Software Requirements Specification

Mobile Cab Booking Application for Passenger

	Name	Role	Location	Revision Date
Prepared By	V.Srinivasamurthy	BA	Bangalore	22/06/15
Reviewed By				
Approved By				
Distribution List	Pruthvi Rao		**Version No.**	1

Purpose of Sign-off

The purpose of the sign-off on this document is to provide alignment and agreement to the content of baseline requirements and scope for the requirements to be considered in future project and/or release plans.

Sign-Off	Typed Name	Date Approved
Executive Sponsor		
Process Owner		
Business Analyst		

Email Approvals/Comments (if any)

REVISION HISTORY

#	Version #	Release Date	Name	Description
1	*1*	*22/06/15*	*V.Srinivasamurthy*	**SRS document for Mobile Solution.**

TABLE OF CONTENTS

1 INTRODUCTION

The Mobile Cab booking Application is a mobile app for booking cabs for Cab Company. The application described here is intended for mobile savvy users who would be interested in booking a Cab on the go. It is targetted to work on three main Operating systems in the market namely Windows,Android and Apple IOS. The application would make use of Location,Mapping and geocoding services to show cabs and passenger location on a interactive map along with the location address of the passenger. It would alow the user to to interactively locate cabs in his vicinity and book cabs of his choice and at his prefered time.It would also allow the Passenger to book a Cab immediately from his Location with turn ou ttime to be thirty minutes.It would also use the latest mobile Wallet features to ease the process of payment and build credibility between the Passenger and the Cab Company.

The applications would be developed using Client-server architecture, where the business Logic and Database access are exposed as web services .The Business Layer consists of Passenger,Driver and Admin modules along with COTS software modules.The module interfaces would be exposed as webservices consumed by the application.

1.1 PURPOSE

The purpose of this document is to provide the stakeholder a comprehensive view of all their requirements in developing mobile cab Booking application. The motive is to develop mobile application targetting passengers who would like to book cab using mobile devices.

1.2 SCOPE

This project is focussed on implementing a Mobile Cab booking application for Passengers.The mobile applications would target Windows,Apple IOS and Android mobile platforms.

2.1 IN SCOPE

The Features of Passenger Mobile Application are as follows:

A) Registration and Login Features.

B) Change Password.

C) Recover Password for a registered Passenger.

D) Show Cab and Passenger Location(s) on Interactive Map.

E) Abilty to relate the Passenger location(GPS coordinates) to an address and vice-versa.

F) Show Cabs based on cab type available in the vicinity of the Passenger on the Map.

G) Track the Location of a booked cab from its current Location to Passenger Pickup Point.

H) Book Cab immediately or at a later date based on Passenger Pickup and Destination location.

I) Cancel a Cab booking.

J) Add Pickup Locations to favourite locations for later retrieval.

K) Choose Pickup locations by the following options:

1. Scroll on map to choose PickUp Location.
2. Input PickUp Location.
3. Choose from Favourite or Recommended Location in the City.

L) Choose type of Cab and view their avialabilty.

M) View Fare List for a given Service Area.

N) Enroll Passsenger to Mobile Wallet feature.

O) Pay for Travel using Mobile Wallet.

P) Ability to store data related to Passenger in the mobile and in the datastore on the server.

Q) Abilty to refill wallet using Debit,Credit or Net Banking facility.

R) Recieve SMS, e-mail and Push notifications from the Cab Company with regard to travel.

S) Ability to Send Feedback about travel.

2.2 OUT OF SCOPE

The following items have been discussed during analysis, but a decision has been made to exclude them from the scope of this project:

I. Any new feature additions that arise later in the project will be considered out of scope for this project.

II. Extension of existing Passenger,Driver and Admin Modules on the server side are out of scope. The Data access and extra interfaces requried would be developed on the Client Server by our recommendation.

III. The Wallet features would used as it is and the Interfaces it exposes, the web services exposed by the Wallet APIs would be used as part of the Application webservices of the API. Any API extension would be part of the Wallet providers job.

IV. Licensing of the Wallet provider Apis would be Clients responsibilty.

V. Web development of the intended system is out of scope.

VI. Applications would be targetting specific operating systems and their versions and any addition to support would be deemed out of scope.

VII. Fleet tracking and asset management would not be part of the solution but would use these modules if necessary.

VIII. Server side resources and their maintainence would be done by Client.

1.3 DEFINITIONS, ACRONYMS AND ABBREVIATIONS

App Store	An installed application on mobile phone which helps user to find new compatible applications and download them from Internet.
CRUD	Create,Read,Update and Delete operations specific to Database.
Geocoding	Feature of the Application where the GPS location can be resolved to street address.
GPS	Global Positioning System
GUI	Graphical User Interface
HTTPS	Hyper Text Transfer Protocol Secured
ITR	Indian Telecom Regulations
MB	Mega Bytes (1024 bytes)
Mobile App	Mobile Cab Booking Booking Application
MoSCoW	Prioritization technique,Must have,Should have,Could have and Would have.
Passenger	Registered User of the Mobile Cab Booking Application.
Reverse Geocoding	Feature of Application where the street address can be converted to GPS coordinate location.
SMS	Short Messaging System
SOAP	Simple Object Access Protocol
SRS	Software requirements Specification.
System	Mobile Cab Booking Application
UI	Mobile Cab booking User Interface
User	Person who has not yet registered for the Mobile Cab Booking Application.
XML	Extensible Markup Language

1.4 OVERVIEW

The remainder of this document includes two chapters. The second one provides an overview of the system functionality and system interaction with other systems. This chapter also introduces different types of stakeholders and their interaction with the system. Further, the chapter also mentions the system constraints and assumptions about the product.
The third chapter provides the requirements specification in detailed terms and a description of the different system interfaces. Different specification techniques are used in order to specify the requirements more precisely for different audiences.

2 OVERALL DESCRIPTION

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and introduce the basic functionality of it. It will also describe what type of stakeholders that will use the system and what functionality is available for each type. At last, the constraints and assumptions for the system will be presented.

2.1 PRODUCT PERSPECTIVE

This system consists of Mobile Application that is used for booking Cabs for travel. The mobile application will be used to locate and book cabs for travel at Passengers chosen PickUp Point. The Passenger will be given the option to book his cab immediately or later at a given date and time. The mobile application will need to communicate to a GPS application within the mobile phone, which in turn communicates with a physical GPS device to find the location of the user. The GPS will provide the mobile application with locations of both the Pasenger and the cabs in the vicinity of the Passenger on a Map. The Map can be zoomed and Panned. The Location details are in address format and will be using the coding and reverse geocoding capabilities. The functionality provided by the GPS will be embedded into the application in order for the user to be able to use the functions in the application in a seamlessly manner.
Webservices would be used in the Application to communicate between the Mobile Application and the business Layer in the server. The Business Layer would constain Passenger,Driver and Admin modules.It would also contain complex Math and Computing algorithms for cab route optimization,Adjacency, Service area Limits,Timing etc. These modules have interfaces to outside world in terms of these webservices. XML based SOAP messages would be used for communication.
The Wallet feature is a third party service. The APIs for interaction with these are embedded in the Mobile App. Also, the Wallet APIs on the server are exposed as WebServices. These module are used for financial transaction with Cab Companys Merchant Bank, Escrow Account Payment Gateways etc.
The Wallert provider has also a secure server on which personal information of the Pasenger including credit/Debit card details and Net banking information would be stored.
All the data with regard to Mobile Application would be stored in Passenger Database on the Server. The mobile application will communicate with the database through webs services. All of the communication between the mobile and the server will go over the Internet on HTTPS protocol.
The mobile application has some restrictions about the resource allocation. To avoid problems with overloading the operating system the application is only allowed to use 20MB of memory while running the application. The maximum amount of hard drive space is also 20 MB.

2.2 PRODUCT FUNCTIONS

The Mobile Cab booking Application should support the following functions:

No.	Functionality	UseCase	Use case Name
1	Use case related to Authorization	USE CASE 01	REGISTER
		USE CASE 02	LOGIN
		USE CASE 03	CHANGE PASSWORD
		USE CASE 04	FORGOT PASSWORD
2	Use case related to Booking	USE CASE 05	BOOK CAB
		USE CASE 06	BOOK CAB NOW
		USE CASE 07	BOOK CAB LATER
3	Use cases related to showing View of Cab Locations, geocoding of Passenger location and Designated Cab Path on Map	USE CASE 08	SHOW MAPVIEW
		USE CASE 09	SHOW CABMAPVIEW
		USE CASE 10	SHOW DESIGNATEDCABMAPVIEW
4	Use case related to Tracking	USE CASE 11	TRACK CAB
5	Use case related to Payment and Fares.	USE CASE 12	REFILL WALLET
		USE CASE 13	PAY FOR TRAVEL
		USE CASE 14	VIEW FARE CHART
6	FeedBack of Travel	USE CASE 15	SEND FEEDBACK
7	Use case related to Choosing	USE CASE 16	CHOOSE CAB TYPE
		USE CASE 17	CHOOSE PASSENGER LOCATION
		USE CASE 18	CHOOSE DESTINATION LOCATION
8	Cancellation	USE CASE 19	CANCEL BOOKED CAB
9	Favourites	USE CASE 20	ADD TO FAVOURITES
10	Passenger Location	USE CASE 21	GET CABS AT PASSENGER LOCATION
11	Booking	USE CASE 21	VIEW BOOKINGS
		USE CASE 23	VIEW PENDING BOOKINGS
		USE CASE 24	VIEW PAST BOOKINGS

2.3 USER CHARECTERISTICS

1. The Passenger shold know about downloading Mobile Application from their respective App Stores on Android,IOS and Windows platforms.
2. The Passenger should know about using Mobile Maps and scrolling of maps along with zooming and Panning.
3. The Passenger should know about Mobile Wallet and its features.
4. Passenger should be well versed with Credit/Debit Card transactions and Netbanking.

2.4 USER

The Primary user of the Application is the Passenger.

2.5 GENERAL CONSTRAINTS

The constraints are as follows:

The Mobile Application is targetted for three Operating systems and the Application is going to be a native App. A development team with suitable experience working on each of the operating system need to be chosen.

The development team needs to have have native mobile development software IDE's and suitable testing API's. Licencing for commercial distribution if any needs to be purchased.

Since there are multiple OS, the UI wil likely not be the same for every one of them. Also, there may be a difference between what navigation features each of them provide.

The Internet connection is also a constraint for the application. Since the application fetches data over the Internet, it is crucial that there is an Internet connection for the application to function.

Suitable COTS software provider for Wallet needs to be chosen, which would agree upon the legal and financial bindings with the Cab company.

The mobile application performance will be constrained by the number of users and load sharing on the server.

The webservice interface functions may require additional functionality for the Mobile application to function.The development of these would be left to the Client.

2.6 ASSUMPTIONS

Serial No.	
1	Application(s) would be developed for two types of users, one being the Passenger and for registered Drivers of the Company on the mobile platform.
2	Already existing modules for Passenger,Driver and the Administrator would be deployed on a centralized server and exposed as web services.
3	Mobile applications(Driver and Passenger Applications) would be developed making use of available web service modules on the server side.
4	The Applications would be be developed as native mobile application for each of the platforms for enhaced security,access to local services and fastness of the application.
5	Passengers who would like to avail the services of the cab services can download the mobile application provided by the company for the following platforms: Windows,Android and Iphone, from their respective mobile stores.
6	Cabs would be provided with a tablet pre-loaded with application specific to cab Driver(s). It would also contain GPS unit to relay its location.
7	Registered Company Drivers would be able to use the application in the cab with the driver login credentials.

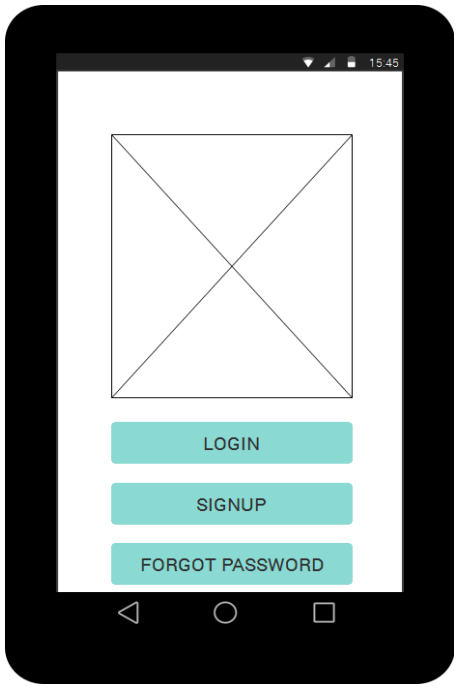
8	The Passenger mobile application would have the same UI components for all the device platforms.
9	Both the Driver and Passenger applications would make use of SMS,e-mail and push notification services.
10	Complex algorithms API's for distance calculation,Path Optimization,route generation and others already exist and can be availed by the application as webservices.
11	The Company would have suitable storage(Database) for storing Passenger details,Driver details and Location details (GPS) of Cabs operated by the company. These would be available to the Mobile App using webservices.
12	The application would make use of map services and location based services(GPS) on the mobile for tracking and showing the map and location of the Passenger on his/her mobile.
13	Wallet services would be developed in-App to the Passenger mobile booking solution and can be refilled using credit card,debit card and net banking.The Wallet service can be accessed seperately.
14	Wallet APIs for transaction management and other services would be exposed on the server as web services.
15	The Third party wallet service would hold the passenger details and passenger card and netbanking information on their secure servers.
16	Server and Database infrastructure would be provided by the Client. Scalability related issues would be taken care of by the Client.
17	Secure transfer of data to and from the server would be guaranteed by the Client by using suitable encryption technologies.
18	Data encryption of all tables on the server would be provided by the Client.
19	No Legal binding would exist for disclosure or breach of Privacy details on the Solution Provider
20	Development of the software for these module would be done at the Client end with pseudo-data. Final testing and conformance would be done on the Client side server Machine.

3 SPECIFIC REQUIREMENTS

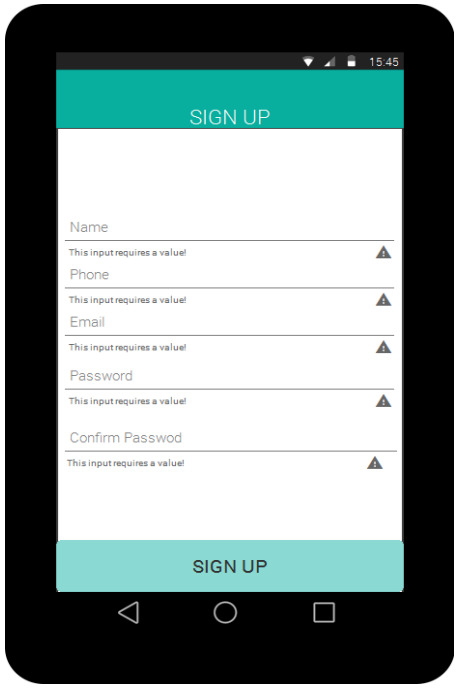
3.1 INTERFACES

1.1 USER INTERFACES

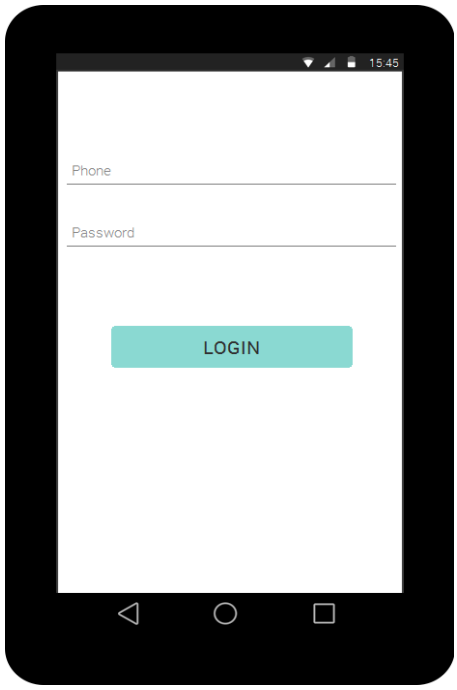
1.1.1 INITIAL PAGE



1.1.2 REGISTRATION PAGE

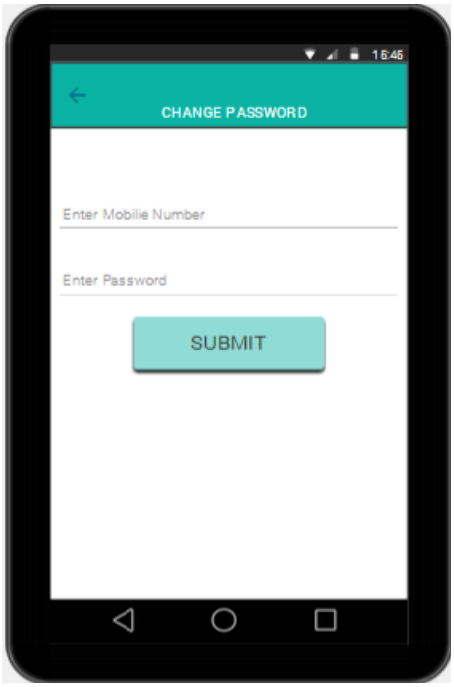


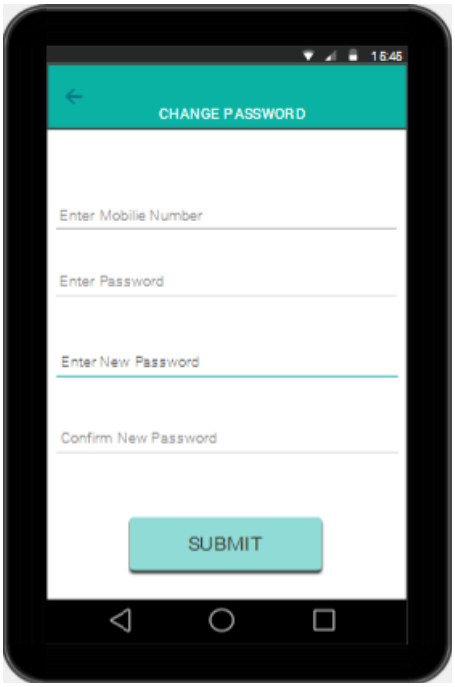
1.1.3 LOGIN PAGE



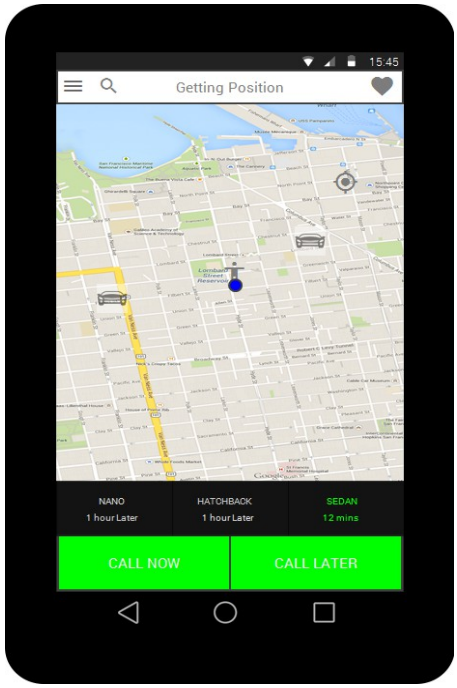
1.1.4 FORGOT PASSWORD PAGE

1.1.5 CHANGE PASSWORD PAGE

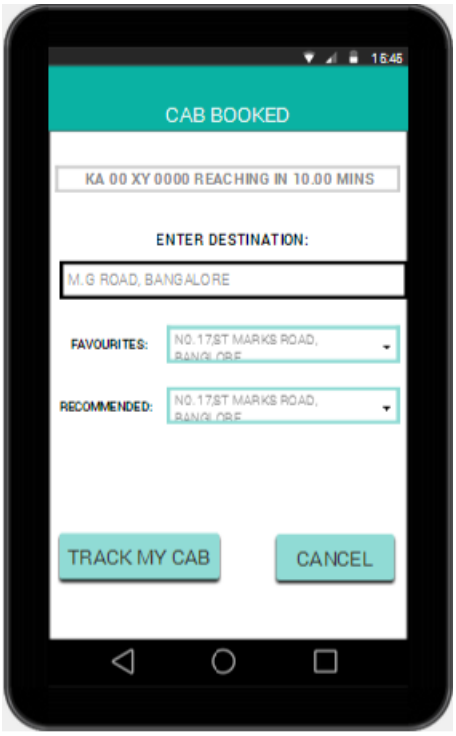




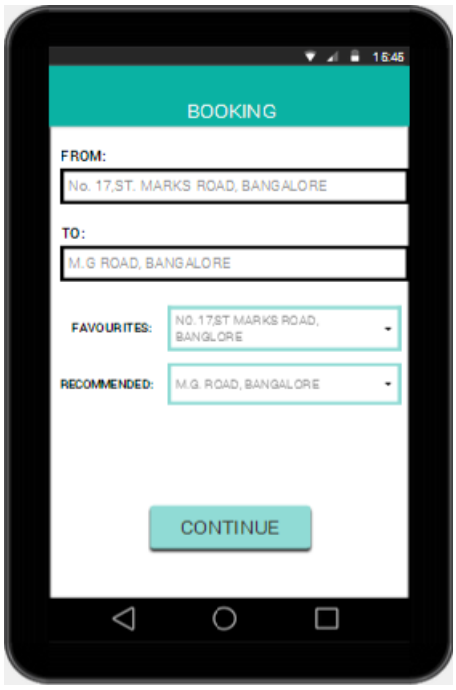
1.1.6 CABSMAPVIEW PAGE



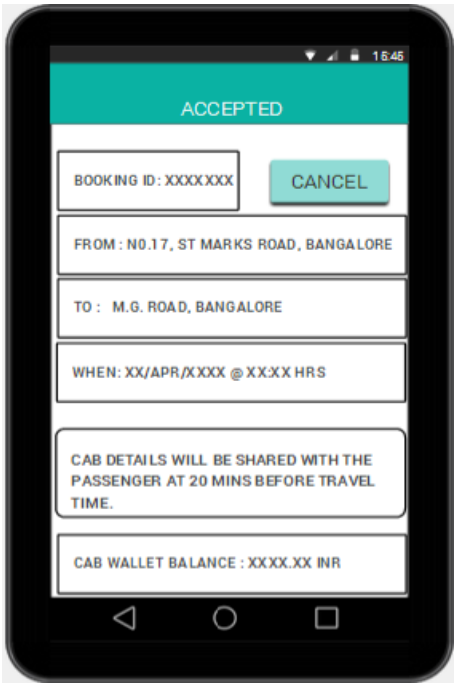
1.1.7 BOOK CAB NOW (CAB BOOKED PAGE)



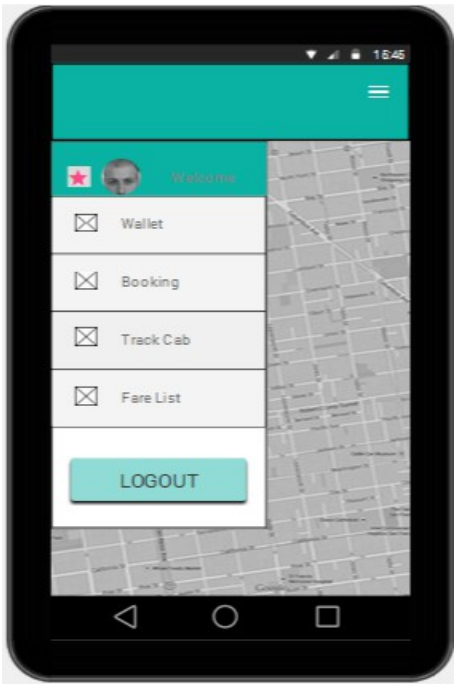
1.1.8 BOOK CAB LATER (DESTINATION AND TIME PAGE)



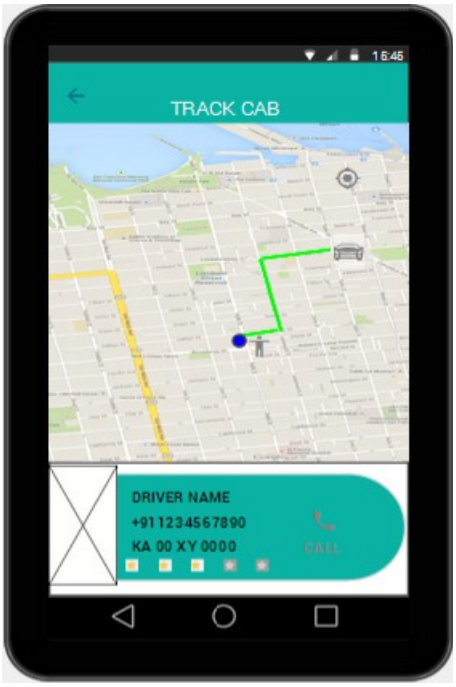
1.1.9 BOOK CAB LATER(ACCEPTED BOOKING PAGE)



