A

PROJECT REPORT ON

ONLINE VEHICLE RENTAL SYSTEM

In a partial fulfilment of the award

For the

Degree of

MASTER OF COMPUTERS APPLICATIONS

Submitted by

APPASHA

2122-20-862-047

Under the Guidance of

Mrs. V. Nagasri

Osmania University

AURORA'S PG COLLEGE (MBA)

(Affiliated to Osmania University)

Punjagutta, HYDERABAD-500082

(2020-2022)

AURORA'S POST -GRADUATE COLLEGE(MBA)
#6-3-542, Punjagutta, Hyderabad-500 082.

CERTIFICATE

This is certify that, this project entitled

ONLINE VEHICLE RENTAL SYSTEM

Is a bonafide work carried out by

APPASHA

2122-20-862-047

As part of their curriculum in the

Department of Computer Science

In partial fulfilment of the award of

MASTER OF COMPUTER APPLICATIONS

This work has been carried out our guidance

INTERNAL GUIDE EXTERNAL EXAMINER

HEAD OF THE DEPARTMENT PRINCIPAL

DECLARATION

I **APPASHA** hereby declare that the project "**ONLINE VEHICLE RENTAL SYSTEM"** this project is submitted to Osmania University, Hyderabad in partial fulfilment of the requirements for the award of the degree of "**MASTER OF COMPUTER APPLICATIONS".** The results embodied in this dissertation have not been submitted to any other university or institution for the award of any degree or diploma.

APPASHA

2122-20-862-047

ACKNOWLEDGEMENT

The success and final outcome of this project required a lot of guidance and assistance from many people and I am extremely privileged to have got this all along the completion of my project. All that I have done is only due to such supervision and assistance and I would not forget to thank them.

I consider myself lucky enough to get such a good project. This project would be added as an asset to my academic profile.

I would like to express my thankfulness to my **Vice Principal Mrs. R. Sushma and my Dept HOD Mrs. Bidyutlata Sahoo** for their constant motivation and valuable help through the project work.

I owe my deep gratitude to my project guide **Ms B.sreelekha** who took keen interest in my project work and guided me all along, till the completion of my project work by providing all the necessary information for developing a good system. I would like to thank my friends who helped me in improving my thesis.

I am thankful to and fortunate enough to get constant encouragement, support and guidance from all Teaching staff of MCA which helped me in successfully completing our project work. Also, I would like to extend our sincere esteems to all staff in the laboratory for their timely support.

Finally, I wish to thank my parents for their love and encouragement, without whom I would never have enjoyed so many opportunities.

APPASHA

2122-20-862-047

PANHYD0341/MAJOR PROJECT/ML/2021-22

COMPLETION CERTIFICATION

This is to certify that **Mr./Ms. APPASHA,** Roll No-**(2122-20-862-047),** who is pursuing **MCA, (Master of Computer Applications),** Department at **AURORA'S PG COLLEGE** has successfully completed his/her Major Project at, **Pantech E Learning Pvt Limited** on **("ONLINE VEHICLE RENTAL SYSTEM")** under the guidance of **Ms. B. SREELEKHA** in our organization.

During the Major Projects period, the candidate has shown keen interest and commitment towards learning and his/her performance was good.

Period of Major Project from: 22-04-2022 to 29-07-2022

Yours truly,

Pantech E Learning Pvt.Ltd,

Pantech E Learning Pvt. Ltd. Hyd-16.

Srinivasan.N

(Branch Manager)

Online Vehicle Rental System

USING WEB APPLICATION

ABSTRACT

A car rental service is wishing to have a user interface that will allow their customers to view the models, descriptions and prices of different cars available. The user has the ability to register and log in to the web site and see their rental plan. The web site will be responsive, allowing for the customer to view it on any device, from tables to mobile phones and desktop computers. The administrator will also be able to login through the same from but have the ability to add/remove new car rentals, change prices, and so on. Potential customers should be able to view all the cars available to rent even without logging in as well as rent without having an account, through the option is provided upon checkout.

The Car Rental System is being developed for customers so that they can book their cars from any part of the world. This application takes information from the customers through filling their details. A customer being registered in the website has the facility to book a Car which he requires. The proposed system is completely integrated online systems. It automates manual procedure in an effective and efficient way. This automated system facilitates customer and provides to fill up the details according to their requirements. It includes type of car they are trying to hire and location. The purpose of this system is to develop a web site for the people who can book theirCar along with requirements from any part of the world. Car rental system provideCar to User in their location on short time.

INDEX

INTRODUCTION

1.1 INTRODUCTION TO ONLINE CAR RENTAL SYSTEM

This project is designed so as to be used by Car Rental Company specialising inrenting cars to customers. It is an online system through which customers can view available cars, register, view profile and book car. Here, User has to Login To book a car. The user can search for cars easily and book. For bookings, the user has to provide information such as Booking Dates and Text Message. All cardetails are provided and it also includes Car's feature and Overview. The user canalso post their Testimonials and the user can update their Profile as well as passwords anytime they want from the site. Admin can Add/Manage car brands, manage cars, bookings, testimonial, pages and many more. It's easy to operate and understand by users. This site makes customers easy for car rental. The designis pretty simple and the user won't find it difficult to understand, use and navigate.

1.2 REASON FOR THE PROJECT

- **Enhance Business Processes**: To be able to use internet technology to project the rental company to the global world instead of limiting their services to their local domain alone, thus increase their return on investment (ROI).
- **Online Car Reservation**: A tools through which customers can Booking available cars online prior to their expected pick-up date or time.
- **Customer's registration**: A registration portal to hold customer's details, monitor their transaction and used same to offer better and improve services to them.
- **Group Booking/Event Management**: Allows the customer to bookspace for a group in the case of weddings or corporate meetings

1.3 PROBLEM STATEMENT

A car rental is a car that can be used temporarily for a fee during a specified period. Getting a rental car helps people get around despite the fact they do not have access to their own personal car or don't own a car at all. The individual whoneeds a car must contact a rental car company and contract out for a car. This system increases customer retention and simplify car and staff management.

1.4 AIMS & OBJECTIVE

- ➢ To produce a web-based system that allow customer to register and Booking car online and for the company to effectively manage their car rental business.
- ➢ To ease customer's task whenever they need to rent a car

1.5 SCOPE

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

- ➢ **Car-rental industry**: This includes study on how the car 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.

1.6 SUMMARY

The main objective of this Car Rental System project will enable the user to renta car. The user shall login to the system and check for availability of cars. The user specifies a type of car and the journey date and time. The Car Rental Systemshall check for the availability of the car and rent the car to the customer. All the data regarding the rental cars are stored in MySQL database. The user has to enterhis name, address, phone details and check for the cars available for rent. The UIis very simple and the connectivity to back end is robust. The main advantage is that the user shall be able to choose a car depending on his budget.

LITERATURE REVIEW

2.1 LITERATURE SURVEY

- **Zoom car Self-drive car rental**

Zoom car is a self-drive car rental service which allows user to rent cars by the hour, day, week or month.

- **Eco Rent a Car**

This venture aims to offer individuals as well as corporates with superlative car hire services. Over time this firm has made a mark in the self-drive car rental segment.

- **Vroom Drive - Self Drive Car Rental**

Good place to rent a car for weekend trips. They had a good business plan until they changed it. The location is only one in the city and not feasible one. They had home pickup and drop service with reasonable price of 500.

- **Zing Car, Self-Drive Car rental**

One of the best place to hire a car, outstanding customer support with simple rules and regulations, friendly customer service with brand new cars far better than other car rental services, must provide all original cards to take vehicle.

- **Drivezy Self Drive Car Rentals**

Drivezy is a very good example of how jugged can make any business run in India

- **MyChoize Self Drive Cars**

One of the better rental services in India. Clean/new cars, transparent documentation, properly serviced cars and good staff! Really happy customer. Pathetic Service and quality of cars. In order to evaluate the effectiveness of the system, the study was successfully done for each type of the criteria. We have developed our project by keeping in mind all the hurdles the customers are facing from other Online Rental Services.

- **Working proposed**

After comparison study the application that is build will have the following features like

• To work on any platform

• Have to work freely

• Have optimal working process

• Ads should not be displayed on the webpage

• User friendly

This project has used the insight on user technology to construct and integrating the web-based system with SMS technology to enhance the service provided by the car rental agencies for the customers about the booking status, and the availability of the car reserved. Thus, the system provides a convenient way of notification. Besides, this system makes it easy to get car information, book a car and quickly rent a car.

2.2 TOOLS AND TECHNOLOGY USED

- **Xamp**

Xamp is a free and open-source, cross-platform, web server solutions start package developed by apache friends, consisting mainly of Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server possible.

- **Notepad ++**

Notepad++ is a text and source code editor for use with Microsoft Windows. It supports tabbed editing, which allows working with multiple open files in a single window. The project's name comes from the C increment operator.

- **Php Myadmin**

phpMyAdmin is a free and open source administration tool for MySQL and MariaDB. As a portable web application written primarily in PHP, it has become one of the most popular MySQL administration tools, especially for web hosting services

SYSTEM ANALYSIS

3.1.1 EXISTING SYSTEM

1. Introduction

Although many online portals have come into the picture for providing online carbooking service . But most of the car renting companies are using traditional way to deal with the customer. Which are time and labour consuming? An existing system can provide manually paper work or excel sheet to track the booking and registered cars details. The user has to go in the office where the user can get the car on rent and book their car. Most of the time user does not get a sight of the car in which he is planning to travel.Which results in compromising the travel comfort. In the existing system, you cannot provide feedback of the user to the admin directly. The user gets fluctuation every time he/she travels. Maintaining excel sheet or paper book record of reservation is very laborious work. Chances of error are more. No automation involves which means they area very slow to process.

2. Problem Statement

The Manual car rental system provides services only during office hours. So; customers have limited time to make any transactions or reservation of the cars. The existence of the online car rental systems nowadays has overcome the limitation of the business operation hour. There are some customers who faced aproblem in choosing car to be rented which suitable with some of the important requirements.

i. To rent a car a prospective renter must first go to the nearest office to register as a client.
ii. Cars that provide difficulties to rent out are normally advertised in local or national newspaper. It involves a lot of paper work and consumes time
iii. Details are stored in papers
iv. Maintenance is a huge problem
v. Updations, changes in details is a tedious task
vi. Performance is not achieved up to the requirements.

3.1.2 PROPOSED SYSTEM

1. Introduction

The proposed system facilitates the customers to fill up their details, and to give a brief description of a car they want to book. This new system is very helpful forcustomers who want to hire their cars through this site.

This Car Rental System project will enable the user to rent a car. The user shall login to the system and check for availability of cars. The user specifies a type ofcar and the journey date and time. The Car Rental System shall check for the availability of the car and rent the car to the customer. All the data regarding therental cars are stored in MySQL database. The user has to enter his name, address, phone details and check for the cars available for rent. The UI is very simple and the connectivity to back end is robust. The main advantage is that the user shall be able to choose a car depending on his budget.

2. Advantages:

- ➢ First the customer has to make a reservation and later on in the process has to do registration.
- ➢ Second if the customer had already registered himself then he can continue booking in his own account by giving his customer id or mail id.
- ➢ Thirdly, the customer can amend details or update his details.
- ➢ Maintenance is easy and performance is good
- ➢ It is easy to use and understand.
- ➢ It reduce the time complexity.

3.2 FEASIBILITY STUDY

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- ➢ Technical Feasibility
- ➢ Operation Feasibility
- ➢ Economical Feasibility
- ➢ Legal Feasibility

1. Technical Feasibility

- ➢ The technical issue usually raised during the feasibility stage of the investigation includes the following:
- ➢ Do the proposed equipments have the technical capacity to hold the datarequired to use the new system?
- ➢ Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
- ➢ Are there technical guarantees of accuracy, reliability, ease of access and data security?

2. Operational Feasibility

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following: -

- ➢ Is there sufficient support for the management from the users?
- ➢ Will the system be used and work properly if it is being developed andimplemented?

- Will there be any resistance from the user that will undermine the possibleapplication benefits?

3. Economical Feasibility

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs.

4. Legal Feasibility

In the legal feasibility it is necessary to check that the software we are going to develop is legally correct which means that the ideas which we have taken for
the proposed system will be legally implemented or not so,it is also an important step in feasibility study.

3.3 SYSTEM REQUIREMENTS

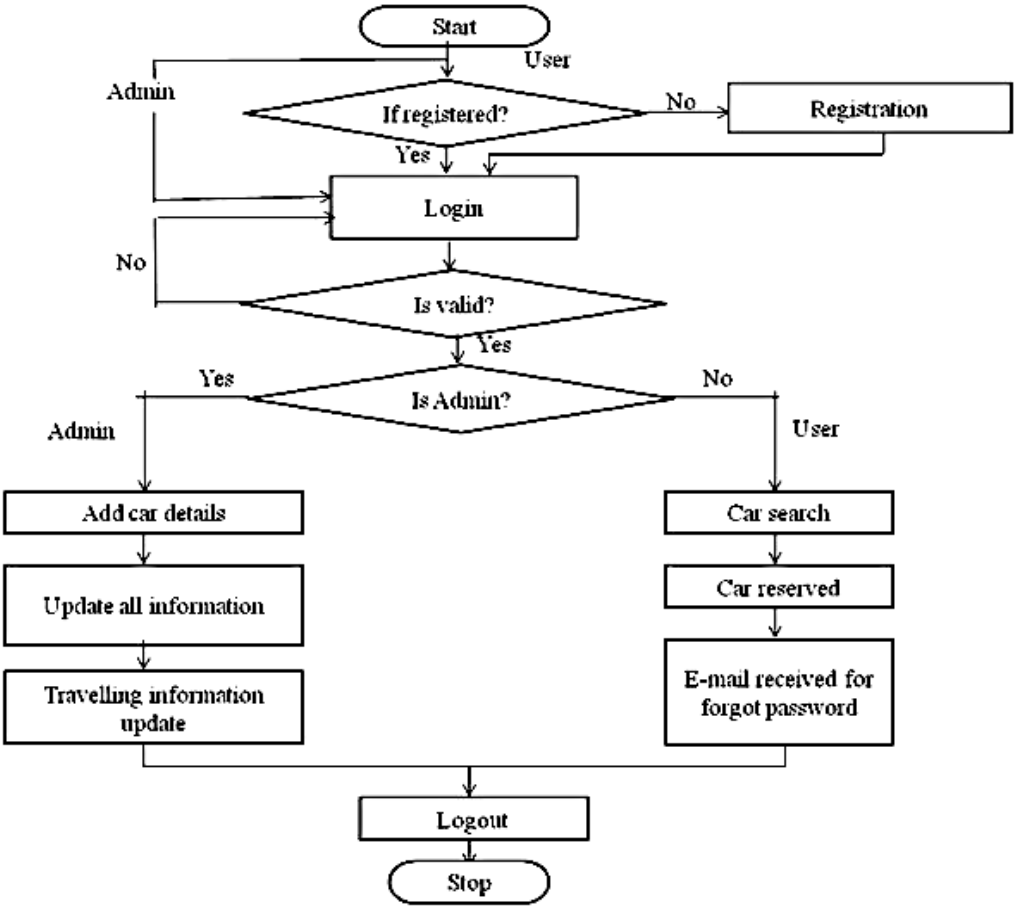
3.3.1 HARDWARE REQUIREMENTS

Processor	:	Intel Pentium Dual Core
RAM	:	512 MB
Hard Disk	:	160 GB Space

3.3.2 SOFTWARE REQUIREMENTS

Operating System	:	Windows /iOS/Unix
Web Browser.	:	IE/Google Chrome/Firefox
Tools.	:	NODE JS,VS CODE
Web Design	:	HTML, CSS, JAVASCRIPT

3.4 SYSTEM ARCHITECTURE



INTRODUCTION TO REACT

React:

ReactJS tutorial provides basic and advanced concepts of ReactJS. Currently, ReactJS is one of the most popular JavaScript front-end libraries which has a strong foundation and a large community.ReactJS is a **declarative**, **efficient**, and flexible **JavaScript library** for building reusable UI components. It is an open-source, component-based front end library which is responsible only for the view layer of the application. It was initially developed and maintained by Facebook and later used in its products like WhatsApp & Instagram.Our ReactJS tutorial includes all the topics which help to learn ReactJS. These are ReactJS Introduction, ReactJS Features, ReactJS Installation, Pros and Cons of ReactJS, ReactJS JSX, ReactJS Components, ReactJS State, ReactJS Props, ReactJS Forms, ReactJS Events, ReactJS Animation and many more.

Why react :

The main objective of ReactJS is to develop User Interfaces (UI) that improves the speed of the apps. It uses virtual DOM (JavaScript object), which improves the performance of the app. The JavaScript virtual DOM is faster than the regular DOM. We can use ReactJS on the client and server-side as well as with other frameworks. It uses component and data patterns that improve readability and helps to maintain larger apps.

Introduction:

ReactJS is a declarative, efficient, and flexible JavaScript library for building reusable UI components. It is an open-source, component-based front end library responsible only for the view layer of the application. It was created by **Jordan Walke,** who was a software engineer at **Facebook.** It was initially developed and maintained by Facebook and was later used in its products like **WhatsApp** & **Instagram.** Facebook developed ReactJS in **2011** in its newsfeed section, but it was released to the public in the month of

May 2013.

Today, most of the websites are built using MVC (model view controller) architecture. In MVC architecture, React is the 'V' which stands for view, whereas the architecture is provided by the Redux or Flux.A ReactJS application is made up of multiple components, each component responsible for outputting a small, reusable piece of HTML code.

The components are the heart of all React applications. These Components can be nested with other components to allow complex applications to be built of simple building blocks.

Why react using

Today, many JavaScript frameworks are available in the market(like angular, node), but still, React came into the market and gained popularity amongst them. The previous frameworks follow the traditional data flow structure, which uses the DOM (Document Object Model). DOM is an object which is created by the browser each time a web page is loaded. It dynamically adds or removes the data at the back end and when any modifications were done, then each time a new DOM is created for the same page. This repeated creation of DOM makes unnecessary memory wastage and reduces the performance of the application.

Therefore, a new technology ReactJS framework invented which remove this drawback. ReactJS allows you to divide your entire application into various components. ReactJS still used the same traditional data flow, but it is not directly operating on the browser's Document Object Model (DOM) immediately; instead, it operates on a virtual DOM.

It means rather than manipulating the document in a browser after changes to our data, it resolves changes on a DOM built and run entirely in memory. After the virtual DOM has been updated, React determines what changes made to the actual browser's DOM. The React Virtual DOM exists entirely in memory and is a representation of the web browser's DOM. Due to this, when we write a React component, we did not write directly to the DOM; instead, we are writing virtual components that react will turn into the DOM.

