REQUIREMENTS DEFINITITION DOCUMENT FOR DINE SWIFT APPLICATION

Document Version: 1.0.2 **Date:** October, 13th, 2025

Status: Approved with Component Specification

Authors: Product Management Team

Revision Focus: Clear component responsibility assignment and phased implementation

---

NAME	RegNo	EMAIL	PHONE
Mushabe Moses	23/U/12131/EVE	Mosesmushae9@gmail.com	0752307875
Drate Hillary	23/U/23611	dratehillary@gmail.om	0758235980
Mukyala Dorcus Nandy	23/U/11911/EVE	mukyaladorcus@gmail.com	0755011795
Kiyimba Fahad	23/U/0628	kiyimbafwitty@gmail.com	0762938957

Contents

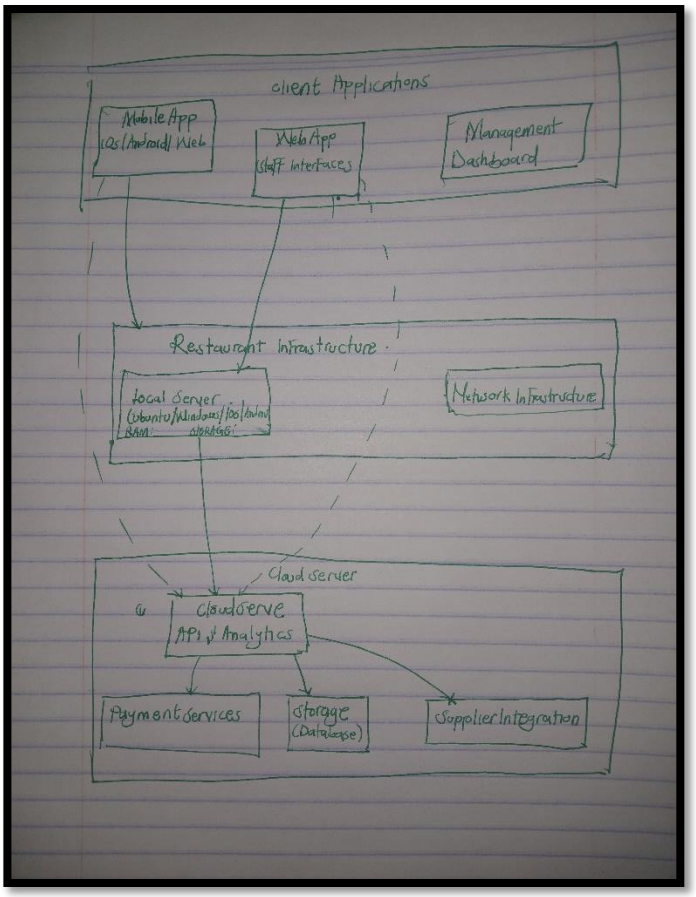
1. INTRODUCTION

1.1. Purpose

This document defines the functional and non-functional requirements for the DineSwift platform, a comprehensive Software-as-a-Service (SaaS) solution designed to digitize and optimize the dining experience for both customers and restaurants.**1.2. Scope**

1.2. Scope

System Components Overview:



Phase 1: Core MVP (weeks) - Component Focus:

- **MOBILE-APP:** Customer-facing mobile application for iOS and Android
- **WEB-APP:** Basic restaurant staff interfaces (order management only)
- **MANAGEMENT-DASHBOARD:** Web-based restaurant management dashboard (essential features)
- **CLOUD-SERVER:** Integration with Momo payment gateway
- **LOCAL-SERVER:** Offline-first architecture with local server support
- **All Components:** Core ordering and booking functionalities

In Scope for Initial Release:

- Customer-facing mobile application for iOS and Android
- Restaurant staff applications (waiter, chef, manager interfaces)
- Web-based restaurant management dashboard
- Integration with specified payment gateways (Momo, Visa)
- Core ordering, booking, inventory, and analytics functionalities
- Offline-first architecture with local server support
- Real-time communication and feedback systems
- Loyalty program management
- Supplier and inventory management