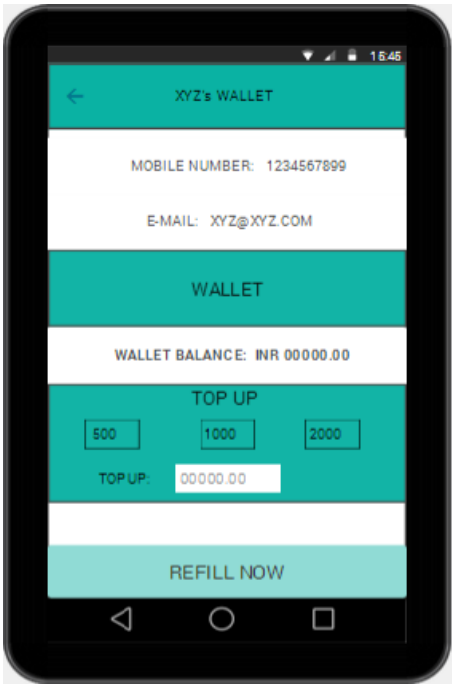
1.1.10 APPLICATION MENU



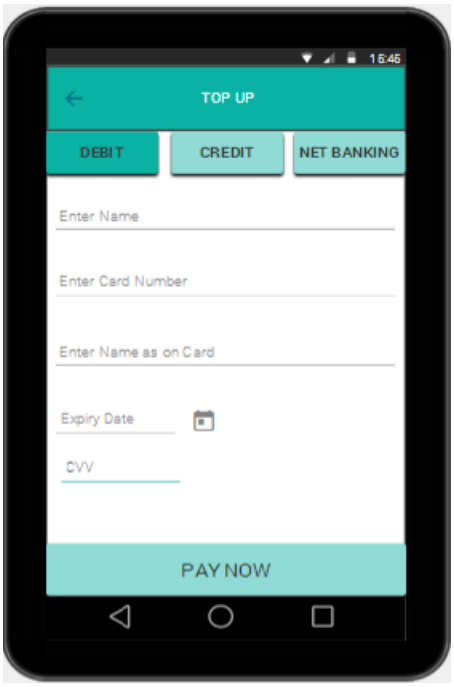
1.1.11 TRACK CAB

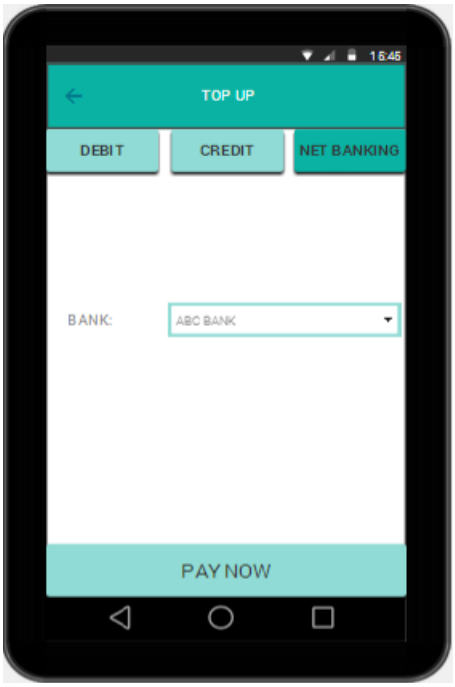


1.1.12 WALLET PAGE

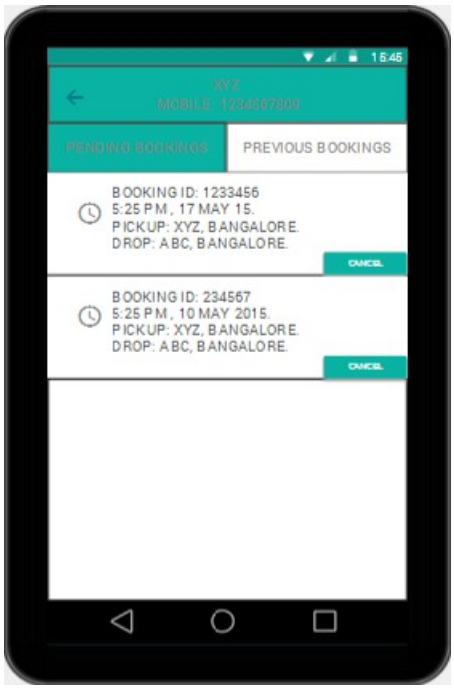


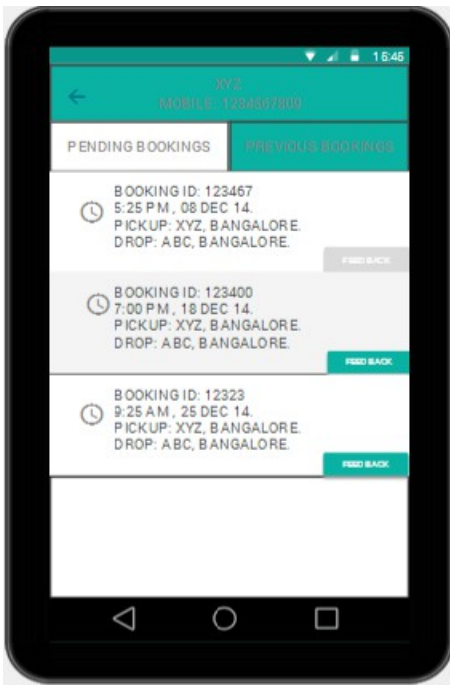
1.1.13 WALLET PAGE (DEBIT/CREDIT CARD, NETBANKING)





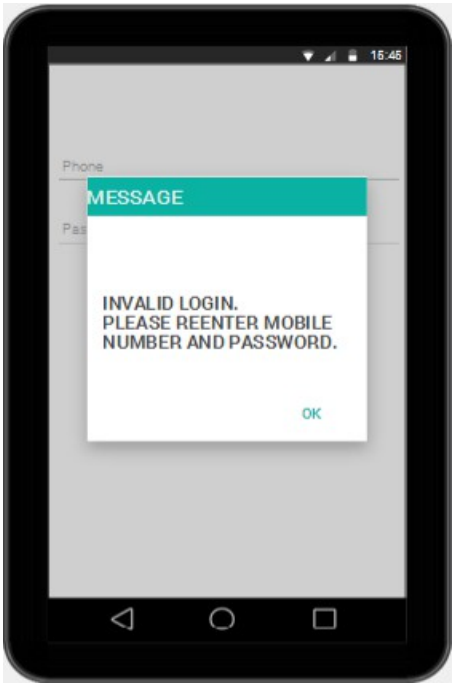
1.1.14 BOOKINGS (PENDING AND PREVIOUS)



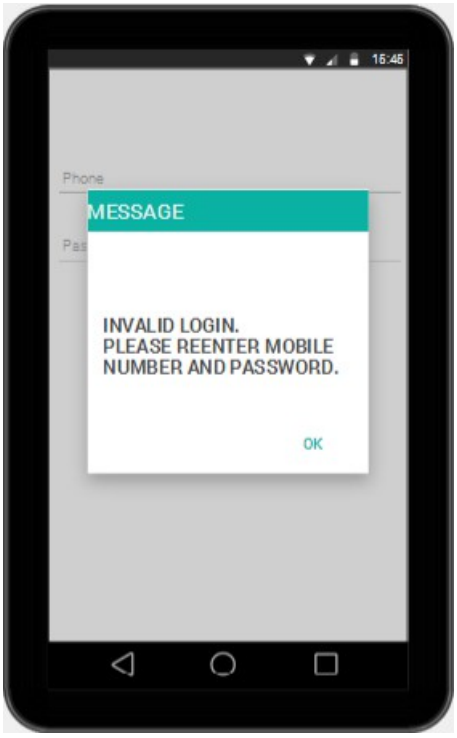


1.1.15 WAIT ICON

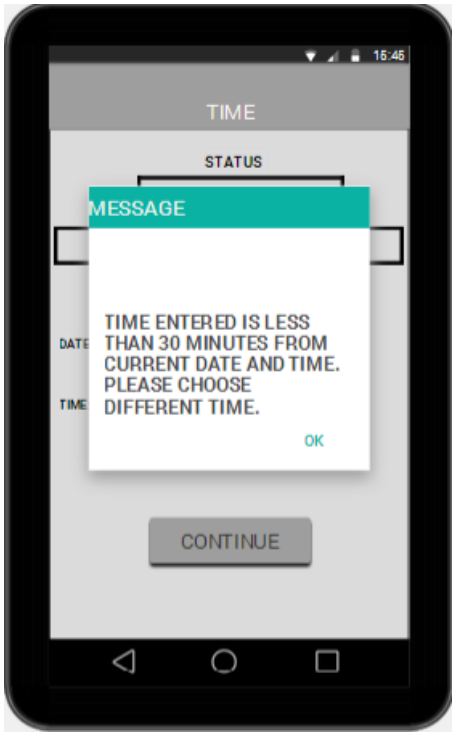
1.1.16 MESSAGES AND WARNINGS




15:45
FORGOT PASSWORD
ENTER REGISTERED MOBILE:
MESSAGE
PASSWORD SENT TO REGISTERED E-MAIL.
OK
NOTE: PASSWORD DETAILS WOULD BE SENT TO REGISTERED E-MAIL.


15:45
Phone
MESSAGE
INVALID LOGIN.
PLEASE REENTER MOBILE NUMBER AND PASSWORD.
OK


15:45
SIGN UP
MESSAGE
INVALID E-MAIL. E-MAIL ALREADY EXISTS.
OK
SIGN UP

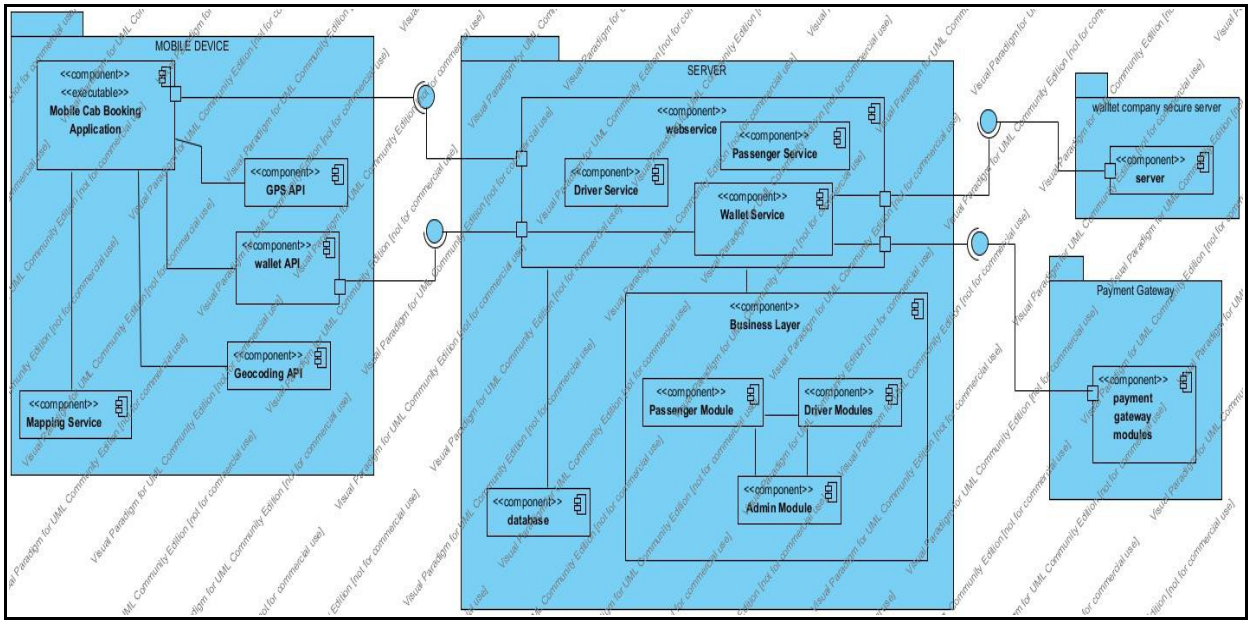

15:45
TIME
STATUS
MESSAGE
TIME ENTERED IS LESS THAN 30 MINUTES FROM CURRENT DATE AND TIME. PLEASE CHOOSE DIFFERENT TIME.
OK
CONTINUE

1.2 HARDWARE INTERFACES

Since the mobile application have any designated hardware, it does not have any direct hardware interfaces. The physical GPS is managed by the GPS application in the mobile phone.

1.3 SOFTWARE INTERFACES

The mobile application communicates with the GPS application in order to get geographical information about where the user is located.The geocoding APIs help in getting the address and GPS coordinated of a location and vice versa. The Mapping APIs on the mobile are used to retreive the map for a particular location of the Passenger PickUp point. Widgets for showing location of Cabs and Passenger are drawn over a layer on top of the Map.The Application communicates with the Server using exposed webservices. SOAP based webservices over HTTP/HTTPS and XML are used. The Webservices communicate with the Business Layer modules i.e; Passenger,Driver and Admin Modules. The Appliction would be able to connect with webservices specifically meant for Passenger services. Internally, the modules can communicate with each other. Also, Webservices would be used to communicate to Database residing on the server. Wallet compnay providers webservices would be used to communicate with the Payment gateway and Wallet company Server.



3.2 FUNCTIONAL REQUIREMENTS

We describe the functional requirements by giving various use cases for Passenger Mobile Application.

2.1 USE CASES FOR PASSENGER MOBILE APPLICATION

2.1.1 USE CASE 01 : REGISTER

ID: UC 01
DESCRIPTION: Given that a user has downloaded the mobile application, the user shall be able to register with the Application. The user must provide valid and complete details including his valid e-mail on the Registration Page of the Application.
ACTORS: Passenger
SECONDARY ACTORS: Server
DEPENDENCIES: None
PRE-CONDITION: The User has dowloaded the App and ready to register himself as a Passenger. The User must have a valid e-mail.
POST-CONDITION(S):
A) SUCCESS END CONDITION:
The User has succesfully registered with the Cab booking Application as a Passenger.
B) FAILURE END CONDITION(s):
The User is not able to register due to invalid data or e-mail already being registered or not valid.
FLOW OF EVENTS
BASIC FLOW:

1. User enters registration fields on Registration Page:
   a) Name,
   b) Phone,
   c) e-mail,
   d) Password and
   e) Confirm Password.
2. User submits Registration details on Registration Page.
3. Application checks for Inconsitency in Passenger Field values.
4. Application finds no fault and forwards the data to webservice.
5. Webservice forwards the data to the database for verification.
6. Database checks the data and returns a valid flag.
7. Webservice requests the User Registration details to be stored in database.
8. The Database stores the User information into the database.
9. The Database returns a succes flag to webservices.
10. WebServices sends SMS and registered e-mail to User about his registration with the Cab booking mobile Application.
11. Mobile application gets success return value from Web Services.
12. The Mobile Application shows succesfull registration.

ALTERNATE FLOW:
3a The Field values are not valid.
3a.1 Clear Registration Fields and allow Passenger to enter Registration

details.

3a.2 resume @ step 1 of Basic Flow.

3b Password and Confirm Password Fields dont match.

3b.1 Clear Registration Fields and allow Passenger to enter Registration details.

3b.2 resume @ step 1 of Basic Flow.

EXCEPTIONAL FLOW:

6a Email already exists in the database.

6a.1 Show e-mail is already registered.

6a.2 End Registration Process.

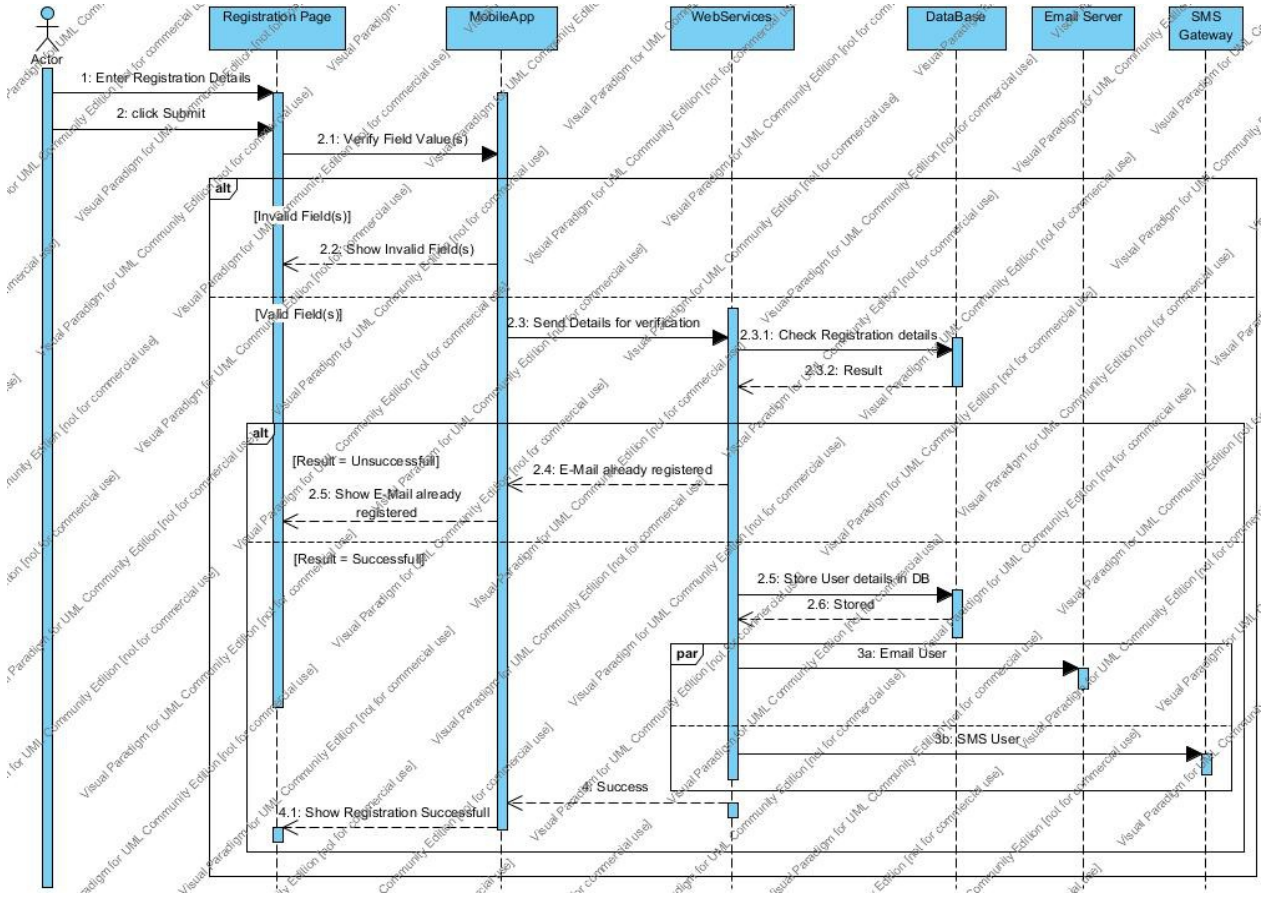
BUSINESS RULES:

1. All fields on the Registration page are necessary.
2. Personal information of the passenger would be governed by privacy rules of the Company.
3. The Passenger has to register himself seperately for using the Wallet services with his Mobile Number.
4. Information with regard to cab booking,cancellation, payment and cab company promotional details can only be sent to registered e-mail.
5. The Passenger data entity created in database must have a unique ID.
6. The password field of Passenger Entity shall be encrypted in Database.
7. Registration shall avail 10% discount on their first travel by using promo code sent to their welcome message in e-mail.

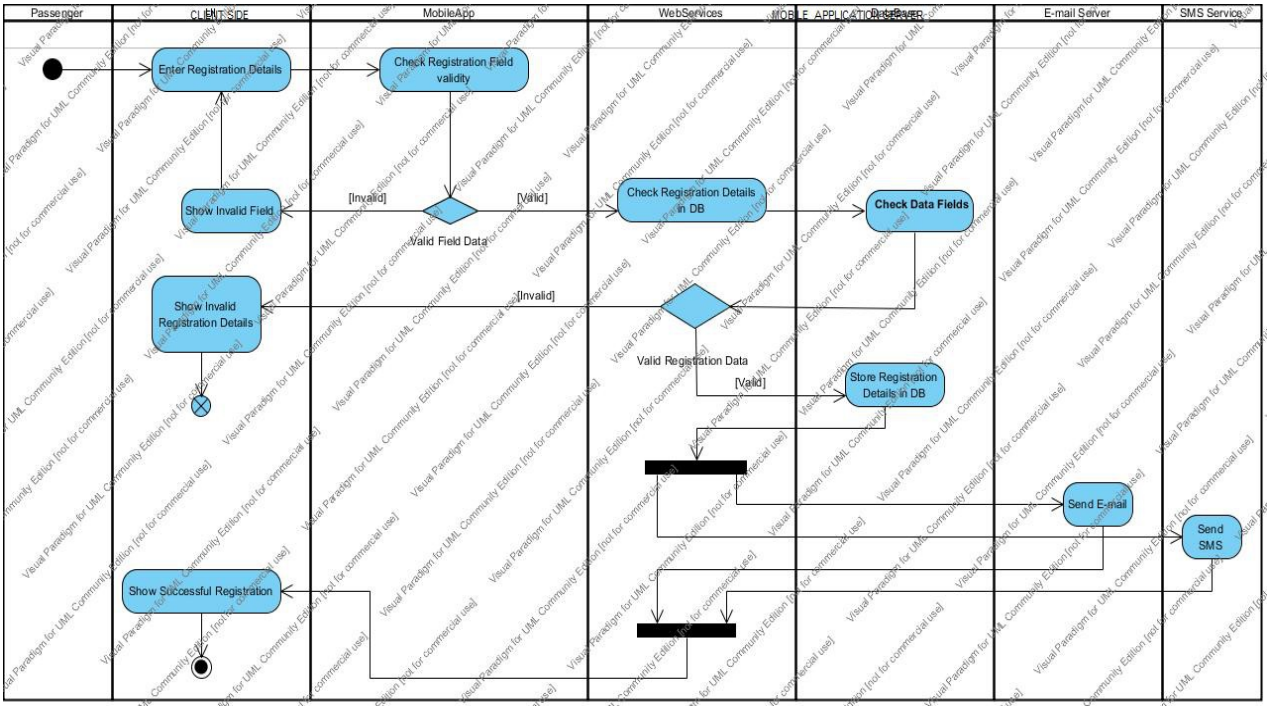
IMPLEMENTATION RULES AND SPECIFICATIONS:

1. Mobile number should be 10 charecters long.
2. Mobile Number field should allow only numerical values.
3. Name field should allow 32 charecters.
4. The field width of Password and confirm Password on Registration page must be 8 charecters long.
5. Password Field(s) values must be a hidden.
6. Password must be alpha-numeric charecters only.
7. Registration Page field validation for length, charecter type for both the fields must be done.
8. Device orientation look and feel should be consistent.

The sequence and activity diagram for the use case are shown in the following diagrams.



Sequence diagram for Registration



Activity Diagram for Registration

2.1.2 USE CASE 02 : LOGIN

ID: UC 02
DESCRIPTION: This usecase describes the process of Logging into the Application. The Registered Passengers of the Application need to Login to avail the features of the Cab Mobile Application. The Passenger needs to give the registered Mobile Number and Password as Login credentials. The Application sends the information to the server and redirects the Passenger to CabsMapView, where the Passenger would be able to view his/her location as well as cabs in the vicinity on a map and book the same for his travel.
ACTORS: Passenger
SECONDARY ACTORS: Server
DEPENDENCIES: UC 01
PRE-CONDITION: The Passenger has registered with the Application and has a valid Password.
POST-CONDITION(S)
SUCCESS END CONDITION:
The Passenger has logged into the application and cab view his Location and Cabs location on the Map.
FAILURE END CONDITION(s):
Not able to login into the Application.
FLOW OF EVENTS
BASIC FLOW:

1. The Passsenger enters the following fields on Login Page
   a. Mobile Number and
   b. Password.
2. The Passenger submits the Login information.
3. Application checks for the validity of submitted fields.
4. Application forwards the data to webservices for Login Authentication.
5. Webservices forward the data to database for Authentication.
6. Database checks Login information, finds the password and mobile number match with Passengers record and return success flag.
7. The webservices returns valid data flag to the Mobile Application.
8. Extension usecase ***SHOW CABSMAPVIEW*** starts. [*Extension point*: OnLogin.]
9. Mobile Application requests CabsMapView Page.
10. The Mobile Application gets the Passenger location using the mobile GPS.
11. CabsMapView requests Map from Map services.
12. CabsMapView renders the CabType and BookCab widgets and selects “Nano” as default CabType along with “Book Cab Now” as default.
13. CabsMapView renders the Map along with Passenger Location and Homing Icon.
14. The Mobile Application request cab location data in the vicinity of the passenger.
15. The webservice forwards the request to the database.
16. The database queries all the cabs that are in the vicinity of the passenger and returns a list to webservices.
17. The web services forwards the list to Application.
18. The Application stores the data in cache for further use.

