ABSTRACT

ONLINE VEHICLE RENTAL SYSTEM

This project is being considered in order to reduce and totally eliminate loss of customers to competitors, and save the company from folding up. The current system is manual and it is time consuming. It is also cost ineffective, and average return is low and diminishing.

Currently, customers can call or walk-in in order to rent or reserve a vehicle. The staff of the company will check their file to see which vehicle is available for rental. The current system is error prone and customers are dissatisfied. The goal of this project is to automate vehicle rental and reservation so that customers do not need to walk-in or call in order to reserve a vehicle.

They can go online and reserve any kind of vehicle they want and that is available. Even when a customer chooses to walk-in, computers are available for him to go online and perform his reservation. When he choose to reserve by phone, any of the customer service representatives can help him reserve the vehicle speedily and issue him a reservation number.

The VRS will maintain the database of all vehicles the company has. It will also keep track of all vehicle reservation and return. Reports will be generated bi-weekly. Reports for the Accounts Manager will detail the cost incurred to maintain each vehicle and revenue accrued on each vehicle.

Reports for the Maintenance Manager will detail the present mileage of the car in order for him to take care of the vehicle servicing, and when each vehicle will be due for tag renewal. The Branch Manager’s report will detail total cost incurred and total revenue accrued, and the status of each vehicle so that he can decide whether to sell the vehicle or still keep it.

CHAPTER 2

INTRODUCTION

This project Online Vehicle Rental System is a web-based online system because it's easier for the customers to rent a car. This project Online Vehicle Rental System has been provided car history details, their engine and parts details, insurance registration and expiration details, car check in and check out details, car servicing details, payment details etc. This project also have to facility to check their customers and suppliers details and their payment mode and status details along with date and time. First time customers will have to create a profile if they are taking a car on rent and select the appropriate payment mode.

However customers are taking this service by visiting the office, they will get their id and password. Customers will have the facility to select any type of car, search car by their brand name. Upon selection of particular type customers will able to get their entire details like rent type, cost for taking a particular car, mileage details in kilometer an hour. This system can also help for customers to fill the basic information details like name, address, total number of family members who also travel through the car, number of days to take service, location to travel etc. The main aim of this project Online Vehicle Rental system project is to maintain records of cars.

Basically this system help Online Vehicle Rental shopper to make daily record and easy billing of customers and also help to keep maintain monthly revenues and help to grow business. This system work 24×7 because of it’s online existence. Customer can use this system from anywhere and anytime. Customers can book car service from any were in the world and take service when they visit that city.CHAPTER 2

CHAPTER 3

OBJECTIVE OF THE PROJECT AND MODULES

The main modules of the projects are Customer Module, which performs all the operations related to customer such as adding new customer, edit the existing customer, search customer and delete customer. Car module, which performs all the operations related to car such as adding new car, edit the existing car, details view car, search car and delete car. Booking module, which performs all the operations related to booking such as adding new booking, edit the existing booking, details view booking, search booking and delete booking. System User module, which performs all the operations related to system user such as adding new system user, edit the existing system user, details view system user, search system user and delete system user.

ADMIN MODULE

Administrator or admin has total control over the application. They can add managers of any kind, set up their profiles, and add, delete, or modify manager records. They can have an overview of all managers from different locations and directly communicate with them. Admin is required to log in to the system with a unique user id consisting of username and password.

CUSTOMER MODULE

The main objective of this module is provide all the functionality related to customer. It tracks all the information of the customer. We have developed all type of CRUD (Create, Read, Update and Delete) operations of the customer. This is a role based module where admin can perform each and every operations on data but the customer will be able to view only his/her data, so access level restrictions has also been implemented on the project. Students can download php projects with database free download for learning.

Features of Customer Module

- Admin can add new customer records
- Admin can see the list of customer details
- Only admin can edit and update the record of the customer
- Admin will be able to delete the records of the customer
- All customer forms are validated on client side using JavaScript

CAR MODULE

The main aim for developing this module is to manage the car. So all car will be managed by admin. It tracks all the information of the car. We have developed all type of CRUD (Create, Read, Update and Delete) operations of the car. We have many best php projects free download with source code and database.

Features of Car Module:

- Admin can manage the car
- Admin can edit/delete the car
- Admin can see the list of all car
- Customer can see his car

BOOKING MODULE

The main objective for developing this module is to manage the booking. This Booking module is an important module in this project Online Online Vehicle Rental System which has been developed on PHP and MYSQL. So all booking will be managed by admin.

Features of Booking Module:

- Admin can manage the booking
- Admin can edit/delete the booking
- Admin can see the list of all booking
- Customer can see his booking

CHAPHTER 3

SYSTEM SPECIFICATION Software specifications

Operating System	:	Windows XP, Windows7,8,9,10
User interface	:	php
Database	:	My SQL
Documentation Tool	:	Ms Office

Hardware specifications

Processor	**:** Standard processor with a speed of 1.6 GHz or more
RAM	**:** 1 GB RAM or more
	: 500 GB or more
Hard Disk	
Monitor	**:** 25-Inch Monitor
Keyboard	**:** Standard keyboard
Mouse	**:** Standard mouse

SOFTWARE SPECIFICATIONS

3.1.Php

PHP is a server-side scripting language designed primarily for web development but also used as a generalpurpose programming language. Originally created by Rasmus Lerdorf in 1994,the PHP reference implementation is now produced by The PHP Development Team. PHP originally stood for personal home page but it now stands for the recursive acronym PHP: Hypertext Preprocessor.

PHP code may be embedded into HTML or HTML5 code, or it can be used in combination with various web template systems, web content management systems and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface

PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications.

The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge.

The PHP language evolved without a written formal specification or standard until 2014, leaving the canonical PHP interpreter as a de facto standard. Since 2014 work has gone on to create a formal PHP specification.PHP development began in 1995 when Rasmus Lerdorf wrote several Common Gateway Interface (CGI) programs in C, which he used to maintain his personal homepage. He extended them to work with web forms and to communicate with databases, and called this implementation "Personal Home Page/Forms Interpreter" or PHP/FI.

PHP/FI could help to build simple, dynamic web applications. To accelerate bug reporting and to improve the code, Lerdorf initially announced the release of PHP/FI as "Personal Home Page Tools(PHPTools)version1.0"onthe Usenet discussiongroup comp.infosystems.www.authoring.cgi on June 8, 1995. This release already had the basic functionality that PHP has as of 2013. This included Perllike variables, form handling, and the ability to embed HTML. The syntax resembled that of Perl but was simpler, more limited and less consistent.

Lerdorf did not intend the early PHP to become a new programming language, but it grew organically, with Lerdorf noting in retrospect. A development team began to form and, after months of work and beta testing, officially released PHP/FI 2 in November 1997.

The fact that PHP lacked an original overall design but instead developed organically has led to inconsistent naming of functions and inconsistent ordering of their parameters. In some cases, the function names were chosen to match the lower-level libraries which PHP was "wrapping" ,while in some very early versions of PHP the length of the function names was used internally as a hash function, so names were chosen to improve the distribution of hash values.

Php 3 and 4

Zeev Suraski and Andi Gutmans rewrote the parser in 1997 and formed the base of PHP 3, changing the language's name to the recursive acronym PHP:Hypertext Preprocessor. Afterwards, public testing of PHP 3 began, and the official launch came in June 1998. Suraski and Gutmans then started a new rewrite of PHP's core, producing the Zend Engine in 1999. They also founded Zend Technologies in Ramat Gan, Israel. On May 22, 2000, PHP 4, powered by the Zend Engine 1.0, was released. As of August 2008 this branch reached version 4.4.9. PHP 4 is no longer under development nor will any security updates be released.

Php 5

On July 13, 2004, PHP 5 was released, powered by the new Zend Engine II. PHP 5 included new features such as improved support for object-oriented programming, the PHP Data Objects (PDO) extension (which defines a lightweight and consistent interface for accessing databases), and numerous performance enhancements. In 2008 PHP 5 became the only stable version under development. Late static binding had been missing from PHP and was added in version 5.3.

Many high-profile open-source projects ceased to support PHP 4 in new code as of February 5, 2008, because of the GoPHP5 initiative, provided by a consortium of PHP developers promoting the transition from PHP 4 to PHP 5. Over time, PHP interpreters became available on most existing 32-bit and 64bit operating systems, either by building them from the PHP source code, or by using pre-built binaries. For the PHP versions 5.3 and 5.4, the only available Microsoft Windows binary distributions were 32bit x86 builds, requiring Windows 32-bit compatibility mode while using Internet Information Services (IIS) on a 64-bit Windows platform. PHP version 5.5 made the 64-bit x86-64 builds available for Microsoft Windows.

Php 6 and Unicode

PHP has received criticism due to lacking native Unicode support at the core language level, instead only supporting byte strings. In 2005, a project headed by Andrei Zmievski was initiated to bring native Unicode support throughout PHP, by embedding the International Components for Unicode (ICU) library, and representing text strings as UTF-16 internally. Since this would cause major changes both to the internals

Since this would cause major changes both to the internals of the language and to user code, it was planned to release this as version 6.0 of the language, along with other major features then in development.

However, a shortage of developers who understood the necessary changes, and performance problems arising from conversion to and from UTF-16, which is rarely used in a web context, led to delays in the project. As a result, a PHP 5.3 release was created in 2009, with many non-Unicode features back-ported from PHP 6, notably namespaces. In March 2010, the project in its current form was officially abandoned, and a PHP 5.4 release was prepared containing most remaining non-Unicode features from PHP 6, such as traits and closure re-binding. Initial hopes were that a new plan would be formed for Unicode integration, but as of 2014 none have been adopted.

Php 7

During 2014 and 2015, a new major PHP version was developed, which was numbered PHP 7. The numbering of this version involved some debate. While the PHP 6 Unicode experiment had never been released, several articles and book titles referenced the PHP 6 name, which might have caused confusion if a new release were to reuse the name. After a vote, the name PHP 7 was chosen.The foundation of PHP 7 is a PHP branch that was originally dubbed PHP next generation (phpng). It was authored by Dmitry Stogov, Xinchen Hui and Nikita Popov, and aimed to optimize PHP performance by refactoring the Zend Engine to use more compact data structures with improved cache locality while retaining near-complete language compatibility. As of 14 July 2014, WordPress-based benchmarks, which served as the main benchmark suite for the phpng project, showed an almost 100% increase in performance. Changes from phpng are also expected to make it easier to improve performance in the future, as more compact data structures and other changes are seen as better suited for a successful migration to a just-in-time (JIT) compiler. Because of the significant changes, the reworked Zend Engine is called Zend Engine 3, succeeding Zend Engine 2 used in PHP 5. Because of major internal changes in phpng, it must receive a new major version number of PHP, rather than a minor PHP 5 release, according to PHP's release process. Major versions of PHP are allowed to break backward-compatibility of code and therefore PHP 7 presented an opportunity for other improvements beyond phpng that require backward-compatibility breaks, including wider use of exceptions, reworking variable syntax to be more consistent and complete, and the deprecation or removal of various legacy features. PHP 7 also introduced new language features, including return type declarations for functions, which complement the existing parameter type declarations, and support for the scalar types (integer, float, string, and boolean) in parameter and return type declarations.

Data Types

PHP stores integers in a platform-dependent range, either a 64-bit or 32-bit signed integer equivalent to the C-language long type. Unsigned integers are converted to signed values in certain situations; this behavior is different from that of other programming languages. Integer variables can be assigned using decimal (positive and negative), octal, hexadecimal, and binary notations. Floating point numbers are also stored in a platform-specific range. They can be specified using floating point notation, or two forms of scientific notation. PHP has a native Boolean type that is similar to the native Boolean types in Java and C++. Using the Boolean type conversion rules, non-zero values are interpreted as true and zero as false, as in Perl and C++.The null data type represents a variable that has no value; NULL is the only allowed value for this data type. Variables of the "resource" type represent references to resources from external sources. These are typically created by functions from a particular extension, and can only be processed by functions from the same extension; examples include file, image, and database resources. Arrays can contain elements of any type that PHP can handle, including resources, objects, and other arrays. Order is preserved in lists of values and in hashes with both keys and values, and the two can be intermingled. PHP also supports strings, which can be used with single quotes, double quotes, now doc or here doc syntax. The Standard PHP Library (SPL) attempts to solve standard problems and implements efficient data access interfaces and classes.

