SOFTWARE REQUIREMENTS SPECIFICATION

Confidential & Technical Document

SmartDine

Restaurant Reservation & Ordering Mobile App

Document Title: Software Requirements Specification
Project Name: SmartDine by Khan
Version: 1.0.0
Status: Final Draft
Date: February 2026
Authors: SmartDine Development Team
Platform: Android & iOS (React Native / Expo)

Created By: Muhammad Fazil
Roll Number: 9179

Table of Contents

1. Introduction

1.1 Purpose

This Software Requirements Specification (SRS) document describes the functional and non-functional requirements for SmartDine, a cross-platform mobile application designed for restaurant management, table reservations, digital ordering, and customer loyalty programs. This document serves as the authoritative reference for developers, designers, QA engineers, and project stakeholders throughout the software development lifecycle.

1.2 Scope

SmartDine is a full-stack mobile application built with React Native (Expo) for the frontend and Node.js/Express for the backend. The system encompasses the following functional areas:

- Customer-facing mobile app for browsing menus, placing orders, and managing table reservations
- Real-time order tracking and status notifications
- Loyalty program with tiered rewards and points redemption
- Admin panel for restaurant staff to manage orders, tables, menus, and reservations
- Secure authentication and user profile management
- Payment processing integration with multiple payment methods

1.3 Definitions, Acronyms, and Abbreviations

Term / Acronym	Definition
SRS	Software Requirements Specification
UI	User Interface
UX	User Experience
API	Application Programming Interface
REST	Representational State Transfer
JWT	JSON Web Token — used for stateless authentication
QR Code	Quick Response Code — used for order/reservation verification
Expo	Open-source platform for building universal React Native apps
Zustand	Lightweight state management library for React
OTP	One-Time Password
CRUD	Create, Read, Update, Delete — standard data operations
POS	Point of Sale
MVP	Minimum Viable Product
FR	Functional Requirement
NFR	Non-Functional Requirement

1.4 Overview

This document is organized into the following sections:

1. Introduction — project background, scope, and definitions
2. Overall Description — product perspective, user classes, and constraints
3. System Features — detailed functional requirements for each module
4. External Interface Requirements — UI, hardware, software, and communications
5. Non-Functional Requirements — performance, security, scalability, and usability
6. System Architecture — technology stack and component diagram
7. Data Requirements — entity descriptions and data models
8. Appendix — use cases, revision history

2. Overall Description

2.1 Product Perspective

SmartDine is a standalone mobile application that replaces traditional paper-based restaurant workflows. It integrates with a RESTful backend API and supports both online and offline-first usage patterns. The system is designed to operate on Android and iOS devices running within the Expo Go environment or as a standalone compiled application.

- Frontend: React Native (Expo SDK 51) with TypeScript
- Backend: Node.js + Express.js REST API
- State Management: Zustand
- Navigation: React Navigation (Stack + Bottom Tabs)
- Auth: JWT-based token authentication
- Deployment: Expo Application Services (EAS)

2.2 Product Functions

SmartDine provides the following high-level functions:

Customer Functions

- User registration, login, and profile management
- Browse restaurant menu with filtering, searching, and sorting
- View detailed dish information including allergens and nutritional data
- Add items to cart with quantity control and coupon application
- Select dine-in or takeaway ordering mode
- Complete checkout with tip selection and payment method choice
- Book tables via a 3-step reservation wizard (date → table map → details)
- Track live order status through an animated timeline
- View and manage order history
- Submit star ratings and written reviews
- Manage loyalty points, view tier status, and redeem rewards
- Toggle dark/light mode and manage notification preferences

Admin Functions

- View dashboard with revenue charts and key metrics
- Manage all active and historical orders with status updates
- Approve or reject reservation requests
- Add, edit, toggle availability, and delete menu items
- Monitor real-time table status on a visual map

2.3 User Classes and Characteristics

User Class	Description	Technical Proficiency	Access Level
Guest User	Unauthenticated visitor viewing public menu	Low	Read-only menu
Registered Customer	Logged-in user who can order and book tables	Low–Medium	Full customer features
Restaurant Admin	Staff managing orders, tables, and menu	Medium	Admin panel + all customer features
Super Admin	Owner with full system access and analytics	High	Unrestricted

2.4 Operating Environment

- Mobile OS: Android 8.0 (API 26) and above; iOS 13.0 and above
- Development Runtime: Expo SDK 51, React Native 0.74
- Backend Runtime: Node.js 18+ with Express.js
- Network: Requires active WiFi or cellular data for real-time features; supports offline demo mode for login
- Database: JSON-based mock data layer (production: PostgreSQL or MongoDB)

