Software Requirement Specification Document

Online Ride Booking System

Team Members :

Maharsh Nayak (IT083)

Mann Patel (IT099)

# Table of Contents

[Table of Contents](#_jruovzusxjri)

[1. Introduction](#_f5yfxosin3nl)

[1.1. Purpose](#_hiui1tc2o1zx)

[1.2. Scope Of The Project](#_cx2rzf526ps2)

[1.3. Overview Of Document](#_j5hntoscikkh)

[2. Functional Requirements](#_4bk09wfhgms8)

[2.1. User Registration and Management](#_l38pj5i36iu2)

[■ Register User](#_wm967yndm6ix)

[■ User Login](#_w432nbr319qx)

[2.2. Update Profile](#_xmpczdwngryn)

[2.3. Password Reset](#_94ubcqk5cqhx)

[2.4. Ride History](#_9jfighzd9zss)

[2.5. Driver Management](#_odzomtvbce95)

[■ Driver Registration](#_nvtrysssaai7)

[■ Driver Login](#_9csyohuxt7o1)

[■ Update Driver Profile](#_mucxvn32m5on)

[■ Availability Status](#_zbet4v3gxs0i)

[■ Accept or Reject Ride Request](#_h3y3jwi9oc3j)

[■ View Ride History](#_4tyce2bzaall)

[■ Earnings & Payout Summary](#_gc6l7lix5xth)

[2.6. Booking System](#_hsij8q5btzhz)

[■ Selecting Ride Types from :](#_1z5a5hv2x4h3)

[■ Booking process according to ride type](#_sbw4ljgizbld)

[2.7. Live location tracking](#_w164y5gf67vc)

[■ Map Implementation](#_8sl4q2j1ymxn)

[2.8. Ride Cancel](#_86ygq3bny1fy)

[■ Cancel Ride(User side)](#_b6pghu6ueslc)

[■ Cancel Ride (Driver Side)](#_lzj099jy6ehn)

[■ View Cancellation History](#_vs3ibn8qdec)

[2.9. Feedback](#_s4ss46m56204)

[■ Submit Feedback(User side)](#_koph4mr24819)

[■ Submit Feedback(Driverside)](#_rzgle2dizs9)

[■ View Feedback](#_5d8lqnnnxhum)

[2.10. Contact details](#_dhpkupjkls8x)

[■ User Side](#_p8l0owq6pg7c)

[■ Driver Side](#_prhrb28w3ky5)

[2.11. Payment System](#_twa89ks1txlo)

[■ Payment Details](#_2e2uyapxeeiz)

[■ Ride Canceled by user](#_nt90e62r6pl8)

[■ Ride canceled by driver](#_xihsrx9uw5n9)

[■ Payment methods](#_6vd15nz0evji)

[■ Driver Payment for system](#_ryr50wbrb8sw)

[3. Non Functional Requirements](#_m9bfwprrsy57)

[3.1. Security](#_hiogzxxem42q)

[3.2. Performance Requirements](#_30bgwc4hykmi)

[3.3. Scalability](#_ifv6u5b2x00u)

[3.4. Reliability & Availability Requirements](#_8r00erxychle)

[3.5. Maintainability](#_8s1d88iwhl1e)

[3.6. Usability](#_ywls0upqvmty)

[3.7. Portability](#_pcjma51lhvuc)

[3.8. Data Integrity & Consistency](#_te2tp9p3ljra)

# Introduction

## Purpose

* + - The purpose of this Software Requirements Specification (SRS) is to define the functional and non-functional requirements of an Online Ride Booking System, similar to services like Ola or Uber. The system will provide a platform for users to book rides conveniently through a mobile or web application, while enabling drivers to receive ride requests and navigate efficiently.
    - This document is intended for:
      * Developers and designers working on the implementation,
      * Project evaluators and instructors,
      * Testers validating system behavior,
      * Stakeholders reviewing system expectations and features.

