

Centre Name: ACE-HCMC-2-FPT.

Address: 590 Cach Mang Thang 8, District 3, Ho Chi Minh City, Viet Nam.

Ice Cream Shop Management System

Supervisor:	Ms. Le Mong Thuy	
Semester:	3	
Batch No:	T1.1804.M1	
Group No:	5	
Order:	Full name	Roll No.
1.	Bui Van Huong	Student1093630
2.	CHAU THANH PHONG	Student1088413
3.	DANG NGUYEN DAN LINH	Student1053724
4.	LY TIEN DAT	Student1093706
5.	MAI HOANG MINH TIEN	Student1057354

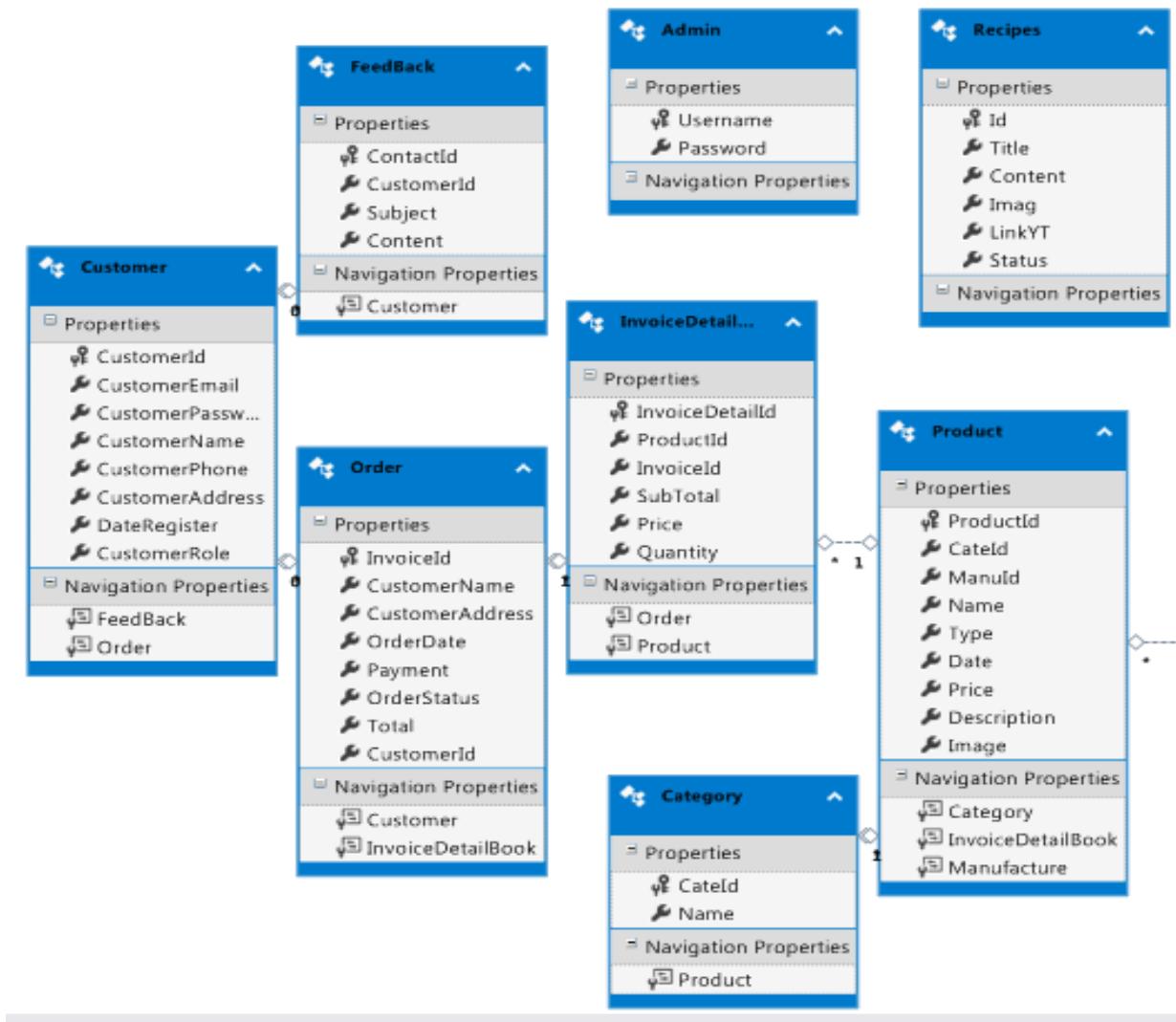
Contents

REVIEW 3	2
I. DATABASE DESIGN	3
1. DATABASE DESIGN DIAGRAM	3
2. DATABASE STRUCTURE.....	4
2.1. <i>Table Admin</i>	4
2.2. <i>Table Customer</i>	4
2.3. <i>Table Feedback</i>	4
2.4. <i>Table Category</i>	4
2.5. <i>Table Product</i>	5
2.6. <i>Table Recipes</i>	5
2.7. <i>Table Order</i>	5
2.8. <i>Table InvoiceDetailBook</i>	6
II. GUI DESIGN	7
1. USER INTERFACE	7
1.1. <i>Home Page</i> : Introduce the website with the navigation bar links.	7
1.2. <i>Contact Us Page</i> : Show contact information of clinic & feedback form for user. (must login first to send feedback).....	11
1.3. <i>Login Page</i> : Login form for user.....	14
1.4. <i>Logout</i> : notification will appear when admin click logout button.....	17
1.5. <i>Register Page</i> : Register form for user if they haven't account.	18
1.6. <i>Change Personal Information Page</i> : User can change their information here.	21
1.7. <i>Update Password Page</i> : User can change their Password here.	24
1.8. <i>Register Member Page</i> :If the user is not register ,user can fill value member is this profiles.....	27
1.9. <i>Book Page</i> : This page display list of medicines, user can search item with search bar.	29
1.10. <i>Product Details Page</i> : This page display information of a product. User also can add this product to cart.....	32
1.11. <i>Free Recipes</i> : This page display list of recipes for your user don't use account.	34
1.12. <i>Full Recipes</i> : This page display list of recipes for your user register a member as month or year	37
1.13. <i>Detail of Recipes</i> : This page display the way to cook this favourite ice cream.....	39
1.14. <i>FAQ Page</i> : This page display list of frequently of question.....	41
1.15. <i>Your Cart Page</i> : Items which user had chosen will display here.	43
1.16. <i>Checkout page</i> : This page show for user confirm payment.	46
1.17. <i>INVOICEDETAILBOOK</i> : THIS PAGE DISPLAY INFORMATION ABOUT ORDER USER JUST BOUGHT.....	50
1.18. <i>DETAILS OF AN ORDER</i> : WHEN USER CLICK DETAILS BUTTON IN ORDER DETAILS PAGE, IT WILL SHOW DETAILS OF IT.	52
1.19. <i>Personal order history</i> : this page display history all orders of personal user. User can view it by click to menu "Order History" appears when user login successfully.	54
2. ADMIN INTERFACE:	58
2.1. <i>Login Page</i> : Login form for admin (default: Username: admin , password: 123456)	58
2.2. <i>Logout</i> : notification will appear when admin click logout button.....	60
2.3. <i>Change Password Page</i> : Admin can change password here.....	61
2.4. <i>Home Page</i> : has navigation bar links to all manage pages.....	63
2.5. <i>Category Management Page</i> : Admin can manage product here.....	64
2.6. <i>Product Management Page</i> : Admin can manage product here	72
2.7. <i>User Management Page</i> : Admin can manage user here.....	82
2.8. <i>Order Management Page</i> : Admin can manage order here	88
2.9. <i>Feedback Management Page</i> : Admin can manage feedback here.....	93
2.10. <i>Recipes Manage Page</i> : Admin can manage recipe in here	97
3. SITE MAP	106
1. TEST PLAN/TEST CASE	108
2. CHECKLIST OF VALIDATIONS	109
3. SUBMISSION CHECKLIST	110
TASK SHEET REVIEW 3.....	111

REVIEW 3

I. DATABASE DESIGN

1. Database Design Diagram



2. Database Structure

2.1. Table Admin

No	Field Name	Data Type	Null	Default	Key	Reference Table	Description
1	Username	varchar(20)	No	admin	PK		Admin's Username
2	Password	varchar(20)	No	123456			Admin's Password

2.2. Table Customer

No	Field Name	Data Type	Null	Default	Key	Reference Table	Description
1	CustomerId	int	No	Identity(1,1)	PK		User's ID
2	CustomerPassword	varchar(50)	Yes				User's Password
3	CustomerName	varchar(50)	Yes				User's Name
4	CustomerPhone	varchar(100)	Yes				User's Phone
5	CustomerAddress	datetime	Yes				User's Address
6	DateRegister	varchar(50)	Yes				User's Date Regist
7	CustomerRole	tinyint	Yes				User's Role
8	CustomerEmail	varchar(50)	Yes				User's Email

2.3. Table Feedback

No	Field Name	Data Type	Null	Default	Key	Reference Table	Description
1	ContactId	int	No	Identity(1,1)	PK		Feedback's ID
2	CustomerId	Int	No		FK	User(ID)	User's ID
3	Subject	varchar(500)	No				Feedback's Subject
4	[Content]	varchar(30)	No				Feedback's Content

2.4. Table Category

No	Field Name	Data Type	Null	Default	Key	Reference Table	Description

1	Cateld	Int	No	Identity(1,1)	PK	Category's ID
2	Name	varchar(50)	No			Category's Name

2.5. Table Product

No	Field Name	Data Type	Null	Default	Key	Reference Table	Description
1	ProductId	int	No	Identity(1,1)	PK		Product's ID
2	Cateld	int	No		FK	Category (Cateld)	Category's ID
3	Manuld	int	No		FK	Manufacture (Manuld)	Manufacture's ID
4	Name	varchar(50)	No				Product's Name
5	Type	varchar(50)	No				Product's Type
6	Date	datetime	No				Recipe Date
7	Price	double	No				Product's Price
8	Description	varchar(1000)	No				Product's Description
9	Image	varchar(50)	No				Product's Image

2.6. Table Recipes

No	Field Name	Data Type	Null	Default	Key	Reference Table	Description
1	Id	int	No	identity(1,1)	PK		Recipe's ID
2	Title	varchar(100)	Yes				Recipe's Title
3	[Content]	text	Yes				Recipe's Content
4	Imag	text	Yes				Recipe's Image
5	LinkYT	text	Yes				Recipe's Link
6	Status	varchar(10)	Yes				Recipe's Status

2.7. Table Order

No	Field Name	Data Type	Null	Default	Key	Reference Table	Description
1	Invoiceld	int	No	Identity(1,1)	PK		Order's ID

2	CustomerId	int	No	FK	Customer(CustomerId)	Customer's ID
3	CustomerName	varchar(50)	No			Order's Customer Name
4	CustomerAddress	varchar(50)	No			Order's Customer Address
5	OrderDate	datetime	No			Order Date
6	Payment	varchar(20)	No			Payment
7	OrderStatus	bit	No			Order Status
8	Total	int	No			Total

2.8. Table InvoiceDetailBook

No	Field Name	Data Type	Null	Default	Key	Reference Table	Description
1	InvoiceDetailId	int	No	identity(1,1)	PK		OrderDetails's ID
2	Invoiceld	int	No		FK	Order (OrderId)	Order's ID
3	ProductId	int	No		FK	Product (ProductId)	Product's ID
4	Price	double	No				Product's Price
5	Quantity	int	No				Product's Quantity
6	SubTotal	double	No				Product's Sub Total

II. GUI DESIGN

1. User Interface

1.1. Home Page: Introduce the website with the navigation bar links.



ICE CREAM PALOR SHOP

Best quality website selling Ice Cream And Dessert

Ice cream is a snack that everyone loves, especially young people. Your shop is selling fresh ice cream to serve people, but you can only sell in very small quantities and not very well known.

About the site

Website is the shortest path from your store to the user.

Having a website will easily promote your brand to everyone through google search engine, social network...

Your website is integrated with shopping cart features and online purchases that customers can buy right from the website without going to the place.

Website is the place to display ice cream products, arranged with the most logical order. For each product, you can detail the price, characteristics and properties of the product for the best user search.

More Info

We have more than 700 stores across 50 countries. You will be surprised to find us almost all over the world with typical ice cream flavors.

Restaurant Development Milestones in the world

- India
- Australia
- Korea
- Vietnam
- China
- Japan
- Canada
- UK

Address: 500 Cách Mạng Tháng Tám
Ward 8 County 3 HCMC

Phone: 034567890

Business Hours: 8AM - 22PM

Member Creator

<p>Bùi Văn Hưởng Team Leader Responsible, high team spirit to lead the good team. Email: bvh1802@yahoo.com</p>	<p>Lý Tiên Đạt Young programmer Is a long-time programmer who has a lot of experience, helps the members, fulfills the assigned tasks. Email: datly1994@gmail.com</p>	<p>Châu Thành Phong Young Programmer Enthusiastic, rich experience, many bold thoughts. Email: thanhphong2407@gmail.com</p>
--	---	---

<p>Mai Hoàng Minh Tiến Young programmer Enthusiastic, rich experience, many bold thoughts. Email: konuybirds@gmail.com</p>	<p>Đặng Nguyễn Dân Linh Young programmer Enthusiastic, rich experience, many bold thoughts. Email: dangnguyendanlinh.95.dl@gmail.com</p>	<p>Team Smart, hope Strong, reliable, cooperative. together for the future.</p>
--	--	--



MY TEAM

Code Your Own Way

f t g l

Bùi Văn Hưởng
Châu Thành Phong
DÀNG NGUYỄN DÂN LINH
LÝ TIỀN ĐẠT

CONTACT

Address: 500-CMT8, District 3, HCMC
Phone: (+8428) 5 846 0846
Email: optach.hcm@fpt.com.vn

No	Name	Type	Source Code	Event	Description	Status
1	Navigation Bar	Hyperlink		Click	Choose & click to go to sub pages	Enable
2	Login	Hyperlink		Click	Click to go to login page	Enable
3	Register	Hyperlink		Click	Click to go to register page	Enable
4	Cart	Hyperlink	Figure 1	Click	Click to go to cart page	Enable
5	Form About Us	TextBox	Figure 2	Focus	Information about website	Enable
6	Book Dessert	Hyperlink		Click	Click to go to Book product page	Enable
7	Free Recipes	Hyperlink		Click	Click to go to Recipes page(login to see Full Recipes)	Enable
8	FAQ	Hyperlink	Figure 1	Click	Frequently asked questions	Enable
9	Contact	Hyperlink	Figure 1	Click	Send feedback	Enable
10	Footer	Hyperlink	Figure 3	Click	Some information to contact with clinic	Enable

Source code:

View:

```

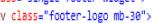
    }
    </ul>
  </div>
</div>
</div>
</div>
<!-- LOGO AND MENU SECTION -->
<div class="top-logo" data-spy="affix" data-offset-top="250">
  <div class="container">
    <div class="row">
      <div class="col-md-1 wed-logo">
        <a href="@Url.Action("Index", "Home")'> <img src "~/Images/OCM.jpg" style="width:auto; height:auto; margin-bottom:-9px;"></a>
      </div>
      <div class="col-md-1">
      </div>
      <div class="main-menu col-lg-8">
        <ul>
          <li><a href="@Url.Action("Index", "Home")'">Home</a></li>
          <!--<li><a class='dropdown-button ed-sub-menu' href='#' data-activates='dropdown1'>Courses</a></li>-->
          <li>
            <a href="@Url.Action("Book", "Product")'">Book Dessert</a>
          </li>
          <li>
            <a href="@Url.Action("FreeRecipes", "Home")'">Free Recipes</a>
          </li>
          <li>
            <a href="@Url.Action("Contact", "Contact")'">Contact Us</a>
          </li>
          <li>
            <a href="@Url.Action("FAQ", "FAQ")'">FAQ</a>
          </li>
          @if (Session["email"] != null)
          {
            <li> <a href="@Url.Action("Orderhistory", "Shoppingcart")'">Order History</a></li>
          }
          <li>
            <a href("~/ShoppingCart/Index")><img src "~/images/Cart3.jpg" height="62" width="55" /></a>
          </li>
        </ul>
      </div>
    </div>
  </div>
</div>

```

Figure 1

```

<footer class="footer-area bg-img" style="background-image: url('../../../images/35.jpg');">
    <!-- Main Footer Area -->
    <div class="main-footer-area">
        <div class="container">
            <div class="row">

                <!-- Single Footer Widget -->
                <div class="col-12 col-sm-6 col-lg-3">
                    <div class="single-footer-widget">
                        <div class="footer-logo mb-30">
                            <a href="#"></a>
                        </div>
                        <p>Cook Your Own Way</p>
                        <div class="social-info">
                            <a href="https://www.facebook.com/aptech.fpt/?ref=br_rs" target="_blank"><i class="fa fa-facebook" aria-hidden="true"></i></a>
                            <a href="https://twitter.com/?lang=en" target="_blank"><i class="fa fa-twitter" aria-hidden="true"></i></a>
                            <a href="https://plus.google.com" target="_blank"><i class="fa fa-google-plus" aria-hidden="true"></i></a>
                            <a href="https://www.instagram.com/?hl=en" target="_blank"><i class="fa fa-instagram" aria-hidden="true"></i></a>
                            <a href="https://vn.linkedin.com/" target="_blank"><i class="fa fa-linkedin" aria-hidden="true"></i></a>
                        </div>
                    </div>
                </div>
                <!-- Single Footer Widget -->

                <!-- Single Footer Widget -->
                <div class="col-12 col-sm-6 col-lg-3">
                    <div class="single-footer-widget">
                        <div class="widget-title">
                            <h5>My Team</h5>
                        </div>

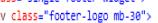
                        <!-- Single Best Seller Products -->
                        <div class="single-best-seller-product d-flex align-items-center">
                            <div class="product-info">
                                <a href="#">Bui Van Huong</a>
                            </div>
                        </div>
                        <!-- Single Best Seller Products -->
                        <div class="single-best-seller-product d-flex align-items-center">
                            <div class="product-info">
                                <a href="#">Chau Thanh Phong</a>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
</footer>

```

Figure 2

```

<footer class="footer-area bg-img" style="background-image: url('../../../images/35.jpg');">
    <!-- Main Footer Area -->
    <div class="main-footer-area">
        <div class="container">
            <div class="row">

                <!-- Single Footer Widget -->
                <div class="col-12 col-sm-6 col-lg-3">
                    <div class="single-footer-widget">
                        <div class="footer-logo mb-30">
                            <a href="#"></a>
                        </div>
                        <p>Cook Your Own Way</p>
                        <div class="social-info">
                            <a href="https://www.facebook.com/aptech.fpt/?ref=br_rs" target="_blank"><i class="fa fa-facebook" aria-hidden="true"></i></a>
                            <a href="https://twitter.com/?lang=en" target="_blank"><i class="fa fa-twitter" aria-hidden="true"></i></a>
                            <a href="https://plus.google.com" target="_blank"><i class="fa fa-google-plus" aria-hidden="true"></i></a>
                            <a href="https://www.instagram.com/?hl=en" target="_blank"><i class="fa fa-instagram" aria-hidden="true"></i></a>
                            <a href="https://vn.linkedin.com/" target="_blank"><i class="fa fa-linkedin" aria-hidden="true"></i></a>
                        </div>
                    </div>
                </div>
                <!-- Single Footer Widget -->

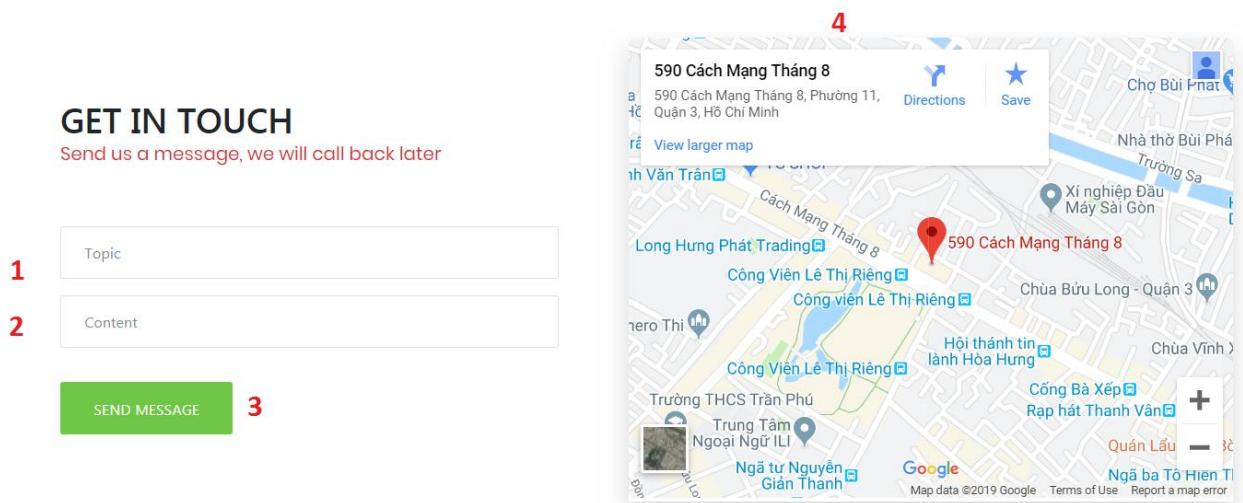
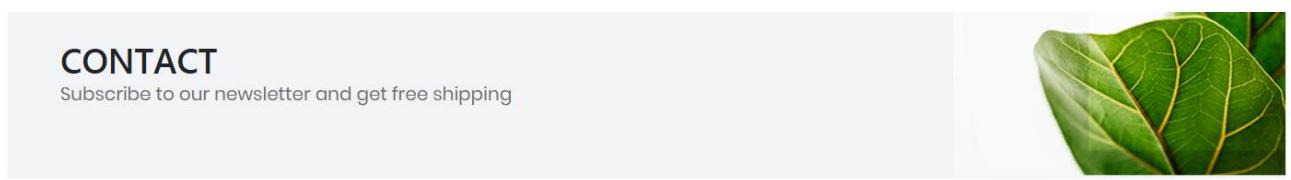
                <!-- Single Footer Widget -->
                <div class="col-12 col-sm-6 col-lg-3">
                    <div class="single-footer-widget">
                        <div class="widget-title">
                            <h5>My Team</h5>
                        </div>

                        <!-- Single Best Seller Products -->
                        <div class="single-best-seller-product d-flex align-items-center">
                            <div class="product-info">
                                <a href="#">Bui Van Huong</a>
                            </div>
                        </div>
                        <!-- Single Best Seller Products -->
                        <div class="single-best-seller-product d-flex align-items-center">
                            <div class="product-info">
                                <a href="#">Chau Thanh Phong</a>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
</footer>

