ACKNOWLEDGEMENT

We would like to express our deep sense of gratitude towards our respected teachers of Kathmandu Bernhardt College for providing a suitable platform to showcase our skills. Our warm respect goes to the Department of Computer Science and Information Technology for providing this opportunity to undertake a project and proceeding with it. We would also like to express our heartiest appreciation to our project supervisor**, Mr. Abhimanu Yadav** for his support and guidance throughout the journey of our project.

We can't remain idle without giving a lot of thanks to the Head of CSIT Department, **Mr. Abhimanu Yadav,** for his tremendous support and attention. Without his support and guidance, it would not have been possible to successfully complete this project.

Sincerely,
Mridani Pandey (28674/078)
Priyanka Luitel (28682/078)
Purna Bahadur Shrestha (28683/078)

ABSTRACT

Commuto is a non-profit web-based and Progressive Web App (PWA) platform designed to connect individuals within the same organization or institution, such as co-workers, students, teachers, and staff who share a similar commuting route. It is created to tackle the commuting difficulties faced by students and office staff, especially those without access to personal vehicles. By connecting riders, employees with bikes or scooters, with non-riders seeking transportation, Commuto offers an eco-friendly, affordable, and community-focused commuting solution. The platform fosters a sense of community while addressing practical challenges related to daily travel in and around organizations. Many individuals within the same organization face daily commuting challenges, leading to inconvenience and delays, but also contribute to increased traffic congestion and carbon emissions. Existing public transport options may not always be reliable or route- efficient, creating a gap in accessible and sustainable commuting solutions for workers. The main objective of Commuto is to provide an efficient and dependable ride-sharing solution tailored specifically for people in an organization who have to reach their workplaces or learning institutions on time. The platform aims to reduce daily commuting costs while promoting environmentally friendly travel by encouraging ride-sharing among verified users. It focuses on offering a secure and user-friendly experience through authentication via official email addresses, ensuring that only eligible participants can access the service. Additionally, Commuto seeks to improve workplace connectivity and foster a sense of community among individuals by enabling them to share rides and interact more regularly during their commutes. This is done using React.js, TypeScript, TailwindCSS, NestJS, PostgreSQL (Prisma ORM), and ride matching is the algorithm used. By leveraging modern web technologies and a user-centered design, Commuto stands as a technically sound and economically viable solution to a real-world transportation issue.

Keywords: *Ride-sharing, Organization-centered, Web application, Progressive Web App, Eco-friendly commuting, Resource utilization*

TABLE OF CONTENT

LIST OF ABBREVIATIONS

API	Application Programming Interface
CSS	Cascading Style Sheet
GUI	Graphical User Interface
IDE	Integrated Development Environment
ORM	Object-Relational Mapping
PWA	Progressive Web App
SSL	Secure Sockets Layer
SQL	Structured Query Language
UAT	User Acceptance Testing
UI	User Interface

LIST OF FIGURES

LIST OF TABLES

Chapter 1: Introduction

1.1 Introduction

Commuto is a web-based and Progressive Web App (PWA) platform designed to connect individuals within the same organization or institution—such as co-workers, students, teachers, and staff—who share similar commuting routes. By facilitating ride-sharing, Commuto offers an innovative solution for people to share rides to and from the same members of an organization in an environmentally-friendly and efficient manner. In the current landscape, many students and office staff face the difficulty of commuting long distances to attend classes and offices, especially if they lack their own vehicle or rely on public transportation. This not only leads to unnecessary expenses but also contributes to traffic congestion and increased carbon emissions. Commuto aims to address these issues by providing a platform where students can easily find others who are traveling along the same route and share rides.

Commuto is a web-based platform that connects individuals within the same organization—such as students or office staff—who have vehicles (riders) with those who need transportation (non-riders). The platform is designed to streamline commuting by offering a secure, efficient, and environmentally friendly ride-sharing solution. By focusing on riders' convenience and ensuring minimal deviations from their routes, Commuto promotes hassle-free, sustainable, and community-driven commuting. The platform enables Riders (students with personal vehicles) to post their available rides, including details such as departure time, pickup locations. Riders can post their available rides, specifying details such as departure time and route, while non-riders can browse these listings and request to join rides that align with their commute. By requiring a valid organizational email for access, Commuto maintains a trusted network of users, enhances safety, and fosters a stronger sense of community.

By leveraging organization-specific email authentication, Commuto ensures a trusted and secure environment where users can confidently connect with colleagues for shared travel. The platform intelligently matches commuters based on their home location and preferred travel time, ensuring that ride-sharing is both convenient and efficient without requiring drivers to deviate from their usual routes. This not only reduces the burden on drivers but also empowers non-drivers with a reliable and cost-free commuting option. As a result, Commuto helps foster a sense of community within the organization while actively contributing to lower transportation costs, reduced traffic congestion, and a more sustainable urban ecosystem.

By operating as a non-profit initiative, Commuto places its core emphasis on community benefit rather than commercial gain. The platform encourages a culture of mutual support, where ride-sharing becomes a practical extension of organizational solidarity. It not only addresses the daily commuting challenges faced by individuals without personal vehicles but also actively contributes to a greener environment by reducing the number of single-occupancy vehicles on the road. Through seamless digital interaction, intuitive features, and transparent communication between riders and passengers, Commuto ensures that shared travel becomes a dependable and socially responsible choice for all members of the organization.

1.2 Problem Statements

Many students and staff members face daily challenges commuting to and from the organization due to the absence of personal vehicles. Public transportation options, where available, are often unreliable, infrequent, or financially burdensome, especially for those living in remote or underserved areas. This can result in missed classes or work hours, increased stress, and reduced academic or professional performance. At the same time, individuals with personal vehicles may be open to sharing rides but lack a structured, secure, and efficient way to coordinate with others on similar routes. The absence of a centralized system leads to missed opportunities for collaboration and cost-saving. Commuto aims to bridge this gap by providing a non-profit, web-based ride-sharing platform that connects riders and non-riders within the same organization. It ensures that ride requests are limited to those located along the rider’s intended route, minimizing inconvenience and encouraging voluntary participation. By promoting shared travel, Commuto helps reduce commuting costs, traffic congestion, and environmental impact while fostering a greater sense of community.

1.3 Objectives

- To facilitate easy ride-sharing by connecting an individual of an organization who need a ride with those members who have a vehicle.
- To minimize rider inconvenience, ensure non-riders can request rides only if they are on the rider’s existing route.

1.4 Scope and Limitation

1.4.1 Scope

The Commuto project is designed to provide an efficient and reliable transportation management system specifically for members of the same organization. It facilitates ride booking, sharing, and tracking for students and staff, helping to optimize commuting for all users. The platform includes user registration, ride scheduling, real-time notifications, route optimization, and a rating system to enhance service quality. By providing a centralized and user-friendly interface, Commuto aims to reduce travel time, minimize traffic congestion on campus, and promote a safer and more organized transportation experience for the entire campus or organization community.

1.4.2 Limitation

Despite its advantages, the Commuto project has certain limitations. The system is limited to campus transportation and does not support off-campus or long-distance travel. Its effectiveness depends on active user participation; low engagement could reduce the availability of rides. Technical challenges such as internet connectivity issues, server downtime, or software bugs may impact performance. The platform also requires users to have access to smartphones and internet services, which may not be universally available. Additionally, while security measures are implemented, there are inherent risks related to storing personal and location data. Unforeseen circumstances such as vehicle unavailability or sudden schedule changes are difficult to handle.

1.5 Development Methodology

Agile methodologies are iterative and incremental, meaning they break a project into smaller parts and adjust to changing requirements. Agile methodology is a project management framework that breaks projects down into several dynamic phases, commonly known as sprints. The Agile framework is an iterative methodology. After every sprint, teams reflect and look back to see if there was anything that could be improved so they can adjust their strategy for the next sprint. Kanban is a popular Agile Software Development Methodology. Kanban is about envisioning the existing workflow in terms of steps. This methodology is selected as improvement is ongoing and data- driven. As Commuto develops, feedback from test users will drive frequent changes in priorities. Kanban is ideal in such situations, as it allows you to re-prioritize tasks in real time without disrupting workflow.

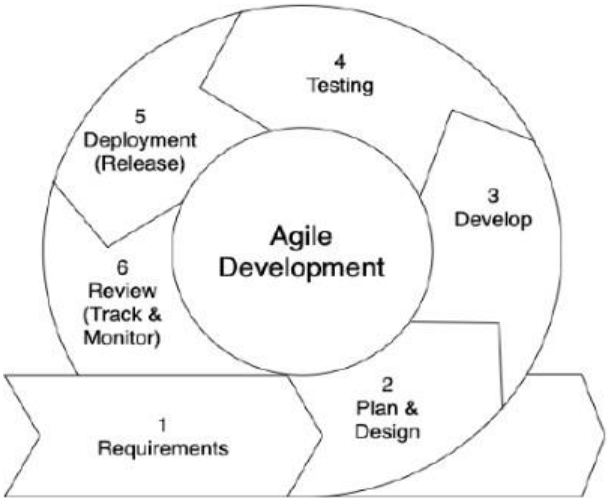


Figure 1.1: Agile Development

1.6 Report Organization

The materials presented in this project are organized into six chapters.

Chapter 1 represents the problem statement, objectives, scope, and limitations of the project.

Chapter 2 describes the fundamental theories and concepts as well as information about existing systems, journals, and references.

Chapter 3 summarizes the keynote on system analysis of the functional and non- functional requirements as well as different feasibility designs.

Chapter 4 represents the design whereas a description of the use case diagram, performance, and the algorithms being used.

Chapter 5 provides an account on implementation and testing, tools used for the preparation of the project. Test cases for unit testing as well as integration testing are done. Implementation details of the modules are traced.

Chapter 6 presents brief summaries of the outcome of the project, the conclusion, reviews as well as future recommendations, improvements that can be done on the upcoming days and feedback of systems, and the stability of the project.

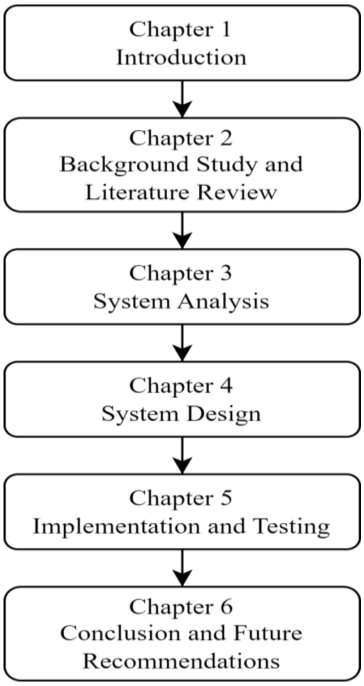


Figure 2.1: Report Organization of Commuto

Chapter 2: Background Study and Literature Review

2.1 Background Study

In recent years, urban transportation challenges have become increasingly prominent, especially among student and staff populations in cities where public transportation is either insufficient, unreliable, or costly. Individuals who lack access to personal vehicles often rely on inconsistent public transit systems, which can lead to missed classes, increased stress, and a decline in academic performance. These issues are particularly prevalent in densely populated urban areas where colleges are widely distributed but public transportation infrastructure has not scaled accordingly.

Ride-sharing platforms like Uber and Lyft have revolutionized commuting globally along with in Nepal, alternatives better suited for students and staff with short routine commutes are offered mainly by local apps like Yango, Pathao, and inDrive. But, such platforms lack the ability to connect members within the same community in a personalized and secure manner. Existing solutions also rarely incorporate route optimization based on the daily schedules and paths of individual users, which is critical for minimizing travel time and inconvenience for both riders and passengers.

Commuto addresses these gaps by offering a platform tailored specifically to both students and staff of the same institution. It allows riders and non-riders within the same institution or area to coordinate daily commutes based on shared routes. By restricting requests to only those who are on the rider's route, the system minimizes disruptions while encouraging community collaboration, cost-saving, and environmentally friendly travel habits. This approach not only enhances accessibility for individuals but also helps reduce traffic congestion and pollution by promoting shared mobility solutions.

2.2 Literature Review

Commuto, referred to as a "Organization Commute Helper" introduces a technological solution aimed at alleviating transportation difficulties encountered by students, faculty, and staff. As student enrollments rise and the demand for shared transit options grows, it becomes crucial to analyze commuting behaviors and the role of technology in addressing these challenges. This review delves into campus commuting issues, trends in shared mobility, technological transportation platforms, and user experience principles to outline the objectives of Commuto. Effective transportation applications hinge on a user-friendly design, real-time functionalities, robust safety measures, and easy navigation to build trust among users.

Commuto seeks to incorporate these technological principles while addressing affordability and equity issues to ensure access for low-income students and those facing technological barriers. By utilizing smart routing algorithms and feedback mechanisms, Commuto aligns with theories of shared mobility and insights into user behavior. Share mobility's primary goals include reducing transportation barriers, minimizing environmental impacts, and enhancing commuting experiences through innovative solutions [1].College campuses face significant challenges such as parking shortages, overcrowded transit systems, and environmental concerns linked to single-occupancy vehicle usage.

Shared mobility solutions like carpooling and ridesharing have emerged as effective strategies to mitigate congestion and reduce emissions, particularly when enhanced by technology-driven platforms like UberPool and RideAmigos, which provide real-time features for convenience and cost-effectiveness globally. Likewise within Nepal, alternatives better suited for students and staff with short routine commutes are offered mainly by local apps like Yango, Pathao, and inDrive. RideAmigos Commute is a cloud-based commute management platform designed to help organizations streamline and improve how their employees or members travel to work or campus [2]. However, campus commuting is characterized by unique factors such as short travel distances and safety issues that necessitate customized rideshare models. This eco-environmental narrative is common to many university car share policies; however, other institutions have begun to highlight the social, emotional and relational benefits of car sharing, such as saving time spent looking for a parking space with dedicated car share bays for car sharers; enjoying the social aspect of commuting with friends and colleagues; and improving work–life balance by leaving work on time (University of Exeter)[3]. The passenger agent plans the trip according to passenger preferences and based on the real- time information available from road-segment agents (travel time) and vehicle agents (routes and dwell times) [4].

For this leaflet, a JavaScript library is used to create interactive, web-based maps. With it, you can create a simple map in as little as three lines of JavaScript, or you can create complex, interactive, editable maps with hundreds of lines of code [5]. By offering a centralized system to manage carpooling, transit options, parking, and incentives, Commute supports both individual commuting needs and broader organizational goals like sustainability, reduced traffic, and efficient parking use. Energy consumption and CO2 emission can also be reduced due to the increased vehicle utilization rate [6].

In summary, Commuto aims to create a more efficient and equitable transportation ecosystem on campus, fostering a sense of community while addressing the pressing challenges of modern campus commuting. The passenger agent plans the trip according to passenger preferences and based on the real-time information available from road- segment agents (travel time) and vehicle agents (routes and dwell times).

Chapter 3: System Analysis

3.1 System Analysis

System analysis is a review of a technological system, like a software package, for troubleshooting, development or improvement purposes. Through in-depth analysis, analysts can uncover errors in code, accessibility issues for end-users or design incompatibilities.

