Development of a practical online application for the

Online Cab Booking System (cabsforyou.com)

**A Project Report for Summer Industrial Training**

###### ***Submitted by***

##### Name of the Students

* Sayan Basu
* Soumeet Basak
* Arundhati Dutta
* Pallavi Sharma

***In partial fulfillment for the award of the degree of***

##### **B. Tech**

In

Computer Science Engineering

**Neotia Institute of Technology, Management and Science**

At

**Ardent Computech Pvt. Ltd.**



**July – August 2016**

**Ardent Computech Pvt. Ltd.**



**BONAFIDE CERTIFICATE**

Certified that this project work was carried out under my supervision

***“Development of a feature-rich, practical online application for the Training and Placement Dept. of the college (T&P automation)”*** is the bonafide work of

***Name of the student: Signature:***

***Name of the student: Signature:***

***Name of the student: Signature:***

***Name of the student: Signature:***

**SIGNATURE**

Name :

**PROJECT MENTOR**

**SIGNATURE**

**Name:**

**EXAMINERS**

**Ardent Original Seal**

**Acknowledgement**

I take this opportunity to express my deep gratitude and sincerest thank to my project mentor, Amitabha Chatterjee for giving most valuable suggestion, helpful guidance and encouragement in the execution of this project work.

I will like to give a special mention to my colleagues. Last but not the least I am grateful to all the faculty members of Ardent Computech Pvt. Ltd. or their support.

|  |  |
| --- | --- |
| **Table of Contents** | **Page No** |
| 1. **Abstract** | 5 |
| 1. **Introduction, Objectives of the Project and Benefits of this project, Project Category(RDBMS/OOPS/Networking/Multimedia/Artificial Intelligence/Expert Systems)** | 6 |
| 1. **SDLC(Software Development Life cycle)** | 7 |
| 1. **Project Analysis** | 8 |
| 1. **Project Planning and schedule** | 15 |
| 1. **Data Modelling** | 16 |
| 1. **Process Modelling** | 20 |
| 1. **Other Software Engineering Paradigm Applied** | 23 |
| 1. **User Interface Design** | 25 |
| 1. **Coding** | 42 |
| 1. **Testing** | 50 |
| 1. **Maintains and cost estimation** | 57 |
| 1. **Project Reports** | 59 |
| 1. **Future scope and further enhancement of the Project** | 66 |
| 1. **Bibliography** | 67 |

**Chapter 1 : ABSTRACT**

Cab booking is a booming business that has the potentiality to generate huge revenue using Cab Booking Software, instead of the traditional cab hailing system. One can enhance the quality of service as well by streamlining and automating the processes by taking advantage of such technology. Our feature loaded and fast Cab Booking Application is enough efficient to handle the intricacies of the simultaneously running processes that ensures smooth growth of the business and reduce the downtime.

There are different ranges of cabs available according to one’s preference. As the internet users are increasing exponentially, companies have introduced Online Cab Booking system. This system improves customer’s experience but also eases the hassles of a customer while taking a ride.

Cab booking software has research driven features and functionalities that will lessen your effort to reduce overall cost, while it will improve the productivity, smoothen the booking procedure and enhance profitability of your company.

**Chapter 2: Introduction, Objectives of the Project and benefit of this project and category**

You must have waited for cabs when you are supposed to go somewhere and you are in a hurry or the weather conditions are bad or there is some unrest (religious, political, etc.) going on the roads. These are the very few problems faced by one while taking a ride. It’s a time consuming process which at times irritates customers.

To resolve many such problems there is an online cab booking system introduced. The location of pick-up and destination is known from before, which makes the entire process very smooth and user-friendly ensuring the comfort of the customer.

**The Objective of this project are**

The main objectives of this ordering system are:

**Payment**- This system will give option to the customer for online as well as cash payment. Customers can choose the mode of payment according to their convenience.

**Better Knowledge**- This system will provide customer all the details of his cab, the driver before booking a cab. This confirmation will help customers to check all the details with the cost per km.

**Know Location and destination**- This system will let the driver know the location of the customer. For pick-ups customers can also fix a time before hand and book a ride.

**Improves Efficiency**- This system will make things easier for drivers as whole booking process is done by customers only.  
These were the objectives of the Cab Booking System. Let us now get into the details of the working of this booking system.

**Benefit of this project are**

Let us now understand the working of this system.  
Whenever a customer visits the web application of the cab booking system, he/she will have to select his/her location so that the nearest cab available can take their booking.  
Different ranges of cabs will be visible to the customer with the approximate fare. There is also an option to avail discount coupons if available.

**Project Category**

Web Application

**Chapter 3:**  **SDLC (Software Development Life cycle)**

**Development Methodology**

Often, a customer defines a set of general objectives for software but does not identify detailed input, processing, or output requirements. In other cases, the developer may be unsure of the efficiency of an algorithm, the adaptability of an operating system, or the form that human/machine interaction should take.

In these, and many other situations, a prototyping paradigm may offer the best approach.



The prototyping paradigm begins with requirements gathering. Developer and customer meet and define the overall objectives for the software, identify whatever requirements are known, and outline areas where further definition is mandatory. A **"quick design"** then occurs. The quick design focuses on a representation of those aspects of the software that will be visible to the customer/user (e.g., input approaches and output formats). The quick design leads to the construction of a prototype. The prototype is evaluated by the customer/user and used to refine requirements for the software to be developed. Iteration occurs as the prototype is tuned to satisfy the needs of the customer, while at the same time enabling the developer to better understand what needs to be done.

Ideally, the prototype serves as a mechanism for identifying software requirements. If a working prototype is built, the developer attempts to use existing program fragments or applies too…. (e.g., report generators, window managers) that enable working programs to be generated quickly.

**Chapter 4: Project Analysis**

You should provide a feasibility report in the following format:

* ***Product:***

A general statement of the product; give a brief description of what the proposed system will do, highlighting where the proposed system meets the specified business requirements of the organization.

* ***Technical Feasibility:***

Will the proposed system perform to the required specification? Outline technical systems options you propose to use, which will give a technical solution satisfying the requirements and constraints of the system, as outlined in the terms of reference.

* ***Social Feasibility:***

Consideration of whether the proposed system would prove acceptable to the people who would be affected by its introduction. Describe the effect on users from the introduction of the new system; consider whether there will be a need for retraining the workforce. Will there be a need for relocation of some of the workforce? Will some jobs become deskilled? Will the current workforce be able to perform effectively any new tasks introduced by the proposed system? Describe how you propose to ensure user co-operation before changes are introduced.

* ***Economic Feasibility:***

Consider the cost/benefits of the proposed system. Detail the costs that will be incurred by the organization adopting the new system; consider development costs and running costs. Detail benefits that the new system will bring, direct economic benefits such as reduced costs, and indirect benefits, such as improved management information and better customer service. Illustrate the cost/benefit of the new system by applying a suitable cost/benefit analysis method such as the payback method.

* ***Market Research:***

A comprehensive market research identifying a need for the product. Detail all market research you carried out, listing sources of information. Justify any conclusions you have drawn from your research. Identify the potential customer base for your product, together with evidence of customer need for the product. Describe how you propose to compete with similar products on the market.

\*

* ***Alternative Solution:***

Consideration of alternative solutions should be documented. At least two alternative business or technical systems options should be considered. Detail the differences between these options and the proposed system. Justify your choice of the proposed system and the reasons for rejecting the alternative options. At this point, all of the planning for the project has been done and if the feasibility study has shown that the project is likely to succeed within its constraints, then it only remains for us to start the requirements analysis and thus proceed with the project.

|  |  |
| --- | --- |
| **Feasibility Study** | |
| System: cabforyou.com | Date: 03/08/2016 |
| Author: | Page: 1 |
| **Product** | |
| The project requires a web application to be developed that will allow online cab booking management. | |
| **Technical Feasibility** | |
| The web application will be developed using Python with HTML, CSS, JS with Oracle11G as database. The team is competent in that. | |
| **Social Feasibility** | |
| Some training for the users/admin are required but all users are IT literate. | |
| **Market Research** | |
| Market research says that this application would be useful for the users as it could seamlessly help them to book cabs. | |
| **Economic Feasibility** | |
| The application can be developed within budget. | |
| **Alternate Solution** | |
| Could be a desktop system but that would not allow documents to be shared online. | |

**Requirement Analysis**

1. **Technical Requirement**

**Software Requirement**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Front End** | **Platform** | **Back End** | **Web Browser** | **Database** |
| HTML,CSS, HTML5, JAVA SCRIPT & JQUERY | Windows 10 x64 | Python 3.4 | Google Chrome | Oracle 11G |

**Hardware Requirement**

|  |  |
| --- | --- |
| **Client Side** | **SERVER Side** |
| * Web browser: Google Chrome * Processor : Intel Core 2 Duo * RAM : 1 GB | * Processor: Intel Core i5 * RAM : 4GB and Above * Disk Space: 10GB |

**Non-Technical Requirement**

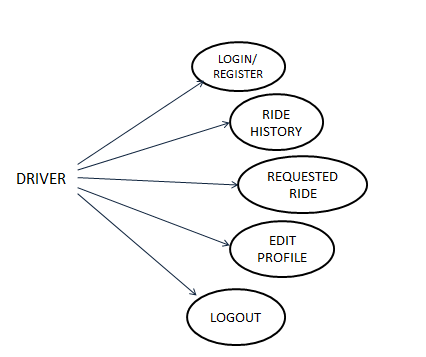
**1. Functional Requirement**

Functional Requirements are those that refer to the functionality of the system, i.e., what services it will provide to the user. Nonfunctional (supplementary) requirements pertain to other information needed to produce the correct system and are detailed separately.

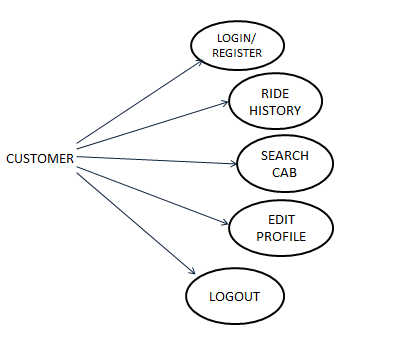
* Any registered user can log in through login procedure.
* Any new user can register through register procedure.
* A registered user can navigate through only those pages which permission is given to them.
* A registered user can book cabs and pay online for the ride through credit card.

An administrator can add or view driver details, cab details, add one new admin and view feedback.

***Use Case Diagram for Driver:***



***Use Case Diagram for Customer:***



***Use Case Description:***

|  |  |
| --- | --- |
| **Use Case Name** | Authentication |
| **Priority** | Essential |
| **Trigger** | Menu selection |
| **Precondition** | User is connected to the Internet and on the cabforyou.com home page |
| **Basic Path** | 1. User enters email id and password. 2. The email id and password is matched with the record in the database. 3. If the authentication parameters are correct the user is directed to the user’s main page, otherwise an error message is displayed. |
| **Alternate Path** | NA |
| **Post Condition** | The user is on the User Home Page |
| **Exception Path** | If there is a connection failure the server returns to the wait state |
| **Use Case Name** | Create booking |
| **Priority** | Essential |
| **Trigger** | Location selection |
| **Precondition** | User is connected to the Internet and on the user’s main page |
| **Basic Path** | 1. User books a cab. 2. User can see ride history |
| **Alternate Path** | NA |
| **Post Condition** | The customer booked cab. |
| **Exception Path** | If there is a connection failure the server returns to the wait state |

|  |  |
| --- | --- |
| **Use Case Name:** | View booking |
| **Priority** | Essential |
| **Trigger** | Location selection |
| **Precondition** | Driver is connected in the driver log in page |
| **Basic Path** | 1. Driver log in the driver account. 2. Clicks on the booking details link. |
| **Alternate Path** | NA |
| **Post Condition** | The driver views the bookings |
| **Exception Path** | If there is a connection failure the server returns to the wait state |

2. **Non Functional Requirements**

In addition to the obvious features and functions that you will provide in your system, there are other requirements that don't actually DO anything, but are important characteristics nevertheless. These are called "non-functional requirements" or sometimes "Quality Attributes." For example, attributes such as performance, security, usability, compatibility. Aren't a "feature" of the system, but are a required characteristic. You can't write a specific line of code to implement them; rather they are "emergent" properties that arise from the entire solution. The specification needs to describe any such attributes the customer requires. You must decide the kind of requirements that apply to your project and include those that are appropriate.

Each requirement is simply stated in English. Each requirement must be objective and quantifiable; there must be some measurable way to assess whether the requirement has been met.

Often deciding on quality attributes requires making tradeoffs, e.g., between performance and maintainability. In the APPENDIX you must include an engineering analysis of any significant decisions regarding tradeoffs between competing attributes.

Here are some examples of non-functional requirements:

***Performance requirements:***

Requirements about resources required, response time, transaction rates, throughput, benchmark specifications or anything else having to do with performance.

For better performance the application will restrict the document size to 5 MB.

***Operating constraints:***

List any run-time constraints. This could include system resources, people, needed software,

The application must run without any manual intervention.

***Platform constraints:***

Discuss the target platform. Be as specific or general as the user requires. If the user doesn't care, there are still platform constraints.

Since the application will be developed in JEE it is platform independent.

***Accuracy and Precision:***

Requirements about the accuracy and precision of the data. (Do you know the difference?) Beware of 100% requirements; they often cost too much.

***Modifiability:***

Requirements about the effort required to make changes in the software. Often, the measurement is personnel effort (person- months).

Minimal

***Portability:***

The effort required to move the software to a different target platform. The measurement is most commonly person-months or % of modules that need changing.

Minimal

***Reliability:***

Requirements about how often the software fails. The measurement is often expressed in MTBF (mean time between failures). The definition of a failure must be clear. Also, don't confuse reliability with availability which is quite a different kind of requirement. Be sure to specify the consequences of software failure, how to protect from failure, a strategy for error detection, and a strategy for correction.

***Security:***

One or more requirements about protection of your system and its data. The measurement can be expressed in a variety of ways (effort , skill level , time , ...) to break into the system. Do not discuss solutions (e.g. passwords) in a requirements document.

Only secured users can access the application.

No one can go to any independent page without logging in.

***Usability:***

Requirements about how difficult it will be to learn and operate the system. The requirements are often expressed in learning time or similar metrics.

***Legal***

There may be legal issues involving privacy of information, intellectual property rights, export of restricted technologies, etc.

**Chapter 5 : Project Planning and Schedule**

Project planning is concerned with identifying the following for every project:

1. Activities
2. Milestones
3. Deliverables.

A plan must be drawn up to guide the development towards the project goal. A plan is drawn up at the start of a project. This plan should be used as the driver for the project. The initial plan is not static, and must be modified as the project progresses. Planning is required for development activities from specification through to delivery of the system.

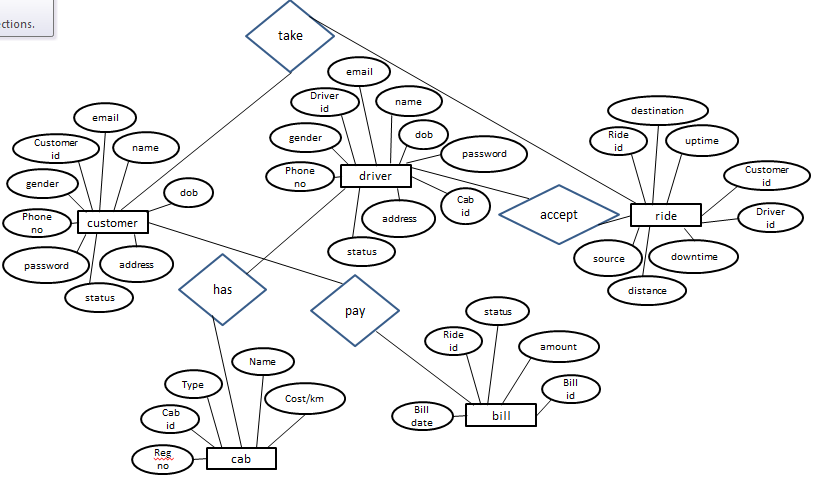
***GANTT CHART:***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Task** | **Person(s) Responsible** | **Week 1** | **Week 2** | **Week 3** | **Week 4** |
| **Communication** |  |  |  |  |  |
| **Quick Plan** |  |  |  |  |  |
| **Modeling Quick Design** |  |  |  |  |  |
| **Construction of Prototype** |  |  |  |  |  |
| **Deployment, Delivery and Feedback** |  |  |  |  |  |

**Chapter 6 : Data Modeling**

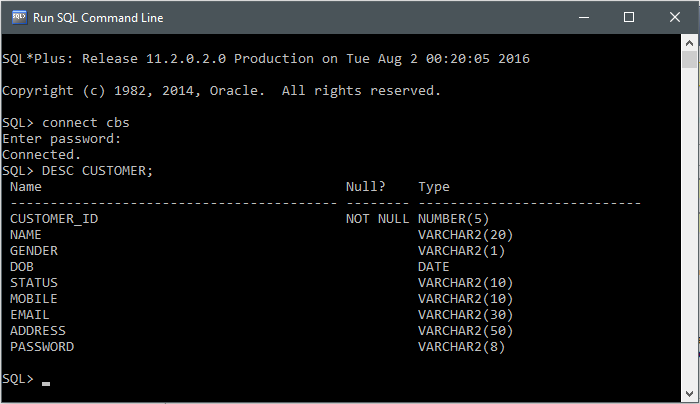
**Entity Relationship Diagram**

ERD stands for Entity relationship diagram. An **ERD** is a data modeling technique that can help **define** business processes and can be used as the foundation for a relational database.

