BILKENT UNIVERSITY

FALL 2017 - CS 353

TERM PROJECT FINAL REPORT

Private Taxi Database Management System

GROUP 12

21301027 Nergiz Ünal Sec-1

21200992 Rıdvan Çelik Sec-2

21401058 Orhun Kar Sec-2

21200509 Zeynep Delal Mutlu Sec-1

1. Application System Description

Private Taxi Database Management System is an online web application designed to manage the relation between taxi drivers and taxi users. The system simulates and works like the application Uber.

Web application will have 3 user types: customer, driver and admin

Users who are logged in as customers will be able to change their preferences and see available drivers according to their preferences after every change in their preferences. After that customers will be able to send requests to these drivers. Also after completing a trip customer can rate and make a comment about the trip they had.

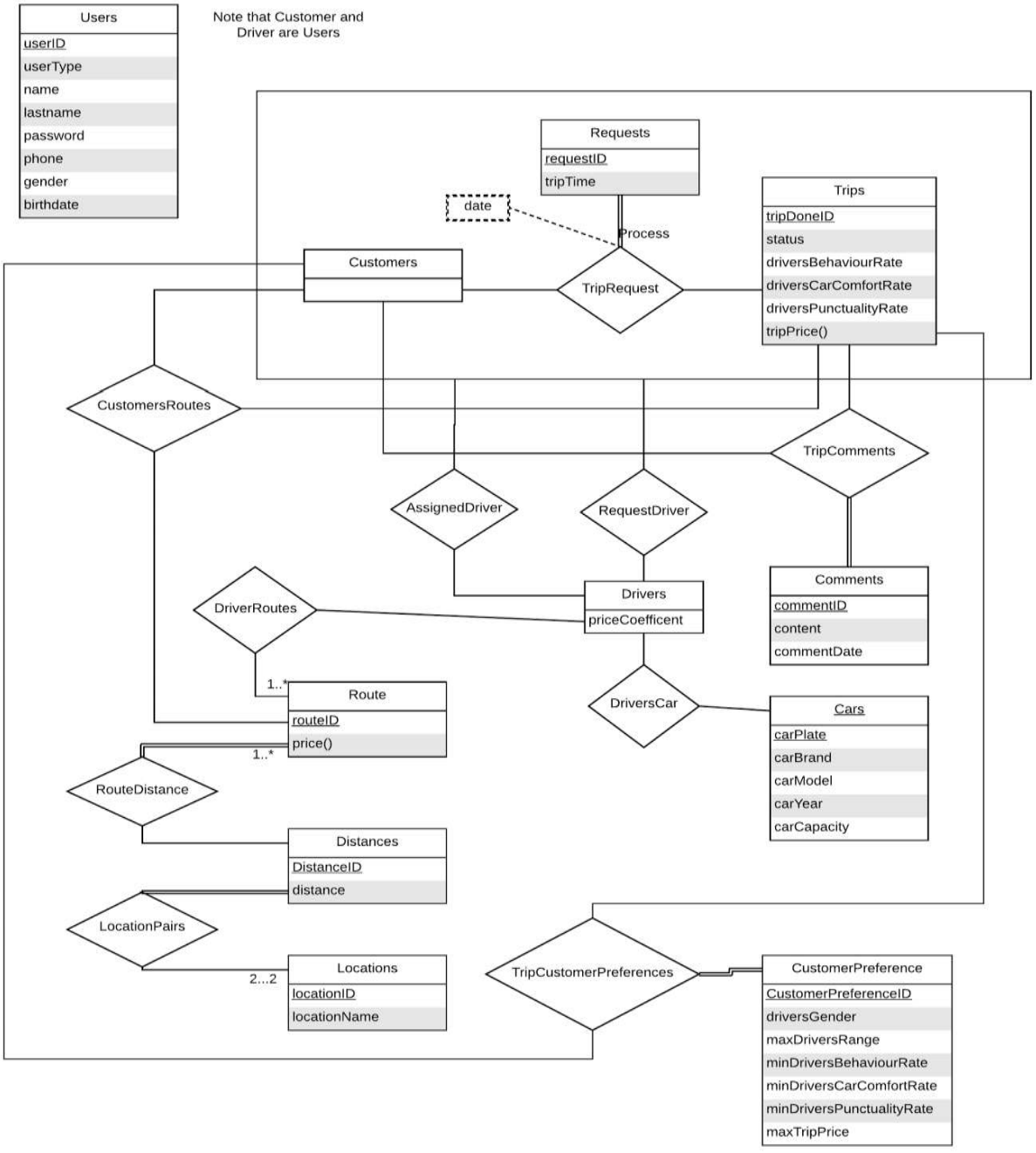
Users who are logged in as drivers will have a list of requests from different customers and will accept or deny these request. Similar to customers, drivers also can comment on completed trips.

Admins have absolute control on the system, they can view all database and manipulate records.

