DETAILED DESIGN

Project Code:	ATA-01
Project Name:	Automation of Travel Agency

Affected Groups

Development Engineering
Quality Assurance
XYZ Corp

List of Reference Documents

Name	**Version No.**
1.RS_ATA	2.20
2.FS_ATA	2.20
3.	
4.	

Prepared by/Date	**Reviewed by/Date**	**Approved by/Date**

Table of Contents

1. Introduction

1.1 Background

XYZ Travels Ltd. provides vehicle booking facilities to users (Customers) across many cities.

1.2 Purpose

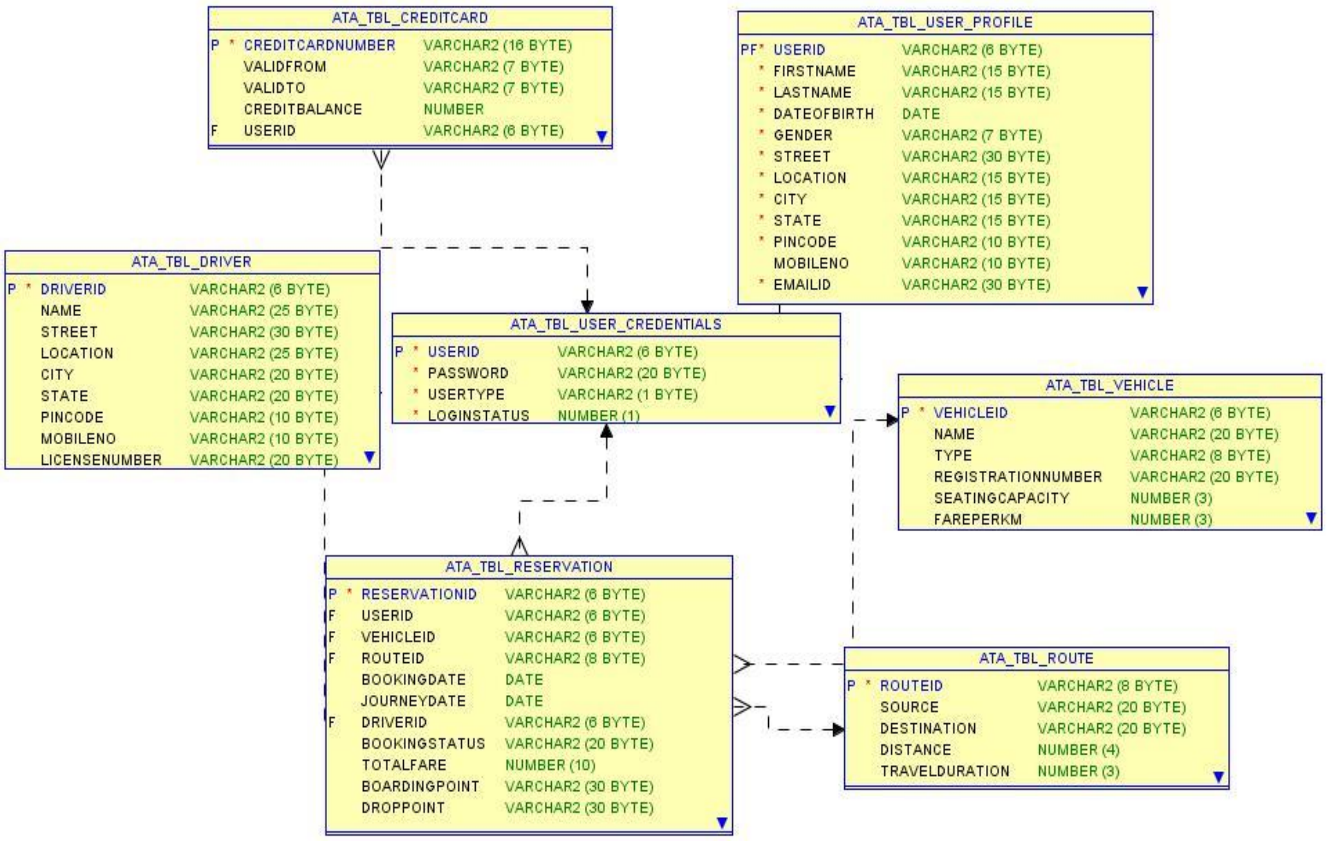
XYZ Travels Ltd. plans to develop "Automation of Travel Agency" - standalone/web application [Core Java Batches - Swing Application; J2EE Batches - Web Application], where users (Customers) can reserve vehicles and manage their reservations.

1.3 Scope

The scope of the Automation of Travel Agency (ATA) will be to provide the functionality as described below. The system will be developed on a Windows operating system using Java/J2EE.

2. Global Data Structures and Shared Data Functions

This section describes the structure of 9 tables to be used for the implementation of requirements as stated in the specification.



