1. **Introduction**

The project "Online Vehicle Reservation" is an online portal which is specially designed & developed to resolve the current market problem of renting a vehicle directly through a web portal. The vehicle owners will be provided the full privileges of managing their vehicle information from data entry to booking management. And, the vehicle renters to upload & manage required legal documents and also view their booking history. The system has been given a smart search engine for searching the vehicle according to journey locations & date required. Once the vehicle has been booked for the certain time period the vehicle is automatically not bookable for that time period but available according to the time period defined by the vehicle owner.

1. **Definition of Problem**

* *Vehicle Owners problem*,
  + People buy vehicles but do not use regularly.
* *Users problem,*
  + Middle class peoples want to buy vehicle but costly.
  + Want to self drive, as most as possible.
  + Tourists like to drive, while visiting but do not get it in affordable price.

1. **Scope And Objectives**

* Targeted for vehicle owners who intend to rent their vehicle for some extra income.
* People wanting to pay for a vehicle for self driving throughout the journey.
* To change the trend of driver/passenger vehicle reservation.
* To digitalize all the paper work required.
* To make easy to rent a vehicle online.

1. **Theoretical Background**

For OVR, it has been assumed that vehicle owners would be able to use the online portal along with updating their vehicle details with pictures to upload. The vehicle information entity has been designed assuming common specification titles provided in the market. The availability of the vehicle has been managed using the date & location of the vehicle recorded into the database. The search engine is accordingly designed too. Three different users are managed according to the system requirement. The entities in database are strongly related so that the data is consistent over the system. Once the vehicle is booked, it is reserved for the booked timeframe to the specific user only.

1. **Feasibility Study**

Since it is a web application, any device or system that has the support for the recent web browsers can render the pages of OVR. The relying Operating System can be Windows, UNIX or Mac. It doesn’t need the special types of the software or the hardware. That’s why it isn’t cost expensive and remains affordable. Though the system it doesn’t need special program or hardware, but it requires a web browser and a basic computer literate to operate. The web browser can be Internet Explorer/Mozilla/Google Chrome and so on. A person having basic knowledge of computer to operate can easily use OVR.

1. **System Planning**

OVR is developed under the process of basic SDLC (System Development Life Cycle). Following table data shows the brief information of the total time division for the project completion.

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Work Description** | **Time Period (Days)** |
| 1. | Field Study and Feasibility | 10 |
| 2. | System Analysis & Design | 12 |
| 3. | Coding & Implementation | 45 |
| 4. | Testing | 10 |
| 5. | Maintenance & others | 5 |
| 6. | Project Documentation | 90 |
| **Total** | | **90 days** |

Table: SDLC Time Period

* 1. **Gantt chart**

**­­**

Figure: Gantt chart for SDLC Time Period

1. **Methodology Adopted**
   1. **Project Methodology**

This project is based on open source applications such as Core Java, JSP & MySQL. JSP is an open-source HTML-embedded scripting language that allows writing dynamically generated professional pages quickly. MySQL, the most popular Open Source SQL database management system, runs as a server providing multi-user access to a number of databases.

Some the striking features of open source Programming approaches are:

* Emphasis is on code rather than procedure.
* Programs are divided into what are known as objects.
* Data structures are designed such that they characterize the objects.
* Functions that operate on the data of and object are tied together in the data structure.
* Data is hidden and cannot be accessed by external functions.
* New data and functions can be easily added whenever necessary.
* Follows bottom-up approaches in program design.
  1. **H/W and S/W Used**

**7.2.1 Hardware**

* For Users, a PC, Tablet or any device that has support for browsers
* For Server, a PC configured with minimum:
  + Pentium IV Processor
  + 512 MB RAM
  + 40 GB HDD
  + Mouse/Keyboard
  + Monitor (Min Resolution: 1024x768)

**7.2.2 Software**

* For Users, a Web Browser (e.g.; Firefox, Chrome, etc.)
* For Server, Web Server (Apache Tomcat + MySQL Server)
* Client-Side Technologies: HTML5, CSS3, JavaScript, AJAX
* Server-Side Technologies: JSP 2.0, MySQL 5.6
* Eclipse Kepler for JEE Development
* SQL Query Browser

1. **System Architecture overview**
   1. **ER Diagram**

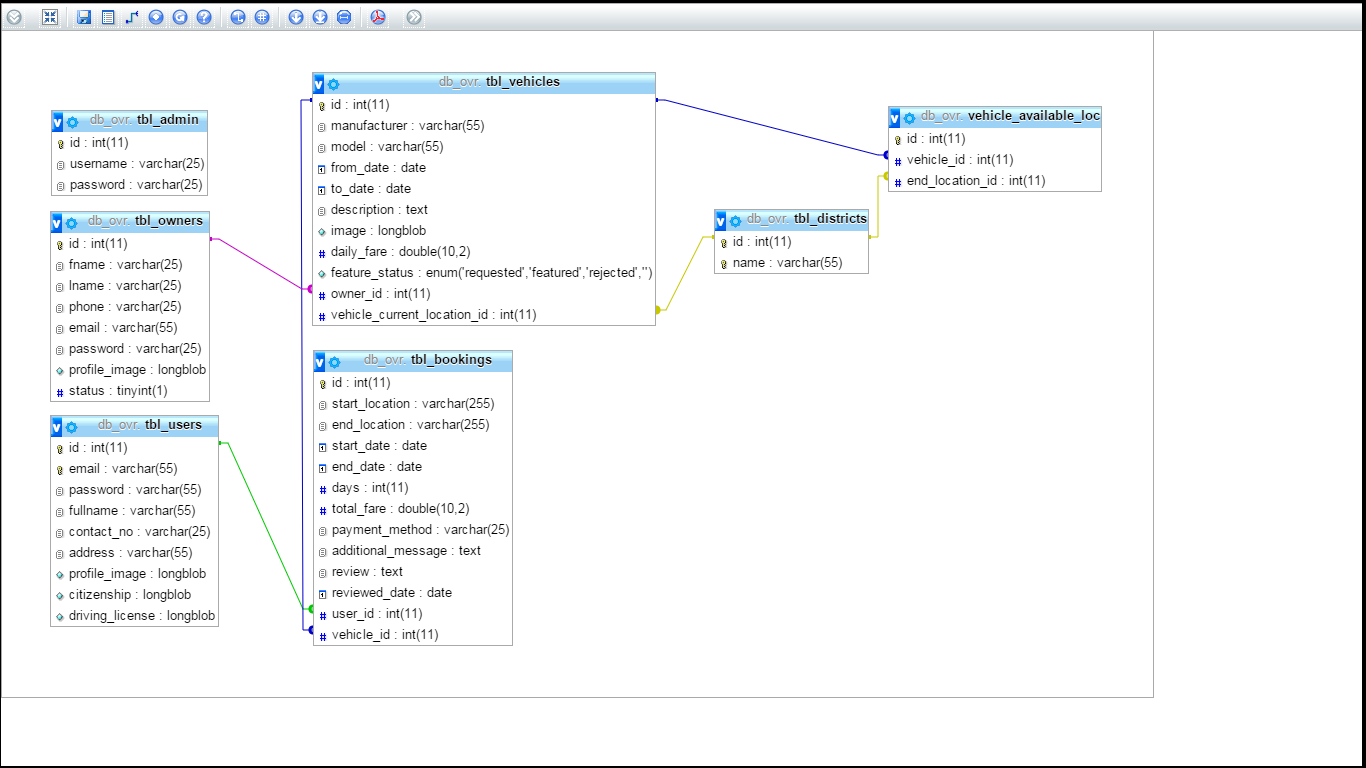


Figure: ER Diagram for “db\_ovr”

* 1. **Data Flow Diagram (Administrator)**

Login Page

If User/Pass

Correct?

