**PROJECT PROPOSAL**

**ON**

**ONLINE MOVIE RENTAL SYSTEM**

**RANJIT BASU**

**00179637**

**COMPUTING PROJECT**

**LEVEL 5 DIPLOMA IN COMPUTING**

**SOFTWARICA COLLEGE IF IT AND E-COMMERCE**

**KATHMANDU, NEPAL**

**04/11/2019**

**C# DOTNET MVC FRAMEWORK**

**Acknowledgement**

I am thankful for my leader Mr. Niman Maharjan for the support and guidance in completing my project.

I am also thankful for all my teachers and friends who helped me in the completion of my project. This is my website project that was assigned as a project.

**Abstract**

The purpose of the project is to search and rent movies. Before the development of website customer and have to visit and borrow movies. Now they can directly rent and get delivered to their homes as well.

The projects main aim is to develop online movie rental system named as 4321movies.com. Through this website user can see wide range of movies. Its main aim is to allow customer to search wide range of movies and rent them.

The system has two users one is admin who is in control of the website and other is customer who can have facilities in using this website.

I have chosen waterfall methodologies in this project and used C# dot net mvc framework pattern for the design.

Contents

[CHAPTER 1: INTRODUCTION 5](#_Toc15289927)

[**1.1 Project Introduction** 5](#_Toc15289928)

[**1.2 Justification for the Project** 5](#_Toc15289929)

[**1.2.1 Background of the Project** 5](#_Toc15289930)

[**1.2.2 Problem Statement** 5](#_Toc15289931)

[**1.3 Description of the Project** 6](#_Toc15289932)

[**1.3.1 Features** 6](#_Toc15289933)

[**1.4 Overview of the Project** 6](#_Toc15289934)

[**1.5 Scope** 6](#_Toc15289935)

[**1.5.1 Limitations** 6](#_Toc15289936)

[**1.5.2 Aims** 7](#_Toc15289937)

[**1.5.3 Objectives** 7](#_Toc15289938)

[**1.5.4 Overview of the Scope** 7](#_Toc15289939)

[CHAPTER 2: ANALYSIS 7](#_Toc15289940)

[**2.1 Introduction to analysis** 7](#_Toc15289941)

[**2.2 Feasibility study** 7](#_Toc15289942)

[**2.3Requirement analysis** 8](#_Toc15289943)

[**2.4 DEVELOPPMENT METHODOLOGIES** 15](#_Toc15289944)

[CHAPTER 3: DESIGN 17](#_Toc15289945)

[**3.1 Structural design** 17](#_Toc15289946)

[**3.2 Behavioral design** 19](#_Toc15289947)

[**3.3 Database design** 22](#_Toc15289948)

[**Architecture** 28](#_Toc15289949)

[CHAPTER 4: IMPLEMENTATION 29](#_Toc15289950)

[**4.1 programming language** 29](#_Toc15289951)

[**4.2 Development Environment** 30](#_Toc15289952)

[CHAPTER 5: TESTING 30](#_Toc15289953)

[**Black box testing:** 30](#_Toc15289954)

[**Unit testing:** 30](#_Toc15289955)

[CHAPTER 6: OTHER PROJECT ISSUES 42](#_Toc15289956)

[**6.1 Scheduling: Gantt chart** 42](#_Toc15289957)

[**6.2 RISK MANAGEMENT** 44](#_Toc15289958)

[CHAPTER 7: CONFIGURATION MANAGEMENT 46](#_Toc15289959)

[CHAPTER 8: CONCLUSION OF THE PROJECT 49](#_Toc15289960)

[REFERENCE & BIBLIOGRAPHIES 50](#_Toc15289961)

[CHAPTER 9: APPENDIX 50](#_Toc15289962)

# CHAPTER 1: INTRODUCTION

## **1.1 Project Introduction**

Nowadays, people are very modern, they surf internet most of the time. Basically, they use internet to use various sites and watching movies is one of them. Sometimes people cannot find exact type of movies that they want. They make the use of internet for searching and buying products. The proposed project “Online Movies Rental System” where customer can search and buy their desired movies in real time. This application lets to interact between customer and employee. It has wide range of movies to search with. Customer can make and have their own account and rent any movie anytime anywhere.

## **1.2 Justification for the Project**

## **1.2.1 Background of the Project**

4321movie.com is Kathmandu’s top online rental movie service company which rents DVD, CD's, of the different Hollywood, Bollywood and Kollywood movies. It specializes in a complete range of rental service of the movies of any category. You can also view special offers movies.

Specific rates of movies and timing is also shown in the browser. You need to register before making a purchase. It is open 24 hours a day. It is a very user-friendly website where users can interact with the system easily. I have c# dot net mvc framework to design this website.

## **1.2.2 Problem Statement**

The company provides rental service to the customer while also maintaining their records. Staffs encounter many problems while maintaining reports. It is very time consuming. Staffs have difficulty in preparing reports when manager asks. The service has been manually managed with traditional filing system. But the organization wants to automate the information and record of every transaction activities and would like a reporting system in the form of a user-friendly web application.

## **1.3 Description of the Project**

## **1.3.1 Features**

* User can register and login to the system

User needs to enter email, password, and other required field for registration. If password and confirm password matches registration will be successful else it will ask user to try again. To login user must enter email and password correctly else it will ask user to try again.

* User can rent a movie

User needs to enter their valid name and movie and enter submit button. If records successfully submitted success message will display else error message will display.

* User can search movies

User can search their desired movies through release date, genre or in search.

* Admin can add movies

Admin can add, update and make changes to the list of movies.

## **1.4 Overview of the Project**

Overall this system lets people to rent various movies. For that they need to provide a little information through registration. After successful registration they can login. They can search various type of movies through release date, search bar and genre.

## **1.5 Scope**

It maintains the rental system in a modern organized way.

* It lets user to search wide range of movies.
* It is for people who wants to rent a movie without too much search effort.
* It lets user to rent movie in a modern way.

## **1.5.1 Limitations**

* This web hasn’t made available in foreign countries.
* Customer can buy one movie at a time.

## **1.5.2 Aims**

* It needs to allow user to rent a movie.
* It needs to provide customer to search wide range of movies.
* It needs to be very simple and customer satisfying.
* It needs to be user friendly and search methods needs to be simple.
* It needs user to register and login.

## **1.5.3 Objectives**

Its main objective is to overcome traditional filling system.

* It needs to keep record of what user brought.
* User can know the details of movies and its price.
* user can send feedback through contact.

## **1.5.4 Overview of the Scope**

Overall the main aim of this site is be as user friendly as possible. Before the development of site, customer have to visit and rent movie. Now they can directly rent online.

# CHAPTER 2: ANALYSIS

## **2.1 Introduction to analysis**

Analysis is the first stage of software development process. During analysis, requirement analysis like functional, non-functional and Moscow prioritization are explored. For further ease of analysis diagrams like use case and Class diagram are designed.

It is the process of production of software. This is the part when programmers understand what the customer wants. It allows teams to transform the needs specified in earlier proposed proposal. Requirements and diagrams help team and user interact each properly where both parties understand how the system will be. Requirements are gathered through the help of processes like interviews, questionnaires, observations, focus-groups etc. it defines what user clearly wants.

## **2.2 Feasibility study**

It is an analysis of how successfully a project can be completed. It is an analysis of proposed project to determine if it is feasible economically, technically, operationally, schedulable and legally. An important purpose of feasibility study is that it helps planners focus on project. It also outlines alternatives. It addresses the probable issues that could affect the project.

**Types of feasibility study:**

* Economic feasibility

It determines the efficiency of a new project also known as cost/benefit analysis. It helps in finding profit from and against investment expected from a project. Its two most essential factors are cost and time.

* Technical feasibility

It defines the needs of current hardware and software for the proposed project. If any modification is necessary what kind of technology would be required.

* Operational feasibility

It refers to usability of system with the user. Will there be any training required for user to operate some tasks? It also tells if any materials are required for the training.

* Schedule feasibility

It relates the allocation of time to complete the development of the system. it also tells about the impact of the product is not delivered in time.

* Legal feasibility

It states about the legal requirement of the project. It is important that the project is following the requirements needed to start a project including licenses, copyrights and other safety measures.

## **2.3Requirement analysis**

**Functional Requirements:**

The most important phase in SDLC is requirement analysis phase because this the part when programmers understands what the customer wants. Functional requirements are those tasks which describes what a system should do and are designed for users and their convenience.

E.g. a system should have a login page where people can login in order to enter their desired sites.

Functional requirements are given below:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Fid** | **Functions** | **Data** | **Rational** | **Dependency** | **Remarks** |
| F01 | Registration | Personal details | User information | F01 | Creates user account |
| F02 | Login | Username password | security | F01 | Opens user’s dashboard |
| F03 | Reset password | email | Update password | F02 | Gets new password |
| F04 | Search movies | \_ | Searching movies | F02 | User can search movies |
| F05 | Rent movie | \_ | Renting movies | F02 | User can rent a movie |
| F06 | Admins customization | Movie details | Add, Update, Delete, Cancel | F06 | User can add, update, delete movies |
| F07 | Admin login | Admin details | Admin | F09 | Admin can update profile |

**Non-Functional**

Non-functional requirements are those tasks that describes how the system will do so on. Non-functional aspects involve performance, usability, responsiveness etc. they are designed which are not addressed by functional testing. Usually they are in the form of how system shall do or part of the system.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **NID** | **Functions** | **Data** | **Rational** | **Dependency** | **Remarks** |
| N01 | Responsive | - | Support different resolutions | N04 | Fits in different screen sizes |
| N02 | Usability | - | Easy and user friendly | N04 | User friendly |
| N03 | Maintainability | - | Easy to change | N04 | Easy maintenance and optimization |
| N04 | Reliable | - | Accurate output | N04 | Gives accurate output as per the input |
| N05 | Multi-Browser support | - | Supports different browsers | N04 | Runs in different browsers |
| N06 | Performance | - | Maintains system’s productivity | N04 | Regular tests are focused |

**Moscow Prioritization**

Moscow prioritization also known as Moscow method is a technique that manages both the functional and non-functional requirements. It has four prioritization categories must have, should have, could have and won’t have. It sets priorities between functionalities about how much importance they meet.

|  |  |  |
| --- | --- | --- |
| **ID** | **Functionalities** | **Priority** |
| F01 | Registration | Must have |
| F02 | Login | Must have |
| F03 | Reset password | Must have |
| F04 | Search movies | Must have |
| F05 | Rent movies | Must have |
| F06 | Admins customization | Must have |
| F07 | Admin login | Must have |
| N01 | Responsive | Must have |
| N02 | Usability | Must have |
| N03 | Maintainability | Must have |
| N04 | Reliable | Should have |
| N05 | Multi-browser support | Must have |
| N06 | Performance | Should have |

**Use case**

use case is a methodology used in system analysis to identify, clarify and organize system requirements. There is an actor and use cases in the diagram. Actor represents the roles played by users of the system.

* Admin can register to the system
* Admin can login to the system
* Admin can add movies
* Admin can add customers
* Admin can update customers info
* Admin can update rental info
* Admin can view movies
* Admin can logout of the system
* User can register, log in and logout
* User can view and rent movies

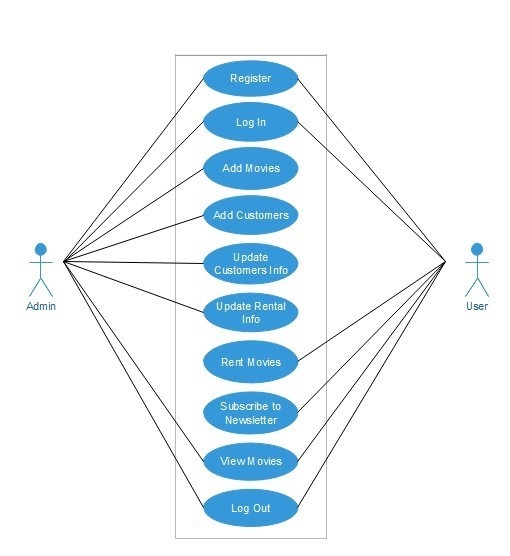
****

Fig: Use case diagram

**Initial class diagram (NLA)**

They are used to show the different objects in a system, their attributes, operations and relationships among them.

