System Design Document (Sprint 3 Update)

1. Introduction

This document outlines the system design, including key components, architecture, dependencies, and error-handling strategies for Sprint 3. This includes High-Level System Architecture as well as CRC cards. This document is as of Sprint 3 and is subject to change in the future.

2. System Overview

Primary Features in Sprint 3:

- Adding and Retrieving Listings
  - Adding new Listings
    - Image Upload
  - Searching Listings based on parameter
  - Retrieving all Listings belonging to a user
  - Auto-recommendations of parking spots based on user preferences
- Geospatial search
- Integration of new Data into Map
- User Preferences Management (Preference Storage & Retrieval)

Goal:

Provide the enhanced functionality for adding listings, searching/retrieving listings based on user preferences, location, and improved auto-recommendation of parking spots.

3. System Interaction & Dependencies

3.1 Operating Environment

Component	Technology Used
Platform	Mobile Application (Expo iOS App)
Backend	Node.js with Express.js
Frontend	React Native (Expo)
Database	MongoDB with Mongoose ORM

Component	Technology Used
Network Configuration	HTTPS API, Cloud-based MongoDB Atlas
External APIs	Google Maps API (Navigation & Location), Firebase (Auth & Image Storage), Stripe/PayPal (Payments)

3.2 Assumptions & Constraints

- Users must have an internet connection to access services.
- External services (Google Maps, Payments, Firebase, MongoDB) must be operational.
- The database must hold entries for listings to provide search functionality.
- User preferences must be stored in the backend to enable auto-recommendation of spots.

4. System Architecture

4.1 Abstract View & Component Diagram

(Currently missing - Update with latest architecture diagram)

4.2 Components & Responsibilities

Component	Responsibilities
User Management	Handles authentication & user profile management
Authentication Service	Manages user login & JWT tokens
Parking Spot Management	Stores and updates parking spots, availability
Preferences Management	Stores user preferences for auto-recommendations
Recommendation System	Suggests best parking spots based on preferences
Booking System	Handles reservations, prevents double booking
Interactive Map	Displays available spots, navigation support

5. CRC Cards for Sprint 3

(Update diagrams to reflect new components like Preferences Management and Recommendation System)

6. System Decomposition & Error Handling

6.1 Error Handling Strategy

Error Scenario	Handling Strategy
Invalid User Input	Validate fields before submission, return meaningful error messages
Authentication Failure	Lock account after multiple failed login attempts, notify user
Double Booking Attempt	Check availability before confirming reservation, prevent conflicts
API Failure (Google Maps, Payment)	Implement retries, display fallback messages to users
Network Failure	Cache essential data, show 'Service Unavailable' message

6.2 Mapping System Architecture to Components

- **Frontend (React Native / Expo):** User authentication, map interaction, booking UI, preference management UI.
- **Backend (Node.js/Express.js):** API to handle authentication, reservations, preferences management, payments.
- **Database (MongoDB):** Stores user data, parking spots, user preferences, and bookings.

7. Next Steps

- Improve UI for preferences management in the Expo app.
- Implement navigation functionality.
- Improve search criteria to better match user preferences.
- Show listing details on a map with user-specific suggestions.