Functions

PHP defines a large array of functions in the core language and many are also available in various extensions, these functions are well documented in the online PHP documentation. However, the built-in library has a wide variety of naming conventions and associated inconsistencies, as described under history above.

In lieu of function pointers, functions in PHP can be referenced by a string containing their name. In this manner, normal PHP functions can be used, for example, as callbacks or within function tables. Userdefined functions may be created at any time without being prototyped. Functions may be defined inside code blocks, permitting a run-time decision as to whether or not a function should be defined. There is a function_exists function that determines whether a function with a given name has already been defined. Function calls must use parentheses, with the exception of zero-argument class constructor functions called with the PHP operator new, in which case parentheses are optional.

Until PHP 5.3, support for anonymous functions and closures did not exist in PHP. While creat_function() exists since PHP 4.0.1, it is merely a thin wrapper around eval() that allows normal PHP functions to be created during program execution. PHP 5.3 added syntax to define an anonymous function or "closure" which can capture variables from the surrounding scope. In the example above, getAdder() function creates a closure using passed argument $x (the keyword use imports a variable from the lexical context), which takes an additional argument $y, and returns the created closure to the caller. Such a function is a first-class object, meaning that it can be stored in a variable, passed as a parameter to other functions, etc. Unusually for a dynamically typed language, PHP supports type declarations on function parameters, which are enforced at runtime. This has been supported for classes and interfaces since PHP 5.0, for arrays since PHP 5.1, for "callables" since PHP 5.4, and scalar (integer, float, string and Boolean) types since PHP 7.0. PHP 7.0 also has type declarations for function return types, expressed by placing the type name after the list of parameters, preceded by a colon. For example, the getAdder function from the earlier example could be annotated with types like so in PHP 7.

Html

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects, such as interactive forms, may be embedded into the rendered page. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as <img/>and<input/> introduce content into the page directly. Others such as ,<p>…</p>

surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.HTML can embed programs written in a scripting language such as JavaScript which affect the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), maintainer of both the HTML and the CSS standards, has encouraged the use of CSS over explicit presentational HTML.

MySQL

MySQL is an open-source relational database management system (RDBMS). Its name is a combination of "My", the name of co-founder Michael Wideners' daughter, and "SQL", the abbreviation for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation.

MySQL, pronounced either "My S-Q-L" or "My Sequel," is an open source relational database management system. It is based on the structure query language (SQL), which is used for adding, removing, and modifying information in the database. Standard SQL commands, such as ADD, DROP, INSERT, and UPDATE can be used with MySQL.

MySQL can be used for a variety of applications, but is most commonly found on Web servers. A website that uses MySQL may include Web pages that access information from a database. These pages are often referred to as "dynamic," meaning the content of each page is generated from a database as the page loads. Websites that use dynamic Web pages are often referred to as database-driven websites.

Many database-driven websites that use MySQL also use a Web scripting language like PHP to access information from the database. MySQL commands can be incorporated into the PHP code, allowing part or all of a Web page to be generated from database information. Because both MySQL and PHP are both open source (meaning they are free to download and use), the PHP/MySQL combination has become a popular choice for database-driven websites.

CHAPTER

SYSTEM ANALYSIS

Requirement analysis for web applications encompasses three major tasks: formulation, requirements gathering and analysis modeling. During formulation, the basic motivation and goals for the web application are identified, and the categories of users are defined. In the requirements gathering phase, the content and functional requirements are listed and interaction scenarios written from end-user's point-of-view are developed. This intent is to establish a basic understanding of why the web application is built, who will use it, and what problems it will solve for its users.

2.1 Existing System

Cool cab Service is an innovative thought to simplify the Transportation problems of Employees of an organization. In the present System, Organization do maintain a person for the allocating and proper functioning of transportation .The Person appointed needs to look after the assigning and movement of cabs. Authorised person maintains the transportation details in papers, which is a tedious task if any updation or changes need to be done.

- Details are stored in Papers.
- Maintenance is a huge problem.
- Updation, changes in details is a tedious task.
- Performance is not achieved up to the requirements.

2.2 Proposed System

In the Previous System, Details are Stored Manually in papers, to share the details between employees was a Financial drawback. Updation in the details is a tedious task.
But a new system was proposed to overcome the above drawbacks.

Functionalities and advantages of proposed system are:

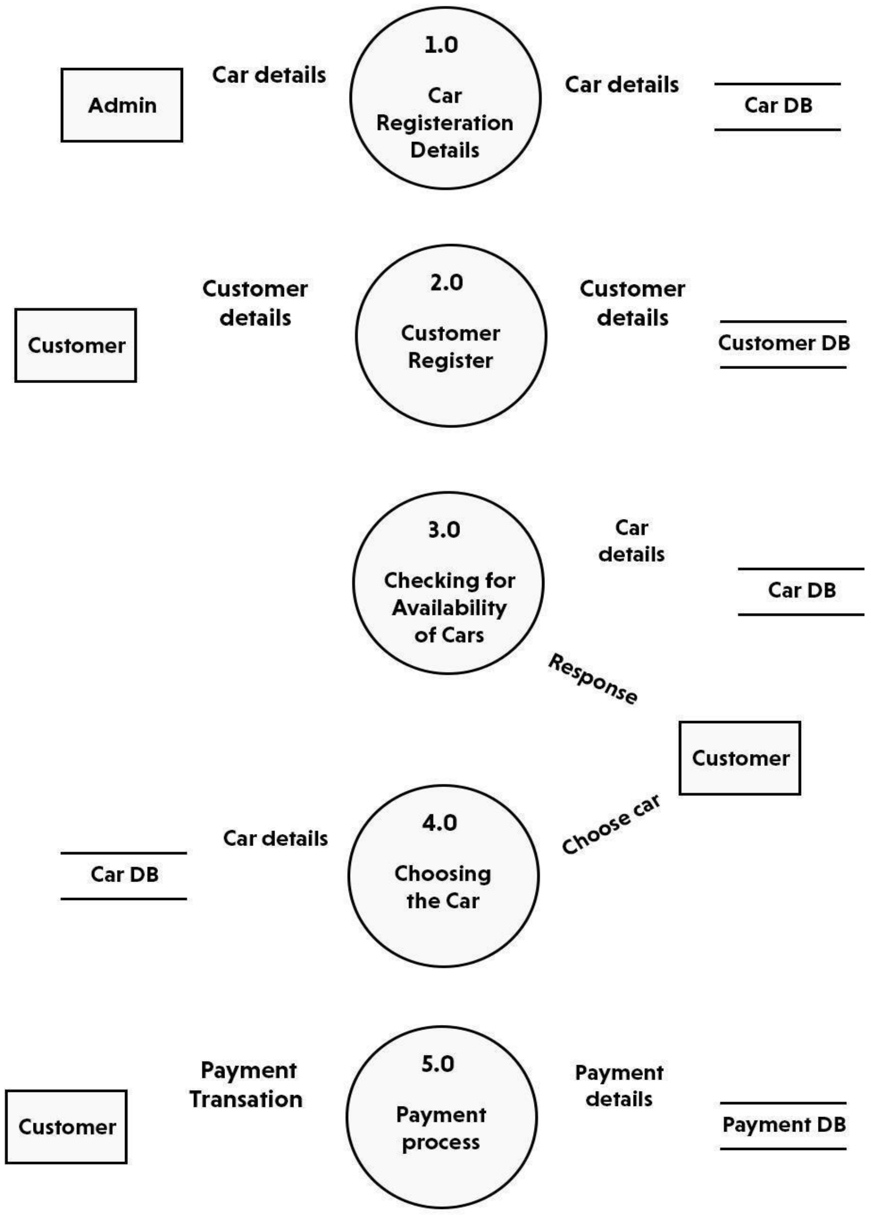
- Data is Centralized which has overcome the Sharing problem in previous system.
- As data is Maintained electronically, it's easy for a person to update the details, which has overcome the tedious updation in previous system.
- Maintenance is easy and performance is good.
- Mainly the system has automated the Transportation Process.

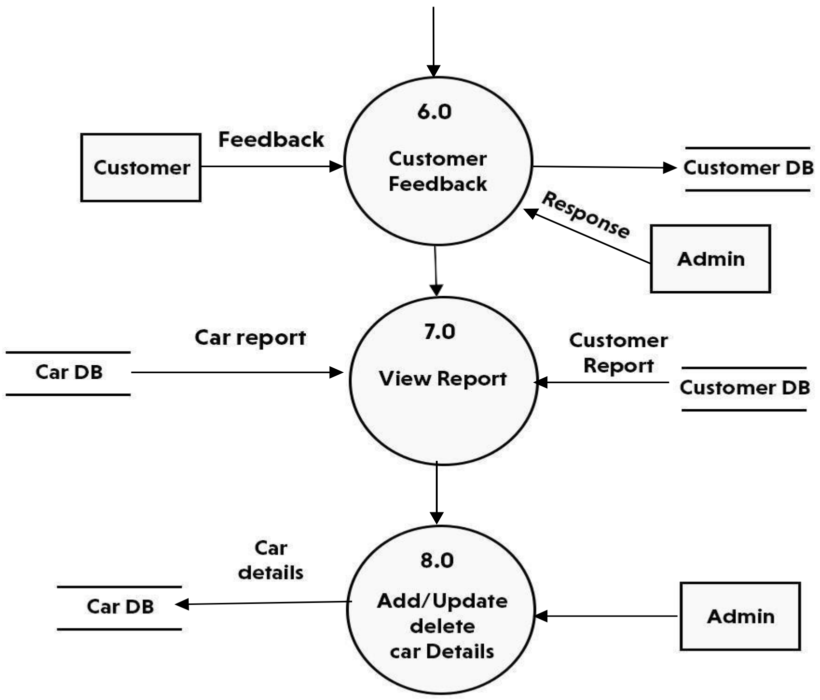
CHAPTER

SYSTEM DESIGN AND DEVLOPMENT

Data Flow Diagram (DFD)

A Data Flow Diagram (DFD) is a graphical representation that depicts the information flow and the transforms that are applied as data moves from input to output.





DFD of Online Online Vehicle Rental System

In this diagram, Customer and Online Vehicle Rental Company are the two entity sets. Functions of

Customer:

- New Registration
- Login Request
- Registration Confirmation by the System
- Reserve Car
- Car Issued by the System
- Email received for Reserved Car Functions of Online Vehicle Rental Company:

- Add Customer

- Send E-Mails for Reserved Car
- View Transaction reports

CHAPTER 6

Use-case diagrams

Actor and Use Case Description

Actor and use case description shows the detail description of interaction between the actors and their use cases. The description enables to have a proper understanding of how actor interacts with the system through their use cases.

Actor	Use Case	Use Case Description
Customer	Register as member	This use case describes the activities of the customer to register online and become a member. Customer's details are required as part of the registration. Login detail is automatically sent to the customer after successful registration.
	Make reservation	This use case enable customer to search and make reservation. Non-register customer will be directed to register before their reservation can be confirmed. Notification is automatically send to the customer after the task is completed.
	Return car	This use case describes the event of customer returning the car borrowed, the use case extends "process rental" use case from the staff actor.
	Give feedback	This use case is used by the customer to provide feedbacks/comment to the company; a confirmation notification will be send to the customer once a feedback has been submitted.
Staff	Add new car	This use case is used by the staff to add new car to the company's fleet database. Staff will need to login to activate this use case.

