Table of Contents

1. Introduction 1

1.1 Description of the problem 1

1.2 Goals 2

1.3 Domain Properties / Assumptions 2

1.4 Proposed System 3

2. Actors 3

3. Requirements 3

3.1 Functional Requirements 3

3.2 Non Functional Requirements 4

4. Specifications 7

5. Scenarios 8

6. UML Models 8

6.1 Use Case Diagram 8

6.2 Use Case Description 9

6.3 Sequence Diagrams 9

6.4 StateChart Diagrams 9

6.5 Class Diagram 9

7. Alloy 9

7.1 Modeling 9

7.2 Alloy Analyzer 9

7.3 Worlds Generated 9

8. Used Tools 10

9. Hours of Work 10

# Introduction

## Description of the problem

Our object is to project a system to optimize the taxi service of Big\_City, simplifying the access of passengers to the service and guaranteeing a fair management of taxi queues. It will be composed of a (web application) and a mobile application allowing users to request a taxi and informing them about the code of the incoming taxi and the waiting time. The system is constantly updated about the availability of the taxi drivers and call confirmations in order to maintain the fairness of queues.

The city is divided in zones (2 km² each) and there is a one-to-one correspondence between them and the queues. NELLE CODE CI SONO SOLO AVAILABLE

-

-

STAKEHOLDER – RAFFAELA MIRANDOLA

-

## Goals

When the system is online, it has to guarantee all these functionalities:

* create a taxi ride request
* create a taxi ride reservation
* confirm/reject a taxi ride
* update taxi driver availability
* management of taxi queues
* management of requests and reservations
* scalability to additional services

## Domain Properties / Assumptions

This is a list of some domain properties and assumptions that we think has to be true:

* the taxi service is entirely controlled by the government of Big\_City, that owns the taxis and assigns an username and password to taxi drivers to let them access the service
* Every taxi has an unique code and must have an embedded Android Auto operating system, with myTaxiService and Google Maps (maps already downloaded and available offline) as the only usable and preinstalled applications
* the city is divided in zones and the range of each one is a precondition
* when taxi drivers inform the system that their taxi is available, they cannot move from the zone where they have declared availability
* taxis have an on-board GPS that always returns their accurate location to the system
* when a passenger requests a taxi, the GPS embedded in the smartphone always returns their accurate location to the system
* there is a number of taxis large enough to assure that the waiting time of all request is 20 minutes at most
* when a taxi driver confirms a reservation, it will always reach the passenger in 10 minutes at most
* when a passenger reserves a taxi, they will always give the right information about meeting date, departure and arrival location, meeting time
* origin and destination of ride requests and reservations must be within Big\_City
* when a user requests or reserves a taxi, the maximum number of passengers of the ride must be 5 included them

## Proposed System

The mobile application used by passengers and taxi drivers is the same but its user interface changes after the login screen since the functionalities needed from them are different.

Interfaccia diversa su macchina screen, mobile e web app per utente, gps installato sulla macchina, codice identificativo univoco per ogni macchina visibile esternamente, ANDROID AUTO APP IOS ANDROID web app

# Actors

The actors involved in this project are four:

* **System:** it has to provide the user with taxi ride request and reservation functionalities and the taxi driver with ride confirmation and rejection functionalities
* **Guest:** a person that opens either the web application or the mobile application for the first time, facing its login page. They are not already registered in the system, so they have to sign up into the service completing a registration form before accessing the service
* **User:** a person that is already registered as a user of the service, they can log in into the system, request or reserve a taxi ride, check the list of their booked taxis
* **Taxi driver:** a person that is employed by the government to drive a taxi in Big\_City. It can inform the system about the availability of their taxi

# Requirements

## Functional Requirements

**GUEST:**

* Sign up in the web application or mobile application by filling up a form with username and password

**USER:**

* Log into the web application or mobile application with username and password
* Create a ride request
* Create a ride reservation
* Cancel a ride reservation
* Check the list of booked taxis

**TAXI DRIVER**:

* Confirm a ride(reservation or request)
* Reject a ride(reservation or request)
* update their availability

**SYSTEM**

* propose a ride to a taxi
* confirm request or reservation to the user with related information
* insert, remove or move a taxi from a queue
* update availability of taxis

## Non Functional Requirements

1. **USER INTERFACE**

The user interface of the web application and the mobile application for the users, and of the Android Auto application for the taxis must be user-friendly. To better explain it, we realized mockups for Android smartphones and Android Auto.

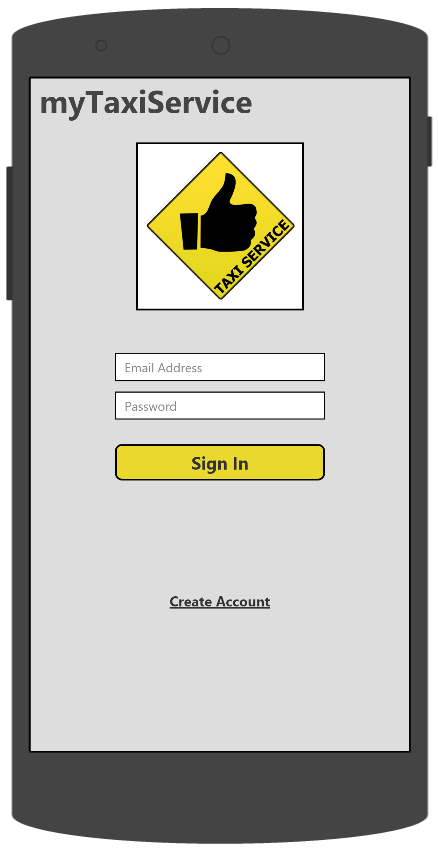
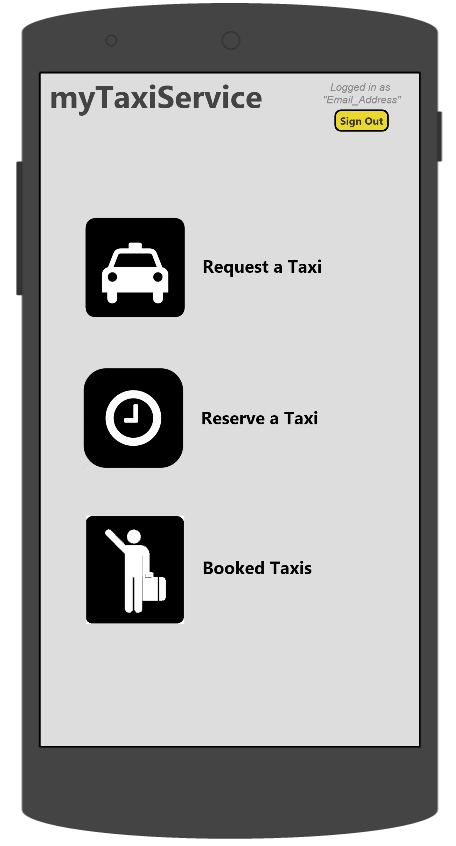
After downloading, installing and running the application from the Google Play Store, the user is facing the Login Page and is requested to insert their Email Address and Password, if already registered. Else, they has to tap on Create Account, action that leads to the registration page where desired Email Address and Password have to be inserted. If correct information are inserted (i.e. they are in the database), both Sign In and Sign Up buttons, when tapped, lead to the Home Page.