2.5 Design and Implementation Constraints

- The application must run on Expo Go without native module ejection for development builds
- The backend API must be RESTful and stateless, using JWT for all authenticated requests
- @expo/vector-icons is incompatible with Expo SDK 51 due to expo-font API changes; a custom emoji-based Icon component is used instead
- Axios v1.7+ is incompatible with React Native due to Node.js crypto module dependency; native fetch() is used
- All monetary values are stored and transmitted in USD with two decimal precision
- QR codes are generated client-side using react-native-qrcode-svg
- The application does not use browser storage APIs (localStorage/sessionStorage)

2.6 Assumptions and Dependencies

- Users have a stable internet connection for real-time order tracking
- Restaurant staff have access to a tablet or smartphone running the admin panel
- The backend server is accessible on the local network during development
- Push notifications require Expo Notifications and Firebase Cloud Messaging in production
- Production payment processing requires Stripe API integration

3. System Features & Functional Requirements

3.1 Authentication Module

The authentication module handles user identity, session management, and access control.

FR-AUTH-01: User Registration

Attribute	Detail
Requirement ID	FR-AUTH-01
Priority	High
Description	Users shall be able to create a new account by providing their full name, email address, password, and optional phone number.
Input	Name (required), Email (required, unique), Password (min 6 chars), Phone (optional)
Output	Account created; JWT token issued; user redirected to Home screen
Validation	Email format validation, password strength check, duplicate email detection
Error Cases	Duplicate email → 409 Conflict; invalid format → 422 Unprocessable Entity

FR-AUTH-02: User Login

Attribute	Detail
Requirement ID	FR-AUTH-02
Priority	High
Description	Registered users shall be able to log in using their email and password. The system shall support an offline demo mode with pre-seeded credentials.
Input	Email address, Password
Output	JWT access token stored in Zustand state; navigation to Main tab stack
Demo Credentials	Email: alex@example.com / Password: password123
Fallback	If backend unreachable, demo credentials bypass network and log in locally
Security	Shake animation on failed attempt; max 5 attempts before 30-second lockout (production)

FR-AUTH-03: Password Recovery

- Users shall be able to request a password reset by entering their registered email address
- The system shall display a confirmation screen indicating the email has been sent
- Reset link expires after 24 hours (production implementation)

FR-AUTH-04: Session Management

- JWT tokens are stored in Zustand in-memory state (cleared on app restart)
- Authenticated routes are protected via navigation guards in the stack navigator
- Logout clears all user state and navigates to the Login screen

3.2 Menu Browsing Module

FR-MENU-01: Menu Display

The system shall display all available menu items organized by category with the following information for each item:

- Item name, description, and high-resolution image
- Category tag (Starters, Mains, Desserts, Cocktails, Wine)
- Price in USD
- Preparation time in minutes
- Caloric content
- Average rating (1–5 stars) and total review count
- Allergen information and dietary tags (Vegan, Gluten-Free, Chef's Pick, etc.)
- Availability status — unavailable items show an overlay and cannot be added to cart

FR-MENU-02: Search and Filter

Filter Type	Options	Behavior
Category	All, Starters, Mains, Desserts, Cocktails, Wine	Single select; filters list in real time
Search	Free-text input	Searches name and description fields; case-insensitive
Sort	Featured, Rating (High to Low), Price (Low to High), Price (High to Low), Most Popular	Applied after category filter
Results Count	Dynamic display	"X items found" updates with each filter change

FR-MENU-03: Dish Detail Screen

- Tapping any dish opens a full-screen detail view with hero image
- Users can adjust quantity (1–20) before adding to cart
- Favorite toggle (heart icon) persists to Zustand favorites state
- Reviews section shows aggregated rating and links to full reviews
- Add to Cart button triggers animated checkmark confirmation feedback

3.3 Shopping Cart Module

FR-CART-01: Cart Management

- Items added from Menu or Dish Detail screens accumulate in the cart

- Quantity can be increased or decreased; reaching zero removes the item
- Cart persists within the app session (cleared on logout or explicit clear)
- Cart item count badge displayed on the Orders tab bar icon

FR-CART-02: Order Type Selection

- Users shall toggle between Dine-In and Takeaway modes
- Dine-In mode activates table number field input
- Takeaway mode shows pickup location information

