“Raahi: Connecting Drivers and Riders for Effortless Mobility”

راہی :ہر سفر آسان

Submitted By: **Zahra Yasin**

Introduction:

In today's fast-paced urban environments, commuters often struggle with unreliable transportation options, long wait times, and the lack of accessible, user-friendly digital platforms for everyday travel. **Raahi (راہی)** addresses this gap by offering a smart, intuitive ride-hailing web application designed to connect passengers with available drivers in real time. The platform streamlines the process of booking a ride through a seamless interface, enabling safe, efficient, and personalized travel experiences for daily commuters. By focusing on simplicity, responsiveness, and role-based functionality, Raahi empowers users to manage their journeys with confidence and ease.

Tech Stack Used and Why:

Technology / Tool	**Purpose**
React.js	Frontend framework for building a dynamic and component-based user interface
Redux Toolkit	Centralized state management for users, rides, and driver availability
React Router	Enables smooth page navigation without full reloads
Tailwind CSS	Utility-first CSS framework for modern, responsive, mobile-friendly UI
React Hot Toast	Provides lightweight and customizable toast notifications for user feedback
Vite	Fast and modern build tool for optimized frontend development
GitHub	Version control, collaboration, and project hosting
Visual Studio Code	Primary development environment for writing, debugging, and managing code

Assumptions:

- The application supports two user roles**: Passenger** and **Driver**, selected during signup.
- **Dummy data** is used for user registration and login; all user information is stored and managed through **local state** and **local storage** for session tracking.
- A **single passenger** is associated with each ride; there is no ride-sharing or pooling functionality in this version.
- The **ride lifecycle** includes the following statuses: "Requested", "Accepted", "In Progress", "Completed", and "Cancelled".
- **Drivers** have a toggleable **availability status**: either "Free" or "Busy", which controls their visibility to passengers.
- Location inputs (pickup and drop-off points) are **text-based only**, with no real-time map or geolocation integration.
- The platform supports three predefined ride types: **Car**, **Rickshaw**, and **Bike**.
- The application includes static **"Home"** and **"About Us"** screens to support **Progressive Web App (PWA)** behavior, improving accessibility and offline readiness.

Features:

Feature	Description
Role-Based Login / Signup	Users can sign up or log in by selecting a role , **Passenger** or **Driver**. The app routes them to their respective dashboards based on the selected role.
Passenger Dashboard	After a successful ride request, passengers are redirected to their **Passenger Dashboard** where they can view **scheduled rides**, **ride history**, and **status updates**.
Find a Ride Form	Passengers can enter details like name, pickup & drop-off locations, date, time, and ride type to request a ride.
Cancel Ride	Passengers can cancel any of their scheduled rides before they are accepted or in progress. Passengers can also view the ride status and interact when the ride has started or is completed.
Ride History	A dedicated section shows the user's completed ride history, including date, route, time, and type of ride.
Driver Dashboard	Drivers are redirected to their **Driver Dashboard** upon login, where they can view a list of all **incoming ride requests** from passengers. Drivers can toggle their status between **Free** and **Busy**.
Accept / Reject Ride Requests	Drivers can accept or reject incoming ride requests. Accepted rides are updated in real-time and marked as "In Progress" or "Completed" accordingly.
In-App Messaging	A built-in chat feature allows **drivers and passengers** to communicate in real time once a ride is matched, ensuring coordination and trust.
Toast Alerts & Feedback	Real-time toast notifications provide immediate user feedback for actions like ride cancelled, ride completed, ride started.
Responsiveness	Fully optimized for all screen sizes.

Entity Relationship Diagram: