Software Requirements Specification Document: **UniRide** *a Ride-Sharing Application*

Development Team: Nour El Houda EL IAMANI & Aya BENJELLOUN

October 7, 2025

1 Introduction

This document is the Software Requirements Specification (SRS) of the UniRide application. UniRide is a campus-based ride-sharing and on-demand transport app that helps students, staff, and visitors move easily across and outside the university campus.

It allows users to request rides, offer rides, and coordinate transport conveniently. This document follows the feasibility report and presents the finalized functional and non-functional requirements obtained after requirement elicitation, analysis, specification, and validation.

2 Functional Requirements

2.1 Requirements of Riders Interface

- Riders can register via email with confirmation.
- Riders can view available ride offers in real time.
- Riders can filter rides by location (inside or outside campus), time, and number of available seats.
- Riders can book a ride by choosing a pickup point, destination, and number of passengers.
- For inside-campus rides, riders choose from predefined pickup and destination points.
- For outside-campus rides, riders can enter custom pickup and destination points.
- The fare is automatically calculated based on distance, number of passengers, and fare per kilometer.
- Riders can schedule rides in advance ("Schedule for Later").

- Riders receive notifications when the driver accepts the request, is on the way, or has arrived.
- Riders can cancel a ride before it starts.
- The system tracks ride cancellations and can suspend riders who cancel too many rides.
- Riders can contact drivers through WhatsApp.
- Riders can view driver details such as name, car model, and license plate.
- After each ride, riders can leave a short review of the driver.
- Riders can report a driver for misconduct.
- Riders can see the driver's location after accepting the ride.

2.2 Requirements of Drivers Interface

- Drivers can register after providing and confirming their email, license, insurance, and student/staff ID, and by providing car information such as license plate.
- Drivers can create and publish ride offers (pickup, destination, available seats, departure time).
- Drivers can view and accept ride requests.
- Drivers can contact riders through WhatsApp.
- Drivers can manage scheduled rides.
- Drivers' real-time location is shared with accepted riders during a ride.
- Drivers can view basic rider information after accepting a booking.
- Drivers can cancel a ride before it starts. After a certain number of cancellations (e.g., 5), the driver is automatically blocked from the app for one week.
- Drivers can view their trip and earnings history, as well as received reviews.
- Drivers can upload and manage verification information such as license, insurance, and student/staff ID.
- Drivers can update car details (model, license plate, capacity).
- Drivers can report a rider for misconduct.

2.3 Requirements of Admin Interface

- Admin can view, add, edit, or delete rider and driver accounts.
- Admin can verify driver documents (license, insurance, ID).
- Admin can review and handle user complaints or reports.
- Admin can access the full ride history, including past, ongoing, and scheduled rides.
- Admin can view and manage reviews and ratings (edit or remove inappropriate ones).
- Admin can suspend or ban users who violate app policies such as fraud, repeated cancellations, or misconduct.
- Admin can generate statistics and reports (total rides, active drivers, ratings, revenue, etc.).
- Admin can configure app settings such as fares, pickup points, and cancellation limits.

3 Non-Functional Requirements

3.1 Performance

- The app supports up to 500 users at the same time without slowing down.
- Ride offers and driver locations update with less than a 2-second delay.

3.2 Security

- All user passwords and personal data are encrypted.
- Communication between users and the server is protected by HTTPS.
- Only verified drivers can offer rides (verified by the admin).
- User data follows basic privacy and protection rules.

3.3 Reliability and Availability

- The app is available at least 99% of the time during campus hours.
- In case of a server failure, the system recovers within a few minutes.
- Ride data and user information are backed up daily.

3.4 Usability

- The interface is simple, intuitive, and mobile-friendly.
- New users can register and book a ride in less than 3 minutes.
- Notifications (ride accepted, driver arrived, etc.) are clear and easy to understand.

3.5 Portability

- The app works on both Android and iOS using a cross-platform framework (e.g., React Native).

