Table of Contents

1. Introduction 1

1.1 Description of the problem 1

1.2 Goals 1

1.3 Domain Properties / Assumptions 2

1.4 Proposed System 3

2. Actors 3

3. Requirements 3

3.1 Goals Requirements 3

3.2 Functional Requirements 4

3.3 Non Functional Requirements 4

4. Specifications 4

5. Scenarios 5

6. UML Models 5

6.1 Use Case Diagram 5

6.2 Use Case Description 5

6.3 Sequence Diagrams 5

6.4 StateChart Diagrams 5

6.5 Class Diagram 5

7. Alloy 6

7.1 Modeling 6

7.2 Alloy Analyzer 6

7.3 Worlds Generated 6

8. Used Tools 6

9. Hours of Work 6

# 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:

* registration of a new passenger
* create a taxi ride request
* reservation of a taxi ride
* confirm/reject a taxi ride request
* 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 employs the taxi drivers and owns the taxi
* the city is divided in zones and the range of each one is a precondition
* there are different FIFO availability queues, each one assigned to a zone
* there is a number of taxis in each zone large enough to assure that all requests and reservations can be fulfilled
* if a taxi driver declares that their taxi is available, then it can be inserted in a queue. Else, it cannot
* each zone has a preassigned range of taxi codes. The system provides taxi drivers only with requests coming from a zone only if the code of their taxi is inside the preassigned range of that zone
* taxi drivers can inform the system that their taxi is available only when they are inside their preassigned zone
* passengers can create taxi ride requests and reservations only after registration and when they are logged in into the system
* when an available taxi driver confirms a ride, their taxi is automatically excluded from the queue of their zone by the system
* taxis have an on-board GPS that always returns their accurate location to the system
* when a taxi driver confirms a request, it will reach the passenger in the shortest possible time
* when the system allocates a taxi to a reservation and the taxi driver confirms, it will reach the passenger in the shortest possible time according to the scheduled meeting time
* when a passenger requests a taxi, they will always give the right information about their current location
* when a passenger reserves a taxi, they will always give the right information about departure and arrival location and meeting time
* the system takes into account the GPS information from a taxi only when the driver declares that they are available
* taxis must return to their preassigned zone after completing a ride

## 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,

# 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 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 and request or reserve a taxi ride
* **Taxi driver:** a person that is employed by the government to drive a taxi in a preassigned zone in the X city. It can inform the system about the availability of their taxi

# Requirements

## Goals Requirements

1. **Registration of a new user**

* a sign up functionality is provided to the user by the system

1. **Create a taxi ride request**

* the system allows a user to create a taxi ride request

1. **Reservation of a taxi ride**

* the system allows a user to reserve a taxi

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

* the system allows a taxi driver to confirm a provided ride request
* the system allows a taxi driver to reject a provided ride request

1. **Update taxi driver availability**

* the system allows taxi drivers to give their availability

1. **Management of taxi queues**

* when a taxi becomes available the system stores its code in the queue
* if
* se taxi rifiuta richiesta finisce in fondo alla coda e la coda viene aggiornata

1. **Management of requests and reservations**

* send the request to the first taxi queueing in the zone where the request comes from until it is accepted
* inform the passenger about the code of the incoming taxi and the waiting time

1. **Scalability to additional services**

* The system provides API for developers

## Functional Requirements

**GUEST:**

* Sign up in the system by filling a form with personal information

**USER:**

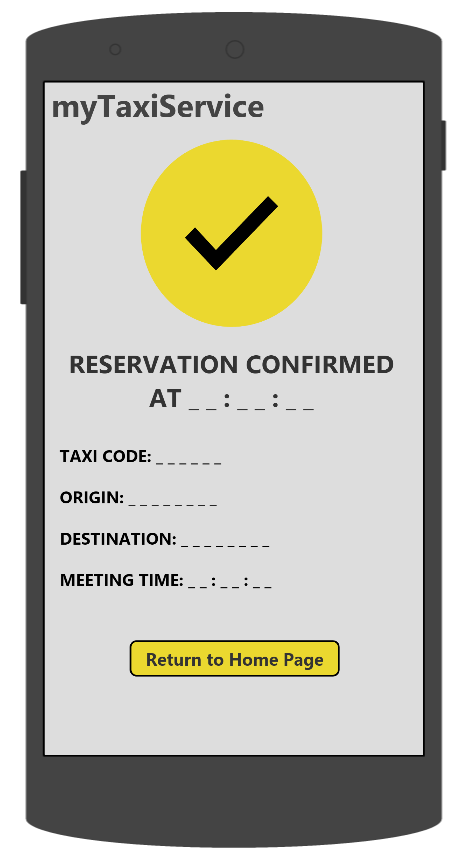
* Log into the system with their credential
* Create a request ride
* Create a reservation ride
* Receive by the system a notification with the waiting time and taxi code of their ride

