Software Requirements Specification (SRS)

Version 1.0

Red Bike Taxi

<u>Documented By Tridib Sarma</u>

Aspirant Software

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to provide a comprehensive Software Requirements Specification (SRS) for "**Red Bike Taxi**". This includes separate applications for users and riders (Android Based Mobile Application), and a multi-role admin panel with API integration. The user and rider applications will be developed in Kotlin/Java, and the API and admin panel will be developed using Laravel. The system will utilize a monolithic architecture with WebSocket for live tracking and two points of failure to ensure high availability and reliability.

1.2 Scope

" **Red Bike Taxi** " aims to provide a seamless and efficient ride-sharing experience. The scope includes the development of:

- A user app for passengers to book rides.

- A rider app for drivers to manage ride requests.

- An admin panel with multi-role access for managing users, rides, payments, and system settings.

- APIs for communication between the apps and the backend system.

- Live tracking of rides using WebSocket.

1.3 Definitions, Acronyms, and Abbreviations

- **Red Bike Taxi**: The name of the ride-sharing application.

- **SRS**: Software Requirements Specification.

- **API**: Application Programming Interface.

- **UI**: User Interface.

- **GPS**: Global Positioning System.

- **OTP**: One-Time Password.

- **GDPR**: General Data Protection Regulation.

- **WebSocket**: A protocol providing full-duplex communication channels over a single TCP connection.

1.4 References

- Android Developer Documentation

- Google Maps API Documentation

- Firebase Authentication Documentation

- Razorpay Payment Gateway Documentation

- Laravel Documentation

2. Overall Description

2.1 Product Perspective

"**Red Bike Taxi**" is a comprehensive ride-sharing solution with distinct applications for users and riders, and a robust admin panel. The system leverages external services like Google Maps for location tracking and Razorpay for payment processing. The system is built on a monolithic architecture, which includes WebSocket for real-time ride tracking with redundancy to ensure high availability.

2.2 Product Functions

The main functions of " **Red Bike Taxi** " include:

- User Registration and Authentication

- Profile Management

- Ride Booking and Acceptance

- Real-time Ride Tracking using WebSocket

- Payment Integration

- Notifications

- Review and Rating System

- Multi-Role Administration

2.3 User Classes and Characteristics

- **Passengers**: Users who book rides.

- **Drivers**: Users who offer ride-sharing services.
- **Administrators**: Users who manage the system and resolve disputes.
- **Super Administrators**: Users with full access to system settings and data.

2.4 Operating Environment

- Android 9.0 (Pie) and above for the mobile applications.
- Web browser compatibility for the admin panel.
- Internet connectivity (Wi-Fi or mobile data).
- GPS-enabled device for users and riders.

2.5 Design and Implementation Constraints

- Compliance with Android development standards for mobile apps.
- Use of Laravel framework for the backend and admin panel.
- Integration with third-party APIs (Google Maps, Razorpay, SMS Gateway).
- Data privacy and security regulations.
- Implementation of WebSocket for real-time communication.

2.6 Assumptions and Dependencies

- Users have GPS-enabled Android devices.
- Users have a valid mobile number for registration.
- The application will use Firebase for authentication and real-time database services.

3. System Features

3.1 User App Features

3.1.1 User Registration and Authentication

- **Description**: Users can register using their mobile number and receive an OTP for verification. They can log in using their registered credentials.

- **Functional Requirements**:

  - FR1: The system shall allow users to register using their mobile number.

  - FR2: The system shall send an OTP for verification.

  - FR3: The system shall authenticate users based on the OTP.

Use Case Diagram

```
          +-------------------+
          |   Register User   |
          +-------------------+
               /            \
  +-----------------+   +-------------+
  |   Send OTP      |   | Verify OTP |
  +-----------------+   +-------------+
```

3.1.2 Profile Management

- **Description**: Users can create and manage their profiles, including personal information and ride preferences.

- **Functional Requirements**:

  - FR4: The system shall allow users to update their profile information.

  - FR5: The system shall store and display user profiles.

