MIT Art Design and Technology University

MIT School of Computing, Pune

Department of Computer Science and Engineering

Third Year B. Tech Academic Year 2022-2023. (SEM-I)

Subject: Web Technology Laboratory

Assignment 8

Assignment Title: Design a shopping application form with following fields[itemID, itemName, itemQuantity] Write a PHP script to add and display the items..

Aim: Create a shopping cart web page

Objectives:

1. To understand how PHP can be used to interact with MariaDB / MySQL
2. Understand the syntax of PHP

Outcomes:

Upon Completion of the assignment the students will be able to

1. Create Connection to MySQL with PHP to access data
2. Understand relationship between HTML and PHP

Name: Rudradev Arya
Branch and Division: CSE TY-CC-B
Roll Number: 2213825
Enrollment Number: MITU21BTCS0489
Course Name: Web Technology Lab

PHP :

```php
<?php

$conn = mysqli_connect('localhost', `username',
`password', `DBname');

if (!$conn)

{

 die("Connection failed: " . mysqli_connect_error());

}

if ($_SERVER["REQUEST_METHOD"] == "POST")

{

 // Get form data

 $itemID = $_POST["itemID"];

 $itemName = $_POST["itemName"];

 $itemQuantity = $_POST["itemQuantity"];
```

```
 // Add item to database

 $sql = "INSERT INTO shopping(itemID, itemName,
itemQuantity)

 VALUES ('$itemID', '$itemName', '$itemQuantity')";

 if (mysqli_query($conn, $sql))

 {

 echo "Item added successfully Kindly Refresh Database
Page";

 }

 else

 {

 echo "Error adding item: Check Your Connection" .
mysqli_error($conn);

 }

}

?>

<html>

<head>
```

```
<title>Shopping Cart</title>

<style>

body

{

background-color: #f2f2f2;

font-family: Arial, sans-serif;

}

h2

{

color: #333333;

}

form

{

background-color: #ffffff;

padding: 20px;

margin-bottom: 20px;

border-radius: 5px;
```

```
}

table

{

border-collapse: collapse;

margin-bottom: 20px;

}

th,

td {

padding: 8px;

border: 1px solid #dddddd;

text-align: left;

}

th {

background-color: #4CAF50;

color: #ffffff;

}

tr:nth-child(even) {
```

```
background-color: #f2f2f2;

}

input[type="text"],

input[type="number"] {

padding: 8px;

border-radius: 5px;

border: 1px solid #cccccc;

width: 100%;

}

input[type="submit"] {

background-color: #4CAF50;

color: #ffffff;

border-radius: 5px;

border: none;

padding: 8px 16px;

margin-top: 10px;

cursor: pointer;
```

