Centre Name: ACE-HCMC-2-FPT.

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

ONLINESHOP4DVDS

Design Document

Supervisor:	Mr. Le Tuan Xuyen	
Semester:	3	
Batch No:	T1.2303.M0	
Group No:	3	
Order:	**Full name**	**Roll No.**
1.	Bùi Trần Nguyên Hưng	Student1462831
2.	Trương Nguyễn Quang Huy	Student1468180
3.	Nguyễn Anh Khoa	Student1459131
4.	Bùi Thị Nhung	Student1459130
5.	Vũ Minh Tuấn	Student1459120

- **This is to certify that**

Mr. Bùi Trần Nguyên Hưng

Mr. Trương Nguyễn Quang Huy

Mr. Nguyễn Anh Khoa

Mr. Vũ Minh Tuấn

Ms. Bùi Thị Nhung

have successfully designed & developed:

eProject:ONLINESHOP4DVDS

- **Submitted by:**

Mr. Le Tuan Xuyen

Date of issue: 02/08/2024

Authorized Signature:

Table of Contents

REVIEW 1

I. Introduction

We'd like to thank you everyone on behalf of Group 3 for your efforts and support in bringing this eProject to a successful conclusion. We received a lot of assistance from our teacher, Mr. Le Tuan Xuyen, starting on the first day of Sem 3 at FPT Aptech. He has led us in this eProject with his zeal and discipline, and it is thanks to him that we will be able to complete this project. Therefore, Mr. Xuyen, let us say once more how much we appreciate your commitment and leadership.

In addition, we want to thank the entire group 3 team for their dedication to finishing this project over the course of four weeks.

Last but not least, we would like to express our gratitude to the Aptech eProject team for giving us the chance to put what we had learned into practice and obtain practical experience that would be valuable for our future careers.

II. Synopsis:

The Objective of this program we aim is to give a sample project to work on real life projects. These applications help us build a larger, more robust application.

The objective is not only to teach us C Sharp, .Net, ReactJS but also to provide us with a real life scenario and help us create basic applications using the tools. We can enhance our skills and value with the project.

Music, Game, Movie is an important part of our life. Nowadays, listening to music online is hobby of most modern people. So you need to build website where users will get relevant information about the latest music, game, movie releases in the market. This website has web pages that give information about the new releases in the field of music, game, and movie in the National market.

It is very essential for us to have a clear understanding of the subject. We think we should go through the project and solve the assignments as per requirements given.

III. Problem Definition:

Music, Game, Movie is an important part of our life. Nowadays, listening to music online is hobby of most modern people. So you need to build website where users will get relevant information about the latest music, game, movie releases in the market. The site will allow users to view all kinds of music, game, movie - includes National and International, listen to some music, see trailers game, movie, and download them for free some but not full, vote for an album, game, movie and send feedback about the site, etc. In addition, this website also provides a forum that users could discuss. The website has search and browse functions too.

When users need to know information about recent releases of music, game, movie they can open this website to see or search some information, and naturally, they can buy some DVDs if they want.

If users want to know information regarding music, game, movie or they wish to buy a new DVDs, they need not visit a shop at all. Instead of this, they can just open the website to refer to information about music displayed on the web pages and place an order. Finally, what's thing that we want customers: come to see, and buy new releases albums.

IV. Customer's Requirement Specifications

A web-site incorporating all the information about the ONLINESHOP4DVDS services is required with following functionality

There should be a menu categorizing the various aspects of the ONLINESHOP4DVDS. The detailed menu options are as mentioned below.

1. Home Page:

a. Header – All Page for client will extend the header:

- There is the 1st navbar including the following:
  - Logo on the left side
  - A search bar is next to the Logo
  - A cart and account login are on the right side
- There will be a 2nd navbar including 5 Tabs:
  - Home
  - Album
  - Movie
  - Game
  - News
  - About
  - Support

b. Body:

- First section is a Slider, will the current seasonal promotion.
- Second section is a 4 respective areas showing promotion or preorder for products of each categories.
- Third section will show 3 areas for each main categories (Music, Game, Movie)
- Section Announcements will show the winner of the contest
- Last section is a slider showing all featured brands/artists.

c. Footer – All Page for client will extend the Footer:

- Show all information of owner (owner email, all services, Quick links)
- At the end of the footer, will have a bar, on the left will show Copyright info, on the right will show Developed info.

2. Login(Register) Page:

- This page will show the form to register or login.

3. Game/Album/Movie Page:

- In this page, will has sections about type of game
- Each product detail page will show information on the specific game such as Price, promotion (if any), description, trailer, etc…
- Review: will show reviews from registered user who bought the product.

4. Contest Page:

- It should provide the form available to fill so users can participate in a contest.

5. About us/Contact Page:

- This page will display the information on the store and contact information.

6. Customer' Page:

- This page will display all the information of a customer account.
- User can edit, remove, or view their own transaction history.

7. Wishlist Page:

- The user can view their personal wishlist. Update their favorite products in the list.

8. Feedback Page:

- The visitor can send email to feedback about the views related to the site.

9. Dashboard Page:

In this page, admins can manage these following functions:

a. Manage Categories :

This function contains view categories, adding a new category, deleting, editing, search.

b. Manage Artists:

- Artist is a composer or a singer or both.
- This function contains view artists, adding a new artist, deleting, editing, search.

c. Manage Albums:

- This function contains view albums, adding a new album, deleting, editing, search.

d. Manage songs, games, movies:

- This function contains view, adding a new, deleting, editing, search.

e. Manage Users:

- User can be a guest or a member or a customer.
- A user hasn't registered at website is a Guest.
- A user has already registered at website is a Member.
- Member is called a Customer if he buys album from the website or he had an account at Website.
- This function contains view users, adding a new user, deleting, editing, search and recovery user.

f. Manage Feedback:

- This function contains view, deleting, reply and search feedback of users.

g. Manage Suppliers:

- Supplier is company which supplies music's discs for website.
- This function contains view suppliers, adding a new supplier, deleting, editing, search and recovery supplier.

h. Manage Producers:

- Producer is a company which publishes.

- This function contains view producers, adding a new producer, deleting, editing, search and recovery producer.

i. Manager Permissions:

- Permission of website is divided into many module functions. Each module function has own functions such as view, adding, deleting and editing.
- This function contains view permissions of an administrator, assigning permissions, unassigned permissions and editing permission.

j. Manage Products:

- A product is an album which is sold on the website.
- This function contains view products, adding a new product, deleting, editing, search.

***k.* Manage Purchasing Invoices:**

- This function contains view purchasing invoices, adding a new purchasing invoice, deleting, editing, search.

l. Manage Orders:

- This function contains view orders, deleting, editing, search.

m. Manage Reports:

- This function helps admin can view reports.

n. Manage Advertising:

- This function contains changing banner of website, and changing all ads of website. The Ads can be placed in any position.

o. Manage News:

- This function contains view, adding, deleting, editing, search and recovery news of music, game, movie. Include National and International news.

p. Manage Administrators:

- Admin manages website and manage other administrators.

- This function contains view administrators, adding a new administrator, deleting, editing, search and recovery administrator.

q. Manage Album Reviews:

- Admin manages all reviews about album of users.
- This function contains adding, deleting, editing and search album reviews.

r. Manage Promotion:

- Admin manages promotion on website.
- This function contains view all promotions and adding new promotion.

s. The registered user:

- Member is a user who registered at website. Signing in the website, a Member can use these below functions:
- All functions of Guest.
- Download music, trailer movie, or movie game.
- Buy DVD on website:
- Write the review about album, movie, game or rating.
- Change profile (view profile, edit profile, change password, cancel account)
- Send feedback.
- Member can manage his Website Account (such as Check Balance, View Transactions Log).
- Member can manage history orders (such as view history orders …).

t. The guest user who is not the member of site:

- Guest is a user who doesn't register at website. He can use these below functions:
- Read news about music, game, movie on website.
- Browse or search information about an artist, an album, a song or a lyric, or see trailer game, movie or information content about them.
- Listen to free trailer movie, and some music online (but not full DVD)

- Join the forum of website.

10. Non-functional Requirement Specifications

- Good using Interface and Experience
- Clean code
- Try hard to improve the performance
- Try limit bugs on this website

V. System Requirement Specifications

The Guest accessing the website for the first time can register right away or just search for a product but lack service. When they want to actually buy a product and pay for it, they need to register to become a customer then all of the service will be available.

1. Server requirements:

a. Hardware:

Component	Requirement
CPU	Processor type: Pentium III-compatible processor or faster Processor speed: Recommended: 1.0 GHz or faster
OS	All OS(Windows ,Linux ,Android ,Mac OS …)
Memory (RAM)	RAM: Minimum: 1 GB Recommended: 2 GB or more Maximum: Operating system maximum
Hard Drive	Free space: Minimum: 10 MB

b. Software:

Component	Requirement
Web Browser	IE 8.0 , Firefox 3.0,Chrome or Higher…..

2. Development Software:

Microsoft SQL server Management Studio

Microsoft Visual Studio 2022

3. Technology:

ASP.Net Core 8.0

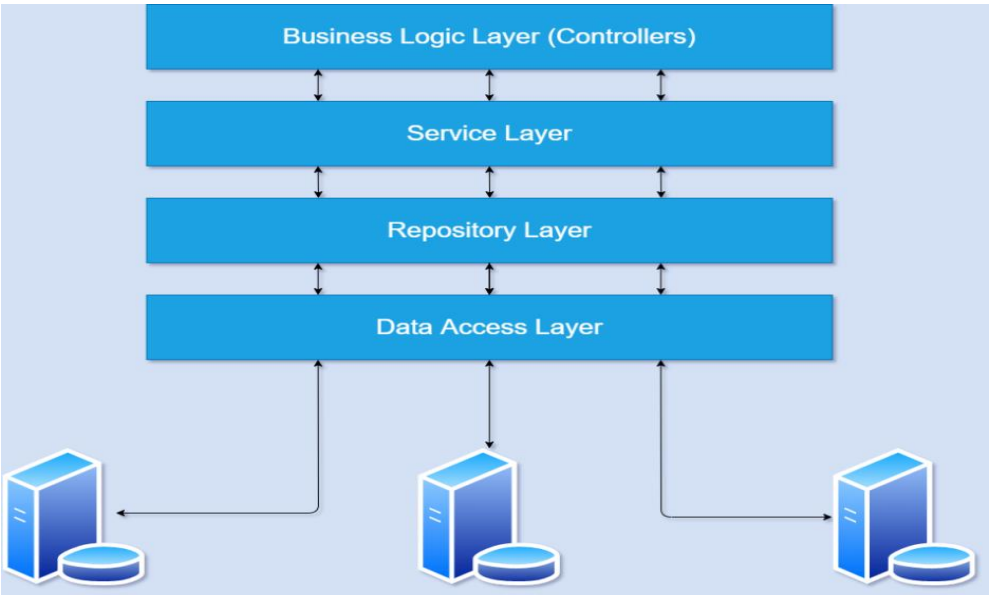
Entity Framework Core 8.0

1. TASK SHEET REVIEW 1

Project Ref. No.: eP/Advertisement Portal Management System/01		Project Title:	Activity Plan Prepared By:	Date of Preparation of Activity Plan:			
Sr. No.	**Task**			**Actual Start Date**	**Actual Days**	**Team Mate Names**	**Status**
1	Introduction	ONLINESHOP4DVDS	All member	05/19/2024	1	Nhung	Completed
2	Problem Definition			05/19/2024	1	Khoa	Completed
3	Requirement & Non-functional requirement			05/19/2024	1	Tuấn & Hưng	Completed
4	Process Analyst & Task sheet review			05/19/2024	1	Tuấn & Huy	Completed

REVIEW 2

I. Architecture & Design of the Project



1. Presentation

- Exposes interaction capabilities for the end users or applications / systems wanting to interact with it.
- Interacts with the Business Logic layer.
- Can interact with other applications (through their presentation layers).

2. Business Logic

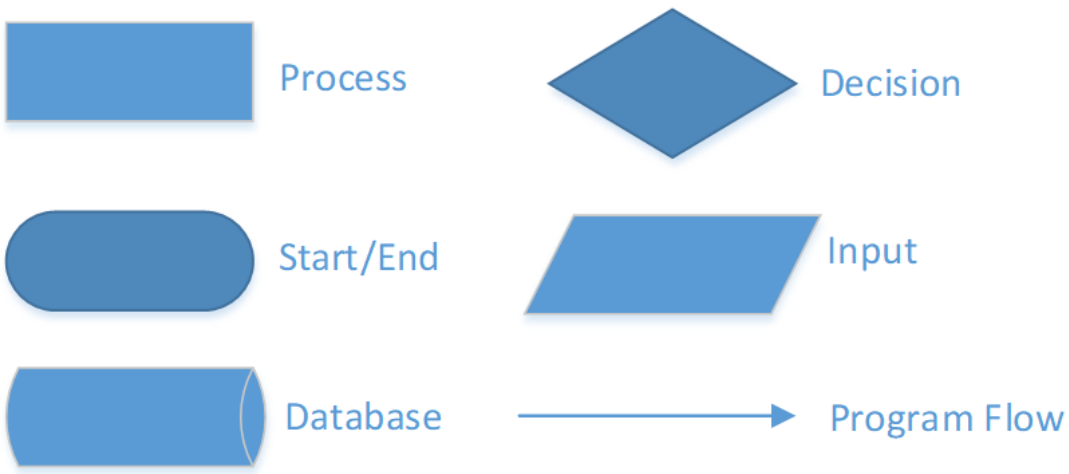
- Expresses domain knowledge, to be utilized from the Presentation layer.
- Interacts with the Data Access layer.
- Can interact with other applications (through their presentation layers).