## Scope Of The Project

* + - The Online Ride Booking System will serve as a digital platform that facilitates real-time ride booking services between customers (riders) and drivers. Key functionalities include:
      * User registration and login
      * Driver registration and onboarding
      * Searching and booking available rides
      * Real-time tracking of driver location
      * Fare calculation and multiple payment options (cash, UPI, wallet)
      * Ratings and feedback system for rides
      * Admin dashboard to manage users, drivers, payments, and disputes
      * The system will support three roles:
      * User (Rider): Can request, book, track rides and make payments.
      * Driver: Can accept ride requests, navigate to pickup/drop points, and receive earnings.
      * Admin: Can manage the overall platform, user data, and payment reconciliation.

## Overview Of Document

* + - The Online Ride Booking System is a digital platform designed to simplify and automate the process of booking rides for commuters and allocating those rides to nearby available drivers. It provides a seamless experience for users to book cabs via a mobile or web application, track the live location of the driver, estimate fare costs, choose payment modes, and provide feedback after the ride.
    - From the driver's perspective, the system helps receive ride requests, navigate to pickup and drop locations, manage earnings, and review trip history. An admin panel is also integrated to monitor the entire platform, manage users and drivers, resolve disputes, and analyze platform usage.
    - The system aims to replicate real-world ride-hailing services like Ola or Uber, with core features such as:
      * Real-time GPS tracking of rides
      * Automated fare estimation and payments
      * Live driver-user communication
      * Ratings and review mechanisms
      * A secure and scalable backend
    - The project is modular and extensible, and can later be upgraded to support:
      * Ride-sharing (carpooling)
      * Outstation rentals
      * Emergency safety features
      * Loyalty programs and subscriptions

# Functional Requirements

## User Registration and Management

### Register User

* Description : Users should be able to register themselves using one of these methods and also provide an initial address.
* Input :
* Using phone number
* Using Email
* Using Google 3rd party services
* Initial Address (with label, e.g., Home/Work)
* Output
  + New User Account
  + Confirmation via SMS/Email

##### Valid Information

* Redirect To booking page

##### Invalid Information

* Ask to register again

### User Login

* Description : Users should be able to login themselves using one of these methods and also provide an initial address.
* Input :
  + Using phone number
  + Using Email
  + Using Google 3rd party services
  + Initial Address (with label, e.g., Home/Work)

##### Correct credentials

* Display booking page with user details

##### Incorrect Credentials

* Ask user to enter correct credentials

## Update Profile

* Description : User should be able to update the profile
* Input :
  + Their phone number with a number that doesn’t already have any account associated with it.
  + Their email with another email which doesn’t have any account associated with it.
  + Add another address and give it a label if they want to.
* Output :
  + Updated User Account

### Password Reset

* Description : System should allow password reset using OTP, which will be sent on either phone number and if user has provided with an email then user can select whether to receive OTP via SMS or email

### Ride History

* Description : The user should be able to see the details of all the rides that have been booked from that account in the past. Information to be shown include :
* Input :
  + Select the ride from list of Previous rides
* Output :
  + Amount of the payment made
  + Pick-up and Destination location
  + Duration of the ride
  + Driver name and vehicle type
  + Date and Time of the ride.

## Driver Management

### Driver Registration

* Description:

Drivers should be able to register themselves on the platform by providing personal, vehicle, and verification details.

* Input:
  + Full Name
  + Phone Number
  + Email ID (optional)
  + Government-issued ID (e.g., Aadhaar, PAN)
  + Driver License (with expiry date)
  + Vehicle Registration Details
  + Vehicle Type (Sedan, SUV, Bike, etc.)
  + Bank Account Details for payouts
  + Profile Photo & Vehicle Photo
* Process:
  + Submit form with required documents
  + Documents go through manual/admin or automated verification
* Output:
  + New Driver Profile Created

#### Correct Documents Provided

* Create new Driver account

#### No/Incorrect Documents Provided

* Ask driver to submit correct documents for re-evaluation

### Driver Login

* Description:

Registered and verified drivers should be able to log in using their phone number or email.

* + - * Input:
        + Phone Number or Email
        + OTP or Password
      * Output:
        + Session token
        + Login successful
        + Access to driver dashboard

#### Correct Credentials

* Bring a driver account and display all information, also bring to the process of accepting the users.