**TAXI DRIVER**:

* Receive from the system a ride(request or reservation)
* Confirm a ride(reservation or request)
* Reject a ride(reservation or request)
* update their availability

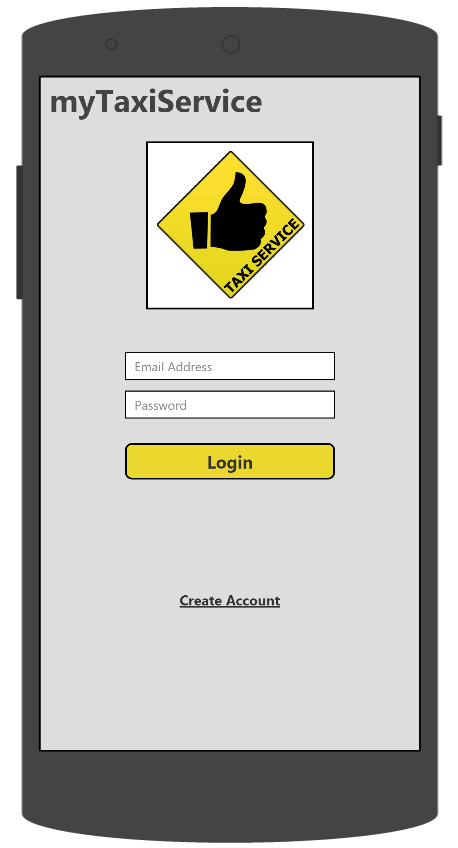
**SYSTEM**

* propose a ride request at the correct taxi driver(the first in the relative queue)
* inform the user about the taxi code and the waiting time of their ride
* insert in the bottom of a relative queue a taxi when becomes available
* update a availability of a taxi when the relative taxi driver accept a ride(request or reservation)
* move the taxi at the last position of the queue if the taxi driver on board it reject a ride(request or reservation)
* calculate the waiting time of a ride request from a gps position of a taxi and the location of a ride request
* propose a ride reservation at the correct taxi driver(scanning the queue and find the first taxi that riesce ad andare a prenderlo)