FR-CART-03: Coupon System

Coupon Code	Discount Type	Value
CHEF20	Percentage	20% off subtotal
SAVE10	Percentage	10% off subtotal
BRUNCH24	Fixed Amount	$15.00 off

- Users enter coupon codes in a text field and press Apply
- Valid coupons display the discount amount in the order summary
- Invalid codes display an inline error message
- Only one coupon can be applied per order

FR-CART-04: Order Summary

- Subtotal calculated from all cart items × quantities
- Discount amount applied after coupon validation
- Tax calculated at 10% of discounted subtotal
- Total = Subtotal − Discount + Tax + Tip

3.4 Checkout & Payment Module

FR-PAY-01: Checkout Flow

9. Cart review → Checkout screen (tip selection)
10. Checkout → Payment screen (method selection + card details)
11. Payment → Order Success screen (confirmation + QR code)

FR-PAY-02: Tip Selection

- Pre-defined tip options: 10%, 15%, 18%, 20%, 25%
- Tip calculated on discounted subtotal (before tax)
- Selected tip highlighted; user can change selection before proceeding

FR-PAY-03: Payment Methods

Method	Status	Notes
Credit / Debit Card	UI Implemented	Card number, expiry, CVV, name fields with animated preview
Apple Pay	UI Placeholder	Full integration requires Expo In-App Purchases
Google Pay	UI Placeholder	Full integration requires Google Pay API
Cash on Delivery	Functional	No card input required; payment at pickup/table

FR-PAY-04: Order Confirmation

- On successful checkout, order is created via POST /checkout API
- Cart is cleared from Zustand state
- Loyalty points are credited: 1 point per $1 spent + 50 bonus points
- Order ID and QR code are displayed on the success screen
- User can navigate to Order Tracking or return to Home

3.5 Table Reservation Module

FR-RES-01: 3-Step Reservation Wizard

The reservation flow is implemented as a guided 3-step wizard:

Step	Screen Element	Validation
Step 1: Date & Time	Guest count selector (1–8), horizontal date picker (14 days), time slot grid	Date and time slot must be selected
Step 2: Table Selection	Visual table map with color-coded status (Available/Occupied/Reserved)	An available table must be selected
Step 3: Details	Booking summary card, special requests text area (multiline)	No required fields; optional special requests

FR-RES-02: Table Status Color Coding

- Green — Available: can be selected
- Red — Occupied: cannot be selected
- Orange — Reserved: cannot be selected
- Gold border — Currently selected table

FR-RES-03: Reservation Confirmation

- Successful reservation creates a booking record via POST /reservation API
- Confirmation screen displays: date, time, guest count, table number, status
- A unique confirmation code is generated and displayed
- QR code is generated for presentation at the restaurant

FR-RES-04: Reservation Cancellation

- Users can cancel pending or confirmed reservations from order history
- Cancellation sends PATCH /reservation/:id/cancel to the API
- Cancelled status is reflected immediately in the UI

3.6 Order Tracking Module

FR-TRACK-01: Live Order Timeline

The order tracking screen displays a 5-step animated status timeline:

Step	Status	Description
1	Placed	Order received by the system
2	Confirmed	Restaurant has confirmed the order
3	Preparing	Kitchen is preparing the order
4	Ready	Order is ready for pickup or serving
5	Delivered	Order delivered to customer

- Completed steps display gold connector lines and checkmark icons
- Current active step displays a pulsing animation
- Estimated delivery time is displayed at the top of the screen
- Users can cancel an active order (confirmation dialog required)

3.7 Reviews & Ratings Module

FR-REV-01: View Reviews

- All users can view reviews for individual menu items or the restaurant overall
- Reviews display: reviewer name, date, star rating (1–5), review text
- Aggregate rating shown as numeric average with total review count
- Helpful vote count displayed per review

FR-REV-02: Submit Review

- Authenticated users can submit one review per dish per order
- Review form requires a star rating (1–5) and written text
- Submitted reviews appear immediately in the list after API confirmation

3.8 Loyalty Program Module

FR-LOY-01: Points System

Action	Points Earned
$1 spent on any order	1 point

Action	Points Earned
Completing a reservation	50 points
Writing a review	25 points
Referring a new user	100 points

FR-LOY-02: Membership Tiers

Tier	Min Points	Key Benefits
Bronze	0	5% off orders, birthday surprise
Silver	1,000	10% off orders, free dessert monthly, priority reservations
Gold	2,500	15% off orders, free wine pairing, chef's table access
Platinum	5,000	20% off all orders, monthly chef's dinner, airport VIP service

