**A Project Report ON**

**PROJECT in MOBILE STORE SYSTEM**

**For**

**EXL – Certified Software Test Engineer**

**BY**

**Mayuri Siddhu Bandgar**

**Enrollment** **Number** :**EBEON0722629683**

**Batch No-7669**

**Under Guidance** **by:-Mohana Priya**



CONTENT

|  |  |
| --- | --- |
| Sr.No | Index |
| 1 | Introduction |
| 2 | Scope and objective |
| 3 | System specification |
| 4 | System design |
| 5 | Implementation |
| 6 | System testing |
| 7 | Conclusion |

**1.INTRODUCTION**

**1.1 Introduction**

Hyper Text Markup Language(HTML)is the main markup Language for creating

Web pages and other information that can be displayed in a web browser.

HTML is written in the form of html elements consisting of pages enclosed in angle brackets <like<html>. Whitin the web page content .HTML pages most commonly come in pairs <h1> and </h1>,although some tags represent empty elements and so are unpaired. for example<img>The first tag in a pair is the start tags and second tag n the end tag in between these tags web designers can add text .further tags, comments and other types of text based content.

HTML elements form the building blocks of all websites.Html allows images and objects to be embedded and can be used to crate interactive forms .it provides a means to crete structured documents by denoting structural semantics for text such as headings ,paragraphs, lists, links, quotes and other items it can embed scripts written in languages such as JavaScript which affect the behaviour of html web page.

Web Browsers can also refer to cascading Style Sheets(CSS) to define the Appearance and Layout of text and other material.

**1.2 Introduction To Mobile Store**

The Mobile Store Limited, is India’s first countrywide chain of telecom retail outlets and largest mobile phone retailer, formerly known as “Essar tecom retail.ltd”. The Mobile store currently has over 1000 outlets across 150cities with its headquarters located at Mumbai, india. The mobile Store Is an essar Group venture set to introduce a pan Indian network of retail telecom outlets.

History-The Mobile Stores was launced on march 22,2006,in Mumbai, india incorporated under the companies act 1956.it has its registered office at esser OUSE,mahalaxmi,Mumbai.

**2.SCOPE AND OBJECTIVE**

**2.1 Scope and Objective**

The Mobile Store is one of the leading telecom retail stores in the country. The Mobile Store has Over 1000 outlines across 150 cities, this covering virtually every state across India. The Mobile Store outlines are in three formats .Large(1000-1500 sq. ft).Medium(800-1000 sq. ft).and small(150-200 sq. ft) They have categorized its mobile device offerings into consumer segments keeping in mind the profile and needs of different consumers.

The Mobile Store markets mobiles. Accessories mobile Connections and recharges, as well as services like bill payments and mobile repairs and exchange The Mobile store also sells music and gaming devices and Direct to Home Packages. The Mobile Store retails mobiles of domestic and international brands like apple ,HTC, BlackBerry, Samsung, Micromax, Maxx, Karbonn Mobiles Motorola Spice and others. The mobile store also provides online mobiles and DTH credit recharge through its website. For this The mobile tied up with al leading operators including Airtel, Vodafone, BPL, Idea, BSNL and reliance communications .

The Mobile Store also provides mobile phone repaires and VAS such as ring tones wallpapers and games. The Stores also provides prepared and postpaid mobile connections and after sales services

**2.2 Abstract**

Information Technology has been growing rapidly, with the course of time, as technology and human knowledge converge to a twilight, the need for the betterment of their own sustaining life also increases. Rapid need of high speed data access and faster delivery is one of the major requirement of a product. Technology has given many gifts to the human race to lead a sophisticated life. One of the fabulous gifts was Mobile Phone which was invented by Joel Engel and Richard Frenkiel in 1983.

              As the havoc increase in the platform of cellular phone, there must be well organized software which makes it possible for a management of a mobile store to maintain a healthy business. Thus getting interested on this matter we started thinking to develop a software name “MOBILE STORE SYSTEM”.

              It has a well define hardcore database of Microsoft SQL Server 2005 were the details of the specific data are safely kept in that Database and can also be retrieved as per need.

               No matter how advanced the Cellular phone, Smart phone be upcoming in the near future the use of this software will not be Back dated. Moreover the mobile Phones get advanced with the upcoming technology but this software will remain as Evergreen Forever.

**2.3 Existing System**

* In the existing system details are maintained manually. Due to this the data retrieved is time consuming. Due to human calculation errors occur. Even when the data is maintained on spreadsheet inconsistency occurs as an order might be missed or wrongly entered or twice.
* Data are stride an excel sheet which takes lot of time and data may be corrupted.
* As storage and exchange of data is achieved only by use of excel sheets which lack validation capabilities, there is always risk of invalid, inaccurate or incomplete data being fed in computer.
* Difficulty in managing multiple forms.
* Lack of security.

**3.SYSTEM SPECIFICATION**

Processing environment includes hardware and software requirements

**3.1 Software Requirement-**

When an application project is considered the three basic software requirements are the platform in which the project is developed, the front-end tool that provides the interaction with the users and the back-end tool that stores the data.

* Operating System-Windows XP, windows 10 pro
* Front-end Tool – Notepad
* Back-end Tool- Ms Oracle 11g

**3.2 Hardware Requirement-**

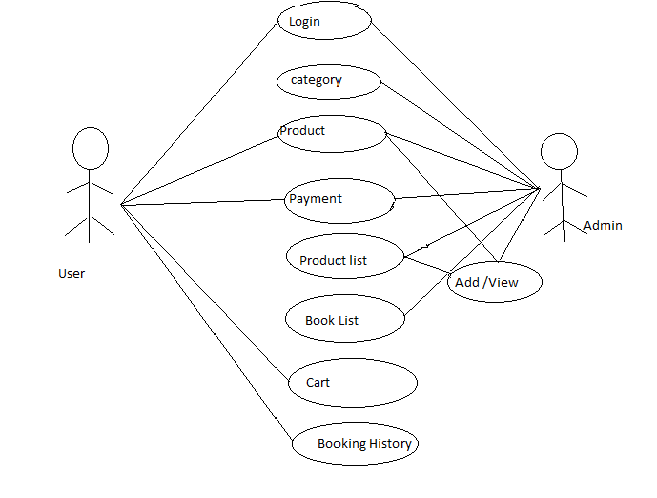
Processors will continue to get faster, smaller and cheaper, whereas memory will continue to get faster, larger and cheaper. The trend except to have a reasonable memory to a powerful processor

* Operating System- Windows xp and above
* RAM-4GB RAM
* Processor-Intel(R)Core(TM)i5-2410 CPU @2.30GHz
* System Type-64-bit operating system,x64-based processor

**4. SYSTEM DESIGN**

**4.1 Use case Diagram-**

This Use Case Diagram is a graphic depiction of the interactions among the elements of Mobile Shop Management System.



**4.1.1 Use case Description**

This use cae diagram is a graphical depiction of the interactions among the elements of mobile store system. It’s represents the methodology used in system analysis to identify, clarify, and organize system requirements of mobile store system. The main actors of mobile store system in this use case diagram are: System user, login category ,product, product list, book list, cart, Booking history different type of use cases such as mobile manage, manage customers ,who perform the different type of use cases such as manage mobile, manage customers, manage brands, manage sells, manage payment, manage stock, manage models, manage users and full mobile store system operation.

|  |  |  |  |
| --- | --- | --- | --- |
| Sr.No | USE CASE | ACTOR | DESCRIPTION |
| 1. | Login | Administrator, customer | Enter username and password |
| 2. | Registration | Customer | First name, middle name, Last name, email ,mobile no, age, gender, |
|  |  |  |  |

**4.1.2 Use case 001: login**

INTORDUCTION: This use case the administrator and customer the need to the store

ACTOR: Administrator, Customer

PRE-CONDITION: user has to have a valid credential

POST-CONDITION: the display the page

BASIC FLOW: the user enter the user name and password

Scenario:

|  |  |
| --- | --- |
| ACTOR | SOFTWARE REACTION |
| User name  Password | If it is valid username and password the system login successful  If it invalid username and password then system login unsuccessful |

ALTERNATE FLOW: the user can enter the register now

SPECIAL REQUIREMENTS: None

ASSOCIATED USE CASE(S): None

**4.1.3Use case 002: Registration**

INTORDUCTION: this use case outlines the step that need to be followed to register form

ACTOR: Customer.

PRE-CONDITION: user has to have a valid credential

POST-CONDITION: the system display the relevant page

BASIC FLOW: the system confirms that the entered details is correct or not.