3.6 Policy and Restrictions

- Riders who cancel rides frequently may be temporarily suspended from using the app.
- Drivers who cancel more than five rides are automatically blocked for one week.

4 Use Case Diagram

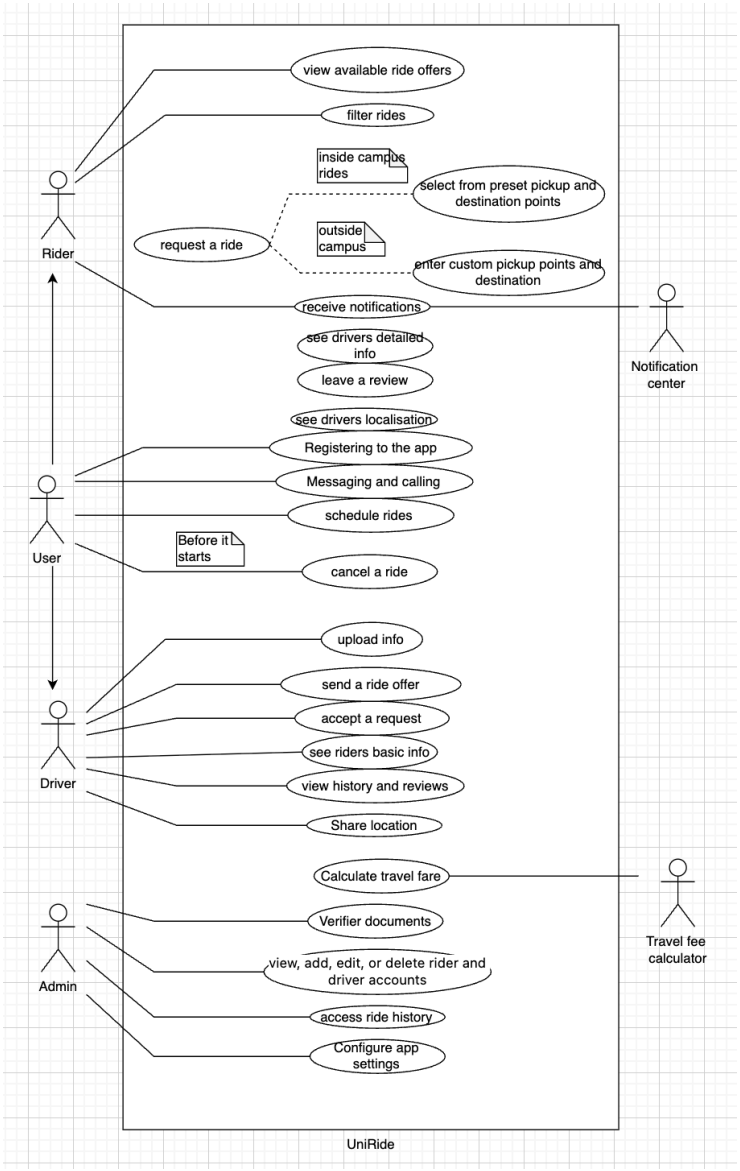

Figure 1: Use Case Diagram of UniRide

5 Use Case diagram explanation

When a user installs the UniRide application, they are first prompted to choose whether they want to use the app as a driver or a rider. If the user chooses to

be a rider, they register and log in using their email, which must be confirmed before accessing the system. Once logged in, the rider can select whether they want to travel inside campus or outside campus. Based on this choice, the system displays a list of available ride offers. The rider can either accept an existing offer or create a new ride request if they need transportation at a later time. They also have the option to schedule a ride in advance.

If the user chooses to be a driver, they must provide detailed information such as their driver's license, car details, insurance, and student or staff ID during registration. After submitting this information, the driver's account remains pending until the admin verifies and validates the provided documents. Once approved, the driver can log in, view ride requests from riders, create and publish ride offers, and schedule rides for future trips.

This use case diagram illustrates how both riders and drivers interact with the system.