```

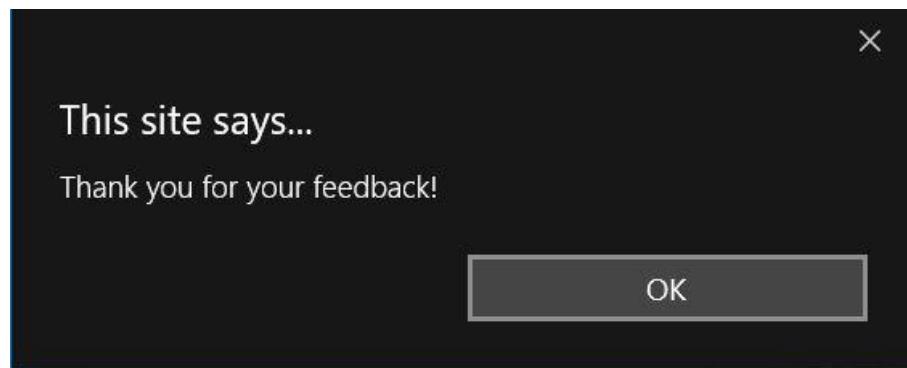
Figure

1.2. Contact Us Page: Show contact information of clinic & feedback form for user. (must login first to send feedback)



No	Name	Type	Validation	Event	Description	Status
1	Subject	Textbox	- Not null - Max length:50	Focus	Input feedback's subject	Enable
2	Message	Textbox	- Not null - Max length:500	Focus	Input feedback's content	Enable
3	Send Message	Button		Click	Click to send feedback	Enable
4	Map				Location of My Store	Enable

If send feedback successfully, a notification will appear, click to go back contact page:



Source code:

Controller:

The screenshot shows the Microsoft Visual Studio interface with the following details:

- Title Bar:** IceCream (Running) - Microsoft Visual Studio (Administrator)
- File Menu:** File Edit View Project Build Debug Team Tools Test Analyze Window Help
- Toolbars:** Standard, Debug, Task List, Solution Explorer, Properties, Task List, Status Bar.
- Status Bar:** Phong Chau PC, The content requires a new version of Internet Explorer, 80%, Error List, Output, Ln 35, Col 17, Ch 17, INS, EN, 9:16 AM, 08/08/2019.
- Solution Explorer:** Shows the project structure for 'IceCream' (2 projects). The 'Controllers' folder contains 'ContactController.cs' (selected), 'FAQController.cs', 'HomeController.cs', 'ProductController.cs', and 'ShoppingCartController.cs'. Other folders include 'Properties', 'References', 'App_Data', 'App_Start', 'bin', 'Content', 'css', 'fonts', 'fontS7', 'ImageLogo', 'Images', 'Models', 'Cart.cs', 'CustomerMeta.cs', 'CustomerModel.cs', 'IceCreamDB.dbml', 'ModelLecture.cs', 'Recipe.cs', 'RecipeData.dbml', and 'RecipeMetadata.cs'.
- Code Editor:** Displays the 'ContactController.cs' file content. The code handles GET and POST requests for contact form submissions, interacting with the 'IceCreamEntities1' database to add feedback and save changes.
- Error List:** Shows no errors or warnings.
- Output:** Shows the output from 'iisexpress.exe' during debugging, indicating module loading and assembly loading.

```

7  namespace IceCream.Controllers
8  {
9      public class ContactController : Controller
10     {
11         private IceCreamEntities1 db = new IceCreamEntities1();
12         // GET: Contact
13         public ActionResult Contact()
14         {
15             return View();
16         }
17         [HttpPost]
18         public ActionResult Contact([Bind(Include = "FeedbackId,Subject,Content,CustomerId")] FeedBack feedback)
19         {
20             if (Session["id"] == null)
21             {
22                 feedback.CustomerId = null;
23             }
24             else
25             {
26                 feedback.CustomerId = Convert.ToInt32(Session["id"]);
27             }
28             if (ModelState.IsValid)
29             {
30                 db.FeedBack.Add(feedback);
31                 db.SaveChanges();
32                 ViewBag.msg = "Thank You For Your Contact.";
33             }
34             return View();
35         }
36     }
37 }
38 
```

View:

```
}

<!-- ##### Subscribe Area Start ##### -->
<section class="subscribe-newsletter-area" style="background-image: url('../images/subscribe.png');">
    <div class="container">
        <div class="row align-items-center justify-content-between">
            <div class="col-12 col-lg-5">
                <!-- Section Heading -->
                <div class="section-heading mb-0">
                    <h2>Contact</h2>
                    <p>Subscribe to our newsletter and get free shipping</p>
                </div>
            </div>
            <div class="col-12 col-lg-6">
            </div>
        </div>
    </div>

    <!-- Subscribe Side Thumbnail -->

</section>
<!-- ##### Subscribe Area End ##### -->
<!-- ##### Contact Area Start ##### -->
<section class="contact-area section-padding-100-0">
    <div class="container">
        <div class="row align-items-center justify-content-between">
            <div class="col-12 col-lg-5">
                <!-- Section Heading -->
                <div class="section-heading">
                    <h2>GET IN TOUCH</h2>

                    <p class="text-success">Send us a message, we will call back later</p>
                </div>
                <!-- Contact Form Area -->
                <div class="contact-form-area mb-100">

                    @using (Html.BeginForm("Contact", "Contact", FormMethod.Post))
                    {

                        <div class="row">
                            <div class="col-12 col-sm-6">
                                <div class="form-group">
                                    @Html.ValidationMessageFor(model => model.CustomerId, "", new { @class = "text-danger" })
                                    @Html.HiddenFor(model => model.CustomerId, new { htmlAttributes = new { @class = "form-control" } })
                                </div>
                            </div>
                            @Html.ValidationSummary(true)
                        <div class="col-12">
```

1.3. Login Page: Login form for user

Phone: +101-1231-1231

1 **2** Sign In Register **3** **4**

Good Lickin'

1 Home Book Dessert Free Recipes Contact Us FAQ

4 Email: lam123@gmail.com

5 Password:

6 Submit

New Customer?
Create your ice-cream parlor shop account!!

No	Name	Type	Validation	Event	Description	Status
1	Navigation Bar	Hyperlink		Click	Choose & click to go to sub pages	Enable
2	Login	Hyperlink		Click	Click to go to login page	Enable
3	Register	Hyperlink		Click	Click to go to register page	Enable
4	Cart	Hyperlink		Click	Click to go to cart page (must be login first)	Enable
5	Email	Textbox	- Not null - Min length:5 - Max length:20	Focus	Input Email	Enable
6	Password	Password	- Not null - Min length:6 - Max length:20	Focus	Input password	Enable
7	Login	Button		Click	Click to login	Enable

Source code:

Controller:

The screenshot shows the Microsoft Visual Studio interface with the following details:

- Title Bar:** IceCream (Running) - Microsoft Visual Studio (Administrator)
- File Menu:** File, Edit, View, Project, Build, Debug, Team, Tools, Test, Analyze, Window, Help
- Toolbar:** Standard toolbar with icons for New, Open, Save, Print, Cut, Copy, Paste, Find, Replace, etc.
- Code Editor:** Displays the `ContactController.cs` file. The code implements a login functionality for a customer. It checks if email and password are provided, retrieves a customer from the database, and compares the provided password with the hashed one. It also handles different customer roles (1 or 2) by adding a year to the date registered and comparing it with the current date.
- Solution Explorer:** Shows the project structure for "IceCream" which contains two projects: "IceCream" and "IceCream.Web". The "IceCream" project includes files like Properties, References, App_Start, bin, Content, Controllers (ContactController.cs, FAQController.cs, HomeController.cs, ProductController.cs, ShoppingCartController.cs), Models (Cart.cs, CustomerMeta.cs, CustomerModel.cs, IceCreamDB.dbml, Model1L.edmx, Recipe.cs, RecipeData.dbml, RecipeMetadata.cs, Status.cs, UserOrderDetail.cs), and various CSS, fonts, and images folders.
- Status Bar:** Shows the status of the solution ("Ready"), current line (Ln 233), column (Col 57), character (Ch 57), and mode (INS). It also displays the date and time (08/08/2019, 9:18 AM).

View:

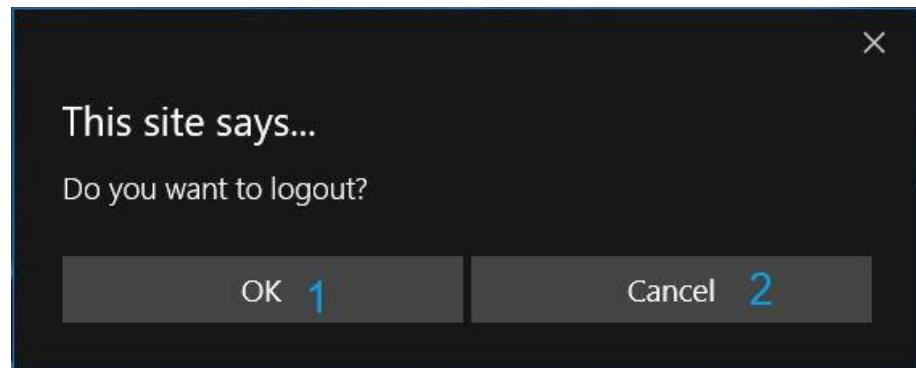
```

<div class="form-horizontal" style="border:groove;border-color:aqua;margin-left:200px;padding:20px;margin-top:80px;display:flex;">
    <div style="padding-right:70px; border-right:groove; border-color:aqua; width:50%; padding-left:20px;">
        <h2 style="margin-left:50px; padding-bottom:20px; font-family:HP-Ashley; text-align:center; font-size:40px">Login</h2>
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
        <div class="form-group">
            <span class="label-input100">Email</span>
            @Html.EditorFor(model => model.CustomerEmail, new { htmlAttributes = new { @class = "form-control", @autofocus = "true" } })
        </div>
        <div class="form-group">
            <span class="label-input100">Password</span>
            @Html.EditorFor(model => model.CustomerPassword, new { htmlAttributes = new { @class = "form-control" } })
        </div>
        <div class="form-group">
            <div class="col-md-9">
                @Html.ValidationMessageFor(model => model.CustomerPassword, "", new { @class = "text-danger" })
            </div>
        </div>
        <div class="form-group">
            <div class="col-md-9">
                <input type="submit" value="Submit" class="btn btn-primary" style="padding-left:50px;padding-right:50px;"/>
            </div>
        </div>
    </div>
    <div style="margin-left:125px; margin-top:75px;">
        <h2>New Customer?</h2>
        <a href='@Url.Action("Register","Home")' style="text-decoration:none; margin-left:-45px; font-size:17px;">Create your ice-cream parlor shop account!!</a>
    </div>
</div>
}

```

Figure 1

1.4. Logout: notification will appear when admin click logout button



No	Name	Type	Validation	Event	Description	Status
1	btnOk	Button		Click	Click to logout	Enable
2	btnCancel	Button		Click	Click to return	Enable

Source code:

Controller:

```

public ActionResult Logout()
{
    Session["email"] = null;
    Session["password"] = null;
    Session["role"] = null;
    Session["id"] = null;
    return RedirectToAction("Index");
}

```

1.5. Register Page: Register form for user if they haven't account.

Create Account

Customer Email
1 lam123@gmail.com

Password
2

Name
3 Ex: Mai Hoang Minh Tien

Phone
4 Ex: 0347669551

Address
5 Ex: 0347669551

6 Check this box if you want to register to become a member to see more flavor recipes!!

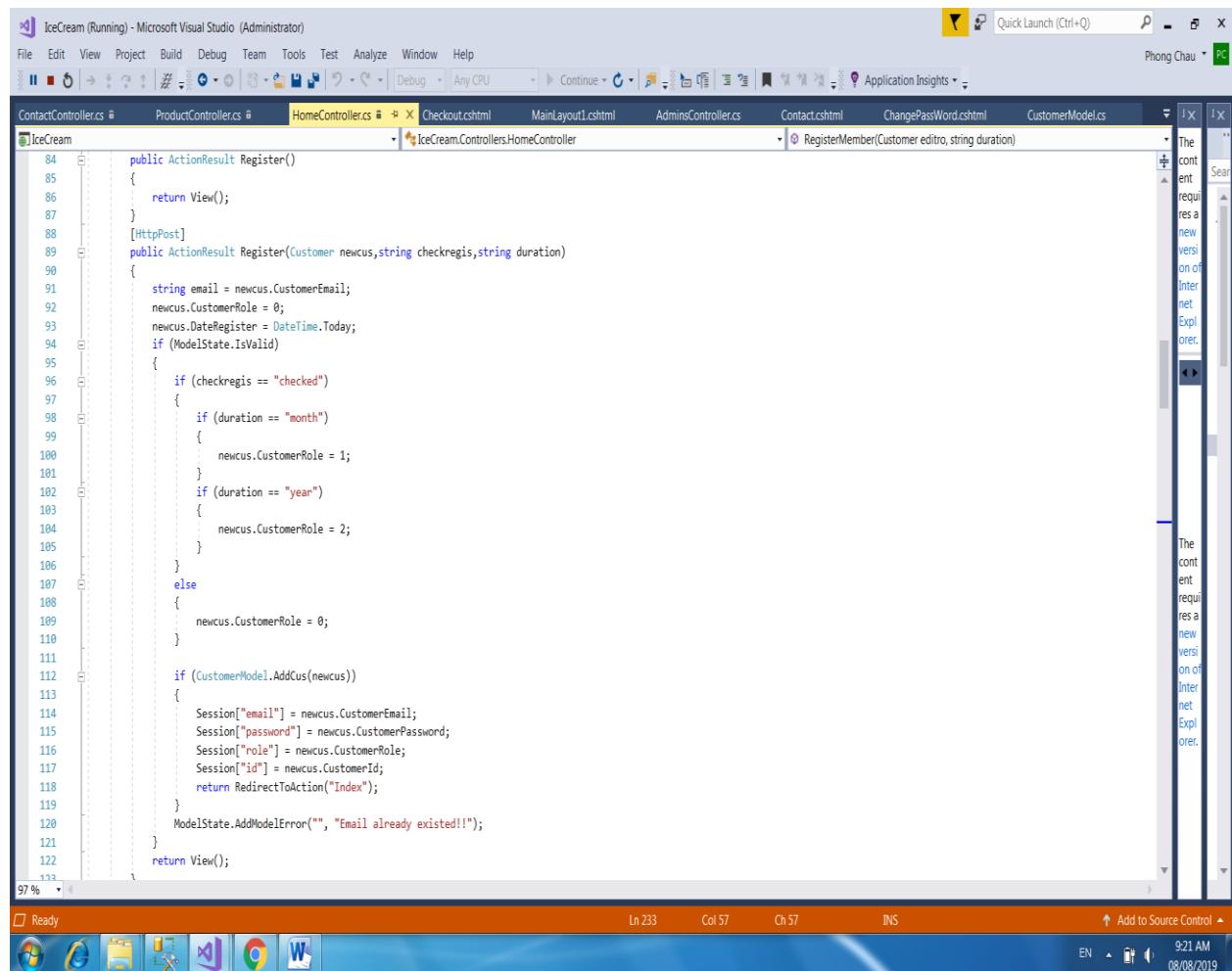
Create

No	Name	Type	Validation	Event	Description	Status
1	CustomerEmail	Textbox	- Not null - Min length:5 - Max length:20	Focus	Show your Email	Enable
2	Password	Password	- Not null - Min length:6 - Max length:20	Focus	Input password	Enable
3	Name	Textbox	- Not null - Max length:50	Focus	Input name	Enable
4	Phone Number	Textbox	- Not null - Max length:15	Focus	Input phone number	Enable
5	Address	Textbox	- Not null - Max length:50	Focus	Input address	Enable

6	CheckBox	Radio	Focus	Check value member if user want	Enable
---	----------	-------	-------	---------------------------------	--------

Source code:

Controller:



The screenshot shows the Microsoft Visual Studio interface with the project 'IceCream' open. The 'HomeController.cs' file is the active code editor. The code implements two methods: 'Register()' and 'Register(Customer newcus, string checkregis, string duration)'. The 'Register()' method returns a view. The 'Register()' method handles an HTTP POST request, setting the customer's role based on the checked radio button ('month' or 'year') and adding the customer to the database. If the email already exists, it adds an error message to the ModelState.

```

84     public ActionResult Register()
85     {
86         return View();
87     }
88     [HttpPost]
89     public ActionResult Register(Customer newcus, string checkregis, string duration)
90     {
91         string email = newcus.CustomerEmail;
92         newcus.CustomerRole = 0;
93         newcus.DateRegister = DateTime.Today;
94         if (ModelState.IsValid)
95         {
96             if (checkregis == "checked")
97             {
98                 if (duration == "month")
99                 {
100                     newcus.CustomerRole = 1;
101                 }
102                 if (duration == "year")
103                 {
104                     newcus.CustomerRole = 2;
105                 }
106             }
107             else
108             {
109                 newcus.CustomerRole = 0;
110             }
111             if (CustomerModel.AddCus(newcus))
112             {
113                 Session["email"] = newcus.CustomerEmail;
114                 Session["password"] = newcus.CustomerPassword;
115                 Session["role"] = newcus.CustomerRole;
116                 Session["id"] = newcus.CustomerId;
117                 return RedirectToAction("Index");
118             }
119             ModelState.AddModelError("", "Email already existed!!!");
120         }
121     }
122     return View();

```

View:

```
3 <div class="form-horizontal" style="border:groove; border-color:aqua; margin-left:150px; margin-right:100px; padding:20px; margin-top:80px;">
4   <h2 style="text-align:center; padding-bottom:20px; font-family:HP-Ashley; font-size:40px;">Create Account</h2>
5   <div style="margin-left:75px; padding-left:10px; padding-right:50px;">
6     <div class="form-group">
7       <span class="label-input100">Customer Email</span>
8       @Html.EditorFor(model => model.CustomerEmail, new { htmlAttributes = new { @class = "form-control", @placeholder = "Ex: kenzytien35@gmail.com" } })
9       <div class="col-md-9">
10
11         @Html.ValidationMessageFor(model => model.CustomerEmail, "", new { @class = "text-danger" })
12         @Html.ValidationSummary(true, "", new { @class = "text-danger" })
13       </div>
14     </div>
15
16     <div class="form-group">
17       <span class="label-input100">Password</span>
18       @Html.EditorFor(model => model.CustomerPassword, new { htmlAttributes = new { @class = "form-control" } })
19
20       <div class="col-md-9">
21
22         @Html.ValidationMessageFor(model => model.CustomerPassword, "", new { @class = "text-danger" })
23       </div>
24     </div>
25
26     <div class="form-group">
27       <span class="label-input100">Name</span>
28       @Html.EditorFor(model => model.CustomerName, new { htmlAttributes = new { @class = "form-control", @placeholder = "Ex: Mai Hoang Minh Tien" } })
29
30       <div class="col-md-9">
31
32         @Html.ValidationMessageFor(model => model.CustomerName, "", new { @class = "text-danger" })
33       </div>
34     </div>
35
36     <div class="form-group">
37       <span class="label-input100">Phone</span>
38       @Html.EditorFor(model => model.CustomerPhone, new { htmlAttributes = new { @class = "form-control", @placeholder = "Ex: 0347669551" } })
39
40       <div class="col-md-9">
41
42         @Html.ValidationMessageFor(model => model.CustomerPhone, "", new { @class = "text-danger" })
43       </div>
44     </div>
45   </div>
```

1.6. Change Personal Information Page: User can change their information here.

No	Name	Type	Validation	Event	Description	Status
1	Customer Email	Textbox (Read only)		Focus	Email	Disable
2	Full name	Textbox	- Not null - Max length:50	Focus	Input to change full name	Enable
3	Phone Number	Textbox	- Not null - Max length:15	Focus	Input to change phone number	Enable
4	Address	Textbox	- Not null - Max length:50	Focus	Input to change address	Enable
5	Profile	Textbox		Focus	Tab to change page	Enable
6	Information Account	Hyperlink		Click	Go to Profile page(must be login)	Enable
7	Register Member	Hyperlink		Click	Go to Register page (must be login)	Enable
8	Change Password	Hyperlink		Click	Go to Change Password Page	Enable
9	Logout	Hyperlink		Click	Sign out the page	Enable
10	Update	Button		Click	Click to update information	Enable

Source code:**Controller:**

```

public static bool EditProfile(Customer editpro)
{
    var b = GetCustomer(editpro.CustomerEmail);
    if (b != null)
    {
        b.CustomerName = editpro.CustomerName;
        b.CustomerPhone = editpro.CustomerPhone;
        b.CustomerAddress = editpro.CustomerAddress;
        db.SubmitChanges();
        return true;
    }
    return false;
}

```

```

129     Session["id"] = null;
130     return RedirectToAction("Index");
131 }
132 public ActionResult Profiles(string email)
133 {
134     if (Session["email"] != null)
135     {
136         email = Session["email"].ToString();
137         return View(CustomerModel.GetCustomer(email));
138     }
139     return RedirectToAction("Login");
140 }
141 [HttpPost]
142 [ValidateAntiForgeryToken]
143 public ActionResult Profiles(Customer editpro)
144 {
145     ModelState.Remove("CustomerId");
146     ModelState.Remove("CustomerRole");
147     ModelState.Remove("DateRegister");
148     ModelState.Remove("CustomerPassword");
149     if (ModelState.IsValid)
150     {
151         if (CustomerModel.EditProfile(editpro))
152         {
153             ModelState.AddModelError("", "Update Successful!!!");
154             return View();
155         }
156         ModelState.AddModelError("", "can't update");
157     }
158     return View();
159 }
160 
161 public ActionResult ChangePassWord()
162 {
163     return View();
164 }

```

View:

```

<div id="title-profile">
    <div id="title-profile-design">
        <h2 style="text-align:center; margin-left:-25px">Profile</h2>
    </div>
</div>

<div id="menu-profile-style">
    <div id="menu-profile">
        <ul>
            <li style="background-color:aqua; border-bottom:groove aqua;"><a href='@Url.Action("Profiles", "Home")'>Information Account</a></li>
            <li><a href='@Url.Action("ChangePassWord", "Home")'>Change Password</a></li>
            <li><a href='@Url.Action("RegisterMember", "Home")'>Register Member</a></li>
            <li style="border-bottom:none;"><a href='@Url.Action("Logout", "Home")'>Logout</a></li>
        </ul>
    </div>
<div id="form-profile">
    <div id="fix-form-profile">
        <h2 style="text-align:center; margin-left:-110px; font-family:HP-Ashley; padding-bottom:20px">Information Account</h2>

        @using (Html.BeginForm())
        {
            @Html.AntiForgeryToken()

            <div class="form-horizontal" style="padding-right:50px">

                @Html.ValidationSummary(true, "", new { @class = "text-danger" })
                @Html.HiddenFor(model => model.CustomerId)

                <div class="wrap-input100 rsl-wrap-input100 validate-input">
                    <span class="label-input100">Customer Email</span>
                    @Html.EditorFor(model => model.CustomerEmail, new { htmlAttributes = new { @class = "form-control", @readonly = "true" } })
                    @Html.ValidationMessageFor(model => model.CustomerEmail, "", new { @class = "text-danger" })
                    <span class="focus-input100"></span>
                </div>

                <div class="wrap-input100 rsl-wrap-input100 validate-input">
                    <span class="label-input100">Full name</span>
                    @Html.EditorFor(model => model.CustomerName, new { htmlAttributes = new { @class = "form-control" } })
                    @Html.ValidationMessageFor(model => model.CustomerName, "", new { @class = "text-danger" })
                </div>
            </div>
        }
    </div>
</div>

```

1.7. Update Password Page: User can change their Password here.

The screenshot shows a user interface for changing a password. On the left, there's a sidebar with a 'Profile' tab and other options like 'Information Account', 'Change Password', 'Register Member', and 'Logout'. The 'Change Password' option is highlighted. The main content area has a light blue background. It contains three input fields labeled 1, 2, and 3: 'Old Password', 'New Password', and 'Re-Password'. At the bottom right is a yellow 'Save' button with the number 4 next to it.

