ABSTRACT

Television is present in every house today. There are so many brands that produce TV every single year. These companies also produce many models of television with separate specifications. There now are numerous electronic shops that sell TV. It becomes difficult for these retailers to maintain manual record of these television models. We have designed software that maintains the accounts for a retail television store, to help them digitize their records. We have used various applications to make sure that these can be accessed only by the owner of the firm.

CONTENTS

INTRODUCTION

In recent times technology has developed drastically and electronic products are increasing day by day. One of them, also probably one of the most popular one, is television. Television is present in every house today. There are so many brands that produce TV every single year. These companies also produce many models of television with separate specifications. There is a pretty good market too for them. Hence there now are numerous electronic shops that sell TV.

It becomes difficult for these retailers to maintain manual record of these television models. So we decided to create software that maintains the accounts for a retail television store, to help them digitize their records.

We have used MySQL as the basic database and web development using- HTML, CSS, JavaScript and PHP and used XAMPP to connect the database to the webpage. We have ensured that the sensitive data (financial accounts) remains accessible to only those with the access privilege in the firm.

CONTEXT DIAGRAM

We have designed seven modules for the software. The basic software of the business accounts has these allied systems which have to be developed to create the software.

These are shown in the context diagram below:

MODULE WISE ELUCIDATION

We have designed seven modules, query execution and a login system for the software. . The basic software of the business accounts has these allied systems which have to be developed to create the software.

A) Login

Input: Username and Password

Output: Home page or error page

B) Query

Input: Query

Output: Query executed or not

1. Product Details

Input:

Product Code
Brand
Description
Wholesale cost
Rating

Output:

Sr. N	Product Code	Brand	Description	Whole sale cost	Rating

2. Customer Details

Input:

Costumer Code
Name
Contact Number
Email Address
Delivery Address
Total purchases till date

Output:

Sr. No	Customer Code	Name	Contact Number	Email Address	Delivery Address	Total purchases till date

3. Wholesaler Details

Input:
Wholesaler code
Name
Contact Number
Shop Address
Total purchased items

Output:

Sr. No.	Wholesaler Code	Name	Contact Number	Shop Address	Total Items Purchased

4. Purchase Details
Information about goods purchases by the retailer

Input:
Product Code
Wholesaler Code
Quantity
Receipt Number
Date
Contact Number
Shop Address
Total purchased items

Output:

Sr. No.	Prod. Code	Wholesaler Code	Quantity	Receipt N.	Date	Contact No.	Address	Total Items

5. Sales Details

Input:

Date
Product Code
Costumer Code
Quantity
Total Cost
Bill Number
Contact Number

Output:

Sr. No.	Date	Product Code	Costumer Code	Quantity	Total Cost	Bill Number	Contact Number

6. Employee Details

Input:

Employee Number
Name
Post
Contact Number
Salary

Output:

Sr. No	Employee Number	Name	Post	Contact Number	Salary

7. Total Balance sheet

Input:

Date
Transaction Number
Details
Amount

Output:

Sr. No	Date	Transaction number	Details	Amount (with – for expense and + for sales)

REQUIREMENT ANALYSIS

Requirement analysis is an important part of the process of software development.

4.1 FUNCTIONAL REQUIREMENTS

Functional requirements define the fundamental actions that system must perform.

1. Login Function

The business accounts of a firm are sensitive documents that should only be accessible only to authorised personnel. Hence an efficient login system should be installed.

2. Query Function

This function is the backbone of the software. It must support execution of any SQL query to update, add and delete any data in the database.

3. Product Details Module

This module shows details of all the products purchased by the firm. It includes details like product code, brand, description, wholesale cost and rating in a tabular format:

Sr. N	Product Code	Brand	Description	Whole sale cost	Rating

This enables analysis in terms of if more stock is required to be purchased and at what cost must the deal be negotiated.
It should allow addition of a new entry and updation or deletion of an old one.

4. Customer Details Module

This module shows details of all the products purchased by the firm. It includes details like costumer code, name, contact number, email address, delivery address and total purchases till date in a tabular format.

Sr. No	Customer Code	Name	Contact Number	Email Address	Delivery Address	Total purchases till date

This enables analysis in terms of how valuable a costumer is, does he give timely payment and what are his/ her choices.
It should allow addition of a new entry and updation or deletion of an old one.

5. Wholesaler Details Module

This module shows details of all the products purchased by the firm. It includes details like wholesaler code, name, contact number, shop address and total purchases till date in a tabular format.

Sr. No.	Wholesaler Code	Name	Contact Number	Shop Address	Total Items Purchased

This enables analysis in terms of how reliable a seller is.
It should allow addition of a new entry and updation or deletion of an old one.

6. Purchase Details Module

This module shows details of all the products purchased by the firm. It includes details like product code, wholesaler code, quantity, receipt number, date, contact number, shop address and total purchases till date in a tabular format

Sr. No.	Prod. Code	Wholesaler Code	Quantity	Receipt N.	Date	Contact No.	Address	Total Items

This enables analysis in terms of if more stock is required to be purchased and from whom.
It should allow addition of a new entry and updation or deletion of an old one.

7. Sales Details Module

This module shows details of all the products purchased by the firm. It includes details like date, product code, costumer code, quantity, bill number and contact number in a tabular format**.**

Sr. No.	Date	Product Code	Costumer Code	Quantity	Total Cost	Bill Number	Contact Number

This enables analysis in terms of how well the product is selling, if more stock is required to be purchased and also records details of sales made for future reference.
It should allow addition of a new entry and updation or deletion of an old one.

8. Employee Details Module

This module shows details of all the products purchased by the firm. It includes details like employee number, name, post, salary and contact number in a tabular format**.**

Sr. No	Employee Number	Name	Post	Contact Number	Salary

This presents data about the employee.

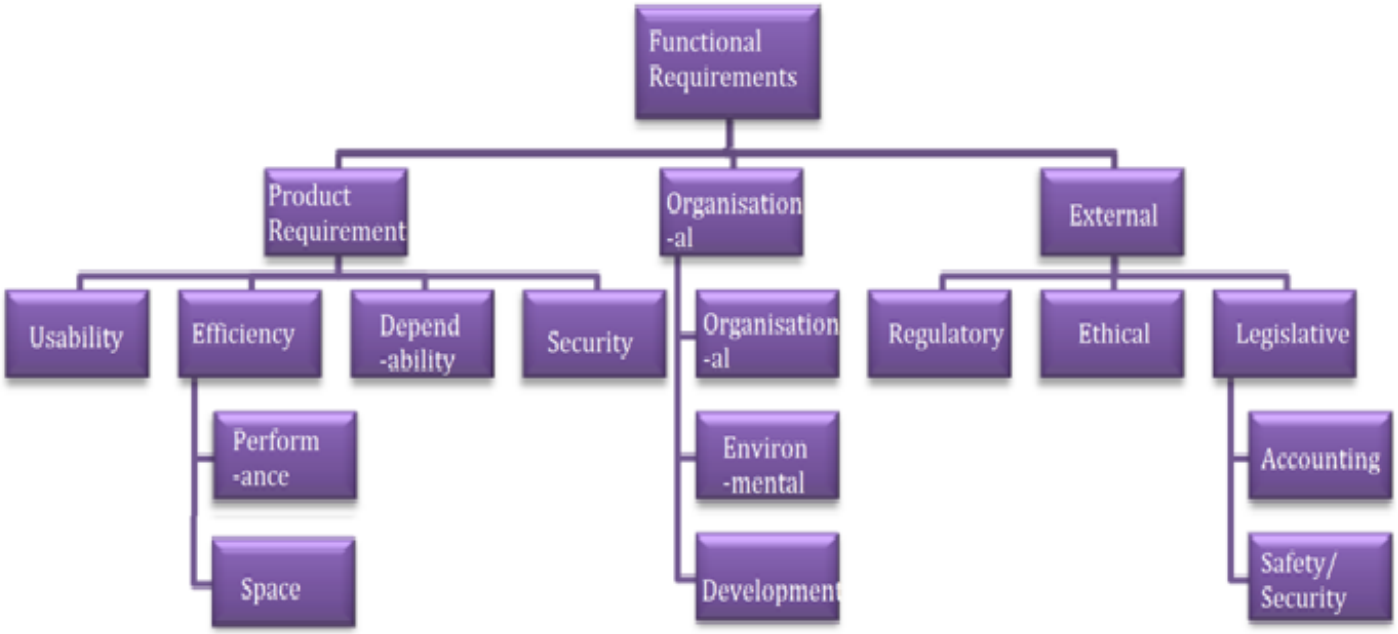
9. Total Balance Sheet Module

This module shows details of all the products purchased by the firm. It includes details like date, transaction number, details and amount involved in a tabular format**.**

Sr. No	Employee Number	Name	Post	Contact Number	Salary

This presents data about the all of the firm’s money exchange transactions.

4.2. NON FUNCTIONAL REQUIREMENTS



Since the flowchart doesn't have space, the details have been listed down category-wise:

A) <u>Product Requirement:</u>

1) Usability Requirements:
The user would authenticate themselves to information using the login function.

2) Efficiency Requirements

A) Performance Requirement: Response time of a query must be less than 15 seconds for all actions.

B) Space Requirements Around 1 Terabyte memory space should be allocated to store the data in the database.

3) Dependability Requirements:

It will be usable everyday with downtime not exceeding 5 seconds and without having breakdown.

4) Security Requirements:

All system data must be backed up every 24 hours and the backup copies stored in another server at different building or location for disaster recovery.
A firewall sytem will be built-in. The IP address of any attempted security breaching shall be noted and notified to the admin.

B) Organisational Requirements:

1) Operational Requirements:

The source codes for the system will be well documented for ease of maintenance and upgrading the system in future.

2) Environmental Requirements:

The software will be used at personal laptops of the authorised personnel.

3) Development Requirements:

The system that runs on different type of browser such as Internet Explorer, Mozilla and Google Chrome. It shall be developed using Java, Javascript, Apache and HTML, CSS programming languages. It will run using MySql Database Server and XAMPP application to connect the browser to the data.

C) External Requirements:

1) Regulatory Requirements:

Admin can view all the information stored in the software database.

2) Ethical Requirements:

The admin can view all employee details, complaints and reviews.
Any other employee can not do the same for his/her subordinates, colleagues or higher authorities.