React version

SN	Version	Release Date	Significant Changes
1.	0.3.0	29/05/2013	Initial Public Release
2.	0.4.0	20/07/2013	Support for comment nodes {/* */} , Improved server-side rendering APIs, Removed React.autoBind, Support for the key prop, Improvements to forms, Fixed bugs.
3.	0.5.0	20/10/2013	Improve Memory usage, Support for Selection and Composition events, Support for getInitialState and getDefaultProps in mixins, Added React.version and React.isValidClass, Improved compatibility for Windows.
4.	0.8.0	20/12/2013	Added support for rows & cols, defer & async, loop for <audio> & <video>, autoCorrect attributes. Added onContextMenu events, Upgraded jstransform and esprima-fb tools, Upgraded browserify.
5.	0.9.0	20/02/2014	Added support for crossOrigin, download and hrefLang, mediaGroup and muted, sandbox, seamless, and srcDoc, scope attributes, Added any, arrayOf, component, oneOfType, renderable, shape to React.PropTypes, Added support for onMouseOver and onMouseOut event, Added support for onLoad and onError on <img> elements.
6.	0.10.0	21-03-2014	Added support for srcSet and textAnchor attributes, add update function for immutable data, Ensure all void elements don't insert a closing tag.
7.	0.11.0	17/07/2014	Improved SVG support, Normalized e.view event, Update $apply command, Added support for namespaces, Added new transformWithDetails API, includes pre-built packages under dist/, MyComponent() now returns a descriptor, not an instance.

8.	0.12.0	21/11/2014	Added new features Spread operator ({...}) introduced to deprecate this.transferPropsTo, Added support for acceptCharset, classID, manifest HTML attributes, React.addons.batchedUpdates added to API, @jsx React.DOM no longer required, Fixed issues with CSS Transitions.
9.	0.13.0	10/03/2015	Deprecated patterns that warned in 0.12 no longer work, ref resolution order has changed, Removed properties this._pendingState and this._rootNodeID, Support ES6 classes, Added API React.findDOMNode(component), Support for iterators and immutable-js sequences, Added new features React.addons.createFragment, deprecated React.addons.classSet.
10.	0.14.1	29/10/2015	Added support for srcLang, default, kind attributes, and color attribute, Ensured legacy .props access on DOM nodes, Fixed scryRenderedDOMComponentsWithClass, Added react-dom.js.
11.	15.0.0	07/04/2016	Initial render now uses document.createElement instead of generating HTML, No more extra s, Improved SVG support, ReactPerf.getLastMeasurements() is opaque, New deprecations introduced with a warning, Fixed multiple small memory leaks, React DOM now supports the cite and profile HTML attributes and cssFloat, gridRow and gridColumn CSS properties.
12.	15.1.0	20/05/2016	Fix a batching bug, Ensure use of the latest object-assign, Fix regression, Remove use of merge utility, Renamed some modules.
13.	15.2.0	01/07/2016	Include component stack information, Stop validating props at mount time, Add React.PropTypes.symbol, Add onLoad handling to <link> and onError handling to <source> element, Add isRunning() API, Fix performance regression.

14.	15.3.0	30/07/2016	Add React.PureComponent, Fix issue with nested server rendering, Add xmlns, xmlnsXlink to support SVG attributes and referrerPolicy to HTML attributes, updates React Perf Add-on, Fixed issue with ref.
15.	15.3.1	19/08/2016	Improve performance of development builds, Cleanup internal hooks, Upgrade fbjs, Improve startup time of React, Fix memory leak in server rendering, fix React Test Renderer, Change trackedTouchCount invariant into a console.error.
16.	15.4.0	16/11/2016	React package and browser build no longer includes React DOM, Improved development performance, Fixed occasional test failures, update batchedUpdates API, React Perf, and ReactTestRenderer.create().
17.	15.4.1	23/11/2016	Restructure variable assignment, Fixed event handling, Fixed compatibility of browser build with AMD environments.
18.	15.4.2	06/01/2017	Fixed build issues, Added missing package dependencies, Improved error messages.
19.	15.5.0	07/04/2017	Added react-dom/test-utils, Removed peerDependencies, Fixed issue with Closure Compiler, Added a deprecation warning for React.createClass and React.PropTypes, Fixed Chrome bug.
20.	15.5.4	11/04/2017	Fix compatibility with Enzyme by exposing batchedUpdates on shallow renderer, Update version of prop-types, Fix react-addons-create-fragment package to include loose-envify transform.
21.	15.6.0	13/06/2017	Add support for CSS variables in style attribute and Grid style properties, Fix AMD support for addons depending on react, Remove unnecessary dependency, Add a deprecation warning for React.createClass and React.DOM factory helpers.

22.	16.0.0	26/09/2017	Improvd error handling with introduction of "error boundaries", React DOM allows passing non-standard attributes, Minor changes to setState behavior, remove react-with-addons.js build, Add React.createClass as create-react-class, React.PropTypes as prop-types, React.DOM as react-dom-factories, changes to the behavior of scheduling and lifecycle methods.
23.	16.1.0	9/11/2017	Discontinuing Bower Releases, Fix an accidental extra global variable in the UMD builds, Fix onMouseEnter and onMouseLeave firing, Fix <textarea> placeholder, Remove unused code, Add a missing package.json dependency, Add support for React DevTools.
24.	16.3.0	29/03/2018	Add a new officially supported context API, Add new packagePrevent an infinite loop when attempting to render portals with SSR, Fix an issue with this.state, Fix an IE/Edge issue.
25.	16.3.1	03/04/2018	Prefix private API, Fix performance regression and error handling bugs in development mode, Add peer dependency, Fix a false positive warning in IE11 when using Fragment.
26.	16.3.2	16/04/2018	Fix an IE crash, Fix labels in User Timing measurements, Add a UMD build, Improve performance of unstable_observedBits API with nesting.
27.	16.4.0	24/05/2018	Add support for Pointer Events specification, Add the ability to specify propTypes, Fix reading context, Fix the getDerivedStateFromProps() support, Fix a testInstance.
28.	16.5.0	05/09/2018	Add support for React DevTools Profiler, Handle errors in more edge cases gracefully, Add react-dom/profiling, Add onAuxClick event for browsers, Add movementX and movementY fields to mouse events, Add tangentialPressure and twist fields to pointer event.

29.	16.6.0	23/10/2018	Add support for contextType, Support priority levels, continuations, and wrapped callbacks, Improve the fallback mechanism, Fix gray overlay on iOS Safari, Add React.lazy() for code splitting components.
30.	16.7.0	20/12/2018	Fix performance of React.lazy for lazily-loaded components, Clear fields on unmount to avoid memory leaks, Fix bug with SSR, Fix a performance regression.
31.	16.8.0	06/02/2019	Add Hooks, Add ReactTestRenderer.act() and ReactTestUtils.act() for batching updates, Support synchronous thenables passed to React.lazy(), Improve useReducer Hook lazy initialization API.
32.	16.8.6	27/03/2019	Fix an incorrect bailout in useReducer(), Fix iframe warnings in Safari DevTools, Warn if contextType is set to Context.Consumer instead of Context, Warn if contextType is set to invalid values.

React environment

In this section, we will learn how to set up an environment for the successful development of ReactJS application.

1. NodeJS and NPM
2. React and React DOM
3. Webpack
4. Babel

There are two ways to set up an environment for successful ReactJS application. They are given below.

1. Using the npm command
2. Using the create-react-app command

Create react app:

Starting a new React project is very complicated, with so many build tools. It uses many dependencies, configuration files, and other requirements such as Babel, Webpack, ESLint before writing a single line of React code. Create React App CLI tool removes all that complexities and makes React app simple. For this, you need to install the package using NPM, and then run a few simple commands to get a new React project.

The **create-react-app** is an excellent tool for beginners, which allows you to create and run React project very quickly. It does not take any configuration manually. This tool is wrapping all of the required dependencies like **Webpack**, **Babel** for React project itself and then you need to focus on writing React code only. This tool sets up the development environment, provides an excellent developer experience, and optimizes the app for production.

The Create React App is maintained by **Facebook** and can works on any **platform**, for example, macOS, Windows, Linux, etc. To create a React Project using create-react-app, you need to have installed the following things in your system.

1. Node version >= 8.10
2. NPM version >= 5.6

MODULES

5.1 USERS

Anyone can register through the registration page. After a successful registrationuser can log in with valid email and password. User can recover own password by providing some registered info.

After successful login user can do the following things–

- Car Booking
- View Car booking history
- Update His/Her profile
- Update his/her password
- View details of car
- Logout

Module Description

The system after careful analysis has been identified to be presented with the **User** module

- **Register** – 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.
- **Login** – Once they have registered they need to login to avail the service at the needy time.
- **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 Admin.

5.2 ADMIN

Admin is the super user of the website who can manage everything on thewebsite.

Admin Features–

- Admin can create car brands
- Manage Car Brands(Edit, Delete)
- Post Car
- Manage car(Edit,Delete)
- Manage Booking(Admin can confirm and Cancel Booking)
- Manage Contact us Query
- Admin Can the details of registered users
- admin can also update the page content
- Admin can update the contact us details
- Manage Subscribers
- Admin Dashboard(Admin can view the count of reg users, total booking, total subscribers, total queries etc)
- Change Password(admin can change own password)
- logout

5.3 GUEST USERS

Guest user can view the website and checkout the information about rental cars. Guest users can also inquiry through contact us page.

SYSTEM DESIGN

Design is the first step in the development phase for any techniques and principles for the purpose of defining a device, a process or system in sufficient detail to permit its physical realization. Once the software requirements have been analysed and specified the software design involves three technical activities - design, coding, implementation and testing that are required to build and verify the software.

The design activities are of main importance in this phase, because in this activity, decisions ultimately affecting the success of the software implementation and its ease of maintenance are made. These decisions have the final bearing upon reliability and maintainability of the system. Design is the only way to accurately translate the customer's requirements into finished software or a system. Design is the place where quality is fostered in development. Software design is a process through which requirements are translated into a representation of software. Software design is conducted in two steps. Preliminary design is concerned with the transformation of requirements into data.

6.1 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a language for specifying, visualizing and documenting the system. This is the step while developing any product after analysis. The goal from this is to produce a model of the entities involved in the project which later need to be built. The representation of the entities that are to be used in the product being developed need to be designed.

There are various kinds of methods in software design:

- **Use case Diagram**
- **Class diagram**
- **Sequence Diagram**
- **Activity diagram**
- **Dataflow diagram**

6.1.1 USE CASE DIAGRAMS

Use case diagrams model behavior within a system and helps the developers understand of what the user require. The stick man represents what's called an actor.Use case diagram can be useful for getting an overall view of the system and clarifying who can do and more importantly what they can't do.

1. 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.
	Booking 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 besend to the customer once a feedback has been submitted.

Admin	Add a new car	This use case is used by the staff to add new car to the company's fleet database. Staff willneed to login to activate this use case.
Update cardetails		This use case is used by the staff to edit and modify car details whenever there is newrenewal (insurance, road tax). It allows the company to keep up-to-date record of their fleet.
	Reply to customer's feedback	This use case described the event by which staff updates the system when customer pickup or when returning car.

Table 1: Actors and Use Case Description

2. Use-case Login

Use-case Number	UC-01	
Use-case Name	Login	
Actor	Customer	
Description	This use case describe how user login into this online car rental system	
Precondition	None	
Post condition	If the use case was successful, the actor is now logged into the application	
Basic course of Action	**User Action**	**System Response**
	1. The user is on the home page to 2.Tlogin to the system 3.The user enters username and 4.The syson login button.	he system promotes the user to enter Username,Password. tem verifies that all the Password,clickfilled out and valid he system successfully logged in The system. Use case exit.

Table no - 2 : Use case - Login

3. Use-case Booking Car

Use-case Number	UC-02	
Use-Case Name	Booking car	
Customer Description	This use case permits customers to Booking and make schedule for rentingcar, based on the availability of the car	
Precondition	Customer wants to Booking a car and reservation details aboutcustomer have to be entered	
Post-condition	Customers Booking successfully	
Basic Course of	User Action	System Response
Action	1. The customer wants to Booking a car. 2. The customer clicks booking page. 4. The customer enters the following information customer (full name, email address, password, Pickup date& return date) 5. The customer clicks Bookingbutton to Booking. 8. The customer accepts the reservation and clicks Accept.	3. The system prompts the customer to fill a reservation form. 6. The system checks all required information had been filled and the date entered dates are valid 7. The system presents information to accept or decline the rental Agreement. 9. The system shows the customer that the reservation has been completed, and presents the customer a reservation

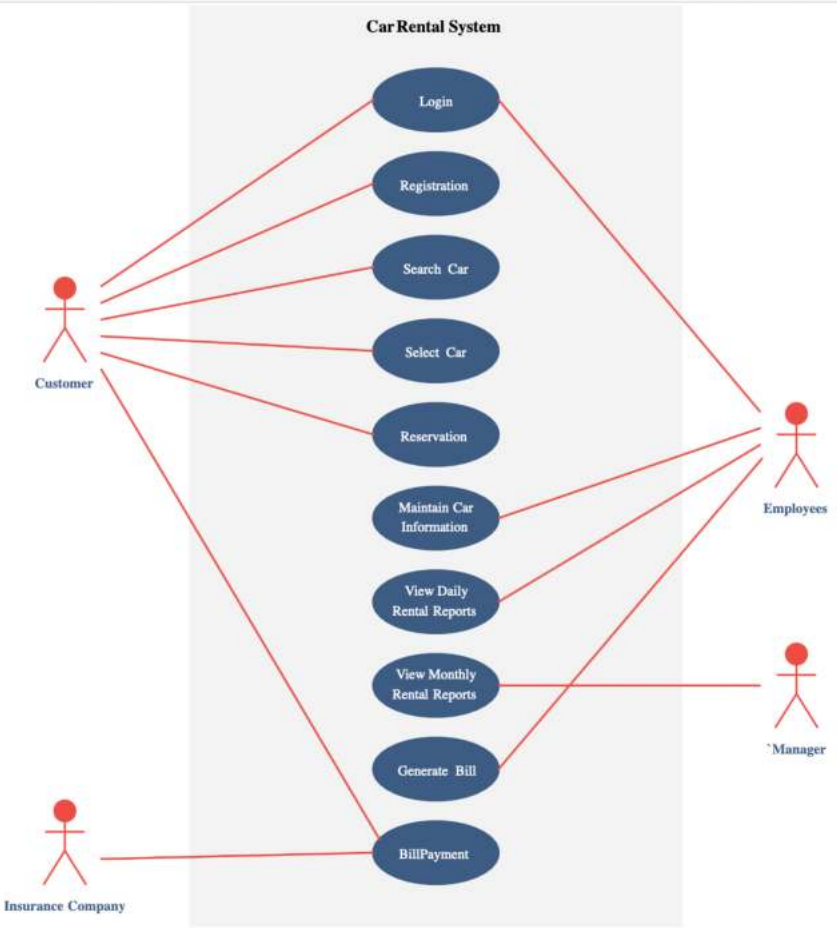
Table No. 3: Use Case Booking Car

4. Use Case View My booking

Use-Case Number	UC-03	
Use-Case Name	My booking	
Actor	User	
Description	These use case allow staff to view or display customer reservati on.	
Precondition	UC-1	
Post Condition	Display All Bookings	
Basic Course of Action	1. The staff wants to view reservation. 2. The staff requests the reservation Page. 4. Then on reservation page the employee clicks view button	3. The system responds the requested page. 5. The system puts on view or displays all reservation information to the employee. 6. Use case ends

Table No. 4 : Use Case View My booking

5. Use Case Diagram



Online Car Rental System [use case]

6.1.2 CLASS DIAGRAM

Car Rental System Class Diagram describes the structure of a Car Rental System classes, their attributes, operations (or methods), and the relationships among objects. The main classes of

the Car Rental System are Cars, Booking, Passenger, Car Routes, Drivers, ##Class6##

Classes of Car Rental System Class Diagram:

Cars Class : Manage all the operations of Cars

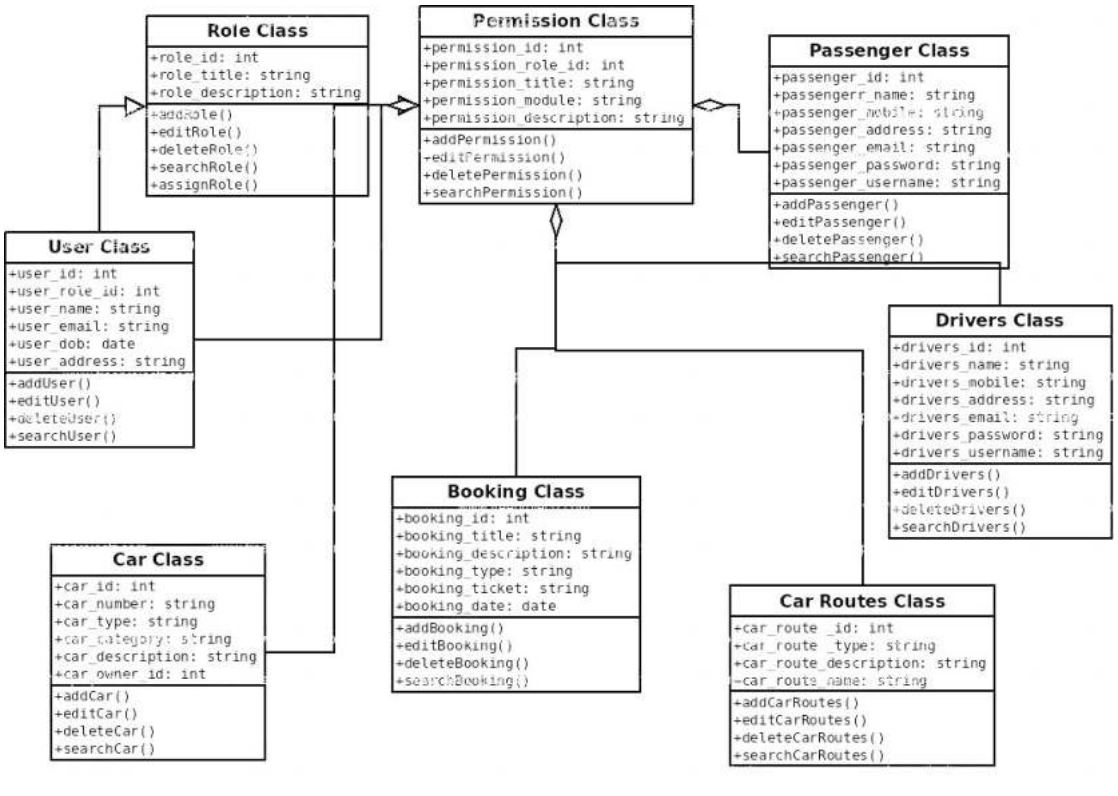
Booking Class : Manage all the operations of Booking