3. Data Access

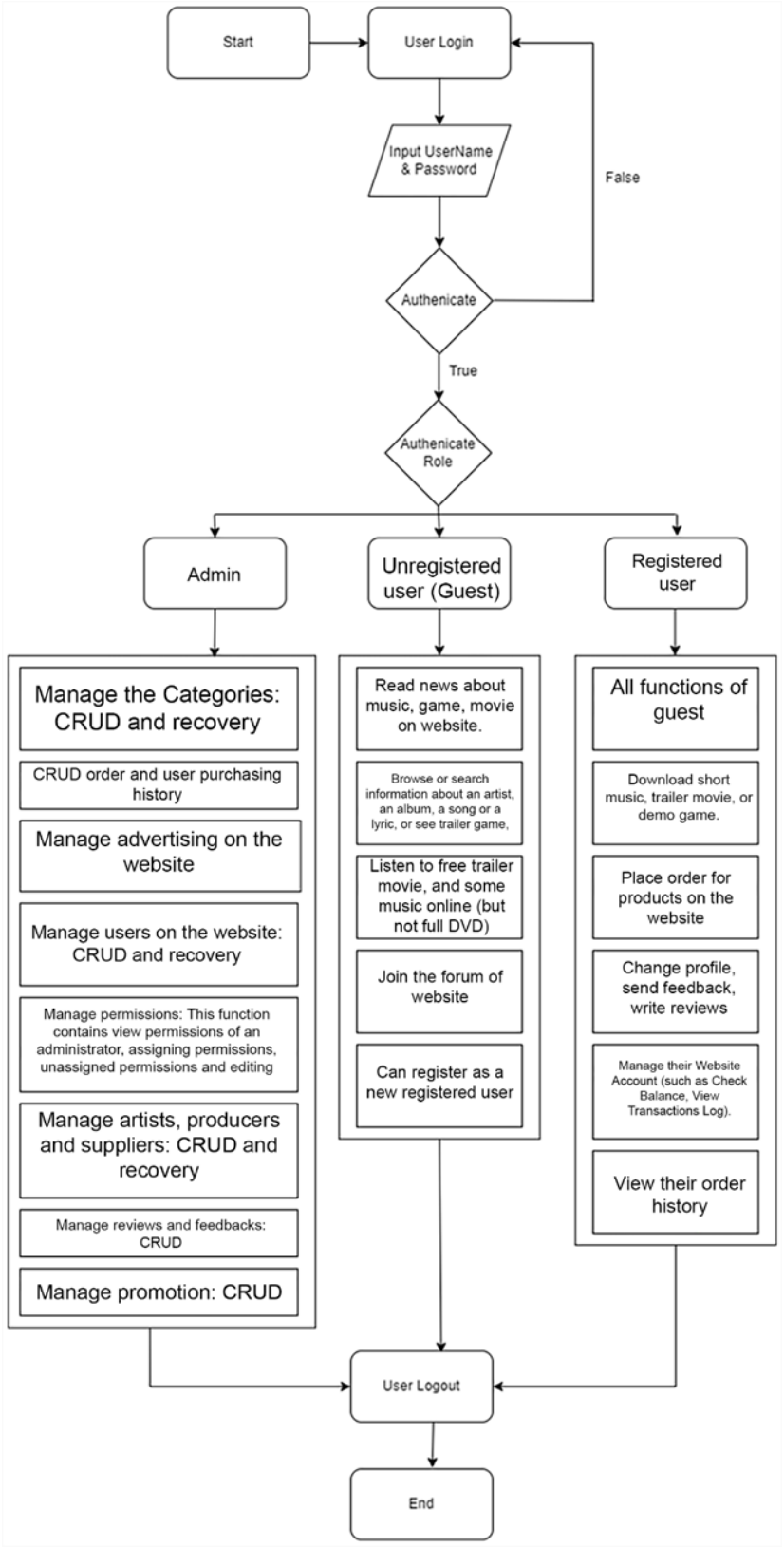
- Acts as a data persistence, to be utilized from Business Logic.
- Can interact with other applications (through their presentation layers).

II. Algorithms – Data Flowchart

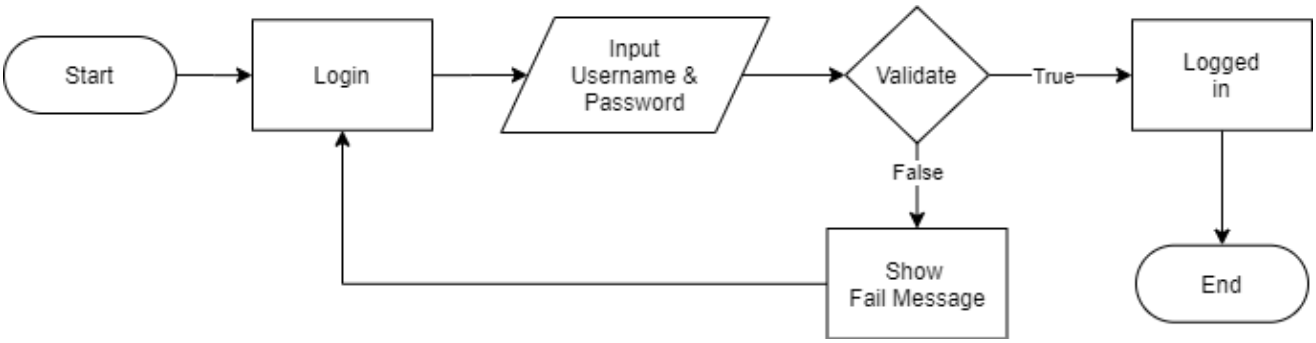
1. Symbol generates:



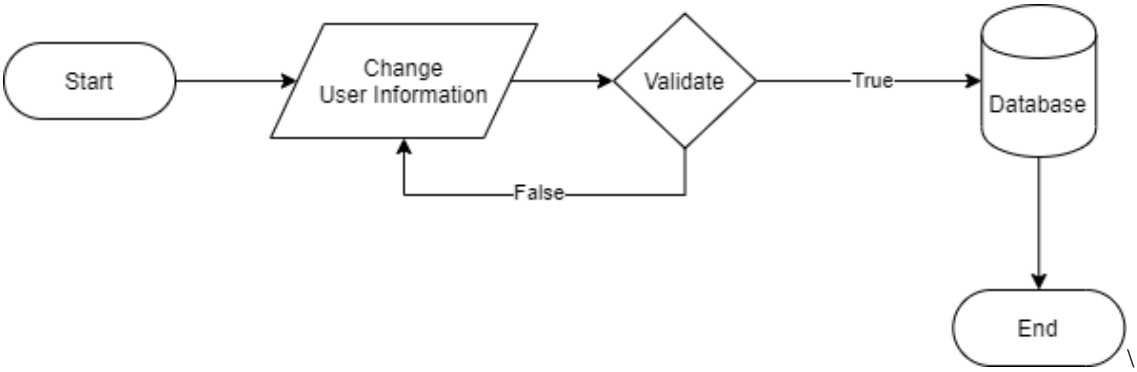
2. Main data flowchart for us



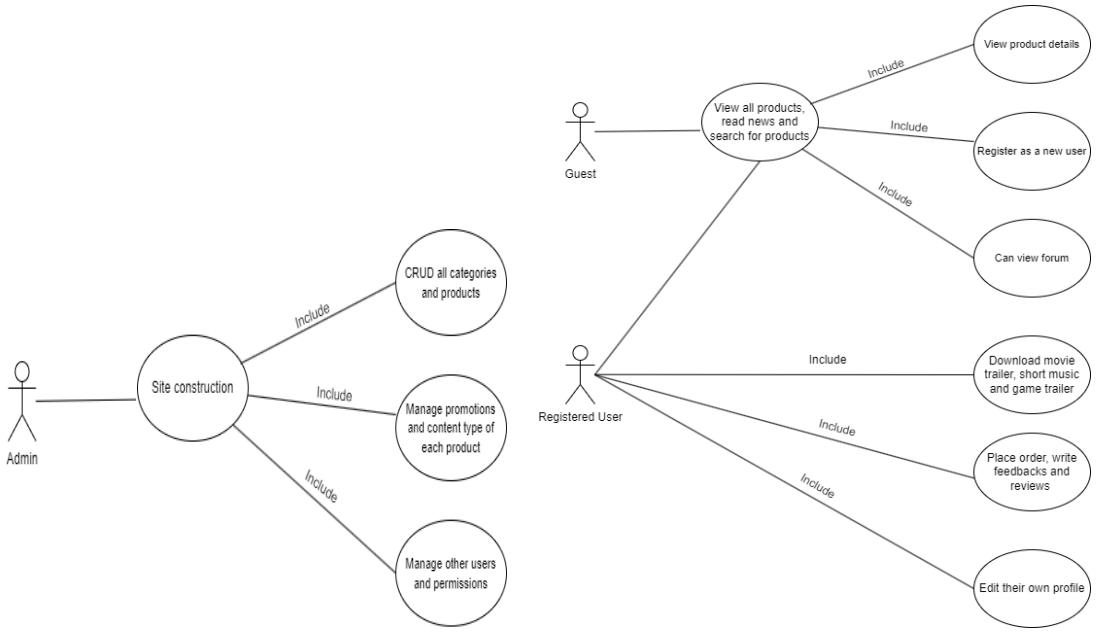
3. Login process (For admins and members)



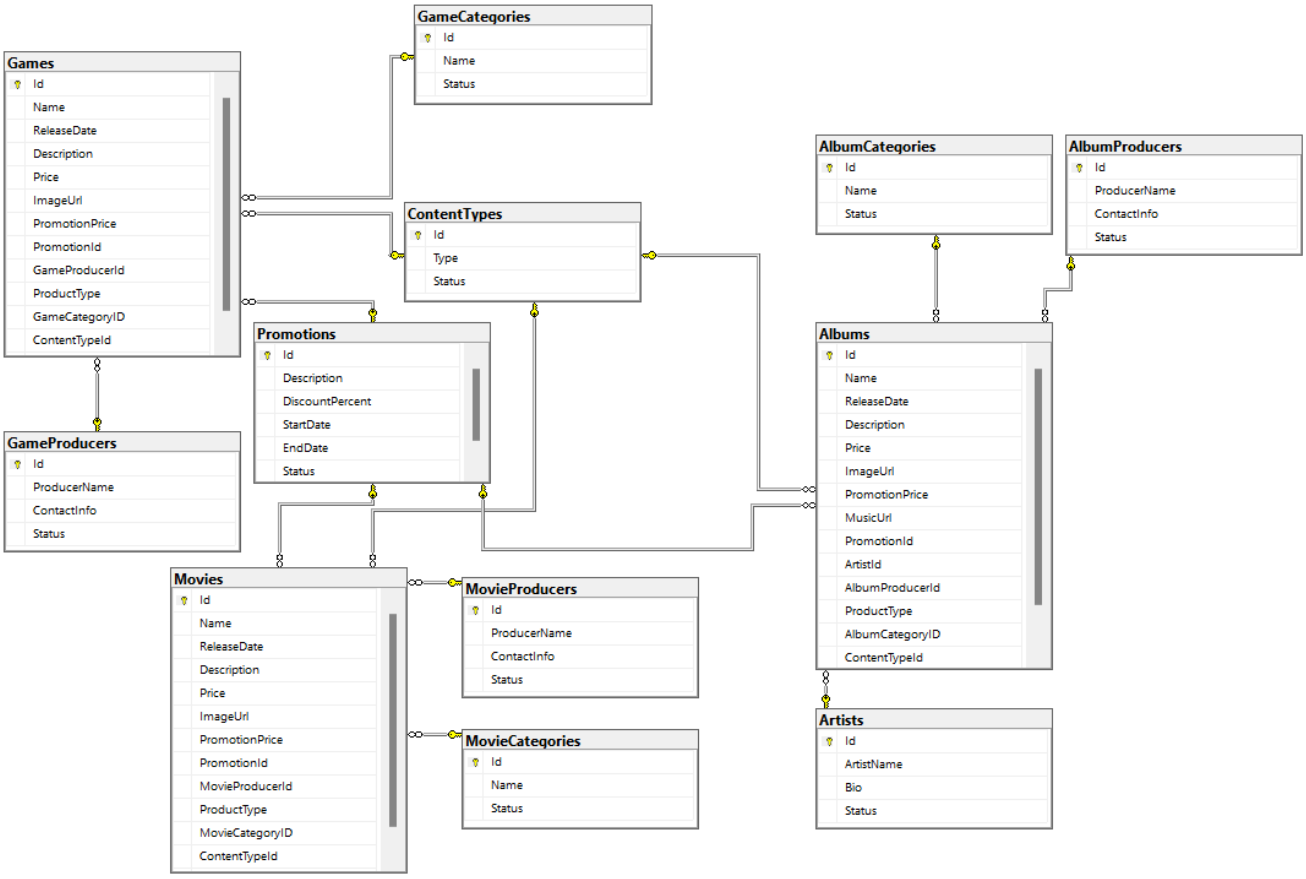
4. Logout process (For admins and members)