Settings

Vehicle Owner Management

Feature Vehicles

Change Password

Activate/Deactivate

Accept/Reject

Logout

Yes

No

View Details

Delete

* 1. **Data Flow Diagram (Vehicle Owner)**

Yes

No

Register

Profile

Vehicles

Bookings

Logout

Edit

Add/Edit/Delete

Feature Request

Activate/Deactivate

Details

* 1. **Data Flow Diagram (User)**

Login PageLogin Page

If User/Pass

Correct?If User/Pass

Correct?

Yes

No

Register

Profile

Bookings

Logout

Edit

Details

Search Vehicle

Book a Vehicle

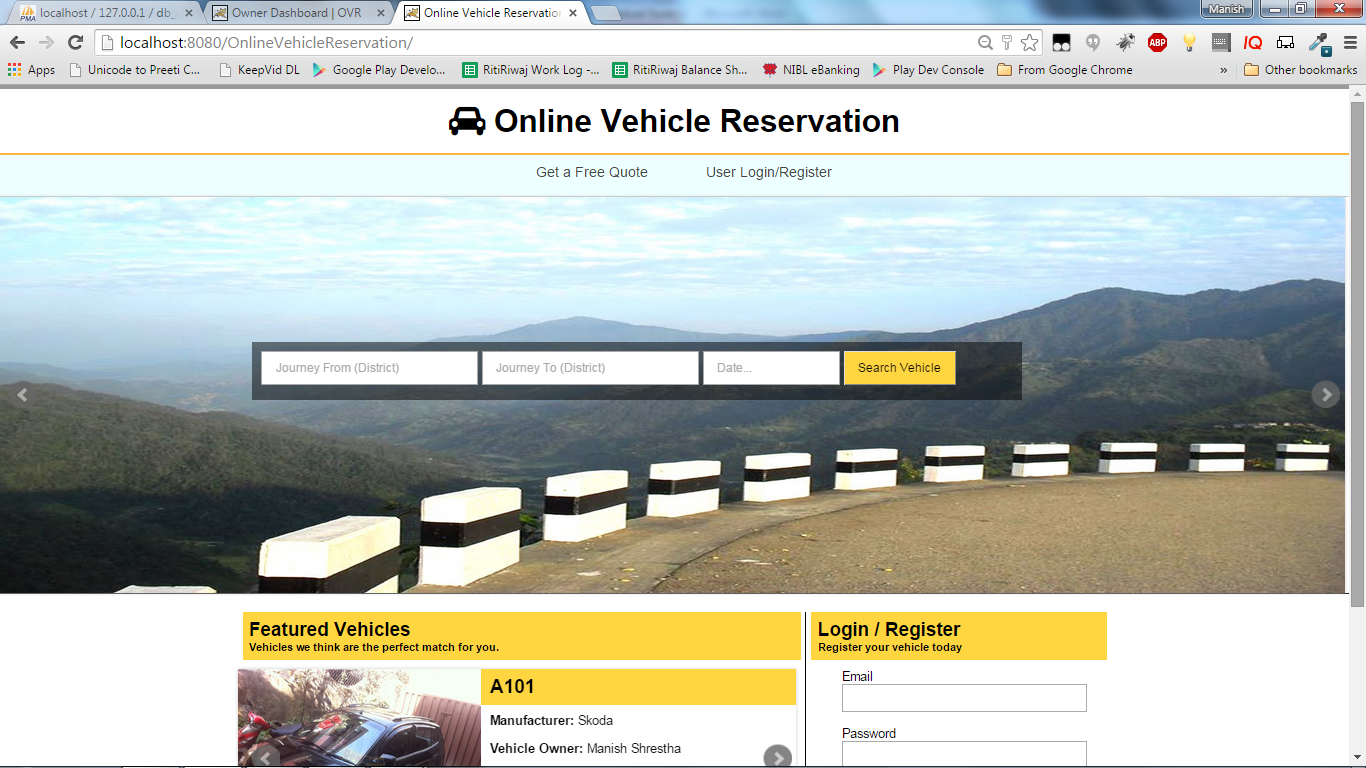
1. **Testing Methods**
   1. **Black Box Testing**

A tester was interacted with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon. A successful Black Box Testing assures the functionality of software from the users view. It checks for the runtime error and helps in making the software more productive & error free.

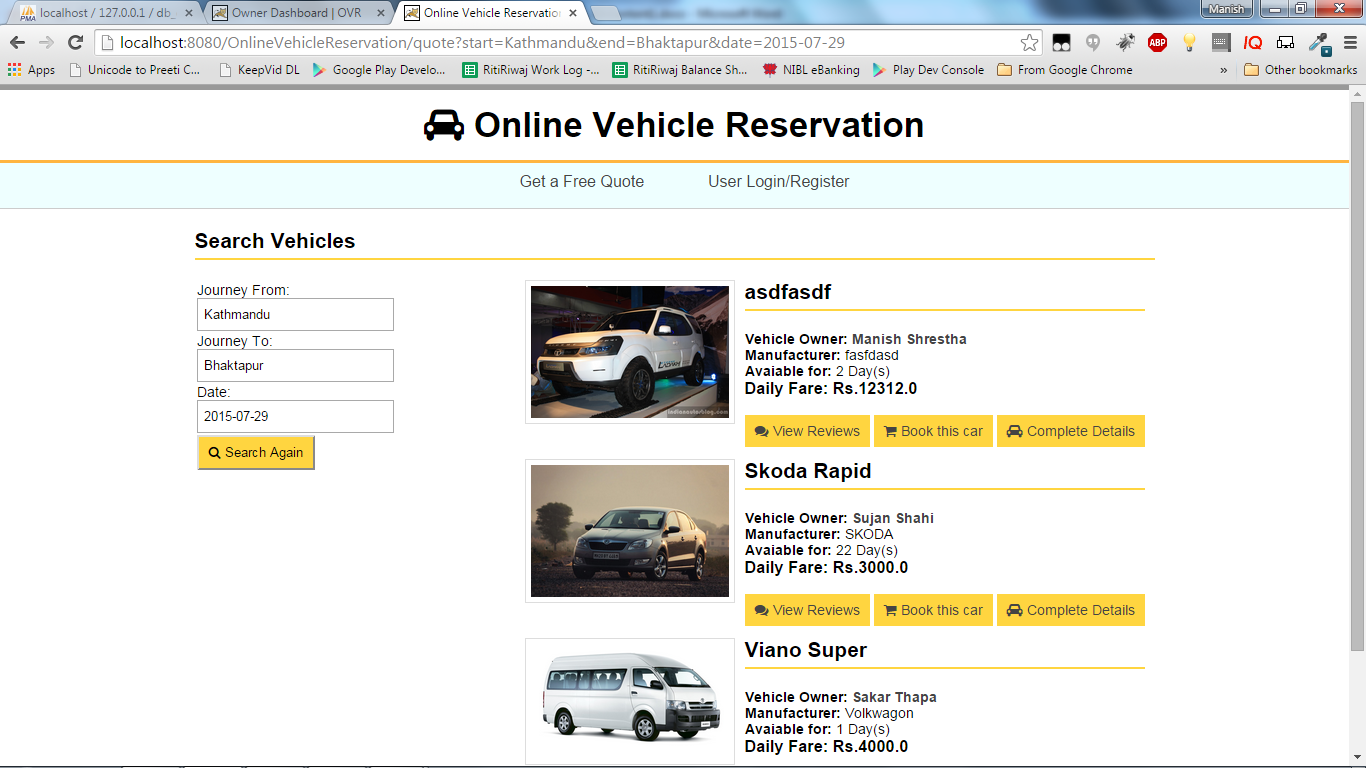
* 1. **White Box Testing**

White box testing is the detailed investigation of internal logic and structure of the code. In order to perform white box testing on an application, the tester needs to possess knowledge of the internal working of the code. It helps in optimizing the code. Extra lines of code can be removed which can bring in hidden defects.