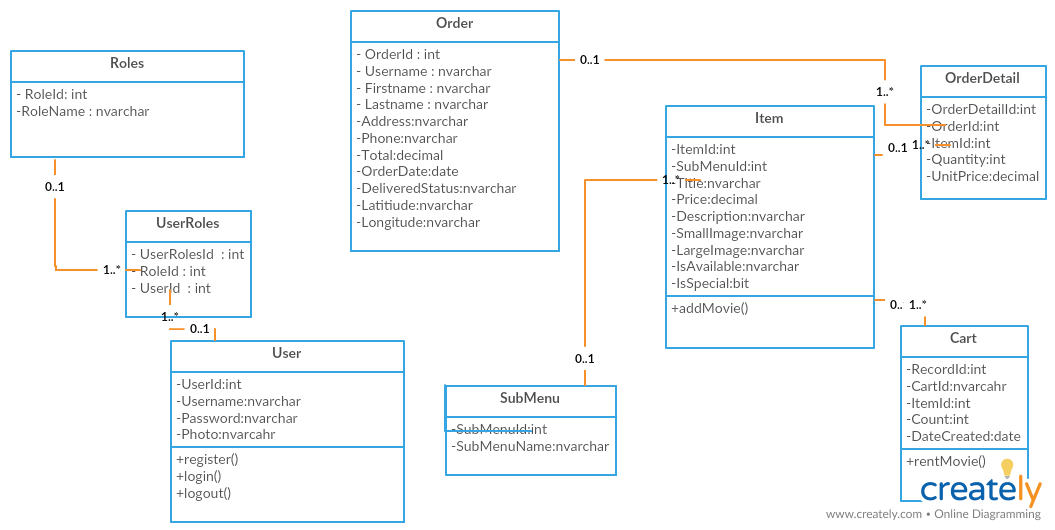


Fig: Initial class diagram

## **2.4 DEVELOPPMENT METHODOLOGIES**

**2.4.1 Waterfall Model**

I have used waterfall approach in this project. It’s a commonly used methodologies in any project. It is simple to understand and use. Each step is divided in this model where only after the completion of one step, another phase is performed. Steps are divided in six different parts i.e. requirement analysis, system design, implementation, testing, deployment and maintenance.

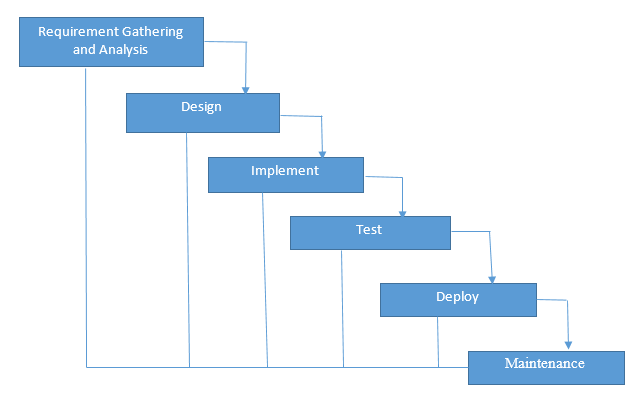


Fig: waterfall model.

**2.4.2 Design pattern**

I have used MVC (Model View Controller) design patter in this project. It is most used framework done in today’s projects. Both desktop and web-based application is performed by using this design pattern.

**2.4.3 Model View Controller Pattern**

Model interacts with data. The model doesn’t know about views and controller. When a model is changed, it notifies that a change has been occurred. View is what’s presented to the user and user interacts with the app. Controller finds the data query in model. Controller acts as an intermediary between model and view to process logic and requests. The controller updates the view when the model changes. It also adds event listeners to the view and updates the model when the user manipulates the view.

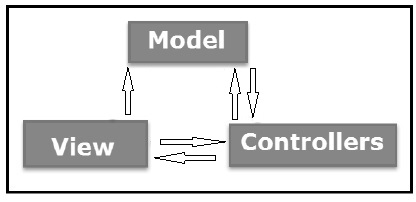


Fig: Model View Controller

**2.4.4 System Architecture**

System architecture is a conceptual model. It defines the structure, behavior and view of the system. It provides a presentation for the whole system. A system architecture consists of system components and the sub-systems developed, that will work together to implement the overall system.

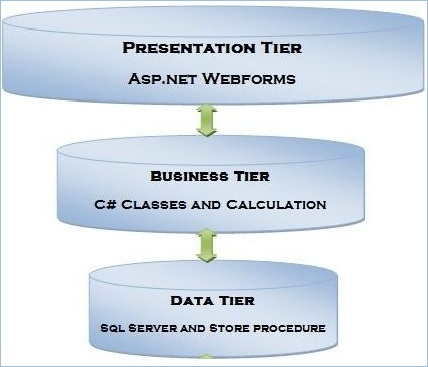


Fig: System Architecture

I choose 3-tier architecture because of following reasons:

It is more reliable.

It helps to maintain code easily.

It provides more opportunity while updating

# CHAPTER 3: DESIGN

Design pattern is a well-defined solution to a common problem. It’s a template, not a solution. That’s why it is language independent. We can use in different other language platform. We need design pattern to properly create a class, instantiate object, interact between objects and write reusable code. Structural, creational and behavioral are its types.

## **3.1 Structural design**

It is made of variables and functions. So, when design pattern’s deal with the class structure then it comes in the structure of design patterns. It simplifies the structure by identifying the relationships. It focuses on, how classes inherit from each other.

Class diagram(final)

They are used to show the different objects in a system, their attributes, operations and relationships among them. It is static view of an application. Class notation consists of class name, class attributes and class operations.

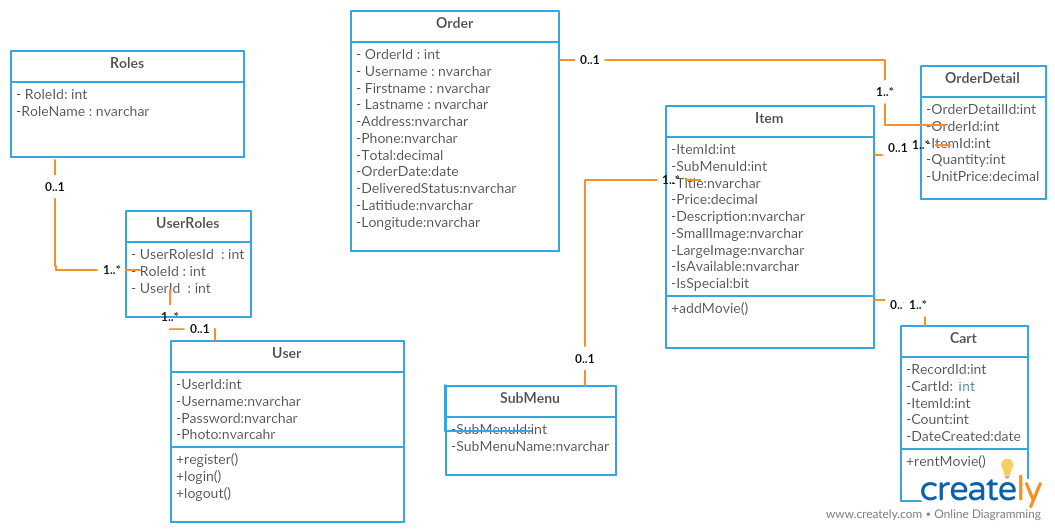


Fig: final class diagram

In above class diagram, user have one to many relationships with user roles also, roles have one to many relationships with user roles. Also, sub menu have one to many relationship with item, item have one to many relationship with cart and order detail and order have one to many relationship with order datail.

**Data flow diagram**

A data flow diagram is a graphical representation of flow of information of any system.it can be divided into logical and physical. Logical flow means flow of a system to perform certain functionality whereas physical flow means implementation of logical flow.

Building blocks of DFD

**Process**: it receives input and provides different output. Every process has its own name that performs some function. A rounded rectangle represents a process.

**Data flow:** a data flow is a path for data to move from one part to another. Straight lines incoming arrow are input data flow and outgoing arrow are output data flow. Process will be in middle of the flows.

**Data store:** a data store is connected to a data-flow. Each data store must have an input data flow and an output

**External entity:** a rectangle represents an external entity. They either supply data or receive data. They do not process data. It must be connected to a data-flow.

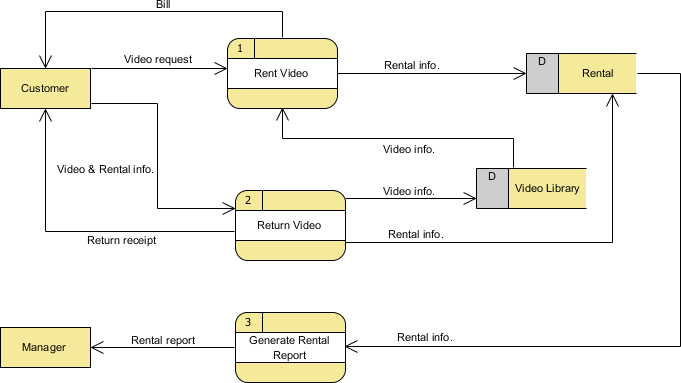


Fig: data flow diagram of online movie rental system

In the diagram the DFD shows how the system works when it is started. First the customer rent a video, in return customer gets the bill. Also, customer can return video, also its receipt gets returned to the customer. Returned video gets stored in video library database. From where customer can again rent a video. Rented video and returned video’s information is stored in rental database. A process of rental report is generated and reported to the manager.

## **3.2 Behavioral design**

When how does one class interact with other, then it comes under behavioral design pattern? It should be loosely coupled. it deals with communication of objects.

**Activity diagram**

It is basically a representation of flow from one activity to another. This activity can be described as operation of the system. It has pre-condition, actor input, system step, basic flow, alternative flow, returning alternative flow, parallel activities and post-conditions.

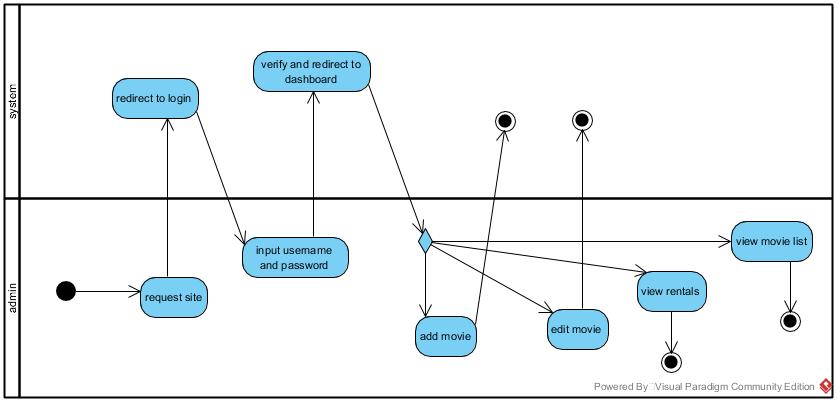


Fig: admin’s activity diagram

In the diagram, admin’s activity is pictured. First the admin requests a site, admin’s gets redirected to login in system. Admin inputs username and password. System verifies the entered username and redirects to dashboard and admin gets to add, edit, view rentals and movie.

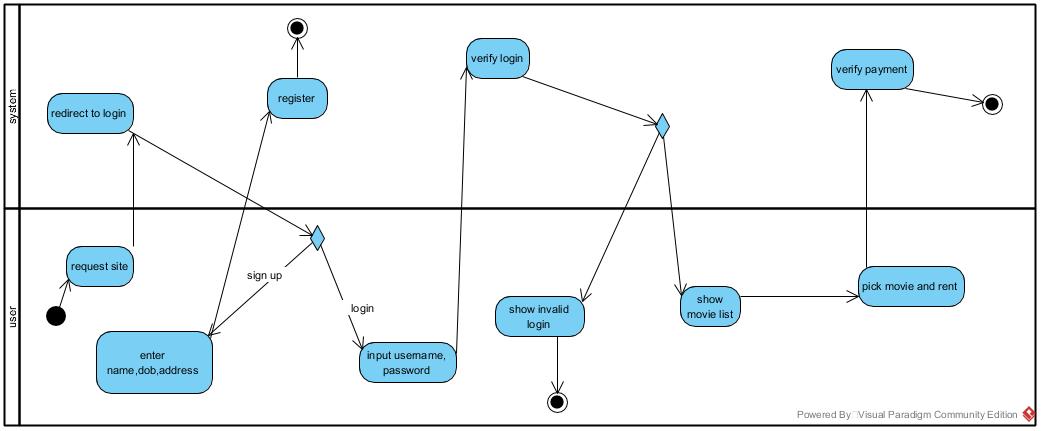


Fig: user’s activity diagram

First user requests a site and gets redirected to system’s login page. If user haven’t registered, they get redirected to register page, if they have already registered, they can login using valid username and password. If the login is successful they can enter into system’s where they can pick and rent a movie else user are shown invalid login error.