Scenario-

|  |  |
| --- | --- |
| ACTOR | SOFTWARE REACTION |
| First name  Middle name  Last name  Mobile no  Email  Date of Birth  Gender | The system verifies that the above items has been filled out  If any data is missing, the system warns the user and the steps continues with software reaction  If all data has been entered the system ask the user to view the result. If the user indicate they want to end the scenario here And If the user indicate that their marks are not correct the scenario continues with software reaction |

ALTERNATE FLOW: None

SPECIAL REQUIREMENTS: None

ASSOCIATED USE CASE(S): None

**4.2 Activity Diagram-**

This is the Activity diagram of mobile store management system which shows the flows between the activity of payment ,mobile brand, models customer. The main activity involved in this Activity diagram of mobile shop management system as follow-

Payment Activity

Mobile activity

Brand Activity

Models Activity

Customer Activity

**Features of the Activity Diagram of Mobile Store System**

Admin user can search payment, view description of a selected payment, add payment, update payment and delete payment

Its shows the activity flow of editing, adding and updating of mobile

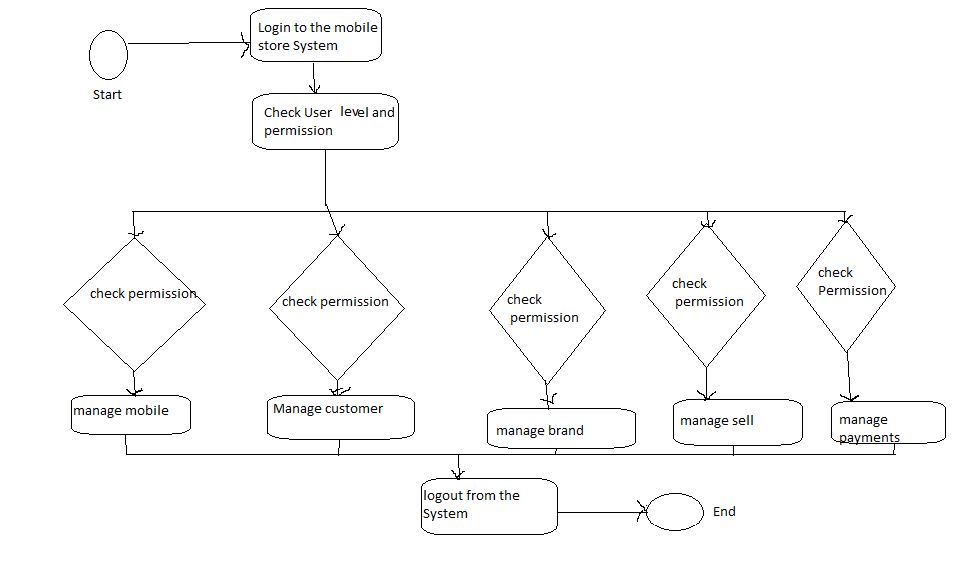
User will be able to research and generate report of brand, models, customer

All objectes such as(payment,mobile, customer)are interlinked

Its shws the full description and flow of payment,models,customer,brand,mobile

**Login Activity Diagram of Mobile Shop System**

This is the login activity diagram of mobile shop system, which shows the flows of login activity where admin will be able to login using username and password. After login user can manage all the operations on brand, payment, mobile, customer, models are secure and user can access these page after login. The diagram below helps demonstrate how the login page works in a mobile shop system. The various objects in the customer, brand, payment, mobile, and models page-interacted over the course of the activity, and user will not be able to access this page without verifying there identity



**4.3 Class Diagram-**

Mobile Shop Management System Class Diagram describes the structure of a Mobile Shop Management System classes, their attributes , operations ,and the relationship among objects. The main classes of the mobile shop management system are Mobile , customer, brands sells ,payment, stock.

The purpose of class diagram is to model the static view of an application. Class diagrams are the only diagrams which can be directly mapped with object-oriented languages and thus widely used at the time of construction

Mobile class-Manage all the operation of Mobile

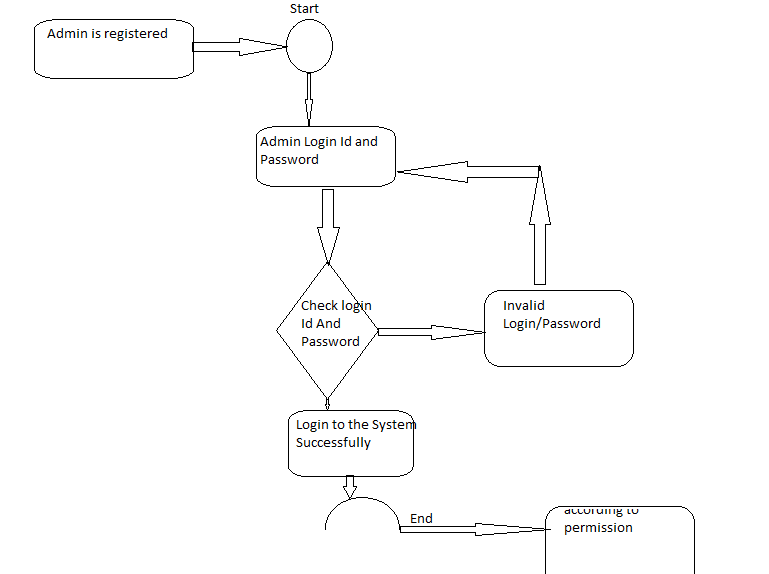
Customers class Manage all the operations of customers

Brands class –Manage all the operations of Brands

Sells Class-Manage all the operation of sells

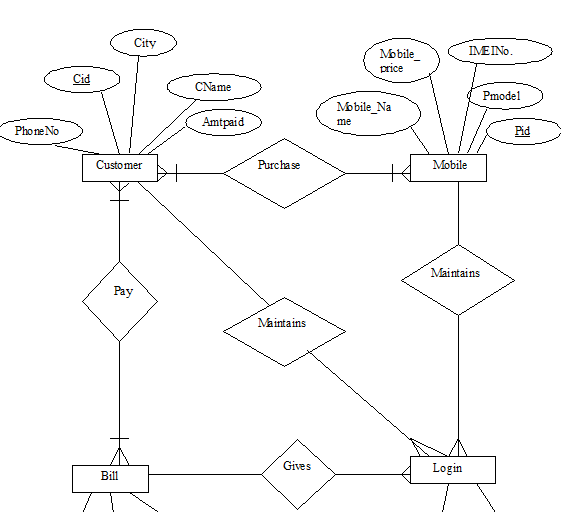
Payment Class-Manage all the operations of Payment

Stock class-Manage all the operations of Stock



**4.4 ER Diagram-**

This ER(Entity Relationship) Diagram represents the model of mobile shop management system entity. The entity –relationship diagram of Mobile Shop management System shows all the visual instrument of database tables and the relations between customers, sells, Mobile, Stock etc. It used structure data and to define the relationship between structured data groups of mobile shop management system functionalities.

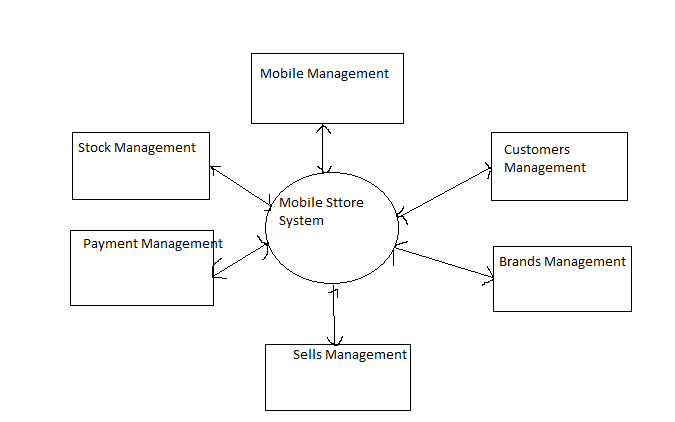


**4.5 Data Level flow Diagram-**

Mobile Store System Data flow diagram is often used as a preliminary step to create an overview of the mobile without going into great detail. Its a basic overview of the whole mobile store system or process being analyzed or modelled. its designed to be an at-a-glance view of payments stock and login showing the system as a single high-level process, with its relationship to external entities of mobile, customer and brands. it should be easily understood by a wide audience, including mobile, Brands and Payments in zero level DFD of mobile Store System.

#### DFD Symbol

1. Process
2. Data Flow
3. External Entity
4. Data Store



#### Table Structure

##### LoginTable name: login

##### Primary key: email id

|  |  |  |
| --- | --- | --- |
| **COLUMN NAME** | **DATA TYPE** | **DESCRIPTION** |
| EMAIL ID | TEXT | EMAIL ID |
| PASSWORD | NUMBER | PASSWORD |

##### Register:

##### Register Name-Register

##### Primary key: email id

|  |  |  |
| --- | --- | --- |
| **COLUMN NAME** | **DATA TYPE** | **DESCRIPTION** |
| FIRST NAME | TEXT | FIRST NAME |
| MIDDLE NAME | TEXT | MIDDLE NAME |
| LAST NAME | TEXT | LAST NAME |
| MOBILE NUMBER | NUMBER | MOBILE NUMBER |

|  |  |  |
| --- | --- | --- |
| BIRTHDAY | NUMBER | BIRTH DATE |
| AGE | NUMBER | NUMBER |
| EMAIL ID | TEXT | EMAIL ID |
| GENDER | TEXT | TEXT |
| ADDRESS | TEXT | ADDRESS |

**5.CODING AND SCREENSHOT**

**MODULES-**

Mobile Phone is an electronic device used for communication and messaging. Now days this has become very popular and essential need of the society. These are available in various make and brands in the market. Therefore repairing and servicing of mobile phones are also required to be done. This has got a prospective market.

**Purpose-** Mobile Shop means a registered means of transportation installed as a point of sale of food and/or non-food products where residents of townships and villages are served in accordance with the procedure laid down by the municipal council.