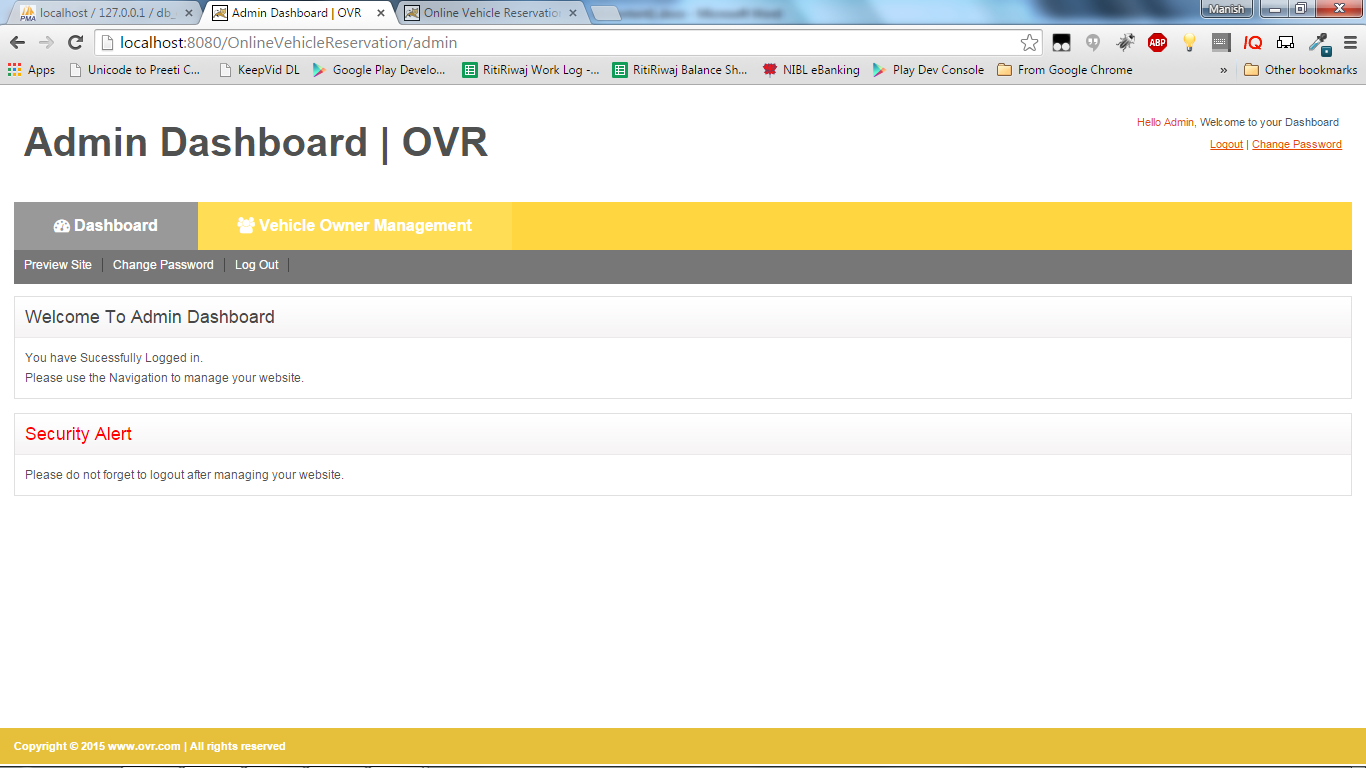
1. **Detail of the project with snapshot and codes**
   1. **Snapshots**
      1. **Home Page (Index)**



