Django Web-App: Ride Sharing Service

Engineering Robust Server Software
Homework 1

For this assignment you will be writing a web-app in Django. This web-app will let users request, drive for, and join rides. In particular, your system should allow three roles:

Ride Owner – When a user requests a ride, he/she becomes the owner of that ride. Requesting a ride should involve specifying a destination address, a required arrival (date & time), the number of total passengers from the owner's party, and optionally, a vehicle type and any other special requests[1]. A request will also indicate whether this ride can be shared or not – a shared ride can be joined by other users (ride sharers). A ride owner would be able to modify a ride request up until it is *confirmed* (a ride becomes *confirmed* once a driver accepts the ride and is in route). A ride is *open* from the time it is requested until that point. A ride owner can also view ride status until the ride is *complete* (a ride becomes closed once a driver finishes the ride and marks it as *complete*).

Ride Driver – A user can register as a driver, and in doing so will provide their name along with their vehicle information. The vehicle information includes the type, license plate number, maximum number of passengers, and optionally any other special vehicle info[1]. To simplify, a driver can only have 1 vehicle. A driver can search for *open* ride requests based on the ride request attributes. A driver can claim and start a ride service, thus confirming it. A driver can also *complete* rides that they service after reaching the destination to indicate that the ride is finished.

Ride Sharer – A user can search for *open* ride requests by specifying a destination, arrival window (the user's earliest and latest acceptable arrival date & time) and number of passengers in their party. The user can then become a ride sharer, by joining that ride. A ride sharer can also view the ride status, similarly to a ride owner. Finally, a ride sharer can edit their ride status as long as the ride is *open*.

Note that your system should support *multiple* rides, and the same user MAY hold different roles in different rides. For example, a user may be an owner of a current ride, a ride sharer on yet a later ride in the day, and a driver of 2 rides scheduled for the following day.

You should support the following functionality:

Create Account – A user should be able to create an account if they do not have one.

Login/Logout – A user with an account should be able to login and logout.

---

[1] The "special requests" of a ride request and the "special vehicle info" of a driver's vehicle should free-text fields. If a "special request is specified with a ride request, it must exactly match a driver's "special vehicle info" for that driver to be eligible to service that ride.

Driver Registration – A logged-in user should be able to register as a driver and enter their personal and vehicle info. They should also be able to access and edit their info.

Ride Selection – If a logged-in user is part of multiple rides, she should be able to select which ride she wants to perform actions on. If a logged in user belongs to only one ride, you MAY display your ride-selection mechanism with the one ride, or you MAY omit it (not show it). Note this should allow selection of any *open* or *confirmed* rides for that user (but not *complete* rides).

Ride Requesting – A logged-in user should be able to request a ride. Requesting a ride should allow the owner to specify the destination address, a required arrival date / time, the number of total passengers from their party, a vehicle type (optionally), whether the ride may be shared by other users or not, and any other special requests.

Ride Request Editing (Owner) – A ride owner should be able to edit the specific requested attributes of the ride as long as the ride is not *confirmed*.

Ride Status Viewing (Owner / Sharer) – A ride owner or sharer should be able to view the status of their non-*complete* rides. For *open* ride requests, this should show the current ride details (from the original request + any updates due to sharers joining the ride). For *confirmed* ride requests, the driver and vehicle details should also be shown.

Ride Status Viewing (Driver) – A ride driver should be able to view the status of their *confirmed* rides, which should show the information for the owner and each sharer of the ride, including the number of passengers in each party. A driver should also be able to edit a ride to mark it as *complete*.

Ride Searching (Driver) – A driver should be able to search for *open* ride requests. Only requests which fit within the driver's vehicle capacity and match the vehicle type and special request info (if either of those were specified in the ride request) should be shown. A driver can claim and start a ride service, thus confirming it. Once closed, the ride owner and each sharer should be notified by email that the ride has been *confirmed* (hence no further changes are allowed).

Ride Searching (Sharer) – A user should be able to search for *open* ride requests by specifying a destination, arrival window (the user's earliest and latest acceptable arrival time) and number of passengers in their party. A sharer should be able to join a selected ride, if any exist in the resulting list of pending rides.

A few notes:

- Your grade will not be affected by the aesthetics of your website (this is not a UI/UX class). However, your site should be use-able in a reasonable way (your TA should be able to easily figure out how to perform each feature).
- You are, however, encourages to make a website which you would be proud to show to your friends, family, and prospective employers.
- You MAY use any CSS/JavaScript libraries that you want. In fact, we strongly encourage it! Bootstrap (https://getbootstrap.com/) is quite popular.
- Many of these requirements are broad, and open to interpretation. We are not specifying the details of how these should be implemented, or the exact nature of the functionality. For example, whether ride selection is a screen displayed after login, or a menu in the corner of every page, or some other completely different mechanism, is totally up to you.
- Remember to keep your "danger log."