3.1.1 Requirement Analysis

Requirement analysis is done to understand and document the needs of the users and the system functionalities essential for developing Commuto. It is divided into two parts. i.e. functional and non-functional requirements.

i. Functional Requirement:

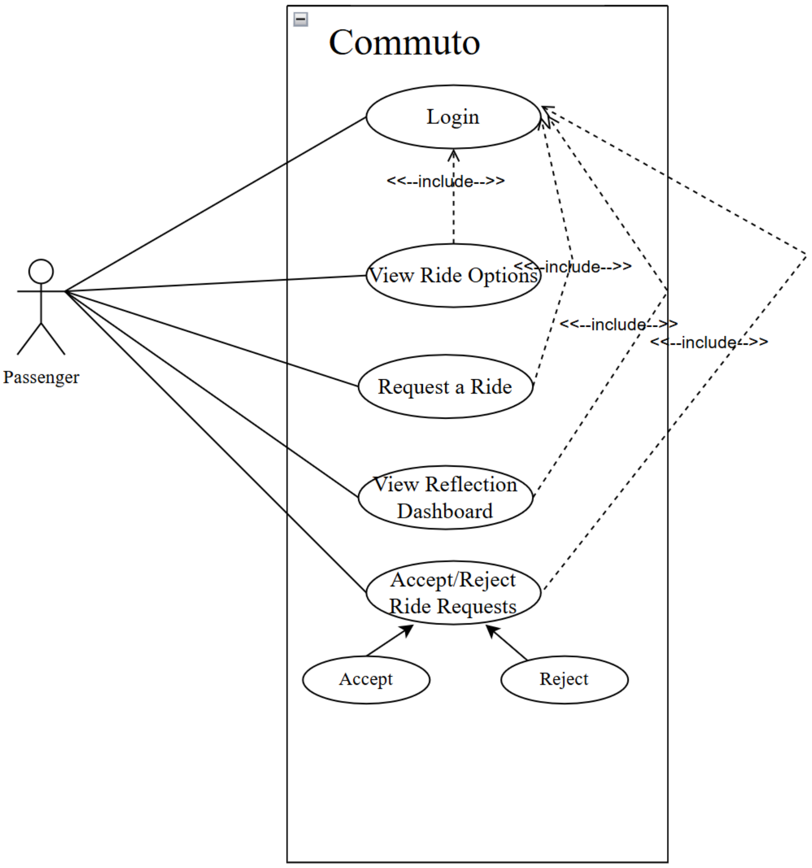


Figure 3.1.1: Use Case Diagram (Passenger) of Commuto

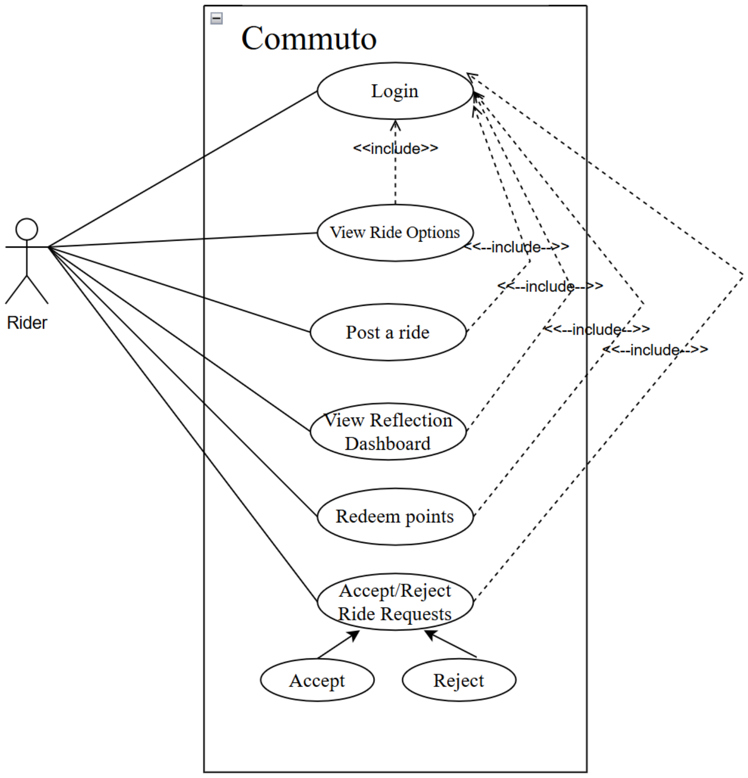


Figure 3.1.2 Use Case Diagram (Rider) of Commuto

Commuto offers a structured and user-friendly ride-sharing experience through a series of integrated features that ensure security, efficiency, and engagement. User authentication enables participants to securely log in using valid organizational emails or phone numbers, with options for session management. Through User profiles, individuals can view their login status and customize preferences such as light or dark mode for a personalized interface. The Ride search and matching system allows passengers to find rides based on pickup, destination, and time, while riders can easily post their trip details through the Ride posting and management feature. Passengers may then submit Ride requests, which riders can accept or decline depending on availability, with confirmed trips notifying both parties of pickup details and rider information. Upon ride completion, passengers can leave reviews, while riders can mark the ride as completed, leading both to a self-reflection dashboard and access to the Karma points redeem page, reinforcing fairness and sustained participation within the community.

ii. Non-Functional requirements:

a) **Performance**: The portal loads within 3 seconds and handle high traffic and large data volumes efficiently.

b) **Scalability:** The system have scalability to accommodate increasing numbers of users and ride postings.

c) **Security:** Implemented SSL encryption, secure data storage, compliance with data protection regulations, and regular security audits**.**

d) **Usability:** The portal have an intuitive, user-friendly interface and accessible design for users with disabilities.

e) **Reliability:** Commuto has high availability with minimal downtime and regular backups to prevent data los**s.**

f) **Compatibility:** Performed cross-browser and cross-device support (Chrome, Firefox, Edge, Safari; mobile + desktop).

3.1.2 Feasibility Analysis

The purpose of the feasibility study is to assess the practicality of the system. This study also helps to foresee any potential problems that likely arise during or after the deployment of the system by taking in all the substantial factornto account. This chapter contains Technical, Operational, Economic and Schedule feasibility.

1. **Technical feasibility**

   The project can be developed using basic web technologies such as React.js, TypeScript, Tailwind CSS and a simple backend framework like Nest.js. PostgreSQL database will be used.

   Table 3.1 Hardware Requirements

Hardware Requirements	
Device	Laptop
Connection	Active & stable internet connection
Processor	i3, i5, i7 or i9
RAM	Minimum 4GB and multi-core

2. **Economic Feasibility**

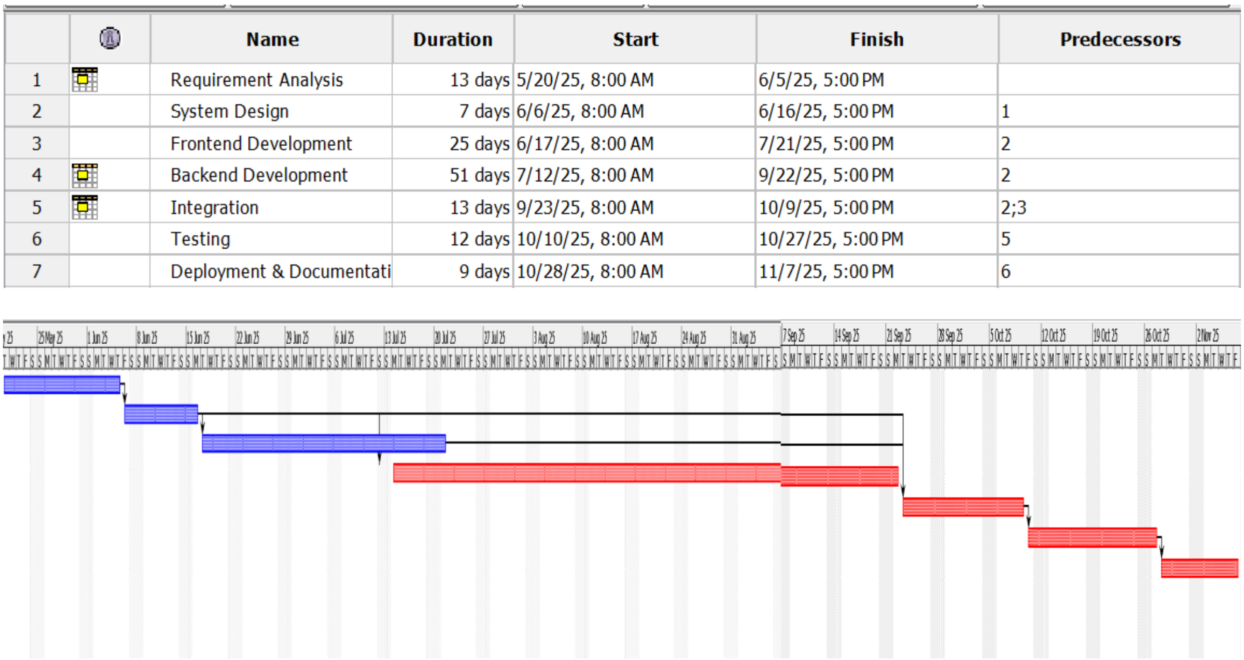
   The resources required for this project are minimal. The project does not require significant financial investment, making it economically feasible for a college project. All developers and designers have the necessary computers and tools, which are either free or open source.

3. **Operational Feasibility**

The portal will be user-friendly and straightforward, focusing on core functionalities like user registration, job posting, and job searching. The scope is limited to ensure it can be completed within the project timeline and with the available resources.

4. **Schedule feasibility**

The project can be completed within 3-4 months. A detailed project plan with milestones for design, development, testing, and deployment phases will help ensure timely completion. Regular progress reviews and adjustments will keep the project on track.



		Name	Duration	Start	Finish	Predecessors
1		Requirement Analysis	13 days	5/20/25, 8:00 AM	6/5/25, 5:00 PM	
2		System Design	7 days	6/6/25, 8:00 AM	6/16/25, 5:00 PM	1
3		Frontend Development	25 days	6/17/25, 8:00 AM	7/21/25, 5:00 PM	2
4		Backend Development	51 days	7/12/25, 8:00 AM	9/22/25, 5:00 PM	2
5		Integration	13 days	9/23/25, 8:00 AM	10/9/25, 5:00 PM	2;3
6		Testing	12 days	10/10/25, 8:00 AM	10/27/25, 5:00 PM	5
7		Deployment & Documentati	9 days	10/28/25, 8:00 AM	11/7/25, 5:00 PM	6

Figure 3.2 Gantt chart of Commuto

3.1.3 Analysis

A. Object Modelling: Class and Object Diagram

a) Class Diagram

The class diagram for the Commuto system defines the structure of the main entities in the system, such as Passenger, Rider, Ride, RideRequest, and Authentication. Each class contains relevant attributes and methods. For instance, the Passenger class includes attributes like name, email, and methods such as requestRide() and viewRideOptions(). Associations between classes represent relationships like a Passenger can make multiple RideRequests, and each RideRequest is associated with a Ride. This diagram helps visualize the static relationships and the system's data model.

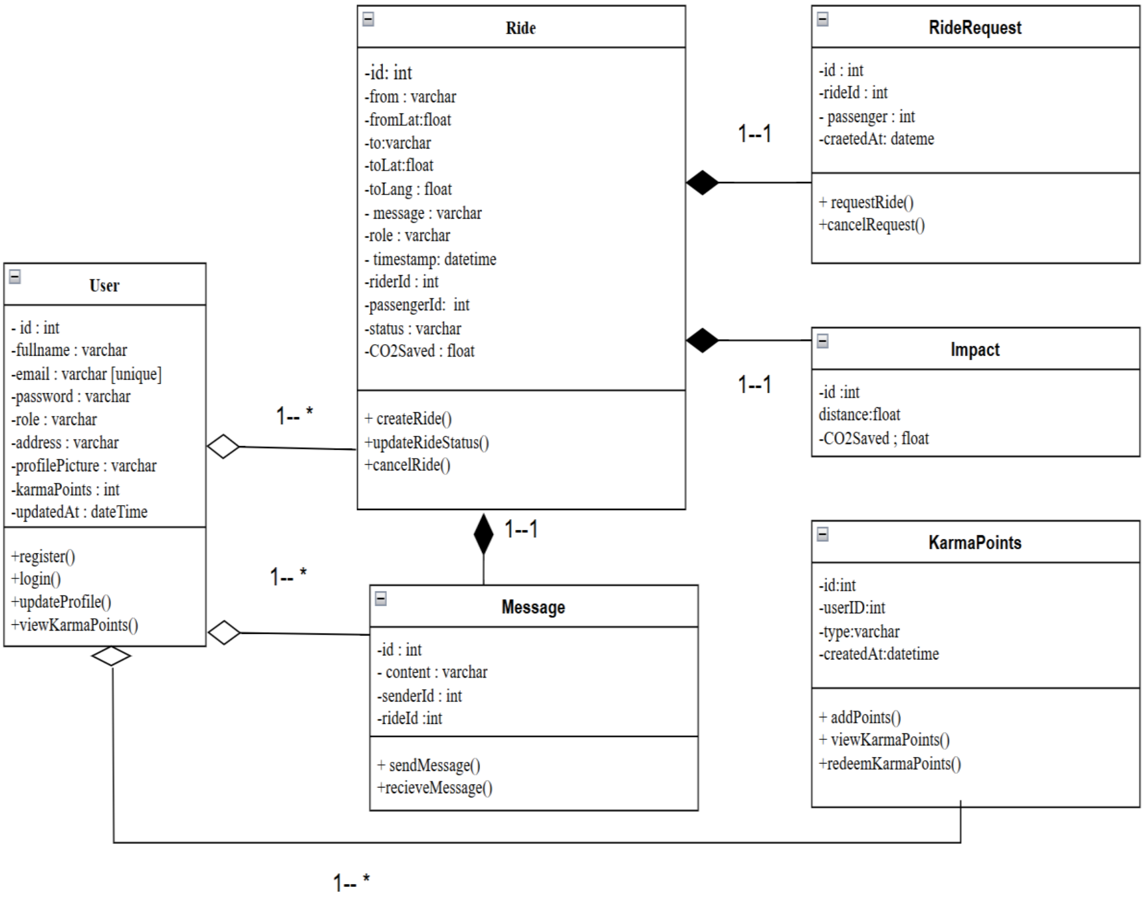


Figure 3.3 Class Diagram of Commuto

b) Object Diagram

The object diagram provides a snapshot of objects from the class diagram at a specific point in time. It shows how a particular passenger is connected to a specific ride request and ride details at runtime. This helps understand the state and relationships between real objects in a particular scenario of the system.

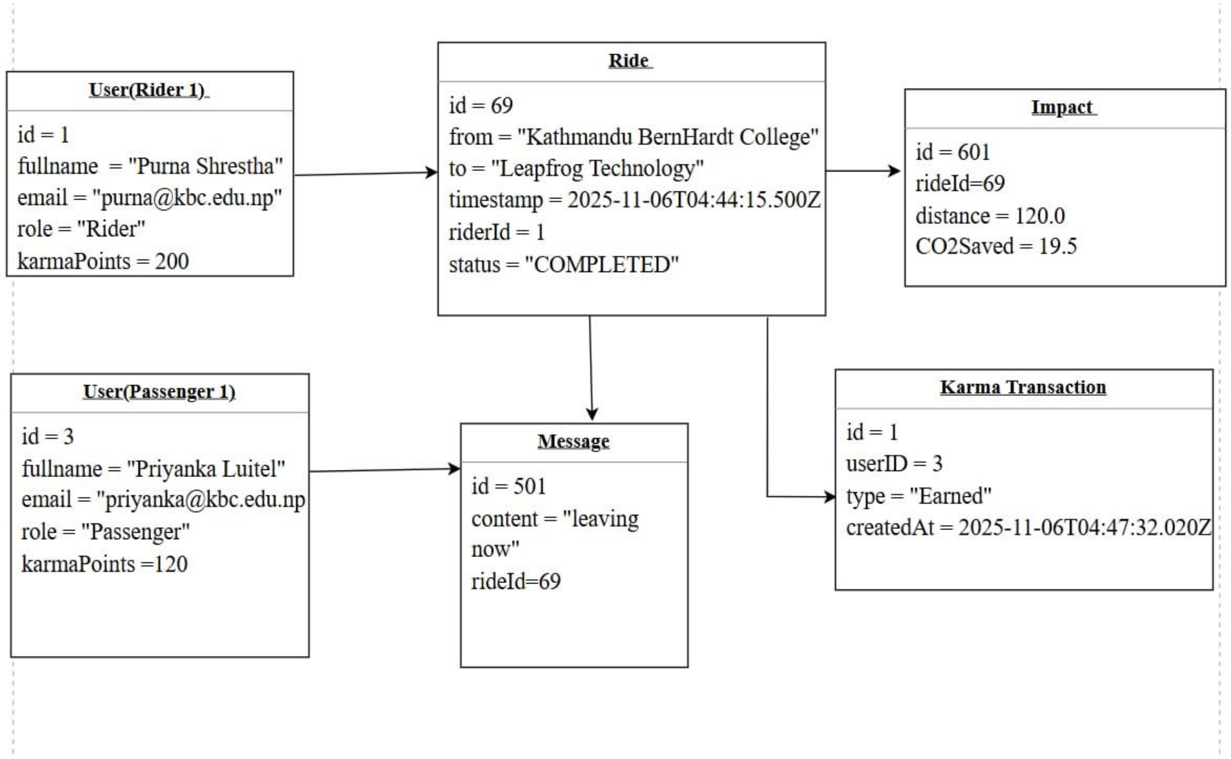


Figure 3.4 Object Diagram of Commuto

ii) Dynamic Modelling: Activity, State and Sequence Diagram

a) Activity Diagram:

This activity diagram illustrates the overall workflow of the Commuto system, where both passengers and drivers interact concurrently. The process begins with a user logging into the system, and only upon successful authentication can they proceed further. Once logged in, two parallel activities may occur: a passenger can request a ride, while a driver can search for or offer a ride. These actions merge at a decision point where the system checks if a ride has been successfully matched or offered. If no ride is available, the process loops back to allow continued searching or requesting. When a ride is offered, the system progresses to the ride being in progress, followed by the ride finishing. Finally,the process terminates, indicating the successful completion of the ride-sharing activity. This diagram emphasizes both the decision-based flow and the concurrency of passenger and driver actions within the system.