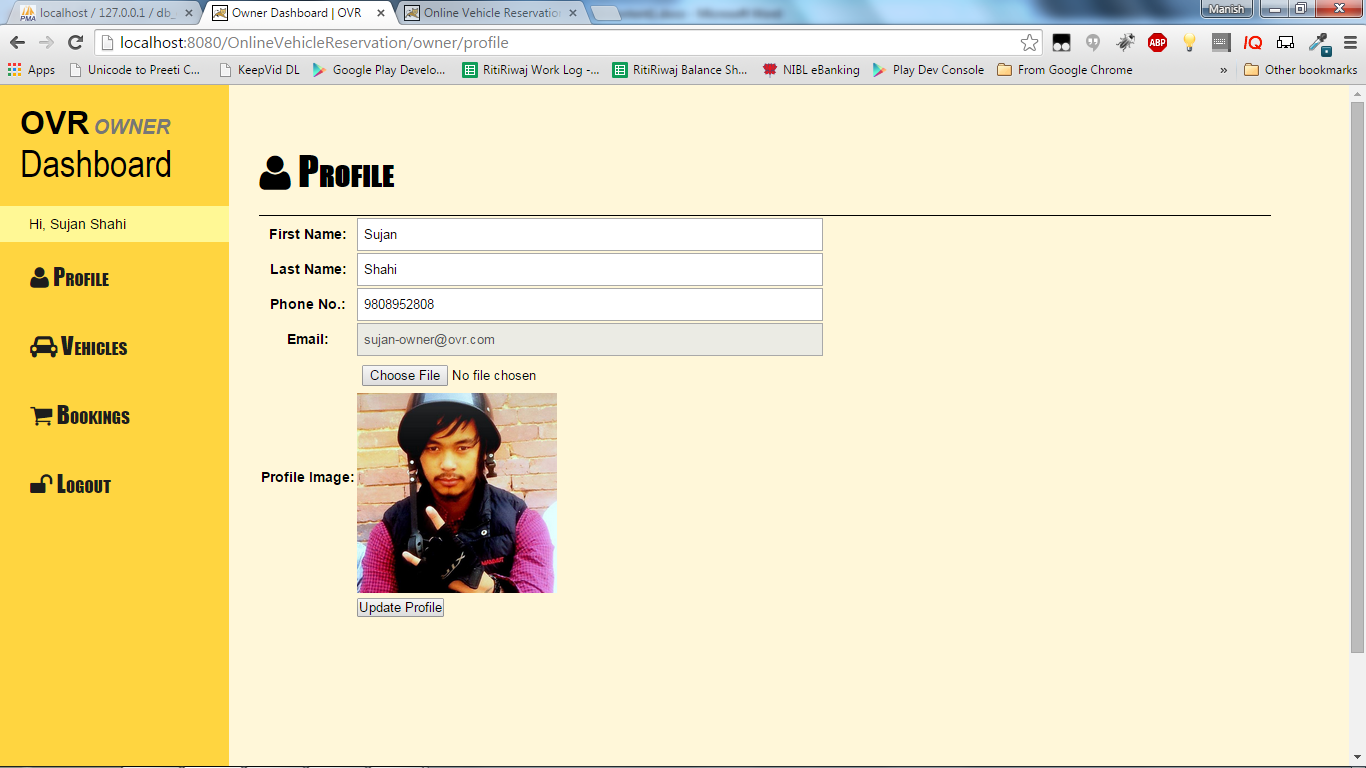
* + 1. **Vehicle Search Results**

****

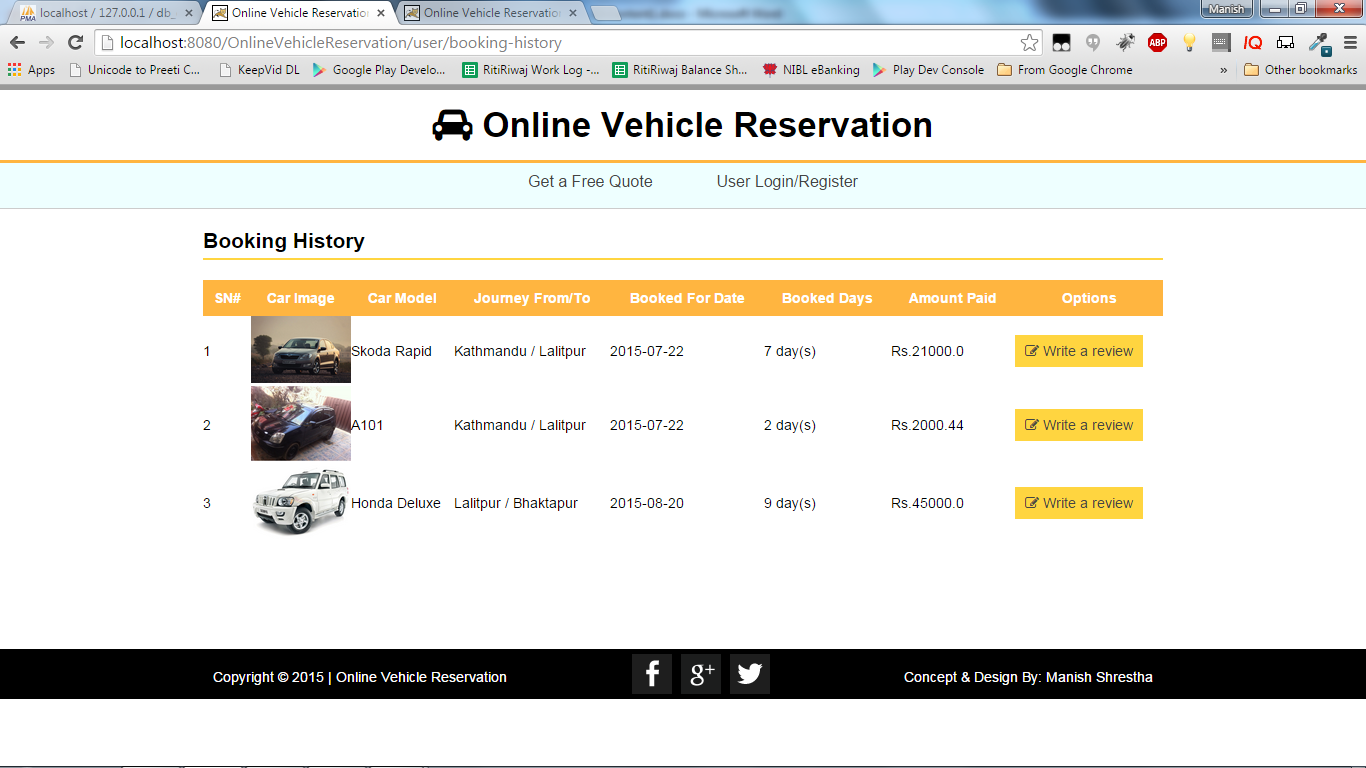
* + 1. **Admin Dashboard**

****

* + 1. **Vehicle Owner Dashboard**

****

* + 1. **User Dashboard**

****

* 1. **Source Code**

### utils/Database.java

package com.ovr.utils;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

abstract public class Database {

static String driver = "com.mysql.jdbc.Driver";

static String url = "jdbc:mysql://localhost:3306/db\_ovr";

static String user = "root";

static String pass = "";

public static Connection connect(){

Connection con = null;

try {

Class.forName(driver);

} catch (ClassNotFoundException e) {

System.err.println("[DATABASE CONNECTION FAILED]: "+e);

}

try {

con = DriverManager.getConnection(url, user, pass);

} catch (SQLException e) {

System.err.println("[DATABASE CONNECTION FAILED]: "+e);

}

return con;

}

}

**utils/Helper.java**

package com.ovr.utils;

import java.io.IOException;

import java.io.PrintWriter;

import java.math.BigDecimal;

import java.math.RoundingMode;

import java.text.DateFormat;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.concurrent.TimeUnit;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

public class Helper {

public static Date stringToDate(String strDate) {

DateFormat formatter;

Date date = null;

formatter = new SimpleDateFormat("yyyy-MM-dd");

try {

date = formatter.parse(strDate);

} catch (ParseException e) {

e.printStackTrace();

}

System.out.println(strDate+": === "+date.toString());

return date;

}

public static String getURISegment(int i, HttpServletRequest request){

String fullPath = (String) request.getAttribute("javax.servlet.forward.request\_uri");

String[] uriSegments = fullPath.split("/");

if(i >= uriSegments.length){

return "";

} else {

return uriSegments[i];

}

}

public static String printSessionMsg(HttpSession session){

if(session.getAttribute("msg-success") != null){

String msg = "<div class='alert-success'>"+session.getAttribute("msg-success")+"</div>";

session.removeAttribute("msg-success");

return msg;

} else if (session.getAttribute("msg-error") != null){

String msg = "<div class='alert-danger'>"+session.getAttribute("msg-error")+"</div>";

session.removeAttribute("msg-error");

return msg;

} else {

return "";

}

}

public static void setSuccessMsg(String msg, HttpSession session){

session.setAttribute("msg-success", msg);

}

public static void setErrorMsg(String msg, HttpSession session){

session.setAttribute("msg-error", msg);

}

/\*

\* @userType (String)

\* owner, admin, user

\*

\*/

public static boolean isLoggedIn(String userType, HttpSession session) {

if(userType.equals("admin"))

return session.getAttribute("admin\_id") != null ? true : false;

else if (userType.equals("owner"))

return session.getAttribute("owner\_id") != null ? true : false;

else if (userType.equals("user"))

return session.getAttribute("user\_id") != null ? true : false;

else

return false;

}

public static long getDateDiff(Date date1, Date date2, TimeUnit timeUnit) {

long diffInMillies = date2.getTime() - date1.getTime();

return timeUnit.convert(diffInMillies,TimeUnit.MILLISECONDS);

}

public static double round(double value, int places) {

if (places < 0) throw new IllegalArgumentException();

BigDecimal bd = new BigDecimal(value);

bd = bd.setScale(places, RoundingMode.HALF\_UP);

return bd.doubleValue();

}

public static void toJson(Object object, HttpServletResponse response){

response.setContentType("application/json");

// Get the printwriter object from response to write the required json object to the output stream

