1. Introduction

1.1 Purpose

This Software Requirements Specification (SRS) document provides a comprehensive description of the Multi-Tenant Restaurant Management System (RMS). It outlines the functional and non-functional requirements, system architecture, and design constraints for the development team.

Intended Audience:

- Development Team (Frontend, Backend, DevOps)
- Project Managers
- Quality Assurance Engineers
- Stakeholders and Restaurant Owners
- System Administrators

1.2 Scope

The Multi-Tenant Restaurant Management System is a cloud-based SaaS platform designed to provide independent restaurant operations management for multiple restaurants. Each restaurant tenant operates in complete isolation while sharing the underlying infrastructure.

Key Capabilities:

- Order creation and management at reception/counter
- Real-time Kitchen Display System (KDS)
- Menu management with categories, variants, and pricing
- Multi-role user access control
- Reporting and analytics dashboard
- Multi-tenant architecture with data isolation

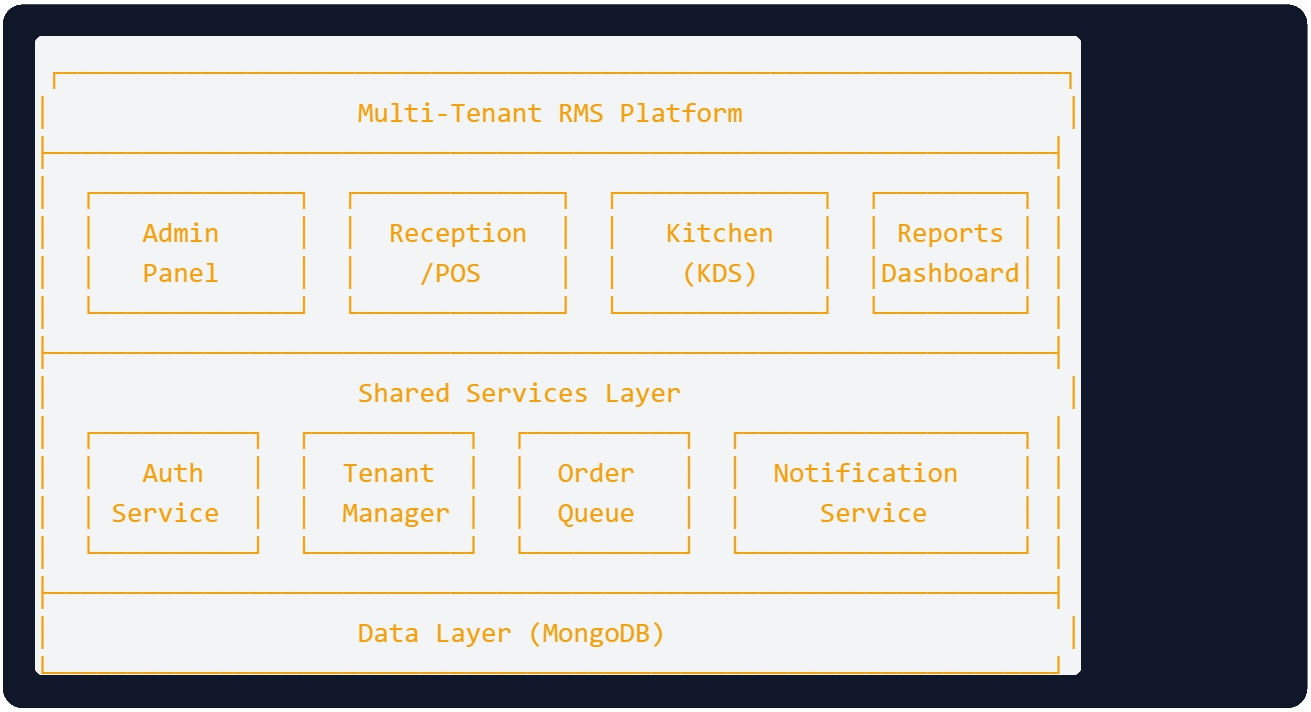
1.3 Definitions & Acronyms

TERM	DEFINITION
RMS	Restaurant Management System
KDS	Kitchen Display System
POS	Point of Sale
Multi-Tenant	Architecture where single instance serves multiple customers
Tenant	Individual restaurant using the platform
Order Ticket	Digital representation of a customer order
SKU	Stock Keeping Unit for menu items