In this application taxi calling system follows these steps:

1. Customer sets their preferences for taxi and driver.
2. Driver sets their preferences(stops)
3. A list of available and suitable drivers offered to customer as a list
4. Customer sends requests to drivers and if the driver accepts the request a match will be created.
5. If both driver and user confirms when the trip is done, a trip record will be created

2. Final E/R


Users
userID
userType
name
lastname
password
phone
gender
birthdate
Note that Customer and Driver are Users
Requests
requestID
tripTime
date
Process
Trips
tripDoneID
status
driversBehaviourRate
driversCarComfortRate
driversPunctualityRate
tripPrice()
Customers
TripRequest
CustomersRoutes
TripComments
AssignedDriver
RequestDriver
Comments
commentID
content
commentDate
DriverRoutes
Drivers
priceCoefficent
1..*
Route
routeID
price()
DriversCar
Cars
carPlate
carBrand
carModel
carYear
carCapacity
1..*
RouteDistance
Distances
DistanceID
distance
LocationPairs
2...2
Locations
locationID
locationName
TripCustomerPreferences
CustomerPreference
CustomerPreferenceID
driversGender
maxDriversRange
minDriversBehaviourRate
minDriversCarComfortRate
minDriversPunctualityRate
maxTripPrice

3. Final List of Tables

3.1 Users

Relational Model:

Users(userID,usertype, name, lastname, password, phone, gender, birthdate)

3.2 Customers

Relational Model:

Customers(userID)

3.3 Drivers

Relational Model:

Drivers(userID)

3.4 Requests

Relational Model:

Requests(requestID,tripTime)

3.5 Trips

Relational Model:

Trips(tripDoneID, status, driversBehaviourRate,driversCarComfortRate, driversPunctualityRate, tripPrice())

3.6 Cars

Relational Model:

Cars(carPlate,carBrand,carModel,carYear,carCapacity)

3.7 Route

Relational Model:

Route(routeID, price())

3.8 Comments

Relational Model:

Comments(commentID, content, commentDate)

3.9 Distances

Relational Model:

Distances(DistanceID,distance)

3.10 Locations

Relational Model:

Locations(locationID, locationName)

3.11 TripRequest

Relational Model:

TripRequest(userID,requestID,tripDoneID,date)

3.12 TripConsruct

Relational Model:

3.13 CustomerRoutes

Relational Model:

CustomerRoutes(tripDoneID,userID,routeID)

3.14 DriverRoutes

Relational Model:

DriverRoutes(userID,routeID)

3.15 TripComments

Relational Model:

TripComments(commentID,userID,tripDoneID)

3.16 DriversCar

Relational Model:

DriversCar(userID)

3.17 LocationPairs

Relational Model:

LocationPairs(locationID1,locationID2)

3. Functional Dependencies and Normalization of Tables

Since our database is not much complex, we will not need to decompose any table.

4. Implementation Details

Our project will be MVC architectural pattern as most web applications are. Data tier will be constructed with mySQL server. In the controller we will use php. View will be provided with

HTML. Visual Studio 2017 will be used for the application logic and the interface of the application, since external tools can be easily added to Visual Studio.

5-) Advanced Database Features

5.1-) Conflict Check (Trigger)

- The number of passengers will be checked if it exceeds the size of the car or the preference of the driver.

5.2-) Information Update (Trigger)

- The rating of each driver is going to be calculated after each given rate by the passengers.

5.3-) Cancellation (Trigger)

- A passenger can send a trip request to more than one drivers for the same trip. After first driver's approval, the other requests will be dropped by the system.

5.4-) Finding the most travelled driver(Report)

