Park Zone

Software Requirements Specifications

Group 7

Revision History

Date	Revision	Description	Author
mm/dd/yyyy	1.0	Initial Version	Your Name

Revision History..... 2
1. Purpose..... 7
1.1. Scope..... 7
1.2. Definitions, Acronyms, Abbreviations..... 8
1.3. References..... 8
1.4. Overview..... 8
2. Overall Description..... 8
2.1. Product Perspective..... 8
2.2. Product Architecture..... 8
2.3. Product Functionality/Features..... 8
2.4. Constraints..... 8
2.5. Assumptions and Dependencies..... 8
3. Functional Requirements..... 8
3.1. Server-Side Core Systems..... 9
3.1.1. Database Management & Real-Time Tracking:..... 9
3.1.2. Payment Processing Backend:..... 9
3.1.3. Reservation Logic & Validation:..... 9
3.1.4. Report Generation Engine:..... 9
3.2. Client Application Interface..... 9
3.2.1 Parking Space Selector GUI:..... 9
3.2.2 Payment Interface & Forms:..... 9
3.2.3 Reservation Booking Interface:..... 9
3.2.4 User Dashboard & Profile Management:..... 9
3.3. System Integration Requirements..... 10
3.3.1 Server-Client Communication Protocols:..... 10
3.3.2 Real-Time Data Synchronization:..... 10
3.3.3 Offline Functionality Requirements:..... 10
3.3.4 External System Interfaces (payment gateways, hardware):..... 10
3.4 User Experience Features..... 10
3.4.1 Ticket Generation & Display:..... 10
3.4.2 Notification Systems (overstay alerts, confirmations):..... 10
3.4.3 Rewards & Gamification Elements:..... 10

3.4.4 User Reporting Capabilities:..... 10
4. Non-Functional Requirements..... 11
4.1. Security and Privacy Requirements..... 11
4.2. Environmental Requirements..... 11
4.3. Performance Requirements..... 11

1. Purpose

ParkZone exists to eliminate the frustration of finding parking by connecting drivers with available parking spaces in real-time while maximizing revenue for parking garage operators through intelligent space management and automated billing systems.

1.1. Scope

This document defines the requirements for developing ParkZone, a comprehensive parking management system consisting of:

- **Server Application**: Backend systems managing databases, payments, reservations, and reporting
- **Client Application**: User-facing mobile/web interface for customers and garage operators
- **Hardware Integration**: Sensors, payment terminals, and access control systems

What's included:

- Real-time parking availability tracking
- Reservation and payment systems
- User profiles and rewards program
- Operator dashboards and reporting
- Multi-level garage support

What's excluded:

- Hardware manufacturing
- Physical installation services
- Integration with municipal parking systems (Phase 2)

1.2. Definitions, Acronyms, Abbreviations

- **GUI**: Graphical User Interface
- **API**: Application Programming Interface
- **SRS**: Software Requirements Specification
- **Overstay**: Parking beyond the paid/reserved time limit
- **Grace Period**: 5-minute buffer before penalties apply
- **Garage Operator**: Business owner managing parking facilities

- **End User**: Driver seeking parking

1.3. References

- IEEE Std 830-1998 (Software Requirements Specifications)
- ParkZone Project Kickoff Presentation
- Team meeting notes and requirements gathering sessions

---

2. Overall Description

2.1. Product Perspective

ParkZone is a new, standalone system that bridges the gap between parking supply and demand. The system operates as:

- **Customer-facing application** (mobile/web) for finding and reserving spots
- **Operator management console** for garage owners
- **Backend infrastructure** connecting all components
- **Integration layer** for payment processors and hardware sensors

2.2. Product Architecture

ParkZone is a new, standalone system that bridges the gap between parking supply and demand. The system operates as:

- **Customer-facing application** (mobile/web) for finding and reserving spots
- **Operator management console** for garage owners
- **Backend infrastructure** connecting all components
- **Integration layer** for payment processors and hardware sensors

2.3. Product Functionality/Features

Java Client-Server Architecture with Centralized Coordinator:

Server Application (Java) - Single-Threaded Coordinator:

