# Part A: Scope Of Work

## 1. Introduction

SKN Cabs is a growing transportation service provider focused on improving accessibility and convenience within its region. Over the past two years, the company has successfully launched and operated a taxi booking platform, enabling customers to connect with drivers through a digital interface instead of traditional offline methods.

The current platform has played an important role in establishing initial market presence and user adoption. However, it is built on a white-label solution hosted on a shared server, with no ownership of the source code. This has resulted in several operational and technical limitations, including dependency on the existing vendor, limited customization capabilities, and increasing performance challenges as usage grows.

As the business continues to expand—particularly in core services such as ride-hailing and tour operations—these limitations have begun to directly impact system performance, reliability, and scalability. This becomes especially critical during peak business periods, where system stability and responsiveness are essential to maintaining service quality and customer satisfaction.

To address these challenges, SKN Cabs is planning to transition to a modern, scalable, and fully controlled platform. The objective is to build a high-performance system that ensures reliability, reduces operational dependencies, and provides complete ownership of the technology stack.

Based on the recent discussions and business priorities, the implementation will follow a phased rollout approach. The initial phase will focus on delivering and deploying the core ride-hailing and tour booking functionalities within an accelerated timeline, enabling the platform to go live ahead of the upcoming peak season. Subsequent phases will introduce additional modules such as delivery services, advanced pricing configurations, and further enhancements without impacting ongoing operations.

The proposed solution focuses on developing a custom-built platform that aligns with current operational requirements while remaining flexible for future growth. It is designed to deliver improved performance, optimized cost structure, enhanced user experience, and seamless scalability, while ensuring long-term control and independence for SKN Cabs.

## 2. Objective

The primary objective of this project is to design and develop a new platform for SKN Cabs that replaces the existing white-label system and resolves current performance, cost, and operational limitations, while enabling a faster and more reliable rollout aligned with business timelines.

# Part A: Scope Of Work

**The specific objectives of the project are as follows:**

- To develop a fully owned and scalable taxi booking platform, eliminating dependency on the existing vendor and shared infrastructure.
- To adopt a phased development and deployment approach, ensuring that core ride-hailing and tour functionalities are delivered and launched ahead of the peak business season, with additional modules rolled out in subsequent phases.
- To improve system performance by reducing latency, optimizing architecture, and ensuring stable operation across devices, including low-end Android devices and low-bandwidth environments.
- To deploy the platform on a dedicated cloud infrastructure to ensure better reliability, scalability, and performance compared to the current shared server setup.
- To optimize operational costs, particularly related to map APIs, through controlled usage, hybrid map integration, and efficient tracking strategies.
- To develop a robust system supporting core ride operations, including booking, driver connectivity, and real-time ride management.
- To implement a driver application that supports multi-service operations and multi-vehicle management, with controlled and efficient request handling.
- To ensure a reliable driver assignment mechanism based on a real-time first-accept model, eliminating duplicate allocations and improving response time.
- To integrate the Jad Cash payment gateway with flexible payment handling, including authorization and capture logic to improve user experience and prevent incorrect charging scenarios.
- To prioritize tour booking functionality in the initial phase, aligned with current business operations, and enable delivery services and additional features in later phases.
- To ensure the system performs effectively in low-bandwidth environments (including 3G conditions) for real-world usability.
- To provide comprehensive admin and dispatch panels for managing rides, drivers, pricing, and operational workflows.
- To enable smooth data migration from the existing system, including users and drivers, with minimal disruption and secure credential handling.
- To ensure full ownership, transparency, and control over the platform, including source code, server, and infrastructure.

# Part A: Scope Of Work

- To provide structured ongoing support and maintenance, ensuring system stability, compliance, and continuous improvement post-deployment.

## 3. Key Goals

### 3.1 Transition to a Fully Controlled Platform

To transition from the existing white-label system to a fully custom-built platform that provides complete ownership, flexibility, and control over system functionality, data, infrastructure, and future enhancements, eliminating dependency on third-party vendors.

### 3.2 Enable Phased Launch Strategy (Business-Critical)

To implement a structured phased rollout approach aligned with the client's operational priorities and peak season timeline:

- Phase 1 (Priority Launch): Core ride-hailing system and tour booking module delivered and launched ahead of the peak season to ensure business continuity.
- Phase 2: Delivery services and additional enhancements such as advanced pricing features, optimizations, and extended modules.

This approach ensures early go-live of critical services while allowing continuous improvement without operational disruption.

### 3.3 Improve System Performance and Reliability

To resolve existing issues related to latency, instability, and scalability by implementing a high-performance architecture optimized for real-world usage, including low-end devices and high-demand peak conditions.

### 3.4 Eliminate Shared Infrastructure Dependency

To migrate from shared hosting to a dedicated cloud infrastructure, ensuring improved stability, performance consistency, scalability, and full operational independence.

