

**PROJECT REPORT ON**

# ISE TASK-2

**Comprehensive System Design and Agile Implementation for a Ride-Hailing Application**

Submitted to

**NMAM Institute of Technology, Nitte**

(Off-Campus Centre, Nitte Deemed to be University, Nitte – 576110, Karnataka, India)

On partial fulfilment of the requirements for the award of the

Degree of Bachelor of Technology

In

**INFORMATION SCIENCE AND ENGINEERING**

**By**

**NAME: Daniel Leonard**

**USN: NNM23IS039**

**Under the guidance of**

**Dr. JASON ELROY MARTIS, Associate Professor,**

**Department of Information Science and Technology,**

**NMAM Institute of Technology, Nitte, Karnataka, India**

# RIDE HAILING APP

**Introduction to System Design for a Ride-Hailing App with Agile Practices**

Ride-hailing apps have transformed the transportation industry by offering timely, on-demand ride-booking services to users. These platforms connect passengers with drivers using mobile apps, ensuring an easy and hassle-free transportation experience. They also incorporate numerous technologies like GPS navigation, automatic payments, and real-time vehicle tracking. The objective of this report is to lay out a systematic development plan for a ride-hailing app by defining critical system models and Agile practices. The report contains a Context Model that shows the app's interaction with external systems including payment gateways, map services, and notification services. Besides, an Interaction Model is also included to document the series of actions involved in reserving a ride, whereas a Behavioral Model graphically portrays the different states of a ride, from requested to completed.

To facilitate effective development and deployment, a Scrum-based approach is suggested. Sprint frameworks, critical user stories for drivers and passengers, and an explicit Definition of Done (DoD) for ensuring quality and usability are delineated in the report. This systematic process if adopted will ensure that the ride-hailing application can be designed with scalability, security, and user experience at its core.

**Key Components of the System Design:**

1. User Management Module: Handles passenger and driver authentication, profile management, and security features such as identity verification.
2. Ride Matching System: Connects passengers with available drivers based on location, estimated arrival time, and ride preferences.
3. Navigation and Mapping: Utilizes a Maps API to provide real-time navigation, route optimization, and estimated time of arrival.
4. Payment Processing: Integrates with a payment gateway to support multiple payment options, including credit cards, digital wallets, and in-app transactions.
5. Ride Tracking and Notifications: Provides real-time updates on ride status through push notifications and in-app tracking.
6. Rating and Feedback System: Allows passengers and drivers to rate each other and submit reviews to maintain service quality.
7. Admin Dashboard: Offers tools for managing users, monitoring rides, handling disputes, and generating reports on app performance.
8. Security and Compliance: Implements data encryption, fraud detection, and regulatory compliance measures to ensure a safe and secure experience.

**Agile Implementation in the Design Process:**

* Sprint Planning – Decomposing system elements into tasks that can be managed.
* Iterative Development – Developing and improving features in incremental steps.
* Continuous Feedback – Feedback from stakeholders at every step.
* Scrum/Kanban Practices – Standups, backlog grooming, and sprint retrospectives.

By integrating system design best practices with Agile, we seek to create a robust, scalable, and user-focused ride-hailing platform that can evolve with market demands

**System Modelling**

**Context Model:**

This context model illustrates how the ride-hailing app system connects with different external systems and users. Below is a description of the external interactions:

**External Systems:**

**1. GPS/Location Services:** Enables real-time rider and driver location tracking

**2. Payment System:** Manages payment processing, transactions, and refunds

**3. Messaging System:** Facilitates messaging between drivers and riders

**4. Mapping Service:** Offers navigation, route calculation, and traffic alerts

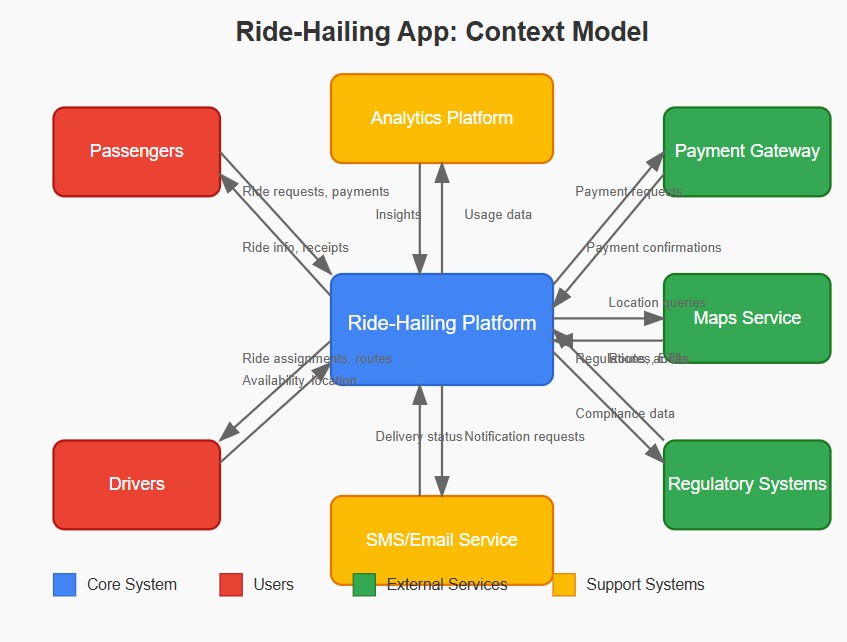
**5. Regulatory Systems:** Interconnects with government/regulatory authorities for compliance

**6. Analytics Service:** Logs and processes application use and business analytics

**7. Notification Service:** Sends push notifications, SMS, and emails

**External Users:**

1. Riders: End customers seeking rides  
2. Drivers: Service providers executing ride orders  
3. Customer Support: Personnel who deal with user complaints and queries  
The diagram illustrates the ride-hailing app as the central system and bidirectional flows of data to all external players, showing how the app is a platform for bringing together diverse stakeholders and services

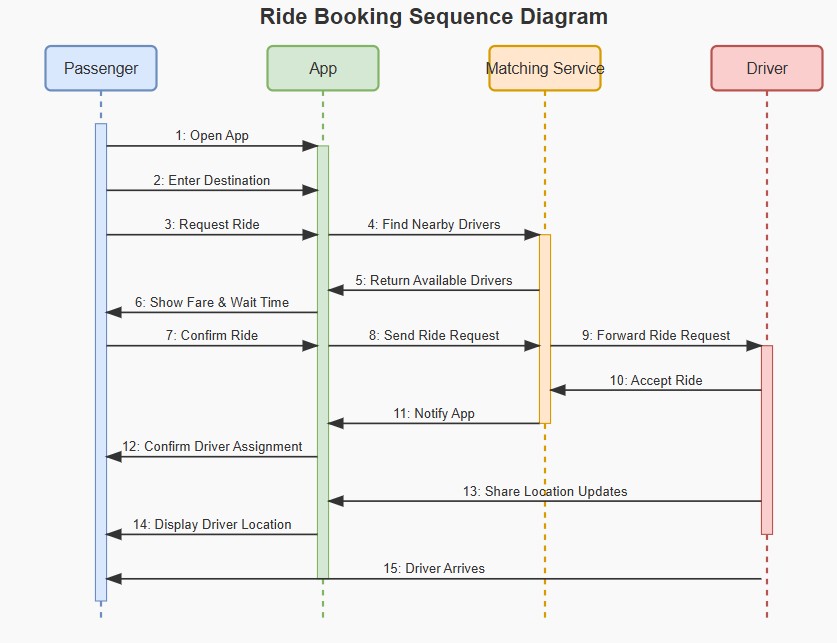


**Interaction Model**

This sequence diagram depicts the interaction flow while booking a ride within a ride-hailing app. The diagram represents the interaction between four participants of prime importance:

**The sequence of interactions:**   
1.Passenger opens the app  
2.Passenger inputs their destination  
3. Passenger makes a request for a ride  
4. App requests to communicate with the Matching Service for available drivers around them  
5. Matching Service responds with available drivers to the App  
6. App indicates passenger estimated wait time and fare  
7. Passenger accepts the request for the ride  
8. App notifies the Matching Service of the confirmed request for the ride  
9. Matching Service relays the ride request to an available Driver  
10. Driver accepts the request for the ride  
11. Matching Service informs the App of driver acceptance  
12. App confirms driver allocation to the Passenger  
13. Driver sends location updates to the App  
14. App shows the driver's location to the Passenger  
15. Driver reaches the passenger's pickup point.