III. Use Case Diagram



IV. Entity – Relationship Diagrams



TASK SHEET REVIEW 2

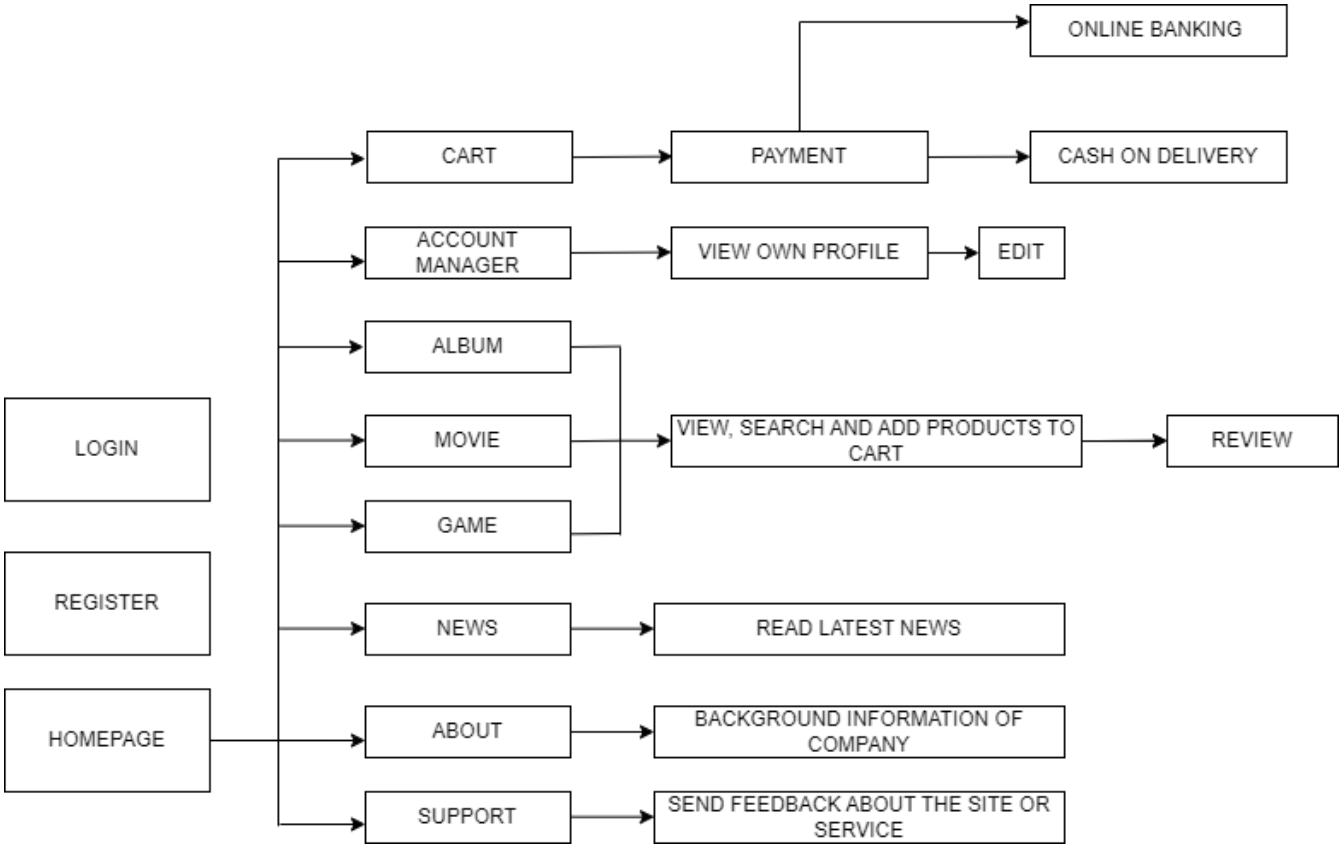
Project Ref. No.: eP/Advertisement Portal Management System/01		Project Title:	Activity Plan Prepared By:	Date of Preparation of Activity Plan:			
Sr. No.	**Task**			**Actual Start Date**	**Actual Days**	**Team Mate Names**	**Status**
1	Architecture & Design of Project	ONLINESHOP4DVDS	All Members	07/29/2024	1	Tuấn	Completed
2	Algorithms – Data Flowchart			07/29/2024	1	Khoa & Huy	Completed
3	Use Case Diagram & Task sheet review			07/29/2024	1	Hưng	Completed
4	Entity-Relationship Diagram			07/29/2024	1	Tuấn	Completed

Date: 07/29/2024	
Signature of Instructor: **Mr. LE TUAN XUYEN**	**Signature of Team Leader:** **VU MINH TUAN**

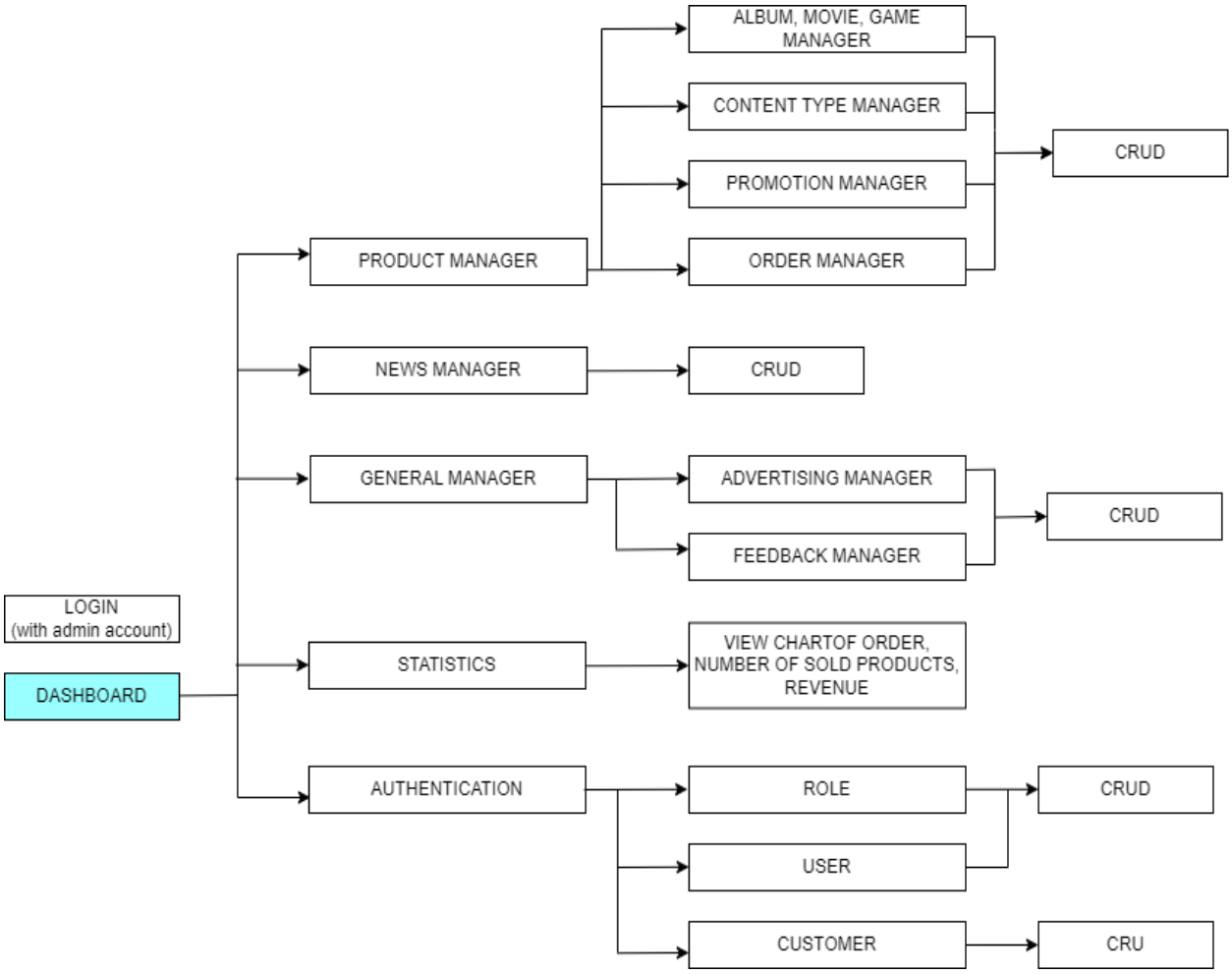
REVIEW 3

I. Site map

1. Client site map

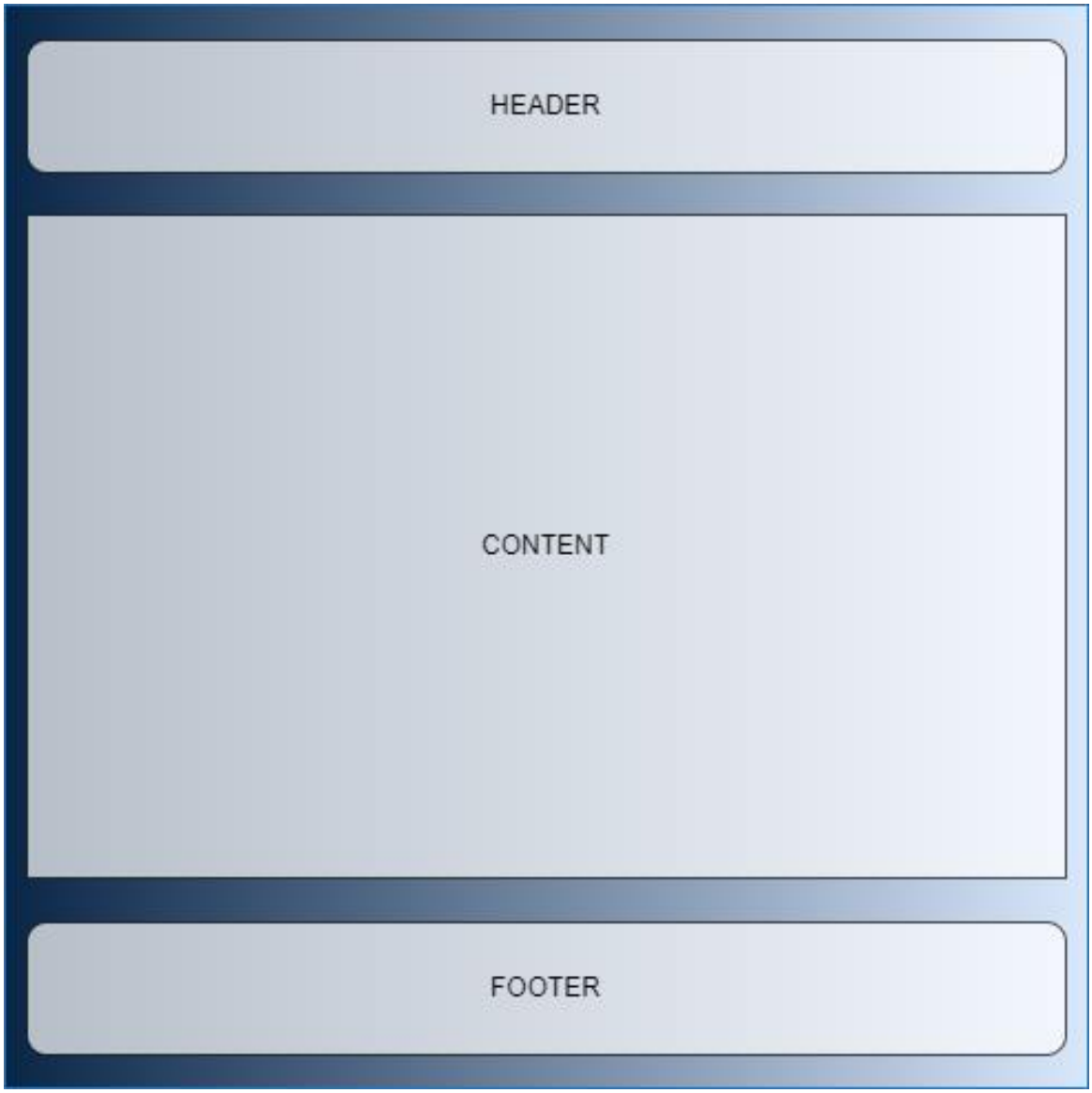


2. Dashboard site map

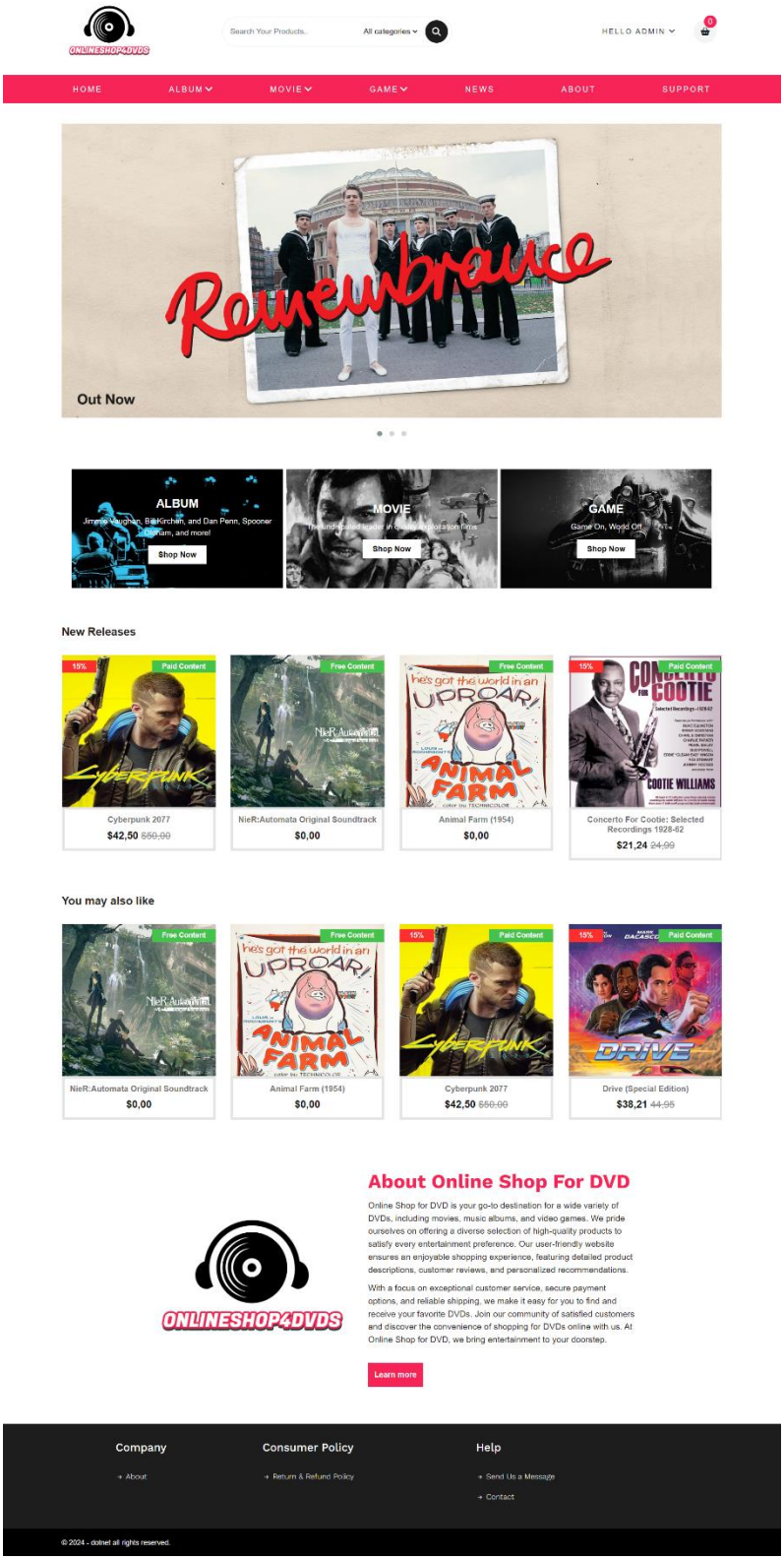


II. Graphical User Interface (GUI) design

1. Client GUI



a. Home page (Index)