ATA_TBL_CREDITCARD

P	*	CREDITCARDNUMBER	VARCHAR2 (16 BYTE)
		VALIDFROM	VARCHAR2 (7 BYTE)
		VALIDTO	VARCHAR2 (7 BYTE)
		CREDITBALANCE	NUMBER
F		USERID	VARCHAR2 (6 BYTE)

ATA_TBL_USER_PROFILE

PF	*	USERID	VARCHAR2 (6 BYTE)
	*	FIRSTNAME	VARCHAR2 (15 BYTE)
	*	LASTNAME	VARCHAR2 (15 BYTE)
	*	DATEOFBIRTH	DATE
	*	GENDER	VARCHAR2 (7 BYTE)
	*	STREET	VARCHAR2 (30 BYTE)
	*	LOCATION	VARCHAR2 (15 BYTE)
	*	CITY	VARCHAR2 (15 BYTE)
	*	STATE	VARCHAR2 (15 BYTE)
	*	PINCODE	VARCHAR2 (10 BYTE)
		MOBILENO	VARCHAR2 (10 BYTE)
	*	EMAILID	VARCHAR2 (30 BYTE)

ATA_TBL_DRIVER

P	*	DRIVERID	VARCHAR2 (6 BYTE)
		NAME	VARCHAR2 (25 BYTE)
		STREET	VARCHAR2 (30 BYTE)
		LOCATION	VARCHAR2 (25 BYTE)
		CITY	VARCHAR2 (20 BYTE)
		STATE	VARCHAR2 (20 BYTE)
		PINCODE	VARCHAR2 (10 BYTE)
		MOBILENO	VARCHAR2 (10 BYTE)
		LICENSENUMBER	VARCHAR2 (20 BYTE)

ATA_TBL_USER_CREDENTIALS

P	*	USERID	VARCHAR2 (6 BYTE)
	*	PASSWORD	VARCHAR2 (20 BYTE)
	*	USERTYPE	VARCHAR2 (1 BYTE)
	*	LOGINSTATUS	NUMBER (1)

ATA_TBL_VEHICLE

P	*	VEHICLEID	VARCHAR2 (6 BYTE)
		NAME	VARCHAR2 (20 BYTE)
		TYPE	VARCHAR2 (8 BYTE)
		REGISTRATIONNUMBER	VARCHAR2 (20 BYTE)
		SEATINGCAPACITY	NUMBER (3)
		FAREPERKM	NUMBER (3)

ATA_TBL_RESERVATION

P	*	RESERVATIONID	VARCHAR2 (6 BYTE)
F		USERID	VARCHAR2 (6 BYTE)
F		VEHICLEID	VARCHAR2 (6 BYTE)
F		ROUTEID	VARCHAR2 (8 BYTE)
		BOOKINGDATE	DATE
		JOURNEYDATE	DATE
F		DRIVERID	VARCHAR2 (6 BYTE)
		BOOKINGSTATUS	VARCHAR2 (20 BYTE)
		TOTALFARE	NUMBER (10)
		BOARDINGPOINT	VARCHAR2 (30 BYTE)
		DROPPOINT	VARCHAR2 (30 BYTE)

ATA_TBL_ROUTE

P	*	ROUTEID	VARCHAR2 (8 BYTE)
		SOURCE	VARCHAR2 (20 BYTE)
		DESTINATION	VARCHAR2 (20 BYTE)
		DISTANCE	NUMBER (4)
		TRAVELDURATION	NUMBER (3)

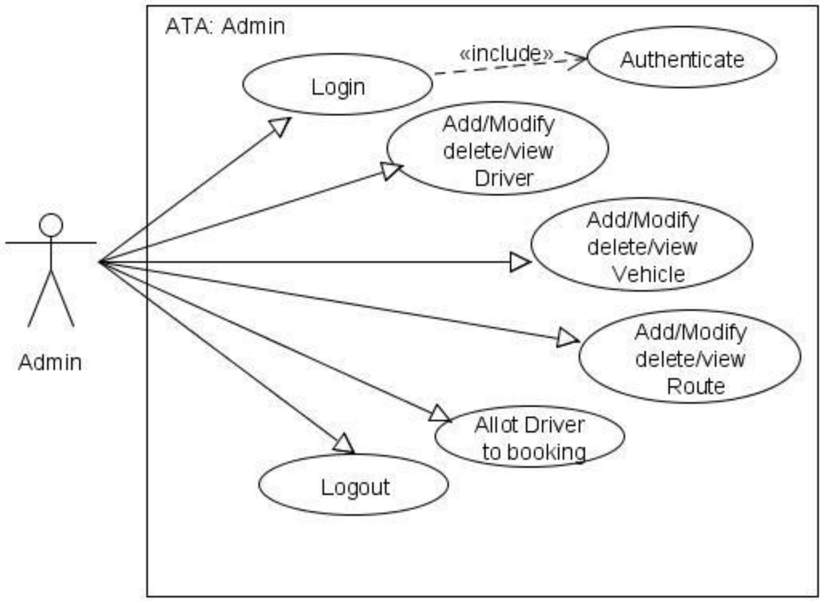
3. High Level Design

This section describes the high level design diagrams. Use case diagram with Use Case definition, Sequence Diagram and Class Diagram which provides a visual representation of the requirements, logical flow and their class representations.