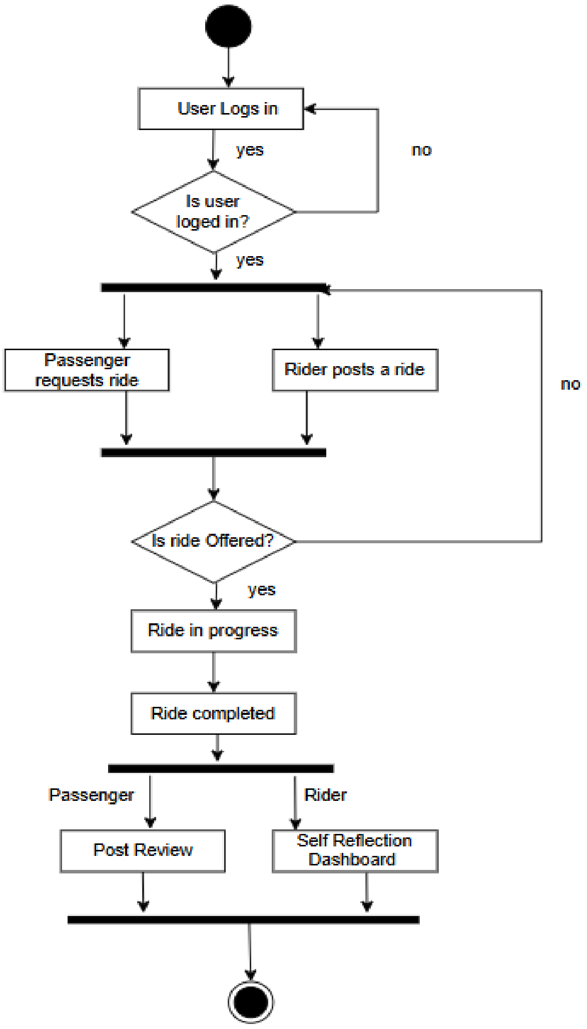


Figure 3.5: Activity Diagram of Commuto

b) State Diagram

The state diagram for the Passenger user in the Commuto system illustrates the various states the passenger can be in during their interaction with the system. It starts from the Logged Out state, transitions to Logged In, then to Browsing Rides, Requesting Ride, and finally waiting for Confirmation. Depending on the Rider's action, the state transitions to either Ride Confirmed or Ride Rejected. This diagram models the lifecycle and behavioral changes of the passenger based on system events.

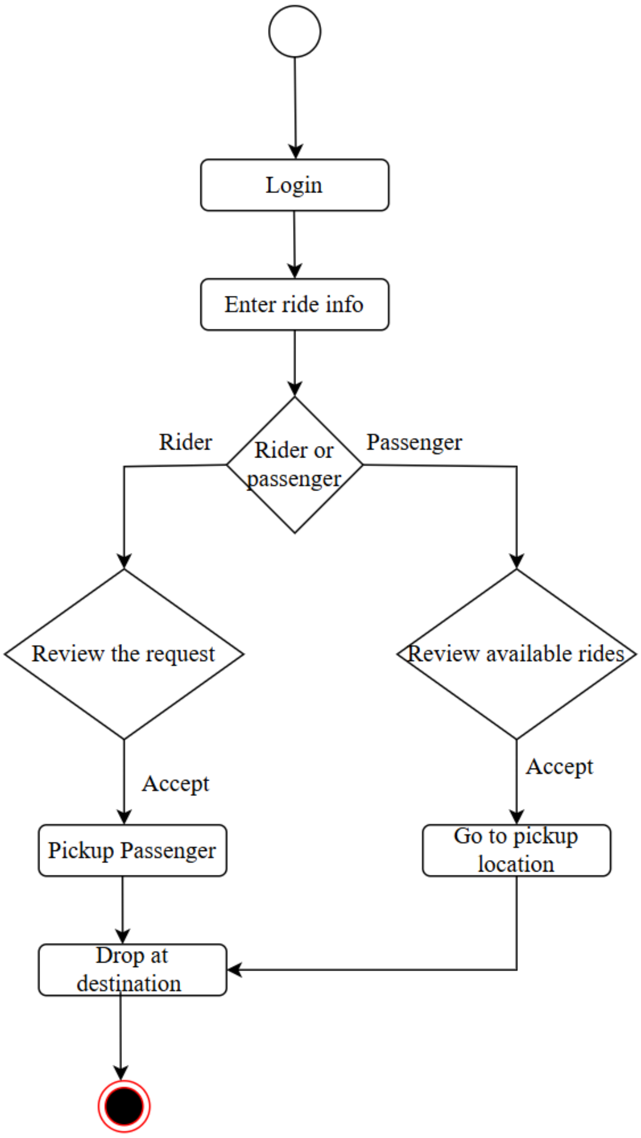


Figure 3.6: State Diagram of Commuto

c) Sequence Diagram

The sequence diagram for the Commuto system represents the interaction flow between objects over time. It starts with the Passenger initiating a login request, followed by viewing available rides, selecting a ride, and sending a request. The system then confirms the request or provides feedback. The main actors include Passenger, Authentication Service, Ride Service, and Ride Request Handler. This diagram highlights the order of message passing and the dynamic behavior of the system during a ride request process.

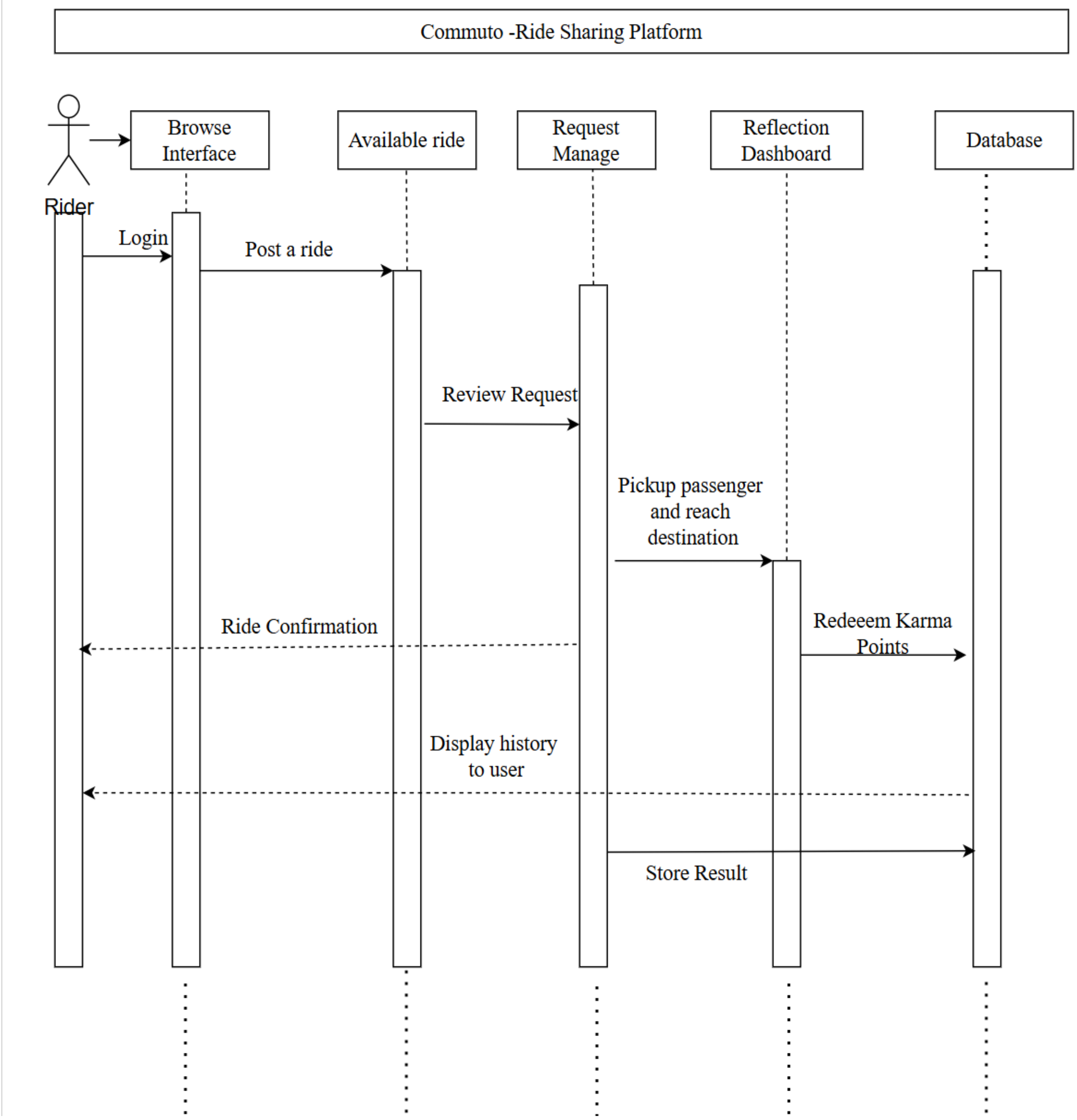


Figure 3.7.1: Sequence Diagram of Commuto(rider)

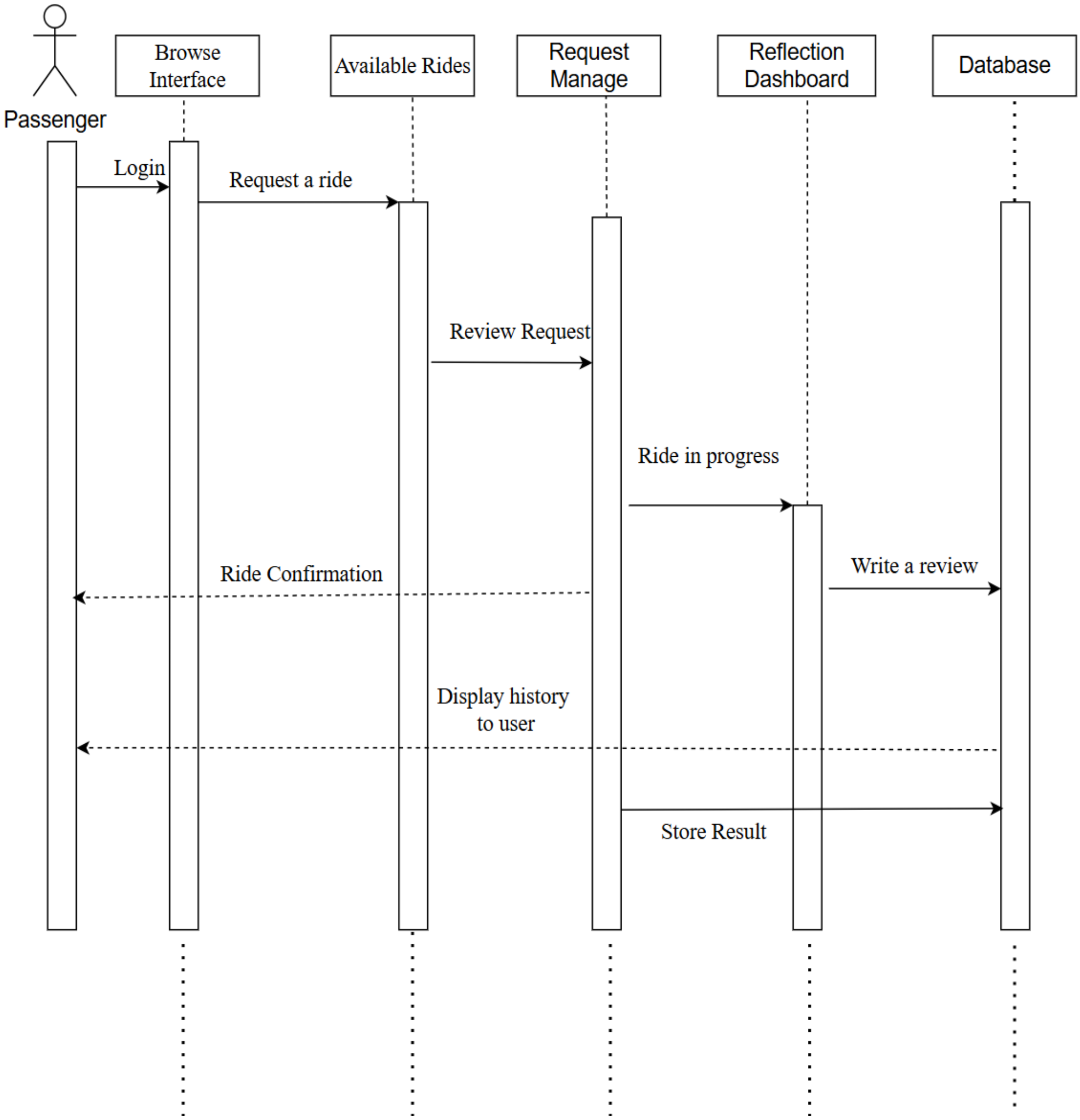


Figure 3.7.2: Sequence Diagram of Commuto(passenger)

Chapter 4: System Design

4.1 Design

4.1.1 System Architecture and Overview

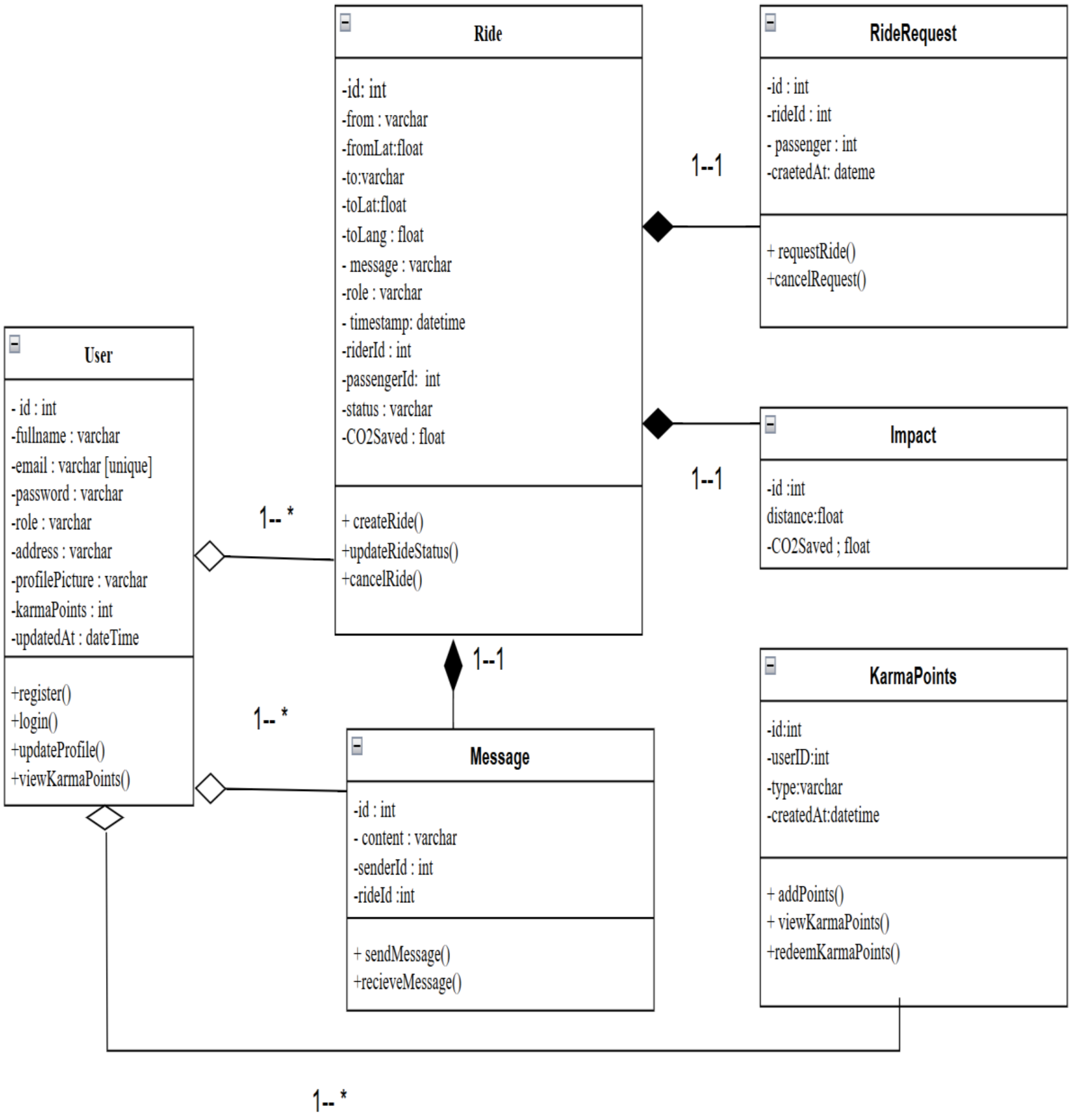


Figure 4.1 Class Diagram of Commuto

This class diagram defines the conceptual framework of the ride-sharing system. It captures how users interact with rides, send messages, and accumulate karma points through eco-friendly actions. It also integrates environmental tracking through the Impact class and manages user engagement via the KarmaPoints mechanism.

