View: Web pages (HTML/JSP templates)
- Controller: Backend controllers handling requests

2. Service Layer Pattern

- Business logic encapsulated in service classes
- Controllers delegate to services
- Services interact with data access layer

3. Repository Pattern

- Data access abstraction
- MongoDB operations encapsulated in repository classes
- CRUD operations standardized

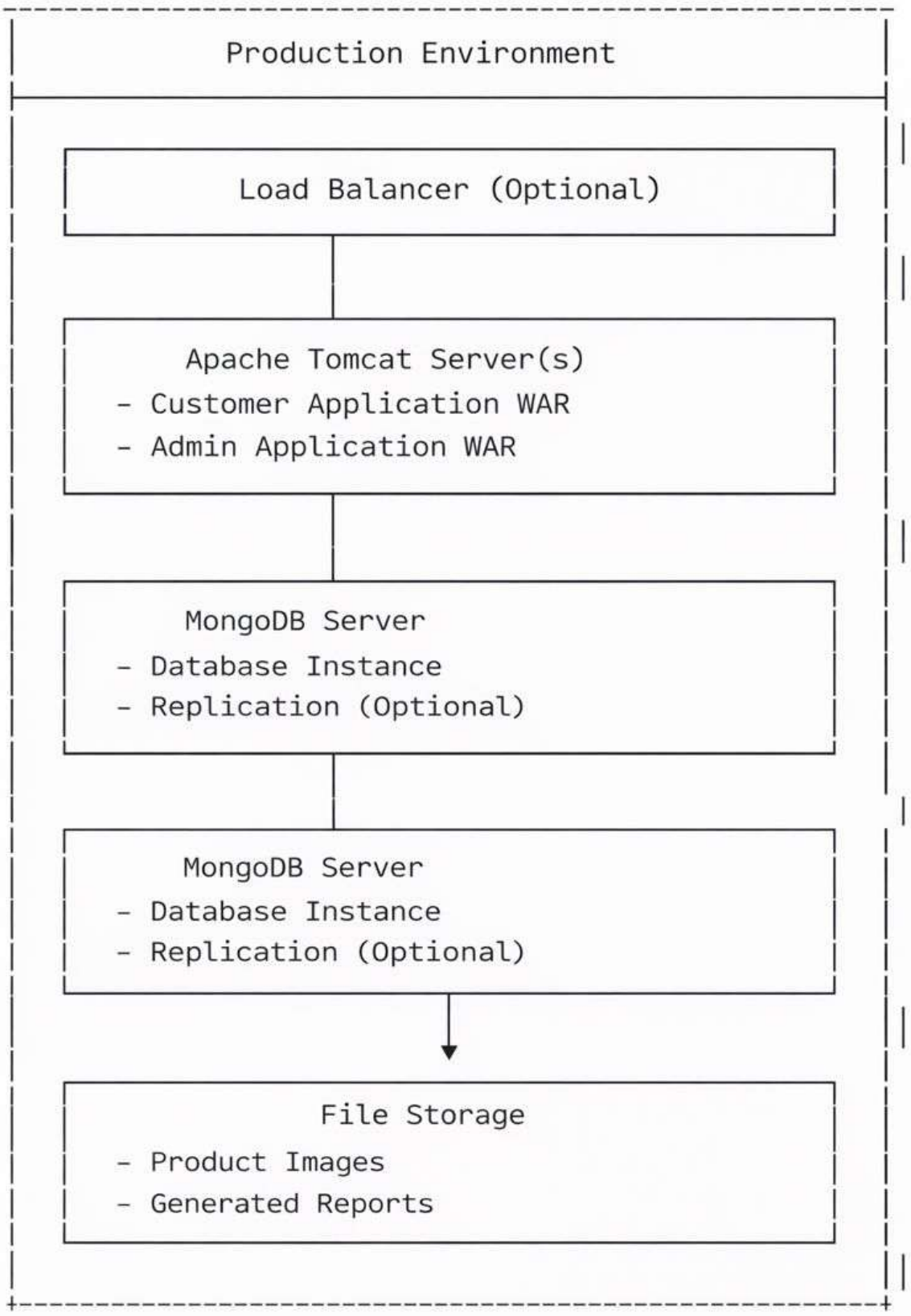
4. Singleton Pattern

- Database connection pool
- Configuration manager

5. Factory Pattern

- Service object creation
- Payment gateway instances

5.4 Deployment Architecture



6. Database Requirements

6.1 Database Schema

Collection 1: users

```
{
  _id: ObjectId,
  email: String (unique, required),
  password: String (hashed, required),
  role: String (enum: ['customer', 'admin'], required),
  name: String (required),
  phone: String (required, 10 digits),
  addresses: [
    {
      addressId: String,
      label: String (e.g., "Home", "Office"),
      houseFlat: String,
      street: String,
      district: String,
      state: String,
      pinCode: String (6 digits),
      isDefault: Boolean
    }
  ],
  isActive: Boolean (default: true),
```

