Roadmap and Use case Bee Online Booking System.

For

Bee Transport Solution

Version 1.0

Prepared by Team Bee

Members: Djimgou Patrick Virgile, Timani Patrick, Vinoth Vinothsarun .

Team Bee.

10/11/2017

The first thing we need to know about developing an Bee Transport app is that we actually have to ***build two slightly different apps***, which would be connected to our admin portal: also one for the passengers and one for the drivers.

1. **- Passenger App: Essential Features**

* **Register/login page:** the best way is to offer social media integration for registering, though an email option is also ok.
* **Booking interface:** A screen where the user will input their travel data and hail a cab or book one for later.
* **Fare calculator** allows to estimate the cost of the ride prior to placing an order.
* **Driver’s rating and reviews:** feedback left by the customers’ on the driver, the trip, and the vehicle.
* **Messaging** to contact the driver and settle the details e.g. exact pickup point.
* **Payment:** Users should receive a fare quote and pay it in-app. They can securely store their credit cards for one-click checkout.
* **Push notifications:** Keep the user updated on the booking progress and deliver the necessary information on their ride e.g. car model, color, number on the plate and ETA (estimated time of arrival).
* **Ride history**: Displays details of the previous trips and receipts, plus can allow re-booking the same ride in one click (e.g. for the standard work-to-home commute).

1. **- Driver App: Essential Features**

Some of the driver app features will duplicate the passenger’s – login, push notifications, support and messaging. And there are additional unique features as well:

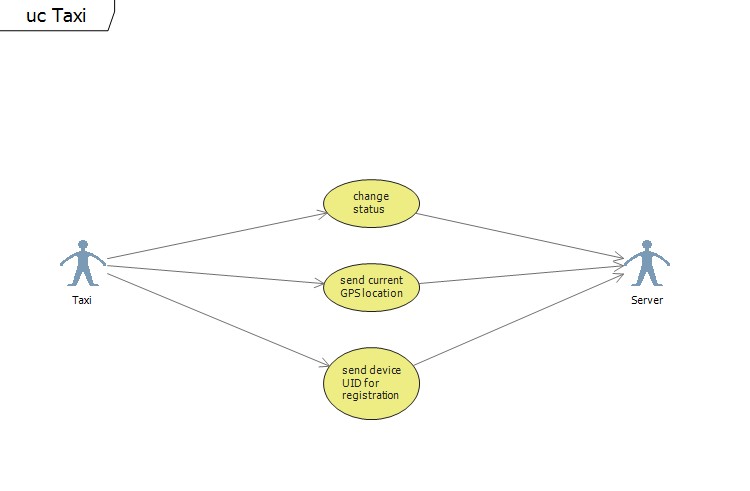
* **Driver profile and status**: Drivers should be required to provide extra verification e.g. their license and tax number when they register. The status functionality should help them adjust their availability and schedule.
* **Order alert**: push notification plus information on the new booking – payment, route details etc. Should allow accepting and canceling orders.
* **Navigation** suggests the best route to the passenger and onwards.
* **Estimation**: States the order price with discounts (if available).
* **Reports** offer a quick glance on trip stats, earnings, and payroll.

1. **--Admin Panel**

Your admin panel is a robust web app to store, manage and review all the in-app activity. It should offer you a quick overview of all the processes; suggest the best navigation for drivers and trips; manage revenue and payroll and collect data on your clientele for further enhancements.

**Use Cases**

**Taxi UC**



*4.1.1 Use case “change status”*

|  |  |
| --- | --- |
| Use case ID | TAXI1 |
| Name | *Change status* |
| Goal | *Change current taxi status* |
| Participating actors | Taxi and server |
| Precondition |  |
| Main scenario | *1. A customer stops a* Taxi /Moto-Taxi *which is roaming through the city*  *2. A* Taxi /Moto-Taxi *receives an order from the customer*  *3. A* Taxi /Moto-Taxi *changes its status from available to busy*  *4. When the order is done, taxi changes status back to free*  *1. Taxi updates status when is “on duty”*  *2. When taxi finishes with the working shift, updates status to “off duty”* |
| Exceptions | *If there is no internet connection,* Taxi /Moto-Taxi *can’t change its status* |
| Extensions |  |
| Dependent UC |  |