PrintWriter out;

try {

out = response.getWriter();

out.print(object);

out.flush();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

### model/User.java

package com.ovr.model;

import java.io.InputStream;

public class User {

private int id;

private String email;

private String password;

private String fullname;

private String contactNo;

private String address;

private InputStream profileImage;

private InputStream citizenship;

private InputStream drivingLicense;

public String getFullname() {

return fullname;

}

public void setFullname(String fullname) {

this.fullname = fullname;

}

public String getContactNo() {

return contactNo;

}

public void setContactNo(String contactNo) {

this.contactNo = contactNo;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public InputStream getProfileImage() {

return profileImage;

}

public void setProfileImage(InputStream profileImage) {

this.profileImage = profileImage;

}

public InputStream getCitizenship() {

return citizenship;

}

public void setCitizenship(InputStream citizenship) {

this.citizenship = citizenship;

}

public InputStream getDrivingLicense() {

return drivingLicense;

}

public void setDrivingLicense(InputStream drivingLicense) {

this.drivingLicense = drivingLicense;

}

public User() {

// TODO Auto-generated constructor stub

}

public User(int id){

this.id = id;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

}

### dao/UserDao.java

package com.ovr.dao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.ArrayList;

import java.util.List;

import java.util.Map;

import com.ovr.model.User;

import com.ovr.utils.Database;

public class UserDao {

private Connection conn;

private PreparedStatement ps;

private ResultSet rs;

private String sql;

private List<User> userList;

private int lastInsertedId;

public int insert(Map<String, Object> data){

conn = Database.connect();

try {

sql = "INSERT INTO `tbl\_users` SET";

for(String key : data.keySet()){

sql += " `"+key+"` = ?,";

}

sql = sql.substring(0, sql.length()-1);

ps = conn.prepareStatement(sql, Statement.RETURN\_GENERATED\_KEYS);

int paramIndex=1;

for(String key : data.keySet()){

ps.setObject(paramIndex, data.get(key));

paramIndex++;

}

System.out.println(ps.toString());

ps.executeUpdate();

rs = ps.getGeneratedKeys();

if(rs.next())

lastInsertedId = rs.getInt(1);

} catch (SQLException e) {

System.err.println("UserDao (insert): "+e.getMessage());

}

return lastInsertedId;

}

public List<User> getWhere(Map<String, Object> condition, String operator){

conn = Database.connect();

try {

sql = "SELECT \* FROM `tbl\_users` WHERE";

for(String key : condition.keySet()){

sql += " `"+key+"` = '"+condition.get(key)+"' "+operator;

}

if(operator.equals("AND"))

sql = sql.substring(0, sql.length()-3);

else if (operator.equals("OR"))

sql = sql.substring(0, sql.length()-2);

ps = conn.prepareStatement(sql);

System.out.println(ps.toString());

ResultSet result = ps.executeQuery();

userList = new ArrayList<User>();

while(result.next()){

User user = new User();

user.setId(result.getInt("id"));

user.setEmail(result.getString("email"));

user.setPassword(result.getString("password"));

user.setFullname(result.getString("fullname"));

user.setContactNo(result.getString("contact\_no"));

user.setAddress(result.getString("address"));

user.setProfileImage(result.getBinaryStream("profile\_image"));

user.setCitizenship(result.getBinaryStream("citizenship"));

user.setDrivingLicense(result.getBinaryStream("driving\_license"));

userList.add(user);

}

} catch (SQLException e) {

System.err.println("UserDao (getWhere): "+e.getMessage());

}

return userList;

}

}

**dao/CommonDao.java**

package com.ovr.dao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import com.ovr.model.District;

import com.ovr.utils.Database;

public class CommonDao {

District district;

List<District> districtList;

Connection conn;

Statement stmt;

ResultSet rs;

int updatedRows;

String sql;

PreparedStatement ps;

int lastInsertedId;

public int insert(String tblName, Map<String, Object> data) {

conn = Database.connect();

try {

sql = "INSERT INTO `" + tblName + "` SET";

for (String key : data.keySet()) {

sql += " `" + key + "` = ?,";

}

sql = sql.substring(0, sql.length() - 1);

ps = conn.prepareStatement(sql, Statement.RETURN\_GENERATED\_KEYS);

int paramIndex = 1;

for (String key : data.keySet()) {

ps.setObject(paramIndex, data.get(key));

paramIndex++;

}

System.out.println(ps.toString());

ps.executeUpdate();

rs = ps.getGeneratedKeys();

if (rs.next())

lastInsertedId = rs.getInt(1);

} catch (SQLException e) {

System.err.println("CommonDao (insert): " + e.getMessage());

}

return lastInsertedId;

}

public List<District> getAllDistrict() {

conn = Database.connect();

districtList = new ArrayList<District>();

try {

String sql = "SELECT \* FROM tbl\_districts";

stmt = conn.createStatement();

rs = stmt.executeQuery(sql);

while (rs.next()) {

district = new District();

district.setId(rs.getInt("id"));

district.setName(rs.getString("name"));

districtList.add(district);

}

} catch (SQLException e) {

System.err.println("SQL ERROR: " + e.getMessage());

}

return districtList;

}

public List getAll(String tblName) {

conn = Database.connect();

List list = new ArrayList<>();

try {

String sql = "SELECT \* FROM `" + tblName + "`";

stmt = conn.createStatement();

rs = stmt.executeQuery(sql);

while (rs.next()) {

list.add(rs);

}

} catch (SQLException e) {

System.err.println("CommonDao (getAll) : " + e.getMessage());

}

return list;

}

public int update(String tblName, Map<String, Object> data,

Map<String, Object> condition) {

conn = Database.connect();

String sql = "UPDATE `" + tblName + "` SET";

for (String key : data.keySet()) {

sql += " `" + key + "` = ?,";

}

sql = sql.substring(0, sql.length() - 1);

sql += " WHERE";

for (String key : condition.keySet()) {

sql += " `" + key + "` = ? AND";

}

sql = sql.substring(0, sql.length() - 3);

System.out.println(sql);

try {

ps = conn.prepareStatement(sql);

int index = 1;

for (String key : data.keySet()) {

ps.setObject(index, data.get(key));

index++;

}

for (String key : condition.keySet()) {

ps.setObject(index, condition.get(key));

index++;

}

System.out.println(ps.toString());

updatedRows = ps.executeUpdate();

} catch (SQLException e) {

System.err.println("SQL Error: " + e.getMessage());

}

return updatedRows;

}

public int getCount(String tblName, Map<String, Object> condition,

String operator) {

String sql;

int count=0;

Connection conn = Database.connect();

try {

sql = "SELECT \* FROM `" + tblName + "` WHERE";

for (String key : condition.keySet()) {

sql += " `" + key + "` = '" + condition.get(key) + "' "

+ operator;

}

if (operator.equals("AND"))

sql = sql.substring(0, sql.length() - 3);

else if (operator.equals("OR"))

sql = sql.substring(0, sql.length() - 2);

PreparedStatement ps = conn.prepareStatement(sql);

System.out.println(ps.toString());

ResultSet result = ps.executeQuery();

while(result.next()){

count++;

}

} catch (SQLException e) {

System.err.println(e.getMessage());

}

return count;

}

}

**controller/UserController.java**

package com.ovr.controller;

import java.io.IOException;

import java.io.InputStream;

import java.io.PrintWriter;

import java.util.Date;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import javax.servlet.ServletException;

import javax.servlet.annotation.MultipartConfig;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import com.ovr.dao.BookingDao;

import com.ovr.dao.CommonDao;

import com.ovr.dao.OwnerDao;

import com.ovr.dao.UserDao;

import com.ovr.model.Booking;

import com.ovr.model.User;

import com.ovr.utils.Helper;

import com.ovr.utils.ImageUtil;