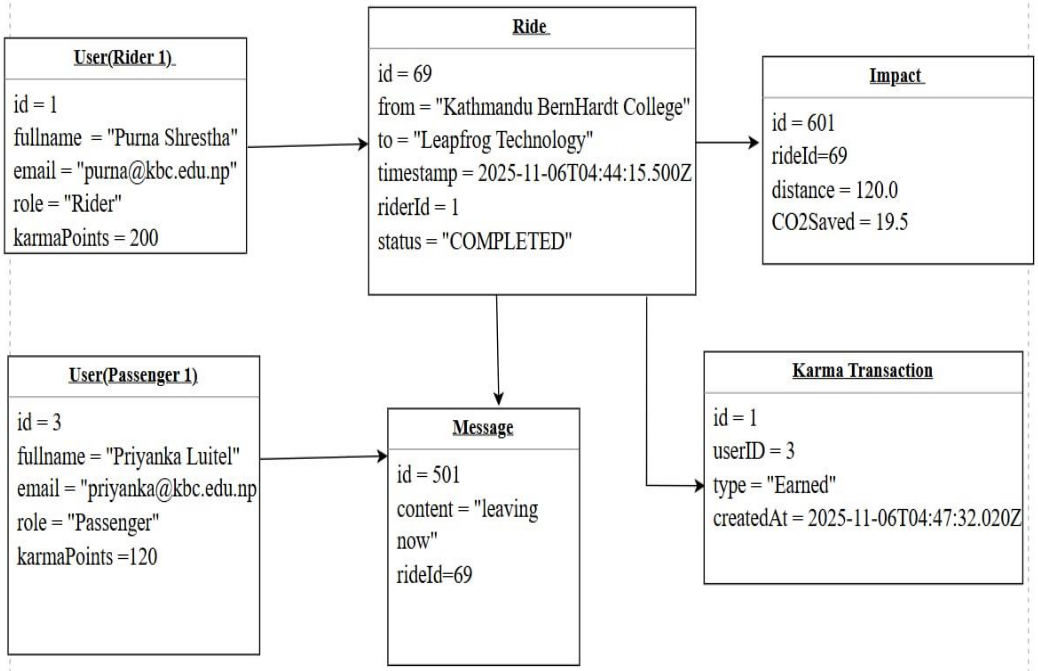


Figure 4.2 Object Diagram of Commuto

This refined object diagram effectively visualizes a real instance of how the ride-sharing system behaves at runtime. It takes the abstract relationships defined in the class diagram and turns them into tangible examples, showing how actual objects interact in a real situation. It visualizes the structure as well as the flow of information and the state of each object once the ride has been completed. In this snapshot, the instance is portrayed as “Purna Shrestha” being a rider and “Priyanka Luitel” being a passenger. The passenger has passed along a “leaving now” message to the rider. Once the ride has been completed, the system calculated an Impact of 19.5 Co2 saved in travelling 120km through Commuto. The rider earned karma points and the passenger earned credit score for completing the ride.

4.1.2 Component Diagram

The component diagram for the Commuto system shows key parts like User Interface, Authentication, Ride Management, Ride Request Handler, and Ride Service. It explains how these parts work together to support login, ride selection, and request confirmation. This diagram highlights the system's main components and how they interact to handle ride requests.

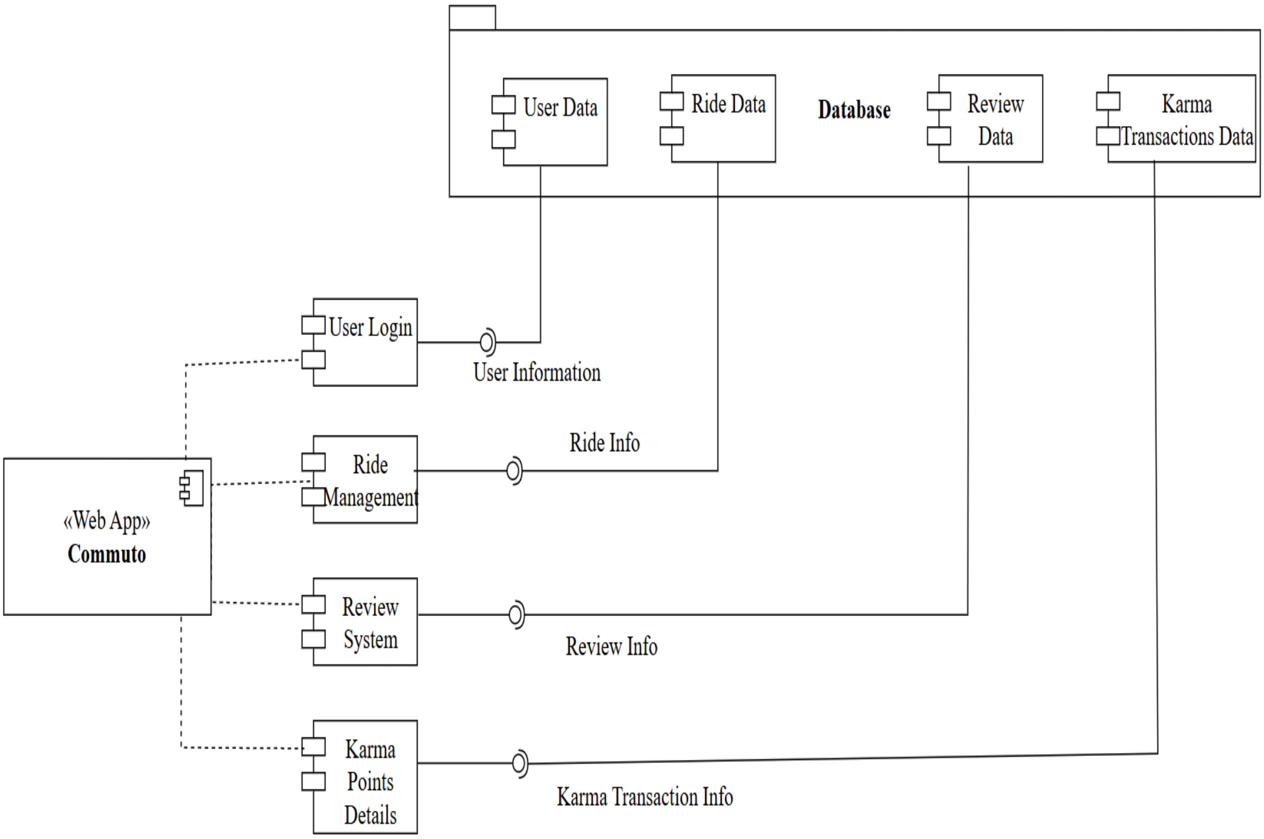


Figure 4.3: Component Diagram of Commuto

4.1.3 Deployment Diagram

The deployment diagram for the Commuto system shows how software components like Authentication Service, Ride Service, Ride Request Handler, and User Interface are deployed on physical hardware nodes such as servers and mobile devices. It illustrates the physical setup, including nodes (hardware devices), artifacts (software components), and communication paths (network connections). This diagram helps visualize how the system is physically distributed and connected in its running environment.

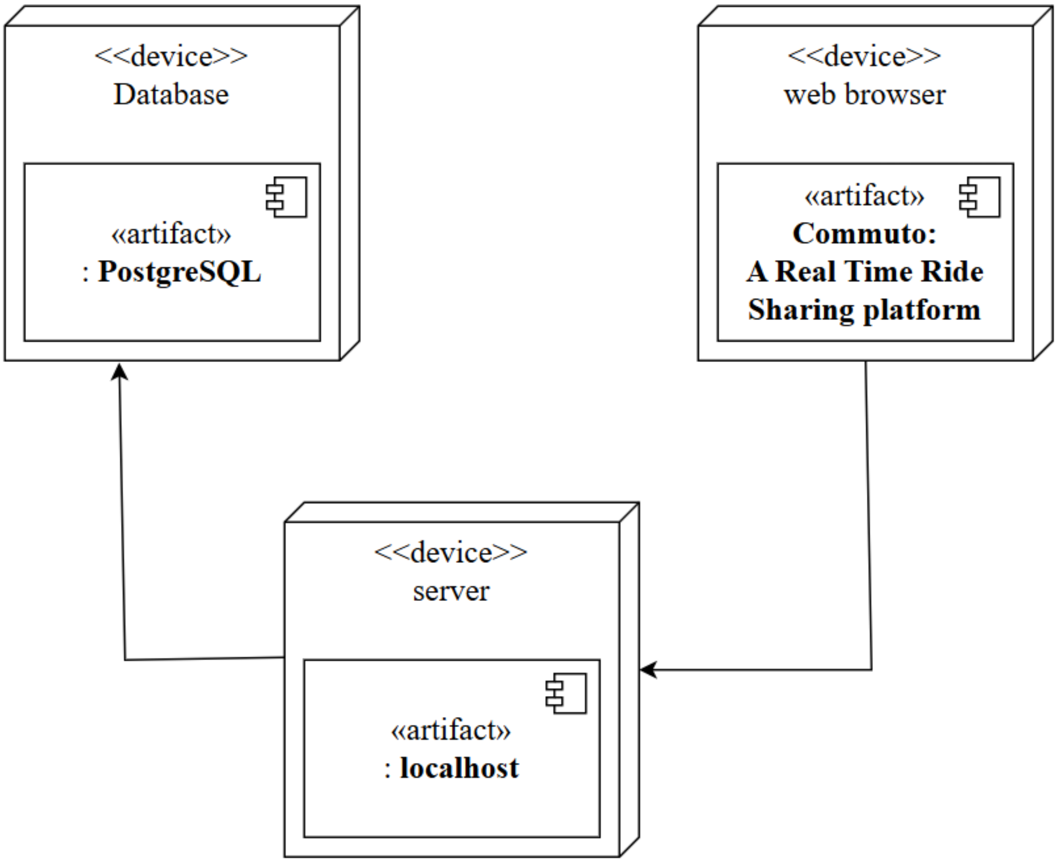


Figure 4.4 Deployment Diagram of Commuto

4.2 Algorithm Details

The following algorithms are used for this project:

4.2.1 Combined Route Matching Algorithm

1. **Input from Rider:**

   Rider provides their current location and confirms they are commuting to college (fixed destination).

2. **Input from passenger:**

   Non-rider users enter their current location when searching for rides.

3. **Matching Logic (Haversine Algorithm):**

   a. Convert both locations into geographical coordinates (latitude & longitude).

   b. Use the Haversine formula to calculate the straight-line distance between the rider and the non-rider.

   c. If the non-rider is within a certain proximity radius (within 3 km) along the rider’s route, display the ride as "Available".

   d. Optionally (future improvement): Analyze the direction of travel to ensure the non-rider is on the rider’s path toward college, not behind or in the opposite direction.

4.2.2 Algorithm Used:

Haversine Formula

Used to calculate the shortest distance between two points on a sphere based on latitude and longitude.

Where φ is latitude, λ is longitude, and r is Earth's radius (~6371 km).

#Constants

EARTH_RADIUS_KM = 6371 PROXIMITY_RADIUS_KM = 3.0

$$d = 2r.\arcsin\left(\sqrt{hav(\varphi_2-\varphi_1)+\left(1-hav(\varphi_1-\varphi_2)-hav(\varphi_1+\varphi_2)\right).hav(\lambda_2-\lambda_1)}\right)$$

$$= 2r.\arcsin\left(\sqrt{sin^2\left(\frac{\varphi_2-\varphi_1}{2}\right)+\left(1-sin^2\left(\frac{\varphi_2-\varphi_1}{2}\right)-sin^2\left(\frac{\varphi_2+\varphi_1}{2}\right)\right).sin^2\left(\frac{\lambda_2-\lambda_1}{2}\right)}\right)$$

$$= 2r.\arcsin\left(\sqrt{sin^2\left(\frac{\varphi_2-\varphi_1}{2}\right)+cos\varphi_1.cos\varphi_2.sin^2\left(\frac{\lambda_2-\lambda_1}{2}\right)}\right)$$

Equation 1

#Function to calculate distance using the Haversine formula

```
function calculateHaversineDistance(lat1, lon1, lat2, lon2):
dLat = radians(lat2 - lat1)dLon = radians(lon2 - lon1)
a = sin(dLat / 2)^2 + cos(radians(lat1)) * cos(radians(lat2)) *
sin(dLon / 2)^2 c = 2 * atan2(sqrt(a), sqrt(1 - a))
return EARTH_RADIUS_KM * c
 #Function to match riders based on proximity
function findMatchingRiders(nonRiderLocation, activeRidersList):
matchingRiders = [] for each rider in activeRidersList:
    riderLocation = rider.currentLocation

    # Step 1: Calculate the distance between the non-rider and the
    rider distance = calculateHaversineDistance(
        riderLocation.latitude,
        riderLocation.longitude,
        nonRiderLocation.latitude,
        nonRiderLocation.longitude
    )
 # Step 2: Check if the non-rider is within the proximity
    threshold if distance <=
    PROXIMITY_RADIUS_KM:
        # Step 3: Optionally check if the rider is heading toward the college # This step can
        be added later with direction vector filtering matchingRiders.append(rider)
return matchingRiders
```

This Haversine algorithm is then optimized in order to take into account time, opposite role of users and their location .This ensures that a rider only gets matched with a non-rider (passenger) who is within a 3km radius of each other. It also takes into consideration the time of request i.e it makes sure that the difference in time is ± 30 minutes.

Tiered Linear Scaling with Sentiment Weighting

$P = \min(\max((B \times D_m) + S_b, F_{min}), F_{max})$

where:

P = Final points awarded

B = Base points (participation reward) = 15

D_m = Distance multiplier (tier-based scaling factor)

S_b = Sentiment bonus (quality weighting)

F_min = Minimum floor (system stability) = 5

F_max = Maximum cap (exploitation prevention) = 100:

This algorithm awards karma points based on both ride distance tiers and feedback sentiment, providing fairer reward distribution. introducing a formula that adjusts the reward according to both the distance of the ride and the sentiment of the feedback received.This base is multiplied by a distance multiplier that increases with the length of the ride, ensuring that longer rides receive proportionally higher rewards. A sentiment bonus is applied based on the positivity or quality of feedback, allowing users who deliver better experiences to earn extra points. To maintain fairness and prevent extreme values, the system will enforce a minimum and maximum limit, ensuring rewards stay within a reasonable range.

Chapter 5: Implementation and Testing

5.1 Implementation

5.1.1 Tools used

Frontend Tools

- **React:**

  In the Commuto system, React is used for building the user interface using components. It plays a central role in delivering a responsive, modular, and maintainable web application. Technologies like TypeScrip**t** combined with a component-based UI library (like React) allow Commuto to manage complex user interactions for both Passengers and Riders efficiently.

- **TypeScript:**

  In the Commuto system, TypeScript plays a crucial role in ensuring a **s**mooth**,** predictabl**e**, and secure user experience across the entire user interface. It offers the benefits of optional static typing, allowing types to be added to variables, functions, properties, etc.

- **Tailwind CSS:**

  In the Commuto system, Tailwind CSS is used to build a clean, responsive, and visually consistent user interface across both Passenger and Rider experiences .Tailwind **CSS** provides a robust and scalable approach to UI design.

- **OpenStreetLayer Map:**

  The map API gives Commuto a free, open-source, and community-driven alternative to commercial map services. By integrating with an open Map API (such as OpenStreetLayerMap **or** Leaflet), Commuto gains full control over map functionality without relying on costly or restrictive third-party services.

Backend Tools

- **NestJS:**

  NestJS provides a TypeScript-first approach, making it easier to maintain consistency across the code base. Built on top of Node.js and leveraging TypeScrip**t** out of the box, NestJS allows Commuto to implement well-structured backend logic while maintaining full control over performance, security, and maintainability.

- **Nest Winston (Logging):**

  The system captures and organizes logs from all core modules such as authentication, ride posting, ride matching, karma points, and emission estimation. Nest Winston provides structured logging, making it easier to debug, monitor, and audit system behavior across the entire backend.

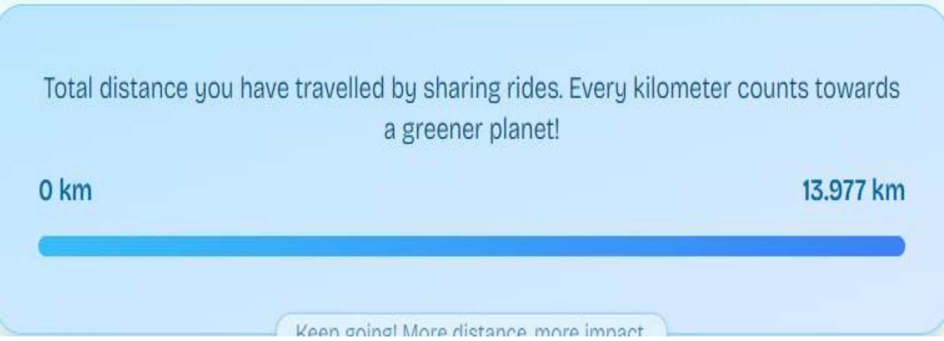
- **PostgreSQL:**

  It is used to maintain critical records such as user authentication details, ride postings, ride requests and confirmations, karma point transactions, and emission estimation data. As an open-source, enterprise-grade database, PostgreSQL gives Commuto the tools to ensure data integrity, security, and performance at every stage of the user journey.