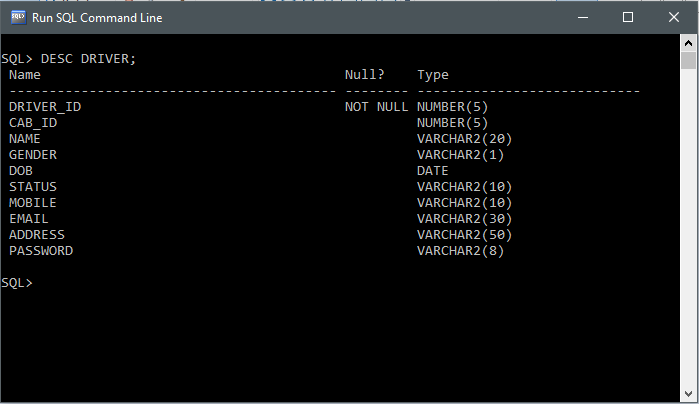


**Data Base Tables**

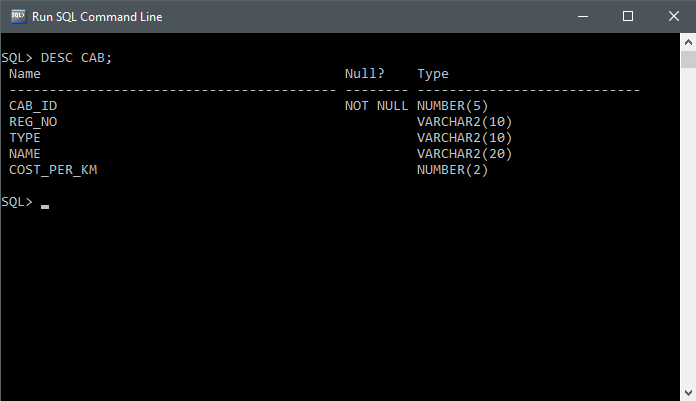
Customer Table



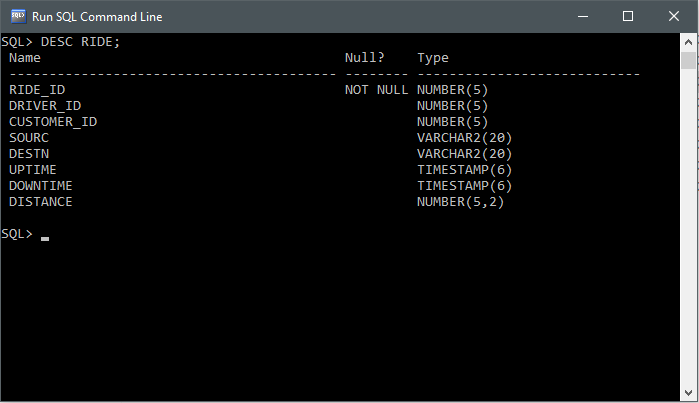
Driver Table



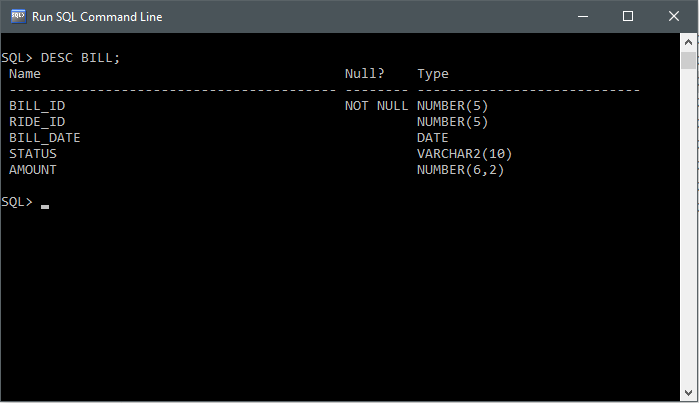
Cab Table



Ride Table



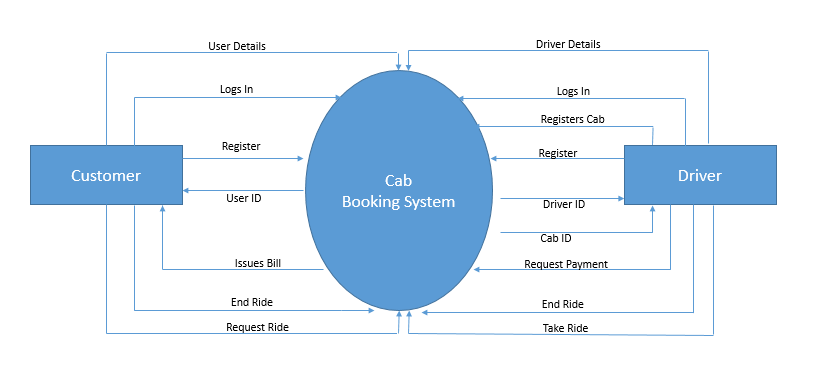
Bill Table



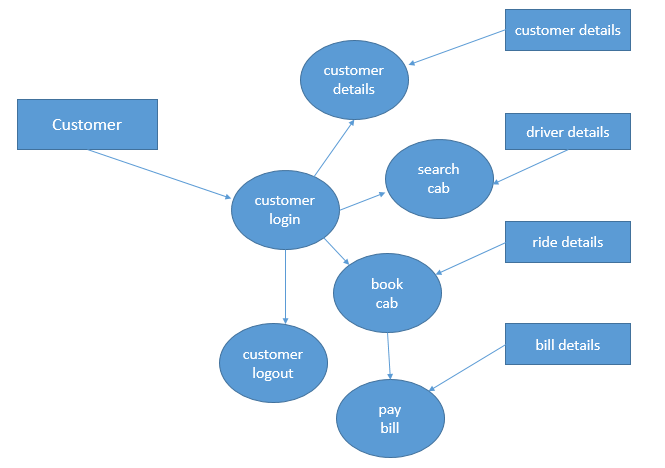
**Chapter 7: Process Modeling**

**DFD**

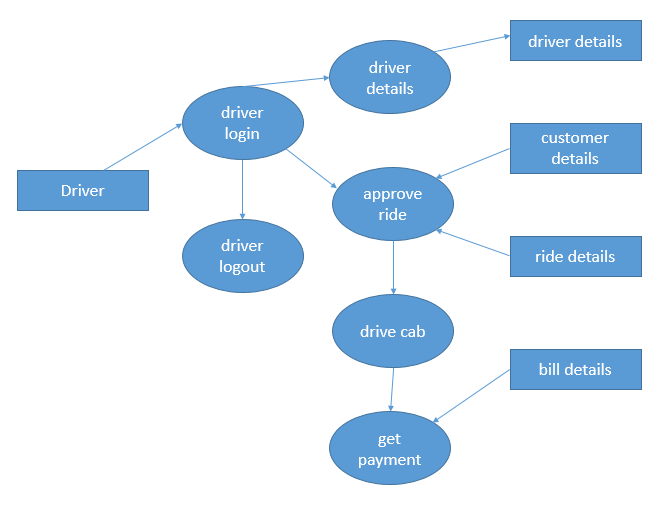
**DFD Level-0(Context Diagram)**



**Level 1 DFD for Customer**

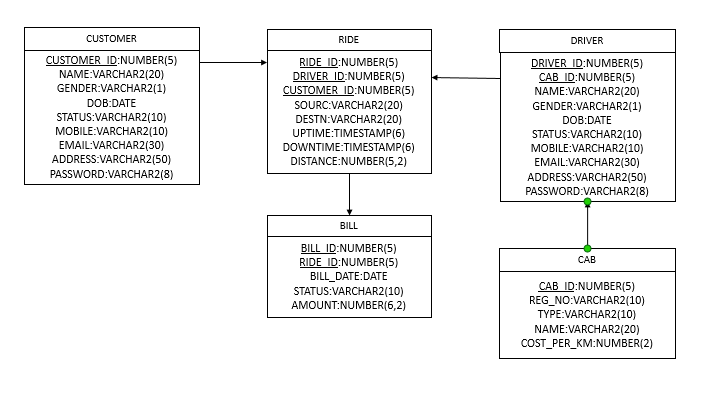


**Level 1 DFD for Driver**

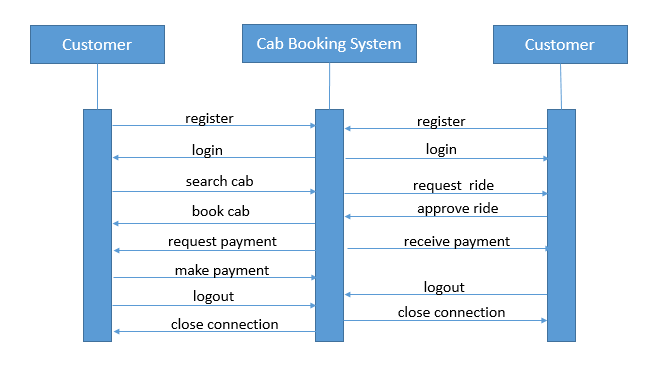
****

**Chapter 8: Other Software Engineering Paradigms Applied**

**Schema/Database Design**



**Sequence Diagram**



**Activity Diagram**

user registration

cabsforyou.com

login

Reject

login correct?

is driver?

View location, view approx. fare

Customer

View user details, booking details

Ride booked

Confirm booking

**Chapter 9: User Interface Design**

**User interface design** (**UI**) or **user interface** engineering is the **design** of **user interfaces** for machines and software, such as computers, home appliances, mobile devices, and other electronic devices, with the focus on maximizing the **user** experience.

## Form created using HTML

home.html:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Cab For You</title>

<meta http-equiv="content-type" content="text/html; charset=utf-8" />

<meta name="description" content="" />

<meta name="keywords" content="" />

<script src="js/jquery.min.js"></script>

<script src="js/skel.min.js"></script>

<script src="js/skel-layers.min.js"></script>

<script src="js/init.js"></script>

<noscript>

<link rel="stylesheet" href="css/skel.css" />

<link rel="stylesheet" href="css/style.css" />

<link rel="stylesheet" href="css/style-xlarge.css" />

</noscript>

</head>

<body class="landing">

<header id="header">

<h1><a href="home.html">Interphase</a></h1>

</header>

<section id="banner">

<h2>Cabs For You</h2>

<p></p>

<ul class="actions">

<li>

<a href="driver.html" class="button big">Driver</a>

</li>

<li>

<a href="customer.html" class="button big">Customer</a>

</li>

</ul>

</section>

<footer id="footer">

<div class="container">

<div class="row">

<section class="4u$ 12u$(medium) 12u$(small)">

<h3>Contact Us</h3>

<ul class="icons">

<li><a href="#" class="icon rounded fa-twitter"><span class="label">Twitter</span></a></li>

<li><a href="#" class="icon rounded fa-facebook"><span class="label">Facebook</span></a></li>

<li><a href="#" class="icon rounded fa-google-plus"><span class="label">Google+</span></a></li>

<li><a href="#" class="icon rounded fa-linkedin"><span class="label">LinkedIn</span></a></li>

</ul>

<ul class="tabular">

<li>

<h3>Address</h3>

NITMAS Campus<br>

DH Road, West Bengal

</li>

<li>

<h3>Mail</h3>

<a href="#">soumeet.basak@gmail.com</a>

</li>

<li>

<h3>Phone</h3>

(983) 507-2430

</li>

</ul>

</section>

</div>

<ul class="copyright">

<li>&copy; Untitled. All rights reserved.</li>

<li>Design: <a href="http://templated.co">Templated</a></li>

<li>Images: <a href="http://unsplash.com">Unsplash</a></li>

</ul>

</div>

</footer>

</body>

</html>

driver.html:

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<title>Customer</title>

<link href='http://fonts.googleapis.com/css?family=Titillium+Web:400,300,600' rel='stylesheet' type='text/css' />

<link rel="stylesheet" href="css/normalize.css" type="text/css" />

<link rel="stylesheet" href="css/style.css" type="text/css" />

</head>

<body>

<div class="form">

<ul class="tab-group">

<li class="tab active"><a href="#signup">Sign Up</a></li>

<li class="tab"><a href="#login">Log In</a></li>

</ul>

<div class="tab-content">

<div id="signup">

<h1>Sign Up for Free</h1>

<form name="fm\_customer\_register" id="fm\_customer\_register" action="python/customer\_register.py" method="post">

<div class="field-wrap">

<label>Name<span class="req">\*</span></label>

<input type="text" autocomplete="off" name="name" id="name" />

</div>

<div class="field-wrap">

<label>Gender<span class="req">\*</span></label>

<input type="text" autocomplete="off" name="gender" id="gender" />

</div>

<div class="field-wrap">

<label>Date Of Birth<span class="req">\*</span></label>

<input type="text" autocomplete="off" name="dob" id="dob" />

</div>

<div class="field-wrap">

<label>Mobile<span class="req">\*</span></label>

<input type="text" required="" autocomplete="off" name="mobile" id="mobile" />

</div>

<div class="field-wrap">

<label>Email<span class="req">\*</span></label>

<input type="email" required="" autocomplete="off" name="email" id="email" />

</div>

<div class="field-wrap">

<label>Address<span class="req">\*</span></label>

<input type="text" required="" autocomplete="off" name="address" id="address" />

</div>

<div class="field-wrap">

<label>Set A Password<span class="req">\*</span></label> <input type=

"password" required="" autocomplete="off" name="password" id="password" />

</div>Get Started

</form>

</div>

<div id="login">

<h1>Welcome Back!</h1>

<form action="python/customer\_login.py" method="post">

<div class="field-wrap">

<label>Email Address<span class="req">\*</span></label> <input type="email"

required="" autocomplete="off" name="email" />

</div>

<div class="field-wrap">

<label>Password<span class="req">\*</span></label> <input type="password"

required="" autocomplete="off" name="password" />

</div>

<p class="forgot"><a href="customer\_forgot.html">Forgot Password?</a></p>Log In

</form>

</div>

</div><!-- tab-content -->

</div><!-- /form -->

<script src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js' type=

"text/javascript">

</script><script src="js/index.js" type="text/javascript">

</script><script type="text/javascript">

$("#fm\_customer\_register").submit(function(e){

var name = $('#name').val();

var gender = $('#gender').val();

var dob = $('#dob').val();

var mobile = $('#mobile').val();

var email = $('#email').val();

var address = $('#address').val();

var password = $('#password').val();

var formURL = $(this).attr("action");

var data = {

name:name,

gender:gender,

dob:dob,

mobile:mobile,

email:email,

address:address,

password:password

}

$.ajax({

url : formURL,

type: "POST",

data : data,

dataType: 'json',

success: function (data) {

if(data[0]==1){

alert('Customer registered with Customer ID: '+data[1]);

$("#fm\_customer\_register")[0].reset();

}

else

alert('Customer registered failed');

}

});

e.preventDefault();

});

</script>

</body>

</html>

**cab\_register.html:**

<!DOCTYPE html>

<html>

<head>

<meta content="HTML Tidy for Linux (vers 25 March 2009), see www.w3.org" name="generator">