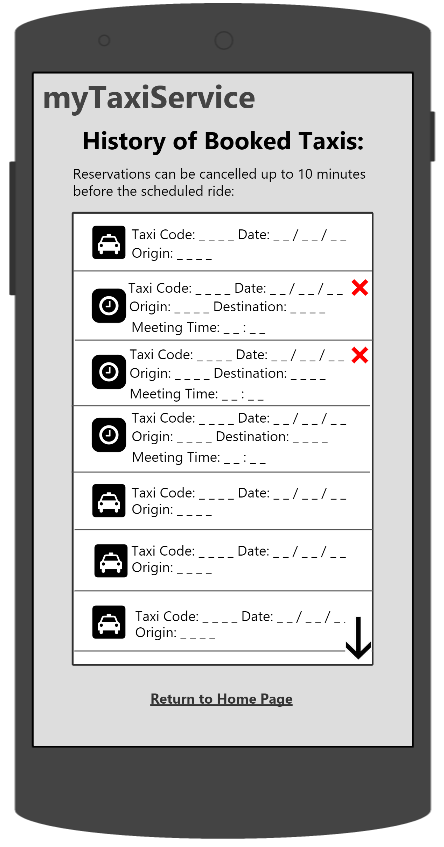
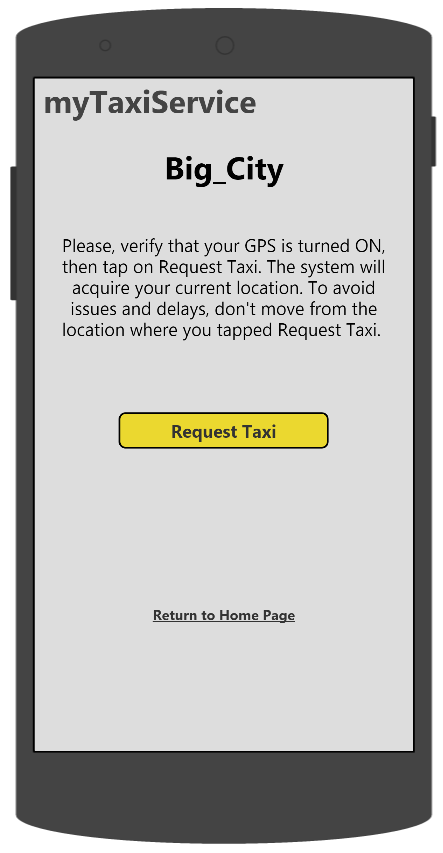
When the user reaches the Home Page, they can Request a Taxi, Reserve a Taxi, check the list of the Booked Taxis and Sign Out from the service and return to the Login Page.

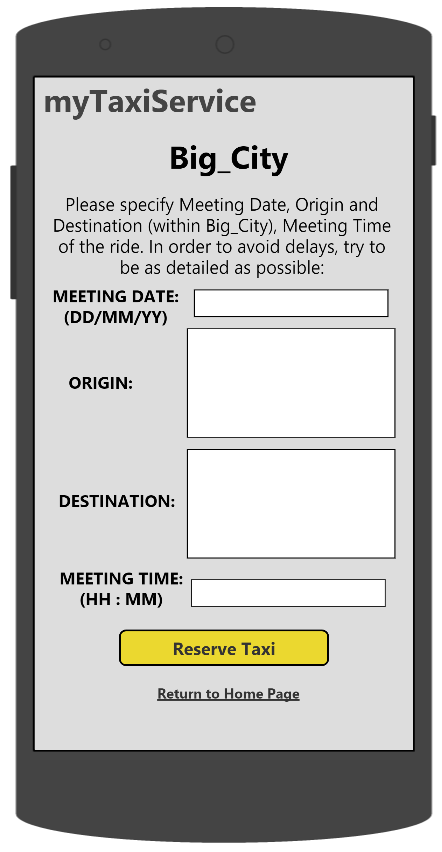
In the Request a Taxi page, the user is requested to turn on the GPS on their smartphone, in order to let the system automatically acquire their current position and consider it as the meeting location. This is the only difference with the web version of the application, since in that case the user is required to specify their location in a form (since laptops do not have GPS module).