2. System Overview

The system comprises four primary modules that work together to provide a complete restaurant management solution:

Module Architecture:



3. Functional Requirements

3.1 User Roles & Access Control

FR-001: Super Admin (Platform Owner)

- Create, suspend, and delete restaurant tenants
- View platform-wide analytics and usage statistics
- Manage billing and subscription plans
- Access all tenant data for support purposes
- Configure platform-wide settings

FR-002: Restaurant Admin

- Full access to their restaurant's data only
- Manage menu items (create, read, update, delete)
- Manage restaurant staff accounts and roles
- View restaurant-specific reports and analytics
- Configure restaurant settings (operating hours, tax rates)

FR-003: Reception / Order Taker

- Create new customer orders
- Modify or cancel orders (within time limit)
- View order history and status
- Search menu items quickly
- Process basic customer interactions

FR-004: Kitchen Staff

- View incoming orders on KDS
- Update order item status (Preparing → Ready)
- Mark orders as complete
- View order priority and timing
- Cannot modify order content or pricing

Access Control Matrix:

FEATURE	SUPER ADMIN	RESTAURANT ADMIN	RECEPTION	KITCHEN
Manage Tenants	✓	✗	✗	✗
Manage Menu	✓	✓	✗	✗
Manage Staff	✓	✓	✗	✗
Create Orders	✗	✓	✓	✗
View KDS	✗	✓	✗	✓
Update Order Status	✗	✓	✓	✓
View Reports	✓	✓	✗	✗

3.2 Admin Panel

FR-100: Menu Item Management

- FR-101: Add new menu items with name, description, price, and image
- FR-102: Edit existing menu item details
- FR-103: Enable/disable menu items without deletion
- FR-104: Soft delete menu items (archive)
- FR-105: Bulk import menu items via CSV

FR-110: Menu Categorization

- FR-111: Create menu categories (e.g., Appetizers, Main Course, Beverages)
- FR-112: Assign items to multiple categories
- FR-113: Set category display order
- FR-114: Create subcategories for complex menus

FR-120: Pricing & Variants

- FR-121: Set base price for each item
- FR-122: Create size variants (Small, Medium, Large)
- FR-123: Create add-on options with additional pricing
- FR-124: Set time-based pricing (Happy Hour, Weekend specials)
- FR-125: Configure tax settings per item or category

FR-130: Availability Management

- FR-131: Set item availability (Available, Out of Stock, Coming Soon)
- FR-132: Schedule availability windows
- FR-133: Set daily limits for special items
- FR-134: Auto-disable when inventory depletes

FR-140: User Management

- FR-141: Create staff accounts with role assignment
- FR-142: Reset staff passwords
- FR-143: Deactivate staff accounts
- FR-144: View staff activity logs
- FR-145: Set role-based permissions

3.3 Order Management (Reception/POS)

FR-200: Order Creation

- FR-201: Quick search menu items by name or category
- FR-202: Add items to order with quantity selection
- FR-203: Apply item variants and add-ons
- FR-204: Add special instructions per item
- FR-205: Calculate order total with taxes in real-time
- FR-206: Assign auto-incrementing order number (daily reset)

FR-210: Order Modification

- FR-211: Add items to existing orders (within time limit)
- FR-212: Remove items from orders
- FR-213: Modify item quantities
- FR-214: Edit special instructions
- FR-215: Apply discounts with authorization