3) Legislative Requirements:

A) Accounting Requirements:
Revenue information must be visible to the admin only. The owner will be the admin.

B) Safety/ Security Requirements:
All access permission for the system data may only be changed by the system's administrator.

RISKS ASSESSMENT

Risks are events, the occurrence of which will hamper the project completion schedule, project quality and the requirement expectation of the project.

We assess the risks that could possibly have occurred during the making of the project

.

5.1. PRODUCT RISKS

Product risks affect the quality or performance of the software being developed.

Estimation Error: The time estimated for requirement procurement has been surpassed and the project is running in delay. Hence coding has to be rushed.
The time required for debugging the code is underestimated.
The memory space or processing system version required is not calculated correctly.

Specifications delay: Microsoft Visual C++ 2008 is not available.

Requirements delay: Due to lack of experience and knowledge in the required domain, members decide to learn the same online which is time taking.

Complaints and revisions: Many requirements change requests; customer complaints.

CASE tool underperformance: The CASE tool of memory storage (MySQL)is not efficient and speedy enough as per the requirements of the software.
The processer (Apache) used in the system cannot process as many transactions per second as expected.

5.2. PROJECT RISKS

Project risks affect schedule or resources.

Ownership Change: The financial manager or the ownership of the software retail shop changes and they decide to this project less priority. Hence the budget and number of programmers reduces.

Organizational financial problems force reductions in the project budget

Requirements Delay: Late delivery of hardware or support software (Eg: DBMS-MySQL, Xampp).

Specifications delay: Microsoft Visual C++ 2008 is not available.

Requirements delay: Due to lack of experience and knowledge in the required domain, members decide to learn the same online which is time taking.

Requirement Change: New module (eg: tax details) can be proposed to be added. Hence additional memory space, time and effort is required. Costumers fail to understand the extent and impact of this revision.

Estimation Error: The time estimated for requirement procurement has been surpassed and the project is running in delay. Hence coding has to be rushed.

Staff: Reluctance by team members to use tools and complaints about CASE tools like that of memory storage (MySQL) and processing software (Apache).
Poor staff morale
Training for members on using web-tools, database connection is not available.

5.3. BUSINESS RISKS

Business risks affect the organisation developing or procuring the software.

Complaints and revisions:
Many requirements change requests.
Many customer complaints.
Many reported technology problems.

Product Competition:
There are so many brands that produce TV every single year. These companies also produce many models of television with separate specifications. There is a pretty good market too for them. Hence there now are numerous electronic shops that sell TV. They have softwares with updated systems and have a faster response time.

Technology Change:
The initial technology of RDBMS (Relative database management system) of MySQL and Oracle is superseded by introduction of new document based DBMS, like mongoDB and Cassandra, which are faster. Hence basic storage structure has to be revised.

Costumers fail to understand the extent and impact of this revision.

CODING DETAILS

❖ **FRONT-END:**

HTML, CSS, JavaScript and PHP: To make the webpage.

❖ **BACK-END:**

Database: MySQL for storing data

PHP: For connecting the data to the web page.

❖ **FRONT-END:**

6.1. Query Function:

```
<html>
<head>
<style type="text/css">
body{
        text-align: center;
        background-color: black;}
th{text-align:center;
        background-color: orange;
        color:black;
        font-family: arial-black;
        font-size:20px;}
td{text-align:center;
        color:white;
        font-family:tahoma;
        font-size:15px; }
CAPTION{text-align:center;
        color:RED;
        font-family: arial-black;
        font-size: 30px;
        text-decoration:underline; }
h1{text-align:center;
        background-color:orange;
        color:Black;
        font-family: arial-black;
        font-size:35px;}
h2{text-align:center;
        color:orange;
        font-family: arial-black;
        font-size:25px;}
form{text-align:center;
        position:relative;
        width:600px;
        left:400px;
        background-color:orange
        border:blue solid 3px;
        color:red;
        font-family: arial-black;
        font-size:25px;}
#lo{position:absolute;
        text-align:center;
        top:530px;
        left:1220px;
        width:100px;
        background-color:orange;
        border:orange solid 3px;
        color:red;
```

```
        font-family: arial-black;
        font-size:25px;
}
ul {list-style-type: none;
margin: 25px;
   padding: 0;
   overflow: hidden;
  border: orange solid 3px;
}
li a {float:left;
         display: block;
        width:18%;
        height: 100px;
        text-align: center;
   padding: 4px;
        color: blue;
        font-size:25px;
}
</style>
</head>
<body>
<h1><img src="tv.jpg" align="Left"
height="100px"><br/>
Television Shop <img src="tv.jpg"
align="Right"
height="100px"><br/><br/></h1><br/>
<form action="Hi.php"
method="post"><br/>
Enter Query: <input type="text"
name="query"
style="width:500px;"><br/>
   <input type="submit" value="
Submit Query     "><br/><br/></form>
<?php
$dbcon =
mysql_connect('localhost','root', '');
if (!$dbcon){
        die('Connection Error'
.mysql_error());
}
$dbselect = mysql_select_db('tv',
$dbcon);
if(!$dbselect){
die('Cant connect: '
.mysql_error());}
$query=$_POST["query"];
$result=mysql_query($query);
if ($result){
echo '<h2>Your query has been
executed.</h2>';}
mysql_close($dbcon);?>
<br/><br/>
<ul>
<li><a
href="Product_Details.php">
Product Details </a></li>
<li><a
href="Employee_Details.php">
Employee Details </a></li>
<li><a
href="Purchase_Details.php">
Purchase Details </a></li>
<li><a href="Sales_Details.php">

Sales Details </a></li>
<li><a
href="Total_Balance_Sheet.php">
Total Balance Sheet </a></li>
</ul>
<form action="index.html"
method="post" id="lo">
<input type="submit" value="
Log out     ">
</form>
</body>
</html>
```

6.2. Login and Logout:

6.2.1. Login Page:

```
<html>
<head>
<style type="text/css">
body{text-align: center;
        background-color: black;}
div{text-align:center;
        position: absolute;
        width: 300px;
        top:250px;
        left:520px;
        background-color:orange;
        border:blue solid 3px;
        color:red;
        font-family: arial-black;
        font-size:25px;}
h1{text-align:center;
        background-color:orange;
        color:Black;
        font-family: arial-black;
        font-size:35px;}
</style> </head>
<body>
<h1><img src="tv.jpg" align="Left"
height="100px"><br/>
Television Shop
<img src="tv.jpg" align="Right"
height="100px"><br/><br/></h1>
<div>
<u>Log-In</u>
<br/>
<form action="login.php"
method="post">
Username <input type="text"
name="user"><br>
Password  <input type="password"
name="pass"><br/>
<input type="submit" value="     Log
in     ">
</form>
</div>
</body>
</html>
```

6.2.2. Login Function:

```
<?php
$user=$_POST["user"];
$pass= $_POST["pass"];
if ($user == "Admin" && $pass == "tv")
{
echo "Welcome";
header("Location: Hi.php");
}
else{
        header("Location: Bye.php");
}
?>
```

6.2.2. Invalid Login page:

```
<html><head>
<style type="text/css">
body{text-align: center;
        background-color: black; }
th{text-align:center;
        background-color: orange;
        color:black;
        font-family: arial-black;
        font-size:20px; }
td{text-align:center;
        color:white;
        font-family:tahoma;
        font-size:15px; }
CAPTION{text-align:center;
        color:RED;
        font-family: arial-black;
        font-size: 30px;
        text-decoration:underline; }
h1{text-align:center;
        background-color:orange;
        color:Black;
        font-family: arial-black;
        font-size:35px;}
h2{text-align:center;
        color:orange;
        font-family: arial-black;
        font-size:25px;}
form{text-align:center;
        position:relative;
        width:600px;
        left:400px;
        background-color:orange;
        border:blue solid 3px;
color:red;
      font-family: arial-black;
      font-size:25px;}
ul { list-style-type: none;
    margin: 25px;
    padding: 0;
    overflow: hidden;
border: orange solid 3px;}
li a {
          float:left;
    display: block;
          width:12.5%;
          height: 100px;
          text-align: center;
    padding: 4px;
          color: blue;
          font-size:25px; }
</style></head>
<body>
<h1><img src="tv.jpg" align="Left"
height="100px"><br/>
Television Shop
<img src="tv.jpg" align="Right"
height="100px"><br/><br/></h1><br/>
<h2>Incorrect Login Credentials.
Please try again.</h2>
</body></html>
```

6.2.3. Valid Login page:

Redirects to query page.

6.2.3. Logout page:

```
<form action="index.html" method="post" id="lo">
<input type="submit" value="     Log out     ">
</form>
```

6.3. Modules :

6.3.1. Product Details:

This is extensively described. All other modules have only the core code written here.

```
<html>
<head>
<style type="text/css">
body{border:blue solid 3px;
        text-align: center;
        background-color: black;}
th{text-align:center;
        background-color: orange;
        color:black;
        font-family: arial-black;
        font-size:20px;}
td{text-align:center;
        color:white;
        font-family:tahoma;
        font-size:15px;}
CAPTION{text-align:center;
        color:RED;
        font-family: arial-black;
        font-size: 30px;
        text-decoration:underline; }
h1{text-align:center;
        background-color:orange;
        color:Black;
        font-family: arial-black;
        font-size:35px;
}
ul {
   list-style-type: none;
   margin: 25px;
   padding: 0;
   overflow: hidden;
        border: orange solid 3px;
        }
li a {
        float:left;
display: block;
        width:12.5%;
        height: 100px;
        text-align: center;
   padding: 4px;
        color: blue;
        font-size:25px; }
</style></head><body>
<h1><img src="tv.jpg"
align="Left"
height="100px"><br/>
Television Shop
<img src="tv.jpg" align="Right"
height="100px"><br/><br/></h1
>
<?php
$dbcon =
mysql_connect('localhost','root',
'');
if (!$dbcon){
        die('Connection Error'
.mysql_error());}
$dbselect = mysql_select_db('tv',
$dbcon);
if(!$dbselect){
        die('Cant connect: '
.mysql_error());
}
//Product Details
$query="SELECT * from
product_details";
$data=mysql_query($query);
echo "<br/>
<br/>
mysql_close($dbcon);
?>
```

```
 <table border = 1
align=Center><CAPTION>
 PRODUCT DETAILS</CAPTION>
<tr>
<th>Sr.No.</th>
<th>Product Code</th>
<th>Brand</th>
<th>Description</th>
<th>Wholesale cost</th>
 <th>Rating</th>
</tr>";
while($record=mysql_fetch_array($data)
){
        echo "<tr>";
        echo "<td>".$record["Sr. No."];
        echo "<td>".$record["Product
Code"];
        echo "<td>".$record["Brand"];
        echo
"<td>".$record["Description"];
        echo "<td>".$record["Wholesale
cost"];
        echo "<td>".$record["Rating"];
        echo "</tr>";}
echo "</table><br/><br/>";
<ul>
<li><a href="Hi.php"> Home
</a></li>
<li><a
href="Customer_Details.php">
Customer Details </a></li>
<li><a
href="Employee_Details.php">
Employee Details </a></li>
<li><a
href="Wholesaler_Details.php">
Wholesaler Details </a></li>
<li><a
href="Purchase_Details.php">
Purchase Details </a></li>
<li><a
href="Sales_Details.php"> Sales
Details </a></li>
<li><a
href="Total_Balance_Sheet.php"
> Total Balance Sheet </a></li>
</ul>
</body></html>
```