*4.1.2 Use case “send current GPS location”*

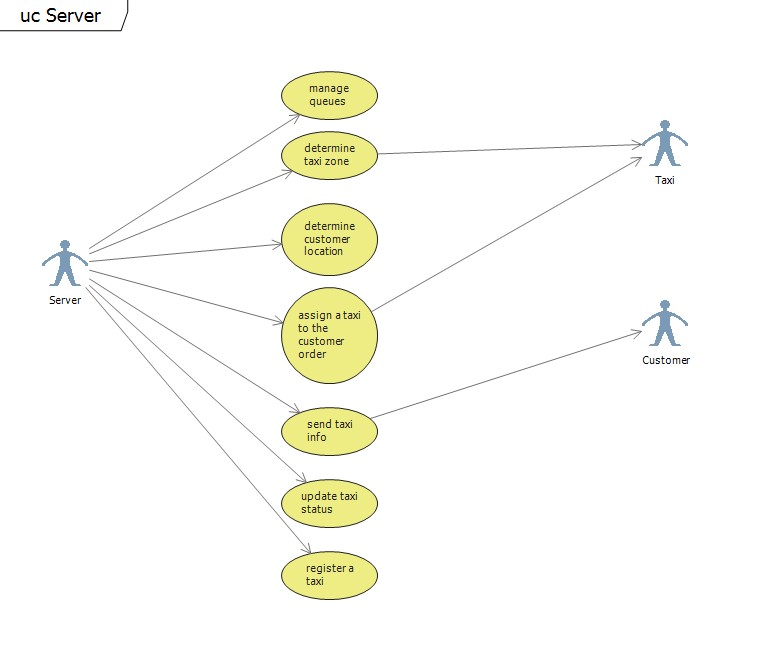
|  |  |
| --- | --- |
| Use case ID | TAXI2 |
| Name | *Send current GPS location* |
| Goal | *Update the location of the taxi* |

|  |  |
| --- | --- |
| Participating actors | Taxi /Moto-Taxi and server |
| Precondition | The Taxi /Moto-Taxi needs to be “on duty” |
| Main scenario | 1. Taxi /Moto-Taxi  *will send its GPS coordinates to the server every several seconds* |
| Exceptions |  |
| Extensions |  |
| Dependent UC |  |

*4.1.3 Use case “apply for taxi registration”*

|  |  |
| --- | --- |
| Use case ID | TAXI3 |
| Name | *Apply for* Taxi /Moto-Taxi *registration* |
| Goal | *Add new taxi or Moto-Taxi to the system* |
| Participating actors | Taxi /Moto-Taxi and server |
| Precondition |  |
| Main scenario | 1. *Before starting to work as taxi, taxi needs to apply for taxi registration* |
| Exceptions | *Taxi is already registered in the system*  *Taxi registration is rejected* |
| Extensions |  |
| Dependent UC |  |

**Server UC**



*4.2.1 Use case “manage queues”*

|  |  |
| --- | --- |
| Use case ID | SERVER1 |
| Name | *Manage queues* |
| Goal | *Keep the queues for every city zone updated* |
| Participating actors | Server |
| Precondition | Change in the queue |
| Main scenario | 1. *When the taxi enters a new city zone, it is automatically put at the end of the virtual queue of that zone*  2. *Whenever a taxi from the queue goes to “busy” or “off duty” status,*  *it is removed from the queue*  3. *The server notifies the taxi about its position in the queue* |
| Exceptions |  |
| Extensions |  |
| Dependent UC |  |

*4.2.2 Use case “determine taxi zone”*

|  |  |
| --- | --- |
| Name | *Determine taxi zone* |
| Goal | *Assigns taxi to the appropriate zone* |
| Participating actors | Server and taxi |
| Precondition | Taxi has sent its GPS location |
| Main scenario | 1. *Server receives the GPS location of the taxi*  2. *Server determines the zone in which taxi currently is*  3. *Server sends the zone number to the taxi* |
| Exceptions |  |
| Extensions |  |
| Dependent UC | TAXI2 |

*4.2.3 Use case “determine customer location”*

|  |  |
| --- | --- |
| Use case ID | SERVER3 |
| Name | *Determine customer location* |
| Goal | *Determine the city zone of the customer* |
| Participating actors | Server |
| Precondition | Customer has made an order for the taxi |
| Main scenario | 1. *Server receives the customer order*  2. *Server determines the zone of the customer* |
| Exceptions |  |
| Extensions |  |
| Dependent UC | CUSTOMER1 |