19. CabsMap View diplays cabs based on the default cab type chosen on the Map along with the homing icon.

ALTERNATIVE FLOW:

3a Passenger enters invalid field values on Login Page.

3a.1. Clear Login Fields and resume @ 2.

6a Database returns false flag for authentication

6a.1 Webservices returns failed for Login Authentication.

6a.2 Login Page displays message about invalid details on Login Page.

6a.3 Allow Passenger to Login for 2 more times.

6a.4 resume @ 2

EXCEPTIONAL FLOW:

6e Database returns false flag and Passenger has tried max number of attempts to login with failure.

6e.1 Webservices returns failed for Login validation.

6e.2 Login Page displays login failed and is Account suspended for 1 day.

6e.3 Use case ends.

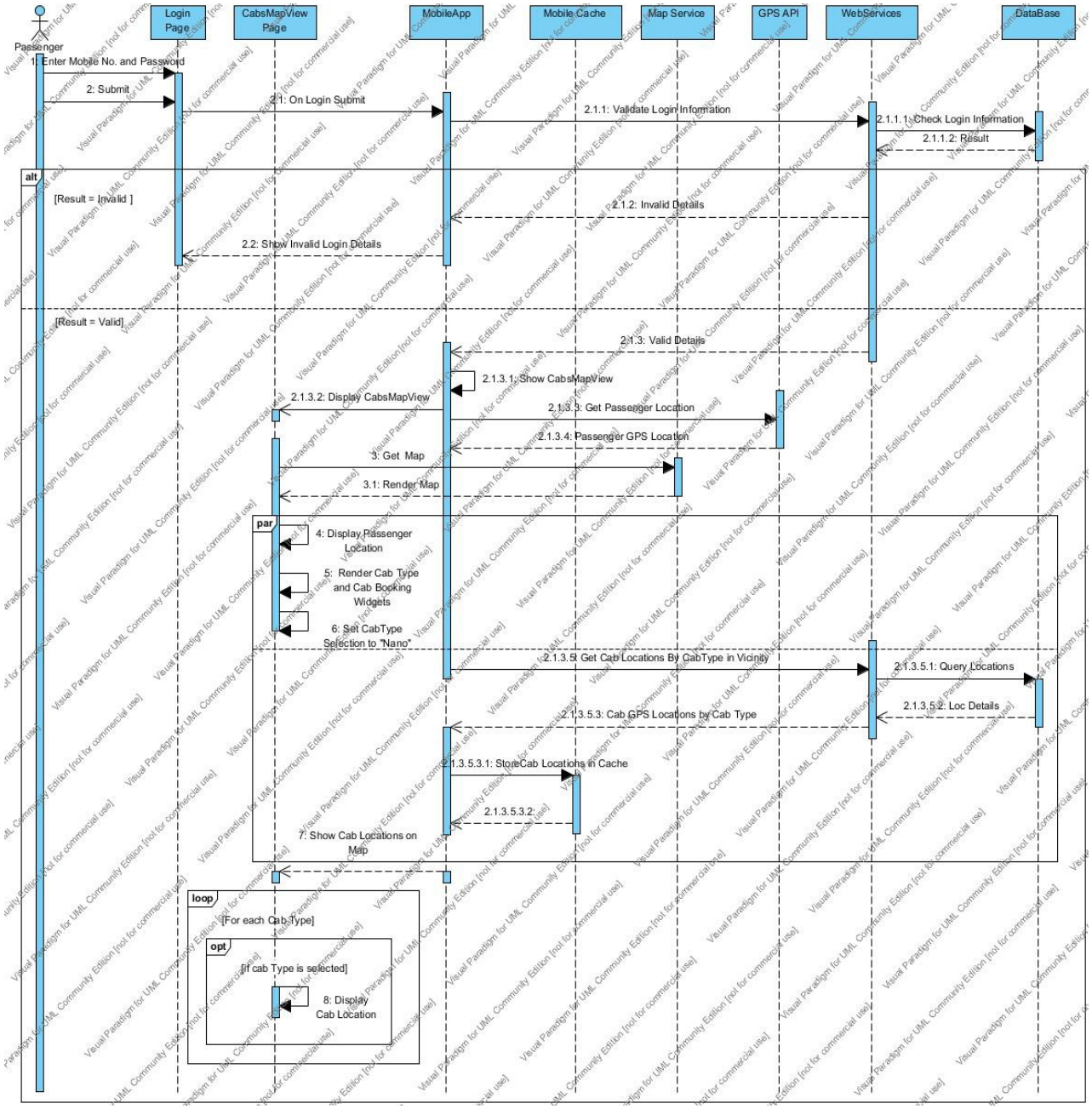
BUSINESS RULES:

1. The Login page will not show any advertisements.
2. In-App purchase should not be allowed.

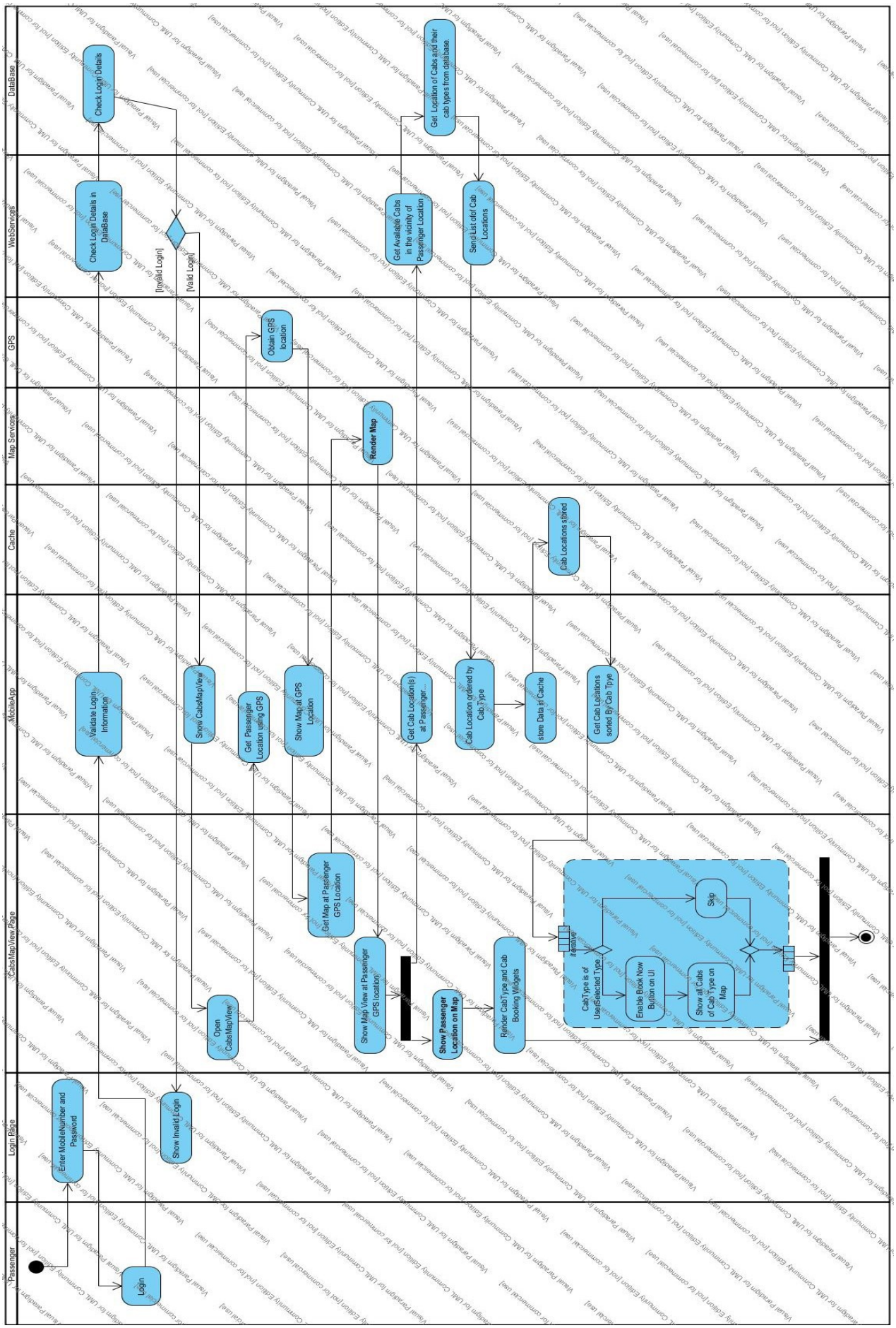
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:

1. The field width of mobile number in login page must be 10 charecters long.
2. Mobile Number field should allow only numerical values.
3. The field width of password in Login page must be 8 charecters long.
4. Password Field values must be a hidden.
5. Password must be alpha-numeric charecters only.
6. Login Page field validation for length, charecter type for both the fields must be done.
7. Device orientation look and feel should be consistent.
8. Passenger accounts marked suspened need to be contacted by Admin through e-mail.

The sequence and activity diagram for the use case are shown in the following diagrams.



Sequence diagram for Login



Activity Diagram for Login

2.1.3 USE CASE 03 : CHANGE PASSWORD

ID: UC 03

DESCRIPTION: Registered Passsengers can change their password using this feature. Initially, the Passenger needs to authenticate on the Change Password Page by using his Mobile number and existing Password.Once authenticated the Passenger will be provided with New Password and Confirm Password fields on the same Page. He can submit the new Pasword details and thus change the Password.

ACTORS: Passenger

SECONDARY ACTORS: Server.

DEPENDENCIES: UC 01,UC 02

PRE-CONDITION:

1. The Passenger is directed from Login Page to the Change Password Page.
2. Passenger has valid Login details.

POST-CONDITION(S)

SUCCESS END CONDITION:

Passsenger is able to change his password and is forwarded to the Login Page.

FAILURE END CONDITIONS:

Passsenger is not able to change his password and the account is suspended.

FLOW OF EVENTS

BASIC FLOW:

1. Passenger enters the following authentication fields in Change Password Page:
   a) Mobile number and
   b) Password.
2. Passenger submits his authentication details.
3. Application does field validation check on entered field values.
4. Application forwards validated details to the webservices for authenticating in Database.
5. Webservices forwards the details to Database for authentication.
6. Database checks the data and return valid flag.
7. Webservices returns success.
8. Application prompts UI to show new password and confirm password fields.
9. Change Password Page shows updates and shows two Password fields.
10. Passenger enters the new Password and Confirm new Password fields.
11. User submits data on Change Password Page.
12. Application checks data for field validation and consistency of Passwords.
13. Application forward the data to webservices.
14. Webservices forwards the data to database for updation.
15. Database updates the password field of the Passenger.
16. Webservice returns a success flag.
17. Change Password Page shows password changed succesfully message.

ALTERNATIVE FLOW:

6a Database returns invald flag for Validation.

6a.1 Webservices return invalid flag to mobile application.

6a.2 Change Password Page shows Invalid Login details message.
6a.3 Allow Passenger to Login for two more times.
6a.3 resume @ 1 of Basic Flow.

EXCEPTIONAL FLOW:

6e Database returns Invalid flag and Passenger has tried max number of attempts to login with failure.

6e.1 Webservices returns false flag to Application.
6e.2 Change Password page displays login failed.Account suspended.

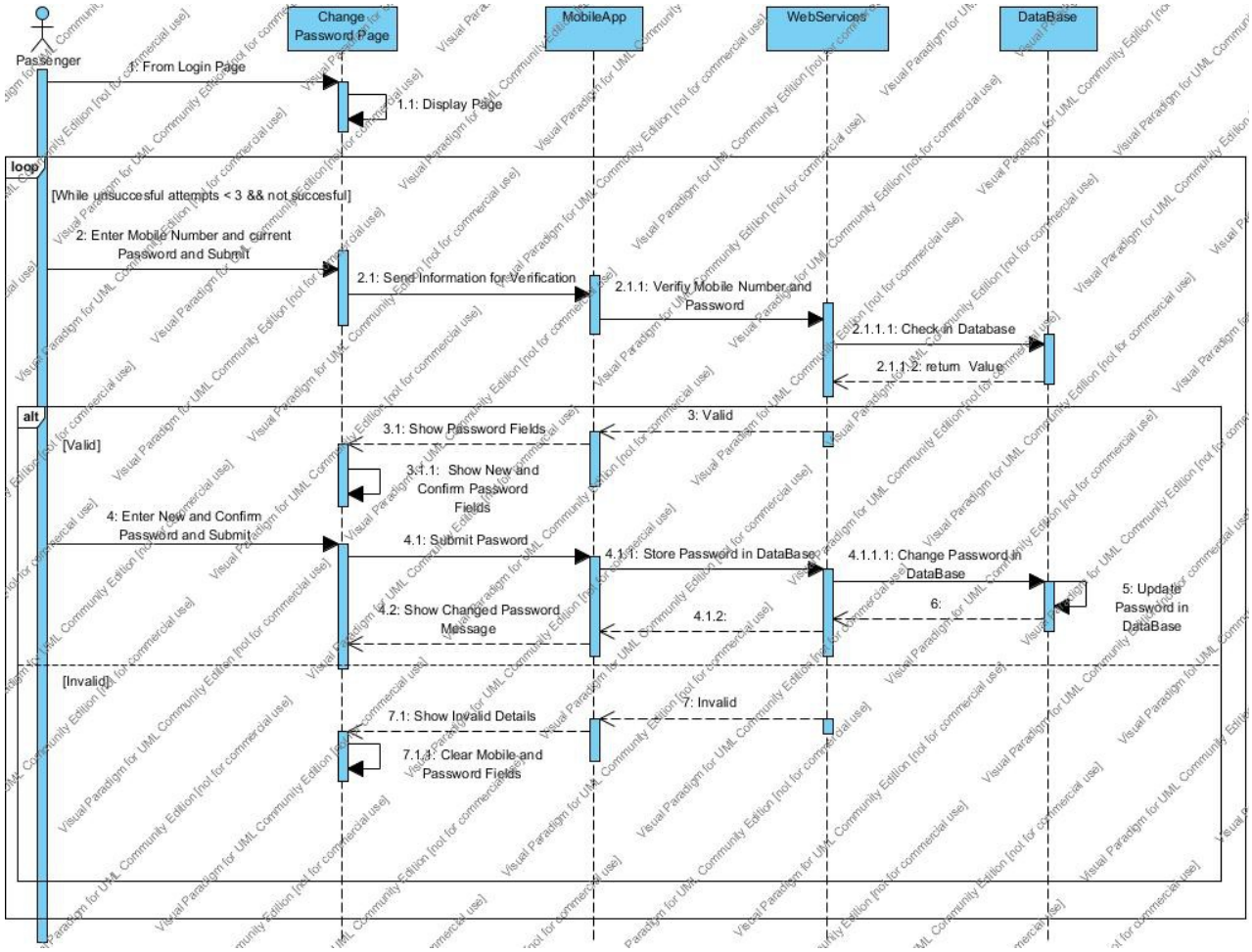
BUSINESS RULES:

1. All suspended accounts need to be taken into account by the Admin on a daily basis.

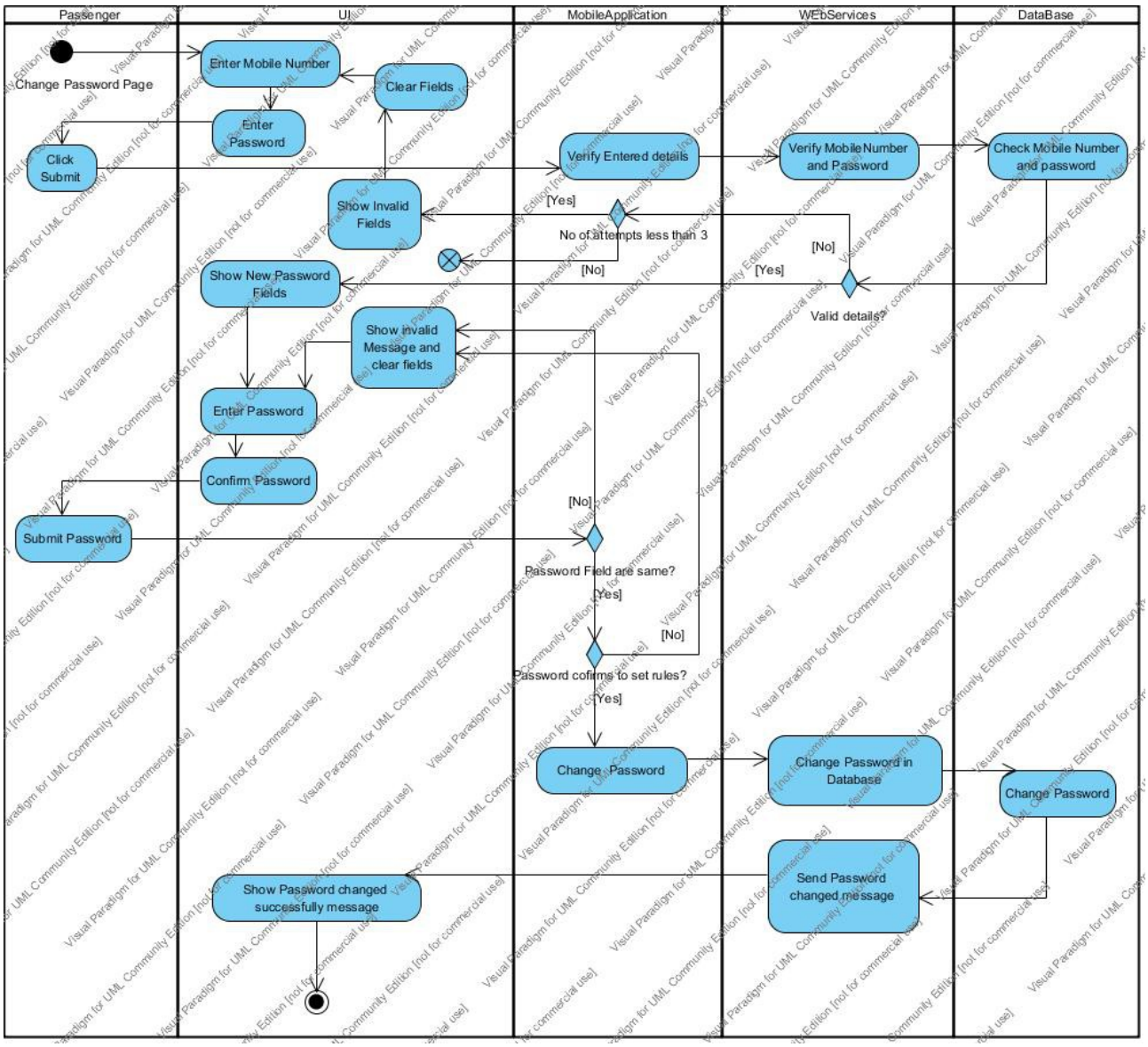
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:

1. The field width of mobile number in login page must be 10 charecters long.
2. Mobile Number field should allow only numerical values.
3. The field width of password in Login page must be 8 charecters long.
4. Password Field values must be a hidden.
5. Password must be alpha-numeric charecters only.
6. Login Page field validation for length, charecter type for both the fields must be done.
7. Device orientation look and feel should be consistent.

The sequence and activity diagram for the use case are shown in the following diagrams.



Sequence diagram for Change Password



Activity Diagram for Change Password

2.1.4 USE CASE 04 : FORGOT PASSWORD

ID: UC 04

DESCRIPTION: This usecase describes the processs of retreiving password if one has forgetten the Password for the Application.The Passenger would be able to retrieve his password using this feature.The passenger enters his registered e-mail phone number in the Forgot password page. He would be able to see his password details in his registered e-mail.

ACTORS: Passenger

SECONDARY ACTORS: Server.

DEPENDENCIES: UC 01

PRE-CONDITION: The user has registered with the application and has the Cab Booking application on his mobile.The Passsenger is directed to the Forgot Password page from the Initial Login Page by clicking on the Forgot password button.

POST-CONDITION(S):

SUCCESS END CONDITION: The Passsenger is able to retrieve his password through e-mail.

FAILURE END CONDITION(s): The Passenger is not able to retrieve his password due to e-mail fields not valid.

FLOW OF EVENTS

BASIC FLOW:

1. Passenger enters mobile number in the Forgot Password Page.
2. Passenger submits details.
3. Application forwards mobile number to web services for verification.
4. Web services forwards mobile number to Passenger Database.
5. Database verifies if mobile number exists in Passenger Database.
6. Database return true flag to webservices.
7. Webservices asks for the registered e-mail and password details from the Passenger Database.
8. Passenger Database retrieves the same and returns it Webservices.
9. Webservices would compose a mail with password information and send it Passengers registered e-mail address through e-mail server.
10. Passenger would get a message that password information has been sent to his registered e-mail.

ALTERNATIVE FLOW:

6a. DataBase returns false flag to webservices.

6a.1 Webservices forwards false flag to Application.

6a.2 Forgot Password Page shows Invalid mobie number on the screen.