The diagram uses color-coding (blue for Passenger, green for App, orange for Matching Service, and red for Driver) to distinguish between the different participants, and shows the chronological flow of messages between them using directional arrows.



**Behavioural Model**

This state diagram covers the entire life of a ride in a ride-sharing app from request initiation to eventual completion. The diagram employs variously colored states to depict how a ride progresses through different stages.

**Main Flow Path:**

1. **Start** → Initial state
2. **Ride Requested** → User initiates a ride request with pickup location and destination
3. **Searching for Driver** → System looks for available drivers nearby
4. **Driver Assigned** → A driver accepts the ride request
5. **Driver En Route** → Driver is traveling to the pickup location
6. **Arrived at Pickup** → Driver reaches the pickup location
7. **Waiting for Passenger** → Driver is waiting for the passenger to board
8. **Ride in Progress** → Passenger is in the vehicle and journey is underway
9. **Ride Completed** → Vehicle arrives at the destination
10. **Payment Processing** → System processes the fare payment
11. **Rating & Feedback** → User rates the driver and provides feedback
12. **End** → Final state

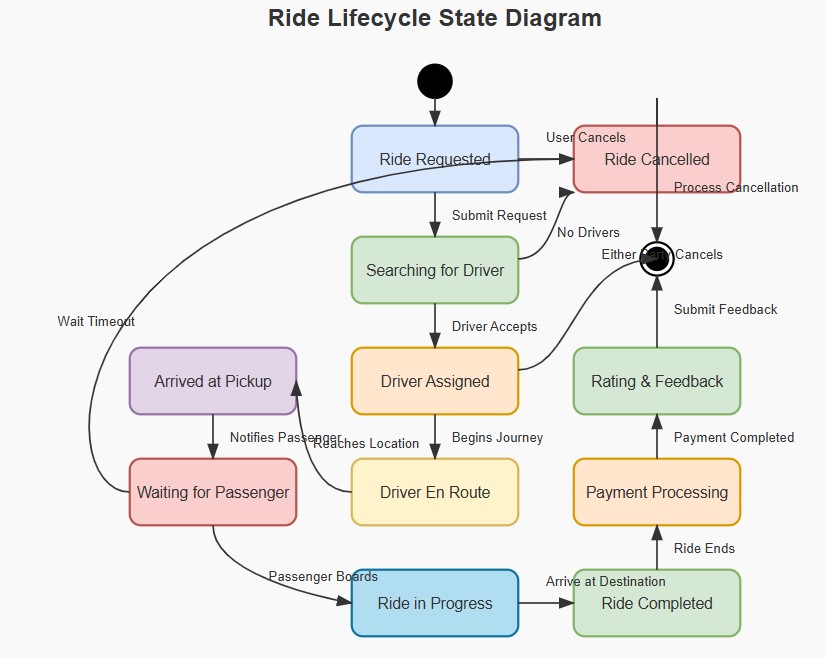
**Alternative Paths:**

● **Cancellation Paths**:

 Passenger can cancel from the "Ride Requested" state  
○ System can cancel because of "No Drivers" from the "Searching for Driver" state  
○ Either user can cancel once a driver has been assigned  
○ Timeout can happen if the passenger fails to show up after the driver has waited.

Every state transition is annotated with the event that causes the transition,

 i.e., "Submit Request," "Driver Accepts," "Passenger Boards," etc.  
The chart adheres to the draw.io visual standard using rounded rectangles for states, transitional arrows, and a neat, well-structured layout that clearly indicates the flow and potential branches in the ride lifecycle.



**Ride-Hailing App: Agile Development Plan**

**Principal Agile Implementation Elements**

**Scrum Team Organization**

**1. Product Owner**: Product backlog, prioritizing, and communications with stakeholders

2. Scrum Master: Coordinates Scrum events, removes obstacles, and guides the team

3. Development Team: Cross-functional development team of 5-7 members with knowledge in mobile, backend systems, UX/UI, and QA

Sprint Architecture with Principal Deliverables

Sprint 0: Project Setup & Foundation (2 weeks) Objective: Set up project foundation and infrastructure Key Deliverables:

● Project repository setup with branching strategy

● CI/CD pipeline setup

● Technical architecture documentation

● Database schema design

● UI/UX wireframes and design system

● Product backlog setup and initial priority

● Definition of Done and team working agreements

Sprint 1: Core User Authentication & Profiles (2 weeks) Objective: Create basic user authentication and profile management

