1.PROJECT DESCRIPTION

COOLING CORNER ICE-CREAM SHOP

1.1 INTRODUCTION

This project developed for the Ice-cream Shop, to managing works like Ordering the products and maintaining them. The Cooling Corner is a web-based application designed to streamline the management of an ice cream shop. The system offers features such as adding new ice cream products, updating or deleting existing products, and facilitating the ordering process for customers. With a user-friendly interface and efficient functionalities, Cooling Corner aims to enhance productivity and customer satisfaction in an ice cream shop setting. It offers a range of functionalities to streamline operations and enhance customer experience. Shop owners can easily add new ice cream products, update their details, or remove them when necessary. Customers can browse the available products, add them to their cart, and place orders conveniently. With its intuitive interface and robust features, Cooling Corner aims to optimize efficiency and drive success in the ice cream shop industry.

Front End: PHP

Back End: My Sql

2.REQUIREMENT SPECIFICATION

2.1 HARDWARE REQUIREMENTS

- Processor: AMD pro system
- Processor Speed: 250MHz to 833MHz
- RAM: 512MB RAM
- Hard Disk: 40GB

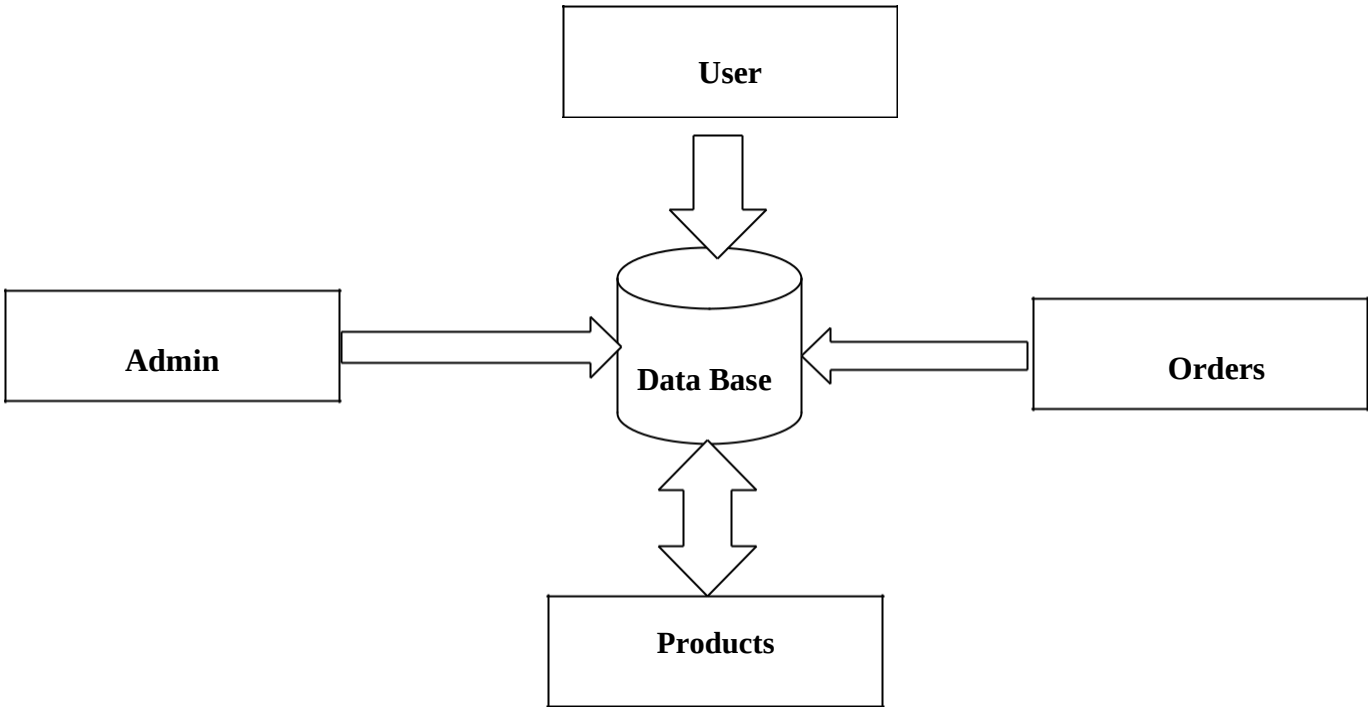
2.2 SOFTWARE REQUIREMENTS:

- Operating System: Windows 10
- Database: My Sql
- Server-side technology: Xampp
- Server-side scripting: PHP
- Client-side scripting: HTML
- Web-Server: Local Host

3.ARCHITECTURE DESIGN

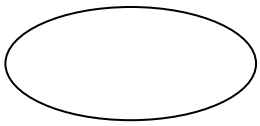
3.1 ARCHITECTURE DIAGRAM

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system. System architecture can comprise system components, the externally visible properties of those components, the relationships (e.g. the behavior) between them. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture; collectively these are called architecture description languages.



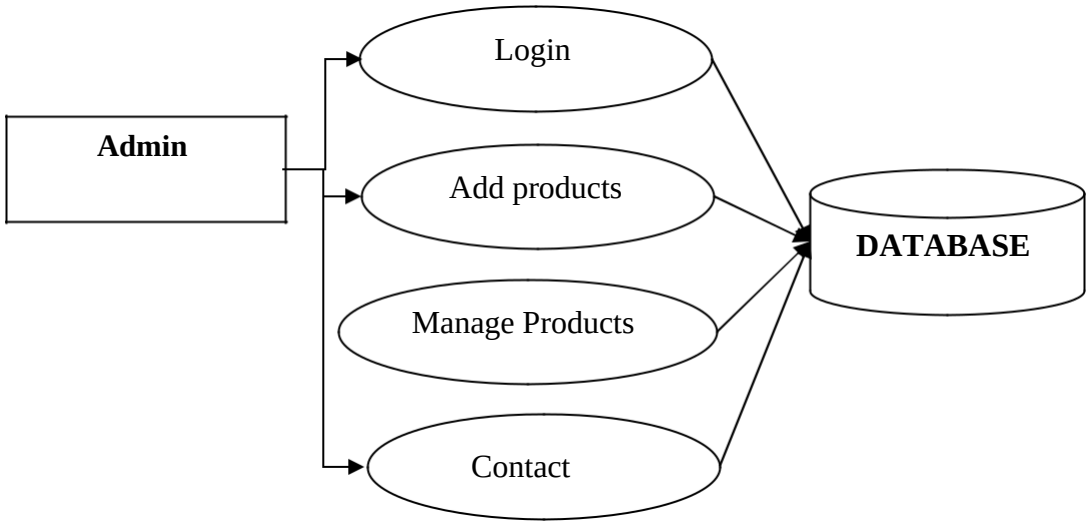
3.2 DATAFLOW DIAGRAM

It is a two-dimensional diagram explains how data is processed and transferred in a system. The graphical depiction identifies each source of data and how it interacts with other data sources to reach a common output. Individuals seeking to draft a data flow diagram must identify external inputs and outputs, determine how the inputs and outputs relate to each other, and explain with graphics how these connections relate and what they result in. This type of diagram helps business development and design teams visualize how data is rocessed and identify or improve certain aspects.

Symbol	**Description**
	An **entity**. A source of data or a destination for data.
	A **process** or task that is performed by the system.
	A **data store**, a place where data is held between processes.
	A **data flow**.