@WebServlet(urlPatterns = { "/user", "/user/login", "/user/logout", "/user/edit-profile", "/user/image", "/user/booking-history", "/user/ajax\_checkIfEmailExists", "/user/authenticate", "/user/review-add-edit" })

@MultipartConfig

public class UserController extends HttpServlet {

private static final long serialVersionUID = 1L;

private HttpSession session;

private UserDao userDao;

private Map<String, Object> data = new HashMap<String, Object>();

private Map<String, Object> condition = new HashMap<String, Object>();

private int updatedRows;

protected void doGet(HttpServletRequest request,

HttpServletResponse response) throws ServletException, IOException {

String uri = request.getServletPath();

session = request.getSession();

switch (uri) {

case "/user":

int userId = Integer.valueOf((String) session.getAttribute("user\_id"));

UserDao uDao = new UserDao();

condition = new HashMap<String, Object>();

condition.put("id", userId);

List<User> u = uDao.getWhere(condition, "");

request.setAttribute("user", u.get(0));

request.getRequestDispatcher(

"/WEB-INF/views/frontend/user/index.jsp").forward(request,

response);

break;

case "/user/login":

if(Helper.isLoggedIn("user", session)) { response.sendRedirect(request.getContextPath()+"/user"); return; }

request.getRequestDispatcher(

"/WEB-INF/views/frontend/user/login.jsp").forward(request,

response);

break;

case "/user/logout":

session.removeAttribute("user\_id");

session.removeAttribute("user\_email");

response.sendRedirect(request.getContextPath()+"/user/login");

break;

case "/user/booking-history":

BookingDao bDao = new BookingDao();

condition = new HashMap<String, Object>();

condition.put("user\_id", Integer.valueOf((String) session.getAttribute("user\_id")));

List<Booking> bookingList = bDao.getWhere(condition, "");

request.setAttribute("bookingList", bookingList);

request.getRequestDispatcher(

"/WEB-INF/views/frontend/user/booking-history.jsp").forward(request,

response);

break;

case "/user/review-add-edit":

bDao = new BookingDao();

condition = new HashMap<String, Object>();

condition.put("b`.`id", Integer.valueOf(request.getParameter("booking\_id")));

List<Booking> b = bDao.getWhere(condition, "");

request.setAttribute("booking", b.get(0));

request.getRequestDispatcher(

"/WEB-INF/views/frontend/user/review-add-edit.jsp").forward(request, response);

break;

case "/user/image":

userDao = new UserDao();

condition = new HashMap<String, Object>();

if(request.getParameterMap().containsKey("user\_id"))

condition.put("id", Integer.valueOf(request.getParameter("user\_id")));

else

condition.put("id", Integer.valueOf((String) session.getAttribute("user\_id")));

List<User> userList = userDao.getWhere(condition, "");

InputStream imageData = null;

if(request.getParameter("type").equals("profile-image"))

imageData = userList.get(0).getProfileImage();

else if(request.getParameter("type").equals("citizenship"))

imageData = userList.get(0).getCitizenship();

else if(request.getParameter("type").equals("driving-license"))

imageData = userList.get(0).getDrivingLicense();

ImageUtil.writeStreamImage(imageData, response);

break;

default:

break;

}

}

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

String uri = request.getServletPath();

session = request.getSession();

session.setMaxInactiveInterval(-1);

switch (uri) {

case "/user/ajax\_checkIfEmailExists":

userDao = new UserDao();

condition = new HashMap<String, Object>();

condition.put("email", request.getParameter("email"));

List<User> userList = userDao.getWhere(condition, "");

response.setContentType("text/plain");

PrintWriter out = response.getWriter();

if (userList.isEmpty()) {

out.write("noexist");

} else {

out.write("exists");

}

out.close();

break;

case "/user/authenticate":

userDao = new UserDao();

if(request.getParameter("action").equals("login")){

System.out.println("LOGIN");

condition = new HashMap<String, Object>();

condition.put("email", request.getParameter("email"));

condition.put("password", request.getParameter("password"));

userList = userDao.getWhere(condition, "AND");

if(!userList.isEmpty()){

session.setAttribute("user\_id", String.valueOf(userList.get(0).getId()));

session.setAttribute("user\_email", userList.get(0).getEmail());

if(session.getAttribute("login-request") != null && session.getAttribute("login-request").equals("from-booking")){

session.removeAttribute("login-request");

response.sendRedirect(request.getContextPath()+"/quote/book");

} else {

response.sendRedirect(request.getContextPath()+"/user");

}

} else {

Helper.setErrorMsg("Invalid User/Pass", session);

response.sendRedirect(request.getContextPath()+"/user/login");

}

} else if (request.getParameter("action").equals("register")){

System.out.println("REGISTER");

data = new HashMap<String, Object>();

data.put("email", request.getParameter("email"));

data.put("password", request.getParameter("password"));

int lastInsertedId = userDao.insert(data);

session.setAttribute("user\_id", String.valueOf(lastInsertedId));

session.setAttribute("user\_email", request.getParameter("email"));

if(session.getAttribute("login-request") != null && session.getAttribute("login-request").equals("from-booking")){

session.removeAttribute("login-request");

response.sendRedirect(request.getContextPath()+"/quote/book");

} else {

response.sendRedirect(request.getContextPath()+"/user");

}

}

break;

case "/user/review-add-edit":

CommonDao cDao = new CommonDao();

condition = new HashMap<String, Object>();

condition.put("id", Integer.valueOf(request.getParameter("booking\_id")));

data = new HashMap<String, Object>();

data.put("review", request.getParameter("review"));

data.put("reviewed\_date", new Date());

updatedRows = cDao.update("tbl\_bookings", data, condition);

if(updatedRows>0)

Helper.setSuccessMsg("Vehicle review added successfully.", session);

else

Helper.setErrorMsg("Something went wrong.", session);

response.sendRedirect(request.getContextPath()+"/user/booking-history");

break;

case "/user/edit-profile":

User user = new User();

user.setFullname(request.getParameter("fullname"));

user.setContactNo(request.getParameter("contact\_no"));

user.setAddress(request.getParameter("address"));

UserDao uDao = new UserDao();

condition = new HashMap<String, Object>();

condition.put("id", Integer.valueOf((String) session.getAttribute("user\_id")));

List<User> prevUserData = uDao.getWhere(condition, "");

if(request.getPart("profile\_image").getSize() > 0)

user.setProfileImage(request.getPart("profile\_image").getInputStream());

else

user.setProfileImage(prevUserData.get(0).getProfileImage());

if(request.getPart("citizenship").getSize() > 0)

user.setCitizenship(request.getPart("citizenship").getInputStream());

else

user.setCitizenship(prevUserData.get(0).getCitizenship());

if(request.getPart("driving\_license").getSize() > 0)

user.setDrivingLicense(request.getPart("driving\_license").getInputStream());

else

user.setDrivingLicense(prevUserData.get(0).getDrivingLicense());

data = new HashMap<String, Object>();

data.put("fullname", user.getFullname());

data.put("contact\_no", user.getContactNo());

data.put("address", user.getAddress());

data.put("profile\_image", user.getProfileImage());

data.put("citizenship", user.getCitizenship());

data.put("driving\_license", user.getDrivingLicense());

cDao = new CommonDao();

cDao.update("tbl\_users", data, condition);