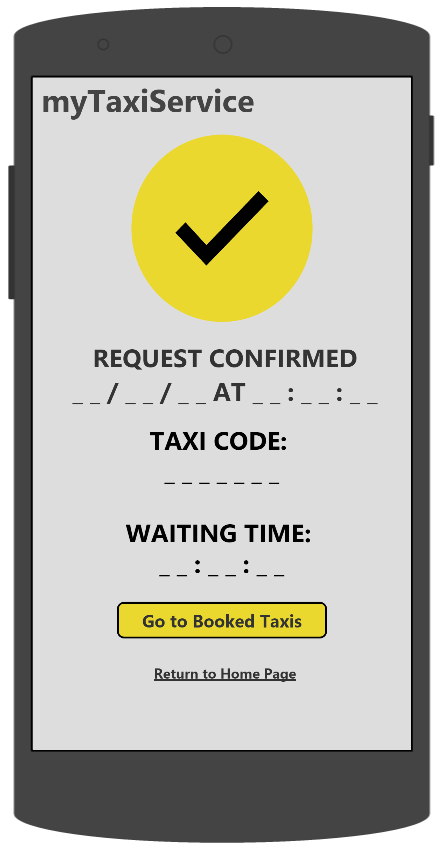
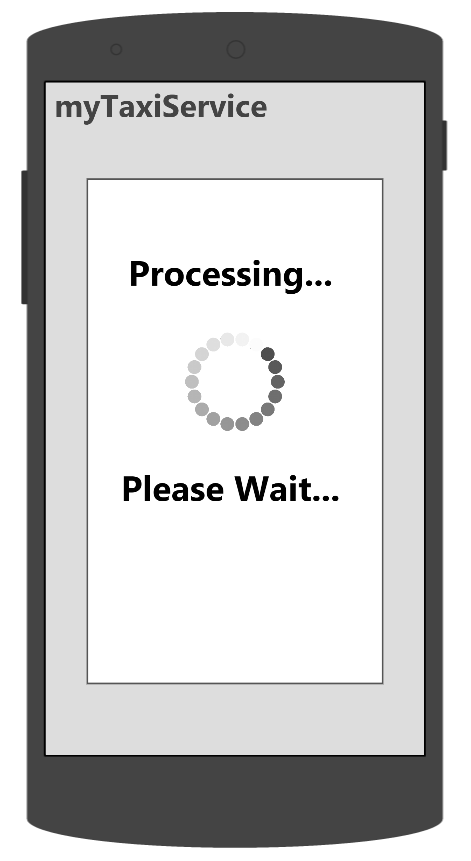
When the Request Taxi button is tapped, the Request Confirmed page appears after the processing of the request (Processing Page), with all the needed information. Then, the user can go to the Booked Taxis page with the relative button, or return to the Home Page. If the user changes their mind and do not want to request a taxi anymore, they can tap on Return to Home Page instead of Request Taxi.

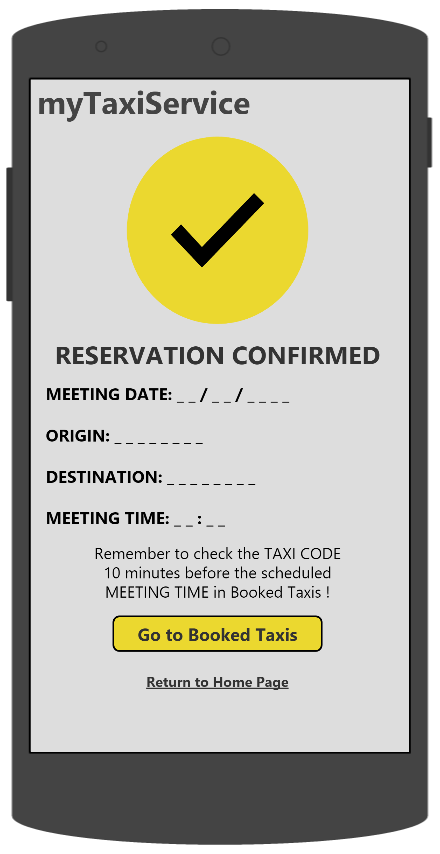
In the Reserve a Taxi page, the user has to fill the required fields with all the requested details and then tap on Reserve Taxi (or on Return to Home Page if they are not convinced also in this case). As the previous case, the user has to wait the processing of the request (Processing Page) and then the Reservation Confirmed page with detailed information appears. Also in this case, the user can then go to the Booked Taxis page, or Return to Home Page. The Booked Taxis page contains a scrollable history of all the requested and reserved taxis with all the related information. The entries of the reserved taxis have also the possibility to be cancelled by tapping on the red cross that appears only up to 10 minutes before the reserved ride. Also for this page, there is the possibility to Return to Home Page.

