***Guest***

1. Sign In

Description:

A guest, who doesn't have a profile yet, can submit his data to the system in order to become a registered User.

Actors:

Guest

Input Condition:

Null

Output Condition:

Guest successfully ends the registration process, and becomes a User. From now on he/she can log in to the application using his/her credential and have access to peculiar functionalities.

Events flow:

* Guest on the home page clicks on the “register” button to start the registration process;
* Guest fills in at least all the mandatory fields;
* Guest clicks on “confirm” button;
* The application will save the data in the database.

Exceptions:

* The guest has already registered;
* The chosen user name is already used;
* The email inserted is already associated to another user;
* One or more mandatory fields are empty or not valid;

All the exceptions cause the application to notify the error to the user with an alert window;

the application then come back to the registration form, to restart the Event Flow from the filling step.

2. Sign Up as Taxi Driver

Description

A guest, who doesn't have a profile yet, can submit his data to the system in order to become a Taxi Driver.

Actors:

Guest

Input Condition:

The Guest must have a driving licence and a taxi licence.

Output Condition:

Guest successfully ends the registration process, and becomes a Taxi Driver. From now on he/she can log in to the application using his/her credential and have access to peculiar functionalities.

Events Flow:

* Guest on the home page clicks on the “sign up as a taxi driver” button to start the registration process;
* Guest fills in at least all the mandatory fields, and upload data related to driving license and taxi license;
* Guest clicks on “confirm” button;
* The application will save the data in the database.
* After a check by administrators, the Guest will be authorized to login as a Taxi Driver;

Exceptions:

* The guest has already registered;
* The chosen user name is already used;
* The email inserted is already associated to another user;
* One or more mandatory fields are empty or not valid;
* Driving license or taxi license are not valid;
* Driving license or taxi license are already associated to another user.

All the exceptions cause the application to notify the error to the user with an alert window;

the application then come back to the registration form, to restart the Events Flow from the filling step.

3. Login

Description:

A guest already registered can provide his credential to log in to the application, becoming a User and thus gaining User or Taxi Driver privileges.

Actors:

Guest

Input Conditions:

The Guest must be already registered to the system.

Output Conditions:

The Guest is promoted to User or Taxi Driver.

Events Flow:

* Guest fills in the log in form already present in the home page.
* The application verifies the inserted credentials, and if they are correct, promotes the guest and shows additional features.

Exceptions:

If the credentials are not correct, so user name and password don't match, an alert window will be shown and the access denied, remaining on the home page, and giving the guest the possibility to try again.

4. Call a taxi

Description:

The guest has the opportunity to call for a taxi ride without being registered to the site.

Actors:

Guest, Taxi Driver

Input Condition:

Null

Output Conditions:

The first free taxi will answer the call and the guest will be served as requested.

Events Flow:

* Guest fills in the form in the home page, providing a name, a surname, the number of passengers and a phone number;
* Guest submits the request.
* His/her data, and his/her position, detected by using GPS, will be processed by the system, that starts sending the request to the first taxi in the zone queue;
* The taxi accept the request, and goes to take the guest in the place reported by the system.

Exceptions:

* If the guest doesn't fill in all the fields, or the values inserted are not valid, the request can not be sent, and the user will be warned;
* If the taxi declines the request, the system will move that driver to the bottom of the queue, and proceeds asking to the next taxi in the queue, following the order.
* If problems arise with positioning or times, taxi drivers can use the customer's phone number to call them and make verbal arrangements.

***User***

1. Logout

Description:

This functionality allows the User to close the current session on the site and brings him/her back to Guest level.

Actors:

User

Input Conditions:

The User must be logged in.

Output Conditions:

The User closes the current session, loses his/her privileges and becomes a Guest.

Events Flow:

* User clicks on the “Logout” button in the page.
* The system close the session, and brings the User back to home page, as a Guest.

Exceptions:

There are no possible exceptions.

2. Call a taxi

Description

This functionality is similar to the one presented for the Guest use case, but in this case most of the data are already present in the database, so the procedure is faster and easier.

Actors:

User, Taxi Driver

Input Conditions:

The User must be logged in.

Output conditions:

The first free taxi will answer the call and the User will be served as requested.

Events Flow:

* User inserts the number of passengers for the requested ride;
* User submits the request;
* The system collects the number inserted, together with User data and his/her position tracked with GPS, and sends a request to the first free taxi in the zone queue;
* The taxi accept the request, and goes to take the User in the place reported by the system.

Exceptions:

* If the User doesn't provide the number of passengers, or the value inserted is not valid, the request can not be sent, and the user will be warned;
* If the taxi declines the request, the system will move that driver to the bottom of the queue, and proceeds asking to the next taxi in the queue, following the order.
* If problems arise with positioning or times, taxi drivers can use the customer's phone number to call them and make verbal arrangements.