- **ParkingServer** class manages ALL space updates in one thread
- Maintains master list of available/occupied spaces
- Processes all client requests sequentially (no race conditions)
- Broadcasts updates to all connected clients immediately
- File-based persistence for garage state

2.4. Constraints

Java Client-Server Architecture with Centralized Coordinator:

Server Application (Java) - Single-Threaded Coordinator:

- **ParkingServer** class manages ALL space updates in one thread
- Maintains master list of available/occupied spaces
- Processes all client requests sequentially (no race conditions)
- Broadcasts updates to all connected clients immediately

- File-based persistence for garage state

2.5. Assumptions and Dependencies

Assumptions:

- Parking garages have basic internet connectivity
- Users have smartphones with GPS capabilities
- Garage operators are willing to adopt new technology
- Sufficient server infrastructure can be secured

Dependencies:

- Third-party payment processor APIs (Stripe, PayPal)
- Cloud hosting services (AWS, Azure, or Google Cloud)
- GPS and mapping services
- Hardware sensor integration capabilities

---

3. Specific Requirements

3.1. Server-Side Core Systems

3.1.1. Database Management & Real-Time Tracking

- **FR-001**: System shall maintain real-time inventory of all parking spaces
- **FR-002**: System shall update space availability within 5 seconds of status change
- **FR-003**: System shall store user profiles, transaction history, and garage data
- **FR-004**: System shall support multi-level garage configurations

3.1.2. Payment Processing Backend

- **FR-005**: System shall process cash and card payments
- **FR-006**: System shall calculate fees based on duration, vehicle type, and pricing rules
- **FR-007**: System shall handle refunds for canceled reservations
- **FR-008**: System shall generate payment receipts and transaction logs

3.1.3. Reservation Logic & Validation

- **FR-009**: System shall allow users to reserve spaces up to 24 hours in advance
- **FR-010**: System shall release reservations after 5-minute grace period
- **FR-011**: System shall prevent double-booking of spaces
- **FR-012**: System shall send confirmation notifications for reservations

3.1.4. Report Generation Engine

- **FR-013**: System shall generate daily/weekly/monthly usage reports
- **FR-014**: System shall track revenue, occupancy rates, and user behavior
- **FR-015**: System shall identify overstay incidents and violations
- **FR-016**: System shall export reports in PDF and CSV formats

3.2. Client Application Interface

3.2.1 Parking Space Selector GUI

- **FR-017:** Interface shall display available spaces in real-time
- **FR-018:** Interface shall allow filtering by accessibility and EV charging
- **FR-019:** Interface shall show pricing information before selection
- **FR-020:** Interface shall provide garage layout visualization

3.2.2 Payment Interface & Forms

- **FR-021:** Interface shall support multiple payment methods
- **FR-022:** Interface shall calculate total costs before payment
- **FR-023:** Interface shall store preferred payment methods securely
- **FR-024:** Interface shall handle payment failures gracefully

3.2.3 Reservation Booking Interface

- **FR-025**: Interface shall allow time-based reservations
- **FR-026**: Interface shall show reservation confirmations
- **FR-027**: Interface shall allow reservation modifications/cancellations
- **FR-028**: Interface shall display countdown timers for reserved spaces

3.2.4 User Dashboard & Profile Management

- **FR-029**: System shall maintain user profiles with vehicle information
- **FR-030**: System shall display parking history and receipts
- **FR-031**: System shall show rewards points and available discounts
- **FR-032**: System shall allow users to report issues and violations

3.3. System Integration Requirements

- **FR-033**: System shall use HTTPS for all client-server communication
- **FR-034**: System shall implement WebSocket connections for real-time updates
- **FR-035**: System shall provide RESTful APIs for all operations
- **FR-036**: System shall handle network interruptions gracefully

3.3.1 Server-Client Communication Protocols

- **FR-037**: System shall sync data across all client instances within 10 seconds
- **FR-038**: System shall maintain data consistency during high-traffic periods
- **FR-039**: System shall queue updates during temporary network outages
- **FR-040**: System shall resolve data conflicts using timestamp precedence

3.3.2 Real-Time Data Synchronization

- **FR-041**: System shall integrate with payment gateway APIs
- **FR-042**: System shall connect with garage sensor hardware
- **FR-043**: System shall interface with access control systems
- **FR-044**: System shall support third-party mapping services