No	Name	Type	Validation	Event	Description	Status
1	Old Password	Old Password	- Not null - Min length:6 - Max length:20	Focus	Input the old password	Enable
2	New Password	New Password	- Not null - Min length:6 - Max length:20	Focus	Input the new password	Enable
	RePassword	Re Password	Not null - Max length:50	Focus	reinput the new password	Enable
3	Save	Button	-	Click	Click to update information	Enable

Source code:

Controller:

```

        public static bool EditPassword(Customer editpass)
    {
        var b = GetCustomer(editpass.CustomerEmail);
        if (b != null)
        {
            b.CustomerPassword = editpass.CustomerPassword;
            db.SubmitChanges();
            return true;
        }
        return false;
    }

    public ActionResult ChangePassWord()
    {
        if (Session["email"] != null)
        {
            return View();
        }
        return RedirectToAction("Login");
    }

    [HttpPost]
    [ValidateAntiForgeryToken]
    public ActionResult ChangePassWord(Customer editpass, string newPassword, string repassword)
    {
        ModelState.Remove("CustomerId");
        ModelState.Remove("CustomerRole");
        ModelState.Remove("DateRegister");
        ModelState.Remove("CustomerName");
        ModelState.Remove("CustomerAddress");
        ModelState.Remove("CustomerPhone");
        ModelState.Remove("CustomerEmail");
        editpass.CustomerEmail = Session["email"].ToString();
        if (newPassword != repassword)
        {
            ModelState.AddModelError("", "Incorrect re-password");
        }
        else
        {
            if (ModelState.IsValid)
            {
                if (editpass.CustomerPassword == Session["password"].ToString())
                {

                    editpass.CustomerPassword = newPassword;
                    if (CustomerModel.EditPassword(editpass))
                    {
                        Session["password"] = newPassword;
                        ModelState.AddModelError("", "Update Successful!!!");
                        return View();
                    }
                    ModelState.AddModelError("", "Incorrect Old Password!!!");
                }
            }
        }
    }
}

```

View:

```

<div id="menu-profile-style">
    <div id="menu-profile">
        <ul>
            <li><a href='@Url.Action("Profiles", "Home")'>Information Account</a></li>
            <li style="background-color:aqua; border-bottom:groove aqua;"><a href='@Url.Action("ChangePassword", "Home")'>Change Password</a></li>
            <li href='@Url.Action("RegisterMember", "Home")'>Register Member</a></li>
            <li style="border-bottom:none;"><a href='@Url.Action("Logout", "Home")'>Logout</a></li>
        </ul>
    </div>
    <div id="form-profile">

        <div id="fix-form-profile">
            <h2 style="text-align:center; margin-left:-110px; font-family:HP-Ashley; padding-bottom:20px">Information Account</h2>

            @using (Html.BeginForm())
            {
                @Html.AntiForgeryToken()

                <div class="form-horizontal">

                    @Html.ValidationSummary(true, "", new { @class = "text-danger" })

                    <div class="form-group">
                        <div class="col-md-4">
                            <span class="label-input100">Old Password</span>
                        </div>

                        <div class="col-md-6">
                            @Html.EditorFor(model => model.CustomerPassword, new { htmlAttributes = new { @class = "form-control", @required = "" } })
                            @Html.ValidationMessageFor(model => model.CustomerPassword, "", new { @class = "text-danger" })
                            <span id="erro-oldpass" style="color:red;"></span>
                        </div>
                    </div>

                    <div class="form-group">
                        <div class="col-md-4">
                            <span class="label-input100">New Password</span>
                        </div>
                        <div class="col-md-6">
                            <input type="password" class="form-control" id="newpass" required="" name="newpassword" />
                            <span id="erro-newpass" style="color:red;"></span>
                        </div>
                    </div>
                </div>
            
```

1.8. Register Member Page: If the user is not register ,user can fill value member is this profiles.

Profile

Information Account

Change Password

Register Member

Logout

Register Member

1 Using Card

2 Duration: 15\$/month 150\$/year

3 Card Number: Ex: 1234 4568 4571 1246

4 Name printed on the card: Ex: Mai Hoang Minh Tien

5 Security code: Ex: 123

4321 123

Save 6

No	Name	Type	Validation	Event	Description	Status
1	Using Card	img		Focus		Enable
2	Duration	Radio Btn		Focus	Choose member price	Enable
3	Name Printed on the card	Textbox		Focus	Input Full Name	Enable
4	Name Printed on the card	TextBox		Focus	Input Name On Card	Enable
5	Security Code	Textbox		Focus	Input the number	Enable

Source code:

Controller:

```

public ActionResult RegisterMember()
{
    if (Session["email"] != null)
    {
        return View();
    }
    return RedirectToAction("Login");
}
[HttpPost]
public ActionResult RegisterMember(Customer editro, string duration)
{
    editro.CustomerEmail = Session["email"].ToString();
    if (duration == "month")
    {
        editro.CustomerRole = 1;
        Session["role"] = 1;
    }
    if (duration == "year")
    {
        editro.CustomerRole = 2;
        Session["role"] = 1;
    }
    if (CustomerModel.editrole(editro))
    {
        return RedirectToAction("RegisterMember");
    }
    ModelState.AddModelError("", "Can't Register");
    return View();
}

```

View:

```

<div id="form-profile">
    <div id="fix-form-profile">
        <h2 style="text-align:center; margin-left:-110px; font-family:HP-Ashley; padding-bottom:20px">Register Member</h2>
        @using (Html.BeginForm())
        {
            @Html.AntiForgeryToken()

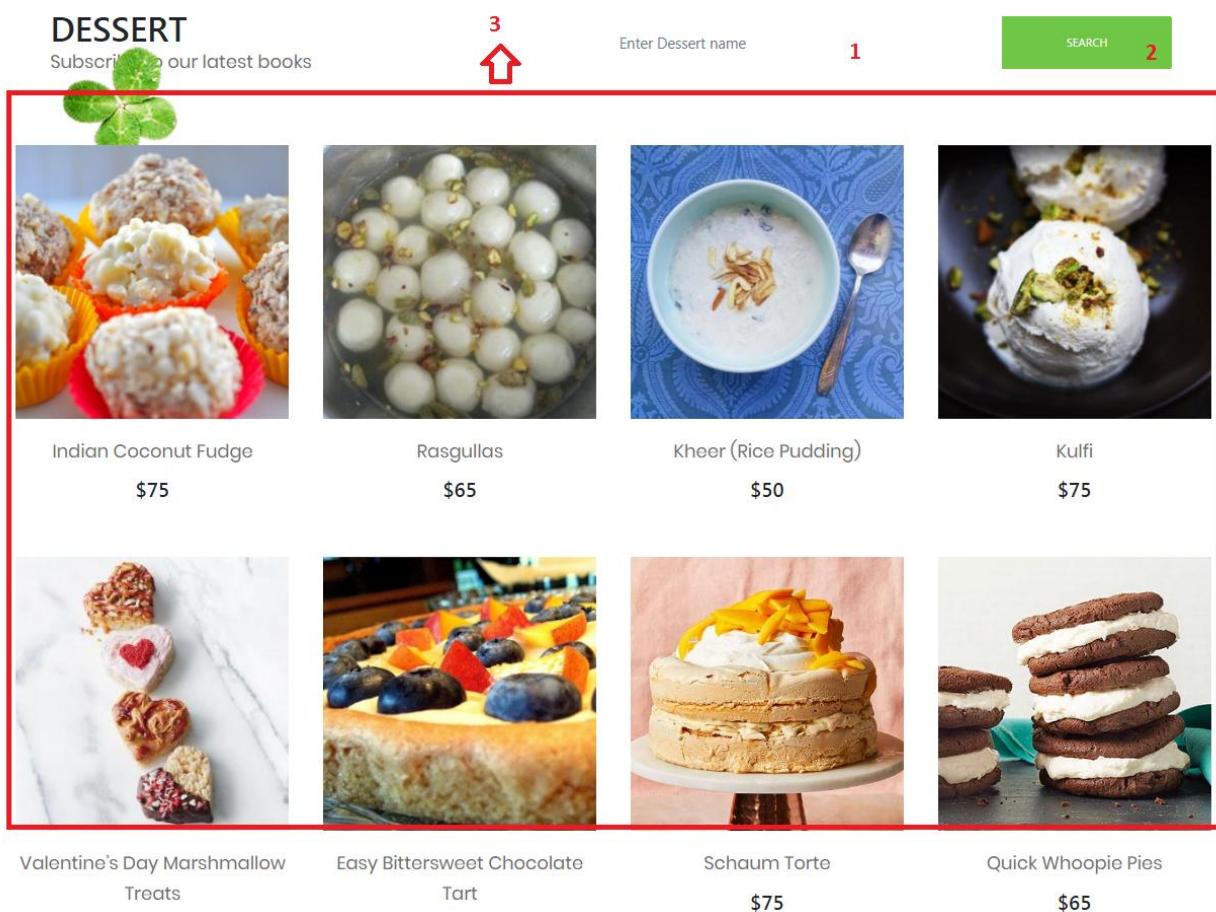
            <div class="form-horizontal">
                @if (sessrole != 0)
                {
                    <h2 style="font-family:'UTM Ong Do Gia'; font-size:115px; color:lawngreen;">You already a member!!!</h2>
                }
                else
                {
                    @Html.ValidationSummary(true, "", new { @class = "text-danger" })
                    @Html.HiddenFor(model => model.CustomerId)

                    <div class="form-group">
                        <div class="control-label col-md-3"><span class="label-input100">Using Card</span></div>
                        <div class="col-md-4">
                            
                            
                            
                        </div>
                    </div>
                    <div class="form-group">
                        <div class="control-label col-md-3"><span class="label-input100">Duration: </span></div>
                        <div class="col-md-9" style="margin-top:5px;">
                            <input type="radio" name="duration" checked="" value="month" /><span class="label-input100">15$/month</span>
                            <input type="radio" name="duration" value="year" /><span class="label-input100">150$/year</span>
                        </div>
                    </div>

                    <div class="form-group">
                        <div class="control-label col-md-3"><span class="label-input100">Card Number:</span> </div>
                        <div class="col-md-9">
                            <input type="text" class="form-control" id="cardnum" name="cardnumber" required="" pattern="[0-9]{12}" placeholder="Ex: 1234 4568 4571 1246" />
                        </div>
                    </div>
                    <div class="form-group">
                        <div class="control-label col-md-3"><span class="label-input100">Name printed on the card: </span></div>

```

1.9. Book Page: This page display list of medicines, user can search item with search bar.



No	Name	Type	Validation	Event	Description	Status
1	txtSearch	Textbox		Focus	Input medicine name to search	Enable
2	btnSearch	Button		Click	Click to search Book	Enable
3	btnAdd to Cart	Button		Click	Click to search medicine	Enable

Source code:

Controller:

The screenshot shows the Microsoft Visual Studio interface with the following details:

- Title Bar:** "IceCream (Running) - Microsoft Visual Studio (Administrator)"
- Toolbar:** Standard Visual Studio toolbar with icons for file operations, search, and navigation.
- Code Editor:** The main window displays the `ProductController.cs` file. The code implements a controller for managing products, including methods for listing products by category and searching by name. It uses Entity Framework (EF) to interact with the database.
- Solution Explorer:** Shows the project structure for "IceCream". It includes files like `Cart.cs`, `CustomerMe.cs`, `IceCreamDB.edmx`, `ModelLeden.cs`, `Recipe.cs`, `RecipeData.cs`, `RecipeMeta.cs`, `Status.cs`, `UserOrderDe.cs`, and various CSS, fonts, and image files.
- Status Bar:** Shows the status "Ready" and other system information like date and time.

View:

```

        </div>
    </div>
</div>

<!-- Subscribe Side Thumbnail --&gt;
&lt;div class="subscribe-side-thumb wow fadeInUp" data-wow-delay="500ms"&gt;
    &lt;img class="first-img" src "~/images/leaf.png" /&gt;
&lt;/div&gt;
&lt;/section&gt;
&lt;!-- ##### Subscribe Area End ##### --&gt;
&lt;!-- Single Product Area --&gt;
&lt;div&gt;
    @foreach (var item in Model)
    {
        &lt;div class="col-xs-12 col-lg-3"&gt;
            &lt;div class="single-product-area mb-50 wow fadeInUp" data-wow-delay="100ms"&gt;
                &lt;!-- Product Image --&gt;
                &lt;div class="product-img"&gt;
                    &lt;a href="@Url.Action("ProductDetails","Product",new { id=item.ProductId},null)" @*class="Show-Details"*@
                        &lt;img src="@item.Image" /&gt;
                    &lt;/a&gt;
                &lt;!-- Product Tag --&gt;

                &lt;div class="product-meta d-flex"&gt;
                    &lt;a href="#" class="wishlist-btn"&gt;&lt;i class="icon_heart_alt"&gt;Like&lt;/i&gt;&lt;/a&gt;
                    &lt;a href="@Url.Action("OrderNow","ShoppingCart",new { id=item.ProductId},null)" class="add-to-cart-btn"&gt;Add to cart&lt;/a&gt;
                    &lt;a href="#" class="compare-btn"&gt;&lt;i class="arrow_left-right_alt"&gt;Buy&lt;/i&gt;&lt;/a&gt;
                &lt;/div&gt;
            &lt;/div&gt;
            &lt;!-- Product Info --&gt;
            &lt;div class="product-info mt-15 text-center"&gt;
                &lt;h6&gt;
                    &lt;p&gt;@item.Name&lt;/p&gt;
                &lt;/h6&gt;
                &lt;h6&gt;$@item.Price&lt;/h6&gt;
            &lt;/div&gt;
        &lt;/div&gt;
    }
</pre>

```

1.10. Product Details Page: This page display information of a product. User also can add this product to cart



No	Name	Type	Validation	Event	Description	Status
1	Image	Image			Product's image	Enable
2	Name	Textbox (Read only)			Product's name	Enable
3	Price	Textbox (Read only)			Product's price	Enable
4	Description	Textbox (Read only)			Product's description	Enable
5	Add to Cart	Button		Click	Click to add product to your cart (must be login first)	Enable

Source code:

Controller:

```

    public ActionResult ProductDetails(int id)
    {
        var model = db.Product.Find(id);
        //to take id from product details view
        Session["IdDetail"] = model.ProductId;
        if (model != null)
        {
            return View(model);
        }
        return View();
    }
}

```

View:

```
<div class="container-contact100">
    <div class="pro-de">
        <div class="row">
            <div class="col-lg-4">
                
            </div>
            <div class="col-lg-8">
                <div class="testi_text">
                    <h4>@Html.DisplayFor(model => model.Name)</h4>
                    <h5>$@Html.DisplayFor(model => model.Price)</h5>
                    <p>@Html.DisplayFor(model => model.Description) </p>
                </div>
            </div>
            <br />
            <br />
            <form>
                <h3 class="contact100-form-btn">
                    <a href="@Url.Action("OrderNow","ShoppingCart",new { id=@Session["IdDetail"]},null)">
                        Add to cart
                        <i class="fa fa-long-arrow-right m-l-7" aria-hidden="true"></i>
                    </a>
                </h3>
            </form>
        </div>
    </div>
    <span class="contact100-more">
        For any question contact our 24/7 call center: <span class="contact100-more-highlight">+001 123 6889</span>
    </span>
</div>
```

1.11. Free Recipes: This page display list of recipes for your user don't use account.



Chocolate Ice-Cream Recipe

Combine sugar, milk, salt, and cocoa powder in a saucepan over medium heat.

[Read More...](#)

1



Strawberry Ice-Cream Recipe

Can you tell that I've been on an ice cream kick lately? If you

[Read More...](#)



Mango Ice-Cream Recipe

Combine the NESTLE MILKMAID Sweetened Condensed Milk, milk, and mango puree in a bowl.

[Read More...](#)



Butterscotch Ice Cream Recipe

Butter and brown sugar meld in this ice cream recipe to create delicious butterscotch

[Read More...](#)



Walnut Ice-Cream Recipe

The walnuts do add a nice crunch and nuttiness to this recipe

[Read More...](#)

2

[See More Recipes...](#)

No	Name	Type	Validation	Event	Description	Status
1	Read	Button		Click	Go to the video detail of recipes	Enable
2	See More Recipe	Button		Click	Click to Full Recipes(must be login)	Enable

Source code:

Controller:

```
public ActionResult Index()
{
    return View();
}

public IEnumerable<Recipe> GetRecipes()
{
    return db.Recipes;
}
public Recipe Get(int id)
{
    return db.Recipes.SingleOrDefault(item => item.Id == id);
}

RecipeDataContext db = new RecipeDataContext();

public ActionResult FreeRecipes()
{
    GetRecipes();
    return View(db.Recipes);
}
```

View:

```

@foreach (var item in Model)
{
    if (item.Status == "Free")
    {
        <div class="container">
            <div class="frameItem">
                <a href="@Url.Action("RecipeDetails", "Home", new { id = item.Id })">
                <div style="display: inline-block; class="contentItem">
                    <a href="@Url.Action("RecipeDetails", "Home", new { id = item.Id })" style="text-decoration: none;"><h5 class="hh5">
                    <p>@Html.DisplayFor(modelItem => item.Content)</p>
                    <a href="@Url.Action("RecipeDetails", "Home", new { id = item.Id })" style="text-decoration: none;">Read More...</a>
                </div>
            </div>

            <br /><br /><br />
        </div>
    }
}

<center>
    @if (Session["email"] == null)
    {
        <a href="@Url.Action("FullRecipes", "Home")" onclick="return confirm('Please create account or login to see full recipes')">
            <button style="margin: 50px; padding: 10px; padding-left: 150px; padding-right: 150px;
                border-radius: 10px; font-family: 'Satisfy', cursive; font-size: 30px; color: white;
                background: crimson;">
                See More Recipes...
            </button>
        </a>
    }
    else
    {
        if (int.Parse(Session["role"].ToString()) == 0)
        {
            <button style="margin: 50px; padding: 10px; padding-left: 150px; padding-right: 150px;
                border-radius: 10px; font-family: 'Satisfy', cursive; font-size: 30px; color: white;
                background: crimson;">
                See More Recipes...
            </button>
        }
    }
</center>

```

1.12. Full Recipes: This page display list of recipes for your user register a member as month or year



Chocolate Chip Ice-Cream Recipe

This ice cream version of the popular chocolate-fllecked stracciatella gelato...

[Read More...](#)

1



Coffee Ice-Cream Recipe

David has provided helpful advice for me on practically every ice cream recipe on this...

[Read More...](#)



Black Currant Ice-Cream Recipe

Looking for a new ice cream flavor? What about homemade blackcurrant ice cream?

[Read More...](#)



Cherry Ice-Cream Recipe

Place the cherry juice, milk, yogurt, and heavy cream into the bowl of a blender. Add the

[Read More...](#)



Vanilla And Strawberry (two in one) Ice-Cream Recipe

Crush 55 wafers finely; place crumbs in medium bowl

[Read More...](#)

No	Name	Type	Validation	Event	Description	Status
1	Read	Button		Click	Go to the video detail of recipes	Enable

Source code:**Controller:**

```

public ActionResult FullRecipes()
{
    if (Session["email"] == null)
    {
        return RedirectToAction("Login");
    }
    if (int.Parse(Session["role"].ToString()) == 0)
    {
        return RedirectToAction("RegisterMember");
    }
    GetRecipes();
    return View(db.Recipes);
}

```

View:

```

@foreach (var item in Model)
{
    if (item.Status == "VIP")
    {

        <div class="container">

            <div class="frameItem">
                <a href=@Url.Action("RecipeDetails", "Home", new { id = item.Id })><img src=@Html.DisplayFor(modelItem => item.Image) class="itemBanner" style="display: inline-block;"></a>
                <div style="display: inline-block;" class="contentItem">
                    <a href=@Url.Action("RecipeDetails", "Home", new { id = item.Id })" style="text-decoration: none;"><h5 class="hh5">@Html.DisplayFor(modelItem => item.Title)</h5></a>
                    <p>@Html.DisplayFor(modelItem => item.Content)</p>
                    <a href=@Url.Action("RecipeDetails", "Home", new { id = item.Id })" style="text-decoration: none;">Read More...</a>
                </div>
            </div>

            <br /><br /><br />
        </div>
    }
}

```

1.13. Detail of Recipes: This page display the way to cook this favourite ice cream.



No	Name	Type	Validation	Event	Description	Status
1	Video	Linkyt			Product's Video	Enable
2	Name	Textbox (Read only)			Recipes's name	Enable
3	btnRecipes	Button	Login before see full recipes		Click go to the full recipes page (must be login first)	Enable

Source code:**Controller:**

```
public ActionResult RecipeDetails(int id)
{
    return View(Get(id));
}
```

View:

```
h2 class="hh2">@Html.DisplayFor(model => model.Title)</h2>

div class="container">
@{
    string preFix = "https://www.youtube.com/embed/";
    string YT = Model.LinkYT;
    YT = YT.Substring(32);
    YT = preFix + YT;
}
<div>
<div>
    <center><iframe width="700" height="400" src=@YT frameborder="0" allow="accelerometer; autoplay; encrypted-media; gyroscope; picture-in-picture" allowfullscreen></iframe></center>
</div>
</div>

<!--Button See More Recipes-->
<center>
    <a href="@Url.Action("FullRecipes", "Home")">
        <button style="margin: 50px; padding: 10px; padding-left: 150px; padding-right: 150px;
border-radius: 10px; background: #ffe82b; text-decoration: none; font-family: 'Satisfy', cursive; font-size: 30px; color: white;">
            See Premium Recipes...
        </button>
    </a>
</center>

</div>

link rel="stylesheet" type="text/css" href="/Content/styledetails.css"
```

1.14. FAQ Page: This page display list of frequently of question.


[Home](#) [Book Dessert](#) [Free Recipes](#) [Contact Us](#) [FAQ](#) [Order History](#)

- What is the difference between Book Store and other ice cream?

1

Where do we start? Our lovingly prepared ice cream is a unique blend of the finest ingredients and lashings of fresh Scottish milk and double cream. And unlike other folk, we don't pump our ice cream full of air, cheap vegetable fat or unnecessary artificial additives. The result is an unrivalled range of deliciously quirky flavours that keep the awards flooding in. 2

- What's your best selling ice cream flavour?

- Do you have allergy information for your ice cream?

- How can I book a table?

- How can I contact your website?

- Are your ice creams suitable for vegetarians?

- How can you send your feedback?