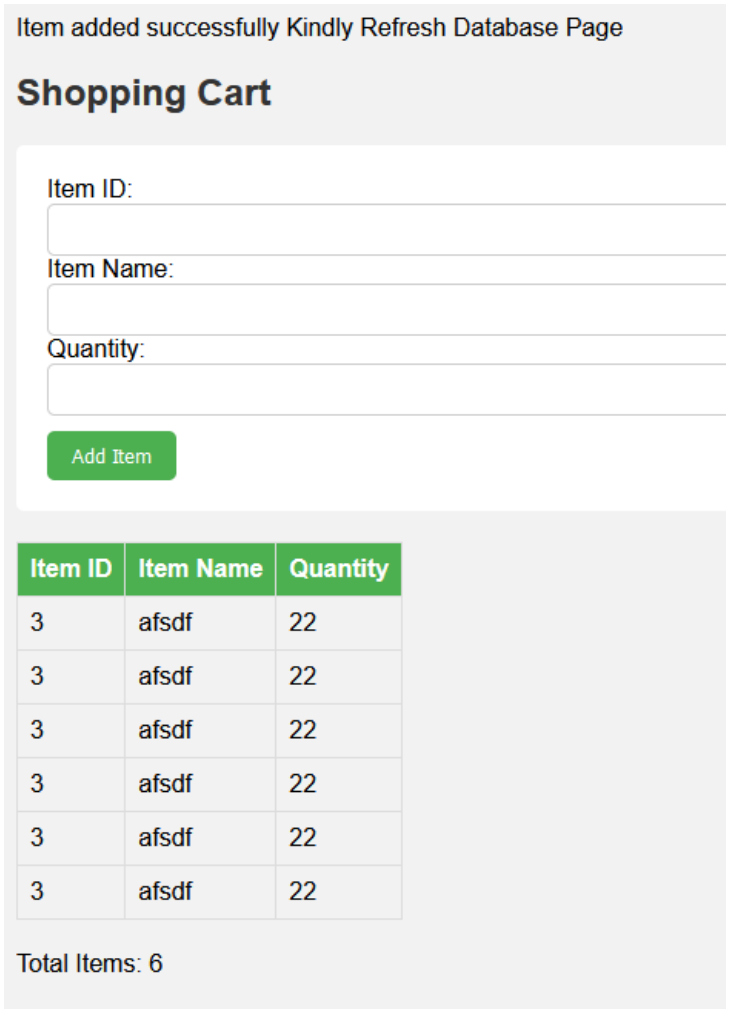
```
 }
 input[type="submit"]:hover {
 background-color: #3e8e41;
 }
</style>
</head>
<body>
 <h2>Shopping Cart</h2>
 <form method="post">
 <label for="itemID">Item ID:</label>
 <input type="text" name="itemID" id="itemID">
 <label for="itemName">Item Name:</label>
 <input type="text" name="itemName" id="itemName">
 <label for="itemQuantity">Quantity:</label>
 <input type="number" name="itemQuantity"
id="itemQuantity">
 <input type="submit" value="Add Item">
```

```
</form>

<table>

<thead>

<tr>

<th>Item ID</th>

<th>Item Name</th>

<th>Quantity</th>

</tr>

</thead>

<tbody>

<?php

$sql = "SELECT * FROM shopping";

$result = mysqli_query($conn, $sql);

if (mysqli_num_rows($result) > 0) {

while ($row = mysqli_fetch_assoc($result)) {

echo "<tr>";

echo "<td>" . $row["itemID"] . "</td>";
```

```
echo "<td>" . $row["itemName"] . "</td>";

echo "<td>" . $row["itemQuantity"] . "</td>";

echo "</tr>";

}

}

?>

</tbody>

</table>

<?php

$sql = "SELECT COUNT(*) as totalCount FROM shopping";

$result = mysqli_query($conn, $sql);

$row = mysqli_fetch_assoc($result);

echo "<p>Total Items: " . $row["totalCount"] . "</p>";

?>

</body>

</html>

<?php
```

```php
mysqli_close($conn);

?>
```

Output :



Explanation :

1. **HTML Form Creation**: The code demonstrates how to create a form in HTML with fields for item ID, item name, and item quantity. Each field is created using the `<input>` tag with appropriate 'type' and 'name' attributes.
2. **PHP and MySQL Integration**: The PHP script at the top of the file shows how to establish a connection to a MySQL database using the `mysqli_connect` function. It also shows how to handle form submissions using the `$_POST` superglobal array and insert data into the database using an SQL INSERT statement.
3. **Error Handling**: The code includes basic error handling for the database connection process. If the connection fails, the `die` function is called with an error message.
4. **Dynamic Content Generation**: The PHP script within the HTML table dynamically generates table rows based on data fetched from the database. This is done using a while loop that iterates over each row of data returned by an SQL SELECT statement.
5. **CSS Styling**: The code includes CSS styles that enhance the appearance of the form and table. This includes styles for form inputs, table cells, and dynamic hover effects for submit buttons.
6. **Server-Side Form Handling**: The PHP script checks if the form has been submitted using a POST request, retrieves the values from the form fields, and uses them to create an SQL INSERT statement.
7. **Closing Database Connection**: At the end of the script, it's important to close the connection to free up resources. This is done using `mysqli_close($conn);`.