<meta charset="UTF-8">

<title>Cab Registration</title>

<link href="css/cab\_register\_style.css" rel="stylesheet" type="text/css">

</head>

<body class="align">

<div class="site\_\_container">

<div class="grid\_\_container">

<form action="python/cab\_register.py" class="form form--login" id="fm\_cab\_register" method="post" name="fm\_cab\_register">

<div class="form\_\_field">

<label for="login\_\_username">

<span class="hidden">Registration No</span>

</label>

<input class="form\_\_input" id="reg\_no" name="reg\_no" placeholder="Registration No" required="" type="text">

</div>

<div class="form\_\_field">

<label for="login\_\_username">

<span class="hidden">Type</span>

</label>

<input class="form\_\_input" id="type" name="type" placeholder="Type" required="" type="text">

</div>

<div class="form\_\_field">

<label for="login\_\_username">

<span class="hidden">Cab Name</span>

</label>

<input class="form\_\_input" id="name" name="name" placeholder="Cab Name" required="" type="text">

</div>

<div class="form\_\_field">

<label for="login\_\_username">

<span class="hidden">Mileage</span>

</label>

<input class="form\_\_input" id="cost\_per\_km" name="cost\_per\_km" placeholder="Mileage" required="" type="text">

</div>

<div class="form\_\_field"><input type="submit" value="Get Registered"></div>

</form>

</div>

</div>

<script src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js' type="text/javascript"></script>

<script type="text/javascript">

$("#fm\_cab\_register").submit(function(e){

var reg\_no = $('#reg\_no').val();

var type = $('#type').val();

var name = $('#name').val();

var cost\_per\_km = $('#cost\_per\_km').val();

var formURL = $(this).attr("action");

var data = {

reg\_no:reg\_no,

type:type,

name:name,

cost\_per\_km:cost\_per\_km

}

$.ajax({

url : formURL,

type: "POST",

data : data,

dataType: 'json',

success: function (data) {

if(data[0]==1){

alert('Cab registered with Cab ID: '+data[1])

$("#fm\_cab\_register")[0].reset();

}

else

alert('Cab registered failed')

}

});

e.preventDefault();

});

</script>

</body>

</html>

**driver\_forgot.html:**

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta charset="UTF-8" />

<title>Driver Password</title>

<link rel="stylesheet" href="css/cab\_register\_style.css" type="text/css" />

</head>

<body class="align">

<div class="site\_\_container">

<div class="grid\_\_container">

<form id="fm\_driver\_forgot" name="fm\_driver\_forgot" action="python/driver\_forgot.py"

method="post" class="form form--login">

<div class="form\_\_field">

<label for="login\_\_username">

<span class="hidden">Registration

No</span></label>

<input name="email" id="email" type="text" class="form\_\_input" placeholder="Email" required="" />

</div>

<div class="form\_\_field">

<label for="login\_\_username">  
<span class="hidden">Mileage</span>

</label>

<input name="dob" id="dob" type="text" class="form\_\_input" placeholder="Date Of Birth" required/>

</div>

<div class="form\_\_field">

<input type="submit" value="Get Password" /></div>

</form>

</div>

</div>

<script src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js' type=

"text/javascript">

</script><script type="text/javascript">

$("#fm\_driver\_forgot").submit(function(e) {

var email = $('#email').val();

var dob = $('#dob').val();

var formURL = $(this).attr("action");

var data = {

email: email,

dob: dob

}

$.ajax({

url: formURL,

type: "POST",

data: data,

dataType: 'json',

success: function(data) {

if (data[0] == 1) {

alert('Password: ' + data[1])

$("#fm\_driver\_forgot")[0].reset();

} else alert('Account not found')

}

});

e.preventDefault();

});

</script>

</body>

</html>

**customer.html:**

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Customer</title>

<link href='http://fonts.googleapis.com/css?family=Titillium+Web:400,300,600' rel='stylesheet' type='text/css'>

<link href="css/normalize.css" rel="stylesheet">

<link href="css/style.css" rel="stylesheet">

</head>

<body>

<div class="form">

<ul class="tab-group">

<li class="tab active"><a href="#signup">Sign Up</a></li>

<li class="tab"><a href="#login">Log In</a></li>

</ul>

<div class="tab-content">

<div id="signup">

<h1>Sign Up for Free</h1>

<form action="python/customer\_register.py" id="fm\_customer\_register" method="post" name="fm\_customer\_register">

<div class="field-wrap"><label>Name<span class="req">\*</span></label> <input autocomplete="off" id="name" name="name" type="text"></div>

<div class="field-wrap"><label>Gender<span class="req">\*</span></label> <input autocomplete="off" id="gender" name="gender" type="text"></div>

<div class="field-wrap"><label>Date Of Birth<span class="req">\*</span></label> <input autocomplete="off" id="dob" name="dob" type="text"></div>

<div class="field-wrap"><label>Mobile<span class="req">\*</span></label> <input autocomplete="off" id="mobile" name="mobile" required="" type="text"></div>

<div class="field-wrap"><label>Email<span class="req">\*</span></label> <input autocomplete="off" id="email" name="email" required="" type="email"></div>

<div class="field-wrap"><label>Address<span class="req">\*</span></label> <input autocomplete="off" id="address" name="address" required="" type="text"></div>

<div class="field-wrap"><label>Set A Password<span class="req">\*</span></label> <input autocomplete="off" id="password" name="password" required="" type="password"></div>

<button class="button button-block" type="submit"></button>Get Started</form>

</div>

<div id="login">

<h1>Welcome Back!</h1>

<form action="python/customer\_login.py" method="post">

<div class="field-wrap"><label>Email Address<span class="req">\*</span></label> <input autocomplete="off" name="email" required="" type="email"></div>

<div class="field-wrap"><label>Password<span class="req">\*</span></label> <input autocomplete="off" name="password" required="" type="password"></div>

<p class="forgot"><a href="customer\_forgot.html">Forgot Password?</a></p>

<button class="button button-block" type="submit"></button>Log In</form>

</div>

</div>

<!-- tab-content --></div>

<!-- /form -->

<script src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>

<script src="js/index.js"></script>

<script>

$("#fm\_customer\_register").submit(function(e){

var name = $('#name').val();

var gender = $('#gender').val();

var dob = $('#dob').val();

var mobile = $('#mobile').val();

var email = $('#email').val();

var address = $('#address').val();

var password = $('#password').val();

var formURL = $(this).attr("action");

var data = {

name:name,

gender:gender,

dob:dob,

mobile:mobile,

email:email,

address:address,

password:password

}

$.ajax({

url : formURL,

type: "POST",

data : data,

dataType: 'json',

success: function (data) {

if(data[0]==1){

alert('Customer registered with Customer ID: '+data[1]);

$("#fm\_customer\_register")[0].reset();

}

else

alert('Customer registered failed');

}

});

e.preventDefault();

});

</script>

</body>

</html>

**customer\_forgot.html:**

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Customer Password</title>

<link href="css/cab\_register\_style.css" rel="stylesheet">

</head>

<body class="align">

<div class="site\_\_container">

<div class="grid\_\_container">

<form action="python/customer\_forgot.py" class="form form--login" id="fm\_customer\_forgot" method="post" name="fm\_customer\_forgot">

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Registration No</span></label> <input class="form\_\_input" id="email" name="email" placeholder="Email" required="" type="text"></div>

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Mileage</span></label> <input class="form\_\_input" id="dob" name="dob" placeholder="Date Of Birth" required="" type="text"></div>

<div class="form\_\_field"><input type="submit" value="Get Password"></div>

</form>

</div>

</div>

<script src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>

<script>

$("#fm\_customer\_forgot").submit(function(e){

var email = $('#email').val();

var dob = $('#dob').val();

var formURL = $(this).attr("action");

var data = {

email:email,

dob:dob

}

$.ajax({

url : formURL,

type: "POST",

data : data,

dataType: 'json',

success: function (data) {

if(data[0]==1){

alert('Password: '+data[1])

$("#fm\_customer\_forgot")[0].reset();

}

else

alert('Account not found')

}

});

e.preventDefault();

});

</script>

</body>

</html>

**driver\_homepage.html:**

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Driver</title>

<link href='http://fonts.googleapis.com/css?family=Titillium+Web:400,300,600' rel='stylesheet' type='text/css'>

<link href="css/normalize.css" rel="stylesheet">

<link href="css/style.css" rel="stylesheet">

<link crossorigin="anonymous" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" rel="stylesheet">

<link crossorigin="anonymous" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap-theme.min.css" integrity="sha384-rHyoN1iRsVXV4nD0JutlnGaslCJuC7uwjduW9SVrLvRYooPp2bWYgmgJQIXwl/Sp" rel="stylesheet">

</head>

<body style="background: #c1bdba">

<div class="form" style="max-width:1200px">

<div class="container" style="padding-right:60px">

<nav class="navbar navbar-default">

<div class="container-fluid">

<div class="navbar-header"><a class="navbar-brand" href="#">Driver</a></div>

<div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">

<ul class="nav navbar-nav navbar-right">

<li><a><b>Welcome!</b></a></li>

<li class="dropdown"><a aria-expanded="false" aria-haspopup="true" class="dropdown-toggle" data-toggle="dropdown" role="button"><span aria-hidden="true" class="glyphicon glyphicon-user"></span> Driver<span class="caret"></span></a>

<ul class="dropdown-menu">

<li><a href="driver\_edit\_profile.html"><span aria-hidden="true" class="glyphicon glyphicon-pencil"></span> Edit Profile</a></li>

<li class="divider" role="separator"></li>

<li><a href="python/driver\_logout.py"><span aria-hidden="true" class="glyphicon glyphicon-off"></span> Logout</a></li>

</ul>

</li>

</ul>

</div>

<!-- /.navbar-collapse --></div>

<!-- /.container-fluid --></nav>

<div class="jumbotron">

<ul class="nav nav-tabs">

<li class="active"><a data-toggle="tab" href="#ride\_now">Drive Now!</a></li>

<li><a data-toggle="tab" href="#ride\_history" id="tb\_ride\_history">History</a></li>

</ul>

<div class="tab-content" id="driver\_tabs">

<div class="tab-pane fade in active" id="ride\_now">

<h3>HOME</h3>

<p>Some content.</p>

</div>

<div class="tab-pane fade" id="ride\_history">

<h3>&nbsp;</h3>

<div id="driver\_history\_msg">

<p>Summary of undertaken rides</p>

</div>

<table class="table" id="table\_driver\_history">

<thead>

<tr>

<th>#</th>

<th>Customer</th>

<th>Date</th>

<th>Source</th>

<th>Destination</th>

<th>Pickup Time</th>

<th>Deliver Time</th>

<th>Distance</th>

<th>Amount</th>

</tr>

</thead>

<tbody id="results">

<tr>

<th scope="row">1</th>

<td>[Customer]</td>

<td>[Date]</td>

<td>[Source]</td>

<td>[Destination]</td>

<td>[Up Time]</td>

<td>[Down Time]</td>

<td>[Distance]</td>

<td>[Amount]</td>

</tr>

</tbody>

</table>

</div>

<div class="tab-pane fade" id="menu2">

<h3>Menu 2</h3>

<p>Some content in menu 2.</p>

</div>

</div>

</div>

</div>

</div>

<!--<script src='https://code.jquery.com/jquery-3.1.0.min.js'></script>-->

<script src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

<script src="js/index.js"></script>

<script>

$('#tb\_ride\_history').click(function(e){

var did = '0';

var formURL = 'python/driver\_history.py';

var data = {

did:did

}

$.ajax({

url : formURL,

type: "POST",

data : data,

dataType: 'json',

success: function (data) {

if(data!=''){

$("#table\_driver\_history tbody > tr").remove();

$('#driver\_history\_msg').html(data.length+' records found');

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

$('#table\_driver\_history > tbody:last-child').append('<tr><td>'+(i+1)+'<\/td><td>'+data[i][0]+'<\/td><td>'+data[i][1]+'<\/td><td>'+data[i][2]+'<\/td><td>'+data[i][3]+'<\/td><td>'+data[i][4]+'<\/td><td>'+data[i][5]+'<\/td><td>'+data[i][6]+'<\/td><td>'+data[i][7]+'<\/td><\/tr>');

};

}else{

$('#driver\_history\_msg').html('no records found');

$("#table\_driver\_history tbody > tr").remove();

}

}

});

e.preventDefault();

})

</script>

</body>

</html>

**driver\_edit\_profile.html:**

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Driver Update</title>

<link href="css/cab\_register\_style.css" rel="stylesheet">

</head>

<body class="align" onload="load\_data()">

<div class="site\_\_container">

<div class="grid\_\_container">

<form action="python/driver\_update.py" class="form form--login" id="fm\_driver\_update" method="post" name="fm\_driver\_update">

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Registration No</span></label> <input class="form\_\_input" id="name" name="name" placeholder="Name" required="" type="text"></div>

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Type</span></label> <input class="form\_\_input" id="gender" name="gender" placeholder="Gender" required="" type="text"></div>

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Cab Name</span></label> <input class="form\_\_input" id="dob" name="dob" placeholder="Date Of Birth" required="" type="text"></div>

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Mileage</span></label> <input class="form\_\_input" id="mobile" name="mobile" placeholder="Mobile" required="" type="text"></div>

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Mileage</span></label> <input class="form\_\_input" id="email" name="email" placeholder="Email" required="" type="text"></div>

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Mileage</span></label> <input class="form\_\_input" id="address" name="address" placeholder="Address" required="" type="text"></div>

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Mileage</span></label> <input class="form\_\_input" id="password" name="password" placeholder="Password" required="" type="text"></div>

<div class="form\_\_field"><input type="submit" value="Save"></div>

</form>

</div>

</div>

<script src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>

<script>

$("#fm\_driver\_update").submit(function(e){

var name = $('#name').val();

var gender = $('#gender').val();

var dob = $('#dob').val();

var mobile = $('#mobile').val();

var email = $('#email').val();

var address = $('#address').val();

var password = $('#password').val();

var formURL = $(this).attr("action");

var data = {

name:name,

gender:gender,

dob:dob,

mobile:mobile,

email:email,

address:address,

password:password

}

$.ajax({

url : formURL,

type: "POST",

data : data,

dataType: 'json',

success: function (data) {

if(data[0]==1){

alert('Driver Profile Updated'+data[1]);

$("#fm\_driver\_update")[0].reset();

}

else

alert('Driver Profile Update Failed');

}

});

e.preventDefault();

});

function load\_data() {

var did = 0;

var formURL = 'python/driver\_details.py';

var data = {

did:did

}

$.ajax({

url : formURL,

type: "POST",

data : data,

dataType: 'json',

success: function (data) {

if(data!=''){

document.getElementById('name').value=data[0];

document.getElementById('gender').value=data[1];

document.getElementById('dob').value=data[2];

document.getElementById('mobile').value=data[3];

document.getElementById('email').value=data[4];

document.getElementById('address').value=data[5];

document.getElementById('password').value=data[6];

}

else

alert('Couldn\'t fetch driver profile');

}

});

e.preventDefault();

}

</script>

</body>

</html>

**customer\_homepage.html:**

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Customer</title>

<link href='http://fonts.googleapis.com/css?family=Titillium+Web:400,300,600' rel='stylesheet' type='text/css'>

<link href="css/normalize.css" rel="stylesheet">

<link href="css/style.css" rel="stylesheet">

<link crossorigin="anonymous" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" rel="stylesheet">

<link crossorigin="anonymous" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap-theme.min.css" integrity="sha384-rHyoN1iRsVXV4nD0JutlnGaslCJuC7uwjduW9SVrLvRYooPp2bWYgmgJQIXwl/Sp" rel="stylesheet">

</head>

<body style="background: #c1bdba">

<div class="form" style="max-width:1200px">

<div class="container" style="padding-right:60px">

<nav class="navbar navbar-default">

<div class="container-fluid"><div class="navbar-header">

<a class="navbar-brand" href="#">Customer</a></div>

<!-- Collect the nav links, forms, and other content for toggling -->

<div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">

<ul class="nav navbar-nav navbar-right">

<li><a><b>Welcome!</b></a></li>

<li class="dropdown"><a aria-expanded="false" aria-haspopup="true" class="dropdown-toggle" data-toggle="dropdown" role="button"><span aria-hidden="true" class="glyphicon glyphicon-user"></span> Customer<span class="caret"></span></a>