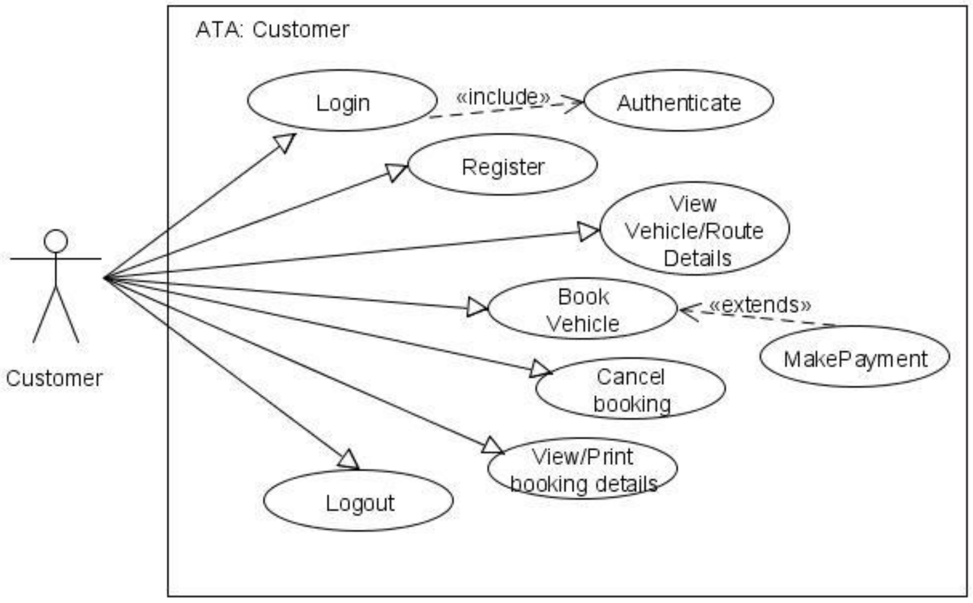
3.1 Use Case Diagrams

The requirements of a system can be represented using a use case model in the Use Case Diagram. The use case diagram for the actors of this case study is given as below.

3.1.1 Use Case Diagram for Admin


ATA: Admin
Login
«include»
Authenticate
Add/Modify delete/view Driver
Add/Modify delete/view Vehicle
Add/Modify delete/view Route
Allot Driver to booking
Logout
Admin

3.1.2 Use Case Diagram for Customer



3.2 Use case Definition

Generally, in a design document, Use case definitions should be written for all the *Requirements* of the system.

Note: Participants are expected to document use case definitions for all requirements. However, for few requirements documented below for reference.

Below table explains 'Use Case' definition for requirement "AA-001" - Login operation for all users.

3.2.1 Login

USE CASE #	AA-001 *Login*	
Goal	*All users logging into the system should be authenticated using a unique login-id and password (operations to be supported based on type of user)*	
Preconditions	If the user type is 'Admin', credential details should exist. If the user type is 'Customer', he/she should be registered by the administrator.	
Success End Condition	If the user type is 'Admin', then redirect to the Admin page. If the user type is 'Customer', then redirect to the Customer page.	
Failed End Condition	*The end user is redirected to an Error Page, and/or is asked to re-enter login credentials.*	
Primary, Secondary Actors	Admin, Customer.	
Trigger	*Login button*	
DESCRIPTION	**Step**	**Action**
	1	*Enter Login credentials (id & password)*
	2	*Click on Login button*
	3	*If id & password is Success, then identify user type Display appropriate(Admin/Customer) home page*
	Step	**Branching Action**
	1	*If 'id' is not existing then return with requesting for registration*
	2	*If password is not matching return with suitable error message say 'Re-enter id & password'*
Related Information/Use cases	*Not Applicable*	
Priority	*P1*	
Performance	*5 seconds*	
Frequency	*10 / hour*	
Assumptions	Admin/Customer login credentials are available in the database and others are already registered with their credentials	

Below table explains 'Use Case' definition for requirement "AD-001" – ADD Vehicle Details operation for Admin user only.