**Sequence diagram**

It shows how operations are carried out. It shows object interactions arranged in time sequence. A lifeline where different object live horizontally and exchange message between then.

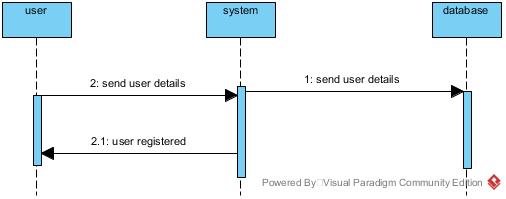


Fig: user’s registration sequence diagram

When user details are sent to a system, it gets saved into database and after that if valid information is provided, user gets registered successfully.

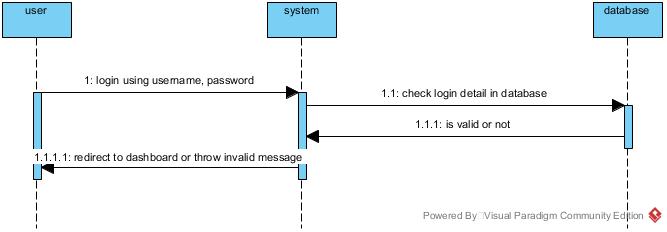


Fig: user’s login sequence diagram

During login valid username and password needs to be given to the system. Database checks login details, if given information is valid user is redirected to dashboard else it throws invalid message.

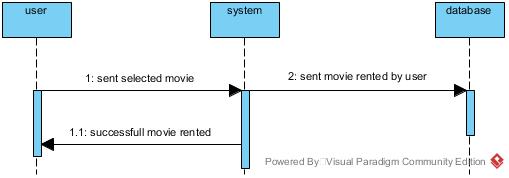


Fig: movie rent sequence diagram

During movie renting, user select a movie and rented movie information is sent to the database. After a movie is selected, it is rented successfully.

## **3.3 Database design**

A collection of process that helps in implementing data management system. Its objective is to produce logical and physical design models of proposed database system.

**Data dictionary**

A data dictionary is a collection of descriptions of the data objects or items in a data model for the benefit of programmers and others who need to refer to them. A first step in analyzing a system of objects with which users interact is to identify each object and its relationship to other objects. This process is called data modeling and results in a picture of object relationships.

Here, I have created the data dictionary for the online Movie Rental System which is as given below:

**Roles**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data Type | Length | Constrains | | Description |
| Null | Key |
| RoleId | int |  | no | Pri | ID number of Primary  Key |
| RoleName | Nvarchar | 50 | yes |  | Provided Role Name |

**UserRoles**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data Type | Length | Constrains | | Description |
| Null | Key |
| UserRolesId | int | - | no | pri | Unique Id of the user |
| UserId | int | - | yes | for | Id from user table |
| RoleId | int | - | yes | for | Id from role table |

**Users**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data Type | Length | Constrains | | Description |
| Null | Key |
| UserId | int | - | no | Pri | Unique Id of the user |
| Password | Nvarchar | -50 | Yes |  | password of the user |
| Username | Nvarchar | 50 | yes |  | Username of the user |
| Photo | Nvarchar | 50 | yes |  | Phone number of the user |

**Cart**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data Type | Length | Constrains | | Description |
| Null | Key |
| RecordId | Int |  | no | Pri | Unique Id of the cart table |
| CartId | Nvarchar | 50 | yes | for | Id provided by cart |
| ItemId | int | - | yes | for | Id provided by item |
| Count | int | - | yes |  | No. of movies |
| DateCreated | Date |  | yes |  | Date of movie rented |

**SubMenu**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data Type | Length | Constrains | | Description |
| Null | Key |
| SubMenuId | int | - | no | Pri | Id number for the SubMenu |
| SubMenuName | Nvarchar | 50 | yes |  | Name of the SubMenu |

**OrderDetail**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data Type | Length | Constrains | | Description |
| Null | Key |
| OrderDetailId | int | - | no | Pri | Unique id of orderdetail |
| OrderId | int | - | yes | for | Id provided by order table |
| ItemId | int | - | yes | for | Id provided by item table |
| Quantity | int | - | yes |  | Number of movie purchase |
| UnitPrice | decimal | (18,0) | yes |  | Price of payment |

**Item**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data Type | Length | Constrains | | Description |
| Null | Key |
| ItemId | Int | - | no | Pri | ID number for the item |
| SubMenuId | int | - | yes | For | submenu id from the submenu table |
| Title | nvarchar | 50 | yes |  | Movie name |
| Price | decimal | (18,0) | yes |  | Movie price |
| Description | nvarchar | -50 | yes |  | Movie description |
| Small Image | nvarchar | 50 | yes |  | Movie small image |
| Large Image | nvarchar | 50 | yes |  | Movie large image |
| IsAvailable | nvarchar | 50 | yes |  | Checking movie available |
| Is Special | bit |  | yes |  | Checking movie special |

**Order**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data Type | Length | Constrains | | Description |
| Null | Key |
| OrderId | Int | - | no | Pri | ID number for the order |
| Username | nvarchar | 50 | yes |  | User registered name |
| First name | nvarchar | 50 | yes |  | Users first name |
| Last name | nvarchar | 50 | yes |  | Users last name |
| address | nvarchar | 50 | yes |  | Users address |
| Phone | nvarchar | 50 | yes |  | Users Phone |
| Total | nvarchar | (18,0) | yes |  | Users payment |
| Order Date | date | 50 |  |  | Date ordered |
| Delivered status | nvarchar |  |  |  | True or false |

**ER diagram**

it is used to sketch out the design of the database. It shows the logical structure of databases by defining their entities, attributes and relationship between them. They use different set of symbols such as rectangles, diamonds and ovals.



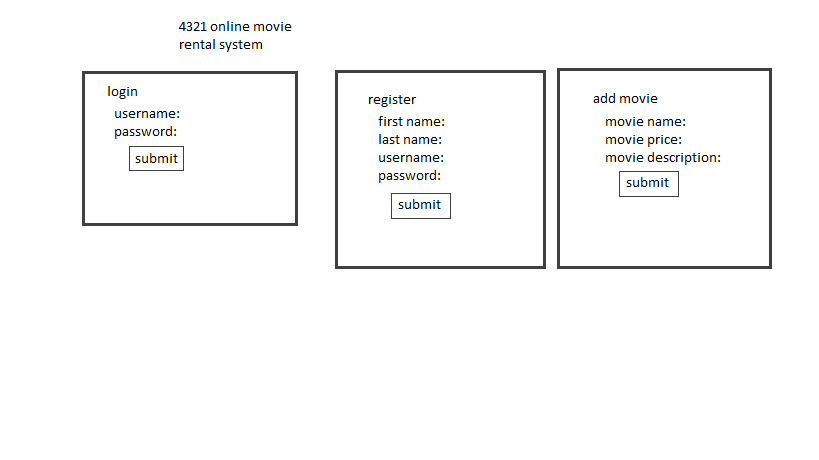
Fig: ER diagram for the project

In above ER diagram, userid is primary key for user table, rolesid is primary key for userroles table, Userroleid is primary key and roleid and userid are foreign key for userroles table. Submenuid is primary key for submenu table, recordid is primary key for cart table. Itemid id primary key for item table. Orderid is primary key for order table and orderdetailid is primary key for order detail table.

## **Architecture**

3.4.1 UI

UI or user interface is a design to make user interaction as simple as possible. It can be finished drawing through hands too. It should show technical functionality and visual elements. It should be adaptable to changing user needs.



# CHAPTER 4: IMPLEMENTATION

Implementation is the third phase of software development lifecycle. It is the part where programmers program code for the project. It’s the part where real working system are built and their responding functions. **Code and UI are in appendix.**

## **4.1 programming language**

Programming language is set of instructions that provides various output. It is instruction for computer that generates or performs special actions. Thousands of programming language have been created and are being created every year. Most popular programming languages are c, C++, java, C#, etc.

For my project I have used C# dot net programming language with MVC framework. I have used .NET MVC with controller and views and razor pages for building web UI more easily. Razor pages covers more features. C# is one of the powerful object-oriented language developed by Microsoft. It can be used to create various application like web, desktop and other types of application using visual studio. MVC stands for Model, View and Controller. Model is used to retrieve and store objects in database. View is a user interface. Controller is request handler.

## **4.2 Development Environment**

Development environment is a collection of tools for the development of programs. It includes testing and debugging as well. It is also called IDE (Integrated development environment) in programming terms. Basically, development environment refers entire environment which includes development and staging and IDE is used to code in local application. Various types of IDE are Microsoft Visual studio, phpstorm, php, mysql etc.

For my project, I have used Microsoft Visual studio for development environment. Visual studio community is the most common used and is free of cost, featured for students and programmers. It supports 16 different programming languages. Many websites, web apps, Mobile apps can be created through this IDE. I have developed a website through the help of this IDE.

# CHAPTER 5: TESTING

Testing is done at the end of software development lifecycle. Its main purpose is to check if actual results have been produced from the designed software. It helps in finding out where the error lies, how the system is performing and supposed to perform. It can be done manually or also through some automated tools.

For my project, I have used black box testing and white box testing.

## **Black box testing:**

Black box testing is the process of testing a software where functionality of the application is checked rather than the internal code structure. The detail structure and paths of is not checked in this part of testing. Just input and output is focused on this type of testing.

## **Unit testing:**

Unit testing is a part of white box testing where test is done and it impacts directly upon your application. Its main aim is to prevent as many bugs from introduced. Unit testing is mainly done by developers themselves.

|  |  |
| --- | --- |
| Test no.1 | 1 |
| Purpose of the test | To check whether login function is working properly or not by providing username and password |
| Test data | User(ramesh@gmail.com)  Admin (Hari) |
| Test name | Login test (user and admin) |
| Expected Result | Login test needs to be success and gets user directed to homepage and admin to dashboard. |
| Actual result | User gets successfully logged in to home page and admin gets successfully logged in to dashboard. |
| Test Status | success |