Out of Scope for Initial Release:

- Hardware procurement and management
- Bitcoin/cryptocurrency payment integration
- Integration with external accounting software systems
- Custom hardware development
- Advanced AI-powered predictive analytics
- Multi-language support beyond English
- Advanced reporting and BI tools

1.3. Definitions, Acronyms, and Abbreviations

Acronym	Definition
OTP	One-Time Password
SLA	Service Level Agreement
API	Application Programming Interface
POS	Point of Sale
QR Code	Quick Response Code
MVP	Minimum Viable Product
SaaS	Software as a Service

PCI DSS	Payment Card Industry Data Security Standard

1.4. References

- UX Research Findings - Customer Journey Mapping
- PCI DSS Compliance Requirements
- Technical Architecture Specification

1.5. Document Conventions

- Requirements are numbered using format: **FR-[MODULE]-[NUMBER]**
- "Shall" indicates mandatory requirements
- "Should" indicates recommended but not mandatory
- Priority: **H** (High - MVP), **M** (Medium - Post-MVP), **L** (Low - Future Enhancement)
- Error handling requirements prefixed with **EH-**
- Data requirements prefixed with **DR-**

1.6. Intended Audience

- Project Sponsors and Stakeholders
- Development and QA Teams
- System Architects and Designers
- Restaurant Management Teams
- UX/UI Design Team
- Operations and Support Teams

1.7. Stakeholders

Role	Responsibility	Priority
Project Sponsor	Budget approval, strategic direction	High
Restaurant Owners	End users, requirement validation	High
Customers	Primary end users	High
Development Team	Implementation	High
QA Team	Validation and testing	High
Operations Team	System maintenance and support	Medium
Marketing Team	Customer acquisition and promotion	Medium

1.8. Assumptions and Constraints

Assumptions:

- Restaurants have stable Wi-Fi connectivity for local servers
- Customers have smartphones with camera capabilities for QR scanning
- Payment gateway APIs are stable and available
- Restaurant staff are trainable on digital systems
- Mobile data connectivity is generally available for customers

Constraints:

- Must comply with **PCI DSS** security standards for payment processing
- Must support **offline functionality** for core operations
- Maximum **3-second menu load time** requirement
- **99.9% system uptime** requirement
- Must support both **iOS and Android** platforms
- Data privacy regulations compliance (GDPR, CCPA)

2. OVERALL DESCRIPTION

2.1. Product Perspective

Component Responsibility Matrix:

Component	Primary Role	Key Responsibilities
Mobile App	Customer Experience	QR scanning, menu browsing, ordering, payments, loyalty
Web App	Staff Operations	Order management, inventory, supplier orders, analytics
Local Server	Restaurant Operations	Offline support, real-time processing, local caching
Cloud Server	Platform Management	Central data, payments, multi-tenant support, analytics
Management Dashboard	Business Intelligence	Reporting, performance monitoring, configuration

2.2. Product Functions

- Contactless ordering via QR codes
- Intelligent waiter dispatch and order tracking
- Table booking with deposit system
- Flexible food ordering and delivery
- Supplier ordering and inventory management
- Real-time communication and feedback
- Gamified loyalty program
- Immersive media integration
- Comprehensive analytics dashboard

2.3. User Characteristics

Primary Users:

- Customers: Varying technical proficiency, mobile-first, expect instant service
- Wait Staff: Need simple, fast interfaces during peak hours
- Chefs/Kirchen Staff: Require clear, prioritized order information
- Restaurant Managers: Need comprehensive oversight and reporting
- Suppliers: Require clear order information and payment processing

2.4. Operating Environment

- Mobile: iOS 13+, Android 8+
- Web: Chrome, Safari, Firefox, Edge (latest versions)
- Local servers: Ubuntu 20.04+/ Windows 11+/Android/iOS, 4GB RAM minimum