HTML-

* HTML is Standard markup language.
* HTML is the standard markup language for creating Web pages.
* HTML describes the structure of a Web page.
* The <!DOCTYPE html> declaration defines that this document is an HTML5 document
* The <html> element is the root element of an HTML page
* The <head> element contains meta information about the HTML page
* The <title> element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)
* The <body> element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
* The <h1> element defines a large heading
* The <p> element defines a paragraph

## The <!DOCTYPE> Declaration

## The <!DOCTYPE> declaration represents the document type, and helps browsers to display web pages correctly.

## It must only appear once, at the top of the page (before any HTML tags).

## HTML Headings

## HTML headings are defined with the <h1> to <h6> tags.

## HTML Attributes

* All HTML elements can have attributes
* Attributes provide additional information about elements
* Attributes are always specified in the start tag
* Attributes usually come in name/value pairs like: name="value"

# HTML Forms

An HTML form is used to collect user input. The user input is most often sent to a server for processing. The <form> element is a container for different types of input elements, such as: text fields, checkboxes, radio buttons, submit buttons, etc.

## Text Fields

The <input type="text"> defines a single-line input field for text input.

## Radio Buttons

The <input type="radio"> defines a radio button.

Radio buttons let a user select ONE of a limited number of choices.

## Checkboxes

The <input type="checkbox"> defines a **checkbox**.

Checkboxes let a user select ZERO or MORE options of a limited number of choices.

## Submit Button

The <input type="submit"> defines a button for submitting the form data to a form-handler.

The form-handler is typically a file on the server with a script for processing input data.

## The <input> Element

One of the most used form element is the <input> element.

The <input> element can be displayed in several ways, depending on the type attribute.

## CSS

Cascading Style Sheets (CSS) is used to format the layout of a webpage.

With CSS, you can control the color, font, the size of text, the spacing between elements, how elements are positioned and laid out, what background images or background colors are to be used, different displays for different devices and screen sizes, and much more!

CSS can be added to HTML documents in 3 ways:

* **Inline** - by using the style attribute inside HTML elements.An inline CSS is used to apply a unique style to a single HTML element.An inline CSS uses the style attribute of an HTML element.
* **Internal** - by using a <style> element in the <head> section.An internal CSS is used to define a style for a single HTML page.An internal CSS is defined in the <head> section of an HTML page, within a <style> element.
* **External** - by using a <link> element to link to an external CSS file.An external style sheet is used to define the style for many HTML pages

## CSS Colors, Fonts and Sizes

Here, we will demonstrate some commonly used CSS properties. You will learn more about them later.

The CSS color property defines the text color to be used.

The CSS font-family property defines the font to be used.

The CSS font-size property defines the text size to be used.

# JAVASCRIPT

JavaScript makes HTML pages more dynamic and interactive.

## The HTML <script> Tag

The HTML <script> tag is used to define a client-side script (JavaScript).

The <script> element either contains script statements, or it points to an external script file through the src attribute.

Common uses for JavaScript are image manipulation, form validation, and dynamic changes of content.

**5.1 HTML CODE-**

<html>

<body bgcolor=" sky blue">

<CENTER>

<FONT COLOR="white" face="times newroman" size="60">

<b>

<u>

MOBILE STORE

</u>

</b>

</font>

</center>

<marquee direction="right" height="40%"scrollamount="10">

<img src="C:\Users\Mayuri\Desktop\mobile img\blackberry.jpg" height="50" width="50" alt="blackberry">

</a>&nbsp;

&nbsp;

<img src="C:\Users\Mayuri\Desktop\mobile img\mobile1.jpg" height="350" width="350" alt="mobile1">

</a>&nbsp;

&nbsp;

<img src="C:\Users\Mayuri\Desktop\mobile img\download (2).jpg" height="350" width="350" alt="download (2)"></a>&nbsp; &nbsp;

<img src="C:\Users\Mayuri\Desktop\mobile img\download3.jpg" height="350" width="350" alt="download3"></a>&nbsp; &nbsp;

<img src="C:\Users\Mayuri\Desktop\mobile img\images.jpg" height="350" width="350" alt="images"></a>&nbsp; &nbsp;

</marquee>

<div class="navbar">

<a href="https://www.amazon.com/">Home</a>

<a href="C:\Users\Mayuri\Desktop\loginpage">About us</a>

<a href="https://www.flipkart.com/">Delivery</a>

<a href="C:\Users\Mayuri\Desktop\loginpage\feedback.html">FeedBack</a>

<a href="https:///https://www.justdial.com/ ">Contact us</a>

<a href="C:\Users\Mayuri\Desktop\loginpage\terms and condition.html">\*Terms & Conditions

</a>

<br>

<BR>

<a href="C:\Users\Mayuri\Desktop\loginpage\index.html" class="left">Login</a>

</div>

<div class="main">

<h2>Mobile</h2>

<p>Welcome to Guys</p>

<br>

<div class="row">

<div class="sidebar">

<style>

table{

font-family:arial,sans-serif;

border-collapse:collapse;

width:100%;

}

td,th

{

border:1px solid;

text-align:left;

padding:8px;

}

tr:nth-child(even)

{

background-color:dddddd;

}

</style>

<h2>

Mobile Discount

</h2>

<table>

<tr>

<th>

Mobile-name

</th>

<th>

Price

</th>

<th>

Discount

</th>

</tr>

<tr>

<td>

SamSung 5G

</td>

<td>

25000

</td>

<td>

50%

</td>

</tr>

<tr>

<td>

Vivo

</td>

<td>

50000

</td>

<td>

50%

</td>

</tr>

<tr>

<td>

Iphone

</td>

<td>

80000

</td>

<td>

50%

</td>

</tr>

<tr>

<td>

Apple

</td>

<td>

100000

</td>

<td>

50%

<td>

</tr>

</table>

<h3>

Register form

</h3>

<form action="/action\_page.php">

<fieldset

<label for="fname">First name:</label>

<input type="text" id="fname" name="fname">

<br>

<br>

<label for="mname">Middle name:</label>

<input type="text" id="mname" name="mname">

<br>

<br>

<label for="lname">Last name:</label>

<input type="text" id="lname" name="lname">

<br>

<br>

<label for="moname">Mobile No:</label>

<input type="text" id="moname" name="moname">

<br>

<br>

<label for="birthday">Birthday:</label>

<input type="date" id="birthday" name="birthday">

<br>

<br>

<label for="agename">Age:</label>

<input type="text" id="agename" name="agename">

<br>

<br>

<label for="ename">Email Id:</label>

<input type="text" id="ename" name="ename">

<br>

<br>

<label for="gname">Gender:</label>

<input type="text" id="gname" name="gname">

<br>

<br>

<label for="aname">Address:</label>

<input type="text" id="aname" name="aname">

<br>

<br>

<input type="submit" value="Submit">

</fieldset>

<br>

<form action="/action\_page.php">

<label for="Mobile">Choose a Mobie:</label>

<select id="Mobile" name="Mobiles">

<option value="apple">Apple</option>

<option value="vivo">Vivo</option>

<option value="oopo">Oopo</option>

<option value="sumsang">Sumsang</option>

</select>

<input type="submit">

</form>

<br>

<font size="5" color="green">

<u>

<b>

<i>My Stores</i>

</b>

</u>

</font>

<br>

<br>

<a href="C:Documents and SettingspalakDesktoplab assignmenteast delhi stores.html">East Delhi Stores</a>

<br>

<a href="C:Documents and SettingspalakDesktoplab assignmentwest delhi stores.html">West Delhi Stores</a>

<br>

<a href="C:Documents and SettingspalakDesktoplab assignmentnorth delhi stores.html">North Delhi Stores</a>

<br>

<a href="C:Documents and SettingspalakDesktoplab assignmentsouth delhi stores.html">South Delhi Stores</a>

<br>

<a href="C:Documents and SettingspalakDesktoplab assignmentcenter delhi stores.html"> Center Delhi Stores</a>

<br>

<BR>

<a href="C:Documents and SettingspalakDesktoplab assignmentMobiles.html">Mobiles</a>

<br>

<font face="time new roman" size="4" color="Green">

<b>

<u>MISSION</U>

</B>

</font>

<BR>

<BR>

<FONT face="time new roman" size="3">

Our most aim is to be the world's best mobile store, where a person can get all types and brands phone. <br>

<BR>

<BR>

<BR>

<FONT face="times new roman" size="4" color="Green">

<b>

<u>VISION</u>

</b>

</font>

<br>

<BR>

<FONT face="time new roman" size="3">

Mobile Store only vision is to provide customers with best feature phone and best service. <br>

<BR>

<BR>

</font>

<FONT color="white" face="time new roman" size="5">

<b>

<u>HISTORY & INFRASTRUCTURE</U>

</B>

</font>

<br>

<br>

<font size="3" color="white" >

The Mobile was launched on.March22,2006,in Mumbai,India incorporated under The Companies Act,1956.

It has its registered office at Essar House,Mahalaxmi,Mumbai.<br>

The Mobilestore is one of the leading telecom retail stores in the country.

The Mobile store has over 1000 outlets across 150 cities,

thus covering virtually every major town in every state across India.

THE Mobile Store outlets are in three formats: Large(1000-1500sq.ft)and small (150-200sq.ft)

They have categorized its mobile device offerings into consumer segments keeping in mind the profiles and need of different consumers.