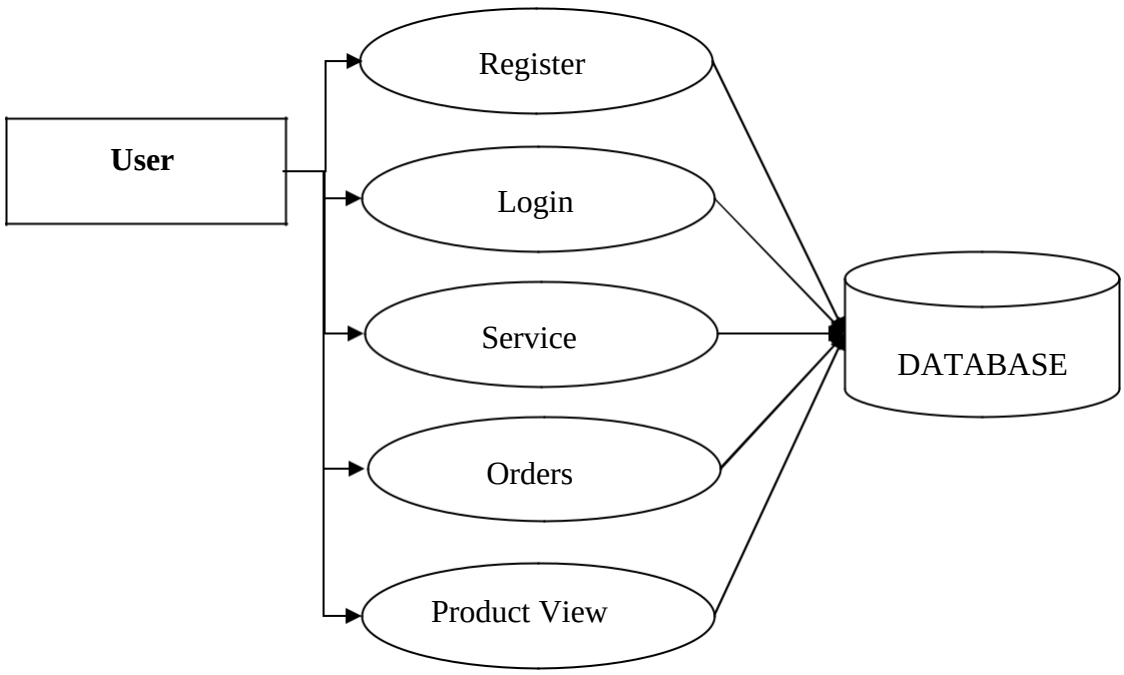
LEVEL 0

It is also known as a context diagram. It's designed to be an abstraction view, showing the system as a single process with its relationship to external entities. It represents the entire system as a single bubble with input and output data indicated by incoming/outgoing arrows.



LEVEL 1

In 1-level DFD, the context diagram is decomposed into multiple bubbles/processes. In this level, we highlight the main functions of the system and breakdown the high-level process of 0-level DFD into sub processes.



4.TABLE DESIGN

4.1 TABLES

A table is a data structure that organizes information into rows and columns. It can be used to both store and display data in a structured format. For example, databases store data in tables so that information can be quickly accessed from specific rows. Websites often use tables to display multiple rows of data on page. Databases often contain multiple tables, with each one designed for a specific purpose. For example, a company database may contain separate tables for employees, clients, and suppliers. Each table may include its own set of fields, based on what data the table needs to store. In database tables, each field is considered a column, while each entry (or record), is considered a row. A specific value can be accessed from the table by requesting data from an individual column and row.

4.1.1 TABLE: USER

Column	Type	Description
id	int(10)	Primary Key
Full name	varchar(50)	User Name
mobile	varchar(10)	Mobile Number
email	varchar(50)	Mail Address
password	varchar(50)	Password

4.1.2 TABLE: PRODUCT

Column	Type	Default
id	int(10)	Primary Key
name	varchar(50)	Product Name
price	varchar(50)	price
image	longblob	Picture

4.1.2 TABLE: CART

Column	Type	Default
pid	int(10)	Primary Key
uid	varchar(50)	Product Name
price	varchar(50)	Price

TABLE NAME: CONTACT

Column Name	Data Type	Default
Id	Int (11)	PRIMARY KEY
Name	Varchar (25)	NULL
Email	Varchar (25)	NULL
Gender	Varchar(25)	NULL
Mobile	Varchar (25)	NULL
Source	Varchar(25)	NULL
Message	Varchar (55)	NULL
City	Varchar (255)	NULL
Created_at	Date(11)	NULL

5.PROGRAM DESIGN

It allow the admin to add the customer details like Name, E-mail, Contact. It also allow the Product details and manage the selling product to the Customer. Sales is used to manage sales items. sales item, customer name. Product details are saved and retrieve report from the database.

MODULES:

5.1 Admin:

- Login
- Add Products
- Manage Product
- Contact

5.1 Admin:

The module allows the admin to add customer details, add new product arrival details and also manage the product details. Manage the selling product details to the customer. Add customer details like Name, Email, Contact details. Those Customer details are stored in the database. Product (gaming -related), product name, product id and product price are stored in the database. In the product details, First select the customer's name which customer want to purchase the product. Admin can see the customer purchase details and he can update and delete the product. These details are stored in the database.

5.1.1 Login:

In the module, Admin can login into the system using her/his username and password. Username and password generated in the database as a admin module. After login the page, next dashboard page will open. Now admin can manage the users and orders. create category(sub-categories), Insert the new product details and manage that product.

Code: LOGIN

```
<?php
session_start();
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "icecream";  ]
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
   die("Connection failed: " . $conn->connect_error);
if ($_SERVER["REQUEST_METHOD"] == "POST") {
   $fullname = $_POST["fullname"];
   $password = $_POST["password"];
   $adminUsername = "surya317";
   $adminPassword = "surya123";
   if ($fullname === $adminUsername && $password === $adminPassword) {
      echo "Admin login successful!";
      header("location:admin_page.php");
   } else {
      $sql = "SELECT * FROM users WHERE fullname = '$fullname' AND password = '$password'";
      $result = $conn->query($sql);
      if ($result->num_rows == 1) {
         echo "User login successful!";
         $_SESSION['uname']= $fullname;
         header("location:registered-user.php");
      } else
         echo "Invalid username or password!"
$conn->close();
 <input type="text"name="fullname"id="fullname"class="form-control" placeholder="username" required/>
```

```
 <input type="text"name="password" id="pass"class="form-control" placeholder="password" required/>
<button class="btn btn-success" type="submit" name="submit">
            Login
        </button>
         Don't have an account?
         <a href="./register.php" style="color:aliceblue">Register now</a>
```

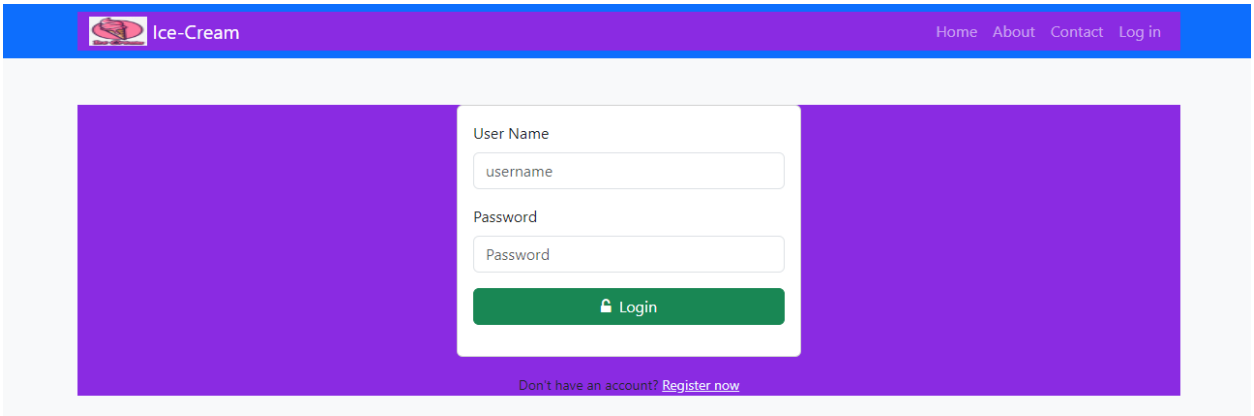


Fig. 5.1.1 Login page

Upon the successful login, the page will be redirected to the Dashboard page of the administrator, where a number of operations to be performed will be present in the panel. In the dashboard page admin can enters the details of product and customer details.