<ul class="dropdown-menu">

<li><a href="customer\_edit\_profile.html"><span aria-hidden="true" class="glyphicon glyphicon-pencil"></span> Edit Profile</a></li>

<li class="divider" role="separator"></li>

<li><a href="python/customer\_logout.py"><span aria-hidden="true" class="glyphicon glyphicon-off"></span> Logout</a></li>

</ul>

</li>

</ul>

</div>

<div class="jumbotron">

<ul class="nav nav-tabs">

<li class="active"><a data-toggle="tab" href="#ride\_now">Ride Now!</a></li>

<li><a data-toggle="tab" href="#ride\_history" id="tb\_ride\_history">History</a></li>

</ul>

<div class="tab-content">

<div class="tab-pane fade in active" id="ride\_now">

<h3>&nbsp;</h3>

<form action="python/driver\_list.py" id="fm\_driver\_list" method="post" name="fm\_driver\_list">

<div class="container">

<div class="row">

<div class="col-md-2 pull-left">

<div class="btn-group"><select class="form-control" id="type" name="type" style="width: 140px">

<option value="TYPE">Vehicle Type</option>

<option value="SCOOTY">Scooty</option>

<option value="BIKE">Bike</option>

<option value="CAR">Car</option>

</select></div>

</div>

<div class="col-md-4">

<div class="input-group"><span class="input-group-addon" id="sizing-addon2">Source</span> <input aria-describedby="sizing-addon2" class="form-control" id="source" name="source" placeholder="Enter Location" required="" type="text"></div>

</div>

<div class="col-md-4">

<div class="input-group"><span class="input-group-addon" id="sizing-addon2">Destination</span> <input aria-describedby="sizing-addon2" class="form-control" id="destination" name="destination" placeholder="Enter Location" required="" type="text"></div>

</div>

<div class="col-md-2">

<div aria-label="..." class="btn-group" role="group"><button class="btn btn-default" style="width: 140px" type="submit">Search</button></div>

</div>

</div>

</div>

</form>

<h3>&nbsp;</h3>

<div id="driver\_list\_msg">Available Drivers</div>

<table class="table" id="table\_driver\_result">

<thead>

<tr>

<th>#</th>

<th>Driver</th>

<th>Vehicle Type</th>

<th>Vehicle Name</th>

<th>Mileage</th>

</tr>

</thead>

<tbody id="results">

<tr>

<th scope="row">1</th>

<td>admin</td>

<td>Sedan</td>

<td>Maruti Swift</td>

<td>24.00</td>

</tr>

</tbody>

</table>

</div>

<div class="tab-pane fade" id="ride\_history">

<h3>&nbsp;</h3>

<div id="customer\_history\_msg">

<p>Summary of undertaken rides</p>

</div>

<table class="table" id="table\_customer\_history">

<thead>

<tr>

<th>#</th>

<th>Driver</th>

<th>Date</th>

<th>Source</th>

<th>Destination</th>

<th>Up Time</th>

<th>Down Time</th>

<th>Distance</th>

<th>Amount</th>

</tr>

</thead>

<tbody>

<tr>

<th scope="row">1</th>

<td>admin</td>

<td>01-JAN-16</td>

<td>Esplanade</td>

<td>Ultadanga</td>

<td>02:00</td>

<td>04:00</td>

<td>14.02</td>

<td>256.00</td>

</tr>

</tbody>

</table>

</div>

<div class="tab-pane fade" id="menu2">

<h3>Menu 2</h3>

<p>Some content in menu 2.</p>

</div>

</div>

</div>

</div>

</div>

<!--<script src='https://code.jquery.com/jquery-3.1.0.min.js'></script>-->

<script src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

<script src="js/index.js"></script>

<script>

$("#fm\_driver\_list").submit(function(e){

var type = $('#type').val();

var source = $('#source').val();

var destination = $('#destination').val();

var formURL = $(this).attr("action");

var data = {

type:type,

source:source,

destination:destination

}

$.ajax({

url : formURL,

type: "POST",

data : data,

dataType: 'json',

success: function (data) {

if(data!=''){

$("#table\_driver\_result tbody > tr").remove();

$('#driver\_list\_msg').html(data.length+' records found');

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

$('#table\_driver\_result > tbody:last-child').append('<tr><td>'+(i+1)+'<\/td><td>'+data[i][0]+'<\/td><td>'+data[i][1]+'<\/td><td>'+data[i][2]+'<\/td><td>'+data[i][3]+'<\/td><\/tr>');

};

}else{

$('#driver\_list\_msg').html('no records found');

$("#table\_driver\_result tbody > tr").remove();

}

}

});

e.preventDefault();

});

$('#tb\_ride\_history').click(function(e){

var cid = '0';

var formURL = 'python/customer\_history.py';

var data = {

cid:cid

}

$.ajax({

url : formURL,

type: "POST",

data : data,

dataType: 'json',

success: function (data) {

if(data!=''){

$("#table\_customer\_history tbody > tr").remove();

$('#customer\_history\_msg').html(data.length+' records found');

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

$('#table\_customer\_history > tbody:last-child').append('<tr><td>'+(i+1)+'<\/td><td>'+data[i][0]+'<\/td><td>'+data[i][1]+'<\/td><td>'+data[i][2]+'<\/td><td>'+data[i][3]+'<\/td><td>'+data[i][4]+'<\/td><td>'+data[i][5]+'<\/td><td>'+data[i][6]+'<\/td><td>'+data[i][7]+'<\/td><\/tr>');

};

}else{

$('#customer\_history\_msg').html('no records found');

$("#table\_customer\_history tbody > tr").remove();

}

}

});

e.preventDefault();

})

</script>

</body>

</html>

**customer\_edit\_profile.html:**

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Customer Update</title>

<link href="css/cab\_register\_style.css" rel="stylesheet">

</head>

<body class="align" onload="load\_data()">

<div class="site\_\_container">

<div class="grid\_\_container">

<form action="python/customer\_update.py" class="form form--login" id="fm\_customer\_update" method="post" name="fm\_customer\_update">

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Registration No</span></label> <input class="form\_\_input" id="name" name="name" placeholder="Name" required="" type="text"></div>

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Type</span></label> <input class="form\_\_input" id="gender" name="gender" placeholder="Gender" required="" type="text"></div>

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Cab Name</span></label> <input class="form\_\_input" id="dob" name="dob" placeholder="Date Of Birth" required="" type="text"></div>

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Mileage</span></label> <input class="form\_\_input" id="mobile" name="mobile" placeholder="Mobile" required="" type="text"></div>

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Mileage</span></label> <input class="form\_\_input" id="email" name="email" placeholder="Email" required="" type="text"></div>

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Mileage</span></label> <input class="form\_\_input" id="address" name="address" placeholder="Address" required="" type="text"></div>

<div class="form\_\_field"><label for="login\_\_username"><span class="hidden">Mileage</span></label> <input class="form\_\_input" id="password" name="password" placeholder="Password" required="" type="text"></div>

<div class="form\_\_field"><input type="submit" value="Save"></div>

</form>

</div>

</div>

<script src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>

<script>

$("#fm\_customer\_update").submit(function(e){

var name = $('#name').val();

var gender = $('#gender').val();

var dob = $('#dob').val();

var mobile = $('#mobile').val();

var email = $('#email').val();

var address = $('#address').val();

var password = $('#password').val();

var formURL = $(this).attr("action");

var data = {

name:name,

gender:gender,

dob:dob,

mobile:mobile,

email:email,

address:address,

password:password

}

$.ajax({

url : formURL,

type: "POST",

data : data,

dataType: 'json',

success: function (data) {

if(data[0]==1){

alert('Customer Profile Updated'+data[1]);

$("#fm\_customer\_update")[0].reset();

}

else

alert('Customer Profile Update Failed');

}

});

e.preventDefault();

});

function load\_data() {

var cid = 0;

var formURL = 'python/customer\_details.py';

var data = {

cid:cid

}

$.ajax({

url : formURL,

type: "POST",

data : data,

dataType: 'json',

success: function (data) {

if(data!=''){

document.getElementById('name').value=data[0];

document.getElementById('gender').value=data[1];

document.getElementById('dob').value=data[2];

document.getElementById('mobile').value=data[3];

document.getElementById('email').value=data[4];

document.getElementById('address').value=data[5];

document.getElementById('password').value=data[6];

}

else

alert('Couldn\'t fetch customer profile');

}

});

e.preventDefault();

}

</script>

</body>

</html>

**Chapter 10: Coding**

**Coding** is the phase of a software development project where developer's actually input the **source code** into a computer that will be compiled into the final **software program**.

**cab\_register.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

import json

print('Content-type: application/json\r\n\r\n')

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

print(con, ' ', con.version)

cur=con.cursor()

data=cgi.FieldStorage()

sts=-1

cab\_id=-1

reg\_no=data.getvalue('reg\_no')

typ=data.getvalue('type')

name=data.getvalue('name')

cost\_per\_km=data.getvalue('cost\_per\_km')

print('<h1> Cab Details: ', reg\_no, typ, name, cost\_per\_km, '</h1>')

sql="INSERT INTO CAB(REG\_NO, TYPE, NAME, COST\_PER\_KM) VALUES('%s', '%s', '%s', %s)" % (reg\_no, typ, name, cost\_per\_km)

cur.execute(sql)

cur.execute('commit')

sql="SELECT CAB\_ID FROM CAB WHERE REG\_NO = '%s'" % (reg\_no)

cur.execute(sql)

for r in cur:

cab\_id=r[0]

sts=1

print("[\"", sts, "\",\"", cab\_id, "\"]")

**customer\_details.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

import datetime

print('Content-type: application/json\r\n\r\n')

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

cur=con.cursor()

data=cgi.FieldStorage()

cid=data.getvalue('cid')

sts=-1

sql="SELECT \* FROM CUSTOMER WHERE CUSTOMER\_ID = %s" % (cid)

cur.execute(sql)

for r in cur:

name=r[1]

gender=r[2]

dob=r[3].strftime('%d-%m-%Y')

mobile=r[5]

email=r[6]

address=r[7]

password=r[8]

print("[\"", name, "\",\"", gender, "\",\"", dob, "\",\"", mobile, "\",\"", email, "\",\"", address, "\",\"", password, "\"]")

**customer\_forgot.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

import json

print('Content-type: application/json\r\n\r\n')

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

cur=con.cursor()

data=cgi.FieldStorage()

sts=-1

password=-1

email=data.getvalue('email')

dob=data.getvalue('dob')

sql="SELECT PASSWORD FROM CUSTOMER WHERE EMAIL = '%s' AND DOB = '%s'" % (email, dob)

cur.execute(sql)

for r in cur:

password=r[0]

sts=1

print("[\"", sts, "\",\"", password, "\"]")

**customer\_history.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

import datetime

print('Content-type: application/json\r\n\r\n')

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

cur=con.cursor()

data=cgi.FieldStorage()

did=data.getvalue('cid')

sts=-1

sql="SELECT D.NAME, B.BILL\_DATE, R.SOURC, R.DESTN, R.UPTIME, R.DOWNTIME, R.DISTANCE, B.AMOUNT FROM BILL B, CUSTOMER C, DRIVER D, RIDE R WHERE R.DRIVER\_ID = D.DRIVER\_ID AND R.CUSTOMER\_ID = C.CUSTOMER\_ID AND B.RIDE\_ID = R.RIDE\_ID AND C.CUSTOMER\_ID = %s" % (did)

cur.execute(sql)

no\_of\_row=1

for r in cur:

no\_of\_row=cur.rowcount

cur.execute(sql)

print("[")

i=1

for r in cur:

driver\_name=r[0]

bill\_date=r[1].strftime('"%d-%m-%Y')

source=r[2]

destination=r[3]

uptime=r[4].strftime('%H:%M')

downtime=r[5].strftime('%H:%M')

distance=r[6]

amount=r[7]

print("[\"", driver\_name, "\",\"", bill\_date, "\",\"", source, "\",\"", destination, "\",\"", uptime, "\",\"", downtime, "\",\"", distance, "\",\"", amount, "\"]")

if i!=no\_of\_row:

print(",")

i=i+1

print("]")

**customer\_login.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

cur=con.cursor()

data=cgi.FieldStorage()

email=data.getvalue('email')

password=data.getvalue('password')

sts=-1

sql="SELECT \* FROM CUSTOMER"

cur.execute(sql)

for r in cur:

db\_email=r[6]

db\_password=r[8]

if db\_email==email and db\_password==password:

sts=1

break

if sts==1:

sql="SELECT \* FROM CUSTOMER WHERE EMAIL = '%s' AND PASSWORD = '%s'" % (email, password)

cur.execute(sql)

for r in cur:

cid=r[0]

name=r[1]

print("location: ../customer\_homepage.html\r\n\r\n")

else:

print("location: ../customer.html\r\n\r\n")

**customer\_logout.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

cur=con.cursor()

data=cgi.FieldStorage()

cid=data.getvalue('cid')

sts=-1

if sts==-1:

sql="UPDATE CUSTOMER SET STATUS = 'OFFLINE' WHERE CUSTOMER\_ID = %s" % (cid)

cur.execute(sql)

cur.execute("commit")

print("location: ../customer.html\r\n\r\n")

else:

print("location: ../customer\_homepage.html\r\n\r\n")

**customer\_register.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

#print('Content-type: text/html\r\n\r\n')

print('Content-type: application/json\r\n\r\n')

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

cur=con.cursor()

data=cgi.FieldStorage()

sts=-1

customer\_id=-1

name=data.getvalue('name')

gender=data.getvalue('gender')

dob=data.getvalue('dob')

mobile=data.getvalue('mobile')

email=data.getvalue('email')

address=data.getvalue('address')

password=data.getvalue('password')

sql="INSERT INTO CUSTOMER(NAME, GENDER, DOB, STATUS, MOBILE, EMAIL, ADDRESS, PASSWORD) VALUES('%s', '%s', '%s', 'OFFLINE', '%s', '%s', '%s', '%s')" % (name, gender, dob, mobile, email, address, password)

#print('<h1>', sql, '</h1>')

cur.execute(sql)

cur.execute('commit')

sql="SELECT CUSTOMER\_ID FROM CUSTOMER WHERE MOBILE = '%s' AND PASSWORD = '%s'" % (mobile, password)

cur.execute(sql)

for r in cur:

customer\_id=r[0]

sts=1

print("[\"", sts, "\",\"", customer\_id, "\"]")

**customer\_update.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

print('Content-type: application/json\r\n\r\n')

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

cur=con.cursor()

data=cgi.FieldStorage()

sts=-1

cid=data.getvalue('cid')

name=data.getvalue('name')

gender=data.getvalue('gender')

dob=data.getvalue('dob')

mobile=data.getvalue('mobile')

email=data.getvalue('email')

address=data.getvalue('address')

password=data.getvalue('password')

sql="UPDATE CUSTOMER SET NAME = '%s', GENDER = '%s', DOB = '%s', MOBILE = '%s', EMAIL = '%s', ADDRESS = '%s', PASSWORD = '%s' WHERE CUSTOMER\_ID = %s" % (name, gender, dob, mobile, email, address, password, cid)

cur.execute(sql)

cur.execute('commit')

sts=1

print("[\"",sts,"\"]")

**driver\_details.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

import datetime

print('Content-type: application/json\r\n\r\n')

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

cur=con.cursor()

data=cgi.FieldStorage()

did=data.getvalue('did')

sts=-1

sql="SELECT \* FROM DRIVER WHERE DRIVER\_ID = %s" % (did)

cur.execute(sql)

for r in cur:

name=r[2]

gender=r[3]

dob=r[4].strftime('%d-%m-%Y')

mobile=r[6]

email=r[7]

address=r[8]

password=r[9]

print("[\"", name, "\",\"", gender, "\",\"", dob, "\",\"", mobile, "\",\"", email, "\",\"", address, "\",\"", password, "\"]")

**driver\_forgot.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

import json

print('Content-type: application/json\r\n\r\n')

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

cur=con.cursor()

data=cgi.FieldStorage()

sts=-1

password=-1

email=data.getvalue('email')

dob=data.getvalue('dob')

sql="SELECT PASSWORD FROM DRIVER WHERE EMAIL = '%s' AND DOB = '%s'" % (email, dob)

cur.execute(sql)

for r in cur:

password=r[0]

sts=1

print("[\"", sts, "\",\"", password, "\"]")