Helper.setSuccessMsg("User profile updated successfully.", session);

response.sendRedirect(request.getContextPath()+"/user");

break;

default:

break;

}

}

}

**controller/PagesController.java**

package com.ovr.controller;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import com.ovr.dao.BookingDao;

import com.ovr.dao.VehicleDao;

import com.ovr.model.Booking;

import com.ovr.model.District;

import com.ovr.model.Vehicle;

import com.ovr.utils.Helper;

@WebServlet (urlPatterns = {"/home", "/about", "/faqs", "/testimonials", "/contact", "/vehicle-details", "/ajax\_getVehicleCurrentLocation", "/ajax\_getVehicleAvailableLocations"})

public class PagesController extends HttpServlet{

private static final long serialVersionUID = 1L;

private HttpSession session;

private Map<String, Object> data = new HashMap<String, Object>();

private Map<String, Object> condition = new HashMap<String, Object>();

private VehicleDao vDao;

private String uri;

private Vehicle v;

private PrintWriter out;

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

uri = request.getServletPath();

session = request.getSession();

session.setMaxInactiveInterval(-1);

switch (uri) {

case "/home":

vDao = new VehicleDao();

condition.put("feature\_status", "featured");

List<Vehicle> featuredVehicles = vDao.getWhere(condition, "");

request.setAttribute("featuredVehicles", featuredVehicles);

request.getRequestDispatcher("/WEB-INF/views/frontend/index.jsp").forward(request, response);

break;

case "/about":

request.getRequestDispatcher("/WEB-INF/views/frontend/default.jsp").forward(request, response);

break;

case "/faqs":

request.getRequestDispatcher("/WEB-INF/views/frontend/default.jsp").forward(request, response);

break;

case "/testimonials":

request.getRequestDispatcher("/WEB-INF/views/frontend/default.jsp").forward(request, response);

break;

case "/contact":

request.getRequestDispatcher("/WEB-INF/views/frontend/default.jsp").forward(request, response);

break;

case "/vehicle-details":

int vehicleId = Integer.valueOf(request.getParameter("id"));

vDao = new VehicleDao();

Vehicle v = vDao.getVehicleDetails(vehicleId);

BookingDao bDao = new BookingDao();

List<Booking> bList = bDao.getAllReviewsByVehicleId(Integer.valueOf(request.getParameter("id")));

request.setAttribute("vehicle", v);

request.setAttribute("bookingList", bList);

request.getRequestDispatcher("/WEB-INF/views/frontend/vehicle-details.jsp").forward(request, response);

break;

default:

System.out.println("No page defined");

break;

}

}

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

uri = request.getServletPath();

session = request.getSession();

switch (uri) {

case "/ajax\_getVehicleCurrentLocation":

vDao = new VehicleDao();

v = vDao.getVehicleDetails(Integer.valueOf(request.getParameter("vehicle\_id")));

response.setContentType("text/plain");

out = response.getWriter();

out.write(v.getDistrict().getName());

out.flush();

out.close();

break;

case "/ajax\_getVehicleAvailableLocations":

vDao = new VehicleDao();

v = vDao.getVehicleDetails(Integer.valueOf(request.getParameter("vehicle\_id")));

response.setContentType("text/plain");

out = response.getWriter();

String locations = "";

for(District d : v.getDistrictList()){

locations += d.getName()+",";

}

locations = locations.substring(0, locations.length()-1);

out.write(locations);

out.flush();

out.close();

break;

default:

break;

}

}

}

**WEB-INF/views/frontend/include/header.jsp**

<!doctype html>

<html lang="en">

<head>

<meta charset="utf-8" />

<title>Online Vehicle Reservation</title>

<script src="<%= request.getContextPath() %>/assets/js/jquery.min.js"></script>

<!-- JQuery UI -->

<link href="<%= request.getContextPath() %>/assets/css/jquery-ui.min.css" rel="stylesheet">

<script src="<%= request.getContextPath() %>/assets/js/jquery-ui.min.js" type="text/javascript"></script>

<!-- BXSlider -->

<link href="<%= request.getContextPath() %>/assets/bxslider/jquery.bxslider.css" rel="stylesheet">

<script src="<%= request.getContextPath() %>/assets/bxslider/jquery.bxslider.min.js" type="text/javascript"></script>

<!-- STYLESHEETS -->

<link href="<%= request.getContextPath() %>/assets/css/style.css" rel="stylesheet" type="text/css" />

<link href="<%= request.getContextPath() %>/assets/css/forms.css" rel="stylesheet" type="text/css" />

<!-- JQuery Validate -->

<script src="<%= request.getContextPath() %>/assets/js/jquery.validate.js" type="text/javascript"></script>

<!-- Font Awesome -->

<link rel="stylesheet" href="<%= request.getContextPath() %>/assets/font-awesome-4.3.0/css/font-awesome.min.css">

<script>

$(document).ready(function () {

$('.bxslider').bxSlider({

mode: 'fade',

auto: true,

pager: false

});

$('.bxslider-featured-box').bxSlider({auto:true});

});

</script>

</head>

<body>

<!-- HEADER SECTION -->

<div id="header">

<div class="wrapper">

<div id="logo">

<h1 align="center"><i class="fa fa-automobile"></i> Online Vehicle Reservation</h1>

</div>

</div>

</div>

<div class="clear"></div>

**WEB-INF/views/frontend/include/navbar.jsp**

<%@page import="com.ovr.utils.Helper"%>

<div id="nav-bar">

<div class="wrapper">

<ul>

<li><a href="<%= request.getContextPath() %>" class="">Get a Free Quote</a></li>

<li>

<a href="<%= request.getContextPath() %>/user/login" class="">User Login/Register</a>

<% if(Helper.isLoggedIn("user", session)){ %>

<ul>

<li><small>Hello,</small><br> <strong><%= session.getAttribute("user\_email") %></strong></li>

<li><a href="<%= request.getContextPath() %>/user">Profile</a></li>

<li><a href="<%= request.getContextPath() %>/user/booking-history">Booking History</a></li>

<li><a href="<%= request.getContextPath() %>/user/logout">Logout</a></li>

</ul>

<% } %>

</li>

</ul>

</div>

</div>

**WEB-INF/views/frontend/include/footer.jsp**

<!-- FOOTER SECTION -->

<div id="footer">

<div id="footer-container">

<div id="copy-text">Copyright &copy; 2015 | Online Vehicle Reservation</div>

<div id="social-buttons">

<a href="#" alt="Follow Us on Facebook" title="Facebook" class="facebook">Facebook</a>

<a href="#" alt="Follow Us on Google Plus" title="Google Plus" class="gplus">Google Plus</a>

<a href="#" alt="Follow Us on Twitter" title="Twitter" class="twitter">Twitter</a>