3. Book a taxi

Description:

The User can book a taxi in advance, to be sure to have a ride on time and without problems.

Actors:

User, Taxi Driver

Input Conditions:

The User must be logged in.

Output Conditions:

The User will have a booked ride for the time and the ride requested.

Events Flow:

* The User fills in the form for the booking procedure, deciding the time of the ride, that has to be at least two hours after the time of the reservation procedure, the starting address, the destination, and the number of passengers;
* The system saves the values inserted;
* Ten minutes before the arranged time, the system generates an automatic request to ask for the ride requested by the User;
* The automatic call works as a normal call for generic guest or users (see use case Call a Taxi for specific informations).

Exceptions

If the User tries to insert a time that is not at least two hours later that moment, the system will reject the request, and will give the User the opportunity to try again with other times.

***Taxi Driver***

1. Logout

Description:

This functionality allows the Taxi Driver to close the current session on the site and brings him/her back to Guest level.

Actors:

Taxi Driver

Input Conditions:

The Taxi Driver must be logged in.

Output Conditions:

The Taxi Driver closes the current session, loses his/her privileges and becomes a Guest.

Event Flow:

* User clicks on the “Logout” button in the page.
* The system close the session, and brings the Taxi Driver back to home page, as a Guest.

Exceptions:

There are no possible exceptions.

2. Give/Remove Availability

Description:

The Taxi Driver let the system knows if he/she is available or not to accept requests.

Actors:

Taxi Driver

Input Conditions:

The Taxi Driver has to be logged in.

Output Conditions:

The Taxi Driver is inserted in a queue, or he/she is removed from it.

Event Flow:

* When ready to accept requests, a Taxi Driver gives the availability, and he/she will be inserted by the system in the zone queue based on his position tracked by GPS.
* When he/she removes the availability, he/she will be removed from the queue where he/she was lastly placed.

Exceptions:

There are no possible exceptions.

3. Accept/Deny a Request

Description:

The Taxi Driver has the possibility to accept or decline a request from a customer.

Actors:

Taxi Driver

Input Conditions:

The Taxi Driver has to be logged in and be available.

Output Conditions:

If the Taxi Driver accepts the request, he/she will go to the place of the meeting with the customer; otherwise he/she will wait for another request.

Event Flow:

* When the Taxi Driver is the first of his zone queue, he/she will receive the next available request;
* Taxi Driver can decide to accept or reject the request;
* If he/she accepts, he/she can go to the place pointed by the system, to take on the customer;
* If he/she rejects, he/she will be put at the bottom of the queue.

Exceptions

If the Taxi Driver doesn't answer the request, after a minutes time the request will be considered as rejected by the Taxi Driver, that will be moved to the bottom of the queue.

***Administrator***

1. Logout

Description:

This functionality allows the Administrator to close the current session on the site and brings him/her back to Guest level.

Actors:

Administrator

Input Conditions:

The Administrator must be logged in.

Output Conditions:

The Administrator closes the current session, loses his/her privileges and becomes a Guest.

Event Flow:

* Administrator clicks on the “Logout” button in the page.
* The system close the session, and brings the Administrator back to home page, as a Guest.

Exceptions:

There are no possible exceptions.

2. Remove Users

Description:

The Administrator has the power to see the list of registered users, and remove any of them

if necessary.

Actors:

Administrator, User (passive presence).

Input Conditions:

Administrator has to be logged in, User has to be registered.

Output Conditions:

The selected User will be removed from the system.

His data, however, will remain saved in the database.

Events Flow:

* Administrator clicks on the “users management” button to access the dedicated page;
* In the management page he/she can see the list of registered users;
* Administrator select one of the users by clicking on his/her user name;
* By clicking the “delete” button, the Administrator removes the user from the system.

Exceptions:

There are no possible exceptions.

3. Taxi Drivers Management

Description:

The Administrator has the job to check every request from new taxi driver, in order to verify if data related to licenses are correct. If so, the Administrator gives the Taxi Driver the permission to work.

Actors:

Administrator, Taxi Driver (passive presence).

Input Conditions:

Administrator has to be logged in, Taxi Driver has to be registered.

Output Condition:

The Taxi Driver considered will be authorized to work, or rejected.

Events Flow:

* Administrator clicks on the “Taxi drivers management” button to access the dedicated page;
* In the management page he/she can see the list of pending requests;
* Administrator select one of the requests by clicking on the user name;
* A list of related data will be shown in the page, letting the Administrator check the information inserted by the user;
* By clicking on the “Accept” button the Taxi Driver will be accepted, otherwise by clicking on the “Deny” button the Taxi Driver will be rejected.

Exceptions

There are no possible exceptions.