Passenger Class : Manage all the operations of Passenger

Car Routes Class : Manage all the operations of Car Routes

Drivers Class : Manage all the operations of Drivers

##Class6## Class : Manage all the operations of ##Class6##

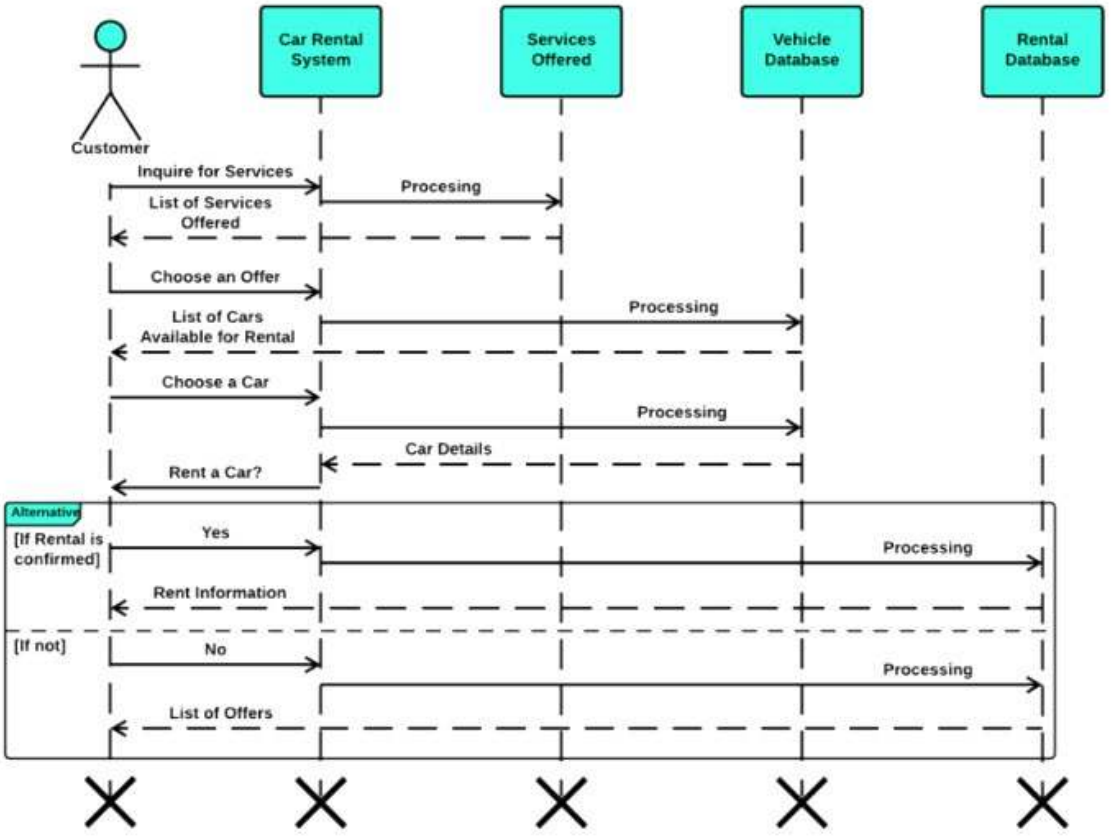


Class Diagram of Car Rental System

6.1.3 SEQUENCE DIAGRAM

Sequence diagram and collaboration diagram are called INTERACTION DIAGRAMS. An interaction diagram shows an interaction, consisting of set of objects and their relationship including the messages that may be dispatched among them.

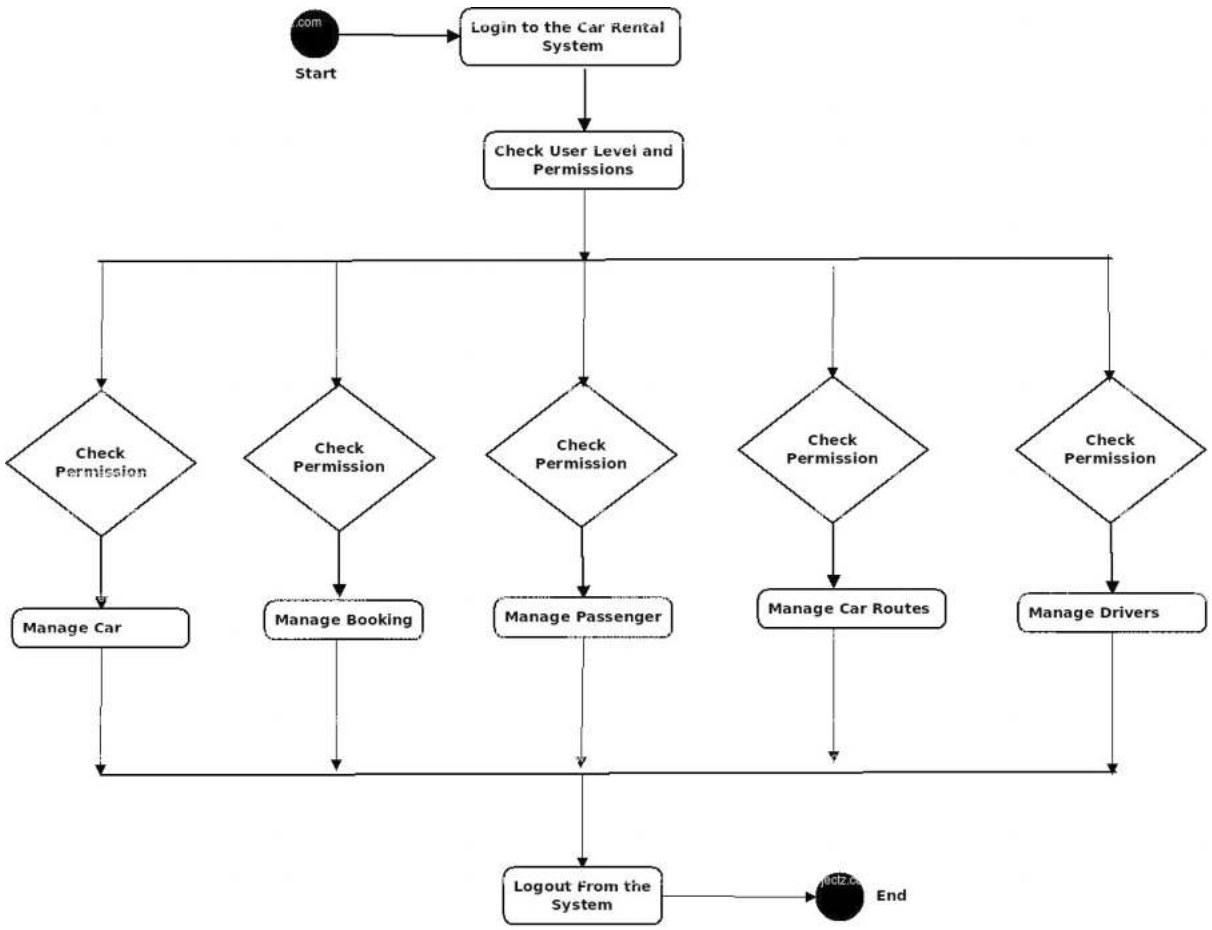
A sequence diagram is an introduction that empathizes the time ordering of messages. Graphically a sequence diagram is a table that shows objects arranged along the X-axis and messages ordered in increasing time along the Y-axis.



6.1.4 ACTIVITY DIAGRAM

This is the Activity UML diagram of Car Rental System which shows the flows between the activity of Booking, Cars, Drivers, Car Routes, Passengar. The main activity involved in this UML Activity Diagram of Car Rental System are as follows:

- Booking Activity
- Cars Activity
- Drivers Activity
- Car Routes Activity
- Passengar Activity



Activity Diagram for Car Rental System

6.2 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.

- **Zero Level Data Flow Diagram**

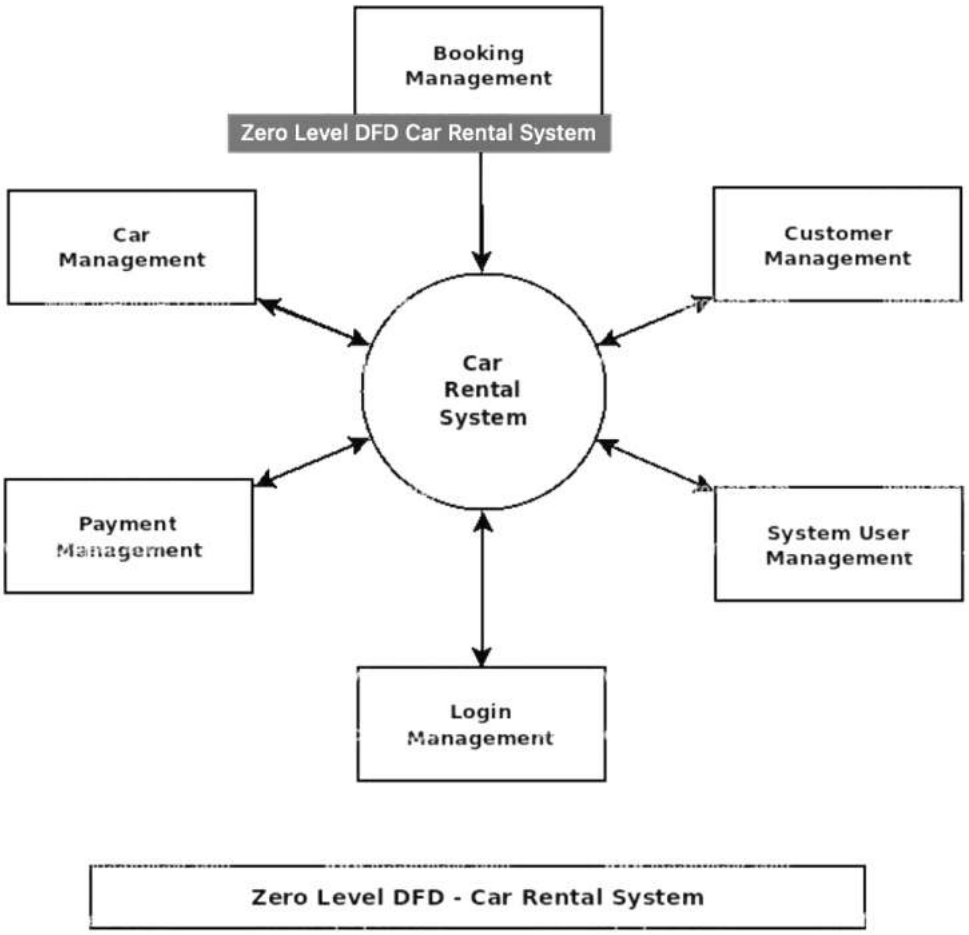


Figure 1: Zero level DFD

Zero Level DFD of online car rental system, it elaborate high level process of online car rental system. It is overview of whole online car rental system there are some high level entities for the process of car rental system.

- **First Level Data Flow Diagram**

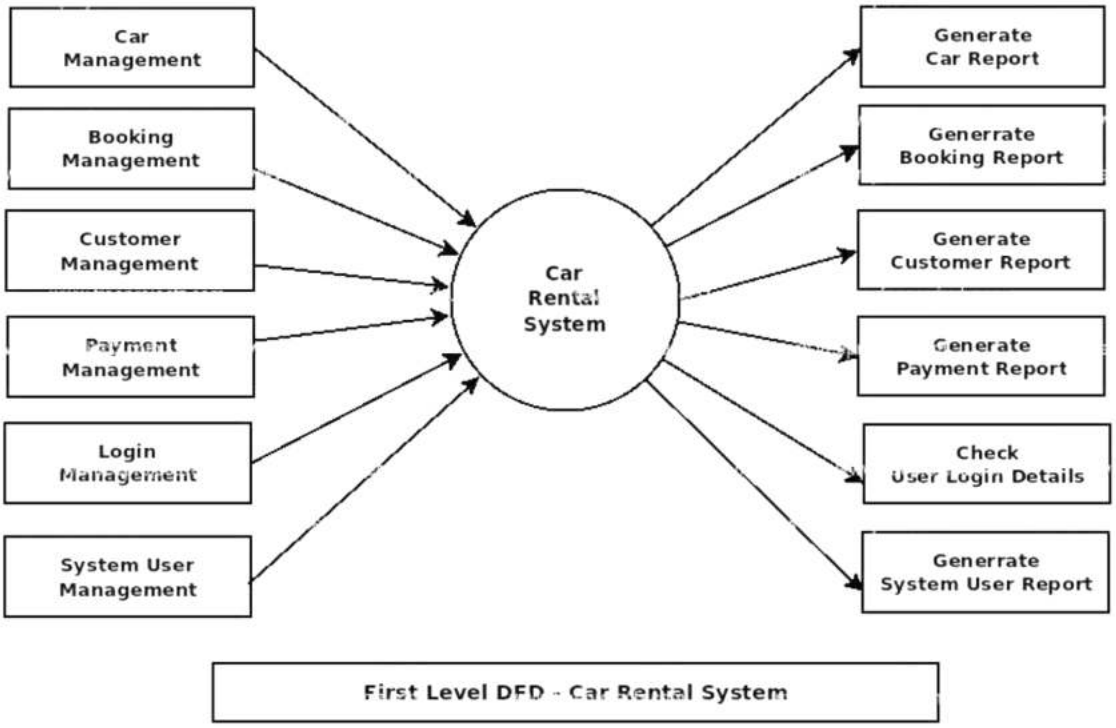


Figure 2 : 1st level DFD

1st Level DFD of online car rental system shows how the system is divided into sub system, each of which deals with one or more of the data flows to or from an external agent which together provide all the functionality of online car rental system as whole, above are some given entities and output of 1st level.

- **Second Level Data Flow Diagram**

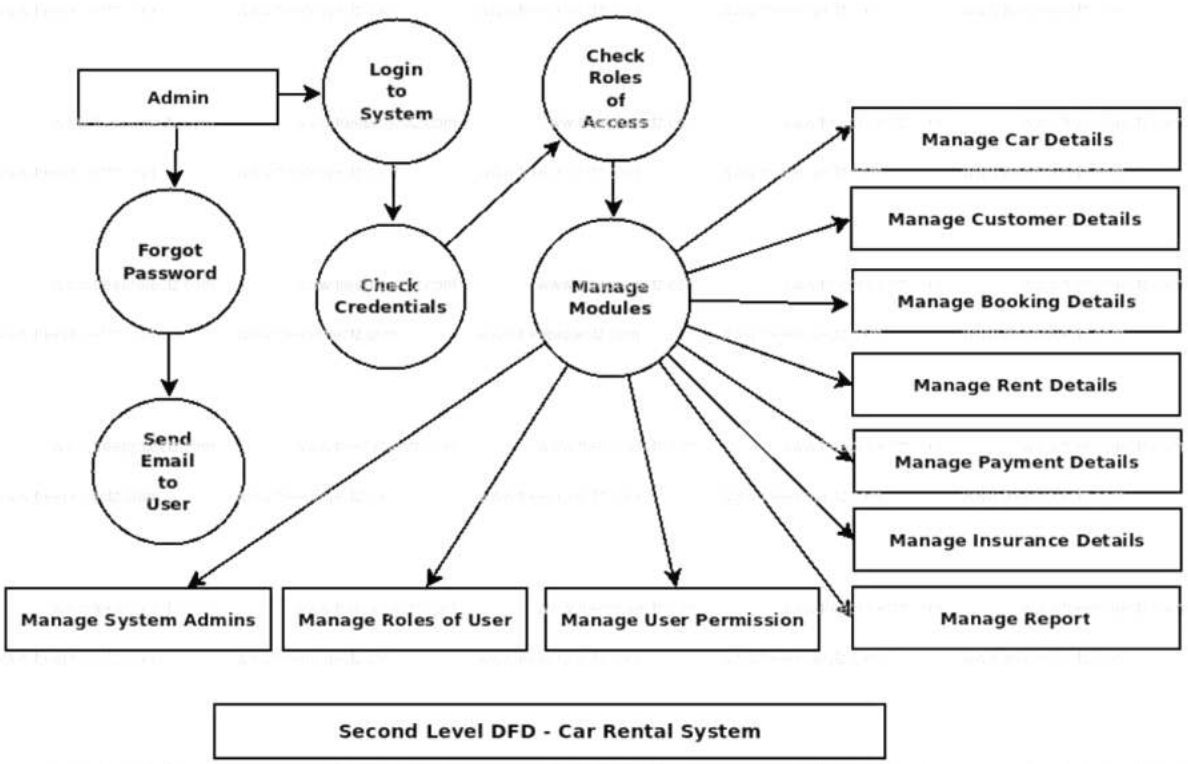
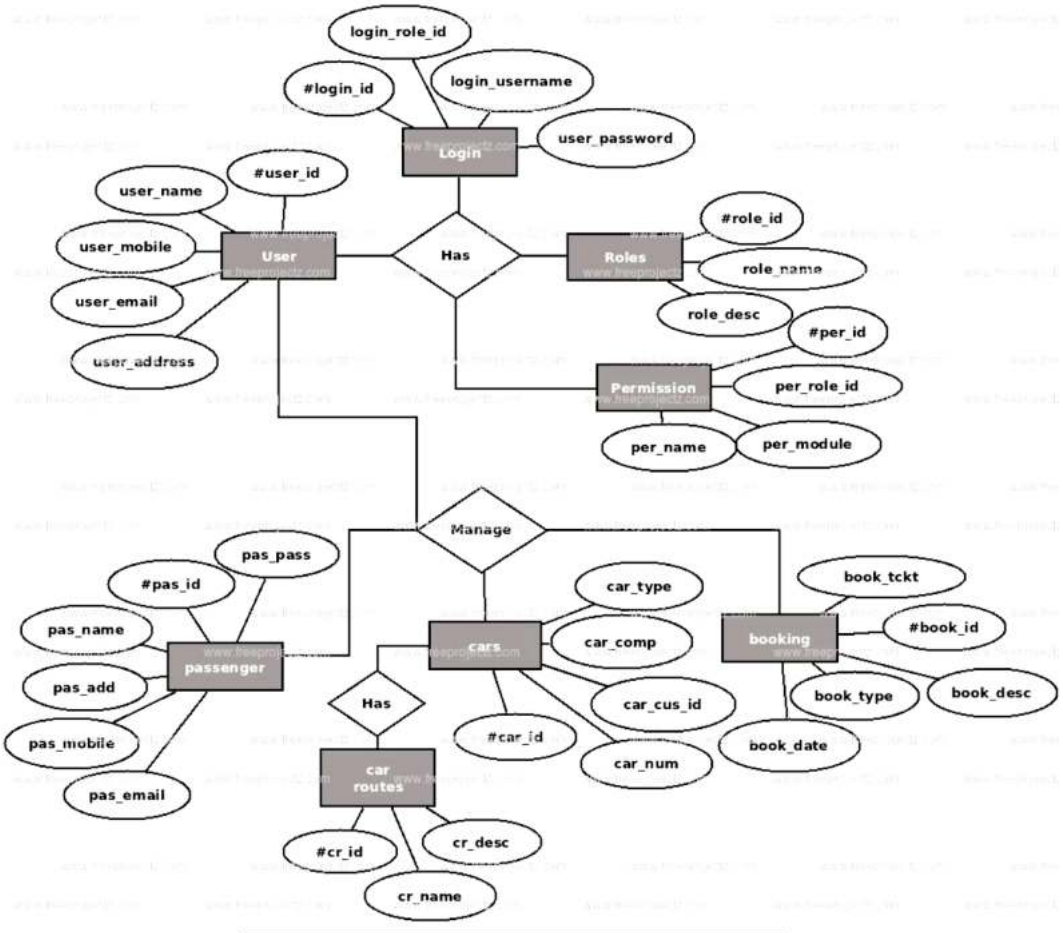


Figure 3: 2nd level DFD

DFD Level 2 then goes one step deeper into parts of Level 1 of Car Rental. It may require more functionalities of Car Rental to reach the necessary level of detail about the Car Rental functioning. First Level DFD (1st Level) of Car Rental Svstem shows how the system is divided into sub-systems (processes). The 2nd Level DFD contains more details of Bill. Rent. Insurance, Payment, Booking, Customer, Car.