	Update car details	This use case is used by the staff to edit and modify car details whenever there is new renewal (Insurance, road tax). It allows the company to keep up-to-date record of their fleet.
	Reply to customer's feedback	This use case describes the event by which staff sends reply to customer's earlier feedback. It depends on `give feedback' use case from the customer.
	Process rental	This use case described the event by which staff updates the system when customer pick up or when returning car.
Admin	Add new staff	This use case describes the event by which Admin add new staff detail to the company's staff database. It is invoke whenever a new staff join the company.
	View report	This use case is used by the Admin to view transaction report.

Table 1.1 Actors and Use Case Description

ER DIAGRAM

This ER (Entity Relationship) Diagram represents the model of Vehicle Management System Entity. The entity-relationship diagram of Vehicle Management System shows all the visual instrument of database tables and the relations between Driver, Vehicle History, Vehicle, Vehicle Type etc. It used structure data and to define the relationships between structured data groups of Vehicle Management System functionalities. The main entities of the Vehicle Management System are Vehicle, Driver, Fuel, Vehicle History, Booking and Vehicle Type.

Vehicle Management System entities and their attributes :

- **Vehicle Entity :** Attributes of Vehicle are vehicle_id, vehicle_customer_id, vehicle_number, vehicle_type, vehicle_description
- **Driver Entity :** Attributes of Driver are driver_id, driver_name, driver_mobile, driver_email, driver_username, driver_password, driver_address Fuel Entity : Attributes of Fuel are fuel_id, fuel_vehicle_id, fuel_name, fuel_expences, fuel_type, fuel_description
- **Vehicle History Entity :** Attributes of Vehicle History are vehicle_history_id, vehicle_history_name, vehicle_history_type, vehicle_history_description
- **Booking Entity:** Attributes of Booking are booking_id, booking_vehicle_id, booking_title, booking_type, booking_date, booking_description
- **Vehicle Type Entity :** Attributes of Vehicle Type are vehicle_type_id, vehicle_type_customer_id, vehicle_type_number, vehicle_type_description

Description of Vehicle Management System Database :

- The details of Vehicle is store into the Vehicle tables respective with all tables
- Each entity (Vehicle Type, Fuel, Booking, Driver, Vehicle) contains primary key and unique keys.
- The entity Fuel, Booking has binded with Vehicle, Driver entities with foreign key
- There is one-to-one and one-to-many relationships available between Booking, Vehicle History, Vehicle Type, Vehicle
- All the entities Vehicle, Booking, Fuel, Vehicle Type are normalized and reduce duplicacy of records
- We have implemented indexing on each tables of vehicle management system tables for fast Query execution

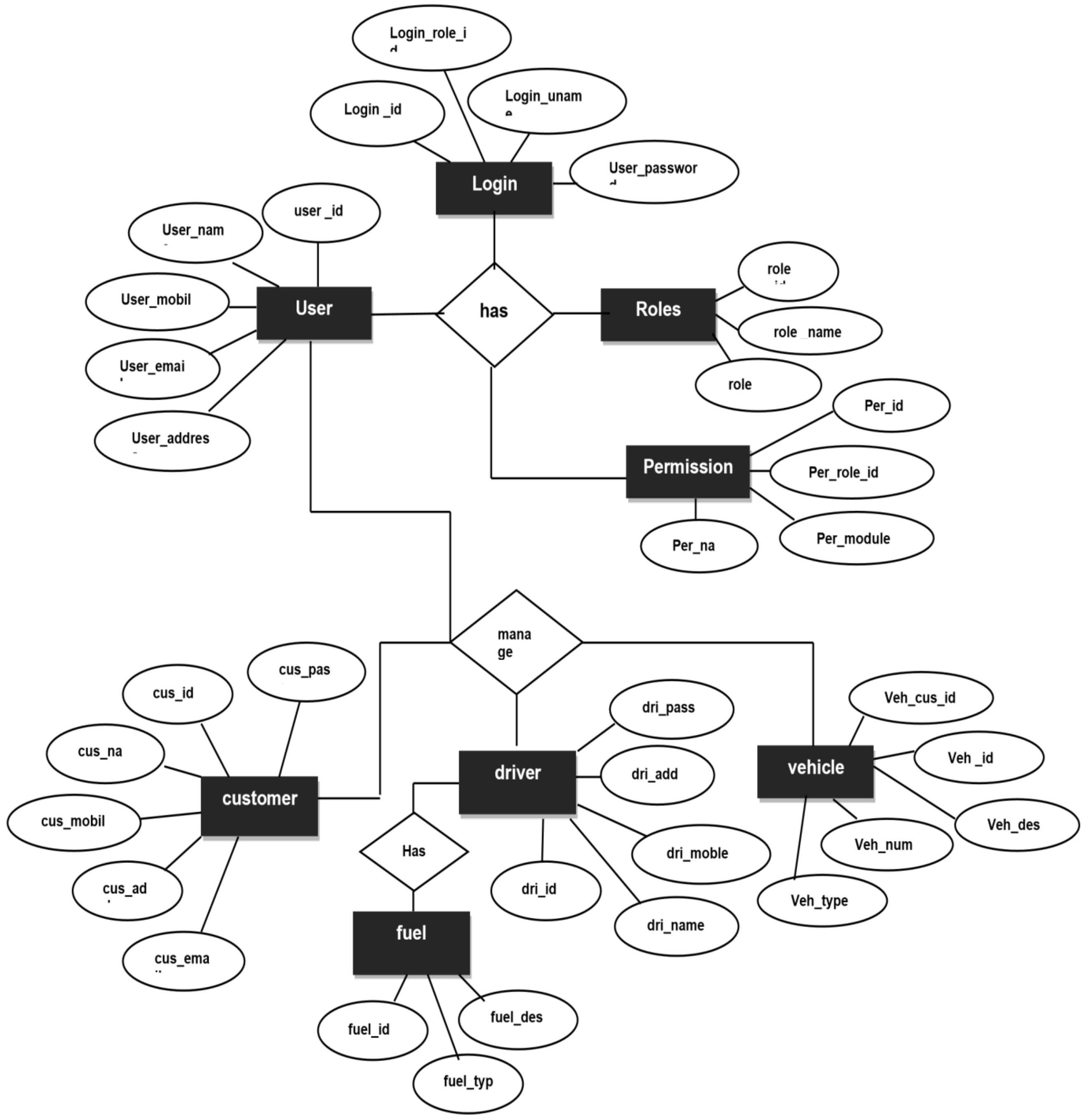


Fig 1.2 **ER Diagram for vehicle management system**

CHAPTER 7

TABLE STRUCTURE

Column	Type	Description	Constraint
id	int(11)	Admin id	PRIMARY KEY
UserName	varchar(100)	User Name	NULL
Password	varchar(100)	Password	NULL
updationDate	timestamp	Create Date	CURRENT_TIMESTAMP

Table B.1:admin (Primary Key : id)

Column	Type	Description	Constraint
id	int(11)	Booking Id	PRIMARY KEY
userEmail	varchar(100)	Email	NULL
VehicleId	int(11)	Vehicle Id	NULL
FromDate	date	From Date	NULL
ToDate	date	To Date	NULL
message	varchar(255)	Message	NULL
Status	int(11)	Status	NULL
PostingDate	timestamp	Posting Date	CURRENT_TIMESTAMP

Table B.2: tblbooking (Primary Key : id)

Column	Type	Description	Constraint
id	int(11)	Brand Id	PRIMARY KEY
BrandName	varchar(120)	Brand name	NULL
CreationDate	timestamp	Creation Date	CURRENT_TIMESTAMP
UpdationDate	timestamp	Updation Date	CURRENT_TIMESTAMP

Table B.3: tblbrands (Primary Key : id)

Column	Type	Description	Constraint
id	int(11)	Contact Id	PRIMARY KEY
Address	tinytext	Address	NULL
EmailId	varchar(255)	Email id	NULL
ContactNo	char(11)	Contact number	NULL

Table B.4: tblcontactusinfo (Primary Key : id)

Column	Type	Description	Constraint
id	int(11)	Query Id	PRIMARY KEY
name	varchar(100)	Query name	NULL
EmailId	varchar(120)	Email Id	NULL
ContactNumber	char(11)	Contact Number	NULL
Message	longtext	Message	NULL
PostingDate	timestamp	Posting Date	CURRENT_TIMESTAMP

status	int(11)	Status	NULL

Table B.5: tblcontactusquery (Primary Key : id)

Column	Type	Description	Constraint
id	int(11)	Page Id	PRIMARY KEY
PageName	varchar(255)	Page Name	NULL
type	varchar(255)	Page Type	NULL
detail	longtext	Details	NULL

Table B.6: tblpages (Primary Key : id)

Column	Type	Description	Constraint
id	int(11)	Subscribe Id	PRIMARY KEY
SubscriberEmail	varchar(120)	Subscribe Email	NULL
PostingDate	timestamp	Posting Date	CURRENT_TIMESTAMP

Table B.7: tblsubscribers (Primary Key : id)

Column	Type	Description	Constraint
id	int(11)	Testimonial Id	PRIMARY KEY
UserEmail	varchar(100)	User Email	NULL
Testimonial	mediumtext	Testimonial	NULL
PostingDate	timestamp	Posting Date	CURRENT_TIMESTAMP

status	int(11)	Status	NULL

Table B.8: tbltestimonial (Primary Key : id)

Column	Type	Description	Constraint
id	int(11)	User Id	PRIMARY KEY
FullName	varchar(120)	Full Name	NULL
EmailId	varchar(100)	Email Id	NULL
Password	varchar(100)	Password	NULL
ContactNo	char(11)	Contact No	NULL
dob	varchar(100)	Date Of Birth	NULL
Address	varchar(255)	Address	NULL
City	varchar(100)	City	NULL
Country	varchar(100)	Country	NULL
RegDate	timestamp	Register Date	CURRENT_TIMESTAMP
UpdationDate	timestamp	Updation Date	CURRENT_TIMESTAMP

Table B.8: tblusers (Primary Key : id)

CHAPTER 8

INPUT DESIGN

In an information system, input is the raw data that is processed to produce output. During the input design, the developers must consider the input devices such as PC, MICR, OMR, etc.

Therefore, the quality of system input determines the quality of system output. Well designed input forms and screens have following properties −

- It should serve specific purpose effectively such as storing, recording, and retrieving the information.
- It ensures proper completion with accuracy.
- It should be easy to fill and straightforward.
- It should focus on user’s attention, consistency, and simplicity.
- All these objectives are obtained using the knowledge of basic design principles regarding − ○ What are the inputs needed for the system?

○ How end users respond to different elements of forms and screens.

Objectives for Input Design

The objectives of input design are − □ To design

data entry and input procedures

- To reduce input volume
- To design source documents for data capture or devise other data capture methods
- To design input data records, data entry screens, user interface screens, etc.
- To use validation checks and develop effective input controls.

OUTPUT DESIGN

The design of output is the most important task of any system. During output design, developers identify the type of outputs needed, and consider the necessary output controls and prototype report layouts.

Objectives of Output Design

The objectives of input design are −

- To develop output design that serves the intended purpose and eliminates the production of unwanted output.
- To develop the output design that meets the end users requirements.
- To deliver the appropriate quantity of output.
- To form the output in appropriate format and direct it to the right person.
- To make the output available on time for making good decisions.

Let us now go through various types of outputs −

External Outputs

Manufacturers create and design external outputs for printers. External outputs enable the system to leave the trigger actions on the part of their recipients or confirm actions to their recipients.

Some of the external outputs are designed as turnaround outputs, which are implemented as a form and reenter the system as an input.

Internal outputs

Internal outputs are present inside the system, and used by end-users and managers. They support the management in decision making and reporting.

There are three types of reports produced by management information −

- **Detailed Reports** − They contain present information which has almost no filtering or restriction generated to assist management planning and control.
- **Summary Reports** − They contain trends and potential problems which are categorized and summarized that are generated for managers who do not want details.
- **Exception Reports** − They contain exceptions, filtered data to some condition or standard before presenting it to the manager, as information.