6.3.2. Customer Details:

```
$query="SELECT * from
customer_details";
$data=mysql_query($query);
echo "<br/><br/><table border = 1
align=Center><CAPTION>CUST
OMER DETAILS</CAPTION>
<tr><th>Sr.No.</th>
<th>Customer Code</th>
<th>Name</th>
<th>Contact No.</th>
<th>Email Address</th>
<th>Delivery Address</th>
<th>Total Purchases till date</th>
</tr>";
while($record=mysql_fetch_array($dat
a)){echo "<tr>";
        echo "<td>".$record["Sr. No."];
        echo "<td>".$record["Customer
Code"];
        echo "<td>".$record["Name"];
        echo "<td>".$record["Contact
Number"];echo "<td>".$record["Email
Address"];
echo "<td>".$record["Delivery
Address"];
        echo "<td>".$record["Total
Purchases"];
        echo "</tr>";}
echo "</table><br/><br/>";
```

6.3.3. Wholesaler Details:

```
$query="SELECT * from
wholesaler_details";
$data=mysql_query($query);
echo "<br/>
<br/>
<table border = 1
align=Center><CAPTION>WHOLE
SALER DETAILS</CAPTION>
<tr>
<th>Sr.No.</th>
<th>Wholesaler Code</th>
<th>Name</th>
<th>Contact Number</th>
<th>Shop Address</th>
<th>Total Purchased</th>
</tr>";
```

```
while($record=mysql_fetch_array($
data)){
        echo "<tr>";
        echo "<td>".$record["Sr.
No."];
        echo
"<td>".$record["Wholesaler Code"];
        echo
"<td>".$record["Name"];
        echo "<td>".$record["Contact
Number"];
echo "<td>".$record["Address"];
        echo "<td>".$record["Total
purchased"];
        echo "</tr>";
}echo "</table><br/><br/>";
```

6.3.4. Purchase Details:

```
$query="SELECT * from
purchase_details";
$data=mysql_query($query);
echo "<br/>
<br/>
<table border = 1
align=Center><CAPTION>PURCHASE
DETAILS</CAPTION>
<tr>
<th>Sr.No.</th>
<th>Product Code</th>
<th>Wholesaler Code</th>
<th>Quantity</th>
<th>Reciept Number</th>
<th>Date</th>
<th>Contact Number</th>
<th>Shop Address</th>
<th>Total Purchases till date</th>
</tr>";
while($record=mysql_fetch_array($data))
{
```

```
w echo "<tr>";
echo "<td>".$record["Sr. No."];
echo "<td>".$record["Product
Code"];
echo
"<td>".$record["Wholesaler
Code"];
echo"<td>".$record["Quantity"]
;
echo "<td>".$record["Reciept
Number"];
echo "<td>".$record["Date"];
echo "<td>".$record["Contact
Number"];
echo
"<td>".$record["Address"];
echo "<td>".$record["Total
Purchased"];
        echo "</tr>";}
echo "</table><br/><br/>";
```

6.3.5. Sales Details:

```
$query="SELECT * from
sales_details";
$data=mysql_query($query);
echo "<br/><br/>
<table border = 1
align=Center><CAPTION>SALES
DETAILS</CAPTION>
<tr><th>Sr.No.</th>
<th>Date</th>
<th>Bill Number</th>
<th>Product Code</th>
<th>Costumer Code</th>
<th>Quantity</th>
<th>Total Cost</th>
<th>Contact Number</th></tr>";
```

```
while($record=mysql_fetch_array($d
ata)){echo "<tr>";
echo "<td>".$record["Sr. No."];
echo "<td>".$record["Date"];
echo "<td>".$record["Bill Number"];
echo "<td>".$record["Product
Code"];
echo "<td>".$record["Costumer
Code"];
echo "<td>".$record["Quantity"];
echo "<td>".$record["Total Cost"];
echo "<td>".$record["Contact
Number"];
echo "</tr>";}
echo "</table><br/><br/>";
```

6.3.6. Employee Details:

```
$query="SELECT * from
employee_details";
$data=mysql_query($query);
echo "<br/><br/><table border = 1
align=Center><CAPTION>EMPLO
YEE DETAILS</CAPTION>
<tr><th>Sr.No.</th>
<th>Employee Number</th>
<th>Name</th>
<th>Post</th>
<th>Contact Number</th>
<th>Salary</th></tr>";
```

```
while($record=mysql_fetch_array($d
ata))
{echo "<tr>";
echo "<td>".$record["Sr. No."];
echo"<td>".$record["Employee_ID"]
; echo "<td>".$record["Name"];
echo "<td>".$record["Post"];
echo "<td>".$record["Contact
Number"];
echo "<td>".$record["Salary"];
echo "</tr>";}
echo "</table><br/><br/>";
```

6.3.7. Total Balance Sheet:

```
$query="SELECT * from
total_balance_sheet";
$data=mysql_query($query);
echo "<br/><br/><table border =
1align=Center><CAPTION>TO
TAL BALANCE
SHEET</CAPTION><tr>
<th>Sr.No.</th><th>Date</th>
<th>Transaction Number</th>
```

```
<th>Details</th> th>Amount</th></tr>";
while($record=mysql_fetch_array($data))
{echo "<tr>";echo "<td>".$record["Sr.
No."];echo "<td>".$record["Date"];
echo "<td>".$record["Transaction
Number"];echo
"<td>".$record["Details"];
echo "<td>".$record["Amount"];
echo "</tr>";}echo "</table><br/><br/>";
```

❖ **BACK-END:**

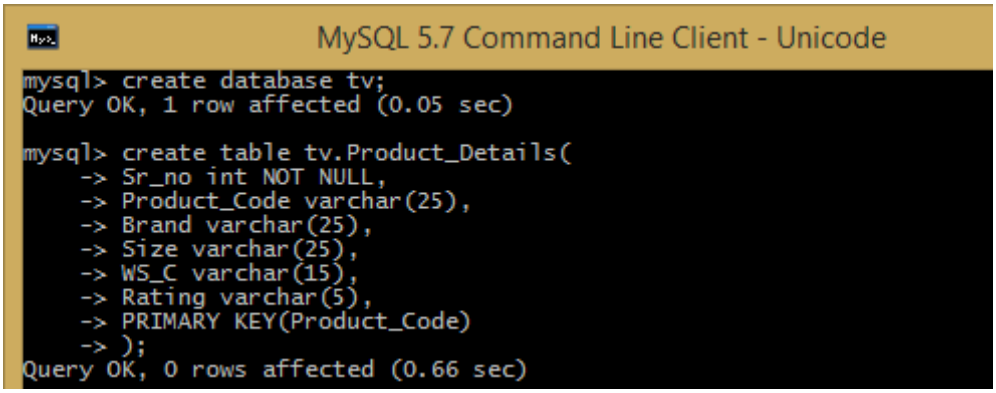
6.4. DATABASE (MySQL) :

6.4.1. Creating Database:

mysql> create database tv;
Query OK, 1 row affected (0.05 sec)

6.4.2. Making a Table:

```
mysql> create table tv.Product_details(
    -> Sr_no int NOT NULL,
    -> Product_Code varchar(25),
    -> Brand varchar(25),
    -> Size varchar(25),
    -> WS_C varchar(15),
    -> Rating varchar(5),
    -> PRIMARY KEY(Product_Code)
    -> );
Query OK, 0 rows affected (0.66 sec)
```



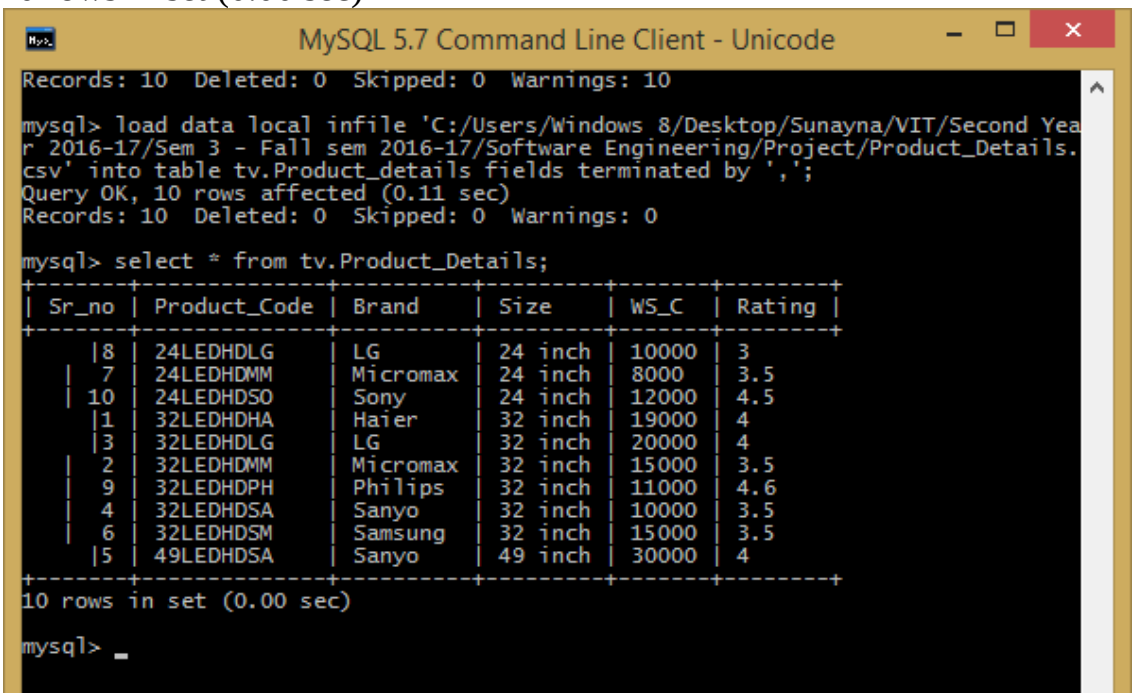
6.4.3. Loading data into the table:

mysql> load data local infile 'C:/Users/Windows 8/Desktop/Sunayna/VIT/Second Yea

r 2016-17/Sem 3 - Fall sem 2016-17/Software Engineering/Project/Product_Details.
csv' into table tv.Product_details fields terminated by ',';
Query OK, 10 rows affected (0.11 sec)
Records: 10 Deleted: 0 Skipped: 0 Warnings: 0