6.3 ER-DIAGRAM

This ER (Entity Relationship) Diagram represents the model of Car Rental System Entity. The entity-relationship diagram of Car Rental System shows all the visual instrument of database tables and the relations between Booking, Car Routes, Cars, ##keyword6## etc. It used structure data and to define the relationships between structured data groups of Car Rental System functionalities. The main entities of the Car Rental System are Cars, Booking, Passenger, Car Routes, Drivers and ##keyword6##



SOURCE CODE

Home page/js code

```
import React from 'react';

import './css/home.css';

import banner from './images/banner.PNG';

import photo8 from './images/photo8.jpg';

import photo4 from "./images/photo4.jpg";

import photo5 from "./images/photo5.jpg";

import photo from "./images/photo.jpg";

import ban from "./images/ban.PNG";

import {Nav,NavLink,NavMenu} from './NavbarElements';

import {Carousel,Card,CardGroup,Form} from 'react-bootstrap';

import {FontAwesomeIcon} from '@fortawesome/react-fontawesome';

import { faGasPump, faCoins, faGift, faLocationArrow, faRoad,
faCarCrash,faIdCard,faUnlock,faCar,faRoute,faStar ,faSmile,faTachometerAlt}
from "@fortawesome/free-solid-svg-icons";


class Home extends React.Component {

    render(){

        return(

                <div>

                <Nav>

                    <NavLink to="/" className="logo">

                        <img src={banner} alt='logo' />

                    </NavLink>

                    <NavMenu>

                        <NavLink to='/' activeStyle>Home</NavLink>

                        <NavLink to='/offers' activeStyle>Offer</NavLink>
```

```
                        <NavLink to='/subscription'
activeStyle>Subscription</NavLink>

                        <NavLink to='/about' activeStyle>About</NavLink>

                        <NavLink to='/services' activeStyle>Services</
NavLink>

                        <NavLink to='/contact' activeStyle>Contact</
NavLink>

                        <NavLink to='/login' activeStyle>Login</NavLink>

                        <NavLink to='/register' activeStyle>Register</
NavLink>

                        <NavLink to='/admin' activeStyle>Admin</NavLink>


                       </NavMenu>

                </Nav>

                </div>

                <div>

                    <div className="full-box">

                        <img src={photo8} alt="logo" height="100%"
width="100%"/>

                    </div>

                </div>

                <br></br>

                <div className="advantage">

                    <h1 style={{textAlign:'center'}}>The Car Advantages</
h1>

                    <p style={{textAlign:'center'}}>We simplified car
rentals, so you can focus on what's important to you.</p>

                    <CardGroup>

                      <Card>

                      <FontAwesomeIcon icon={faGasPump} size="7x"
pull="right" />

                        <Card.Body>

                          <Card.Title>Fuel Cost Included</Card.Title>
```

```
                        <Card.Text>
                        Don't worry about mileage! All fuel costs are
included. If you refill fuel, we'll pay you back!
                        </Card.Text>
                      </Card.Body>
                    </Card>
                    <Card>
                    <FontAwesomeIcon icon={faCoins} size="7x"
pull="right" />
                      <Card.Body>
                        <Card.Title>No Hidden Charges</Card.Title>
                        <Card.Text>
                        Our prices include taxes and insurance.What you
see is what you really pay!
                        </Card.Text>
                      </Card.Body>
                    </Card>
                    <Card>
                    <FontAwesomeIcon icon={faGift} size="7x" pull="right"
/>
                      <Card.Body>
                        <Card.Title>Flexi Pricing Packages</Card.Title>
                        <Card.Text>
                        One size never fits all! Choose a balance of time
and kilometers that works best for you.
                        </Card.Text>
                      </Card.Body>
                    </Card>
                  </CardGroup>
                  <CardGroup>
                    <Card>
```

```
                    <FontAwesomeIcon icon={faLocationArrow} size="7x"
pull="right" />
                      <Card.Body>
                        <Card.Title>Go Anywhere</Card.Title>
                        <Card.Text>
                        Our cars have all-India permits. Just remember to
pay state tolls and entry taxes.
                        </Card.Text>
                      </Card.Body>
                    </Card>
                    <Card>
                    <FontAwesomeIcon icon={faRoad} size="7x" pull="right"
/>
                      <Card.Body>
                        <Card.Title>24x7 Roadside Assistance</Card.Title>
                        <Card.Text>
                        We have round-the-clock, pan India partners. Help
is never far away from you.
                        </Card.Text>
                      </Card.Body>
                    </Card>
                    <Card>
                    <FontAwesomeIcon icon={faCarCrash} size="7x"
pull="right" />
                      <Card.Body>
                        <Card.Title>Damage Insurance</Card.Title>
                        <Card.Text>
                        All your bookings include damage insurance! Drive
safe, but don’t worry!
                        </Card.Text>
                      </Card.Body>
                    </Card>
```

```
        </CardGroup>
      </div>
    <div>
      <Carousel>
       <Carousel.Item>
       <img src={photo5} width="100%" height="100%"></img>
       <Carousel.Caption>
       <h3>First slide label</h3>
       <p></p>
       </Carousel.Caption>
      </Carousel.Item>
      <Carousel.Item>
      <img src={photo4} width="100%" height="100%"></img>
          <Carousel.Caption>
          <h3>Second slide label</h3>
          <p></p>
          </Carousel.Caption>
      </Carousel.Item>
      <Carousel.Item>
      <img src={photo} width="100%" height="100%"></img>
          <Carousel.Caption>
          <h3>Third slide label</h3>
          <p></p>
          </Carousel.Caption>
      </Carousel.Item>
      </Carousel>
      <br></br>
      <br></br>
      </div>
      <div className="row" >
```

```
                    <div className="col-3 img-fluid">
                    <FontAwesomeIcon icon={faCar} size="7x" pull="center" /
>
                    <br></br>
                          <h3>Book</h3>
                          <p>Search for and book a car on our site.</p>
                    </div>
                    <div className="col-3 img-fluid">
                    <FontAwesomeIcon icon={faIdCard} size="7x"
pull="center" />
                    <br></br>
                    <h3>UPLOAD LICENSE</h3>

 <p>Upload your driver’s license, and pay a small security deposit.</p>
                    </div>
                    <div className="col-3 img-fluid">
                    <FontAwesomeIcon icon={faUnlock} size="7x"
pull="center" />
                    <br></br>
                    <h3>UNLOCK</h3>
                    <p>We SMS your car details 20 minutes before
pickup.Unlock it via the Zoomcar app.</p>
                    </div>
                    <div className="col-3 img-fluid">
                    <FontAwesomeIcon icon={faRoute} size="7x" pull="center"
/>
                    <br></br>
                    <h3>RETURN</h3>
                    <p>Return the car to the same location and fill the end
checklist to end your trip.</p>
                    </div>
                </div>
```

```
                <div>
                <div class="text-center">
                  <button type="button" class="btn btn-primary"><NavLink
to="/register">Sign Up</NavLink></button>
                </div>
                 </div>
                <div>
                    <img src={ban} width="100%"></img>
                </div>
                <br></br>
                <div className="main-box">
                  <div className="text-big3">
                <div className="row" >
                    <div className="col-4 img-fluid">
                    <FontAwesomeIcon icon={faSmile} size="7x" pull="center"
/>
                    <br></br>
                          <h3>48,00,000+</h3>
                          <p>Happy Users</p>
                    </div>
                    <div className="col-4 img-fluid">
                    <FontAwesomeIcon icon={faTachometerAlt} size="7x"
pull="center" />
                    <br></br>

  <h3>36,00,000+</h3>
                    <p>Km Travelled(enough for 470 round trips to the
moon!)</p>
                    </div>
                    <div className="col-4 img-fluid">
                    <FontAwesomeIcon icon={faStar} size="7x"
pull="center" />
```

```
                    <br></br>

                    <h3>RATING 4.5/5</h3>

                    <p>Rated by 3,00,000+ customers over 10,00,000+
bookings</p>

                    </div>

                </div>

                </div>

                </div>

                <div className="main-box1">

                  <Form>

                    <h3>LOOKING FOR SELF DRIVE CARS IN BANGALORE?</h3>

                    <p>Bangalore being a burgeoning cosmopolitan city is
witnessing a shift in culture. With youngsters now being able to establish
an equal focus on work and leisure frequent road trips have now become a
norm. In addition to this, the added pleasure of a self-driving has given
rise in the demand for, self-drive car rentals which has emerged as the
answer to many of their prayers.</p>

                    <h3>HOW MUCH DOES IT COST TO RENT A CAR FOR A DAY?</h3>

                    <p>If want to rent one out a self-drive car, choose
Zoomcar, where you can pick out your self-drive car based on your
travelling needs. While small self-drive cars allow you to easily manoeuvre
through city traffic, sedans and SUV's offer more comfort. Affordable
prices starting at 60/hr helps you rent a car easily. Whether you want to
go on a road trip from Bangalore to Coorg or just need a car to travel
within the city the self-drive Zoomcar are your best solution. All Self-
drive Zoomcar comes with a limited liability, enabling you to enjoy the
comfort of a four-wheeler without having to invest in one, with the added
benefit of not having to actually drive it. The Self-drive Zoomcar have
even made travel to and from the airport easy. Avoid exorbitantly priced
Bangalore airport cabs by choosing to travel in a Zoomcar instead. The
Bangalore airport even has a Zoomcar parking lot where you can get a self-
drive car rental as soon as you get off your flight, thus eliminating the
time taken to look for a cab or hire one.</p>

                    <h3>WHAT STATES ALLOW SELF DRIVING CARS?</h3>

                    <p>Drive seamlessly:<br></br>
```

```
                    Our Self-Drive cars come with all India permits,
Road-Side Assistance (RSA), vehicle insurance and provision of multiple
parking sites across the city.</p>

                  <p>Book Through the App:<br></br>

                    No matter how long you need our Self Drive cars for
you can conveniently hire a Self-Drive car online in Bangalore through the
Zoomcar app.</p>

                  <p>Array of services:<br></br>

                    Enjoy tech-enabled booking, tracking and hassle-free
payment options too. Make every special occasion memorable by opting to
ride in a luxury self-drive car yourself. You could opt for it just for the
sheer pleasure and experience too!</p>

                  <h3>STEPS FOR SELF DRIVE CAR BOOKING IN BANGALORE</h3>


 <p>Drive across the city hassle free with a self-drive car from Zoomcar in
the following steps:</p>

                  <ol>

                    <li>Download the Zoomcar App from the Play Store or
the App Store.</li><br></br>

                    <li>Search for a car from the wide range available
and book a self-drive car of your choice</li><br></br>

                    <li>Upload License and pay a small security deposit</
li><br></br>

                    <li>Car details will be sent on SMS 20mins before
pickup. Unlock it via the Zoomcar App</li><br></br>

                    <li>Fill the start checklist in the Zoomcar App. Grab
the keys from the glove box and zoom away</li><br></br>

                      <p>After the trip, return the car and fill the end
checklist from the App to end your trip. We refund your deposit if
applicableBook, Drive, Repeat. Zoom On.</p>

                  </ol>

                </Form>

              </div>

              <br></br>

            </>    } } } export default Home;
```

About page/js code

```
import React from 'react';
import './css/home.css';
import banner from './images/banner.PNG';
import photo8 from './images/photo8.jpg';
import {Nav,NavLink,NavMenu} from './NavbarElements';
import {Form} from 'react-bootstrap';


class About extends React.Component {
    render(){
        return(
            <>
                <div>
                <Nav>
                    <NavLink to="/" className="logo">
                        <img src={banner} alt='logo' />
                    </NavLink>
                    <NavMenu>
                        <NavLink to='/' activeStyle>Home</NavLink>
                        <NavLink to='/offer' activeStyle>Offer</NavLink>
                        <NavLink to='/subscription'
activeStyle>Subscription</NavLink>
                        <NavLink to='/about' activeStyle>About</NavLink>
                        <NavLink to='/services' activeStyle>Services</
NavLink>
                        <NavLink to='/contact' activeStyle>Contact</
NavLink>
                        <NavLink to='/login' activeStyle>Login</NavLink>
                        <NavLink to='/register' activeStyle>Register</
NavLink>
```

```
                        <NavLink to='/admin' activeStyle>Admin</NavLink>
                    </NavMenu>
                </Nav>
                </div>
                <div>
                    <div className="full-box">
                        <img src={photo8} alt="logo" height="100%"
width="100%"/>
                    </div>
                </div>
                <div>
                  <Form>
                    <h3>LOOKING FOR SELF DRIVE CARS IN BANGALORE?</h3>


  <p>Bangalore being a burgeoning cosmopolitan city is witnessing a shift
in culture. With youngsters now being able to establish an equal focus on
work and leisure frequent road trips have now become a norm. In addition to
this, the added pleasure of a self-driving has given rise in the demand
for, self-drive car rentals which has emerged as the answer to many of
their prayers.</p>

                    <h3>HOW MUCH DOES IT COST TO RENT A CAR FOR A DAY?</h3>

                    <p>If want to rent one out a self-drive car, choose
Zoomcar, where you can pick out your self-drive car based on your
travelling needs. While small self-drive cars allow you to easily manoeuvre
through city traffic, sedans and SUV's offer more comfort. Affordable
prices starting at 60/hr helps you rent a car easily. Whether you want to
go on a road trip from Bangalore to Coorg or just need a car to travel
within the city the self-drive Zoomcar are your best solution. All Self-
drive Zoomcar comes with a limited liability, enabling you to enjoy the
comfort of a four-wheeler without having to invest in one, with the added
benefit of not having to actually drive it. The Self-drive Zoomcar have
even made travel to and from the airport easy. Avoid exorbitantly priced
Bangalore airport cabs by choosing to travel in a Zoomcar instead. The
Bangalore airport even has a Zoomcar parking lot where you can get a self-
drive car rental as soon as you get off your flight, thus eliminating the
time taken to look for a cab or hire one.</p>

                    <h3>WHAT STATES ALLOW SELF DRIVING CARS?</h3>
```

```
                    <p>Drive seamlessly:<br></br>

                      Our Self-Drive cars come with all India permits,
Road-Side Assistance (RSA), vehicle insurance and provision of multiple
parking sites across the city.</p>

                    <p>Book Through the App:<br></br>

                      No matter how long you need our Self Drive cars for
you can conveniently hire a Self-Drive car online in Bangalore through the
Zoomcar app.</p>

                    <p>Array of services:<br></br>

                      Enjoy tech-enabled booking, tracking and hassle-free
payment options too. Make every special occasion memorable by opting to
ride in a luxury self-drive car yourself. You could opt for it just for the
sheer pleasure and experience too!</p>

                    <h3>STEPS FOR SELF DRIVE CAR BOOKING IN BANGALORE</h3>

                    <p>Drive across the city hassle free with a self-drive
car from Zoomcar in the following steps:</p>

                    <ol>

                      <li>Download the Zoomcar App from the Play Store or
the App Store.</li><br></br>

                      <li>Search for a car from the wide range available
and book a self-drive car of your choice</li><br></br>

                      <li>Upload License and pay a small security deposit</
li><br></br>

                      <li>Car details will be sent on SMS 20mins before
pickup. Unlock it via the Zoomcar App</li><br></br>

                      <li>Fill the start checklist in the Zoomcar App. Grab
the keys from the glove box and zoom away</li><br></br>

                        <p>After the trip, return the car and fill the end
checklist from the App to end your trip. We refund your deposit if
applicableBook, Drive, Repeat. Zoom On.</p>

                    </ol>

                  </Form>

                </div>


              <br></br>
```

```
                <br></br>
                <div>
                   <div className="footer-wrapper">
                     <h4>About Us</h4>
                     <ul class="Container" style={{justifyContent:
"normal"}}>
                       <li>Zoomcar Team</li><span></span>
                       <li>Zoomcar Subscription</li><span></span>
                       <li>Zoomcar Blog</li><span></span>
                       <li>Careers @ Zoomcar</li><span></span>
                       <li>Location & Cars</li><span></span>
                       <li>Self Drive Cars</li><span></span>
                       <li>Invest via OurCrowd</li><span></span>
                     </ul>
                     <h4>Cities</h4>
                     <ul className="Container">
                       <li><span>Self Drive Cars in Bangalore</span></li>
                       <li><span>Self Drive Cars in Pune</span></li>
                       <li>Self Drive Cars in Delhi</li><span></span>
                       <li>Self Drive Cars in Chennai</li><span></span>
                       <li>Self Drive Cars in Hyderabad</li><span></span>
                       <li>Self Drive Cars in Chandigarh</li><span></span>
                       <li>Slef Drive Cars in Ahmedabad</li><span></span>
                       <li>Slef Drive Cars in Coimbatore</li><span></span>
                       <li>Slef Drive Cars in Indore</li><span></span>
                       <li>Self Drive Cars in Ludhiana</li><span></span>
                       <li>Self Drive Cars in Mysore</li><span></span>
                       <li>Slef Drive Cars in Nagpur</li><span></span>
                     </ul>
                     <h4>Airport Services</h4>
```

```
                    <ul className="Container">
                      <li>Car Rental From Bangalore Airport</li>
                      <li>Car Rental From Pune Airport</li>
                      <li>Car Rental From Delhi Airport</li>
                      <li>Car Rental From Chennai Airport</li>
                      <li>Car Rental From Hyderabad Airport</li>
                      <li>Car Rental From Chadigarh Airport</li>
                      <li>Car Rental From Ahmedabad Airport</li>

                     <li>Car Rental From Coimbatore Airport</li>
                      <li>Car Rental From Chandigarh Airport</li>
                      <li>Car Rental From Chennai Airport</li>
                      <li>Car Rental From Indore Airport</li>
                      <li>Car Rental From Vizag Airport</li>
                    </ul>
                  </div>
                </div>
            </>
        )
    }
}

export default About;
```