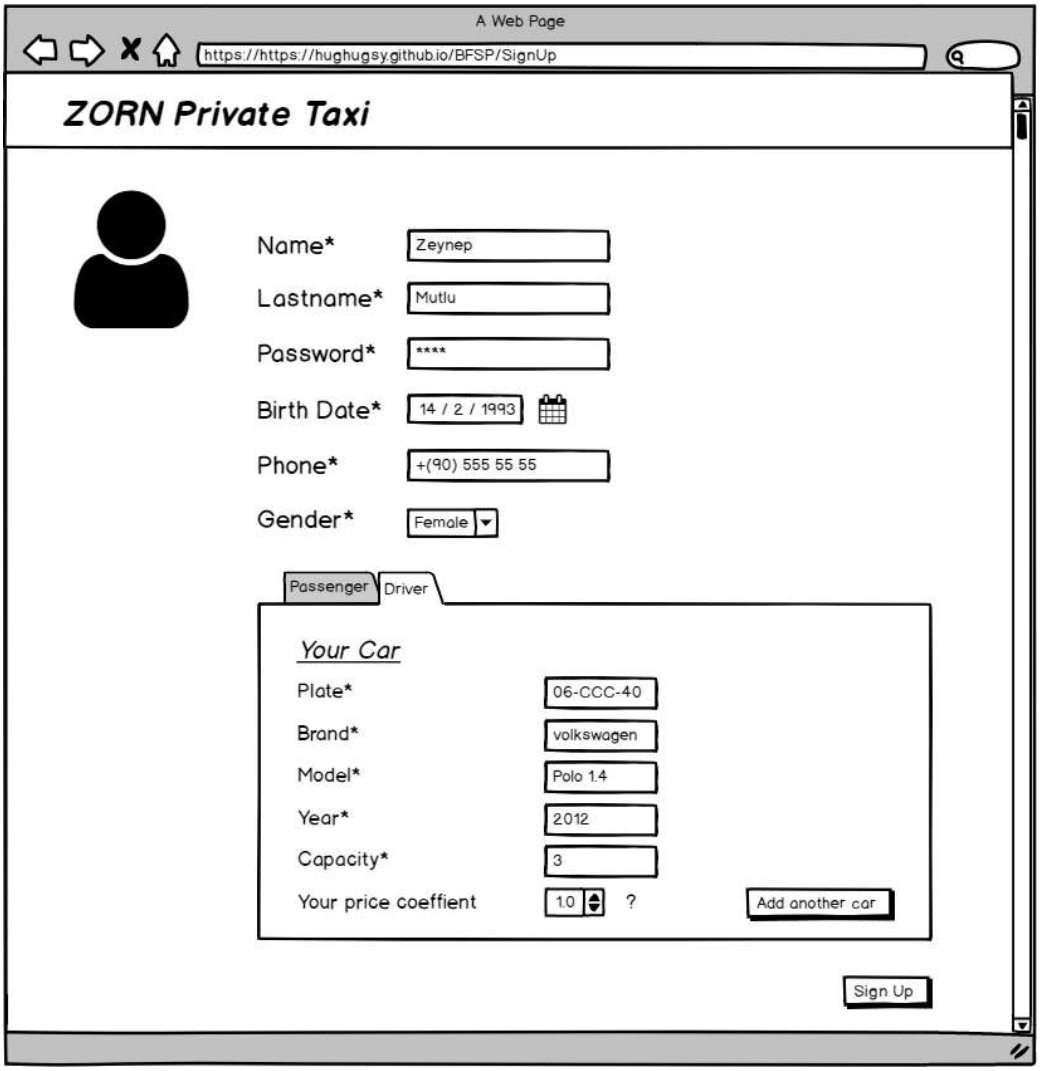
- After summing up all distances of the trips of a driver. All drives will be sorted to find most travelled drives.

5.5-) Price calculation (Stored Procedures)

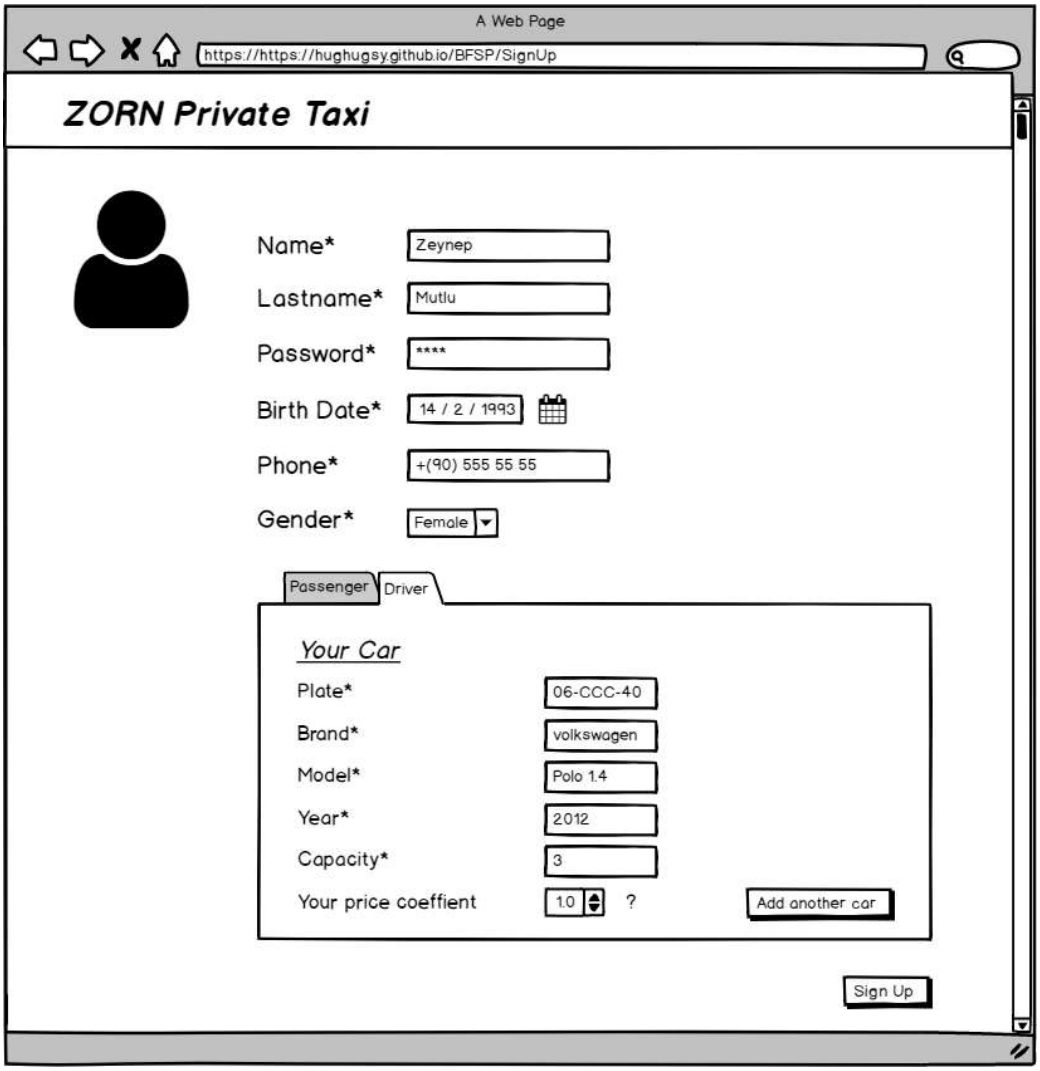
- In this database standard procedure will be used for price calculation. Each driver sets a price index for their trips. This index times the total kilometer travelled gives the price of the trip. This procedure will be applied to all trips, so this will be a stored procedure.