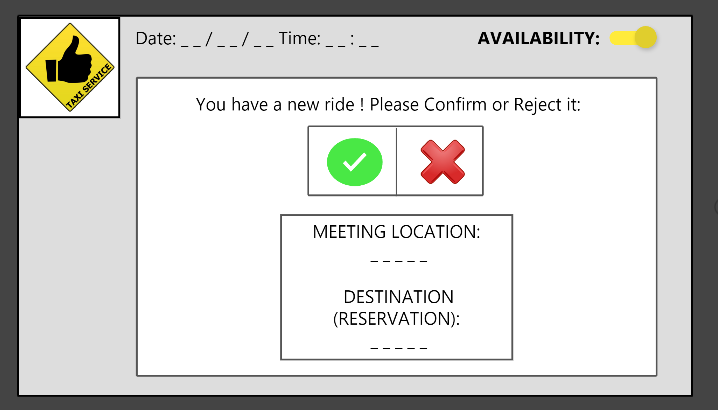
****

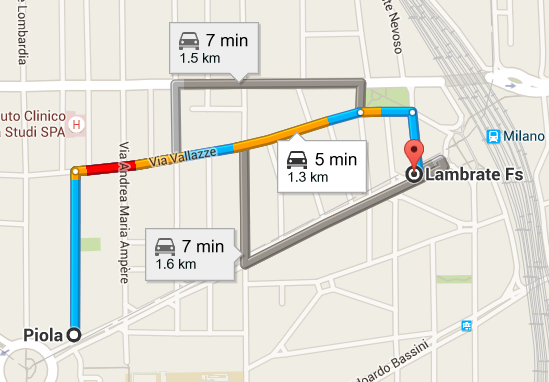
****

****

****

****

****The Android Auto user interface of myTaxiService is designed to meet all the functionalities needed by taxi drivers. It is the default application that starts when the engine of the car is turned on and the driver cannot close it. When a driver begins their daily shift, they have to insert the username and password the government has provided them on a Login Page, in order to access the service and include their taxi into the system. Then, the driver is redirected to the Home Page (shown below), which provides them with the possibility to switch on and off their availability. If the availability is off the central screen remains clear, else the system can allocate a ride to the driver, which is displayed with the possibility of confirmation and rejection along with the location of the passenger (and also the destination in case the ride is reserved, else the field is empty). If the driver rejects the screen turns clear again, else Google Maps is automatically opened with the directions to the meeting location. When the passenger is aboard, the driver searches for the directions to the desired destination (declared at boarding time in case of requested ride). When the taxis reaches destination, the driver can close Google Maps and then the central screen of the user interface turns clear again.

****

1. **DOCUMENTATION**

These are the documents about myTaxiService that will be released:

• **RASD**: contains the description of the scenarios, the use cases that describe

them, and the models describing requirements and specification

• **DD**: contains a functional description of the system, and any other view

useful to provide

1. **SOFTWARE SYSTEM ATTRIBUTES**

* **Reliability:** System shall have an availability of 99.99% (four nines), which implies a 52.56 minutes downtime per year. The reliability of the system is strictly related to the reliability of the server it runs on.
* **Availability:** The system is running 24/7 so that a user can always request or reserves a taxi.Any kind of updates must not stop the normal operations.
* **Capacity:** the system should be able to manage 1000 requests/second and manage at least 3000 connected users (passengers and taxi drivers) at a given time
* **Security:** All the communications between server and clients must be protected by strong encryption using the SSL protocol. Users' passwords be hashed and salted and then stored in the database
* **Portability:** The back-end server software will be written in Java. It must run on every platform that supports the Java Virtual Machine. The web application shall run on every modern browser. The mobile application for users must be supported by the last 2 major versions of Android and iOS. The mobile application for taxis must be supported by Android Auto
* **Robustness:** If the taxi driver loses the GPS connection for a while, the ride is not canceled, available taxis remain in the queues and updates are simply suspended

# Specifications

**La lista delle code attive è una lista in cui saranno presenti tutte le corse attive, ovvero tutte le corse che non sono state ancora assegnate e che devono essere assegnate. La lista funziona in base alla priorità, ci saranno due priorità una alta e una media.Quando avremo due corse con la stessa priorità la lista si comporterà come una fifo.**