FR-LOY-03: Rewards Redemption

- Users can redeem points for rewards (Free Starter, Free Dessert, Wine Bottle, $25 Voucher, Private Dining)
- Points are deducted from balance upon redemption confirmation
- Insufficient points disables the Redeem button and shows required amount

3.9 Profile & Settings Module

FR-PROF-01: User Profile

- Displays avatar, name, email, tier badge, and loyalty statistics
- Quick navigation to Edit Profile, Saved Addresses, Order History, Loyalty, Favorites, Settings
- Admin button visible for admin-role users to access admin dashboard

FR-PROF-02: Settings

- Dark Mode / Light Mode toggle (persisted in Zustand state)
- Push Notification toggle
- Email Notification toggle
- Language selection: English, Español, Français, العربية
- Account management: Change Password, Biometric Login, Delete Account

3.10 Admin Panel Module

FR-ADMIN-01: Dashboard

- Key metrics: Today's Revenue, Total Orders, Tables Occupied/Total, Customer Satisfaction score
- Weekly revenue bar chart with day-by-day breakdown
- Active orders count and list with current status

- Pending reservations count and list

FR-ADMIN-02: Menu Manager

- List all menu items with name, category, price, and availability status
- Toggle item availability (Show/Hide from customer menu)
- Delete items with confirmation dialog
- Add new menu item via form (placeholder for production implementation)

FR-ADMIN-03: Orders Panel

- View all orders with order ID, type, item count, and total
- Update order status via horizontal scrollable status chip selector
- Available status transitions: Confirmed → Preparing → Ready → Delivered / Cancelled

FR-ADMIN-04: Reservations Manager

- View all reservations with date, time, table number, and guest count
- One-tap approve (green ✓) or reject (red ✗) for pending reservations
- Approved and rejected reservations show status chip

FR-ADMIN-05: Table Manager

- Grid view of all tables with shape-coded representations (round/rectangular)
- Color-coded by status: Available (green), Occupied (red), Reserved (orange)
- Tap any table to view details: capacity, location, shape, current status

4. External Interface Requirements

4.1 User Interface Requirements

- The application shall follow a luxury dark-themed design system with gold (#C9A96E) as the primary brand color
- All screens shall support both dark mode (#0D0D0D background) and light mode (#F8F5F0 background)
- Bottom tab navigation shall include: Home, Menu, Reservations, Orders, Profile
- All interactive elements shall provide visual feedback (color change, scale animation, haptic)
- Loading states shall display activity indicators or skeleton placeholder screens
- Empty states shall display descriptive messages and call-to-action buttons
- The minimum tap target size shall be 44×44 points as per iOS HIG and Android guidelines

4.2 Hardware Interfaces

- Camera: Optional — used for QR code scanning and profile photo capture (production)
- Biometric Sensor: Optional — fingerprint or Face ID for login (production)
- Network Interface: WiFi or cellular data required for all API operations
- Storage: Minimum 50MB free device storage for app installation

4.3 Software Interfaces

System	Interface Type	Purpose
REST API (Node.js/Express)	HTTP/JSON	All data operations — auth, menu, orders, reservations
Expo Go / EAS	SDK	Development runtime and production build system
React Navigation	Library	Screen navigation and deep linking
Zustand	Library	Global state management
react-native-svg	Library	QR code rendering
expo-linear-gradient	Expo Module	Gradient backgrounds and buttons
Stripe API (future)	HTTP/JSON	Payment processing in production
Firebase FCM (future)	SDK	Push notification delivery

4.4 Communications Interfaces

- All API communication uses HTTPS (HTTP during local development)
- Request/response format: application/json
- Authentication: Bearer token in Authorization header for protected endpoints
- Timeout: 8 seconds per API request before AbortController cancels the fetch
- CORS: Backend configured to accept requests from the mobile app origin