- Cloud: AWS/Azure cloud infrastructure/ Google cloud
- Network: 4G/LTE/Wi-Fi connectivity

2.5. Design and Implementation Constraints

- Must use **React Native** for mobile applications
- **PostgreSQL** for primary database
- Real-time updates via **WebSocket** connections
- **RESTful API** architecture
- **JSON** for data interchange
- **JWT** for authentication

2.6. User Documentation

- Online help system and knowledge base
- Mobile app in-app tutorials
- Staff training materials and videos
- API documentation for integration partners
- Administrator guide for restaurant setup

2.7. Assumptions and Dependencies

- Dependent on third-party payment gateway availability
- Assumes restaurant compliance with setup requirements
- Dependent on mobile app store approval processes
- Assumes adequate staff training for system adoption

3. SYSTEM FEATURES AND REQUIREMENTS - COMPLETELY REVISED WITH COMPONENT ASSIGNMENT

3.1. Contactless Ordering & Payments - PHASE 1 (P1)

3.1.1. Functional Requirements - COMPONENT-SPECIFIC

- **MOBILE-APP-FR-001-P1:** The Mobile App shall scan QR codes using device camera and extract restaurant/table identifiers
  - **Priority:** H
  - **Rationale:** Core customer interaction point
- **CLOUD-FR-001-P1:** The Cloud Server shall resolve QR codes to restaurant-specific menu URLs with table mapping
  - **Priority:** H
  - **Rationale:** Centralized restaurant configuration
- **LOCAL-FR-001-P1:** The Local Server shall cache menu data and serve it to mobile apps with version control
  - **Priority:** H
  - **Rationale:** Offline functionality and performance
- **MOBILE-APP-FR-002-P1:** The Mobile App shall provide real-time shopping cart functionality with running totals and tax calculations
  - **Priority:** H
  - **Rationale:** Core user experience
- **CLOUD-FR-002-P1:** The Cloud Server shall integrate with Momo payment gateway with idempotent transaction handling
  - **Priority:** H
  - **Rationale:** Payment processing reliability
- **LOCAL-FR-002-P1:** The Local Server shall generate unique OTP codes for order verification with 5-minute expiration
  - **Priority:** H
  - **Rationale:** Order security and accuracy
- **LOCAL-FR-003-P2:** The Local Server shall calculate estimated preparation times using historical data and current kitchen load
  - **Priority:** M
  - **Rationale:** Customer expectation management

3.1.2. Data Requirements - COMPONENT-ASSIGNED

- **LOCAL-DR-001-P1:** The Local Server shall cache menu items with name, description, price, category, availability status, preparation time, and associated media
  - **Priority:** H
- **CLOUD-DR-001-P1:** The Cloud Server shall maintain order records including items, quantities, special instructions, timestamps, table information, payment status, and sync version
  - **Priority:** H

- **CLOUD-DR-002-P1:** The Cloud Server shall store payment transaction records with status, amount, gateway reference, customer information, and idempotency key
  - **Priority:** H
- **LOCAL-DR-002-P1:** The Local Server shall cache menu data with version control, last update timestamps, and checksum validation
  - **Priority:** H

3.1.3. Error Handling & Edge Cases

- **MOBILE-APP-EH-001-P1:** The Mobile App shall handle payment failures by preserving cart state and displaying clear error messages with retry options
  - **Priority:** H
  - **Component Recovery:** Mobile App preserves cart, suggests alternative payment
  - **Business Impact:** Medium
- **MOBILE-APP-EH-002-P1:** The Mobile App shall fall back to cloud-based menu retrieval when local server is unavailable with user notification
  - **Priority:** H
  - **Component Recovery:** Mobile App switches data source, notifies user
  - **Business Impact:** High
- **LOCAL-EH-001-P1:** The Local Server shall sync queued offline orders using conflict-free replicated data types (CRDT) when connectivity is restored
  - **Priority:** H
  - **Component Recovery:** Local Server handles sync with conflict resolution
  - **Business Impact:** Medium