EXCEPTIONAL FLOW:

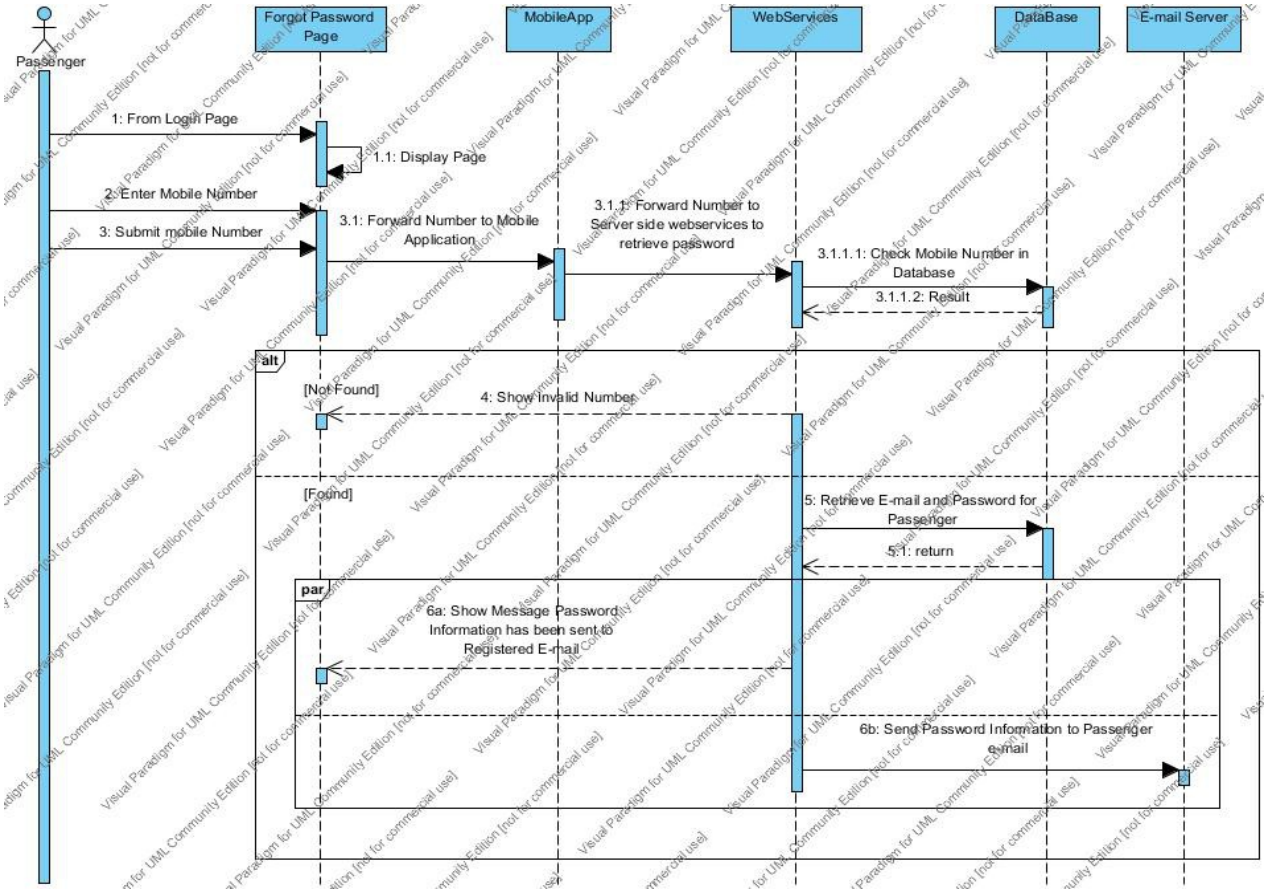
BUSINESS RULES:

1. Company would not be held responsible for misuse of password sent through e-mail.

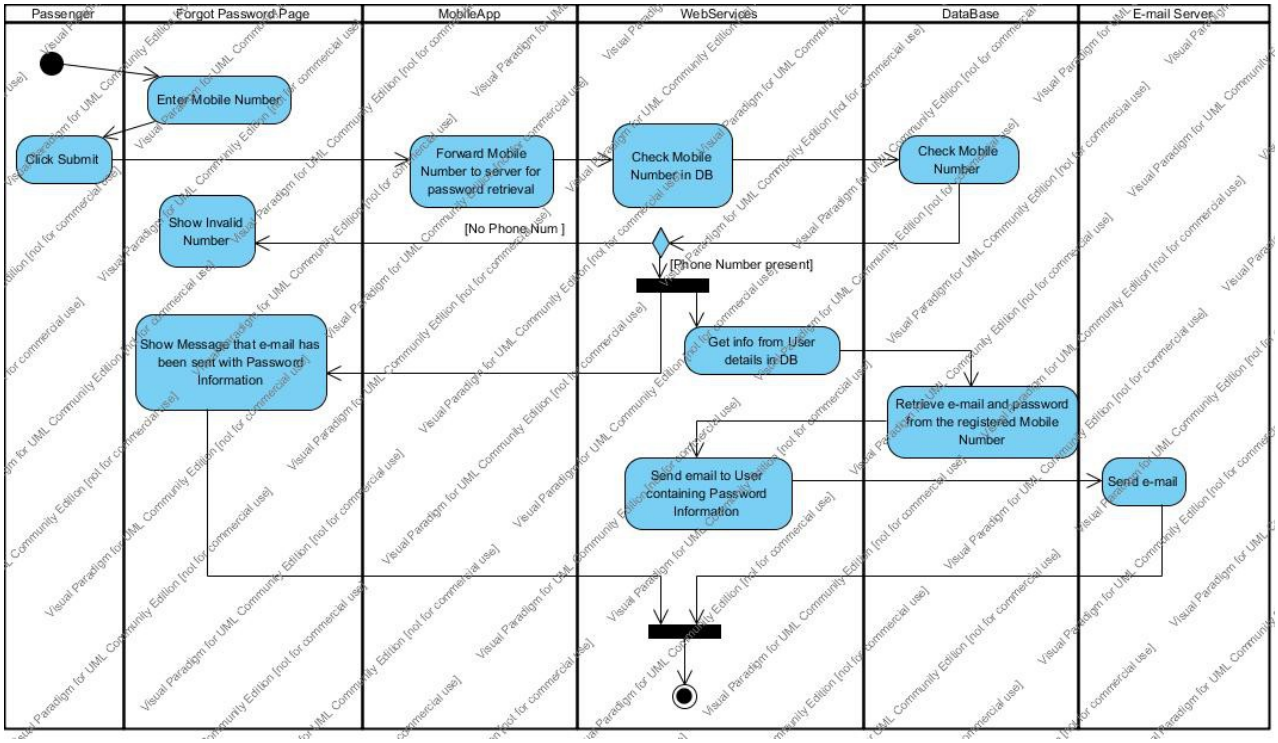
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:

1. The field width of mobile number in login page must be 10 charecters long.
2. Mobile Number field should allow only numerical values.
3. Device orientation look and feel should be consistent.

The sequence and activity diagram for the use case are shown in the following diagrams.



Sequence diagram for Forgot Password



Activity diagram for Forgot Password

2.1.5 USE CASE 05 : *BOOK CAB*

ID: UC 05
DESCRIPTION: This Use case describes the process of booking a cab. It is a base use case for ***BOOK CAB NOW*** and ***BOOK CAB LATER***.Passenger once logged in to the application will be able to view CabsMapView page showing Cabs available in the vicinity of the current Passenger location on a Map based on the Cab type selected. Passenger can opt to book a cab immediately for a given cab type or later at a given date and time.
ACTORS: Passenger
SECONDARY ACTORS: GPS API,Map Services,Geocoding services.
DEPENDENCIES: UC 01**,**UC 02
PRE-CONDITION:

1. Passenger has logged in to the Application.
2. Cabs of a Cab type are shown the Map at Passenger Location.

POST-CONDITION(S)
SUCCESS END CONDITION:
Passenger is able to view cabs and book cabs of his cab type.
FAILURE END CONDITION(s):
Passenger has abandonded the Cab booking Process.
FLOW OF EVENTS
BASIC FLOW:

1. Include ***Choose PickUp Location***.
2. Passenger has made a choice of his Pickup location and shows the address of the location of the Passenger.
3. Include ***Choose CabType***.
4. CabsMapView shows cabs available in the vicinity of the Passenger Pickup Location.
5. CabsMapView shows the cabs for the given cab type on the Map.

ALTERNATIVE FLOW:
***a** Passenger can choose a different PickUp Location any time.
*a.1 Allow Passenger to choose another Location.
*a.2 Include ***CHOOSE PICKUP LOCATION.***
*a.3 Include ***GETCABSATPASSENGERLOCATION.***
*a.2 resume @ step 2 of Basic Flow.
2a Passenger wants to store current Location into his favourites.
2a.1 Extend ***ADD TO FAVOURITES[***Extension Point: Favourites]
2a.2 resume @ 3
4a Passenge wants to choose a different cab type.
4a.1 Allow Passenger to select cab type.
4a.2 Include ***CHOOSE CABTYPE.***
4a.3 resume @ step 4 of Basic Flow.

EXCEPTIONAL FLOW:
***** Any time the Passenger can abandon the Cab booking process.

BUSINESS RULES:

IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:

1. Cab and Passenger locations shown on the map would depend on the accuracy provided by GPS API.
2. There can be variation about location of Passenger and Cabs shown on different mobile platforms at same location.

2.1.6 USE CASE 06 : BOOK CAB NOW

ID: UC 06
DESCRIPTION: This Use case describes the process of booking a cab immediately for travel for Registered Passengers at their current Location.This use case inherits ***BOOK CAB*** use case.Passenger once logged in to the application will be directed to CabsMapView page. The page shows Cabs available in the vicinity of the passenger on a Map based on the Cab type selected. Passenger can opt to book a cab immediately for a given cab type that are available in the vicinity of chosen Passenger location.
PARENT USECASE: BOOK CAB (Use case ID: 05)
ACTORS: Passenger
SECONDARY ACTORS: GPS API,Map Services.
DEPENDENCIES: UC 01**,**UC 02
PRE-CONDITION:
Passenger has logged in to the Application.
Cabs of a Cab type are available for booking immediately as shown in the CabsMapView of the Application.
POST-CONDITION(S)
SUCCESS END CONDITION:
Passsenger is able to book the cab immediately.
FAILURE END CONDITION(s):
Passenger has cancelled the Cab Booking.
Passenger is not able to book the Cab.
Passenger has abandonded the Cab booking Process.
FLOW OF EVENTS
BASIC FLOW:
1. Include ***BOOK CAB***.
2. Cabs of chosen Cab type are available and the Passenger chooses to Book Cab immediately.
3. Application sends request to Webservices to get designated Cab and its Driver details along with GPS location of the Cab.
4. Webservices forwards request to the Database.
5. Cab details and its GPS location,Driver Information are retrieved from the Database.
6. WebService creates a new Booking with Booking ID and forwards it to store in Database.
7. Database stores cab booking associated to the Passenger.
8. WebServices forwards the booking details through registered e-mail and SMS to the Passenger.
9. Webservices forwards Booking details to Application.
10. Application redirects from CabsMapView to Booking Confirmed page.
11. Booking Confirmed Page shows the Cab and Driver details along with Estimated Time of Arrival.

ALTERNATIVE FLOW:
2a Cabs of chosen cab type are not avialable and wants to choose another

cab type.

2a.1 Allow passenger to choose another Cab Type.

2a.2 resume @ 4 of Basic Flow.

2b Cabs of Cab type are not available and Passenger wants to wait till a cab is avaiable of his cab type and is on the CabsMapView Page.

2b.1 Passenger waits for avaiable cab of given cab type.

2b.2 Application refershes the cabs Location every 30 secs or when page is updated.

2b.3 Passenger chooses Cab once available.

2b.4 resume @ 6 of Basic Flow.

1. EXCEPTIONAL FLOW:

***** At any time the Passenger can abandon the Cab booking process.

2e Cabs of Cab type are not available and Passenger does not want to wait for cabs of cab type to be avaiable.

2e.1 Passenger exits/Logsout from the Application.

11e Passenger has changed his mind and wants to cancel his travel.

11e.1 Passenger opts for Cancellation on the Booking Confirmed Page. Extend ***CANCEL BOOKEDCAB*** [Extension Point: Cancel Booked Cab]

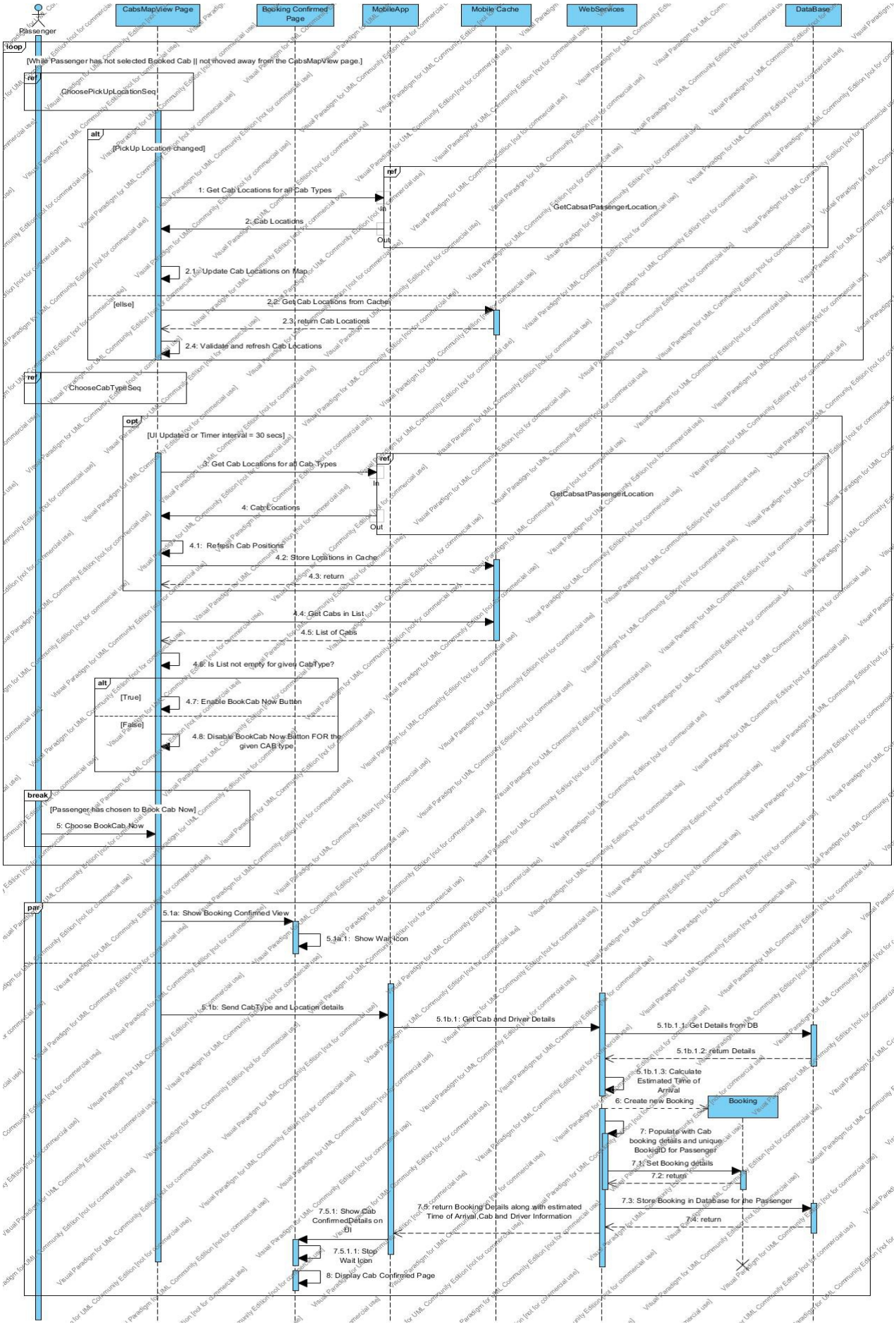
BUSINESS RULES:

1. Priority for choosing a Cab of a given Cabtype would be based on the fastest arrival time from a set of cabs and notifying the Driver of the Cab.
2. Designated Cabs need to arrive at Passenger PickUp Location no later than 30 mins after booking.
3. Waiting Time charger at Passenger PickUp Location is free for the first ten minutes and later a surcharge of INR 50.00 would be charged for each additional 30 mins.
4. An unique BookingID has to be created for Passenger Booking and needs to be 6 digits long.
5. The Destination field would be optional for the Booking data entity.

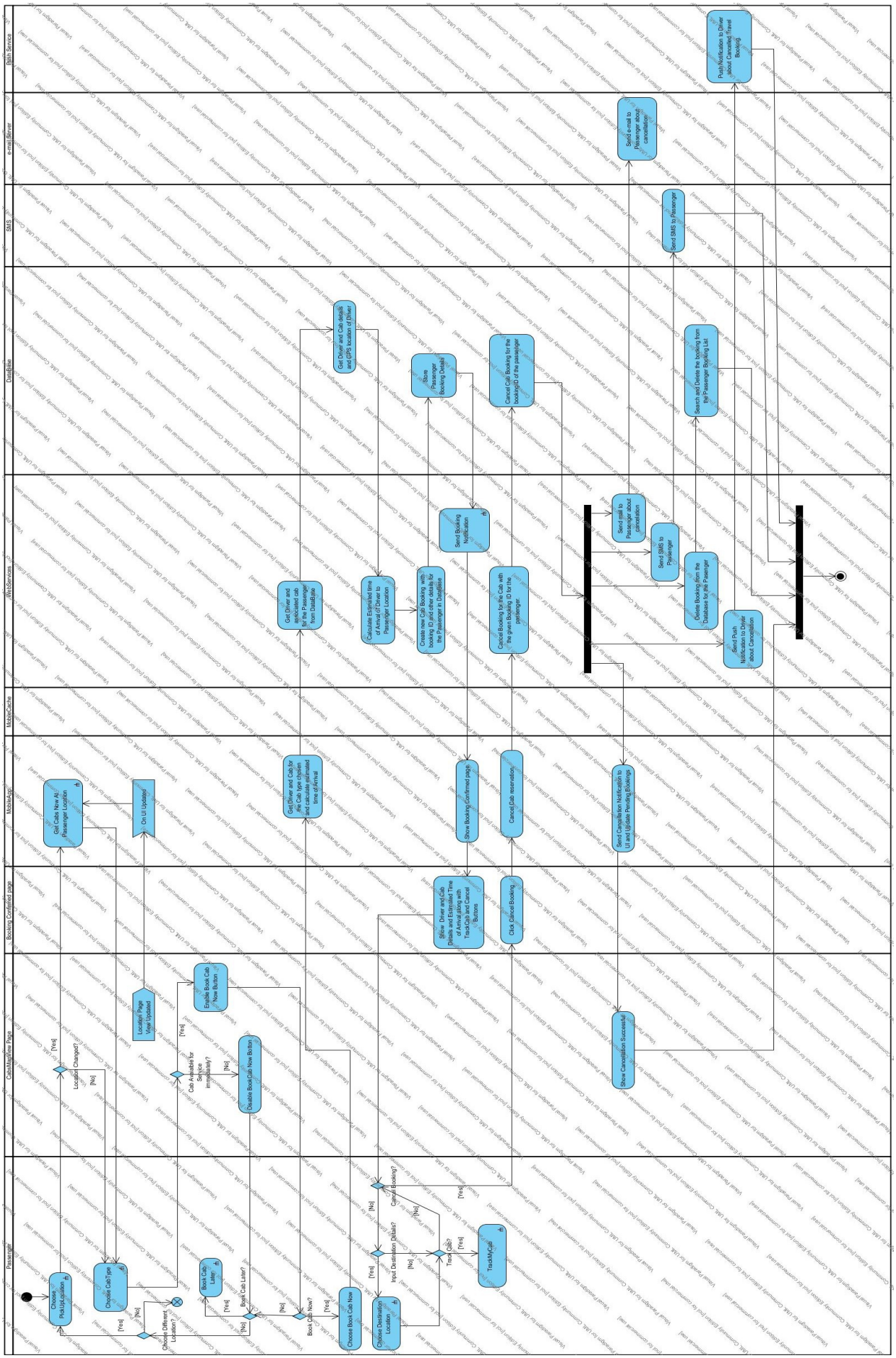
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:

1. Booking Confirmed Page shall have Cancel and Track Cab buttons.

The sequence and activity diagram for the use case are shown in the following diagrams.



Sequence Diagram for Book Cab Now.



Activity Diagram for Book Cab Now

2.1.7 USE CASE 07 : BOOK CAB LATER

ID: UC 07

DESCRIPTION: This use case describes the process of booking a cab at a given date and time. Passenger logged into the Application will be able to view all cabs of a cab type chosen on the CabsMapView for a particular Passenger Location. The Passenger can choose to book a cab of a given cab type at a later date and time by opting to book cab later on the View. The details of his destination, date and time of travel are noted. The passenge would be notified 20 minutes prior to his travel time about designated cab and Driver details.

PARENT USE CASE: BOOK CAB (Use case ID: 05)

ACTORS: Passenger

SECONDARY ACTORS: Server.

DEPENDENCIES: UC 01**,** UC 02

PRE-CONDITION: The passenger has logged in to the application and wants to book a cab at a given date and time of his choice.

POST-CONDITION(S):

SUCCESS END CONDITION:

The Passenger is able to book a cab at a given time and date of his choice.

FAILURE END CONDITION(s):

The Passenger is not able book a cab.

The Passenger moves away from the Booking process.

The Passenger has cancelled the booking immediately or at a later date.

FLOW OF EVENTS

BASIC FLOW:

1. Include ***BOOK CAB***.
2. Passenger wants to book cab for a Particular CabType, at later date and time and clicks on Book Cab Later.
3. Application redirects from CabsMapView page to Destination page.
4. Passenger inputs his destination Location on the Destination page. Include ***CHOOSE DESTINATION LOCATION*** .
5. Passenger is directed to the Time Page.
6. Passenger enters the date and time of travel on the Time Page.
7. Cab is available for booking at Passenger stipulated Date and time.
8. Passenger confirms Booking.
9. Application passes the Booking information to Webservices.
10. The webservices creates an booking object filled booking details and booking ID and forward the object to Database for storing.
11. The Database stores the information.
12. Webservices forwards booking details to registered e-mail and mobile phone through SMS.
13. Passenger is shown the Confirmed Booking Page with details of the Booking.

2. ALTERNATIVE FLOW:

7a Cabs of chosen cab type are not avialable at given date and time and Passenger wants to choose another cab type.

2a.1 Passenger returns to CabsMapView Page and chooses another Cab Type.

2a.2 resume @ step 1 of Basic Flow.

7b If time chosen for the Booking is within 30 mins of the current time and Day of travel is the day of booking, then the Passenger needs to be informed.

4a.1 Passenger will be given a message about the constraint and will need to give a different timing for his booking.

4a.2 resume @ step 4 of Basic Flow.

7c If a Cab booking already exists for the Passenger within 30 mins before or after the Cab booking, then the Passenger needs to be informed.

4b.1 Passenger will be shown a message that already a booking exists, and to choose different time.

4b.2 resume @ step 4 of Basic Flow.

7d Cab is unavailable at Passengers stipulated time.

6a.1 Passenger can choose different time or date or both and check for cab Availability.

6a.2 resume @ step 4 of Basic Flow.

3. EXCEPTIONAL FLOW:

*Any time during the Cab Booking process,Passenger can abandon the Cab booking process by moving away from the Page.

13e Passenger cancels Booking on Confirm Booking Page.

13e.1 The Passenger cancels booking on the Time&Destination Page.

13e.2 Application prompts for confirmation using message.

13e.2 Passenger confirms the Cancellation.

13e.3 Extend ***CANCEL BOOKED CAB*** [Extension Point: Cancel Booked Cab]

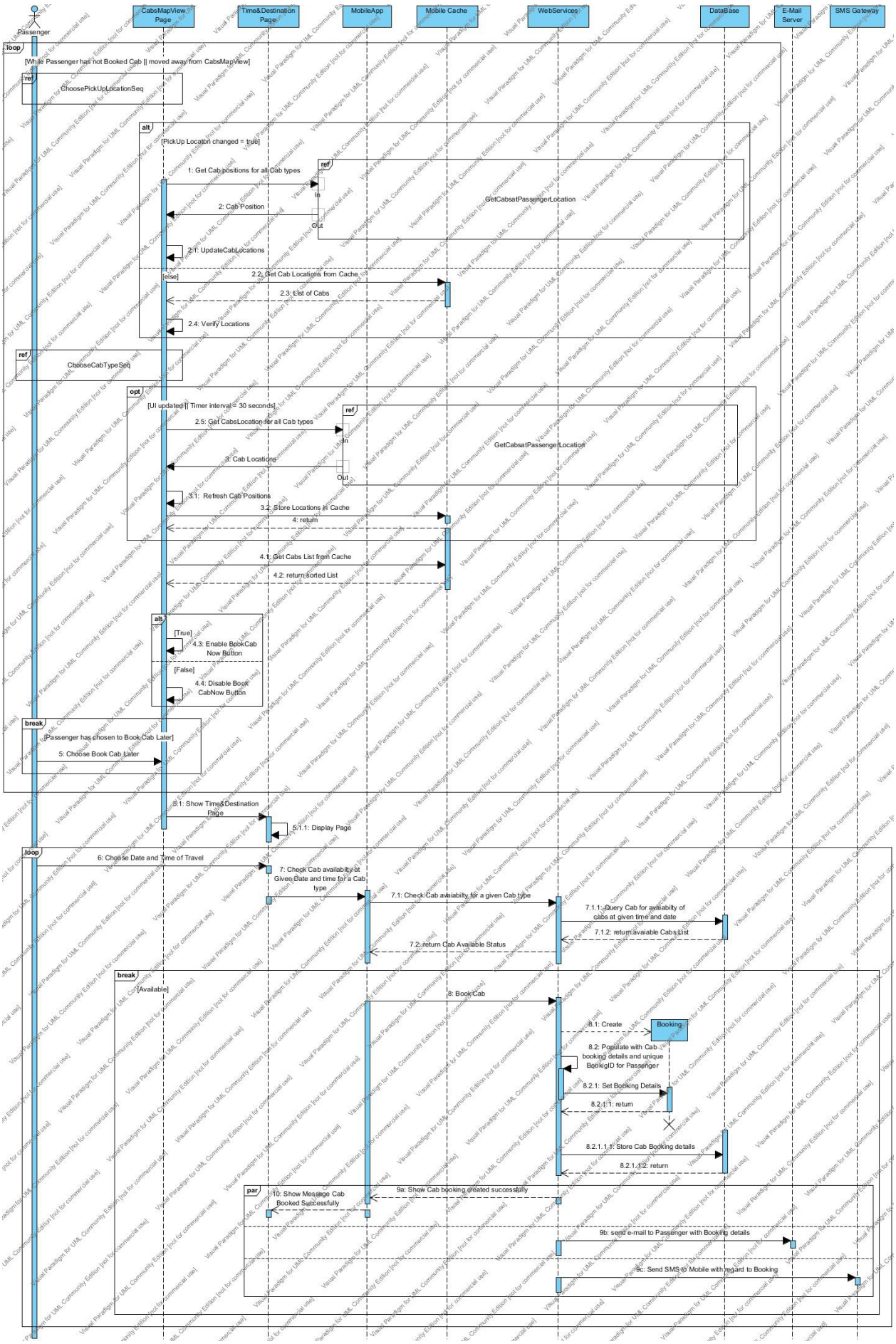
BUSINESS RULES:

1. Passenger will be able to book a cab upto three days in advance.
2. Passenger will be notified 20 minutes before date and time and travel about his cab and Driver details.
3. Designated Cabs need to arrive at Passenger PickUp Location no later than 30 mins after booking.
4. Waiting Time charge at Passenger PickUp Location is free for the first ten minutes and later a surcharge of INR 50.00 would be charged for each additional 30 mins.
5. An Unique Booking ID has to be generated for Passenger booking and has to be 6 digits long.
6. Destination fields in the Booking data entity will be optional and not a neccasary field.
7.