### 3.5 Optimize Operational Costs

To significantly reduce operational expenses by:

- Implementing hybrid map providers (Google Maps, Mapbox, OpenStreetMap, and region-specific alternatives such as Mappls/Ola Maps where applicable)
- Reducing dependency on continuous live tracking
- Using caching, static maps, and event-based location updates
- Minimizing unnecessary API calls for routing and geocoding

### 3.6 Implement Flexible and Configurable Pricing Models

**To build a fully configurable pricing engine supporting:**

- Local vs tourist pricing (logic-driven with optional admin override)
- Passenger-based pricing with automatic vehicle category filtering
- Luggage-based pricing
- Promotional and dynamic pricing rules with admin control

# Part A: Scope Of Work

### 3.7 Enable Multi-Service Platform Capability

To support expansion beyond ride-hailing into a scalable platform ecosystem:

- Phase 1: Ride-hailing + Tour booking (core business priority)
- Phase 2: Delivery services and additional service modules

### 3.8 Support Multi-Vehicle and Multi-Service Driver Operations

To allow drivers to operate across multiple service categories and vehicle types under a single account, with configurable controls and intelligent request distribution.

### 3.9 Ensure Reliable Driver Assignment System

To implement a real-time first-accept driver assignment model ensuring:

- One driver per ride allocation
- No duplicate assignments
- Faster response and improved dispatch efficiency

### 3.10 Optimize Tracking Strategy for Cost and Performance

To replace continuous GPS tracking with:

- Status-based tracking as the default (Searching, Assigned, Arrived, Trip Started, Completed)
- Optional live tracking during active trips only when required

This ensures a balance between cost optimization and operational visibility.

### 3.11 Implement Efficient Communication System

To align with client preference for lightweight communication:

- Direct calling as primary communication method
- In-app notifications for booking updates
- WhatsApp integration as optional channel
- SMS removed as default (used only if required as fallback)

### 3.12 Strengthen Operational Control via Admin & Dispatch Panels

To provide full operational visibility and control, including:

- Ride management and monitoring
- Driver assignment and tracking
- Pricing configuration
- Exception handling (no driver, stuck rides, cancellations)
- Reporting and operational insights

### 3.13 Enable Tour and Experience Management (Phase 1 Priority)

To include a dedicated tour module in Phase 1, allowing:

- Tour creation and management
- Pricing configuration
- Booking interface for customers
- Scheduling and dispatch assignment

### 3.14 Ensure System Usability in Low Network Conditions

To optimize application performance for 3G and low-bandwidth environments, ensuring smooth booking, tracking, and communication with minimal data usage.

# Part A: Scope Of Work

**3.15 Ensure Scalable and Future-Ready Architecture**

To design a modular and scalable architecture capable of handling increasing users, bookings, and feature expansion without performance degradation.

**3.16 Provide Full Ownership, Access & Transparency**

To ensure complete ownership of:

- Source code (mobile apps, backend, admin panel)
- Database
- Server and deployment infrastructure

No proprietary restrictions will be applied.

# Part B: Product Function

## Functional Requirements

### 1. Customer Application

#### 1. User Registration and Authentication

- The system shall allow users to register using email-based authentication or mobile number (configurable).
- The system shall allow users to log in using email/password authentication with OTP verification at each login for security.
- The system shall allow users to manage their profile information.

#### 1.2 Ride Booking Management

- The system shall allow users to book rides by selecting pickup and drop locations.
- The system shall display accurate ETA and estimated pricing before booking confirmation.
- The system shall display ride status updates including:
- Searching
- Assigned
- Arrived
- Trip Started
- Completed
- The system shall allow users to cancel a booking before driver assignment or as per cancellation policy.

#### 1.3 Pricing Display

- The system shall support logic-based pricing for:
- Local vs Tourist pricing
- Passenger-based pricing
- Luggage-based pricing
- The system shall automatically filter available vehicle options based on passenger capacity and booking requirements.

#### 1.4 Tour and Experience Booking (Phase 1 Priority)

- The system shall allow users to browse available tours and experiences.
- The system shall display tour details including duration, stops, and pricing.
- The system shall allow users to book tours by selecting date, number of persons, and pickup location.

#### 1.5 Delivery Booking (Phase 2)

- The system shall allow users to create delivery requests with pickup and drop details (to be introduced in Phase 2 rollout).

# Part B: Product Function

## Functional Requirements

### 1.6 Payment Management

- The system shall integrate the Jad Cash payment gateway.
- The system shall support flexible payment handling:
- Authorization at booking time
- Capture after ride acceptance or ride completion (configurable logic based on business rules)
- The system shall ensure users are not charged in case of non-accepted rides (as per agreed flow).