#### Incorrect CredentialsI

* Ask driver to input correct credentials

### Update Driver Profile

* + - * Description**:** Drivers should be able to update personal or vehicle details and upload new documents when necessary.
      * Input:
        + New phone number or email
        + New vehicle or license info
        + Updated address or bank details
        + Upload of new documents/photos
      * Output:
        + Updated Driver Profile
        + Some fields (like documents) may trigger re-verification

### Availability Status

* Description:

Drivers can toggle their availability to receive ride requests.

* Input:
  + Availability toggle in driver app interface
* Output:
  + Driver availability status updated in system
  + Only Online drivers are eligible for ride matching

### Accept or Reject Ride Request

* Description:

Drivers receive incoming ride requests and can accept or reject based on availability.

* Input
  + Ride request details (pickup location, fare estimate)
  + Accept/Reject action

#### If accepted

* Ride assigned and navigation enabled

#### If rejected

* Request passed to another nearby driver

### View Ride History

* Description:

Drivers should be able to view past rides they’ve completed along with earnings details.

* Input
  + Filter options: Date range, status (completed/cancelled), fare amount
* Output
  + List of rides with:

Pickup & drop-off locations

Fare amount

Ride duration

Passenger rating

Date and time

### Earnings & Payout Summary

* Description
  + Drivers can view their total earnings, bonuses, and pending payouts.
* Input
  + Date Range
* Output
  + Daily/weekly/monthly earnings
  + Commission deductions
  + Payout status (completed/pending)
  + Downloadable summary

## Booking System

### Selecting Ride Types from :

* Description :

The user should be able to select 1 of the 3 ride options provided.

* Input : selection of ride:
  + Inter city ride
  + Intra city ride
  + Rental
* Output : Further booking process according to the ride type selected.

### Booking process according to ride type

* Description : The user should be able to process the booking of the ride and book a ride.

#### Intra City Ride

* Input :
  + Selecting pick-up and drop off location
  + Selection of the vehicle type :
    - Rickshaw
    - Bike
    - Normal Car
    - SUV
    - Premium Car
* Output :
  + Show estimated cost
  + Connect to nearby drives available for ride
    - Select the 1 with highest rating
  + Confirming booking and sending OTP to users.

#### Inter City Ride

* Input :
  + Selection of pick up location
  + Selection of drop off city
  + Selection for time of the ride
  + Selection of vehicle type
    - Rickshaw
    - Bike
    - Normal Car
    - SUV
    - Premium Car
* Output :
  + Showing Estimated cost
  + Searching and connecting with a driver
  + Confirming the driver
  + Sending the user OTP

#### Rental

* Input :
  + Selection for Number of hours for which the rental is needed
  + Type of the vehicle
    - Rickshaw
    - Bike
    - Normal Car
    - SUV
    - Premium Car
* Output :
  + Showing Estimated price
  + Searching and connecting with a driver
  + Confirming the driver
  + Sending the user OTP

## Live location tracking

* Description : The user should be able to track the ride and live location on a map.

### Map Implementation

User Side

* Description : The user should be able to See the pick up, Destination, and the route taken for inter city and intra city ride.

Driver Side

* Description : When the driver accepts a ride they can see the user pick up location and once the ride starts they can see the destination and suggested route.

### tracking of ride

* Description : Users and driver should be able to see the live location of the path the driver is following and also see if it is in the path that leads to the destination.

## Ride Cancel

### Cancel Ride(User side)

* Description
  + Users should be able to cancel a ride that they have booked, either before the ride starts or during driver arrival.
* Input
  + Ride ID (selected from upcoming rides)
  + Reason for cancellation (optional selection or text input)
* Process
  + User initiates cancellation request
  + System checks ride status and policy rules
  + If within cancellation window, ride is cancelled and user is notified
  + If cancellation fee applies, it is shown to the user before confirmation
* Output
  + Ride status updated to "Cancelled"
  + Notification sent to driver
  + Refund or charge processed accordingly
  + Cancellation confirmation shown to user

#### Driver not yet arrived to the pick up location

* No bill to be payed
* Ask user for reason of cancellation

#### Driver has picked up user and is on the way

* User has to pay the bill of the distance travelled so far
* The user has to provide a reason for the cancellation.