No	Name	Type	Validation	Event	Description	Status
1	PanelTitle	Textbox		Focus	Question's Customer	Enable
2	PanelBody	Textbox		Focus	Answer from my website	Enable

Source code:

View:

```
<div class="panel-group" id="accordion" role="tablist" aria-multiselectable="true">
  <div class="panel panel-default">
    <div class="panel-heading" role="tab" id="headingOne">
      <h4 class="panel-title">
        <a role="button" data-toggle="collapse" data-parent="#accordion" href="#collapseOne" aria-expanded="true" aria-controls="collapseOne">
          What is the difference between Book Store and other ice cream?
        </a>
      </h4>
    </div>
    <div id="collapseOne" class="panel-collapse collapse in" role="tabpanel" aria-labelledby="headingOne">
      <div class="panel-body">
        <ul>
          <li>Where do we start? Our lovingly prepared ice cream is a unique blend of the finest ingredients and lashings of fresh Scottish milk and double cream. And unlike other fol</li>
        </ul>
      </div>
    </div>
  </div>
  <div class="panel panel-default">
    <div class="panel-heading" role="tab" id="headingTwo">
      <h4 class="panel-title">
        <a class="collapsed" role="button" data-toggle="collapse" data-parent="#accordion" href="#collapseTwo" aria-expanded="false" aria-controls="collapseTwo">
          What's your best selling ice cream flavour?
        </a>
      </h4>
    </div>
    <div id="collapseTwo" class="panel-collapse collapse" role="tabpanel" aria-labelledby="headingTwo">
      <div class="panel-body">
        <ul>
          <li>Double cream vanilla of course!</li>
        </ul>
      </div>
    </div>
  </div>
  <div class="panel panel-default">
    <div class="panel-heading" role="tab" id="headingThree">
      <h4 class="panel-title">
        <a class="collapsed" role="button" data-toggle="collapse" data-parent="#accordion" href="#collapseThree" aria-expanded="false" aria-controls="collapseThree">

```

1.15. Your Cart Page: Items which user had chosen will display here.

Shopping Cart

		Price 3	Quantity 4	Total 5
	1 THE EASIEST HOMEMADE POP Add Remove	\$75	Qty 1	\$75
	2 Easy Bittersweet Chocolate Tart Add 7 Remove 6	\$50	Qty 1	\$50
Continue Shopping 8				

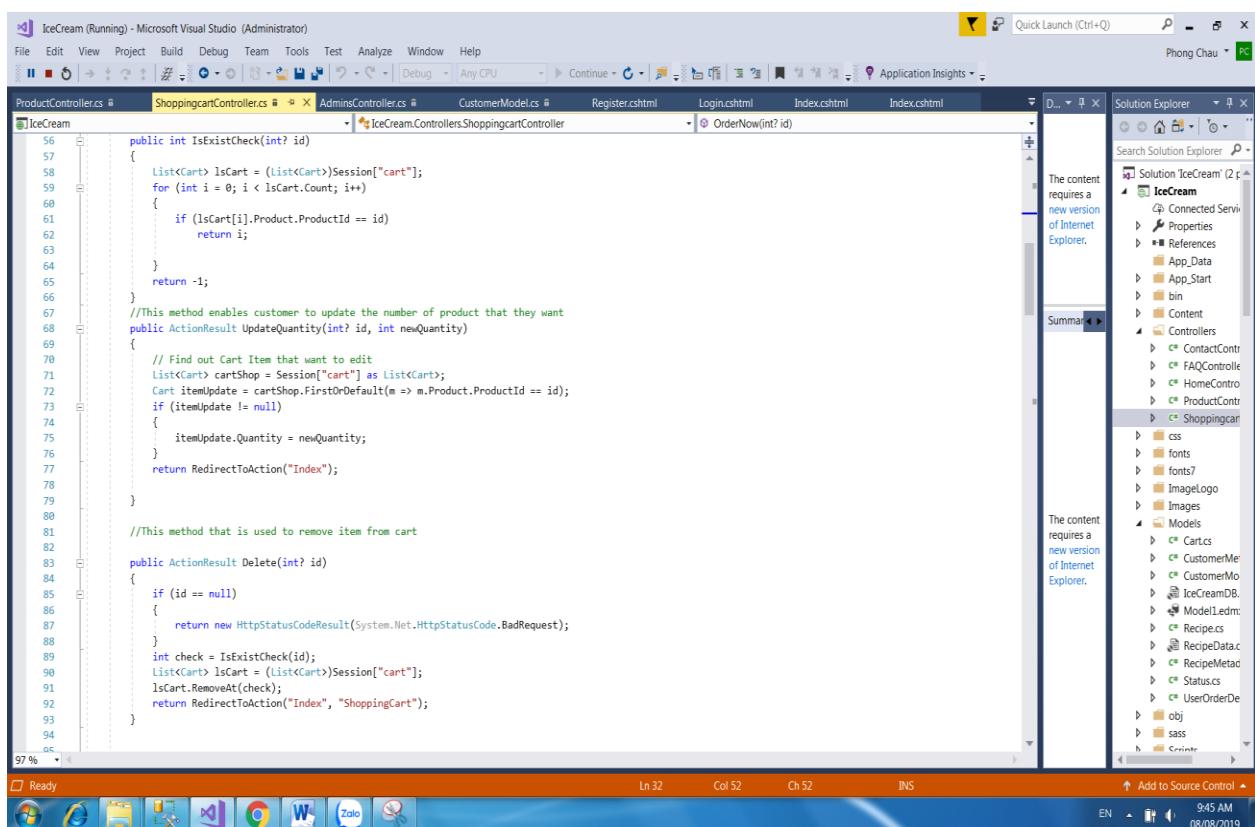
No	Name	Type	Validation	Event	Description	Status
1	Image	Image			Product's image	Enable
2	Name	Textbox (Read only)			Product's name	Enable
3	Price	Textbox (Read only)			Product's unit price	Enable
4	Quantity	Textbox		Focus	Product's quantity, if user want update quantity, type number of product want to buy and then press enter, the cart will reload & update	Enable
5	Subtotal	Textbox (Read only)			Product's subtotal price	Enable
6	Remove	Button		Click	Click to clear cart	Enable
7	Add	Button		Click	Click to update quantity	Enable
8	Continue shopping	Button		Click	Click to go back home page	Enable

Source code:**Controller:**

```
// GET: ShoppingCart
public ActionResult Index()
{
    return View();
}
public ActionResult OrderNow(int? id)
{
    if (id == null)
    {
        return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
    }
    if (Session["cart"] == null)
    {
        List<Cart> lsCart = new List<Cart>
        {
            new Cart(db.Product.Find(id),1)
        };
        Session["cart"] = lsCart;
    }
    else
    {
        List<Cart> lsCart = (List<Cart>)Session["cart"];
        int check = IsExistCheck(id);
        if (check == -1)
        {
            lsCart.Add(new Cart(db.Product.Find(id), 1));
        }
        else

            lsCart[check].Quantity++;
        Session["cart"] = lsCart;
    }
}

return View("Index");
}
```



View:

```
<div class="super_container">

    @if (Session["cart"] != null)
    {
        <div class="cart_info">
            <div class="container">
                <div class="row">
                    <div class="col">
                        <!-- Column Titles -->
                        <div class="cart_info_columns clearfix">
                            <div class="cart_info_col cart_info_col_product">Shopping Cart</div>
                            <div class="cart_info_col cart_info_col_price">Price</div>
                            <div class="cart_info_col cart_info_col_quantity">Quantity</div>
                            <div class="cart_info_col cart_info_col_total">Total</div>
                        </div>
                    </div>
                </div>
                <div class="row cart_items_row">
                    <div class="col">

                        <!-- Cart Item -->
                        @foreach (var item in (List<Cart>)Session["cart"])
                        {
                            <div class="cart_item d-flex flex-lg-row flex-column align-items-lg-center align-items-start justify-content-start">
                                <!-- Name -->
                                <div class="cart_item_product d-flex flex-row align-items-center justify-content-start">
                                    <div class="cart_item_image">
                                        <div></div>
                                    </div>
                                    <div class="cart_item_name_container">
                                        <div class="cart_item_name"><a href="#">@item.Product.Name</a></div>
                                        <div class="cart_item_edit"><a href="#">Add</a></div>
                                        @Html.ActionLink("Remove", "Delete", "ShoppingCart", new { id = item.Product.ProductId }, new { @onclick = "return confirm('Are you sure want to remove this item?')" })
                                    </div>
                                <div class="remove-btn">
                                    <div class="cart_item_name">
                                        @Html.ActionLink("Update", "UpdateQuantity", "ShoppingCart", new { id = item.Product.ProductId })@*
                                    </div>
                            </div>
                        }
                    </div>
                </div>
            </div>
        </div>
    }
</div>
```

1.16. Checkout page: This page show for user confirm payment.

The screenshot shows the 'Checkout Information' section (1) containing fields for Customer Name (2), Address (3), and Payment Option (4). To the right, the 'Cart total' section (10) displays the final info with Customer ID (1), Shipping (Free), and Total (125). Below these are options for gender (6), email (7), phone number (9), and address (8). A confirmation dialog (13) at the bottom asks if the user wants to checkout, with OK and Cancel buttons. A red box labeled '5' highlights the list of contact information.

Checkout Information

Select the one you want

Customer Name 2
Enter Name

Address 3
Enter Address Delivery

Payment Option 4
Credit Card ▾

Cart total

Final info

10	Customer ID	1
Shipping	Free	
11	Total	125

Order 12

- 6 Mr/Ms Chau Thanh Phong
- 7 Email phong2397@gmail.com
- 9 Phone number 0123456781
- 8 Address 331 Vung Tau

5

localhost:49409 says

Are you sure want to checkout?

13 OK Cancel

No	Name	Type	Validation	Event	Description	Status
1	User'Order	Textbox			Form Order	Enable
2	Customer name	Textbox	Not null Max length :30		User input into the name (both user register and no register)	Enable
3	Address	Textbox	Not null Max length :30		User input into the Address (both user register and no register)	Enable
4	Payment option	Select	Not null		User choose your payment to order(both user register and no register)	Enable
5	Inforamtion user	Textbox (Read only)			User's information(must be login)	Enable
6	User Name	Textbox (Read only)			User's Name	Enable

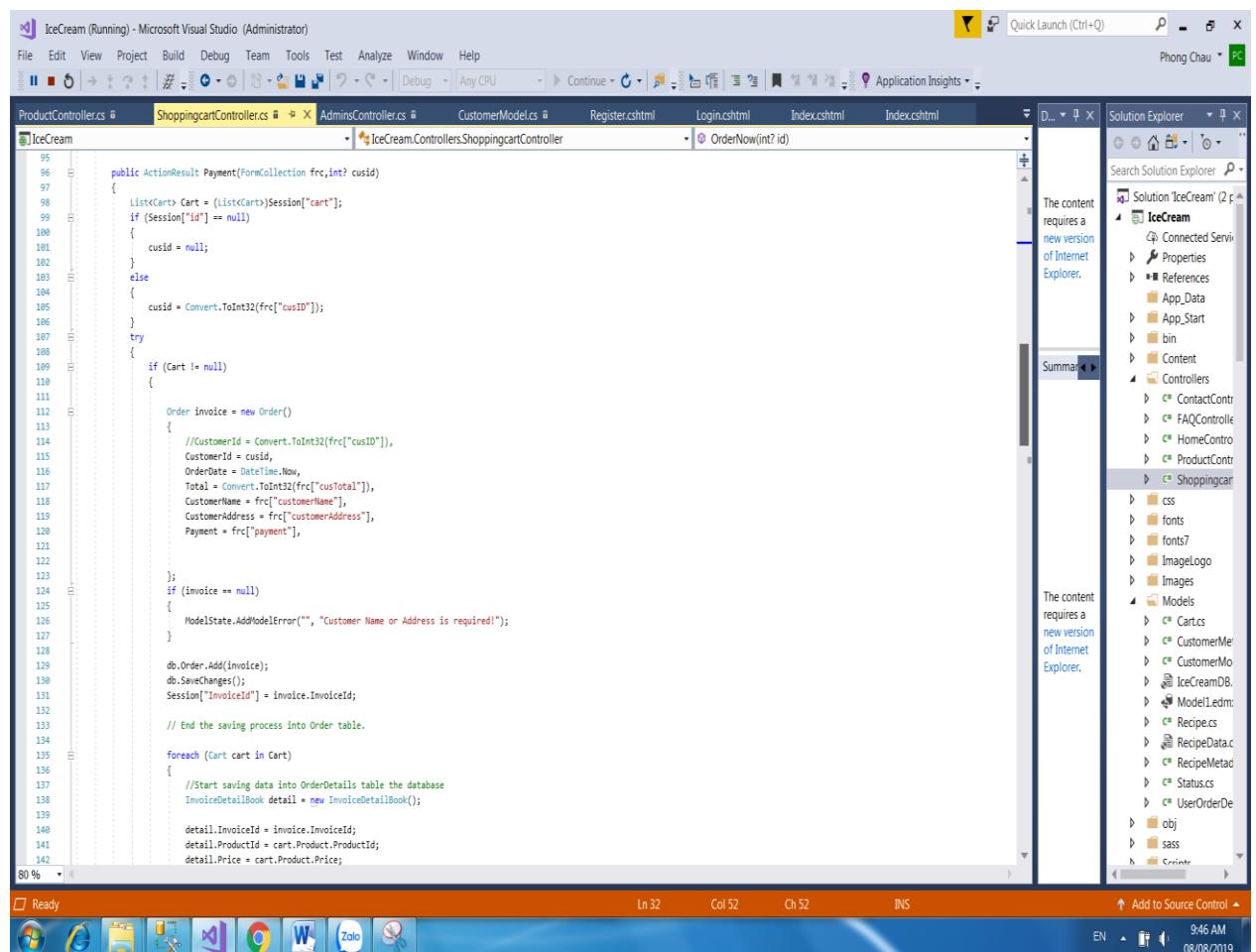
7	Email	Textbox (Read only)	User's Email	Enable
8	Address	Textbox (Read only)	User's Address	Enable
9	Phone Number	Textbox (Read only)	User's Phone	Enable
10	CustomerID	TextBox(Read only)	Focus Primary key of Customer	Enable
11	Total	TextBox(read only)	Focus Sum of quantity and price of product	Enable
12	Order	Button	Click Click to checkout order	Enable
13	Ok	Button	Click Click to confirm checkout	Enable

Source code:**Controller:**

```

public PartialViewResult Shipping()
{
    if (Session["email"] != null)
    {
        var user = Session["email"].ToString();
        var model = CustomerModel.GetCustomer(user);
        if (model != null)
        {
            return PartialView(model);
        }
    }
    return PartialView();
}

```



```

1 public ActionResult Payment(FormCollection frc,int? cusid)
2 {
3     List<Cart> Cart = (List<Cart>)Session["cart"];
4     if (Session["id"] == null)
5     {
6         cusid = null;
7     }
8     else
9     {
10        cusid = Convert.ToInt32(frc["cusID"]);
11    }
12    try
13    {
14        if (Cart != null)
15        {
16            Order invoice = new Order()
17            {
18                //CustomerId = Convert.ToInt32(frc["cusID"]),
19                CustomerId = cusid,
20                OrderDate = DateTime.Now,
21                Total = Convert.ToInt32(frc["cusTotal"]),
22                CustomerName = frc["customerName"],
23                CustomerAddress = frc["customerAddress"],
24                Payment = frc["payment"];
25            };
26            if (invoice == null)
27            {
28                ModelState.AddModelError("", "Customer Name or Address is required!");
29            }
30            db.Order.Add(invoice);
31            db.SaveChanges();
32            Session["InvoiceId"] = invoice.InvoiceId;
33        }
34        // End the saving process into Order table.
35        foreach (Cart cart in Cart)
36        {
37            //Start saving data into OrderDetails table the database
38            InvoiceDetailBook detail = new InvoiceDetailBook();
39            detail.InvoiceId = invoice.InvoiceId;
40            detail.ProductId = cart.Product.ProductId;
41            detail.Price = cart.Product.Price;
42        }
43    }
44 }

```

View:

```

@using (Html.BeginForm("Payment", "ShoppingCart", new { id = "customer-form" }, FormMethod.Post))
{
    if (Session["email"] != null)
    {
        if (Session["cart"] != null)
        {
            <form action="@Url.Action("Payment")" method="get">
                <div class="row row_extra">
                    <div class="col-lg-4">

                        <!-- Delivery -->
                        <div class="delivery">
                            <div class="section_title">Checkout Information</div>
                            <div class="section_subtitle">Select the one you want</div>
                            @Html.RenderAction("Checkout", "Shoppingcart");
                            @Html.RenderAction("Shipping", "Shoppingcart");
                        </div>
                    <!-- Coupon Code -->
                    @{
                        List<Cart> temp = (List<Cart>)Session["cart"];
                        var total = temp.Sum(x => x.Quantity * x.Product.Price);
                        // var total = String.Format("{0:N0}", temp.Sum(x => x.Quantity * x.Product.Price));
                    }
                </div>

                <div class="col-lg-6 offset-lg-2">
                    <div class="cart_total">
                        <div class="section_title">Cart total</div>
                        <div class="section_subtitle">Final info</div>
                        <div class="cart_total_container">
                            <ul>

                                <input type="text" name="cusID" class="cart_total_value ml-auto" hidden="" value="@Session["id"]" />

                            <li class="d-flex flex-row align-items-center justify-content-start">
                                <div class="cart_total_title">Shipping</div>
                                <div class="cart_total_value ml-auto">Free</div>
                            </li>
                        </ul>
                    </div>
                </div>
            </form>
        }
    }
}

```

1.17. InvoiceDetailBook: this page display information about order user just bought

Transaction is Completely **1**

2

The Detail Order

To provide the best to you

Order Id	33
Customer Name	Chau Thanh Phong
Order date	27-Thg7-2019
Customer Address	331 Vung Tau
Total	\$130

3

Detail:

No	Name	Type	Validation	Event	Description	Status
1	Result	Textbox (Read only)			Message to customer order complete	Enable
2	Order information	Textbox (Read only)			Order information	Enable
3	Details	Button			Click to view details of this order	Enable

Source code:

```

public ActionResult Order()
{
    int id = (int)Session["InvoiceId"];
    var model = db.Order.Find(id);
    return View(model);
}

public ActionResult Checkout()
{
    return View();
}

```

View:

```

<div class="col">
    <!-- Column Titles -->
    <div class="cart_info_columns clearfix">
        <div class="cart_info_col cart_info_col_product"><h3 style="font-weight:bolder">Transaction is Completely</h3></div>
    </div>
</div>

<div class="row row_extra">
    <div class="container">

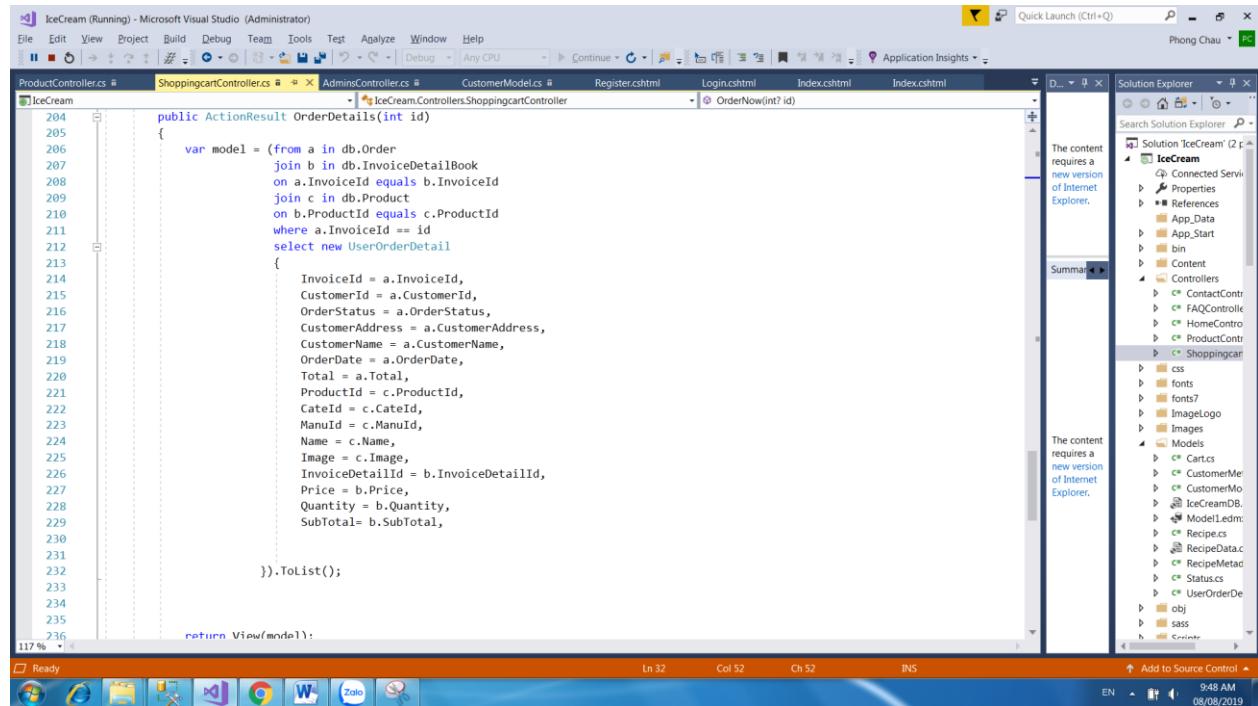
        <div class="col-lg-6 offset-lg-2">
            <div class="cart_total">
                <div class="section_title">The Detail Order</div>
                <div class="section_subtitle">To provide the best to you</div>
                <div class="cart_total_container">
                    <ul>
                        <li class="d-flex flex-row align-items-center justify-content-start">
                            <div class="cart_total_title">Order Id</div>
                            <div class="cart_total_value ml-auto">@Html.DisplayFor(model => model.InvoiceId)</div>
                        </li>
                        <li class="d-flex flex-row align-items-center justify-content-start">
                            <div class="cart_total_title">Customer Name</div>
                            <div class="cart_total_value ml-auto">@Html.DisplayFor(model => model.CustomerName)</div>
                        </li>
                        <li class="d-flex flex-row align-items-center justify-content-start">
                            <div class="cart_total_title">Order date</div>
                            <div class="cart_total_value ml-auto">@Html.DisplayFor(model => model.OrderDate)</div>
                        </li>
                        <li class="d-flex flex-row align-items-center justify-content-start">
                            <div class="cart_total_title">Customer Address</div>
                            <div class="cart_total_value ml-auto">@Html.DisplayFor(model => model.CustomerAddress)</div>
                        </li>
                        <li class="d-flex flex-row align-items-center justify-content-start">
                            <div class="cart_total_title">Total</div>
                            <div class="cart_total_value ml-auto">$@Html.DisplayFor(model => model.Total)</div>
                        </li>
                    </ul>
                </div>
            </div>
        </div>
    </div>
</div>

```

1.18. Details of an order: when user click details button in order details page, it will show details of it.

Product		4 Price	5 Quantity	6 Total
 1	2 Strawberry 27/07/2019 0:00:00 AM 3	\$65	2	\$130

