Domain Model Description

The domain model captures the main entities, attributes, and associations for the ride-sharing application, based on the specified functional requirements.

The abstract entity **User** contains common attributes (`name`, `email`, `password`, `role`) and is specialized into **Rider** and **Driver**. Each user has exactly one **Wallet** (`balance`). Riders can request zero or more **Ride** instances, while Drivers can accept rides and configure zero or more **AvailabilityWindow** entries (`startTime`, `endTime`) to indicate availability.

The **Ride** entity stores `pickUpLocation`, `destination`, `fareEstimate`, `requestedTime`, and `status`, and is associated with exactly one Rider and optionally one Driver. The ride status follows the defined lifecycle and valid state transitions.

Upon ride completion, a **Payment** record is created with attributes `payTime` and `rideReference`. Each ride can have at most one associated payment, which supports the deduction from the Rider's wallet and credit to the Driver's wallet.

Ride history is captured through the abstract **RideHistory** (`pickupLocation`, `destination`, `timestamp`, `status`), specialized as **RidersRideHistory** (`farePaid`, `driversName`) and **DriversRideHistory** (`fareReceived`, `ridersName`). These maintain role-specific historical records for completed or cancelled rides.

The model applies generalization to avoid duplication of common attributes, and associations with multiplicities are defined to enforce business rules such as one wallet per user, one payment per completed ride, and multiple ride requests or acceptances. The explicit modelling of **Wallet**, **Payment**, and **AvailabilityWindow** enables clear separation of concerns for financial transactions, scheduling, and ride management.