**Ride-Sharing Service**

PREKSHA S – PES1UG22CS450

ROJA ARPITH A V – PES1UG22CS484

S PRIYANKA – PES1UG22CS491

Here we are creating a ride booking system. User can add passengers, add drivers, book a ride for a passenger, cancel a ride along with updating the system and calculate the total fare of any ride.

**Structures:**

* Location Structure: Represents latitude and longitude coordinates.

**Driver Structure:**

* name: Name of the driver.
* car\_details: Details about the driver's car.
* availability: Status of availability (0 for occupied, 1 for free).
* location: Driver's current location.

**Passenger Structure:**

* name: Name of the passenger.
* location: Passenger's current location.

**Ride Structure:**

* name\_P: Passenger's name.
* car\_details: Details about the car for the ride.
* status: Status of the ride (0 for Pending, 1 for Completed, -1 for Cancelled).
* current: Current location of the ride.
* destination: Destination of the ride.

**Functions:**

* distance(struct Location l1, struct Location l2): Calculates the distance between two locations using their latitude and longitude.
* calculateFare(struct Location l1, struct Location l2): Calculates the fare based on the distance between two locations.
* completeride(struct ride\* head\_R,struct driver\* head\_D): Completes a ride by updating the ride status and driver availability. Calculates the fare using calculateFare.
* cancelride(struct ride\* head\_R, struct driver\* head\_D): Cancels a ride by updating the ride status and driver availability.
* addRide(struct ride \*head\_R, struct driver \*head\_D): Adds a new ride request. Finds the nearest available driver and assigns the ride to them.
* addDriver(struct driver \*head\_D): Adds a new driver to the system with details like name, car details, availability, and location.
* addPassenger(struct passenger \*head\_P): Adds a new passenger to the system with details like name and location.

**Main Function**:

* Menu-Based Interaction: The main function presents a menu-driven interface to interact with the ride-sharing system.
* Switch Case: Handles user choices from the menu to perform various actions like adding drivers/passengers, requesting a ride, completing/canceling rides, and calculating ride fares.
* Driver, Passenger, Ride Handling: The menu options call respective functions to perform operations related to drivers, passengers, and rides.
* Input Handling: Collects user input for various functionalities, such as adding drivers, passengers, locations, etc.
* Loop: Continuously prompts the user for input until the user chooses to exit (by inputting 0 or another specified option).