No	Name	Type	Validation	Event	Description	Status
1	Image	Textbox (Read only)			Product's image	Enable
2	Product Name	Textbox (Read only)			Product's name	Enable
3	Order Date	Textbox (Read only)			Order date	Enable
4	Unit price	Textbox (Read only)			Product's unit price	Enable
5	Quantity	Textbox (Read only)			Quantity	Enable
6	Subtotal	Textbox (Read only)			Product's subtotal	Enable

Source code:**Controller:**


The screenshot shows the Microsoft Visual Studio interface with the 'IceCream (Running)' project open. The main window displays the code for the ShoppingCartController.cs file. The code implements a method named OrderDetails that performs a complex query across three tables (Order, InvoiceDetailBook, and Product) to retrieve user order details. The Solution Explorer on the right shows the project structure, including controllers like ContactController, FAQController, HomeController, and ShoppingCartController, along with various models, views, and shared files.

```

public ActionResult OrderDetails(int id)
{
    var model = (from a in db.Order
                 join b in db.InvoiceDetailBook
                 on a.InvoiceId equals b.InvoiceId
                 join c in db.Product
                 on b.ProductId equals c.ProductId
                 where a.InvoiceId == id
                 select new UserOrderDetail
                {
                    InvoiceId = a.InvoiceId,
                    CustomerId = a.CustomerId,
                    OrderStatus = a.OrderStatus,
                    CustomerAddress = a.CustomerAddress,
                    CustomerName = a.CustomerName,
                    OrderDate = a.OrderDate,
                    Total = a.Total,
                    ProductId = c.ProductId,
                    CateId = c.CateId,
                    ManuId = c.ManuId,
                    Name = c.Name,
                    Image = c.Image,
                    InvoiceDetailId = b.InvoiceDetailId,
                    Price = b.Price,
                    Quantity = b.Quantity,
                    SubTotal = b.SubTotal,
                }).ToList();

    return View(model);
}

```

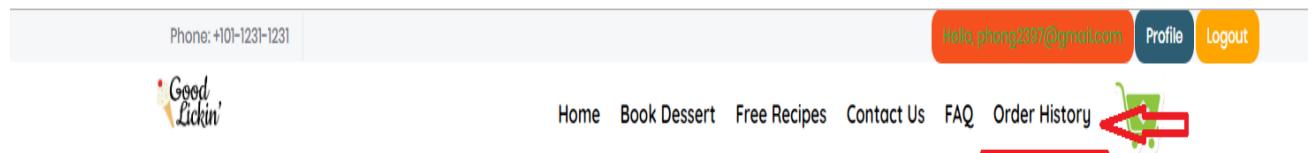
View:

```

<div class="super_container">
    <div class="cart_info">
        <div class="container">
            <div class="row">
                <div class="col">
                    <!-- Column Titles -->
                    <div class="cart_info_columns clearfix">
                        <div class="cart_info_col cart_info_col_product">Product</div>
                        <div class="cart_info_col cart_info_col_price">Price</div>
                        <div class="cart_info_col cart_info_col_quantity">Quantity</div>
                        <div class="cart_info_col cart_info_col_total">Total</div>
                    </div>
                </div>
            </div>
            <div class="row cart_items_row">
                <div class="col">
                    <!-- Cart Item -->
                    @foreach (var item in Model)
                    {
                        <div class="cart_item d-flex flex-lg-row flex-column align-items-lg-center align-items-start justify-content-start">
                            <!-- Name -->
                            <div class="cart_item_product d-flex flex-row align-items-center justify-content-start">
                                <div class="cart_item_image">
                                    <div></div>
                                </div>
                                <div class="cart_item_name_container">
                                    <div class="cart_item_name"><a href="#">@item.Name</a></div>
                                    <div class="cart_item_edit"><a href="#">@item.OrderDate</a></div>
                                </div>
                            </div>
                            <!-- Price -->
                            <div class="cart_item_price">$@item.Price</div>
                            <!-- Quantity -->
                        </div>
                    }
                </div>
            </div>
        </div>
    </div>
</div>

```

1.19. Personal order history: this page display history all orders of personal user. User can view it by click to menu “Order History” appears when user login successfully.



1 Order NO	3 Customer Name	3 Order Date	4 Quantity	5 Total	6 Detail
19	Ly Tien Dat	26/07/2019 0:00:00 AM	1	\$75	
19	Ly Tien Dat	26/07/2019 0:00:00 AM	1	\$65	

7

No	Name	Type	Validation	Event	Description	Status
1	OrderId	Textbox (Read only)			Id of Order	Enable
2	Customer Name	Textbox (Read only)			Customer's Name	Enable
3	Order Date	Textbox (Read only)			Order date	Enable
4	Quantity	Textbox (Read only)			Quantity	Enable
5	Subtotal	Textbox (Read only)			Product's subtotal	Enable
6	btnDetail	Button		Click	Details of Order	Enable
7	Number Page	Hyperlink		Click	Number Page Order history	

Source code:

Controller:

The screenshot shows the Microsoft Visual Studio interface for a project named 'IceCream'. The code editor displays the 'ShoppingCartController.cs' file, which contains C# code for handling orders. The Solution Explorer on the right lists the project structure, including controllers like 'ContactController', 'FAQController', 'HomeController', 'ProductController', and 'ShoppingCartController', along with various models, views, and shared files.

```
127 }  
128 }  
129 public ActionResult Orderhistory(int? page)  
130 {  
131     int id = Convert.ToInt32(Session["id"]);  
132     var itemList = from t in db.Order  
133                     where t.CustomerId == id  
134                     orderby t.InvoiceId descending  
135                     select t;  
136     var model = itemList.ToList();  
137  
138     int pageSize = 8;  
139     int pageNumber = (page ?? 1);  
140     return View(model.ToPagedList(pageNumber, pageSize));  
141 }  
142 }  
143 }  
144 }  
145 }  
146 }  
147 }  
148 }  
149 }  
150 }  
151 }  
152 }  
153 }  
154 }  
155 }  
156 }  
157 }  
158 }  
159 }
```

View:

```

<div class="container">
    <div class="row">
        <div class="col">
            <!-- Column Titles -->
            <div class="cart_info_columns clearfix">
                <div class="cart_info_col cart_info_col_price">Order NO</div>
                <div class="cart_info_col cart_info_col_price">Customer Name</div>
                <div class="cart_info_col cart_info_col_price" style="margin-left:100px">Order Date</div>
                <div class="cart_info_col cart_info_col_quantity">Quantity</div>
                <div class="cart_info_col cart_info_col_total">Total</div>
            </div>
        </div>
    </div>
    <div class="row cart_items_row">
        <div class="col">

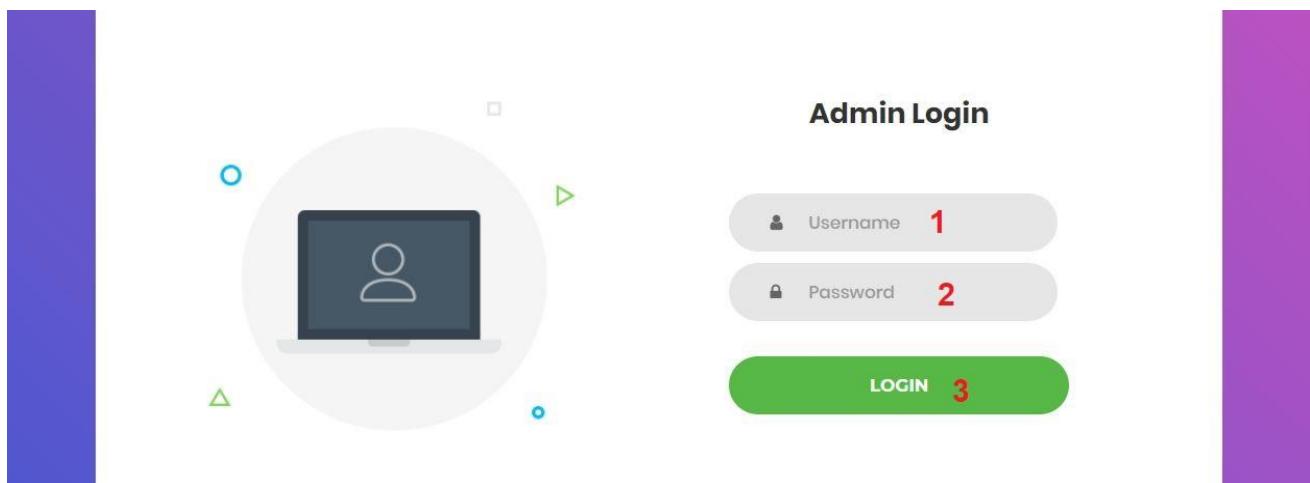
            <!-- Cart Item -->
            @foreach (var item in Model)
            {
                <div class="cart_item d-flex flex-lg-row flex-column align-items-lg-center align-items-start justify-content-start">
                    <!-- Name -->
                    <div class="cart_item_product">@item.InvoiceId</div>
                    <div class="cart_item_price">@item.CustomerName</div>
                    <div class="cart_item_total">
                        <div class="cart_item_total" style="margin-left:100px">@item.OrderDate</div>
                    </div>

                    <!-- Price -->
                    <!-- Quantity -->
                    <div class="cart_item_quantity">
                        <div class="product_quantity_container" style="margin-left:180px">
                            @*<div class="product_quantity clearfix">*@
                            <h4>@item.Quantity</h4>
                            @*</div>*@
                        </div>
                    </div>
                </div>
            <!-- Total -->
        </div>
    </div>

```

2. Admin Interface:

2.1. **Login Page:** Login form for admin (default: Username: **admin**, password: **123456**)



No	Name	Type	Validation	Event	Description	Status
1	txtUsername	Textbox	-Not null -Min length:5 -Max length:20	Focus	Input username	Enable
2	txtPassword	Textbox (password)	-Not null -Min length:6 -Max length:20	Focus	Input password	Enable
3	btnSubmit	Button		Click	Click to login	Enable

Source code:

Controller:

```
[HttpPost]
public ActionResult Login(Admin loginAdmin)
{
    try
    {
        var admin = db.Admin.SingleOrDefault(a => a.Username.Equals(loginAdmin.Username));
        if (ModelState.IsValid)
        {
            if (admin != null)
            {

                Session["aname"] = admin.Username;
                if (admin.Password.Equals(loginAdmin.Password))
                {
                    TempData["aName"] = admin;
                    return RedirectToAction("Index", "Admins");
                }
                else
                {
                    ModelState.AddModelError("", "Invalid password...");
                }
            }
            else
            {
                ModelState.AddModelError("", "Invalid username...");
            }
        }
    }
}
```

View:

```
<div class="container-login100">
    <div class="wrap-login100">
        <div class="login100-pic js-tilt" data-tilt>
            
        </div>

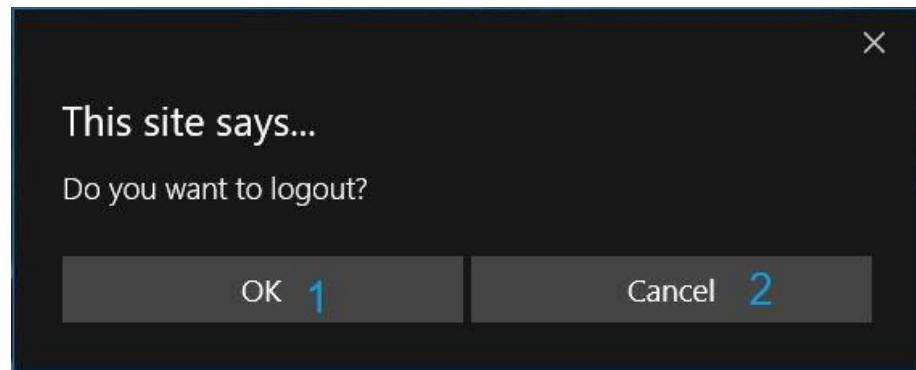
        <form id="Login" class="login100-form validate-form" method="post">
            <span class="login100-form-title">
                Admin Login
            </span>
            @Html.AntiForgeryToken()
            @Html.ValidationSummary(true)
            <div class="wrap-input100 validate-input">
                <input class="input100" type="text" name="Username" placeholder="Username">
                @Html.ValidationMessageFor(m => m.Username)
                <span class="focus-input100"></span>
                <span class="symbol-input100">
                    <i class="fa fa-user" aria-hidden="true"></i>
                </span>
            </div>

            <div class="wrap-input100 validate-input">
                <input class="input100" type="password" name="Password" placeholder="Password">
                @Html.ValidationMessageFor(m => m.Password)
                <span class="focus-input100"></span>
                <span class="symbol-input100">
                    <i class="fa fa-lock" aria-hidden="true"></i>
                </span>
            </div>

            <div class="container-login100-form-btn">
                <button class="login100-form-btn" type="submit">
                    Login
                </button>
            </div>
        </form>
    </div>

```

2.2. Logout: notification will appear when admin click logout button

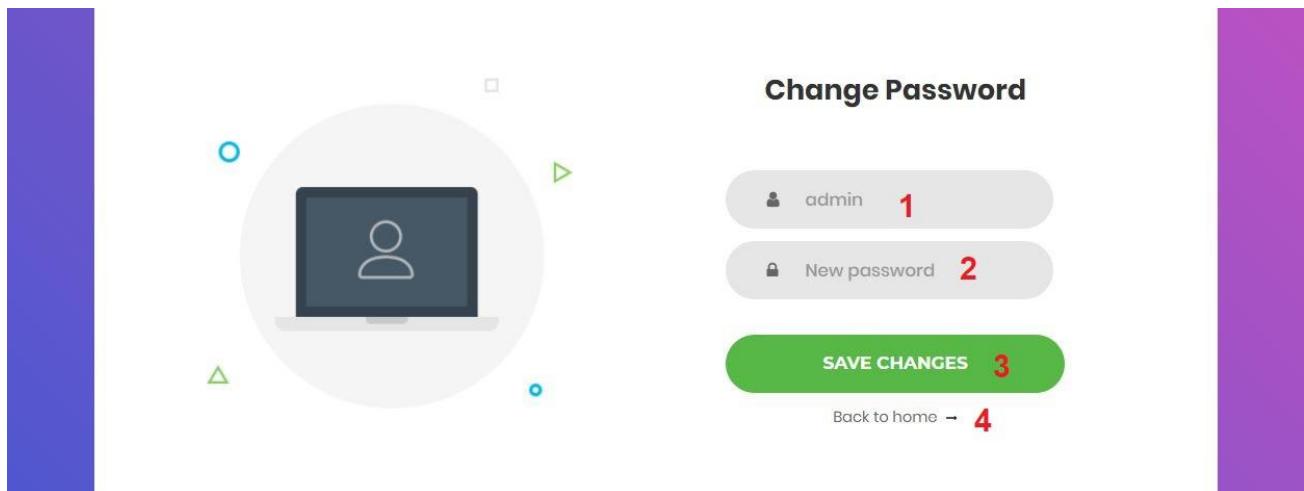


No	Name	Type	Validation	Event	Description	Status
1	btnOk	Button		Click	Click to logout	Enable
2	btnCancel	Button		Click	Click to return	Enable

Source code:

```
// GET: Admins/Logout
public ActionResult Logout()
{
    Session["aname"] = null;
    Session.Abandon();
    return RedirectToAction("Login", "Admins");
}
```

2.3. Change Password Page: Admin can change password here



No	Name	Type	Validation	Event	Description	Status
1	Username	Textbox (Read only)		Focus	Username	
2	New Password	Textbox (password)	- Not null - Min length:6 - Max length:20	Focus	Input new password	Enable
3	Save changes	Button		Click	Click to change password	Enable
4	Back to Home	Hyperlink		Click	Click to return home page	Enable

Source code:

Controller:

```

// GET: Admins/Edit/5
public ActionResult Edit(string id)
{
    if (id == null)
    {
        return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
    }
    Admin admin = db.Admin.Find(id);
    if (admin == null)
    {
        return HttpNotFound();
    }
    return View(admin);
}

// POST: Admins/Edit/5
// To protect from overposting attacks, please enable the specific properties you want to bind to, for
// more details see https://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
public ActionResult Edit([Bind(Include = "Username,Password")] Admin admin)
{
    if (ModelState.IsValid)
    {
        db.Entry(admin).State = EntityState.Modified;
        db.SaveChanges();
        return RedirectToAction("Index");
    }
    return View(admin);
}

```

View:

```

<div class="container-login100">
    <div class="wrap-login100">
        <div class="login100-pic js-tilt" data-tilt>
            
        </div>

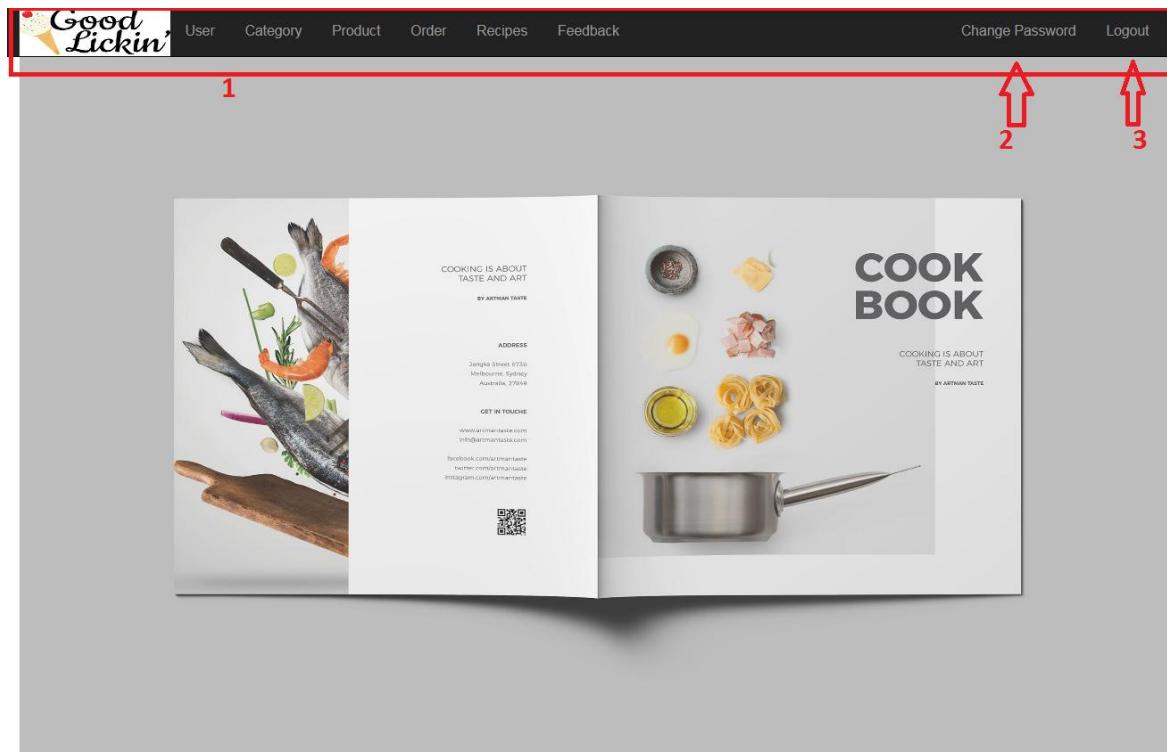
        <form id="Edit" class="login100-form validate-form" method="post">
            <span class="login100-form-title">
                Change Password
            </span>
            @Html.HiddenFor(model => model.Username)
            <div class="wrap-input100 validate-input">
                <input class="input100" type="text" placeholder="@Session["aname"]" readonly>
                <span class="focus-input100"></span>
                <span class="symbol-input100">
                    <i class="fa fa-user" aria-hidden="true"></i>
                </span>
            </div>

            <div class="wrap-input100 validate-input">
                <input class="input100" type="text" name="Password" placeholder="New password">
                @Html.ValidationMessageFor(m => m.Password)
                <span class="focus-input100"></span>
                <span class="symbol-input100">
                    <i class="fa fa-lock" aria-hidden="true"></i>
                </span>
            </div>

            <div class="container-login100-form-btn">
                <button class="login100-form-btn" type="submit" onclick="return confirm('Are you sure you want to change password')">
                    Save changes
                </button>
            </div>
        </form>
    </div>
</div>

```

2.4. Home Page: has navigation bar links to all manage pages



No	Name	Type	Validation	Event	Description	Status
1	Navigation Bar	Hyperlink		Click	Click to go to management pages	Enable
2	Change Password	Hyperlink		Click	Click to go to change password page	Enable
3	Logout	Hyperlink		Click	Click to logout	Enable

Source code:

FrontEnd:

```
// GET: Admins/Index
public ActionResult Index()
{
    if (Session["aname"] == null)
    {
        return RedirectToAction("Login", "Admins");
    }
    else
    {
        return View();
    }
}
```

```

<body>
    <div class="navbar navbar-inverse navbar-fixed-top">
        <div class="container">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                </button>
                <a href="~/Admins/Index"></a>
            </div>
            <div>
                <ul class="nav navbar-nav navbar-right">
                    @if (@Session["aname"] != null)
                    {
                        <ul class="nav navbar-nav navbar-right">
                            <li>@Html.ActionLink("Change Password", "Edit", new { id = @Session["aname"] })</li>
                            <li>@Html.ActionLink("Logout", "Logout", "Admins", new { onclick = "return confirm('Do you want to logout?')"} )</li>
                        </ul>
                    }
                </ul>
            </div>
            <div class="navbar-collapse collapse ">
                <ul class="nav navbar-nav">
                    @if (@Session["aname"] != null)
                    {
                        <li>@Html.ActionLink("User", "UserIndex", "Admins")</li>
                        <li>@Html.ActionLink("Category", "CategoryIndex", "Admins")</li>
                        <li>@Html.ActionLink("Product", "ProductIndex", "Admins")</li>
                        <li>@Html.ActionLink("Order", "OrderIndex", "Admins")</li>
                        <li>@Html.ActionLink("Recipes", "ReIndex", "Admins")</li>
                        <li>@Html.ActionLink("Feedback", "FeedbackIndex", "Admins")</li>
                    }
                </ul>
            </div>
        </div>
        <div class="container body-content">

```

2.5. Category Management Page: Admin can manage product here

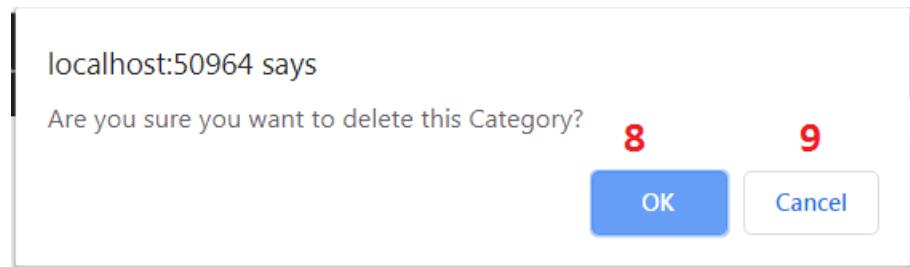
2.5.1. Add new Category: Create category form for admin