**driver\_history.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

import datetime

print('Content-type: application/json\r\n\r\n')

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

cur=con.cursor()

data=cgi.FieldStorage()

did=data.getvalue('did')

sts=-1

sql="SELECT C.NAME, B.BILL\_DATE, R.SOURC, R.DESTN, R.UPTIME, R.DOWNTIME, R.DISTANCE, B.AMOUNT FROM BILL B, CUSTOMER C, DRIVER D, RIDE R WHERE R.DRIVER\_ID = D.DRIVER\_ID AND R.CUSTOMER\_ID = C.CUSTOMER\_ID AND B.RIDE\_ID = R.RIDE\_ID AND D.DRIVER\_ID = %s" % (did)

cur.execute(sql)

no\_of\_row=1

for r in cur:

no\_of\_row=cur.rowcount

cur.execute(sql)

print("[")

i=1

for r in cur:

customer\_name=r[0]

bill\_date=r[1].strftime('"%d-%m-%Y')

source=r[2]

destination=r[3]

uptime=r[4].strftime('%H:%M')

downtime=r[5].strftime('%H:%M')

distance=r[6]

amount=r[7]

print("[\"", customer\_name, "\",\"", bill\_date, "\",\"", source, "\",\"", destination, "\",\"", uptime, "\",\"", downtime, "\",\"", distance, "\",\"", amount, "\"]")

if i!=no\_of\_row:

print(",")

i=i+1

print("]")

**driver\_list.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

import json

print('Content-type: application/json\r\n\r\n')

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

cur=con.cursor()

data=cgi.FieldStorage()

source=data.getvalue('source')

destination=data.getvalue('destination')

typ=data.getvalue('type')

sts=-1

if typ=="TYPE":

sql="SELECT D.NAME, C.TYPE, C.NAME, C.COST\_PER\_KM FROM DRIVER D, CAB C WHERE D.CAB\_ID = C.CAB\_ID"

else:

sql="SELECT D.NAME, C.TYPE, C.NAME, C.COST\_PER\_KM FROM DRIVER D, CAB C WHERE D.CAB\_ID = C.CAB\_ID AND C.TYPE = '%s'" % (typ)

cur.execute(sql)

no\_of\_row=1

for r in cur:

no\_of\_row=cur.rowcount

cur.execute(sql)

print("[")

i=1

for r in cur:

driver\_name=r[0]

vehicle\_type=r[1]

vehicle\_name=r[2]

mileage=r[3]

print("[\"", driver\_name, "\",\"", vehicle\_type, "\",\"", vehicle\_name, "\",\"", mileage, "\"]")

if i!=no\_of\_row:

print(",")

i=i+1

print("]")

**driver\_login.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

import beaker

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

cur=con.cursor()

data=cgi.FieldStorage()

email=data.getvalue('log\_email')

password=data.getvalue('log\_password')

sts=-1

sql="SELECT \* FROM DRIVER"

cur.execute(sql)

for r in cur:

db\_email=r[7]

db\_password=r[9]

if db\_email==email and db\_password==password:

sts=1

break

if sts==1:

sql="SELECT \* FROM DRIVER WHERE EMAIL = '%s' AND PASSWORD = '%s'" % (email, password)

cur.execute(sql)

for r in cur:

did=r[0]

name=r[2]

print("location: ../driver\_homepage.html\r\n\r\n")

else:

print("location: ../driver.html\r\n\r\n")

**driver\_logout.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

cur=con.cursor()

data=cgi.FieldStorage()

did=data.getvalue('did')

sts=-1

if sts==-1:

sql="UPDATE DRIVER SET STATUS = 'OFFLINE' WHERE DRIVER\_ID = %s" % (did)

cur.execute(sql)

cur.execute("commit")

print("location: ../driver.html\r\n\r\n")

else:

print("location: ../driver\_homepage.html\r\n\r\n")

**driver\_register.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

import json

print('Content-type: application/json\r\n\r\n')

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

cur=con.cursor()

data=cgi.FieldStorage()

sts=-1

driver\_id=-1

cab\_id=data.getvalue('cab\_id')

name=data.getvalue('name')

gender=data.getvalue('gender')

dob=data.getvalue('dob')

mobile=data.getvalue('mobile')

email=data.getvalue('email')

address=data.getvalue('address')

password=data.getvalue('password')

sql="INSERT INTO DRIVER(CAB\_ID, NAME, GENDER, DOB, STATUS, MOBILE, EMAIL, ADDRESS, PASSWORD) VALUES(%s, '%s', '%s', '%s', 'OFFLINE', '%s', '%s', '%s', '%s')" % (cab\_id, name, gender, dob, mobile, email, address, password)

cur.execute(sql)

cur.execute('commit')

sql="SELECT DRIVER\_ID FROM DRIVER WHERE CAB\_ID = %s" % (cab\_id)

cur.execute(sql)

for r in cur:

driver\_id=r[0]

sts=1

print("[\"", sts, "\",\"", driver\_id, "\"]")

**driver\_update.py:**

#!C:\Python34\python.exe

import cgi

import cx\_Oracle

print('Content-type: application/json\r\n\r\n')

con=cx\_Oracle.connect('cbs/apss@localhost/xe')

cur=con.cursor()

data=cgi.FieldStorage()

sts=-1

did=data.getvalue('did')

name=data.getvalue('name')

gender=data.getvalue('gender')

dob=data.getvalue('dob')

mobile=data.getvalue('mobile')

email=data.getvalue('email')

address=data.getvalue('address')

password=data.getvalue('password')

sql="UPDATE DRIVER SET NAME = '%s', GENDER = '%s', DOB = '%s', MOBILE = '%s', EMAIL = '%s', ADDRESS = '%s', PASSWORD = '%s' WHERE DRIVER\_ID = %s" % (name, gender, dob, mobile, email, address, password, did)

cur.execute(sql)

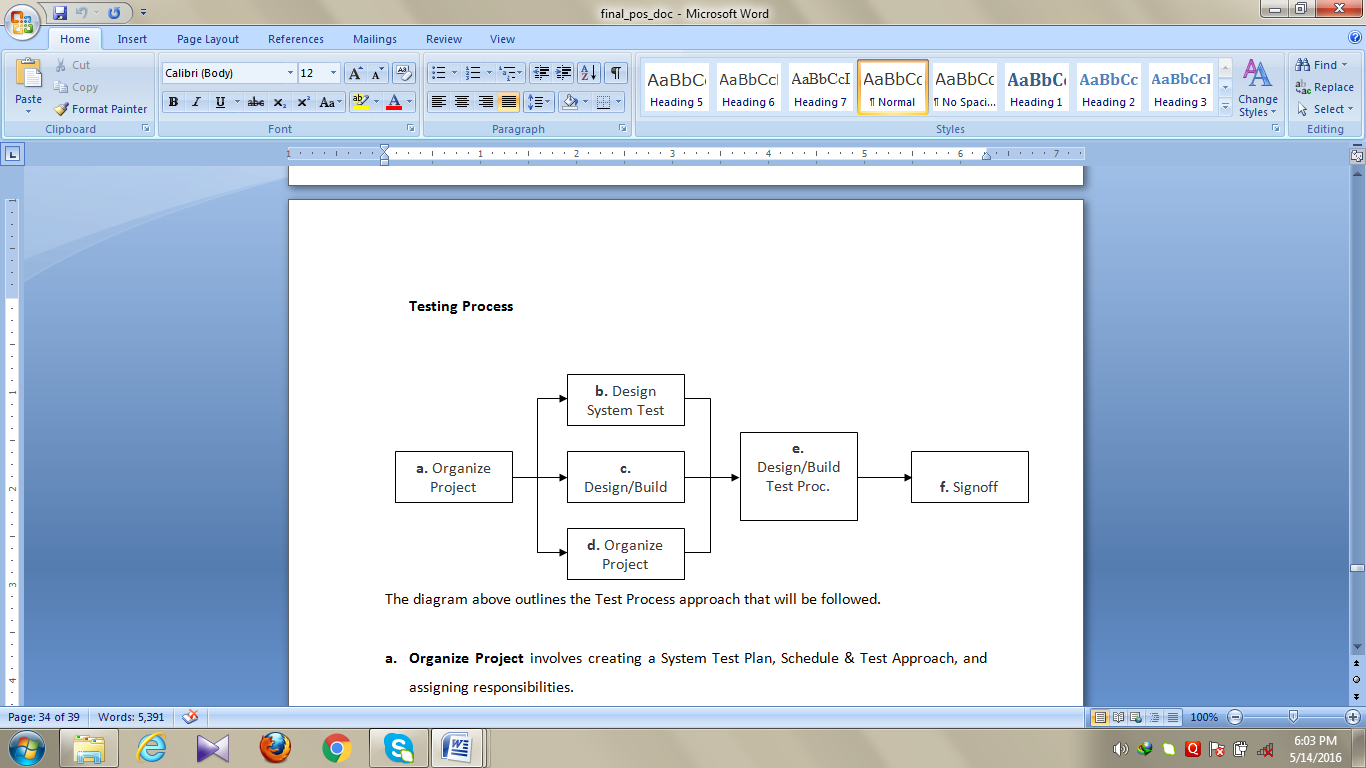
cur.execute('commit')

sts=1

print("[\"",sts,"\"]")

**Chapter 11: Testing**

**Testing Process**

****

The diagram above outlines the Test Process approach that will be followed.

**a. Organize Project** involves creating a System Test Plan, Schedule & Test Approach, and assigning responsibilities.

**b.** **Design/Build System Test** involves identifying Test Cycles, Test Cases, Entrance & Exit Criteria, Expected Results, etc. In general, test conditions/expected results will be identified by the Test Team in conjunction with the Development Team. The Test Team will then identify Test Cases and the Data required. The Test conditions are derived from the Program Specifications Document.

**c.** **Design/Build Test Procedures** includes setting up procedures such as Error Management systems and Status reporting.

**d.** **Build Test Environment** includes requesting/building hardware, software and data set-ups.

**e. Execute System Tests –** The tests identified in the Design/Build Test Procedures will be executed. All results will be documented and Bug Report Forms filled out and given to the Development Team as necessary.

**f.** **Signoff** - Signoff happens when all pre-defined exit criteria have been achieved.

# 

# Testing Strategy

The following outlines the types of testing that will be done for unit, integration, and system testing. While it includes what will be tested, the specific use cases that determine how the testing is done will be detailed in the Test Design Document. The test cases that will be used for designing use cases is shown in Figure 2.1 and onwards.

**Test Cases**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tested By:** | | Sayan Basu, Soumeet Basak | |
| **Test Type** | | Unit Testing | |
| **Test Case Number** | | 1 | |
| **Test Case Name** | | User Identification | |
| **Test Case Description** | | The user should enter his/ her accurate userid and password so that he/she can able to go for the further options. The test case will check the application for the same since a user can only login with the correct userid , password. | |
| **Item(s) to be tested** | | | |
| 1 | Verification of the userid and password with the record in the database. | | |
| **Specifications** | | | |
| **Input** | | | **Expected**  **Output/Result** |
| 1. Correct User id and password      1. Incorrect Id or Password | | | 1. Successful login 2. Failure Message |

|  |  |  |  |
| --- | --- | --- | --- |
| **Tested By:** | | Pallavi Sharma , Arundhati Dutta | |
| **Test Type** | | Unit Testing | |
| **Test Case Number** | | 2 | |
| **Test Case Name** | | Booking identification | |
| **Test Case Description** | | Driver login is must to accept ride request. | |
| **Item(s) to be tested** | | | |
| 1 | Check whether the driver logged in. | | |
| **Specifications** | | | |
| **Input** | | | **Expected**  **Output/Result** |
| 1. Customer id, Customer name, Address | | | Successfully booked. |

## 

## Unit Testing

Unit Testing is done at the source or code level for language-specific programming errors such as bad syntax, logic errors, or to test particular functions or code modules. The unit test cases shall be designed to test the validity of the programs correctness.

### White Box Testing

In white box testing, the UI is bypassed. Inputs and outputs are tested directly at the code level and the results are compared against specifications. This form of testing ignores the function of the program under test and will focus only on its code and the structure of that code. Test case designers shall generate cases that not only cause each condition to take on all possible values at least once, but that cause each such condition to be executed at least once. To ensure this happens, we will be applying Branch Testing. Because the functionality of the program is relatively simple, this method will be feasible to apply.

Each function of the binary tree repository is executed independently; therefore, a program flow for each function has been derived from the code.

### Black Box Testing

Black box testing typically involves running through every possible input to verify that it results in the right outputs using the software as an end-user would. We have decided to perform Equivalence Partitioning and Boundary Value Analysis testing on our application.

## 

## System Testing

The goals of system testing are to detect faults that can only be exposed by testing the entire integrated system or some major part of it. Generally, system testing is mainly concerned with areas such as performance, security, validation, load/stress, and configuration sensitivity. But in our case well focus only on function validation and performance. And in both cases we will use the black-box method of testing.

**Decision Table based testing**

**Test case Login**

|  |  |  |  |
| --- | --- | --- | --- |
| **Condition** | Email id | Password | **Action (Expected result)** |
| C1. | Yes | Yes | Correct |
| C2. | Yes | No | Incorrect |
| C3. | No | Yes | Incorrect |
| C4 | No | No | Incorrect |

**Test case Registration**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Condition** | Name | Email id | Password | Confirm password | Address | **Action**  **(Expected result)** |
| C1. | Yes | Yes | Yes | Yes | Yes | Correct |
| C2. | Yes | Yes | Yes | Yes | No | Incorrect |
| C3. | Yes | Yes | Yes | No | Yes | Incorrect |
| C4. | Yes | Yes | No | Yes | Yes | Incorrect |
| C5. | Yes | No | Yes | Yes | Yes | Incorrect |
| C6. | Yes | Yes | Yes | Yes | Yes | Incorrect |
| C7. | No | Yes | Yes | Yes | Yes | Incorrect |
| …. | ..… | ….. | ….. | …… | ….. | …. |
| C62. | No | No | No | No | No | Incorrect |
| C63. | Yes | No | No | No | No | Incorrect |
| C64: | No | No | No | No | No | Incorrect |

**Unit Testing**

This type of testing is performed by the developers before the setup is handed over to the testing team to formally execute the test cases. Unit testing is performed by the respective developers on the individual units of source code assigned areas. The developers use test data that is separate from the test data of the quality assurance team.

The goal of unit testing is to isolate each part of the program and show that individual parts are correct in terms of requirements and functionality. In our Food Recipe Web Portal project all the code units are tested properly.

**Integration Testing**

The testing of combined parts of an application to determine if they function correctly together is Integration testing. There are few methods of doing integration testing.

1. Top-Down.
2. Bottom-Up.
3. Sandwich Testing.

**Top-Down Integration Testing**

In top-down integration testing, the highest-level modules are tested first and progressively lower-level modules are tested after that. In a comprehensive software development environment, bottom-up testing is usually done first, followed by top-down testing.

Bottom-Up.

In bottom-up integration testing, the lowest-level modules are tested first and progressively higher-level modules are tested after that.

Sandwich Testing

It is a mixed approach of bottom up and top down testing.

**System Testing**

This is the next level in the testing and tests the system as a whole. Once all the components are integrated, the application as a whole is tested rigorously to see that it meets Quality Standards. This type of testing is performed by a specialized testing team.

**Why is System Testing so Important?**

**System Testing** is the first step in the Software Development Life Cycle, where the application is tested as a whole.

**The application** is tested thoroughly to verify that it meets the functional and technical specifications. The application **is tested in an environment** which is very close to the production environment where the application will be deployed.

**System Testing** enables us to test, verify and validate both the business requirements as well as the Applications Architecture.

**Performance Testing:** It is a process of measuring various efficiency characteristics of a system such as response time, through put, load, stress transactions per minutes, transaction mix.

**Types of Performance Testing**

**Load Testing:** Load Testing is type of performance testing to check system with constantly increasing the load on the system until the time load is reaches to its threshold value. Here Increasing load means increasing number of concurrent users, transactions & check the behavior of application under test.

**Stress Testing:** Stress Testing is performance testing type to check the stability of software when hardware resources are not sufficient like CPU, memory, disk space etc. We have done all this kind of system testing and it work according its functional requirement.