5.1.2 Implementation Details of Modules

i. Module 1: Distance Calculation

```
export const EARTH_RADIUS_KM = 6371;
export const DEG_TO_RAD = Math.PI / 180;
export const MAX_RIDE_PROXIMITY_KM = 3;
export function haversineDistance(
 lat1: number,
 lon1: number,
 lat2: number,
 lon2: number, ): number {
const dLat = (lat2 - lat1) * DEG_TO_RAD;
const dLon = (lon2 - lon1) * DEG_TO_RAD;
const a = Math.sin(dLat / 2) * Math.sin(dLat / 2) +Math.cos(lat1 *
DEG_TO_RAD)* Math.cos(lat2 * DEG_TO_RAD) * Math.sin(dLon / 2) *
Math.sin(dLon / 2);
const c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));
return EARTH_RADIUS_KM * c;
}
```

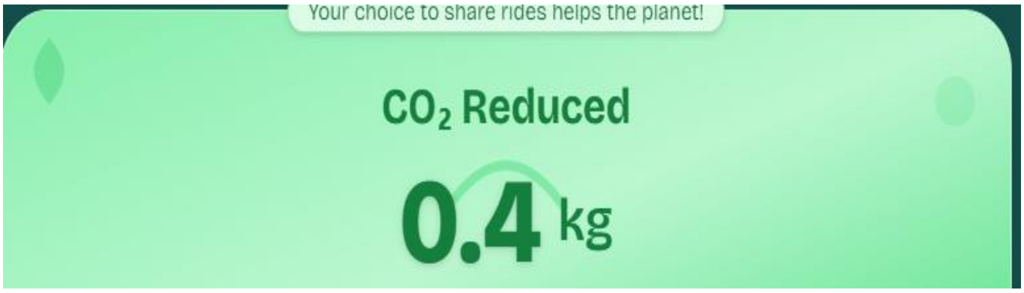


Total distance you have travelled by sharing rides. Every kilometer counts towards a greener planet!

Fig 5.1: Distance calculation using haversine

ii. Module 2: Carbon Emission Calculation

```
export enum TransportSpeed {
  BIKE = 38.28,
}
export enum EmissionFactor {
  BIKE = 0.016,
  CAR = 0.17144,
}
export function estimateCO2FromDistance(
distanceKm: number,
vehicle: EmissionFactor = EmissionFactor.BIKE, // Default to bike
): number {
return distanceKm * vehicle;
}
```

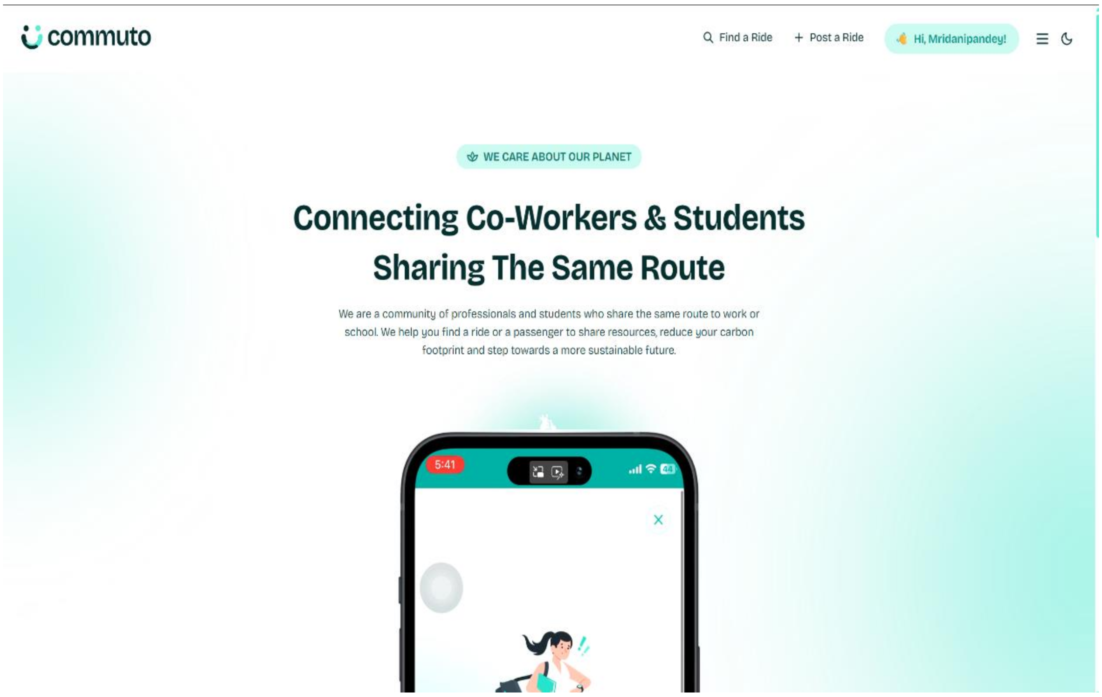


Your choice to share rides helps the planet!

Fig 5.2: Carbon emission calculation

iii. Module 3:User authentication

```
const user = await this.prisma.user.findUnique({
      where: { email: body.email },
    });
    if (!user) {
      this.logger.log({
        level: 'warn',
        message: `Login failed for email: ${body.email} - User not found`,
        tag: 'error',
        email: body.email,
      });
      throw new UnauthorizedException('User not found');
    }
    const isPasswordValid = await bcrypt.compare(body.password, user.password);
    if (!isPasswordValid) {
      this.logger.log({
        level: 'warn',
        message: `Login failed for email: ${body.email} - Invalid password`,
        tag: 'error',
        email: body.email,
      });
      throw new UnauthorizedException('Invalid password');
    }
    // Remove password field from user object for response
    const userWithoutPassword = Object.fromEntries(
      Object.entries(user).filter(([key]) => key !== 'password'),
    );
    this.logger.log({
      level: 'info',
      message: `Login successful for email: ${body.email}`,
      tag: 'auth',
      email: body.email,
      userId: user.id,
    });
    return { message: 'Login successful', user: userWithoutPassword };
  }
```



WE CARE ABOUT OUR PLANET

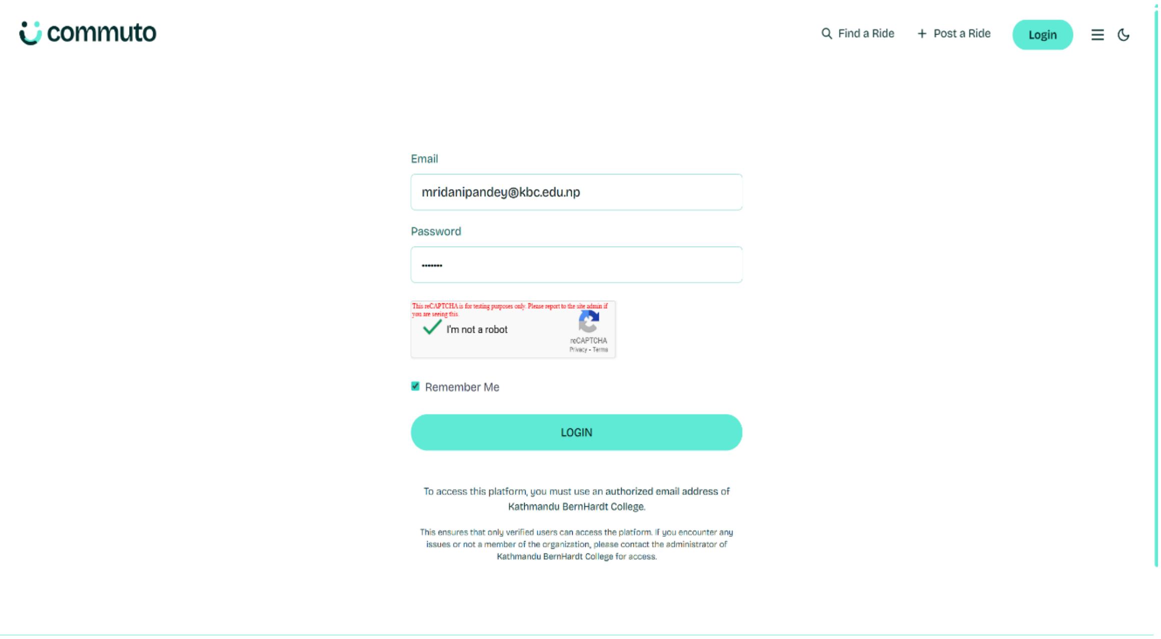
Connecting Co-Workers & Students Sharing The Same Route

We are a community of professionals and students who share the same route to work or school. We help you find a ride or a passenger to share resources, reduce your carbon footprint and step towards a more sustainable future.

Fig 5.3: User Authentication successful

iv. Module 4: Captcha verification

```
const verifyUrl = `https://www.google.com/recaptcha/api/siteverify`;
   let response: any;
   try {
   response = await axios.post(
   verifyUrl,
   new URLSearchParams({
   secret: recaptchaSecret || '',
   response: body.recaptchaToken,
     }),
    );
   } catch {
    this.logger.log({
      level: 'error',
      message: `Login failed for email: ${body.email} - reCAPTCHA verification request failed`,
      tag: 'error',
      email: body.email,
    });
    throw new UnauthorizedException('Failed to verify reCAPTCHA');
    }
```



Email

mridanipandey@kbc.edu.np

Password

••••••••

This reCAPTCHA is for testing purposes only. Please report to the site admin if you are seeing this.

✓ I'm not a robot

☑ Remember Me

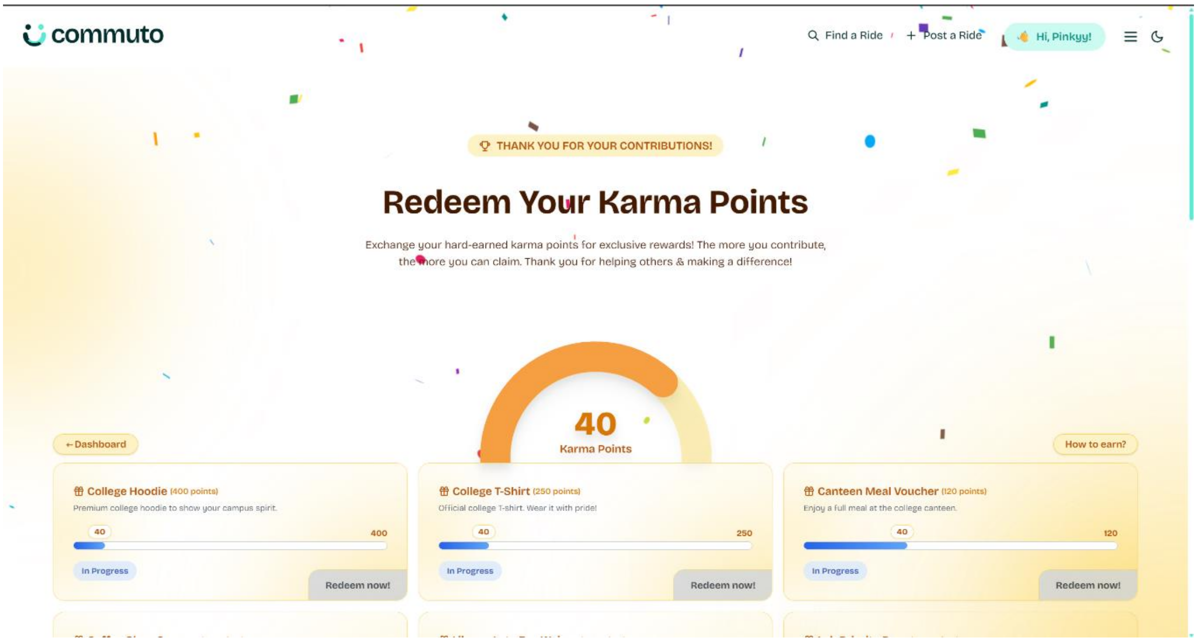
To access this platform, you must use an authorized email address of Kathmandu BernHardt College.

This ensures that only verified users can access the platform. If you encounter any issues or not a member of the organization, please contact the administrator of Kathmandu BernHardt College for access.

Fig 5.4: Captcha verification

Module 5:Karma Points Calculation

```
static getDistanceTierDescription(tier: DISTANCE_TIER): string {
    const thresholds = DISTANCE_THRESHOLDS;
    const threshold: DistanceThreshold = thresholds[tier];
    if (tier === DISTANCE_TIER.VERY_LONG) {
      return `${threshold.min}+ km (${threshold.multiplier}x multiplier)`;
    }
    return `${threshold.min}-${threshold.max} km (${threshold.multiplier}x multiplier)`;
  }
  static getFeedbackRatingDescription(rating: FEEDBACK_EMOJI): string {
    const descriptions: Record<FEEDBACK_EMOJI, string> = {
      [FEEDBACK_EMOJI.SATISFIED]: 'Satisfied',
      [FEEDBACK_EMOJI.NEUTRAL]: 'Neutral',
      [FEEDBACK_EMOJI.DISSATISFIED]: 'Dissatisfied',
    };
    return descriptions[rating] ?? `Unknown rating (${rating})`;
  }
}
```



Redeem Your Karma Points

Exchange your hard-earned karma points for exclusive rewards! The more you contribute, the more you can claim. Thank you for helping others & making a difference!

College Hoodie (400 points)
Premium college hoodie to show your campus spirit.

College T-Shirt (250 points)
Official college T-shirt. Wear it with pride!

Canteen Meal Voucher (120 points)
Enjoy a full meal at the college canteen.

Fig 5.5: Karma Points Calculated

5.2 Testing

Testing is the process of determining whether the system works effectively and efficiently. Testing does not only include debugging. It also checks for quality assurance, validation and verification, reliability, and estimation.

5.2.1 Test Cases for Unit Testing

The modules of the application are tested individually to ensure that each component functions correctly. The following test cases will validate key functionalities of the Commuto application.

- **Unit Testing 1: Authentication Module**

Objective: Ensure that user authentication (login) works as intended, and errors are handled properly.

Table 5.1: Test case for Authentication Module

No.	Test Case Description	Test Data	Expected Result	Actual Result	Test Result
1	Test invalid user authentication	Using an invalid email priyakb@gmail.com Password :Commuto	An error message should display.	An error message is displayed	Pass
2	Test a valid login	Using registered credentials Email:priya@kbc.edu.np Password:Commuto	User should logged in and redirected to the dashboard	User is logged in and redirected to the dashboard	Pass
3	Test invalid login	Using an in correct email or password Email:priya@kbc.edu.np Password: commuto	An error message should be displayed	An error message is displayed	Pass

- **Unit Testing 2: Ride Management Module**

Objective: Ensure that users can create, edit, and delete ride offers without errors.

Table 5.2: Test Case Ride Management Module

No.	Test Case Description	Test Data	Expected Result	Actual Test	Result
1	Test posting a ride	Pickup – Kathmandu BernHardt College Destination-Leapfrog Technology Message – I will be leaving in 5 minutes.	Ride should be posted successfully	Ride is posted successfully	Pass
2	Test cancelling a ride	Clicking Reject button.	The ride should be canceled from the database	The ride is canceled from the database	Pass
3	Test show available rides	2 Riders consecutively. Pickup – Kathmandu BernHardt College Destination-Leapfrog Technology Message – I will be leaving .	Rides should be displayed correctly based on availability	Rides are displayed correctly based on availability	Pass

- **Unit Testing 3: Route Matching Module**

Objective: Ensure that the route matching logic correctly matches non-riders with riders based on the defined route.

Table 5.3: Test Case for Route Matching

No.	Test Case Description	Attribute and Value	Expected Result	Actual Result	Result
1	Test route matching for passengers	Enter passenger's pickup location Pickup–Kathmandu BernHardt College Destination- Leapfrog Technology Message – I will be	Only rides along the same route should be suggested.	Only rides along the same route are suggested	Pass

		leaving .			
2	Test invalid route match	Input passenger's location not matching any ride route Rider:Pickup Kathmandu BernHardt College Destination- Leapfrog Technology Message – I will be leaving . Passenger: Pickup–Kathmandu BernHardt College Destination-New Summit College. Message – I will beleaving .	No rides should be shown.	No rides are shown	Pass

5.2.2 Test Cases for System Testing

System testing evaluates the end-to-end functionality of the integrated application, ensuring that all modules work together as expected.

System Testing 1: Application Starting

Objective: Verify that the application starts without any errors and loads the main

Table 5.4: Test Case Application Start 1

No.	Test Case/Test Script	Expected Result	Observed Result	Result
1	Launch the Commuto application	Application should start without errors,	Application starts successfully;	Pass

		loading the main page	main page loaded	

System Testing 2: Ride Creation Flow

Objective: Ensure that the ride creation process works as expected from user input to database storage.

Table 5.5: Test Case for Ride Creation

No.	Test Case/Test Script	Expected Result	Observed Result	Result
1	User creates a ride	User inputs valid ride details and clicks "Submit"	Ride is added to the database and shown in the user's dashboard	Pass
2	User creates an invalid ride	User inputs invalid details (missing destination)	Error message is displayed prompting to fill in required fields	Pass

System Testing 3: User Dashboard Functionality

Objective*:* Ensure that the user dashboard displays rides and requests correctly, both for riders and non-riders.

Table 5.6: Test Case for User Dashboard

No.	Test Case/Test Script	Expected Result	Observed Result	Result
1	Display the rides	Rider views their listed rides on the dashboard	Rides are displayed correctly with correct information	Pass
2	Display ride requests for non-riders	Non-rider views available rides to request	Rides matching their route are displayed correctly	Pass

System Testing 4: User Interaction with Ride Requests

Objective: Verify that non-riders can request rides and the ride request system works correctly.

