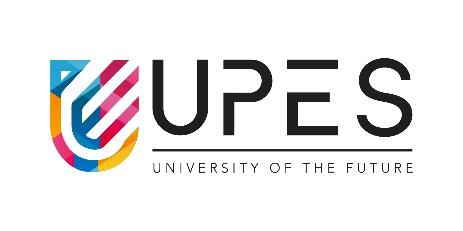
# MINOR PROJECT-1

**SYNOPSIS REPORT**

**ON**

## Trekker On the Go

|  | **Submitted By** |  |
| --- | --- | --- |
| Vyom Singhal | R2142220212 500105031 | FSAI(Hons) |
| Divya Darshan Tiwari | R2142220070 500105299    ***Under the guidance of***  **Dr. Rohit Tanwar**  Senior Associate Professor  Department of Computer Science | FSAI(Hons) |

**School of Computer Science** 

# UNIVERSITY OF PETROLEUM AND ENERGY STUDIES

**Dehradun-248007**

**Aug-Nov, 2024**

**INDEX**

| **Content** | **Page No** |
| --- | --- |
| Abstract | 3 |
| Introduction | 4 |
| Literature | 5 |
| Problem Statement | 6 |
| Objectives | 7 |
| Methodology | 7 |
| System Requirements | 11 |
| PERT Chart | 12 |
| SWOT | 13 |
| References | 13 |

# 1.Abstract

Students who don’t have their personal vehicles do travel most of the time through Trekkers, but they also face a lot of problems while traveling through them. The problems are like waiting for the trekker to get filled so that the driver starts the journey, if there is no trekker currently at the stand then waiting for one. During the vacation time, 80% of the students go home and they rely on trekkers to reach the city which causes an increase in the number of students but not enough trekkers.

Also, after 6 p.m. the frequency of the trekkers drops significantly, and any student who has to get a trekker after 6 p.m., it is completely his/her fortune whether he/she will be getting a trekker.

*Trekker On the Go* is designed to be an optimal solution to optimize the share auto (known as ‘Trekkers’) which are the major form of transportation from UPES to Dehradun City. It will be a system to monitor the seat availability, trekker departures providing students the information beforehand so they can arrange any alternative if trekker isn’t available.

*Key Features*: driver and passenger interface, real-time updates, seat-availability, departure time

# 2.Introduction

In today's fast-paced world, efficient transportation systems are essential, especially for students commuting between their university and nearby cities. The lack of a structured system for managing trekker services has led to significant inconveniences, with students often facing long wait times without information on when the next trekker will depart or if seats are available, leading to unnecessary stress and delays.

**Trekker On the Go** is designed to address these challenges by providing a comprehensive solution that delivers real-time updates on trekker availability and departure schedules. This system will feature two user-friendly interfaces: one for students (users) and another for trekker drivers (service providers). The primary goal is to ensure simplicity and intuitiveness in the interface, minimizing the learning curve so that drivers can quickly and easily adapt to the software.

**Drawing inspiration** from the straightforward tracking systems used by platforms like **Domino's and Zomato**, which provide real-time updates on the status of food orders, Trekker On the Go will offer students similar real-time insights. Students will be able to see whether any trekkers are en route, have arrived at the college, or are ready to depart, making their commute more predictable and stress-free.

To ensure that **Trekker On the Go** remains accessible and practical for all users, we have strategically decided to avoid using GPS and other IoT-based technologies. While these technologies offer advanced tracking capabilities, they also come with challenges, including the need for additional equipment, potential privacy concerns, and costs that might not be feasible for all drivers.

Instead, **Trekker On the Go** will employ a streamlined, manual approach. All updates regarding seat availability, vehicle departure, and arrival times will be managed through simple button clicks. This approach ensures that the system remains cost-effective, easy to use, and adaptable to the needs of both students and drivers.

By allowing drivers to manually update the status of their vehicles—such as indicating when seats are available or when they have reached a specific point—the system empowers drivers to control the information shared without requiring invasive technology. Simultaneously, students receive timely updates, enabling them to plan their commutes effectively without needing complex tracking equipment.

This approach aligns with the core values of **Trekker On the Go: simplicity, accessibility, and user empowerment**. By focusing on manual updates, we create a solution that respects drivers' preferences, keeps costs low, and still provides the essential functionality that students need for a reliable and efficient transportation experience.