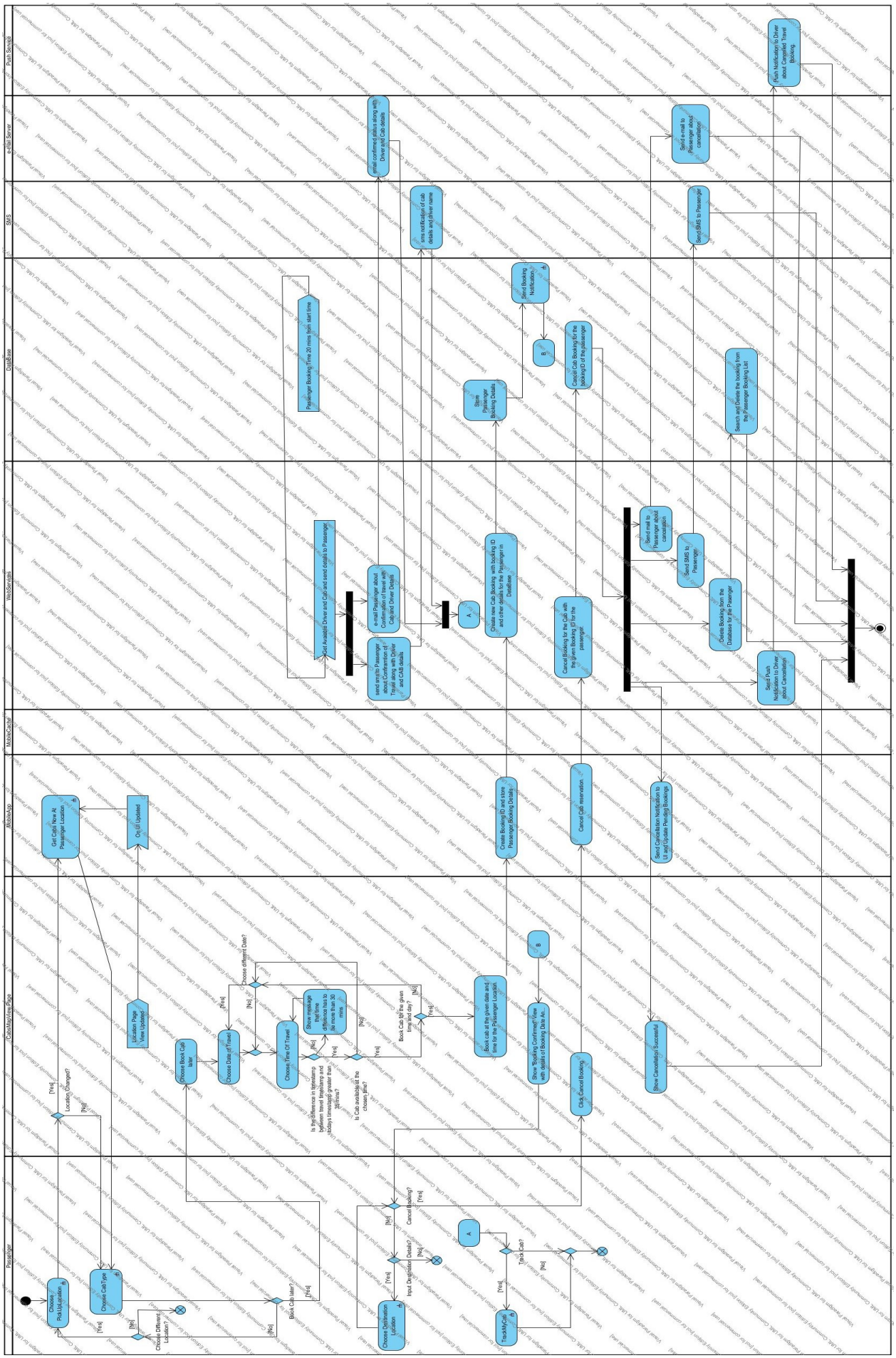
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS:

1. Time&Destination Page shall have Confirm and Cancel buttons.
2. Pickup location accuracy will depend on the GPS accuracy and Mapping detail for that location.

The sequence and activity diagram for the use case are shown in the following diagrams.



Sequence diagram for Book Cab Later



Activity Diagram for Book Cab Later.

2.1.8 USE CASE 08 : SHOW MAPVIEW

ID: UC 08

DESCRIPTION: Show MapView is the base usecase for ***SHOW CABMAPVIEW*** and ***SHOW DESIGNATEDCABMAPVIEW*** usecases.Show MapView renders the Map and passenger location along with homing widget. This view is extended by derived views to render other information.

ACTORS: Passenger

SECONDARY ACTORS: GPS API,MappingService,Geocoding API.

DEPENDENCIES: UC 01, UC 02

PRE-CONDITION:

Passenger is Logged in to the Application or the Passenger wants to track his booked cab.

POST-CONDITION(S):

SUCCESS END CONDITION:

Map is rendered in MapView showing Passenger Location along with homing widget.The Passenger location is shown as icon on the Map.

FAILURE END CONDITION(s):

Map with Passenger location is not rendered.

FLOW OF EVENTS

BASIC FLOW:

1. Application requests Passenger Location from the GPS API.
2. GPS API returns the GPS coordinates of the Passenger Location.
3. Application requests Map from Mapping services based on Passenger Location.
4. Mapping services returns the Map to be rendered on the MapsViewPage.
5. The Application renders the Passenger Icon at Passenger PickUp Point on the Map. Homing widget is also displayed on the Map.The Location of the Passenger is shown as address in the Address textbox of the View.

ALTERNATIVE FLOW:

2a Application is not able to communicate with GPS API.

2a.1 Passenger Location is taken previous known location from Cache.
2a.2 resume @ step 3 in Basic Flow.

EXCEPTIONAL FLOW:

1e GPS API is not able to locate Passenger Location.

1e.1 MapView reports an error and stops rendering the view.

1f Passenger Location is not in Cab Service Area.

1f.1 The Application reports the Passenger PickUp Locations is not in the cab service area.

BUSINESS RULES:

1. Mapping service would return a Map based on bounding box calculations at Passenger Location and at particular zoom level defined in the Application.
2. Maps should provide street level details for locations within Cab Service area where the Cab company provides it services to Passenger.
3. The Application has to detect if Location services and Mapping services are available on the Mobile based on the Mobile OS.
4. In case the Mobile Application is not able to find the Location services or Mapping APIs, it needs to report to the Passenger the name and versions of the Location services or Mapping services it needs for its functioning.
5. Mapping APIs used in Mobile Application must provide touchscreen gestures

including zooming and panning capability.

6. Mapping APIs must support overlay of objects for drawing Passenger Locations etc.

IMPLEMENTATION CONSTRAINTS AND SPECIFICATION:

1. MapView depends on Location services and Mapping service specific to the platform(OS) for the functioning of this feature.Smart phones not having Location services(GPS) or Mapping Service would not be able to view the Map and Passenger Location.
2. MapView would have an text box showing the current Pasenger Location.
3. Street level details on the map would depend on the tiled cache and hence would show the details to which information is present in the Map.

2.1.9 USE CASE 09 : SHOW CABMAPVIEW

ID: UC 09

DESCRIPTION: This use case is a subcase of ***SHOW MAPVIEW*** and it extends the feature of the base case. The ***SHOW MAPVIEW*** shows a Map at a given Passenger location and the address is shown in the text area of the View. This view extends the base use case by showing cabs for a chosen cab type, available on the map in the vicinity of Passenger Location. The Passenger will have the capabilty to choose a cab type and view the cab locations on the map for that cab type.It also has the capabilty to choose Passenger location based on his favourite locations, recommended locations or location based on moving his location dynamically on the map. The homing device widget will be used to position the Passenger back to his current location.

PARENT USE CASE:SHOW MAPVIEW (USECASE ID: 08).

ACTORS: Passenger

SECONDARY ACTORS: Application and actors inherited from base view.

DEPENDENCIES: UC 01**,** UC 02

PRE-CONDITION: The Passenger is logged in to the Mobile Application and wants view Avaliable Cabs on the Map based on Cab type.

POST-CONDITION(S):

SUCCESS END CONDITION:

The Passenger is able to View Cabs avaiable in the vicinity of Passenger location on the Map based on the cab type chosen. The Passenger has the option of booking a cab immediately or at a later date and time.

FAILURE END CONDITION(s):

Passenger is not able to View Cabs on Map.

FLOW OF EVENTS

BASIC FLOW:

1. Include ***SHOW MAPVIEW.***
2. The CabsMapView places CabType and Booking widgets on the View.
3. The Application requests webservices to send a list of cabs sorted by cab type for the given Passenger location.
4. Web services forwards this request to the Database.
5. Database queries all the Cabs available in the vicinity of Passenger location and send the list containing the Cab Locations to webservices.
6. Webservices sends the List to the Application for CabsMapView to render Cab Locations.

7. The Application stores the List of cab locations in the Mobile Cache.
8. CabsMapView renders the cab on the Map based on the cab type chosen.By default “Nano” car type is the default car type.

ALTERNATIVE FLOW:
EXCEPTIONAL FLOW:
BUSINESS RULES:

1. The Cab types provided by the Cab campany are Tata Nano, Hatchbacks and Sedans.

CONSTRAINTS AND SPECIFICATION:

1. The accuracy of the cab location would depend on the internet speed and frequency of updation of Cab positions in the Server database.
2. The server should have suitable storage to store incoming GPS data for the fleet of cars.
3. Each of the car types will be rendered on the map with car icons representing these three cab types.
4. The Book Cab widgets will be placed at the bottom of the screen and the Cabtype Widgets will be place above.

2.1.10 USE CASE 10 : SHOW DESIGNATEDCABMAPVIEW

ID: UC 10
DESCRIPTION: ***SHOW DESIGNATEDCABMAPVIEW*** is a subcase of ***SHOW MAPVIEW*** and it extends the features of the base use case. The ***SHOW MAPVIEW*** use case shows the Map in the vicinity of Passenger location along with the address of the Pasenger . This view extends the base use case by showing Passenger's designated Cab relative to the Passengers Location and the path in which it would travel to reach Passengers Pickup Location.
PARENT USECASE: SHOW MAPVIEW(USECASE ID: 08)
ACTORS: Passenger
SECONDARY ACTORS: Application and actors inherited from base view.
DEPENDENCIES: UC 01,UC02.
PRE-CONDITION: The Passenger is logged in to the Mobile Application and has booked a cab.The Passenger is in 20 minutes of travel time and wants to track his designated Cab and Passenger details.
POST-CONDITION(S):
SUCCESS END CONDITION:
The Passenger is able to view his designated Cab relative to Passenger Location and see the recommended or optimized path between the Cab and PickUp Point.
FAILURE END CONDITION(s):
Passenger is not able to track his Cab on the Map.
FLOW OF EVENTS
BASIC FLOW:

1. Include ***SHOW MAPVIEW.***
2. The Application requests webservices to send Cab and Driver details along with Optimized path for the cab to reach Passenger PickUp location.
3. The Application forwards the details to webserver.
4. The webserver requests the database to retrieve Cab Booking details for the Passenger.
5. Database searches for the Cab booking that is in status at that time and retrieves the Cab booking along with Cab and Driver details.
6. Webservice calculates the optimized path between the Cab and Passenger Location and forwards Cab booking,Cab,Driver and Optimized path to Mobile Application.
7. The Application stores the cabs GPS location and Optimized path in the Cache.
8. DesignatedCabView renders Cab location on the Map.The path between the Cab Location and Passenger is shown.

ALTERNATIVE FLOW:

EXCEPTIONAL FLOW:
*****At any time the Passenger can move away from this Page and stop the View being rendered.
BUSINESS RULES:
***The Passenger can change his recommended path at any time.**
CONSTRAINTS AND SPECIFICATION:

1. The cab location would be rendered by the cab type icon.
2. The path between the cab and Passenger would be shown in green.

2.1.11 USE CASE 11 : TRACK CAB

ID: UC 11

DESCRIPTION: This use case describes the process of tracking a booked cab. It uses the ***SHOW DESIGNATEDCABMAPVIEW***.The view will show the cab loation relative to the Passenger location and the optimized Path it would travel on. The position of the cab would be updated every 30 seconds and the new Position of the cab will be shown on the Map. Once the Cab has reached the Passenger PickUp Location the Application will stop polling for Cab position.

ACTORS: Passenger

SECONDARY ACTORS: Server.

DEPENDENCIES: UC 01, UC 02.

PRE-CONDITION: The Passenger has book a cab immediately or booked a cab at later date. The Pasenger wants to track the location of designated cab to his location. For a Passenger who has booked a cab at a later date he would be able to track designated cab 20 mins prior to his journey time and would be able to avail it from Track Cab menu option. In case of a passenger who has booked his cab immediately for travel("Book Cab Now"), the designated Cab can be tracked by clicking on the Track Cab option on the Booking Confirmed Page.

POST-CONDITION(S)

SUCCESS END CONDITION:

The Passenger is able to track the designated cab on the map relative to his current Location and view the Driver details of the Cab.

FAILURE END CONDITION(s):

Passenger is not able to track the cab.

Passenger has cancelled.

FLOW OF EVENTS

BASIC FLOW:

A1.Include ***SHOW DESIGNATEDCABMAPVIEW.***

A2.Show the position of the Cab on the Optimized path between the Cab and the Passenger.

The Application polls the server every twenty minutes for change in cab location:

B1.Loop till Designated Cab has reached the Passenger PickUp Location or has traversed away from the Page.

B1.2 Application requests Cab Location every 30 seconds from webservice .

B1.3 Web service requests database to give the cab location.

B1.4 Database retrieves the latest cab position.

B1.5 Webservice forwards the Cab location data to Application.

B1.6 Application stores it in Cache.

B1.7 Track Cab page updates the Cab Location from Cache.

ALTERNATIVE FLOW:

B1.5a The Cab driver chooses an alternate path instead of the optimized path.

B1.5a.1 Show the Cab position on the Map.(recalculate?)

B1.5a.1 resume @ B1

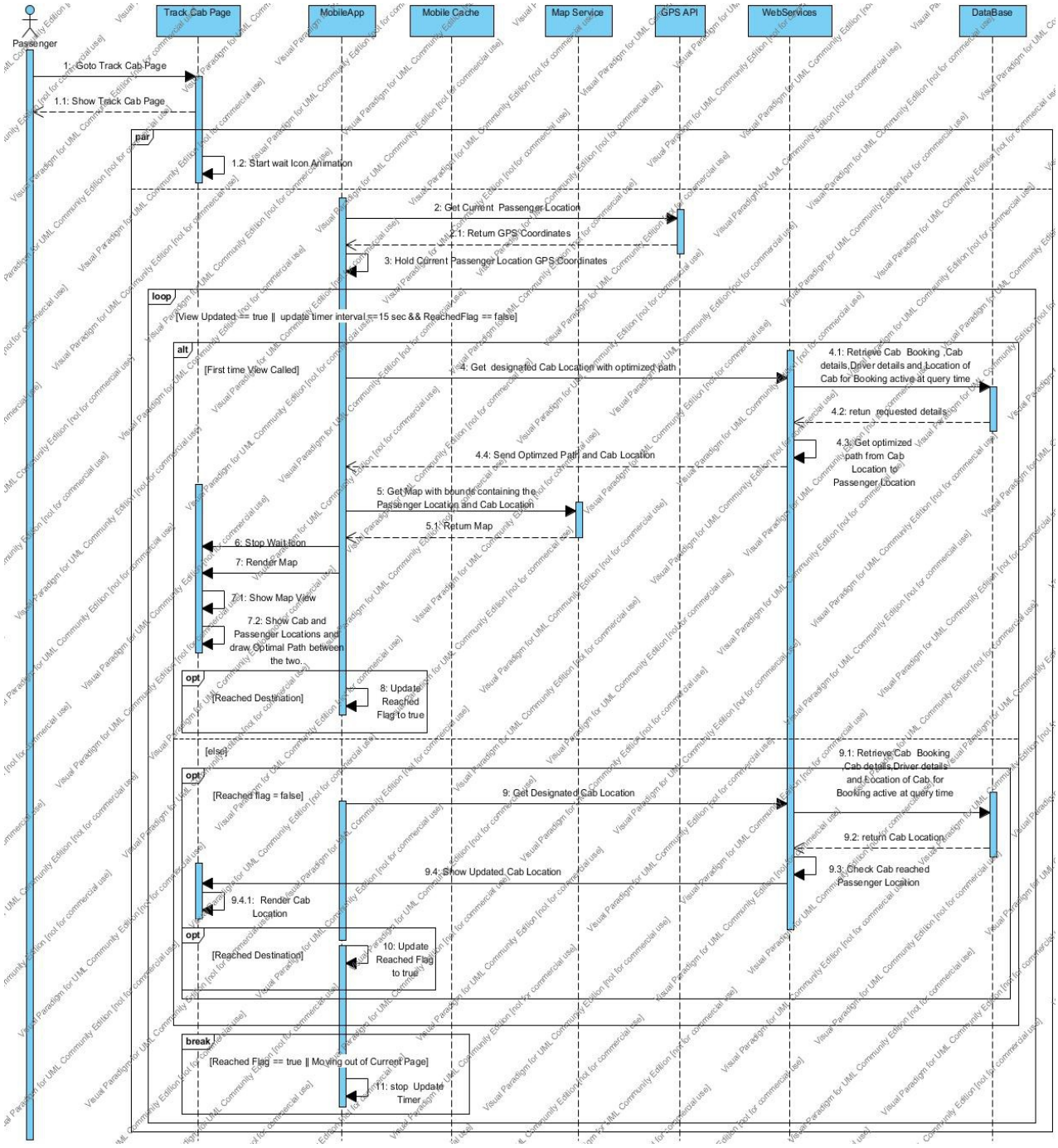
EXCEPTIONAL FLOW:

BUSINESS RULES:

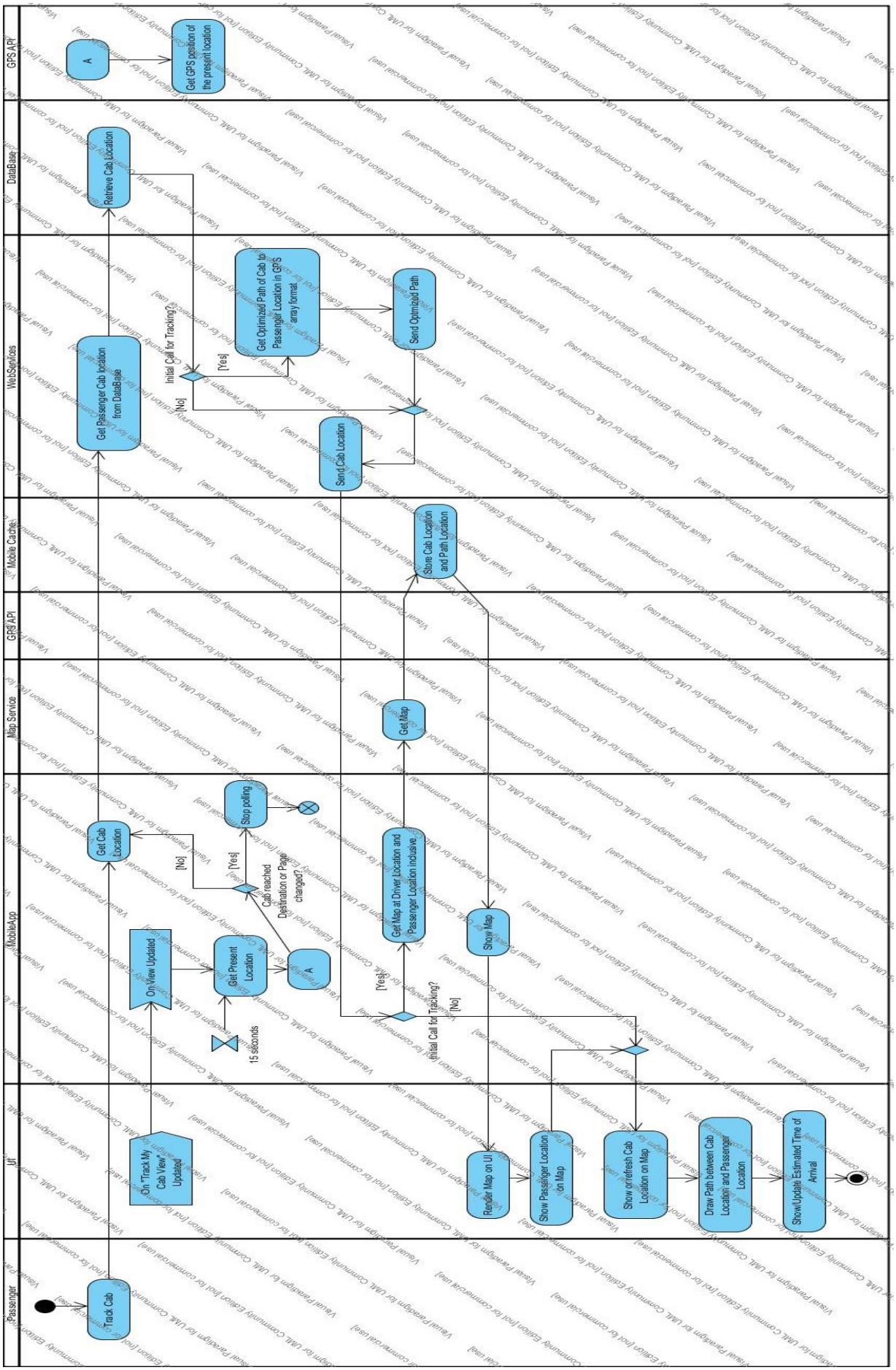
Cab company would not be held responsible for cab drivers choosing to travel on alternate path than that of the Optimized path.

IMPLEMENTATION CONSTRAINTS AND SPECIFICATION:

The sequence and activity diagram for the use case are shown in the following diagrams.



Sequence Diagram for Track Cab



Activity Diagram for Track Cab

2.1.12 USE CASE 12 : REFILL WALLET

ID: UC 12

DESCRIPTION: This use case describes the process of adding money to Wallet . The Passenger can use the Wallet feature provided in the Application to pay for travel.The Wallet helps pay for travel and would be directly debited from the Passengers Wallet to Driver, if sufficient balance exists in Wallet. The Wallet can be filled with Indian currency. Credit/Debit cards or NetBanking can be used by the Passenger to transfer money to the Wallet.

ACTORS: Passenger

SECONDARY ACTORS: Wallet API,Payment gateway, Escrow Account, Wallet Provider Server.

DEPENDENCIES: UC 01**,UC 02**

PRE-CONDITION: The Passenger has logged in to the Application and wants to View and refill his Wallet.

The Passenger has a valid card(Debit/Credit) or net banking account to refill his Wallet.

Passengers credit/debit card or bank has sufficient balance to be credited to Wallet.

The Passenger has already registered with the Cab Wallet provider.

POST-CONDITION(S)

SUCCESS END CONDITION:

The Passenger has successfully added amount to his Wallet.

FAILURE END CONDITION(s):

The Passenger fails to update his Wallet by the amount.

FLOW OF EVENTS

BASIC FLOW:

1. The Passenger inputs the required amount in INR on the Wallet Page and clicks on “Refill” Button.
2. The Passenger is redirected to the Payment Page.
3. The Application requests the Wallet provider APIs to get all the card information for the customer.
4. The request is sent to the web services.
5. The web services forward the request to Wallet provider secure server.
6. The secure server returns all the card details.
7. The card details are sent back to the Application and the List of already registered cards are displayed om the Payment Page.
8. Passenger chooses to refill his wallet by a new credit card.
9. Passenger enters his Credit card credentials and card CVV field and clicks on “Pay”.
10. The Application forwards the details to web server.
11. The Web server forwards the data to Payment Gateway.
12. The payment Gateway redirects the user from Payment Page to their 3d secure page.
13. The Passenger enter his pin and validates his account.
14. The amount is debited from the credit card and place in the Passengers escrow account.
15. The Payment Gateway redirects the Passenger to the Payment Page.
16. The Payment page shows the updated Wallet amount.