Code:

```
<!-- HEADER -->
<header class="float-start w-100">
    <div class="container">
        <div class="row">
            <div class="top-main-haade py-4">
                <div class="logo-main-div me-5">
                    <!-- logo -->
                    <a asp-area="Customer" asp-controller="Home" asp-action="Index" class="logo">
                        <img src="~/images/logo.png" alt="logo" style="width: 75%; height:auto"/>
                    </a>
                </div>
                <div class="search-div-main d-flex align-items-center w-100 ms-5">
                    <div class="seach-div">
                        <form class="form-group d-flex align-items-center" asp-area="Customer" asp-controller="Home" asp-action="MainSearch">
                            <input type="text" name="search" class="form-control" placeholder="Search Your Products.." />
                            <div class="me-3">
                                <select name="productType" class="focus-visible:ring-0" aria-required="true" style="background-color: transparent; border: none; outline: none; box-shadow: none;">
                                    <option value="All categories" selected>All categories</option>
                                    <option value="Album">Album</option>
                                    <option value="Movie">Movie</option>
                                    <option value="Game">Game</option>
                                </select>
                            </div>
                            <div>
                                <button type="submit" class="btn serch-btn">
                                    <i class="fas fa-search"></i>
                                </button>
                            </div>
                        </form>
                    </div>
                </div>
                <div class="right-cart-section">
                    <ul class="d-flex align-items-center justify-content-end lgoin-sectoon-top">
                        <!-- login -->
                        <partial name="_LoginPartial" />

                        <!-- cart -->
                        <li class="dropdown position-relative mx-3">
                            <a class="btn com-link cart-now-icon whilist ml-3" type="button" asp-area="Customer" asp-controller="Cart" asp-action="Index">
                                <i class="fas fa-shopping-basket"></i>
                                @await Component.InvokeAsync("ShoppingCart")
                            </a>
                        </li>
                    </ul>
                </div>
            </div>
        </div>
    </div>
```

```
    <!-- menu -->
    <div class="menu-bar">
        <div class="container">
            <div class="row">
                <nav class="navbar navbar-expand-lg">
                    <div class="container-fluid">
                        <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
                            <span class="navbar-toggler-icon"></span>
                        </button>
                        <div class="collapse navbar-collapse" id="navbarSupportedContent">
                            <ul class="navbar-nav me-auto mb-2 mb-lg-0">
                                <!-- home -->
                                <li class="nav-item dropdown">
                                    <a class="nav-link" role="button" aria-expanded="false" asp-area="Customer" asp-controller="Home" asp-action="Index">
                                        HOME
                                    </a>

                                <!-- album -->
                                <li class="nav-item dropdown position-relative">
                                    <a class="nav-link dropdown-toggle" asp-area="Customer" asp-controller="Album" asp-action="Index" role="button">
                                        Album
                                    </a>
                                    <ul class="dropdown-menu normaldrop">
                                            @foreach (var item in albumCategoryList)
                                            {
                                            <li><a class="dropdown-item" asp-area="Customer" asp-controller="Album" asp-action="Index" asp-route-categoryId="@item.Id">@item.Name</a></li>
                                            }
                                    </ul>
                                </li>

                                <!-- movie -->
                                <li class="nav-item dropdown position-relative">
                                    <a class="nav-link dropdown-toggle" asp-area="Customer" asp-controller="Movie" asp-action="Index" role="button">
                                        Movie
                                    </a>
                                    <ul class="dropdown-menu normaldrop">
                                            @foreach (var item in movieCategoryList)
                                            {
                                            <li><a class="dropdown-item" asp-area="Customer" asp-controller="Movie" asp-action="Index" asp-route-categoryId="@item.Id">@item.Name</a></li>
                                            }
                                    </ul>
                                </li>
```

```
                        <!-- game -->
                        <li class="nav-item dropdown position-relative">
                            <a class="nav-link dropdown-toggle" asp-area="Customer" asp-controller="Game" asp-action="Index" role="button">
                                Game
                            </a>
                            <ul class="dropdown-menu normaldrop">
                                    @foreach (var item in gameCategoryList)
                                    {
                                    <li><a class="dropdown-item" asp-area="Customer" asp-controller="Game" asp-action="Index" asp-route-categoryId="@item.Id">@item.Name</a></li>
                                    }
                            </ul>
                        </li>

                        <!-- news -->
                        <li class="nav-item">
                            <a class="nav-link" asp-area="Customer" asp-controller="News" asp-action="Index" role="button">
                                News
                            </a>
                        </li>

                        <!-- about -->
                        <li class="nav-item">
                            <a class="nav-link" asp-area="Customer" asp-controller="About" asp-action="Index"> About </a>
                        </li>

                        <!-- contact -->
                        <li class="nav-item">
                            <a class="nav-link" asp-area="Customer" asp-controller="Contact" asp-action="Index"> Support </a>
                        </li>

                    </ul>
                </div>
            </div>
        </nav>
    </div>
</div>
</header>
```

```
<div id="layoutSidenav_content">
    <!-- BODY -->
    <main class="float-start w-100 main-body">
        <section class="d-inline-block w-100">
            <div class="container">
                <partial name="_ToastrNotification" /> @* thông báo dùng chung cho mọi trang *@
                <partial name="_SwalNotification" />
                @RenderBody()
            </div>
        </section>
    </main>

    <!-- FOOTER -->
    <footer>
        <div class="main-footer01 d-inline-block w-100 pt-5">
            <div class="container">
                <div class="row row-cols-2 row-cols-lg-4">
                    <div class="col d-grid justify-content-lg-center">
                        <div class="comon-footer-mns d-inline-block w-100">
                            <h5> Company </h5>

                            <ul class="main-links mt-5">
                                <li class="d-flex align-items-center">
                                    <a asp-area="Customer" asp-controller="About" asp-action="Index">
                                        <span>
                                            <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-arrow-right-short" viewBox="0 0 16 16">
                                                <path fill-rule="evenodd" d="M4 8a.5.5 0 0 1 .5-.5h5.793L8.146 5.354a.5.5 0 1 1 .708-.708l3 3a.5.5 0 0 1 0 .708l-3 3a.5.5 0 0 1-.708-.708L10.293 8.5H4.5A.5.5 0 0 1 4 8z" />
                                            </svg>
                                        </span>
                                        About
                                    </a>
                                </li>
                            </ul>
                        </div>
                    </div>
                    <div class="col  d-grid justify-content-lg-center">
                        <div class="comon-footer-mns d-inline-block w-100">
                            <h5> Consumer Policy </h5>

                            <ul class="main-links mt-5">
                                <li class="d-flex align-items-center">
                                    <a href="#">
                                        <span>
                                            <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-arrow-right-short" viewBox="0 0 16 16">
                                                <path fill-rule="evenodd" d="M4 8a.5.5 0 0 1 .5-.5h5.793L8.146 5.354a.5.5 0 1 1 .708-.708l3 3a.5.5 0 0 1 0 .708l-3 3a.5.5 0 0 1-.708-.708L10.293 8.5H4.5A.5.5 0 0 1 4 8z" />
                                            </svg>
                                        </span>
                                        Return & Refund Policy
                                    </a>
                                </li>
```

```
                            </div>
                        </div>
                        <div class="col d-grid justify-content-lg-end">
                            <div class="comon-footer-mns d-inline-block w-100">
                                <h5> Help </h5>
                                <ul class="main-links mt-5">
                                    <li class="d-flex align-items-center">
                                        <a asp-area="Customer" asp-controller="Contact" asp-action="Index">
                                            <span>
                                                <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-arrow-right-short" viewBox="0 0 16 16">
                                                    <path fill-rule="evenodd" d="M4 8a.5.5 0 0 1 .5-.5h5.793L8.146 5.354a.5.5 0 1 1 .708-.708l3 3a.5.5 0 0 1 0 .708l-3 3a.5.5 0 0 1-.708-.708L10.293 8.5H4.5A.5.5 0 0 1 4 8z" />
                                                </svg>
                                            </span>
                                            Send Us a Message
                                        </a>
                                    </li>
                                    <li>
                                        <a href="#">
                                            <span>
                                                <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-arrow-right-short" viewBox="0 0 16 16">
                                                    <path fill-rule="evenodd" d="M4 8a.5.5 0 0 1 .5-.5h5.793L8.146 5.354a.5.5 0 1 1 .708-.708l3 3a.5.5 0 0 1 0 .708l-3 3a.5.5 0 0 1-.708-.708L10.293 8.5H4.5A.5.5 0 0 1 4 8z" />
                                                </svg>
                                            </span>
                                            Contact
                                        </a>
                                    </li>
                                </ul>
                            </div>
                        </div>
                    </div>
                </div>
                <div class="last-footer d-inline-block mt-5 w-100">
                    <div class="container">
                        <div class="d-md-flex coply align-items-center justify-content-between">
                            <p> © 2024 - dotnet all rights reserved.</p>
                        </div>
                    </div>
                </div>
            </div>
        </footer>
```

b. Album/Game/Music

We will be using Album as example to show our page and code in the following section.

```
{
    [Area("Admin")]
    [Authorize(Policy = "NotCustomerPermission")]
    1 reference
    public class AlbumController : Controller
    {
        private readonly IUnitOfWork _unitOfWork;
        private readonly IWebHostEnvironment _webHostEnvironment;

        0 references
        public AlbumController(IUnitOfWork unitOfWork, IWebHostEnvironment webHostEnvironment)
        {
            _unitOfWork = unitOfWork;
            _webHostEnvironment = webHostEnvironment;
        }

        0 references
        public IActionResult Index()
        {
            return View();
        }

        [Authorize(Policy = "AdminOrCreatePermission")]
        0 references
        public async Task<IActionResult> Create()
        {
            var promotionList = await _unitOfWork.PromotionRepository.GetAll(p => p.Status == StaticDetail.PromotionStatus_Active);

            var selectList = promotionList.Select(p => new PromotionViewModel
            {
                Id = p.Id,
                DiscountPercent = p.DiscountPercent
            }).ToList();

            ViewBag.PromotionList = selectList;

            var artistList = await _unitOfWork.ArtistRepository.GetAll(a => a.Status == false);
            ViewBag.ArtistList = new SelectList(artistList, "Id", "ArtistName", "Id");

            var albumProducerList = await _unitOfWork.AlbumProducerRepository.GetAll(p => p.Status == false);
            ViewBag.AlbumProducerList = new SelectList(albumProducerList, "Id", "ProducerName", "Id");

            var albumCategoryList = await _unitOfWork.AlbumCategoryRepository.GetAll(c => c.Status == false);
            ViewBag.AlbumCategoryList = new SelectList(albumCategoryList, "Id", "Name", "Id");

            var contentTypeList = await _unitOfWork.ContentTypeRepository.GetAll(c => c.Status == false);
            ViewBag.ContentTypeList = new SelectList(contentTypeList, "Id", "Type", "Id");

            return View(new Album());
        }
```

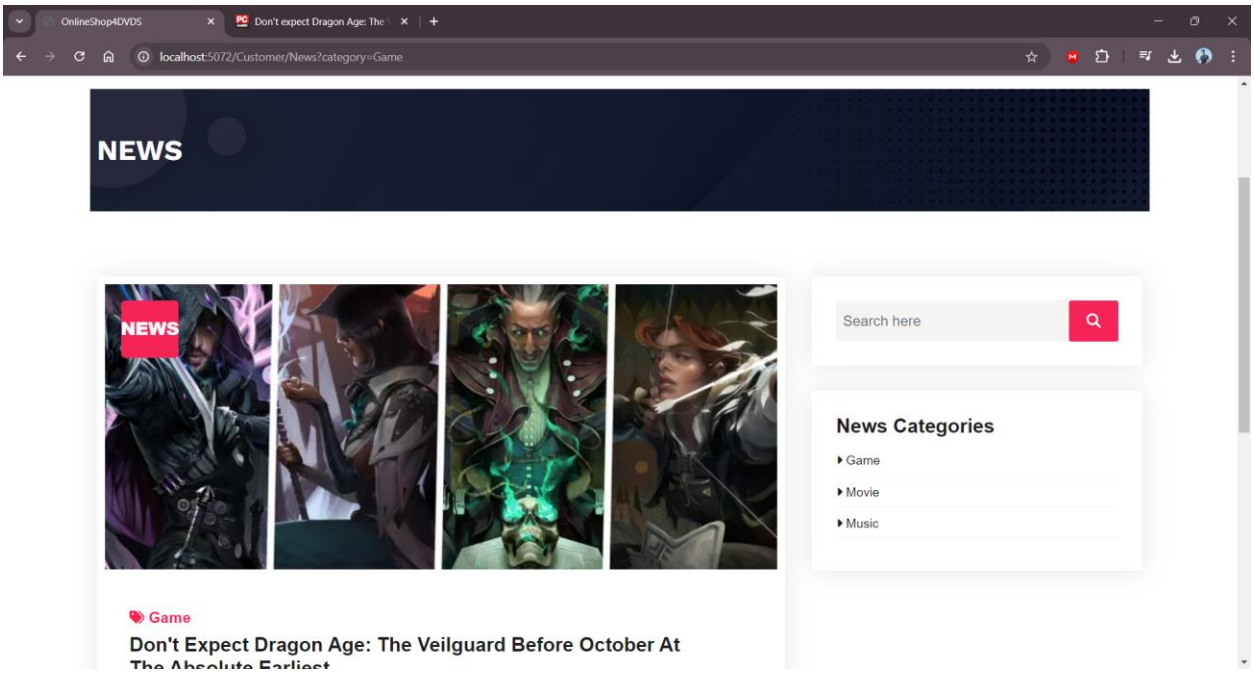
Example Code (in this case we use the Album Controller as example)