This basic yet effective tracking system will not only enhance the commuting experience for students but also foster better communication and coordination between students and drivers. Trekker On the Go aims to establish a reliable, efficient, and stress-free transportation solution that meets the needs of both students and drivers. Ultimately, this approach will streamline the transportation process and improve the overall efficiency of trekker services, making daily commutes smoother and more predictable for everyone involved.

# 3.Literature

In recent years, several innovative systems have been developed to enhance public transportation services, particularly in urban areas. Apps like ZingBus and RedBus have revolutionized the way users plan their commutes by providing real-time updates on bus arrivals and seat availability. However, these solutions are typically designed for large-scale public transportation networks and do not cater to the unique needs of smaller, community-based services like shared trekkers.

Trekker On the Go is designed specifically to address the transportation challenges faced by students at our university. By offering a platform that connects students with available trekker drivers, our system aims to minimize wait times and provide greater flexibility to students. Unlike large-scale solutions, Trekker On the Go is tailored to the specific dynamics of our campus and surrounding areas.

Existing research, such as the study on Dynamic Matching for Real-Time Ride Sharing[1], has explored matching policies based on continuous linear programming to account for the varying arrival rates of customers and drivers. This approach, while effective for large urban environments, highlights a common problem that Trekker On the Go will address on a smaller scale. Our solution will utilize a two-interface system—one for students and one for drivers—to manage the supply and demand of trekkers, especially during peak times.

To handle the fluctuating number of students and trekkers, we will implement a queue-based system, recognizing this as a classic random queuing problem. Previous studies have addressed the complexity of queuing at transportation hubs by developing models that simulate dynamic queuing behaviors of passengers and taxis[2]. These models take into account random factors such as passenger numbers and varying arrival or departure times, providing valuable insights into improving transport hub operations. Our project will leverage similar concepts to enhance the efficiency of trekker services, making it easier for students to plan their travel.

In contrast to IoT-based solutions, which have been used in large-scale public transport systems to provide real-time operational data, our project will avoid the high costs associated with hardware like sensors. For instance, the IoT-based Bus Seating Technology System[3], designed to dynamically monitor vehicle occupancy through an array of sensors, is a robust solution for city-wide bus systems but is impractical for trekker services due to cost constraints. Trekker On the Go focuses on a more cost-effective, software-driven approach, making it accessible and affordable for both students and trekker drivers.

By building on these existing concepts and tailoring them to the specific needs of our campus community, Trekker On the Go aims to significantly improve the reliability and efficiency of trekker services, ultimately enhancing the daily commute experience for students.

# 4.Problem Statement

The university’s student struggles with unreliable and disorganized shared transportation services, specifically with trekkers operating between the campus and cities. The primary issues include a lack of updates on seat availability, unclear departure times, and no efficient way for students to plan their travel. These inefficiencies result in wasted time, frustration, and often missed opportunities for timely commutes, especially affecting freshmen and those new to the area.

**5.Objective**

1. Develop a user interface
2. Implement real-time data management system
3. Create driver and passenger interface
4. Enhance travel efficiency

# 6.Methodology

**Phase 1: Planning and Requirement Analysis**

Objectives:

* Gather and document requirements, including user needs and system specifications.
* Define clear project goals and break down the project into manageable tasks.

Activities:

* Collaborate with students to understand and document the requirements.
* Create a detailed project plan, including timelines, milestones, and resource allocation.

Tools:

* Project Management: Usage of Jira for task tracking and collaboration.
* Documentation: Google Docs for requirement documentation

**Phase 2: System Design**

Objectives:

* Design the architecture of the system, focusing on the interaction between different components (frontend, backend, database and data structures).
* Ensure that the system's design supports scalability, performance, and ease of use.

Activities:

* Create wireframes and mockups for the user interface to visualize the user experience.
* Design the database schema to effectively store and manage trekker and seat availability data.
* Define the API endpoints and their interactions between frontend and backend.

Tools:

* Database Design: MongoDB for database schema design and data storage.

**Phase 3: Backend Development**

Objectives:

* Build a robust backend system that handles data processing, real-time updates, and API management.

Activities:

* Develop RESTful APIs using Node.js and Express to manage the communication between the frontend and backend.
* Integrate MongoDB to store and retrieve data on trekker availability, ensuring efficient data management.
* Implement real-time features using WebSocket, enabling the server to push data to the client without the client needing to initiate a request.