3.2.2 ADD Vehicle Details

USE CASE #	AD-001 *Add Vehicle details*	
Goal	*To enable Administrator to create and add new Vehicle*	
Preconditions	Administrator must be logged in to be able to create a new Vehicle.	
Success End Condition	*"Redirect to Admin home page"*	
Failed End Condition	*"Redirect to Error Page"*	
Primary, Secondary Actors	*Administrator*	
Trigger	*'Add Vehicle button*	
DESCRIPTION	**Step**	**Action**
	1	*Provide appropriate vehicle details*
	2	*Click on Add Vehicle button*
	Step	**Branching Action**
	1	*If failed to add vehicle details*
	2	*Display appropriate message to the admin*
Related Information/Use cases	*Not applicable*	
Priority	*P1*	
Performance	*Approx. 4 sec*	
Frequency	*2 / Month*	
Assumptions	Admin login credentials are available in the database	

3.3 Class Diagram

The class diagram is a very basic concept in object-oriented world. Class diagrams demonstrate a model, describing what attributes and behavior it has rather than describing the methods for accomplishing operations. Class diagrams are very useful in representing relationships between classes and interfaces.

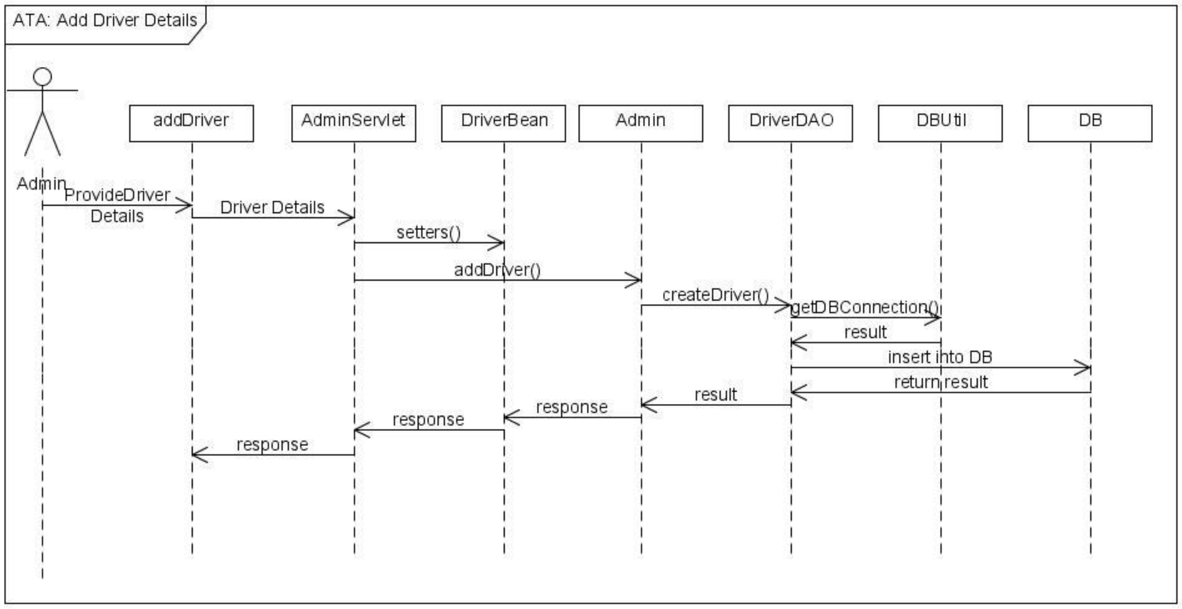
<class diagram to be drawn by RLL participants>

3.4 Sequence Diagram

A graphical representation of a module's function invoking functions of other modules in order to achieve a task (specific user requirement) is called a sequence diagram. A sequence diagram for the authentication process is given below for reference. The below example is for a Web Application using servlets/jsp.

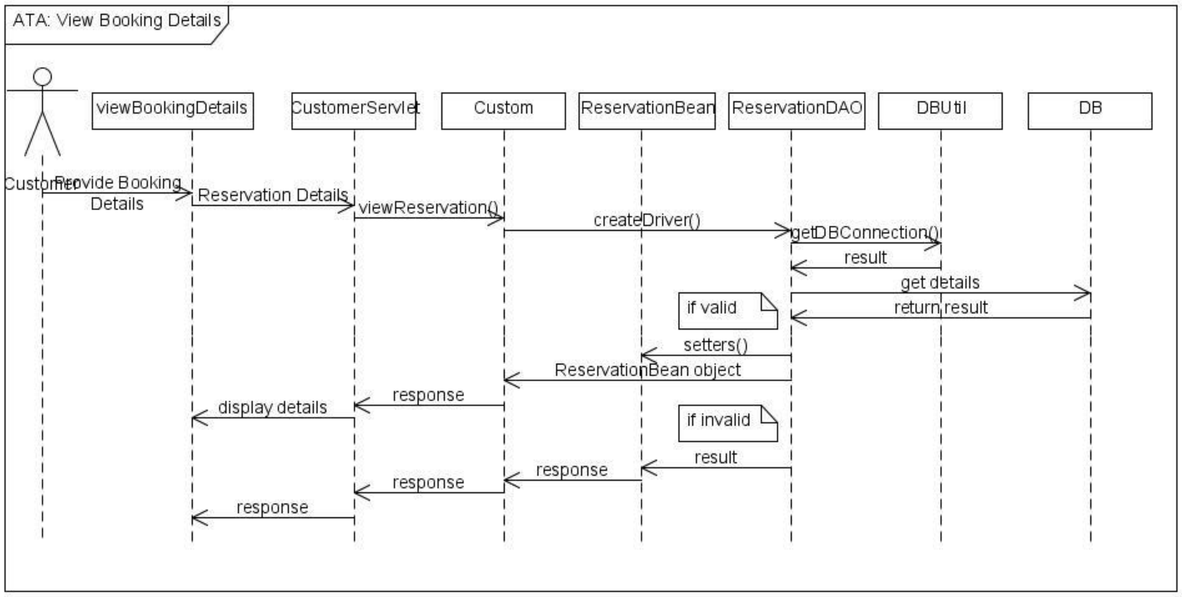
3.4.1 Add Driver details:

Admin performing add Driver details.



3.4.2 Add Driver details:

Customer performing view booking**.**



3.5 Packages / Classes / Interface

This section provides a brief outlook on the packaging hierarchy along with the respective classes to be used for the implementation.

The 4 packages mentioned below are for both GUI and Web Application.

Packages	
Package	**Description**
com.ata.service	This package contains all the Service classes
com.ata.bean	This package contains all the bean classes
com. ata.dao	This package contains all the DAO functionality classes
com. ata.util	This package contains all the generic functionality classes

This package is used only for a GUI application.

com. ata.ui	This package contains all the UI related classes [For Core Java]

The package for the controller class should be used as below based on the type of application

com.ata.listener	**listener -** core java	
or		
com.ata.servlet	**servlet -** Web Applications	
or		
com.wirpo.ata.action	**action -** Struts	
or		
com.ata.controller	**controller -** Spring	

Package com. ata.bean

Class Name	Attributes	Data Type
ProfileBean	userID	String
	firstName	String
	lastName	String
	dateOfBirth	Date
	gender	String
	street	String
	location	String
	city	String
	state	String
	pincode	String
	mobileNo	String
	emailID	String
	password	String
CredentialsBean	userID	String
	password	String
	userType	String
	loginStatus	int
DriverBean	driverID	String
	name	String
	street	String
	location	String
	city	String
	state	String
	pincode	String
	mobileNo	String
	licenseNumber	String
ReservationBean	reservationID	String
	userID	String
	routeID	String
	bookingDate	Date
	journeyDate	Date
	vehicleID	String
	driverID	String

	bookingStatus	String
	totalFare	double
	boardingPoint	String
	dropPoint	String

RouteBean	routeID	String
	source	String
	destination	String
	distance	int
	travelDuration	int

VehicleBean	vehicleID	String
	name	String
	type	String
	registrationNumber	String
	seatingCapacity	int
	farePerKM	double

Package com.ata.service

Interface Summary	
Interface	**Description**
Administrator	Entity interface for Administrator dealing with the admin process functionalities **Method Summary** String **addVehicle**(VehicleBean vehicleBean) int **deleteVehicle**(ArrayList<String> vehicleID) VehicleBean **viewVehicle**(String vehicleID) boolean **modifyVehicle**(VehicleBean vehicleBean) String **addDriver**(DriverBean driverBean) int **deleteDriver**(ArrayList<String> driverID) boolean **allotDriver**(String reservationID, String driverID) boolean **modifyDriver**(DriverBean driverBean) String **addRoute**(RouteBean routeBean) int **deleteRoute**(ArrayList<String> routeID) RouteBean **viewRoute**(String routeID)

	boolean **modifyRoute**(RouteBean routeBean)
	ArrayList<ReservationBean> **viewBookingDetails**(Date journeyDate,String source, String destination)
Customer	Entity interface of Customer for dealing with the Customer process functionalities
	Method Summary
	ArrayList<VehicleBean> **viewVehiclesByType**(String vehicleType)
	ArrayList<VehicleBean> **viewVehicleBySeats**(int noOfSeats)
	ArrayList<RouteBean> **viewAllRoutes**()
	String **bookVehicle**(ReservationBean reservationBean)
	boolean **cancelBooking**(String userID, String reservationID)
	ReservationBean **viewBookingDetails**(String reservationID)
	ReservationBean **printBookingDetails**(String reservationID)

Package com.ata.dao

Find below the suggestive approach for CRUD operations [method naming & signature] for the DAO Interface/classes. Create the necessary DAO classes.