6.4.4. Viewing loaded data:

```
mysql> select * from tv.Product_Details;
+-------+--------------+----------+---------+-------+--------+
| Sr_no | Product_Code | Brand    | Size    | WS_C  | Rating |
+-------+--------------+----------+---------+-------+--------+
    |8 | 24LEDHDLG    | LG       | 24 inch | 10000 | 3
  |  7 | 24LEDHDMM    | Micromax | 24 inch | 8000  | 3.5
  | 10 | 24LEDHDSO    | Sony     | 24 inch | 12000 | 4.5
    |1 | 32LEDHDHA    | Haier    | 32 inch | 19000 | 4
    |3 | 32LEDHDLG    | LG       | 32 inch | 20000 | 4
  |  2 | 32LEDHDMM    | Micromax | 32 inch | 15000 | 3.5
  |  9 | 32LEDHDPH    | Philips  | 32 inch | 11000 | 4.6
  |  4 | 32LEDHDSA    | Sanyo    | 32 inch | 10000 | 3.5
  |  6 | 32LEDHDSM    | Samsung  | 32 inch | 15000 | 3.5
    |5 | 49LEDHDSA    | Sanyo    | 49 inch | 30000 | 4
+-------+--------------+----------+---------+-------+--------+
10 rows in set (0.00 sec)
```



6.5. CONNECTING TO THE DATABASE (PHP) :

6.5.1. Connecting to the Database:

```
<?php

$dbcon = mysql_connect('localhost','root', '');

if (!$dbcon){

        die('Connection Error' .mysql_error());}

?>
```

6.5.2. Selecting the Database:

```
<?php

$dbselect = mysql_select_db('tv', $dbcon);

if(!$dbselect){

        die('Cant connect: ' .mysql_error());


?>
```

6.5.3. Disconnecting from the Database:

```
<?php

mysql_close($dbcon);


?>
```

VERIFICATION AND VALIDATION WITH TEST CASES

In software project management the software validation and testing is very important. It is the process of checking that the software meets the specifications and fulfils the intended purpose. It may be referred as the software quality control. It is normally responsible for the software quality control. It is normally responsible for the software testing and the SDLC cycle(software development lifecycle.

7.1. UNIT TESTING

Unit testing is undertaken when a module has been created and successfully reviewed. In order to create and test a single module, we need to provide a complete environment . That is besides the module that we would require.

- The procedures belong to the other modules that are under test calls
- Non local data structures and the module access.
- A procedure to call the function of the module under test with appropriate parameters. The unit testing was done each and every module that is described under the module description.

7.1.1. Test for Login:

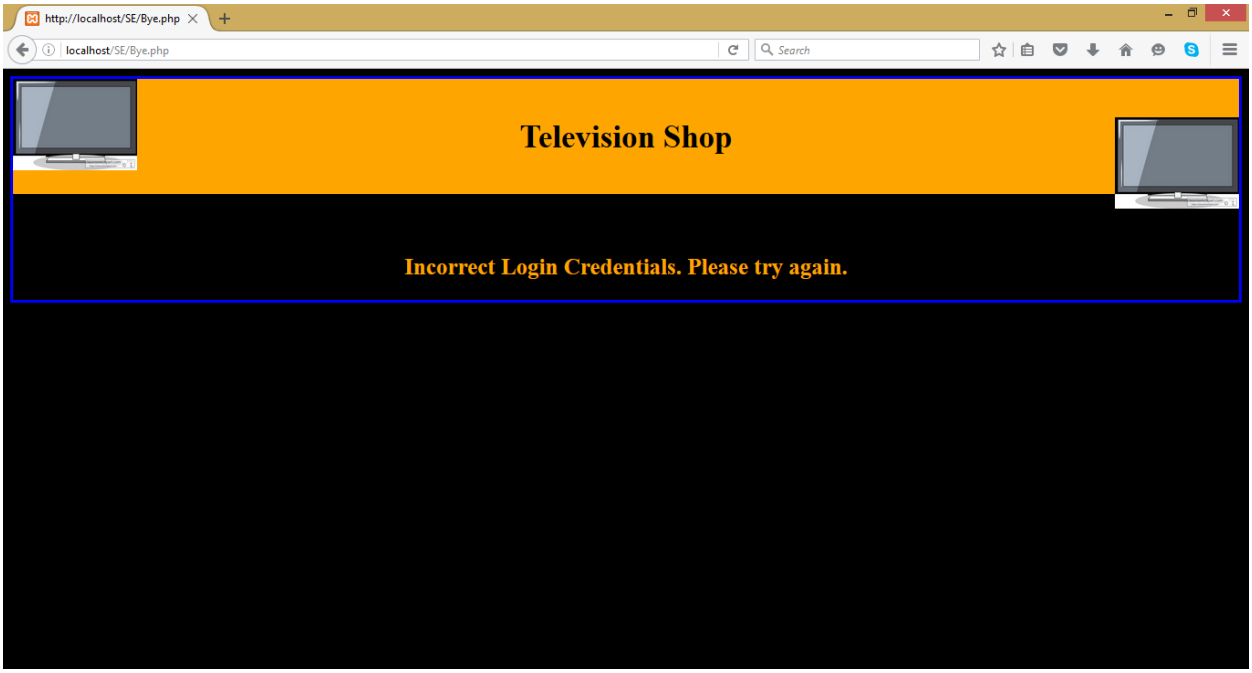
The form is used for the log in of the page is accessible only for the particular customer. Designed only for the particular customer. Login is very safe and secure that can only be accessed by a particular person.

Test case1: To open the software connect apache and MySQL from XAMPP and open web browser and type: localhost\SE

If the username and the password is wrong then it cannot be accessed. It will be redirected to a different page displaying a error message " Invalid Login credentials. Please try again".

Test case2:

Wrong: Directed to another page:



Test case3:

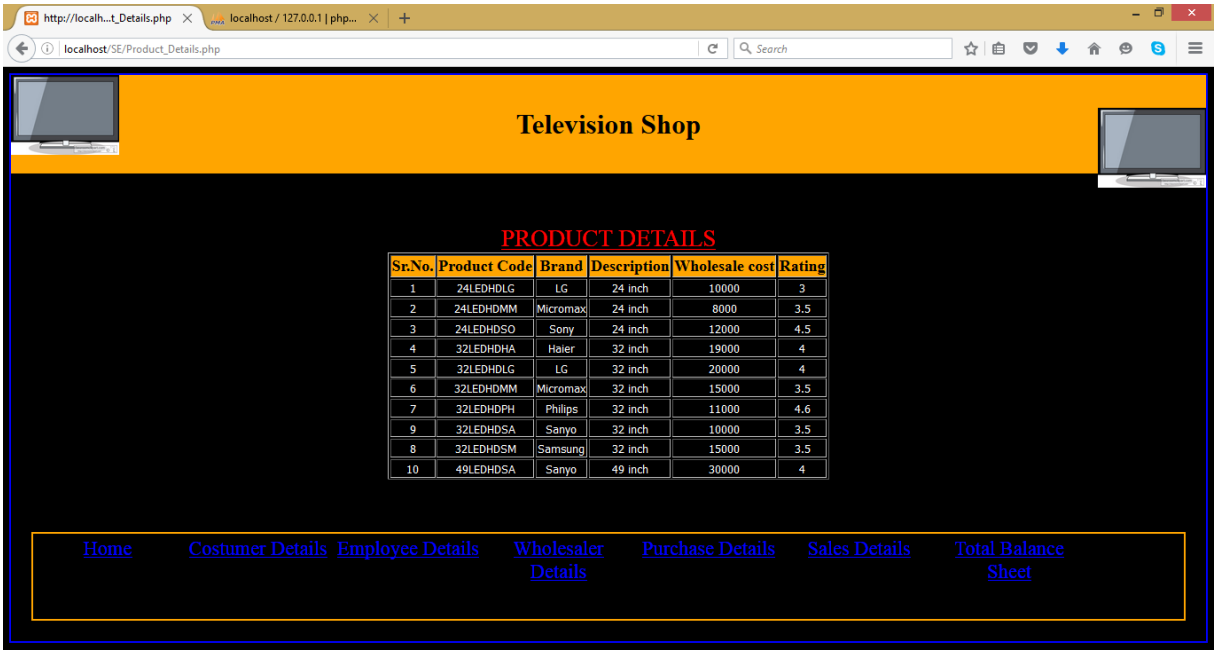
Correct Login input: Username: Admin and Password: tv: Goes to query page.

7.2. INTEGRATION TESTING

The process of the integration testing is to verify that to a very functional performance, and reliability requirements paced in major design items
In this type of testing we test various integration of the project module by providing the input. The primary objective is to test the module interfaces in order to ensure that no errors are occurring when one module invokes the other module. Like from the account.