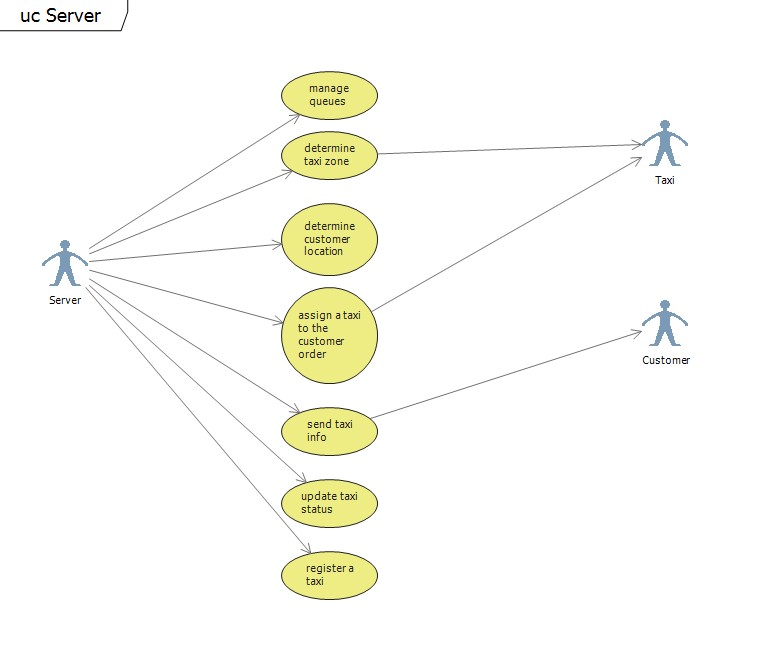
*4.2.4 Use case “*assign a taxi to the customer order*”*

|  |  |
| --- | --- |
| Use case ID | SERVER4 |
| Name | *Assign a taxi to the customer order* |
| Goal | *Dispatch the first taxi in the queue to the customer* |
| Participating actors | Server and taxi |
| Precondition | The server has determined the customer zone |
| Main scenario | 1. *Server sends the customer order to the first taxi of the appropriate queue* |
| Exceptions | *Taxi refuses to take the order*  *There is no taxis in the queue* |
| Extensions |  |
| Dependent UC | SERVER3 |

*4.2.5 Use case “*send taxi info*”*

|  |  |
| --- | --- |
| Use case ID | SERVER5 |
| Name | *Send taxi info* |
| Goal | *Inform the customer about the taxi which will pick him up* |
| Participating actors | Server and customer |
| Precondition | The order has been assigned to a taxi |
| Main scenario | 1. *Server sends the information about taxi and ETA to the customer* |
| Exceptions |  |
| Extensions |  |

**Server UC**



*4.2.1 Use case “manage queues”*

|  |  |
| --- | --- |
| Use case ID | SERVER1 |
| Name | *Manage queues* |
| Goal | *Keep the queues for every city zone updated* |
| Participating actors | Server |
| Precondition | Change in the queue |
| Main scenario | 1. *When the taxi enters a new city zone, it is automatically put at the end of the virtual queue of that zone*  2. *Whenever a taxi from the queue goes to “busy” or “off duty” status,*  *it is removed from the queue*  3. *The server notifies the taxi about its position in the queue* |
| Exceptions |  |
| Extensions |  |
| Dependent UC |  |

*4.2.2 Use case “determine taxi zone”*

|  |  |
| --- | --- |
| Name | *Determine taxi zone* |
| Goal | *Assigns taxi to the appropriate zone* |
| Participating actors | Server and taxi |
| Precondition | Taxi has sent its GPS location |

|  |  |
| --- | --- |
| Main scenario | 1. *Server receives the GPS location of the taxi*  2. *Server determines the zone in which taxi currently is*  3. *Server sends the zone number to the taxi* |
| Exceptions |  |
| Extensions |  |
| Dependent UC | TAXI2 |

*4.2.3 Use case “determine customer location”*

|  |  |
| --- | --- |
| Use case ID | SERVER3 |
| Name | *Determine customer location* |
| Goal | *Determine the city zone of the customer* |
| Participating actors | Server |
| Precondition | Customer has made an order for the taxi |
| Main scenario | 1. *Server receives the customer order*  2. *Server determines the zone of the customer* |
| Exceptions |  |
| Extensions |  |
| Dependent UC | CUSTOMER1 |