Interface Name	Description
xyzDAO	DAO interface/class to deal with operations related to the specific table.
	Method Summary
	String createXYZ(BeanObject)
	int deleteXYZ(ArrayList<String>)
	boolean updateXYZ(BeanObject)
	BeanObject findByID(String)
	ArrayList<BeanObject> findAll()

- If required, additional find methods can be created.

Package com.ata.util

Interface Summary	
Interface	**Description**
Authentication	This interface is responsible for performing the Authentication and Authorization process.
	Methods
	boolean **authenticate**(**CredentialsBean** credentialsBean)
	String **authorize**(String userID)

	boolean **changeLoginStatus**(**CredentialsBean** credentialsBean, int loginStatus)
DBUtil	This class is responsible for the Database connection establishment.
	Methods static Connection **getDBConnection**(String driverType)
User	Interface for handling different types of users **Method Summary** String **login**(**CredentialsBean** credentialsBean) Return value must be either: "A", "C", "FAIL", “INVALID” A->Admin, C->Customer **Wrong username/password should return INVALID**. boolean **logout**(String userId) String **changePassword**(**CredentialsBean** credentialsBean, String newPassword) Return value must be either: "SUCCESS", "FAIL", “INVALID” String **register**(ProfileBean profileBean) Return value must be either: <userId of lenght 6>, "FAIL" Note: userId-> first 2 letter of first name followed by 4 digit auto generated number
Payment	Class for handling payment related information String creditCardNumber, validFrom, validTo, double balance **Methods** boolean **findByCardNumber**(**String** userID**, String** cardNumber) String **process**(Payment payment)

Note: **Wherever empty or NULL is the response in all such cases suitable message has to be displayed for user**

3.6 UI Templates

3.6.1 UI Principle

The UI [Presentation Layer] should be designed with the below mentioned principles which helps easy interaction by the user to the application.

3.6.2 UI controls and Usage Principle

UI Type	Controls	Description
Direct Entry	Text Box, Text Area	Any input that cannot be predicted and needs the user to key in. e.g Name, Address, contact no etc.
Static Selection	Option Button, Check Box, Drop Down	Should be used where the input can be predefined. e.g gender, month [Jan – Dec] etc. If number of items is more, drop down is preferred.
Dynamic Selection	Drop Down	The items for the drop down should be retrieved from a stored data. e.g Displaying Districts in a drop down from places table.

Automation	Label Text Field [Read Only]	Data’s that are calculative or an output of a function. e.g : Displaying system date, showing total amount etc.
Decision Control	Button	Operations like submit, save, clear should be executed only upon clicking respective buttons.

3.6.3 UI Template

This section contains the design template for the website home page [Fig. 1] that will be displayed at the time of opening this web application and Actor specific home page [Fig. 2].