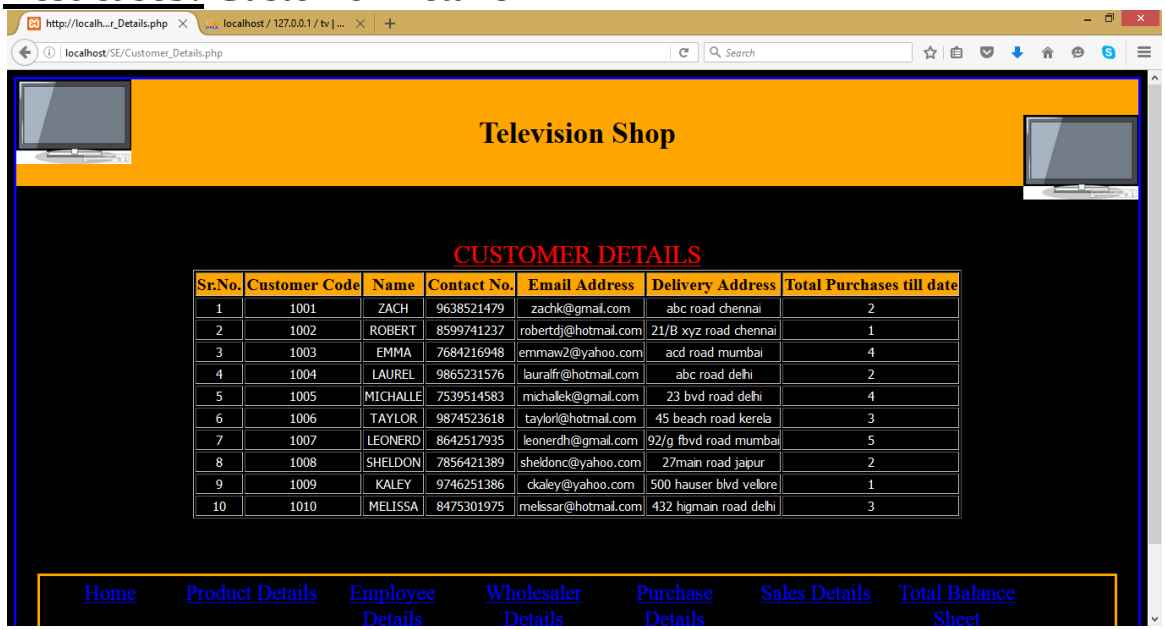
7.2.1. Test for View of Modules:

Test case4: Product Details



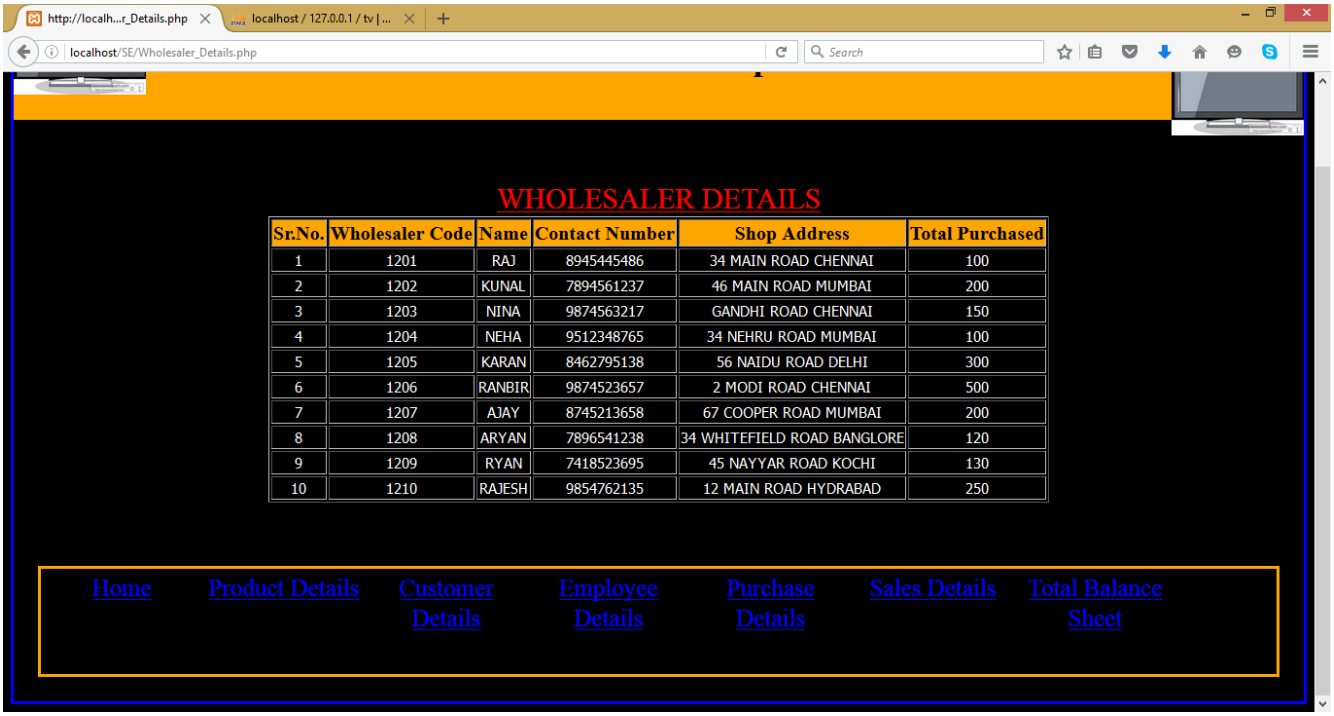
Sr.No.	Product Code	Brand	Description	Wholesale cost	Rating
1	24LEDHDLG	LG	24 inch	10000	3
2	24LEDHDMM	Micromax	24 inch	8000	3.5
3	24LEDHDSO	Sony	24 inch	12000	4.5
4	32LEDHDHA	Haier	32 inch	19000	4
5	32LEDHDLG	LG	32 inch	20000	4
6	32LEDHDMM	Micromax	32 inch	15000	3.5
7	32LEDHDPH	Philips	32 inch	11000	4.6
9	32LEDHDSA	Sanyo	32 inch	10000	3.5
8	32LEDHDSM	Samsung	32 inch	15000	3.5
10	49LEDHDSA	Sanyo	49 inch	30000	4

Test case5: Customer Details



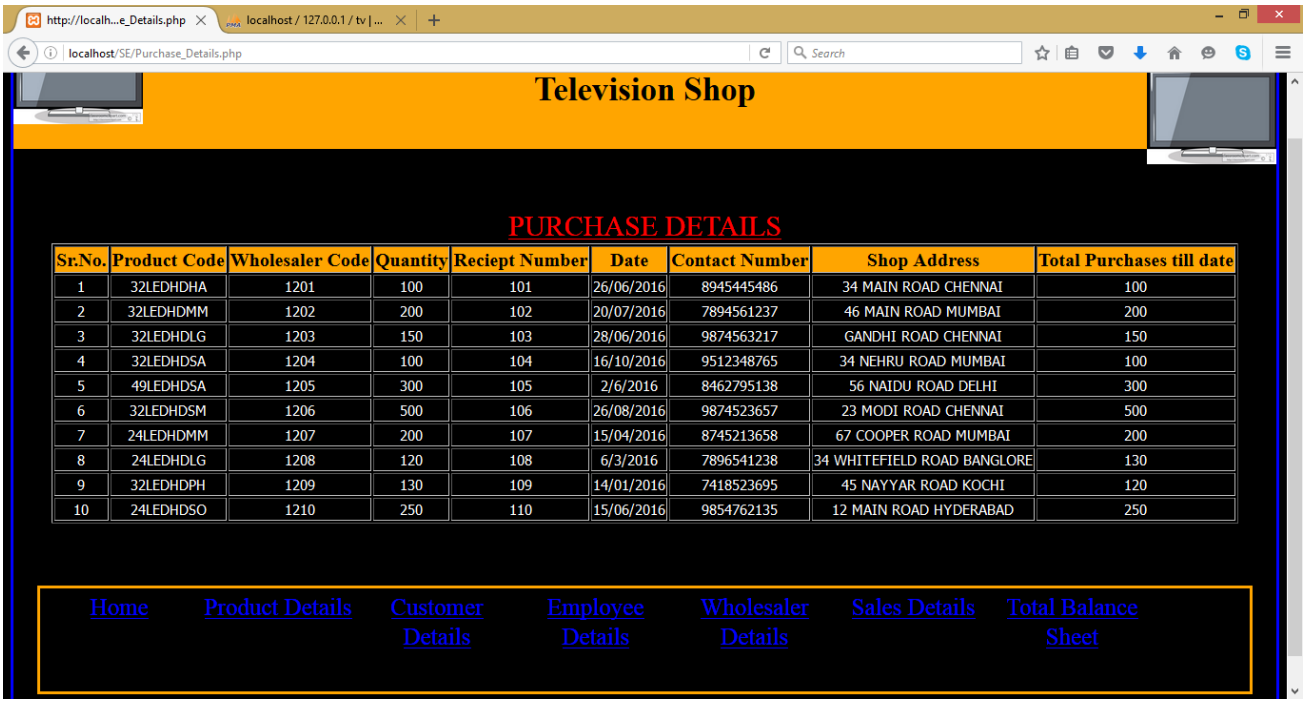
Sr.No.	Customer Code	Name	Contact No.	Email Address	Delivery Address	Total Purchases till date
1	1001	ZACH	9638521479	zachk@gmail.com	abc road chennai	2
2	1002	ROBERT	8599741237	robertdj@hotmail.com	21/B xyz road chennai	1
3	1003	EMMA	7684216948	emmaw2@yahoo.com	acd road mumbai	4
4	1004	LAUREL	9865231576	lauralfr@hotmail.com	abc road delhi	2
5	1005	MICHALLE	7539514583	michallek@gmail.com	23 bvd road delhi	4
6	1006	TAYLOR	9874523618	taylorl@hotmail.com	45 beach road kerela	3
7	1007	LEONERD	8642517935	leonerdh@gmail.com	92/g fbvd road mumbai	5
8	1008	SHELDON	7856421389	sheldonc@yahoo.com	27main road jaipur	2
9	1009	KALEY	9746251386	ckaley@yahoo.com	500 hauser blvd vellore	1
10	1010	MELISSA	8475301975	melissar@hotmail.com	432 higmain road delhi	3