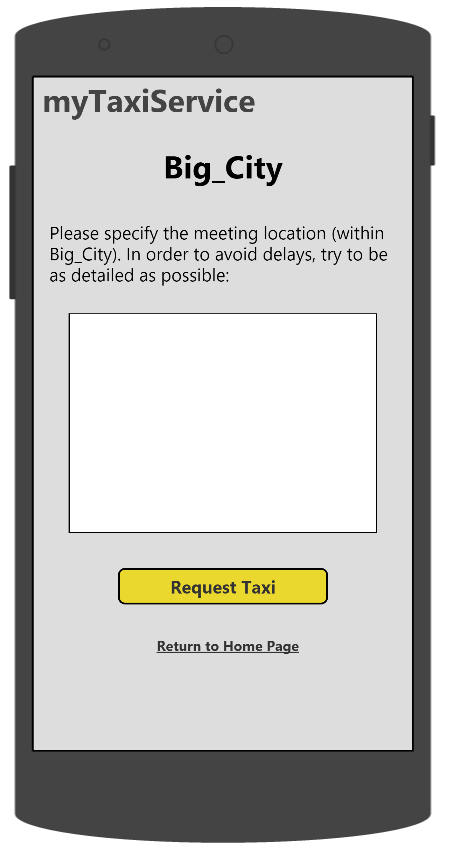


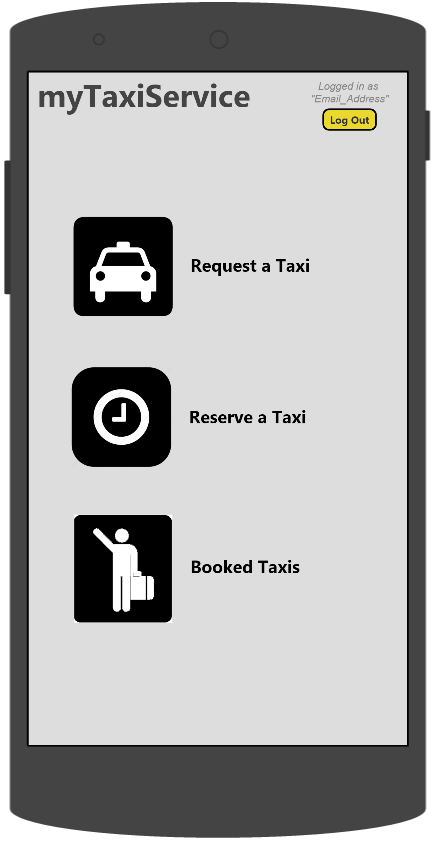
## Non Functional Requirements

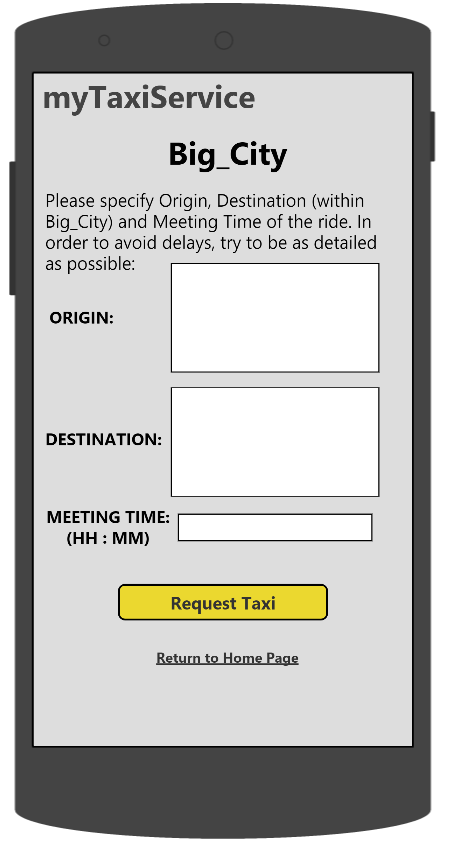
1. **USER INTERFACES**

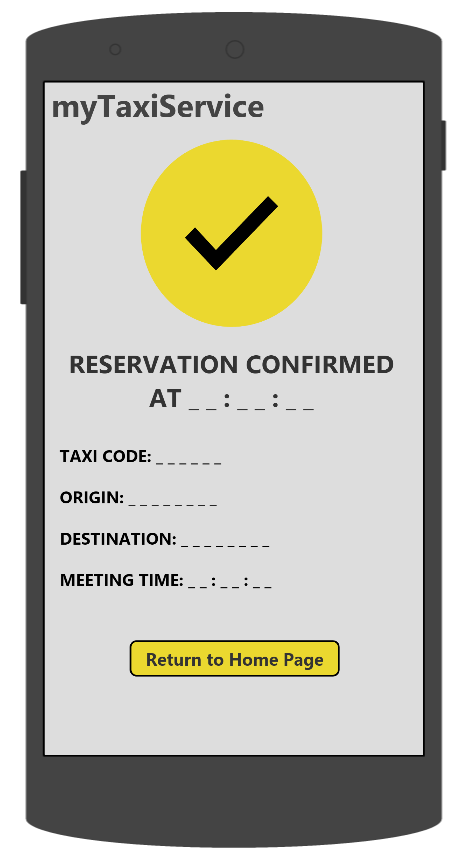
****

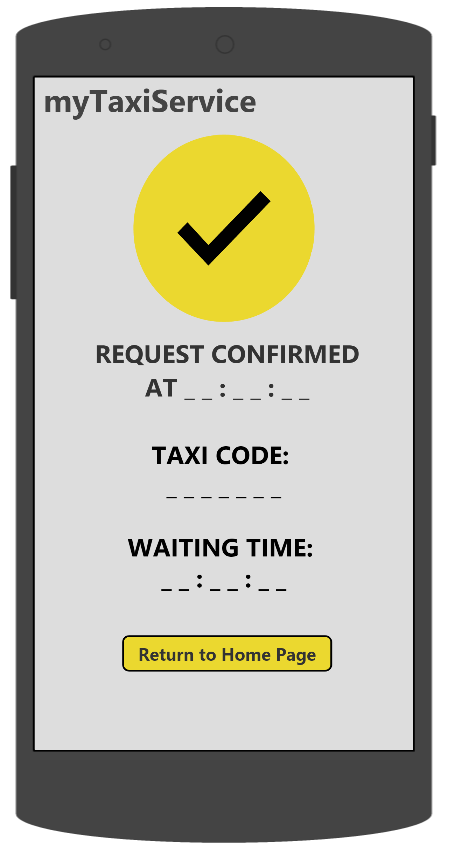


****



****





1. **DOCUMENTATION**
2. **ARCHITECTURAL CONSIDERATIONS**

# Specifications

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table Name** | **Field Name** | **Data Type** | **Allow Nulls** | **Field Description** |
| BinderRequest | SellingRep | Varchar(50) |  | Get this field from BankPlan.ProposalRepCode  But use actual name (Roy Pinnell) |
| BinderRequest | SigningRep | Varchar(50) |  | Get this field from Bankplan.SigningRepCode  But use actual name (Roy Pinnell) |
| ProcessBP | ParentProcessID | Int | Yes | This will tie a subprocess to a process |
| BinderRequest | ProjectedWireDate | Date |  | The earliest Policy.ProposalWireDate of all included policies in the scenario. |
|  |  |  |  |  |
|  |  |  |  |  |

*For each field change (such as data types, required/not required, or renaming), please complete a row of the following table. (Insert additional rows as needed.)*

|  |  |  |
| --- | --- | --- |
| **Table Name** | **Field Name** | **What to change?** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Scenarios

# 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