</font>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<title>HTML</title>

<link rel="stylesheet" href="https://en.wikipedia.org/wiki/Mobile\_phone">

<style>

.About-us {

width: 100%;

text-align: center;

background-color: #ccc;

margin-top: 30px;

margin-bottom: 30px;

}

h1{

color: red;

font-size: 35px;

}

img {

border-radius: 50%;

}

p {

font-size: 20px;

}

h3 {

text-shadow:

}

a {

padding: 20px;

text-align: center;

text-decoration: none;

margin: 5px 2px;

background: #3B5998;

color: white;

border-radius: 8px;

font-size: 50px;

}

.column {

float: left;

width: 30%;

margin-bottom: 16px;

padding: 0 8px;

}

.card {

box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.3);

margin: 10px;

background-color: white;

}

.card img {

height: 100px;

}

</style>

</head>

<body>

<div class="About-us">

<h1> About Us </h1>

<href="https://en.wikipedia.org/wiki/Mobile\_phone">

<div class="row">

<div class="column">

</div>

</div>

<div class="column">

<div class="card">

<href="https://en.wikipedia.org/wiki/Mobile\_phone" style="width:100%">

<div class="container">

<p class="title">About us</p>

<p align="center">

When you have a great story about how your product or service was built to change lives, share it. <br>

The "About Us" page is a great place for it to live, too. Good stories humanize your brand, providing<br>

context and meaning for your product. Whatâ€™s more, good stories are sticky -- which means people are<br>

more likely to connect with them and pass them on.People tend to think that "About Us" pages have to<br>

sound formal to gain credibility and trust. But most people find it easier to trust real human beings,<br>

rather than a description that sounds like it came from an automaton. Trying to sound too professional<br>

on your "About Us" page results in stiff, â€œsafeâ€? copy and design -- the perfect way to make sure your <br>

company blends in with the masses.We know -- no industry jargon. If you think it makes you sound super <br>

smart on your "About Us" page, think again. People want and appreciate straight talk about what your <br>

business does. After all, if people can't figure out what you do, how will they know they need your <br>

product or service?Instead of following the classic "About Us" script and writing a few paragraphs <br>

about the company's mission and origins, try something different -- there are plenty of ways to make <br>

it more visually compelling.Yes, this post is about, well, "About Us" pages. But sometimes, you don't<br>

always need to wait for users to get there in order to make a statement. That's part of breaking the <br>

mold to showcase your company's personality.Who would you trust more: A company talking about how <br>

awesome it is, or a colleague raving about the company's work? I'd bet that you prefer the latter <br>

that colleagueis more likely to be unbiased and give you a realistic understanding of what the company<br>

is like. Including customer testimonials on your "About Us" page can give prospects and leads a more <br>

down-to-earth view of your company.

</p>

</div>

</div>

</div>

</div>

<HTML>

<HEAD>

<TITLE>Feedback Form</TITLE>

</HEAD>

<BODY>

<H1>Customer Feedback Form</H1>

<FORM Action="Mailto:simon.Long@Cit.Ie" Method="Post" Enctype="Text/Plain">

First Name : <INPUT Type="Text" Name="Firstname" Placeholder="Enter Name Here">

Second Name : <INPUT Type="Text" Name="Secondname" Placeholder="Enter Surname Here">

<BR>

<BR>

<!--It's Important That Both Of These Radio Button Have The Same Name So They Behave As One Component (I.E. Only One Can Be Selected At A Time)-->

<INPUT Type="Radio" Name="Gender" Value="Male">Male<BR>

<INPUT Type="Radio" Name="Gender" Value="Female">Female

<BR>

<BR>

<!--This Uses The New HTML "Email" INPUT Type Which Will Automatically Validates The Email Address When The SUBMIT Button Is Clicked-->

Email: <INPUT Type="Email" Name="Mail" Placeholder="E-Mail Address">

<BR>

<BR>

Comment :<BR>

<TEXTAREA Rows="6" Cols="50" Name="Commentfield"></TEXTAREA>

<BR>

<SELECT Name="Service Feedback">

<OPTION Value="Very Good">VeryGood</OPTION>

<OPTION Value="Good">Good</OPTION>

<OPTION Value="poor">Poor</OPTION>

<OPTION Value="VeryPoor">Very Poor</OPTION>

</SELECT>

<BR>

<BR>

<INPUT TYPE="Submit" Value="Send Feedback">

<INPUT TYPE="Reset" Value="Reset">

</FORM>

</BODY>

</HTML>

<!DOCTYPE html

<!-- Code by CodeWithNepal - codewithnepal -->

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Login Form validation using HTML CSS & JS | CodeWithNepal</title>

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

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.3/css/all.min.css"/>

</head>

<body>

<div class="wrapper">

<header>Login Form</header>

<form action="#">

<div class="field email">

<div class="input-area">

<input type="text" placeholder="Email Address">

<i class="icon fas fa-envelope"></i>

<i class="error error-icon fas fa-exclamation-circle"></i>

</div>

<div class="error error-txt">Email can't be blank</div>

</div>

<div class="field password">

<div class="input-area">

<input type="password" placeholder="Password">

<i class="icon fas fa-lock"></i>

<i class="error error-icon fas fa-exclamation-circle"></i>

</div>

<div class="error error-txt">Password can't be blank</div>

</div>

<div class="pass-txt"><a href="#">Forgot password?</a></div>

<input type="submit" value="Login">

</form>

<div class="sign-txt">Not yet member? <a href="#">Signup now</a></div>

</div>

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

</body>

</html>>

<!--<div class="tacbox">

<input type="checkbox" />

I agree to these <a href="#">Terms and Conditions</a>.

</div>

-->

<div class="tacbox">

<input id="checkbox" type="checkbox" />

<label for="checkbox"> I agree to these <a href="#">Terms and Conditions</a>.</label>

</div>

<!--

<label class="tacbox">

<input type="checkbox" />

I agree to these <a href="#">Terms and Conditions</a>.

</label>

<label for="taccheck" class="tacbox">

<input id="taccheck" type="checkbox" />

I agree to these <a href="#">Terms and Conditions</a>.

</label>

**5.2 CSS CODE-**

@import url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600&display=swap')

\*{

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: "Poppins", sans-serif;

}

body{

width: 100%;

height: 100vh;

display: flex;

align-items: center;

justify-content: center;

background: #3853bf;

}

::selection{

color: #fff;

background: #3853bf;

}

.wrapper{

width: 380px;

padding: 40px 30px 50px 30px;

background: #fff;

border-radius: 5px;

text-align: center;

box-shadow: 10px 10px 15px rgba(0,0,0,0.1);

}

.wrapper header{

font-size: 35px;

font-weight: 600;

}

.wrapper form{

margin: 40px 0;

}

form .field{

width: 100%;

margin-bottom: 20px;

}

form .field.shake{

animation: shake 0.3s ease-in-out;

}

@keyframes shake {

0%, 100%{

margin-left: 0px;

}

20%, 80%{

margin-left: -12px;

}

40%, 60%{

margin-left: 12px;

}

}

form .field .input-area{

height: 50px;

width: 100%;

position: relative;

}

form input{

width: 100%;

height: 100%;

outline: none;

padding: 0 45px;

font-size: 18px;

background: none;

caret-color: #5372F0;

border-radius: 5px;

border: 1px solid #bfbfbf;

border-bottom-width: 2px;

transition: all 0.2s ease;

}

form .field input:focus,

form .field.valid input{

border-color: #5372F0;

}

form .field.shake input,

form .field.error input{

border-color: #dc3545;

}

.field .input-area i{

position: absolute;

top: 50%;

font-size: 18px;

pointer-events: none;

transform: translateY(-50%);

}

.input-area .icon{

left: 15px;

color: #bfbfbf;

transition: color 0.2s ease;

}

.input-area .error-icon{

right: 15px;

color: #dc3545;

}

form input:focus ~ .icon,

form .field.valid .icon{

color: #5372F0;

}

form .field.shake input:focus ~ .icon,

form .field.error input:focus ~ .icon{

color: #bfbfbf;

}

form input::placeholder{

color: #bfbfbf;

font-size: 17px;

}

form .field .error-txt{

color: #dc3545;

text-align: left;

margin-top: 5px;

}

form .field .error{

display: none;

}

form .field.shake .error,

form .field.error .error{

display: block;

}

form .pass-txt{

text-align: left;

margin-top: -10px;

}

.wrapper a{

color: #5372F0;

text-decoration: none;

}

.wrapper a:hover{

text-decoration: underline;

}

form input[type="submit"]{

height: 50px;

margin-top: 30px;

color: #fff;

padding: 0;

border: none;

background: #5372F0;

cursor: pointer;

border-bottom: 2px solid rgba(0,0,0,0.1);

transition: all 0.3s ease;

}

form input[type="submit"]:hover{

background: #2c52ed;

};

**5.3 JAVA SCRIPT**

const form = document.querySelector("form")

eField = form.querySelector(".email"),

eInput = eField.querySelector("input"),

pField = form.querySelector(".password"),

pInput = pField.querySelector("input");

form.onsubmit = (e)=>{

e.preventDefault(); //preventing from form submitting

//if email and password is blank then add shake class in it else call specified function

(eInput.value == "") ? eField.classList.add("shake", "error") : checkEmail();