3.2. Intelligent Waiter Dispatch & Order Tracking - PHASE 2 (P2)

3.2.1. Functional Requirements - COMPONENT-SPECIFIC

- **WEB-APP-FR-001-P2:** The Web App (Kitchen) shall provide interface for chefs to mark orders as "Ready" with preparation completion timestamps
  - **Priority:** H
  - **Rationale:** Core dispatch trigger
- **LOCAL-FR-101-P2:** The Local Server shall track waiter availability and current status in real-time with manual override capability
  - **Priority:** H
  - **Rationale:** Dispatch efficiency foundation
- **LOCAL-FR-102-P2:** The Local Server shall implement dynamic order batching algorithm based on table proximity and readiness timing
  - **Priority:** H
  - **Rationale:** Operational efficiency core
- **LOCAL-FR-103-P2:** The Local Server shall calculate optimal delivery routes for batch assignments with manual adjustment capability
  - **Priority:** M
  - **Rationale:** Movement optimization
- **WEB-APP-FR-002-P2:** The Web App (Waiter) shall require OTP verification for order delivery with manager override capability

- **Priority:** H
- **Rationale:** Order accuracy and security

- **CLOUD-FR-101-P2:** The Cloud Server shall track order preparation times and delivery performance metrics with anomaly detection
  - **Priority:** M
  - **Rationale:** Performance monitoring

3.2.2. Data Requirements

- **LOCAL-DR-101-P2:** The Local Server shall store waiter efficiency scores, current status, and location data
  - **Priority:** H
- **CLOUD-DR-101-P2:** The Cloud Server shall maintain delivery batch records with optimized routes and completion times
  - **Priority:** M
- **CLOUD-DR-102-P2:** The Cloud Server shall track order timing metrics from placement to delivery
  - **Priority:** M

3.2.3. Error Handling & Edge Cases

- **LOCAL-EH-101-P2:** The Local Server shall automatically reassign batches when waiters become unavailable with notification to affected staff
  - **Priority:** H
  - **Component Recovery:** Local Server handles reassignment, Web App notifies staff
  - **Business Impact:** Medium
- **WEB-APP-EH-001-P2:** The Web App (Waiter) shall provide alternative verification methods when OTP verification fails
  - **Priority:** H
  - **Component Recovery:** Web App enables manager override with audit logging
  - **Business Impact:** Low
  -

3.3. Table Booking & Deposit System - PHASE 1 (P1)

3.3.1. Functional Requirements

- **MOBILE-APP-FR-101-P1:** The Mobile App shall display real-time table availability with visual layout and maintenance status
  - **Priority:** H
  - **Rationale:** Customer booking experience
- **CLOUD-FR-102-P1:** The Cloud Server shall require refundable deposits for booking confirmation with clear policy communication
  - **Priority:** H
  - **Rationale:** Revenue protection

- **CLOUD-FR-103-P1:** The Cloud Server shall generate digital tickets with QR codes for check-in and automatic staff notification
  - **Priority:** H
  - **Rationale:** Streamlined check-in process
- **LOCAL-FR-104-P1:** The Local Server shall automatically apply deposits to final bills upon validation with manual adjustment capability
  - **Priority:** H
  - **Rationale:** Customer convenience
- **CLOUD-FR-104-P1:** The Cloud Server shall enforce configurable advance booking requirements
  - **Priority:** M
  - **Rationale:** Operational planning

3.3.2. Data Requirements

- **CLOUD-DR-103-P1:** The Cloud Server shall store booking records with customer information, table assignment, deposit status, and special requests
  - **Priority:** H
- **CLOUD-DR-104-P1:** The Cloud Server shall maintain table capacity, layout coordinates, and availability schedules
  - **Priority:** H

3.3.3. Error Handling & Edge Cases

- **WEB-APP-EH-002-P1:** The Web App (Host) shall provide manual lookup by booking ID or customer details when digital ticket is unavailable
  - **Priority:** H
  - **Component Recovery:** Web App enables alternative verification
  - **Business Impact:** Low
- **CLOUD-EH-001-P1:** The Cloud Server shall automatically process deposit forfeiture for no-shows with customer notification
  - **Priority:** M
  - **Component Recovery:** Cloud Server handles compensation according to policies
  - **Business Impact:** Low