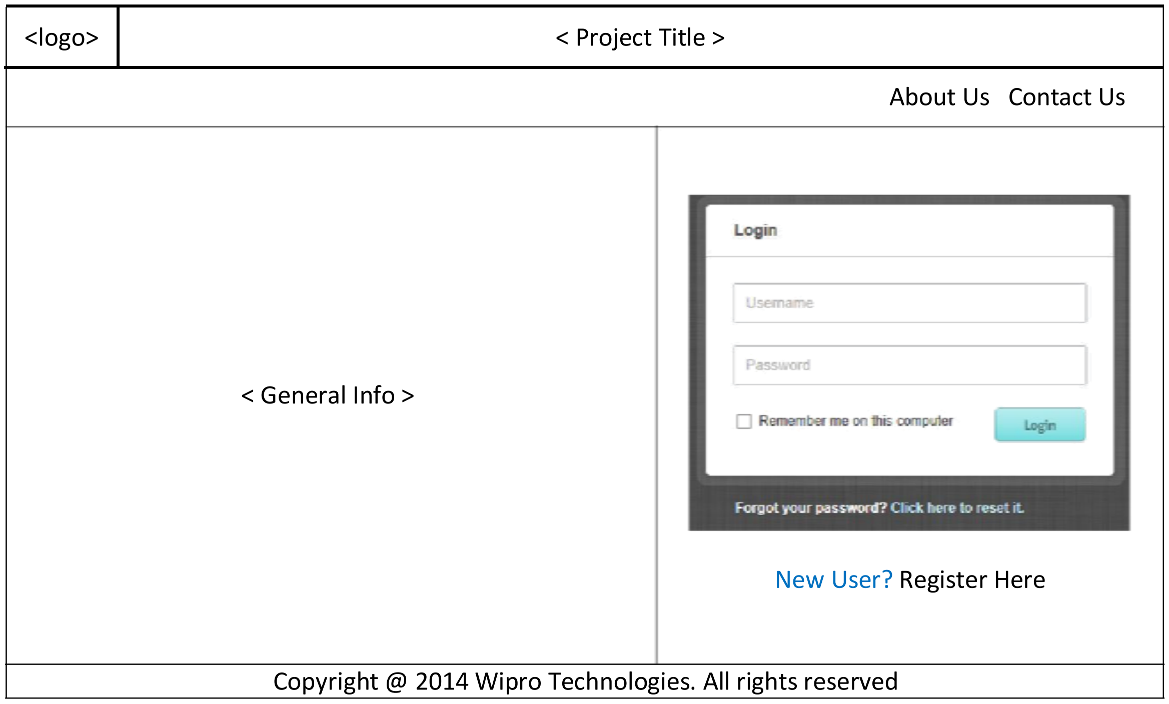

<logo>
< Project Title >
About Us
Contact Us
< General Info >
Login
Username
Password
Remember me on this computer
Login
Forgot your password? Click here to reset it.
New User? Register Here
Copyright @ 2014 Wipro Technologies. All rights reserved

Fig. 1 - Main Page [First Page to open]

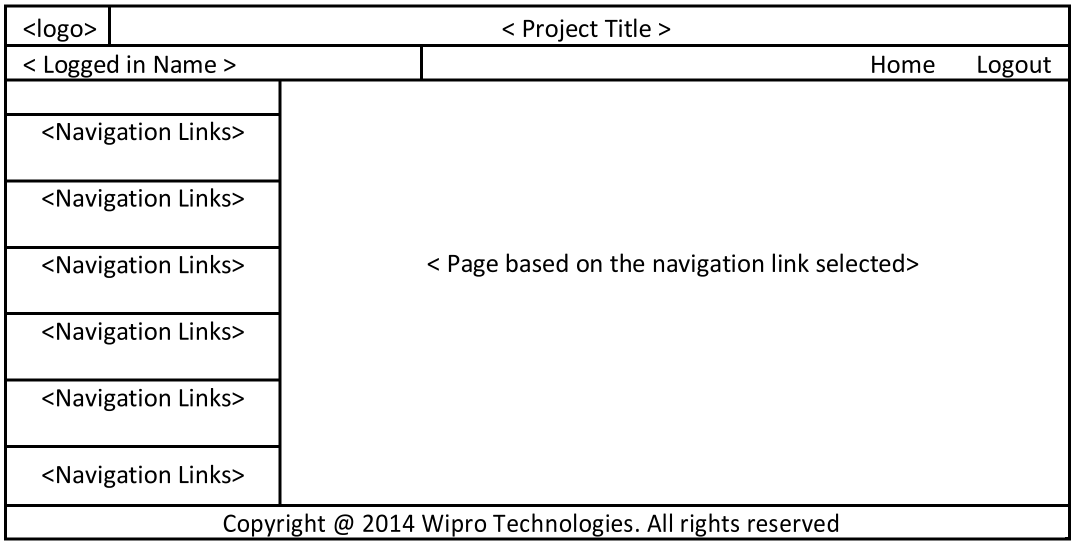


Fig. 2 - Home Page for Actor

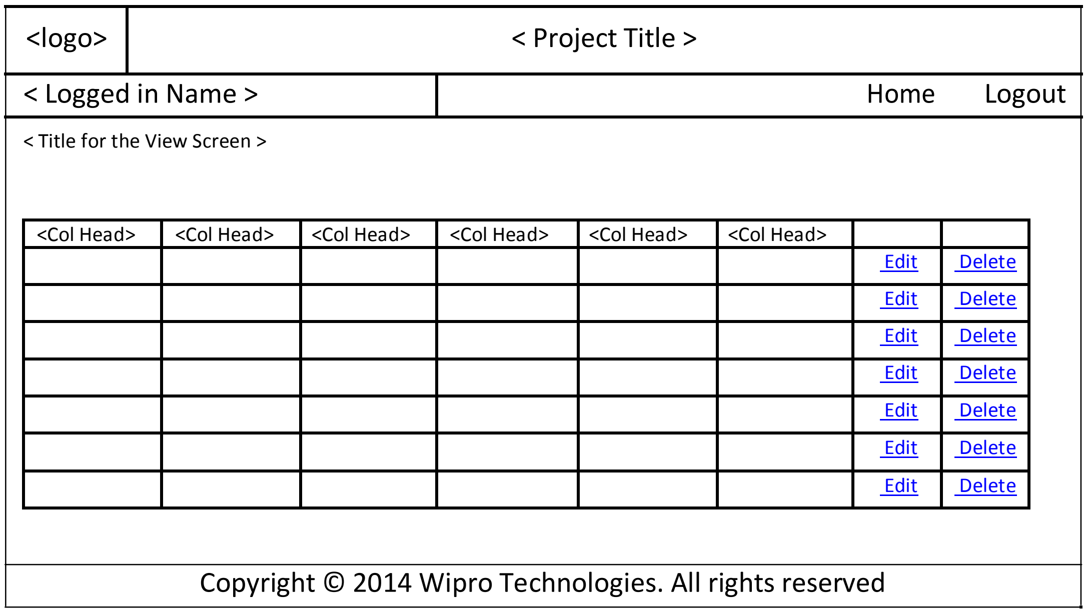


Fig. 3 – View Screen with Edit and Delete Functionality

4. Critical Functions and Focus for Testing

login(), addVehicle(), reservation(), makePayment(), cancelTicket().