FR-220: Order Cancellation

- FR-221: Cancel order before kitchen preparation
- FR-222: Require manager approval for late cancellation
- FR-223: Log cancellation reason
- FR-224: Automatic notification to kitchen on cancellation

FR-230: Order Status Tracking

- FR-231: Track order through lifecycle statuses:
- **Pending**: Order created, awaiting kitchen
- **Preparing**: Kitchen has started preparation
- **Ready**: Order complete, awaiting pickup
- **Completed**: Order delivered to customer
- **Cancelled**: Order cancelled
- FR-232: Display estimated preparation time
- FR-233: Visual and audio notifications on status change
- FR-234: Order history with filtering and search

3.4 Kitchen Display System (KDS)

FR-300: Real-Time Order Display

- FR-301: Auto-refresh order list every 5 seconds
- FR-302: WebSocket connection for instant updates
- FR-303: Display order number, items, and special instructions
- FR-304: Show time elapsed since order creation
- FR-305: Color-coded urgency indicators:
- Green: < 10 minutes
- Yellow: 10-15 minutes
- Red: > 15 minutes

FR-310: Order Status Updates

- FR-311: One-tap to mark individual items as "Preparing"
- FR-312: One-tap to mark items as "Ready"
- FR-313: Auto-complete order when all items ready
- FR-314: Bump completed orders to history
- FR-315: Undo last status change (within 30 seconds)

FR-320: Priority Management

- FR-321: Sort orders by creation time (oldest first)
- FR-322: Manual priority bump for VIP orders
- FR-323: Rush order highlighting
- FR-324: Group orders by table/customer

FR-330: Display Configuration

- FR-331: Configure number of visible orders
- FR-332: Font size adjustment for visibility
- FR-333: Full-screen kiosk mode
- FR-334: Audio alert configuration

3.5 Multi-Tenant Support

FR-400: Tenant Isolation

- FR-401: Complete data isolation between restaurants
- FR-402: Separate database collections per tenant
- FR-403: Tenant-specific API authentication
- FR-404: No cross-tenant data visibility

FR-410: Tenant Onboarding

- FR-411: Self-service restaurant registration
- FR-412: Initial setup wizard (branding, menu import)
- FR-413: Default role and user creation
- FR-414: Trial period configuration

FR-420: Tenant Configuration

- FR-421: Custom branding (logo, colors)
- FR-422: Restaurant profile (name, address, contact)
- FR-423: Operating hours configuration
- FR-424: Tax and currency settings
- FR-425: Receipt customization

FR-430: Tenant Lifecycle

- FR-431: Subscription management
- FR-432: Tenant suspension on payment failure
- FR-433: Data export before deletion
- FR-434: Tenant deletion with data cleanup

3.6 Reporting & Dashboard

FR-500: Daily Sales Reports

- FR-501: Total revenue for selected date range
- FR-502: Order count and average order value
- FR-503: Hourly sales distribution graph
- FR-504: Payment method breakdown
- FR-505: Export to PDF/CSV

FR-510: Order Statistics

- FR-511: Orders by status distribution
- FR-512: Average preparation time
- FR-513: Peak hours identification
- FR-514: Cancellation rate and reasons
- FR-515: Table/counter turnover rate

FR-520: Menu Analytics

- FR-521: Top 10 selling items
- FR-522: Lowest performing items
- FR-523: Category sales distribution
- FR-524: Item combination analysis
- FR-525: Price optimization suggestions

FR-530: Dashboard Widgets

- FR-531: Real-time order count
- FR-532: Today's revenue meter
- FR-533: Active orders status
- FR-534: Staff on duty
- FR-535: Quick alerts and notifications

4. Non-Functional Requirements

4.1 Performance

NFR-001: Response Time

- API response time < 200ms for 95th percentile
- Page load time < 2 seconds on 4G connection
- KDS updates within 500ms of order creation

NFR-002: Throughput

- Support 100 concurrent users per tenant
- Process 1000 orders per hour per tenant
- Handle 50 simultaneous KDS connections