(pInput.value == "") ? pField.classList.add("shake", "error") : checkPass();

setTimeout(()=>{ //remove shake class after 500ms

eField.classList.remove("shake");

pField.classList.remove("shake");

}, 500);

eInput.onkeyup = ()=>{checkEmail();} //calling checkEmail function on email input keyup

pInput.onkeyup = ()=>{checkPass();} //calling checkPassword function on pass input keyup function checkEmail(){ //checkEmail function

let pattern = /^[^ ]+@[^ ]+\.[a-z]{2,3}$/; //pattern for validate email

if(!eInput.value.match(pattern)){ //if pattern not matched then add error and remove valid class

eField.classList.add("error");

eField.classList.remove("valid");

let errorTxt = eField.querySelector(".error-txt");

//if email value is not empty then show please enter valid email else show Email can't be blank

(eInput.value != "") ? errorTxt.innerText = "Enter a valid email address" : errorTxt.innerText = "Email can't be blank";

}else{ //if pattern matched then remove error and add valid class

eField.classList.remove("error");

eField.classList.add("valid");

}

}function checkPass(){ //checkPass function

if(pInput.value == ""){ //if pass is empty then add error and remove valid class

pField.classList.add("error");

pField.classList.remove("valid");

}else{ //if pass is empty then remove error and add valid class

pField.classList.remove("error");

pField.classList.add("valid");} } //if eField and pField doesn't contains error class that mean user filled details properly

if(!eField.classList.contains("error") && !pField.classList.contains("error")){

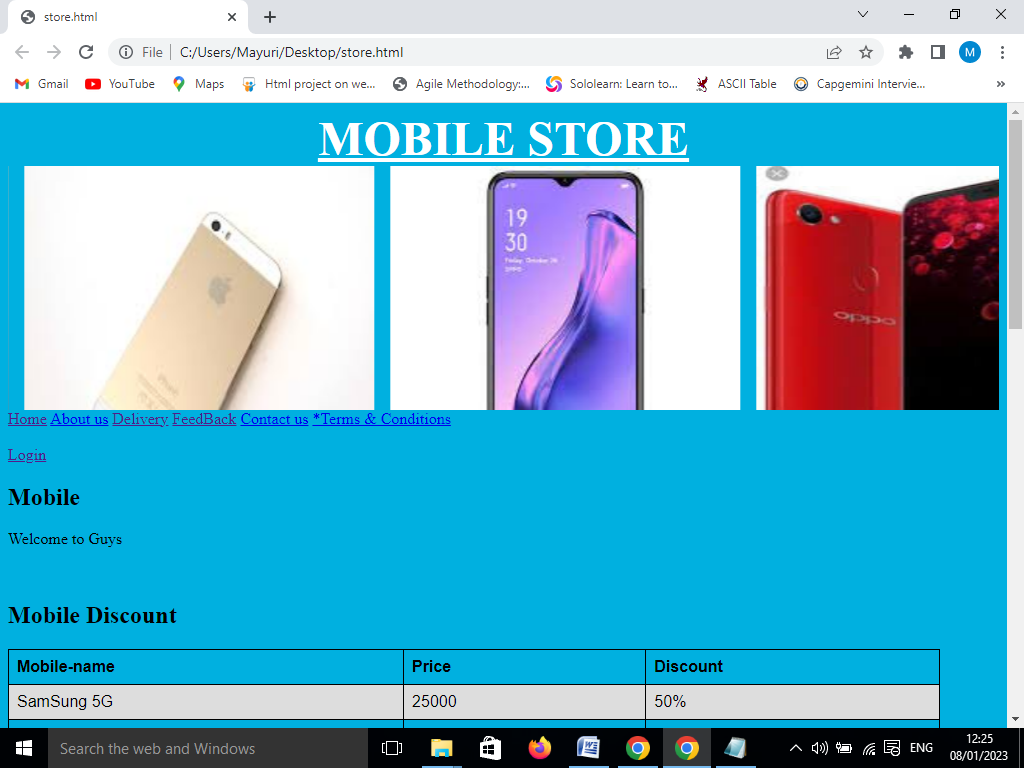
window.location.href = form.getAttribute("action"); //redirecting user to the specified url which is inside action attribute of form tag

}

};

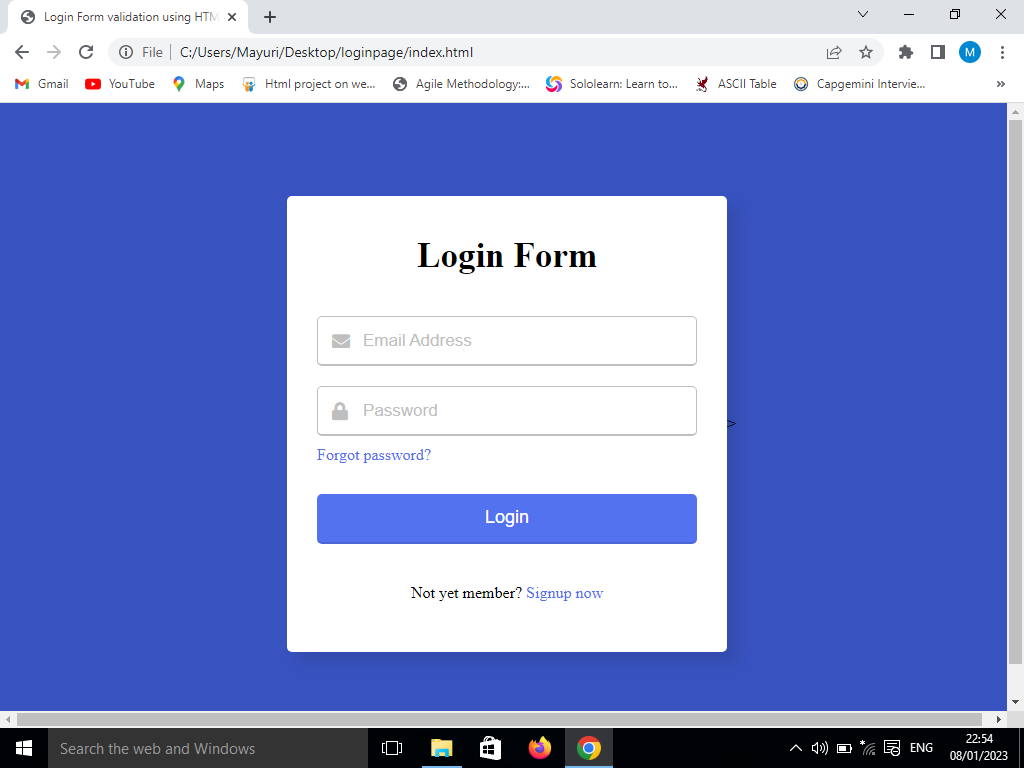
**MODULE 1- HOME PAGE-**

Download this Mobile Store template for free. This mobile shop template is responsive, cross mobile supportive and is built on HTML & CSS



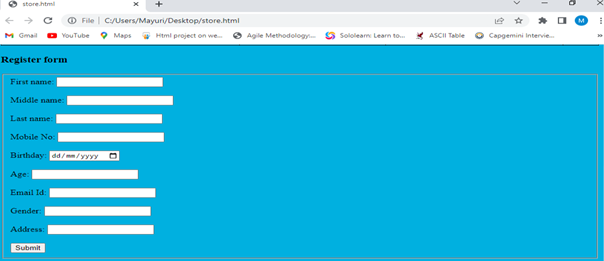
**MODULE 2- LOGIN PAGE-**

login page is of extreme importance to web and app design, especially for online stores or e-commerce websites. A creative and attractive login page will quickly catch the user’s attention, direct a high volume of visitors to your website, and increase the customer base.



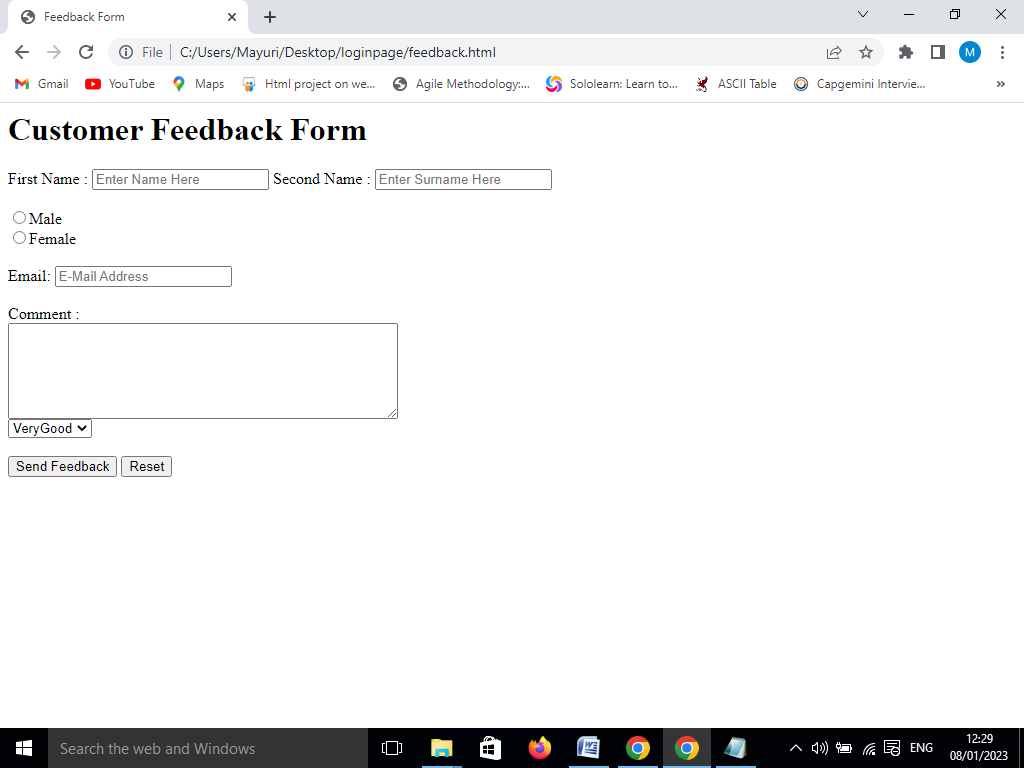
**MODULE 3- REGISTRATION FORM-**

HTML Registration Form. Following are some different types of Forms: Code 1: The following code describes how to create a simple registration Page.