3.4. Supplier Ordering & Inventory Management - PHASE 2 (P2)

3.4.1. Functional Requirements

- **LOCAL-FR-201-P2:** The Local Server shall trigger automatic low-stock alerts at configurable thresholds with escalation
  - **Priority:** H
  - **Rationale:** Inventory optimization
- **WEB-APP-FR-101-P2:** The Web App (Manager) shall generate pre-populated digital order forms with suggested quantities
  - **Priority:** H
  - **Rationale:** Ordering efficiency

- **WEB-APP-FR-102-P2:** The Web App (Manager) shall support item rejection with photographic proof and digital signatures
  - **Priority:** H
  - **Rationale:** Quality control
- **CLOUD-FR-201-P2:** The Cloud Server shall automatically reconcile payments based on accepted quantities
  - **Priority:** H
  - **Rationale:** Financial accuracy
- **CLOUD-FR-202-P3:** The Cloud Server shall track supplier performance metrics and ratings
  - **Priority:** M
  - **Rationale:** Vendor management

3.4.2. Data Requirements

- **LOCAL-DR-201-P2:** The Local Server shall store inventory items with current stock levels, thresholds, cost prices, and supplier information
  - **Priority:** H
- **CLOUD-DR-201-P2:** The Cloud Server shall maintain supplier records with contact information, payment terms, and performance history
  - **Priority:** H
- **CLOUD-DR-202-P2:** The Cloud Server shall track order item rejections with reasons, proof images, and resolution status
  - **Priority:** M

3.4.3. Error Handling & Edge Cases

- **CLOUD-EH-101-P2:** The Cloud Server shall queue orders and send via alternative channels when supplier API is unavailable
  - **Priority:** H
  - **Component Recovery:** Cloud Server implements multi-channel order placement
  - **Business Impact:** Medium
- **WEB-APP-EH-101-P2:** The Web App (Manager) shall support manual inventory adjustment with reason tracking and approval workflow
  - **Priority:** M
  - **Component Recovery:** Web App provides audit trail for discrepancies
  - **Business Impact:** Low

3.5. Loyalty Program Management - PHASE 2 (P2)

3.5.1. Functional Requirements

- **CLOUD-FR-301-P2:** The Cloud Server shall automatically enroll customers upon first registered order with opt-out capability
  - **Priority:** H
  - **Rationale:** Program adoption
- **CLOUD-FR-302-P2:** The Cloud Server shall implement tier-based progression using lifetime spend and order count metrics
  - **Priority:** H
  - **Rationale:** Customer retention
- **CLOUD-FR-303-P2:** The Cloud Server shall maintain points currency earned with every purchase and expiration rules
  - **Priority:** H
  - **Rationale:** Engagement mechanism
- **MOBILE-APP-FR-201-P2:** The Mobile App shall display loyalty status, points balance, and available rewards
  - **Priority:** H
  - **Rationale:** Customer visibility
- **CLOUD-FR-304-P3:** The Cloud Server shall provide tier-specific rewards in addition to points-based rewards
  - **Priority:** M
  - **Rationale:** Tier differentiation

3.5.2. Data Requirements

- **CLOUD-DR-301-P2:** The Cloud Server shall store customer loyalty profiles with tier status, points balance, lifetime spend, and visit history
  - **Priority:** H
- **CLOUD-DR-302-P2:** The Cloud Server shall maintain loyalty rewards catalog with point costs, tier requirements, and redemption rules
  - **Priority:** H
- **CLOUD-DR-303-P2:** The Cloud Server shall track reward redemption history with points used and redemption value
  - **Priority:** M