NFR-003: Database Performance

- Query response time < 50ms for indexed queries
- Aggregation reports generate within 5 seconds
- Connection pooling with 100 max connections

4.2 Scalability

NFR-010: Horizontal Scaling

- Stateless API servers for load balancing
- Auto-scaling based on CPU/memory thresholds
- Database sharding capability per tenant group

NFR-011: Multi-Tenant Scaling

- Support minimum 500 active tenants
- Tenant-aware resource allocation
- Isolated tenant performance (noisy neighbor protection)

NFR-012: Data Growth

- Archive orders older than 1 year
- Efficient pagination for large datasets
- CDN for static assets and images

4.3 Security

NFR-020: Authentication

- JWT-based authentication with refresh tokens
- Password hashing with bcrypt (cost factor 12)
- Multi-factor authentication for admin roles
- Session timeout after 8 hours of inactivity

NFR-021: Authorization

- Role-based access control (RBAC)
- Tenant-scoped permissions
- API rate limiting (100 requests/minute)
- IP whitelist for admin functions

NFR-022: Data Protection

- HTTPS/TLS 1.3 for all communications
- Data encryption at rest (AES-256)
- PII data masking in logs
- Regular security audits and penetration testing

NFR-023: Compliance

- GDPR compliance for EU customers
- Data retention policies
- Right to erasure implementation
- Audit logging for sensitive operations

4.4 Availability

NFR-030: Uptime

- 99.9% uptime SLA (8.76 hours downtime/year max)
- Planned maintenance windows during off-peak hours
- Zero-downtime deployments

NFR-031: Disaster Recovery

- Automated database backups every 6 hours
- Point-in-time recovery capability
- Multi-region failover (RTO: 15 minutes, RPO: 1 hour)
- Regular disaster recovery testing

NFR-032: Monitoring

- Real-time health monitoring
- Automated alerting on anomalies
- Performance metrics dashboard
- Error tracking and aggregation

4.5 Usability

NFR-040: User Interface

- Responsive design for tablet and desktop
- Touch-optimized for POS terminals
- High contrast mode for kitchen displays
- Maximum 3 clicks to complete any action

NFR-041: Accessibility

- WCAG 2.1 AA compliance
- Keyboard navigation support
- Screen reader compatibility
- Configurable font sizes