</div>

<div id="design-by">Concept & Design By: Manish Shrestha</div>

</div>

</div>

</body>

</html>

**WEB-INF/views/frontend/index.jsp**

<%@page import="java.util.ArrayList"%>

<%@page import="com.ovr.model.Vehicle"%>

<%@page import="java.util.List"%>

<%@ include file="include/header.jsp"%>

<%@ include file="include/navbar.jsp"%>

<script>

$(function() {

var districts = [

'Achham','Arghakhanchi','Baglung','Baitadi','Bajhang','Bajura','Banke','Bara','Bardiya','Bhaktapur','Bhojpur','Chitwan','Dadeldhura','Dailekh','Dang Deukhuri','Darchula','Dhading','Dhankuta','Dhanusa','Dholkha','Dolpa','Doti','Gorkha','Gulmi','Humla','Ilam','Jajarkot','Jhapa','Jumla','Kailali','Kalikot','Kanchanpur','Kapilvastu','Kaski','Kathmandu','Kavrepalanchok','Khotang','Lalitpur','Lamjung','Mahottari','Makwanpur','Manang','Morang','Mugu','Mustang','Myagdi','Nawalparasi','Nuwakot','Okhaldhunga','Palpa','Panchthar','Parbat','Parsa','Pyuthan','Ramechhap','Rasuwa','Rautahat','Rolpa','Rukum','Rupandehi','Salyan','Sankhuwasabha','Saptari','Sarlahi','Sindhuli','Sindhupalchok','Siraha','Solukhumbu','Sunsari','Surkhet','Syangja','Tanahu','Taplejung','Terhathum','Udayapur'

];

$( "#start, #end" ).autocomplete({

source: districts

});

});

</script>

<!-- SLIDER SECTION -->

<div id="slider">

<ul class="bxslider">

<li><img src="<%= request.getContextPath() %>/assets/images/banner1.jpg" alt="Slider Image 1" /></li>

<li><img src="<%= request.getContextPath() %>/assets/images/banner2.jpg" alt="Slider Image 1" /></li>

</ul>

<div class="search-box">

<form action="<%= request.getContextPath() %>/quote" id="banner-quote-form">

<input type="text" name="start" id="start" class="journey-input" placeholder="Journey From (District)" required>

<input type="text" name="end" id="end" class="journey-input" placeholder="Journey To (District)" required>

<input type="text" name="date" class="datepicker journey-input-date" placeholder="Date..." required>

<button type="submit">Search Vehicle</button>

</form>

</div>

</div>

<div class="hr"></div>

<!-- QUICK NAVIGATION SECTION -->

<div id="quick-nav">

<div id="featured-box">

<h2>Featured Vehicles <span>Vehicles we think are the perfect match for you.</span></h2>

<ul class="bxslider-featured-box">

<%

List<Vehicle> featuredVehicles = (ArrayList<Vehicle>) request.getAttribute("featuredVehicles");

if(!featuredVehicles.isEmpty()){

for(Vehicle v : featuredVehicles){

%>

<li>

<div class="vehicle-image">

<img src="<%= request.getContextPath() %>/owner/vehicle/image?vehicle\_id=<%= v.getId() %>">

</div>

<div class="vehicle-desc">

<h2><%= v.getModel() %></h2>

<p><strong>Manufacturer:</strong> <%= v.getManufacturer() %></p><br>

<p><strong>Vehicle Owner:</strong> <%= v.getOwner().getFname()+" "+v.getOwner().getLname() %></p><br>

<p><strong>Available for: <span class="text-success">Rs.<%= (int) v.getDailyFare() %> / Day</span></strong></p>

<p><strong>Dates Available: <%= v.getFromDate()+" to "+v.getToDate() %></strong></p><br>

<button type="button" class="btn" onclick="quoteModal(<%= v.getId() %>, '<%= v.getModel() %>')">Book Today</button>

</div>

</li>

<%

}

}

%>

</ul>

</div>

<div id="vehicle-owner-login">

<h2>Login / Register <span>Register your vehicle today</span></h2>

<div id="login-form">

<form action="<%= request.getContextPath() %>/owner/login" method="post" id="owner-login-form">

<table>

<tr>

<td>

<label>Email</label><br>

<input type="email" name="email" required>

<br><br>

</td>

</tr>

<tr>

<td>

<label>Password</label><br>

<input type="password" name="password" required>

</td>

</tr>

<tr>

<td class="buttons">

<button type="submit">Login</button>

<a href="<%= request.getContextPath() %>/owner/register" class="btn">Register Now</a>

</td>

</tr>

</table>

</form>

</div>

</div>

<div class="clear"></div>

</div>

<div class="modal">

<form action="<%= request.getContextPath() %>/quote" id="featured-vehicle-quote-form">

<table>

<tr>

<td>

<label>Journey From</label>

<select name="start"></select>

</td>

</tr>

<tr>

<td>

<label>Journey To</label>

<select name="end"></select>

</td>

</tr>

<tr>

<td>

<label>Date</label>

<input type="text" name="date" class="datepicker" required>

</td>

</tr>

<tr>

<td><button type="submit" class="btn">Search for date</button></td>

</tr>

</table>

</form>

</div>

<script>

$(function(){

$('.modal').dialog({

autoOpen: false

});

$('#banner-quote-form').validate();

$('#owner-login-form').validate();

$('#featured-vehicle-quote-form').validate();

$('.datepicker').datepicker({minDate: 0, dateFormat: 'yy-mm-dd'});

});

function quoteModal(vehicleId, vehicleModal){

$('.modal').attr('title', 'Quote Vehicle - '+vehicleModal);

$.post(

"<%= request.getContextPath() %>/ajax\_getVehicleCurrentLocation",

{'vehicle\_id': vehicleId},

function(response1){

var option1 = "<option>"+response1+"</option>";

$('.modal select[name="start"]').html(option1);

$.post(

"<%= request.getContextPath() %>/ajax\_getVehicleAvailableLocations",

{'vehicle\_id': vehicleId},

function (response2){

var locations = response2.split(',');

var option2 = "";

for(var i=0; i<locations.length; i++){

option2 += "<option>"+locations[i]+"</option>";

}

$('.modal select[name="end"]').html(option2);

}

);

}

);

$('.modal').dialog({

'title': "Quote Vehicle - "+vehicleModal

});

$('.modal').dialog('open');

}

</script>

<%@ include file="include/footer.jsp"%>  
  
**Note\*: Full source code is available in the compact disk provided along with this documentation.**

1. **Backup and Recovery**

The database management system used for this project is MySQL, an open source DBMS. Whole content of OVR is stored into MySQL Database which is very easy to Backup & Restore. Since, OVR is dynamic online software you won’t need to be worried of insertion of data into the database but should be aware of process of making a backup & use it later for restore.

* 1. **Backup of MySQL Database**

For backing up the MySQL Database you need to login to your web server where you’ll find “PhpMyAdmin”, which is also a web platform providing user friendly interface for managing MySQL Database.

Steps involved in making Backup:

1. Open PhpMyAdmin
2. Login using the user credentials.
3. Select the QMS database, named “db\_qms”, then Click on Export. Make sure that entire database is selected, not only one table.
4. Select “SQL” for output format, Check Mark on “Structure” & “Add AUTO\_INCREMENT” value. Check "Enclose table and field name with back quotes". Check "DATA", check use "Hexadecimal for Binary Field". Export type set to "INSERT".
5. Check "Save as file", use compression if you want. Then click "GO" to download the backup file.
   1. **Restoring a Backup of MySQL Database**

Steps involved in restoring a backup:

1. Open PhpMyAdmin
2. Login using the user credentials.
3. Click on “SQL” Tab
4. Then, on SQL page, uncheck the “Show Query here again”.
5. Browse to your backup of the database (e.g. db\_ovr.sql)
6. Click on “GO” to restore your database. If an error occurs retry dropping the current database & re-creating it.