### Cancel Ride (Driver Side)

* Description**:**
  + Drivers may cancel an accepted ride before pickup under valid circumstances (e.g., user unreachable, location not found).
* Preconditions**:**
  + Driver must have accepted the ride  
    - * + Ride not yet started

Input**:**

* + - * + Ride ID
        + Reason for cancellation (selected from predefined list or entered manually)
      * Process**:** 
        + Driver initiates cancellation
        + System verifies if cancellation is allowed
        + User is notified instantly
        + Penalty may apply based on driver behavior history or cancellation frequency
      * Output**:** 
        + Ride status updated to "Cancelled by Driver"
        + User notified with reason
        + Reassignment to nearby driver if applicable
        + Cancellation log stored

### View Cancellation History

* Description**:** Users and drivers should be able to view their past cancelled rides and associated details.
* Input**:**
  + Filter by date range or ride status (Cancelled)
* Output**:**
  + List of cancelled rides showing
    - * + Ride ID
        + Cancellation date and time
        + Who cancelled
        + Reason (if any)
        + Penalty applied (if any)

## Feedback

### Submit Feedback(User side)

* Description
  + After completing a ride, users should be able to rate the ride experience and provide feedback on the driver and vehicle.
* Input
  + Ride ID
  + Star Rating (1 to 5 stars)
  + Optional written comment
  + Optional tags (e.g., "Clean Car", "Rude Driver", "Late Arrival")
* Process
  + User is prompted to submit feedback post-ride
  + Rating and comment are saved and associated with driver profile
* Output
  + Feedback saved
  + Driver rating updated

### Submit Feedback(Driverside)

* Description
  + After completing a ride, drivers should be able to rate the passenger and provide feedback.
* PreCondition
  + Ride must be marked as "Completed"
* Input
  + Ride ID
  + Star Rating (1 to 5 stars)
  + Optional comment
* Process
  + Driver is prompted post-ride to give a rating
  + System stores the feedback associated with the user’s profile
* Output
  + Feedback stored
  + User rating updated
  + Pattern analysis for repeated complaints (optional)

### View Feedback

* Description
  + Admin should be able to view, filter, and act on feedback submitted by users and drivers.
* Input
  + Filters: Date range, ride ID, user ID, driver ID, rating threshold
* Output
  + Feedback displayed with status

### Contact details

#### User Side

* Description : The user should be able to contact the driver through the system, without sharing the number with the driver.
* Input
  + There should be a button that will make a call
* Output
  + The call should be made

#### Driver Side

* Description : The driver should be able to contact the user using system without sharing of number with the user
* Input
  + There should be a button that will make a call
* Output
  + The call should be made

## Payment System

* Description : The User should be able to pay the driver. The amount to pay is fixed in a range which is calculated by system.

### Payment Details

#### User Side

* The user should be given the cost estimation once the booking is done and the driver is on its way to pick up the user.

#### Driver Side

* The driver should be able to see the payment range once the destination is reached or the ride is ended by the user

### Ride Canceled by user

#### Driver reached the location

* Description : The user has to pay 30% of the total estimated value, again in the range provided, but exact value to be decided between user and driver.
* Input :
  + The user needs to give a reason for cancelation, it can be from the options provided or specify another reason in text.
* Output
  + The ride is canceled and the updated payment amount is displayed to both the user and driver.

#### Driver yet to reach the location

* The user doesn’t need to pay anything

#### Ride started

* The user has to pay the price according to the distance travelled, the amount is calculated by the system and given in a range.

### Ride canceled by driver

#### Ride not started

* The driver will get a negative rating if canceled 10 minutes after accepting the ride.

#### Ride started

* The driver cannot cancel the ride. However, in case of an emergency mutually agreed upon by the rider, the driver may cancel the ride.

### Payment methods

* Description: The system shall allow users to pay using cash, debit/credit card, UPI, wallets (e.g., Ola Money), or net banking.
* Inputs:
  + User's selected payment method
* Outputs:
  + Payment initiation and confirmation

### Driver Payment for system

#### Online Payment done by user

* Description : In case user pays the amount through the system then the commission for that ride is automatically deducted from that payment and then forwarded to the driver

#### Any other mode of payment

* Description : Until the driver pays the commission the driver doesn’t get the next ride order.