Service Page/js code

```
import React from 'react';
import './css/home.css';
import banner from './images/banner.PNG';
import photo8 from './images/photo8.jpg';
import {Nav,NavLink,NavMenu} from './NavbarElements';


class Services extends React.Component {
    render(){
        return(
            <>
                <div>
                <Nav>
                    <NavLink to="/" className="logo">
                        <img src={banner} alt='logo' />
                    </NavLink>
                    <NavMenu>
                        <NavLink to='/' activeStyle>Home</NavLink>
                        <NavLink to='/offers' activeStyle>Offer</NavLink>
                        <NavLink to='/subscription'
activeStyle>Subscription</NavLink>
                        <NavLink to='/about' activeStyle>About</NavLink>
                        <NavLink to='/services' activeStyle>Services</
NavLink>
                        <NavLink to='/contact' activeStyle>Contact</
NavLink>
                        <NavLink to='/login' activeStyle>Login</NavLink>
                        <NavLink to='/register' activeStyle>Register</
NavLink>
                        <NavLink to='/admin' activeStyle>Admin</NavLink>
```

```
                    </NavMenu>
                </Nav>
                </div>
                <div>
                    <div className="full-box">
                        <img src={photo8} alt="logo" height="100%"
width="100%"/>
                    </div>
                </div>
                <div>
                   <div className="footer-wrapper">
                     <h4>About Us</h4>
                     <ul class="Container" style={{justifyContent:
"normal"}}>
                       <li>Zoomcar Team</li><span></span>



                       <li>Zoomcar Subscription</li><span></span>
                       <li>Zoomcar Blog</li><span></span>
                       <li>Careers @ Zoomcar</li><span></span>
                       <li>Location & Cars</li><span></span>
                       <li>Self Drive Cars</li><span></span>
                       <li>Invest via OurCrowd</li><span></span>
                     </ul>
                     <h4>Cities</h4>
                     <ul className="Container">
                       <li><span>Self Drive Cars in Bangalore</span></li>
                       <li><span>Self Drive Cars in Pune</span></li>
                       <li>Self Drive Cars in Delhi</li><span></span>
                       <li>Self Drive Cars in Chennai</li><span></span>
                       <li>Self Drive Cars in Hyderabad</li><span></span>
```

```
                <li>Self Drive Cars in Chandigarh</li><span></span>
                <li>Slef Drive Cars in Ahmedabad</li><span></span>
                <li>Slef Drive Cars in Coimbatore</li><span></span>
                <li>Slef Drive Cars in Indore</li><span></span>
                <li>Self Drive Cars in Ludhiana</li><span></span>
                <li>Self Drive Cars in Mysore</li><span></span>
                <li>Slef Drive Cars in Nagpur</li><span></span>
              </ul>
              <h4>Airport Services</h4>
              <ul className="Container">
                <li>Car Rental From Bangalore Airport</li>
                <li>Car Rental From Pune Airport</li>
                <li>Car Rental From Delhi Airport</li>
                <li>Car Rental From Chennai Airport</li>
                <li>Car Rental From Hyderabad Airport</li>
                <li>Car Rental From Chadigarh Airport</li>
                <li>Car Rental From Ahmedabad Airport</li>
                <li>Car Rental From Coimbatore Airport</li>
                <li>Car Rental From Chandigarh Airport</li>
                <li>Car Rental From Chennai Airport</li>
                <li>Car Rental From Indore Airport</li>
                <li>Car Rental From Vizag Airport</li>
              </ul>
            </div>
          </div>
      </>
) } } export default Services;
```

Login Page/js code

```
import React from 'react';
import './css/home.css';
import "./css/adminLogin.css";
import banner from './images/banner.PNG';
import { faKey, faEnvelope } from "@fortawesome/free-solid-svg-icons";
import {Nav,NavLink,NavMenu} from './NavbarElements';
import {FontAwesomeIcon} from '@fortawesome/react-fontawesome';


class Login extends React.Component {
  constructor(props) {
    super();
    this.onChangeAdminEmail = this.onChangeAdminEmail.bind(this);
    this.onChangeAdminPassword = this.onChangeAdminPassword.bind(this);
    this.onSubmit = this.onSubmit.bind(this);
    this.state = {
      adminEmail: "",
      adminPassword: "",
      msg: "",
    };
  }

  onChangeAdminEmail(e) {
    this.setState({ adminEmail: e.target.value });
  }
  onChangeAdminPassword(e) {
    this.setState({ adminPassword: e.target.value });
  }
  onSubmit(e) {
```

```
    e.preventDefault();
    console.log("Login Successfull");
    console.log(`ADMIN EMAIL ID : ${this.state.adminEmail}`);
    console.log(`ADMIN PASSWORD : ${this.state.adminPassword}`);
    if (
      this.state.adminEmail === "admin@gmail.com" &&
      this.state.adminPassword === "admin1234 )
{
      this.setState({ msg: "WELCOME ADMIN" });
      sessionStorage.setItem("Key_Veriable", "admin@gmail.com");
      this.props.history.push("/adminafterlogin");
    } else {
      this.setState({ msg: "INVALID" });
    }
    this.setState({ adminEmail: "", adminPassword: "" });
  }
    render(){
        return(
            <>
                <div>
                <Nav>
                    <NavLink to="/" className="logo">
                        <img src={banner} alt='logo' />
                    </NavLink>
                    <NavMenu>
                        <NavLink to='/' activeStyle>Home</NavLink>
                        <NavLink to='/offer' activeStyle>Offer</NavLink>
                        <NavLink to='/subscription'
activeStyle>Subscription</NavLink>
                        <NavLink to='/about' activeStyle>About</NavLink>
```

```
                        <NavLink to='/services' activeStyle>Services</
NavLink>
                        <NavLink to='/contact' activeStyle>Contact</
NavLink>
                        <NavLink to='/login' activeStyle>Login</NavLink>
                        <NavLink to='/register' activeStyle>Register</
NavLink>
                        <NavLink to='/admin' activeStyle>Admin</NavLink>
                    </NavMenu>
                </Nav>
                </div>
                <div className="container">
            <div className="signin-container">
                <h1 style={{color:'green'}}>Login Here</h1>
                <h4 className="msg">{this.state.msg}</h4>
                <form onSubmit={this.onSubmit}>
                <div className="box">
                <FontAwesomeIcon icon={faEnvelope} />

               <input
                type="text"
                name="adminEmail"
                id="Email"
                placeholder="Email"
                value={this.state.adminEmail}
                onChange={this.onChangeAdminEmail}
                />
                </div>
                <div className="box">
                <FontAwesomeIcon icon={faKey} />
                <input
```

```
                type="password"
                name="adminPassword"
                id="Password"
                placeholder="Password"
                value={this.state.adminPassword}
                onChange={this.onChangeAdminPassword}
                />
                </div>
                <button className="btn-signin" style={{color:'green'}}
><NavLink to="/dashboard">Log in</NavLink></button>
                </form>
            </div>
            </div>
            </>
        )
    }
}

export default Login;
```

Register Page/js code

```
import React from 'react';
import './css/home.css';
import "./css/adminLogin.css";
import banner from './images/banner.PNG';
import {Nav,NavLink,NavMenu} from './NavbarElements';
import {FontAwesomeIcon} from '@fortawesome/react-fontawesome';
import { faKey, faEnvelope, faUser } from "@fortawesome/free-solid-svg-
icons";


class Register extends React.Component {
  constructor(props) {
    super();
    this.onChangeUser =this.onChangeUser.bind(this);
    this.onChangeEmail = this.onChangeEmail.bind(this);
    this.onChangePassword = this.onChangePassword.bind(this);
    this.onChangeCPassword = this.onChangeCPassword.bind(this);
    this.onSubmit = this.onSubmit.bind(this);
    this.state = {
      User:"",
      Email: "",
      Password: "",
      CPassword:"",
      msg: "",
    };
  }

  onChangeUser(e){
    this.setState({ User: e.target.value});
```

```
  }

  onChangeEmail(e) {
    this.setState({ Email: e.target.value });
  }
  onChangePassword(e) {
    this.setState({ Password: e.target.value });
  }
  onChangeCPassword(e) {
    this.setState({ CPassword: e.target.value });
  }
onSubmit(e) {
    e.preventDefault();
    console.log("Login Successfull");
    console.log(`USER NAME: ${this.state.User}`);
    console.log(`EMAIL ID : ${this.state.Email}`);
    console.log(`PASSWORD : ${this.state.Password}`);
    console.log(`CPASSWORD : ${this.state.CPassword}`);
    if (
      this.state.Email === "admin@gmail.com" &&
      this.state.Password === "admin1234"
    ) {
      this.setState({ msg: "WELCOME ADMIN" });
      sessionStorage.setItem("Key_Veriable", "admin@gmail.com");
      this.props.history.push("/adminafterlogin");
    } else {
      this.setState({ msg: "INVALID" });
    }
    this.setState({ Email: "", Password: "" });
  }
```

```
    render(){
        return(
            <>
                <div>
                <Nav>
                    <NavLink to="/" className="logo">
                        <img src={banner} alt='logo' />
                    </NavLink>
                    <NavMenu>
                        <NavLink to='/' activeStyle>Home</NavLink>
                        <NavLink to='/offer' activeStyle>Offer</NavLink>
                        <NavLink to='/subscription'
activeStyle>Subscription</NavLink>
                        <NavLink to='/about' activeStyle>About</NavLink>
                        <NavLink to='/services' activeStyle>Services</
NavLink>
                        <NavLink to='/contact' activeStyle>Contact</
NavLink>
                        <NavLink to='/login' activeStyle>Login</NavLink>
                        <NavLink to='/register' activeStyle>Register</
NavLink>
                        <NavLink to='/admin' activeStyle>Admin</NavLink>
                    </NavMenu>
                </Nav>
                </div>

            <div className="container">
            <div className="signin-container">
                <h1 style={{color:'green'}}>Register Here</h1>
                <h4 className="msg">{this.state.msg}</h4>
                <form onSubmit={this.onSubmit}>
```

```
<div className="box">
<FontAwesomeIcon icon={faUser} />
<input
type="text"
name="User"
id="User"
placeholder="User Name"
value={this.state.User}
onChange={this.onChangeUser}
/>
</div>
<div className="box">
<FontAwesomeIcon icon={faEnvelope} />
<input
type="text"
name="Email"
id="Email"
placeholder="Email"
value={this.state.Email}
onChange={this.onChangeEmail}
/>
</div>
<div className="box">
<FontAwesomeIcon icon={faKey} />
<input
type="Password"
name="Password"
id="Password"
placeholder="Password"
value={this.state.Password}
```

```
                onChange={this.onChangePassword}
                />
                </div>



                <div className="box">
                <FontAwesomeIcon icon={faKey} />
                <input
                type="password"
                name="CPassword"
                id="CPassword"
                placeholder="Confirm Password"
                value={this.state.CPassword}
                onChange={this.onChangeCPassword}
                />
                </div>
                <button className="btn-signin" style={{color:'green'}}
><NavLink to="/login">Create Account</NavLink></button>
                </form>
            </div>
            </div>
            </>
        )
    }
}


export default Register;
```

Admin Page/js code

```
import React from 'react';
import './css/home.css';
import "./css/adminLogin.css";
import banner from './images/banner.PNG';
import {Nav,NavLink,NavMenu} from './NavbarElements';
import {FontAwesomeIcon} from '@fortawesome/react-fontawesome';
import { faKey, faEnvelope } from "@fortawesome/free-solid-svg-icons";

class Admin extends React.Component {
  constructor(props) {
    super();
    this.onChangeAdminEmail = this.onChangeAdminEmail.bind(this);
    this.onChangeAdminPassword = this.onChangeAdminPassword.bind(this);
    this.onSubmit = this.onSubmit.bind(this);
    this.state = {
      adminEmail: "",
      adminPassword: "",
      msg: "",
    };
  }

  onChangeAdminEmail(e) {
    this.setState({ adminEmail: e.target.value });
  }
  onChangeAdminPassword(e) {
    this.setState({ adminPassword: e.target.value });
  }
  onSubmit(e) {
```

```
    e.preventDefault();
    console.log("Login Successfull");
    console.log(`ADMIN EMAIL ID : ${this.state.adminEmail}`);
    console.log(`ADMIN PASSWORD : ${this.state.adminPassword}`);
    if (
      this.state.adminEmail === "admin@gmail.com" &&
      this.state.adminPassword === "admin1234")
{
      this.setState({ msg: "WELCOME ADMIN" });
      sessionStorage.setItem("Key_Veriable", "admin@gmail.com");
      this.props.history.push("/adminafterlogin");
    } else {
      this.setState({ msg: "INVALID" });
    }
    this.setState({ adminEmail: "", adminPassword: "" });
  }
    render(){
        return(
            <>
                <div>
                <Nav>
                    <NavLink to="/" className="logo">
                        <img src={banner} alt='logo' />
                    </NavLink>
                    <NavMenu>
                        <NavLink to='/' activeStyle>Home</NavLink>
                        <NavLink to='/offer' activeStyle>Offer</NavLink>
                        <NavLink to='/subscription'
activeStyle>Subscription</NavLink>
                        <NavLink to='/about' activeStyle>About</NavLink>
```

```
                        <NavLink to='/services' activeStyle>Services</
NavLink>
                        <NavLink to='/contact' activeStyle>Contact</
NavLink>
                        <NavLink to='/login' activeStyle>Login</NavLink>
                        <NavLink to='/register' activeStyle>Register</
NavLink>
                        <NavLink to='/admin' activeStyle>Admin</NavLink>
                    </NavMenu>
                </Nav>
                </div>
                <div className="container">
            <div className="signin-container">
                <h1 style={{color:'green'}}>Welcome Back</h1>
                <h4 className="msg">{this.state.msg}</h4>
                <form onSubmit={this.onSubmit}>
                <div className="box">
                <FontAwesomeIcon icon={faEnvelope} />

                <input
                type="text"
                name="adminEmail"
                id="Email"
                placeholder="Email"
                value={this.state.adminEmail}
                onChange={this.onChangeAdminEmail}
                />
                </div>
                <div className="box">
                <FontAwesomeIcon icon={faKey} />
                <input
```

```
                type="password"
                name="adminPassword"
                id="Password"
                placeholder="Password"
                value={this.state.adminPassword}
                onChange={this.onChangeAdminPassword}
                />
                </div>
                <button className="btn-signin">Log in</button>
                </form>
            </div>
            </div>
            </>
        )
    }
}

export default Admin;
```

Contact Page/js code

```
import React from 'react';
import './css/home.css';
import banner from './images/banner.PNG';
import photo8 from './images/photo8.jpg';
import { Form } from 'react-bootstrap';
import {Button} from 'react-bootstrap';
import {Row} from 'react-bootstrap';
import {Col} from 'react-bootstrap';
import {Nav,NavLink,NavMenu} from './NavbarElements';


class Contact extends React.Component {
    render(){
        return(
            <>
                <div>
                <Nav>
                    <NavLink to="/" className="logo">
                        <img src={banner} alt='logo' />
                    </NavLink>
                    <NavMenu>
                        <NavLink to='/' activeStyle>Home</NavLink>
                        <NavLink to='/offer' activeStyle>Offer</NavLink>
                        <NavLink to='/subscription'
activeStyle>Subscription</NavLink>
                        <NavLink to='/about' activeStyle>About</NavLink>
                        <NavLink to='/services' activeStyle>Services</
NavLink>
                        <NavLink to='/contact' activeStyle>Contact</
NavLink>
```

```
                        <NavLink to='/login' activeStyle>Login</NavLink>
                        <NavLink to='/register' activeStyle>Register</
NavLink>
                        <NavLink to='/admin' activeStyle>Admin</NavLink>
                    </NavMenu>
                </Nav>
                </div>
                <div>
                    <div className="full-box">
                        <img src={photo8} alt="logo" height="100%"
width="100%"/>
                    </div>
                </div>
                <br></br>



                    <div>
                    <Form>
                    <Form.Group as={Row} className="mb-3"
controlId="formPlaintextEmail">
                        <Form.Label column sm="2">
                        Email
                        </Form.Label>
                        <Col sm="10">
                        <Form.Control plaintext readOnly
defaultValue="email@example.com" />
                        </Col>
                    </Form.Group>
                    <Form.Group as={Row} className="mb-3"
controlId="formPlaintextPassword">
                        <Form.Label column sm="2">
                        Contact Number
```

```
                        </Form.Label>
                        <Col sm="10">
                        <Form.Control type="number"
placeholder="9999999999" />
                        </Col>
                    </Form.Group>
                    <Button variant="primary" type="submit"><NavLink
to="/">Submit</NavLink></Button>
                    </Form>
                    <footer>
                            <h4>Let's Keep in Touch</h4>
                            <p>
                                7th Floor, Tower-B, Diamond District, 150,
HAL Airport Road, Kodihalli, Bangalore – 560008
                            </p>
                            <div className="copy-rights">
                                © Copyright 2017 HomeXRental India Private
Ltd. All rights reserved.
                            </div>
                    </footer>
                </div>
            </>
        )
    }
}

export default Contact;
```