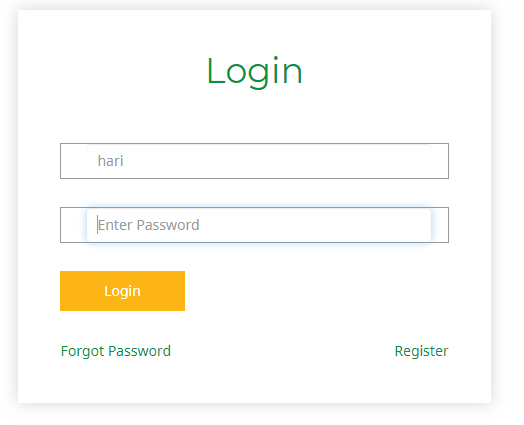


Fig: Black Box testing for admin login

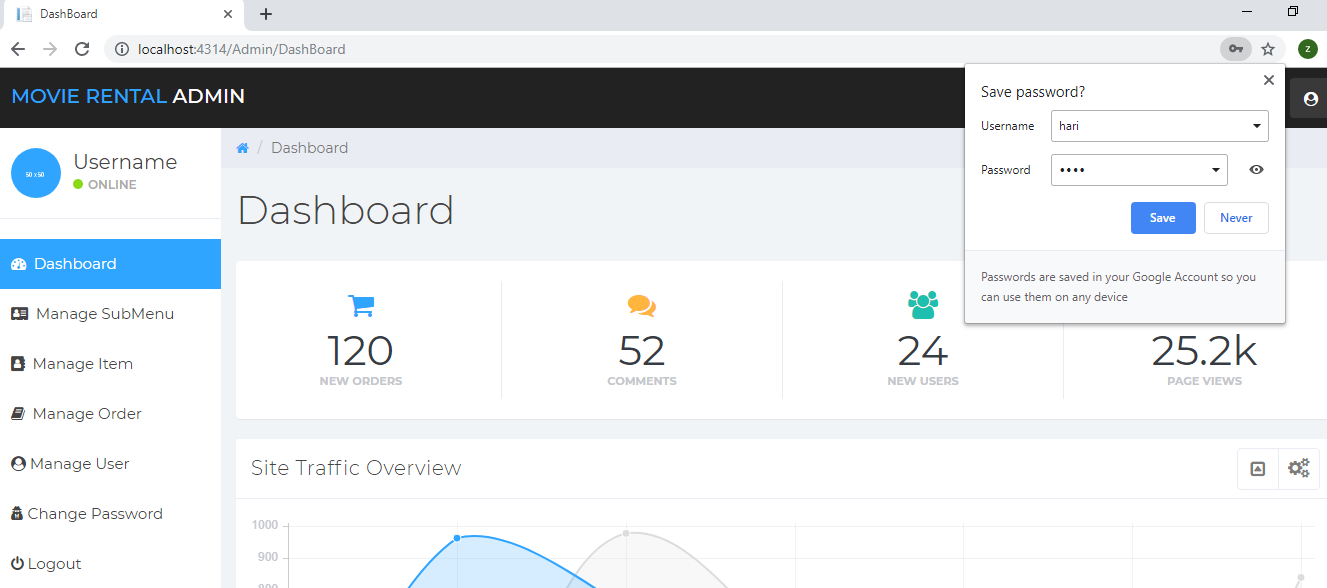


Fig: redirected to admin dashboard after login

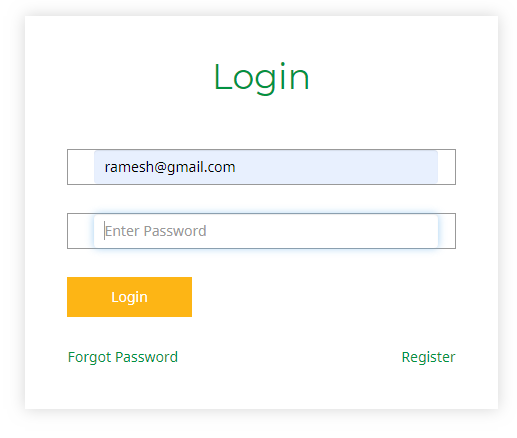


Fig: Black box testing for user login

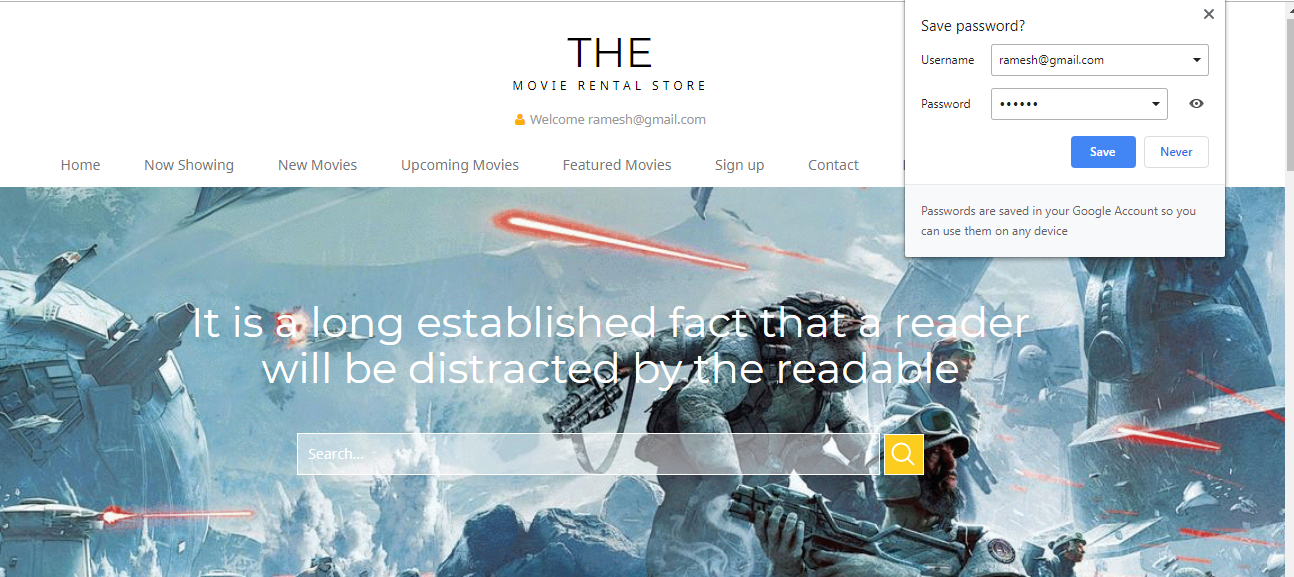


Fig: user gets redirected to homepage after login

|  |  |
| --- | --- |
| Test no.1 | 2 |
| Purpose of the test | To check whether registration function is working properly or not by providing required information fields |
| Test data | shyam@gmail.com |
| Test name | Registration test |
| Expected Result | User needs to gets registered successfully |
| Actual result | User gets registered successfully |
| Test Status | success |

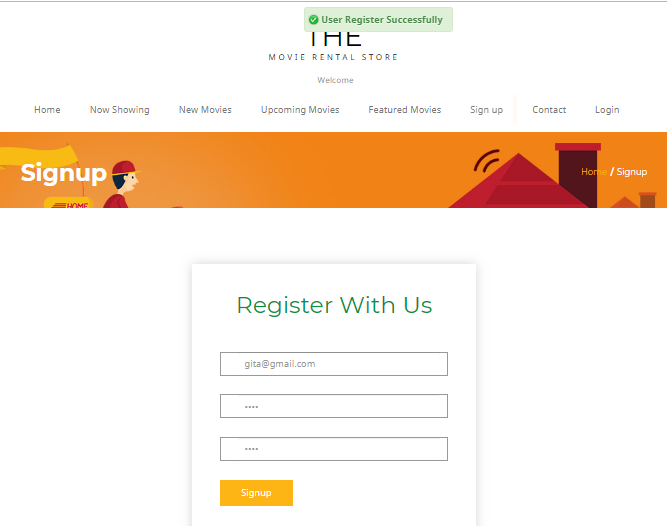


Fig: Black box testing for user registration

|  |  |
| --- | --- |
| Test no.1 | 3 |
| Purpose of the test | To check whether movie gets managed in the system |
| Test data | Spiderman |
| Test name | Movie manage test |
| Expected Result | Movie needs to be added, edited or deleted in the system |
| Actual result | Movie added, deleted successful |
| Test Status | success |

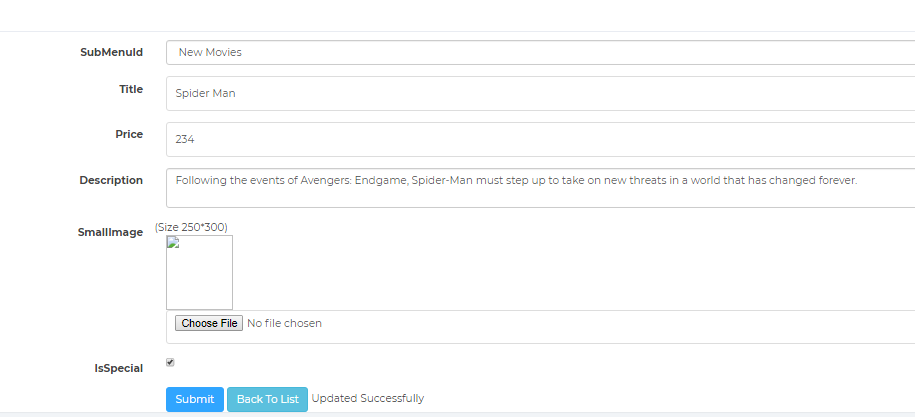


Fig: Black box testing for managing movie (Update)

|  |  |
| --- | --- |
| Test no.1 | 4 |
| Purpose of the test | To check whether movie gets rented successfully |
| Test data | Spiderman |
| Test name | Rent movie test |
| Expected Result | Movie gets rented and is shown on the cart items |
| Actual result | Movie rented successfully |
| Test Status | success |

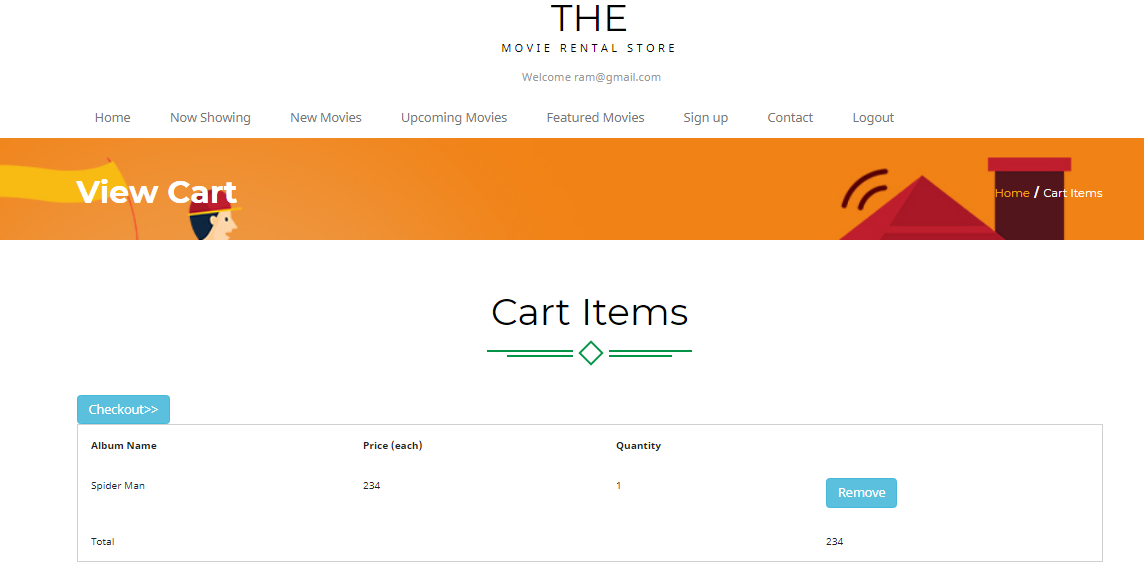


Fig: Black box testing for renting a movie

|  |  |
| --- | --- |
| Test no.1 | 5 |
| Purpose of the test | To check if password changing function is working or not |
| Test data | Hari(admin) |
| Test name | Change password test |
| Expected Result | Password changed successfully |
| Actual result | Password change successful |
| Test Status | successful |

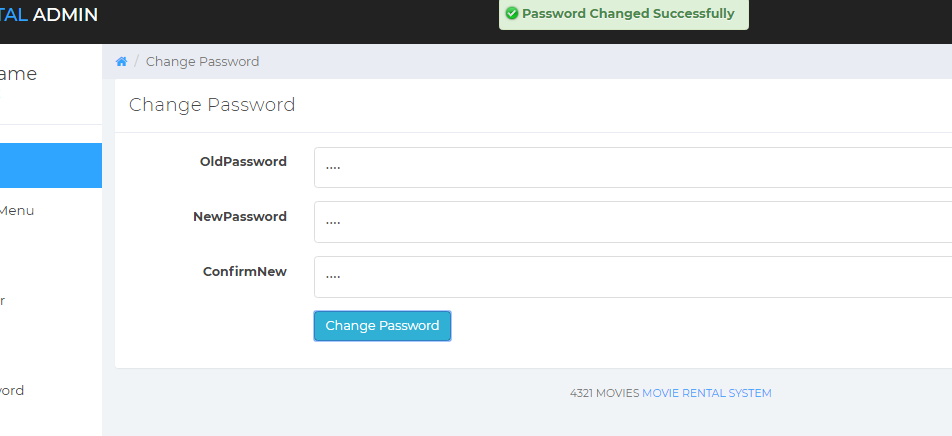


Fig: Black box testing for changing password

|  |  |
| --- | --- |
| Test no.1 | 6 |
| Purpose of the test | To check whether user can manage rented movie |
| Test data | Order 6 (Suraj) |
| Test name | manage rented movie |
| Expected Result | Admin can confirm or delete movie |
| Actual result | Confirm or delete successful |
| Test Status | success |