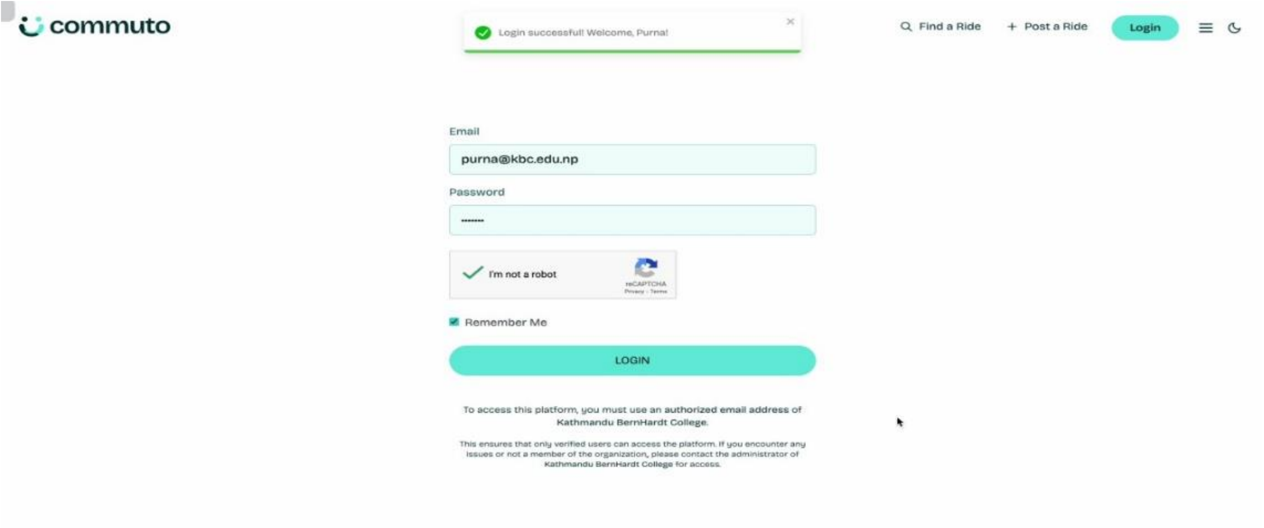
Table 5.7: Test Case Ride Request System

No.	Test Case Description	Expected Result	Observed Result	Result

1	Passenger sends a ride request	Passenger select a ride and sends a request	The ride request is sent and saved in the database	Pass
2	Rider receives and responds to a request	Rider views the ride request and accepts it	Ride request status is updated successfully	Pass

5.3 Result analysis

The development of Commuto has effectively demonstrated the practicality of a community-based ride-sharing system. The implemented features such as ride posting, joining, and mapping worked as intended, validating the requirements set during the design phase. The adoption of tools like React, NestJS and Prisma, ensured that the platform is responsive, scalable, and interactive. From the user perspective, the platform is simple to navigate, with an intuitive interface that minimizes learning effort. The PWA nature makes it adaptable across devices, increasing accessibility. Initial trials indicate that Commuto meets its objectives of reducing individual travel costs, enhancing convenience, and promoting eco-friendly commuting practices. Overall, the project successfully delivers a working prototype of an institutional ride-sharing service, proving its potential to address urban commuting challenges in a sustainable and socially engaging way.



Login successful! Welcome, Purna!

Email: purna@kbc.edu.np

Password: ••••••••

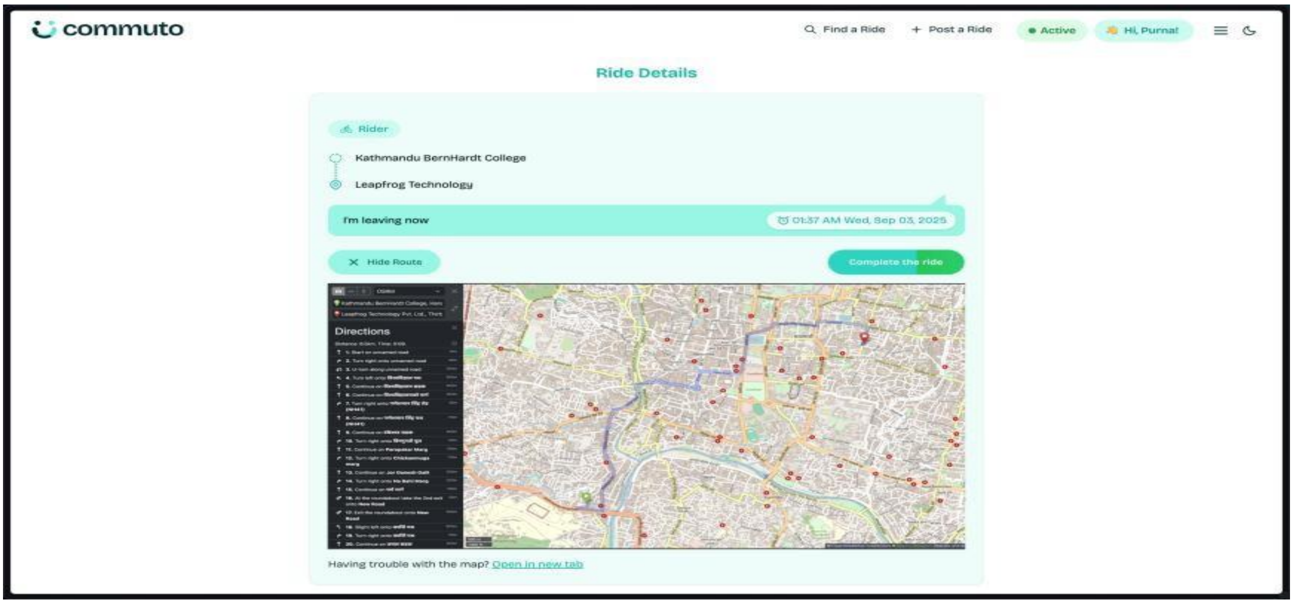
✓ I'm not a robot

☑ Remember Me

To access this platform, you must use an authorized email address of Kathmandu BernHardt College.

This ensures that only verified users can access the platform. If you encounter any issues or not a member of the organization, please contact the administrator of Kathmandu BernHardt College for access.

Fig 5.6: Login Successful



Ride Details

Kathmandu BernHardt College

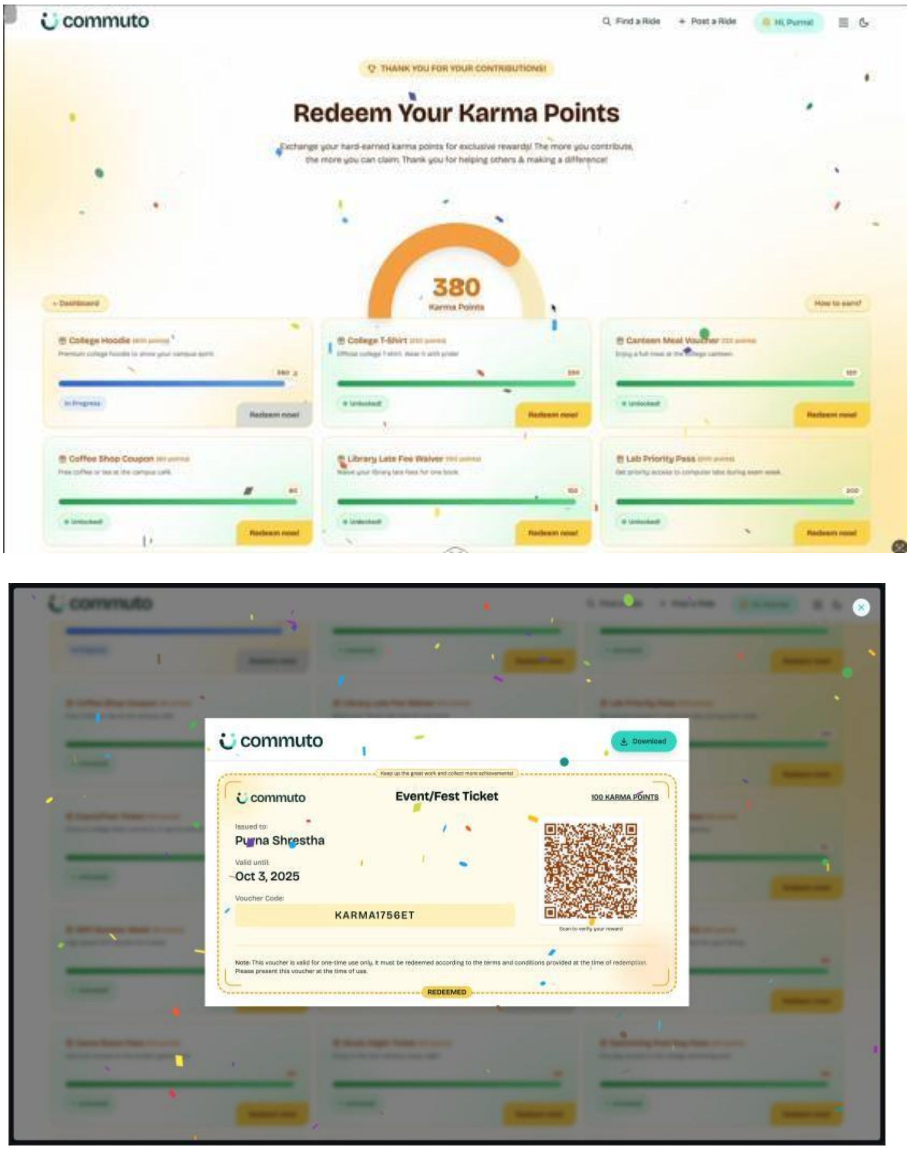
Leapfrog Technology

I'm leaving now

01:37 AM Wed, Sep 03, 2025

Having trouble with the map? Open in new tab

Fig 5.7: Ride Successful



THANK YOU FOR YOUR CONTRIBUTIONS!

Redeem Your Karma Points

Exchange your hard-earned karma points for exclusive rewards! The more you contribute, the more you can claim. Thank you for helping others & making a difference!

380
Karma Points

College Hoodie
Premium college hoodie to show your campus spirit.
In Progress
Redeem now!

College T-Shirt
Official college T-shirt. Wear it with pride!
Unlocked!
Redeem now!

Canteen Meal Voucher
Enjoy a full meal at the college canteen.
Unlocked!
Redeem now!

Coffee Shop Coupon
Free coffee or tea at the campus café.
Unlocked!
Redeem now!

Library Late Fee Waiver
Waive your library late fees for one book.
Unlocked!
Redeem now!

Lab Priority Pass
Get priority access to computer labs during exam week.
Unlocked!
Redeem now!

Keep up the great work and collect more achievements!

commuto

Event/Fest Ticket

100 KARMA POINTS

Issued to:
Purna Shrestha

Valid until:
Oct 3, 2025

Voucher Code:
KARMA1756ET

Scan to verify your reward

Note: This voucher is valid for one-time use only. It must be redeemed according to the terms and conditions provided at the time of redemption. Please present this voucher at the time of use.

REDEEMED

Fig 5.8: Karma Redemption

Chapter 6: Conclusion and Future Recommendation

6.1 Conclusion

The Commuto project successfully addresses a key issue faced by individuals—difficulty commuting due to the lack of personal vehicles and unreliable public transportation. By providing a platform that connects riders and non-riders based on matching routes and destinations (especially around organization-centered areas), the application promotes efficient and sustainable ride-sharing. With features such as user authentication, real-time ride management, and route-based matching, the system offers a convenient, secure, and user-friendly solution for daily commutes. The use of Nest JS as a backend enables real-time data handling and scalability, while the React and TypeScript frontend ensures a smooth and responsive user experience.

After several development cycles, the system underwent thorough testing to ensure reliability, usability, and performance. Providing a smooth and effective user experience remains the central objective, reflecting the broader goals of innovation and convenience in the field of computer science.

6.2 Future Recommendation

There are many scopes available for the improvement of this project. Some of the plans for the future of this application could possibly be:

1. Integrate advanced ride-matching algorithms for better route and time optimization.
2. AI-based ride recommendation for smarter matching and suggestions.
3. Improve accessibility for visibly impaired people, ensuring inclusivity.
4. Integrate Live Location Tracking: Add real-time GPS tracking using map APIs to visualize routes, pickups, and current vehicle locations.
5. Data Analytics Dashboard: Provide analytics for admins or users showing ride frequency, preferred routes, peak hours, etc to help optimize future planning.

REFERENCES:

[1] S. Shaheen and A. Cohen, *Planning for Shared Mobility*, PAS Report No. 583. Chicago, IL: American Planning Association, 2016.

[2] Goodall, W., Fishman, T. D., Bornstein, J., & Bonthron. B,“The rise of mobility as a service”, *Deloitte Review*, vol no. 20, pp. 113–129, 2017.

[3] Finn, K, “Sharing the load: commuting and belonging for staff and students in UK universities*”, Families, Relationships and Societies,* vol. 8, no. 1, pp.161-166, 2019

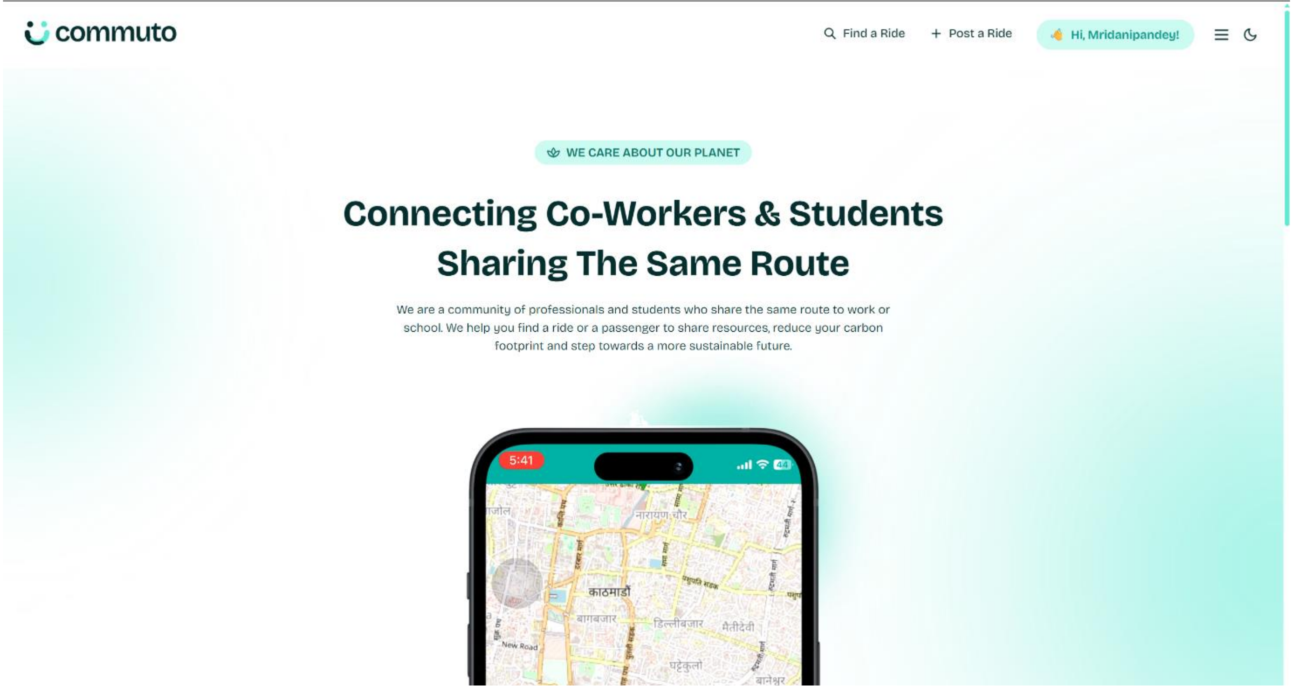
[4] Ceder.Avishai, *Public transit planning and operation:* Modeling, practice and behavior , CRC press, 2016.

[5] Crickard III, Paul, Leaflet. *js essentials*.,Packt Publishing Ltd, 2014.

[6] Wang, Y., Gu, J., Wang, S. and Wang, J., “Understanding consumers’ willingness to use ride-sharing services: The roles of perceived value and perceived risk.”, *Transportation Research Part C: Emerging Technologies*, vol. 105, pp.504-519, 2019.