Use Case Diagram

```
        +-----------------+
        |   Manage Profile|
        +-----------------+
            /           \
+---------------+   +--------------+
|   Update Info |   | Display Info |
+---------------+   +--------------+
```

3.1.3 Ride Booking

- **Description**: Users can book a ride by entering their destination.
- **Functional Requirements**:
  - FR6: The system shall allow users to book rides.
  - FR7: The system shall notify users of the ride status.
  - FR8: The system shall update the ride status in real-time.

Use Case Diagram

```
          +-------------+
          | Book Ride   |
          +-------------+
              /      \
+----------------+  +---------------+
| Notify Drivers |  | Update Status |
+----------------+  +---------------+
```

3.1.4 Ride Tracking

- **Description**: Users can track the real-time location of their rides.

- **Functional Requirements**:

  - FR9: The system shall provide real-time ride tracking using WebSocket.

  - FR10: The system shall display the estimated time of arrival.

<u>Use Case Diagram</u>

```
          +--------------+
          | Track Ride   |
          +--------------+
              /      \
+----------------+  +------------------+
| Provide GPS    |  | Display ETA      |
+----------------+  +------------------+
```

3.1.5 Payment System

- **Description**: Users can pay for their rides through integrated payment gateways.

- **Functional Requirements**:

  - FR11: The system shall integrate with Razorpay for payment processing. User Can also pay offline

  - FR12: The system shall generate and display payment receipts.

Use Case Diagram

```
            +------------+
            | Pay for Ride|
            +------------+
                 /    \
+--------------+        +-----------------+
| Payment Integrate |     | Generate Receipt |
+--------------+        +-----------------+
```

3.1.6 Notifications

- **Description**: Users receive notifications for ride status updates, payment confirmations, and promotional offers.

- **Functional Requirements**:

  - FR13: The system shall send push notifications for ride status updates.

  - FR14: The system shall notify users of successful payments.

Use Case Diagram

```
            +------------+
            | Notifications|
            +------------+
                 /    \
   +--------------+  +-----------------+
| Status Updates |  | Payment Confirmations |
   +--------------+  +-----------------+
```

3.1.7 Review and Rating System

- **Description**: Users can rate and review their ride experience.

- **Functional Requirements**:

  - FR15: The system shall allow users to rate drivers and rides.

  - FR16: The system shall display average ratings on user profiles.

Use Case Diagram

```
           +-------------+
           | Review and Rate|
           +-------------+
               /     \
+---------------+  +-----------------+
| Rate Drivers |  | Display Ratings |
+---------------+  +-----------------+
```

3.2 Rider App Features

3.2.1 Rider Registration and Authentication

- **Description**: Riders can register using their mobile number and receive an OTP for verification. They can log in using their registered credentials.

- **Functional Requirements**:

  - FR17: The system shall allow riders to register using their mobile number.

  - FR18: The system shall send an OTP for verification.

  - FR19: The system shall authenticate riders based on the OTP.

Use Case Diagram

```
      +-------------------+
      |   Register Rider  |
      +-------------------+
          /            \
+-----------------+   +-------------+
|   Send OTP      | | Verify OTP  |
+-----------------+   +-------------+
```

3.2.2 Profile Management

- **Description**: Riders can create and manage their profiles, including personal information and vehicle details.

- **Functional Requirements**:

  - FR20: The system shall allow riders to update their profile information.

  - FR21: The system shall store and display rider profiles.

Use Case Diagram

```
      +-----------------+
      |   Manage Profile|
      +-----------------+
          /          \
+---------------+   +--------------+
|   Update Info |   | Display Info |
+---------------+   +--------------+
```

3.2.3 Ride Management

- **Description**: Riders can manage ride requests, accept rides, and update ride status.

- **Functional Requirements**:

  - FR22: The system shall notify riders of new ride requests.

  - FR23: The system shall allow riders to accept or decline ride requests.

  - FR24: The system shall update the ride status in real-time.

<u>Use Case Diagram</u>

```
              +--------------+
              | Manage Ride  |
              +--------------+
                   /     \
   +---------------+  +-----------------+
   | Accept Ride   |  | Update Ride Status |
   +---------------+  +-----------------+
```

#### 3.2.4 Ride Tracking

- **Description**: Riders can track the real-time location of their rides and navigate to the destination.