**Acceptance Testing**

It is one kind of testing which is done by the client not by the developer or tester. The main goal of this testing is demonstrate the system is ready or not. Here, choice of test is done by the client.

**Functional System Testing**

**Regression Testing**

Whenever a change in a software application is made it is quite possible that other areas within the application have been affected by this change. To verify that a fixed bug hasn’t resulted in another functionality or business rule violation is Regression testing. The intent of Regression testing is to ensure that a change, such as a bug fix did not result in another fault being uncovered in the application. Importance of regression testing:

* **Minimize the gaps** in testing when an application with changes made has to be tested.
* **Testing the new changes** to verify that the change made did not affect any other area of the application.
* **Test coverage** is increased without compromising timelines.
* **Increase speed** to market the product.

**Non Functional System Testing**

**UI / GUI Testing:** Validating if all user interfaces are professionally designed or not is called UI Testing. Through UI testing we verify the followings:

1. Check if all basic elements are available in the page or not.
2. Check spelling of the objects.
3. Check alignments of the objects.
4. Check content displayed in web pages.
5. Check if the mandatory fields are highlights or not.
6. Check consistency in background color, font type and fond size etc.

**Installation Testing:** checking if we are able to install the software successfully or not as per the guidelines given in installation document.

**Un-Installation Testing:** checking if we are able to uninstall the software from the system successfully or not.

**Compatibility Testing:** checking if the application is compatible with the different software and hardware environment.

**Cross-Browser Testing:** checking if the application is running with different browser or not.

**System Security**

There are several issues with respect to system security

1. **Authorized:** Only authorized users are allowed to access the particular application, which suggests that only logged in user can access the inner pages of this application.
2. If a user types any url to access a page for which he is not authorized then the page will redirect him to Index page.

**Chapter 12: Maintains and cost estimation**

**System Security measures (Implementation of security for the project developed)**

* Only authorized users are allowed.
* Without signing in users are not allowed to go an intermediate page by typing an URL. For all such efforts, users will be redirected to the home page.

**Database/Data security**

* Database is present in remote machine.
* SQL server’s default securities are applied.

**Creation of User profiles and access rights**

* The admin must create users manually
* The admin can create more admin

**Cost Estimation of the Project along with Cost Estimation Model**

# Analogous estimate of effort or cost

Used for Early Estimate or Individual Activity Estimate

Sample example shown below is for two major deliverables of a software project. You use a previous project as a benchmark for analogous estimation. Using your experience you will estimate a multiplier.

Multipliers:

1. Prototyping: 0.75.
2. Testing: 0.5
3. Deployment: 0.5

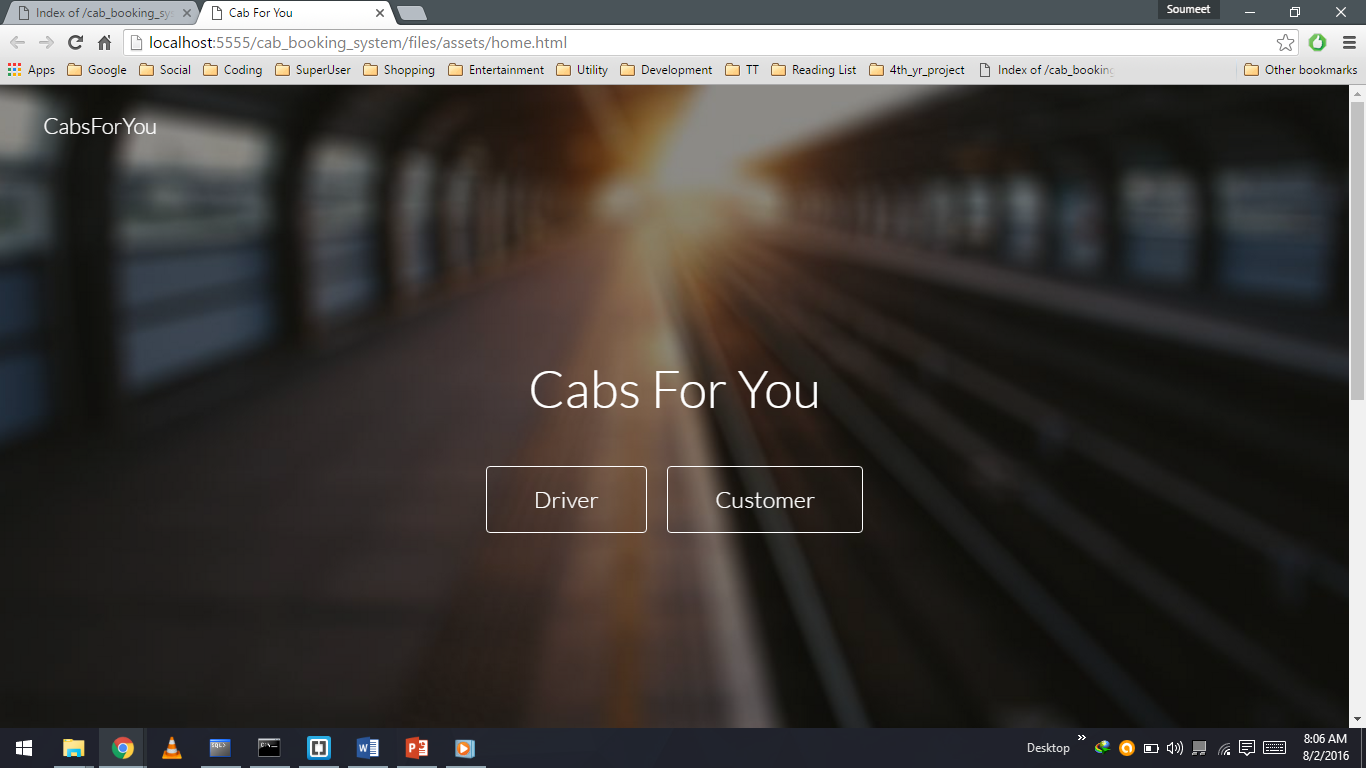
Finally, if you want to convert to cost, you would use current rates for the resource.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **WBS ID** | **Previous**  **Similar Project**  **Activity** | **Previous**  **Effort** | **Current**  **Project**  **Estimate** | **Multiplier** | **Effort**  **(Previous Effort \* 0.75)** | **Cost**  **(Rs. 500/hr.)** |
| 1 | Prototyping | 40 Work-Hours | Prototyping | 0.75 | 30 Work-hours | Rs. 15000/- |
| 2 | Testing | 20 Work-Hours | Testing | 0.50 | 10 Work-Hours | Rs. 5000/- |
| **Total** |  |  |  |  | **40 Work- Hours** | **Rs. 20000/-** |

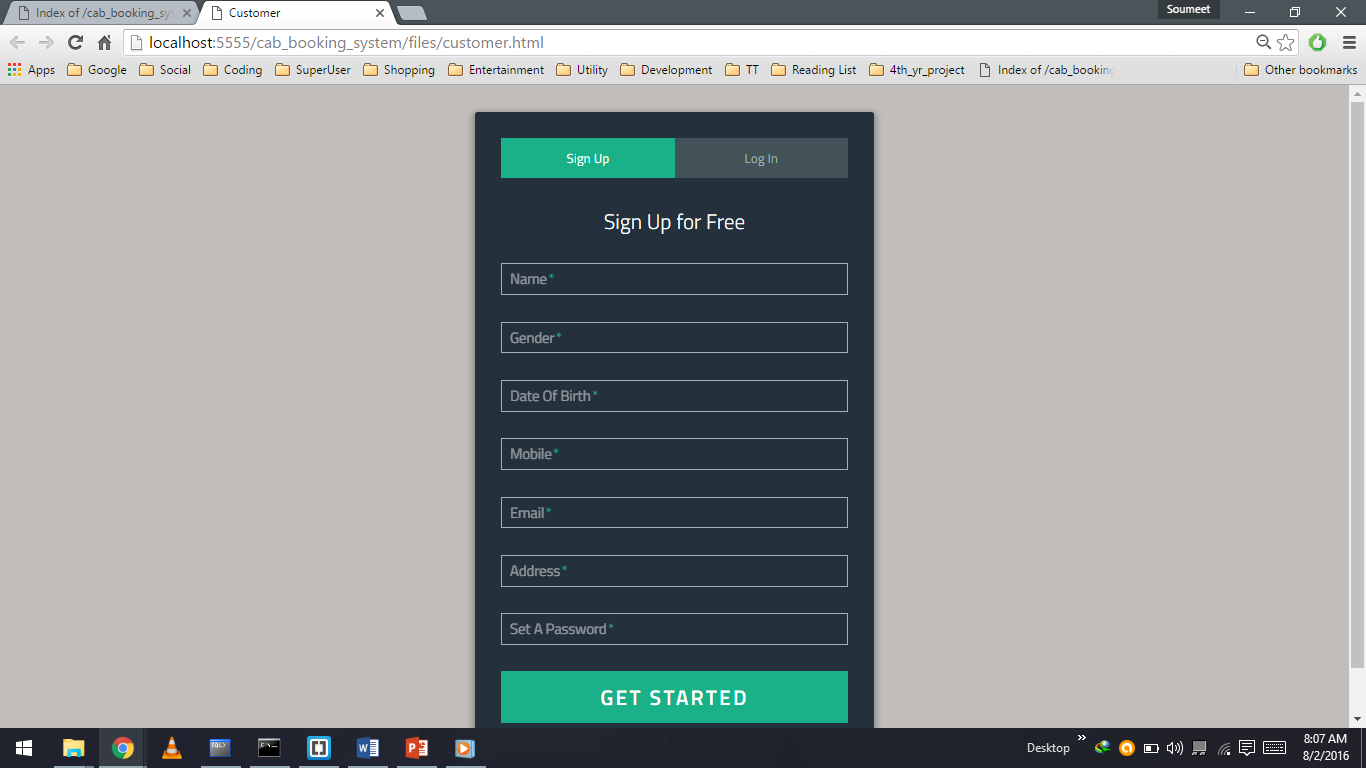
Note: Effort is also called Size and unit of estimation is called either Work-Hour, person-hours..