ALTERNATIVE FLOW:

8a Passenger choose to refill the Wallet using debit card.

8a.1 Passenger fills the debit card details and CVV and submits the Data.

8a.2 resume @ 10.

8b Passenger chooses to refill his Wallet using Net Banking.

8b.1 Passenger chooses the bank name from a pull down List of Bank names.

8b.2 Passenger submits the Data.

8b.3 The Application forwards the details to web server.

8b.4 The Web server forwards the data to Bank Portal.

8b.5 The BankPotal redirects the Passenger from payment Page to Login Page.

8b.6 Passenger enters his login information.

8b.7 Login is validated and the money is transferred to Passengers Escrow Account.

8b.8 resume @ step 16 of basic flow.

8c. Passenger chooses to refill his Wallet account using registered Credit / Debit card.

8c.1 Passenger chooses the registered Credit/Debit cards.

8c.2 Passener enters the CVV field for the Card.

8c.3 resume @ step 10 of basic flow.

The Passenger enter his pin and validates his account.

EXCEPTIONAL FLOW:

13e Passenger enters wrong 3D PIN for the Credit/Debit card.

13e.1 Passenger enters wrong 3d pin in the payment gateway page.

13e.2 Passenger is redirected to the Payment page.

13e.3 Payment page shows message that there is an error in transaction.

13e.4 Use case ends.

14e Insufficient balance in Credit/Debit Card.

14e.1 There is insufficient balance in Passengers account.

14e.2 Payment Gateway redirects the Passenger to Payment page.

14e.3 Payment page shows error that there is insufficient balance.

14e.4 Use case ends.

8b.6e Passenger enters invalid Login details.

8b.6e.1 Passenger enters wrong Login information.

8b.6e.2 Bank Portal verifies Login details and redirects Passenger to Payment Page.

8b.6e.3 The Payment page shows message of Invlid Login.

8b.7e Passenger has got insufficient balance in his account.

8b.7e.1 Bank Portal verifies account balance and redirects the Passenger to Payment page.

8b.7e.2 Payment Page shows Message with regard to insufficient balance.

8b.7e.3 Use case ends.

BUSINESS RULES:

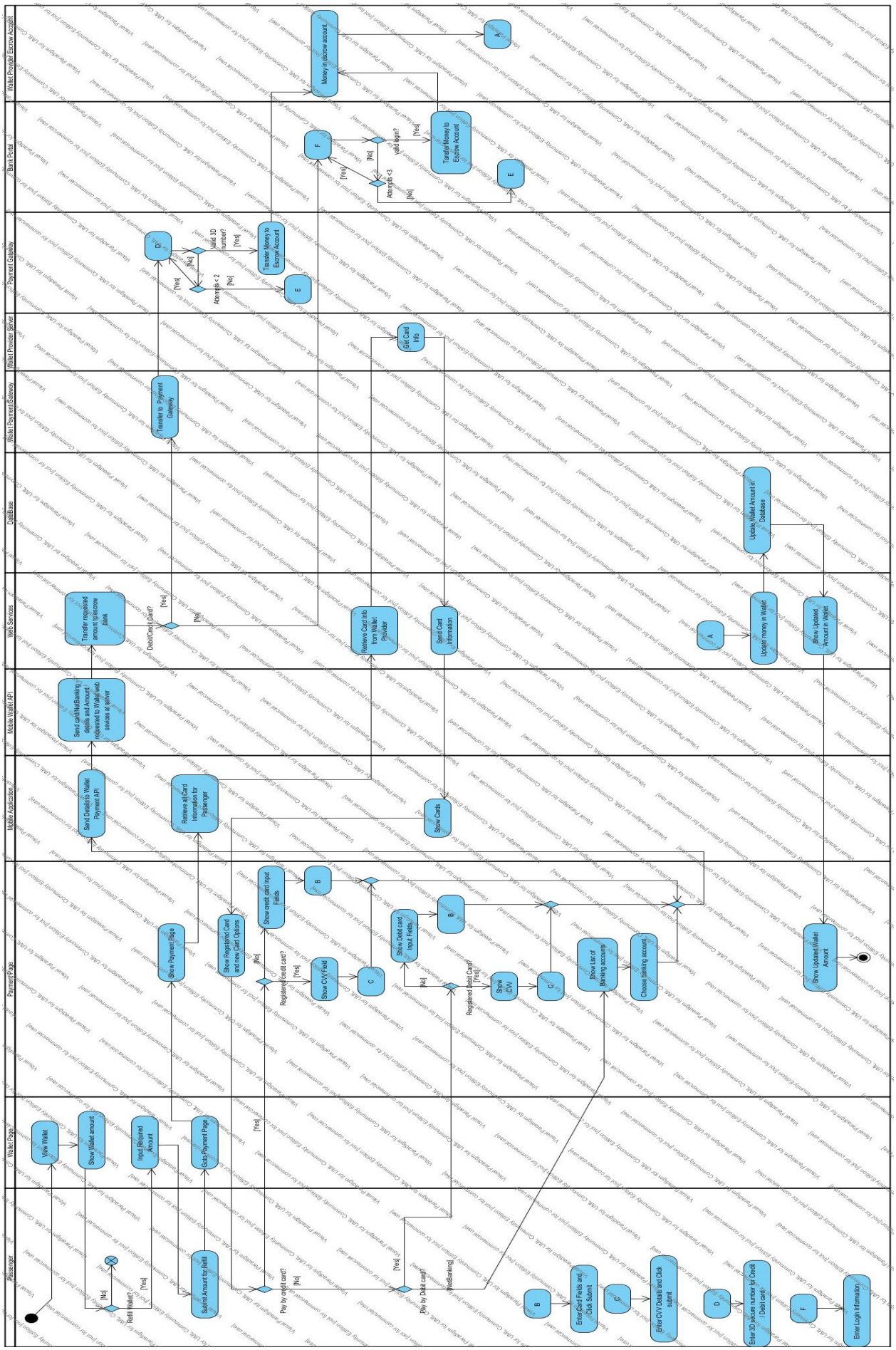
1. All the money transacted from the Passenger would be stored in Escrow account.
2. The Cab Company's merchant account would be credited for Passenergers payment.
3. The account details of Passengers credit or debit cards would be stored on a Wallet providers secure server.
4. The Wallet provider shall follow strict guidelines of Reserve Bank of India(RBI)

for the amount that can be debited from Passengers account and maximum limit that can be as balance in the Passengers escrow account.

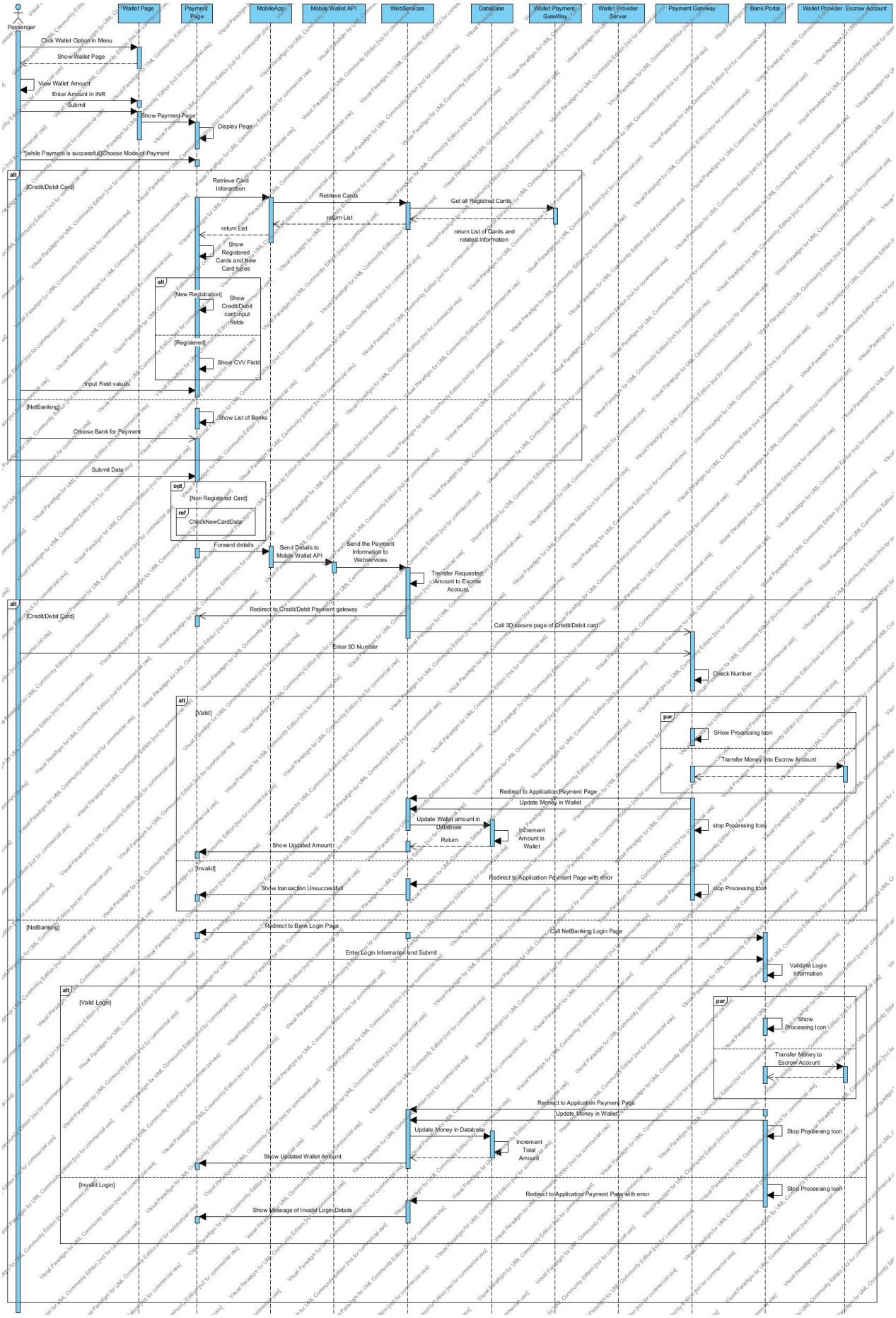
5. The Passengers balance in the escrow account would be shown as Passengers Wallet amount .
6. Interest earnings on the money deposited in the account will be accounted as redeem to Passenger in the form of travel points.
7. The Cab Wallets server would be located in India.

CONSTRAINTS AND SPECIFICATIONS:

The sequence and activity diagram for the use case are shown in the following diagrams.



Activity Diagram for Refill Wallet.



Sequence Diagram for Refill Wallet.

2.1.13 USE CASE 13: PAY FOR TRAVEL

ID: UC 13

DESCRIPTION: This use case describes the process of travel Payment on reaching Passengers destination Location. The driver will end the travel on his mobile, once Driver confirms from Passenger that he has reached the desired location. Webservice on the server would recieve travel end confirmation for the Passenger travel and calculate total travel amount. Web services would check if there is sufficient blance in Passengers Wallet so that it can directly deduct the amount from it. If there is sufficient balance in Wallet, the amount is deducted from Wallet and SMS is sent to the Passenger with regard to the transaction, else the passenger is requested to pay the amount in cash. The travel invoice is sent to the Passengers e-mail and as SMS of the travel summary is sent to the Passenger.

ACTORS: Passenger

SECONDARY ACTORS: Server,Wallet,Wallet Provider's Server, Cab Providers Merchant Bank.

DEPENDENCIES: UC 01**, UC 02**

PRE-CONDITION: Passenger has his mobile and has reached his destination location. The Passenger needs to pay for travel.

POST-CONDITION(S):

SUCCESS END CONDITION:

The Passenger has paid for travel using Wallet.

FAILURE END CONDITION(s):

The Passenger has paid by cash for the travel based on the cab meter.

FLOW OF EVENTS

BASIC FLOW:

1. The Pasenger confirms to the Driver that he has reached his destination.
2. The Driver stops the cab meter manually and clicks on travel end button on his mobile.
3. The Webservice on the server recieves the end of travel message for the Passengers booking.
4. Webservices calculates the total amount for the travel.
5. Webservice checks balance in the Passengers Wallet.
6. Webservice deducts the required amount from the Wallet.
7. Webservice sends the travel invoice to the registered e-mail and sends an SMS with regard to succesful payment through Wallet.
8. Webservice sends notification to the Drivers tab that the amount has been credited to his account.
9. Passenge alights from the cab.

ALTERNATIVE FLOW:

5a. Webservice finds balance less than travel amount.

5a.1 Webservice sends an SMS to Passenger with regard to insufficient balance and asks the Passenger to Pay by cash.

5a.2 Driver is sent an notification to take payment by cash.

5a.3 Passenger pays by cash.

5a.4 resume @ step 9 of Basic Flow.

EXCEPTIONAL FLOW:

7a Webservice messages are not recievable on Passengers Mobile or Passenger mobile lost its charge.

7a.1 Passenger pays for his travel based on the cab meter.

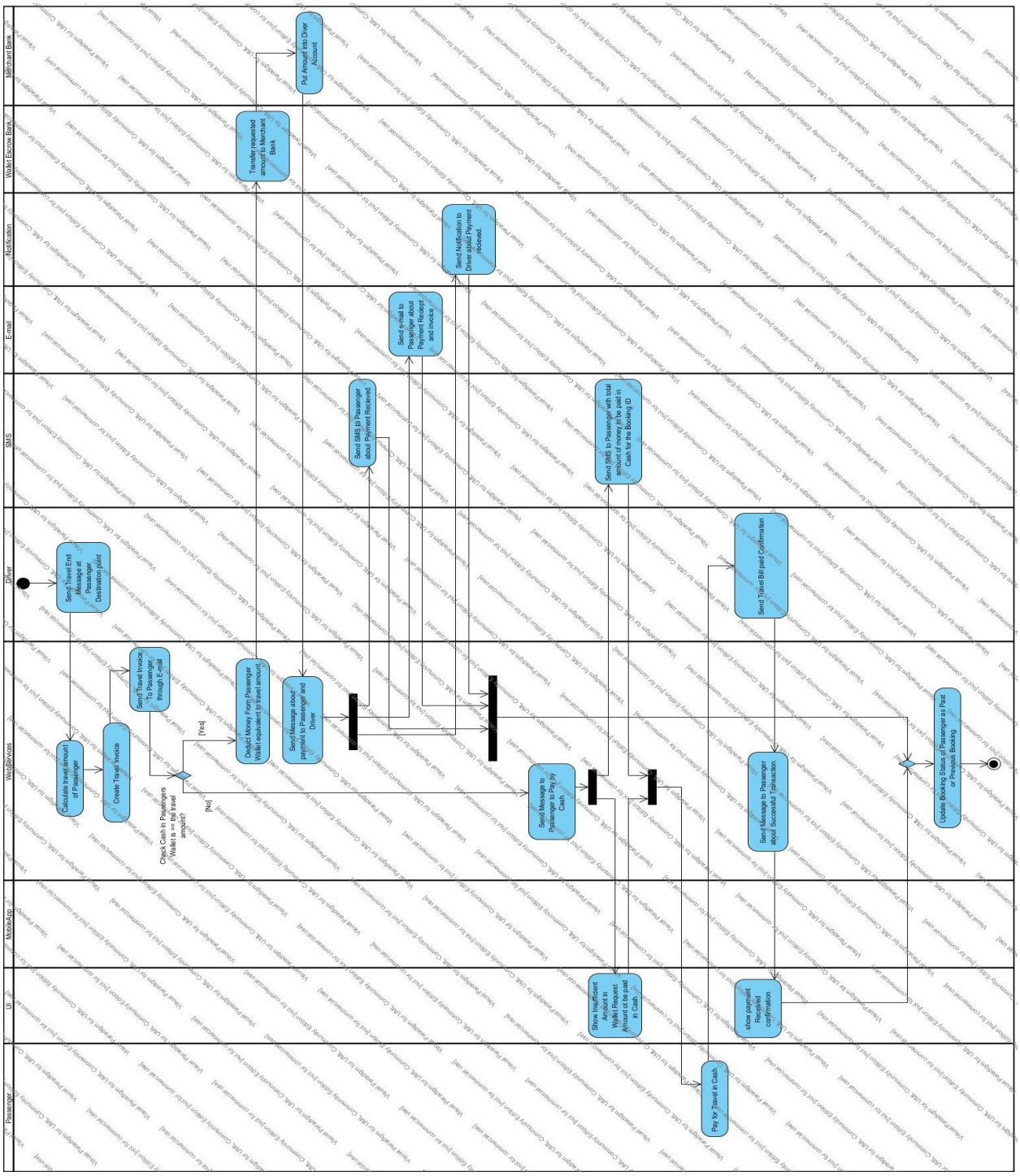
7a.2 Use case ends.

BUSINESS RULES:

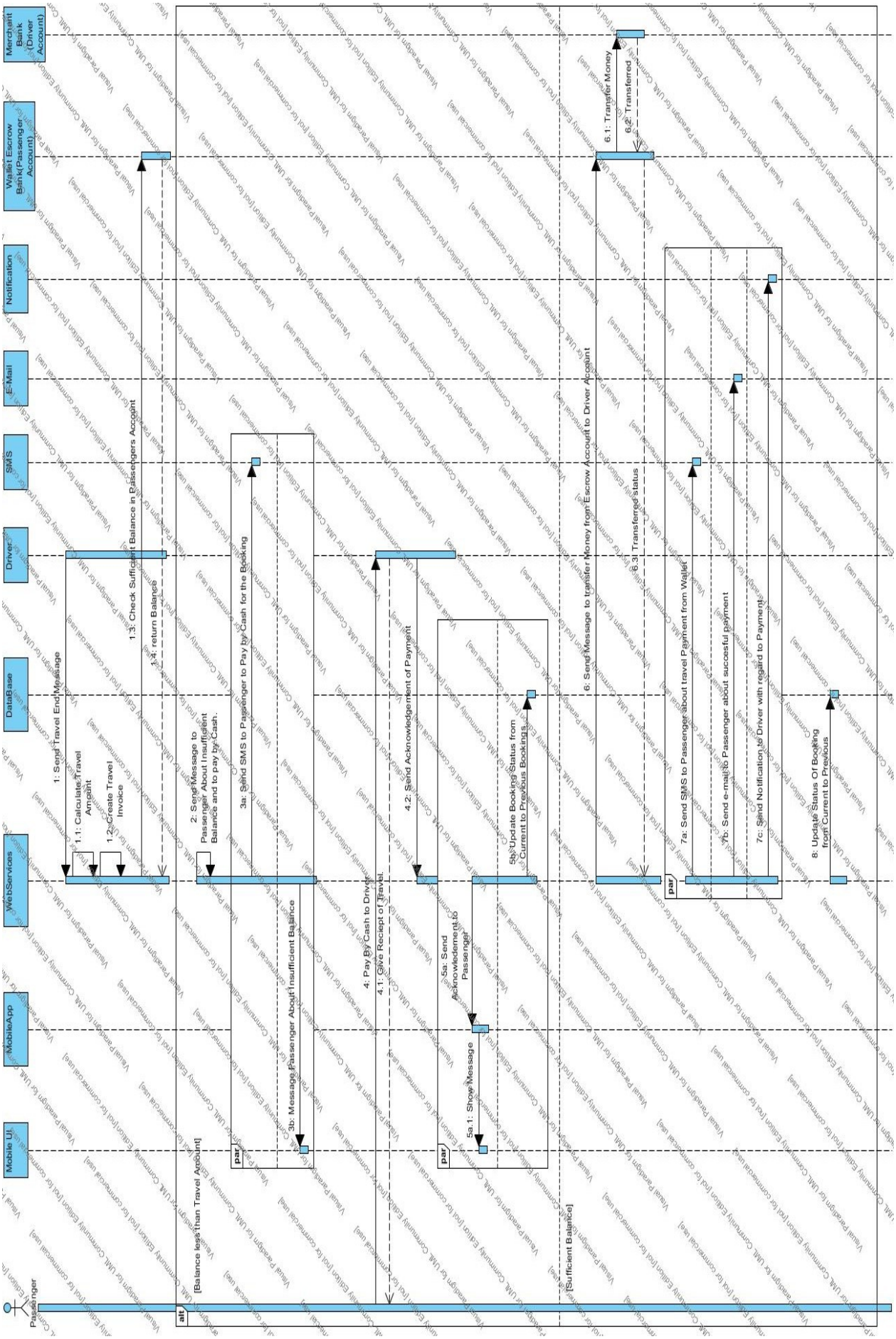
1. Waiting time and toll charges will be aded to travel invoice if the driver has chosen the same.
2. Toll charges will be paid by the Driver at toll gate.

IMPLEMENTATION CONSTRAINTS AND RESTRICTIONS:

The sequence and activity diagram for the use case are shown in the following diagrams.



Activity Daigram for Pay for Travel



Sequence Diagram for Pay for Travel

2.1.14 USE CASE 14 : VIEW FARE CHART

ID: UC 14
DESCRIPTION: This use case describes the process of looking at the Cab fare chart for a city. The base fare and additional rate per kilometer are shown for three cab types irrespective of the model or Make of cabs are shown.
ACTORS: Passenger
SECONDARY ACTORS: Server.
DEPENDENCIES: UC 01**,UC 02**
PRE-CONDITION: The User has logged in to the Application, and wants to check the fare in the service Area(City).
POST-CONDITION(S)
SUCCESS END CONDITION:
The User is able to view fare chart based on the Service area he is located.
FAILURE END CONDITION(s):
The user is not able to view fare chart.
FLOW OF EVENTS
BASIC FLOW:

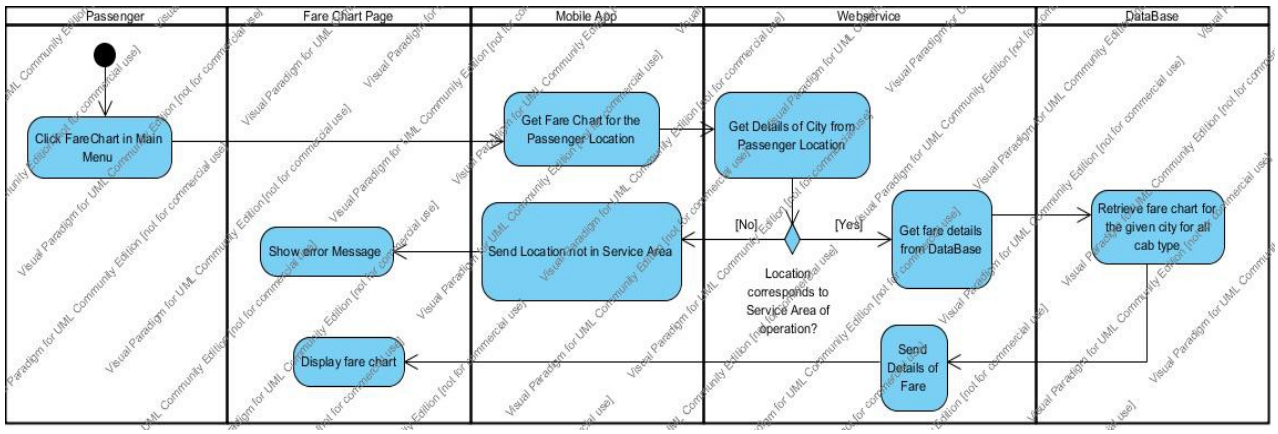
1. The Passenger opts for “Fare Chart” option in the Main menu.
2. The request for fare chart is sent to the Webservices along with the Passenger GPS Location.
3. Websevice runs algorithm on the GPS location and gets the Service area(city).
4. Webservice requests the fare chart from database for that city.
5. Database retrieves the fare details for the city and returns it to webservices.
6. Webservices forwards the fare details to Application.
7. The fare chart Page shows the cab fares for the city sorted by Cabtype.

