**NAME**: PARTHA SARATHI BISWAS (2318513)

**PROJECT TITLE: MOVIE STORE**

**Abstract**:

The movie store is a web application designed to enhance the experience of watching and buying movies built using ASP.NET MVC, C# and SQL. Customers can register, providing essential details such as name, gender, age, and contact information. They can then select the movies they want to purchase. The system maintains enables them to successfully check out after purchasing the movies.

**Key features:**

* **Customer Registration:** Customer can register by providing essential details such as name, gender, age and contact information, with the help of MVC we can also update and delete this information**.**
* **Search Movies: Customers can search movies they want to purchase.**
* **Authentication**: With the help of login page only registered users can login into the system.
* **Database Integration:** The system uses SQL databases to store customer data, movie details, and other relevant information.
* **Admin Controls**: Admins manage CRUD operations on movie list. They can add new movies, update movie information, and handle system configurations.

**PROJECT GOALS:**

1. Streamline the process of searching and purchasing movies under the same hood. Ensure customers can easily view the available movies and purchase them.
2. To develop a responsive web application using ASP.NET MVC. Also providing a simple yet effective interface for customers and admins.
3. To allow admins to manage movies list, pricing and customers. Improve overall movie store efficiency by automating tasks and reducing the need to go out to theatres.

**Technical Stack:**

• **Frontend**: HTML, CSS, Bootstrap and JavaScript

• **Backend**: C#, .NET Framework

• **Database**: SQL Server

**DB SCHEMA:**

namespace signup\_with\_login.Models

{

using System;

using System.Collections.Generic;

using System.Web;

public partial class movie

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public movie()

{

this.orderts = new HashSet<ordert>();

}

public int Id { get; set; }

public string name { get; set; }

public int realease\_year { get; set; }

public string image\_path { get; set; }

public Nullable<int> price { get; set; }

public HttpPostedFileBase ImageFile { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<ordert> orderts { get; set; }

}

}

namespace signup\_with\_login.Models

{

using System;

using System.Collections.Generic;

public partial class ordert

{

public int Id { get; set; }

public Nullable<int> movie\_id { get; set; }

public Nullable<int> cust\_id { get; set; }

public Nullable<int> price { get; set; }

public Nullable<int> total\_price { get; set; }

public Nullable<int> quantity { get; set; }

public Nullable<System.DateTime> order\_date { get; set; }

public virtual movie movie { get; set; }

public virtual user user { get; set; }

}

}

namespace signup\_with\_login.Models

{

using System;

using System.Collections.Generic;

public partial class role

{

public int Id { get; set; }

public int UserId { get; set; }

public string Role1 { get; set; }

public virtual user user { get; set; }

}

}

namespace signup\_with\_login.Models

{

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

public partial class user

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public user()

{

this.roles = new HashSet<role>();

this.orderts = new HashSet<ordert>();

}

public int Id { get; set; }

public string FirstName { get; set; }

public string LastName { get; set; }

public string Gender { get; set; }

[Required(ErrorMessage = "Age is required")]

[Range(typeof(int), "18", "80", ErrorMessage = "Age can only be between 18 and 80")]

public int age { get; set; }

[DataType(DataType.EmailAddress)]

[Required(ErrorMessage = "Please enter Email ID")]

[RegularExpression(@"^\w+([-+.']\w+)\*@\w+([-.]\w+)\*\.\w+([-.]\w+)\*$", ErrorMessage = "Email is not valid.")]

public string email { get; set; }

public string username { get; set; }

[DataType(DataType.Password)]

public string password { get; set; }

[Required(ErrorMessage = "Confirm Password is required")]

[DataType(DataType.Password)]

[Compare("password")]

public string confirm\_password { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<role> roles { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<ordert> orderts { get; set; }