emailVerified: Boolean (default: false),

phoneVerified: Boolean (default: false),

orderCount: Number (default: 0),

totalSpent: Number (default: 0),

lastOrderDate: Date,

preferences: {

emailNotifications: Boolean (default: true),

smsNotifications: Boolean (default: true),

newProductNotifications: Boolean (default: true)

},

createdAt: Date,

updatedAt: Date

}

Indexes:

- email (unique)
- phone
- role
- createdAt

Collection 2: products

{

_id: ObjectId,

sku: String (unique, required),

name: String (required),

category: String (required),

```
  description: String (required),
  shortDescription: String,
  price: Number (required, positive),
  unit: String (enum: ['kg', 'packet', 'piece'], required),
  images: [
   {
    url: String,
    isPrimary: Boolean,
    order: Number
   }
  ],
  stockQuantity: Number (required, non-negative),
  lowStockThreshold: Number (default: 10),
  isActive: Boolean (default: true),
  featured: Boolean (default: false),
  specifications: Object, // Additional product specs
  tags: [String], // For search optimization
  viewCount: Number (default: 0),
  salesCount: Number (default: 0),
  createdAt: Date,
  updatedAt: Date,
  createdBy: ObjectId (ref: users),
  updatedBy: ObjectId (ref: users)
}
```

Indexes:

- sku (unique)
- name (text index)
- category
- price
- isActive
- featured
- createdAt

Collection 3: categories

```
{
  _id: ObjectId,
  name: String (unique, required),
  description: String,
  displayOrder: Number,
  imageUrl: String,
  isActive: Boolean (default: true),
  productCount: Number (default: 0),
  createdAt: Date,
  updatedAt: Date
}
```

Indexes:

- name (unique)
- displayOrder

Collection 4: orders

```
{
  _id: ObjectId,
  orderId: String (unique, auto-generated, e.g., "ORD20260127001"),
  customerId: ObjectId (ref: users, required),
  customerDetails: {
    name: String,
    email: String,
    phone: String
  },
  items: [
    {
      productId: ObjectId (ref: products),
      productName: String,
      sku: String,
      price: Number,
      quantity: Number,
      unit: String,
      subtotal: Number
    }
  ],
  deliveryAddress: {
    name: String,
    phone: String,
    houseFlat: String,
    street: String,
```

```
  district: String,
  state: String,
  pinCode: String
 },
 orderSummary: {
  subtotal: Number,
  deliveryCharge: Number (default: 0),
  discount: Number (default: 0),
  totalAmount: Number
 },
 payment: {
  method: String (enum: ['UPI', 'Card', 'NetBanking']),
  transactionId: String,
  status: String (enum: ['Pending', 'Completed', 'Failed']),
  paidAt: Date,
  gatewayResponse: Object
 },
 status: String (enum: ['Order Placed', 'Processing', 'Shipped', 'Out for Delivery', 'Delivered'], default: 'Order Placed'),
 statusHistory: [
  {
   status: String,
   timestamp: Date,
   updatedBy: ObjectId (ref: users),
   notes: String
```

```
  }
 ],
 estimatedDeliveryDate: Date,
 actualDeliveryDate: Date,
 notes: String,
 createdAt: Date,
 updatedAt: Date
}
```

Indexes:

- orderId (unique)
- customerId
- status
- createdAt (descending)
- deliveryAddress.district

Collection 5: cart

```
{
 _id: ObjectId,
 customerId: ObjectId (ref: users, required),
 items: [
  {
   productId: ObjectId (ref: products, required),
   quantity: Number (required, positive),
   addedAt: Date
  }
```

],

updatedAt: Date,

expiresAt: Date (TTL index for auto-cleanup)

}

Indexes:

- customerId (unique)

- expiresAt (TTL index)