Categories List

Add New Category **2**

Search by name: **3** Search **4**

Name	5	6	7
Book 1	Edit	Details	Delete



No	Name	Type	Validation	Event	Description	Status
1	Category List	Table			List of Category	Enable
2	Add New Category	Hyperlink		Click	Click to go to add new category page	Enable
3	txtSearch	Textbox		Click	Input name of Search to search	Enable
4	btnFilter	Button		Click	Click to search product	Enable
5	Edit	Hyperlink		Click	Click to go to edit category page	Enable
6	Details	Hyperlink		Click	Click to go to product details page	Enable
7	Delete	Hyperlink		Click	Process to delete product	Enable
8	btnOk	Button		Click	Click to delete product	Enable
9	btnCancel	Button		Click	Click to return	Enable

Source code:**Controller:**

```
// GET: Admins/CategorieIndex
public ActionResult CategoryIndex(string cName)
{
    if (Session["aname"] == null)
    {
        return RedirectToAction("Login", "Admins");
    }
    var model = db.Category.ToList();
    if (!string.IsNullOrEmpty(cName))
    {
        model = model.Where(p => p.Name.ToUpper().Contains(cName)
                        || p.Name.ToLower().Contains(cName)).ToList();
        return View(model);
    }
    else
    {
        return View(model);
    }
}
```

View:

```
<p>
    @Html.ActionLink("Add New Category", "CategoryCreate", "Admins")
</p>
<div>
    @using (Html.BeginForm("Index", "User", FormMethod.Get))
    {

        <label>Search by name:</label>
        <input type="text" name="name" placeholder="Enter Name User" />
        <input type="submit" value="Search" class="btn btn-warning" />

    }
</div>
<table class="table">
    <tr>
        <th>
            @Html.DisplayNameFor(model => model.Name)
        </th>
        <th></th>
    </tr>
    @foreach (var item in Model)
    {
        <tr>
            <td>
                @Html.DisplayFor(modelItem => item.Name)
            </td>
            <td>
                @Html.ActionLink("Edit", "CategoryEdit", new { id = item.CateId }, new { @class = "btn btn-primary" })
                @Html.ActionLink("Details", "CategoryDetails", new { id = item.CateId }, new { @class = "btn btn-success" })
                @Html.ActionLink("Delete", "CategoryIndex", new { id=item.CateId },
                                new { onclick = "return confirm('Are you sure you want to delete this Category?')", @class = "btn btn-danger" })
            </td>
        </tr>
    }
}
```

2.5.2. Add new Category: Create category form for admin

Create New Category

Category

Name

Create
2

Back to List 3

No	Name	Type	Validation	Event	Description	Status
1	txtCateName	Textbox	- Not null - Max length:50	Focus	Input category name	Enable
2	btnSubmit	Button		Click	Click to add new category	Enable
3	Back to list	Hyperlink		Click	Click to go back category list	Enable

Source code:

Controller:

```

    // GET: Admins/CategoryCreate
    public ActionResult CategoryCreate()
    {
        if (Session["aname"] == null)
        {
            return RedirectToAction("Login", "Admins");
        }
        return View();
    }
,
```

View:

```
<h2>Add New Category</h2>

@using (Html.BeginForm())
{
    @Html.AntiForgeryToken()

    <div class="form-horizontal">
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
        <div class="form-group">
            @Html.LabelFor(model => model.Name, htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.Name, "", new { @class = "text-danger" })
            </div>
        </div>

        <div class="form-group">
            <div class="col-md-offset-2 col-md-10">
                <input type="submit" value="Create" class="btn btn-success" />
            </div>
        </div>
    </div>
}

<div>
    @Html.ActionLink("Back to List", "CategoryIndex")
</div>
```

2.5.3. Edit Category: Edit category form for admin

Edit Category

Name 1

Save 2

Back to List 3

No	Name	Type	Validation	Event	Description	Status
1	txtCateName	Textbox	- Not null - Max length:50	Focus	Input to edit category name	Enable
2	btnSubmit	Button		Click	Click to edit category	Enable
3	Back to list	Hyperlink		Click	Click to go back category list	Enable

Source code:

Controller:

```

public ActionResult CategoryEdit(int? id)
{
    if (Session["aname"] == null)
    {
        return RedirectToAction("Login", "Admins");
    }
    if (id == null)
    {
        return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
    }
    Category category = db.Category.Find(id);
    if (category == null)
    {
        return HttpNotFound();
    }
    return View(category);
}

// POST: Admins/CategoryEdit
// To protect from overposting attacks, please enable the specific properties you want to bind to, for
// more details see https://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult CategoryEdit([Bind(Include = "CategoryId,Name")] Category category)
{
    if (ModelState.IsValid)
    {
        db.Entry(category).State = EntityState.Modified;
        db.SaveChanges();
        return RedirectToAction("CategoryIndex");
    }
    return View(category);
}

```

View:

```

@using (Html.BeginForm()
{
    @Html.AntiForgeryToken()

    <div class="form-horizontal">
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
        @Html.HiddenFor(model => model.CateId)

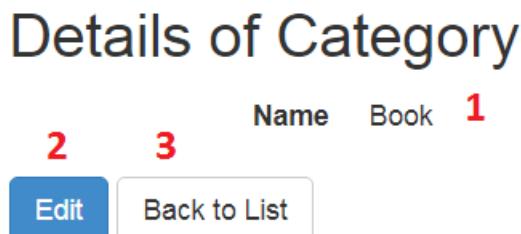
        <div class="form-group">
            @Html.LabelFor(model => model.Name, htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.Name, "", new { @class = "text-danger" })
            </div>
        </div>

        <div class="form-group">
            <div class="col-md-offset-2 col-md-10">
                <input type="submit" value="Save" class="btn btn-warning" onclick="return confirm('Are you sure you want to edit this category?')"/>
            </div>
        </div>
    </div>
}

<div>
    @Html.ActionLink("Back to List", "CategoryIndex", "", new { @class = "btn btn-default" })
</div>

```

2.5.4. Details of Category: This page display details of a category



No	Name	Type	Validation	Event	Description	Status
1	txtCateName	Textbox (Read only)			Category's name	Enable
2	Edit	Hyperlink		Click	Click to go to edit category page	Enable
3	Back to list	Hyperlink		Click	Click to go back category list	Enable

Source code:**Controller:**

```
// GET: Admins/CategoryDetails/5
public ActionResult CategoryDetails(int? id)
{
    if (Session["aname"] == null)
    {
        return RedirectToAction("Login", "Admins");
    }
    if (id == null)
    {
        return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
    }
    Category category = db.Category.Find(id);
    if (category == null)
    {
        return HttpNotFound();
    }
    return View(category);
}
```

View:

```
@model WebAdmin.Models.Category
@{
    ViewBag.Title = "Details";
    Layout = "~/Views/Shared/_AdminLayout.cshtml";
}

<h2>Details of Category</h2>



<dl class="dl-horizontal">
        <dt>
            @Html.DisplayNameFor(model => model.Name)
        </dt>

        <dd>
            @Html.DisplayFor(model => model.Name)
        </dd>
    </dl>
</div>
<p>
    @Html.ActionLink("Edit", "CategoryEdit", new { id = Model.CateId }, new { @class = "btn btn-primary" })

    @Html.ActionLink("Back to List", "CategoryIndex", "", new { @class = "btn btn-default" })
</p>


```

2.6. Product Management Page: Admin can manage product here

2.6.2. Manage Product List: Display list of product, admin can create, view details, edit or delete a product from here.

Product List

The screenshot shows a web application interface for managing products. At the top left is a blue button labeled "Add New Product" with a red number "2" indicating pending actions. To its right is a search bar with the placeholder "Enter any character" and a yellow "Search" button with a red number "4". A red box labeled "1" highlights the top right corner of the search bar area. Below the search bar is a table with four rows of product data. Each row contains columns for Name, Category, Type, Price, and three buttons: Edit (blue), Details (green), and Delete (red). Red numbers 5, 6, and 7 are placed above the "Edit", "Details", and "Delete" buttons respectively. A red box labeled "2" surrounds the entire table area. A modal dialog box is displayed in the center, containing the text "localhost:50964 says" and "Are you sure you want to delete this product?". It has two buttons at the bottom: a blue "OK" button with a red number "8" and a white "Cancel" button with a red number "9".

No	Name	Type	Validation	Event	Description	Status
1	Product List	Table			List of products	Enable
2	Add New	Hyperlink		Click	Click to go to add new product page	Enable
3	txtSearch	Textbox		Click	Input name of product to search	Enable
4	btnFilter	Button		Click	Click to search product	Enable
5	Edit	Hyperlink		Click	Click to go to edit product page	Enable
6	Details	Hyperlink		Click	Click to go to product details page	Enable
7	Delete	Hyperlink		Click	Process to delete product	Enable
8	btnOk	Button		Click	Click to delete product	Enable
9	btnCancel	Button		Click	Click to return	Enable

Source code:

Controller:

```
// GET: Admins/ProductIndex
public ActionResult ProductIndex(string pName)
{
    if (Session["aname"] == null)
    {
        return RedirectToAction("Login", "Admins");
    }
    var products = db.Product.Include(p => p.Category).Include(p => p.Manufacture);
    var model = products.ToList();
    if (!string.IsNullOrEmpty(pName))
    {
        model = model.Where(p => p.Name.ToUpper().Contains(pName)
                        || p.Name.ToLower().Contains(pName)).ToList();
        return View(model);
    }
    else
    {
        return View(model);
    }
}
```

View:

```
<label>Search by name:</label>
<input type="text" name="pName" placeholder="Enter any character" />
<input type="submit" value="Search" class="btn btn-warning" />

}

</div>
<table class="table">
    <tr>
        <th>
            @Html.DisplayNameFor(model => model.Name)
        </th>
        <th>
            @Html.DisplayName("Category")
        </th>
        <th>
            @Html.DisplayNameFor(model => model.Type)
        </th>
        <th>
            @Html.DisplayNameFor(model => model.Price)
        </th>
    </tr>

    @foreach (var item in Model)
    {
        <tr>
            <td>
                @Html.DisplayFor(modelItem => item.Name)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.Category.Name)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.Type)
            </td>
            <td>
                ...
            </td>
        </tr>
    }
</table>
```

2.6.3. Add new Product: Create product form for admin

Add New Product

Category	<input style="width: 150px; height: 25px; border: 1px solid #ccc; border-radius: 5px; padding: 5px; margin-bottom: 10px;" type="text" value="Book"/>	1
Manufacture	<input style="width: 150px; height: 25px; border: 1px solid #ccc; border-radius: 5px; padding: 5px; margin-bottom: 10px;" type="text" value="HarperCollins"/>	2
Name	<input style="width: 150px; height: 25px; border: 1px solid #ccc; border-radius: 5px; padding: 5px; margin-bottom: 10px;" type="text"/>	3
Type	<input style="width: 150px; height: 25px; border: 1px solid #ccc; border-radius: 5px; padding: 5px; margin-bottom: 10px;" type="text"/>	4
Recipe Date	<input style="width: 150px; height: 25px; border: 1px solid #ccc; border-radius: 5px; padding: 5px; margin-bottom: 10px;" type="text" value="mm/dd/yyyy"/>	5
Price	<input style="width: 150px; height: 25px; border: 1px solid #ccc; border-radius: 5px; padding: 5px; margin-bottom: 10px;" type="text"/>	6
Description	<input style="width: 150px; height: 50px; border: 1px solid #ccc; border-radius: 5px; padding: 5px; margin-bottom: 10px;" type="text"/>	7
Image	<input style="width: 150px; height: 25px; border: 1px solid #ccc; border-radius: 5px; padding: 5px; margin-bottom: 10px;" type="file" value="Choose File"/> <input style="width: 150px; height: 25px; border: 1px solid #ccc; border-radius: 5px; padding: 5px; margin-bottom: 10px;" type="text" value="No file chosen"/>	8
<input style="background-color: #28a745; color: white; border: none; border-radius: 5px; padding: 5px 10px; font-weight: bold; width: 150px; height: 30px;" type="button" value="Add New"/> 9		

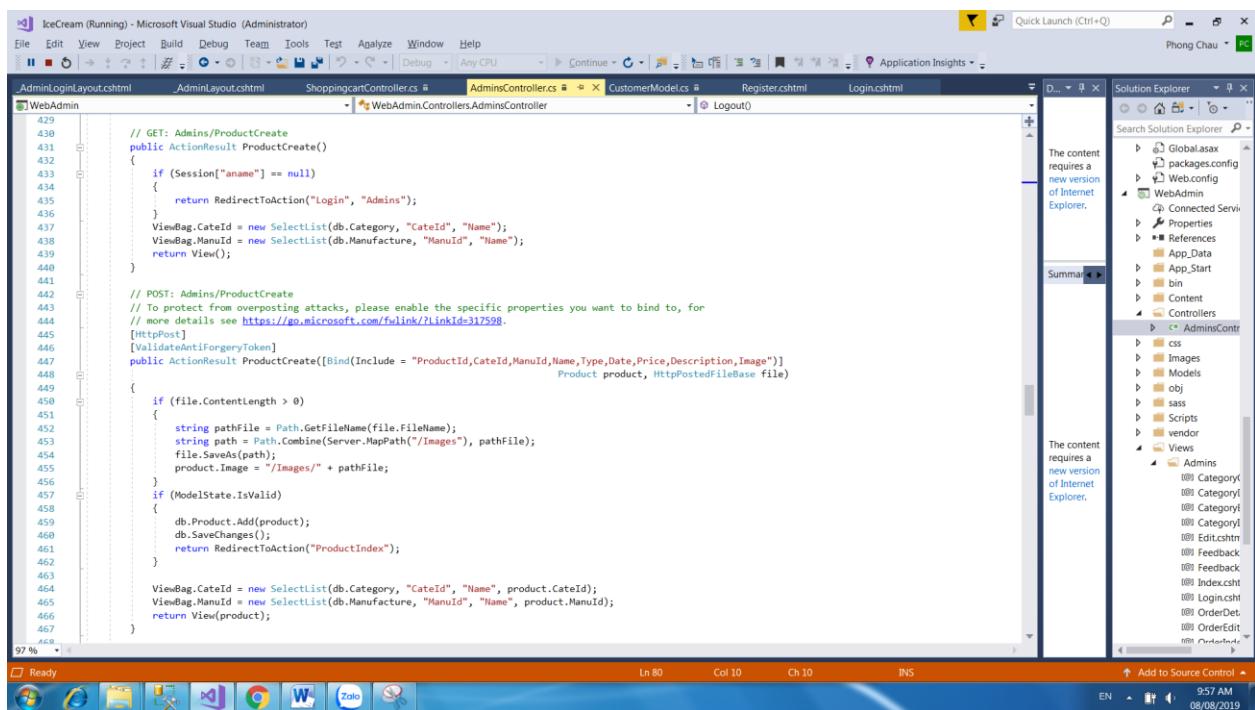
[Back to List](#) **10**

No	Name	Type	Validation	Event	Description	Status
2	ddCategory	Dropdown List	Not Null	Focus	Choose name of category	Enable
1	ddManufacture	Dropdown List	Not Null	Focus	Choose name of manufacture	Enable
3	txtProductName	Textbox	- Not null - Max length:50	Focus	Input product name	Enable
4	txtType	Textbox	- Not null - Max length:50	Focus	Input product type	Enable
5	Date	Calendar	Not Null	Focus	Input product recipe date	Enable
6	txtPrice	Textbox	- Not null - From 1 to 10000	Focus	Input product price	Enable
7	txtDescription	Textbox	- Not null - Max length:1000	Focus	Input product description	Enable
8	fulmage	File Upload	Not null	Focus	Upload product image	Enable

9	btnSubmit	Button	Click	Click to add new product	Enable
10	Back to List	Hyperlink	Click	Click to return product list	Enable

Source code:

Controller:



```

429 // GET: Admins/ProductCreate
430 public ActionResult ProductCreate()
431 {
432     if (Session["aname"] == null)
433     {
434         return RedirectToAction("Login", "Admins");
435     }
436     ViewBag.CatId = new SelectList(db.Category, "CatId", "Name");
437     ViewBag.ManId = new SelectList(db.Manufacture, "Manuid", "Name");
438     return View();
439 }
440
441
442 // POST: Admins/ProductCreate
443 // To protect from overposting attacks, please enable the specific properties you want to bind to, for
444 // more details see https://go.microsoft.com/fwlink/?LinkId=317598.
445 [HttpPost]
446 [ValidateAntiForgeryToken]
447 public ActionResult ProductCreate([Bind(Include = "ProductId,CatId,ManId,Name,Type,Date,Price,Description,Image")]
448                                     Product product, HttpPostedFileBase file)
449 {
450     if (file.ContentLength > 0)
451     {
452         string pathfile = Path.GetFileName(file.FileName);
453         string path = Path.Combine(Server.MapPath("~/Images"), pathfile);
454         file.SaveAs(path);
455         product.Image = "/Images/" + pathfile;
456     }
457     if (ModelState.IsValid)
458     {
459         db.Product.Add(product);
460         db.SaveChanges();
461         return RedirectToAction("ProductIndex");
462     }
463
464     ViewBag.CatId = new SelectList(db.Category, "CatId", "Name", product.CatId);
465     ViewBag.ManId = new SelectList(db.Manufacture, "Manuid", "Name", product.ManId);
466     return View(product);
467 }

```

View:

```

@using (Html.BeginForm("ProductCreate", "Admins", FormMethod.Post, new { enctype = "multipart/form-data" }))
{
    @Html.AntiForgeryToken()

    <div class="form-horizontal">
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
        <div class="form-group">
            @Html.LabelFor(model => model.CateId, "Category", htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.DropDownList("CateId", null, htmlAttributes: new { @class = "form-control" })
                @Html.ValidationMessageFor(model => model.CateId, "", new { @class = "text-danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.ManuId, "Manufacture", htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.DropDownList("ManuId", null, htmlAttributes: new { @class = "form-control" })
                @Html.ValidationMessageFor(model => model.ManuId, "", new { @class = "text-danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.Name, htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.Name, "", new { @class = "text-danger" })
            </div>
        </div>
    </div>
}

```

2.6.4. Product Details: This page display details of a product

Details of Product						7	8
1	2	3	4	5	6	Category	Manufacture
Name	Type	Date	Price	Image	Description		
Queijadas	Waffle	07/02/2017 0:00:00 AM	50		Preheat oven to 325 degrees F (165 degrees C). In a blender, combine eggs, sugar and butter. Blend until smooth. Pour in flour and milk, a little at a time, blending until smooth again. Stir in vanilla. Pour into muffin tins, filling 3/4 full. Bake in preheated oven 45 minutes, until golden brown. Serve hot or cold.	Book	HarperCollins

9 10

[Edit](#) [Back to List](#)

No	Name	Type	Validation	Event	Description	Status
1	Product Name	Textbox (Read only)			Product's name	Enable
2	Type	Textbox (Read only)			Product's type	Enable
3	Date	Calendar (Read only)			Product's recipe date	Enable
4	Price	Textbox (Read only)			Product's price	Enable
5	Image	Image (Read only)			Product's image	Enable
6	Description	Textbox (Read only)			Product's description	Enable
7	Category Name	Dropdown List			Category's Name	Enable
8	Manufacture Name	Dropdown List			Manufacture's Name	Enable
9	Edit	Hyperlink		Click	Click to go to edit product page	Enable
10	Back to List	Hyperlink		Click	Click to return product list	Enable

Source code:**Controller:**

```
// GET: Admins/ProductDetails/5
public ActionResult ProductDetails(int? id)
{
    if (Session["aname"] == null)
    {
        return RedirectToAction("Login", "Admins");
    }
    if (id == null)
    {
        return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
    }
    Product product = db.Product.Find(id);
    if (product == null)
    {
        return HttpNotFound();
    }
    return View(product);
}
```

View:

```
<h2>Details of Product</h2>



| @Html.DisplayNameFor(model => model.Name)        |
|--------------------------------------------------|
| @Html.DisplayNameFor(model => model.Type)        |
| @Html.DisplayNameFor(model => model.Date)        |
| @Html.DisplayNameFor(model => model.Price)       |
| @Html.DisplayNameFor(model => model.Image)       |
| @Html.DisplayNameFor(model => model.Description) |
| @Html.DisplayName("Category")                    |
| @Html.DisplayName("Manufacture")                 |


```

2.6.5. Edit Product: Edit product form for admin

Edit Product

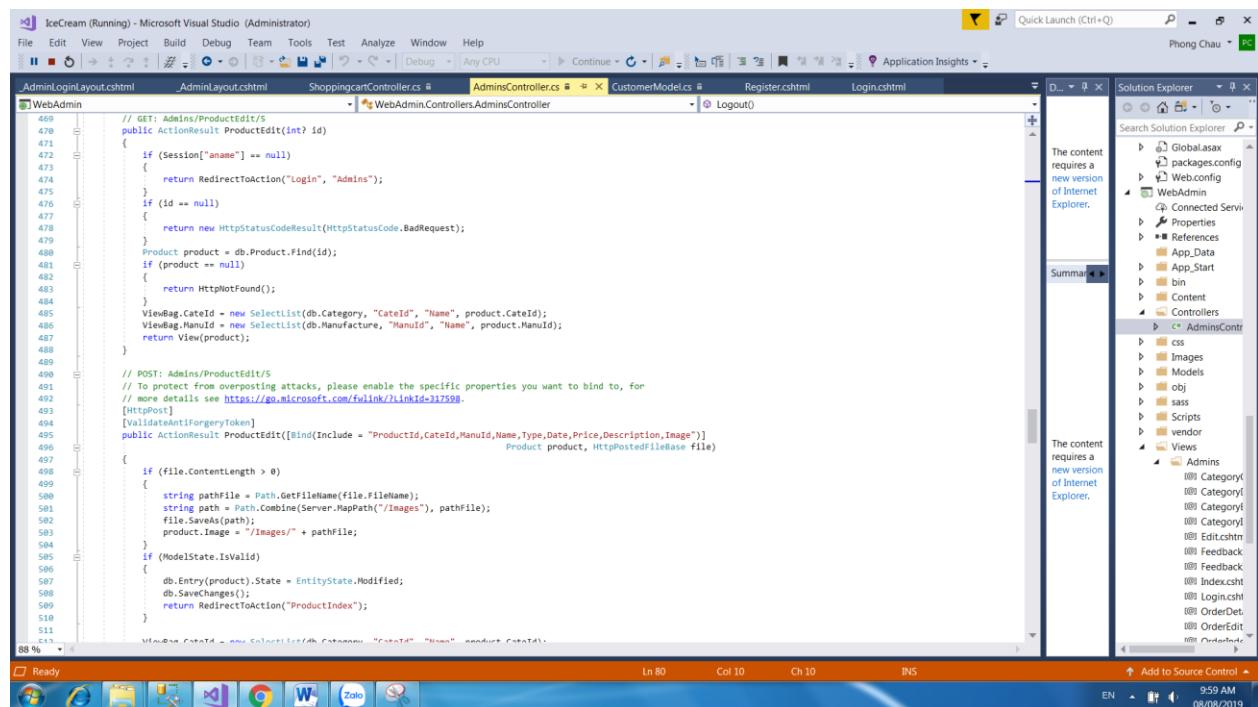
Category	<input style="width: 150px; border: 1px solid #ccc; padding: 2px;" type="text" value="Book"/> 1
Manufacture	<input style="width: 150px; border: 1px solid #ccc; padding: 2px;" type="text" value="HarperCollins"/> 2
Name	<input style="width: 150px; border: 1px solid #ccc; padding: 2px;" type="text" value="Queijadas"/> 3
Type	<input style="width: 150px; border: 1px solid #ccc; padding: 2px;" type="text" value="Wfafle"/> 4
Recipe Date	<input style="width: 150px; border: 1px solid #ccc; padding: 2px;" type="text" value="mm/dd/yyyy"/> 5
Price	<input style="width: 150px; border: 1px solid #ccc; padding: 2px;" type="text" value="50"/> 6
Description	<input style="width: 150px; border: 1px solid #ccc; padding: 2px;" type="text" value="Preheat oven to 325 degrees F (165"/> 7
Image	<input style="border: 1px solid #ccc; padding: 2px;" type="button" value="Choose File"/> No file chosen 8
<input style="background-color: #ff9933; color: white; border: 1px solid #ff9933; padding: 5px; margin-top: 10px;" type="button" value="Save changes"/> 9	