4.5 API Endpoint Reference

Method	Endpoint	Auth Required	Description
POST	/login	No	Authenticate user and return JWT token
POST	/register	No	Create new user account
GET	/menu	No	Fetch menu items (supports ?category, ?search, ?sort)
GET	/specials	No	Fetch today's special dishes
GET	/featured	No	Fetch featured/promoted dishes
GET	/promotions	No	Fetch active promotions and discounts
GET	/tables	Yes	Fetch all tables with availability status
GET	/time-slots	Yes	Fetch available reservation time slots for a date
POST	/reservation	Yes	Create a new table reservation
PATCH	/reservation/:id/cancel	Yes	Cancel an existing reservation
GET	/orders	Yes	Fetch order history for authenticated user
POST	/checkout	Yes	Place a new order with cart items
PATCH	/orders/:id/status	Yes	Update order status (admin)
GET	/reviews	No	Fetch reviews (optional ?menuItemId filter)
POST	/reviews	Yes	Submit a new review
POST	/validate-coupon	No	Validate a coupon code and return discount
GET	/admin/dashboard	Admin	Fetch dashboard statistics and charts
GET	/admin/orders	Admin	Fetch all orders across all users
GET	/admin/reservations	Admin	Fetch all reservations
PATCH	/admin/reservation/:id	Admin	Update reservation status (approve/reject)
POST	/admin/menu	Admin	Add a new menu item
PATCH	/admin/menu/:id	Admin	Update an existing menu item
DELETE	/admin/menu/:id	Admin	Remove a menu item from the menu

5. Non-Functional Requirements

5.1 Performance Requirements

- NFR-PERF-01: App launch time shall not exceed 3 seconds on mid-range devices (2GB RAM)
- NFR-PERF-02: Menu list shall render up to 100 items with no perceptible scroll lag (60fps)
- NFR-PERF-03: API responses shall complete within 2 seconds under normal network conditions
- NFR-PERF-04: Backend simulates realistic API latency (300–1,500ms) during development
- NFR-PERF-05: All screen transitions shall complete within 300ms using native-driver animations
- NFR-PERF-06: Images shall be lazy-loaded and cached to prevent repeated network requests

5.2 Security Requirements

- NFR-SEC-01: All passwords shall be hashed using bcrypt with a minimum salt factor of 10 (production)
- NFR-SEC-02: JWT tokens shall expire after 7 days; refresh token rotation required for production
- NFR-SEC-03: All API endpoints except login and register shall require a valid Bearer token
- NFR-SEC-04: Admin endpoints shall verify admin role claim in JWT payload
- NFR-SEC-05: Credit card data shall never be stored server-side; Stripe tokenization required in production
- NFR-SEC-06: All network traffic shall use TLS/HTTPS in production deployments
- NFR-SEC-07: Input validation shall be enforced on both client and server sides

5.3 Reliability & Availability

- NFR-REL-01: The application shall handle network failures gracefully with user-visible error messages
- NFR-REL-02: The demo login fallback ensures users can access the app even when the backend is unreachable
- NFR-REL-03: All API calls include an 8-second timeout with AbortController to prevent indefinite hangs
- NFR-REL-04: Production backend target uptime: 99.5% monthly
- NFR-REL-05: Database transactions shall be atomic to prevent partial order creation

5.4 Usability Requirements

- NFR-USE-01: New users shall be able to place their first order within 5 minutes without training
- NFR-USE-02: All primary actions shall be reachable within 3 taps from the home screen
- NFR-USE-03: Error messages shall be human-readable and include suggested remediation steps
- NFR-USE-04: The onboarding flow (3 slides) shall be skippable at any point
- NFR-USE-05: Form validation errors shall appear inline below the relevant input field

- NFR-USE-06: All screens shall be accessible without login except ordering and reservation screens

5.5 Maintainability & Scalability

- NFR-MAINT-01: All source code shall be written in TypeScript with strict mode enabled
- NFR-MAINT-02: Business logic shall be separated from UI components using the services layer
- NFR-MAINT-03: The API service layer (api.ts) shall be the single point of backend communication
- NFR-MAINT-04: Zustand store shall be the single source of truth for all shared UI state
- NFR-MAINT-05: The backend shall support horizontal scaling behind a load balancer in production
- NFR-MAINT-06: New menu categories and loyalty tiers shall be addable via admin panel without code changes

