# **1.0 Introduction**

The aim of this project is to develop a website using Asp.net core razor pages, HTML and CSS. The topic chosen is a website for Golf Club Card sale. It allows users to browse through wide range of Golf Club Cards and search for Cards based on certain attributes like title, description etc. Customers can add clubcards to cart and proceed to place an order. They can also update the cart items or remove them permanently. The website also uses entity framework for the database functionality. Related model classes have been created and data is stored in the DB.

# **2.0 Success Criteria**

|  |  |
| --- | --- |
| Homepage | Complete |
| Product pages/Cart | Complete |
| Login/Register forms | Complete |
| Admin pages(Add items, Edit, Delete, View details of golfclub cards) | Complete |
| Contact us page | Complete |

# **3.0 Initial Design**

* The target audience for the website would be irrespective gender and the website is based on Golf Club Cards, hence we have been kept Premium Cards to give an attractive look.

**Note**: User which we registered first time will be our admin and keep the credentials safe after running commands in PackageManagerConsole for database migration and commands you can find in readme.txt file in the project itself

# **4.0 Feature Development**

## **Feature : Homepage**

The homepage consists of links to Privacy and Manage Catalogue pages.

**Admin Dashboard:**

A picture containing graphical user interface

Description automatically generated

### **Image slideshow**

This page consists of a slide show container for 3 images. There is a javascript method that gets the ui element based on the class. It then sets the display for all the classes as “None”. At a time only one section is displayed. This is achieved using another counter variable that keeps incrementing.

document.getElementsByClassName("mySlides") - method to load the sections with class name mySlides

for (i = 0; i < slides.length; i++) {

slides[i].style.display = "none";

}

The above for loop hides all the image sections.

slides[slideIndex - 1].style.display = "block"; // This line sets the display as block, as a result we can see the image.

## **Feature : Login/Register forms for authentication/registration**

There are login and register pages added to the website. Also, there are two roles being created for admin and customers. Users can register by providing their first, last names, email id and password. The login and register links are available on all the pages for users to login any time. These register and login pages are added by creating new scaffolded items.

**Sample screenshot for Registration form**

Graphical user interface, text, application, email

Description automatically generated

**Sample screenshot for login form**

Graphical user interface, text, application, email

Description automatically generated

## **Feature : Admin pages for adding/updating products**

Using the ClubCards entity model, pages for database operations were created. There are separate views for viewing, creating and updating the table. Records are inserted using the OnPost() method and details are fetched in the OnGet() method.

**OnGet() method sample code:**

ClubCards = await \_context.ClubCardsList.ToListAsync();

The above code retrieves records from ClubCardList table.

**OnPost() method sample code:**

context.ClubCardList.Add(ClubCards);

await \_context.SaveChangesAsync();

Sample screenshot for added clubcards by admin and delete option as well

Graphical user interface, text, email

Description automatically generated

This code sample created a table row in the clubcardlist database with the passed object value ClubCards.

**Sample screenshot for admin page to add product**

Graphical user interface, text, email

Description automatically generated

**Sample screenshot for ContactUs page**

Graphical user interface, text

Description automatically generated

**Sample screenshot for Edit the club card details which is already added by admin**

Graphical user interface, text

Description automatically generated

**Sample screenshot for view the club card details which is already added by admin**

Graphical user interface, text, email

Description automatically generated

**Sample screenshot for Index page for customer after successful login**

A screenshot of a video game

Description automatically generated with low confidence

**Sample screenshot for Clubcards added by admin are reflecting in customer login and to place an order on the same**

Graphical user interface, website

Description automatically generated

**Sample screenshot after selecting a GolfClubCard by the customer to purchase**

Graphical user interface

Description automatically generated

## **Feature : Product page/Cart Functionality**

For displaying all the clubcards on the web pages, database is required to store related details namely clubcard images, prices etc. Hence, I have installed the required entity framework dependencies and created relevant model class for creating the EF table for golfclubcard entity. I have also created the data context file ApplicationDbContext.cs and added started adding the object collections for creating the tables.

Here is my model class for ClubCards:

public class ClubCards

{

[DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int Id { get; set; }

public string Title { get; set; }

public string Description { get; set; }

public int Price { get; set; }

public string Image { get; set; }

}

**ViewClubCardList.cshtml: View behind the page displaying ClubCardList**

@foreach (var clubCard in Model.ClubCards)

{

<section class="clubCard">

<a asp-page="ViewClubCards" asp-route-id="@clubCard.Id">

<img src="./images/@clubCard.Image">

</a>

<h6>@clubCard.Title</h6>

<p>GBP @clubCard.Price</p>

</section>

}

**ViewClubCardList.cshtml.cs: Model class behind the page displaying ClubCardList**

The OnGet() method in this model class gets the data from ClubCardList table and binds it to the ClubCardList data member created in this class. As a result of the data binding, the view can access ClubCardList object values using @ tag.

clubCardList = \_db. ClubCardList.ToList();

**Note**: Sessions were handled by SessionObjectHandler.cs

public static class SessionObjectHandler

{

public static void SetObjectAsJsonString(this ISession session, string key, object value)

{

session.SetString(key, JsonConvert.SerializeObject(value));

}

public static T GetObjectFromJsonString<T>(this ISession session, string key)

{9

var value = session.GetString(key);

return value == null ? default(T) : JsonConvert.DeserializeObject<T>(value);

}

}

The values stored in session will be retrieved in all the web pages by using the below code snippet

clubCardInCart = SessionObjectHandler.GetObjectFromJsonString<List<ClubCardInCart>>(HttpContext.Session, "shoppingCardbag");

**Sample screenshot for shopping cart page after adding an club card to the item**

Graphical user interface, text

Description automatically generated

**Sample screenshot after placing an order by customer**

A picture containing shape

Description automatically generated