3.5.3. Error Handling & Edge Cases

- **CLOUD-EH-201-P2:** The Cloud Server shall queue points calculations and retry with transaction rollback protection during failures
  - **Priority:** H
  - **Component Recovery:** Cloud Server ensures eventually consistent points allocation
  - **Business Impact:** Medium
- **CLOUD-EH-202-P2:** The Cloud Server shall provide alternative rewards or point refunds when reward inventory is depleted

- **Priority:** M
- **Component Recovery:** Cloud Server implements reward substitution logic
- **Business Impact:** Low

4. EXTERNAL INTERFACE REQUIREMENTS

4.1. User Interfaces

Mobile App Interfaces:

- React Native, supporting iOS and Android
- QR code scanner interface
- Menu browsing and cart management
- Payment processing screens
- Order tracking and status
- Loyalty program display

Web App (Staff) Interfaces:

- Kitchen order management dashboard
- Waiter assignment and tracking
- Inventory management screens
- Supplier ordering interface
- Customer communication hub

Management Dashboard Interfaces:

- Real-time analytics and reporting
- Restaurant configuration
- Staff performance monitoring
- Financial reporting

4.2. Hardware Interfaces

Local Server Hardware Management:

- **LOCAL-FR-401-P1:** The Local Server shall manage network connectivity for up to 50 concurrent client connections
- **LOCAL-FR-402-P1:** The Local Server shall monitor system health and trigger alerts for hardware issues

Mobile App Hardware Integration:

- **MOBILE-APP-FR-301-P1:** The Mobile App shall access device camera for QR code scanning
- **MOBILE-APP-FR-302-P1:** The Mobile App shall utilize device storage for offline data caching

4.3. Software Interfaces

Cloud Server External Integrations:

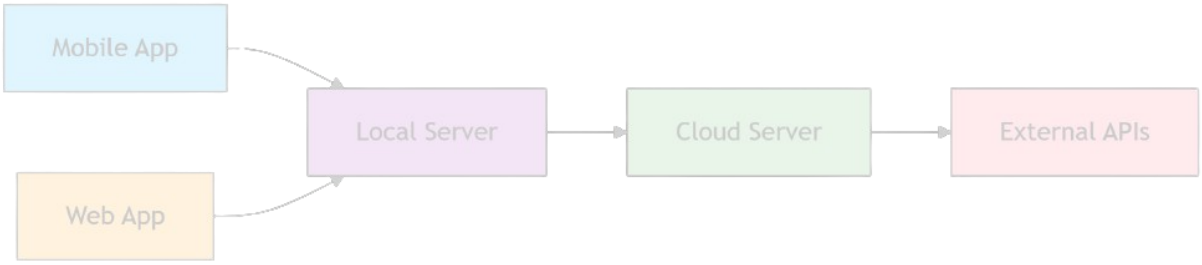
- **CLOUD-FR-401-P1:** The Cloud Server shall integrate with Momo, Bitcoin, payment gateway API
- **CLOUD-FR-402-P2:** The Cloud Server shall integrate with Visa Direct API
- **CLOUD-FR-403-P2:** The Cloud Server shall integrate with supplier ordering APIs
- **CLOUD-FR-404-P2:** The Cloud Server shall integrate with delivery partner APIs (Uber Eats, Glovo)

Local Server Internal Interfaces:

- **LOCAL-FR-403-P1:** The Local Server shall provide REST API for mobile app connectivity
- **LOCAL-FR-404-P1:** The Local Server shall maintain WebSocket connections for real-time updates

4.4. Communications Interfaces

- **Inter-Component Communication:**



-
    1. **Mobile App Communications:**
        - **MOBILE-APP-FR-303-P1:** The Mobile App shall **maintain WebSocket connection** to Local Server for real-time order updates.
        - **MOBILE-APP-FR-304-P1:** The Mobile App shall **implement offline queue** for orders when disconnected.
    2. **Cloud Server Communications:**
        - **CLOUD-FR-405-P1:** The Cloud Server shall **provide RESTful APIs** for all client components.