5.6 Portability

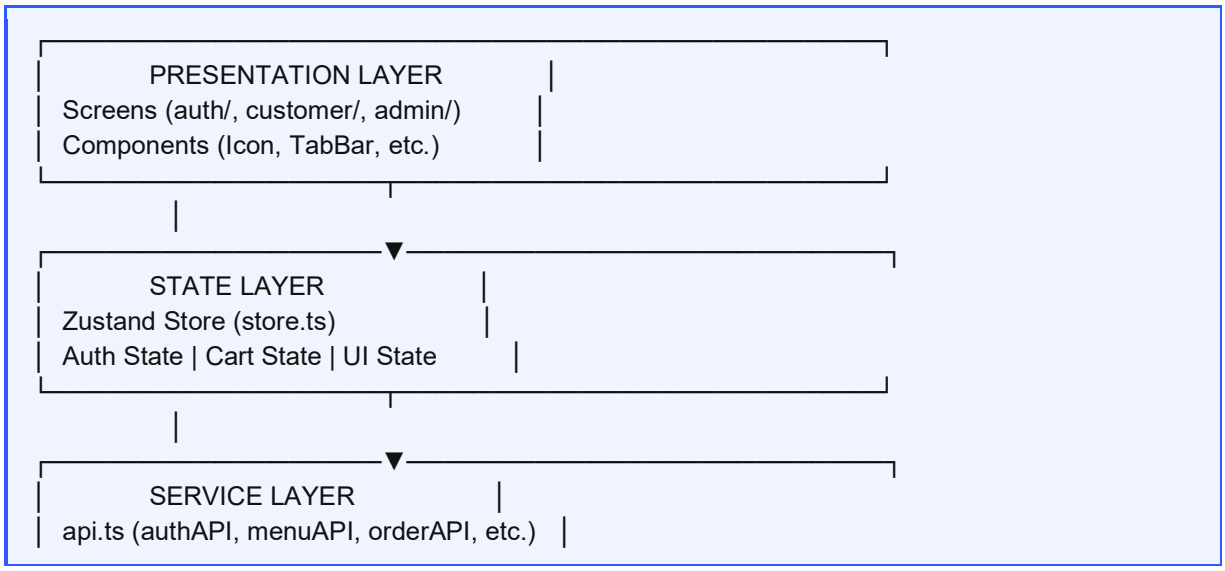
- NFR-PORT-01: The application shall render identically on Android 8.0+ and iOS 13.0+
- NFR-PORT-02: The application shall function on screen sizes from 4.7" to 6.7" without layout issues
- NFR-PORT-03: The backend API shall be deployable to any Node.js 18+ compatible hosting platform
- NFR-PORT-04: No platform-specific (iOS-only or Android-only) APIs shall be used without fallbacks

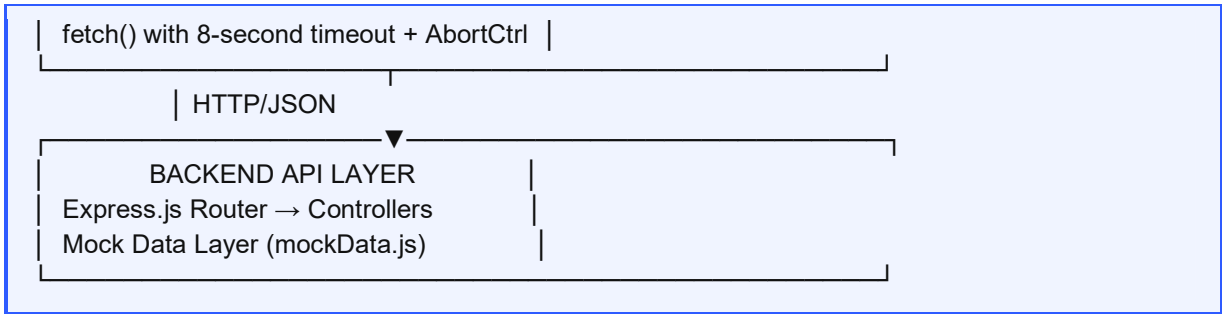
6. System Architecture

6.1 Technology Stack

Layer	Technology	Version	Purpose
Mobile App	React Native (Expo)	SDK 51 / RN 0.74	Cross-platform iOS & Android UI
Language	TypeScript	^5.3.0	Type-safe development
State Management	Zustand	^4.5.2	Global app state (auth, cart, theme)
Navigation	React Navigation	^6.x	Stack + Bottom Tab navigation
HTTP Client	Native fetch()	Built-in	API communication with timeout support
Animations	React Native Reanimated	~3.10.1	60fps native animations
Gradients	expo-linear-gradient	~13.0.2	UI gradient effects
QR Codes	react-native-qrcode-svg	^6.3.1	Order & reservation QR generation
Backend	Node.js + Express	18+ / ^4.x	REST API server
Mock Data	In-memory JSON	N/A	Development data layer
Prod Database	PostgreSQL (planned)	Latest	Persistent data storage
Auth	JWT (jsonwebtoken)	^9.x	Stateless authentication tokens
Build System	Expo EAS	Latest	iOS & Android production builds