*4.2.4 Use case “*assign a taxi to the customer order*”*

|  |  |
| --- | --- |
| Use case ID | SERVER4 |
| Name | *Assign a taxi to the customer order* |
| Goal | *Dispatch the first taxi in the queue to the customer* |
| Participating actors | Server and taxi |
| Precondition | The server has determined the customer zone |
| Main scenario | 1. *Server sends the customer order to the first taxi of the appropriate queue* |
| Exceptions | *Taxi refuses to take the order*  *There is no taxis in the queue* |
| Extensions |  |
| Dependent UC | SERVER3 |

*4.2.5 Use case “*send taxi info*”*

|  |  |
| --- | --- |
| Use case ID | SERVER5 |
| Name | *Send taxi info* |
| Goal | *Inform the customer about the taxi which will pick him up* |
| Participating actors | Server and customer |
| Precondition | The order has been assigned to a taxi |
| Main scenario | 1. *Server sends the information about taxi and ETA to the customer* |
| Exceptions |  |
| Extensions |  |

Dependent UC SERVER4

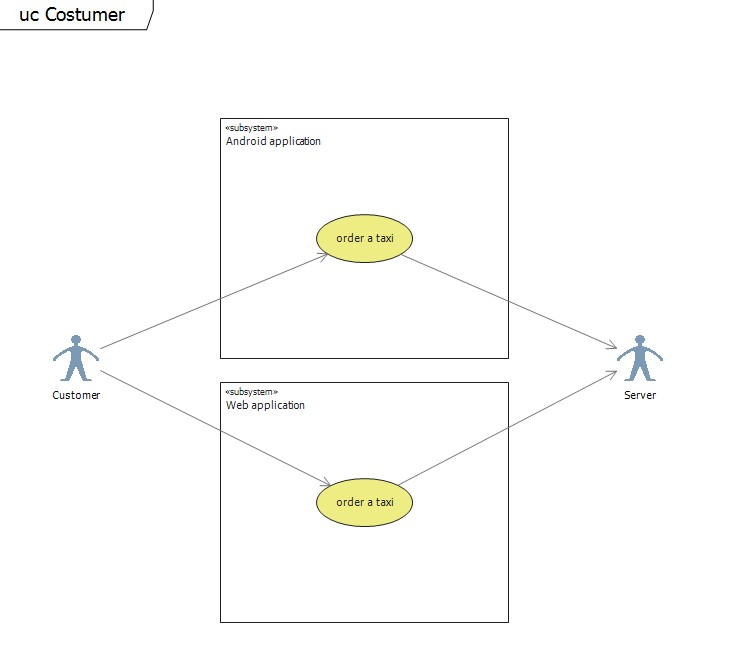
*4.2.6 Use case “*update taxi status*”*

|  |  |
| --- | --- |
| Use case ID | SERVER6 |
| Name | *Update taxi status* |
| Goal | *Update the status of the taxi on the server* |
| Participating actors | Server |
| Precondition | Taxi changed its status |
| Main scenario | 1. *Server receives the status change from taxi*  2. *Server updates the queue* |
| Exceptions |  |
| Extensions |  |
| Dependent UC | TAXI1 |

*4.2.7 Use case “*register a taxi*”*

|  |  |
| --- | --- |
| Use case ID | SERVER7 |
| Name | *Register a taxi* |
| Goal | *Add new taxi to the system* |
| Participating actors | Server |
| Precondition | Taxi had sent the application for registration |
| Main scenario | 1. *Server receives the application*  2. *Server adds taxi to the system* |
| Exceptions | *The taxi is already in the system*  *The application is rejected* |
| Extensions |  |
| Dependent UC | TAXI3 |

**4.3 Customer UC**



*4.3.1 Use case “order a taxi”*

|  |  |
| --- | --- |
| Use case ID | CUSTOMER1 |
| Name | *Order a taxi* |
| Goal | *Order a taxi to customer location* |
| Participating actors | Server and customer |
| Precondition |  |
| Main scenario | 1. *Customer clicks the button for ordering a taxi*  2. *Application sends the GPS coordinates to the server* |
| Exceptions |  |
| Extensions |  |
| Dependent UC |  |