6-) User Manual

To use the system all users must sign up for the application first by using this two screens. First one is for drivers.

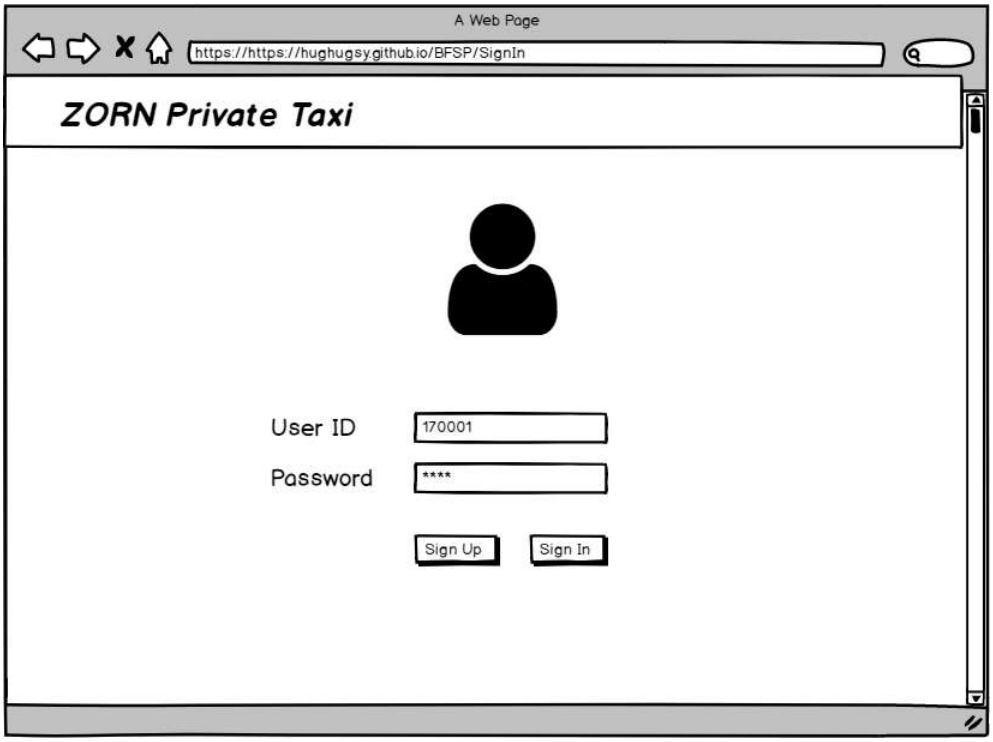


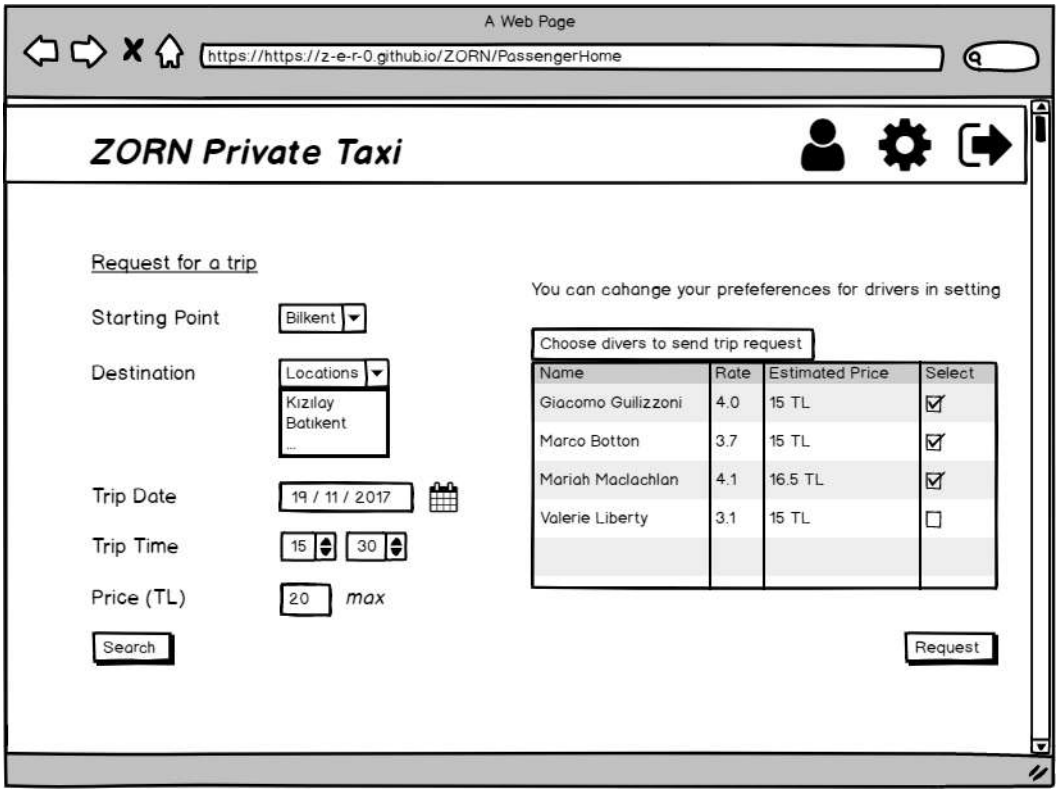