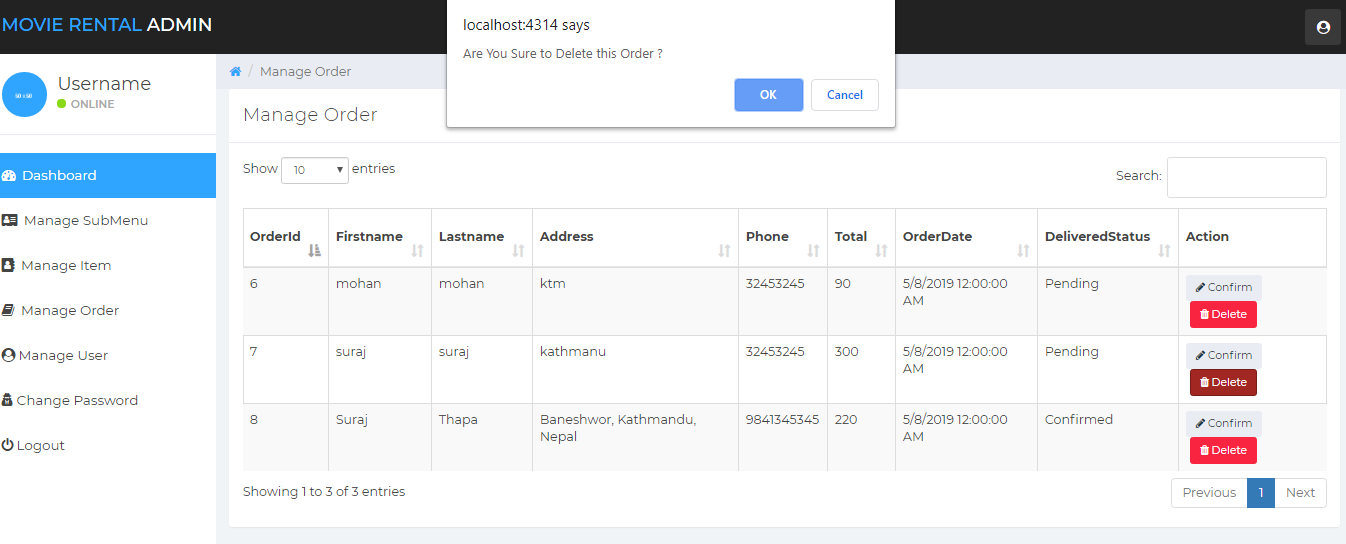


Fig: Black box testing for managing rented movie

|  |  |
| --- | --- |
| Test no.1 | 7 |
| Purpose of the test | To check whether provided address for delivery information gets submitted or not |
| Test data | Address for order |
| Test name | Order test |
| Expected Result | User orders is displayed successfully |
| Actual result | Ordered done successful |
| Test Status | success |



Fig: Black box testing for successful ordering

|  |  |
| --- | --- |
| Test no.1 | 8 |
| Purpose of the test | To check if when user clicks on the movie icon overview of the movie is shown or not |
| Test data | Spiderman |
| Test name | Quick overview test |
| Expected Result | Movie quick overview needs to be shown |
| Actual result | Over view successful |
| Test Status | success |

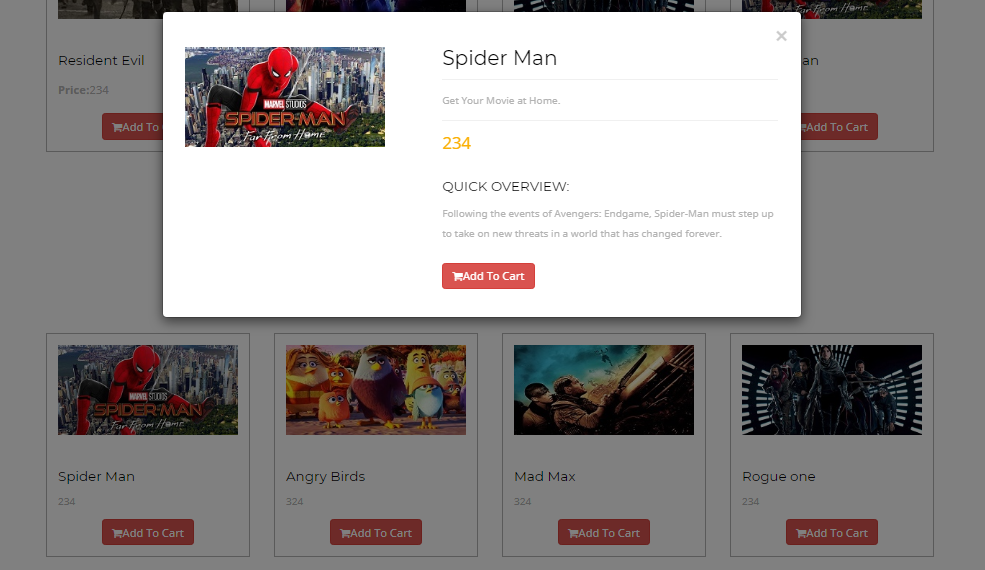


Fig: Black box testing for overview of movie

|  |  |
| --- | --- |
| Test no.1 | 9 |
| Purpose of the test | To check if admin gets to manage user |
| Test data | aa@gmail.com |
| Test name | User control test |
| Expected Result | User gets deleted successfully |
| Actual result | User deleted successful |
| Test Status | success |

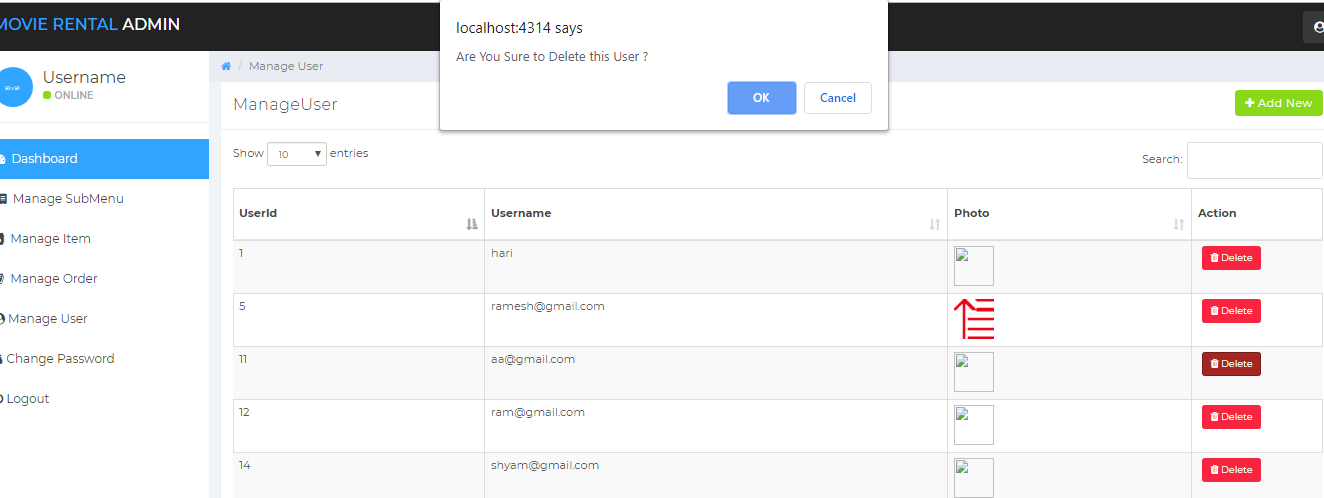


Fig: Black box testing for user control

|  |  |
| --- | --- |
| Test no.1 | 10 |
| Purpose of the test | To check if when user clicks on view more button on the home page, it needs to show list of movies |
| Test data | View more |
| Test name | View more test |
| Expected Result | Page should be directed to list of movies after clicking view more button |
| Actual result | List of movies page load successful |
| Test Status | success |

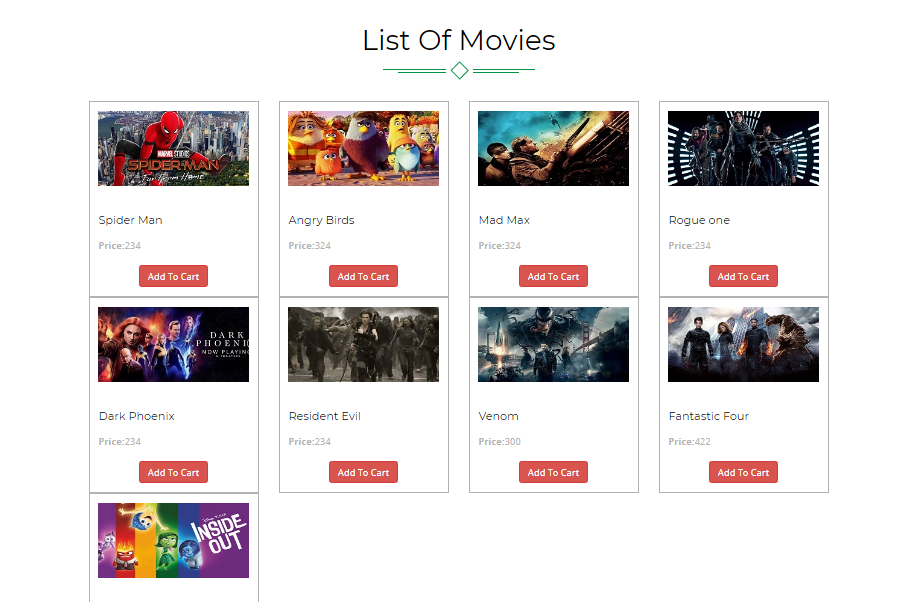


Fig: Black box testing for view more test

# CHAPTER 6: OTHER PROJECT ISSUES

## **6.1 Scheduling: Gantt chart**

In this section of my project, I will divide the days and schedule them for my working days. So, for the scheduling purpose I prepared time estimation tale and Gantt chart.

**6.1.1 Time Estimation Table**

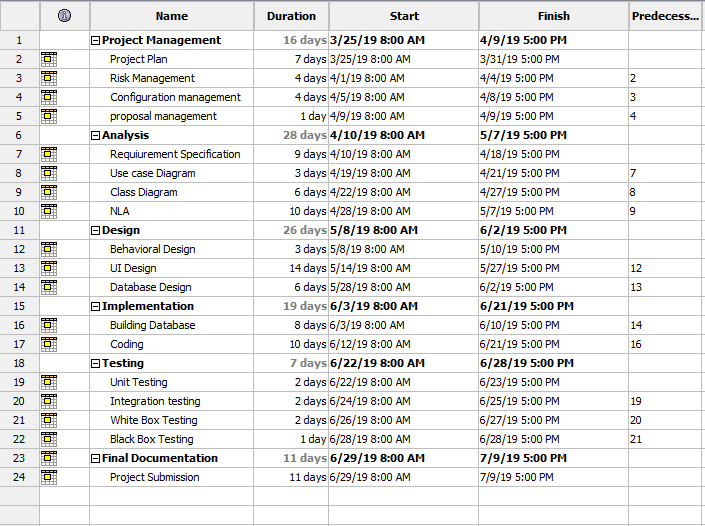


Fig: time estimation

**6.1.2 GANTT Chart**

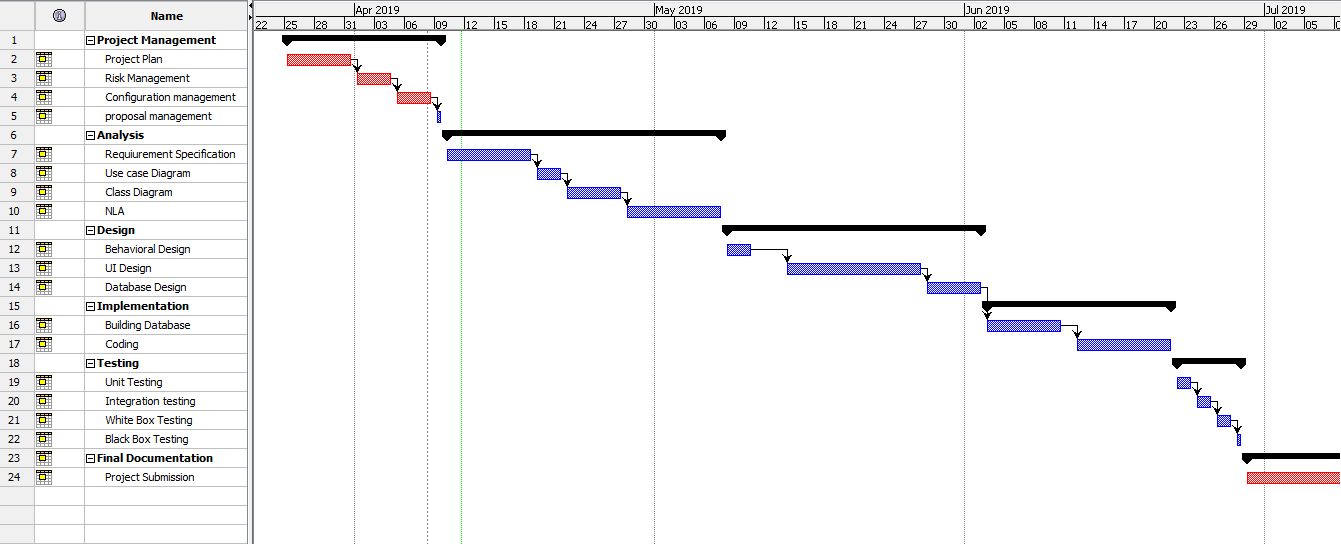


Fig: Gantt chart

## **6.2 RISK MANAGEMENT**

Risk management is the process of identifying, evaluating and prioritizing risks to minimize the probability of impact and maximizing realization of opportunities.