Subscription page/js code

```
import React from 'react';

import './css/home.css';

import banner from './images/banner.PNG';

import photo8 from './images/photo8.jpg';

import car3 from './images/car3.jpg';

import car1 from './images/car1.jpg';

import car2 from './images/car2.jpg';

import car6 from './images/car6.jpg';

import car5 from './images/car5.jpg';

import {Nav,NavLink,NavMenu} from './NavbarElements';

import {Card,CardGroup} from 'react-bootstrap';

import {FontAwesomeIcon} from '@fortawesome/react-fontawesome';

import { faCar,faCalendar,faWallet} from "@fortawesome/free-solid-svg-
icons";


class Subscription extends React.Component {

    render(){

        return(

            <>

                <div>

                <Nav>

                    <NavLink to="/" className="logo">

                        <img src={banner} alt='logo' />

                    </NavLink>

                    <NavMenu>

                        <NavLink to='/' activeStyle>Home</NavLink>

                        <NavLink to='/offer' activeStyle>Offer</NavLink>

                        <NavLink to='/subscription'
activeStyle>Subscription</NavLink>

                        <NavLink to='/about' activeStyle>About</NavLink>
```

```
                        <NavLink to='/services' activeStyle>Services</
NavLink>
                        <NavLink to='/contact' activeStyle>Contact</
NavLink>
                        <NavLink to='/login' activeStyle>Login</NavLink>
                        <NavLink to='/register' activeStyle>Register</
NavLink>
                        <NavLink to='/admin' activeStyle>Admin</NavLink>
                    </NavMenu>
                </Nav>
                </div>
                <div>
                    <div className="full-box">


                     <img src={photo8} alt="logo" height="100%"
width="100%"/>
                    </div>
                </div>
                <h1 style={{textAlign:'center'}}>Top Selling Cars</h1>
                <div className="main-box">
                    <CardGroup>
                      <Card>
                      <img src={car1} alt="logo" height="100%"
width="100%"/>
                        <Card.Body>
                          <Card.Title>Go Anywhere</Card.Title>
                          <Card.Text>
                          Our cars have all-India permits. Just remember to
pay state                         tolls and entry taxes.
                          </Card.Text>
                        </Card.Body>
```

```
                    </Card>
                    <Card>
                    <img src={car2} alt="logo" height="100%"
width="100%"/>
                      <Card.Body>
                        <Card.Title>24x7 Roadside Assistance</Card.Title>
                        <Card.Text>
                        We have round-the-clock, pan India partners. Help
is never far away from you.
                        </Card.Text>
                      </Card.Body>
                    </Card>
                    <Card>
                    <img src={car3} alt="logo" height="100%"
width="100%"/>
                      <Card.Body>
                        <Card.Title>24x7 Roadside Assistance</Card.Title>
                        <Card.Text>
                        We have round-the-clock, pan India partners. Help
is never far away from you.
                        </Card.Text>
                      </Card.Body>
                    </Card>
                </CardGroup>
                </div>
                <br></br>
                <br></br>


                    <div className="main-box">
                    <div className="Row">
```

```
                        <div className="Col"
style={{float:'left',width:'50%',padding:'5px'}}>

                            <img src={car5} alt="logo" height="100%"
width="100%"/>

                        </div>

                        <div className="Col"
style={{float:'right',width:'50%',padding:'5px',fontFamily:'Ubuntu,sans-
serif'}}>

                            <h3>Why Subscribe to a Personal Car?</h3>

                            <p>A pandemic (from Greek πᾶν, pan, "all" and
δῆμος, demos, "local people" the 'crowd') is an epidemic of an infectious
disease that has spread across a large region, for instance multiple
continents or worldwide, affecting a substantial number of people. A
widespread endemic disease with a stable number of infected people is not a
pandemic. Widespread endemic diseases with a stable number of infected
people such as recurrences of seasonal influenza are generally excluded as
they occur simultaneously in large regions of the globe rather than being
spread worldwide.</p>

                        </div>

                    </div>

                </div>

                <br></br>

                <div className="main-box">

                <div className="Row">

                        <div className="Col"
style={{float:'left',width:'50%',padding:'5px',fontFamily:'Ubuntu,sans-
serif'}}>

                            <h3>Why Subscribe to a Personal Car?</h3>

                            <p>A pandemic (from Greek πᾶν, pan, "all" and
δῆμος, demos, "local people" the 'crowd') is an epidemic of an infectious
disease that has spread across a large region, for instance multiple
continents or worldwide, affecting a substantial number of people. A
widespread endemic disease with a stable number of infected people is not a
pandemic. Widespread endemic diseases with a stable number of infected
people such as recurrences of seasonal influenza are generally excluded as
they occur simultaneously in large regions of the globe rather than being
spread worldwide.</p>
```

```
                        </div>
                        <div className="Col"
style={{float:'right',width:'50%',padding:'5px'}}>
                            <img src={car6} alt="logo" height="100%"
width="100%"/>
                        </div>
                    </div>
                </div>
                <br></br>
                <div className="main-box">
                <Card style={{ width: '18rem' ,height:'15rem'}}>
                    <Card.Body>
                        <Card.Title>01</Card.Title>
                        <Card.Text>


                        <h3>Select A Car</h3>
                        <br></br>
                        <p>We offer 45+ cars in your city.Check them out
here</p>
                        </Card.Text>
                    </Card.Body>
                </Card>
                <Card style={{ width: '18rem' ,height:'15rem'}}>
                    <Card.Body>
                        <Card.Title>02</Card.Title>
                        <Card.Text>
                        <h3>Apply Referral Code</h3>
                        <br></br>
                        <p>Use referral code.Know More</p>
                        </Card.Text>
```

```
                    </Card.Body>
                </Card>
                <Card style={{ width: '18rem',height:'15rem' }}>
                    <Card.Body>
                        <Card.Title>03</Card.Title>
                        <Card.Text>
                        <h3>Pay & Book</h3>
                        <br></br>
                        <p>Choose from multiple payment options.We
recommend you to select
                            Instapay for lower security deposit!</p>
                        </Card.Text>
                    </Card.Body>
                </Card>
                <Card style={{ width: '18rem',height:'15rem' }}>
                    <Card.Body>
                        <Card.Title>04</Card.Title>
                        <Card.Text>
                        <h3>Delivery</h3>
                        <br></br>
                        <p>We will deliver the car at your doorstep</p>
                        </Card.Text>
                    </Card.Body>
                </Card>
                </div>
                <br></br>


                  <div className="main-box">
                  <div className="text-big3">
```

```
                    <div className="row" >
                        <div className="col-4 img-fluid">
                        <FontAwesomeIcon icon={faCalendar} size="3x"
pull="center" />
                        <br></br>
                            <h3>Pick Dates</h3>
                            <p>Choose the dates you want to share your car
using Zoomcar Subscription app.</p>
                        </div>
                        <div className="col-4 img-fluid">
                        <FontAwesomeIcon icon={faCar} size="3x"
pull="center" />
                        <br></br>
                        <h3>Booking</h3>
                        <p>We will allocate booking subject to your car's
according to the availability you have choosen</p>
                        </div>
                        <div className="col-4 img-fluid">
                        <FontAwesomeIcon icon={faWallet} size="3x"
pull="center" />
                        <br></br>
                        <h3>Earn</h3>
                        <p>Earning will be credited in the first week of
every month or adjusted against your monthly subscription.</p>
                        </div>
                    </div>
                  </div>
                </div>
                <br></br>
                <br></br>
                <br></br>
                <br></br>
```

```
                <br></br>
                <br></br>
                <br></br>
                <div>
                   <div className="footer-wrapper">
                     <h4>About Us</h4>
                     <ul class="Container" style={{justifyContent:
"normal"}}>
                       <li>Zoomcar Team</li><span></span>
                       <li>Zoomcar Subscription</li><span></span>


                       <li>Zoomcar Blog</li><span></span>
                       <li>Careers @ Zoomcar</li><span></span>
                       <li>Location & Cars</li><span></span>
                       <li>Self Drive Cars</li><span></span>
                       <li>Invest via OurCrowd</li><span></span>
                     </ul>
                     <h4>Cities</h4>
                     <ul className="Container">
                       <li><span>Self Drive Cars in Bangalore</span></li>
                       <li><span>Self Drive Cars in Pune</span></li>
                       <li>Self Drive Cars in Delhi</li><span></span>
                       <li>Self Drive Cars in Chennai</li><span></span>
                       <li>Self Drive Cars in Hyderabad</li><span></span>
                       <li>Self Drive Cars in Chandigarh</li><span></span>
                       <li>Slef Drive Cars in Ahmedabad</li><span></span>
                       <li>Slef Drive Cars in Coimbatore</li><span></span>
                       <li>Slef Drive Cars in Indore</li><span></span>
                       <li>Self Drive Cars in Ludhiana</li><span></span>
```

```
                <li>Self Drive Cars in Mysore</li><span></span>
                <li>Slef Drive Cars in Nagpur</li><span></span>
              </ul>
              <h4>Airport Services</h4>
              <ul className="Container">
                <li>Car Rental From Bangalore Airport</li>
                <li>Car Rental From Pune Airport</li>
                <li>Car Rental From Delhi Airport</li>
                <li>Car Rental From Chennai Airport</li>
                <li>Car Rental From Hyderabad Airport</li>
                <li>Car Rental From Chadigarh Airport</li>
                <li>Car Rental From Ahmedabad Airport</li>
                <li>Car Rental From Coimbatore Airport</li>
                <li>Car Rental From Chandigarh Airport</li>
                <li>Car Rental From Chennai Airport</li>
                <li>Car Rental From Indore Airport</li>
                <li>Car Rental From Vizag Airport</li>
              </ul>
            </div>
          </div>
      </>
) } } export default Subscription;
```

//Instruction to run this programme

 # Getting Started with Create React App

This project was bootstrapped with [Create React App](https://github.com/facebook/create-react-app).

Available Scripts

In the project directory, you can run:

`npm start`

Runs the app in the development mode.\
Open http://localhost:3000 to view it in the browser.

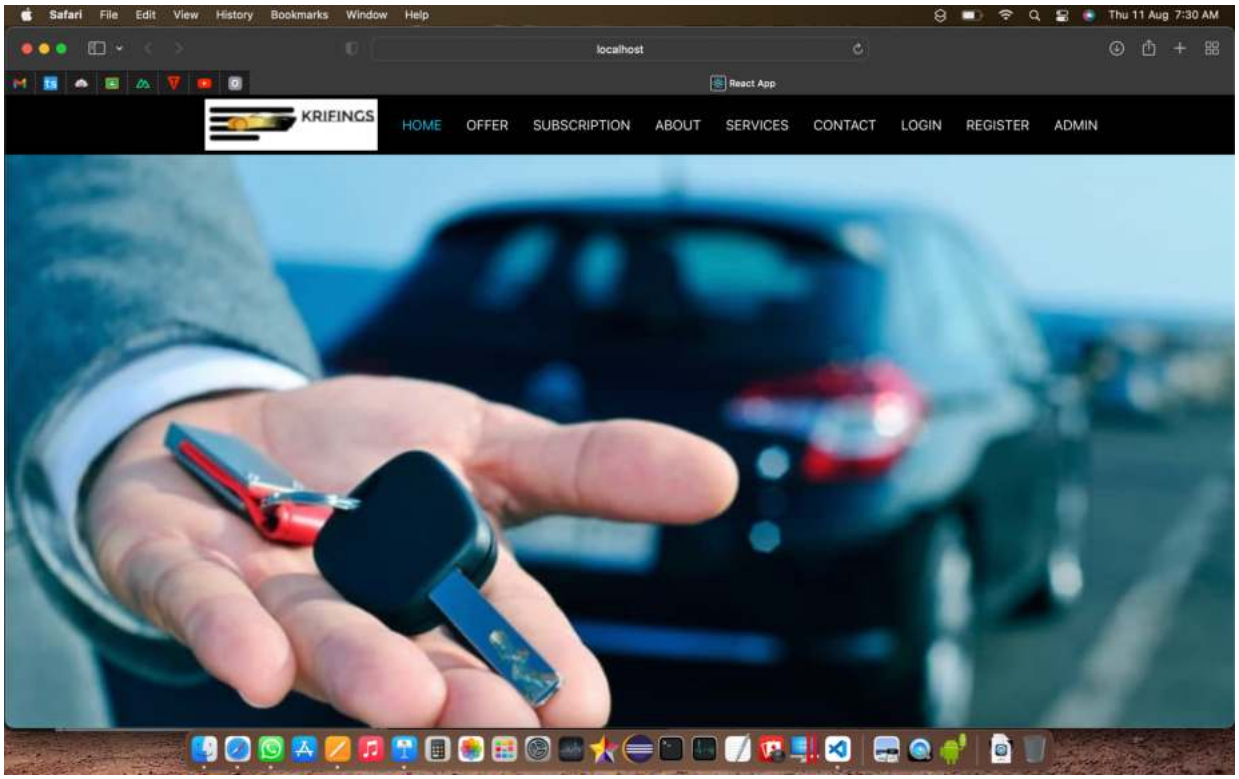
The page will reload if you make edits.\
You will also see any lint errors in the console.

`npm test`

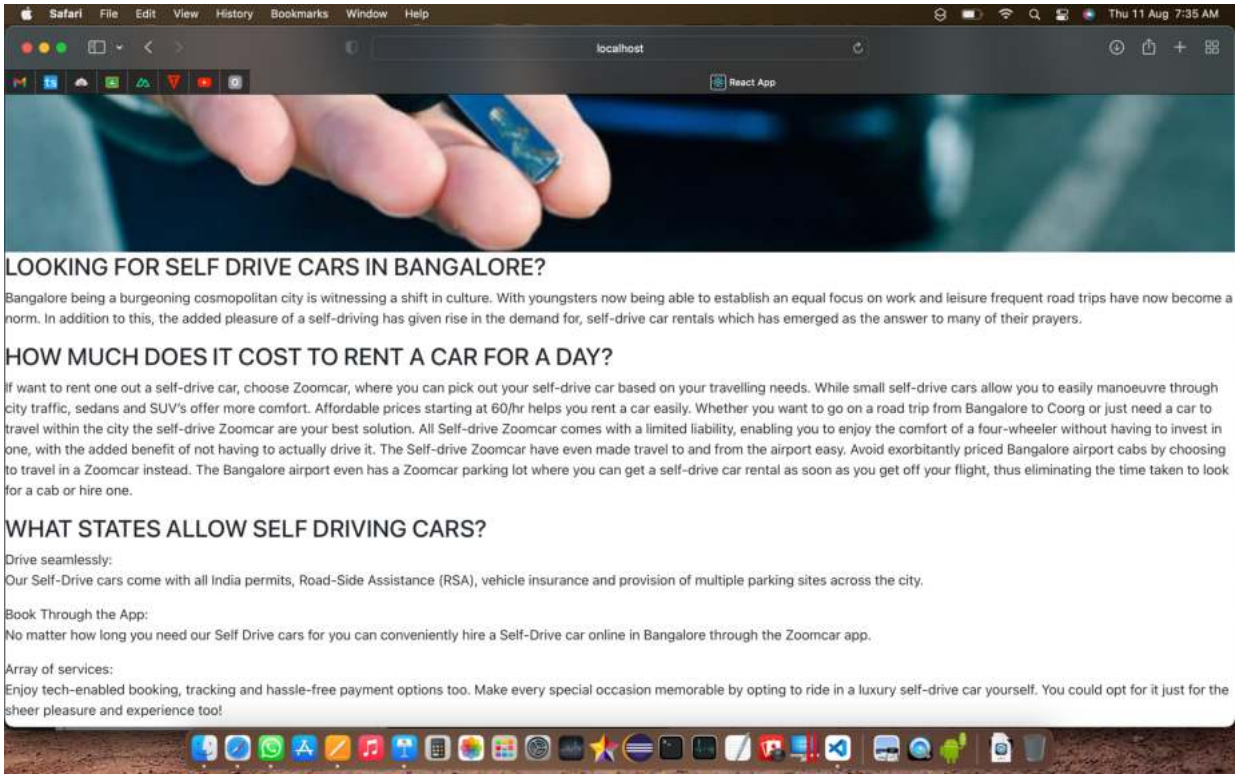
Launches the test runner in the interactive watch mode.\
See the section about [running tests](https://facebook.github.io/create-react-app/docs/running-tests) for more information.