}

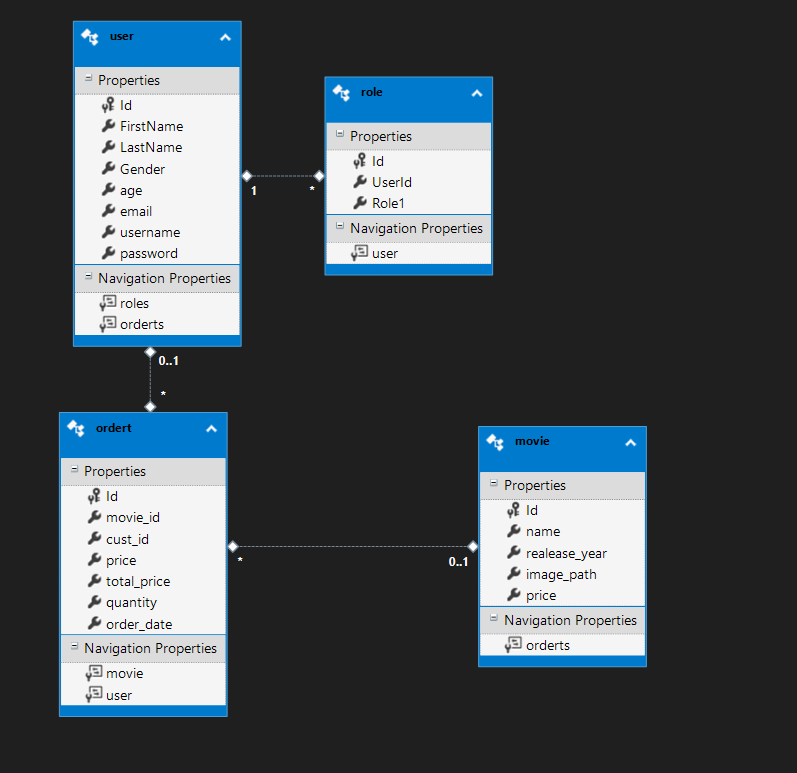
}

Here is the database diagram of Movie store Web Application:

A screenshot of a computer

Description automatically generated

**EDMX DIAGRAM FOR MOVIE STORE MANAGEMENT SYSTEM:**

****

**High-Level Design Document: Movie Store management System**

**1.Introduction:**

The Movie store is a web application designed to provide a user-friendly platform for managing movies, customers and their purchases. The main aim of the system is to streamline the movie operations by using efficient management techniques.

**2. Architecture Overview:**

The system follows a multi-tier architecture, consisting of the following layers:

• **Presentation Layer**: This layer includes the user interface components built using ASP.NET MVC, HTML and CSS. It provides the frontend for user interaction.

• **Application Layer**: The application layer contains the business logic implemented in C# using the .NET Framework. It handles user requests, processes data, and interacts with the data access layer.

• **Data Access Layer**: This layer is responsible for interacting with the database (SQL Server).

**3.Key Components :**

* **Customer Details:** Customer can register by providing essential details such as name, age, and contact information(Contact number and Email), with the help of MVC we can also update and delete this information**.**
* **Search Functionality :** Customer can search their desired movies.
* **Movie management**: Movies list can be easily managed.
* **Security**: With the help of login page only registered users can login into the system.

1. **Database Design**: The database design includes the following tables:

* **Users:** This table is responsible to store all the login credentials of both the customer and the admin. So upon entering appropriate credentials of either admin or customer the respective user will be able to log in the application and perform their necessary actions. The authentication process also takes place while accessing this table.
* **Movie:**  This table holds all the details related to the movie like their name, release year and price.
* **Order:**  This table holds the data of all total orders of the customer.

1. **Technologies Used :**

• **Frontend**: ASP.NET MVC(to be used in future dev.), HTML, CSS, JavaScript( very less)

• **Backend**: C#, .NET Framework

• **Database**: SQL Server

**6.Deployment:**

The system can be deployed on a web server that supports ASP.NET applications.