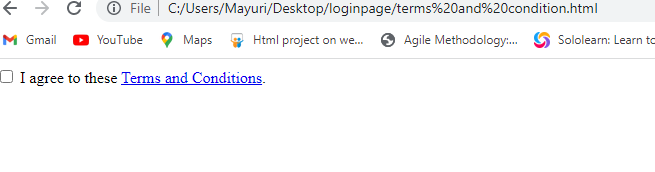
**MODULE 4- FEEDBACK FORM-**

The Customer how to service for the store then use of the Feedback Form used. They create the webpage of feedback form



**MODULE 5- TERMS AND CONDITION**

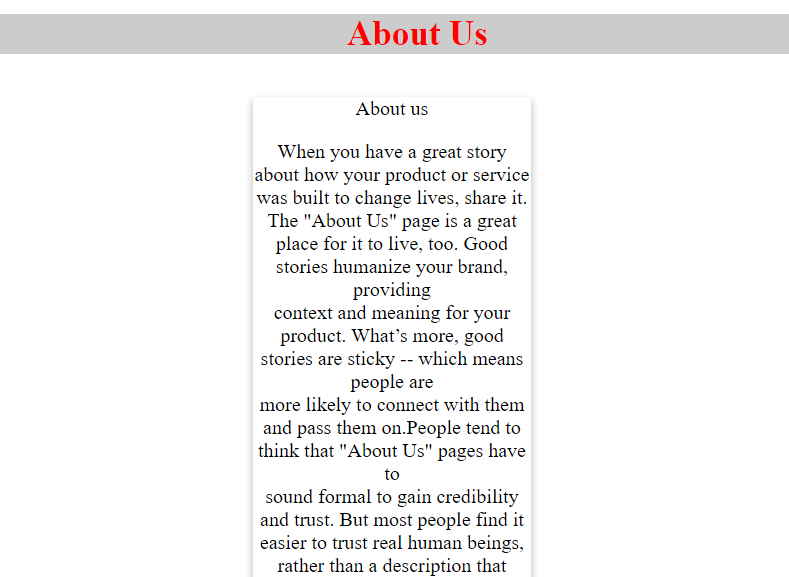
The customers buy getting the mobile or devices the privacy to use terms and condition are required.



**MODULE 6- ABOUT US**

About Us pages are where you showcase your history, what is unique about your work, your company's values, and who you serve.

The about us page is often a reflection of the purpose and personality of the business and its owners or top employees. Finally, the page can also incorporate contact or locational information. One way to view the about us concept is as a text self-portrait or short autobiography created by a business.



**6.SYSTEM TESTING**

**6.1 SOFTWARE TESTING –**

This testing checking and completeness of the software.

There are Two types of Software

1.Manual Testing- Manual testing is the Software testing process in which test case are executed manually without using automation tool is called Manual testing.

2.Automation Testing-Automation testing is process of changing any manual test cases into the test script by using automation tool.

**6.2 LEVELS OF SOFTWARE TESTING-**

Level Of Testing- There are four level of testing

1.Unit Testing

2.intergration Testing

3.System Testing

4.Acceptance Testing

Unit Testing-

Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation. This testing methodology is done during the development process by the software developers and sometimes QA staff.

Integration testing-

Integration testing -- also known as integration and testing (I&T) -- is a type of software testing in which the different units, modules or components of a software application are tested as a combined entity. However, these modules may be coded by different programmers.

System Testing-

**System Testing** is a level of testing that validates the complete and fully integrated software product. The purpose of a system test is to evaluate the end-to-end system specifications. Usually, the software is only one element of a larger computer-based system. Ultimately, the software is interfaced with other software/hardware systems. System Testing is defined as a series of different tests whose sole purpose is to exercise the full computer-based system.

Acceptance testing-

Acceptance Testing is the last phase of software testing performed after System Testing and before making the system available for actual use.

**6.3 The seven principles of testing**

* Testing shows the *presence* of defects, not their absence. ...
* Exhaustive testing is impossible. ...
* Early testing saves time and money. ...
* Defects cluster together. ...
* Beware of the pesticide paradox. ...
* Testing is context dependent. ...
* Absence-of-errors is a fallacy.

**6.4 WBT- WHITE BOX TESTING**

White Box testing is done by coder It is also called as –code level testing, unit testing, clear box testing.

In the white box testing whenever coder completeness his code writing, he check or compile code then if any bug found coder have to solve it Coder can not send code tested without doing white box testing

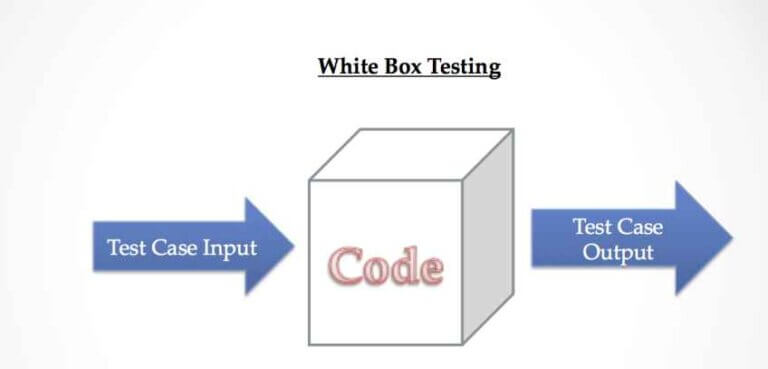
Coder check or test only positive scenario. White box testing has purpose to test correctness and completeness of the program

What do you verify in White Box Testing?

White box testing involves the testing of the software code for the following:

* Internal security holes
* Broken or poorly structured paths in the coding processes
* The flow of specific inputs through the code
* Expected output
* The functionality of conditional loops
* Testing of each statement, object, and function on an individual basis

White Box Testing is a testing technique in which software’s internal structure, design, and coding are tested to verify input-output flow and improve design, usability, and security. In white box testing, code is visible to testers, so it is also called Clear box testing, Open box testing, Transparent box testing, Code-based testing, and Glass box testing.



**Following are important White Box Testing Techniques:**

* Statement Coverage
* Decision Coverage
* Branch Coverage
* Condition Coverage
* Multiple Condition Coverage
* Finite State Machine Coverage
* Path Coverage
* Control flow testing
* Data flow testing

**Advantages:**

* White box testing is very thorough as the entire code and structures are tested.
* It results in the optimization of code removing error and helps in removing extra lines of code.
* It can start at an earlier stage as it doesn’t require any interface as in case of black box testing.
* Easy to automate.

**Disadvantages:**

* Main disadvantage is that it is very expensive.
* Redesign of code and rewriting code needs test cases to be written again.
* Testers are required to have in-depth knowledge of the code and programming language as opposed to black box testing.
* Missing functionalities cannot be detected as the code that exists is tested.
* Very complex and at times not realistic.

6.5 **BBT** -**BLACK BOX TESTING**

**Black Box Testing** is a software testing method in which the functionalities of software applications are tested without having knowledge of internal code structure, implementation details and internal paths. Black Box Testing mainly focuses on input and output of software applications and it is entirely based on software requirements and specifications. It is also known as Behavioral Testing.

requirements and specifications. It is also known as Behavioral Testing.



black box testing is also know as system &function testing this testing is done by tester. Overall functionality get checked in this type.Tester check internal functionality depend upon external functionality.

External behaviour testing is calles bbt

### How is Black Box Testing done?

The steps for carrying out Black Box Testing are as follows:

* At first, the application to be tested is studied to find out the requirements and specifications. The SRS (Software Requirement Specification) document should be maintained with accuracy.
* The inputs and test scenarios are evaluated. Efficient and time-saving techniques are incorporated.
* Test cases are generated. These test cases are made in such a way that the input range is maximum.
* The test cases are then processed to obtain the output. The generated output is compared with the expected output to understand the success of the result.
* If there are unsuccessful steps, they are sent to the [software development teams](https://openxcell.com/software-development-company/) for fixing.
* The defects are fixed.
* Run the tests again for confirmation.

### 6.5.1 Types of Black Box Testing

There are three types of black-box testing namely- functional testing, non-functional testing, and regression testing.

#### 1. Functional Testing

If a particular function or feature of the software is tested then it is categorized under black-box testing. For example, if the correct pin is entered then money can be transferred and if the incorrect pin is entered, the transaction fails.