Impact = Likelihood\*Consequence

Risk likelihood are shown below:

|  |  |
| --- | --- |
| likelihood | value |
| Low | 1 |
| medium | 2 |
| high | 3 |

Risk consequence values are shown below:

|  |  |
| --- | --- |
| Consequence | Value |
| Very low | 1 |
| low | 2 |
| Medium | 3 |
| High | 4 |
| Very High | 5 |

Risk Consequence values are shown below:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S. N | Risks | Likelihood | Consequences | Impact | Solution |
| 1 | Insufficient resources | 2 | 3 | 6 | Collect all required resources |
| 2 | Hard disk failure | 1 | 2 | 2 | Backup plan |
| 3 | Server failure | 1 | 3 | 3 | Online backup plan |
| 4 | Lack of skill | 1 | 3 | 3 | Providing training for employees |
| 5 | scheduling problem | 2 | 4 | 8 | Dividing tasks in different schedule |

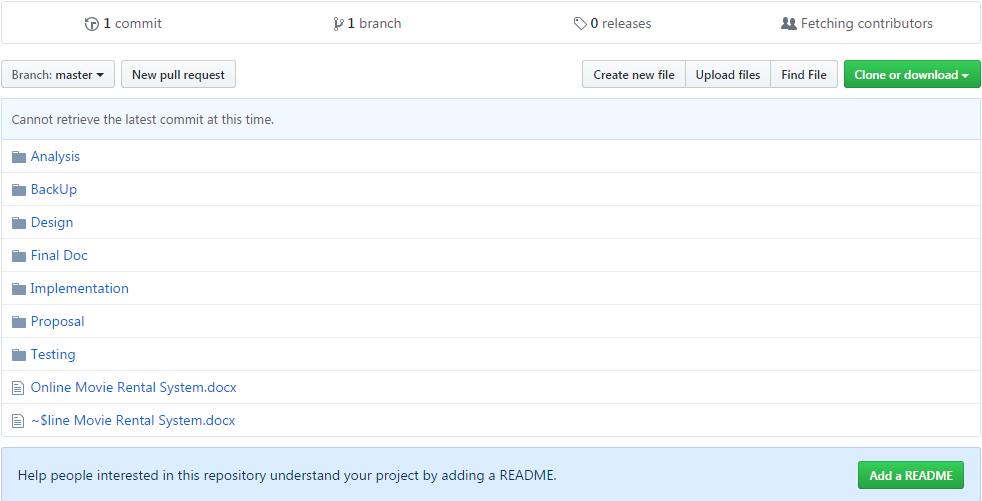
# CHAPTER 7: CONFIGURATION MANAGEMENT

Configuration management is a process for establishing and maintaining consistency of a product’s performance, functional, design and operational information.

This application helps in maintaining the functionality and operation. The main advantage of configuration management is to help in reliability.

Below is the files name that I have uploaded in GitHub:

(Link: https://github.com/ranjit0/computing-project)



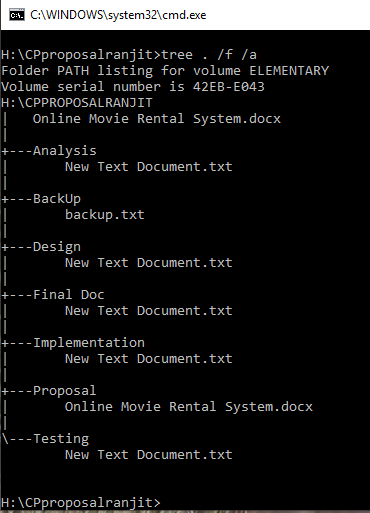


Fig: folder tree structure

**FUTURE WORK**

On this project many features could be added in the future. Some of them are listed below:

* More link buttons like Facebook, twitter and other possible live chat features could come up to date.
* Rating and reviewing pages could be added
* Advanced security features like authentications could be added
* Could be rentable from mobile applications like google, apple and Microsoft.
* Could be available in foreign country.

**User Manual**

The user manual in our web application is to provide support for the user. Through the help of manual user can possibly make use of the site. The diagram of user manual looks like this.

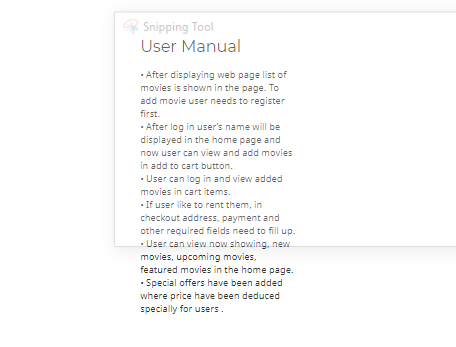


Fig: screen shot of user manual

For more ease of access here are some steps to guide user completely:

* For registration, there is a sign-up button in the home page where after providing required fields user gets registered.
* For log in, there is a log in button on the home page where after providing valid email and password user gets logged in.
* To view movie information, user just have to click in the icon of the movie.
* To rent a movie you can click add to cart button on the list of movies, then you get redirected to page where you can remove selected movies or go checkout for address and payment.
* If you like using our website you can send your thanks in contact page.

# CHAPTER 8: CONCLUSION OF THE PROJECT

Online Movie Rental System is user friendly web application where you can rent a movie anytime anywhere. Its features make customer easy to search, rent and buy movie. Admin can make changes to the user. Waterfall methodology and design pattern MVC is used in this project. Its only limitation is that it’s not currently in use in foreign countries.

In my project I have added some of the features which are not listed before during the start which are as follows:

* User can view movie information just by clicking on the icon of the image.
* After user clicks on add to cart, they get to remove the added movies.
* User can send their feedback in contact page.
* Admin can manage user.
* Admin can manage orders.
* Admin can change password for security purpose.
* Admin can make movie special for special offers purpose.
* In limitation, customer could rent one movie now they can rent multiple.

In my project some of the features or functions have been removed because of complexity which are:

* Integration testing have not been developed.
* Search movie does not work which was must have in Moscow prioritization
* Admin cannot register.
* User cannot subscribe to newsletter instead they can contact straight.

# REFERENCE & BIBLIOGRAPHIES

Google. (2019). MVC architecture. [online]Available at:

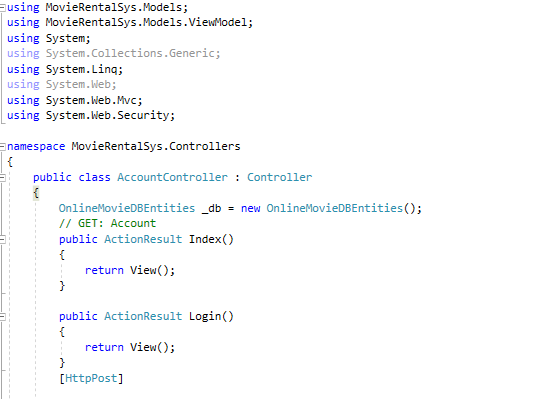
[https://developer.chrome.com/apps/app\_frameworks](https://developer.chrome.com/apps/app_frameworks%5baccessed) [accessed 6 April. 2019]

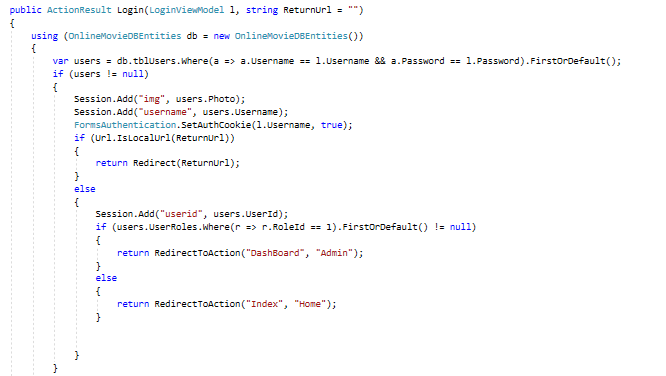
SEoK. (2019). system architecture. [online]Available at:

<https://www.sebokwiki.org/wiki/System_Architecture> [accessed 6 April. 2019]

# CHAPTER 9: APPENDIX

Implementation of dot net mvc framework







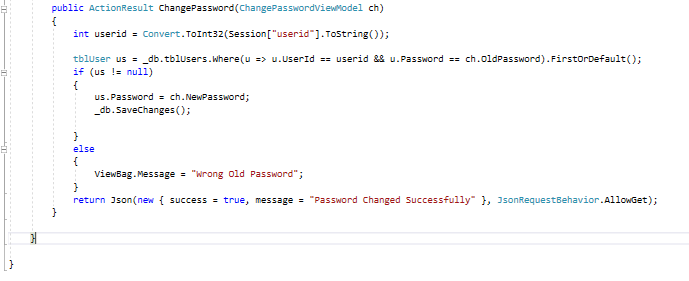


Fig: Account Controller

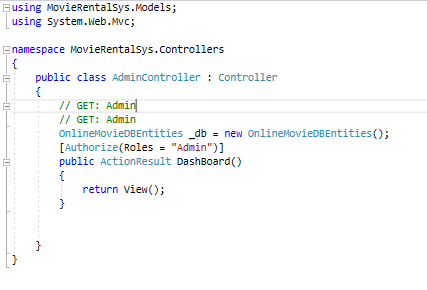


Fig: Admin Controller





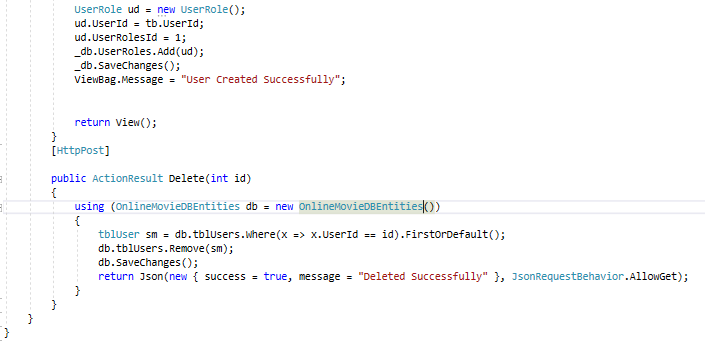
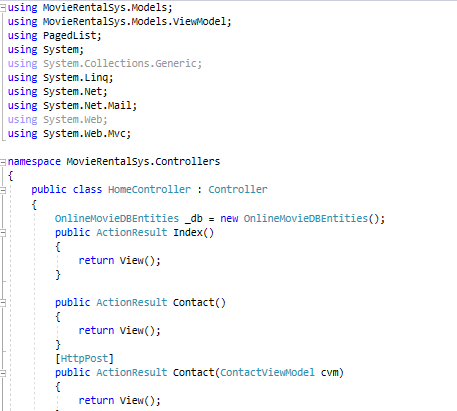