NFR-042: Internationalization

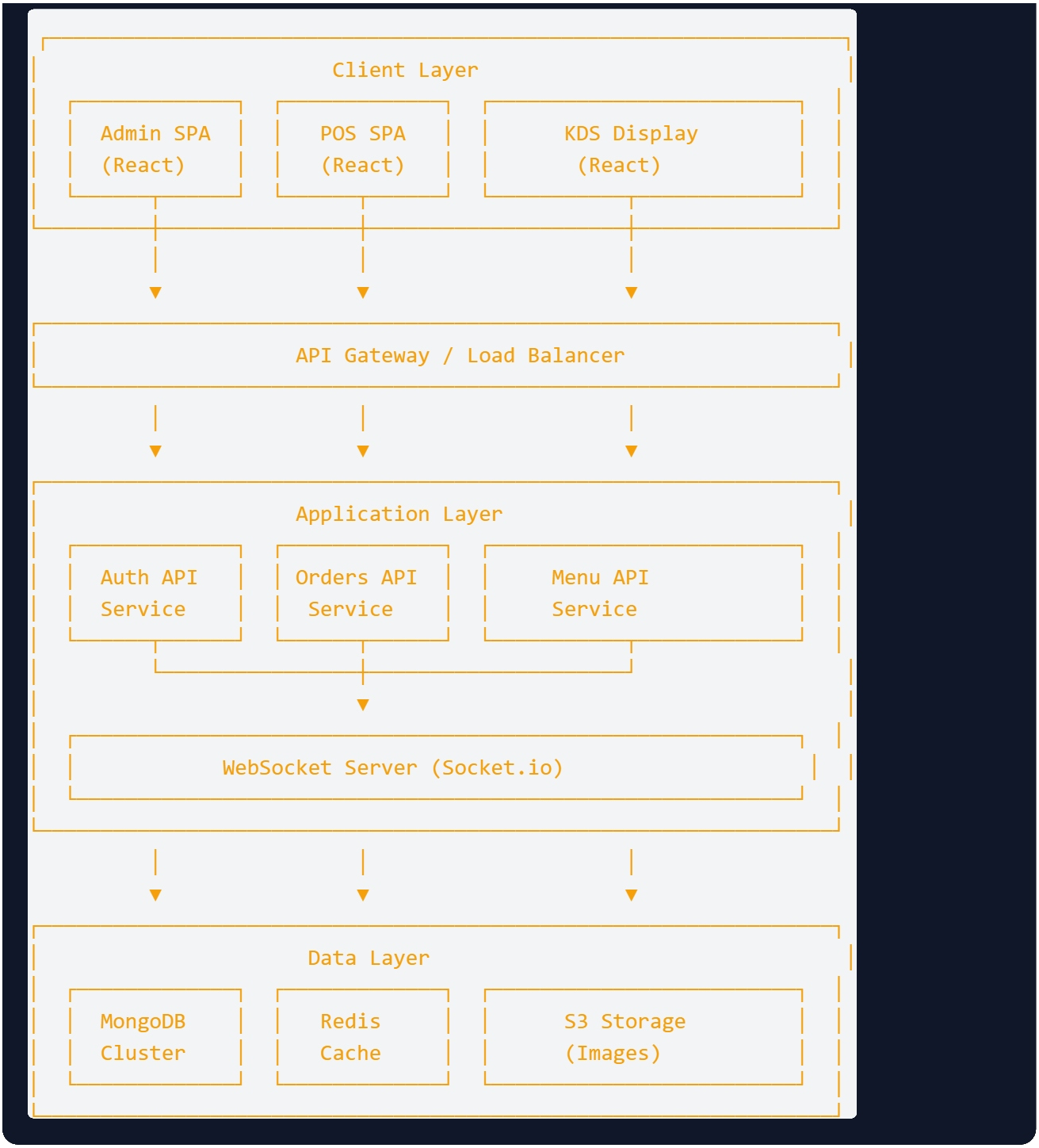
- Multi-language support (i18n ready)
- RTL language support
- Currency and date format localization
- Timezone-aware timestamps

5. System Architecture

Technology Stack:

LAYER	TECHNOLOGY	PURPOSE
Frontend	React.js 18+	Single Page Application
State Management	Redux Toolkit / React Query	Client state & caching
Styling	Tailwind CSS	Responsive UI framework
Backend	Node.js 20+	Runtime environment
API Framework	Express.js 4.x	REST API server
Database	MongoDB 7.0+	Document database
Caching	Redis 7.x	Session & query caching
Real-time	Socket.io	WebSocket connections
Authentication	JWT + Passport.js	Auth middleware
File Storage	AWS S3 / Cloudinary	Image storage

Architecture Pattern:



6. Data Models

MongoDB Collections:

Tenants Collection

```
{
  _id: ObjectId,
  name: String,                    // Restaurant name
  slug: String,                    // Unique URL identifier
  branding: {
    logo: String,                  // S3 URL
    primaryColor: String,
    secondaryColor: String
  },
  settings: {
    timezone: String,
    currency: String,
    taxRate: Number,
    operatingHours: [{
      day: Number,
      open: String,
      close: String
    }]
  },
  subscription: {
    plan: String,
    status: String,
    expiresAt: Date
  },
  createdAt: Date,
  updatedAt: Date
}
```