DATABASE DESIGN

Database Design is a collection of processes that facilitate the designing, development, implementation and maintenance of enterprise data management systems. Properly designed database are easy to maintain, improves data consistency and are cost effective in terms of disk storage space. The database designer decides how the data elements correlate and what data must be stored.

The main objectives of database designing are to produce logical and physical designs models of the proposed database system.

The logical model concentrates on the data requirements and the data to be stored independent of physical considerations. It does not concern itself with how the data will be stored or where it will be stored physically.

The physical data design model involves translating the logical design of the database onto physical media using hardware resources and software systems such as database management systems (DBMS).

CHAPTER 9

INPUT DESIGN

In an information system, input is the raw data that is processed to produce output. During the input design, the developers must consider the input devices such as PC, MICR, OMR, etc.

Therefore, the quality of system input determines the quality of system output. Well designed input forms and screens have following properties −

- It should serve specific purpose effectively such as storing, recording, and retrieving the information.
- It ensures proper completion with accuracy.
- It should be easy to fill and straightforward.

− ○ What are the

- It should focus on user's attention, consistency, and simplicity.
- All these objectives are obtained using the knowledge of basic design principles regarding inputs needed for the system?

○ How end users respond to different elements of forms and screens.

Objectives for Input Design

The objectives of input design are − To design

data entry and input procedures

- To reduce input volume
- To design source documents for data capture or devise other data capture methods
- To design input data records, data entry screens, user interface screens, etc.
- To use validation checks and develop effective input controls.

OUTPUT DESIGN

The design of output is the most important task of any system. During output design, developers identify the type of outputs needed, and consider the necessary output controls and prototype report layouts.

Objectives of Output Design

The objectives of input design are −

- To develop output design that serves the intended purpose and eliminates the production of unwanted output.
- To develop the output design that meets the end users requirements.
- To deliver the appropriate quantity of output.
- To form the output in appropriate format and direct it to the right person.
- To make the output available on time for making good decisions.

Let us now go through various types of outputs −

External Outputs

Manufacturers create and design external outputs for printers. External outputs enable the system to leave the trigger actions on the part of their recipients or confirm actions to their recipients.

Some of the external outputs are designed as turnaround outputs, which are implemented as a form and reenter the system as an input.

Internal outputs

Internal outputs are present inside the system, and used by end-users and managers. They support the management in decision making and reporting.

There are three types of reports produced by management information −

- **Detailed Reports** − They contain present information which has almost no filtering or restriction generated to assist management planning and control.
- **Summary Reports** − They contain trends and potential problems which are categorized and summarized that are generated for managers who do not want details.
- **Exception Reports** − They contain exceptions, filtered data to some condition or standard before presenting it to the manager, as information.

DATABASE DESIGN

Database Design is a collection of processes that facilitate the designing, development, implementation and maintenance of enterprise data management systems. Properly designed database are easy to maintain, improves data consistency and are cost effective in terms of disk storage space. The database designer decides how the data elements correlate and what data must be stored.

The main objectives of database designing are to produce logical and physical designs models of the proposed database system.

The logical model concentrates on the data requirements and the data to be stored independent of physical considerations. It does not concern itself with how the data will be stored or where it will be stored physically.

The physical data design model involves translating the logical design of the database onto physical media using hardware resources and software systems such as database management systems (DBMS).

SYSTEM IMPLEMENTATION

Detailed Design of Implementation

This phase of the systems development life cycle refines hardware and software specifications, establishes programming plans, trains users and implements extensive testing procedures, to evaluate design and operating specifications and/or provide the basis for further modification.

Technical Design

This activity builds upon specifications produced during new system design, adding detailed technical specifications and documentation.

Test Specifications and Planning

This activity prepares detailed test specifications for individual modules and programs, job streams, subsystems, and for the system as a whole.

Programming and Testing

This activity encompasses actual development, writing, and testing of program units or modules.

User Training

This activity encompasses writing user procedure manuals, preparation of user training materials, conducting training programs, and testing procedures.

Acceptance Test

A final procedural review to demonstrate a system and secure user approval before a system becomes operational.

Installation Phase

In this phase the new Computerized system is installed, the conversion to new procedures is fully implemented, and the potential of the new system is explored.

System Installation

The process of starting the actual use of a system and training user personnel in its operation.

Review Phase

This phase evaluates the successes and failures during a systems development project, and to measure the results of a new Computerized Tran system in terms of benefits and savings projected at the start of the project.

Development Recap

A review of a project immediately after completion to find successes and potential problems in future work.

Post-Implementation Review

A review, conducted after a new system has been in operation for some time, to evaluate actual system performance against original expectations and projections for cost-benefit improvements. Also identifies maintenance projects to enhance or improve the system.

THE STEPS IN THE SOFTWARE TESTING

The steps involved during Unit testing are as follows:

- ❖ Preparation of the test cases.
- ❖ Preparation of the possible test data with all the validation checks.
- ❖ Complete code review of the module.
- ❖ Actual testing done manually.
- ❖ Modifications done for the errors found during testing.
- ❖ Prepared the test result scripts.

SCOPE OF FEATURE ENHANCEMENT

This project traverses a lot of areas ranging from business concept to computing field, and required to perform several researches to be able to achieve the project objectives. The area covers include:

- Online Vehicle Rental industry: This includes study on how the Online Vehicle Rental business is being done, process involved and opportunity that exist for improvement.
- PHP Technology used for the development of the application.

- General customers as well as the company’s staff will be able to use the system effectively.

Web-platform means that the system will be available for access 24/7 except when there is a temporary server issue which is expected to be minimal.

CHAPTER 10

CONCLUSION

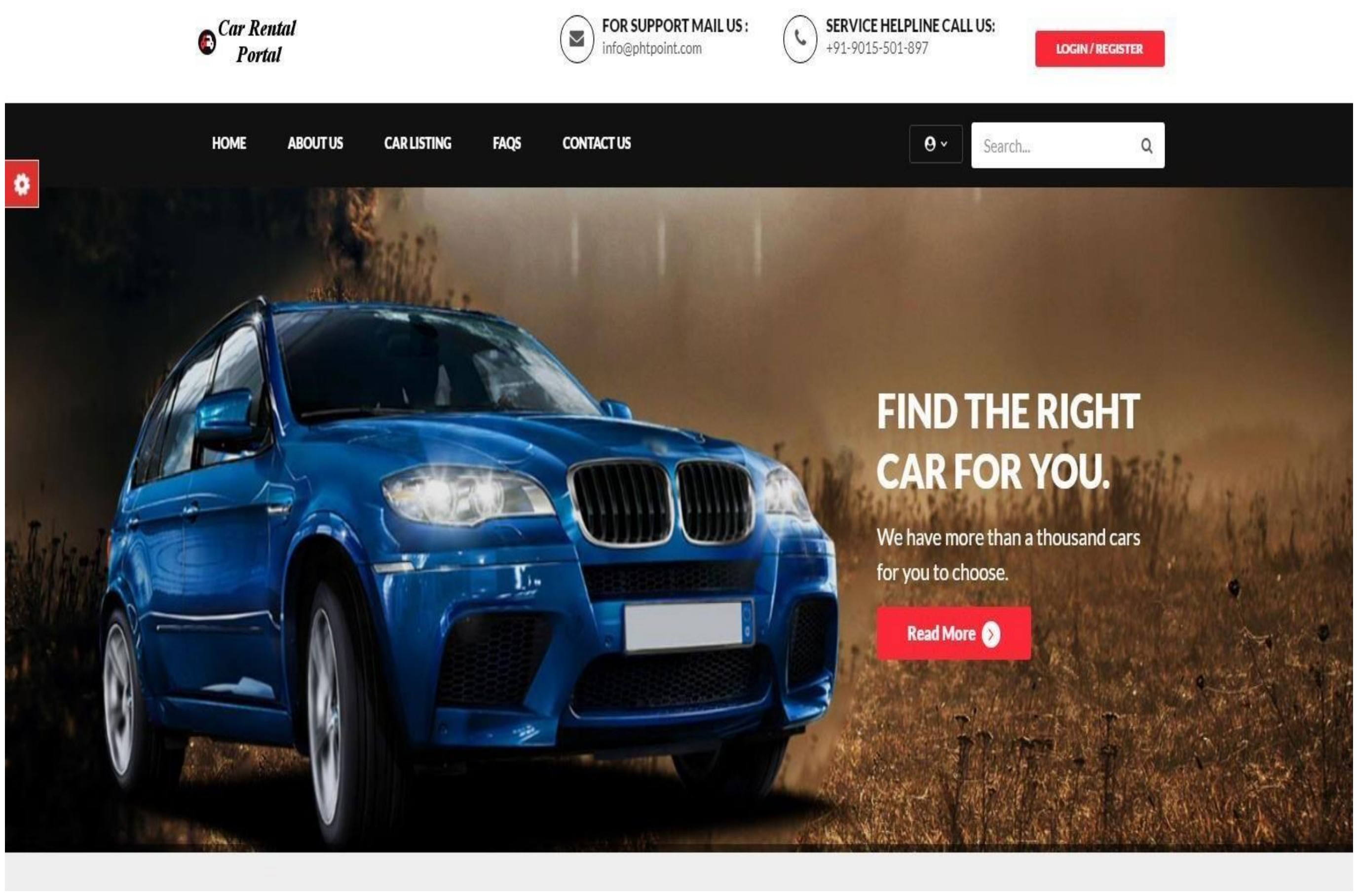
Online Vehicle Rental business has emerged with a new goodies compared to the past experience where every activity concerning Online Vehicle Rental business is limited to a physical location only. Even though the physical location has not been totally eradicated; the nature of functions and how these functions are achieved has been reshaped by the power of internet. Nowadays, customers can reserve cars online, rent car online, and have the car brought to their door step once the customer is a registered member or go to the office to pick the car.

The web based Online Vehicle Rental system has offered an advantage to both customers as well as Online

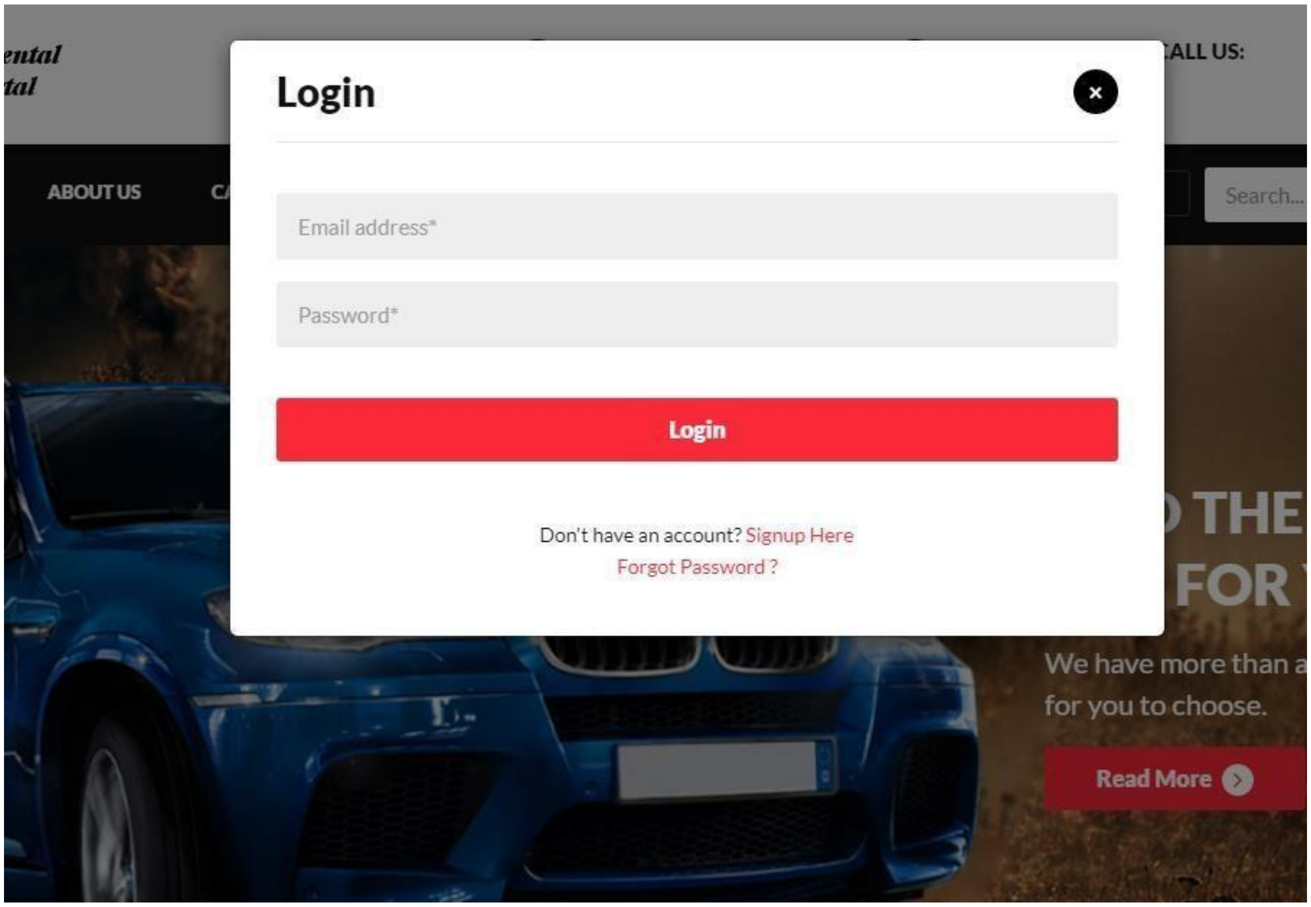
Vehicle Rental Company to efficiently and effectively manage the business and satisfies customers' need at the click of a button.

