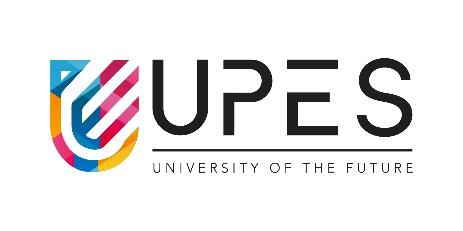
# MINOR PROJECT-1

**SYNOPSIS REPORT**

**ON**

## Trekker On the Go

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

**School of Computer Science** 

# UPES

**Dehradun-248007**

**Aug-Nov, 2024**

**Table of Contents**

1. Introduction

1.1 Purpose of the Project

1.2 Target Beneficiary

1.3 Project Scope

1.4 References

2. Project Description

2.1 Reference Algorithm

2.2 Data and Data Structure

2.3 SWOT Analysis

2.4 Project Features

2.5 User Classes and Characteristics

2.6 Design and Implementation Constraints

2.7 Use Case Diagram

2.8 Assumptions and Dependencies

3. System Requirements

3.1 User Interface

3.2 Software Interface

3.3 Database Interface

3.4 Protocols

4. Non-functional Requirements

4.1 Performance Requirements

4.2 Security Requirements

4.3 Software Quality Attributes

5. Other Requirements

6. Future Prospects

7. Appendix A

8. Appendix B

9. Appendix C

**1. Introduction**

1.1 Purpose of the Project

"Trekker on the Go" aims to streamline the shared transportation experience for university students relying on trekkers (shared auto-rickshaws) for commuting between campus and nearby cities. The project addresses inefficiencies such as unclear seat availability and departure times, providing real-time updates to both drivers and passengers. By enhancing communication between the two, the project seeks to reduce wait times, improve travel planning, and foster a more organized transportation system.

1.2 Target Beneficiary

The primary beneficiaries of "Trekker on the Go" are university students who depend on trekkers for their daily commutes. Secondary beneficiaries include trekker drivers who will benefit from improved passenger management and communication regarding seat availability.

1.3 Project Scope

The project will deliver a user-friendly application that allows:

* Passengers to check real-time seat availability and departure schedules.
* Drivers to update the availability of seats and manage their departure schedules.
* A notification system to inform users about trekker availability.

The deliverables will include:

* A fully functioning mobile-friendly web application.
* User interface design for passengers and drivers.
* Back-end development to support real-time updates and data management.
* Comprehensive testing to ensure reliability and security.

1.4 References

Transportation Management Systems (TMS) – Overview of TMS software features.

User Experience Design Principles – Best practices for designing user interfaces.

**2. Project Description**

2.1 Reference Algorithm

The core logic of the system will include:

* User authentication (registration, login, logout).
* Real-time updates for seat availability and departure times.
* Data management for user preferences and trip history.

2.2 Data and Data Structure

The application will collect the following data:

* User information (student ID, name, contact information).
* Trekker information (driver details, seat availability, and departure times).
* Historical data on user trips.
* Data will be stored in a structured MongoDB database, utilizing user IDs and trekker IDs as primary keys to ensure quick retrieval and efficient data management.

2.3 SWOT Analysis

**Strengths:**

* Real-time updates enhance user experience.
* Simplified interface for ease of use.
* Direct communication between drivers and passengers.

**Weaknesses:**

* Dependence on internet connectivity for real-time updates.
* Limited features at launch; potential for future expansions.

**Opportunities:**

* Potential integration with other transport systems.
* Expansion to serve other regions or cities.

**Threats:**

* Competition from existing transportation apps.
* User concerns regarding data privacy.

2.4 Project Features

* Driver Interface: A dashboard for drivers to update seat availability and departure times.
* Passenger Interface: A simple UI for checking trekker availability and scheduling rides.
* Real-time Notifications: Alerts to notify users about changes in trekker status.
* History Log: Records of past trips for users to reference.