Collection 6: inventory

{

_id: ObjectId,

productId: ObjectId (ref: products, unique, required),

currentStock: Number (required),

lowStockThreshold: Number,

transactions: [

{

transactionId: String,

type: String (enum: ['Addition', 'Reduction', 'Adjustment']),

quantity: Number,

balanceAfter: Number,

reason: String (enum: ['Purchase', 'Sale', 'Return', 'Damage', 'Manual Adjustment']),

referenceId: ObjectId, // OrderId or PurchaseId

performedBy: ObjectId (ref: users),

notes: String,

timestamp: Date

}

],

lastRestocked: Date,

updatedAt: Date

}

Indexes:

- productId (unique)

- transactions.timestamp

Collection 7: notifications

{

_id: ObjectId,

recipientId: ObjectId (ref: users, required),

recipientEmail: String,

recipientPhone: String,

type: String (enum: ['Order_Confirmation', 'Order_Status', 'New_Product', 'Low_Stock', 'General']),

channel: String (enum: ['Email', 'SMS', 'Both']),

subject: String,

message: String (required),

metadata: Object, // Additional data (orderId, productId, etc.)

status: String (enum: ['Pending', 'Sent', 'Failed'], default: 'Pending'),

sentAt: Date,

failureReason: String,

retryCount: Number (default: 0),

createdAt: Date

}

Indexes:

- recipientId
- type
- status
- createdAt

Collection 8: reports

{

_id: ObjectId,

reportType: String (enum: ['Daily_Sales', 'Monthly_Revenue', 'Product_Sales', 'District_Orders', 'Customer_Report', 'Inventory']),

generatedBy: ObjectId (ref: users),

parameters: {

dateFrom: Date,

dateTo: Date,

filters: Object // Report-specific filters

},

data: Object, // Computed report data

fileUrl: String, // If exported to file

status: String (enum: ['Generating', 'Completed', 'Failed']),

generatedAt: Date,

expiresAt: Date (for auto-cleanup)

}

Indexes:

- reportType
- generatedBy
- generatedAt
- expiresAt (TTL index)

Collection 9: sessions

{
  _id: ObjectId,
  userId: ObjectId (ref: users, required),
  token: String (unique, required),
  ipAddress: String,
  userAgent: String,
  isActive: Boolean (default: true),
  lastActivity: Date,
  expiresAt: Date,
  createdAt: Date
}

Indexes:

- token (unique)
- userId
- expiresAt (TTL index)

Collection 10: auditLogs

{

_id: ObjectId,

userId: ObjectId (ref: users),

userEmail: String,

action: String (required), // e.g., "Product_Added", "Order_Placed", "Stock_Updated"

module: String (enum: ['Auth', 'Product', 'Order', 'Inventory', 'Customer', 'Report']),

details: Object, // Action-specific details

ipAddress: String,

userAgent: String,

timestamp: Date (default: current time)

}

Indexes:

- userId

- action

- module

- timestamp

6.2 Data Validation Rules

At Database Level:

- Required fields enforced
- Data type validation
- Enum value validation
- Unique constraints on email, phone, sku, orderId
- Positive number validation for prices and quantities

At Application Level:

- Email format validation
- Phone number format (10 digits)
- Password strength (min 8 chars, uppercase, lowercase, number)
- PIN code format (6 digits)
- Image file type and size validation
- Stock quantity non-negative
- Price positive value

6.3 Database Optimization

Indexing Strategy:

- Primary indexes on frequently queried fields (email, phone, orderId, productId)
- Compound indexes for common query patterns (status + createdAt, category + isActive)
- Text indexes for search functionality (product name, description)
- TTL indexes for automatic cleanup (sessions, expired reports, old notifications)

Query Optimization:

- Use projection to fetch only required fields
- Pagination for large result sets
- Aggregation pipelines for complex reports
- Caching frequently accessed data (product catalog)

Data Archival:

- Archive old orders (older than 2 years) to separate collection
- Periodic cleanup of old notifications (older than 90 days)
- Compress old audit logs

7. Security Requirements

7.1 Authentication Security

Password Policy:

- Minimum 8 characters
- Must contain at least one uppercase letter
- Must contain at least one lowercase letter
- Must contain at least one number
- Optional: Special character requirement
- Password hashing using bcrypt (minimum 10 rounds)
- No password stored in plain text

Session Management:

- JWT-based authentication
- Token stored securely (HttpOnly cookies or localStorage with XSS protection)
- Token expiry: 24 hours for customers, 8 hours for admin
- Automatic token refresh mechanism
- Session invalidation on logout
- Concurrent session limit: 3 per user

Account Security:

- Account lockout after 5 consecutive failed login attempts
- Lockout duration: 15 minutes
- CAPTCHA after 3 failed attempts
- Email/SMS notification on suspicious login activity
- Password reset via OTP (6-digit, valid for 10 minutes)