**La lista delle reservation è una lista in cui sono presenti tutte le reservation.**

**Il Sistema gestirà una richiesta alla volta e prenderà la corsa dalla lista delle corse attive.**

**Il Sistema per ogni zona avrà una coda in cui saranno presenti I taxi available. La coda è di tipo FIFO.**

1. **taxi ride request**

**L’utente se userà la web application dovrà inserire la posizione, mentre dalla app il Sistema acquisirà la posizione attuale da GPS.**

**Il Sistema permette di avere solamente una corsa attiva alla volta. Se l’utente prova a effettuare una request quando ha una corsa attiva il Sistema darà errore.**

**Quando la corsa sarà la prima nella lista il Sistema proporrà la richiesta al primo taxi driver della relativa coda. Se non ci sarà nessun taxi nella zona oppure se tutti I taxi presenti rifiuteranno la corsa il Sistema cercarà un taxi nella coda più vicina. Il Sistema memorizza il primo taxi nella coda a cui propone la corsa per evitare il loop dei taxi che rifiutano da mettere o è troppo specifico??? Una volta trovato il taxi il Sistema invierà allo user il codice del taxi e il suo waiting time.**

1. **taxi ride reservation**

**L’utente dovrà inserire destination origne e tempo.**

**Se un utente effettua una request entro due ore il Sistema gli dirà errore e li farà reinserire I campi.**

**Quando l’utente effettua una reservation il Sistema la inserirà nella lista delle reservation e confermerà all’utente la reservation. Dieci minuti prima il Sistema prende la reservation e la sposta nella lista delle code attive impostando la massima priorità portandola in cima alla lista. Il Sistema utilizza la stessa tecnica della request per assegnare il taxi. Una volta trovato il taxi informerà l’utente del codice del taxi che sta venendo a prenderlo.**

**Il Sistema permette all’utente di cancellare una reservation fino a 10 minuti prima dell’incontro, ovvero fino a quando sarà presente nella lista delle reservation.Quando un utente cancella una reservation verrà cancellata dalla lista delle reservation..**

1. **Confirm/Reject a taxi ride**

**Il Sistema permette al taxi driver di rifiutare o accettare una richiesta. Quando il taxi driver accetta la richiesta verrà visualizzato il percorso, se rifiuta il Sistema spostera il taxi in ultima posizione nella relative coda.**

**Il taxi driver avrà un tempo pari a 10 secondi per poter rispondere alla richiesta, se scade il tempo il Sistema vedrà la richeista come un rifiuto**

1. **Update taxi driver availability**

**Il Sistema permette al taxi driver di impostare la sua disponibilità. Quando un taxi driver accetta una richiesta il Sistema metterà il taxi non available. Dopo aver portato il passeggero sceglie una apposita area di sosta per I taxi e si dichiarerà nuovamente available. Un taxi driver può togliere la sua disponibilità ogni qualvolta vuole.**

1. **Management of taxi queues**

**Quando un taxi diventa disponibile il Sistema lo inserirà nella coda della relativa zona prendendo la sua posizione dal GPS.**

**Quando un taxi diventa non dispobile il Sistema toglierà il taxi dalla relative coda.**

**Quando un taxi rifiuta una corsa il Sistema lo sposterà in ultima posizione.**

1. **Scalability to additional services**

The system provides API for developers

1. **Registrazione**

**Il Sistema permette a un utente di registrarsi una sola volta con la stessa email. Se l’utente prova a crearne due il Sistema darà errore.**

1. **Booked list**

**Il Sistema permette all’utente di visualizzare tutte le sue corse precendenti oltre a quelle pendenti.**

# Scenarios

Ape con i soci:

Christian è andato a fare un ape con i soci e vuole ritornare a casa. He takes his phone, open the app MyTaxiService and do the log in into the system. Christian selezionerà request a taxi e il sistema acquisirà la posizione e avvertirà christian che sta cercando un taxi. Luigi è un taxi driver che ha dato la sua disponibilità e viene avvertito dal sistema che c’è una corsa per lui. Luigi accetta la richeista e va a prendere Il cliente. Christian riceve una notifica con il codice del taxi di Luigi e il waiting time di 8 minuti. Christian si dimentica il codice del taxi e guarderà nella booked list per vedere la sua request attiva. Luigi prende Christian e lo porta alla destinazione desiderata e subito dopo dedide di rimanere nella zona quindi rimette l’avaibility e si ferma ad aspettare.