Fig: Admin User Controller



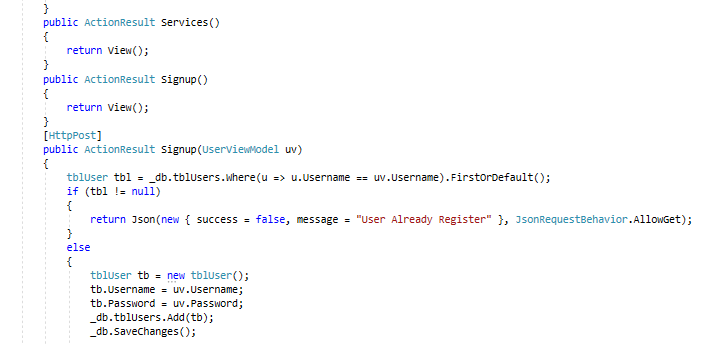
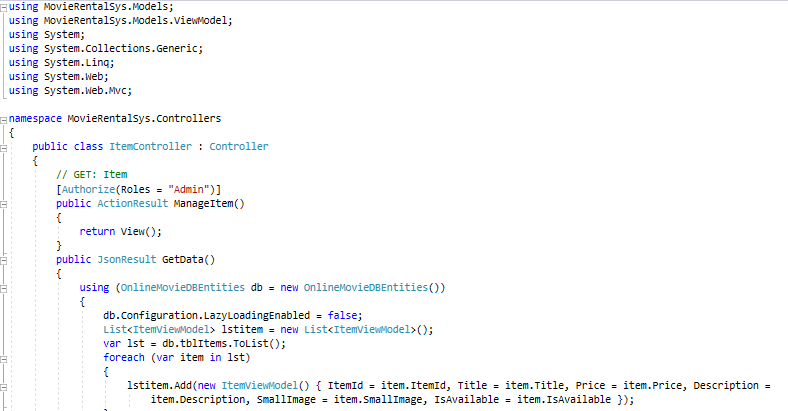
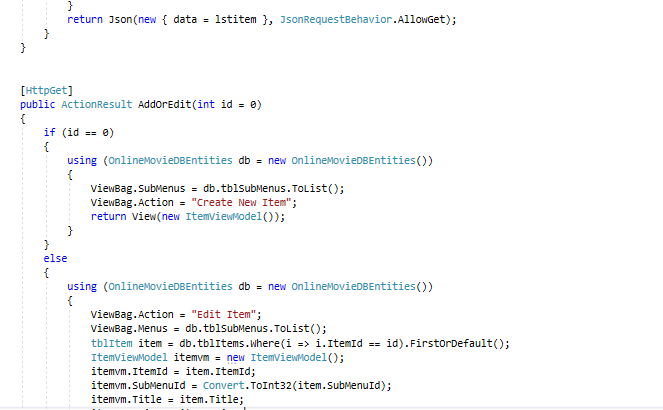


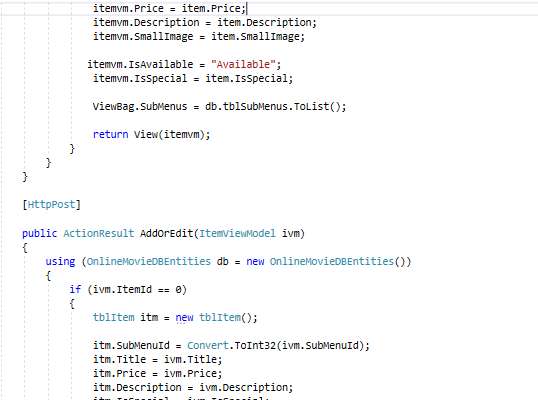




Fig: Home Controller









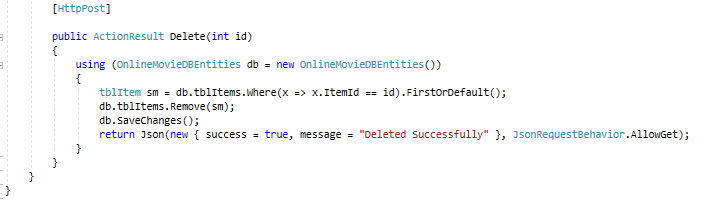
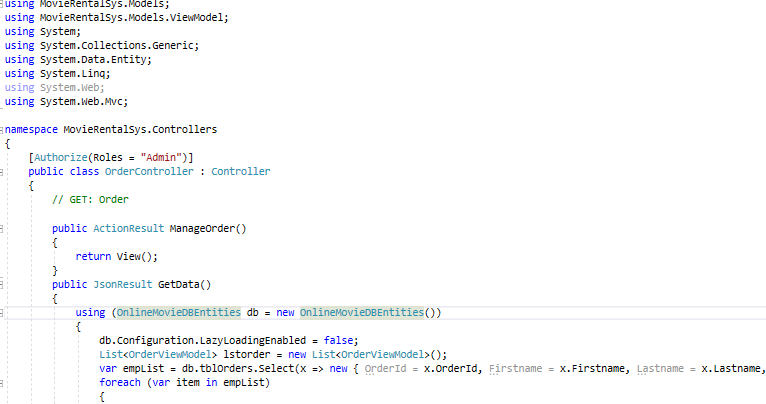
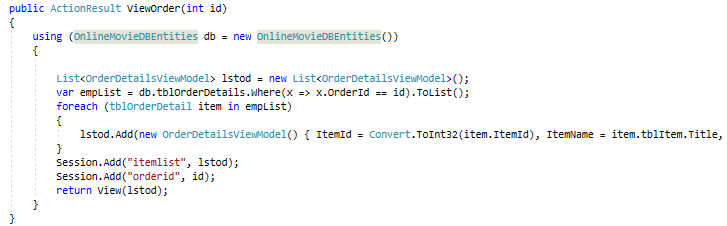
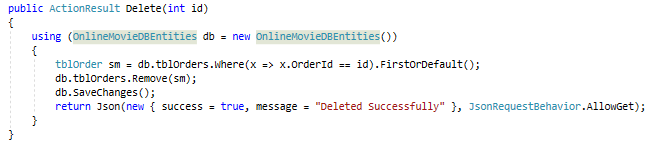


Fig: Item Controller







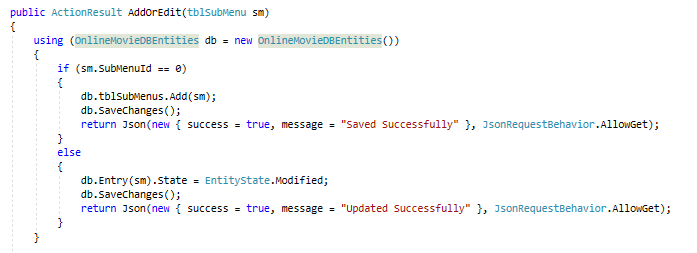
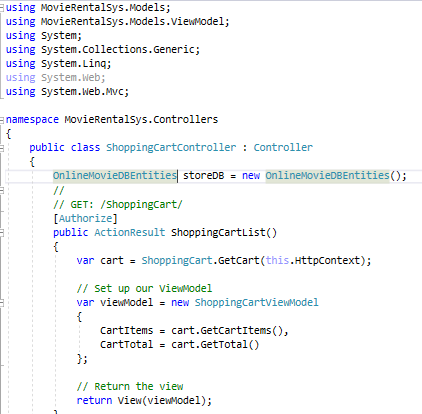
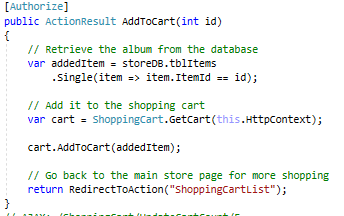
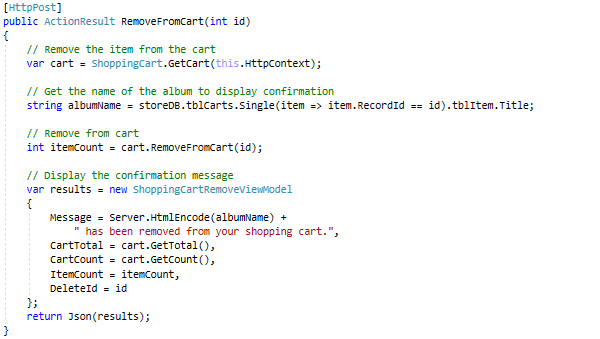


Fig: Order Controller







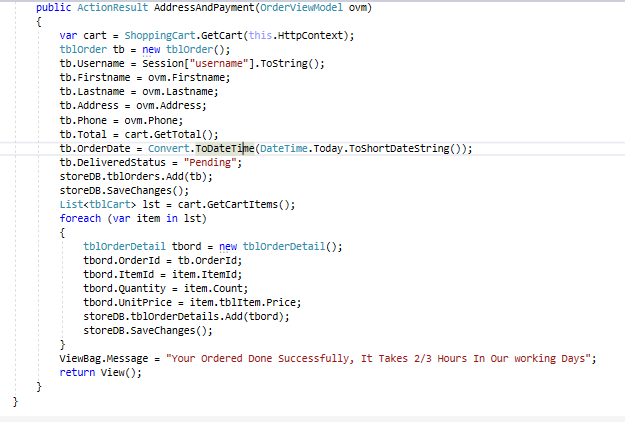
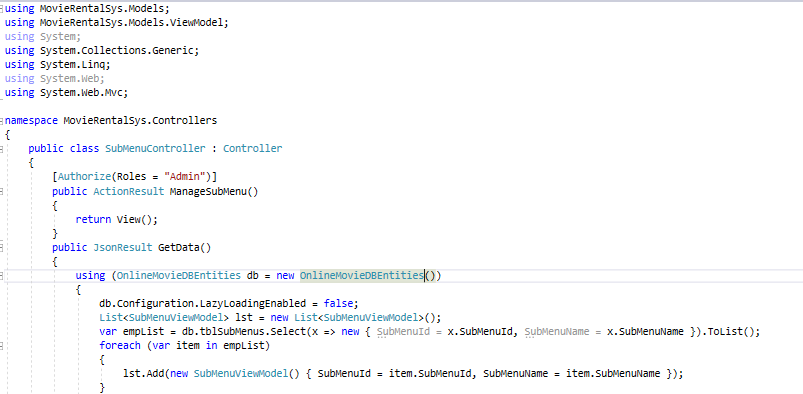
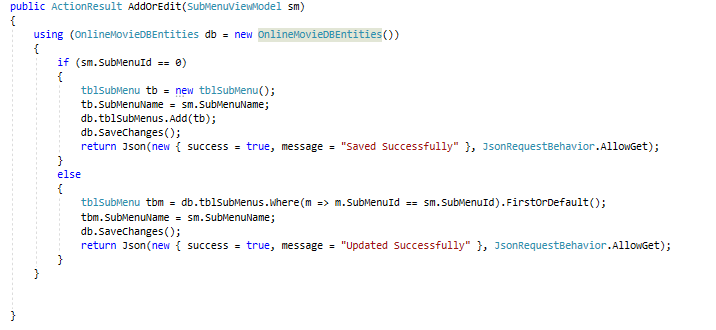