3.3.3 External System Interfaces

- **FR-045**: System shall send push notifications for reservations
- **FR-046**: System shall alert users of overstay situations
- **FR-047**: System shall notify operators of system issues
- **FR-048**: System shall send promotional offers to frequent users

3.4 User Experience Features

3.4.1 External System Interfaces

- **FR-045**: System shall send push notifications for reservations
- **FR-046**: System shall alert users of overstay situations
- **FR-047**: System shall notify operators of system issues
- **FR-048**: System shall send promotional offers to frequent users

3.4.2 Rewards & Gamification Elements

- **FR-049**: System shall award points based on usage duration
- **FR-050**: System shall allow points redemption for free parking
- **FR-051**: System shall provide loyalty tier benefits
- **FR-052**: System shall track and display usage statistics

4. Non-Functional Requirements

4.1. Security and Privacy Requirements

Data Protection

// Sophia
—
10:06 PM
for some part we need to take out encryption, the professor said we are not doing any of that.

- **NFR-001**: System shall encrypt all sensitive data using AES-256 encryption
- **NFR-002**: System shall comply with PCI DSS standards for payment data
- **NFR-003**: System shall implement GDPR-compliant data handling
- **NFR-004**: System shall provide data deletion capabilities for user requests

Authentication and Authorization

- **NFR-005**: System shall require strong password policies (8+ characters, mixed case, numbers)
- **NFR-006**: System shall implement multi-factor authentication for operators
- **NFR-007**: System shall use role-based access control (customer, operator, admin)
- **NFR-008**: System shall automatically log out inactive sessions after 30 minutes

System Security

- **NFR-009**: System shall log all security-relevant events
- **NFR-010**: System shall implement rate limiting to prevent abuse
- **NFR-011**: System shall use HTTPS/TLS 1.3 for all communications
- **NFR-012**: System shall regularly update and patch security vulnerabilities

4.2. Environmental Requirements

Hardware Environment

- **NFR-013**: Client app shall run on iOS 14+ and Android 8+
- **NFR-014**: System shall operate with minimum 4G/LTE connectivity
- **NFR-015**: Server shall run on cloud infrastructure (AWS/Azure/GCP)
- **NFR-016**: System shall support both portrait and landscape orientations

Software Environment

- **NFR-017**: Server shall be compatible with Linux-based operating systems
- **NFR-018**: Database shall use PostgreSQL or equivalent relational database
- **NFR-019**: System shall support modern web browsers (Chrome 90+, Safari 14+, Firefox 88+)
- **NFR-020**: System shall be deployment-ready using containerization (Docker)

Integration Environment

- **NFR-021**: System shall integrate with existing garage management systems
- **NFR-022**: System shall support common sensor protocols (REST APIs, MQTT)
- **NFR-023**: System shall work with standard payment terminals
- **NFR-024**: System shall accommodate various garage layouts and configurations

4.3. Performance Requirements

Response Time

- **NFR-025**: System shall respond to user actions within 2 seconds under normal load
- **NFR-026**: Real-time updates shall propagate within 5 seconds
- **NFR-027**: Payment processing shall complete within 10 seconds
- **NFR-028**: Report generation shall complete within 30 seconds for standard reports

Throughput and Scalability

- **NFR-029**: System shall support minimum 1,000 concurrent users
- **NFR-030**: System shall handle 10,000+ parking transactions per day
- **NFR-031**: Database shall support horizontal scaling for growth
- **NFR-032**: System shall maintain performance during peak usage (weekends, events)

Availability and Reliability

- **NFR-033**: System shall maintain 99.5% uptime during business hours
- **NFR-034**: System shall implement automatic failover for critical components
- **NFR-035**: System shall perform automated daily backups
- **NFR-036**: System shall recover from failures within 15 minutes

Resource Usage

- **NFR-037**: Mobile app shall consume less than 100MB storage space
- **NFR-038**: Mobile app shall use minimal battery power during background operation
- **NFR-039**: Server shall efficiently handle database queries without performance degradation
- **NFR-040**: System shall optimize bandwidth usage for real-time updates