CHAPTER 11

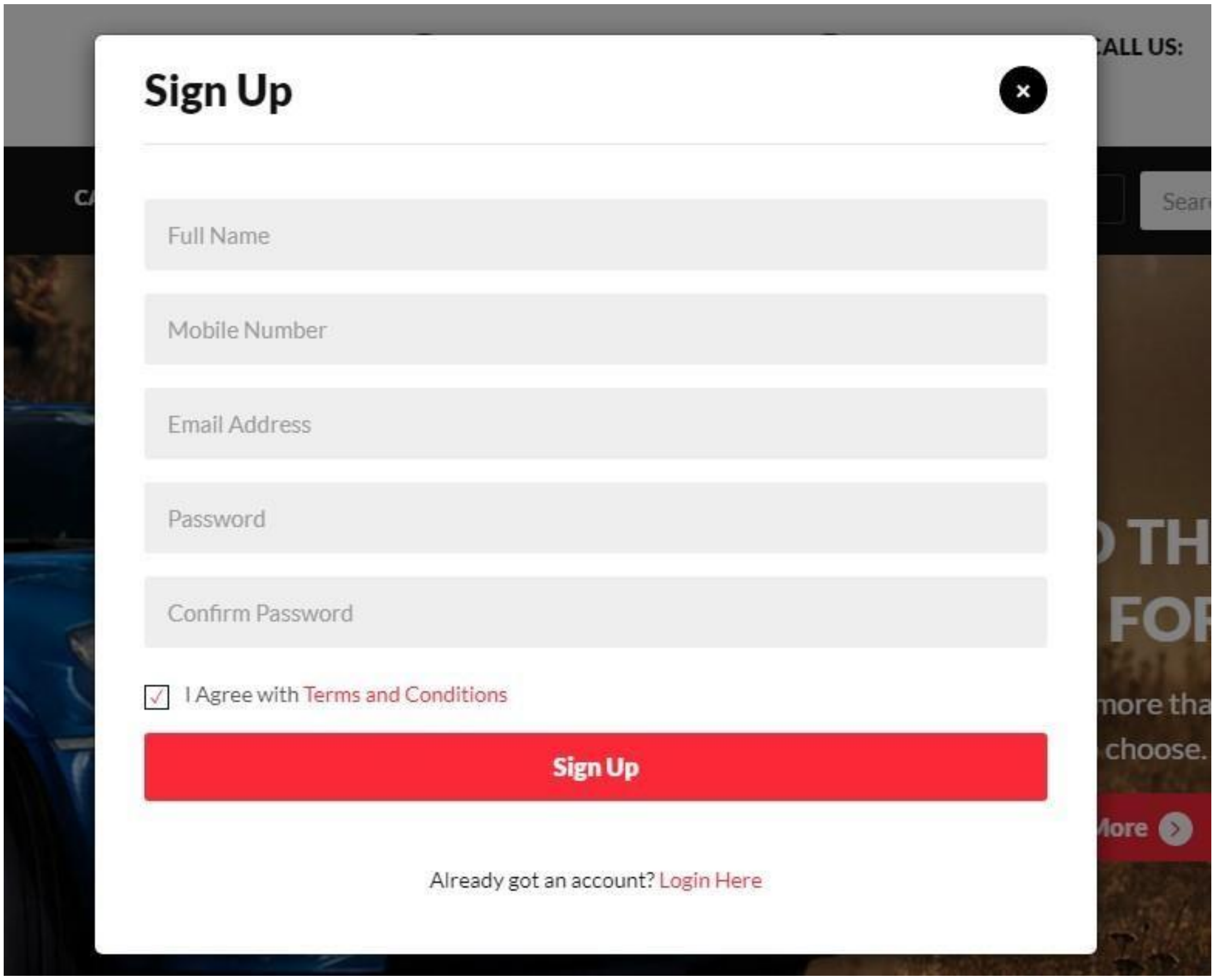
INPUT AND OUTPUT SCREENS



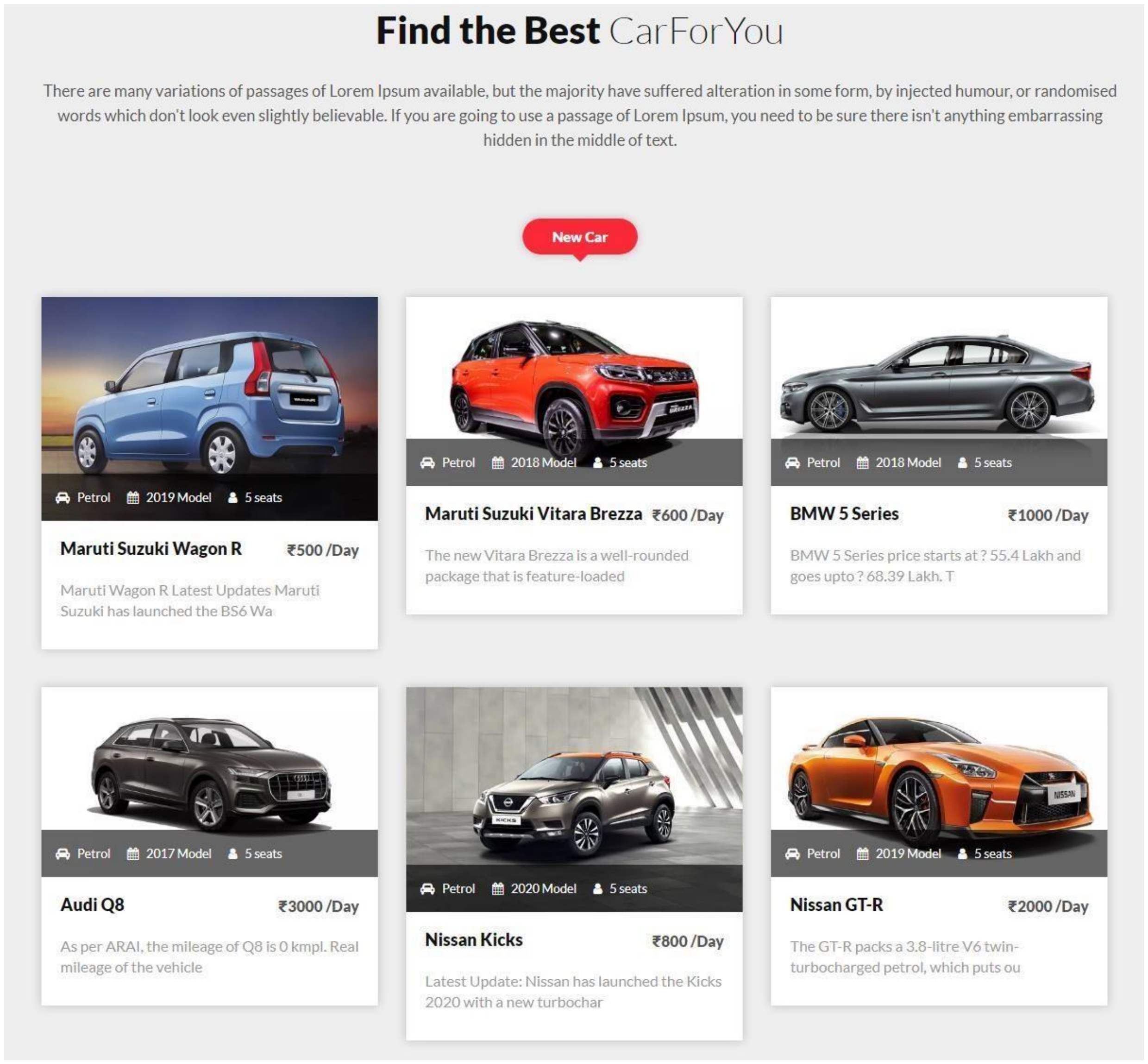
SNAPSHOT 1 : HOMEPAGE



SNAPSHOT 2: USER LOGIN



SNAPSHOT 3: USER REGISTER



Find the Best CarForYou

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the middle of text.

Petrol 2019 Model 5 seats

Maruti Suzuki Wagon R ₹500 /Day

Maruti Wagon R Latest Updates Maruti Suzuki has launched the BS6 Wa

Petrol 2018 Model 5 seats

Maruti Suzuki Vitara Brezza ₹600 /Day

The new Vitara Brezza is a well-rounded package that is feature-loaded

Petrol 2018 Model 5 seats

BMW 5 Series ₹1000 /Day

BMW 5 Series price starts at ? 55.4 Lakh and goes upto ? 68.39 Lakh. T

Petrol 2017 Model 5 seats

Audi Q8 ₹3000 /Day

As per ARAI, the mileage of Q8 is 0 kmpl. Real mileage of the vehicle

Petrol 2020 Model 5 seats

Nissan Kicks ₹800 /Day

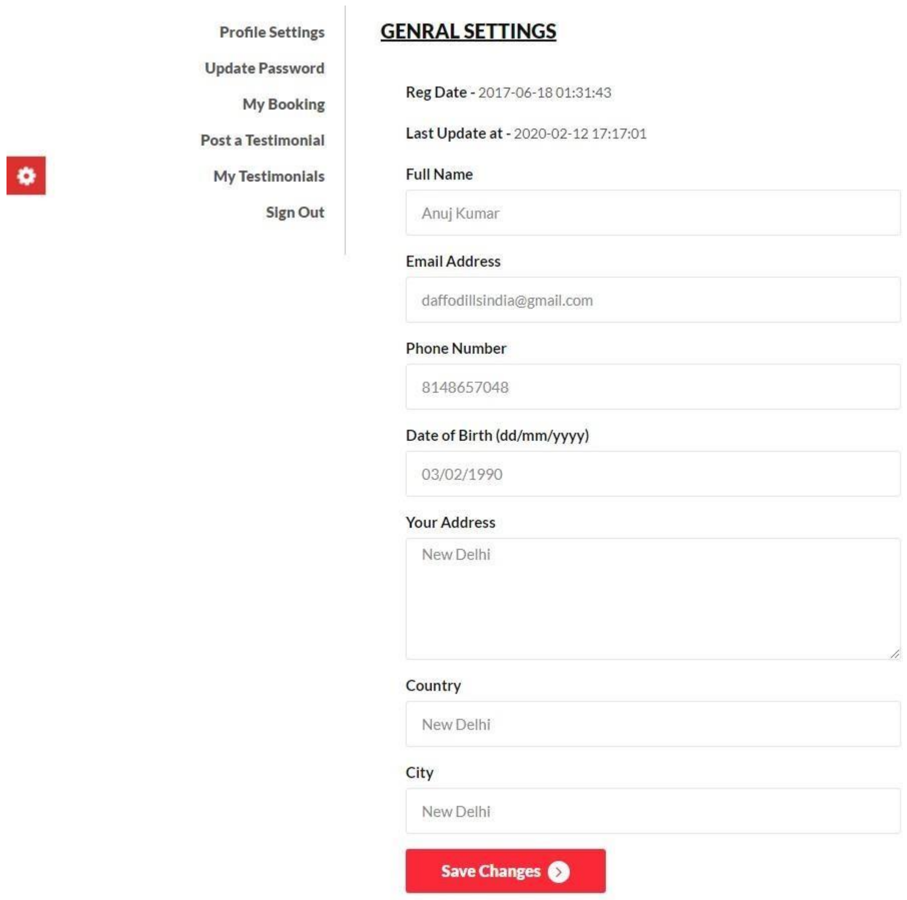
Latest Update: Nissan has launched the Kicks 2020 with a new turbochar