Key Deliverables:

● User registration and login system

● User profile management and creation

● Role-based access control (driver vs passenger)

● Secure authentication system

● Password recovery system

Sprint 2: Location Services & Mapping (2 weeks)

Objective: Add basic location and mapping functionality

Key Deliverables:

● Integration with maps API

● Detection and tracking of current location

● Address search and geocoding

● Route calculation and visualization

● Handling of location permission

Sprint 3: Ride Request & Matching (2 weeks)

Goal: Build core ride request and driver matching functionality

Key Deliverables:

● Ride request form with location input

● Fare estimation algorithm

● Driver discovery and matching service

● Ride request handling system

● Notification system for ride status updates

Sprint 4: Payment Integration (2 weeks) Goal: Implement secure payment processing

Key Deliverables:

● Payment gateway integration

● Credit/debit card management

● Fare calculation service

● Payment receipt generation

● Transaction history

MVP RELEASE

Sprint 5: Ride Navigation & Management (2 weeks) Goal: Improve the ride experience through navigation capabilities

Key Deliverables:

●Navigation within the app for drivers

●Real-time tracking of driver locations

●ETA calculations and notifications

●Turn-by-turn instructions

●"Safety feature: Share my ride"

Sprint 6: Rating & Feedback System (2 weeks)

Goal: Introduce extensive rating and feedback systems

Key Deliverables:

●Rating system after the ride for both sides

●In-depth feedback gathering

●Views of rating history

● Issue reporting system

● Highly-rated driver reward system

User Stories for Passengers

User Story 1: Account Registration

As a passenger

I want to register an account using my email and phone number So that I can safely use the ride-hailing service

Definition of Done:

● User is able to register via email, phone number, and password

● Verification process for phone number and email is done

● Password strength is enforced

● Account creation is verified with proper notifications

● User information is stored safely in the database

● Registration flow is tested on all devices and platforms supported

● UI is equal to approved design specs

● Performance test indicates registration takes less than 3 seconds

User Story 2: Requesting a Ride

As a passenger

I'd like to enter my destination and ask for a ride

So that I can obtain transportation to my chosen location

Definition of Done:

●User is able to enter destination through address search or map choice

●Current location is detected automatically with manual override capability

●Estimated time of arrival and fare are shown prior to confirmation

●Alternative vehicle types/service levels can be chosen

●Ride request is properly sent to the matching system

●Error handling for regions with no service is done

●UI meets approved design specs

●Feature operates offline with proper messaging when network goes down

●Request process finishes in less than 5 seconds

3: Driver's Location Tracking

Passenger

to view my assigned driver's real-time location on a map

so that I can know when they will arrive and prepare in advance

Definition of Done:

●Driver's location is shown on an interactive map.

●Location is updated at least every 5 seconds.

●Estimated time of arrival is updated in real-time.

●Visual indicator indicates driver's progress along the route.

● Automatically scales map zoom and positioning for best viewing.

● Feature functions on all supported devices with proper responsiveness.

● Location tracking continues when app is in background.

● Battery optimization implemented and tested.

● Performance testing reflects insignificant impact on device resources.

**User Story 4: Making In-App Payments**

**As a** passenger

**I want to** securely pay for my ride through the app

**So that** I can complete transactions without cash

**Definition of Done:**

* Multiple payment methods can be added and stored securely
* Payment processing complies with PCI-DSS standards
* Ride fare is calculated accurately based on distance, time, and service type
* Receipt is generated and accessible in the app
* Transaction history is stored and viewable
* Error handling for failed payments is implemented
* Payment flow works across all supported devices and platforms
* Security testing completed with no critical or high vulnerabilities
* Performance testing shows payment processing completes in under 10 seconds

**5: Rating and Reviewing Driver**

passenger

**to** rate and review my driver after the ride

**that** I can provide feedback on my experience

**Definition of Done:**

* Rating prompt appears automatically after ride completion
* Rating scale (1-5 stars) is intuitive and accessible
* Optional text review field is available
* Pre-defined feedback categories are provided for quick selection
* Submitted ratings and reviews are stored in the database
* User can view their past ratings and reviews in profile
* UI/UX is consistent with app design guidelines
* Feature works in offline mode with syncing when connection is restored
* Accessibility testing ensures feature is usable by all users including those with disabilities