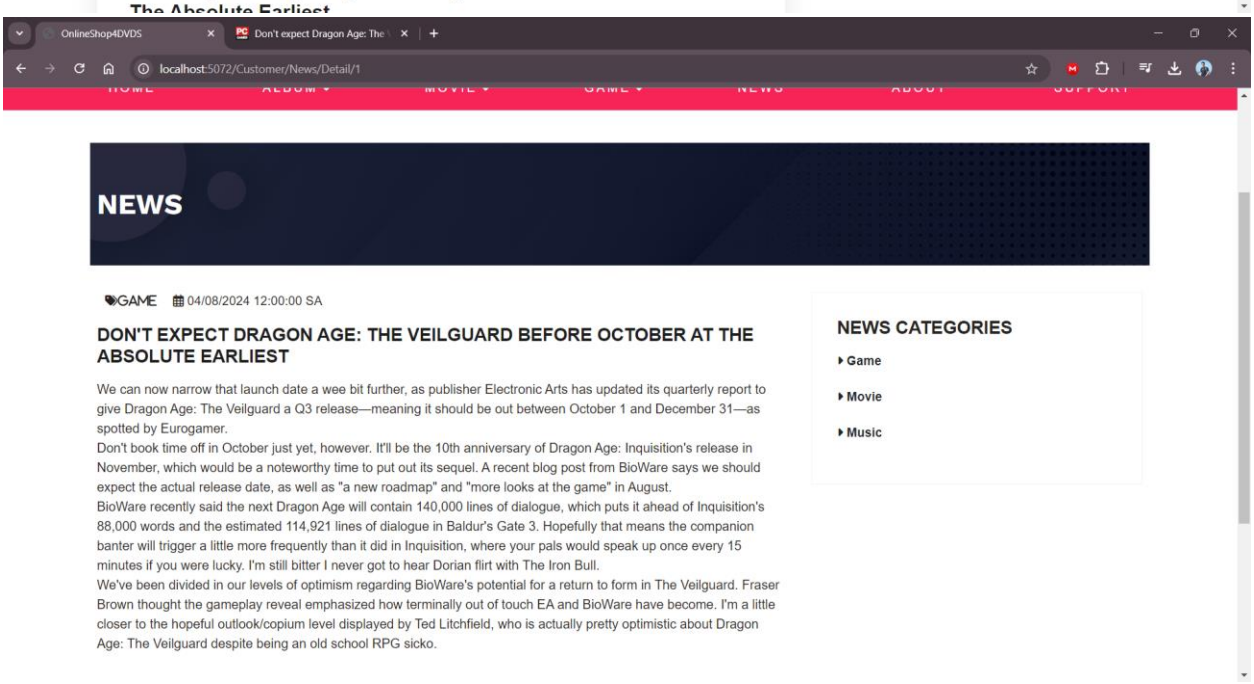
```
[HttpPost]
0 references
public async Task<IActionResult> Create(Album album, IFormFile? MainImage, IFormFile? File)
{
    try
    {
        if (ModelState.IsValid)
        {
            // Create album with no image
            album.ProductType = StaticDetail.ProductType_Album;
            await _unitOfWork.AlbumRepository.Add(album);
            await _unitOfWork.Save();
            TempData["success"] = "Album created successfully";

            // Update ImageUrl to album
            if (MainImage != null)
            {
                string ImageUrl = await UploadImage.SaveImage(album.Id, album.ProductType, MainImage, _webHostEnvironment);
                // Update image
                album.ImageUrl = ImageUrl;

                _unitOfWork.AlbumRepository.Update(album);
                await _unitOfWork.Save();
            }

            // Update FileUrl to album
            if (File != null)
            {
                string FileUrl = await UploadFile.SaveFile(album.Id, album.ProductType, File, _webHostEnvironment);
                // Update image
                album.FileUrl = FileUrl;

                _unitOfWork.AlbumRepository.Update(album);
                await _unitOfWork.Save();
            }

            TempData["success"] = "Album created successfully";
            return RedirectToAction("Index");
        }
    }
    catch (Exception ex)
    {
        if (ex.InnerException is SqlException sqlEx && sqlEx.Number == 2601) // Unique constraint error number
        {
            ModelState.AddModelError("AlbumName", "Album with the same name already exists.");
        }
        else
        {
            ModelState.AddModelError("", "An error occurred during this operation");
        }
    }
    var promotionList = await _unitOfWork.PromotionRepository.GetAll(p => p.Status == StaticDetail.PromotionStatus_Active);

    var selectList = promotionList.Select(p => new PromotionViewModel
    {
        Id = p.Id,
        DiscountPercent = p.DiscountPercent
    }).ToList();

    ViewBag.PromotionList = selectList;

    var albumProducerList = await _unitOfWork.AlbumProducerRepository.GetAll(m => m.Status == false);
    ViewBag.AlbumProducerList = new SelectList(albumProducerList, "Id", "ProducerName", "Id");

    var artistList = await _unitOfWork.ArtistRepository.GetAll(a => a.Status == false);
    ViewBag.ArtistList = new SelectList(artistList, "Id", "ArtistName", "Id");

    var albumCategoryList = await _unitOfWork.AlbumCategoryRepository.GetAll(m => m.Status == false);
    ViewBag.AlbumCategoryList = new SelectList(albumCategoryList, "Id", "Name", "Id");

    var contentTypeList = await _unitOfWork.ContentTypeRepository.GetAll(c => c.Status == false);
    ViewBag.ContentTypeList = new SelectList(contentTypeList, "Id", "Type", "Id");
    return View(album);
}
```

```
[Authorize(Policy = "AdminOrEditPermission")]
0 references
public async Task<IActionResult> Edit(int? id)
{
    var album = await _unitOfWork.AlbumRepository.GetFirstOrDefault(m => m.Id == id, includeProperties: "Promotion,AlbumProducer,AlbumCategory,ContentType,Artist");

    var promotionList = await _unitOfWork.PromotionRepository.GetAll(p => p.Status == StaticDetail.PromotionStatus_Active);

    var selectList = promotionList.Select(p => new PromotionViewModel
    {
        Id = p.Id,
        DiscountPercent = p.DiscountPercent
    }).ToList();

    ViewBag.PromotionList = selectList;

    var albumProducerList = await _unitOfWork.AlbumProducerRepository.GetAll(p => p.Status == false);
    ViewBag.AlbumProducerList = new SelectList(albumProducerList, "Id", "ProducerName", "Id");

    var artistList = await _unitOfWork.ArtistRepository.GetAll(a => a.Status == false);
    ViewBag.ArtistList = new SelectList(artistList, "Id", "ArtistName", "Id");

    var albumCategoryList = await _unitOfWork.AlbumCategoryRepository.GetAll(c => c.Status == false);
    ViewBag.AlbumCategoryList = new SelectList(albumCategoryList, "Id", "Name", "Id");

    var contentTypeList = await _unitOfWork.ContentTypeRepository.GetAll(c => c.Status == false);
    ViewBag.ContentTypeList = new SelectList(contentTypeList, "Id", "Type", "Id");

    if (album == null)
    {
        return NotFound();
    }
    return View(album);
}
```

```
[HttpPost]
0 references
public async Task<IActionResult> Edit(Album album, IFormFile? MainImage, IFormFile? File)
{
    try
    {
        if (ModelState.IsValid)
        {
            var albumUpdated = await _unitOfWork.AlbumRepository.GetFirstOrDefault(m => m.Id == album.Id);
            if (albumUpdated == null)
            {
                return NotFound();
            }

            if (MainImage != null)
            {
                UploadImage.DeleteImage(_webHostEnvironment, albumUpdated?.ImageUrl);

                album.ImageUrl = await UploadImage.SaveImage(albumUpdated.Id, albumUpdated.ProductType, MainImage, _webHostEnvironment);
            }

            if(File != null)
            {
                UploadFile.DeleteFile(_webHostEnvironment, albumUpdated?.FileUrl);

                album.FileUrl = await UploadFile.SaveFile(albumUpdated.Id, albumUpdated.ProductType, File, _webHostEnvironment);
            }

            _unitOfWork.AlbumRepository.Update(album);
            await _unitOfWork.Save();
            TempData["success"] = "Album update successfully";
            return RedirectToAction("Index");
        }
    }
    catch (Exception ex)
    {
        if (ex.InnerException is SqlException sqlEx && sqlEx.Number == 2601) // Unique constraint error number
        {
            ModelState.AddModelError("AlbumName", "Album with the same name already exists.");
        }
        else
        {
            ModelState.AddModelError("", "An error occurred during this operation");
        }
    }
    var promotionList = await _unitOfWork.PromotionRepository.GetAll(p => p.Status == StaticDetail.PromotionStatus_Active);

    var selectList = promotionList.Select(p => new PromotionViewModel
    {
        Id = p.Id,
        DiscountPercent = p.DiscountPercent
    }).ToList();

    ViewBag.PromotionList = selectList;

    var albumProducerList = await _unitOfWork.AlbumProducerRepository.GetAll(m => m.Status == false);
    ViewBag.AlbumProducerList = new SelectList(albumProducerList, "Id", "ProducerName", "Id");

    var artistList = await _unitOfWork.ArtistRepository.GetAll(a => a.Status == false);
    ViewBag.ArtistList = new SelectList(artistList, "Id", "ArtistName", "Id");

    var albumCategoryList = await _unitOfWork.AlbumCategoryRepository.GetAll(m => m.Status == false);
    ViewBag.AlbumCategoryList = new SelectList(albumCategoryList, "Id", "Name", "Id");

    var contentTypeList = await _unitOfWork.ContentTypeRepository.GetAll(c => c.Status == false);
    ViewBag.ContentTypeList = new SelectList(contentTypeList, "Id", "Type", "Id");
    return View(album);
}
```

c. News


OnlineShop4DVDS
Don't expect Dragon Age: The
localhost:5072/Customer/News?category=Game
NEWS
NEWS
Search here
News Categories
Game
Movie
Music
Game
Don't Expect Dragon Age: The Veilguard Before October At


OnlineShop4DVDS
Don't expect Dragon Age: The
localhost:5072/Customer/News/Detail/1
NEWS
GAME
04/08/2024 12:00:00 SA
DON'T EXPECT DRAGON AGE: THE VEILGUARD BEFORE OCTOBER AT THE ABSOLUTE EARLIEST
We can now narrow that launch date a wee bit further, as publisher Electronic Arts has updated its quarterly report to give Dragon Age: The Veilguard a Q3 release—meaning it should be out between October 1 and December 31—as spotted by Eurogamer.
Don't book time off in October just yet, however. It'll be the 10th anniversary of Dragon Age: Inquisition's release in November, which would be a noteworthy time to put out its sequel. A recent blog post from BioWare says we should expect the actual release date, as well as "a new roadmap" and "more looks at the game" in August.
BioWare recently said the next Dragon Age will contain 140,000 lines of dialogue, which puts it ahead of Inquisition's 88,000 words and the estimated 114,921 lines of dialogue in Baldur's Gate 3. Hopefully that means the companion banter will trigger a little more frequently than it did in Inquisition, where your pals would speak up once every 15 minutes if you were lucky. I'm still bitter I never got to hear Dorian flirt with The Iron Bull.
We've been divided in our levels of optimism regarding BioWare's potential for a return to form in The Veilguard. Fraser Brown thought the gameplay reveal emphasized how terminally out of touch EA and BioWare have become. I'm a little closer to the hopeful outlook/copium level displayed by Ted Litchfield, who is actually pretty optimistic about Dragon Age: The Veilguard despite being an old school RPG sicko.
NEWS CATEGORIES
Game
Movie
Music

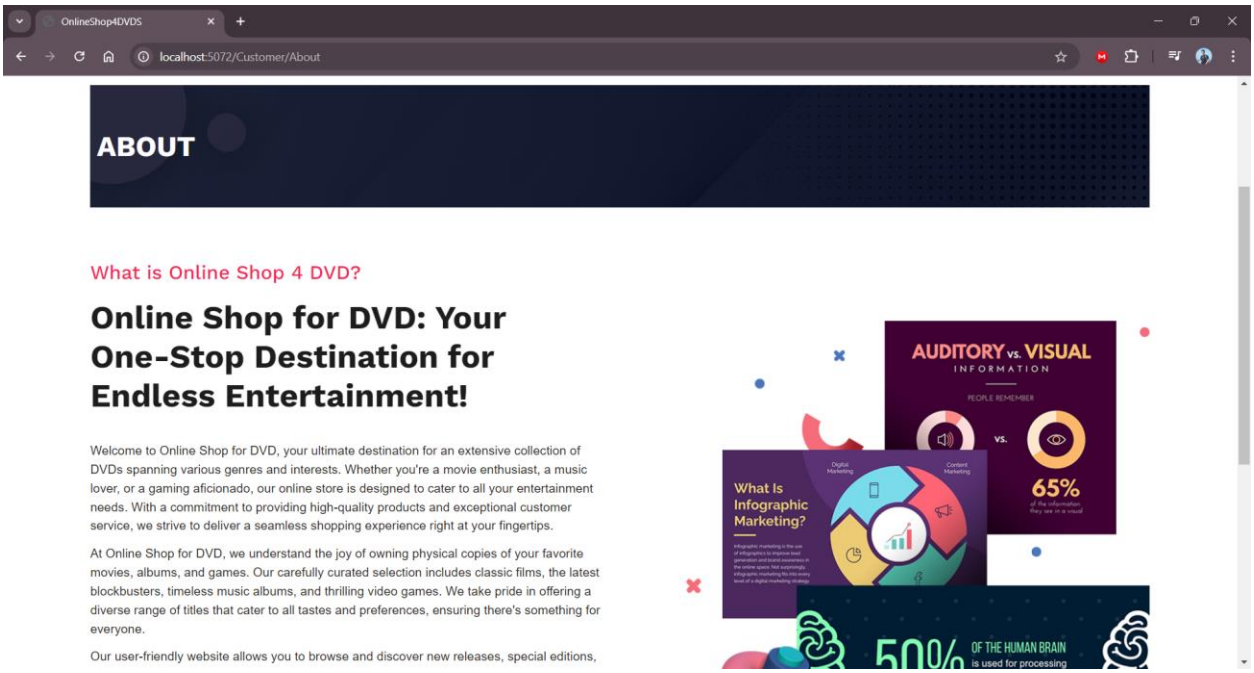
Code:

```
namespace OnlineShop4DVDS.Areas.Admin.Controllers
{
    [Area("Admin")]
    1 reference
    public class NewsController : Controller
    {
        private readonly IUnitOfWork _unitOfWork;
        private readonly IWebHostEnvironment _webHostEnvironment;

        0 references
        public NewsController(IUnitOfWork unitOfWork, IWebHostEnvironment webHostEnvironment)
        {
            _unitOfWork = unitOfWork;
            _webHostEnvironment = webHostEnvironment;
        }

        0 references
        public IActionResult Index()
        {
            return View();
        }

        [Authorize(Policy = "AdminOrCreatePermission")]
        0 references
        public async Task<IActionResult> Create()
        {
            return View(new News());
        }
```

```
[HttpPost]
0 references
public async Task<IActionResult> Create(News news, IFormFile? MainImage)
{
    try
    {
        if (ModelState.IsValid)
        {
            await _unitOfWork.NewsRepository.Add(news);
            await _unitOfWork.Save();
            // Update ImageUrl to news
            if (MainImage != null)
            {
                string ImageUrl = await UploadPhoto.SaveImage(news.Id, news.NewsCategory, MainImage, _webHostEnvironment);
                // Update image
                news.ImageUrl = ImageUrl;

                _unitOfWork.NewsRepository.Update(news);
                await _unitOfWork.Save();
            }

            TempData["success"] = "News created successfully";
            return RedirectToAction("Index", "News", new { area = "Customer", category = news.NewsCategory });
        }
    }
    catch (Exception ex)
    {
        if (ex.InnerException is SqlException sqlEx && sqlEx.Number == 2601) // Unique constraint error number
        {
            ModelState.AddModelError("Title", "News with the same title already exists.");
        }
        else
        {
            ModelState.AddModelError("", "An error occurred during this operation");
        }
    }
    return View(news);
}
```

```
[Authorize(Policy = "AdminOrEditPermission")]
0 references
public async Task<IActionResult> Edit(int? id)
{
    var news = await _unitOfWork.NewsRepository.GetFirstOrDefault(n => n.Id == id);

    if (news == null)
    {
        return NotFound();
    }
    return View(news);
}

[HttpPost]
0 references
public async Task<IActionResult> Edit(News news, IFormFile? MainImage)
{
    try
    {
        if (ModelState.IsValid)
        {
            if (MainImage != null)
            {
                var newsUpdated = await _unitOfWork.NewsRepository.GetFirstOrDefault(m => m.Id == news.Id);
                if (newsUpdated == null)
                {
                    return NotFound();
                }

                UploadPhoto.DeleteImage(_webHostEnvironment, newsUpdated?.ImageUrl);

                news.ImageUrl = await UploadPhoto.SaveImage(newsUpdated.Id, newsUpdated.NewsCategory, MainImage, _webHostEnvironment);
            }

            _unitOfWork.NewsRepository.Update(news);
            await _unitOfWork.Save();
            TempData["success"] = "News update successfully";
            return RedirectToAction("Index");
        }
    }
    catch (Exception ex)
    {
        if (ex.InnerException is SqlException sqlEx && sqlEx.Number == 2601) // Unique constraint error number
        {
            ModelState.AddModelError("Name", "News with the same name already exists.");
        }
        else
        {
            ModelState.AddModelError("", "An error occurred during this operation");
        }
    }

    return View(news);
}
```