Petrol 2019 Model 5 seats

Nissan GT-R ₹2000 /Day

The GT-R packs a 3.8-litre V6 twin-turbocharged petrol, which puts ou

SNAPSHOT 4: CAR LIST



GENRAL SETTINGS

Reg Date - 2017-06-18 01:31:43

Last Update at - 2020-02-12 17:17:01

Full Name: Anuj Kumar

Email Address: daffodillsindia@gmail.com

Phone Number: 8148657048

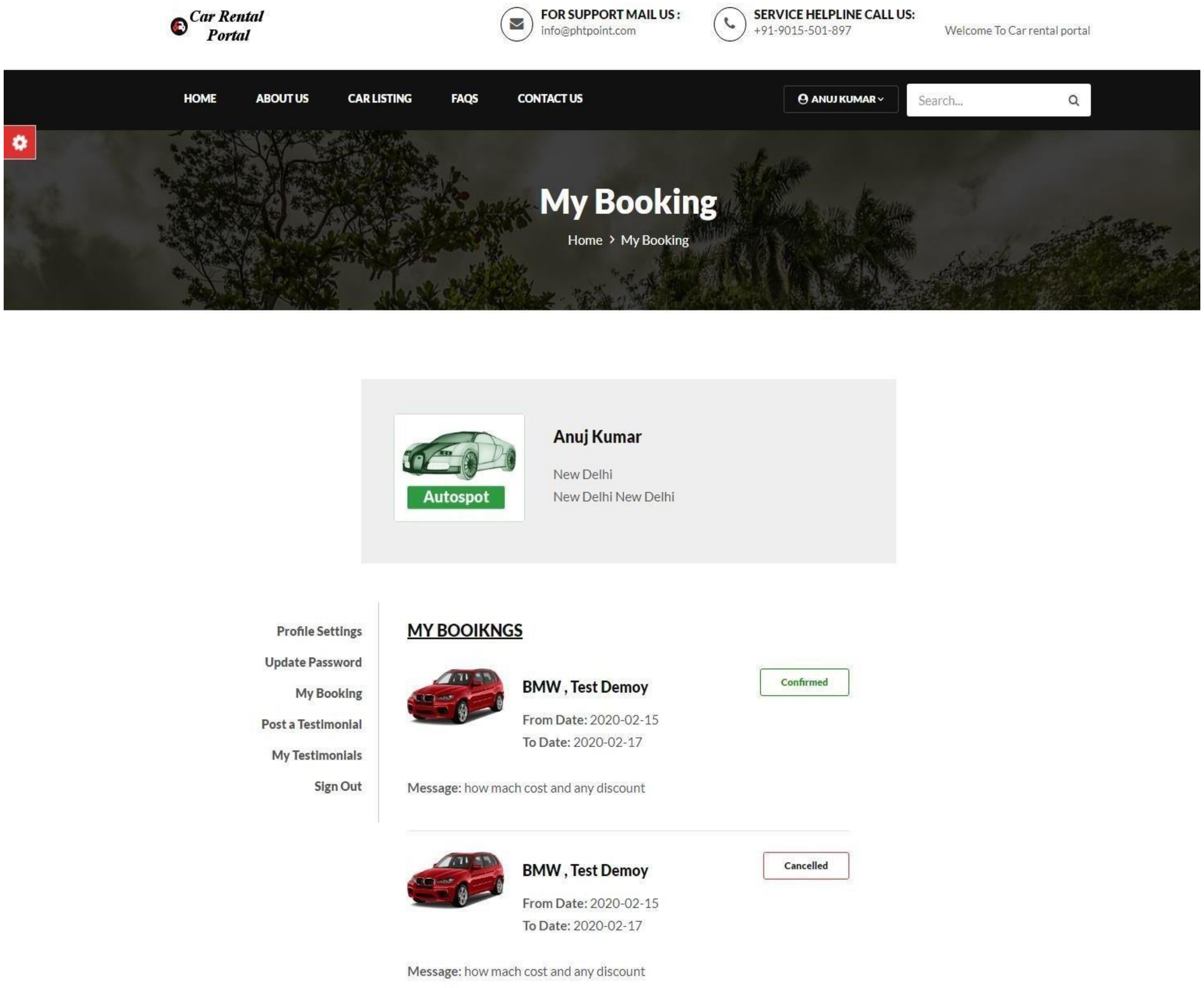
Date of Birth (dd/mm/yyyy): 03/02/1990

Your Address: New Delhi

Country: New Delhi

City: New Delhi

SNAPSHOT 5: USER DETAILS



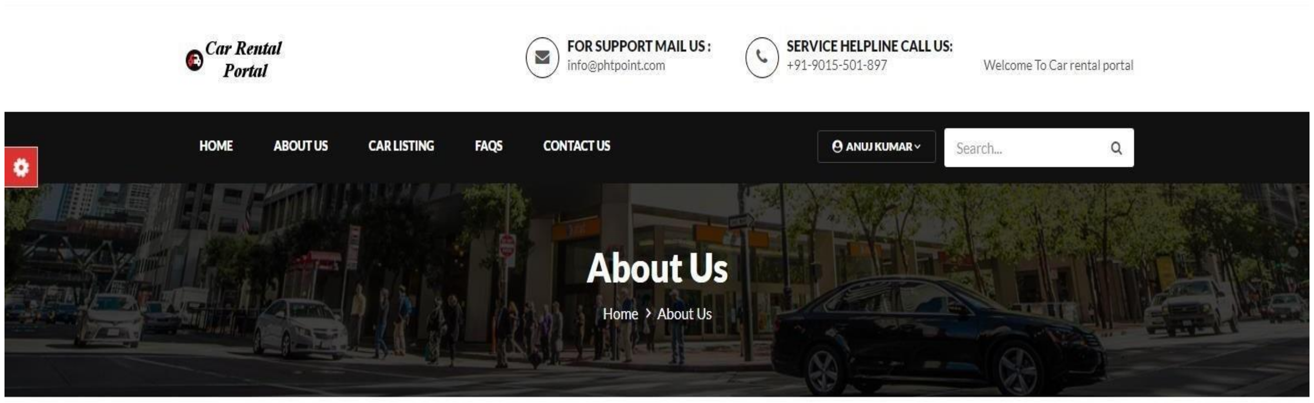
Anuj Kumar
New Delhi
New Delhi New Delhi

MY BOOIKNGS

BMW , Test Demoy
From Date: 2020-02-15
To Date: 2020-02-17
Message: how mach cost and any discount

BMW , Test Demoy
From Date: 2020-02-15
To Date: 2020-02-17
Message: how mach cost and any discount

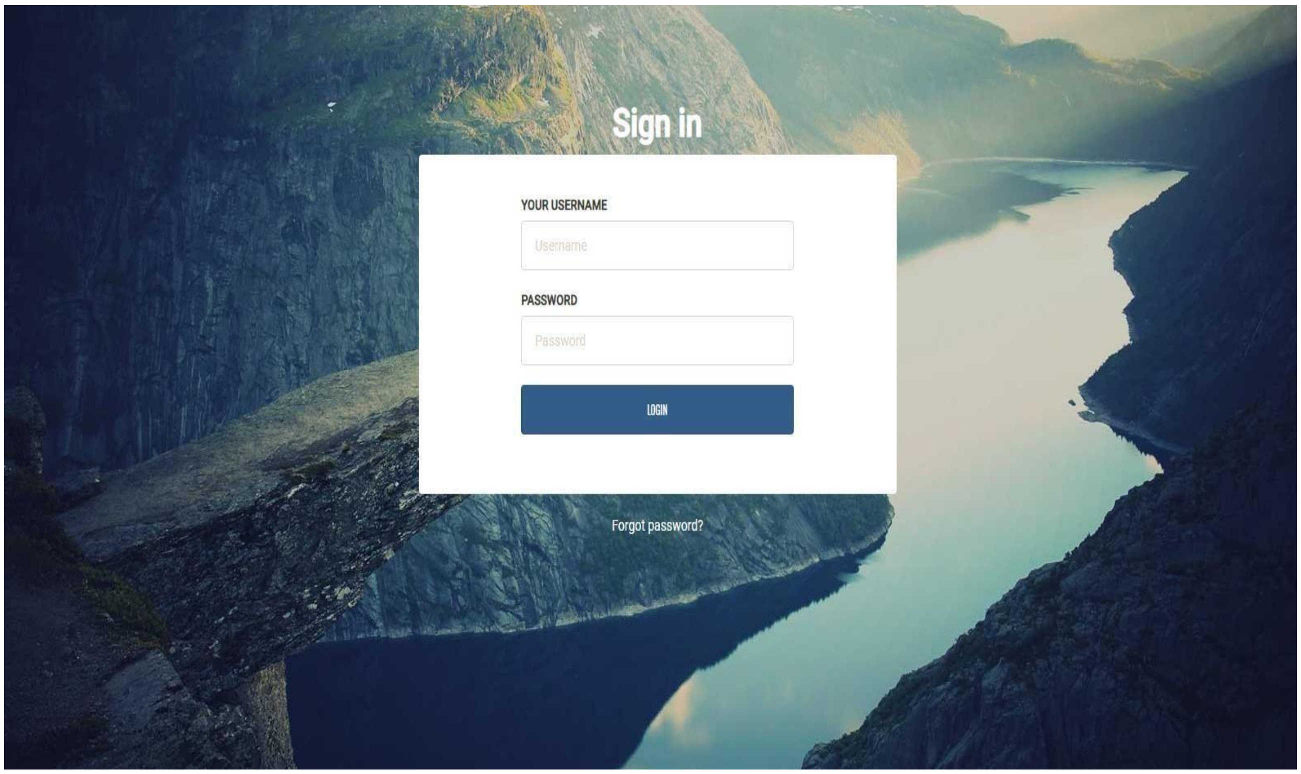
SNAPSHOT 6: BOOKING DETAILS.