## Admin Management system

### Admin Registration & Login

* Description**:** Only authorized personnel should be able to access the admin dashboard through secure credentials.
* Input**:**
  + Username / Email
  + Password (encrypted in storage)
  + 2FA (OTP via Email/SMS) for login
* Output**:**
  + Successful login grants access to Admin Dashboard
  + Failed login shows appropriate error

### User Management (Admin Side)

* Description**:** Admins can view, update, block, or delete user accounts when required (e.g., fraud detection, abuse).
* Input**:**
  + User ID / Phone / Email search
  + Action: View / Edit / Suspend / Delete
* Output**:**
  + Updated user status reflected in system
  + Action logged for audit

### Driver Management (Admin Side)

* Description: Admins can approve, reject, or suspend driver accounts after verification, and monitor their performance.
* Input**:**
  + Driver ID / Name / Vehicle Number search
  + Action: Approve / Reject / Suspend / Edit Details
* Output**:**
  + Updated driver profile status
  + Document verification records stored

### Ride & Booking Oversight

* Description: Admins can view ongoing, completed, and cancelled rides for dispute resolution or platform monitoring.
* Input:
  + Ride ID / Date Range / Status filter
* Output:
  + Ride details displayed
  + Ability to cancel ride manually in emergencies

### Payment & Commission Management

* Description**:** Admins oversee payment settlements, refunds, and commission deductions.
* Input**:**
  + Payment ID / Driver ID / User ID
  + Action: Approve payout, Process refund, Adjust commission
* Output**:**
  + Updated payment records
  + Transaction logs stored

### Analytics & Reporting

* Description**:** Admins can generate platform usage reports, driver earnings summaries, and performance insights.
* Input**:**
  + Date Range
  + Filters: City, Ride Type, Driver/User stats
* Output**:**
  + Downloadable reports in CSV/PDF format

### Feedback & Dispute Handling

* Description**:** Admins can view feedback, investigate complaints, and take action against users or drivers.
* Input**:**
  + Feedback ID / Ride ID search
  + Action: Mark resolved, Suspend account, Add notes
* Output**:**
  + Updated feedback status
  + Actions logged

# Non Functional Requirements

## Security

* + - The system shall encrypt all sensitive user and driver data (e.g., passwords, payment info) using industry-standard encryption.
    - User and driver authentication shall be handled using secure login methods (e.g., OTP).
    - Data transmission between client and server must use HTTPS.

## Performance Requirements

* + 1. The system shall support a high number of users without significant degradation in performance.
    2. Ride request and matching operations should complete within 2 seconds under normal load.
    3. Login and registration response time must be minimal.
    4. The backend should handle 50 ride requests per second during peak hours.

## Scalability

* + - The system architecture shall be horizontally scalable, supporting addition of servers or microservices for high traffic regions.
    - Database and caching layers shall be designed to handle increased data volumes.
    - The platform should support operations in multiple cities.

## Reliability & Availability Requirements

* + - The system shall maintain 99.9% uptime annually to ensure continuous 24/7 service availability.
    - Failover mechanisms must be in place for critical components like ride matching and payment processing.

## Maintainability

* + - The system shall be modular and follow clean code principlesto ease debugging and future upgrades.
    - All major components shall be unit tested and integration tested.

## Usability

* + - The application interface should follow platform-specific UI-UX guidelines.
    - Users and drivers should be able to complete core tasks (book ride, accept ride) with minimum steps.
    - The UI/UX shall be designed for clarity and simplicity to support a wide range of users, including those with minimal technical experience.

## Portability

* + - The application shall be available on Android and iOS platforms, along with a web-based portal for users, drivers, and administrators.

## Data Integrity & Consistency

* + - All data should be consistent.
    - The system shall ensure atomicityin critical operations (e.g., ride booking, payments).
    - All financial transactions should follow ACID-compliantoperationsin the database.
    - The system should prevent double-booking of drivers or duplicate ride requests.