By navigating to the passenger view, a passenger can sign up



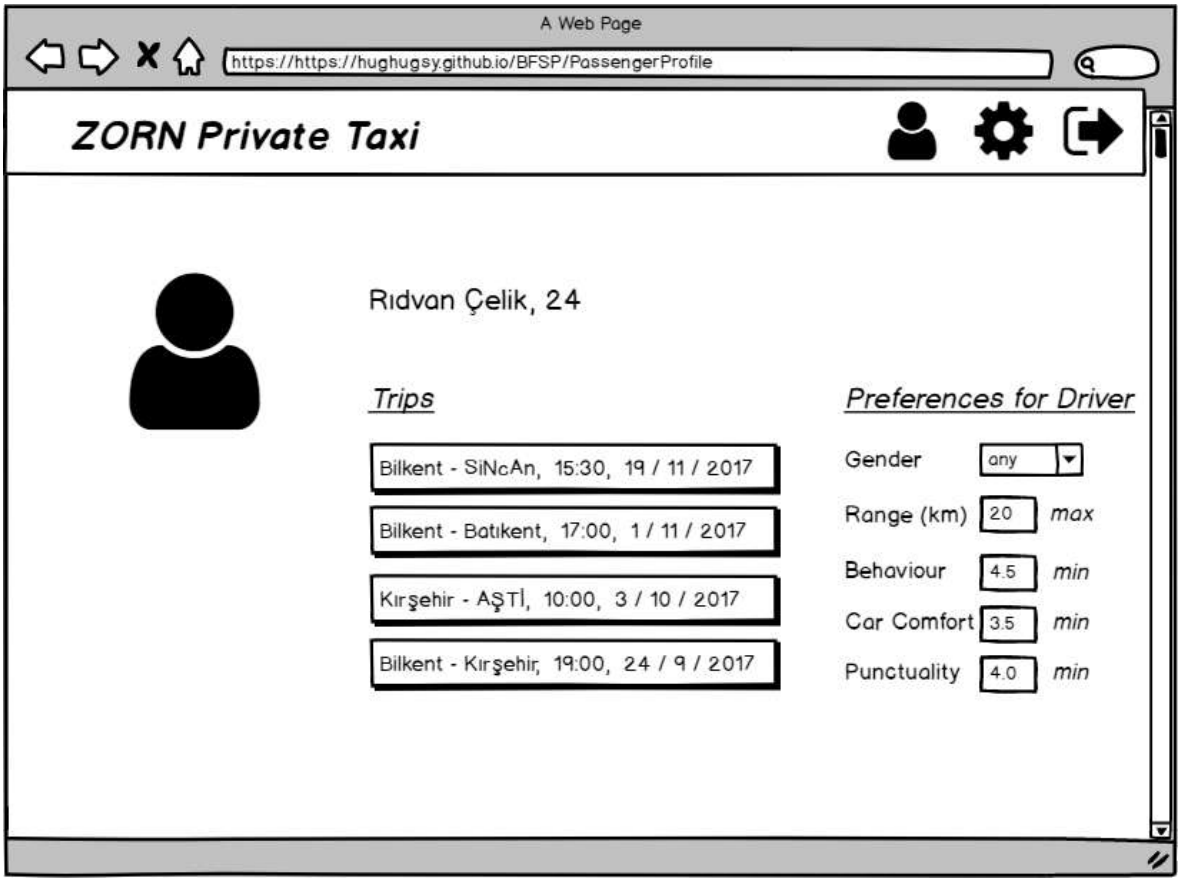
After signing up, Users now can easily login to the application by using this screen.