Code : Home Page(User)

```
<!-- carousel starts -->
   <div id="carouselExampleControls" class="carousel faded" data-bs-ride="carousel">
      <div class="carousel-inner">
         <div class="carousel-item carousel-image bg-img-1 active">
```

```
<img class="w-100" src="img/carousel-1.jpg" alt="Image">
<div class="carousel-caption d-flex flex-column align-items-center justify-content-center">
<div class="p-3" style="max-width: 900px;">
<h4 class="text-white text-uppercase mb-md-3">Cooling Corner</h4>
<h1 class="display-3 text-white mb-md-4">Worldwide Flavours</h1>
<a href="register.php" class="btn btn-light py-md-3 px-md-5 mt-2 rounded-pill">Load More</a>
</div>
</div>
```

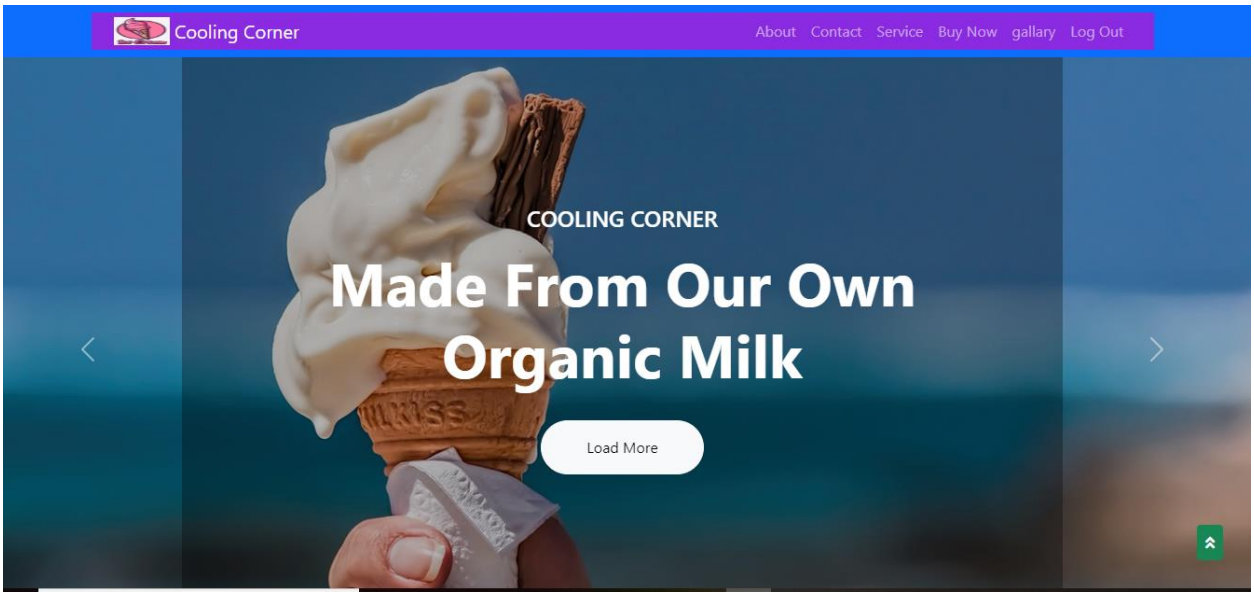


Fig. 5.1.2 Carousel page

```
<!-- Services Start -->

<div class="container-fluid py-5 ">
<div class="container py-5" style="background-color: #eb5fac;">
<div class="container">
<div class="row">
<div class="col-lg-6">
```

<h1 class="section-title position-relative mb-5 text-light">Best Services We Provide For Our Clients</h1>

<!-- Content for service item 1 -->

<img class="rounded-circle w-100 h-100 bg-light p-3" src="img/service-1.jpg" style="object-fit: cover;">

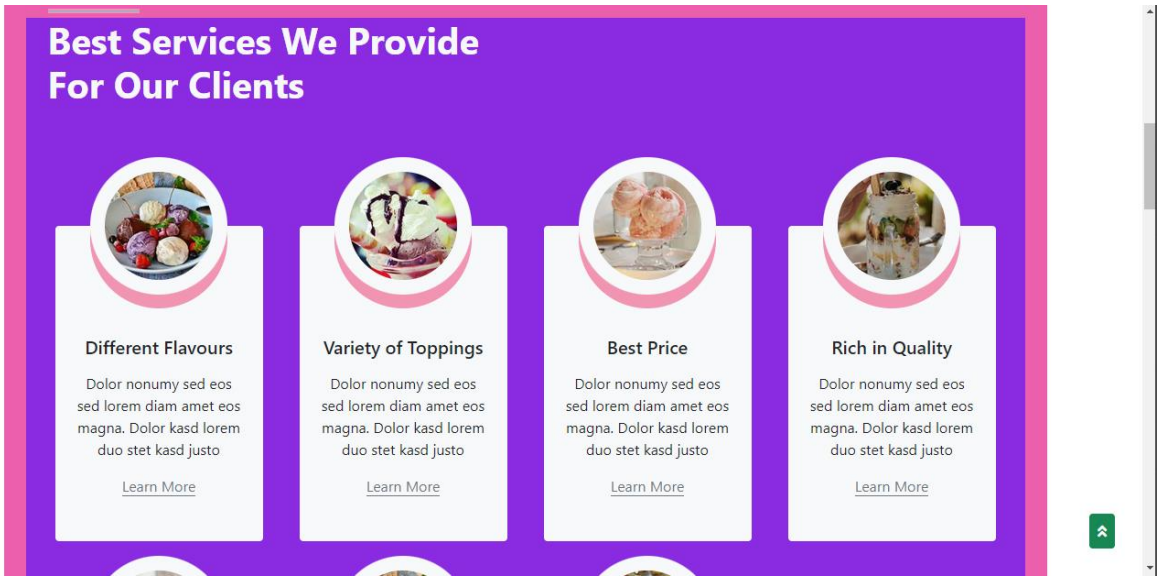


Fig. 5.1.3 Service page

<!-- Gallary Start -->

```
<div class="container-fluid py-5">
<div class="container py-5 ">
  <div class="row justify-content-center m-0">
    <div class="col-lg-5">
      <h1 class="section-title position-relative text-center text-light mb-5">Delicious Ice Cream Made From Our Very Own Organic Milk</h1>
    </div>
  </div>
  <div class="row m-0 portfolio-container ">
    <div class="col-lg-4 col-md-6 p-2 portfolio-item">
      <div class="position-relative overflow-hidden">
        <img class="img-fluid w-100" src="img/portfolio-1.jpg" alt="">
        <a class="portfolio-btn" href="img/portfolio-1.jpg" data-lightbox="portfolio" target="_blank">
          <i class="fa fa-plus text-primary" style="font-size: 60px;"></i></a>
```

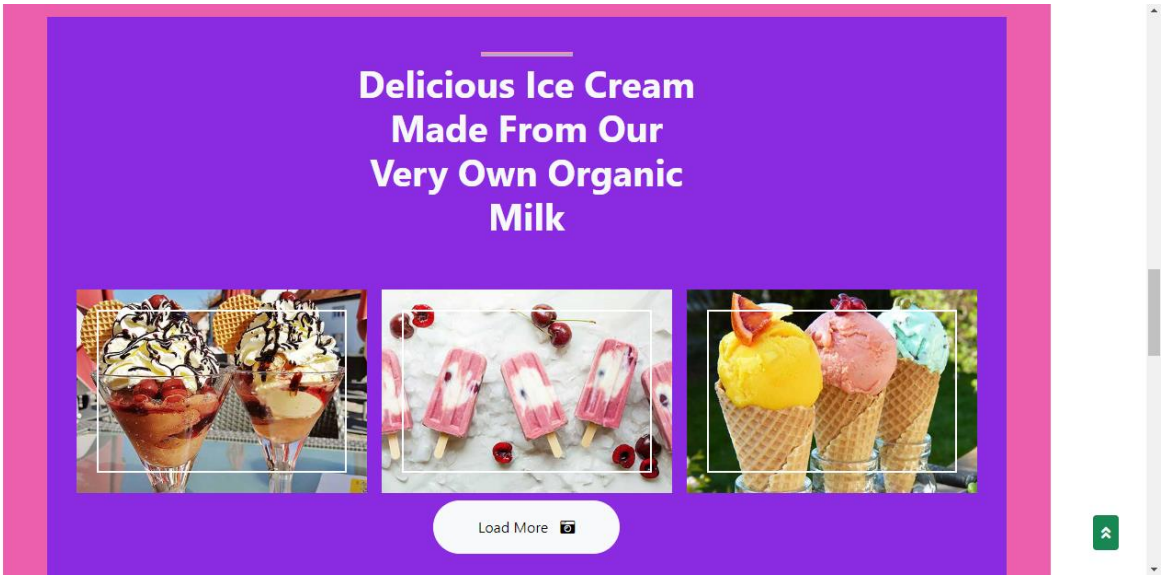


Fig. 5.1.4 Gallary page

5.2 Add Product:

In this module, Admin can add new products and details like product id, product name, product price. These Details are stored in the “Product” database. This module is used to stored product name like Joystick, PlayStation, Game-CD and so on. While click the insert button, the data grid view shows the previously stored data.