* Enable customers to watch and rent movie at the comfort of home

**Future Enhancements**

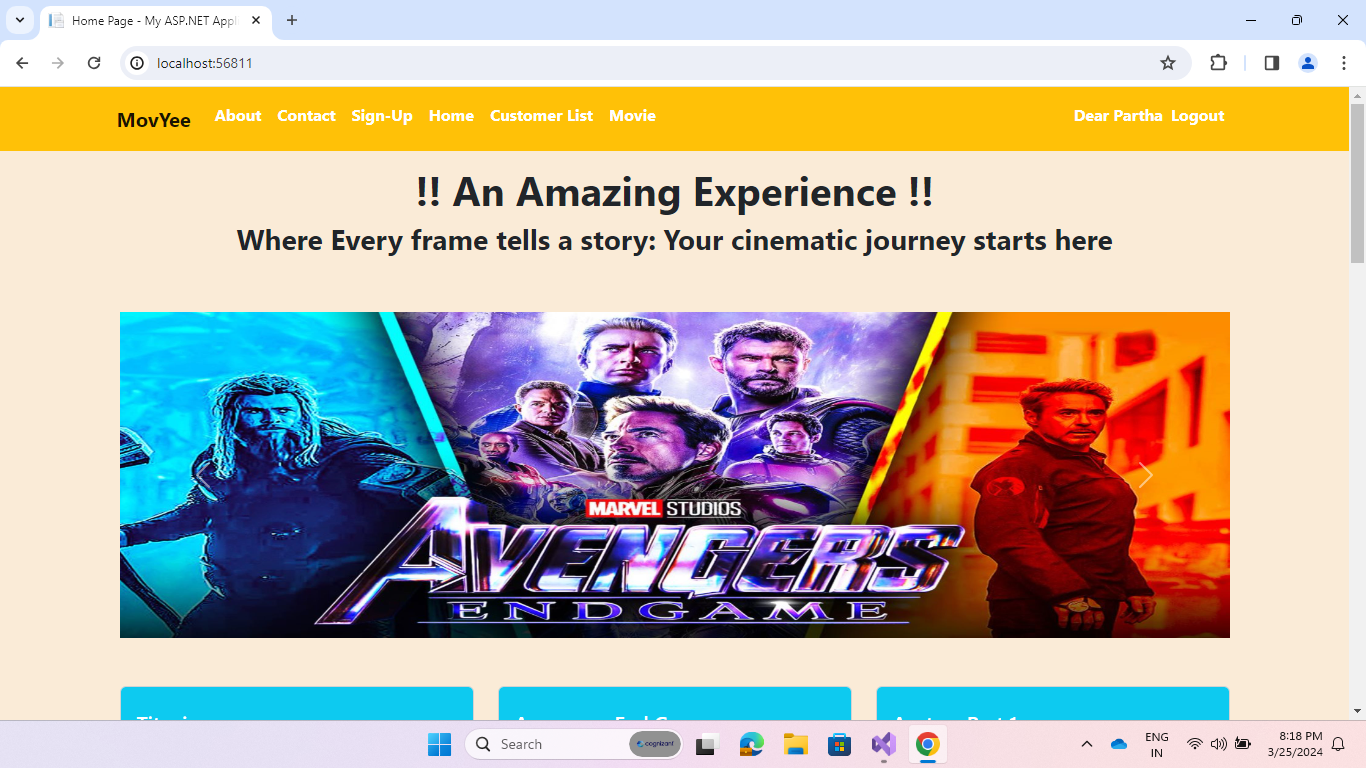
1. In future we can add a payment gateway using Cards in this management system.
2. The system can be deployed on a web server that supports ASP.NET applications. It can be hosted on-premises or on a cloud platform such as Microsoft Azure.

A screenshot of a computer

Description automatically generated**Login Page:**

**A screenshot of a computer

Description automatically generatedSignUp Page:**

**Front Page:**

**A screenshot of a computer

Description automatically generatedSome pages where Admin Can Perform CRUD Operations:**

**A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated**

**THANK YOU**