- **Functional Requirements**:

  - FR25: The system shall provide real-time ride tracking using WebSocket.

  - FR26: The system shall provide navigation assistance.

<u>Use Case Diagram</u>

```
          +--------------+
          | Track Ride   |
          +--------------+
               /     \
+---------------+  +-----------------+
| Provide GPS    |  | Provide Navigation |
+---------------+  +-----------------+
```

3.2.5 Payment System

- **Description**: Riders can receive payments for their rides through integrated payment gateways.

- **Functional Requirements**:

  - FR27: The system shall integrate with Razorpay for payment processing.

  - FR28: The system shall generate and display payment receipts.

<u>Use Case Diagram</u>

```
          +-------------+
          | Receive Payment |
          +-------------+
               /     \
+--------------+  +-----------------+
| Payment Integrate |  | Generate Receipt |
+--------------+  +-----------------+
```

3.2.6 Notifications

- **Description**: Riders receive notifications for ride requests, ride status updates, and payment confirmations.

- **Functional Requirements**:

  - FR29: The system shall send push notifications for ride requests and status updates.

  - FR30: The system shall notify riders of successful payments.

Use Case Diagram

```
              +-------------+
              | Notifications|
              +-------------+
                 /      \
+---------------+  +------------------+
| Ride Requests |  | Payment Confirmations |
+---------------+  +------------------+
```

#### 3.2.7 Review and Rating System

- **Description**: Riders can rate and review their passengers.

- **Functional Requirements**:

  - FR31: The system shall allow riders to rate passengers.

  - FR32: The system shall display average ratings on passenger profiles.

Use Case Diagram

```
        +-------------+
       | Review and Rate|
        +-------------+
           /      \
+---------------+  +------------------+
| Rate Passengers |  | Display Ratings |
+---------------+  +------------------+
```

3.3 Admin Panel Features

3.3.1 User Management

- **Description**: Admins can manage user accounts, including passengers and riders.

- **Functional Requirements**:

  - FR33: The system shall allow admins to view and manage user accounts.

  - FR34: The system shall allow admins to deactivate or delete user accounts.

Use Case Diagram

```
        +-------------+
      | User Management|
        +-------------+
           /      \
+-----------------+  +------------------+
| View Users      |  | Manage Users     |
+-----------------+  +------------------+
```

3.3.2 Ride Management

- **Description**: Admins can manage ride requests, monitor ride status, and resolve disputes.

- **Functional Requirements**:

  - FR35: The system shall allow admins to view and manage ride requests.

  - FR36: The system shall provide tools for resolving ride disputes.

Use Case Diagram

```
              +-------------+
              | Ride Management|
              +-------------+
                    /     \
+-----------------+  +------------------+
| View Rides      |  | Manage Disputes  |
+-----------------+  +------------------+
```

3.3.3 Payment Management

- **Description**: Admins can monitor payments and handle financial transactions.

- **Functional Requirements**:

  - FR37: The system shall allow admins to view payment transactions.

  - FR38: The system shall generate financial reports.

Use Case Diagram

```
+-------------+
| Payment Management|
+-------------+
/      \
+-----------------+  +------------------+
| View Transactions|  | Generate Reports|
+-----------------+  +------------------+
```

3.3.4 Notifications Management

- **Description**: Admins can manage system notifications and promotional offers.

- **Functional Requirements**:

  - FR39: The system shall allow admins to create and manage notifications.

  - FR40: The system shall allow admins to send promotional offers.

Use Case Diagram

```
+-------------+
| Notifications Management|
+-------------+
/      \
+-----------------+  +------------------+
| Create Notifications|  | Send Offers|
+-----------------+  +------------------+
```

3.3.5 Review and Rating Management

- **Description**: Admins can monitor and manage reviews and ratings.
- **Functional Requirements**:
  - FR41: The system shall allow admins to view and manage user reviews.
  - FR42: The system shall allow admins to take action on inappropriate reviews.