10

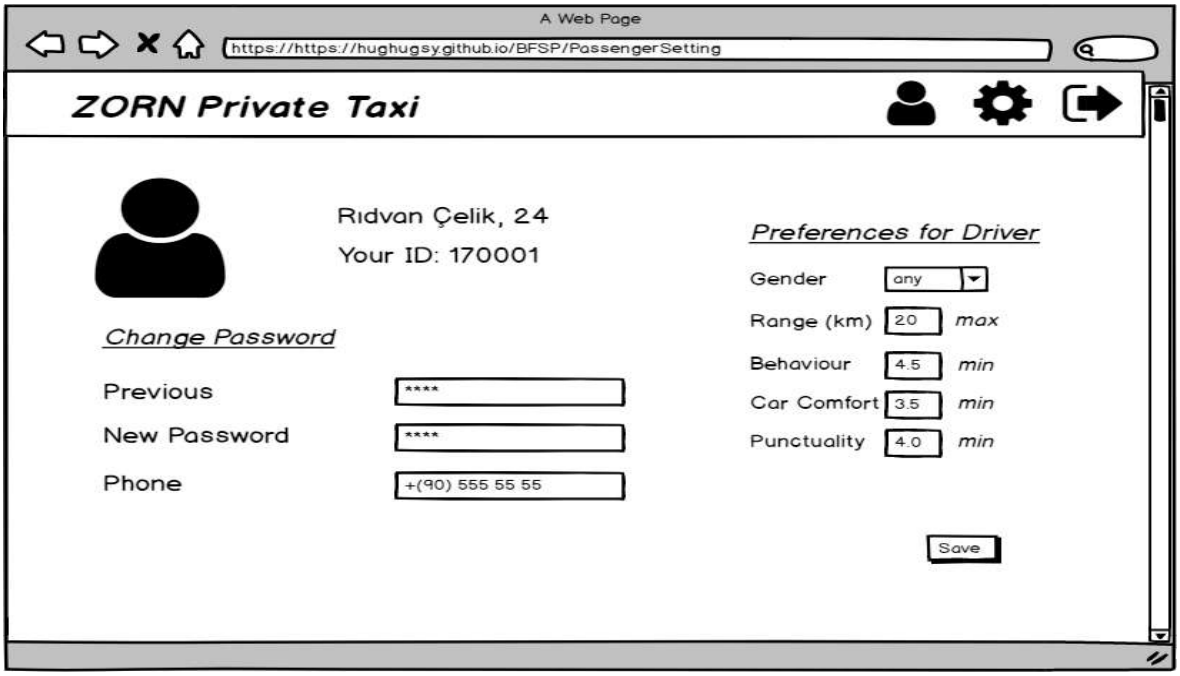

A Web Page
https://https://hughugsy.github.io/BFSP/SignIn
ZORN Private Taxi
User ID
170001
Password
****
Sign Up
Sign In



This screen is the passengers main screen. Here left side of the screen there are several variables to set for passenger. After setting them and clicking to the search button right side of the screen will be updated. Available and appropriate drivers to the passengers will be shown there. By selecting the little boxes next to the drivers and clicking request button customer will send requests to these drivers.