Appendices

A) Appendices 1: Landing Page



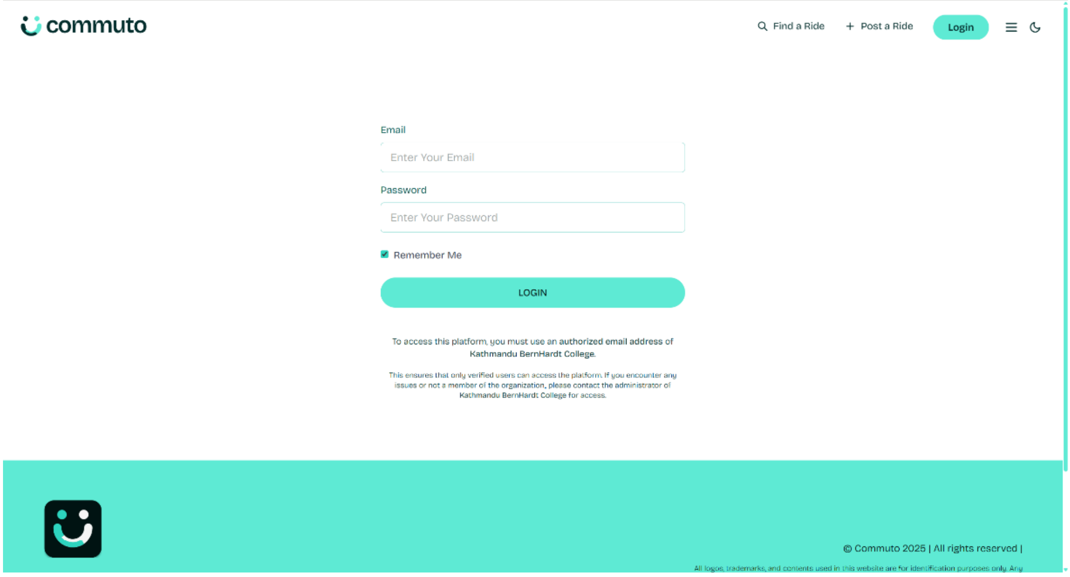
WE CARE ABOUT OUR PLANET

Connecting Co-Workers & Students Sharing The Same Route

We are a community of professionals and students who share the same route to work or school. We help you find a ride or a passenger to share resources, reduce your carbon footprint and step towards a more sustainable future.

Figure 4 Appendices For Landing Page

Appendices 2: Login Page



Email

Enter Your Email

Password

Enter Your Password

☑ Remember Me

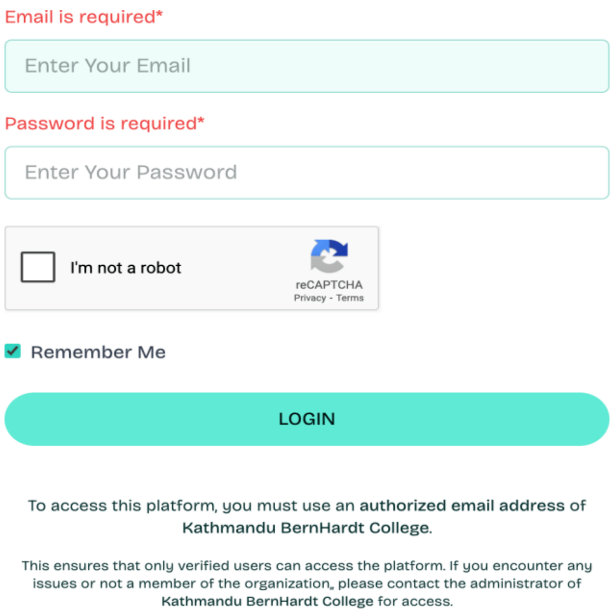
To access this platform, you must use an authorized email address of Kathmandu BernHardt College.

This ensures that only verified users can access the platform. If you encounter any issues or not a member of the organization, please contact the administrator of Kathmandu BernHardt College for access.

© Commuto 2025 | All rights reserved |

All logos, trademarks, and contents used in this website are for identification purposes only. Any

Fig: Login page



Email is required*

Enter Your Email

Password is required*

Enter Your Password

□ I'm not a robot

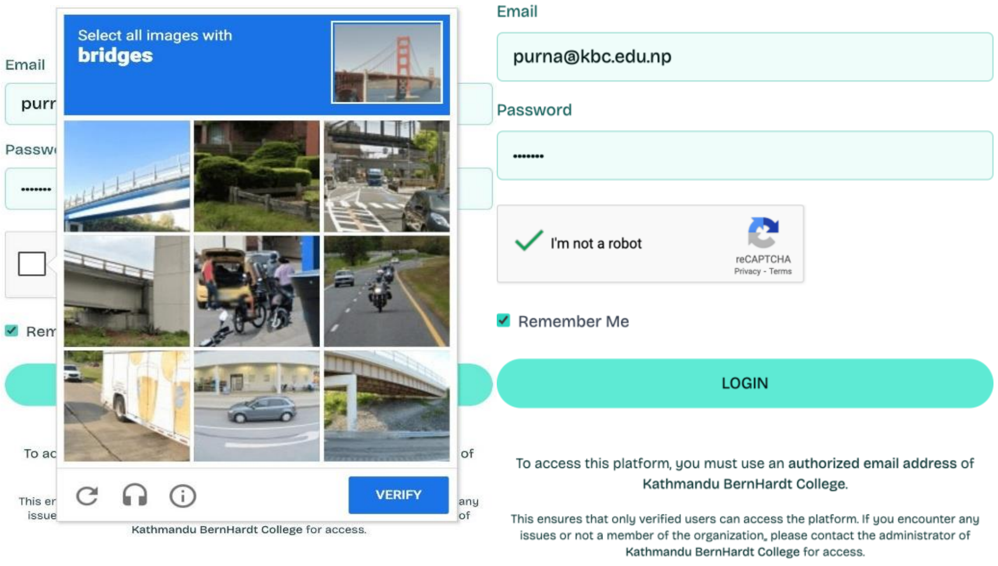
☑ Remember Me

LOGIN

To access this platform, you must use an **authorized email address** of **Kathmandu BernHardt College.**

This ensures that only verified users can access the platform. If you encounter any issues or not a member of the organization, please contact the administrator of **Kathmandu BernHardt College** for access.

Fig: Login page validation



Select all images with **bridges**

Email

purna@kbc.edu.np

Password

•••••••

✓ I'm not a robot

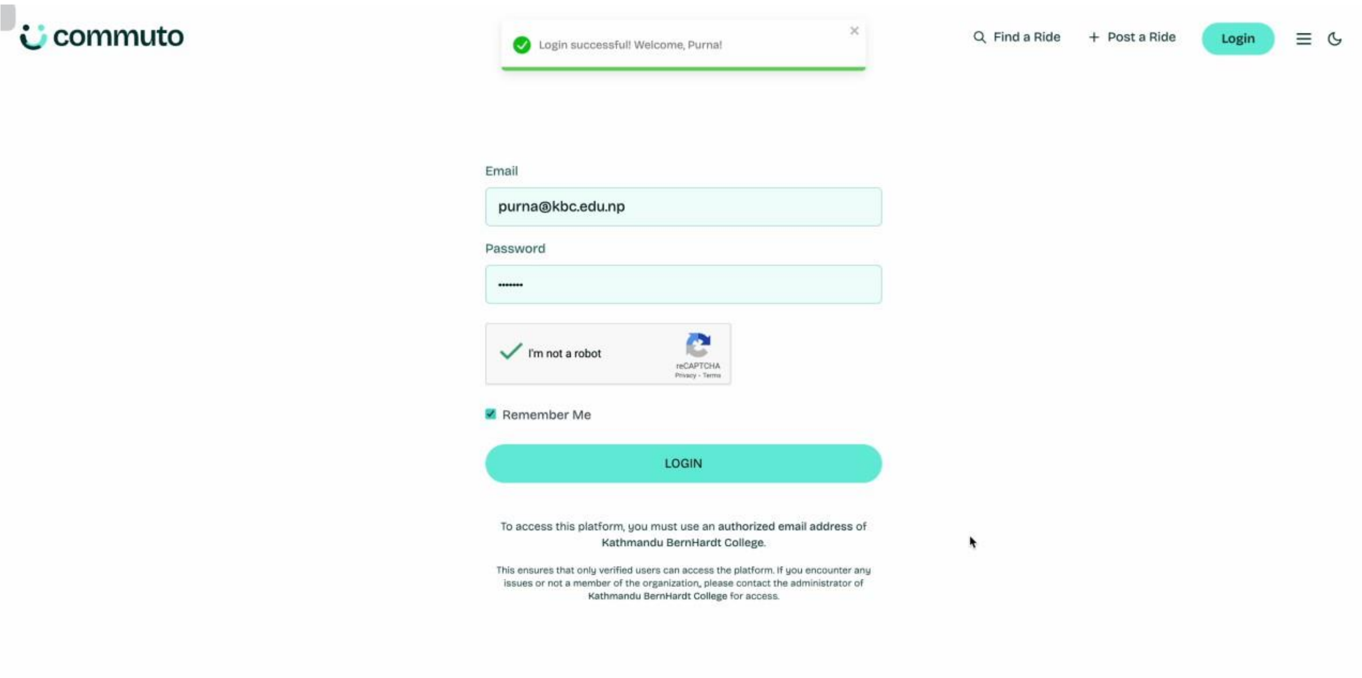
☑ Remember Me

LOGIN

To access this platform, you must use an **authorized email address** of **Kathmandu BernHardt College.**

This ensures that only verified users can access the platform. If you encounter any issues or not a member of the organization, please contact the administrator of **Kathmandu BernHardt College** for access.

Fig: Captcha Verification



Login successful! Welcome, Purna!

Email

purna@kbc.edu.np

Password

•••••••

☑ I'm not a robot

☑ Remember Me

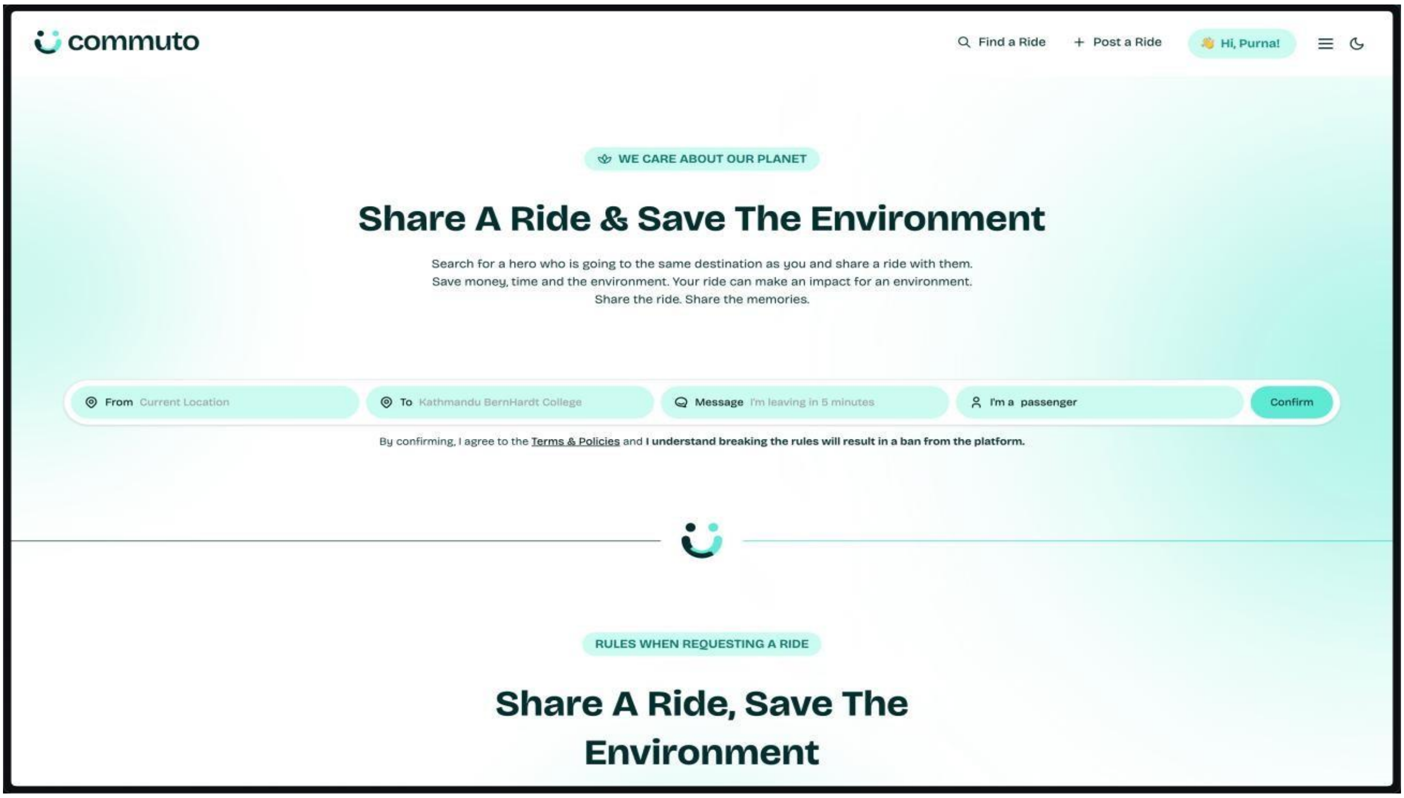
LOGIN

To access this platform, you must use an authorized email address of Kathmandu BernHardt College.

This ensures that only verified users can access the platform. If you encounter any issues or not a member of the organization, please contact the administrator of Kathmandu BernHardt College for access.

Fig: Login Successful.

B) Appendices For Ride posting



WE CARE ABOUT OUR PLANET

Share A Ride & Save The Environment

Search for a hero who is going to the same destination as you and share a ride with them. Save money, time and the environment. Your ride can make an impact for an environment. Share the ride. Share the memories.

From Current Location

To Kathmandu BernHardt College

Message I'm leaving in 5 minutes

I'm a passenger

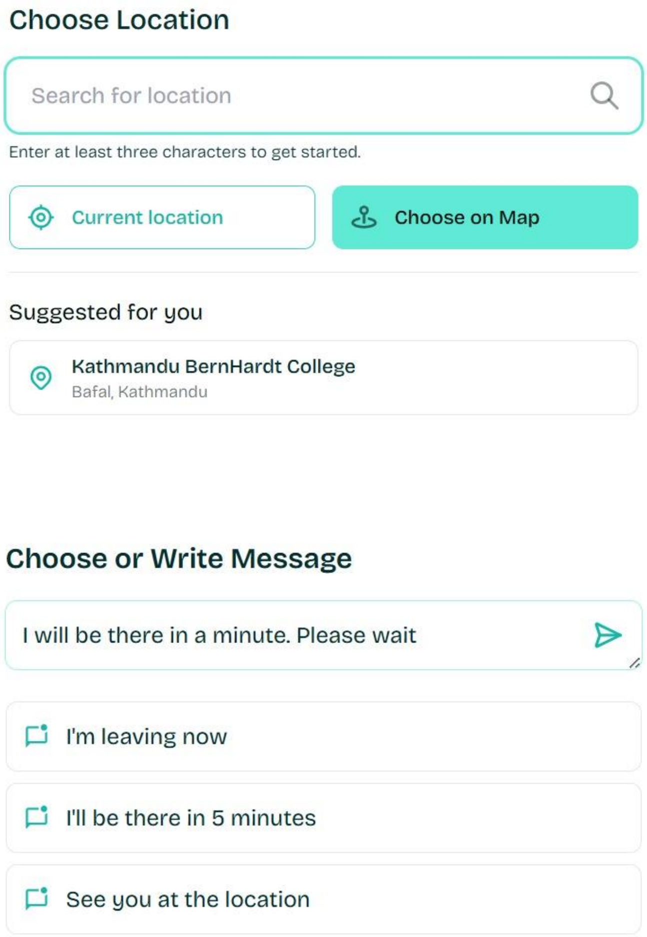
Confirm

By confirming, I agree to the Terms & Policies and I understand breaking the rules will result in a ban from the platform.

RULES WHEN REQUESTING A RIDE

Share A Ride, Save The Environment

Figure 5 Appendices 3 Ride Posting



Choose Location

Search for location

Enter at least three characters to get started.