Add product:

```
<!DOCTYPE html>
<html>
<head>
  <title>Admin Products</title>
  <linkrel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
</head>
<body>
  <div class="container text-light m-5" style="width:auto; max-width:auto">
  <div class="container m-auto">
    <div class="h2 d-flex justify-content-center">Admin Products</div>

    <!-- Display success or error messages if any -->
    <?php
    if (isset($_GET['message'])) {
      echo '<div class="alert alert-primary">' . $_GET['message'] . '</div>';
    }
    ?>
    <!-- Add New Product Form -->
    <div class="h4 d-flex justify-content-center">Add New Product</div>
    <form method="POST" name="form1" action="" enctype="multipart/form-data">
    <div class="form-row d-flex justify-content-center">
        <div class="col-md-4">

     <div class="form-group">
```

```
    <label for="name">Product ID</label>
    <input type="text" class="form-control" id="id" name="id" required>
  </div>
    </div>
</div>
<div class="form-row d-flex justify-content-center">
    <div class="col-md-4">
  <div class="form-group">
    <label for="name">Product Name</label>
    <input type="text" class="form-control" id="name" name="name" required>
  </div>
    </div>
</div>
<div class="form-row d-flex justify-content-center">
    <div class="col-md-4">
  <div class="form-group">
    <label for="price">Product Price</label>
    <input type="number" class="form-control" id="price" name="price" required>
  </div>
    </div>
</div>
<div class="form-row d-flex justify-content-center">
    <div class="col-md-4">
  <div class="form-group">
    <label for="image">Product Image</label>
    <input type="file" class="form-control-file" id="image" name="image" required>
  </div>
    </div>
</div>
<div class="d-flex justify-content-center">
  <button type="submit" class="btn btn-primary m-3" name="btn">Add Product</button>
```

```
</div>
</form>
</div>
```

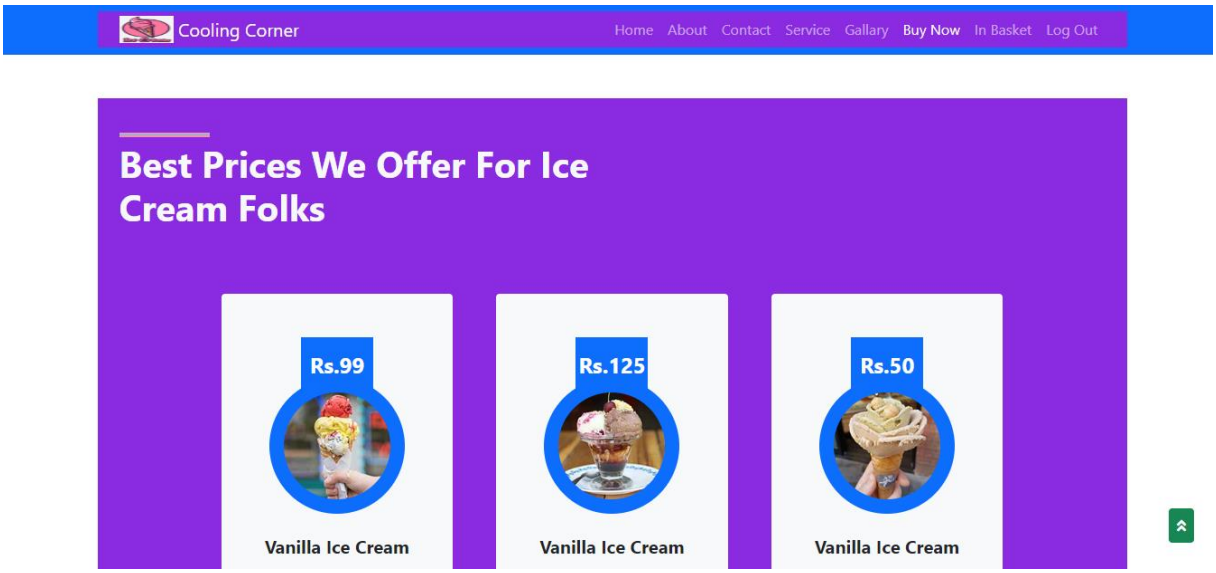


Fig. 5.1.5 Product view

```
<!-- Product List -->
<!-- <h4>Product List</h4> -->
```

```
<?php
$connect = mysqli_connect("localhost", "root", "", "icecream");

if (isset($_POST['btn']))
 {
  // Retrieve form data
 $t1 = isset($_POST['id']) ? $_POST['id'] : '';
 $t2 = isset($_POST['name']) ? $_POST['name'] : '';
 $t3 = isset($_POST['price']) ? $_POST['price'] : '';

  // Handle image upload
  $image = $_FILES['image']['tmp_name']; // Path to the temporary uploaded file
  $imageData = file_get_contents($image); // Retrieve the image data
```

```
 $qry = mysqli_prepare($connect, "INSERT INTO products (id, image, name, price) VALUES (?,
?, ?, ?)");
  mysqli_stmt_bind_param($qry, 'ssss', $t1, $imageData, $t2, $t3);

  // Execute the prepared statement
  mysqli_stmt_execute($qry);

  if (mysqli_stmt_affected_rows($qry) > 0) {

    echo "Data inserted successfully!";
    ?>
    <script language="javascript">alert("Data Inserted Successfully.")</script>
    <?php
  } else {
    echo "Error: " . mysqli_error($connect);
  }

  mysqli_stmt_close($qry);


}
?>
```

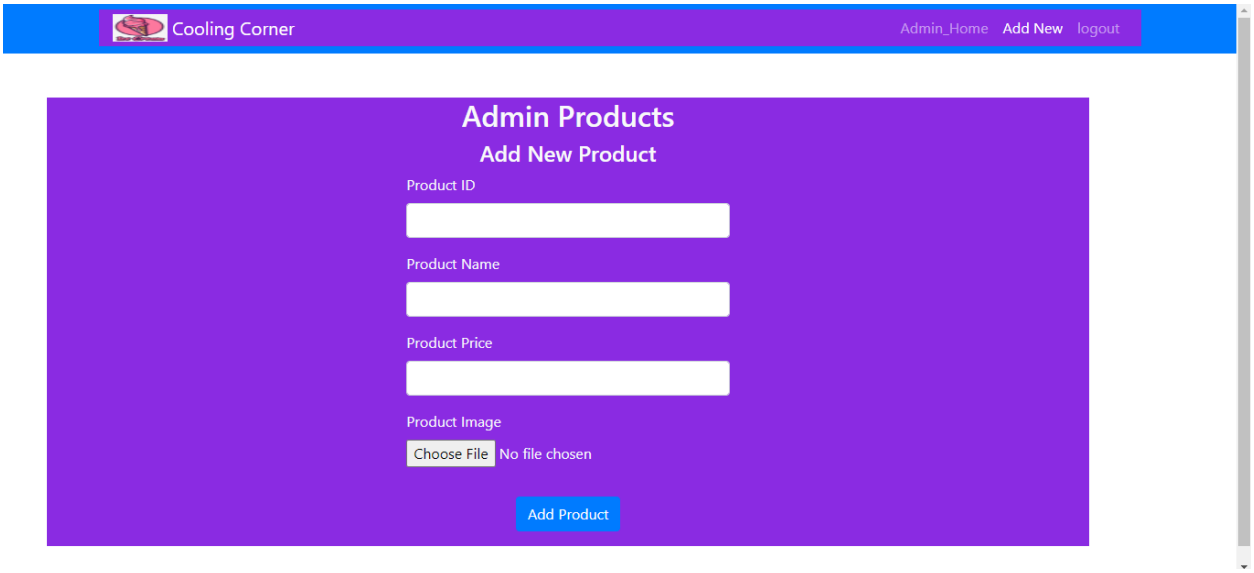


Fig. 5.1.6 Add product page

5.1.4 Manage Product:

In this module, Admin can update and delete products and details like product id, product name. These Details are stored in the “Product” database. After the updation and deletion, Admin can view the product details.