- **CLOUD-FR-406-P1:** The Cloud Server shall **implement Webhook notifications** for external systems.

5. NON-FUNCTIONAL REQUIREMENTS

5.1. Performance Requirements

- **Mobile App Performance:**
  1. **MOBILE-NFR-001-P1:** The Mobile App shall **load menus within 3 seconds** on 4G connection (P95).
  2. **MOBILE-NFR-002-P1:** The Mobile App shall **respond to user interactions within 100ms**.
- **Local Server Performance:**
  1. **LOCAL-NFR-001-P1:** The Local Server shall **process and route orders to kitchen within 500ms** of payment confirmation.
  2. **LOCAL-NFR-002-P1:** The Local Server shall **support 50+ concurrent mobile app connections**.
- **Cloud Server Performance:**
  1. **CLOUD-NFR-001-P1:** The Cloud Server shall **support 100,000+ concurrent users** across 1,000+ restaurant tenants.
  2. **CLOUD-NFR-002-P1:** The Cloud Server shall **complete database queries within 100ms** under normal load (P95).

5.2. Security Requirements

- **Mobile App Security:**
  1. **MOBILE-NFR-101-P1:** The Mobile App shall **never store customer payment data on device**.
  2. **MOBILE-NFR-102-P1:** The Mobile App shall **validate SSL certificates** for all API calls.
- **Cloud Server Security:**
  1. **CLOUD-NFR-101-P1:** The Cloud Server shall **maintain PCI DSS Level 1 compliance** for payment processing.
  2. **CLOUD-NFR-102-P1:** The Cloud Server shall **implement role-based access control** for all data access.
- **Local Server Security:**
  1. **LOCAL-NFR-101-P1:** The Local Server shall **encrypt cached data at rest**.
  2. **LOCAL-NFR-102-P1:** The Local Server shall **authenticate all client connections**.

5.3. Reliability & Availability

- **Local Server Reliability:**
  1. **LOCAL-NFR-201-P1:** The Local Server shall **ensure menu browsing and order building during internet outages of up to 8 hours**.
  2. **LOCAL-NFR-202-P1:** The Local Server shall **automatically recover after power restoration**.
- **Cloud Server Availability:**

1. **CLOUD-NFR-201-P1:** The Cloud Server shall **maintain 99.9% uptime** excluding scheduled maintenance.
2. **CLOUD-NFR-202-P1:** The Cloud Server shall **implement automatic failover** for critical services.

5.4. Usability Requirements

- **Mobile App Usability:**
  1. **MOBILE-NFR-301-P1:** The Mobile App shall **be usable with minimal training or instructions**.
  2. **MOBILE-NFR-302-P1:** The Mobile App shall **provide contextual help** for complex features.
- **Web App Usability:**
  1. **WEB-NFR-001-P1:** The Web App shall **support one-tap actions** for common operations during peak hours.
  2. **WEB-NFR-002-P1:** The Web App shall **provide clear visual indicators** for order priority.

5.5. Compatibility Requirements

1. **NFR-COMP-001:** Mobile apps shall support **iOS 14+** and **Android 8+**.
   - **Priority:** H
2. **NFR-COMP-002:** Web interfaces shall support **Chrome, Safari, Firefox, and Edge** (latest versions).
   - **Priority:** H
3. **NFR-COMP-003:** APIs shall support **JSON** data format with **UTF-8** encoding.
   - **Priority:** H

5.6. Scalability Requirements

1. **NFR-SCAL-001:** Database architecture shall support **horizontal partitioning** by restaurant and date.
   - **Priority:** H
2. **NFR-SCAL-002:** The system shall implement **caching layers** for frequently accessed data.
   - **Priority:** H
3. **NFR-SCAL-003:** Microservices architecture shall allow **independent scaling of components**.
   - **Priority:** M

5.7. Maintainability Requirements

1. **NFR-MAIN-001:** Codebase shall maintain **80%+ test coverage** for critical paths.
   - **Priority:** M
2. **NFR-MAIN-002: Comprehensive logging and monitoring** shall be implemented.
   - **Priority:** H