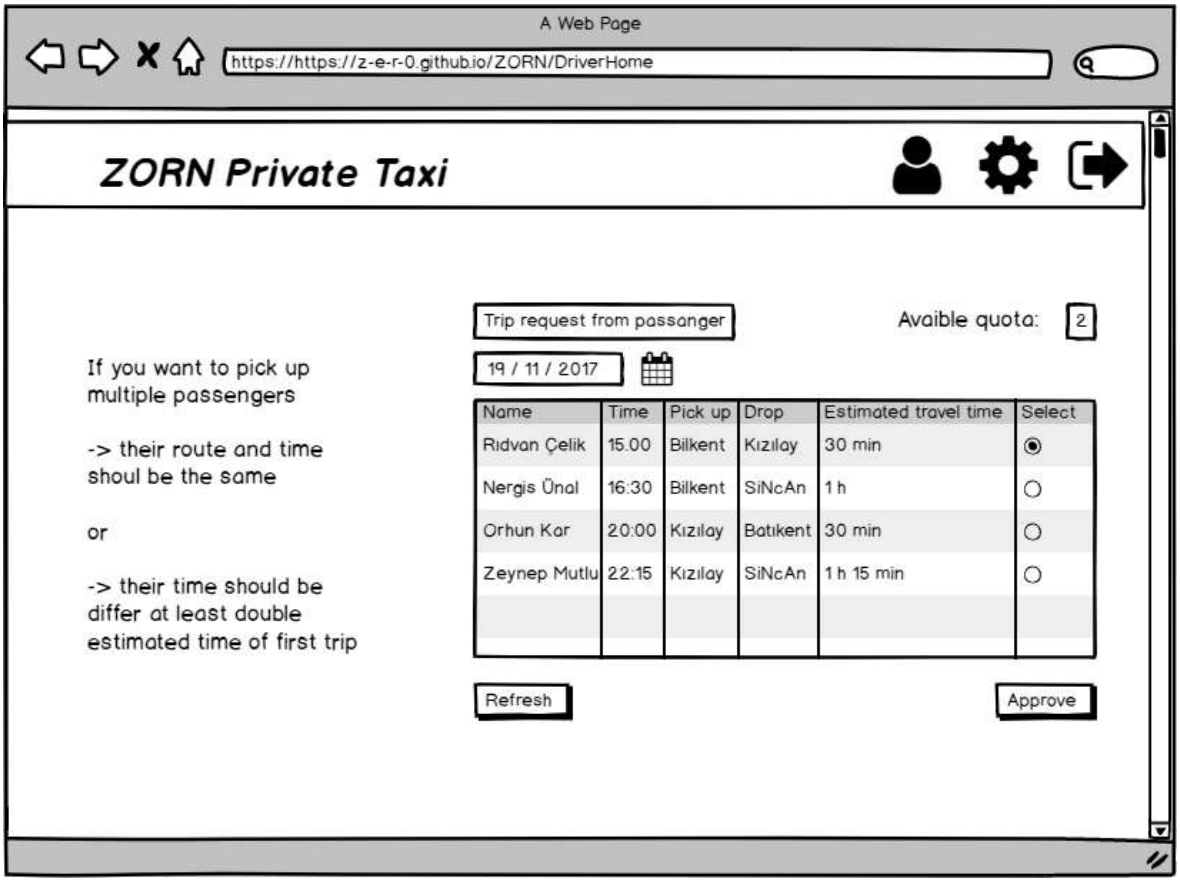


This screen is the profile of the passenger. Here passenger can see his/her past trips and by clicking the cogwheel button he/she can update their preferences.

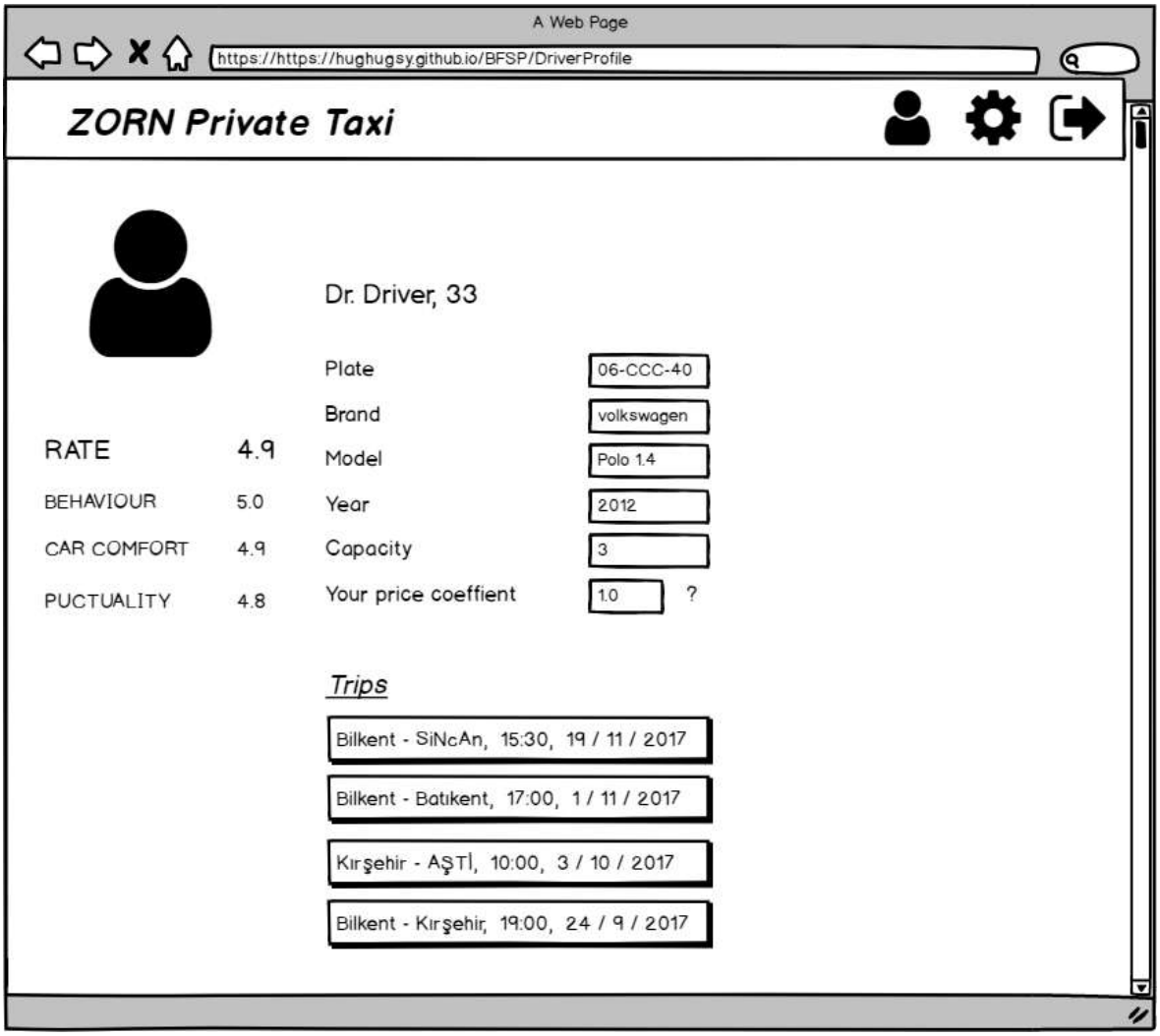


This is the settings page of the customer here he can update his information.