Manage:

```
<?php
session_start();
if (isset($_POST['btn']))
        {
   $clickedId = $_POST['id'];
   echo "Clicked ID: " . $clickedId;
        $_SESSION['id']=$clickedId;
        header("location:product_edit_2.php");
}

if (isset($_POST['delete']))
        {
   $clickedId = $_POST['id'];
        $connect = mysqli_connect("localhost", "root", "", "icecream");
```

```
        $query=mysqli_query($connect,"delete from products where id='$clickedId'");
}

?>
<html>
<head>
  <title>package</title>
  <title>Bootstrap Example</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link                href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css"
rel="stylesheet">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js"></script>
  <style>
    .card {
      height: 100%;
    }

    .card-image-container {
      position: relative;
      padding-top: 100%; /* Set the aspect ratio as per your preference (e.g., 1:1 for square) */
      overflow: hidden;
    }

    .card-image-container img {
      position: absolute;
      top: 0;
      left: 0;
      width: 100%;
      height: 100%;
```

```
      object-fit: cover;
    }
  </style>
</head>
<body>


<div class="container m-5">
  <h2 style="color:#fff">Products</h2>

  <div class="row">
    <?php

    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "icecream";

    $conn = new mysqli($servername, $username, $password, $dbname);
    if ($conn->connect_error) {
      die("Connection failed: " . $conn->connect_error);
    }

    $query = "SELECT * FROM products";
    $result = $conn->query($query);

    if ($result->num_rows > 0) {
      while ($row = $result->fetch_assoc()) {
        $imageData = $row['image'];
        ?>
        <div class="col-lg-3 col-md-6 mb-4 p-10">
```

```
        <div class="card ">
          <div class="card-image-container ">
            <img src="<?= base64_encode($imageData) ?>" alt="Card image">
          </div>
          <div class="card-body">
            <h4 class="card-title"><?= $row['name'] ?></h4>
            <p class="card-text text-dark">Price: <?= $row['price'] ?>Rs.</p>
                        <form method='POST'><input type="hidden" name="id" value="<?php echo  $row['id'] ?> ">
                        <button type="submit" name="btn" style="width:200px;background-color: #35bed4; color: #fff ;border:4px solid #fff; border-radius: 50px">Update</button>
        <br>
                        <button type="submit" name="delete" style="width:200px;background-color: #d43535; color: #fff ;border:4px solid #fff; border-radius: 50px">delete </button></form>

          </div>
        </div>
      </div>
    <?php
     }
    }
    $conn->close();
    ?>
```

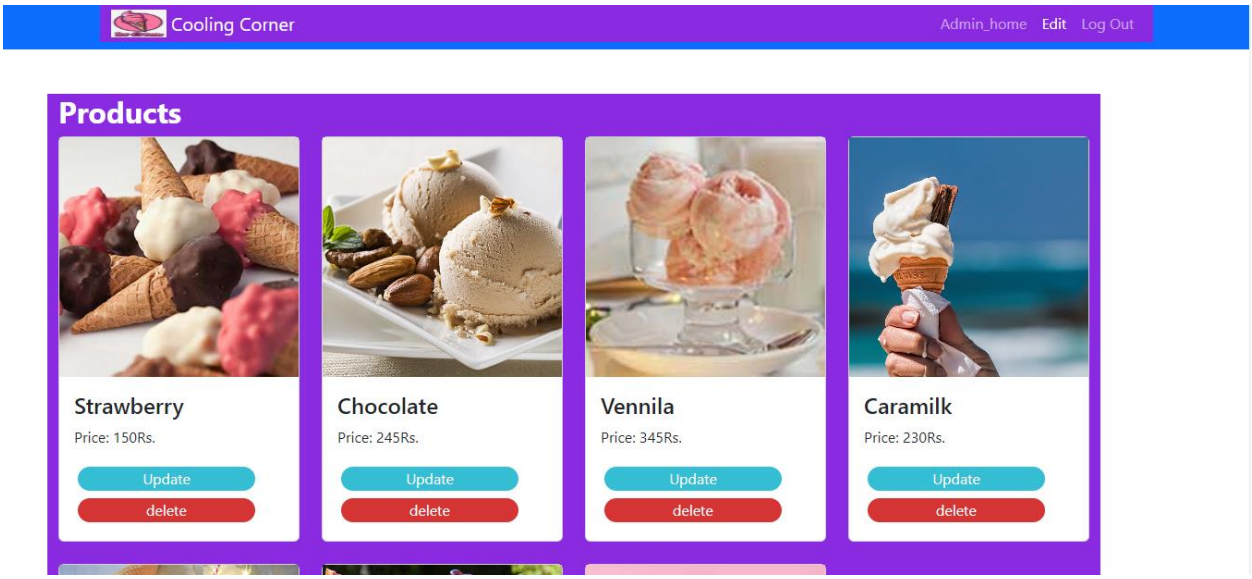


Fig. 5.1.7 Managing product page

5.1.5 Contact:

In this module, admin can able to view the purchased product details . we can able to view todays order ,pending and delivered orders. The purchased product details will be retrieve from the database. When select any sub- categories, The product details are retrieve from the database. While click the search button after select the sub- categories , the data grid view shows the previous sales.

```
?php
// Check if the form is submitted
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Retrieve form data
    $name = $_POST["name"];
    $gender = $_POST["gender"];
    $mobile = $_POST["mobile"];
    $email = $_POST["email"];
    $source = $_POST["source"];
    $message = $_POST["message"];

    // Validate and sanitize the data (you can add more validation if needed)
```

```
$name = htmlspecialchars($name);
$gender = htmlspecialchars($gender);
$mobile = htmlspecialchars($mobile);
$email = htmlspecialchars($email);
$source = htmlspecialchars($source);
$message = htmlspecialchars($message);

// Database connection settings
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "icecream";

// Create a new PDO instance
try {
    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    // Prepare and execute the SQL statement
    $stmt = $conn->prepare("INSERT INTO contact (name, gender, mobile, email, source, message) VALUES (?, ?, ?, ?, ?, ?)");
    $stmt->execute([$name, $gender, $mobile, $email, $source, $message]);

    echo "Data forwarded successfully!";
} catch (PDOException $e) {
    echo "Error: " . $e->getMessage();
}

// Close the database connection
$conn = null;
```

```
}
?>
<!-- PHP Ends -->

    <!-- Content -->
    <div class="container my-5">
      <div class="row">
        <div class="col-md-8">
          <div class="mapouter">
            <div class="gmap_canvas">
              <iframe
                class="col-md-12"
                height="800"
                id="gmap_canvas"
                src="https://maps.google.com/maps?q=Trichy&z=13&output=embed"
                frameborder="0"
                scrolling="no"
                marginheight="0"
                marginwidth="0"
              ></iframe>
            </div>
```

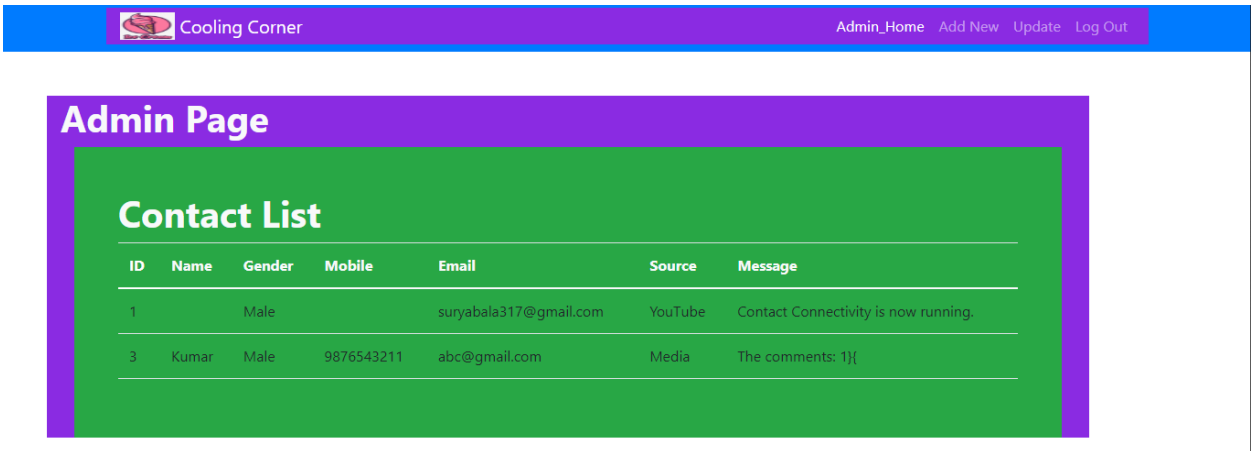


Fig. 5.1.8 Comments of users

5.2 Users:

- Register & Login
- View product
- Orders
- Contact

5.2Users:

In this module, the user can register their details. After login, he can able to search the product what he wants, once the user select the product and place the order.