**Examples of Functional Testing are:**

* Unit Testing
* Smoke Testing
* Sanity Testing
* Integration Testing  `
* User Acceptance Testing

#### 2. Non-functional Testing-

If Black box testing is used to test more aspects other than functionalities and features it comes under the umbrella of non-functional testing. Non-functional testing revolves around examining how well the system does a job.

Capable of working when the load is maximum

* Compatible to work with different devices
* Easy to use

#### 3. Regression Testing

Regression testing helps to find if the new codes have had any ill effects on the already existing ones. Regression testing is basically selecting complete or parts of test cases that had already been executed to make sure that the functionalities do have any abnormalities. In easy words, it tests to make sure that the new codes do not have side effects on the old codes. The different regression testing tools are Selenium, Quick Test Professional or QTP, Rational Functional Tester.

**6.5.2 Techniques of Black Box Testing-**

#### 1. Equivalence Partitioning

As the name suggests, the inputs are partitioned into groups or more literally partitions. Only one input from every group is tested to find the results. The inputs are usually numeric values or a set of values or Boolean conditions. For example, if the field accepts an integer in the range 1 and 20 then:

Valid Equivalence Class Partition: 1 to 20 inclusive.

Invalid Equivalence Class Partition: Less than 1 or more than 20, decimal numbers or alphabets and other non-numeric characters.

-

#### 2.Boundary Value Testing

In boundary value analysis the answers are within specific boundaries. The two ends, the inner and the outer limits are considered in this type of testing. For example, an offer is valid for customers between the ages of 18 and 30 only. Therefore other values such as 17, 18, 30, or 31 can be tested to check whether the inputs are accepted.

**3.State Transition Testing**

This testing technique uses the inputs, outputs, and the state of the system during the testing phase. It checks the software against the sequence of transitions or events among the test data.

Based on the type of software that is tested, it checks for the behavioral changes of a system in a particular state or another state while maintaining the same inputs.

For example, A login page will let you input username and password until three attempts. Each incorrect password will be sent the user to the login page. After the third attempt, the user will be sent to an error page. This state transition method considers the various states of the system and the inputs to pass only the right sequence of the testing.

**4. Decision Table Testing**

This approach creates test cases based on various possibilities. It considers multiple test cases in a [decision table](https://reqtest.com/requirements-blog/a-guide-to-using-decision-tables/) format where each condition is checked and fulfilled, to pass the test and provide accurate output. It is preferred in case of various input combinations and multiple possibilities

**5.Graph-Based Testing:**

It is similar to a decision-based test case design approach where the relationship between links and input cases are considered.

**6. Error Guessing Technique:**

This method of designing test cases is about guessing the output and input to fix any errors that might be present in the system. It depends on the skills and judgment of the tester.

Compari

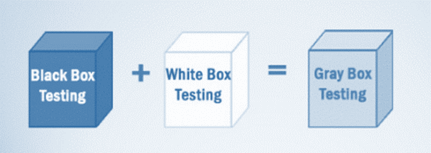
**6.6 Gray Box Testing:**

Gray box testing is a software testing technique to test a software product or application with partial knowledge of internal structure of the application. The purpose of grey box testing is to search and identify the defects due to improper code structure or improper use of applications.

Gray Box Testing is a software testing method, which is a combination of both White Box Testing and Black Box Testing method.

It is primarily used in integration testing and penetration testing.

Gray box testing is a software testing technique to test a software product or application with partial knowledge of internal structure of the application. The purpose of grey box testing is to search and identify the defects due to improper code structure or improper use of applications.



**Techniques used for Grey box Testing are-**

* **Matrix Testing**

This testing technique comes under Grey Box testing. It defines all the used variables of a particular program. In any program, variable are the elements through which values can travel inside the program. It should be as per requirement otherwise, it will reduce the readability of the program and speed of the software. Matrix technique is a method to remove unused and uninitialized variables by identifying used variables from the program.

* **Orthogonal Array Testing**

The purpose of this testing is to cover maximum code with minimum test cases. Test cases are designed in a way that can cover maximum code as well as GUI functions with a smaller number of test cases.

* **Pattern Testing**

Pattern testing is applicable to such type of software that is developed by following the same pattern of previous software. In these type of software possibility to occur the same type of defects. Pattern testing determines reasons of the failure so they can be fixed in the next software.

**6.7 Test Plan and Test Cases**

Implementation stage starts the testing process. In development stage code is reviewed. Test

Plan included all phases of testing and also used as a guide for the overall testing process. The

Test plan was designed before the implementation of the system. Test Objectives, Test Schedule

Test Logistics and Test Strategies are included in Test plan. Test cases are highly considered in here.

Test cases were created according to the designed test plan. That contains data, procedure, and expected result and represents which use to system or part of the system run. To reduce complexity of the testing process test cases were designed for each module independently. The following tables specify some test cases.

Manual Testing method and procedure used for testing rather than automation tools and technologies.

**6.7.1 Test case for Home Page**

|  |  |
| --- | --- |
| Test Scenario id | TS\_01 |
| Test Scenario | Check Home page |
| Test case | Check Url |
| Test case id | TC\_01 |
| Steps | Go to home page url link  click enter |
| Input | <file:///C:/Users/Mayuri/Desktop/store.html> |
| Expected Result | Display Home page |
| Actual result | Same as Expected result |
| Status | Pass |

**6.7.2 Test case for Login page**

|  |  |
| --- | --- |
| Test Scenario id | TS\_02 |
| Test Scenario | Check Login page |
| Test case | Check Url |
| Test case id | TC\_02 |
| Steps-1  Step-2  Step-3  Step-4 | Go to login page and enter  Username-correct  Password –correct  Click on login button  Username-incorrect  Password-correct  Click on login button  Username-correct  Password-incorrect  Click on login button  Username-incorrect  Password-incorrect  Click on login button |
| Input | <file:///C:/Users/Mayuri/Desktop/loginpage/index.html> |
| Expected Result | Display Login page and only correct username and password then login the page |
| Actual result | Same as Expected result |
| Status | Pass |

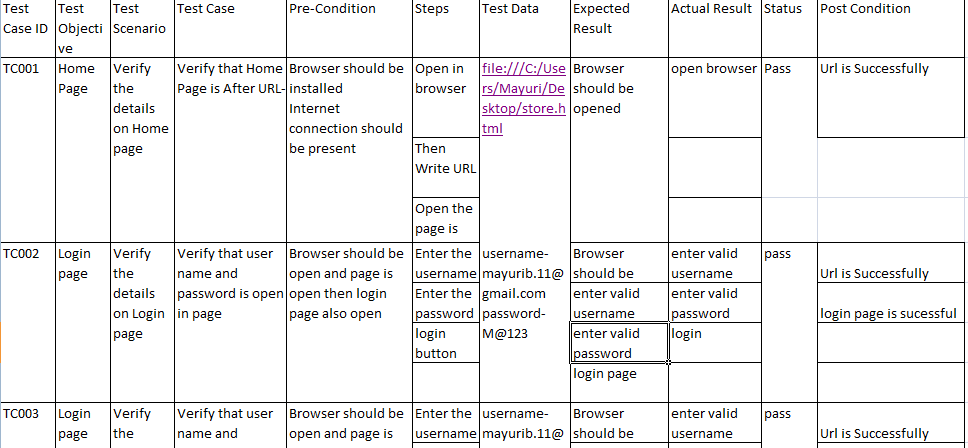
**6.7.3 Test case for About page**

|  |  |
| --- | --- |
| Test Scenario id | TS\_03 |
| Test Scenario | Check About page |
| Test case | Check Url |
| Test case id | TC\_03 |
| Steps | Go to About page and click enter |
| Input | <file:///C:/Users/Mayuri/Desktop/loginpage/about%20us.html> |
| Expected Result | Display About us page |
| Actual result | Same as Expected result |
| Status | Pass |

**6.7.4 Test case for FeedBack page**

|  |  |
| --- | --- |
| Test Scenario id | TS\_04 |
| Test Scenario | Check feedback page |
| Test case | Check Url |
| Test case id | TC\_04 |
| Steps | Go to feedback page and enter the feedback |
| Input | <file:///C:/Users/Mayuri/Desktop/loginpage/feedback.html> |
| Expected Result | Displayfeedback page |
| Actual result | Same as Expected result |
| Status | Pass |

**TESTCASE REPORT**

****

**6.8 AUTOMATION TESTING-**

Automation testing is process of changing any manual test cases into the test script by using automation tool. Making Testing Fast, Reliable, and Easy Throughout The Software Development Process. Perform Automated Testing in Popular Frameworks, include Appium, Espresso, XCTest/XCUITest. device cloud testing. real device testing. real device. android testing.

Automation testing is the process of testing software and other tech products to ensure it meets strict requirements. Essentially, it’s a test to double-check that the equipment or software does exactly what it was designed to do. It tests for bugs, defects, and any other issues that can arise with product development.

Although some types of testing, such as regression or functional testing can be done manually, there are greater benefits of doing it automatically. Automation testing can be run at any time of the day. It uses scripted sequences to examine the software. It then reports on what’s been found, and this information can be compared with earlier test runs. Automation developers generally write in the following programming languages: C#, JavaScript, and Ruby.