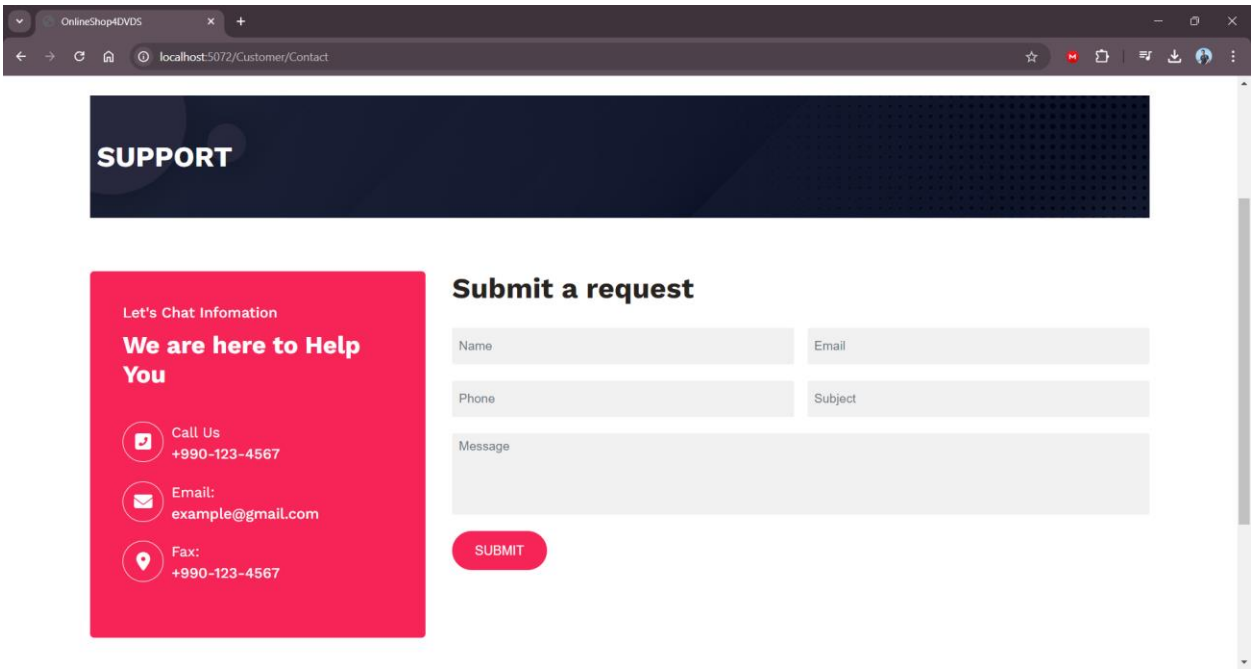
d. About Us



Code

```
<div class="listing-banner-part d-inline-block w-100 mb-5">
    <div class="container">
        <h2 class="text-white"> About </h2>
    </div>
</div>

<div class="row row-cols-1 row-cols-lg-2 align-items-center my-5">
    <div class="col">
        <h5 class="sub-text"> What is Online Shop 4 DVD? </h5>
        <h2 class="main-head-text col-lg-10  mt-4"> Online Shop for DVD: Your One-Stop Destination for Endless Entertainment!  </h2>

        <p class="mt-5">
            Welcome to Online Shop for DVD, your ultimate destination for an extensive collection of DVDs spanning various genres and interests. Wheth
        </p>

        <p class="mt-3">
            At Online Shop for DVD, we understand the joy of owning physical copies of your favorite movies, albums, and games. Our carefully curated
        </p>

        <p class="mt-3">
            Our user-friendly website allows you to browse and discover new releases, special editions, and exclusive collections with ease. We also p
        </p>

        <p class="mt-3">
            Join our community of satisfied customers and experience the convenience and excitement of shopping for DVDs online. At Online Shop for DV
        </p>
    </div>
    <div class="col">
        <figure class="m-0">
            <img alt="seu" src="/images/Template-Header-Infographics.png" />
        </figure>
    </div>
</div>
```

e. Support (Feedback)



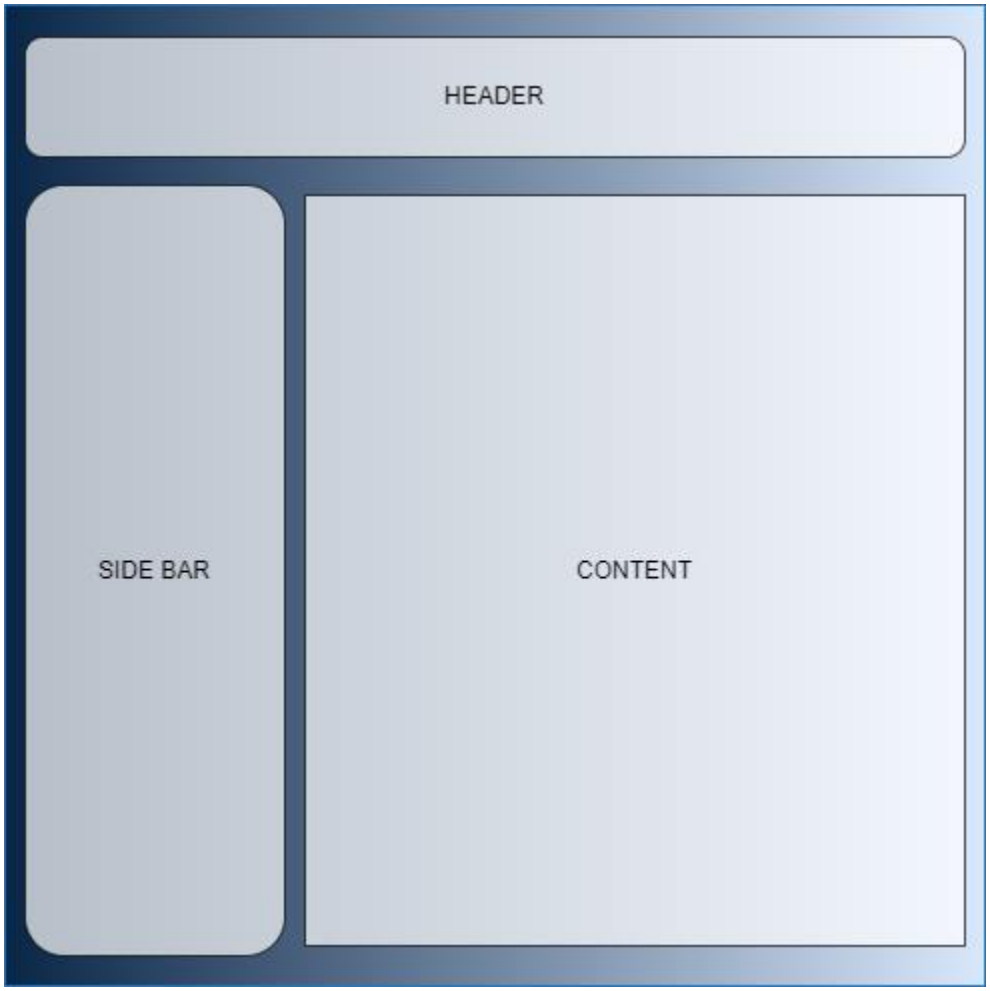
Code:

```
namespace OnlineShop4DVDS.Areas.Admin.Controllers
{
    [Area("Admin")]
    [Authorize(Policy = "NotCustomerPermission")]

    1 reference
    public class FeedbackController : Controller
    {
        private readonly IUnitOfWork _unitOfWork;

        private readonly IEmailSender _emailSender;

        0 references
        public FeedbackController(IUnitOfWork unitOfWork, IEmailSender emailSender)
        {
            _unitOfWork = unitOfWork;
            _emailSender = emailSender;
        }

        0 references
        public IActionResult Index()
        {
            return View();
        }

        [Authorize(Policy = "AdminOrEditPermission")]
        0 references
        public async Task<IActionResult> Reply(int? id)
        {
            var contact = await _unitOfWork.ContactRepository.GetFirstOrDefault(c => c.Id == id);

            if (contact == null)
            {
                return NotFound();
            }
            return View(contact);
        }
```

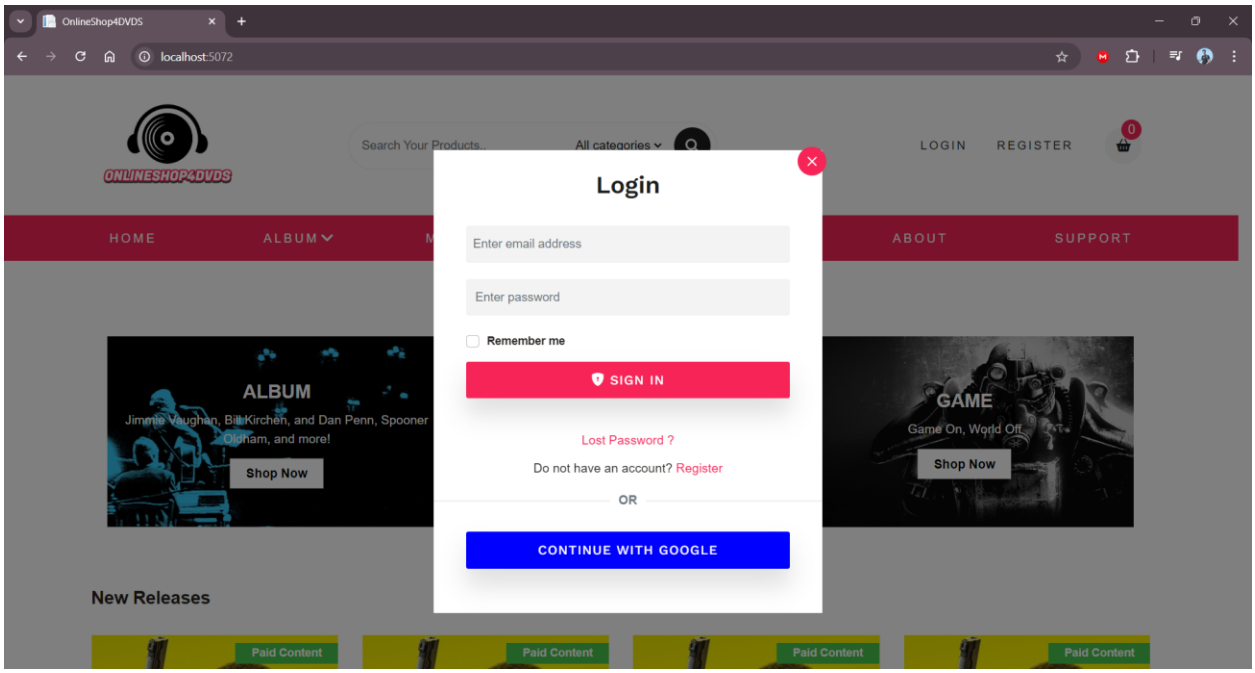
```
        [HttpPost]
        0 references
        public async Task<IActionResult> Reply(Contact contact)
        {
            if (string.IsNullOrWhiteSpace(contact.Reply))
            {
                ModelState.AddModelError("Reply", "Reply cannot be empty");
                return View(contact);
            }

            using (var transaction = new System.Transactions.TransactionScope(System.Transactions.TransactionScopeAsyncFlowOption.Enabled))
            {
                try
                {

                    if (ModelState.IsValid)
                    {
                        var contactCheck = await _unitOfWork.ContactRepository.GetFirstOrDefault(c => c.Id == contact.Id);
                        if (contactCheck == null)
                        {
                            return NotFound();
                        }

                        // Update the reply in the database
                        contactCheck.Reply = contact.Reply;
                        contactCheck.IsSended = true;
                        _unitOfWork.ContactRepository.Update(contactCheck);
                        await _unitOfWork.Save();

                        // Send the email
                        string emailBody = $@"
<!DOCTYPE html>
<html>
<head>
    <meta charset='UTF-8'>
    <title>Reply to Your Feedback</title>
    <style>
        body {{
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
        }}
        .container {{
            width: 100%;
            max-width: 600px;
            margin: 0 auto;
            background-color: #ffffff;
            padding: 20px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            border-radius: 8px;
        }}
        .header {{
            background-color: #007bff;
            color: #ffffff;
            padding: 20px;
            text-align: center;
            border-radius: 8px 8px 0 0;
        }}
        .content {{
            margin: 20px 0;
            line-height: 1.6;
        }}
        .footer {{
            text-align: center;
            color: #888888;
            font-size: 12px;
            margin-top: 20px;
        }}
        .section {{
            margin-bottom: 20px;
        }}
        .section h2 {{
            font-size: 18px;
            color: #333333;
            margin-bottom: 10px;
        }}
        .section p {{
            font-size: 16px;
            color: #333333;
        }}
        .highlight {{
            background-color: #f8f9fa;
            border-left: 4px solid #007bff;
            padding: 10px;
            margin-top: 10px;
        }}
    </style>
```