5.2.1 Register & Login:

There is registration form available where new user can create their account by providing required information to the system. The registration form details are like name, email, contact These details are stored in the database. And then can getting to the username and password in the system.

User reg & log:

```
<?php
session_start();


// Check if the form is submitted
if ($_SERVER["REQUEST_METHOD"] == "POST") {
  // Retrieve the form data
  $fullname = $_POST["fullname"];
  $mobile = $_POST["mobile"];
  $email = $_POST["email"];
  $password = $_POST["password"];


  // Create a database connection
  $servername = "localhost";
  $username = "root";
  $password_db = "";
```

```
  $database = "icecream";

  $conn = new mysqli($servername, $username, $password_db, $database);

  // Check connection
  if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
  }

  // Prepare and bind the insert statement
  $stmt = $conn->prepare("INSERT INTO users (id, fullname, mobile, email, password) VALUES
(NULL, ?, ?, ?, ?)");

  if (!$stmt) {
    die("Prepare failed: " . $conn->error);
  }

  $stmt->bind_param("ssss", $fullname, $mobile, $email, $password);

  // Execute the statement
  if ($stmt->execute()) {
    // Data inserted successfully
    ?>
<script language="javascript">
alert("Registered Successfully");
window.location.href="register.php";
</script>
  <?php
    echo "Registration successful!";
    header("location: log in.php");
    exit;
```

```
    }
   else {
    // Failed to insert data
    echo "Error: " . $stmt->error;
   }

  // Close the statement and database connection
  $stmt->close();
  $conn->close();
}
?>

<!-- php ends -->

    <!-- Content -->
    <div class="container my-5">
      <div class="row">
        <div class="col-md-12">
          <div class="card">
            <div class="card-body">
              <!--<form action="product.php" method="POST">-->
                <form action=" " method="POST">

                <div class="form-group">
                  <label for="fullname">User Name</label>
                  <input
                    type="text"
                    class="form-control"
                    name="fullname"
                    placeholder="Enter your User name"
                    required
```

/>

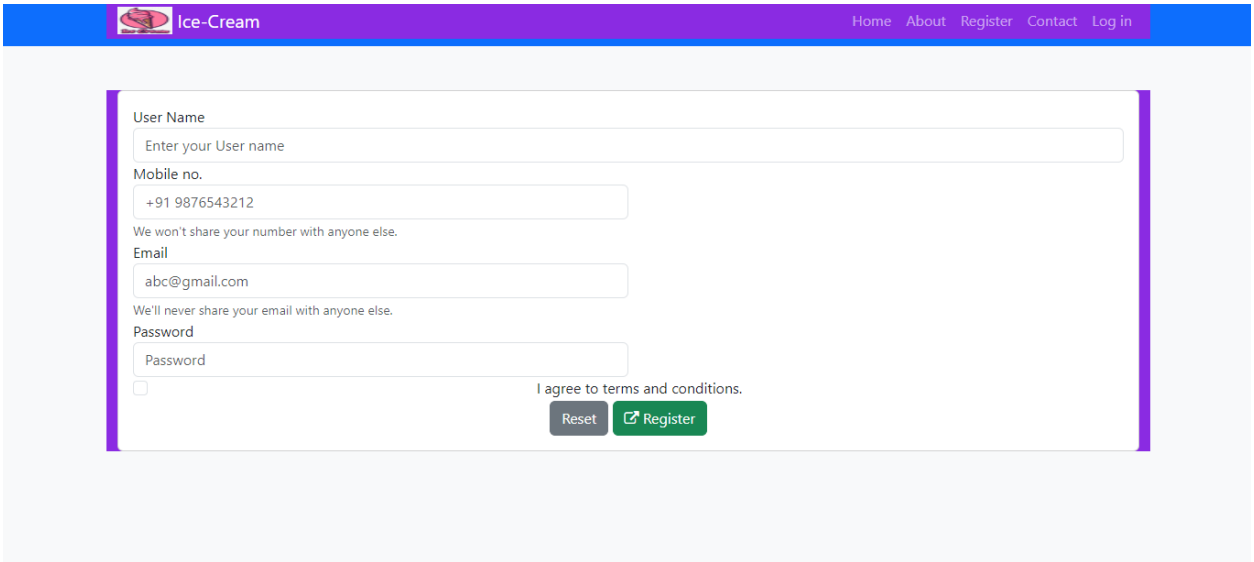


Fig. 5.2.1 Register & Login of users

5.2.3 Orders:

After viewing all products, the user can buy product using this module.

Order:

```
<div class="container text-light m-5">
  <h2>CART</h2>

  <div class="row">
  <?php
  $total = 0;
  $cost = 0;
  session_start();
  $em = $_SESSION['uname'];
  echo "<h1>".$em."</h1>";
  $servername = "localhost";
  $username = "root";
  $password = "";
  $dbname = "icecream";
```

```
  $conn = new mysqli($servername, $username, $password, $dbname);
  if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
  }

  $query1 = "SELECT * FROM cart where uid = '$em'";
  $result = $conn->query($query1);

  if ($result->num_rows > 0) {
    echo "<table class='table'>";
    echo
"<thead><tr><th>Name</th><th>Price</th><th>Quantity</th><th>Cost</th><th></th></tr></t
head>";
    echo "<tbody>";
    $uid = ""; // Define $uid variable before the while loop
    while ($row= $result->fetch_assoc()) {
      $pid = $row['pid'];
      $uid = $row['uid'];
      $query2 = "SELECT * FROM cart where uid = '$em'";
      $result2 = $conn->query($query2);
      $row2 = $result2->fetch_assoc();
      echo "<tr>";
      echo "<td>".$row2['uid']."</td>";
      echo "<td>".$row2['price']."</td>";
      echo "<td>".$row['quantity']."</td>";

     echo $cost = (int)$row['quantity'] *(int) $row2['price'];
     echo $total = $total + $cost;
      echo "<td>".$cost."</td>";
      echo "<td><form method='POST'>";
      echo "<input type='hidden' name='id' value='$pid'>        ";
```

```
    echo "<button type='submit' name='delete' class='btn btn-dark'>Remove</button></form></td>";
     echo "</tr>";
   }
   echo "</tbody>";
   echo "</table>";
  } else {
   echo "No items in the cart.";
  }
  echo "<h2> Total: ".$total."$</h2>";

  if (isset($_POST['delete'])) {
   $clickedId = $_POST['id'];
   $query = mysqli_query($conn, "DELETE FROM cart WHERE pid = '$clickedId' AND uid = '$uid'");
   echo $clickedId."".$uid;
   header("location:cart.php");
  }

  $conn->close();
  ?>
  <button type="submit" name="btn" style="width:120px;margin: 0 auto" class="btn btn-primary">BUY</button>
</div>
```