ALTERNATIVE FLOW:
EXCEPTIONAL FLOW:
3e. The algorithm returns false for finding the service area.
3e.1 The webservice returns Service Area not found flag to Application.
3e.2 The Farechart page shows the error that present location is out of bounds of any service area.
3e.3 Use case stops.
BUSINESS RULES:

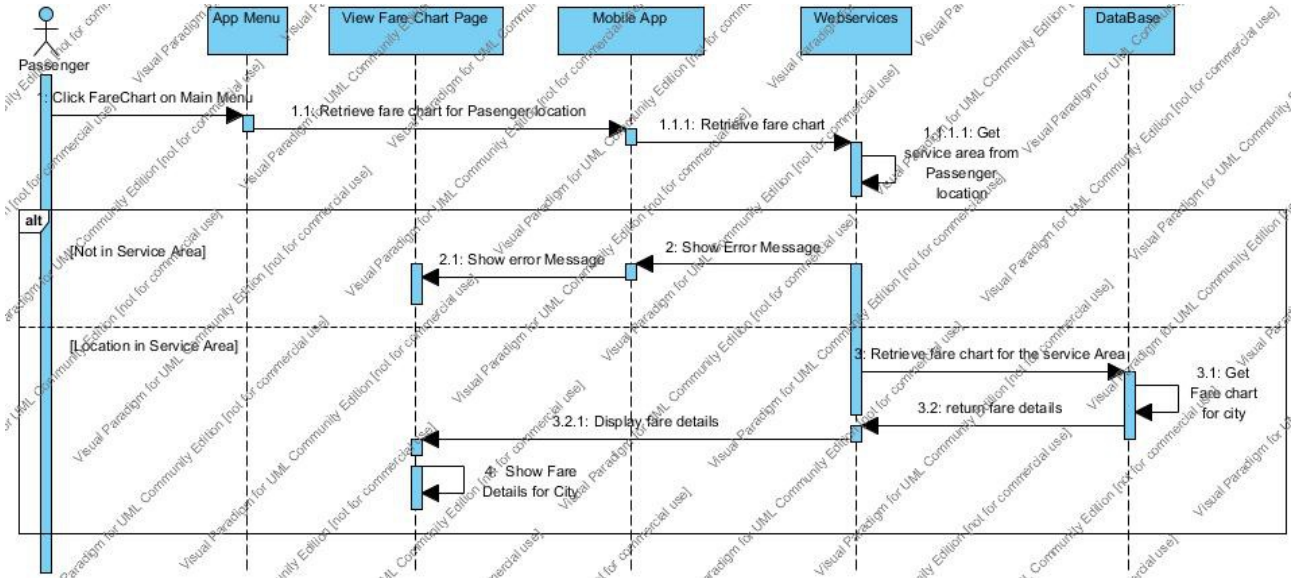
1. Algoirthms are run on the GPS location of the Passenger to find if the Passenger location is within any Service areas(City) in India.
2. This service is meant only for selected cities in which the cab company offers its services.

IMPLEMENTATION CONSTRAITS AND SPECIFICATION:

The sequence and activity diagram for the use case are shown in Fig XX.X and Fig. XX.x respectively.



Activity Diagram for View Fare Chart



Sequence Diagram for View Fare Chart.

2.1.15 USE CASE 15 : SEND FEEDBACK

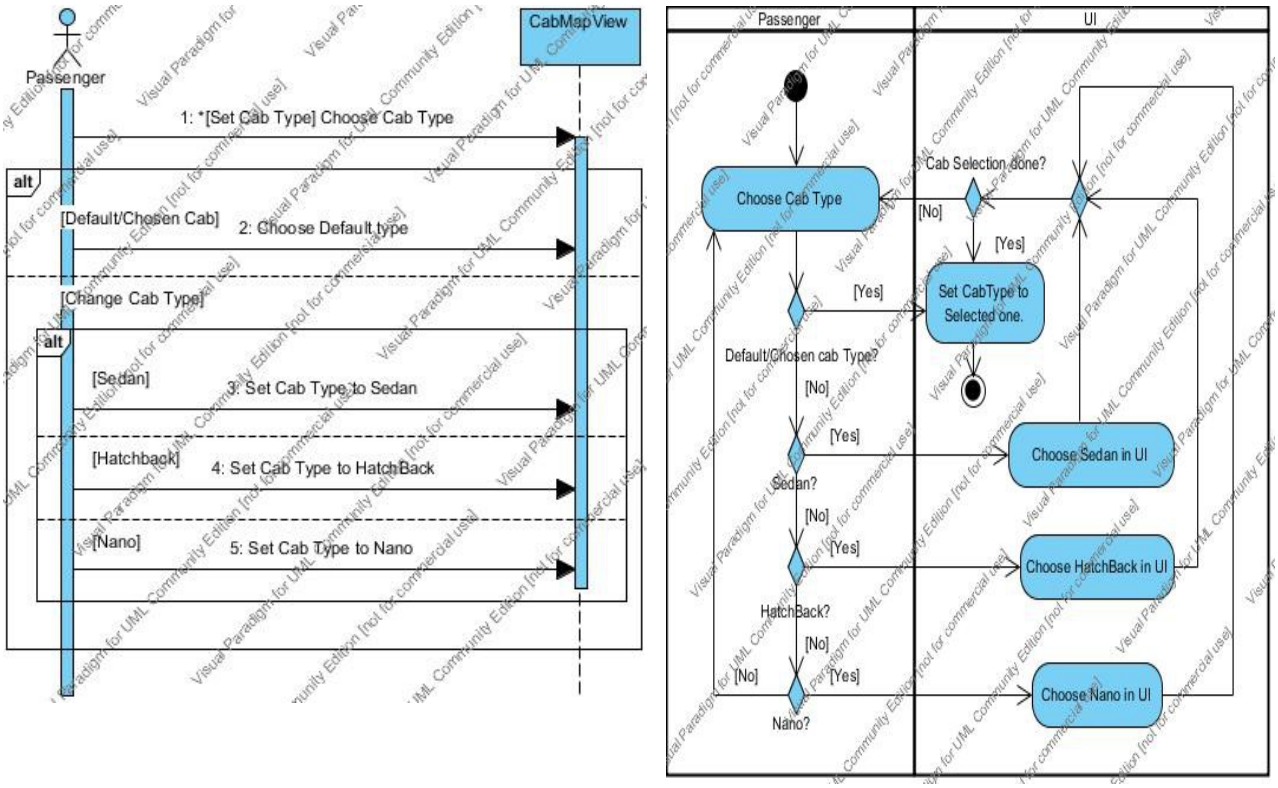
ID: UC 15
DESCRIPTION: This use case describes the process of sending a feedback of the travel done for a cab booking. The Passenger would be able to rate the driver, cab, travel experience ratings and send it.
ACTORS: Passenger
SECONDARY ACTORS: Server
DEPENDENCIES: UC 01**,UC 02**
PRE-CONDITION:
The Passenger has logged in to the application and wants to rate one of his Past bookings travel.The Passenger is viewing bookings in the Past Bookings tab on the Bookings Page.
POST-CONDITION(S)
SUCCESS END CONDITION:
The Passenger is able to send his feedback for a particular booking.
FAILURE END CONDITION(s):
FLOW OF EVENTS
BASIC FLOW:

1. The Passenger clicks on the feedback button for the Booking that he wants to send feed back.
2. The application forwards him to the Send FeedBack Page.
3. The ratings are done and submitted.
4. The rating information is sent to webservices which forwards it
5. The databse stores the feedback in database.
6. A Message is displayed that the feedback is succesfully sent on the FeedBackPage.
7. The Pasenger is redirected to Pending Bookings Tab of the Bookings Page.

ALTERNATIVE FLOW:
EXCEPTIONAL FLOW:

2.1.16 USE CASE 16 : CHOOSE CAB TYPE

ID: UC 16
DESCRIPTION: This use case describes the process of choosing a cab type in CabsMapView from the available Cab types.
ACTORS: Passenger
SECONDARY ACTORS:
DEPENDENCIES: UC 01**, UC 02**
PRE-CONDITION: The Passenger has logged in to the Application and is in the CabsMap View. The User wants to select one cab type from the avaiable cabtypes.
POST-CONDITION(S)
SUCCESS END CONDITION:
The Cabtype is chosen by the user and the use
FAILURE END CONDITION(s):
FLOW OF EVENTS
BASIC FLOW:
ALTERNATIVE FLOW:
EXCEPTIONAL FLOW:
BUSINESS RULES:
IMPLEMENTATION CONSTRAINTS AND SPECIFICATION:



Sequence and Activity Diagram for Choose Cab type.

2.1.17 USE CASE 17 : CHOOSE PASSENGER LOCATION

ID: UC 17

DESCRIPTION: This use case describes the Process of choosing a Passenger PickUp Point Location while booking a cab. The PickUp Loation location can be chosen from Recommended/Favorite Locations List ,can be Manually entered in the Address Location or scrolled on the map to the desired Location. The Passenger Icon would be updated.

ACTORS: Passenger

SECONDARY ACTORS:

DEPENDENCIES: UC 01**, UC 02**

PRE-CONDITION: The User has logged into the Application and is in the CabsMapView and the PickUp location Teext area shows the Address of the current Location of the Passenger.

POST-CONDITION(S)

SUCCESS END CONDITION:

Passenger is able to choose a PickUp Location.

FAILURE END CONDITION(s):

Passenger is not able to choose a PickUp Point.

FLOW OF EVENTS

BASIC FLOW:

1. The Passenger wants to change the Location and enters manually the address in the Location text box.
2. The Application validates the Address.
3. The PickUp Text area shows the address.
4. Passenger Icon is moved to the new Position as shown in the Passenger Text Area.

ALTERNATIVE FLOW:

1a Pasenger wants to choose Location from Favourite List.

1a.1 Passenger clicks on the Favourite Button.

1a.2 Passenger chooses one of the Items from the Favourites.

1a.e resume @ 3 of Basis Flow.

1b Pasenger wants to choose Location from Recommended List.

1b.1 Passenger clicks on the recommended Button.

1b.2 Passenger chooses one of the Items from the drop dowm Reccommede List.

1b.c Resume @ 3 of Basis Flow.

1c Pasenger wants to scroll to a new Location on the Map.

1c.1 Passenger scrolls along the map and stops to choose his new Location

1c.2 On scrolling stop, the Text area shows the PickUp Location.

1c.3 Resume @ 3 of Basis Flow.

EXCEPTIONAL FLOW:

1Ex Passenger has input a location that is out of service area(City) limits.

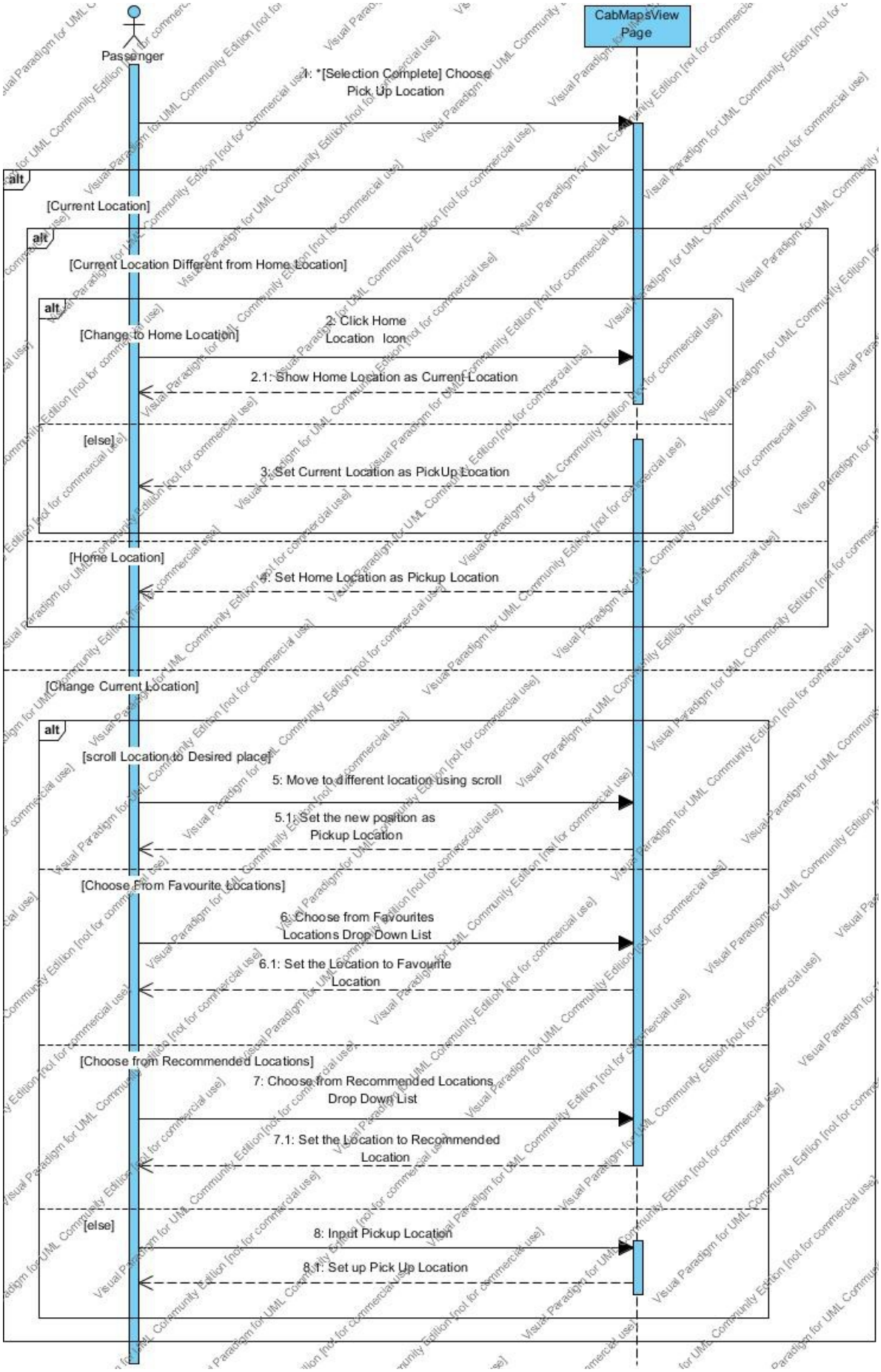
1Ex.1 Passenger is notified about the error using a Message on the CabsMapView.

1cEx Passenger has scrolled to a location beyond the service area(City) limits.

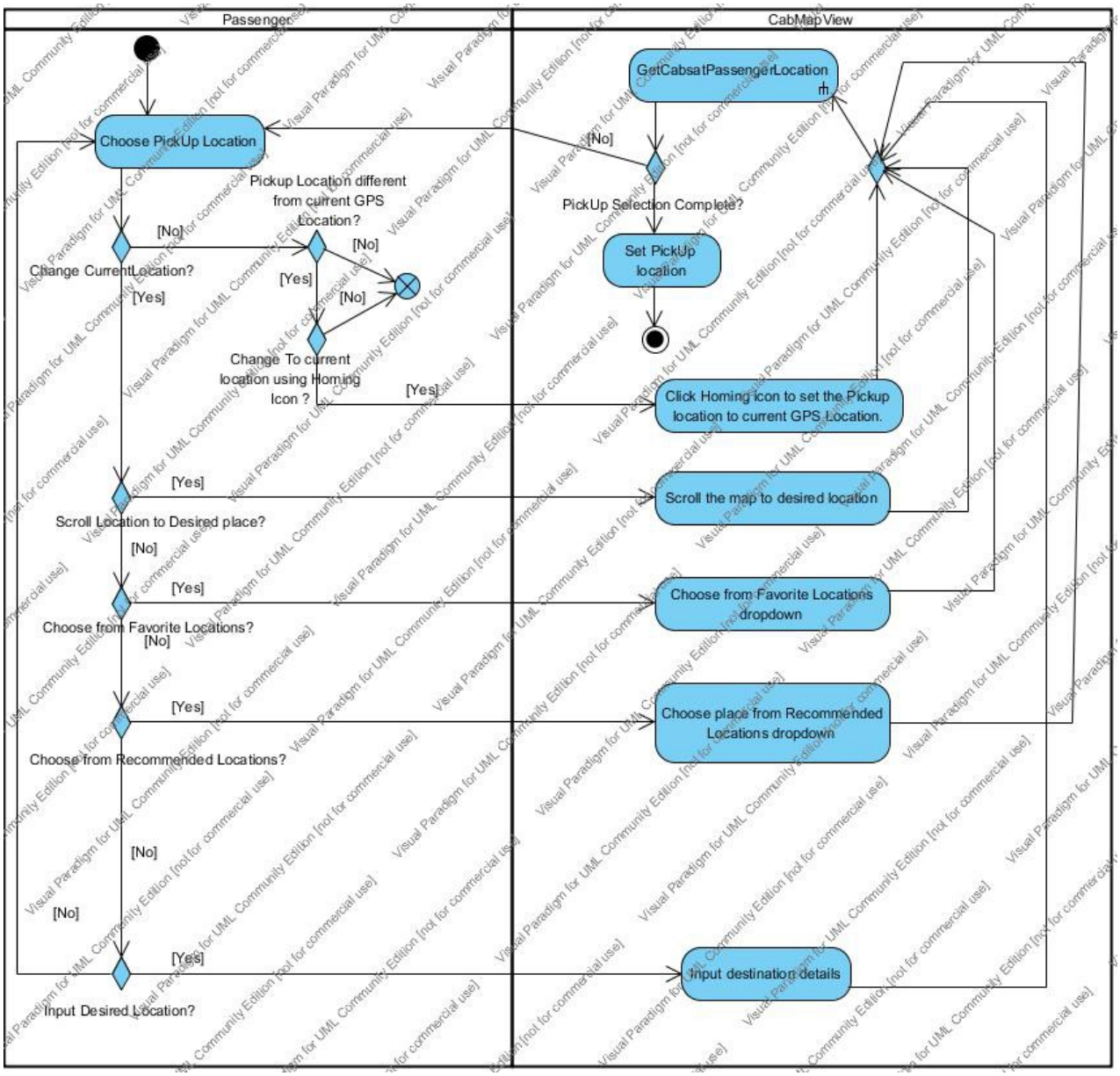
1cEx.1 Passenger is notified of the error.

BUSINESS RULES:

IMPLEMENTATION CONSTRAINTS AND SPECIFICATION:



Sequence Diagram for Choose Passenger Location.



Activity Diagram for Choose Passenger Location.

2.1.18 USE CASE 18 : CHOOSE DESTINATION LOCATION

ID: UC 18

DESCRIPTION: This use case describes the Process of choosing a destination Location after booking a cab. The destination location can be chosen from Recommended,Favorite Locations List or can be manually entered in the destination location.

ACTORS: Passenger

SECONDARY ACTORS:System

DEPENDENCIES: UC 01**, UC 02**

PRE-CONDITION: The passenger has logged in to the Application and has booked a Cab using Book Cab Later or Book Cab Now Usecase.

POST-CONDITION(S)

SUCCESS END CONDITION:

The Passenger has successfully entered his Location details.

FAILURE END CONDITION(s):

The Passenger is not able to find or locate his destination location by the Application.

FLOW OF EVENTS

BASIC FLOW:.

1. The Passenger enters his Destination Location manually by typing the address in the Destination field.
2. The Application is able to validate the location.
3. The destination field shows the Destination Address.

ALTERNATIVE FLOW:

1a. Passenger enters the Destination field using Recommended List.

1a.1 Pasenger clicks on the recommended button and is able to see the List of Recommended locations.

1a.2 Passenger chooses to use one of the locations.

1a.3 Resume @ step 3 of basic Flow.

1b. Passenger enters the Destination fied using Favourite List.

1b.1 Passenger cliccks on Favourites Icon and is able to view the List of his/her Favourite Locations.

1b.2 Passenger opts one of the location that he intends to travel.

1b.3 Resume @ step 3 of basic Flow.

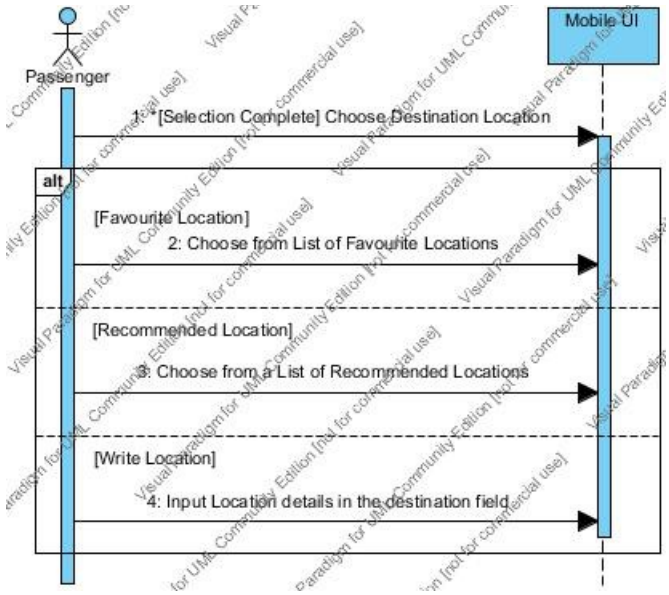
EXCEPTIONAL FLOW:

1Ex Passenger has input a location that is out of service area(City) limits.

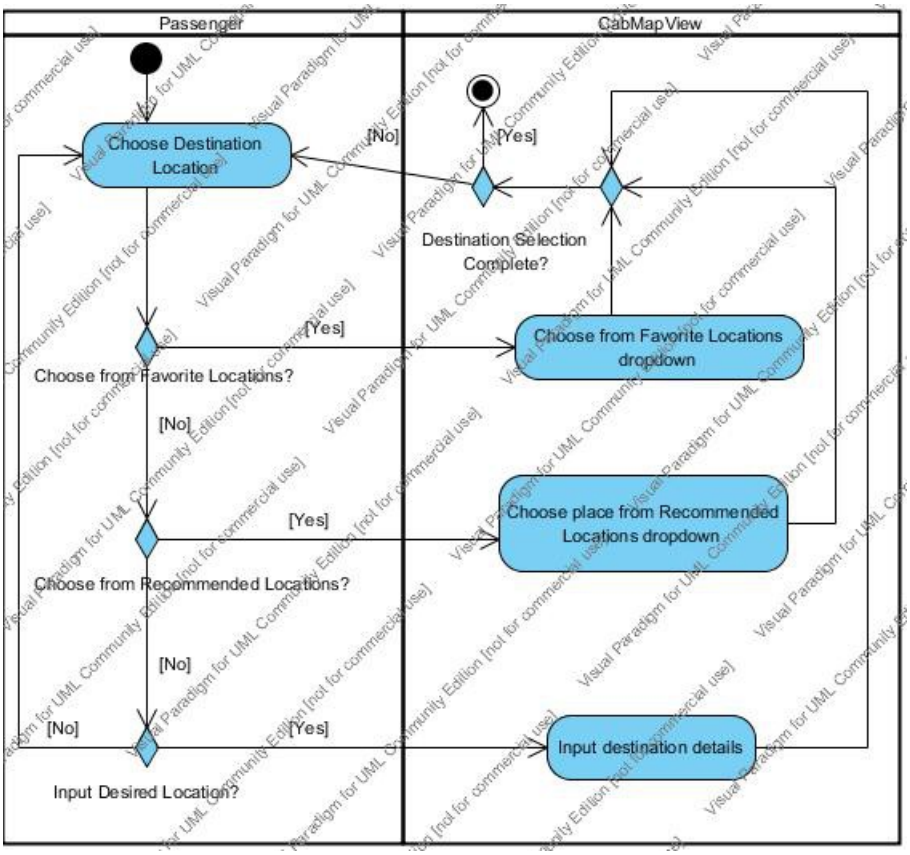
1Ex.1 Passenger is notified about the error using a Message on the CabsMapView.

BUSINESS RULES:

IMPLEMENTATION CONSTRAINTS AND SPECIFICATION:



Sequence Diagram for Choose Destination Location



Activity Diagram for Choose Destination Location.

2.1.19 USE CASE 19 : CANCEL BOOKED CAB

ID: UC 19
DESCRIPTION: This Usecase describes the process of cancelling a booked cab. Once the Passenger has booked a cab, the Passenger can cancel the booking immediately or at a later time depending on the Page on which they are using at that moment.
ACTORS: Passenger
SECONDARY ACTORS: Server.
DEPENDENCIES: UC 01**, UC 02**
PRE-CONDITION: The Passenger has booked a cab and wants to cancel his booking for that travel.
POST-CONDITION(S)
SUCCESS END CONDITION:
The Passenger is able to cancel the booking.
FAILURE END CONDITION(s):
The Passenger is unable to cancel booking.
FLOW OF EVENTS
BASIC FLOW:

1. The Passenger has booked a cab using BookCab Now in CabsMapView.
2. The Passenger is directed to Booking Confimation Page.
3. Passenger clicks on Cancel Cab.
4. Application retrieves the BookingID from the Cache.
5. Application forwards the request to webservices to cancel cab booking for Booking ID.
6. Webservices forwards the Booking ID to Database to delete the booking.
7. The Database deletes the Booking with the Booking ID and return success to webservices.
8. Webservices returns success to Application.
9. The Application deletes the item with Booking ID in the Booking List present in the Cache.
10. Booking Page Display a message that the Booking has been deleted.Booking Page takes the List from the Cache and shows the Pending and Past booking tabs with updated List items.