Test case6: Wholesaler Details



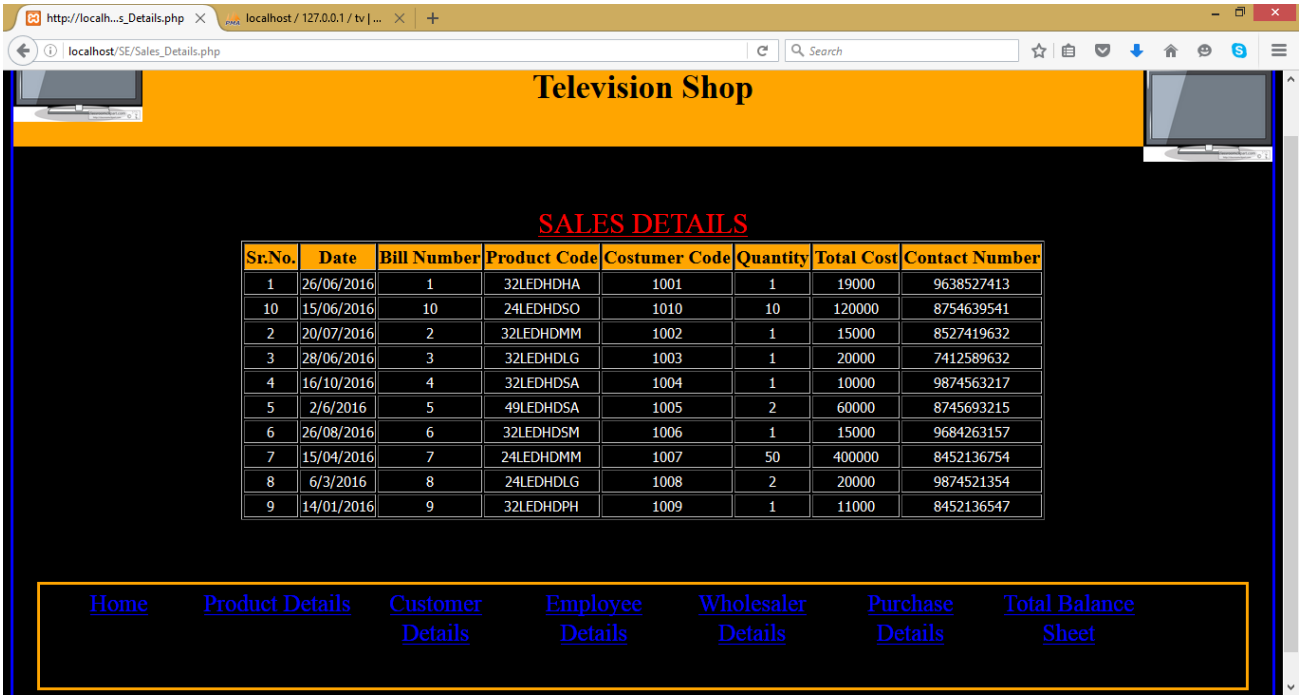
Sr.No.	Wholesaler Code	Name	Contact Number	Shop Address	Total Purchased
1	1201	RAJ	8945445486	34 MAIN ROAD CHENNAI	100
2	1202	KUNAL	7894561237	46 MAIN ROAD MUMBAI	200
3	1203	NINA	9874563217	GANDHI ROAD CHENNAI	150
4	1204	NEHA	9512348765	34 NEHRU ROAD MUMBAI	100
5	1205	KARAN	8462795138	56 NAIDU ROAD DELHI	300
6	1206	RANBIR	9874523657	2 MODI ROAD CHENNAI	500
7	1207	AJAY	8745213658	67 COOPER ROAD MUMBAI	200
8	1208	ARYAN	7896541238	34 WHITEFIELD ROAD BANGLORE	120
9	1209	RYAN	7418523695	45 NAYYAR ROAD KOCHI	130
10	1210	RAJESH	9854762135	12 MAIN ROAD HYDRABAD	250

Test case7: Purchase Details



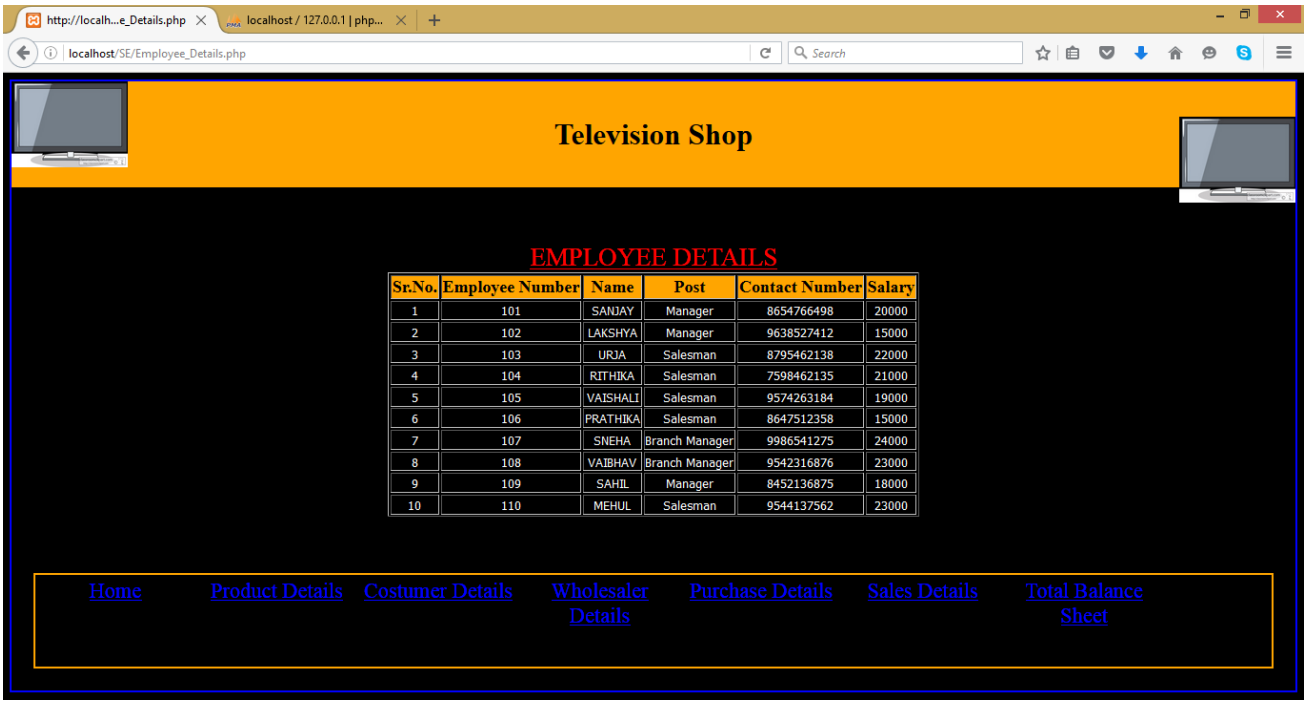
Sr.No.	Product Code	Wholesaler Code	Quantity	Reciept Number	Date	Contact Number	Shop Address	Total Purchases till date
1	32LEDHDHA	1201	100	101	26/06/2016	8945445486	34 MAIN ROAD CHENNAI	100
2	32LEDHDMM	1202	200	102	20/07/2016	7894561237	46 MAIN ROAD MUMBAI	200
3	32LEDHDLG	1203	150	103	28/06/2016	9874563217	GANDHI ROAD CHENNAI	150
4	32LEDHDSA	1204	100	104	16/10/2016	9512348765	34 NEHRU ROAD MUMBAI	100
5	49LEDHDSA	1205	300	105	2/6/2016	8462795138	56 NAIDU ROAD DELHI	300
6	32LEDHDSM	1206	500	106	26/08/2016	9874523657	23 MODI ROAD CHENNAI	500
7	24LEDHDMM	1207	200	107	15/04/2016	8745213658	67 COOPER ROAD MUMBAI	200
8	24LEDHDLG	1208	120	108	6/3/2016	7896541238	34 WHITEFIELD ROAD BANGLORE	130
9	32LEDHDPH	1209	130	109	14/01/2016	7418523695	45 NAYYAR ROAD KOCHI	120
10	24LEDHDSO	1210	250	110	15/06/2016	9854762135	12 MAIN ROAD HYDERABAD	250

Test case9: Sales Details



Sr.No.	Date	Bill Number	Product Code	Costumer Code	Quantity	Total Cost	Contact Number
1	26/06/2016	1	32LEDHDHA	1001	1	19000	9638527413
10	15/06/2016	10	24LEDHDSO	1010	10	120000	8754639541
2	20/07/2016	2	32LEDHDMM	1002	1	15000	8527419632
3	28/06/2016	3	32LEDHDLG	1003	1	20000	7412589632
4	16/10/2016	4	32LEDHDSA	1004	1	10000	9874563217
5	2/6/2016	5	49LEDHDSA	1005	2	60000	8745693215
6	26/08/2016	6	32LEDHDSM	1006	1	15000	9684263157
7	15/04/2016	7	24LEDHDMM	1007	50	400000	8452136754
8	6/3/2016	8	24LEDHDLG	1008	2	20000	9874521354
9	14/01/2016	9	32LEDHDPH	1009	1	11000	8452136547

Test case10: Employee Details



Sr.No.	Employee Number	Name	Post	Contact Number	Salary
1	101	SANJAY	Manager	8654766498	20000
2	102	LAKSHYA	Manager	9638527412	15000
3	103	URJA	Salesman	8795462138	22000
4	104	RITHIKA	Salesman	7598462135	21000
5	105	VAISHALI	Salesman	9574263184	19000
6	106	PRATHIKA	Salesman	8647512358	15000
7	107	SNEHA	Branch Manager	9986541275	24000
8	108	VAIBHAV	Branch Manager	9542316876	23000
9	109	SAHIL	Manager	8452136875	18000
10	110	MEHUL	Salesman	9544137562	23000