Suggested for you

Kathmandu BernHardt College
Bafal, Kathmandu

Choose or Write Message

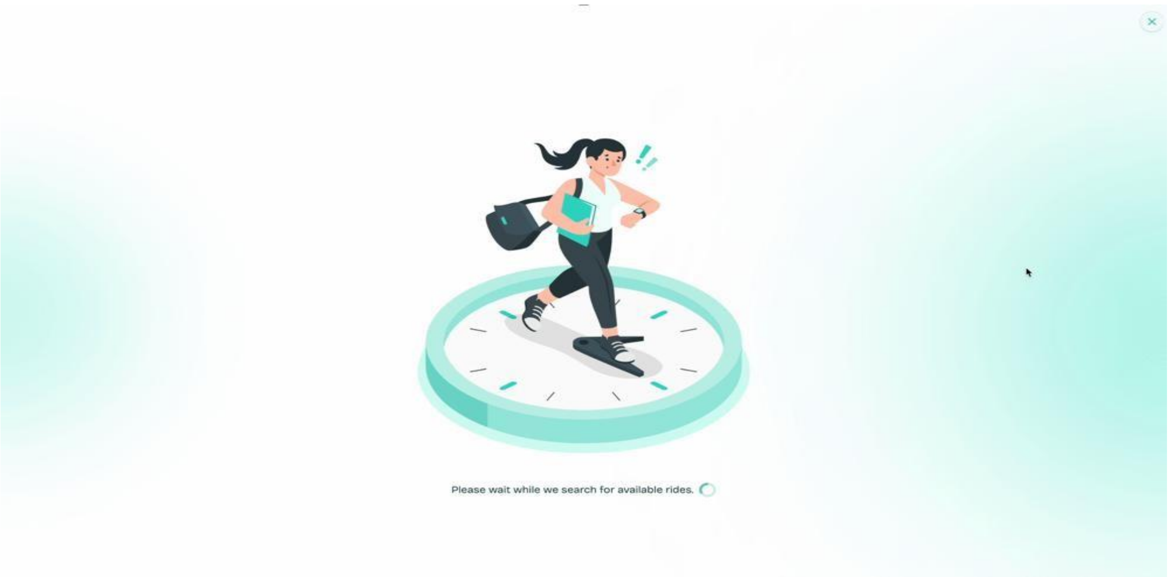
I will be there in a minute. Please wait

I'm leaving now

I'll be there in 5 minutes

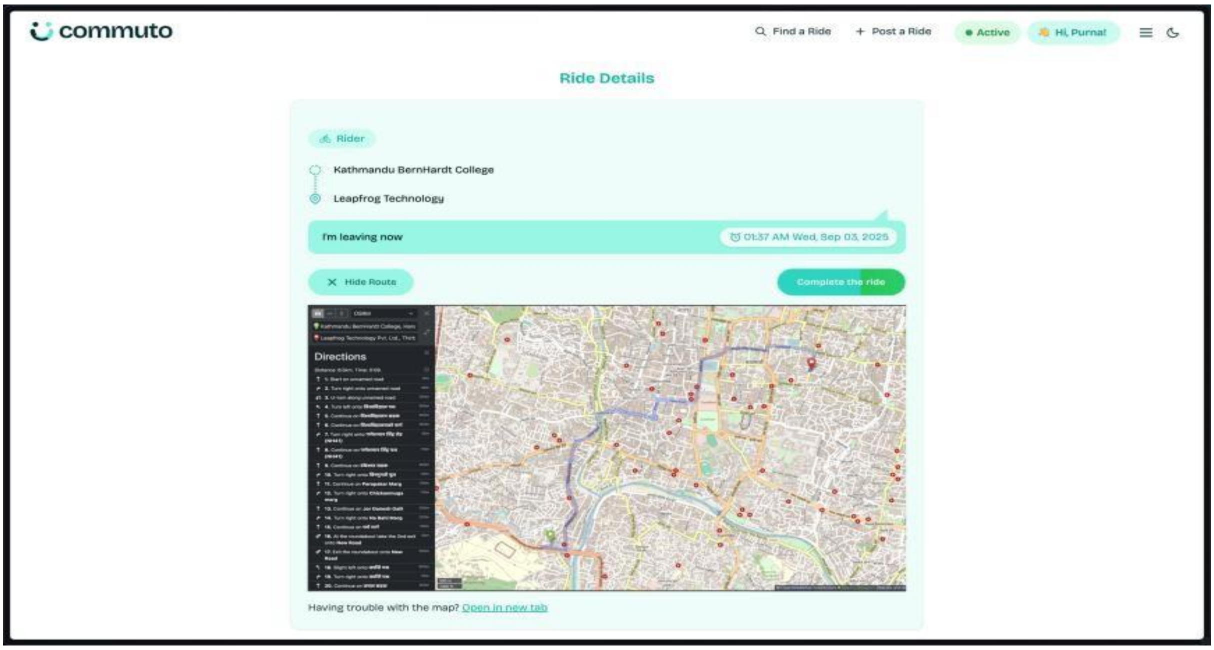
See you at the location

Fig: Ride Request and Message



Please wait while we search for available rides.

Fig: Waiting for Rides



Ride Details

Kathmandu BernHardt College

Leapfrog Technology

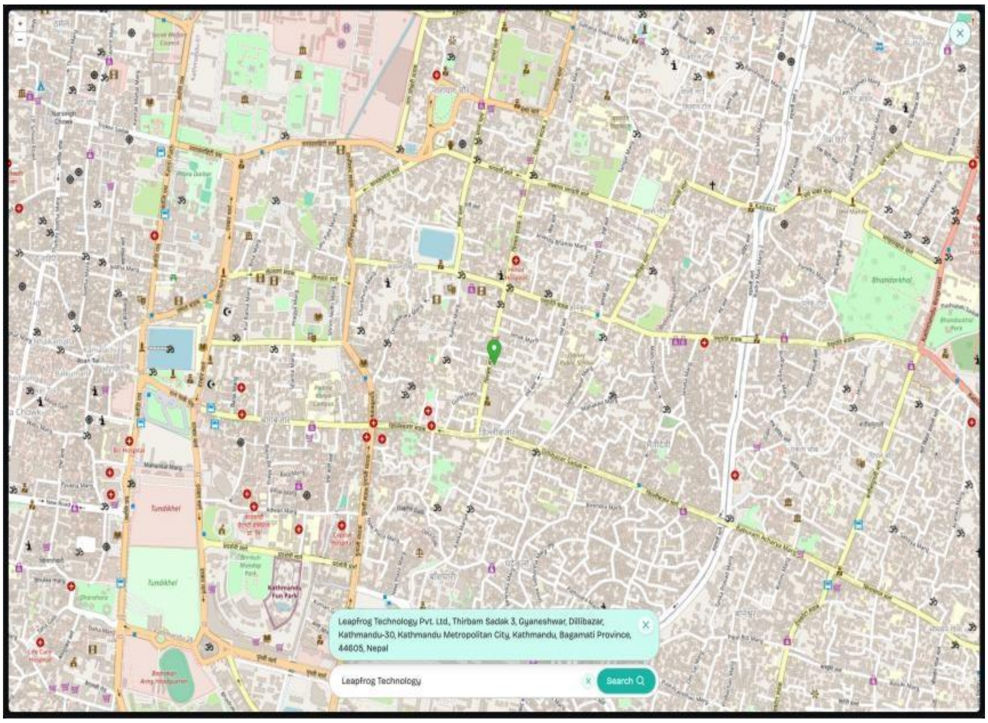
I'm leaving now

01:37 AM Wed, Sep 03, 2025

✕ Hide Route
Complete the ride
Kathmandu BernHardt College, Hars
Leapfrog Technology Pvt. Ltd., Thirb
Directions

1. Start on unnamed road
2. Turn right onto unnamed road
3. U-turn along unnamed road
4. Turn left onto [illegible]
5. Continue on [illegible]
6. Continue on [illegible]
7. Turn right onto [illegible]
8. Continue on [illegible]
9. Continue on [illegible]
10. Turn right onto [illegible]
11. Continue on [illegible] Marg
12. Turn right onto Chickamuga marg
13. Continue on [illegible]
14. Turn right onto [illegible]
15. Continue on [illegible]
16. At the roundabout take the 2nd exit onto New Road
17. Exit the roundabout onto New Road
18. Slight left onto [illegible]
19. Turn right onto [illegible]
20. Continue on [illegible]

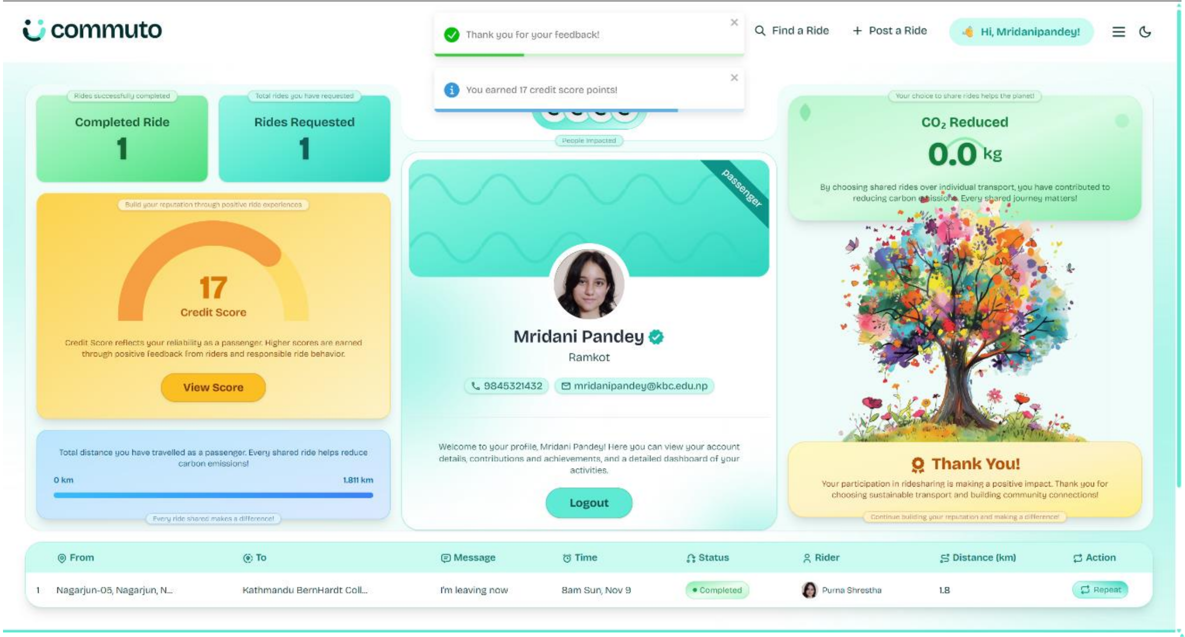
Having trouble with the map? Open in new tab



Leapfrog Technology Pvt. Ltd., Thirbam Sadak 3, Gyaneshwar, Dillibazar, Kathmandu-30, Kathmandu Metropolitan City, Kathmandu, Bagamati Province, 44605, Nepal

Fig: The map shows the route

C) Appendices showing karma Redemption Process


commuto
Find a Ride
+ Post a Ride
Hi, Mridanipandey!
Rides successfully completed
Total rides you have requested
Build your reputation through positive ride experiences
View Score
0 km
1.811 km
Every ride shared makes a difference!
People impacted
passenger
Logout
Your choice to share rides helps the planet!
Continue building your reputation and making a difference!

Thank you for your feedback!

You earned 17 credit score points!

Completed Ride
1

Rides Requested
1

17
Credit Score

Credit Score reflects your reliability as a passenger. Higher scores are earned through positive feedback from riders and responsible ride behavior.

Total distance you have travelled as a passenger. Every shared ride helps reduce carbon emissions!

Mridani Pandey
Ramkot
9845321432
mridanipandey@kbc.edu.np

Welcome to your profile, Mridani Pandey! Here you can view your account details, contributions and achievements, and a detailed dashboard of your activities.

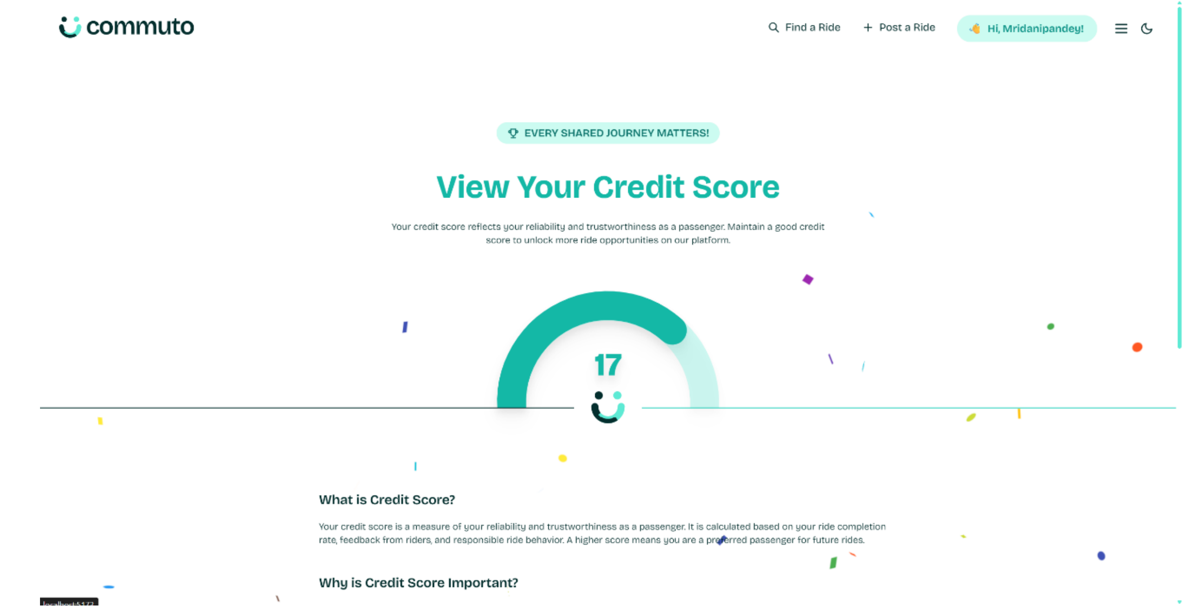
CO_2 Reduced
0.0 kg

By choosing shared rides over individual transport, you have contributed to reducing carbon emissions. Every shared journey matters!

Thank You!

Your participation in ridesharing is making a positive impact. Thank you for choosing sustainable transport and building community connections!

	From	To	Message	Time	Status	Rider	Distance (km)	Action
1	Nagarjun-05, Nagarjun, N...	Kathmandu BernHardt Coll...	I'm leaving now	8am Sun, Nov 9	● Completed	Purna Shrestha	1.8	Repeat



EVERY SHARED JOURNEY MATTERS!

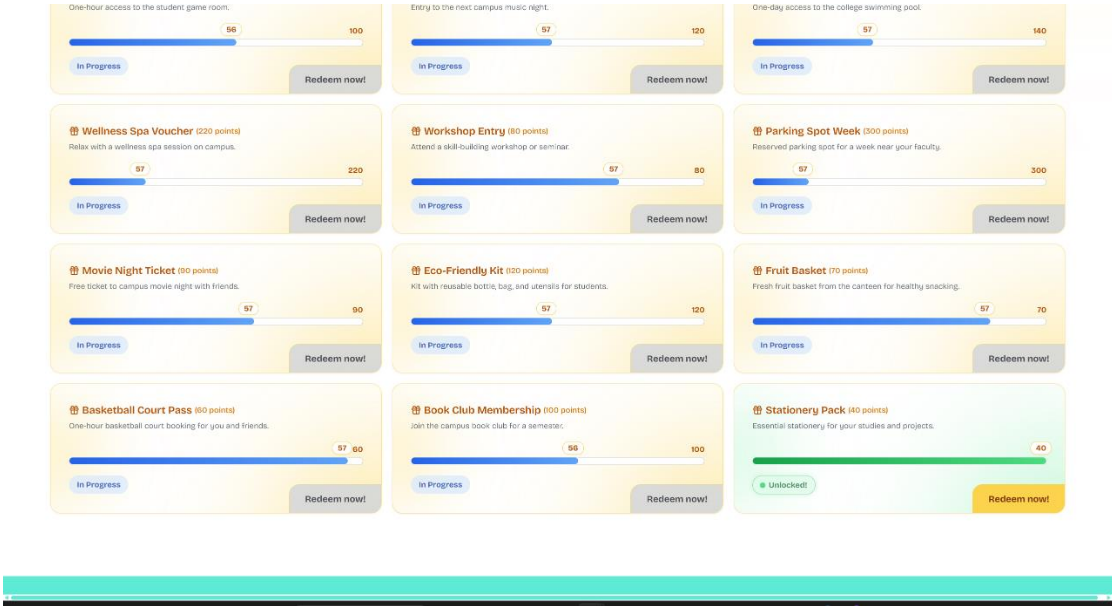
View Your Credit Score

Your credit score reflects your reliability and trustworthiness as a passenger. Maintain a good credit score to unlock more ride opportunities on our platform.

What is Credit Score?

Your credit score is a measure of your reliability and trustworthiness as a passenger. It is calculated based on your ride completion rate, feedback from riders, and responsible ride behavior. A higher score means you are a preferred passenger for future rides.

Why is Credit Score Important?



One-hour access to the student game room.

56 100

In Progress

Redeem now!

Entry to the next campus music night.

57 120

In Progress

Redeem now!

One-day access to the college swimming pool.

57 140

In Progress

Redeem now!

🎁 Wellness Spa Voucher (220 points)

Relax with a wellness spa session on campus.

57 220

In Progress

Redeem now!

🎁 Workshop Entry (80 points)

Attend a skill-building workshop or seminar.

57 80

In Progress

Redeem now!

🎁 Parking Spot Week (300 points)

Reserved parking spot for a week near your faculty.

57 300

In Progress

Redeem now!

🎁 Movie Night Ticket (90 points)

Free ticket to campus movie night with friends.

57 90

In Progress

Redeem now!

🎁 Eco-Friendly Kit (120 points)

Kit with reusable bottle, bag, and utensils for students.

57 120

In Progress

Redeem now!

🎁 Fruit Basket (70 points)

Fresh fruit basket from the canteen for healthy snacking.

57 70

In Progress

Redeem now!

🎁 Basketball Court Pass (60 points)

One-hour basketball court booking for you and friends.

57 60

In Progress

Redeem now!

🎁 Book Club Membership (100 points)

Join the campus book club for a semester.

56 100

In Progress

Redeem now!

🎁 Stationery Pack (40 points)

Essential stationery for your studies and projects.

40

● Unlocked!

Redeem now!

Figure 6 Karma Points Redemption