ALTERNATIVE FLOW:
1a Passenger has booked the cab using Book Cab later and wants to Cancel the Cab booking.

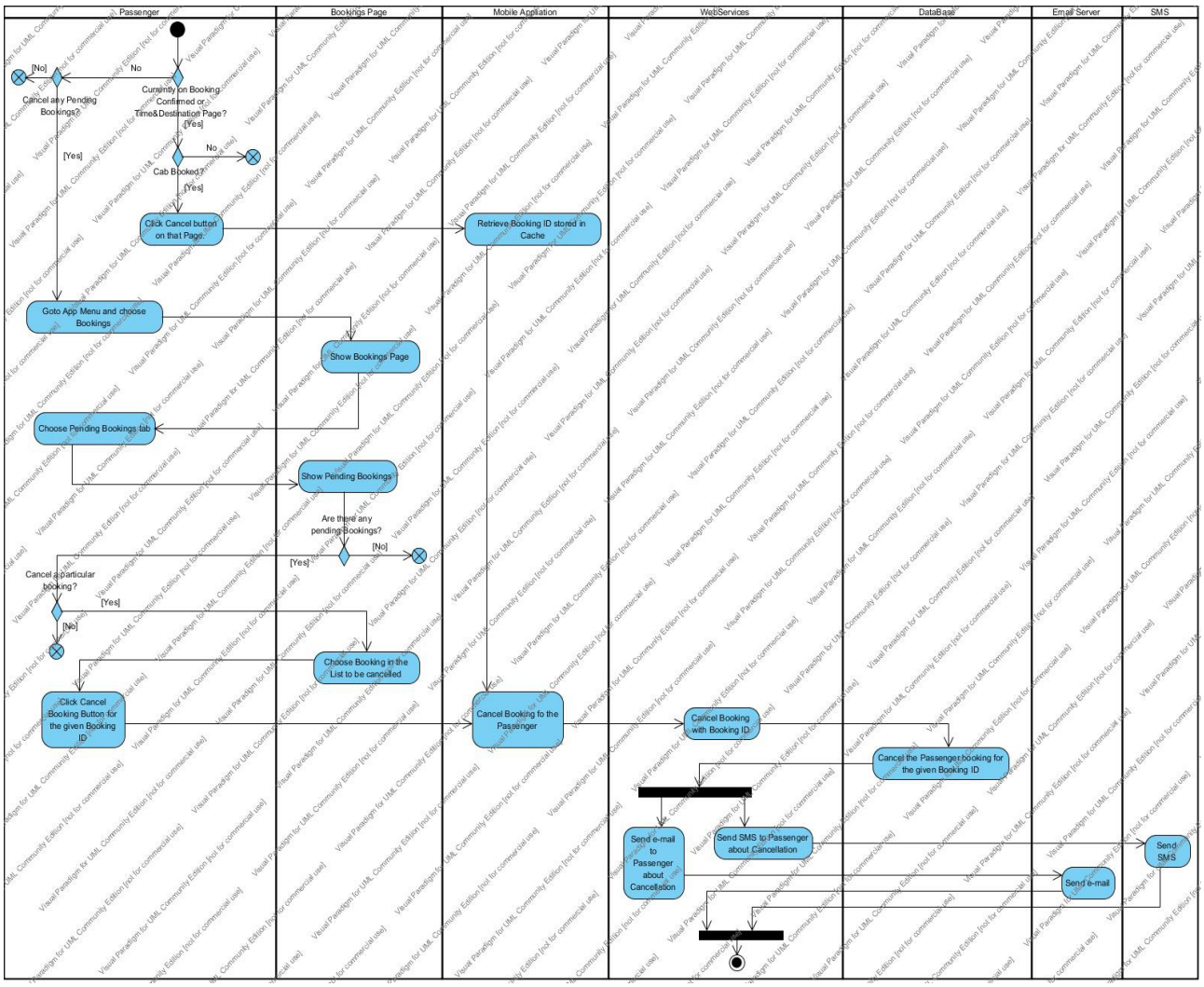
1a.1 Passenger is on the Booking Confirmed Page and clicks on Cancel button.
1a.2 resume @ step 4 of basic flow.

2a. Pasenger has booked a cab and wants to Preview his Pending Bookings and Cancel a Cab.

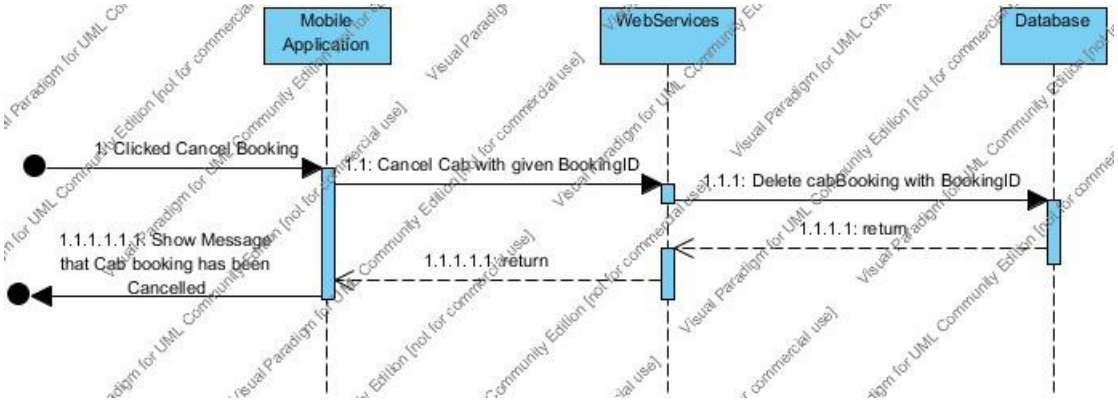
2a.1 Passenger chooses the Booking to be cancelled and clicks on the Cancel button.
2a.2 resume @ step 5 of Basic Flow.

EXCEPTIONAL FLOW:
BUSINESS RULES:

IMPLEMENTATION CONSTRAINTS AND SPECIFICATION:



Activity Diagram for Cancel Booked Cab



Sequence Diagram for Cancel Booked Cab.

2.1.20 USE CASE 20 : ADD TO FAVOURITES

ID: UC 20

DESCRIPTION: This Use case describes the Process of adding Passenger PickUp Location to favourites.It acts as a custom repository for locations that Passenger can get back when he needs it. The CabsMapViewView shows the cabs available in the vicnity of Passenger Location and Passenge location. This Location can be added to the favourites, so that it can be used at later date and time. The favourites are added to the Cache of the Mobile.

ACTORS: Passenger.

SECONDARY ACTORS: Server.

DEPENDENCIES: UC 01**, UC 02**

PRE-CONDITION: The Passenger has logged in to the application and is on CabsMapView. The Passenger is looking at PickUp locations and wants to add his current location to favourites.

POST-CONDITION(S)

SUCCESS END CONDITION:

Passenger has added the location to his favourites in the Mobile Cache.

FAILURE END CONDITION(s):

Pasenger is unable to add the location to his favourites.

FLOW OF EVENTS

BASIC FLOW:

1. Include ***CHOOSE PICKUP LOCATION.***
2. Passenger wants to add the current location to favourites.
3. Passenger chooses the Favourite Icon.
4. A modal window “Naming Location” pops on CabsMapView and asks for a name for the Location.
5. Passenger gives a name and submits.
6. Application checks the Name,Location,its GPS coordinates in the Cache.
7. Applicatiion finds that name is not present in Cache.
8. Application forwards the Location GPS to the WebServices for checking if location already exists.
9. Database retrieves all GPS data and gives the List.
10. Webservices runs adjacency check algoritm on the given Location to determine if the location is already present.
11. Webservices check succeeds and stores the Data in database.
12. Database stores the data and returns.
13. Webservice returns success to Application.
14. Application adds the Data to the local Cache.
15. The CabsMapView shows a message that it has been added successfully and Naming Location Modal Pop up closes.

ALTERNATIVE FLOW:

6a Name Check fails.

6a.1 Application check returns Name is present .

6a.2 The CabsMapView shows a Message that Name is already present.

6a.3 rename the Location in Modal “Naming Location” Popup.

6a.4 resume @ step 5 of Basic Flow.

EXCEPTIONAL FLOW:

11a Webservice check fails for Location, Location already present.

11a.1 Webservice returns Location already present to Application.

11a.2 CabsMapView shows the error Message that the Location already exists with the name and choose another Location.

11a.3 The Modal window “Naming Location” closes.

11a.3 Application doesnot add it to the local Cache.

11a.34 The Use case ends.

BUSINESS RULES:

1. Each passenger can store a maximum of 20 favourite locations irrespective of city.
2. The favourites locations are stored in the Favourite data entity and also in the local Mobile Cache, to retrieve the data on the Application.
3. Unique key indicated by the favourite ID needs to be generated.
4. GPS resolution of 0.15 kms circular radius needs to be used to differentiate one location from another.
5. Each Location needs to have a name assosciated with the location**.**

IMPLEMENTATION CONSTRAINTS AND SPECIFICATION:

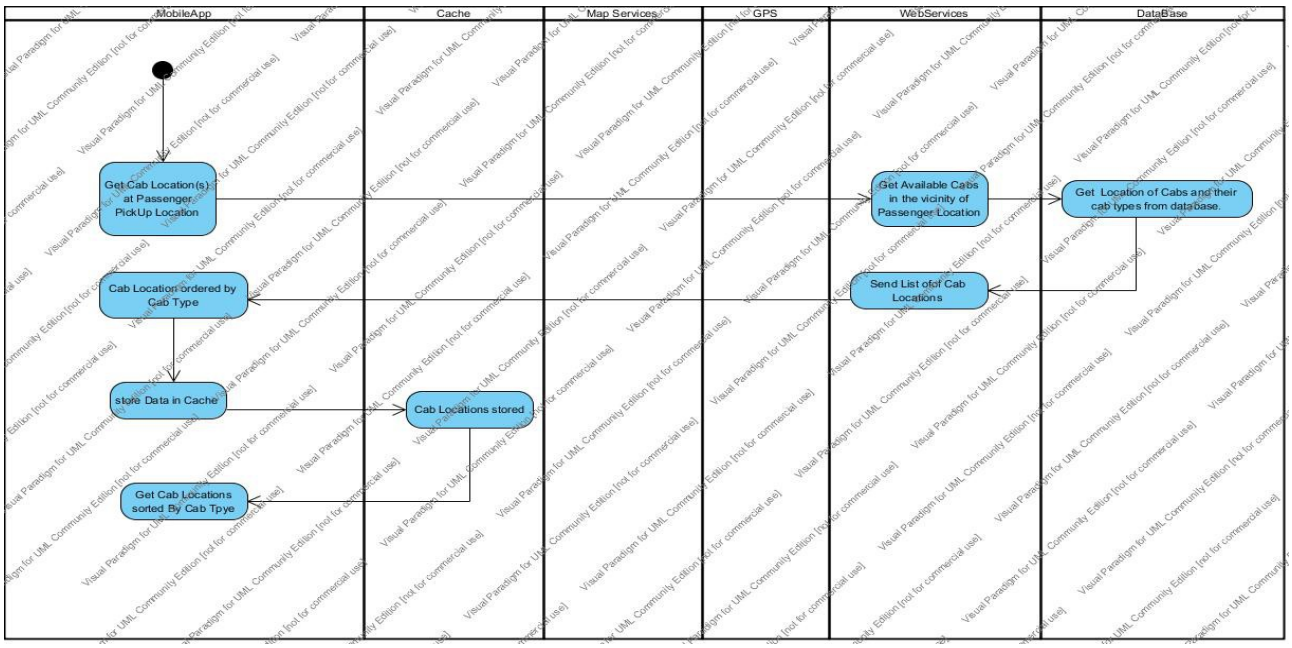
1. GPS data of PickUp Location may not be accurate and the same would be stored in favourites.

2.1.21 USE CASE 21 : GET CABS AT PASSENGER LOCATION.

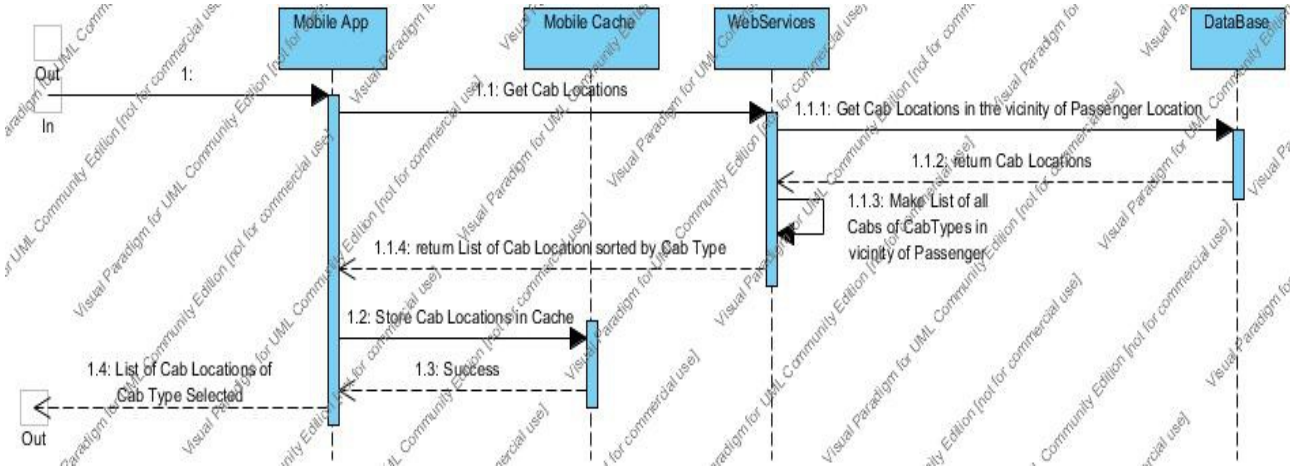
ID: UC 21
DESCRIPTION: This use case describes the scenario of getting new Cabs position when the PickUp Location of the Passenger changes. This scenario is used in ***BOOK CAB NOW*** and ***BOOK CAB LATER*** use cases.
ACTORS: Passenger
SECONDARY ACTORS: Server
DEPENDENCIES: UC 01**, UC 02**
PRE-CONDITION: The Passenger is on the CabsMapView Page and wants to change his current PickUp Location and get a list of cabs in the vicinity of the Passenger Location. The view has been updated and the CabsMapView needs to get the current cab postions.The Application keeps polling for cab positions every 30 seconds, once the CabsMapView page is current.
POST-CONDITION(S)
SUCCESS END CONDITION:
The updated cab position in the vicinIty of the Passenger locations are shown.
FAILURE END CONDITION(s):
The CabMapView is not able to get the updated Positions.
FLOW OF EVENTS
BASIC FLOW:

1. The Application requests Webservices to get cab positions at a given Passenger Location.
2. Webservices forwards the request to the Database.
3. The Database retrieves the Cab location and retuns to the Webservice.
4. The Webservice forward the List sorted by Cabtype to the Mobile Application.
5. Application stores cab information in Mobile Cache.
6. The CabsMapView retrieves the List and renders the Cab Locations on it.

ALTERNATIVE FLOW:
EXCEPTIONAL FLOW:
BUSINESS RULES:
IMPLEMENTATION CONSTRAINTS AND SPECIFICATION:



Activity Diagram for Get Cabs at Passenger Location



Sequence Diagram for Get Cabs at Passenger Location.

2.1.22 USE CASE 21 : VIEW BOOKINGS

ID: UC 21
DESCRIPTION: This use case describes the process of viewing Cab bookings. This usecase is the prerequiste for View Pending Booking and View Past Bookings.
ACTORS: Passenger
SECONDARY ACTORS:System.
DEPENDENCIES: UC 01**, UC 02**
PRE-CONDITION: The Pasenger has logged in to the application and wants to view his history of Bookings.
POST-CONDITION(S)
SUCCESS END CONDITION:
The Passenger is able to view the Bookings segregated as Pending and Previous Bookings.
FAILURE END CONDITION(s):
FLOW OF EVENTS
BASIC FLOW:

1. The Passenger opts for “View Booking” on the Main Menu.
2. The Application requests Webservices for Passengers Bookings List from the database.
3. The Webservice forwards the request to the database.
4. The database retrieves all the Cab bookings and returns the list to WebServices.
5. Webservices sorts the Booking List as Previous and Pending Bookings and sends it Application.
6. The Booking Page puts the Previous and Pending Bookings in separate tabs and Lists them in descending order of their Booking date.
7. By default the Pending Booking tab with List of Bookings in this category for the Passenger is Shown.

ALTERNATIVE FLOW:
EXCEPTIONAL FLOW:
BUSINESS RULES:
IMPLEMENTATION CONSTRAINTS AND SPECIFICATION:

2.1.23 USECASE 22: VIEW PENDING BOOKINGS

ID: UC 23
DESCRIPTION: This use case describes the scenario to view Pending Bookings.
ACTORS: Passenger
SECONDARY ACTORS:
DEPENDENCIES: UC 01
PRE-CONDITION: The Passenger has logged in to the Application and has opted to view Pending Bookings in the Application.
POST-CONDITION(S)
SUCCESS END CONDITION:
The Passenge is able to view Pending bookings in descending order of date and time. The top one showing the Latest Booking.
FAILURE END CONDITION(s):
FLOW OF EVENTS
BASIC FLOW:

1. Include ***View Bookings.***
2. The Pasenger has opted to View Pending Bookings and chooses that tab in

Bookings Page.

3. The Passenger is shown Pending Bookings List in descending order of date and time with each Booking having information about :
    a) Booking ID,
    b) Date and Time,
    c) From Location,
    d) Destination Location and
    e) a button to Cancel the Booking.
4. Passenger can choose and cancel any of the bookings.

ALTERNATIVE FLOW:

4a Passenger has opted to Cancel Booking in the Pending Bookings.

4a.1 Passenge clicks on the cancel option for the chosen Booking ID.

4a.2 Extend "***CANCEL BOOKED CAB***"[Extension Point: Cancel Booked Cab]

4a.3 resume @ 3

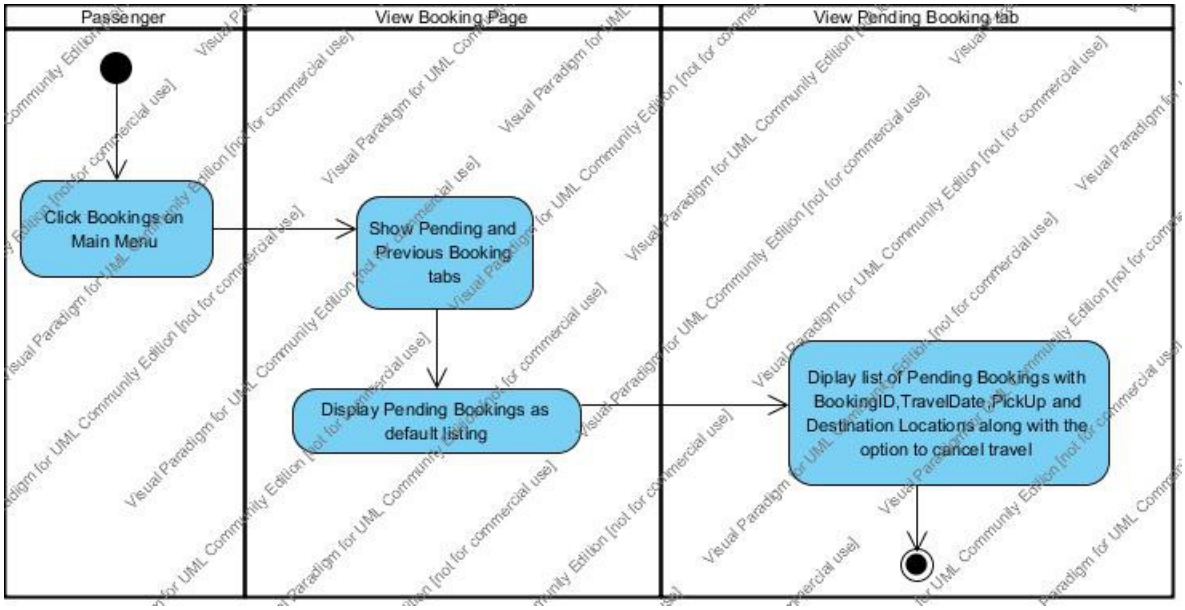
3a There are no Pending Bookings for the Passenger.

3a.1 Passenger is shown "No Pending Cabs for Passenger" on the Pending Bookings Canvas Area.

EXCEPTIONAL FLOW:

BUSINESS RULES:

IMPLEMENTATION CONSTRAINTS AND SPECIFICATION:



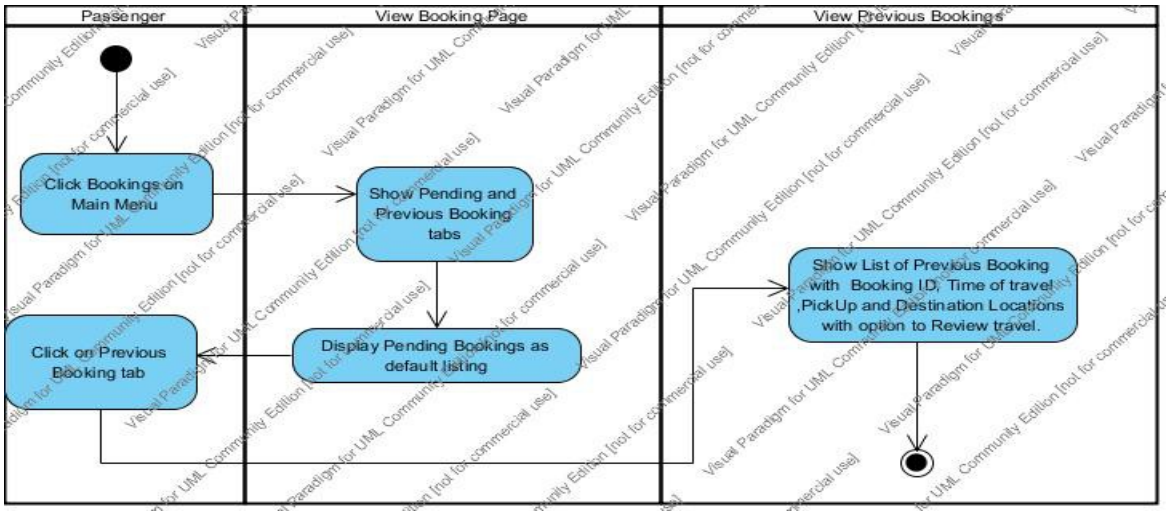
Activity diagram for Display Pending Bookings.

2.1.24 USE CASE 24 : VIEW PREVIOUS BOOKINGS

ID: UC 24
DESCRIPTION: This use case describes the scenario of viewing cabs that have been previously booked,travelled and paid for by the Passenger.
ACTORS: Passenger
SECONDARY ACTORS:
DEPENDENCIES: UC 01**, UC 02**
PRE-CONDITION: The Passenger has Logged in to the Application and wants to view his Previous bookings of travel.
POST-CONDITION(S)
SUCCESS END CONDITION:
The Passenger is able to view his Previous Bookings, sorted in descending order of time and date, with the most recent travel at the top of the List.
FAILURE END CONDITION(s):
FLOW OF EVENTS
BASIC FLOW:

1. Include ***VIEW BOOKINGS.***
2. The Passenger has opted to View Previous Bookings and chooses that tab in Bookings Page.
3. The Passenger is shown Previous Bookings Listing in descending order of date and time with each Booking having information about :
   a) Booking ID,
   b) Date and Time,
   c) From Location,
   d) Destination Location and
   e) a button to send feedback.
4. Passenger can chose to give a feedback if they have already not given the same for the Cab and its Driver.

ALTERNATIVE FLOW:
4a The Passenger chooses to send feedback about the travel.
4a.1The Passenger choose to write feedback.
4a.2 Extend ***SEND FEEDBACK*** [Extension Point: FeedBack]
EXCEPTIONAL FLOW:
BUSINESS RULES:
IMPLEMENTATION CONSTRAINTS AND SPECIFICATION:



Activity Diagram for Display Previous Booking.

3.3 DESIGN CONSTRAINTS

This section includes the design constraints on the software caused by the hardware.

3.3.1 Hard drive space

The application's need of hard drive space.
MUST: No more than 20 MB.
PLAN: No more than 15 MB.
WISH: No more than 10 MB.

3.3.2 Application memory usage

The amount of Operate System memory occupied by the application.
MUST: No more than 20 MB.
PLAN: No more than 16 MB
WISH: No more than 10 MB
Operating System:
DEFINED: The mobile Operate System which the application is running on.