SCREENSHOT

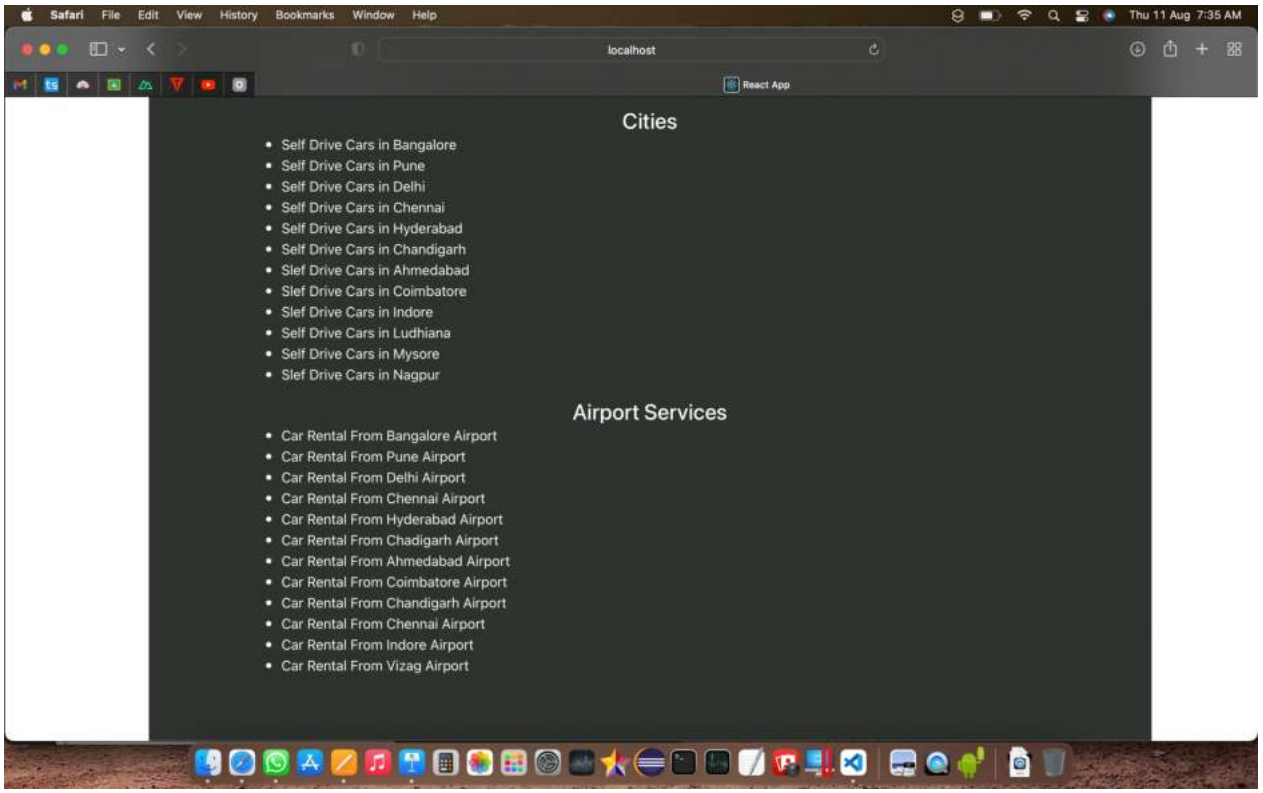
Home Page



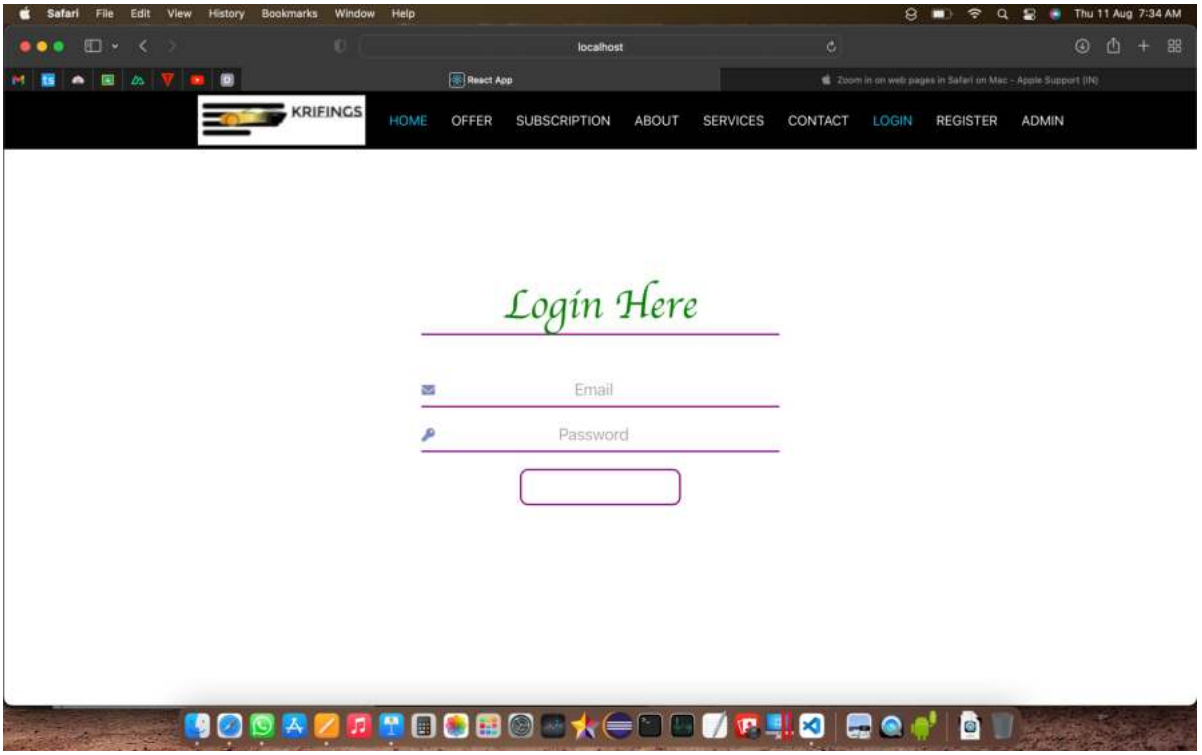
About page


Safari File Edit View History Bookmarks Window Help
Thu 11 Aug 7:35 AM
localhost
React App
LOOKING FOR SELF DRIVE CARS IN BANGALORE?
Bangalore being a burgeoning cosmopolitan city is witnessing a shift in culture. With youngsters now being able to establish an equal focus on work and leisure frequent road trips have now become a norm. In addition to this, the added pleasure of a self-driving has given rise in the demand for, self-drive car rentals which has emerged as the answer to many of their prayers.
HOW MUCH DOES IT COST TO RENT A CAR FOR A DAY?
If want to rent one out a self-drive car, choose Zoomcar, where you can pick out your self-drive car based on your travelling needs. While small self-drive cars allow you to easily manoeuvre through city traffic, sedans and SUV's offer more comfort. Affordable prices starting at 60/hr helps you rent a car easily. Whether you want to go on a road trip from Bangalore to Coorg or just need a car to travel within the city the self-drive Zoomcar are your best solution. All Self-drive Zoomcar comes with a limited liability, enabling you to enjoy the comfort of a four-wheeler without having to invest in one, with the added benefit of not having to actually drive it. The Self-drive Zoomcar have even made travel to and from the airport easy. Avoid exorbitantly priced Bangalore airport cabs by choosing to travel in a Zoomcar instead. The Bangalore airport even has a Zoomcar parking lot where you can get a self-drive car rental as soon as you get off your flight, thus eliminating the time taken to look for a cab or hire one.
WHAT STATES ALLOW SELF DRIVING CARS?
Drive seamlessly:
Our Self-Drive cars come with all India permits, Road-Side Assistance (RSA), vehicle insurance and provision of multiple parking sites across the city.
Book Through the App:
No matter how long you need our Self Drive cars for you can conveniently hire a Self-Drive car online in Bangalore through the Zoomcar app.
Array of services:
Enjoy tech-enabled booking, tracking and hassle-free payment options too. Make every special occasion memorable by opting to ride in a luxury self-drive car yourself. You could opt for it just for the sheer pleasure and experience too!

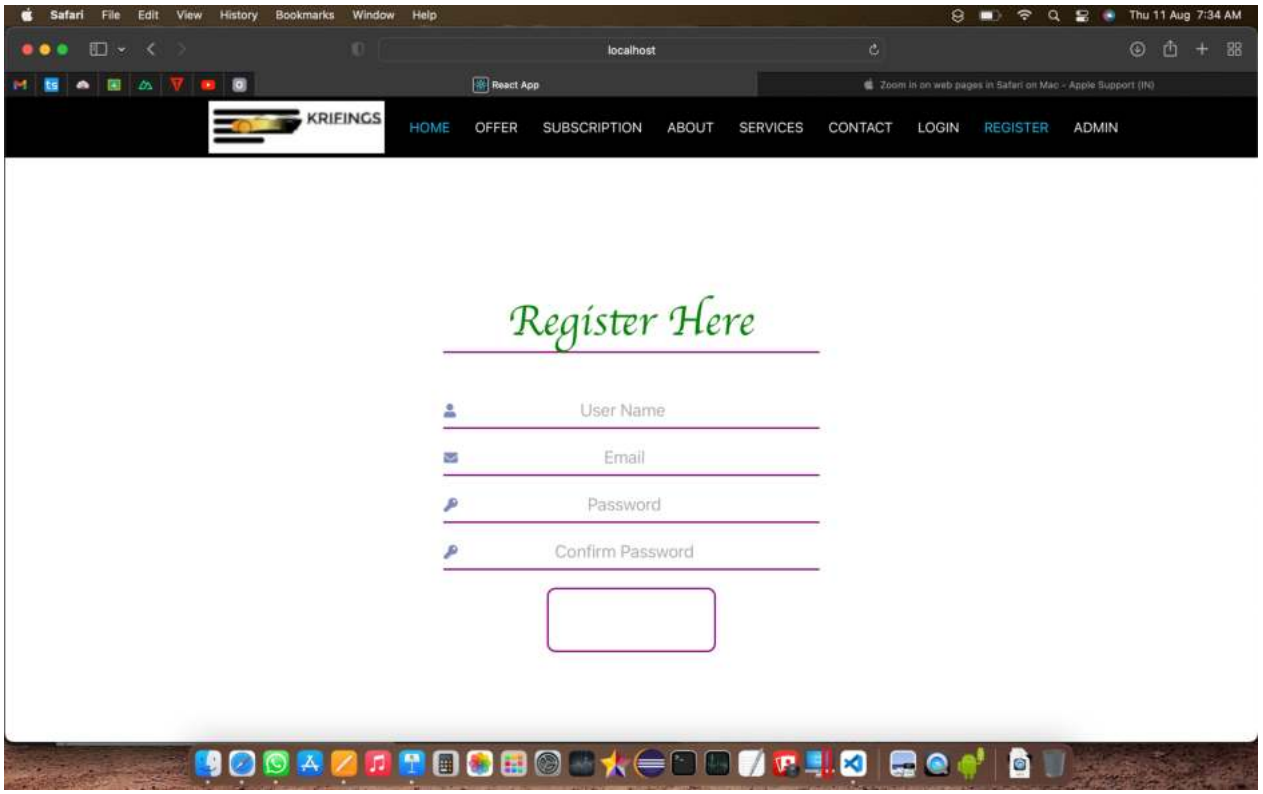
Services page



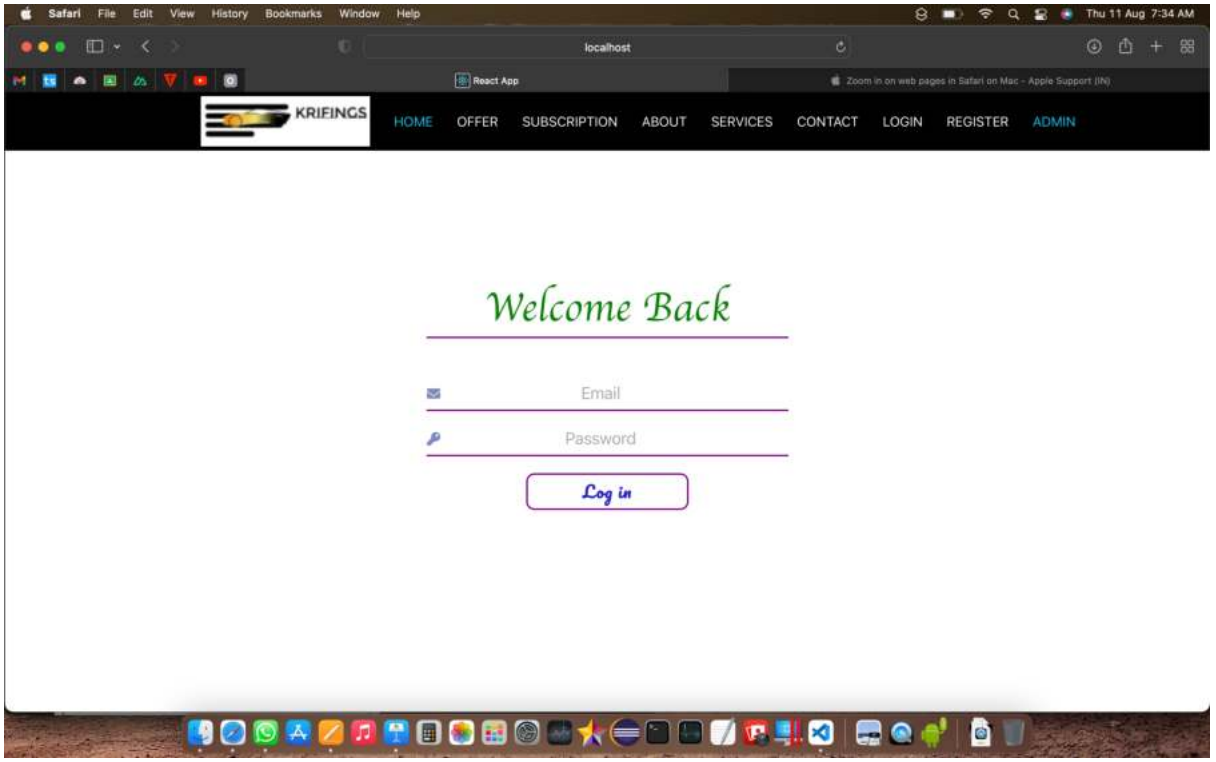
Login page


Safari File Edit View History Bookmarks Window Help
Thu 11 Aug 7:34 AM
localhost
React App
KRIFINGS
HOME
OFFER
SUBSCRIPTION
ABOUT
SERVICES
CONTACT
LOGIN
REGISTER
ADMIN
Login Here
Email
Password

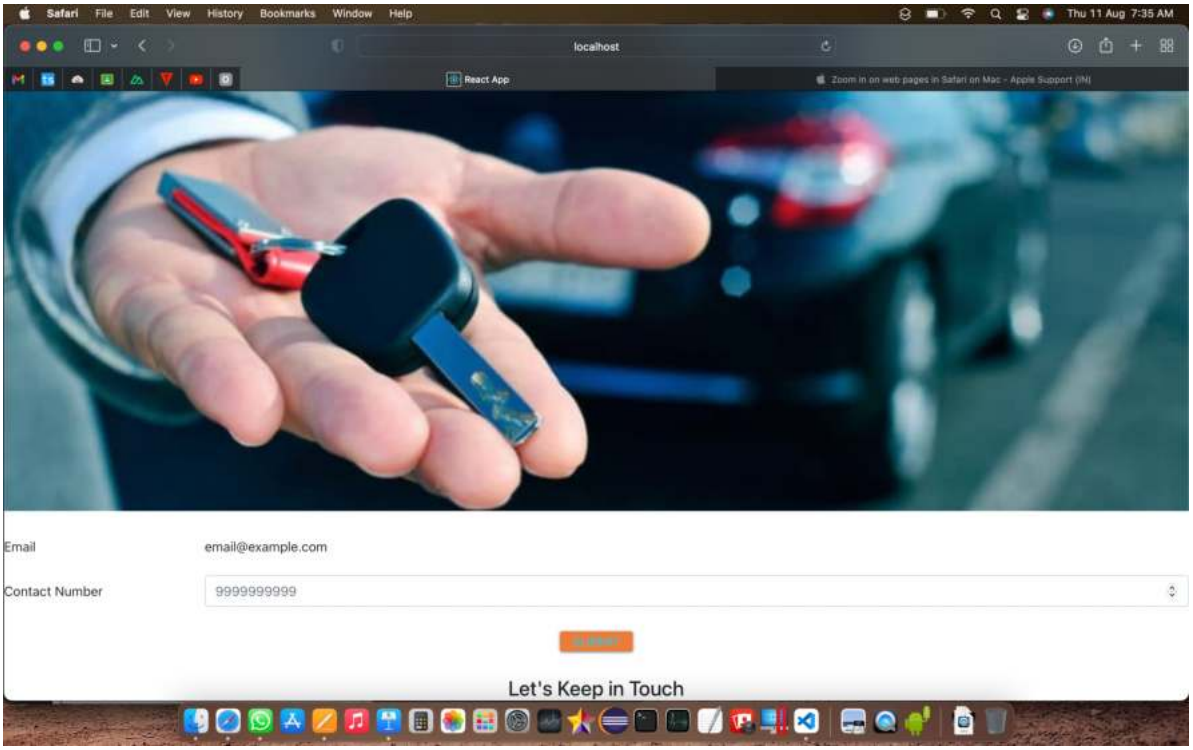
Registration Page



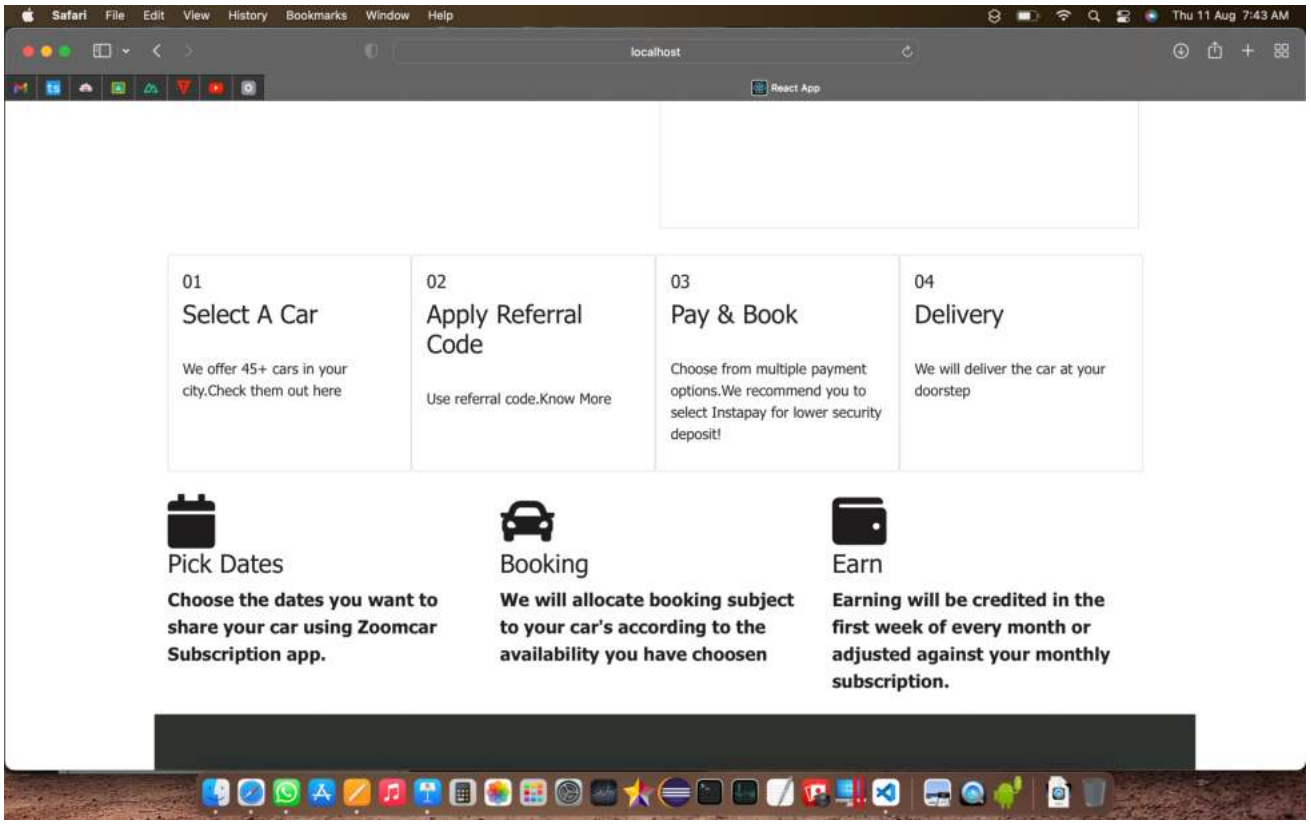
Admin Login Page



Contact Page



Subscription Page



TESTING

The various levels of testing are

1. White Box Testing
2. Black Box Testing
3. Unit Testing
4. Functional Testing
5. Performance Testing
6. Configuration testing
7. Isolation testing
8. Integration Testing
9. Validation Testing
10. System Testing
11. Structure Testing
12. Output Testing
13. User Acceptance Testing

1.WHITE BOX TESTING

White-box testing (also known as **clear box testing**, **glass box testing**, **transparent box testing**, and **structural testing**) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality (i.e. black-box testing). In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT).While white-box testing can be applied at the unit, integration and system levels of the software testing process, it is usually done at the unit level. It can test paths within a unit, paths between units during integration, and between subsystems during a system–level test. Though this method of test design can uncover many errors or problems, it might not detect unimplemented parts of the specification or missing requirements.

White-box test design techniques include:

- Control flow testing
- Data flow testing
- Branch testing
- Path testing
- Statement coverage

White-box testing is a method of testing the application at the level of the source code. The test cases are derived through the use of the design techniques mentioned above: control flow testing, data flow testing, branch testing, path testing, statement coverage and decision coverage as well as modified condition/decision coverage. White-box testing is the use of these techniques as guidelines to create an error free environment by examining any fragile code.

These White-box testing techniques are the building blocks of white-box testing, whose essence is the careful testing of the application at the source code level to prevent any hidden errors later on. These different techniques exercise every visible path of the source code to minimize errors and create an error-free environment. The whole point of white-box testing is the ability to know which line of the code is being executed and being able to identify what the correct output should be.

Levels

1. Unit testing. White-box testing is done during unit testing to ensure that the code is working as intended, before any integration happens with previously tested code. White-box testing during unit testing catches any defects early on and aids in any defects that happen later on after the code is integrated with the rest of the application and therefore prevents any type of errors later on.
2. Integration testing. White-box testing at this level are written to test the interactions of each interface with each other. The Unit level testing made sure that each code was tested and working accordingly in an isolated environment and integration examines the correctness of the behaviour in an open environment through the use of white-box testing for any interactions of interfaces that are known to the programmer.
3. Regression testing. White-box testing during regression testing is the use of recycled white-box test cases at the unit and integration testing levels.

White-box testing's basic procedures involve the understanding of the source code that you are testing at a deep level to be able to test them. The programmer must have a deep understanding of the application to know what kinds of test cases to create so that every visible path is exercised for testing. Once the source code is understood then the source code can be analysed for test cases to be created.