- Force password change on first login (optional for admin)

7.2 Authorization and Access Control

Role-Based Access Control (RBAC):

- Two roles: Admin, Customer
- Admin: Full access to admin application, all data, all operations
- Customer: Access to own data only (profile, orders, cart)
- API endpoints protected with role-based middleware
- Unauthorized access returns HTTP 403 Forbidden

Data Access Rules:

- Customers can view/edit only their own profile and orders
- Admin can view all customer data but cannot modify passwords
- Payment details viewable only by owning customer (masked)
- Audit logs accessible only to admin

7.3 Data Security

Encryption:

- All communication via HTTPS/TLS 1.2+
- Passwords hashed using bcrypt
- Sensitive data encrypted at rest (payment info if stored)
- Database connection encrypted (MongoDB SSL/TLS)
- API keys and secrets stored in environment variables, not in code

Data Privacy:

- Comply with data protection regulations
- Customer data not shared with third parties (except payment gateway)
- Secure deletion of customer data on request
- Anonymize data in analytics/reports

- Regular privacy policy updates

Payment Security:

- PCI-DSS compliant payment gateway
- No storage of complete card details in database
- Only masked card numbers stored (last 4 digits)
- Payment gateway tokenization
- Secure callback/webhook verification

7.4 Input Validation and Sanitization

Server-Side Validation:

- All inputs validated before processing
- Data type checking
- Range and length validation
- Format validation (email, phone, PIN)
- Whitelist validation for enums

Protection Against Attacks:

- **SQL Injection:** Use parameterized queries (MongoDB prevents this by design)
- **XSS (Cross-Site Scripting):** Sanitize all user inputs, escape outputs, Content Security Policy
- **CSRF (Cross-Site Request Forgery):** CSRF tokens for state-changing operations
- **File Upload Attacks:** Validate file type, size, scan for malware, store in secure location
- **Brute Force:** Rate limiting, CAPTCHA, account lockout
- **Session Hijacking:** Secure session tokens, HTTPS only, HttpOnly cookies

- **Clickjacking:** X-Frame-Options header

7.5 API Security

Authentication:

- JWT tokens in Authorization header
- Token validation on every request
- Token blacklisting on logout

Rate Limiting:

- 100 requests per minute per IP address
- 1000 requests per hour per authenticated user
- Stricter limits for sensitive operations (login: 5/minute)

CORS Policy:

- Whitelist allowed origins
- Restrict HTTP methods
- Credentials handling

API Security Headers:

- X-Content-Type-Options: nosniff
- X-Frame-Options: DENY
- Content-Security-Policy
- Strict-Transport-Security (HSTS)

7.6 Logging and Monitoring

Security Logging:

- Log all authentication attempts (success and failure)
- Log all authorization failures
- Log all data modifications (who, what, when)
- Log API access (endpoint, user, timestamp, IP)

- Log file uploads and downloads
- Secure log storage (restricted access)

Security Monitoring:

- Real-time monitoring for suspicious activities
- Automated alerts for:
    - Multiple failed login attempts
    - Unauthorized access attempts
    - Unusual data access patterns
    - High volume of requests from single IP
    - Payment failures
- Regular security audit reviews

7.7 Compliance and Best Practices

Security Best Practices:

- Keep all software dependencies up-to-date
- Regular security patches and updates
- Secure configuration management
- Principle of least privilege
- Regular security audits and penetration testing
- Security training for development team
- Incident response plan

Compliance:

- Data protection regulations (GDPR, local laws)
- PCI-DSS for payment processing
- Secure coding standards (OWASP)
- Regular compliance audits

8. Quality Attributes

8.1 Reliability

Availability:

- Target uptime: 99.5%
- Maximum unplanned downtime: 3.65 hours/month
- Planned maintenance window: Off-peak hours (announced in advance)

Fault Tolerance:

- Graceful error handling
- Automatic retry for transient failures
- Fallback mechanisms for external service failures
- Database connection pooling with reconnection logic

Data Integrity:

- ACID transactions for critical operations
- Referential integrity maintained
- Data validation at multiple layers
- Regular automated backups

Recovery:

- Automated daily backups (retained for 30 days)
- Recovery Time Objective (RTO): < 4 hours
- Recovery Point Objective (RPO): < 24 hours
- Disaster recovery plan documented and tested

8.2 Maintainability

Code Quality:

- Follow Java coding standards
- Modular and reusable code
- Separation of concerns (MVC pattern)
- Comprehensive inline documentation
- Code review process before deployment

Documentation:

- API documentation (endpoints, parameters, responses)
- Database schema documentation
- Architecture documentation
- Deployment guide
- User manuals (admin and customer)
- Troubleshooting guide

Testability:

- Unit tests for critical modules (70% coverage minimum)
- Integration tests for API endpoints
- End-to-end tests for key workflows
- Test data generation scripts
- Automated testing in CI/CD pipeline

Logging and Debugging:

- Structured logging framework
- Log levels (DEBUG, INFO, WARN, ERROR)
- Contextual logging (user, action, timestamp)
- Centralized log management

- Error stack traces for debugging

8.3 Scalability

Horizontal Scalability:

- Stateless application design
- Session data in database (not in memory)
- Load balancer support for multiple server instances
- Database replication support (read replicas)

Vertical Scalability:

- Efficient resource utilization
- Connection pooling
- Query optimization
- Caching frequently accessed data

Growth Capacity:

- Support 50% annual growth in users and orders
- Database schema extensible for new features
- API versioning for backward compatibility
- Modular architecture for adding new modules

8.4 Usability

Ease of Use:

- Intuitive navigation
- Consistent UI/UX across pages
- Clear visual hierarchy
- Helpful error messages
- Contextual help/tooltips
- Minimal steps to complete tasks

Accessibility:

- WCAG 2.1 Level AA compliance
- Keyboard navigation support
- Screen reader compatibility
- Sufficient color contrast
- Alt text for images
- Proper heading structure

Learnability:

- New customer can place order within 5 minutes
- Admin can be trained in under 2 hours
- Inline help and documentation
- Consistent design patterns

Responsiveness:

- Mobile-first design approach
- Adaptive layouts for all screen sizes
- Touch-friendly interface elements
- Fast page load times
- Smooth transitions and interactions

8.5 Performance

Response Time:

- Page load: < 3 seconds
- API response: < 500ms (95th percentile)
- Search results: < 2 seconds
- Report generation: < 10 seconds

Throughput:

- Handle 500 concurrent users
- Process 50-100 orders per day
- Support 5000 registered users
- 500 products in catalog

Resource Utilization:

- CPU usage: < 70% under normal load
- Memory usage: < 80% of available RAM
- Database connections: Efficiently pooled
- Network bandwidth: Optimized (image compression, minified assets)

Optimization:

- Database query optimization
- Caching strategy (product catalog, categories)
- Image optimization (compression, lazy loading)
- Code minification and bundling
- CDN for static assets (optional)

---

9. Appendices

Appendix A: Glossary

Term	Definition
Admin	System administrator with full access to manage products, orders, inventory, and reports
Cart	Temporary storage of products selected by customer before checkout
Checkout	Process of finalizing an order with payment and delivery details
Customer	Registered user who can browse products, place orders, and track deliveries
District	Geographic region used for delivery address specification
Inventory	Stock of products available for sale
Low Stock Alert	Notification when product quantity falls below threshold
Order ID	Unique identifier for each order
Pappad	Traditional Indian flatbread/crisp (primary product)
Payment Gateway	Third-party service for processing online payments
Regular Customer	Customer who has placed 5 or more orders
SKU	Stock Keeping Unit - Unique product identifier
Stock Threshold	Minimum quantity level that triggers low stock alert
TLS	Transport Layer Security - Encryption protocol

Appendix B: Assumptions

1. **Technical Assumptions:**
    - Users have access to internet-enabled devices
    - Modern web browsers are used (Chrome, Firefox, Safari, Edge)
    - Payment gateway APIs are reliable and available
    - SMS/Email services are available 99% of the time
    - Server infrastructure meets minimum requirements
2. **Business Assumptions:**
    - Orders cannot be cancelled after placement (business policy)
    - Delivery is handled manually by shop via local logistics
    - No integration required with logistics partners
    - Online payment only (no cash on delivery)
    - English language is sufficient for initial version
    - GST billing not required (simple order summary sufficient)
3. **User Assumptions:**
    - Users have basic computer/smartphone literacy
    - Customers have valid email and phone for registration
    - Customers understand online payment process
    - Admin staff have moderate computer skills
4. **Operational Assumptions:**
    - Shop has reliable internet connection
    - System monitored during business hours
    - Technical support available for critical issues
    - Regular backups performed by hosting provider or manually
    - Security updates applied regularly