6.2 Application Layer Architecture





6.3 Directory Structure

```
smartdine/
├── app/                    # React Native Frontend
│   ├── App.tsx             # App entry point
│   ├── app.json            # Expo configuration
│   ├── babel.config.js     # Babel with @/ path alias
│   ├── tsconfig.json       # TypeScript config
│   ├── metro.config.js     # Metro bundler config
│   ├── assets/             # Static images & icons
│   ├── components/
│   │   └── Icon.tsx        # Emoji icon component
│   ├── navigation/
│   │   └── index.tsx       # Stack + Tab navigators
│   ├── services/
│   │   ├── api.ts          # All API calls (fetch)
│   │   └── store.ts        # Zustand global state
│   ├── theme/
│   │   └── index.ts        # Colors, spacing, shadows
│   └── screens/
│       ├── auth/           # 5 auth screens
│       ├── customer/       # 15 customer screens
│       └── admin/          # 5 admin screens
└── backend/
    ├── server.js           # Express API (27 endpoints)
    ├── mockData.js         # Seed data
    └── package.json
```

7. Data Requirements

7.1 User Entity

Field	Type	Required	Constraints
id	String (UUID)	Yes	Primary key, auto-generated
name	String	Yes	Max 100 characters
email	String	Yes	Unique, valid email format
password	String (hashed)	Yes	Min 6 characters, bcrypt hashed
phone	String	No	E.164 format
avatar	String (URL)	No	Profile image URL
loyaltyPoints	Integer	Yes	Default: 0, Min: 0
tier	Enum	Yes	Bronze | Silver | Gold | Platinum
role	Enum	Yes	customer | admin
createdAt	DateTime	Yes	ISO 8601 timestamp

7.2 Menu Item Entity

Field	Type	Required	Constraints
id	String (UUID)	Yes	Primary key
name	String	Yes	Max 100 characters
description	String	Yes	Max 500 characters
category	Enum	Yes	Starters | Mains | Desserts | Cocktails | Wine
price	Decimal	Yes	USD, 2 decimal places, Min: 0.01
image	String (URL)	Yes	Valid image URL
prepTime	Integer	Yes	Minutes, Min: 1
calories	Integer	No	kcal
rating	Decimal	Yes	1.0–5.0, 1 decimal place
reviewCount	Integer	Yes	Default: 0
isAvailable	Boolean	Yes	Default: true
tags	String[]	No	e.g. [Vegan, Gluten-Free, Chef's Pick]
allergens	String[]	No	e.g. [Nuts, Dairy, Gluten]

7.3 Order Entity

Field	Type	Required	Constraints

Field	Type	Required	Constraints
id	String (UUID)	Yes	Primary key
userId	String	Yes	Foreign key → User.id
items	Object[]	Yes	Array of {id, name, price, quantity}
type	Enum	Yes	dine-in | takeaway
tableNumber	String	No	Required if type = dine-in
status	Enum	Yes	placed | confirmed | preparing | ready | delivered | cancelled
subtotal	Decimal	Yes	Sum of item prices × quantities
discount	Decimal	Yes	Default: 0.00
tax	Decimal	Yes	10% of discounted subtotal
tip	Decimal	Yes	Default: 0.00
total	Decimal	Yes	subtotal - discount + tax + tip
couponCode	String	No	Applied coupon code if any
paymentMethod	String	Yes	card | apple_pay | google_pay | cash
createdAt	DateTime	Yes	ISO 8601 timestamp

7.4 Reservation Entity

Field	Type	Required	Constraints
id	String (UUID)	Yes	Primary key
userId	String	Yes	Foreign key → User.id
tableId	String	Yes	Foreign key → Table.id
tableNumber	Integer	Yes	Denormalized for display
date	String	Yes	Format: YYYY-MM-DD
time	String	Yes	Format: HH:MM (24h)
guests	Integer	Yes	Min: 1, Max: 8
status	Enum	Yes	pending | confirmed | cancelled
specialRequests	String	No	Max 500 characters
confirmationCode	String	Yes	Auto-generated alphanumeric code
createdAt	DateTime	Yes	ISO 8601 timestamp

7.5 Table Entity

Field	Type	Required	Constraints
id	String (UUID)	Yes	Primary key

Field	Type	Required	Constraints
number	Integer	Yes	Unique table number (1–10+)
capacity	Integer	Yes	Max seating: 2, 4, 6, or 8
location	String	Yes	e.g. Window, Terrace, Private, Center
shape	Enum	Yes	round | rectangle
status	Enum	Yes	available | occupied | reserved