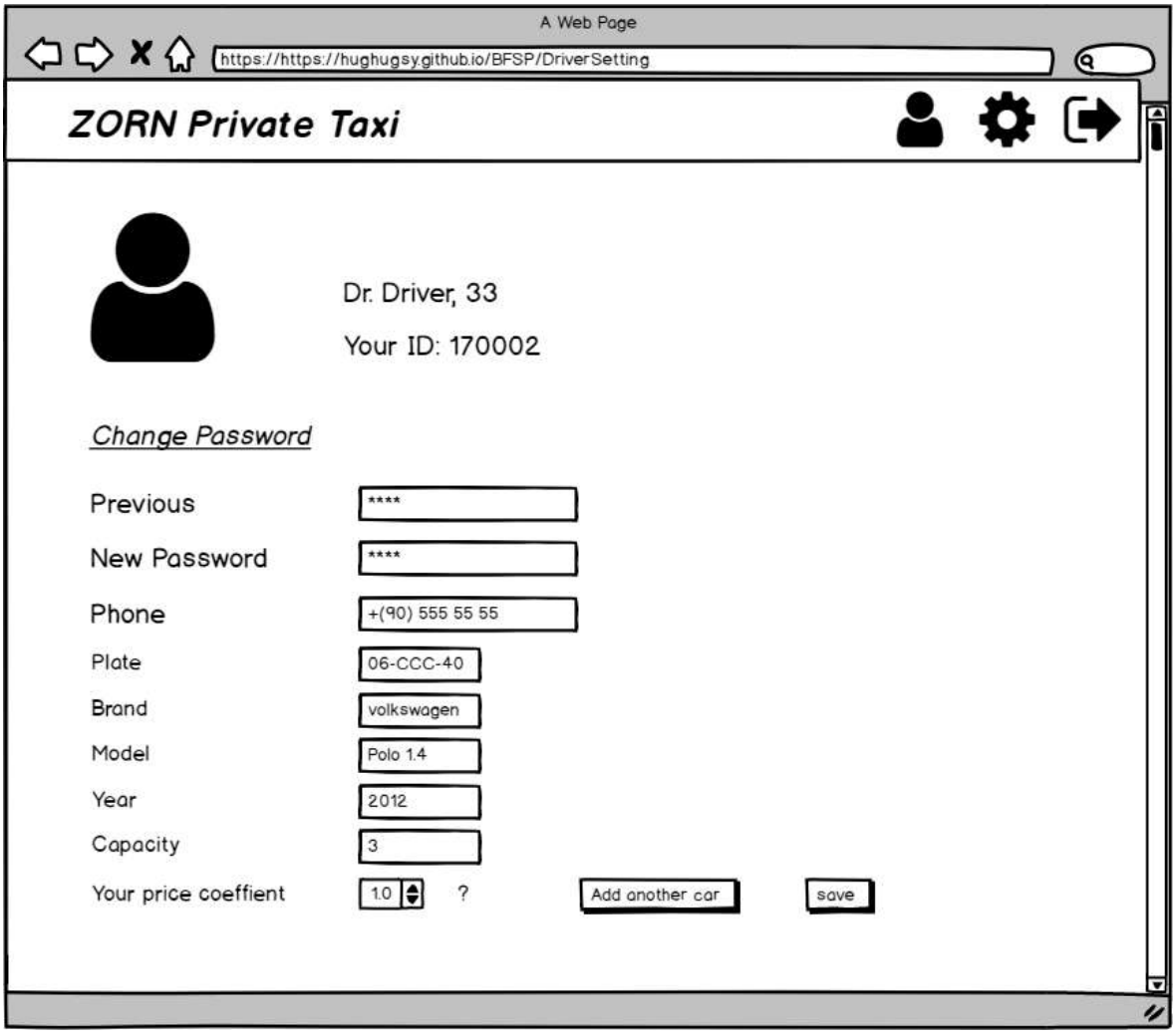
14



This is the main screen of the drivers. Here a driver can see the current requests from passengers. Here driver can select the customer he wants to serve and click approve button to construct a trip.



This is the driver profile page. Here driver can update his information by clicking the cogwheel. Also can see his car and his price index. In addition to that a driver also can see his past trips.



This is the settings page of the driver here he can update his informations and add another car and set his price coefficient