Registrazione:

Simone sta tornando a casa a piedi quando gli viene in mente l’app MyServiceTaxis consigliata da Christian così decide di scaricare e installare l’app.Una volta aperta l’app Simone cliccherà su Sign up e inserirà I campi richiesti e darà la conferma. Il Sistema confermerà la sua registrazione e subito dopo Simone effettua una request. Il Sistema invia la richiesta di Simone a Pietro che sta parlando con un suo amico fuori dal taxi. Pietro non si accorge della chiamata e dopo 10 secondi scade il time out e la richiesta viene rifiutata in automatico. Allora sarà Giulio a ricevere la chiamata di Simone che accetterà e si avvierà per andare a prenderlo.

Update availability:

Marco è un taxista che dopo aver portaro un cliente andrà nella zona B e si metterà available ad aspettare.Dopo 15 minuti in cui non ha ricevuto nessuna corsa decide di mettersi non available e andare a fare pausa pranzo. Dopo aver finito la pausa Marco clicca su available e rimane ad aspettare. Dopo 15 minuti Il Sistema gli proporrà un passeggero. Marco nota che questo passeggero non è nella sua zona ma in quella accanto, accetta comunque. Dopo aver portato il passeggero Marco torna nella zona da cui era arrivato la richiesta si ferma e si mette disponibile.

Reservation 1

Jhon è andato in vacanza a Napoli ma sa già che farà ritorno il giorno X alle ore Y con il treno(sempre se non viene accoltellato prima). Così il giorno prima decide di effettuare una reservation, allora entrà dal suo computer in mytaxiservice e prenota un taxi alle ore Y del giorno X. Jhon riceverà subito la conferma della sua richiesta. Mezz’ora prima del waiting time Jhon viene informato che il treno è in ritardo di 2 oren così decide di cancellare la reservation. Il treno arriverà con 1 ore e 50 minuti di ritardo e una volta sceso Jhon effettua una request dalla sua app. Jhon è arrivato nell’ora di punta e il taxi gli viene assegnato dopo 7 minuti con un waiting time di 12 minuti.

Reservation 2

# UML Models

This section provides user interface design descriptions that directly support construction of user interface screens.

## Use Case Diagram

Detail the common behavior that all screens will have. Common look and feel details such as menus, popup menus, toolbars, status bar, title bars, drag and drop mouse behavior should be described here.

## Use Case Description

Illustrate all major user interface screens and describe the behavior and state changes that the user will experience.

## Sequence Diagrams

## StateChart Diagrams

## Class Diagram

|  |  |  |
| --- | --- | --- |
| **Label Name** | **Note** | **Source** |
| NB Specialist |  | SELECT UWUserID FROM PolicyGroup GROUP BY UWUserID ORDER BY PolicyGroup.UWUserID; |
| Inserted By | Change to be a dropdown containing the NB Cordinators (approve group).  Add to the BinderRequest table a new filed ‘NBCordinator’ | SELECT InsertBy FROM PolicyGroup GROUP BY InsertBy ORDER BY InsertBy;  Query active directory to return the NBCordinator group. |
| Binder Type |  | SELECT CodeToText.Code, CodeToText.Text FROM CodeToText WHERE (((CodeToText.TableDotField)='PolicyGroup.Type')) ORDER BY CodeToText.SortOrder; |
| Status | This will not be used. | SELECT PolicyStatus.Status, PolicyStatus.PolicyGroupApply FROM PolicyStatus WHERE (((PolicyStatus.PolicyGroupApply)<>0)) ORDER BY PolicyStatus.Status; |
| Insurance Carrier |  | SELECT InsCo.InsCo, InsCo.CompanyName FROM InsCo ORDER BY InsCo.InsCo, InsCo.CompanyName; |
| Main Rep | Will this be the ***selling*** or ***signing*** rep? | SELECT Rep.RepCode, [LName]+", " & [FName] AS Name, Rep.RepCode AS AcctgRepCode FROM Rep ORDER BY [LName]+", " & [FName]; |
|  |  |  |

# Alloy

## Modeling

## Alloy Analyzer

## Worlds Generated

# Used Tools

# Hours of Work