[Back to List](#) **10**

No	Name	Type	Validation	Event	Description	Status
2	ddCategory	Dropdown List	Not Null	Focus	Choose name of category	Enable
1	ddManufacture	Dropdown List	Not Null	Focus	Choose name of manufacture	Enable
3	txtProductName	Textbox	- Not null - Max length:50	Focus	Input product name	Enable
4	txtType	Textbox	- Not null - Max length:50	Focus	Input product type	Enable
5	Date	Calendar	Not Null	Focus	Input product recipe date	Enable
6	txtPrice	Textbox	- Not null - From 1 to 10000	Focus	Input product price	Enable
7	txtDescription	Textbox	- Not null - Max length:1000	Focus	Input product description	Enable
8	fulImage	File Upload	Not null	Focus	Upload product image	Enable
9	btnSaveChanges	Button		Click	Click to edit product	Enable
10	Back to List	Hyperlink		Click	Click to return product list	Enable

Source code:

Controller:



```

469 // GET: Admins/ProductEdit/{id}
470 [HttpGet]
471 public ActionResult ProductEdit(int? id)
472 {
473     if (Session["aname"] == null)
474     {
475         return RedirectToAction("Login", "Admins");
476     }
477     if (id == null)
478     {
479         return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
480     }
481     Product product = db.Product.Find(id);
482     if (product == null)
483     {
484         return HttpNotFound();
485     }
486     ViewData["CatId"] = new SelectList(db.Category, "CatId", "Name", product.CatId);
487     ViewData["ManuId"] = new SelectList(db.Manufacture, "ManuId", "Name", product.ManuId);
488     return View(product);
489 }
490
491 // POST: Admins/ProductEdit/
492 // To protect from overposting attacks, please enable the specific properties you want to bind to, for
493 // more details see https://go.microsoft.com/fwlink/?LinkId=317598.
494 [HttpPost]
495 [ValidateAntiForgeryToken]
496 public ActionResult ProductEdit([Bind(Include = "ProductId,CatId,ManuId,Name>Type,Date,Price,Description,Image")]
497                                     Product product, HttpPostedFileBase file)
498 {
499     if (file.ContentLength > 0)
500     {
501         string pathfile = Path.GetFileName(file.FileName);
502         string path = Path.Combine(Server.MapPath("~/Images"), pathfile);
503         file.SaveAs(path);
504         product.Image = "/Images/" + pathfile;
505     }
506     if (ModelState.IsValid)
507     {
508         db.Entry(product).State = EntityState.Modified;
509         db.SaveChanges();
510         return RedirectToAction("ProductIndex");
511     }
512     ViewData["CatId"] = new SelectList(db.Category, "CatId", "Name", product.CatId);
513 }

```

View:

```
<h2>Edit Product</h2>

@using (Html.BeginForm("ProductEdit", "Admins", FormMethod.Post, new { enctype = "multipart/form-data" }))
{
    @Html.AntiForgeryToken()

    <div class="form-horizontal">
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
        @Html.HiddenFor(model => model.ProductId)

        <div class="form-group">
            @Html.LabelFor(model => model.CateId, "Category", htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.DropDownList("CateId", null, htmlAttributes: new { @class = "form-control" })
                @Html.ValidationMessageFor(model => model.CateId, "", new { @class = "text-danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.ManuId, "Manufacture", htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.DropDownList("ManuId", null, htmlAttributes: new { @class = "form-control" })
                @Html.ValidationMessageFor(model => model.ManuId, "", new { @class = "text-danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.Name, htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
```

2.7. User Management Page: Admin can manage user here

2.7.2. Manage User List: Display list of user, admin can create, view details user from here

User List

Email	Password	Name	
phong2397@gmail.com	123456	Chau Thanh Phong	Details 1
kenzytien35@gmail.com	123456	mai hoang minh tien	Details
hyouka@gmail.com	123456	mai hoang minh tien	Details
kenzy@gmail.com	123456	Tien	Details
tien6399@gmail.com	123456	mai hoang minh tien	Details
lam123@gmail.com	123456	Ly Tien Dat	Details

No	Name	Type	Validation	Event	Description	Status
1	User List	Table			List of user	Enable
2	Add New	Hyperlink		Click	Click to go to add new user page	Enable
3	txtSearch	Textbox		Click	Input name of user to search	Enable
4	btnFilter	Button		Click	Click to search user	Enable
5	Details	Hyperlink		Click	Click to go to user details page	Enable

Source code:

Controller:

```
// GET: Admins/User index/id
public ActionResult UserIndex(string uName)
{
    if (Session["aname"] == null)
    {
        return RedirectToAction("Login", "Admins");
    }
    var model = db.Customer.ToList();
    if (!string.IsNullOrEmpty(uName))
    {
        model = model.Where(p => p.CustomerName.ToUpper().Contains(uName)
                        || p.CustomerName.ToLower().Contains(uName)).ToList();
        return View(model);
    }
    else
    {
        return View(model);
    }
}
```

View:

```
<div>
    @using (Html.BeginForm("UserIndex", "Admins", FormMethod.Get))
    {

        <label>Search by name:</label>
        <input type="text" name="uName" placeholder="Enter any character" />
        <input type="submit" value="Search" class="btn btn-warning" />

    }
</div>
<table class="table">
    <tr>
        <th>
            @Html.DisplayNameFor(model => model.CustomerEmail)
        </th>
        <th>
            @Html.DisplayNameFor(model => model.CustomerPassword)
        </th>
        <th>
            @Html.DisplayNameFor(model => model.CustomerName)
        </th>
    </tr>

    @foreach (var item in Model)
    {
        <tr>
            <td>
                @Html.DisplayFor(modelItem => item.CustomerEmail)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.CustomerPassword)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.CustomerName)
            </td>
        </tr>
    }
</table>
```

2.7.3. Add New User: Create user form for admin

Add New User

Email	<input type="text" value="1"/>
Password	<input type="text" value="2"/>
Name	<input type="text" value="3"/>
CustomerRole	<input type="text" value="4"/>
Day of Regist	<input type="text" value="mm/dd/yyyy 5"/>
Address	<input type="text" value="6"/>
Phone Number	<input type="text" value="7"/>
<input style="background-color: #28A745; color: white; border-radius: 5px; padding: 5px; margin-right: 10px;" type="button" value="Add New"/> 8	

[Back to List](#) 9

No	Name	Type	Validation	Event	Description	Status
1	txtEmail	Textbox	- Not null -DataType EmailAddress	Focus	Input email	Enable
2	txtPassword	Textbox	- Not null - Max length:50	Focus	Input password	Enable
3	txtName	Textbox	- Not null - Max length:50	Focus	Input user's full name	Enable
4	txtRole	Textbox		Focus	Input value role member	Enable
5	txtDayofRegist	Calendar		Focus	Input user's register	Enable
6	txtAddress	Textbox	- Not null - Max length:50	Focus	Input user's address	Enable
7	txtPhone	Textbox	- Not null - Max length:15	Focus	Input user's phone number	Enable
8	btnSubmit	Button		Click	Click to add new user	Enable
9	Back to List	Hyperlink		Click	Click to return user list	Enable

Source code:

Controller:

```
// GET: Admins/UserCreate
public ActionResult UserCreate()
{
    if (Session["aname"] == null)
    {
        return RedirectToAction("Login", "Admins");
    }
    return View();
}

// POST: Admins/UserCreate
// To protect from overposting attacks, please enable the specific properties you want to bind to, for
// more details see https://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult UserCreate([Bind(Include = "CustomerId,CustomerEmail,CustomerPassword,CustomerName,CustomerAddress,CustomerPhone,DateRegister,CustomerRole")] Customer user)
{
    if (ModelState.IsValid)
    {
        db.Customer.Add(user);
        db.SaveChanges();
        return RedirectToAction("UserIndex");
    }
    return View(user);
}
```

View:

```
<h2>Add New User</h2>

@using (Html.BeginForm())
{
    @Html.AntiForgeryToken()



@Html.ValidationSummary(true, "", new { @class = "text-danger" })
    <div class="form-group">
        @Html.LabelFor(model => model.CustomerEmail, htmlAttributes: new { @class = "control-label col-md-2" })
        <div class="col-md-10">
            @Html.EditorFor(model => model.CustomerEmail, new { htmlAttributes = new { @class = "form-control" } })
            @Html.ValidationMessageFor(model => model.CustomerEmail, "", new { @class = "text-danger" })
        </div>
    </div>

    <div class="form-group">
        @Html.LabelFor(model => model.CustomerPassword, htmlAttributes: new { @class = "control-label col-md-2" })
        <div class="col-md-10">
            @Html.EditorFor(model => model.CustomerPassword, new { htmlAttributes = new { @class = "form-control" } })
            @Html.ValidationMessageFor(model => model.CustomerPassword, "", new { @class = "text-danger" })
        </div>
    </div>

    <div class="form-group">
        @Html.LabelFor(model => model.CustomerName, htmlAttributes: new { @class = "control-label col-md-2" })
        <div class="col-md-10">
            @Html.EditorFor(model => model.CustomerName, new { htmlAttributes = new { @class = "form-control" } })
            @Html.ValidationMessageFor(model => model.CustomerName, "", new { @class = "text-danger" })
        </div>
    </div>

    <div class="form-group">
        @Html.LabelFor(model => model.CustomerRole, htmlAttributes: new { @class = "control-label col-md-2" })
        <div class="col-md-10">


```

2.7.4. User Details: This page display details of a user

Details of User

1	CustomerId	1
2	Password	123456
3	Name	Chau Thanh Phong
4	DateRegister	23/07/2019 0:00:00 AM
5	CustomerRole	1
6	Address	331 Vung Tau
7	Phone Number	0123456781
8	Email	phong2397@gmail.com

[Back to List](#) 9

No	Name	Type	Validation	Event	Description	Status
1	txtCustomerId	Textbox (Read only)		Focus	Username	Enable
2	txtPassword	Textbox (Read only)		Focus	Password	Enable
3	txtName	Textbox (Read only)		Focus	User's full name	Enable
4	txtDayofRegist	Textbox (Read only)		Focus	User's Regist	Enable
5	txCustomerRole	Textbox (Read only)		Focus	User's Role	Enable
6	txtAddress	Textbox (Read only)		Focus	User's address	Enable
7	txtPhone	Textbox (Read only)		Focus	User's phone number	Enable
8	txtEmail	Textbox (Read only)		Focus	User's email	Enable
9	Back to List	Hyperlink		Click	Click to return user list	Enable

Source code:**Controller:**

```
// GET: Admins/UserDetails/5
public ActionResult UserDetails(int? id)
{
    if (Session["aname"] == null)
    {
        return RedirectToAction("Login", "Admins");
    }
    if (id == null)
    {
        return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
    }
    Customer user = db.Customer.Find(id);
    if (user == null)
    {
        return HttpNotFound();
    }
    return View(user);
}
```

View:

```
<div>
    <dl class="dl-horizontal">
        <dt>
            @Html.DisplayNameFor(model => model.CustomerId)
        </dt>

        <dd>
            @Html.DisplayFor(model => model.CustomerId)
        </dd>

        <dt>
            @Html.DisplayNameFor(model => model.CustomerPassword)
        </dt>

        <dd>
            @Html.DisplayFor(model => model.CustomerPassword)
        </dd>

        <dt>
            @Html.DisplayNameFor(model => model.CustomerName)
        </dt>

        <dd>
            @Html.DisplayFor(model => model.CustomerName)
        </dd>

        <dt>
            @Html.DisplayNameFor(model => model.DateRegister)
        </dt>

        <dd>
            @Html.DisplayFor(model => model.DateRegister)
        </dd>

        <dt>
```

2.8. Order Management Page: Admin can manage order here

2.8.2. Manage Order List: Display list of order, admin can view details an order from here

Order List

OrderDate	User Account	Name	Address Delivery	OrderStatus	
2019-07-25	phong2397@gmail.com	Tien	331 Vung Tau	Confirmed	Details
2019-07-25	phong2397@gmail.com	Chau Thanh Phong	12 Nguyen Kim	Pending	Details
2019-07-25	phong2397@gmail.com			Pending	Details
2019-07-25	kenzytien35@gmail.com	Chau Thanh Phong	12 Nguyen Kim	Pending	Details
2019-07-26	kenzytien35@gmail.com	Chau Thanh Phong	12 Nguyen Kim	Pending	Details

No	Name	Type	Validation	Event	Description	Status
1	Order List	Table			List of order	Enable
2	txtdFrom	Textbox		Focus	Input start date want to search	Enable
3	txtdTo	Textbox		Focus	Input end date want to search	Enable
4	btnFilter	Button		Click	Click to search order	Enable
5	BtnStatus	Button		Click	Click to update status order	Enable
6	Details	Hyperlink		Click	Click to go to order details page	Enable

Source code:**Controller:**

```

// GET: Admins/OrderIndex
public ActionResult OrderIndex(DateTime? dFrom, DateTime? dTo)
{
    if (Session["aname"] == null)
    {
        return RedirectToAction("Login", "Admins");
    }
    var orders = db.Order.Include(o => o.Customer);
    var details = orders.Include(o => o.InvoiceDetailBook);
    if (!string.IsNullOrEmpty(dFrom.ToString()) && !string.IsNullOrEmpty(dTo.ToString()))
    {
        var model = details.ToList().Where(p => (p.OrderDate >= dFrom) && (p.OrderDate <= dTo));
        return View(model);
    }
    else
    {
        return View(details.ToList());
    }
}

public ActionResult ChangeStatus(int? id)
{
    var model = db.Order.Find(id);
    if (model == null)
    {
        return HttpNotFound();
    }
    //Update all field to avoid errors
    //Phone number must be in valid format
    if (model.OrderStatus == true)
    {
        model.CustomerId = model.CustomerId;
        model.CustomerName = model.CustomerName;
        model.CustomerAddress = model.CustomerAddress;
        model.OrderDate = model.OrderDate;
        model.Payment = model.Payment;
        model.Total = model.Total;

        model.OrderStatus = false;
        db.SaveChanges();
    }
    else if (model.OrderStatus == false)
    {
        model.CustomerId = model.CustomerId;
        model.CustomerName = model.CustomerName;
        model.CustomerAddress = model.CustomerAddress;
        model.OrderDate = model.OrderDate;
        model.Payment = model.Payment;
        model.Total = model.Total;

        model.OrderStatus = true;
        db.SaveChanges();
    }
}

return RedirectToAction("OrderIndex", "Admins");
}
:
```

View:

```
@model IEnumerable<WebAdmin.Models.Order>

@{
    ViewBag.Title = "Index";
    Layout = "~/Views/Shared/_AdminLayout.cshtml";
}



## Order List



@using (Html.BeginForm())
{
    <label>From Date:</label>
    <input type="date" name="dFrom" placeholder="MM/dd/yyyy" />
    <label>To Date:</label>
    <input type="date" name="dTo" placeholder="MM/dd/yyyy" />
    <input type="submit" value="Search" class="btn btn-warning" />
}



| <@Html.DisplayNameFor(model => model.OrderDate)> | <@Html.DisplayName("User Account")> | <@Html.DisplayNameFor(model => model.CustomerName)> | <@Html.DisplayName("Address Delivery")> | <@Html.DisplayNameFor(model => model.OrderStatus)> |
|--------------------------------------------------|-------------------------------------|-----------------------------------------------------|-----------------------------------------|----------------------------------------------------|
|--------------------------------------------------|-------------------------------------|-----------------------------------------------------|-----------------------------------------|----------------------------------------------------|


```

2.8.3. Order Details Page: This page display details of an order

Order Details

1

Product Name	Image	Price	Quantity	SubTotal
Golden Oreo Truffles		75	1	75
Ice Cream Sandwich Cake		50	1	50
Total:				125

[Back to List](#) 2

No	Name	Type	Validation	Event	Description	Status
1	Order Details List	Table			Order details of 1 order	Enable
2	Back to List	Hyperlink		Click	Click to return order list	Enable

Source code:

Controller:

```
// GET: Adminss/OrderDetailIndex
public ActionResult OrderDetailsIndex(int id)
{
    if (Session["aname"] == null)
    {
        return RedirectToAction("Login", "Admins");
    }
    var orderDetails = db.InvoiceDetailBook.Include(o => o.Order).Include(o => o.Product);
    var model = orderDetails.ToList().Where(o => o.InvoiceId == id);
    return View(model);
}
```

View:

```
ViewBag.Title = "Index";
Layout = "~/Views/Shared/_AdminLayout.cshtml";
}

<h2>Order Details</h2>


| @Html.DisplayName("Product Name")                                                                                                                                                                                                                    | @Html.DisplayNameFor(model => model.Product.Image) | @Html.DisplayNameFor(model => model.Price) | @Html.DisplayNameFor(model => model.Quantity) | @Html.DisplayNameFor(model => model.SubTotal) |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------|--------------------------------------------|-----------------------------------------------|-----------------------------------------------|
| @foreach (var item in Model)     {         <tr>...</tr>     }         <tr>             <td></td>             <td></td>             <td></td>             <td>Total:</td>             <td>@Model.Sum(s => s.SubTotal)</td>         </tr>     </table> |                                                    |                                            |                                               |                                               |


```

2.9. Feedback Management Page: Admin can manage feedback here

2.9.2. Feedback List: Display list of feedback, view details, recipe from here

Feedback List

Search by name: 2 Search 3

User's Fullname	Subject	
	Vani	Details 4
1	Book Dessert	Details
	Video	Details
Chau Thanh Phong	Website	Details

No	Name	Type	Validation	Event	Description	Status
1	Feedback List	Table			List of feedbacks	Enable
2	txtSearch	Textbox		Focus	Input user's full name to search feedback	Enable
3	btnFilter	Button		Click	Click to search feedback	Enable
4	Details	Hyperlink		Click	Click to go to details & delete feedback page	Enable

Source code:**Controller:**

```

    // GET: Admins/FeedbackIndex
    public ActionResult FeedbackIndex(string fName)
    {
        if (Session["aname"] == null)
        {
            return RedirectToAction("Login", "Admins");
        }
        var feedbacks = db.FeedBack.Include(f => f.Customer);
        var model = feedbacks.ToList();
        if (!string.IsNullOrEmpty(fName))
        {
            model = model.Where(p => p.Customer.CustomerName.ToUpper().Contains(fName)
                                || p.Customer.CustomerName.ToLower().Contains(fName)).ToList();
            return View(model);
        }
        else
        {
            return View(model);
        }
    }
}

```

View:

```

<h2>Feedback List</h2>



@using (Html.BeginForm("FeedbackIndex", "Admins", FormMethod.Get))
    {

        <label>Search by name:</label>
        <input type="text" name="fName" placeholder="Enter any character" />
        <input type="submit" value="Search" class="btn btn-warning"/>

    }



| @Html.DisplayName("User's Fullname") | @Html.DisplayNameFor(model => model.Subject) |  |
|--------------------------------------|----------------------------------------------|--|
|--------------------------------------|----------------------------------------------|--|


```

2.9.3. Details Feedback: Display details of a feedback's user.

Details Feedback

1 User's Fullname Chau Thanh Phong
2 Subject Website
3 Content design eco with customer

[Back to List](#)

4

No	Name	Type	Validation	Event	Description	Status
1	txtUsername	Textbox (Read only)		Focus	Username	Enable
2	txtSubject	Textbox (Read only)		Focus	Feedback's Subject	Enable
3	txtContent	Textbox (Read only)		Focus	Feedback's Content	Enable
4	Back to List	Hyperlink		Click	Click to return feedback list	Enable

Source code

Controller:

```
}

// GET: Admins/FeedbackDelete/5
public ActionResult FeedbackDelete(int? id)
{
    if (Session["aname"] == null)
    {
        return RedirectToAction("Login", "Admins");
    }
    if (id == null)
    {
        return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
    }
    FeedBack feedback = db.FeedBack.Find(id);
    if (feedback == null)
    {
        return HttpNotFound();
    }
    return View(feedback);
}

// POST: Admins/FeedbackDelete/5
[HttpPost, ActionName("FeedbackDelete")]
[ValidateAntiForgeryToken]
public ActionResult DeleteConfirmed(int? id)
{
    FeedBack feedback = db.FeedBack.Find(id);
    db.FeedBack.Remove(feedback);
    db.SaveChanges();
    return RedirectToAction("FeedbackIndex");
}
```

View:

```

<h2>Details Feedback</h2>
<div>
    <dl class="dl-horizontal">
        <dt>
            @Html.DisplayName("User's Fullname")
        </dt>

        <dd>
            @Html.DisplayFor(model => model.Customer.CustomerName)
        </dd>

        <dt>
            @Html.DisplayNameFor(model => model.Subject)
        </dt>

        <dd>
            @Html.DisplayFor(model => model.Subject)
        </dd>

        <dt>
            @Html.DisplayNameFor(model => model.Content)
        </dt>

        <dd>
            @Html.DisplayFor(model => model.Content)
        </dd>
    </dl>

    @using (Html.BeginForm())
    {
        @Html.AntiForgeryToken()
    }

```

2.10. Recipes Manage Page: Admin can manage recipe in here

2.10.1. Recipes List: Display list of Recipe, admin can view details, edit or delete a feedback from here

Recipes List

Add New Recipes 2

Search by name: Enter any character 3 Search 4

1
↑

Title	LinkYT	Status	5	6	7
Vanilla Ice-Cream Recipe	https://www.youtube.com/watch?v=6XwFsl2S9Ro	VIP	Edit	Details	Delete
Chocolate Ice-Cream Recipe	https://www.youtube.com/watch?v=X9KBvedOAOE	Free	Edit	Details	Delete
Strawberry Ice-Cream Recipe	https://www.youtube.com/watch?v=xDW-BZuElns	Free	Edit	Details	Delete
Chocolate Chip Ice-Cream Recipe	https://www.youtube.com/watch?v=X9KBvedOAOE	VIP	Edit	Details	Delete
Mango Ice-Cream Recipe	https://www.youtube.com/watch?v=QtLHSYbdCPI	Free	Edit	Details	Delete