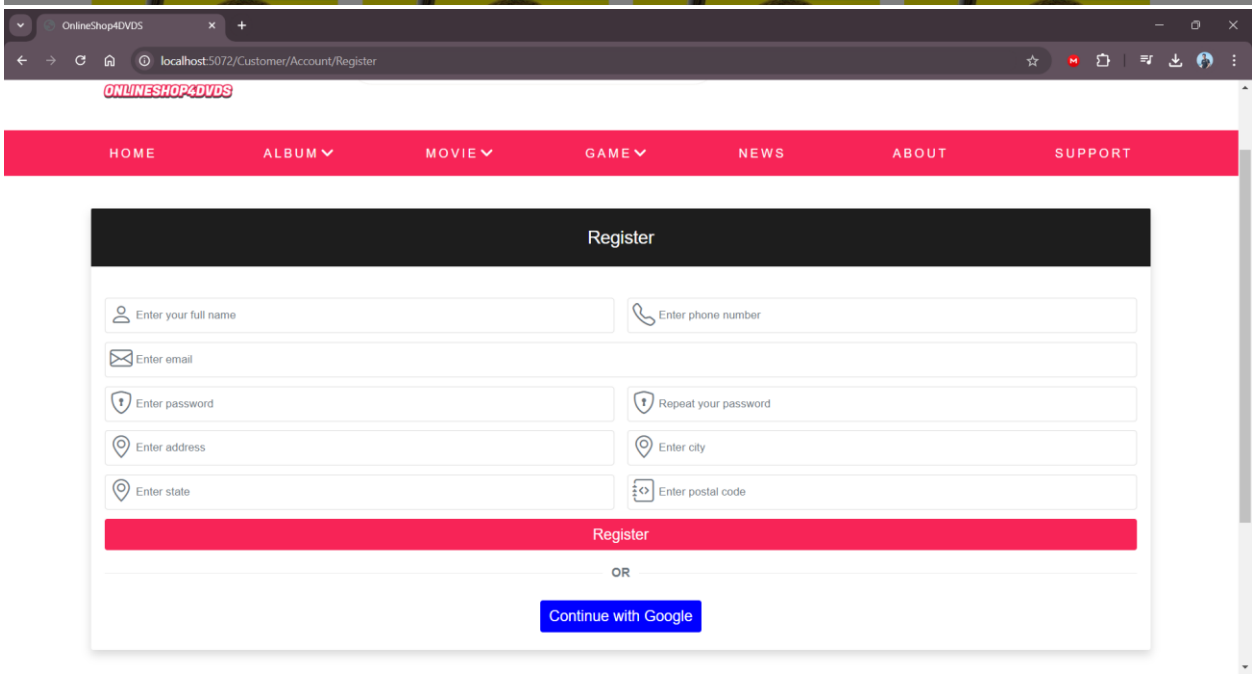
```
</head>
<body>
    <div class='container'>
        <div class='header'>
            <h1>Reply to Your Feedback</h1>
        </div>
        <div class='content'>
            <div class='section'>
                <p>Hi {contact.Name},</p>
                <p>Thank you for your feedback. We appreciate the time you took to share your thoughts with us. Below, you'll find our response to your feedback:</p>
            </div>
            <div class='section'>
                <h2>Your Feedback:</h2>
                <p class='highlight'>{contact.Feedback}</p>
            </div>
            <div class='section'>
                <h2>Our Reply:</h2>
                <p class='highlight'>{contact.Reply}</p>
            </div>
        </div>
        <div class='footer'>
            <p>&copy; {DateTime.Now.Year} OnlineShop4DVDS. All rights reserved.</p>
        </div>
    </div>
</body>
</html>";
                    await _emailSender.SendEmailAsync(contact.Email, "Reply to Your Feedback", emailBody);

                    // Commit the transaction
                    transaction.Complete();
                    TempData["swalsuccess"] = "Reply email have been send to customer";
                    return RedirectToAction("Index");
                }
            }
            catch (Exception ex)
            {
                ModelState.AddModelError("", "An error occurred during this operation");
            }
        }
        return View(contact);
    }

    0 references
    public async Task<IActionResult> Detail(int? id)
    {
        var contact = await _unitOfWork.ContactRepository.GetFirstOrDefault(c => c.Id == id);

        if (contact == null)
        {
            return NotFound();
        }
        return View(contact);
    }
```

2. Dashboard GUI



a. Login & Register





Code:

```
namespace Identity.Areas.Customer.Controllers
{
    [Area("Customer")]
    1 reference
    public class AccountController : Controller
    {
        private readonly UserManager<IdentityUser> _userManager;

        private readonly SignInManager<IdentityUser> _signInManager;

        private readonly IEmailSender _emailSender;

        0 references
        public AccountController(UserManager<IdentityUser> userManager, SignInManager<IdentityUser> signInManager, IEmailSender emailSender)
        {
            _userManager = userManager;
            _signInManager = signInManager;
            _emailSender = emailSender;
        }

        0 references
        public IActionResult Register()
        {
            RegisterViewModel registerViewModel = new();
            return View(registerViewModel);
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        0 references
        public async Task<IActionResult> Register(RegisterViewModel registerViewModel)
        {
            if (ModelState.IsValid)
            {
                var user = new ApplicationUser
                {
                    UserName = registerViewModel.Email,
                    Email = registerViewModel.Email,
                    Name = registerViewModel.Name,
                    PhoneNumber = registerViewModel.PhoneNumber,
                    StreetAddress = registerViewModel.StreetAddress,
                    City = registerViewModel.City,
                    State = registerViewModel.State,
                    PostalCode = registerViewModel.PostalCode
                };

                var result = await _userManager.CreateAsync(user, registerViewModel.Password);

                if (result.Succeeded)
                {
                    await _userManager.AddToRoleAsync(user, StaticDetail.Role_Customer);
                    var tokenConfirm = await _userManager.GenerateEmailConfirmationTokenAsync(user);
                    var callBackUrl = Url.Action("ConfirmEmail", "Account", new { userId = user.Id, code = tokenConfirm }, protocol: HttpContext.Request.Scheme);
                    string emailBody = $@"
```

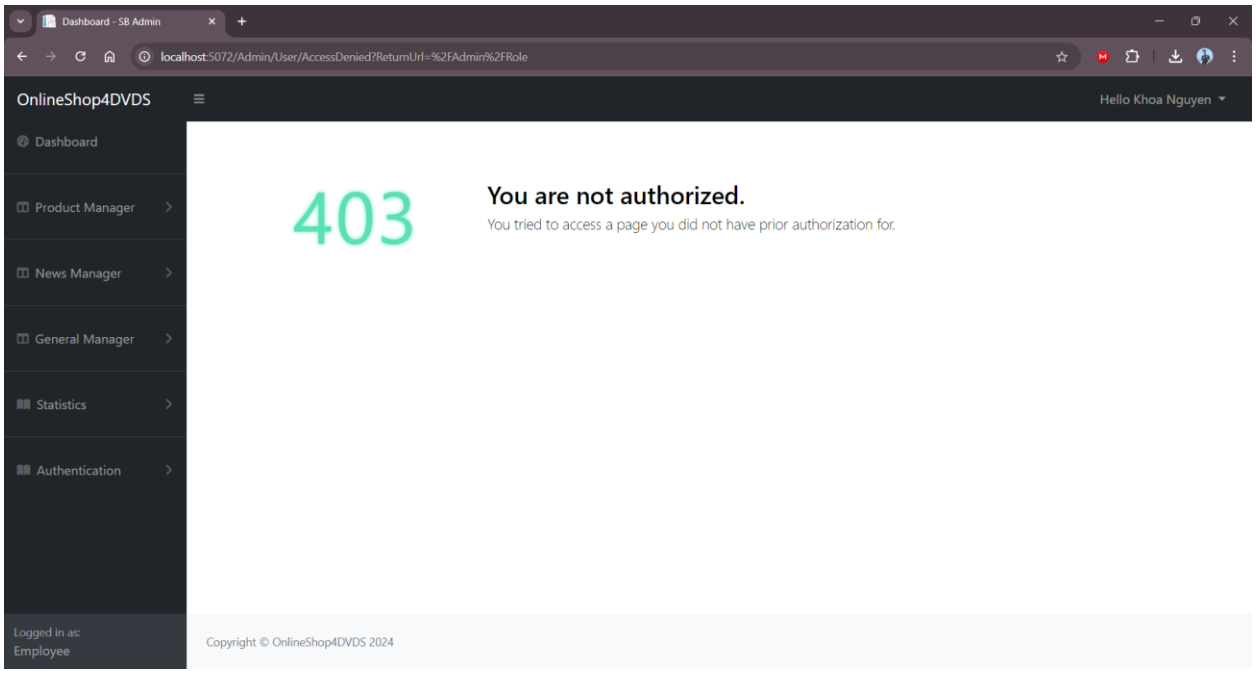
```
<!DOCTYPE html>
<html>
<head>
    <meta charset='UTF-8'>
    <title>Reset Your Password</title>
    <style>
        body {{
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
        }}
        .container {{
            width: 100%;
            max-width: 600px;
            margin: 0 auto;
            background-color: #ffffff;
            padding: 20px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        }}
        .header {{
            background-color: #007bff;
            color: #ffffff;
            padding: 10px 20px;
            text-align: center;
        }}
        .content {{
            margin: 20px 0;
            line-height: 1.6;
        }}
        .content a {{
            color: #007bff;
            text-decoration: none;
        }}
        .footer {{
            text-align: center;
            color: #888888;
            font-size: 12px;
            margin-top: 20px;
        }}
    </style>
</head>
<body>
    <div class='container'>
        <div class='header'>
            <h1>Password Reset Request</h1>
        </div>
        <div class='content'>
            <p>Hi {user.UserName},</p>
            <p>We received a request to reset your password. Click the link below to reset your password:</p>
            <p><a href='{callBackUrl}' target='_blank'>Reset your password</a></p>
            <p>If you did not request a password reset, please ignore this email or contact support if you have questions.</p>
            <p>Thank you,<br/>OnlineShop4DVDS</p>
        </div>
        <div class='footer'>
            <p>&copy; {DateTime.Now.Year} OnlineShop4DVDS. All rights reserved.</p>
        </div>
    </div>
</body>
</html>";

        await _emailSender.SendEmailAsync(forgotPasswordViewModel.Email, "Reset Password", emailBody);
        TempData["swalsuccess"] = "Reset email has been sent";
        return LocalRedirect(returnUrl);
    }
    return RedirectToAction("Index", "Home", new { area = "Customer" });
```

```
            await _emailSender.SendEmailAsync(forgotPasswordViewModel.Email, "Reset Password", emailBody);
            TempData["swalsuccess"] = "Reset email has been sent";
            return LocalRedirect(returnUrl);
        }
        return RedirectToAction("Index", "Home", new { area = "Customer" });
    }

    0 references
    public async Task<IActionResult> ResetPassword(string? code)
    {
        if(code == null)
        {
            return View("Error");
        }

        if (User.Identity.IsAuthenticated)
        {
            await _signInManager.SignOutAsync();
        }

        return RedirectToAction("ResetPasswordReload", new { code });
    }

    0 references
    public async Task<IActionResult> ResetPasswordReload(string code)
    {
        return View("ResetPassword");
    }

    [HttpPost]
    [ValidateAntiForgeryToken]
    0 references
    public async Task<IActionResult> ResetPassword(ResetPasswordViewModel resetPasswordViewModel)
    {
        if (ModelState.IsValid)
        {
            var user = await _userManager.FindByEmailAsync(resetPasswordViewModel.Email);
            if (user == null)
            {
                return RedirectToAction("ResetPasswordConfirmation");
            }

            var result = await _userManager.ResetPasswordAsync(user, resetPasswordViewModel.Code, resetPasswordViewModel.Password);

            if (result.Succeeded)
            {
                return RedirectToAction("ResetPasswordConfirmation");
            }
            AddErrors(result);

        }
        return View();
    }

    0 references
    public async Task<IActionResult> ResetPasswordConfirmation()
    {
        await _signInManager.SignOutAsync();
        return View();
    }
```

```
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public IActionResult ExternalLogin(string provider)
{
    // Request to Google
    var redirectUrl = Url.Action("ExternalLoginCallBack", "Account");
    var properties = _signInManager.ConfigureExternalAuthenticationProperties(provider, redirectUrl);
    return Challenge(properties, provider);
}

0 references
public async Task<IActionResult> ExternalLoginCallBack(string? remoteError)
{

    if (remoteError != null)
    {
        return RedirectToAction("Index", "Home", new { area = "Customer" });
    }

    var info = await _signInManager.GetExternalLoginInfoAsync();
    if (info == null)
    {
        return RedirectToAction("Login");
    }

    // Sign in user with Google login provider
    var result = await _signInManager.ExternalLoginSignInAsync(info.LoginProvider, info.ProviderKey, isPersistent: false);
    if (result.Succeeded) // User already has login by using Google before
    {
        // Update authentication tokens
        await _signInManager.UpdateExternalAuthenticationTokensAsync(info);
        return RedirectToAction("Index", "Home", new { area = "Customer" });
    }
    else // User has login by using Google for the first time
    {
        ViewData["ProviderDisplayName"] = info.ProviderDisplayName;
        var email = info.Principal.FindFirstValue(ClaimTypes.Email);
        var name = info.Principal.FindFirstValue(ClaimTypes.Name);
        return View("ExternalLoginConfirmation", new ExternalLoginConfirmationViewModel { Email = email, Name = name });
    }
}
```