**Chapter 13: Project Report**

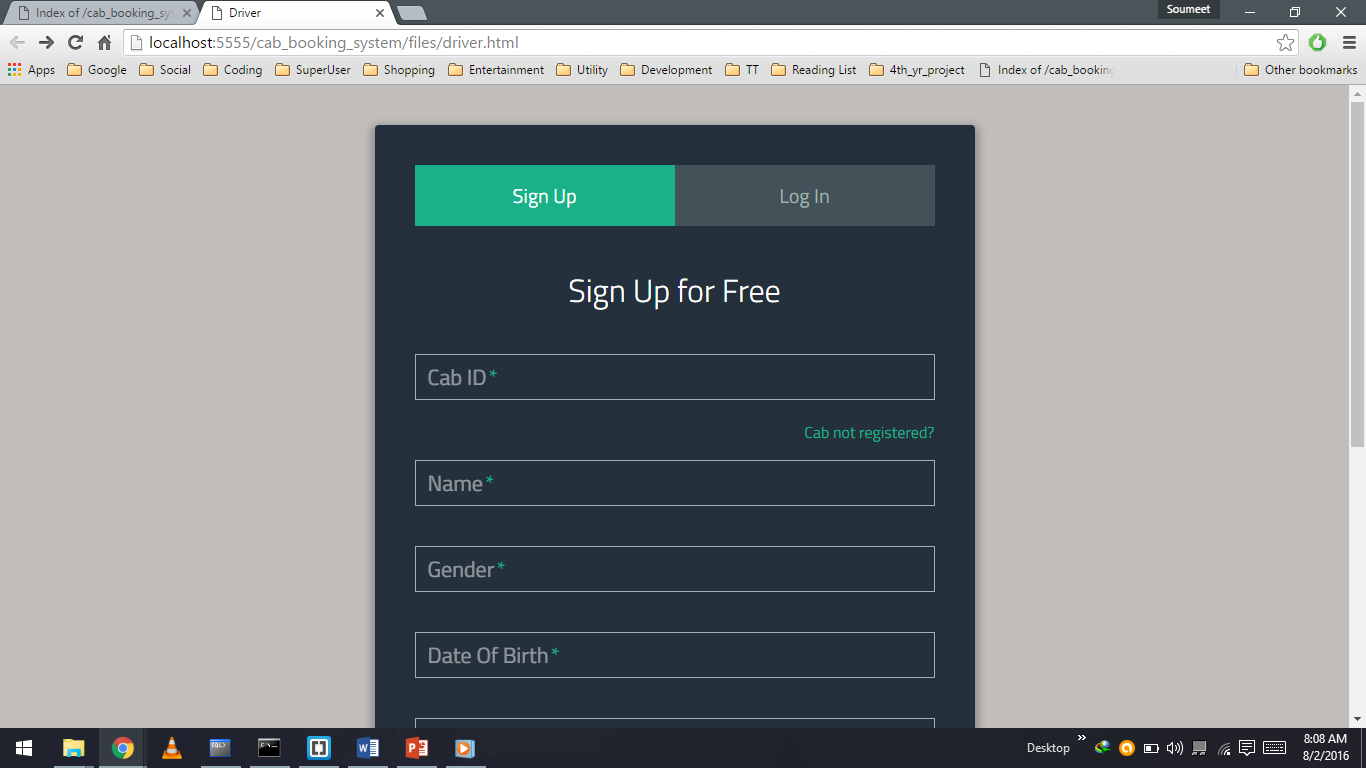
**Home**

****

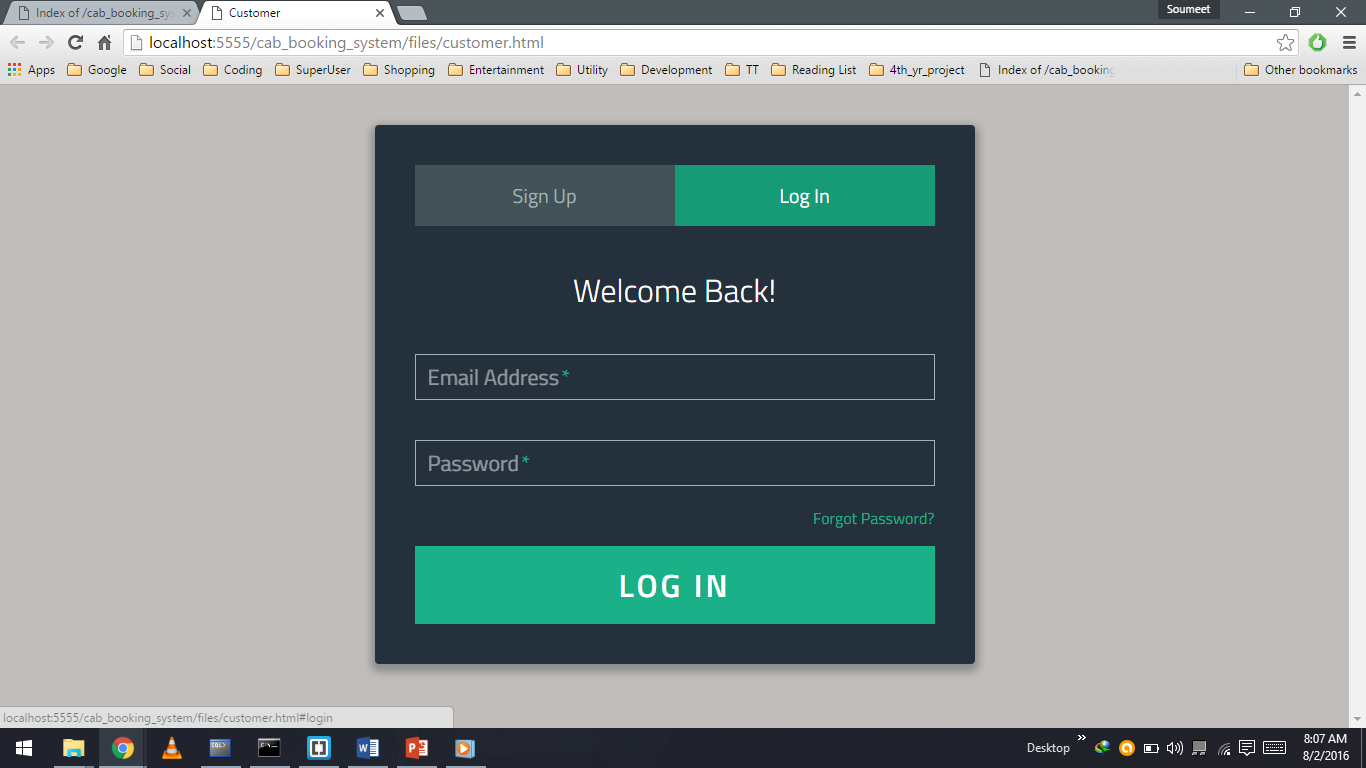
**Customer Register**

****

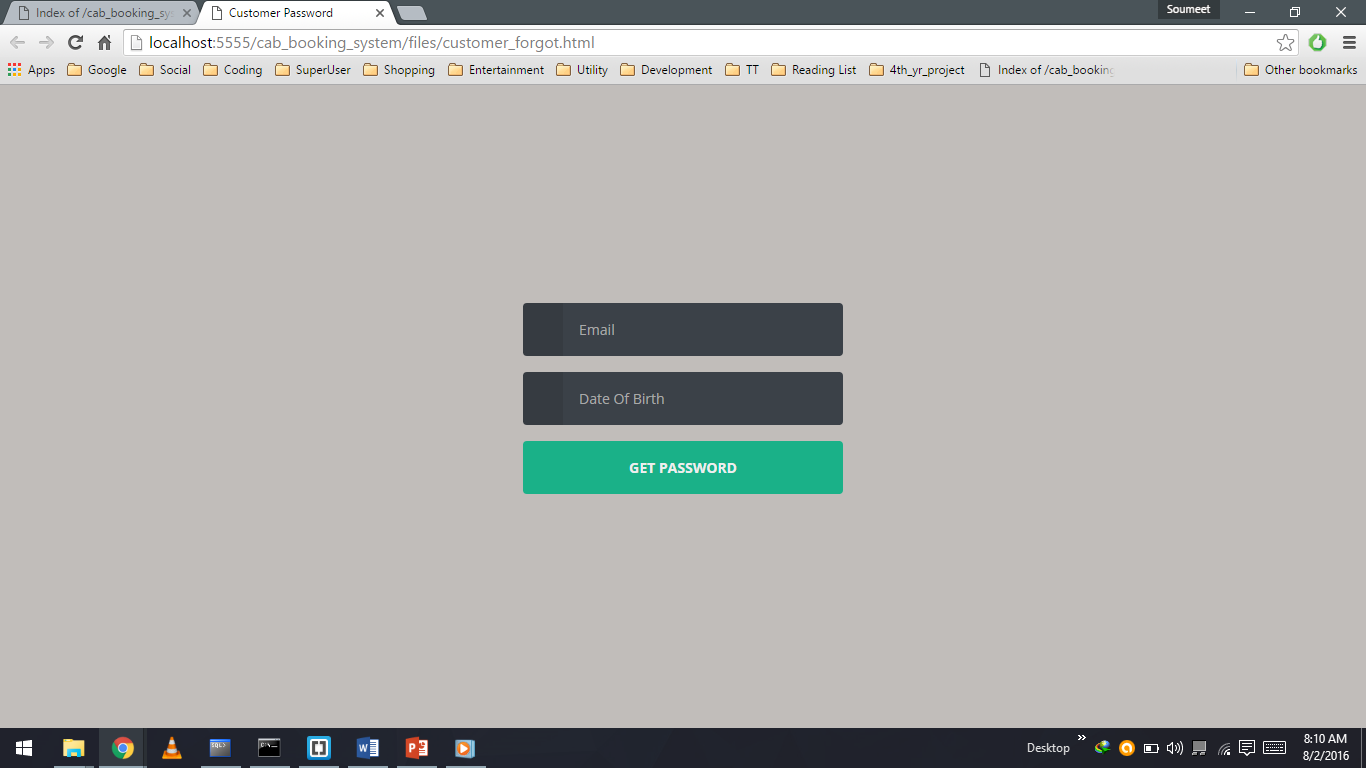
**Driver Register**

****

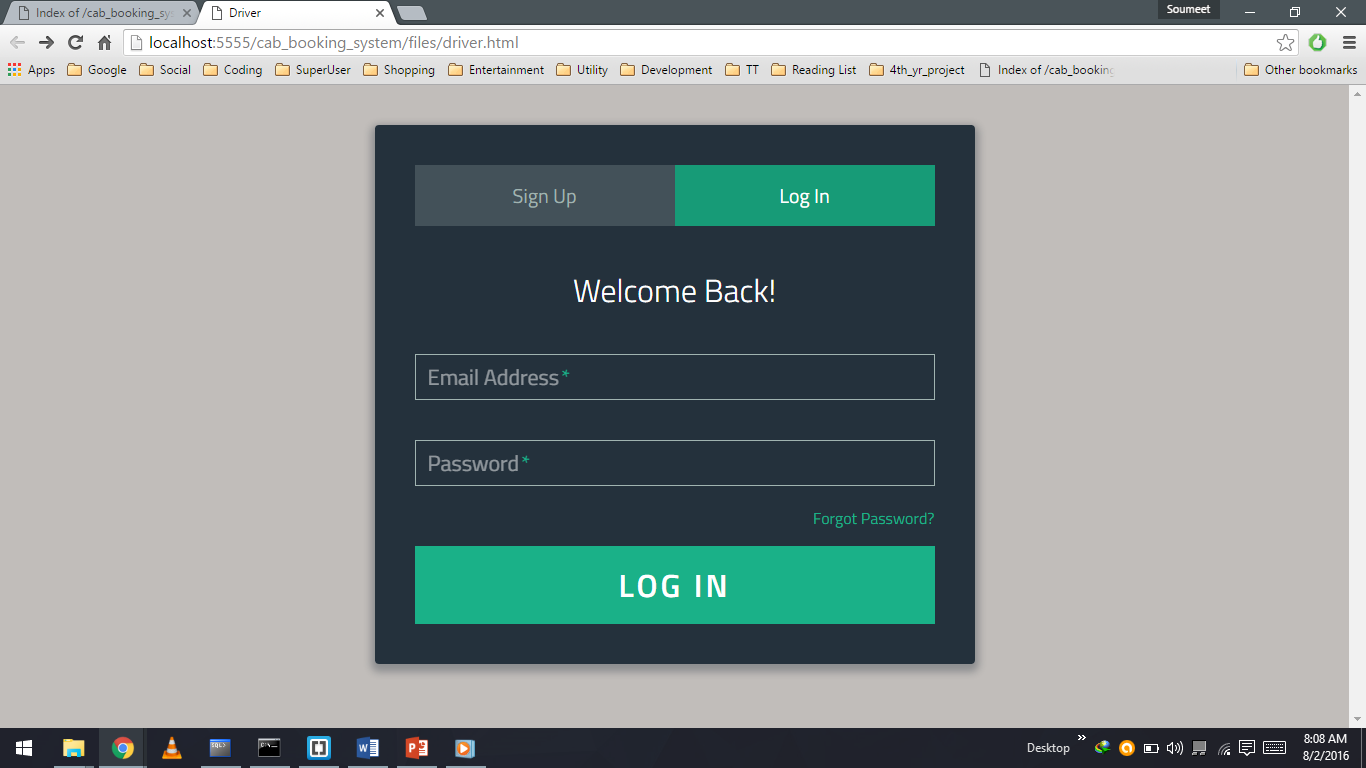
**Customer Login**

****

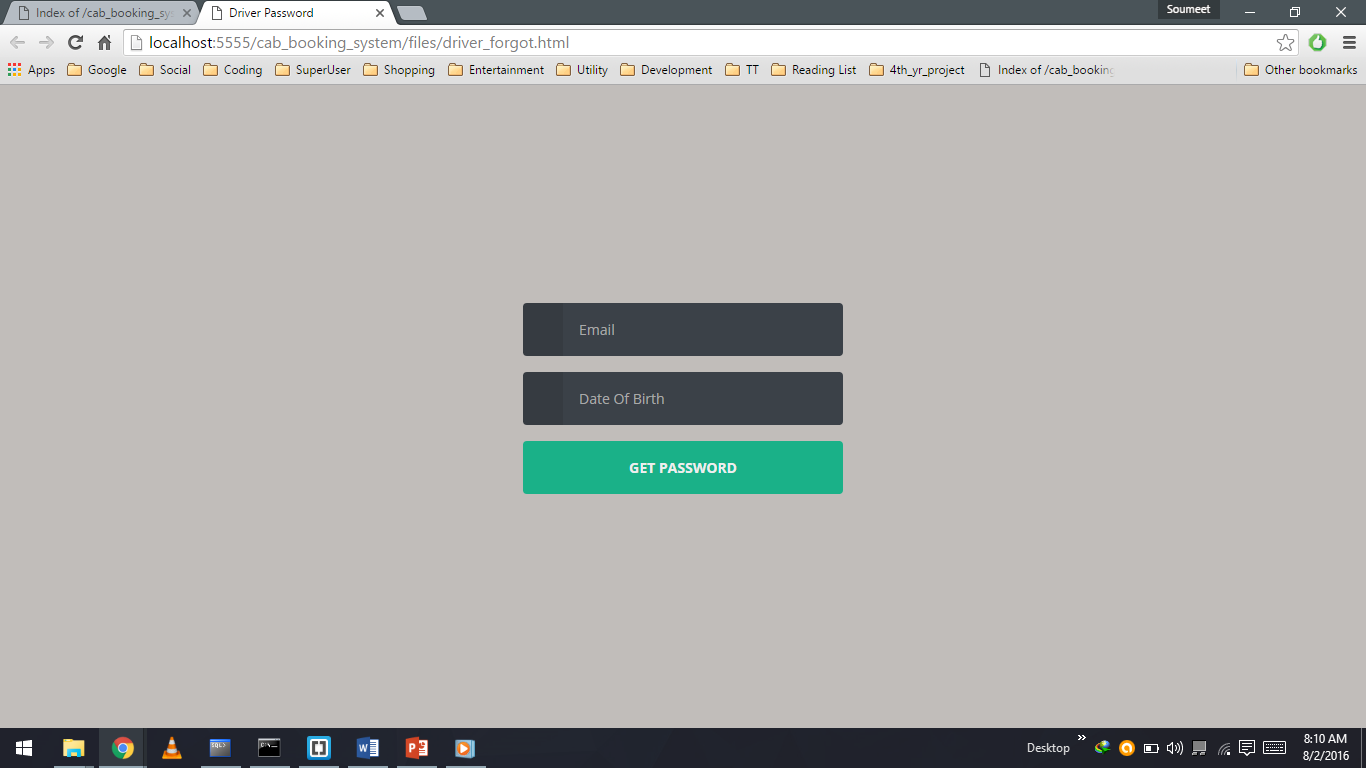
**Customer Forgot Password**

****

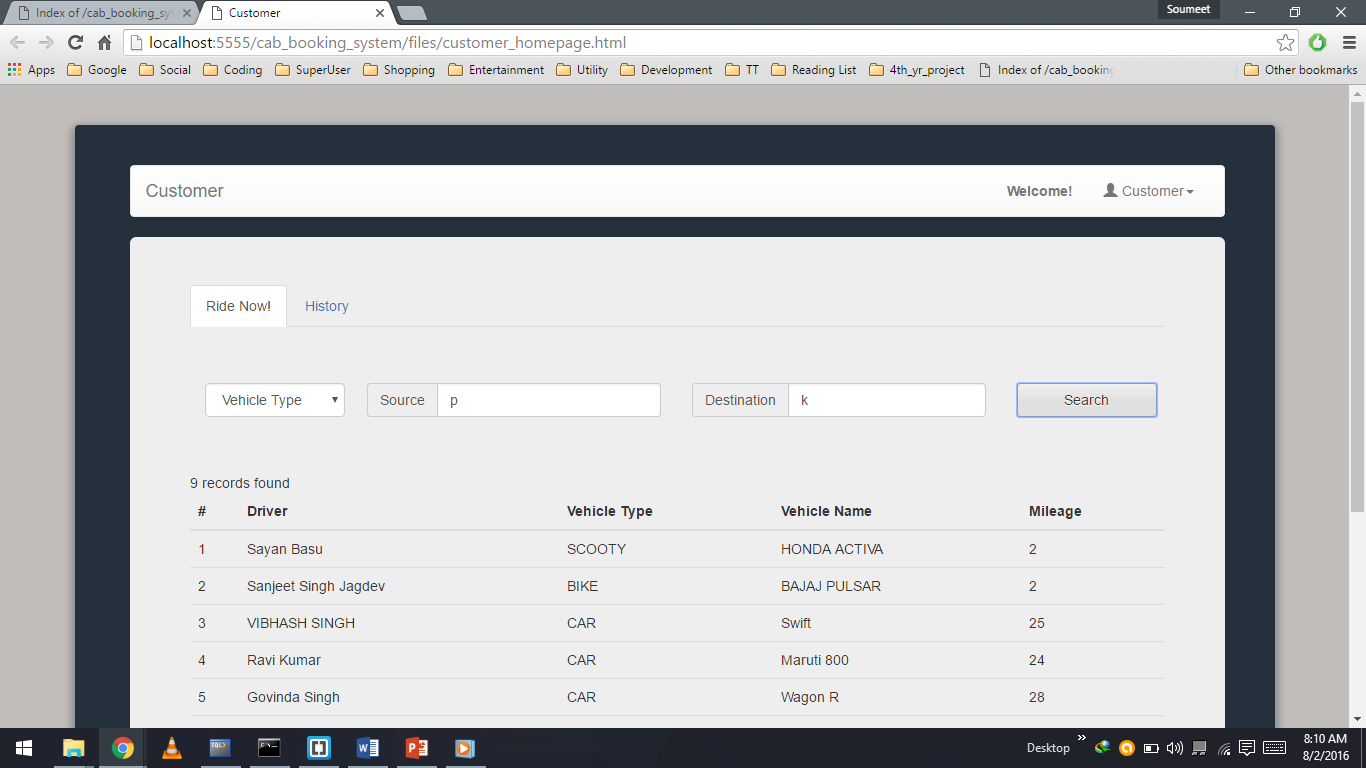
**Driver Login**

****

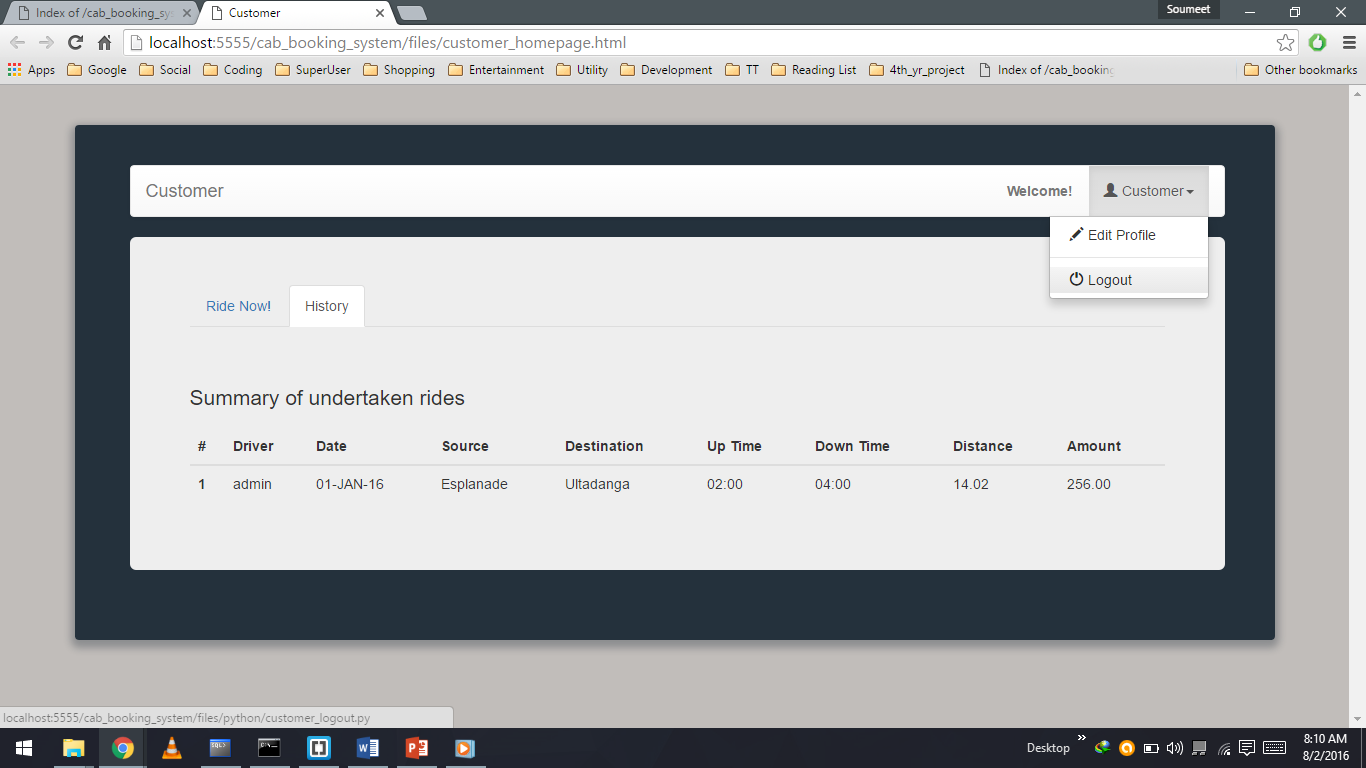
**Driver Forgot Password**

****

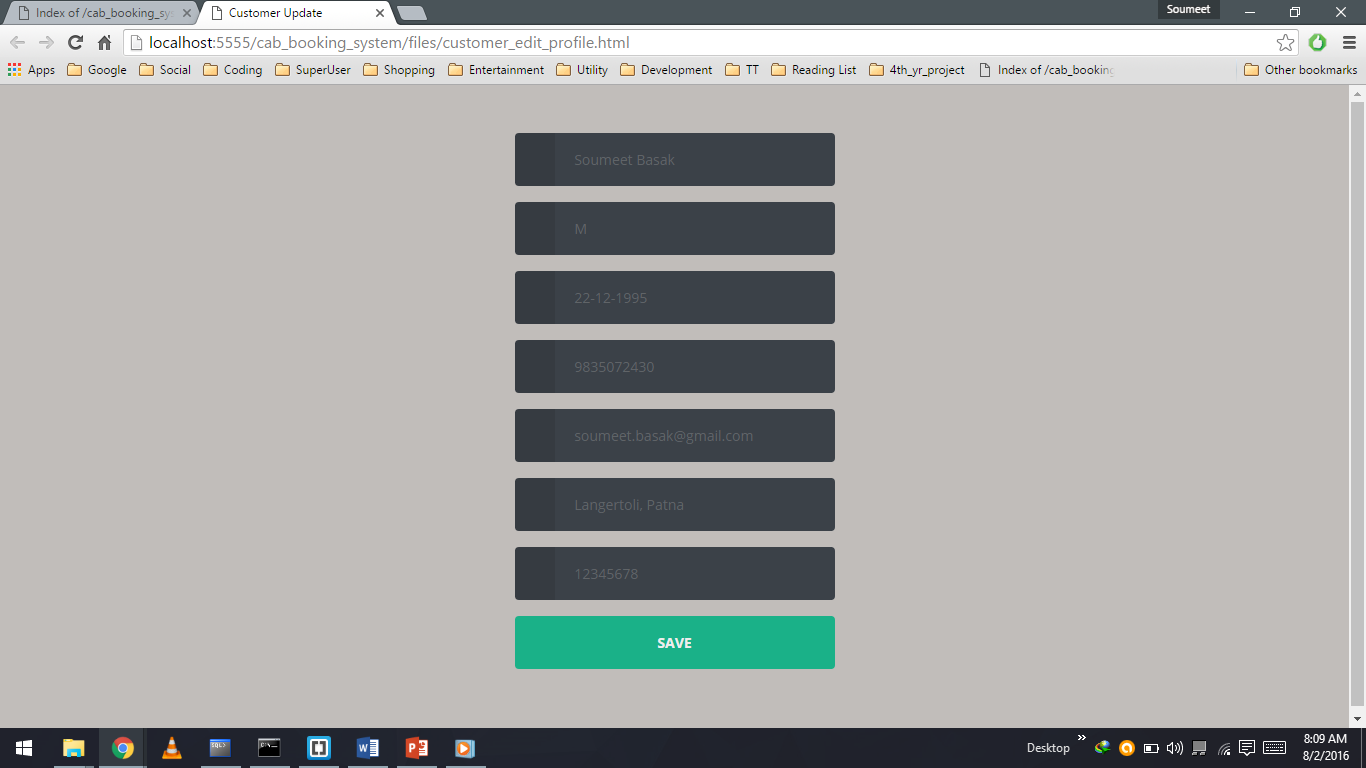
**Customer Home**

****

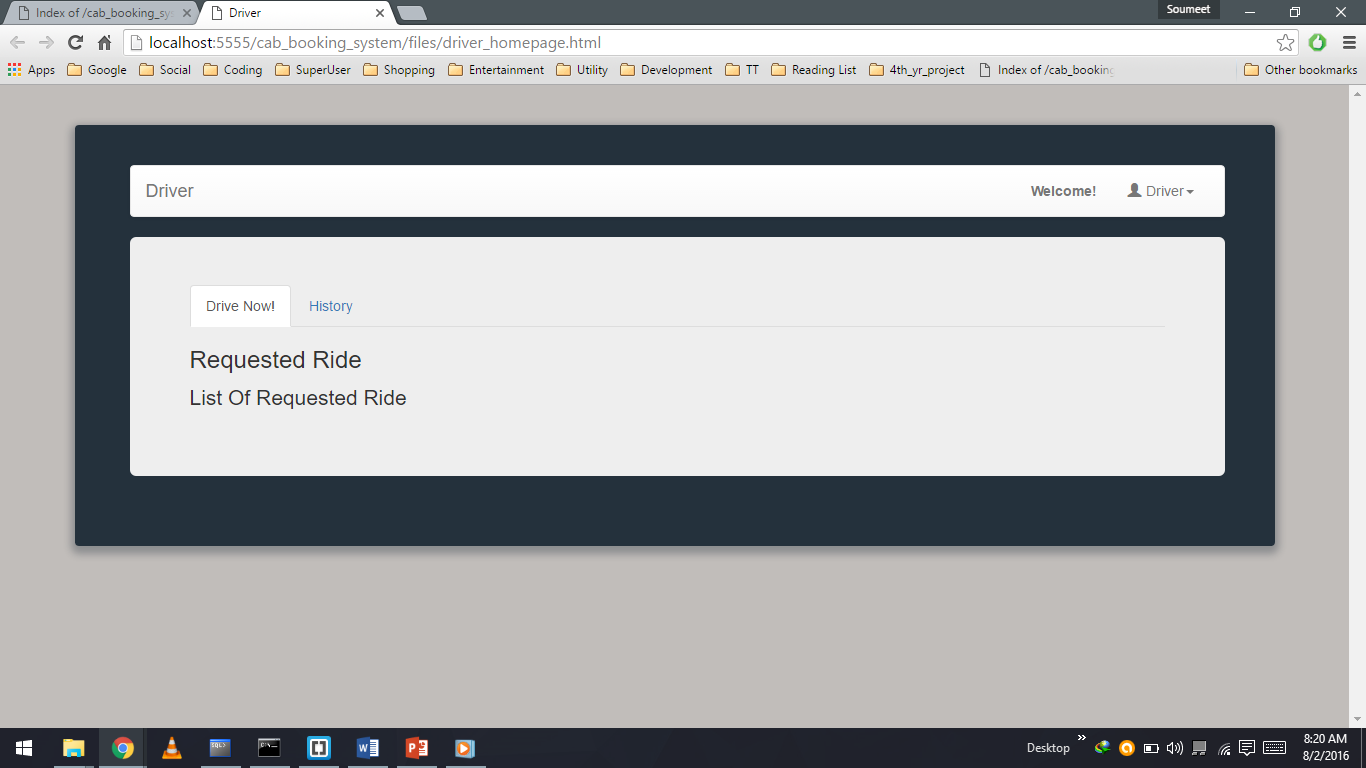
**Customer Ride History**

****

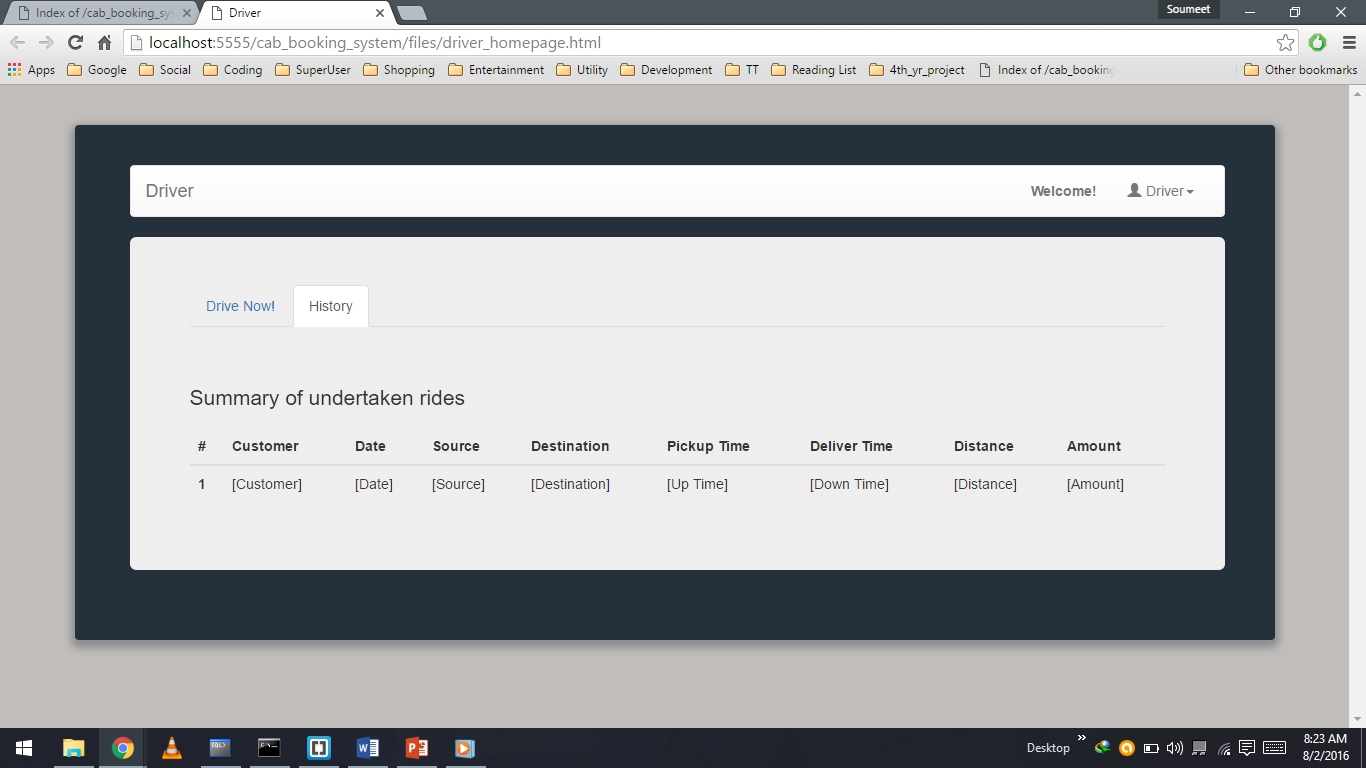
**Customer Profile Update**

****

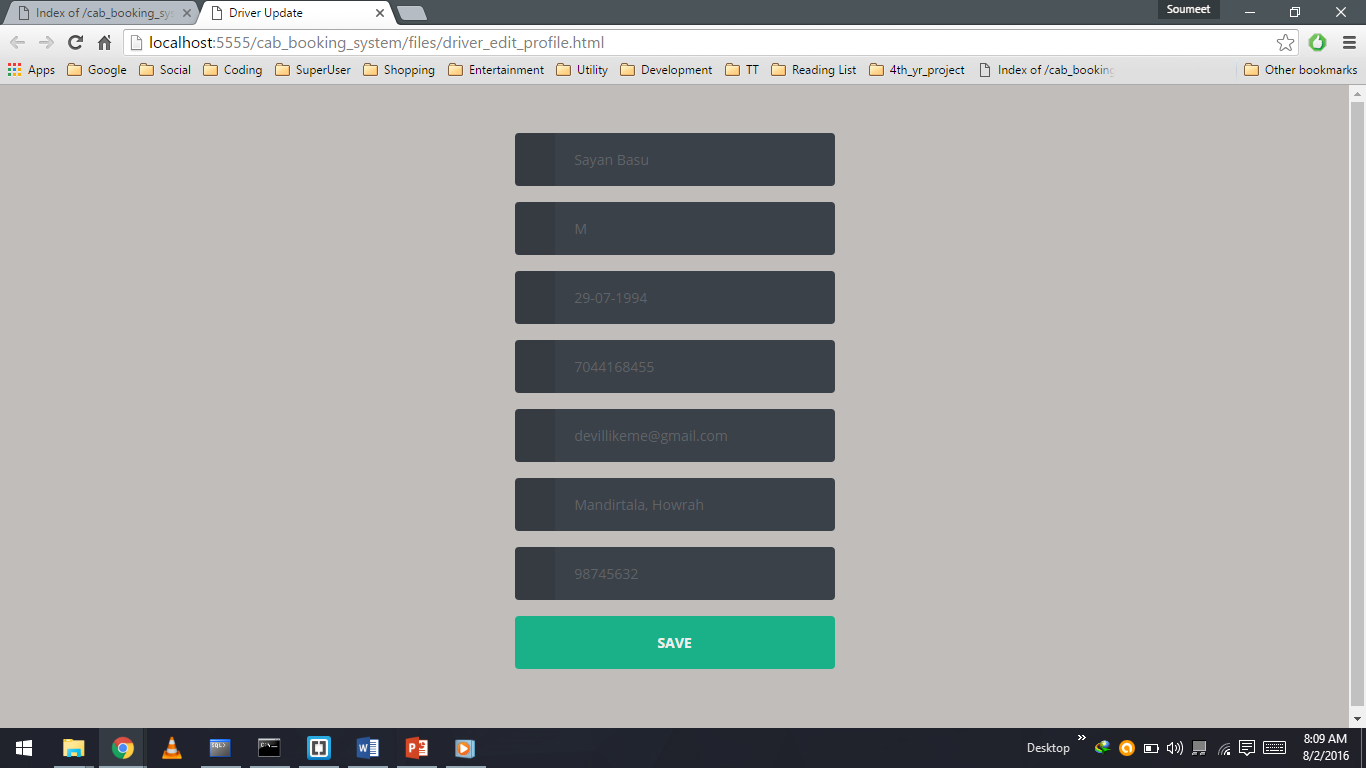
**Driver Home**

****

**Driver Ride History**

****

**Driver Profile Update**

****

**Chapter 13: Future Scope**

 **Mobile web-friendly**. Our websites are optimized for display on smartphones and tablets so that your customers can book on the go from their portable devices.

 **Secure Credit Card Processing & Storing**. With industry-standard encryption, your customers can rest assured that their payments will always remain secure with our SSL connection. We also offer the convenience of saving your customers' payment information in our secure vault to speed up repeat bookings.

 **Booking Zones**. You can select cabs from a range of cabs according to the fare/km in that area. You can also apply free or discount coupons on the bookings if available.

 **SMS** (text message) ,call and **email notifications**. The customer can be kept updated about the cab’s arrival time via the notification system. After the ride the customer receives a receipt of the ride via SMS or email.

**Chapter 14: Bibliography**

* 1. [Software Engineering: A Practitioner's Approach, 8/e](http://highered.mcgraw-hill.com/sites/0078022126)

By: R.S Pressman

Tata McGraw Hill Education Private Limited

# Fundamentals of Software Engineering, 3/E.

By: Rajib Mall

PHI Learning Private Limited

### Mastering Integrated HTML & CSS, 1st Edition

### By: Virginia DeBolt

### Wiley Publishing, Inc.

### Fundamental of Database System, 5th Edition

By: Ramez Elmasri, Shamkant B. Navathe

Pearson Education

* 1. Python in Easy Steps

By: Mike McGrath

Python

**Web Reference:**

1. http://www.python.org/
2. http:// Tutorialspoint.com/
3. http://uml-diagram.org/