*4.3.2 Use case “order a taxi”*

|  |  |
| --- | --- |
| Use case ID | CUSTOMER2 |
| Name | *Order a taxi* |
| Goal | *Order a taxi to customer location* |
| Participating actors | Server and customer |
| Precondition |  |
| Main scenario | 1. *Customer uses web application to select his/hers position on the map*  2. *The position is sent to the server* |
| Exceptions |  |
| Extensions |  |
| Dependent UC |  |
|  |  |

**5. Requirements Definition**

**5.1 Requirement Group Definitions**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Identification** | **Requirement Group** | **Rem.** |  |
| CA | | Customer application |  | |
| TA | | Taxi application |  | |
| SER | | Server |  | |
| NFR | | Non-functional requirements |  | |

**5.2 Requirement Sources**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Source** | **Description** | **Rem.** |  |
| CTM | | Customer |  | |
| SYS | | System |  | |
| DEV | | Developer |  | |

**5.3 Requirement definitions**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identity** | **Sta**  **tus** | **Prio**  **rity** | **Description** | **Sour**  **ce** |
|  |  | Android application for customers |  |
| CA-1 | I | 1 | User can send request for taxi | CTM |
| CA-2 | I | 1 | User can receive information about taxi that is assigned to pick him up and the estimated time of arrival | CTM |
|  |  |  | Android application for taxis |  |
| TA-1 | I | 1 | User sends information about his position periodically | SYS |
| TA-2 | I | 1 | User can receive request to pick up customer | CTM |
| TA-3 | I | 1 | User can receive information about zone he is currently in | CTM |
| TA-4 | I | 1 | User can change its status | SYS |
| TA-5 | I | 2 | User can accept or decline call | CTM |
| TA-6 | I | 2 | User can see if connection with server is established | DEV |
| TA-7 | I | 2 | User can register taxi | SYS |
|  |  |  | Server |  |
| SER-1 | I | 1 | Server can determine client location | SYS |
| SER-2 | I | 1 | Server can determine zone based on position | SYS |
| SER-3 | I | 1 | Server can notify customer | CTM |
| SER-4 | I | 1 | Server can manage queues | SYS |
| SER-5 | I | 1 | Server can assign taxi to customer | CTM |
| SER-6 | I | 1 | Server can update taxi status | SYS |
| SER-7 | I | 2 | Server can recognize taxis by its id | SYS |
| SER-7 | H | 3 | Server can calculate which taxi is nearest to certain position | DEV |
|  |  |  | Non-functional requirements |  |
| NFR-1 | I | 1 | System has to be reliable always | CTM |
| NFR-2 | I | 1 | Mobile applications are developed for Android OS | SYS |
| NFR-3 | I | 1 | Mobile applications should be user friendly | DEV |
| NFR-4 | I | 1 | System should be able to work with large number of customers | CTM |

Requirement status:

*I = initial* (this requirement has been identified at the beginning of the project),

*D = dropped* (this requirement has been deleted from the requirement definitions),

*H = on hold* (decision to be implemented or dropped will be made later),

*A = additional* (this requirement was introduced during the project course).

Requirement priority:

1-high priority

2-medium priority

3-low priority

*5.3.1 Change Log*

**Identity Acti on**

**Date Comments**

*D = dropped* (this requirement has been deleted from the requirement definitions),

*H = on hold* (decision to be implemented or dropped will be made later),

*A = added* (this requirement was introduced during the project course).

R = resurrected (dropped or on hold requirement was reactivated)

**6. Future Development**

Initial version of product will consist only of features needed for core functionality. Additional features were considered and discussed, but will be added after the main goals are accomplished. Some of these additional features are:

* Advertisements system
* Custom Navi (show directions to taxi driver)
* Connect Application with social Networks
* **“Request my favorite driver”** – ok, so all of us develop certain likings towards some peeps. Some users may want to hire their “regular” for a drive if s/he is working.
* **Voice commands** – when you are too buzzed to type.
* **Waitlist** – during peak hours users can add themselves on a waiting list, rather than refreshing the app over and over again in search for a ride.
* Final product could be developed for other operating systems besides Android, such as iOS or WP7.
* **Free in-app calls** – can be added via VoIP integration, though that may increase the overall taxi app development price.
* **“Split the bill”** – Sharing the ride costs with friends.
* **SMS Gateway** – Custom SMS Gateway.