3. **NFR-MAIN-003:** API documentation shall be **automatically generated and maintained**.
    - **Priority:** M

6. SUCCESS METRICS & ACCEPTANCE CRITERIA - COMPONENT-ALIGNED

6.1. Key Performance Indicators

- **Mobile App KPIs:**
    1. **KPI-MOBILE-001: Customer Adoption Rate** - Target: 40% of restaurant visitors using digital ordering (P1)
    2. **KPI-MOBILE-002: App Store Rating** - Target: 4.5+ stars average rating
- **Local Server KPIs:**
    1. **KPI-LOCAL-001: Offline Operation Success** - Target: 99% successful order processing during outages
    2. **KPI-LOCAL-002: Sync Performance** - Target: <30 second sync time when connectivity restored
- **Cloud Server KPIs:**
    1. **KPI-CLOUD-001: System Uptime** - Target: 99.9% platform availability
    2. **KPI-CLOUD-002: Payment Success Rate** - Target: 95% first-time payment success

6.2. Acceptance Criteria

- **Component-Specific Acceptance:**
    1. **AC-MOBILE-001:** All Mobile App high-priority requirements implemented with 95% test coverage
    2. **AC-LOCAL-001:** Local Server offline functionality validated through simulated 8-hour outage test
    3. **AC-CLOUD-001:** Cloud Server payment integration tested with 99.9% transaction success rate
    4. **AC-WEB-001:** Web App staff interfaces validated with restaurant staff user testing

7. APPENDICES

APPENDIX A: Process Flows & Use Cases

- Your Use Case Diagrams (all 8 diagrams with systematic naming)
- Process flow descriptions for all user journeys
- User journey maps with swimlanes
- Error handling and exception workflows

APPENDIX B: Data Models & Dictionary

- Your Data Dictionary (table relationships, field definitions)
- Your PostgreSQL DDL (complete schema scripts)
- Data retention policies and archiving strategies
- Data migration plans

APPENDIX C: Requirements Traceability

- Your Use Case-Requirements Matrix (P1, P2, P3)
- Your Implementation Priority Matrix (quadrant analysis)
- Your Sprint Planning Matrix (by component team)
- Requirements validation and verification methods

APPENDIX D: API Specifications

- REST API endpoints with full documentation
- Request/response payload examples
- Authentication and authorization protocols
- WebSocket connections for real-time features

APPENDIX E: System Architecture

- Component architecture diagrams
- API contracts between components
- Data flow diagrams
- Security protocols between components

APPENDIX F: Deployment & Operations

- Mobile App deployment guides
- Local Server hardware setup
- Cloud infrastructure provisioning
- Monitoring and maintenance procedures

APPENDIX G: Testing Strategy

- Test automation frameworks by component
- Integration test scenarios
- Performance testing methodology
- Security testing protocols

APPENDIX H: Risk Assessment

- Technical, business, and security risks
- Mitigation strategies and contingency plans
- Risk monitoring and escalation procedures

APPENDIX I: Success Metrics & KPIs

- ~~Key Performance Indicators by component~~
- ~~Measurement methodologies and tools~~
- ~~Reporting schedules and dashboards~~
- ~~Continuous improvement feedback loops~~

APPENDIX J: Training & Documentation

- ~~Staff training materials and schedules~~
- ~~Customer onboarding guides~~
- ~~Technical documentation for developers~~
- ~~Troubleshooting guides and FAQs~~
- ~~Support escalation procedures~~

8. APPROVAL

This Requirements Definition Document has been reviewed and approved by:

Role	Name	Signature	Date
Product Manager			
Technical Lead			
Quality Assurance Manager			
Project Sponsor			
Restaurant Representative			

9. DOCUMENT REVISION HISTORY

Version	Date	Changes	Author
1.0	2025-09-25	Initial draft	Product Team
2.0	2025-10-15	Comprehensive restructuring with error handling, priorities, and professional format	Product Team