5. Limitations

- Booking vehicle is on daily basis (no hourly booking is permitted).
- Driver allocation will be done only by the Admin but not Customer.
- Return journey booking is not available

6. APPENDIX

1. Table: ATA_TBL_User_Credentials

This table contains Authentication Information for Administrator & Customer

Field Name	Data Type	Description
Userid	VARCHAR2(6)	Primary Key*
Password	VARCHAR2(20)	Not Null
Usertype	VARCHAR2(1)	Either ['A','C']
Loginstatus	NUMBER(1)	Either [1 ,0]

* First 2 letters of First Name followed by 4 digits auto generated number

2. Table: ATA_TBL_User_Profile

This table contains User specific details entered during User Registration.

Field Name	Data Type	Description
UserId*	VARCHAR2(6)	Foreign Key
Firstname	VARCHAR2(15)	Not Null
Lastname	VARCHAR2(15)	Not Null
Dateofbirth	DATE	Not Null
Gender	VARCHAR2(7)	Not Null
Street	VARCHAR2(30)	Not Null
Location	VARCHAR2(15)	Not Null
City	VARCHAR2(15)	Not Null
State	VARCHAR2(15)	Not Null
Pincode	VARCHAR2(6)	Not Null
MobileNo	VARCHAR(10)	Exact 10 digit only
EmailId	VARCHAR2(30)	

* First 2 letters of First Name followed by 4 digits auto generated number

3. Table: ATA_TBL_Vehicle

This table contains Vehicle specific information.

Field Name	Data Type	Description
vehicleId*	VARCHAR2(6)	Primary Key
Name	VARCHAR2(20)	Not Null
Type	VARCHAR2(8)	Not Null
RegistrationNumber	VARCHAR2(12)	Not Null
SeatingCapacity	NUMBER(3)	Not Null
FarePerKM	NUMBER(3)	

* Id should be First 2 letters of VehicleName followed by 4 digits auto generated number

4. Table: ATA_TBL_Driver

This table contains Driver details like Name, address & License number etc.

Field Name	Data Type	Description
driverId*	VARCHAR2(6)	Primary Key
Name	VARCHAR2(25)	Not Null, Either[A|B]**
Street	VARCHAR2(30)	Not Null
Location	VARCHAR2(15)	Not Null
City	VARCHAR2(15)	Not Null
State	VARCHAR2(15)	Not Null
Pincode	VARCHAR2(6)	Not Null
MobileNo	VARCHAR(10)	Exact 10 digit only
LicenseNumber	VARCHAR2(20)	Unique

* Id should be First 2 letters of First Name followed by 4 digits auto generated number

5. Table: ATA_TBL_Route

This table contains details about route, distance, duration etc.

Field Name	Data Type	Description
routeId*	VARCHAR2 (8)	Primary Key
Source	VARCHAR2(20)	Not Null
Destination	VARCHAR2 (20)	Not Null
Distance	NUMBER(4)	Not Null
TravelDuration	NUMBER(3)	Not Null

*** Id should be First 2 letters of Source and 2 letters of destination followed by 4 digits auto generated number**

6. Table: ATA_TBL_Reservation

This table contains details about vehicle reservation made by customer.

Field Name	Data Type	Description
reservationId	VARCHAR2 (6)	Primary Key

UserId	VARCHAR2 (6)	Foreign Key
VehicleId	VARCHAR2 (6)	Foreign Key
RouteId	VARCHAR2 (8)	Foreign Key
BookingDate	Date	Not Null
JourneyDate	Date	Not Null
DriverId	VARCHAR2 (6)	Foreign Key
BookingStatus	VARCHAR2 (20)	Not Null
TotalFare	NUMBER(10)	Not Null
BoardingPoint	VARCHAR2(30)	Not Null
DropPoint	VARCHAR2(30)	Not Null

7. Table: ATA_TBL_CreditCard

This table contains CreditCard details of the Customer for booking vehicle.

Field Name	Data Type	Description
CreditCardNumber	VARCHAR(16)	Primary Key
ValidFrom	VARCHAR(7)	Not Null
ValidTo	VARCHAR(7)	Not Null
CreditBalance	NUMBER	Not Null
UserId	VARCHAR(6)	Foreign Key

Database Sequences

Sequence Name	Purpose	Start With
ata_seq_userId	User ID	1000
ata_seq_routeId	Route ID	1000
ata_seq_driverId	Driver ID	1000
ata_seq_vehicleId	Vehicle ID	1000
ata_seq_reservationId	Reservation ID	1000