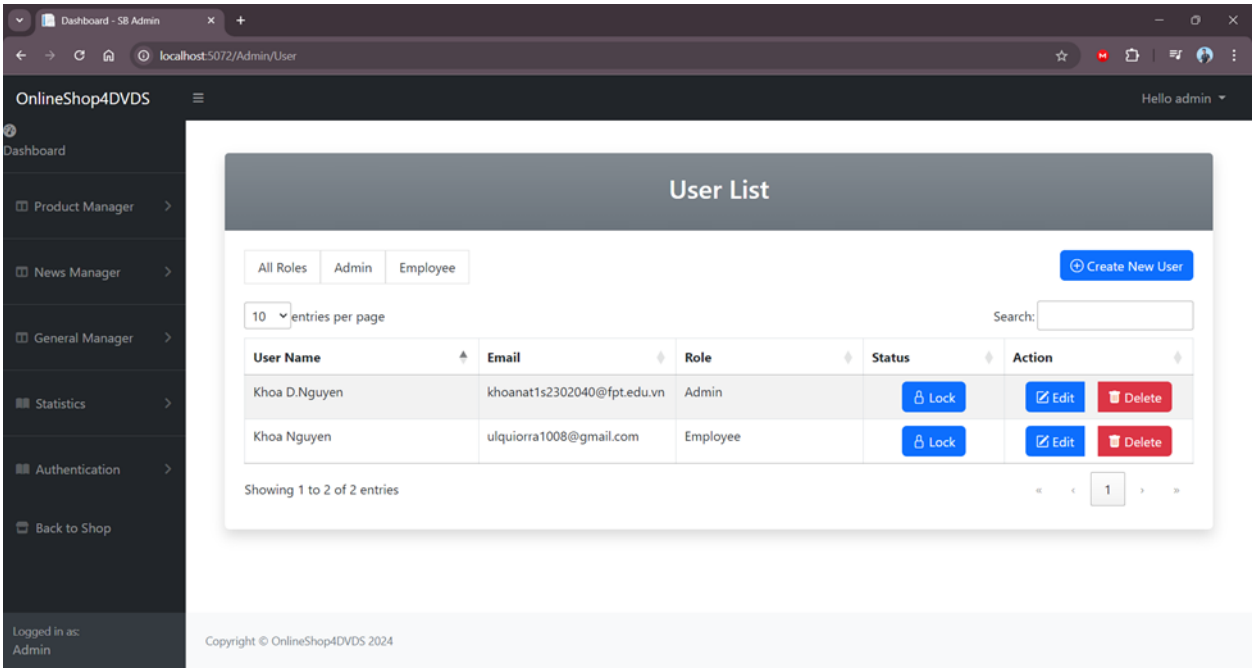
```
        [HttpPost]
        [ValidateAntiForgeryToken]
        0 references
        public async Task<IActionResult> ExternalLoginConfirmation(ExternalLoginConfirmationViewModel externalLoginConfirmationViewModel)
        {
            if (ModelState.IsValid)
            {
                // Get the info about the user from external login provider
                var info = await _signInManager.GetExternalLoginInfoAsync();
                if (info == null)
                {
                    return View("Error");
                }
                var user = new ApplicationUser
                {
                    UserName = externalLoginConfirmationViewModel.Email,
                    Email = externalLoginConfirmationViewModel.Email,
                    Name = externalLoginConfirmationViewModel.Name
                };
                var result = await _userManager.CreateAsync(user);
                if (result.Succeeded)
                {
                    await _userManager.AddToRoleAsync(user, StaticDetail.Role_Customer); // Tạo role customer cho user
                    var tokenConfirm = await _userManager.GenerateEmailConfirmationTokenAsync(user); // Tạo token confirm email cho user
                    result = await _userManager.ConfirmEmailAsync(user, tokenConfirm);
                    if (result.Succeeded)
                    {
                        result = await _userManager.AddLoginAsync(user, info);
                        if (result.Succeeded)
                        {
                            await _signInManager.SignInAsync(user, isPersistent: false);
                            // Update authentication tokens
                            await _signInManager.UpdateExternalAuthenticationTokensAsync(info);
                            return RedirectToAction("Index", "Home", new { area = "Customer" });
                        }
                        AddErrors(result);
                    }
                    AddErrors(result);
                }
                AddErrors(result);
            }

            return View(externalLoginConfirmationViewModel);
        }

        //Add error of creating user to View Create
        5 references
        private void AddErrors(IdentityResult result)
        {
            foreach (var error in result.Errors)
            {
                ModelState.AddModelError(string.Empty, error.Description);
            }
        }

    }
}
```

b. *Authorized*

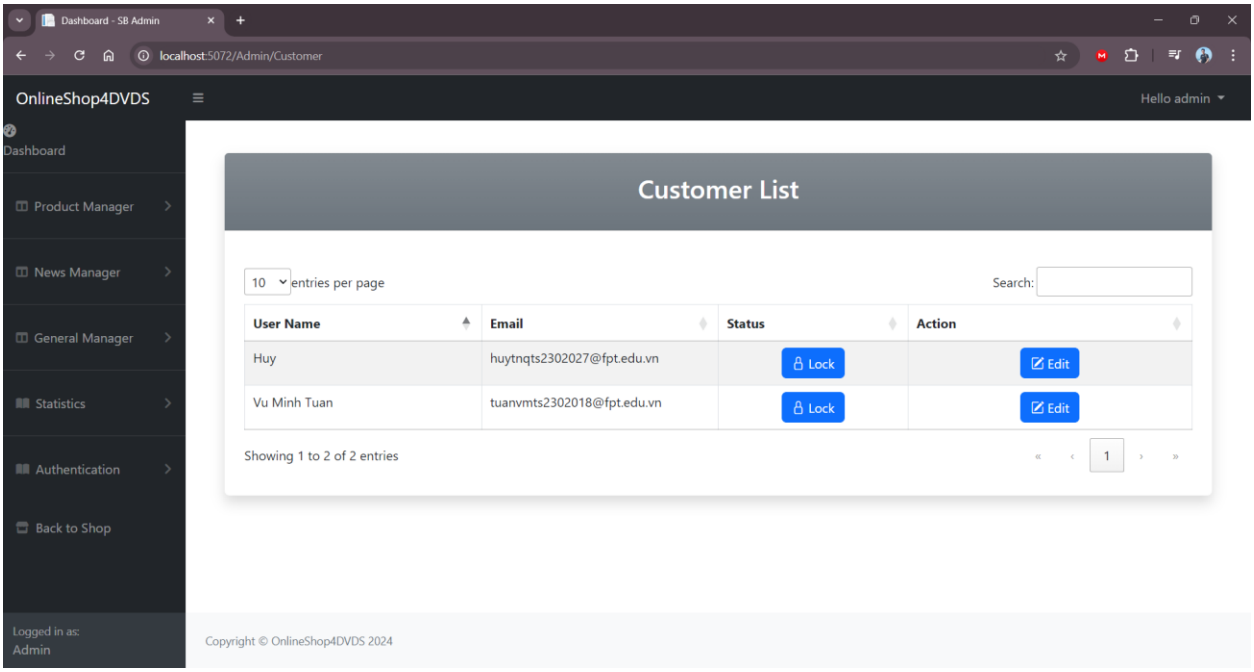


III. Dashboard

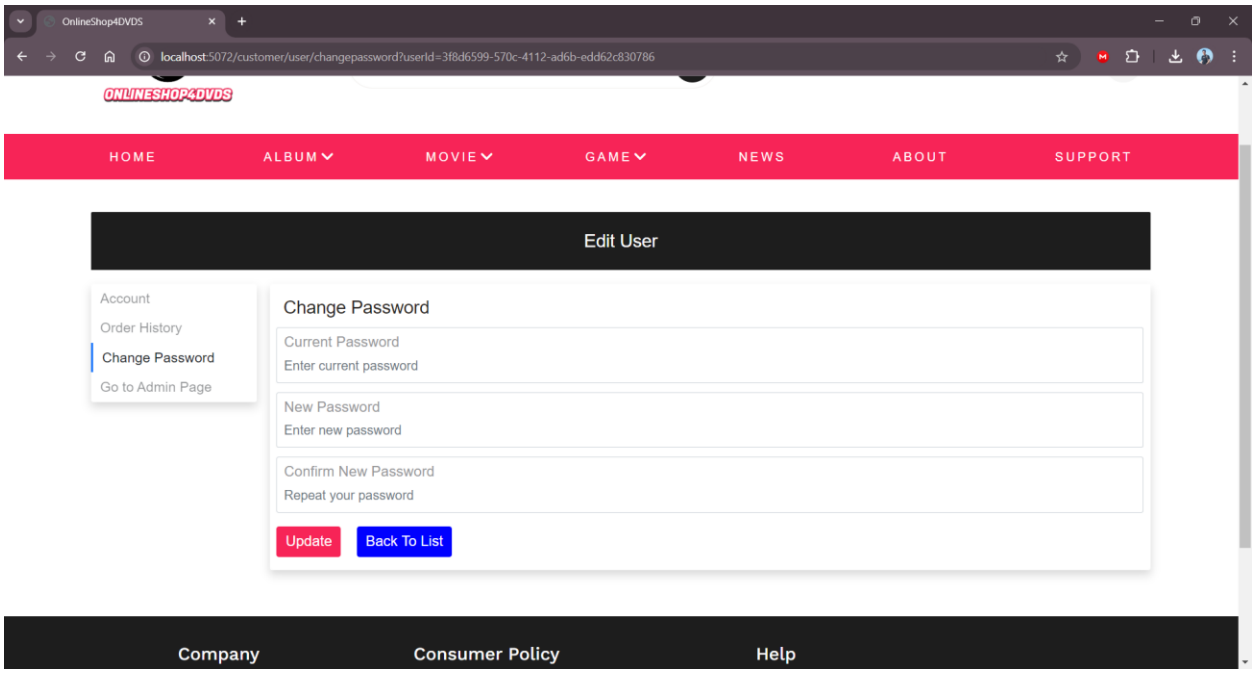
1. User management



2. Customer management



Customer can edit their password in the Account manager



Code:

```
[Authorize(Policy = "AdminOrOnlySelfPermission")]
0 references
public async Task<IActionResult> ChangePassword(string userId)
{
    var userUpdated = await _dbContext.ApplicationUsers.FirstOrDefaultAsync(u => u.Id == userId);
    if (userUpdated == null)
    {
        return NotFound();
    }

    ChangePasswordViewModel changePasswordViewModel = new()
    {
        UserId = userId
    };

    return View(changePasswordViewModel);
}
```

```
[HttpPost]
0 references
public async Task<IActionResult> ChangePassword(ChangePasswordViewModel changePasswordViewModel)
{
    if (ModelState.IsValid)
    {
        var userUpdated = await _userManager.FindByIdAsync(changePasswordViewModel.UserId);
        if (userUpdated == null)
        {
            return NotFound();
        }

        // Check if new password is the same as the current password
        var passwordCheck = _userManager.PasswordHasher.VerifyHashedPassword(userUpdated, userUpdated.PasswordHash, changePasswordViewModel.Password);
        if (passwordCheck == PasswordVerificationResult.Success)
        {
            ModelState.AddModelError(string.Empty, "New password must be different from the current password");
            ChangePasswordViewModel samePasswordError = new()
            {
                UserId = userUpdated.Id
            };
            return View(samePasswordError);
        }

        var result = await _userManager.ChangePasswordAsync(userUpdated, changePasswordViewModel.CurrentPassword, changePasswordViewModel.Password);

        if (result.Succeeded)
        {
            TempData["swalsuccess"] = "Account update successful. Your password has been changed";

            // Logout if the deleted user is the current user
            if (userUpdated.Id == _userManager.GetUserId(User))
            {
                await _signInManager.SignOutAsync();
                return RedirectToAction("Index", "Home", new { area = "Customer"});
            }
            else
            {
                var user = await _userManager.FindByIdAsync(userUpdated.Id);
                var roleCheck = await _userManager.GetRolesAsync(user);

                if (!roleCheck.Contains("Customer"))
                {
                    TempData["success"] = "Change password successful";
                    return RedirectToAction("Index", "User", new { area = "Admin" });
                }

                return RedirectToAction("ChangePassword", "User", new { area = "Customer", userId = userUpdated.Id });
            }
        }
        AddErrors(result);
        ChangePasswordViewModel changePasswordError = new()
        {
            UserId = userUpdated.Id,
        };
        return View(changePasswordError);

    }
    ChangePasswordViewModel modelStateInvalid = new()
    {
        UserId = changePasswordViewModel.UserId,
    };
    return View(modelStateInvalid);
}
```

```
        [Authorize(Policy = "AdminOrOnlySelfPermission")]
        0 references
        public IActionResult ChangePasswordConfirmation()
        {
            return View();
        }

        0 references
        public IActionResult AccessDenied()
        {
            return View();
        }

        //Add error of creating user to View Create
        2 references
        private void AddErrors(IdentityResult result)
        {
            foreach (var error in result.Errors)
            {
                ModelState.AddModelError(string.Empty, error.Description);
            }
        }

        #region API CALLS
    }
}
```

TASK SHEET REVIEW 3

Project Ref. No.: eP/Advertisement Portal Management System/01		Activity Plan Prepared By:	Date of Preparation of Activity Plan:			
Sr. No.	**Task**		**Actual Start Date**	**Actual Days**	**Team Mate Names**	**Status**
1	User – login/logout	All Members	07/12/2024	2	Tuan	Completed
2	Admin - authorized		07/12/2024	2		Completed
3	CRUD User		07/12/2024	1		Completed
4	Layout		07/12/2024	2		Completed
5	CRUD Feedback		07/12/2024	1		Completed
6	CRUD Cart		07/12/2024	2	Hung	Completed
7	Payment		07/12/2024	4		Completed
8	CRUD Promotion		07/12/2024	2	Khoa	Completed
9	CRUD News		07/12/2024	1		Completed
10	CRUD Album		07/12/2024	2		Completed
11	Data Prep		07/12/2024	1		Completed
12	CRUD Game/Music		07/12/2024	3	Nhung	Completed
13	Stats/Chart		07/12/2024	1	Huy	Completed
14	CRUD Review		07/12/2024	1		Completed
15	CRUD Order		07/12/2024	1		Completed

Date: 08/09/2024

Signature of Instructor:	**Signature of Team Leader:**
MR. LE TUAN XUYEN	**VU MINH TUAN**