Many software businesses will have an appointed [QA (quality assurance) automation tester](https://www.globalapptesting.com/how-we-help/qa-teams). They design and write the test scripts in the beginning stages. The QA automation tester will work with automation test engineers and product developers to actually test the software and products. They will form a team and control the test automation initiatives, and use different types of test automation frameworks to establish the best one for successful test automation.

When starting to work with unit testing frameworks, the team needs to know about its attributes, runners, assertions, screen shots, test suites, and CI (continuous integration). Popular user testing frameworks include JUnit for Java and Pytest for Python.

**6.8.1 Selenium**

It is open source and most commonly used tool in automation testing

It support multiple languages like java, c#, python, ruby etc

It support multiple platforms like windows, Mac, Linux etc

It supports multiple browsers like chrome, edge, Firefox, safari opera etc

This tool is used to test web based application with the help of test script and test script can be written in java programming language

Parallel testing means at a one time testing on different browsers

Third party tools plug-in /integrate like testing, Github, Jenkins etc

Large community

**6.8.2 TESTNG**

TestNG is a testing framework inspired from JUnit and NUnit but introducing some new functionalities that make it more powerful and easier to use, such as:

* Annotations.
* Run your tests in arbitrarily big thread pools with various policies available (all methods in their own thread, one thread per test class, etc...).
* Test that your code is multithread safe.
* Flexible test configuration.
* Support for data-driven testing (with @DataProvider).
* Support for parameters.
* Powerful execution model (no more TestSuite).
* Supported by a variety of tools and plug-ins (Eclipse, IDEA, Maven, etc...).
* Embeds BeanShell for further flexibility.
* Default JDK functions for runtime and logging (no dependencies).
* Dependent methods for application server testing.

TestNG is designed to cover all categories of tests:  unit, functional, end-to-end, integration, etc...

I started TestNG out of frustration for some JUnit deficiencies which I have documented on my weblog [here](https://beust.com/weblog/testsetup-and-evil-static-methods/) and [here](https://beust.com/weblog/junit-pain/) Reading these entries might give you a better idea of the goal I am trying to achieve with TestNG.  You can also check out a quick [overview of the main features](https://beust.com/weblog/announcing-testng-1-0/) and an [article](https://beust.com/weblog/using-annotation-inheritance-for-testing/) describing a very concrete example where the combined use of several TestNG's features provides for a very intuitive and maintainable testing design.

**7.8.3Selenium code**

**package** mypage;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.chrome.ChromeDriver;

**public** **class** System {

**public** **static** **void** main(String args[]) **throws** InterruptedException {

WebDriver driver = **new** ChromeDriver();

System.*setProperty*("webdriver.chrome.driver","C:\\Users\\Mayuri\\Downloads\\chromedriver.exe");

driver.get("file:///C:/Users/Mayuri/Desktop/store.html");

driver.manage().window().maximize();

Thread.*sleep*(2000);

driver.findElement(By.*xpath*("/html/body/div[1]/a[7]")).click();

Thread.*sleep*(1000);

driver.findElement(By.*cssSelector*("input[placeholder='Email address']")).sendKeys("mayurib.11");

Thread.*sleep*(1000);

driver.findElement(By.*cssSelector*("input[placeholder='Password']")).sendKeys("M@123456");

Thread.*sleep*(1000);

driver.findElement(By.*xpath*("/html/body/div/form/input")).click();

driver.navigate().back();

Thread.*sleep*(3000);

driver.findElement(By.*xpath*("/html/body/div[2]/div/div/h3")).click();

driver.findElement(By.*id*("fname")).sendKeys("Mayuri");

driver.findElement(By.*id*("mname")).sendKeys("Siddhu");

driver.findElement(By.*id*("lname")).sendKeys("Bandgar");

driver.findElement(By.*id*("moname")).sendKeys("7058517538");

driver.findElement(By.*name*("birthday")).sendKeys("11-11-1997");

driver.findElement(By.*name*("agename")).sendKeys("25");

driver.findElement(By.*name*("ename")).sendKeys("Mayuri123@gmail.com");

driver.findElement(By.*name*("gname")).sendKeys("female");

driver.findElement(By.*id*("aname")).sendKeys("Pune");

driver.findElement(By.*xpath*("/html/body/div[2]/div/div/form/fieldset/input[10]")).click();

driver.quit();

}

}

**7.8.3 CUCUMBER FRAMEWORK**

Cucumber Framework in Selenium is an open-source testing framework that supports Behavior Driven Development for automation testing of web applications. The tests are first written in a simple scenario form that describes the expected behavior of the system from the user's perspective.

Cucumber is an open-source software testing tool written in Ruby. Cucumber enables you to write test cases that anyone can easily understand regardless of their technical knowledge.

Cucumber code-

Feature: functionality

Scenario: verify sucessful login

Given user navigates to the website

When user enter valid username

When enter valid password

Then login must be successfully

package cucumber;

import org.junit.Assert;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import io.cucumber.java.en.And;

import io.cucumber.java.en.Given;

import io.cucumber.java.en.Then;

import io.cucumber.java.en.When;

import io.github.bonigarcia.wdm.WebDriverManager;

public class Loginpage1 {

WebDriver driver=null;

@Given("^User Launch the application on browser$")

public void user\_launch\_the\_application\_on\_browser() {

System.out.println("Lunching the Url");

WebDriverManager.chromedriver().setup();

WebDriver driver=new ChromeDriver();

driver.manage().window().maximize();

driver.get("file:///C:/Users/Mayuri/Desktop/store.html ");

System.out.println("login page is launched"); }

@When("^User is on login page and verify login page$")

public void user\_is\_on\_login\_page\_and\_verify\_login\_page() {

String Pagetitle =driver.getTitle();

if(Pagetitle.equals("login page"))

{

System.out.println("login page verified");

}

else

{

System.out.println("login page verification field");

}

}@And("^user enters username and password$")

public void user\_enters\_username\_and\_password() {

driver.findElement(By.id("email")).sendKeys("mayuribandgar1@gmail.com");

driver.findElement(By.id("pass")).sendKeys("Mayu@123");

driver.findElement(By.id("//button[normalize-space()='log In']")).click();

}@Then("message displayed login successful")

public void message\_displayed\_login\_successful() throws Throwable {

String exp\_message = "Welcome to your account. Here you can manage all of your personal information and orders.";

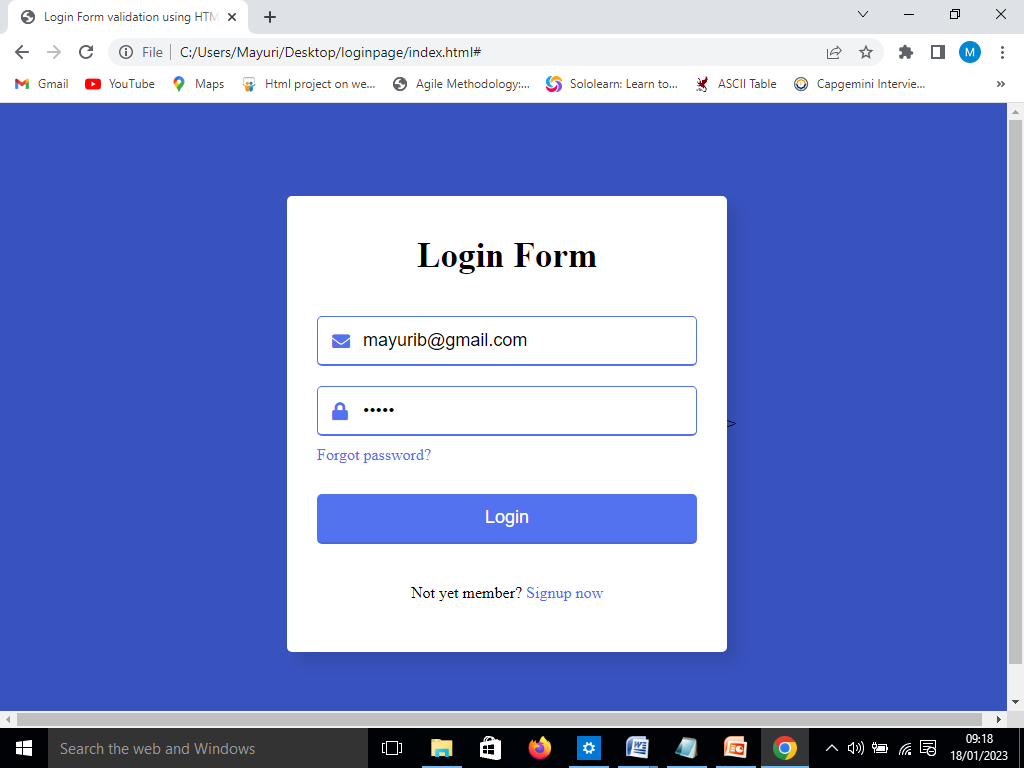
String actual = driver.findElement(By.cssSelector(".info-account")).getText();

Assert.assertEquals(exp\_message, actual);

driver.quit();

}

**Output-**

****

**8.CONCLUSION**

Conclusion of our project is, the mobile store system needs advance technology for smartly and speedily. In this software we will try to all feature including in this, like bulling staff information, how many products sold, how many products we order, all information we can store in this software. We will try to a helpful software for mobile shop.