Appendix C: Constraints

1. **Technical Constraints:**
    - Must use Java (Servlets/Spring Boot) for backend
    - Must use MongoDB as database
    - Must deploy on Apache Tomcat
    - Must support all major browsers
    - Must work on devices from smartphones to desktops
2. **Business Constraints:**
    - Maximum 5000 users
    - Maximum 500 products
    - Expected daily order volume: 50 orders
    - Budget constraints (use open-source technologies)
    - Timeline: 5 weeks for development (as per project plan)
3. **Regulatory Constraints:**
    - Comply with data protection laws
    - Secure payment processing (PCI-DSS)
    - Privacy policy compliance
    - Consumer protection laws
4. **Functional Constraints:**
    - No order cancellation after placement
    - No order modification after placement
    - No offline mode support
    - No integration with external logistics systems
    - No GST invoice generation (simple summary only)

Appendix D: Future Enhancements

Phase 2 (Future):

- Mobile applications (Android and iOS)
- Customer reviews and ratings for products
- Wishlist functionality
- Product recommendations based on purchase history
- Loyalty program and reward points
- Discount coupons and promotional codes
- Bulk order management for wholesale customers
- Multiple payment options (Cash on Delivery)
- Integration with logistics partners for automated delivery tracking
- Multi-language support (Tamil, Hindi)
- Advanced analytics and business intelligence dashboard
- Inventory forecasting and demand prediction
- Automated email marketing campaigns
- Social media integration
- Live chat support
- Voice search functionality
- AR/VR product visualization
- Subscription-based ordering for regular customers

Appendix E: Acronyms and Abbreviations

- **API:** Application Programming Interface
- **BSON:** Binary JSON
- **CORS:** Cross-Origin Resource Sharing
- **CRUD:** Create, Read, Update, Delete
- **CSRF:** Cross-Site Request Forgery

- **CSS:** Cascading Style Sheets
- **GST:** Goods and Services Tax
- **HTML:** HyperText Markup Language
- **HTTP:** HyperText Transfer Protocol
- **HTTPS:** HyperText Transfer Protocol Secure
- **JS:** JavaScript
- **JSON:** JavaScript Object Notation
- **JWT:** JSON Web Token
- **MVC:** Model-View-Controller
- **OTP:** One-Time Password
- **PCI-DSS:** Payment Card Industry Data Security Standard
- **PDF:** Portable Document Format
- **PIN:** Postal Index Number
- **RBAC:** Role-Based Access Control
- **REST:** Representational State Transfer
- **RPO:** Recovery Point Objective
- **RTO:** Recovery Time Objective
- **SDK:** Software Development Kit
- **SKU:** Stock Keeping Unit
- **SMS:** Short Message Service
- **SQL:** Structured Query Language
- **SRS:** Software Requirements Specification
- **SSL:** Secure Sockets Layer
- **TLS:** Transport Layer Security
- **TTL:** Time To Live

- **UI:** User Interface
- **UPI:** Unified Payments Interface
- **URL:** Uniform Resource Locator
- **UX:** User Experience
- **WCAG:** Web Content Accessibility Guidelines
- **XSS:** Cross-Site Scripting

Appendix F: Reference Documents

1. **Project Synopsis - Pappad Shop Management System**
2. **IEEE Std 830-1998** - IEEE Recommended Practice for Software Requirements Specifications
3. **MongoDB Documentation** - https://docs.mongodb.com/
4. **Spring Boot Documentation** - https://spring.io/projects/spring-boot
5. **Java Servlet Specification** - Oracle Java EE Documentation
6. **Bootstrap Framework Documentation** - https://getbootstrap.com/
7. **OWASP Security Guidelines** - https://owasp.org/
8. **PCI-DSS Compliance Standards** - https://www.pcisecuritystandards.org/
9. **WCAG 2.1 Accessibility Guidelines** - https://www.w3.org/WAI/WCAG21/quickref/

Appendix G: Approval Signatures

This SRS document must be reviewed and approved by the following stakeholders:

Role	Name	Signature	Date
Project Manager			
Business Analyst			
Lead Developer			
Database Administrator			
QA Lead			
Client Representative			

Document History:

Version	Date	Author	Changes
1.0	January 27, 2026	Development Team	Initial SRS document created

---

End of Software Requirements Specification Document

Total Pages: 93
Word Count: ~15,000 words
Classification: Project Documentation - Confidential

This SRS document provides a comprehensive specification for the Pappad Shop Management System. It covers all functional and non-functional requirements, system architecture, database design, security requirements, and quality attributes needed for successful system development and deployment.