CampusLink: Product Requirement Documment

Introduction:

CampusLink aims to revolutionize transport services for students and staff across the university's campuses. By developing a digital solution, the project will enhance transportation efficiency, reduce wait times, and provide a convenient and reliable transportation system for users.

Problem Statement:

The university's current transportation system is inefficient and traditional, which is outdated in the technological era, leading to inconveniences for students and drivers. Students struggle with scheduling and payment issues, while drivers find it challenging to manage their travel records. Additionally, commuters to and from other campuses and external locations face difficulties due to a lack of bus stops. There is a pressing need for a comprehensive transport solution to improve accessibility and efficiency across all university campuses and for external commuting.

User Personas:

Bus Drivers: will use the app to view their schedules, manage their travel history, receive notifications about their routes, and provide feedback on their experiences.

Passengers (staff, students, and external commuters): will use the app to book seats, track bus locations in real-time, receive notifications about bus arrivals and departures, provide feedback on their experience, and access important transport information.

Key Features:

For passengers: .

1. Instant Order: passengers can request immediate transportation services.

2. Special Order: Passengers can request transportation to and from locations without bus stops.

3. Payment System: Passengers can make payments for their transportation electronically.

4. Transport history: Passengers can view their past transportation orders and routes.

5. Feedback system: Passengers can provide feedback on their transportation experience.

For Drivers:

1. Receive Order: Drivers can accept payment transportation request from passengers.

2. Accept Payment: Drivers can receive payments from passengers for transportation serrvices.

3. Manage History: Drivers can view their past transportation orders and routes.

4. Feedback: Drivers can receive feedback from passengers on their service.

Technical requirements:

Frontend UI: Next.js with React for building the frontend user interface.

Styling: Tailwind CSS for styling the frontend components.

Backend: Node.js for building the backend server application.

Database: MongoDB for storing data related to users, bookings, routes, and feedback.

APIs: Restful APIs with expressJs for communication between the frontend and the backend.

Development aproach:

Agile software development.

User experience:

The app will have a user-friendly interface with clear, simple, and intuitive navigation.