Tools:

* Backend Development: Node.js and Express for server-side development.
* Database Management: MongoDB for data storage and retrieval.
* API Testing: Postman for testing and validating APIs.

**Phase 4: Frontend Development**

Objectives:

* Develop a responsive and user-friendly interface that meets the design specifications.

Activities:

* Implement the user interface using ReactJS, focusing on creating dynamic and reusable components.
* Style the application using CSS and SCSS, ensuring consistency and responsiveness across devices.
* Ensure accessibility and usability by adhering to best practices in frontend development.

Tools:

* Development: ReactJS for frontend development, Visual Studio Code as the development environment.
* Version Control: Git for version control and collaboration.
* Styling: CSS/SCSS for styling and animations.

#### **Phase 5: Integration and Testing**

Objectives:

* Ensure that all components work together seamlessly and that the system meets performance and reliability standards.
* Checking for any bug in the software , removal of the bug and testing again.

Activities:

* Integrate the frontend and backend, ensuring smooth data exchange and real-time updates.
* Testing of the APIs using postman and Websockets.
* Conducting test for the features that have been integrated within the software.
* Conduct comprehensive testing, including unit tests, integration tests.
* Perform stress and load testing to ensure the system can handle peak traffic as well as slow network.

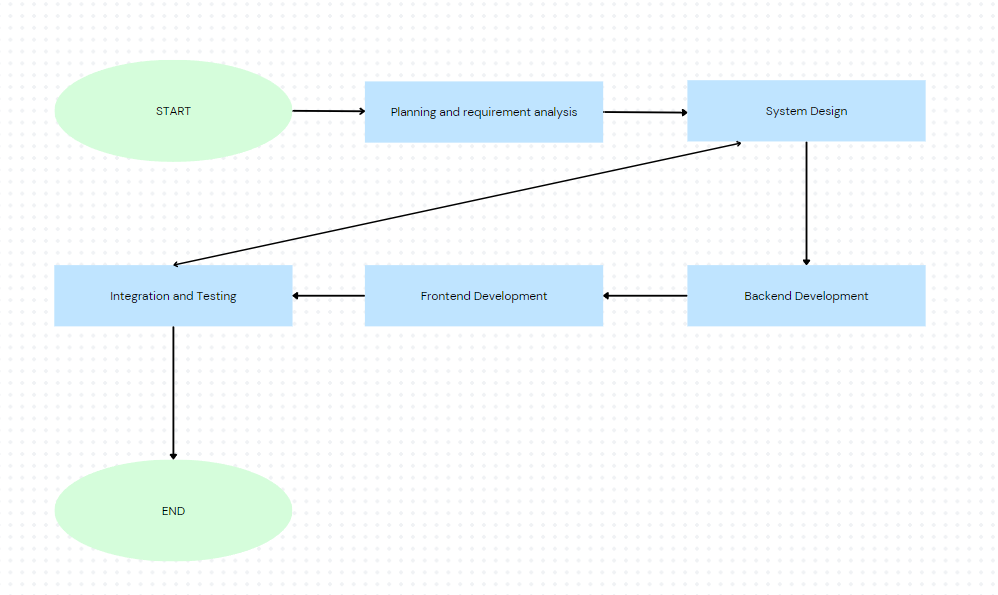


Figure.1: Methodology FlowChart

# 7.System Requirement

## 1. Software Requirements

Operating System:

* Windows 10/8/7 (32-bit or 64-bit)

Software:

* Text Editor/IDE: Visual Studio Code, or any other preferred text editor or IDE
* Node.js: Version 18.x or later (for running the server)
* React: Version 18.x or later (for the frontend development)
* MongoDB: Version 6.x or later (for the database)
* Web Browser: Google Chrome, Mozilla Firefox, or similar (for testing the application)

Compiler/Interpreter:

* Node.js (for executing JavaScript on the server-side)
* NPM (Node Package Manager)