<u>Use Case Diagram</u>

```
              +-------------+
             | Review Management|
              +-------------+
                  /      \
      +-----------------+  +------------------+
     | View Reviews     |  | Manage Reviews   |
      +-----------------+  +------------------+
```

3.3.6 Role-Based Access Control

- **Description**: The system supports multiple admin roles with varying access levels.
- **Functional Requirements**:
  - FR43: The system shall support role-based access control for admins.
  - FR44: The system shall allow super admins to define and manage roles and permissions.

Use Case Diagram

```
+-------------+
| Role-Based Access Control|
+-------------+
/      \
+-----------------+  +------------------+
| Define Roles    |  | Manage Permissions|
+-----------------+  +------------------+
```

4. External Interface Requirements

4.1 User Interfaces

- **Passenger Interface**: User-friendly interface for booking rides, tracking rides, and making payments.

- **Driver Interface**: Interface for managing ride requests, navigation, and payment receipt.

- **Admin Interface**: Dashboard for managing users, rides, and disputes.

4.2 Hardware Interfaces

- **Smartphones**: Android devices with GPS and internet connectivity.

- **Web Browsers**: For accessing the admin panel.

4.3 Software Interfaces

- **Google Maps API**: For location services and ride tracking.

- **Firebase Authentication**: For user authentication.

- **Razorpay API**: For payment processing.

- **SMS Gateway**: For OTP processing.

4.4 Communication Interfaces

- **Internet**: Required for all application functionalities.

5. System Requirements

5.1 Functional Requirements

(As detailed in section 3)

5.2 Non-Functional Requirements

- **Performance**: The system should handle up to 220 concurrent users.
- **Reliability**: The system should have 91.9% uptime.
- **Usability**: The application should have an intuitive and user-friendly interface.
- **Scalability**: The system should be able to scale to accommodate more users and features.
- **Security**: User data (password) should be encrypted and securely stored.

6. Other Requirements

6.1 Performance Requirements

- The application should respond to user inputs within 10 seconds.
- The application should update the ride status in real-time.

6.2 Safety Requirements

- The application should have emergency features for users to contact authorities in case of safety concerns.

6.3 Security Requirements

- User data must be encrypted in transit and at rest.

6.4 Software Quality Attributes

- **Maintainability**: The codebase should be well-documented and modular.
- **Portability**: The application should be compatible with various Android devices.
- **Interoperability**: The application should seamlessly integrate with third-party APIs.

7. API Specifications

7.1 API Endpoints

#### User Registration and Authentication

- **POST /api/register**: Register a new user.
- **POST /api/login**: User login.
- **POST /api/verify-otp**: Verify OTP for authentication.

#### Profile Management

- **GET /api/profile**: Retrieve user profile.
- **PUT /api/profile**: Update user profile.

#### Ride Management

- **POST /api/rides**: Book a ride.
- **GET /api/rides/{id}**: Retrieve ride details.
- **DELETE /api/rides/{id}/cancel**: Ride Cancel.
- **PUT /api/rides/{id}/status**: Update ride status.

#### Payment Processing

- **POST /api/payments**: Process a payment.

- **GET /api/payments/{id}**: Retrieve payment details.

#### Notifications
- **GET /api/notifications**: Retrieve notifications.
- **POST /api/notifications**: Create a notification.

#### Reviews and Ratings
- **POST /api/reviews**: Submit a review.
- **GET /api/reviews/{id}**: Retrieve reviews for a user.

#### Admin Management
- **GET /api/admin/users**: Retrieve list of users.
- **PUT /api/admin/users/{id}**: Update user details.
- **DELETE /api/admin/users/{id}**: Delete a user.

7.2 Authentication and Authorization

- **JWT Tokens**: JSON Web Tokens will be used for securing API endpoints.
- **Role-Based Access**: Different roles will have different permissions for accessing API endpoints.

8. Appendix

8.1 Glossary

- **OTP**: One-Time Password
- **GDPR**: General Data Protection Regulation
- **JWT**: JSON Web Token

8.2 Analysis Models

- **UML Diagrams**:

  - **Use Case Diagrams**: Depicting the different use cases for users (Passengers, Drivers, Admins).

  - **Class Diagrams**: Showing the structure of the application.

  - **Sequence Diagrams**: Illustrating interactions between different components.

  - **Activity Diagrams**: Describing the flow of different activities within the system.