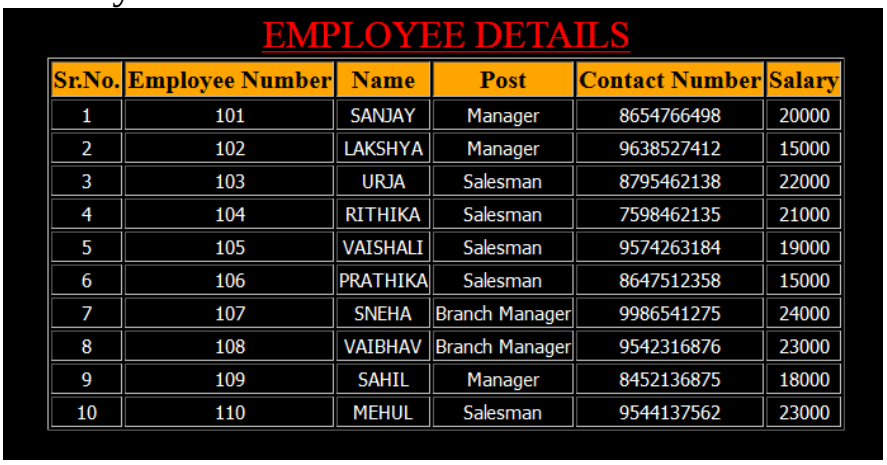
Test case11: Total Balance Sheet

Sr.No.	Date	Transaction Number	Details	Amount
1	26/06/2016	100001	32LEDHDHA	50000
2	20/07/2016	100002	32LEDHDMM	-65000
3	28/06/2016	100003	32LEDHDLG	60000
4	16/10/2016	100004	32LEDHDSA	70000
5	2/6/2016	100005	49LEDHDSA	-45000
6	26/08/2016	100006	32LEDHDSM	-60000
7	15/04/2016	100007	24LEDHDMM	25000
8	6/3/2016	100008	24LEDHDLG	-35000
9	14/01/2016	100009	32LEDHDPH	50000
10	15/06/2016	100010	24LEDHDSO	-37000

7.2.3. Editing of Modules:

Test case11: Updating data

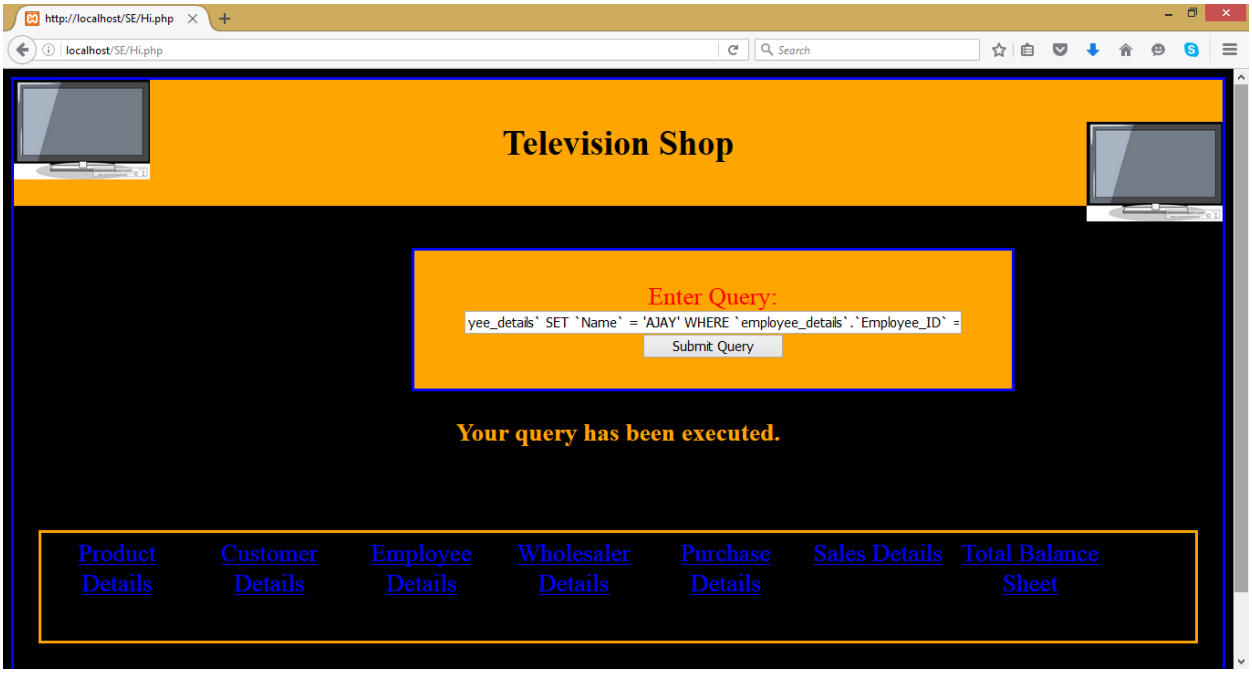
Initially:



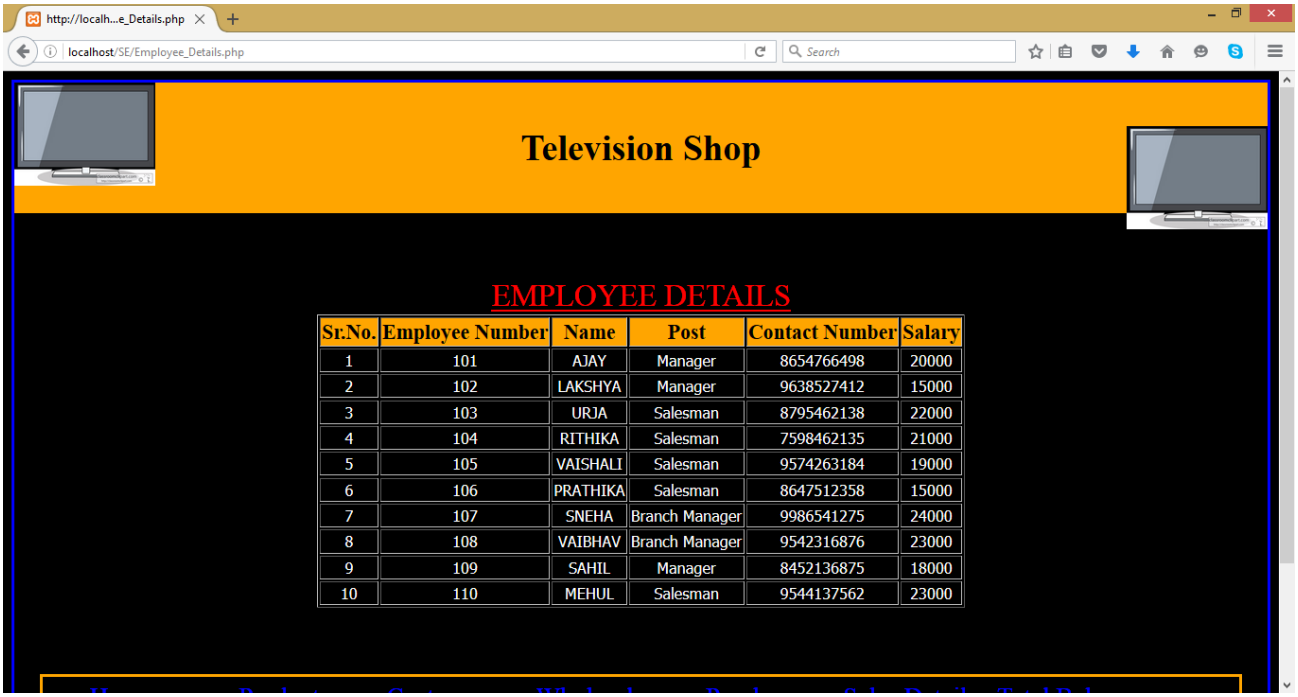
EMPLOYEE DETAILS

Sr.No.	Employee Number	Name	Post	Contact Number	Salary
1	101	SANJAY	Manager	8654766498	20000
2	102	LAKSHYA	Manager	9638527412	15000
3	103	URJA	Salesman	8795462138	22000
4	104	RITHIKA	Salesman	7598462135	21000
5	105	VAISHALI	Salesman	9574263184	19000
6	106	PRATHIKA	Salesman	8647512358	15000
7	107	SNEHA	Branch Manager	9986541275	24000
8	108	VAIBHAV	Branch Manager	9542316876	23000
9	109	SAHIL	Manager	8452136875	18000
10	110	MEHUL	Salesman	9544137562	23000

Input: (Query to update name of 1st employee)
UPDATE `employee_details` SET `Name` = 'AJAY' WHERE `employee_details`.`Employee_ID` = '101'



Output:



Sr.No.	Employee Number	Name	Post	Contact Number	Salary
1	101	AJAY	Manager	8654766498	20000
2	102	LAKSHYA	Manager	9638527412	15000
3	103	URJA	Salesman	8795462138	22000
4	104	RITHIKA	Salesman	7598462135	21000
5	105	VAISHALI	Salesman	9574263184	19000
6	106	PRATHIKA	Salesman	8647512358	15000
7	107	SNEHA	Branch Manager	9986541275	24000
8	108	VAIBHAV	Branch Manager	9542316876	23000
9	109	SAHIL	Manager	8452136875	18000
10	110	MEHUL	Salesman	9544137562	23000

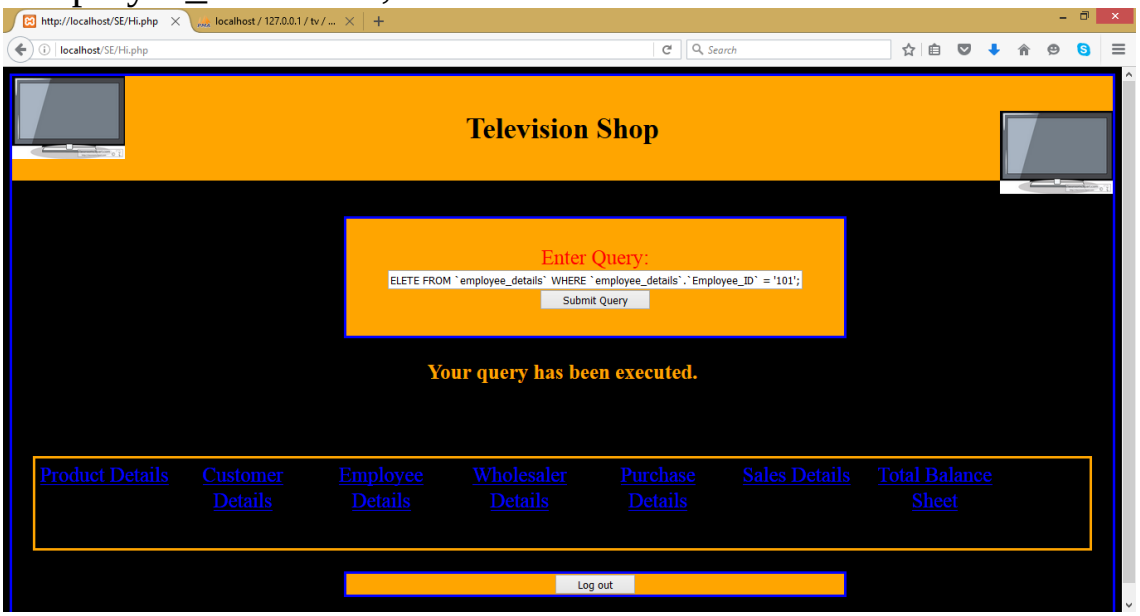
Test case12: Deleting data

Initially:

EMPLOYEE DETAILS

Sr.No.	Employee Number	Name	Post	Contact Number	Salary
1	101	SANJAY	Manager	8654766498	20000
2	102	LAKSHYA	Manager	9638527412	15000
3	103	URJA	Salesman	8795462138	22000
4	104	RITHIKA	Salesman	7598462135	21000
5	105	VAISHALI	Salesman	9574263184	19000
6	106	PRATHIKA	Salesman	8647512358	15000
7	107	SNEHA	Branch Manager	9986541275	24000
8	108	VAIBHAV	Branch Manager	9542316876	23000
9	109	SAHIL	Manager	8452136875	18000
10	110	MEHUL	Salesman	9544137562	23000

Input: (Query to remove 1st employee)
Delete from `employee_details` where `employee_details`.
`Employee_ID`= '101';



Output:

EMPLOYEE DETAILS

Sr.No.	Employee Number	Name	Post	Contact Number	Salary
2	102	LAKSHYA	Manager	9638527412	15000
3	103	URJA	Salesman	8795462138	22000
4	104	RITHIKA	Salesman	7598462135	21000
5	105	VAISHALI	Salesman	9574263184	19000
6	106	PRATHIKA	Salesman	8647512358	15000
7	107	SNEHA	Branch Manager	9986541275	24000
8	108	VAIBHAV	Branch Manager	9542316876	23000
9	109	SAHIL	Manager	8452136875	18000
10	110	MEHUL	Salesman	9544137562	23000

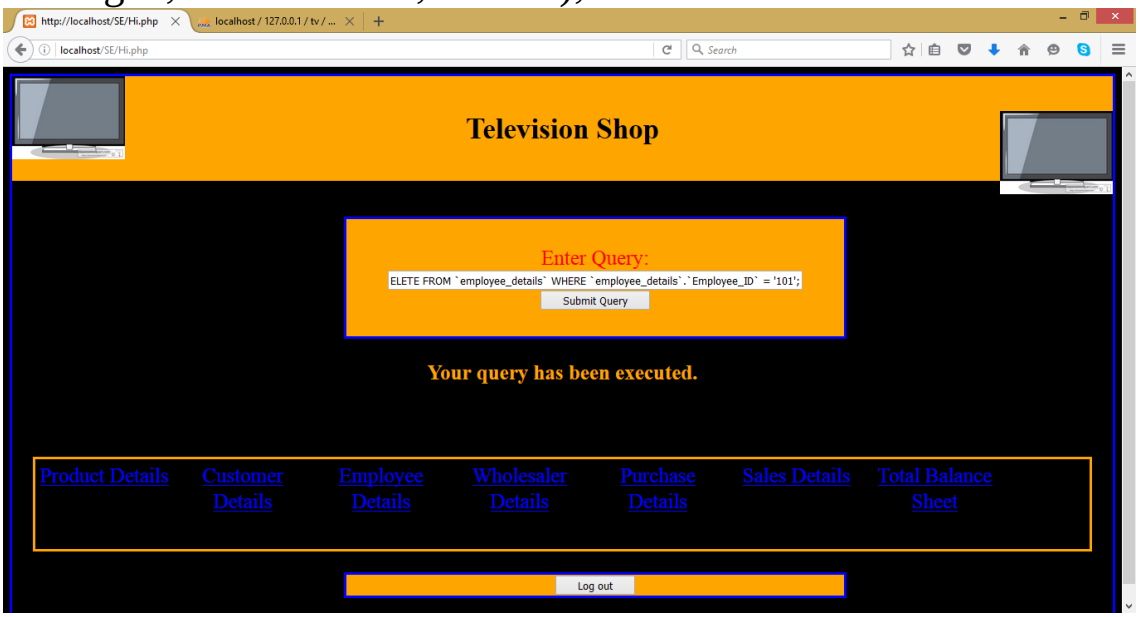
Test case13: Adding data

Initially:

EMPLOYEE DETAILS

Sr.No.	Employee Number	Name	Post	Contact Number	Salary
2	102	LAKSHYA	Manager	9638527412	15000
3	103	URJA	Salesman	8795462138	22000
4	104	RITHIKA	Salesman	7598462135	21000
5	105	VAISHALI	Salesman	9574263184	19000
6	106	PRATHIKA	Salesman	8647512358	15000
7	107	SNEHA	Branch Manager	9986541275	24000
8	108	VAIBHAV	Branch Manager	9542316876	23000
9	109	SAHIL	Manager	8452136875	18000
10	110	MEHUL	Salesman	9544137562	23000

Input: (Query to adding an employee)
INSERT INTO `employee_details` VALUES ('1', '101', 'AJAY', 'Manager', '8654766498', '20000');



Output:

EMPLOYEE DETAILS

Sr.No.	Employee Number	Name	Post	Contact Number	Salary
1	101	AJAY	Manager	8654766498	20000
2	102	LAKSHYA	Manager	9638527412	15000
3	103	URJA	Salesman	8795462138	22000
4	104	RITHIKA	Salesman	7598462135	21000
5	105	VAISHALI	Salesman	9574263184	19000
6	106	PRATHIKA	Salesman	8647512358	15000
7	107	SNEHA	Branch Manager	9986541275	24000
8	108	VAIBHAV	Branch Manager	9542316876	23000
9	109	SAHIL	Manager	8452136875	18000
10	110	MEHUL	Salesman	9544137562	23000

7.2.4. Logging out:

Test case14: Input: Press Log out button.
Output:

IMPLEMENTATION AND MAINTENANCE

8.1. IMPLEMENTATION

To implement this software the **requirements** to run it are as follows-

Operating System- Windows XP or higher, Ubuntu Linux 11.04 or similar, iOS X or similar.

Processor- minimum 1.3-1.5 GHz quad or octa core is required as lower than this will not work fast enough while translating web pages and data abstraction.

XAMPP version 3.2.2 or WAMP server or a similar kind of server that should have

a. php 5.5.0
b. MySQL 5.5 or newer
c. Apache support
d. phpmyadmin version 1.7 or higher.

Web Browser like Mozilla Firefox or Internet Explorer or Google Chrome for running html, php.

How to start the software-
To start the software, just start your server with XAMPP or WAMP and ready your database in phpmyadmin.
NOTE- All your files should be in server location so that they can be manipulated.
Now open your web browser and type- “localhost/SE/index.php”. This will open our website’s home page.

8.2. MAINTAINENCE

Any Software undergoes four types of maintenance after it has been developed and tested.

1. CORRECTIVE MAINTENANCE-

Reactive modification of our product after delivery to any institute to CORRECT discovered problems. Taking existing code and Correcting a fault that causes the code to behave in some way that deviates from its documented requirements.

Focuses on bug fixing and reporting errors fixing

.• i.e. defects generally need to be Corrected either immediately

• or in the near future. Fixing a fault has 20 to 50 % chances of introducing another

• fault System- response to malfunction

2. ADAPTIVE MAINTENANCE-

Modification of software product performed after delivery to keep a software product usable in a changed or changing environment.

3. PREVENTIVE MAINTENANCE-

Modification of software product after delivery to detect and CORRECT latent faults in the software before they become effective faults.

4. PERFECTIVE MAINTENANCE-

Modification of software product after delivery to improve performance or maintainability.

CONCLUSION

This software enables the retail store owner and the concerned staff personnel to to maintain documentation about sensitive information like their business accounts providing authentication.

Thus, the project for “Business Accounts of a Television Shop” with oracle MySQL database was developed and executed successfully.

FUTURE SCOPE

For future scope on enhancing the software, we now have come up with few functions that can further be added. Also, styling can be notched up to appeal to a greater number of users.

In terms of the software, considering the current scenario of expanding market for televisions and the extensive number of stores opening up, this software has a good scope of being widely used. This is because it provides the retailer with the ability of storing any data that he would like to and also displays it properly. Moreover, it is highly safe and secure in terms of the access to the documents, since such information is sensitive to the firm.

REFERENCES

Books:

1) Bob Bryla and Kevin Loney, Oracle Database 12c The complete Reference, Tata McGraw Hill, 1 s t edition, 2013 .

2) Jon Duckett, HTML & CSS Design and Build Websites, Wiley, 2011

3) Ian Sommerville, Software Engineering, Ninth Edition, Pearson, 2011.

4) Web development and application development by Ivan Byross BPB publications

Websites:

1. Kaner, Cem (November 17, 2006). "Exploratory Testing" (PDF). Florida Institute of Technology, Quality Assurance Institute Worldwide Annual Software Testing Conference, Orlando, FL. Retrieved November 22, 2014.

2. Software Testing by Jiantao Pan, Carnegie Mellon University

3. Leitner, A., Ciupa, I., Oriol, M., Meyer, B., Fiva, A., "Contract Driven Development = Test Driven Development – Writing Test Cases", Proceedings of ESEC/FSE'07: European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering 2007, (Dubrovnik, Croatia), September 2007

4. Kolawa, Adam; Huizinga, Dorota (2007). Automated Defect Prevention: Best Practices in Software Management. Wiley-

5. Kolawa, Adam; Huizinga, Dorota (2007). Automated Defect Prevention: Best Practices in Software Management. Wiley-IEEE Computer Society Press. p. 426. ISBN 0-470-04212-5

6. Coding: **www.w3schools.com**
The New Boston tutorial videos.

7. IEEE (1998). IEEE standard for software test documentation. New York: IEEE. ISBN 0-7381-1443-X