## 2. Hardware Requirements

Processor : Dual Core 2.7 GHz or better

RAM : 4GB or higher

Disk Space : 1GB or more

**8.PERT Chart**

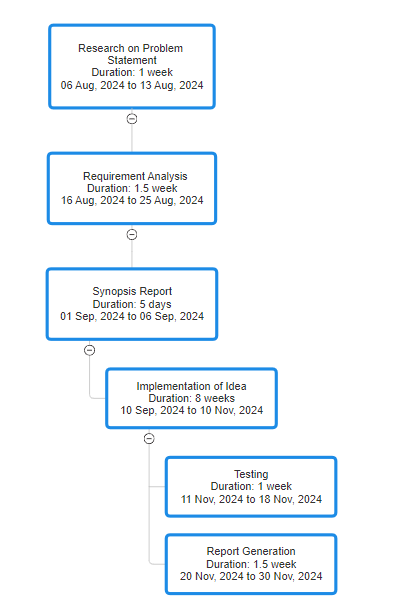


Figure.2: PERT Chart

# 9. SWOT Analysis

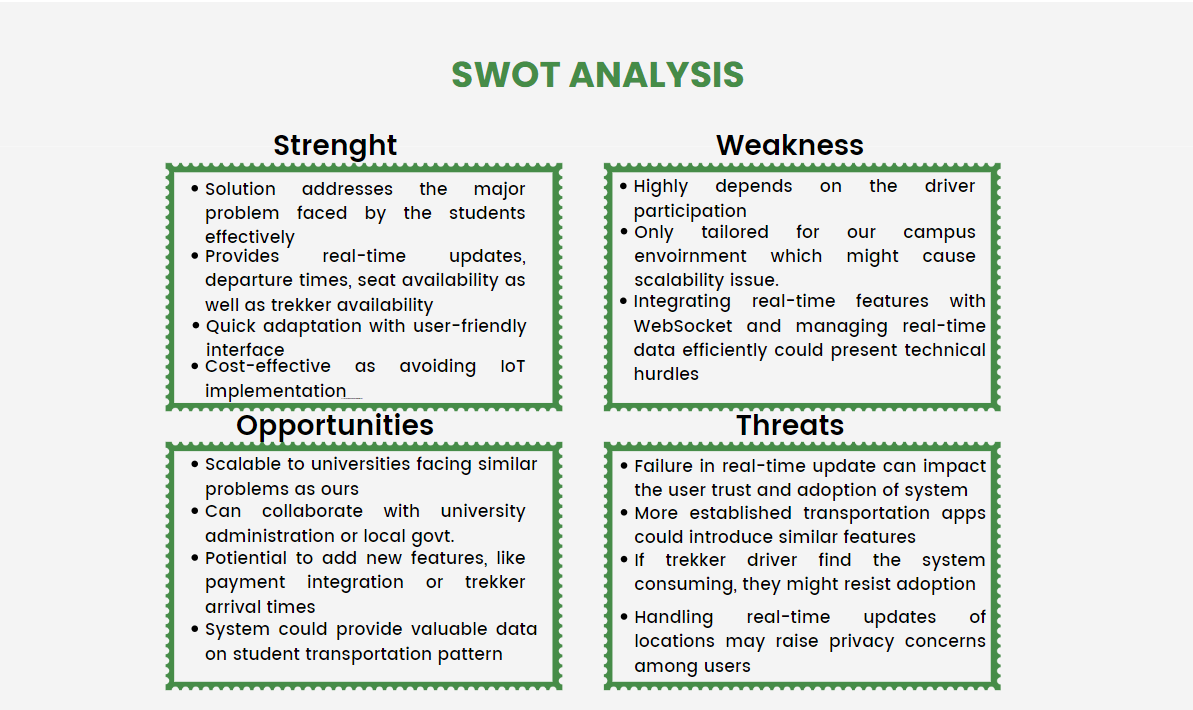


Figure.3: SWOT Analysis

# 10.References

[1][Erhun Özkan, Amy R. Ward (2020) Dynamic Matching for Real-Time Ride Sharing. Stochastic Systems 10(1):29-70.](https://doi.org/10.1287/stsy.2019.0037)

[2][Qiaoli Yang, Bo Yang, Zheng Qiao, Min-an Tang, Fengyang Gao,Impact of possible random factors on queue behaviors of passengers and taxis at taxi stand of transport hubs,Volume 580,2021](https://doi.org/10.1016/j.physa.2021.126131)

[3][Murdan, A.P., Bucktowar, V., Oree, V. and Enoch, M.P. (2020), Low-cost bus seating information technology system. IET Intell. Transp. Syst., 14: 1303-1310.](https://doi.org/10.1049/iet-its.2019.0529)