These are the three basic steps that white-box testing takes in order to create test cases:

1. Input, involves different types of requirements, functional specifications, detailed designing of documents, proper source code, security specifications. This is the preparation stage of white-box testing to layout all of the basic information.
2. Processing Unit, involves performing risk analysis to guide whole testing process, proper test plan, execute test cases and communicate results. This is the phase of building test cases to make sure they thoroughly test the application the given results are recorded accordingly.
3. Output, prepare final report that encompasses all of the above preparations and results.

2.BLACK BOX TESTING

Black-box testing is a method of software testing that examines the functionality of an application (e.g. what the software does) without peering into its internal structures or workings (see white-box testing). This method of test can be applied to virtually every level of software testing: unit, integration, system and acceptance. It typically comprises most if not all higher level testing, but can also dominate unit testing as well **Test procedures** Specific knowledge of the application's code/internal structure and programming knowledge in general is not required. The tester is aware of *what* the software is supposed to do but is not aware of *how* it does it. For instance, the tester is aware that a particular input returns a certain, invariable output but is not aware of *how* the software produces the output in the first place.

Test cases Test cases are built around specifications and requirements, i.e., what the application is supposed to do. Test cases are generally derived from external descriptions of the software, including specifications, requirements and design parameters. Although the tests used are primarily *functional* in nature, *non-functional* tests may also be used. The test designer selects both valid and invalid inputs and determines the correct output without any knowledge of the test object's internal structure.

Test design techniques

Typical black-box test design techniques include:

- Decision table testing
- All-pairs testing
- State transition tables
- Equivalence partitioning
- Boundary value analysis

3. UNIT TESTING

In computer programming, **unit testing** is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures are tested to determine if they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could be an entire module, but is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are created by programmers or occasionally by white box testers during the development process. Ideally, each test case is independent from the others. Substitutes such as method stubs, mock objects, fakes, and test harnesses can be used to assist testing a module in isolation. Unit tests are typically written and run by software developers to ensure that code meets its design and behaves as intended.

Its implementation can vary from being very manual (pencil and paper)to being formalized as part of build automation. Testing will not catch every error in the program, since it cannot evaluate every execution path in any but the most trivial programs. The same is true for unit testing. Additionally, unit testing by definition only tests the functionality of the units themselves. Therefore, it will not catch integration errors or broader system-level errors (such as functions performed across multiple units, or non-functional test areas such as performance). Unit testing should be done in conjunction with other software testing activities, as they can only show the presence or absence of particular errors; they cannot prove a complete absence of errors. In order to guarantee correct behaviour for every execution path and every possible input, and ensure the absence of errors, other techniques are required, namely the application of formal methods to proving that a software component has no unexpected behaviour. Software testing is a combinatorial problem. For example, every Boolean decision statement requires at least two tests: one with an outcome of "true" and one with an outcome of "false". As a result, for every line of code written, programmers often need 3 to 5 lines of test code. This obviously takes time and its investment may not be worth the effort. There are also many problems that cannot easily be tested at all – for example those that are nondeterministic or involve multiple threads. In addition, code for a

unit test is likely to be at least as buggy as the code it is testing. Fred Brooks in The Mythical Man-Month quotes: *never take two chronometers to sea. Always take one or three.* Meaning, if two chronometers contradict, how do you know which one is correct? Another challenge related to writing the unit tests is the difficulty of setting up realistic and useful tests. It is necessary to create relevant initial conditions so the part of the application being tested behaves like part of the complete system. If these initial conditions are not set correctly, the test will not be exercising the code in a realistic context, which diminishes the value and accuracy of unit test results.

To obtain the intended benefits from unit testing, rigorous discipline is needed throughout the software development process. It is essential to keep careful records not only of the tests that have been performed, but also of all changes that have been made to the source code of this or any other unit in the software. Use of a version control system is essential. If a later version of the unit fails a particular test that it had previously passed, the version-control software can provide a list of the source code changes (if any) that have been applied to the unit since that time. It is also essential to implement a sustainable process for ensuring that test case failures are reviewed daily and addressed immediately if such a process is not implemented and ingrained into the team's workflow, the application will evolve out of sync with the unit test suite, increasing false positives and reducing the effectiveness of the test suite. Unit testing embedded system software presents a unique challenge: Since the software is being developed on a different platform than the one it will eventually run on, you cannot readily run a test program in the actual deployment environment, as is possible with desktop programs.

4. FUNCTIONAL TESTING

Functional testing is a quality assurance (QA) process and a type of black box testing that bases its test cases on the specifications of the software component under test. Functions are tested by feeding them input and examining the output, and internal program structure is rarely considered (not like in white-box testing).

Functional Testing usually describes *what* the system does. Functional testing differs from system testing in that functional testing "*verifies* a program by checking it against ... design document(s) or specification(s)", while system testing "*validate* a program by checking it against the published user or system requirements" (Kane, Falk, Nguyen 1999, p. 52). Functional testing typically involves five steps .The identification of functions that the software is expected to perform

1. The creation of input data based on the function's specifications
2. The determination of output based on the function's specifications
3. The execution of the test case
4. The comparison of actual and expected outputs

5. PERFORMANCE TESTING

In software engineering, **performance testing** is in general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload. It can also serve to investigate, measure, validate or verify other quality attributes of the system, such as scalability, reliability and resource usage. Performance testing is a subset of performance engineering, an emerging computer science practice which strives to build performance into the implementation, design and architecture of a system.

- **Load testing**

Load testing is the simplest form of performance testing. A load test is usually conducted to understand the behaviour of the system under a specific expected load. This load can be the expected concurrent number of users on the application performing a specific number of transactions within the set duration. This test will give out the response times of all the important business critical transactions. If the database, application server, etc. are also monitored, then this simple test can itself point towards bottlenecks in the application software. Stress testing Stress testing is normally used to understand the upper limits of capacity within the system. This kind of test is done to determine the system's robustness in terms of extreme load and helps application administrators to determine if the system will perform sufficiently if the current load goes well above the expected maximum.

- **Soak testing**

Soak testing, also known as endurance testing, is usually done to determine if the system can sustain the continuous expected load. During soak tests, memory utilization is monitored to detect potential leaks. Also important, but often overlooked is performance degradation. That is, to ensure that the throughput and/or response times after some long period of sustained activity are as good as or better than at the beginning of the test. It essentially involves applying a significant load to a system for an extended, significant period of time. The goal is to discover how the system behaves under sustained use.

- **Spike testing**

Spike testing is done by suddenly increasing the number of or load generated by, users by a very large amount and observing the behaviour of the system. The goal is to determine whether performance will suffer, the system will fail, or it will be able to handle dramatic changes in load.

6 . Configuration testing

Rather than testing for performance from the perspective of load, tests are created to determine the effects of configuration changes to the system's components on the system's performance and behaviour. A common example would be experimenting with different methods of load-balancing.

7 . Isolation testing

Isolation testing is not unique to performance testing but involves repeating a test execution that resulted in a system problem. Often used to isolate and confirm the fault domain.

8 . Integration testing

Integration testing (sometimes called **integration and testing**, abbreviated **I&T**) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

Purpose

The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items. These "design items", i.e. assemblages (or groups of units), are exercised through their interfaces using black box testing, success and error cases being simulated via appropriate parameter and data inputs.

Simulated usage of shared data areas and inter-process communication is tested and individual subsystems are exercised through their input interface. Test cases are constructed to test whether all the components within assemblages interact correctly, for example across procedure calls or process activations, and this is done after testing individual modules, i.e. unit testing. The overall idea is a "building block" approach, in which verified assemblages are added to a verified base which is then used to support the integration testing of further assemblages.

• Big Bang

In this approach, all or most of the developed modules are coupled together to form a complete software system or major part of the system and then used for integration testing. The Big Bang method is very effective for saving time in the integration testing process. However, if the test cases and their results are not recorded properly, the entire integration process will be more complicated and may prevent the testing team from achieving the goal of integration testing. A type of Big Bang Integration testing is called **Usage Model testing**. Usage Model Testing can be used in both software and hardware integration testing. The basis behind this type of integration testing is to run user-like workloads in integrated user-like environments. In doing the testing in this manner, the environment is proofed, while the individual components are proofed indirectly through their use. Usage Model testing takes an optimistic approach to testing, because it expects to have few problems with the individual components. The strategy relies heavily on the component developers to do the isolated unit testing for their product. The goal of the strategy is to avoid redoing the testing done by the developers, and instead flesh-out problems caused by the interaction of the components in the environment. For integration testing, Usage Model testing can be more efficient and provides better test coverage than traditional focused functional integration testing. To be more efficient and accurate, care must be used in defining the user-like workloads for creating realistic scenarios in exercising the environment. This gives confidence that the integrated environment will work as expected for the target customers.

• Top-down and Bottom-up

Bottom Up Testing is an approach to integrated testing where the lowest level components are tested first, then used to facilitate the testing of higher level components. The process is repeated until the component at the top of the hierarchy is tested. All the bottom or low-level modules, procedures or functions are integrated and then tested. After the integration testing of lower level integrated modules, the next level of modules will be formed and can be used for integration testing.

This approach is helpful only when all or most of the modules of the same development level are ready. This method also helps to determine the levels of software developed and makes it easier to report testing progress in the form of a percentage.

Top Down Testing is an approach to integrated testing where the top integrated modules are tested and the branch of the module is tested step by step until the end of the related module.

Sandwich Testing is an approach to combine top down testing with bottom up testing.

The main advantage of the Bottom-Up approach is that bugs are more easily found. With Top-Down, it is easier to find a missing branch link

9 . Verification and validation

Verification and Validation are independent procedures that are used together for checking that a product, service, or system meets requirements and specifications and that it full fills its intended purpose. These are critical components of a quality management system such as ISO 9000. The words "verification" and "validation" are sometimes preceded with "Independent" (or IV&V), indicating that the verification and validation is to be performed by a disinterested third party. It is sometimes said that validation can be expressed by the query "Are you building the right thing?" and verification by "Are you building it right?"In practice, the usage of these terms varies. Sometimes they are even used interchangeably.

The PMBOK guide, an IEEE standard, defines them as follows in its 4th edition

- "**Validation.** The assurance that a product, service, or system meets the needs of the customer and other identified stakeholders. It often involves acceptance and suitability with external customers. Contrast with *verification*."
- "**Verification**. The evaluation of whether or not a product, service, or system complies with a regulation, requirement, specification, or imposed condition. It is often an internal process. Contrast with *validation*."
    - Verification is intended to check that a product, service, or system (or portion thereof, or set thereof) meets a set of initial design specifications. In the development phase, verification procedures involve performing special tests to model or simulate a portion, or the entirety, of a product, service or system, then performing a review or analysis of the modelling results. In the post-development phase, verification procedures involve regularly repeating tests devised specifically to ensure that the product, service, or system continues to meet the initial design requirements, specifications, and regulations as time progresses. It is a process that is used to evaluate whether a product, service, or system complies with regulations, specifications, or conditions imposed at the start of a development phase. Verification can be in development, scale-up, or production. This is often an internal process.

- Validation is intended to check that development and verification procedures for a product, service, or system (or portion thereof, or set thereof) result in a product, service, or system (or portion thereof, or set thereof) that meets initial requirements. For a new development flow or verification flow, validation procedures may involve modelling either flow and using simulations to predict faults or gaps that might lead to invalid or incomplete verification or development of a product, service, or system (or portion thereof, or set thereof). A set of validation requirements, specifications, and regulations may then be used as a basis for qualifying a development flow or verification flow for a product, service, or system (or portion thereof, or set thereof). Additional validation procedures also include those that are designed specifically to ensure that modifications made to an existing qualified development flow or verification flow will have the effect of producing a product, service, or system (or portion thereof, or set thereof) that meets the initial design requirements, specifications, and regulations; these validations help to keep the flow qualified. It is a process of establishing evidence that provides a high degree of assurance that a product, service, or system accomplishes its intended requirements.
- It is sometimes said that validation can be expressed by the query "Are you building the right thing?" and verification by "Are you building it right?". "Building the right thing" refers back to the user's needs, while "building it right" checks that the specifications are correctly implemented by the system. In some contexts, it is required to have written requirements for both as well as formal procedures or protocols for determining compliance.
- It is entirely possible that a product passes when verified but fails when validated. This can happen when, say, a product is built as per the specifications but the specifications themselves fail to address the user’s needs.

Activities

Verification of machinery and equipment usually consists of design qualification (DQ), installation qualification (IQ), operational qualification (OQ), and performance qualification (PQ). DQ is usually a vendor's job. However, DQ can also be performed by the user, by confirming through review and testing that the equipment meets the written acquisition specification. If the relevant document or manuals of machinery/equipment are provided by vendors, the later 3Q needs to be thoroughly performed by the users who work in an industrial regulatory environment. Otherwise, the process of IQ, OQ and PQ is the task of validation. The typical example of such a case could be the loss or absence of vendor's documentation for legacy equipment or do-it-yourself (DIY) assemblies (e.g., cars, computers etc.) and, therefore, users should endeavour to acquire DQ document beforehand.

Each template of DQ, IQ, OQ and PQ usually can be found on the internet respectively, whereas the DIY qualifications of machinery/equipment can be assisted either by the vendor's training course materials and tutorials, or by the published guidance books, such as *step-by-step* series if the acquisition of machinery/equipment is not bundled with on-site qualification services. This kind of the DIY approach is also applicable to the qualifications of software, computer operating systems and a manufacturing process. The most important and critical task as the last step of the activity is to generating and archiving machinery/equipment qualification reports for auditing purposes, if regulatory compliances are mandatory. Qualification of machinery/equipment is venue dependent, in particular items that are shock sensitive and require balancing or calibration, and re-qualification needs to be conducted once the objects are relocated. The full scales of some equipment qualifications are even time dependent as consumables are used up (i.e. filters) or springs stretch out, requiring recalibration, and hence re-certification is necessary when a specified due time lapse Re-qualification of machinery/equipment should also be conducted when replacement of parts, or coupling with another device, or installing a new application software and restructuring of the computer which affects especially the pre-settings, such as on BIOS, registry, disk drive partition table, dynamically-linked (shared) libraries, or an ini file etc., have been necessary. In such a situation, the specifications of the parts/devices/ software and restructuring proposals should be appended to the qualification document whether the parts/devices/software are genuine or not. Torres and Hyman have discussed the

suitability of non-genuine parts for clinical use and provided guidelines for equipment users to select appropriate substitutes which are capable to avoid adverse effects. In the case when genuine parts/devices/software are demanded by some of regulatory requirements, then re-qualification does not need to be conducted on the non-genuine assemblies. Instead, the asset has to be recycled for non-regulatory purposes. When machinery/equipment qualification is conducted by a standard endorsed third party such as by an ISO standard accredited company for a particular division, the process is called certification. Currently, the coverage of ISO/ IEC 15408 certification by an ISO/IEC 27001 accredited organization is limited; the scheme requires a fair amount of efforts to get popularized.

10 . System testing

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic. As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware system(s).

The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called *assemblages*) or between any of the *assemblages* and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole. System testing is performed on the entire system in the context of a **Functional Requirement Specification**(s) (FRS) and/or a **System Requirement Specification** (SRS). System testing tests not only the design, but also the behavior and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specification

Types of tests to include in system testing

The following examples are different types of testing that should be considered during System testing:

- Graphical user interface testing
- Usability testing
- Software performance testing
- Compatibility testing
- Exception handling
- Load testing
- Volume testing
- Stress testing
- Security testing
- Scalability testing
- Sanity testing
- Smoke testing
- Exploratory testing
- Ad hoc testing
- Regression testing
- Installation testing
- Maintenance testing Recovery testing and failover testing.
- Accessibility testing, including compliance with :
  - Americans with Disabilities Act of 1990
  - Section 508 Amendment to the Rehabilitation Act of 1973
  - Web Accessibility Initiative (WAI) of the World Wide Web Consortium (W3C)

Although different testing organizations may prescribe different tests as part of System testing, this list serves as a general framework or foundation to begin with.

11 . Structure Testing

It is concerned with exercising the internal logic of a program and traversing particular execution paths.

12 . Output Testing

- Output of test cases compared with the expected results created during design of test cases.
- Asking the user about the format required by them tests the output generated or displayed by the system under consideration.
- Here, the output format is considered into two was, one is on screen and another one is printed format.
- The output on the screen is found to be correct as the format was designed in the system design phase according to user needs.
- The output comes out as the specified requirements as the user's hard copy.

13 . User acceptance Testing

- Final Stage, before handling over to the customer which is usually carried out by the customer where the test cases are executed with actual data.
- The system under consideration is tested for user acceptance and constantly keeping touch with the prospective system user at the time of developing and making changes whenever required.
- It involves planning and execution of various types of test in order to demonstrate that the implemented software system satisfies the requirements stated in the requirement document.

Two set of acceptance test to be run

1. Those developed by quality assurance group.
2. Those developed by customer.

10.1 SYSTEM SECURITY

Security :

The proposed system will be highly secured, every user will be required registration and username/password to use the system. The system will do the proper authorization and authentication of the users based on their types and their requirements. The proposed system will be designed persistently to avoid any misuse of the application.

10.2 FUTURE ENHANCEMENT

In near future, we are planning to hire cars daily bases. So that clients can give their car to the customer on daily bases. We are planning to add new feature i.e. pay after the trip. We are working to increase automation in the system to increaseuser experience great

10.3 BIBLIOGRAPHY AND REFERENCES

Books Used:

- Software Engineering - R.S. Pressman
- PHP For Dummies
- PHP Begineers Guide By McGrawhill Publication
- Javascript By McGrawhill Publication

References Used:

- http://www.carrentingsolutions.com/
- http://www.flashvortex.com/
- http://www.imscart.com/car_rental_software.html
- Wikipedia.org
- www.w3schools.com

10.4 GLOSSARY

When it comes to booking a rental car, there is a lot to think about. The last thing you want is to hear some car rental terminology that you don't understand. VroomVroomVroom has put together a list of frequently used car rental industry terms to save you any confusion at the rental desk.

Additional Driver

This refers to any drivers listed on the rental agreement in addition to the main driver. Suppliers often charge a daily fee to add other drivers.

Age Surcharge

For drivers under 25, suppliers will charge an additional daily surcharge. The amount charged differs depending on the car rental company.

Child Seat

You can request a child seat with your booking. Keep in mind there is limited availability, so make sure you supply all relevant information as soon as possible.

Collision Damage Waiver

Without purchasing Collision Damage Waiver, the customer is liable for the full cost of the vehicle.

Debit Card

While some suppliers may accept debit cards, this is not the case for all car rental companies. Most will require that you have a credit card in the lead drivers name.

Driving Restrictions

Read your rental agreement carefully to see if there are any restrictions to where you are allowed to take the car. For example, most suppliers do not permit you to drive the vehicle on unsealed roads.

Loss Damage Waiver

Purchasing Loss Damage Waiver reduces the renters liability for any damage caused to the rental vehicle. The cost of this may vary by supplier and location.

One-Way

If you are picking up a vehicle at one location and returning it to another, your rental may incur a one-way charge.

Personal Accident Insurance

This is offered by car rental companies. This is a type of insurance covering medical expenses.