__Use Case Diagram for User App__

```
  +----------------------+
  |       User App       |
  +----------------------+
           /  |  \
+-----------+  +-----------+
|  Register |  |  Login    |
+-----------+  +-----------+
          |         |
+-----------+  +-----------+
  | Book Ride |   | Track/Cancel Ride|
+-----------+  +-----------+
          |         |
+-----------+  +-----------+
| Pay Ride  |  | Rate Ride |
+-----------+  +-----------+
```

Use Case Diagram for Rider App

```
+----------------------+
|      Rider App      |
+----------------------+
         /   |   \
+-----------+  +-----------+
|  Register |  |  Login    |
+-----------+  +-----------+
        |          |
+-----------+  +-----------+
| Accept Ride|  | Track Ride|
+-----------+  +-----------+
        |          |
+-----------+  +-----------+
| Complete/Cancel Ride | | Rate Passenger |
+-----------+  +-----------+
```

Use Case Diagram for Admin Panel

```
+----------------------+
|     Admin Panel      |
+----------------------+
/   |   \
+-----------+  +-----------+
|  Manage Users|  | Manage Rides|
+-----------+  +-----------+
|          |
+-----------+  +-----------+
| Manage Payments |  | Manage Notifications|
+-----------+  +-----------+
|          |
+-----------+  +-----------+
| Manage Reviews |  | Manage Roles |
+-----------+  +-----------+
```

Class Diagram

```
+------------------------+
|        User            |
+------------------------+
| -userId: String        |
| -name: String          |
| -email: String         |
| -password: String      |
| -mobile: String        |
| -role: String          |
+------------------------+
| +register()            |
| +login()               |
| +updateProfile()       |
+------------------------+

+------------------------+
|      Passenger         |
+------------------------+
| -rideHistory: List<Ride>|
+------------------------+
| +bookRide()            |
| +trackRide()           |
+------------------------+

+------------------------+
|        Driver          |
+------------------------+
| -vehicleDetails: String|
| -rideHistory: List<Ride>|
+------------------------+
| +acceptRide()          |
| +updateRideStatus()    |
+------------------------+
+------------------------+
 |       Admin          |
+------------------------+
| +manageUsers()         |
| +resolveDisputes()     |
+------------------------+


+------------------------+
  |       Ride          |
+------------------------+
 | -rideId: String       |
 | -source: String       |
| -destination: String   |
 | -status: String       |
 | -driver: Driver       |
| -passenger: Passenger  |
+------------------------+
 | +startRide()          |
 | +endRide()            |
| +calculateFare()       |
+------------------------+
```

Sequence Diagram for Ride Booking

```
Passenger      QuickRideApp     Server     Driver
   |               |             |          |
   |---bookRide()-->|             |          |
   |               |---requestRide()------->|
   |               |             |<--notifyDriver()--|
   |               |             |          |        |
   |               |<--rideStatusUpdate()-->|        |
   |<--rideStatusUpdate()--|                |        |
   |               |             |          |        |
   |               |             |---acceptRide()--->|
   |               |             |<--rideStatusUpdate()--|
   |               |<--rideStatusUpdate()--|          |
   |<--rideStatusUpdate()--|                |        |
   |               |             |          |        |
```

Activity Diagram for User Registration

```
        +------------------------+
           |  Start                |
        +------------------------+
                    |
                    v
        +------------------------+
      |  Enter Mobile Number   |
        +------------------------+
            |
            v
+------------------------+
| Send OTP               |
+------------------------+
            |
            v
+------------------------+
| Enter OTP              |
+------------------------+
            |
            v
+------------------------+
| Verify OTP             |
+------------------------+
            |
            v
+------------------------+
| Registration Success   |
+------------------------+
            |
            v
+------------------------+
| End                    |
+------------------------+
```

8.3 Issues List

- Any identified issues and their resolution status.

This SRS document provides a comprehensive overview of the requirements for the "**Red Bike Taxi**" Android application, ensuring that all stakeholders have a clear understanding of the system's capabilities and constraints. The detailed use case, class, sequence, and activity diagrams further clarify the system's design and functionality. The document also includes specifications for the APIs, ensuring smooth communication between the mobile apps and the backend system.