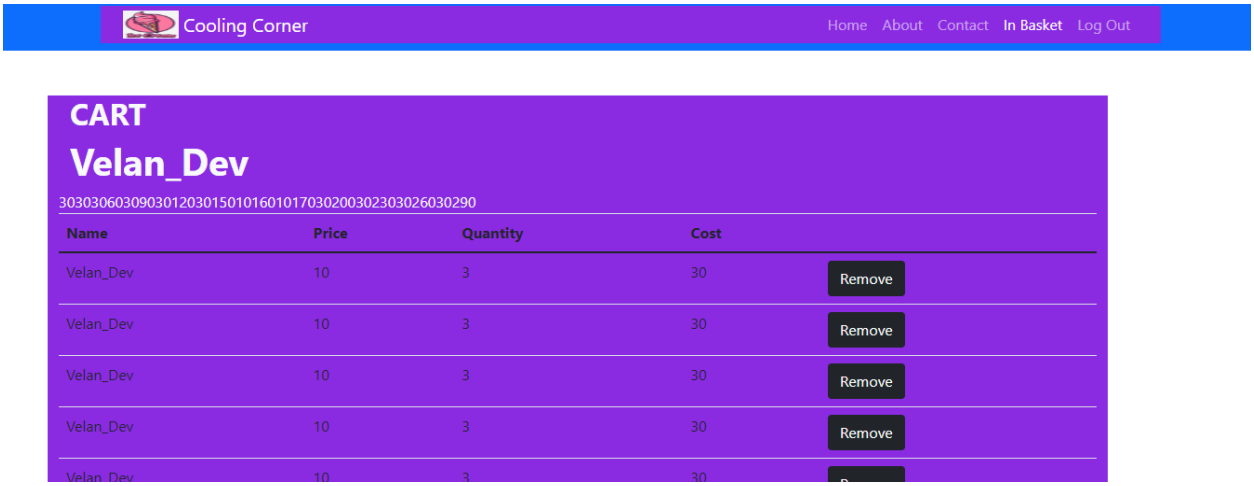


Fig. 5.2.2 Orders of users

5.2.4.Contact:

Code:

```
<!-- Content -->
<div class="container my-5">

<iframe
class="col-md-12"
height="800"
id="gmap_canvas"
src="https://maps.google.com/maps?q=Trichy&z=13&output=embed"
frameborder="0"
scrolling="no"
marginheight="0"
marginwidth="0"
></iframe>
</div>
<style>
/* .mapouter {
position: relative;
text-align: right;
height: 500px;
width: 600px;
```

```
}
.gmap_canvas {
overflow: hidden;
background: none !important;
height: 500px;
width: 600px;
} */
</style>
</div>
</div>
<div class="col-md-4">
<div class="card">
<div class="card-header bg-primary text-white">
<i class="fa fa-envelope mr-2"></i>
Contact us anytime!
</div>
<div class="card-body">
<form action="" method="POST">
<!--
name [text]
number [number]
email [email]
message [textarea]
source [select]
-->
<div class="form-group">
<label for="name">Name</label>
<input
type="text"
name="name"
class="form-control"
placeholder="Enter your name"
/>
</div>
<div class="form-group">
<label for="gender">Gender:</label>
<div class="ml-2 form-check form-check-inline">
<input
```

```
class="form-check-input"
type="radio"
name="gender"
value="Male"
id="gender"
/>
<label class="form-check-label" for="gender">Male</label>
</div>
<div class="form-check form-check-inline">
<input
class="form-check-input"
type="radio"
name="gender"
id="female"
value="Female"
/>
<label class="form-check-label" for="female">Female</label>
</div>
<div class="form-check form-check-inline">
<input
class="form-check-input"
type="radio"
name="gender"
id="others"
value="Others"
/>
<label class="form-check-label" for="others">Others</label>
</div>
</div>
<div class="form-group">
<label for="mobile">Mobile no.</label>
<input
type="text"
name="mobile"
class="form-control"
placeholder="+91 9876543212"
/>
</div>
```

```
<div class="form-group">
<label for="email">Email</label>
<input
type="email"
name="email"
id="email"
class="form-control"
placeholder="abc@gmail.com"
aria-describedby="emailHelp"
/>
```

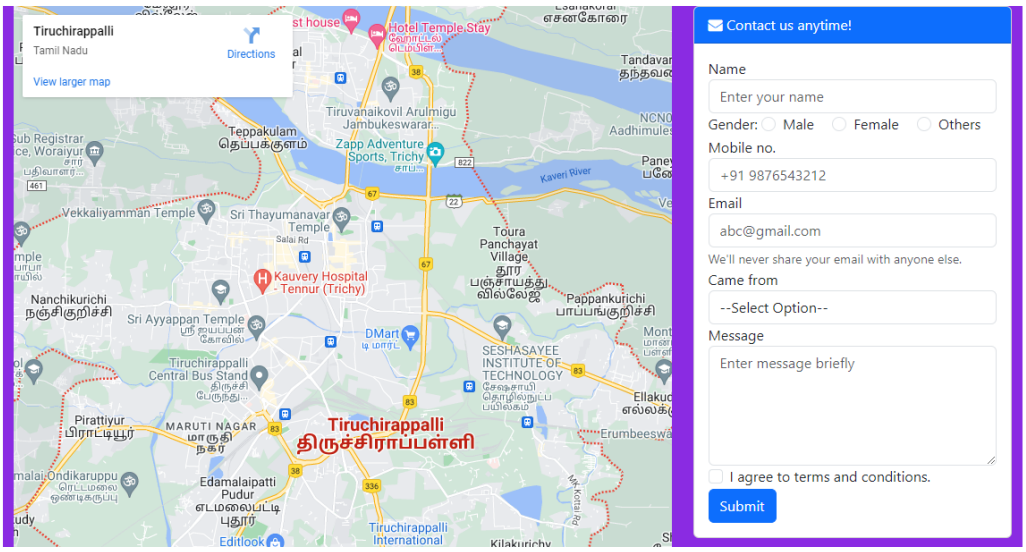


Fig. 5.2.4 Contact of users

6.TESTING

Testing is a set activity that can be planned and conducted systematically. Testing begins at the module level and work towards the integration of entire computers-based system. Nothing is complete without testing, as it is vital success of the system.

Testing Objectives:

There are several rules that can serve as testing objectives, they are

1. Testing is a process of executing a program with the intent of finding an error
2. A good test case is one that has high probability of finding an undiscovered error.
3. A successful test is one that uncovers an undiscovered error.

Tests used for implementation efficiency attempt to find ways to make a correct program faster or use less storage. It is a code-refining process, which re-examines the implementation phase of algorithm development. Tests for computational complexity amount to an experimental analysis of the complexity of an algorithm or an experimental comparison of two or more algorithms, which solve the same problem.

The data is entered in all forms separately and whenever an error occurred, it is corrected immediately. A quality team deputed by the management verified all the necessary documents and tested the Software while entering the data at all levels.

TYPES OF TESTING:

6.1 Unit Testing:

The first test in the development process is the unit test. The source code is normally divided into modules, which in turn are divided into smaller units called units. These units have specific behavior. The test done on these units of code is called unit test. Unit test depends upon the language on which the project is developed. Unit tests ensure that each unique path of the project performs accurately to the documented specifications and contains clearly defined inputs and expected results.