No	Name	Type	Validation	Event	Description	Status
1	Recipe List	Table			List of Recipe	Enable
2	Add New	Hyperlink		Click	Click to go to add new recipes page	Enable
3	txtSearch	Textbox		Click	Input name of recipe to search	Enable
4	btnFilter	Button		Click	Click to search recipe	Enable
5	Edit	Hyperlink		Click	Click to go to edit recipe page	Enable
6	Details	Hyperlink		Click	Click to go to recipe details page	Enable
7	Delete	Hyperlink		Click	Process to delete recipe	Enable
8	btnOk	Button		Click	Click to delete recipe	Enable
9	btnCancel	Button		Click	Click to return	Enable

Source code:**Controller:**

```

public ActionResult ReIndex(string pName)
{
    if (Session["aname"] == null)
    {
        return RedirectToAction("Login", "Admins");
    }

    var model = db.Recipes.ToList();
    if (!string.IsNullOrEmpty(pName))
    {
        model = model.Where(p => p.Title.ToUpper().Contains(pName)
                           || p.Title.ToLower().Contains(pName)).ToList();
        return View(model);
    }
    else
    {
        return View(model);
    }
}

```

View:

```

<div>
    @using (Html.BeginForm("ReIndex", "Admins", FormMethod.Get))
    {

        <label>Search by name:</label>
        <input type="text" name="pName" placeholder="Enter any character" />
        <input type="submit" value="Search" class="btn btn-warning" />

    }
</div>
<table class="table">
    <tr>
        <th>
            @Html.DisplayNameFor(model => model.Title)
        </th>
        <th>
            @Html.DisplayNameFor(model => model.LinkYT)
        </th>
        <th>
            @Html.DisplayNameFor(model => model.Status)
        </th>
    </tr>
    @foreach (var item in Model)
    {
        <tr>
            <td>
                @Html.DisplayFor(modelItem => item.Title)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.LinkYT)
            </td>

```

2.10.2. Add new Recipes: Create Recipes form for admin

Add New Recipes

1	Title	<input type="text"/>
2	Content	<input type="text"/>
3	Image	<input type="text"/>
4	LinkYT	<input type="text"/>
5	Status	<input type="text"/>
		Save Changes
		6
Back to List 7		

No	Name	Type	Validation	Event	Description	Status
1	txtTitle	Textbox		Focus	Input Title	Enable
2	txtContent	Textbox		Focus	Input Content	Enable
3	Image	Textbox		Focus	Choose File Image of recipe	Enable
4	txtLinkYT	Textbox		Focus	Input Link	Enable
5	txtStatus	TextBox		Focus	Input Status of Recipe	Enable
6	btnSubmit	Button		Click	Click to add new recipe	Enable
7	Back to List	Hyperlink		Click	Click to return recipe list	Enable

Source code

Controller:

```

669
670
671     // GET: Admins/UserCreate
672     public ActionResult RecipeCreate()
673     {
674         if (Session["aname"] == null)
675         {
676             return RedirectToAction("Login", "Admins");
677         }
678         return View();
679     }
680
681     // POST: Admins/UserCreate
682     // To protect from overposting attacks, please enable the specific properties you want to bind to, for
683     // more details see https://go.microsoft.com/fwlink/?LinkId=317598.
684     [HttpPost]
685     [ValidateAntiForgeryToken]
686     public ActionResult RecipeCreate([Bind(Include = "Id,Ttitle,Content,Imag,LinkYT,Status")] Recipes recipe)
687     {
688
689         if (ModelState.IsValid)
690         {
691             db.Recipes.Add(recipe);
692             db.SaveChanges();
693             return RedirectToAction("ReIndex");
694         }
695
696
697         return View(recipe);
698     }

```

View:

```

<h2>Add New Recipes</h2>

@using (Html.BeginForm())
{
    @Html.AntiForgeryToken()

    <div class="form-horizontal">
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })

        <div class="form-group">
            @Html.LabelFor(model => model.Title, htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Title, new { htmlAttributes = new { @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.Title, "", new { @class = "text-danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.Content, htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Content, new { htmlAttributes = new { @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.Content, "", new { @class = "text-danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.Imag, htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Imag, new { htmlAttributes = new { @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.Imag, "", new { @class = "text-danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.LinkYT, htmlAttributes: new { @class = "control-label col-md-2" })

```

2.10.3. Edit Recipes: Edit Recipe form for admin

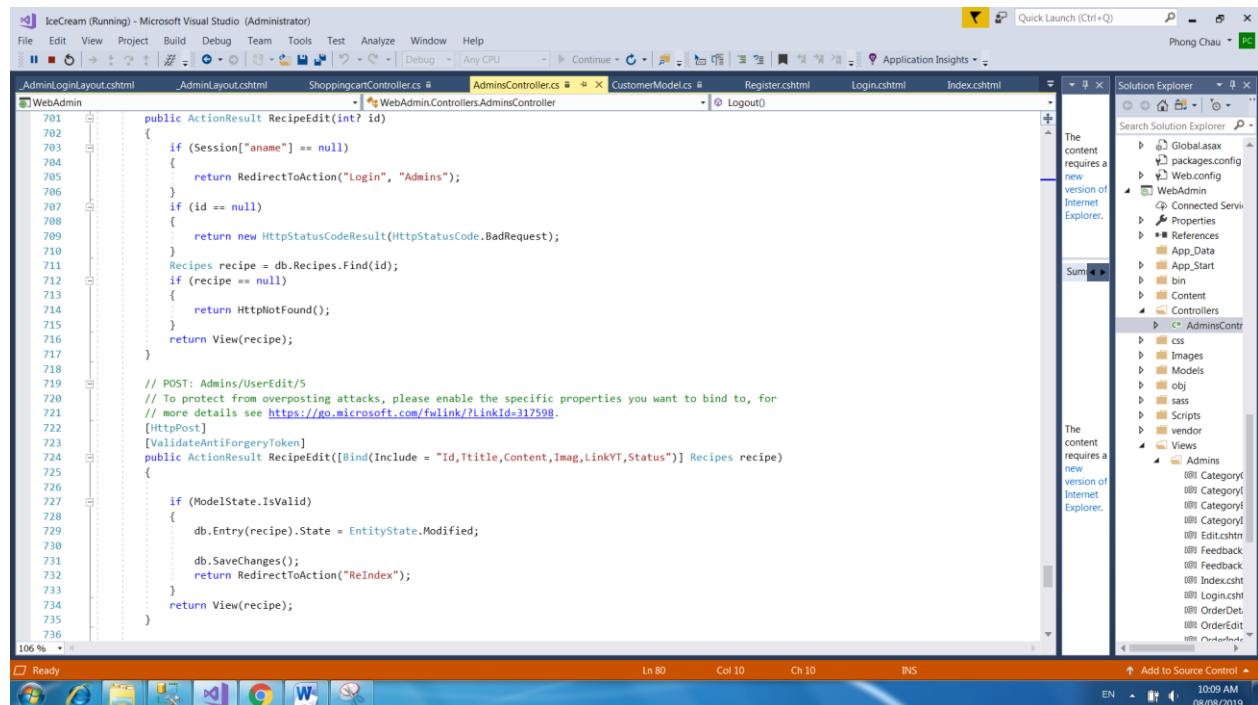
Edit Recipes

1	Title	Vanilla Ice-Cream Recipe
2	Content	Combine 1 1/2 cups heavy cream, 1 cup
3	Image	/Images/Vanilla.jpg
4	LinkYT	https://www.youtube.com/watch?v=6XwF
5	Status	VIP
		Save Changes
		6
Back to List		7

No	Name	Type	Validation	Event	Description	Status
1	txtTitle	Textbox (Read Only)		Focus	Input Title	Enable
2	txtContent	Textbox (Read Only)		Focus	Input Content	Enable
3	Image	Textbox		Focus	Input link Image	Enable
4	txtLinkYT	Textbox (Read Only)		Focus	Input Link	Enable
5	txtStatus	Textbox (Read Only)		Focus	Input Status of Recipe	Enable
8	btnSubmit	Button		Click	Click to add new recipe	Enable
9	Back to List	Hyperlink		Click	Click to return recipe list	Enable

Source code:

Controller:



```

    public ActionResult RecipeEdit(int? id)
    {
        if (Session["aname"] == null)
        {
            return RedirectToAction("Login", "Admins");
        }
        if (id == null)
        {
            return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
        }
        Recipes recipe = db.Recipes.Find(id);
        if (recipe == null)
        {
            return HttpNotFound();
        }
        return View(recipe);
    }

    // POST: Admins/UserEdit/5
    // To protect from overposting attacks, please enable the specific properties you want to bind to, for
    // more details see https://go.microsoft.com/fwlink/?LinkId=317598.
    [HttpPost]
    [ValidateAntiForgeryToken]
    public ActionResult RecipeEdit([Bind(Include = "Id,Title,Content,Image,LinkYT,Status")] Recipes recipe)
    {
        if (ModelState.IsValid)
        {
            db.Entry(recipe).State = EntityState.Modified;
            db.SaveChanges();
            return RedirectToAction("ReIndex");
        }
        return View(recipe);
    }
}

```

View:

```

<h2>Edit Recipes</h2>

@using (Html.BeginForm())
{
    @Html.AntiForgeryToken()

    <div class="form-horizontal">
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })

        <div class="form-group">
            @Html.LabelFor(model => model.Title, htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Title, new { htmlAttributes = new { @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.Title, "", new { @class = "text-danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.Content, htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Content, new { htmlAttributes = new { @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.Content, "", new { @class = "text-danger" })
            </div>
        </div>

        <div class="form-group">

```

2.10.4. Details of Recipes: This page display details of a Recipes

Details of Recipes

- 1 Title Vanilla Ice-Cream Recipe
- 2 Content Combine 1 1/2 cups heavy cream, 1 cup whole milk, 1/4 cup sugar
- 3 Imag /Images/Vanilla.jpg
- 4 LinkYT <https://www.youtube.com/watch?v=6XwFsl2S9Ro>
- 5 Status VIP

6

[Edit](#)[Back to List](#)

7

No	Name	Type	Validation	Event	Description	Status
1	txtTitle	Textbox (Read Only)		Focus	Title	Enable
2	txtContent	Textbox (Read Only)		Focus	Content	Enable
3	Imag	Textbox		Focus	Link Image	Enable
4	txtLinkYT	Textbox (Read Only)		Focus	Link	Enable
5	txtStatus	Textbox (Read Only)		Focus	Status of Recipe	Enable
6	Edit	Hyperlink		Click	Click to return edit recipe	Enable
7	Back to List	Hyperlink		Click	Click to return recipe list	Enable

Source code:

Controller:

```

public ActionResult RecipeDetails(int? id)
{
    if (Session["aname"] == null)
    {
        return RedirectToAction("Login", "Admins");
    }
    if (id == null)
    {
        return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
    }
    Recipes recipe = db.Recipes.Find(id);
    if (recipe == null)
    {
        return HttpNotFound();
    }
    return View(recipe);
}

```

View:

```

<h2>Details of Recipes</h2>

<div>
    <dl class="dl-horizontal">

        <dt>
            @Html.DisplayNameFor(model => model.Title)
        </dt>

        <dd>
            @Html.DisplayFor(model => model.Title)
        </dd>

        <dt>
            @Html.DisplayNameFor(model => model.Content)
        </dt>

        <dd>
            @Html.DisplayFor(model => model.Content)
        </dd>

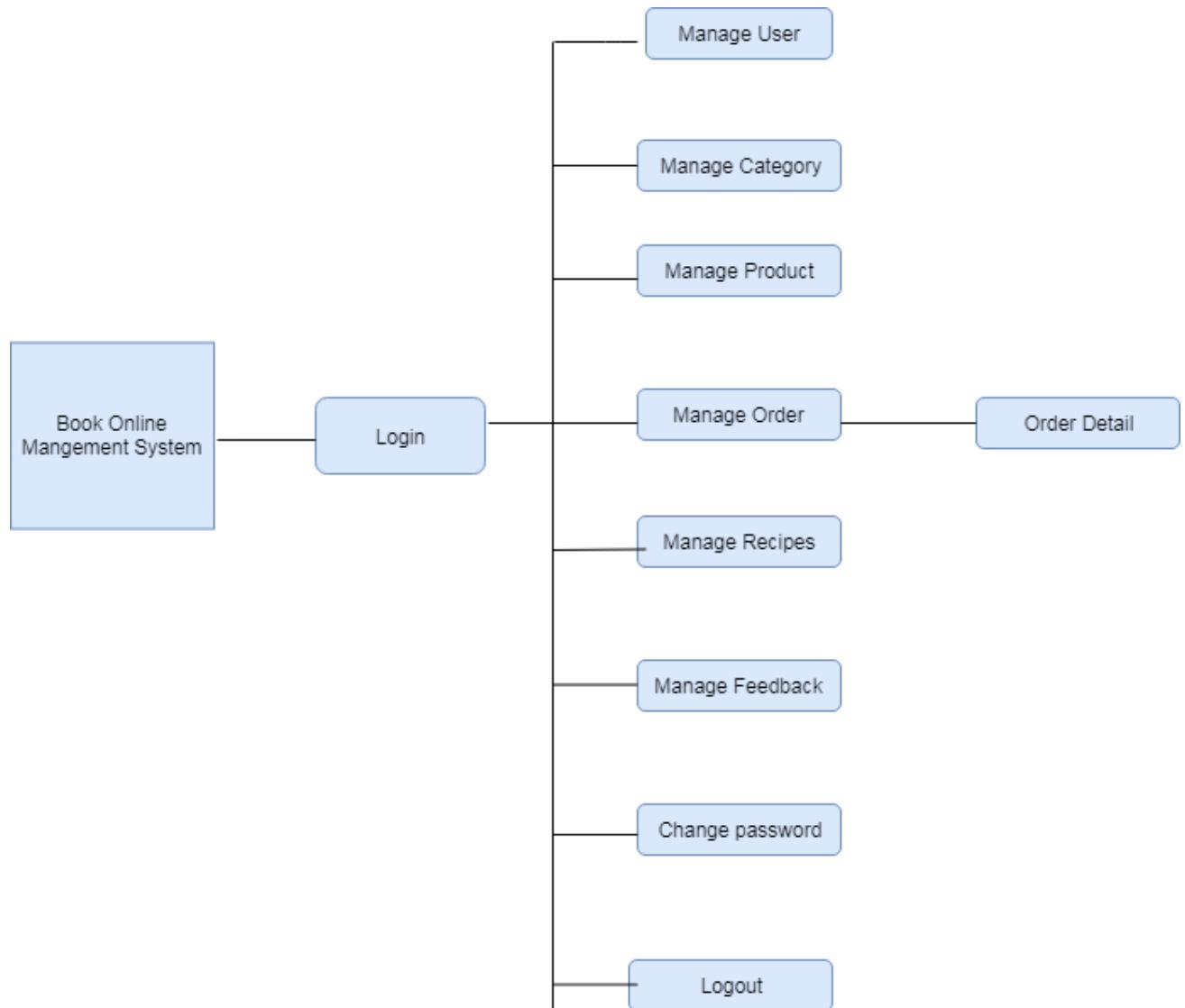
        <dt>
            @Html.DisplayNameFor(model => model.Img)
        </dt>

        <dd>
            @Html.DisplayFor(model => model.Img)
        </dd>
    </dl>
</div>

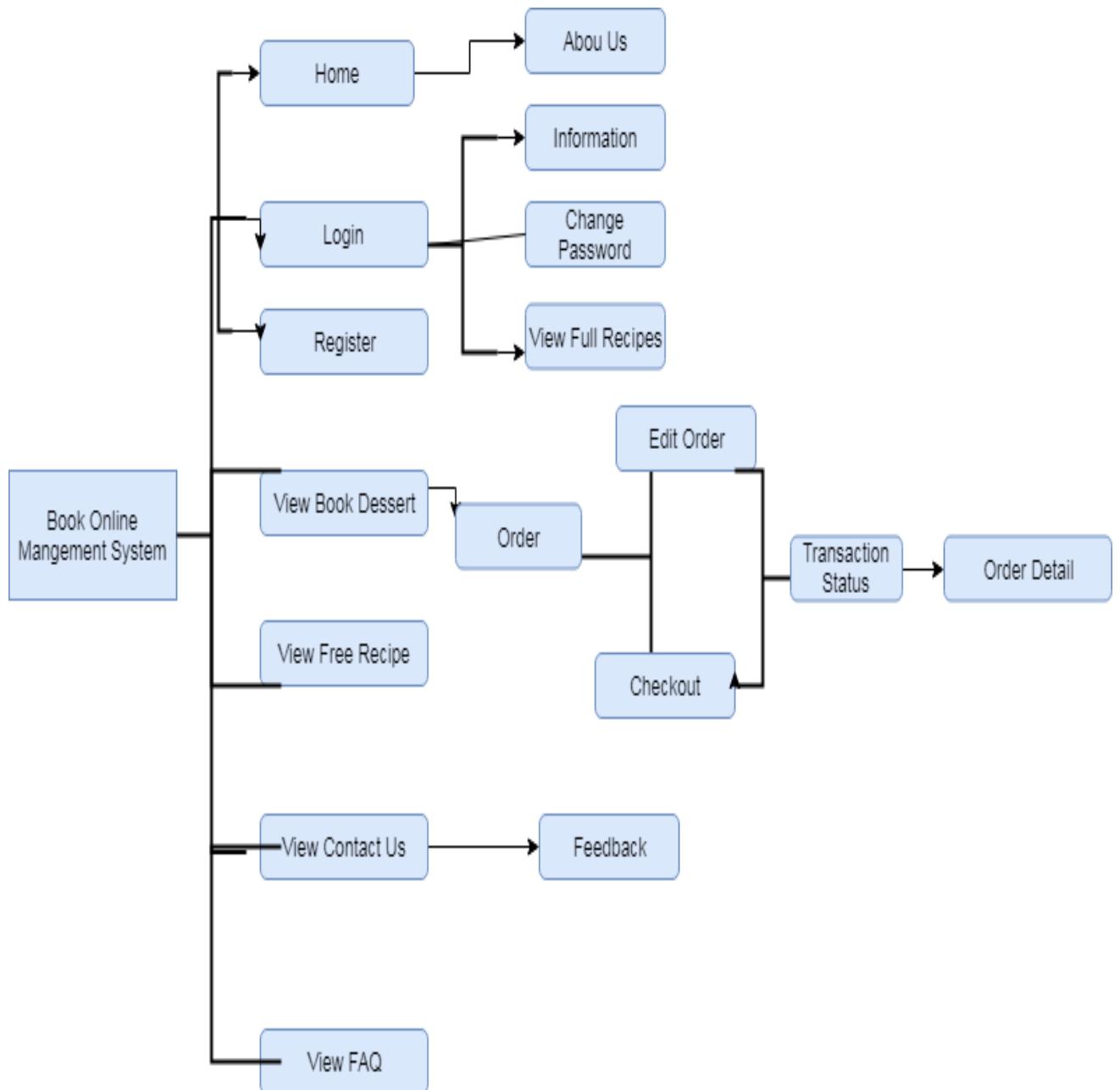
```

3. SITE MAP

1. Back-End Site Map



2. Front-End Site Map



1. TEST PLAN/TEST CASE

Test	What is being tested	How	Test data used	Expected Results
1	Order of input on data entry screen	Enter data from data capture sheet into the form on the data entry screen	Data set 1	Data entry order is the same as that on the data capture sheet
2	Validation of input	Enter typical values, boundary value, values that should be rejected	Data set 2	Good data accepted, bad data rejected
3	Accurate calculations		Data set 3	Data set 3A
4	Scaled output		Data set 4	Paginated output

2. CHECKLIST OF VALIDATIONS

Option	Validated
Do all numeric variables have a default value of zero?	Yes
Does the administrator have all the rights to create and delete the records?	Yes
Are all the records properly fed into the appropriate database?	Yes
Have all the modules been properly integrated and are completely functional?	Yes
Have all the design and coding standards been followed and implemented?	Yes
Is the design consistent all over?	Yes
Is the navigation sequence correct through all the screens in the application?	Yes
Is exception handling mechanism implemented in all the screens?	Yes
Are all the program codes working?	Yes

3. SUBMISSION CHECKLIST

Sr.No.	Particulars	Yes	No	NA	Comments
1	Have all the modules been properly integrated and are they completely functional?	Yes			
2	Does each unit meet its objective and purpose? Are all the validations happening as specified in Process Design?	Yes			
3	Have all Design and Coding standards been followed and implemented?	Yes			
4	Is the GUI design consistent all over?	Yes			
5	Are the codes working as per the specification?	Yes			.
6	Does the application's functionality resolve the client problem, and satisfy his/her needs completely?	Yes			
7	Have the hardware and software been correctly chosen?	Yes			
8	Additional features and utilities that give value addition to the entire project.	Yes			

TASK SHEET REVIEW 3

Project Ref. No.:		Activity	Date of Preparation of Activity Plan:				
Airlines Reservation System Project		Plan	Prepared By:	Actual Start Date	Actual Days	Team Mate Names	Status
Sr.No.	Task	Title:					
1	Admin Site Map			15/07/2019		Chau Thanh Phong	Completed
2	Client Site Map				1	Mai Hoang Minh Tien	Completed
Client Pages							
3	Login Page				1	Mai Hoang Minh Tien	Completed
4	Dashboard Page			21/07/2019	1	Bui Van Huong	Completed
5	Register Page				1	Mai Hoang Minh Tien	Completed
6	Members Page				1	Mai Hoang Minh Tien	Completed
7	Change Password Page					Mai Hoang Minh Tien	Completed
8	Product Book Page	Ice Cream Shop	Bui Van	22/07/2019	5	Chau Thanh Phong	Completed
9	Detail Book Page	Management System	Huong			Chau Thanh Phong	Completed
10	Shopping Cart Page				4	Chau Thanh Phong	Completed
11	Order Page			26/07/2019	4	Chau Thanh Phong	Completed
12	Detail Order Page				2	Chau Thanh Phong	Completed
13	Free Recipes Page				1	Ly Tien Dat	Completed
14	Full Recipes Page				1	Ly Tien Dat	Completed
15	Detail Recipes Page			27/07/2019	2	Ly Tien Dat	Completed
16	Feedback Page			28/07/2019	1	Dang Nguyen Dan Linh	Completed
17	FAQ Page			29/07/2019	1	Dang Nguyen Dan Linh	Completed

Admin Pages

13	Login Page				Completed
14	Dashboard Page	25/07/2019	1	Bui Van huong	Completed
15	Change Password Page				Completed
16	User Page				Completed
17	Category Page				Completed
18	Product page				Completed
19	Order page	29/07/2019	2	Chau Thanh Phong	Completed
20	Recipes Page				Completed
21	Feedback page				Completed

Date:

Signature of Instructor:

MS. Le Mong Thuy

Signature of Team Leader:

Bui Van Huong