## User Stories for Drivers

**User Story 1: Driver Registration and Onboarding**

**As a** driver

**I want to** register and complete the verification process

**So that** I can start accepting ride requests

**Definition of Done:**

* Driver registration form collects all required information
* Document upload for license, insurance, and vehicle registration is functional
* Background check integration is implemented
* Step-by-step onboarding progress is tracked and displayed
* Approval/rejection notifications are sent appropriately
* Account status is clearly indicated in the driver dashboard
* All verification data is securely stored with appropriate access controls
* Process works across all supported devices
* Security testing completed with no critical vulnerabilities identified

**2: Accepting or Declining Ride Requests**

**As a** driver

**I want to** receive, view, and accept/decline incoming ride requests

**So that** I can choose which rides to take based on my preferences

**Definition of Done:**

* Incoming ride requests display pickup location, destination, and estimated fare
* Clear accept/decline buttons with appropriate timeout period
* Audible and visual notifications alert driver to new requests
* Accept action confirms assignment and provides navigation to pickup
* Decline action removes request and records reason (if provided)
* Request handling works when app is in background
* Performance testing shows notification delay under 2 seconds
* Feature functions correctly across all network conditions with appropriate error handling
* Analytics tracking implemented for acceptance/decline rates

**User Story 3: Navigation to Pickup and Destination**

**As a** driver

**I want to** get turn-by-turn directions to the pickup location and then to the destination

**So that** I can efficiently complete rides without getting lost

**Definition of Done:**

* Turn-by-turn navigation with voice guidance is implemented
* Route optimization accounts for traffic conditions
* Toggle between pickup and destination navigation is seamless
* Alternative routes are suggested when available
* Navigation continues in background if driver switches apps
* Offline maps functionality works when connectivity is lost
* Battery usage optimization is implemented and verified
* Feature works correctly across all supported devices
* Performance testing shows minimal impact on device resources
* Safety features prevent interaction while vehicle is in motion

**4: Tracking Earnings and Payments**

**As a** driver

**I want to** view my earnings, trips, and payment history

**So that** I can manage my finances and ensure I'm paid correctly

**Definition of Done:**

* Daily, weekly, and monthly earning summaries are displayed
* Trip-by-trip breakdown with fare details is available
* Payment schedule and status are clearly indicated
* Tax information is calculated and exportable
* Graphs and visualizations show earning trends
* Data export functionality for personal records is implemented
* Feature works across all supported devices with appropriate responsiveness
* All financial calculations are verified for accuracy
* Database queries optimized for performance with large transaction histories
* UI matches approved design specifications

**User Story 5: Managing Availability Status**

**As a** driver

**I want to** set my availability status (online/offline) and working hours

**So that** I can control when I receive ride requests

**Definition of Done:**

* Clear toggle for online/offline status
* Scheduled availability can be set for future time periods
* Current status is prominently displayed in the driver dashboard
* Automatic offline mode after specified period of inactivity
* Status changes are reflected in real-time on the server
* Notifications inform driver when status changes automatically
* Feature works across all supported devices
* Status changes work in poor connectivity with appropriate retry logic
* Battery usage optimization implemented for online status monitoring ● Analytics tracking implemented for availability patterns.

**7. Conclusion**: This report presents a holistic system design for a ride-hailing application, integrating contextual, interactional, and behavioral models to define its architecture, workflows, and dependencies. By adopting Agile methodologies, the design emphasizes iterative development, modularity, and adaptability, ensuring alignment with real-world complexities and user needs. Key insights include:

**1. System Scalability:** The integration of external systems (e.g., Stripe for payments, Google Maps for navigation) ensures scalability while maintaining loose coupling. This allows independent updates and minimizes downtime risks.

**2. User-Centric Workflows:** The interaction and behavioral models prioritize seamless user experiences, from ride booking to post-trip feedback. Real-time communication (SMS, push notifications) and dynamic pricing adjustments enhance reliability and transparency.

**3. Agile Success:** o Sprint-based development, user stories, and retrospectives enabled incremental delivery and risk mitigation. Tools like GitHub facilitated collaboration, version control, and conflict resolution, demonstrating the value of structured teamwork.

**4. Resilience & Compliance:** The system accounts for edge cases (e.g., payment failures, cancellations) and regulatory requirements through automated compliance reporting and redundant service integrations.