8. Appendix

8.1 Screen Inventory

Module	Screen Name	Route Name	Auth Required
Auth	Splash Screen	Splash	No
Auth	Onboarding	Onboarding	No
Auth	Login	Login	No
Auth	Register	Register	No
Auth	Forgot Password	ForgotPassword	No
Customer	Home	Home	Yes
Customer	Menu	Menu	Yes
Customer	Dish Detail	DishDetail	Yes
Customer	Cart	Cart	Yes
Customer	Checkout	Checkout	Yes
Customer	Payment	Payment	Yes
Customer	Order Success	OrderSuccess	Yes
Customer	Reservation Wizard	Reservation	Yes
Customer	Reservation Confirm	ReservationConfirm	Yes
Customer	Order Tracking	OrderTracking	Yes
Customer	Order History	Orders (tab)	Yes
Customer	Reviews	Reviews	Yes
Customer	Profile	Profile (tab)	Yes
Customer	Settings	Settings	Yes
Customer	Loyalty Program	Loyalty	Yes
Admin	Admin Dashboard	AdminDashboard	Admin
Admin	Menu Manager	AdminMenu	Admin
Admin	Orders Panel	AdminOrders	Admin
Admin	Reservations	AdminReservations	Admin
Admin	Table Manager	AdminTables	Admin

8.2 Known Issues & Technical Debt

Issue	Status	Resolution
@expo/vector-icons incompatible with Expo SDK	Resolved	Replaced with custom emoji Icon component (components/Icon.tsx)

Issue	Status	Resolution
51		
Axios v1.7+ uses Node crypto module	Resolved	Replaced with native fetch() + AbortController timeout
Hardcoded BASE_URL for Android emulator	Known	User must manually update to device IP in api.ts for physical device
Mock data stored in-memory (resets on server restart)	Known	Production requires persistent database (PostgreSQL/MongoDB)
No real payment processing	Planned	Stripe API integration required for production
Push notifications not implemented	Planned	Requires Expo Notifications + Firebase FCM setup
Admin auth not enforced client-side	Planned	Admin button visible to all users; server-side role check required
Lottie animations placeholders only	Planned	Replace ActivityIndicator with Lottie JSON animations
No image picker for reviews/profile	Planned	Requires expo-image-picker integration
i18n translations not implemented	Planned	Language selector UI exists; i18next strings not loaded

8.3 Future Enhancements (Post-MVP)

- Real-time order status updates via WebSocket (Socket.io) instead of manual refresh
- Google Maps integration showing restaurant location and directions
- AI-powered dish recommendations based on order history
- Split-bill functionality for group dining
- Multi-language support using i18next with full translation strings
- Accessibility improvements: VoiceOver/TalkBack support, high-contrast mode
- Waitlist system for fully booked time slots
- Staff mobile app for kitchen display and table management
- Analytics dashboard with Mixpanel or Amplitude integration
- Customer re-engagement push notifications for abandoned carts

8.4 Revision History

Version	Date	Author	Description
0.1	Jan 2026	Dev Team	Initial draft — project scope defined
0.5	Jan 2026	Dev Team	Added functional requirements for auth, menu, cart, reservations
0.8	Feb 2026	Dev Team	Added admin panel requirements, API reference, data models
1.0	Feb 2026	Dev Team	Final draft — added NFRs, architecture diagram,

Version	Date	Author	Description
			known issues

8.5 Glossary

Term	Definition
Dine-In	Customer visits the restaurant and eats on-premises at a table
Takeaway	Customer collects their order from the restaurant for consumption elsewhere
QR Code	Machine-readable barcode displayed on order success and reservation confirmation screens for staff verification
Loyalty Tier	Membership level (Bronze, Silver, Gold, Platinum) determined by cumulative points earned
Coupon Code	Alphanumeric promotional code entered at checkout for a discount on the order total
Demo Mode	Offline fallback login using pre-seeded credentials when the backend server is unreachable
AbortController	Browser/React Native API used to cancel fetch requests that exceed the 8-second timeout
JWT	JSON Web Token — digitally signed string containing user identity claims for stateless authentication
Zustand Store	Single in-memory state container managing authentication, cart contents, favorites, and UI preferences
EAS	Expo Application Services — cloud build and submission service for iOS and Android app stores

— End of Document —

SmartDine SRS v1.0.0 • February 2026 • Confidential