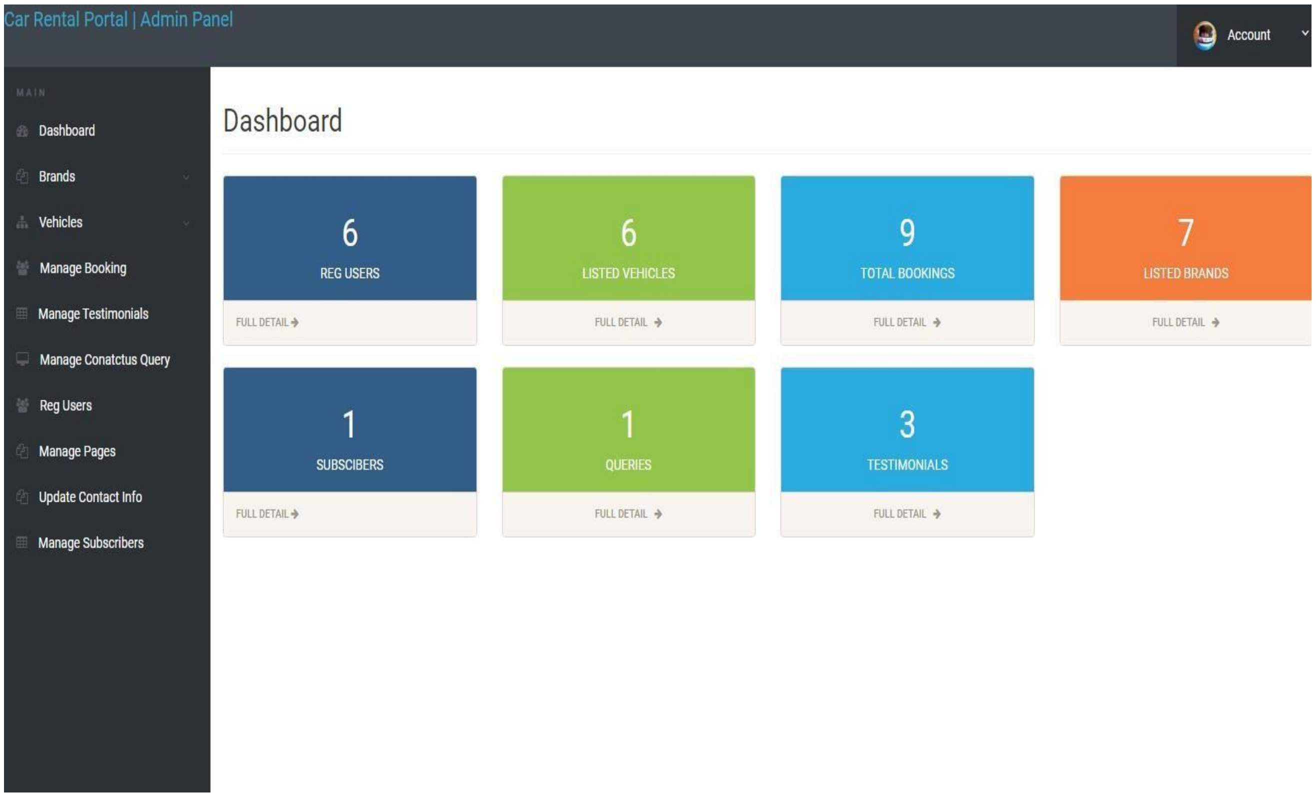
About Us

At vero eos et accusamus et iusto odio dignissimos ducimus qui blanditiis praesentium voluptatum deleniti atque corrupti quos dolores et quas molestias excepturi sint occaecati cupiditate non provident, similique sunt in culpa qui officia deserunt mollitia animi, id est laborum et dolorum fuga. Et harum quidem rerum facilis est et expedita distinctio. Nam libero tempore, cum soluta nobis est eligendi optio cumque nihil impedit quo minus id quod maxime placeat facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores alias consequatur aut perferendis doloribus asperiores repellat

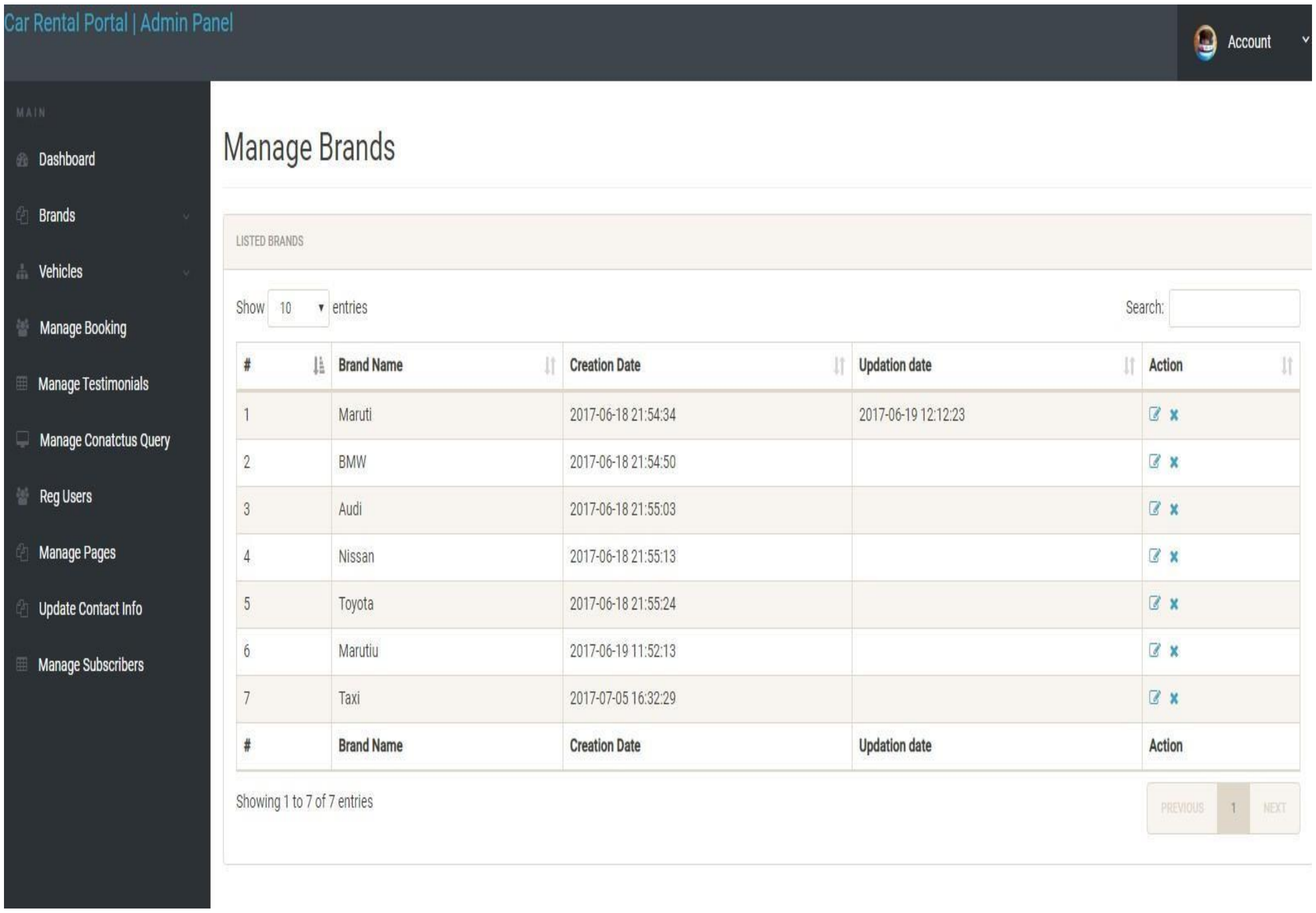
SNAPSHOT 7: ABOUT US



SNAPSHOT 8: ADMIN LOGIN



SNAPSHOT 9: ADMIN DASHBORD



Car Rental Portal | Admin Panel

Manage Brands

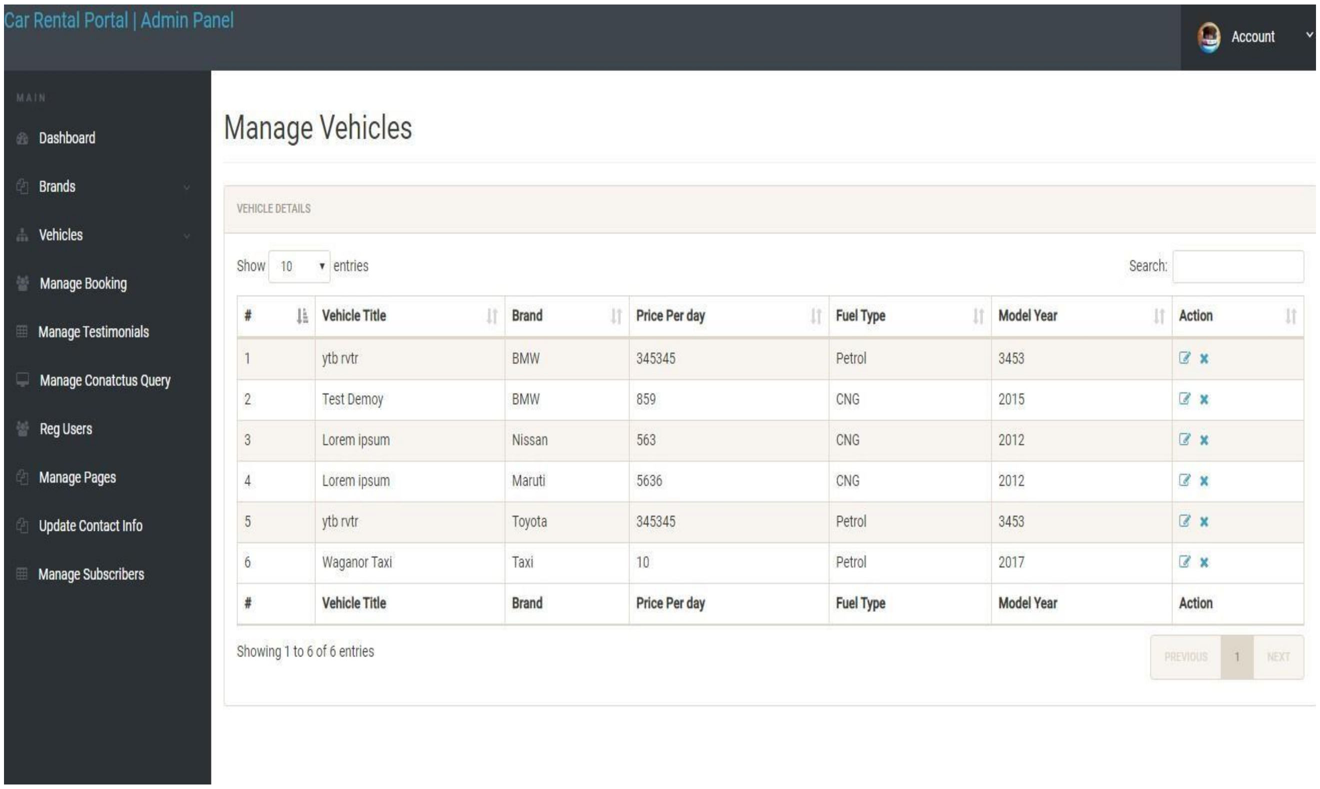
LISTED BRANDS

Show 10 entries Search:

#	Brand Name	Creation Date	Updation date	Action
1	Maruti	2017-06-18 21:54:34	2017-06-19 12:12:23	
2	BMW	2017-06-18 21:54:50		
3	Audi	2017-06-18 21:55:03		
4	Nissan	2017-06-18 21:55:13		
5	Toyota	2017-06-18 21:55:24		
6	Marutiu	2017-06-19 11:52:13		
7	Taxi	2017-07-05 16:32:29		
#	Brand Name	Creation Date	Updation date	Action