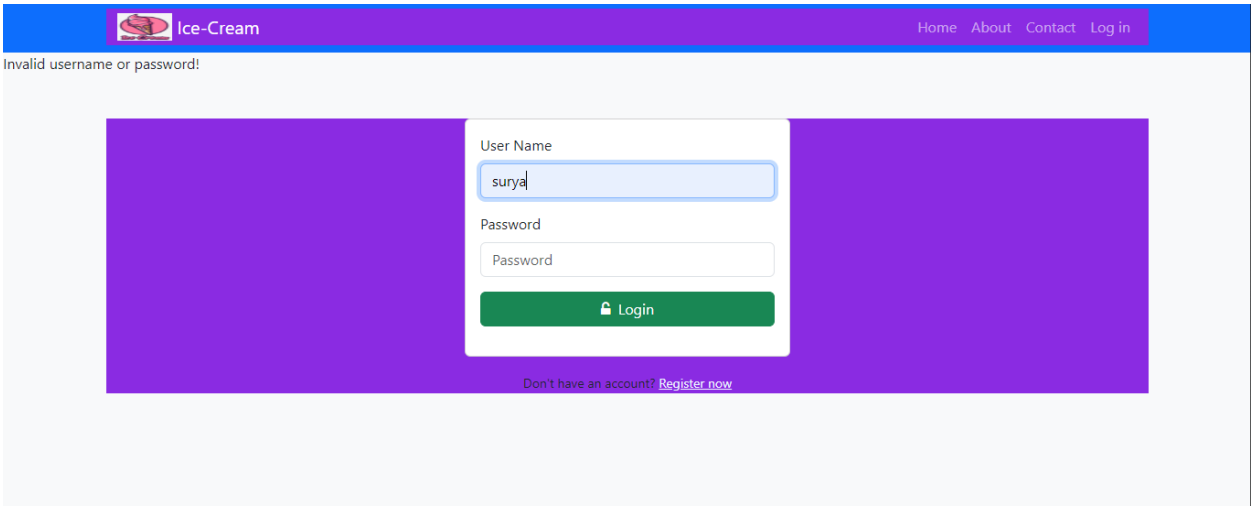
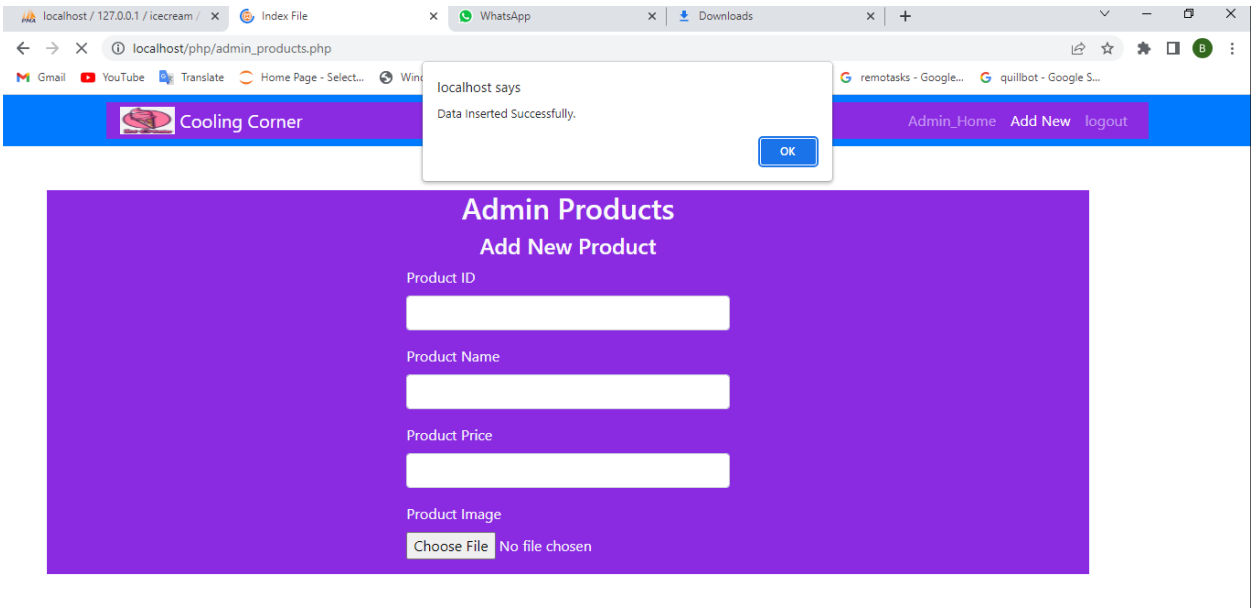


Fig. 6.1 Unit Testing of user

6.2 Integration Testing:

Testing in which modules are combined and tested as a group. Modules are typically code modules, individual applications, source and destination applications on a network, etc. Integration Testing follows unit testing and precedes system testing. Testing the links and buttons that connect the modules are done here(Navigation bar, Login button) Testing after the product is code complete. Betas are often widely distributed or even distributed to the public at large in hopes that they will buy the final product when it is release.



6.3 System Testing:

After a system has been verified, it needs to be throughly tested to ensure that every component of the system is performing in accordance with the specific requirements and that it is operating as it should including when the wrong functions are requested or the wrong data is introduced.

Testing measures consist of developing a set of test criteria either for the entire system or for specific hardware, software and communications components. For an important and sensitive system such as an electronic voting system, a structured system testing program may be established to ensure that all aspects of the system are throughly tested.

Testing measures that are followed include:

Applying qualitative assessments to determine whether the test criteria have been met.

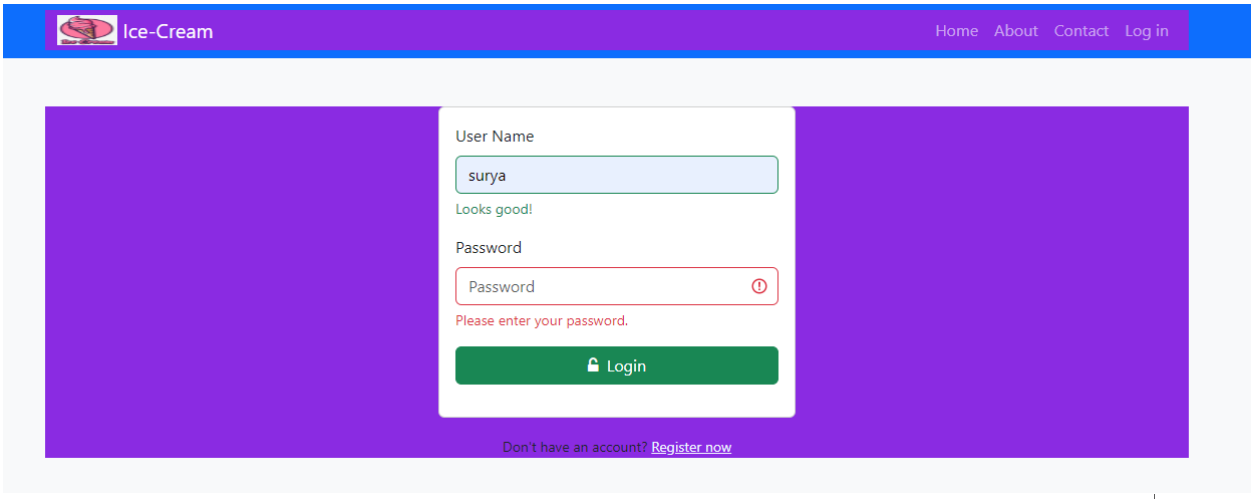
Conducting tests in “laboratory” conditions and conducting tests in a variety of “real life” conditions.

Conducting tests over an extended period of time to ensure systems can perform consistently.

Conducting “load tests”, simulating as close as possible likely conditions while using or exceeding the amounts of data that can be expected to be handled in an actual situation.

6.4 Validation Testing:

Valid and invalid data should be created and the program should be made to process this data to catch errors. When the user of each module wants to enter into the page by the login page using the username and password. If the user gives the wrong password or username then the information is provided to the user like “you must enter username and password”. Here the inputs given by the user are validated. That is password validation, format of date are correct, textbox validation. Changes that need to be done after result of this testing.



7.CONCLUSION

This Web-application" COOLING CORNER ICECREAM SHOP WEB PORTAL" avoids the manual work and the problems concern with it. It is an easy way to obtain the information regarding the various products information that is present in the markets. This system is an improved application better than the existing one's regarding the information about the various products. This project is a computerized solution for storing the details of products, customer, order. Here, we can conclude that the application been developed is to reduce manpower, time, cost and various complexities. The system is highly interactive and flexible for further enhancement.

FUTURE ENHANCEMENT:

It is not possible to develop a system that makes all the requirements of the user. User requirements keep changing as the system is being used. Some of the future enhancements that can be done to this system are As the technology emerges, it is possible to upgrade the system and can be adaptable to desired environment. In the future, we can track the product by using the order ID.

And with the help of users email and contact details we will inform them via messages or mail for the upcoming offers and cashbacks related to the our ice-cream products.

8. REFERENCES

BOOK REFERENCE:

1. Alan Forbes - The joy of PHP Programming.
2. Hill Publication - PHP Beginners Guide by McGraw
3. Hill Publication - JavaScript by McGraw
4. Kevin Yank - PHP & MySQL Novice to Ninja

WEBSITE REFERENCE:

- https://www.mysqltutorial.org
- https://getbootstrap.com
- https://github.com
- www.w3schools.com