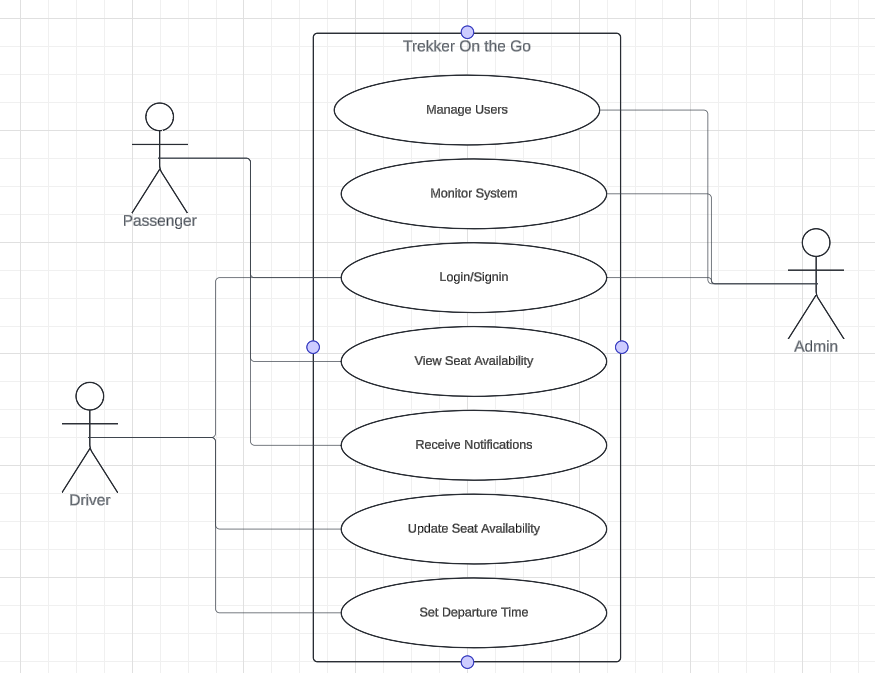
2.5 User Classes and Characteristics

* Students: Primary users looking for efficient transportation options.
* Drivers: Users managing the trekker services, responsible for updating seat availability.
* Admin: Users overseeing the system, ensuring its smooth operation and managing user accounts.

2.6 Design and Implementation Constraints

* Hardware Constraints: Must be compatible with smartphones and tablets.
* Software Constraints: The project will use Node.js for the server, Express.js for the back-end, and MongoDB for data storage.
* Security Constraints: User data must be protected, requiring encryption and secure authentication protocols.

2.7 Use Case Diagram



2.8 Assumptions and Dependencies

* Assumptions: Users have access to the internet and can navigate web applications.
* Dependencies: The project relies on third-party libraries and services for notifications and data management.

**3. System Requirements**

3.1 User Interface

The UI will be designed for accessibility and simplicity, featuring:

* A home screen for easy navigation.
* A real-time dashboard showing trekker availability.
* Profile settings for user customization.

3.2 Software Interface

The back-end will communicate with the front-end using RESTful APIs for:

* User authentication.
* Retrieval and update of trekker and user data.

3.3 Database Interface

The application will use MongoDB to manage user and trekker data, including:

* User profiles.
* Trekker schedules and seat availability.

3.4 Protocols

All data transmission will be secured using HTTPS to protect sensitive information, including user credentials and trip details.

**4. Non-functional Requirements**

4.1 Performance Requirements

The application should support at least 1,000 concurrent users with a maximum loading time of 3 seconds.

Real-time updates should reflect changes within 2 seconds.

4.2 Security Requirements

User data encryption must be implemented.

A two-factor authentication mechanism will enhance security.

4.3 Software Quality Attributes

* Availability: The system should maintain at least 95% uptime.
* Usability: Focus on intuitive design for ease of navigation.
* Maintainability: Modular design for easy updates and feature additions.
* Portability: Accessible across various devices (mobile and desktop).
* Reliability: Critical functions (user authentication, seat updates) must be thoroughly tested.

**5. Other Requirements**

* Legal Compliance: The system must comply with local data privacy laws.
* Documentation: User manuals and technical documentation will be provided**.**

**6. Future Prospects**

Future enhancements may include:

* Integration with mobile payment systems for cashless transactions.
* User feedback and rating system for drivers and trekkers.
* Expansion to additional cities and regions, tailoring the system to broader user needs.

**Appendix A: Glossary**

* Trekker: A shared auto-rickshaw used for commuting.
* API: Application Programming Interface, facilitating communication between software components.
* UI: User Interface, the means by which a user interacts with the application**.**

**Appendix B: Analysis Model**

Initial flow diagrams detailing user interactions and data processes**.**

**Appendix C: Issues List**

* Pending technical issues regarding real-time notifications.
* User interface feedback from initial testing phases**.**