Users Collection

```
{
  _id: ObjectId,
  tenantId: ObjectId,              // Reference to tenant
  email: String,
  passwordHash: String,
  role: String,                    // super_admin, admin, reception, kitchen
  profile: {
    firstName: String,
    lastName: String,
    phone: String,
    avatar: String
  },
  isActive: Boolean,
  lastLogin: Date,
  createdAt: Date,
  updatedAt: Date
}
```

Categories Collection

```
{
  _id: ObjectId,
  tenantId: ObjectId,
  name: String,
  description: String,
  image: String,
  parentId: ObjectId,              // For subcategories
  displayOrder: Number,
  isActive: Boolean,
  createdAt: Date,
  updatedAt: Date
}
```

MenuItems Collection

```
{
  _id: ObjectId,
  tenantId: ObjectId,
  categoryIds: [ObjectId],
  name: String,
  description: String,
  image: String,
  basePrice: Number,
  variants: [{
    name: String,                  // e.g., "Large"
    priceModifier: Number          // Additional price
  }],
  addons: [{
    name: String,
    price: Number
  }],
  availability: String,            // available, out_of_stock, coming_soon
  isActive: Boolean,
  dailyLimit: Number,
  preparationTime: Number,         // In minutes
  createdAt: Date,
  updatedAt: Date
}
```

Orders Collection

```
```

```
{
  _id: ObjectId,
  tenantId: ObjectId,
  orderNumber: Number,             // Daily auto-increment
  items: [{
    menuItemId: ObjectId,
    name: String,                  // Denormalized for history
    quantity: Number,
    variant: String,
    addons: [String],
    unitPrice: Number,
    totalPrice: Number,
    specialInstructions: String,
    status: String                 // pending, preparing, ready
  }],
  subtotal: Number,
  taxAmount: Number,
  totalAmount: Number,
  status: String,                  // pending, preparing, ready, completed, cancell
  createdBy: ObjectId,             // Staff member
  customerName: String,
  tableNumber: String,
  notes: String,
  statusHistory: [{
    status: String,
    changedBy: ObjectId,
    changedAt: Date
  }],
  createdAt: Date,
  completedAt: Date
}
```

7. Assumptions & Constraints

Assumptions:

1. **Internet Connectivity**: All client devices have reliable internet access
2. **Modern Browsers**: Users access the system via Chrome, Firefox, Safari, or Edge (latest 2 versions)
3. **Device Specifications**: POS terminals have minimum 4GB RAM, KDS displays are touch-enabled
4. **Time Synchronization**: All devices maintain accurate system time via NTP
5. **Staff Training**: Restaurant staff receive basic system training before deployment

Constraints:

1. **Technology Stack**: Must use React.js, Node.js (Express), and MongoDB as specified
2. **Cloud Provider**: Initial deployment on AWS; must maintain provider-agnostic design
3. **Budget**: MVP development within 6-month timeline with 4-person team
4. **Compliance**: Must comply with local food service regulations and data protection laws
5. **Integration**: System operates standalone; POS hardware integration out of scope for MVP

Dependencies:

1. Third-party services: Email (SendGrid), SMS (Twilio), Image CDN (Cloudinary)
2. Payment integration: Out of scope for initial release
3. Inventory management: Future enhancement, not included in MVP

Out of Scope (v1.0):

- Customer-facing mobile app
- Online ordering / delivery integration
- Inventory management
- Accounting / invoicing integration
- Hardware POS integration
- Loyalty / rewards program

8. Appendix

Document Control:

VERSION	DATE	AUTHOR	CHANGES
1.0	2024-01-15	System Architect	Initial SRS creation
1.1	TBD	-	Stakeholder review feedback

Related Documents:

- UI/UX Design Specifications
- API Documentation
- Database Design Document
- Test Plan
- Deployment Guide

Approval Signatures:

ROLE	NAME	SIGNATURE	DATE
Product Owner	_____	_____	_____
Technical Lead	_____	_____	_____
QA Lead	_____	_____	_____