Showing 1 to 7 of 7 entries

SNAPSHOT 10: MANAGE BRAND



Manage Vehicles

VEHICLE DETAILS

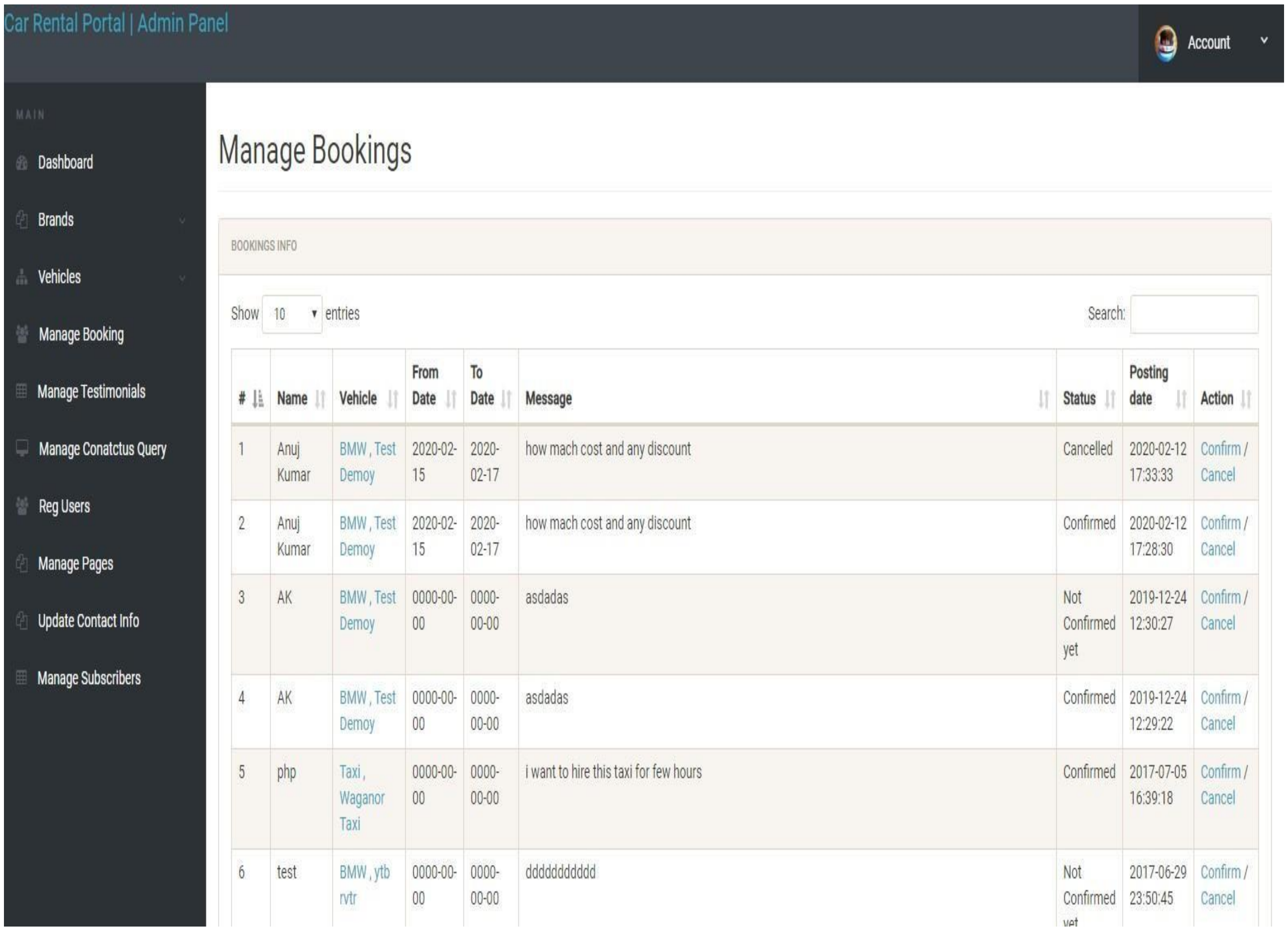
Show 10 entries Search:

#	Vehicle Title	Brand	Price Per day	Fuel Type	Model Year	Action
1	ytb rvtr	BMW	345345	Petrol	3453	
2	Test Demoy	BMW	859	CNG	2015	
3	Lorem ipsum	Nissan	563	CNG	2012	
4	Lorem ipsum	Maruti	5636	CNG	2012	
5	ytb rvtr	Toyota	345345	Petrol	3453	
6	Waganor Taxi	Taxi	10	Petrol	2017	
#	Vehicle Title	Brand	Price Per day	Fuel Type	Model Year	Action

Showing 1 to 6 of 6 entries

PREVIOUS 1 NEXT

SNAPSHOT 11: MANAGE VEHICLE



Manage Bookings

BOOKINGS INFO

Show 10 entries Search:

#	Name	Vehicle	From Date	To Date	Message	Status	Posting date	Action
1	Anuj Kumar	BMW , Test Demoy	2020-02-15	2020-02-17	how mach cost and any discount	Cancelled	2020-02-12 17:33:33	Confirm / Cancel
2	Anuj Kumar	BMW , Test Demoy	2020-02-15	2020-02-17	how mach cost and any discount	Confirmed	2020-02-12 17:28:30	Confirm / Cancel
3	AK	BMW , Test Demoy	0000-00-00	0000-00-00	asdadas	Not Confirmed yet	2019-12-24 12:30:27	Confirm / Cancel
4	AK	BMW , Test Demoy	0000-00-00	0000-00-00	asdadas	Confirmed	2019-12-24 12:29:22	Confirm / Cancel
5	php	Taxi , Waganor Taxi	0000-00-00	0000-00-00	i want to hire this taxi for few hours	Confirmed	2017-07-05 16:39:18	Confirm / Cancel
6	test	BMW , ytb rvtr	0000-00-00	0000-00-00	dddddddddd	Not Confirmed yet	2017-06-29 23:50:45	Confirm / Cancel

SNAPSHOT 12: MANAGE BOOKING

CHAPTER 12

SAMPLE CODING

```
<?php
// DB credentials.
define('DB_HOST','loca
lhost');
define('DB_USER','root
'); define('DB_PASS','');
define('DB_NAME','car
rental'); // Establish
database connection.
try
{
$dbh = new PDO("mysql:host=".DB_HOST.";dbname=".DB_NAME,DB_USER,
DB_PASS,array(PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES 'utf8'"));
} catch
(PDOException
$e)
{ exit("Error: " . $e-
>getMessage());
}
?>
<?php $sql = "SELECT
tblvehicles.VehiclesTitle,tblbrands.BrandName,tblvehicles.PricePerDay,tblvehicles.Fu
elType,tb
lvehicles.ModelYear,tblvehicles.id,tblvehicles.SeatingCapacity,tblvehicles.VehiclesOv
```

```
erview,tb lvehicles.Vimage1 from tblvehicles join tblbrands on
tblbrands.id=tblvehicles.VehiclesBrand"; $query = $dbh -> prepare($sql);
$query->execute();
$results=$query->fetchAll(PDO::FETCH_OBJ);
$cnt=1; if($query-
>rowCount() > 0)
{
foreach($results
as $result)
{
?>
<?php $sql = "SELECT
tblusers.FullName,tblbrands.BrandName,tblvehicles.VehiclesTitle,tblbooking.FromD
ate,tblboo king.ToDate,tblbooking.message,tblbooking.VehicleId as
vid,tblbooking.Status,tblbooking.PostingDate,tblbooking.id from tblbooking join
tblvehicles on tblvehicles.id=tblbooking.VehicleId join tblusers on
tblusers.EmailId=tblbooking.userEmail join tblbrands on
tblvehicles.VehiclesBrand=tblbrands.id ORDER BY tblbooking.id DESC "; $query =
$dbh -> prepare($sql);
$query->execute();
$results=$query->fetchAll(PDO::FETCH_OBJ);
$cnt=1; if($query-
>rowCount() > 0)
{
foreach($results
as $result)
{                                    ?>
<?php
session_start();
error_reporting(0);
```

```
include('includes/config.
php');
if(strlen($_SESSION['alog
in'])==0)
        {
header('location:index.php');
} else{
if(isset($_G
ET['del']))
{
$id=$_GET['del'];
$sql = "delete from tblbrands  WHERE id=:id";
$query = $dbh->prepare($sql);
$query -> bindParam(':id',$id, PDO::PARAM_STR);
$query -> execute();
$msg="Page data updated  successfully";

}
 ?>
<?php
session_start();
include('includes/c
onfig.php');
error_reporting(0);
if(isset($_POST['sub
mit']))
{
$fromdate=$_POST['fromdate'];
$todate=$_POST['todate'];
$message=$_POST['message'];
$useremail=$_SESSION['login'];
$status=0;
$vhid=$_GET['vhid'];
$sql="INSERT INTO
tblbooking(userEmail,VehicleId,FromDate,ToDate,message,Status)
VALUES(:useremail,:vhid,:fromdate,:todate,:message,:status)";
$query = $dbh->prepare($sql);
$query->bindParam(':useremail',$useremail,PDO::PARAM_STR);
$query->bindParam(':vhid',$vhid,PDO::PARAM_STR);
$query->bindParam(':fromdate',$fromdate,PDO::PARAM_STR);
$query->bindParam(':todate',$todate,PDO::PARAM_STR);
$query->bindParam(':message',$message,PDO::PARAM_STR);
$query->bindParam(':status',$status,PDO::PARAM_STR);
$query->execute();
$lastInsertId = $dbh->lastInsertId();
if($lastInsertId)
{ echo "<script>alert('Booking
successfull.');</script>"; } else { echo
"<script>alert('Something went wrong.
Please try again');</script>"; }

}

?>

<?php
```

```
session_start();
error_reporting(0);
include('includes/config
.php');
if(strlen($_SESSION['log
in'])==0)
  {
header('location:index.php');
} else{
if(isset($_POST['updat
eprofile']))
  {
$name=$_POST['fullname'];
$mobileno=$_POST['mobilenumber'];
$dob=$_POST['dob'];
$adress=$_POST['address'];
$city=$_POST['city'];
$country=$_POST['country'];
$email=$_SESSION['login'];
$sql="update tblusers set
FullName=:name,ContactNo=:mobileno,dob=:dob,Address=:adress,City=:city,Countr
y=:countr y where EmailId=:email"; $query = $dbh->prepare($sql);
$query->bindParam(':name',$name,PDO::PARAM_STR);
$query->bindParam(':mobileno',$mobileno,PDO::PARAM_STR);
$query->bindParam(':dob',$dob,PDO::PARAM_STR);
$query->bindParam(':adress',$adress,PDO::PARAM_STR);
$query->bindParam(':city',$city,PDO::PARAM_STR);
$query->bindParam(':country',$country,PDO::PARAM_STR);
$query->bindParam(':email',$email,PDO::PARAM_STR);
$query->execute();
$msg="Profile Updated Successfully";
}

?>
<?php
//Query for Listing count
$brand=$_POST['brand'];
$fueltype=$_POST['fueltype'];
$sql = "SELECT id from tblvehicles where tblvehicles.VehiclesBrand=:brand and
tblvehicles.FuelType=:fueltype";
$query = $dbh -> prepare($sql);
$query -> bindParam(':brand',$brand, PDO::PARAM_STR);
$query -> bindParam(':fueltype',$fueltype, PDO::PARAM_STR);
$query->execute();
$results=$query->fetchAll(PDO::FETCH_OBJ);
$cnt=$query->rowCount();
?>
<p><span><?php echo htmlentities($cnt);?> Listings</span></p>
</div>
</div>

<?php
```

```
$sql = "SELECT tblvehicles.*,tblbrands.BrandName,tblbrands.id as bid from
tblvehicles join tblbrands on tblbrands.id=tblvehicles.VehiclesBrand where
tblvehicles.VehiclesBrand=:brand and tblvehicles.FuelType=:fueltype";
$query = $dbh -> prepare($sql);
$query -> bindParam(':brand',$brand, PDO::PARAM_STR);
$query -> bindParam(':fueltype',$fueltype, PDO::PARAM_STR);
$query->execute();
$results=$query->fetchAll(PDO::FETCH_OBJ);
$cnt=1; if($query-
>rowCount() > 0)
{
foreach($results
as $result)
{  ?>
<?php $sql = "SELECT * from  tblbrands ";
$query = $dbh -> prepare($sql);
$query->execute();
$results=$query->fetchAll(PDO::FETCH_OBJ);
$cnt=1; if($query-
>rowCount() > 0)
{
foreach($results
as $result)
{       ?>
<option value="<?php echo htmlentities($result->id);?>"><?php echo
htmlentities($result-
>BrandName);?></option>
<?php }} ?>
```