### 1.7 Communication & Notifications

- The system shall allow users to contact drivers via direct calling (primary method).
- The system shall provide in-app notifications for all booking updates (replacing SMS as default).
- WhatsApp integration shall be optional based on availability.

## 2. Driver Application

### 2.1 Driver Registration and Authentication

- The system shall allow drivers to register and log in using email and password.
- The system shall support OTP verification for secure login.

### 2.2 Ride Request Management

- The system shall allow drivers to receive ride requests across multiple service categories.
- The system shall allow drivers to accept ride requests.
- The system shall assign the ride to the first driver who accepts the request.
- The system shall automatically remove the request from other drivers once accepted.
- The system shall prevent new requests during an active trip to avoid overload.

### 2.3 Trip Management

The system shall allow drivers to update ride status:

- Arrived
- Trip Started
- Completed
- The system shall allow drivers to view assigned bookings and trip details.
- The system shall allow drivers to contact customers using in-app details (call-enabled).

### 2.4 Multi-Service Support

- The system shall allow drivers to operate across multiple services including:
  - Taxi (Phase 1)
  - Tours (Phase 1)
  - Delivery (Phase 2)

The system shall support multi-vehicle configuration under a single driver account.

# Part B: Product Function

## Functional Requirements

### 2.5 Multi-Vehicle Support

- The system shall allow drivers to manage and operate multiple vehicle types under a single account, enabling them to switch between available vehicles based on service requirements and availability.
- The system shall ensure that vehicle selection is validated against the booking requirements (such as passenger capacity and service type) before assignment.

### 2.6 Navigation

- The system shall allow drivers to use external navigation apps such as Google Maps or Waze.

## 3. Dispatch Panel

### 3.1 Ride Monitoring

- The system shall display ride statuses: New, Assigned, In Progress, Completed, Cancelled.

### 3.2 Driver Assignment

- The system shall allow manual driver assignment and reassignment.
- The system shall support cancellation of rides by dispatch.

### 3.3 Driver Monitoring

- The system shall display driver availability status: Online, Offline, Busy, Available.

### 3.4 Service Management

- **The system shall allow dispatchers to manage:**
  - Ride bookings
  - Delivery requests (Phase 2)
  - Tour bookings (Phase 1)

### 3.5 Operational Data Export

- The system shall allow dispatch/admin users to export ride data in CSV/Excel format, including:
  - Booking ID
  - Ride details
  - Fare
  - Driver assigned
  - Date & time
  - Tips (if applicable)
  - Ride status

# Part B: Product Function

## Functional Requirements

### 4 Admin Panel

#### 4.1 Dashboard
- The system shall provide a dashboard displaying revenue, active drivers, and failed rides.

#### 4.2 User Management
- The system shall allow management of customers, drivers, and dispatchers.

#### 4.3 Pricing Configuration
- The system shall allow configuration of:
- Local vs Tourist pricing
- Passenger-based pricing
- Luggage-based pricing
- Pricing rules shall be dynamically editable without app updates.

#### 4.4 Tour Management
- The system shall allow creation and management of tours.
- The system shall allow configuration of tour pricing.
- The system shall allow uploading of tour descriptions and images.

#### 4.5 Reports and Logs
**The system shall provide:**
- Trip reports
- Revenue reports
- Driver performance reports
- System logs (errors, OTP logs, API usage)

### 5. System-Level Functional Requirements

#### 5.1 Cost Optimization (Maps & Tracking)
- The system shall reduce dependency on continuous live tracking.
- The system shall use status-based tracking as default.
- The system shall minimize API usage through caching and hybrid map providers.

#### 5.2 Communication System
- The system shall prioritize:
- Direct calling
- In-app notifications
- SMS shall not be primary communication channel.

#### 5.3 App Migration / Upgrade Requirement
- The system shall support upgrade without requiring users to reinstall the app.
- Deployment shall be supported via existing App Store and Play Store accounts.

# Part B: Product Function

## Functional Requirements

### 5.4 Low Bandwidth Optimization

- The system shall function efficiently in 3G and low-network environments.

### 5.5 Dedicated Infrastructure

- The system shall be deployed on a dedicated cloud server (non-shared hosting).

### 5.6 Cloud Hosting Flexibility (Location-Based)

- The system shall support cloud hosting with selectable server location based on operational region.

### 5.7 API Cost Control (Explicit)

- The system shall minimize third-party API usage to control operational costs, especially map-related APIs.

### 5.8 External Payment Gateway Integration (Custom API)

- The system shall support integration with custom or region-specific payment gateways such as Jad Cash via provided APIs.

### 5.9 Maintenance & Support Continuity

- The system shall support ongoing maintenance and technical support for long-term operations.

### 5.10 App Store Compliance Handling

- The system shall support updates required for compliance with App Store and Play Store policies.