Fig: shopping cart controller





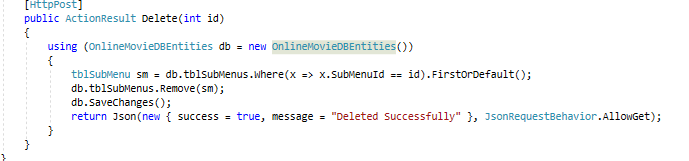


Fig: Submenu controller

Fig: Bill view Model

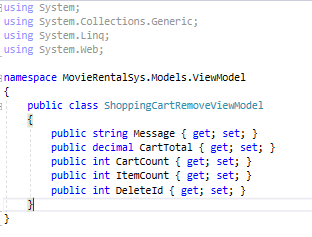


Fig: Cart remove view Model

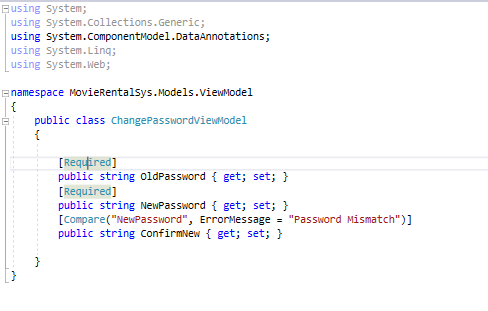


Fig: Change password view model

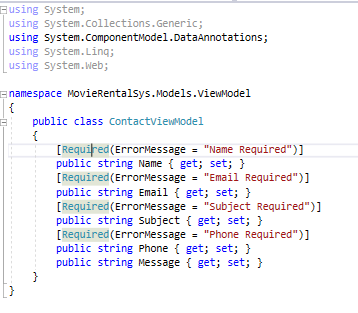


Fig: contact view model

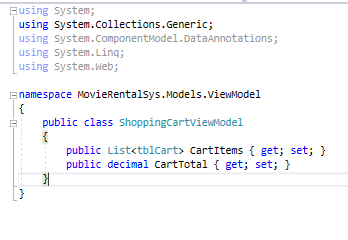


Fig: Shopping cart view model

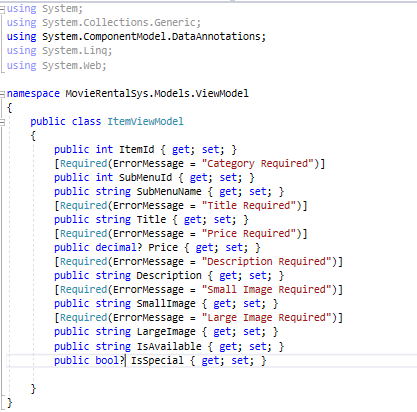


Fig: item view model

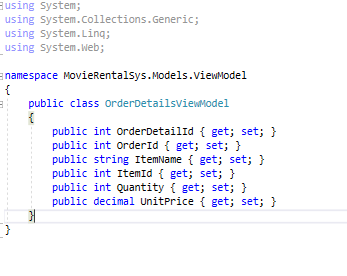


Fig: order detail view model

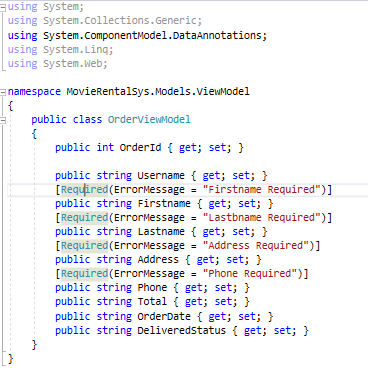


Fig: order view model

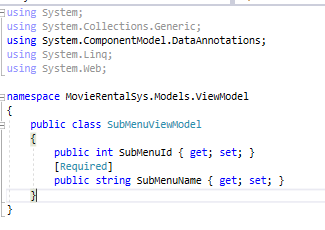


Fig: sub menu view model

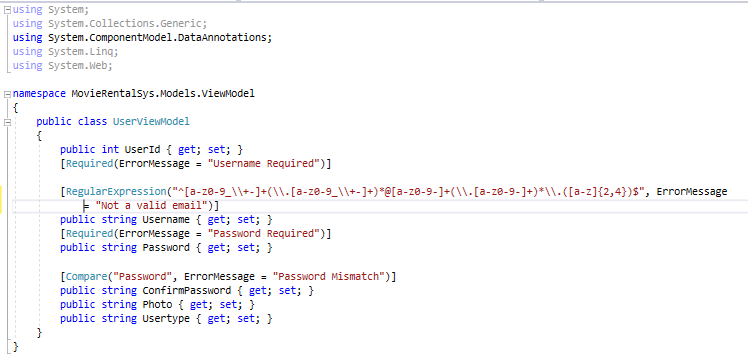


Fig: user view model



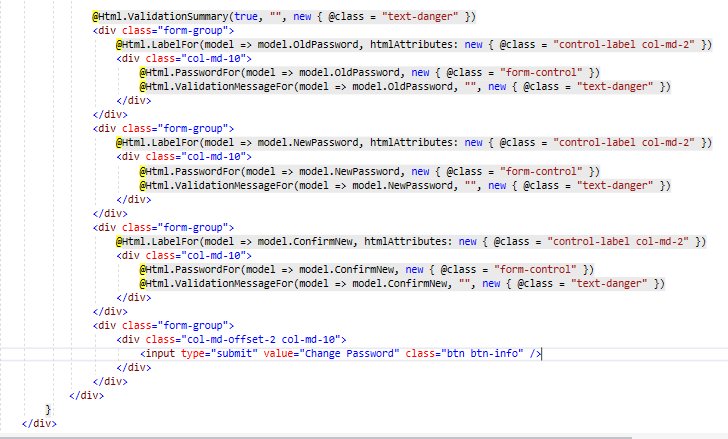


Fig: change password.cshtml





Fig: login.cshtml



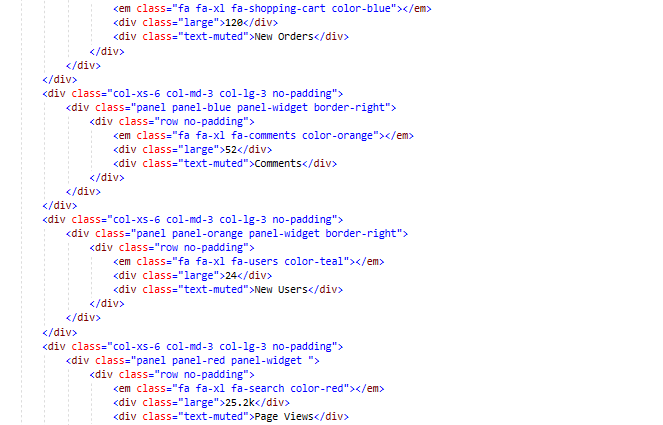








Fig: dashboard.cshtml









Fig: adminuser(addoredit.cshtml)





Fig: adminuser(manageuser.cshtml)



Fig: View item.cshtml





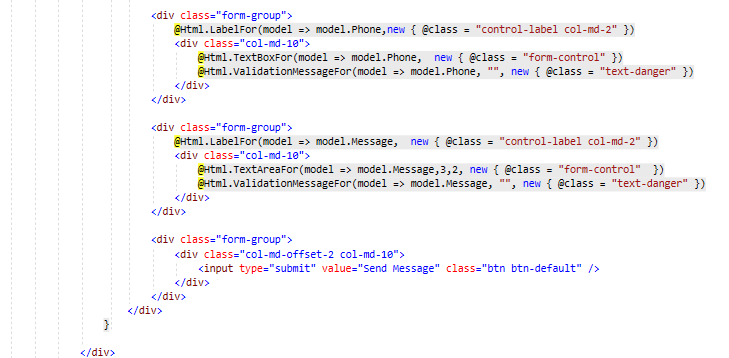


Fig:Contact.cshtml





Fig:ForgetPassword.cshtml

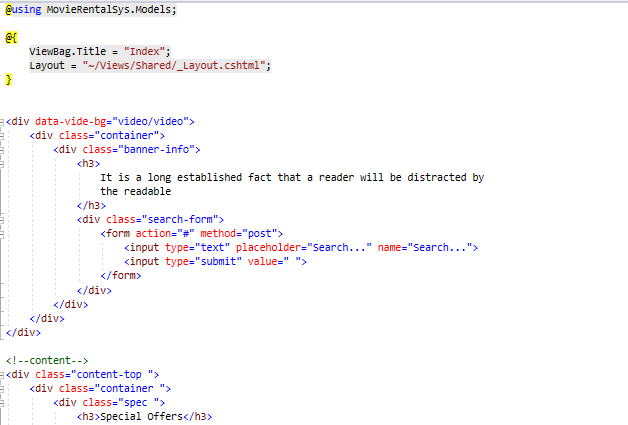










Fig: index.cshtml



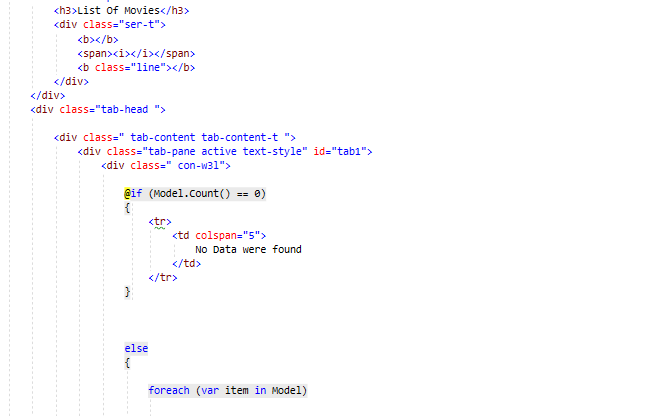






Fig: productlist.cshtml





Fig: signup.cshtml

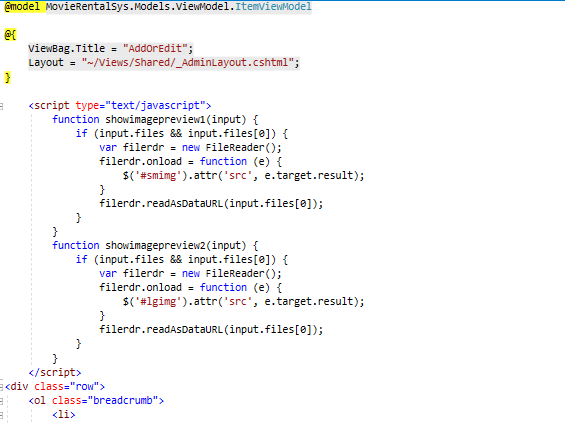








Fig: addoredit.cshtml(item)



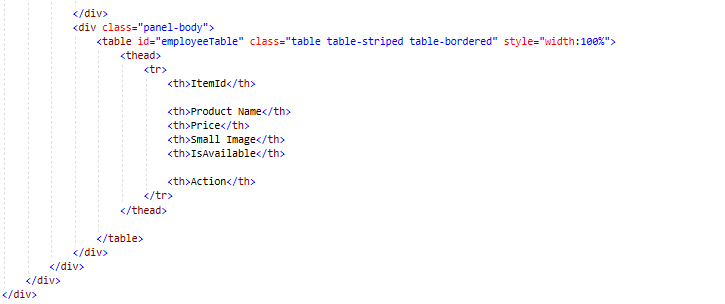


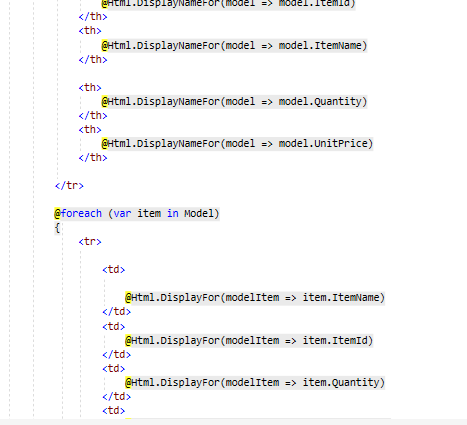
Fig: Manageitem.cshtml(item)





Fig: manageorder.cshtml(order)





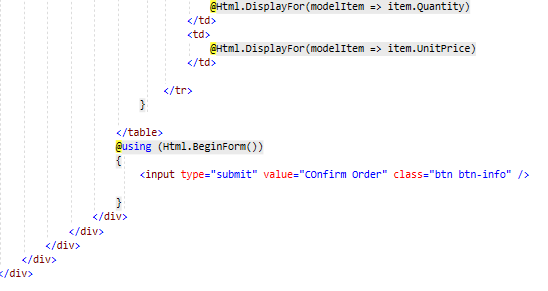


Fig: View order.cshtml(order)









Fig: adminlayout.cshtml







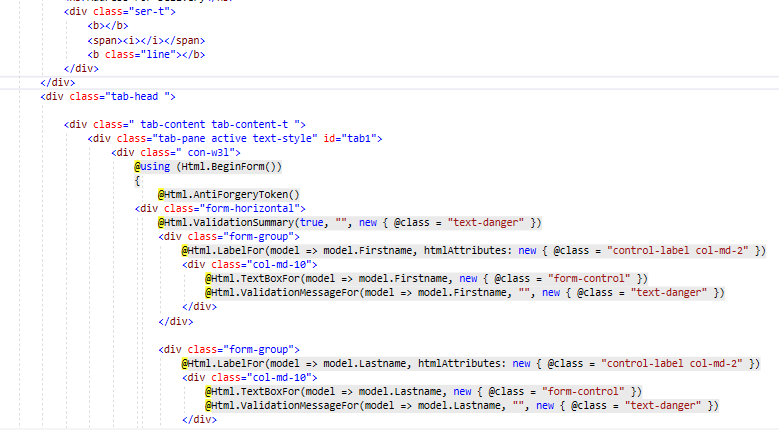


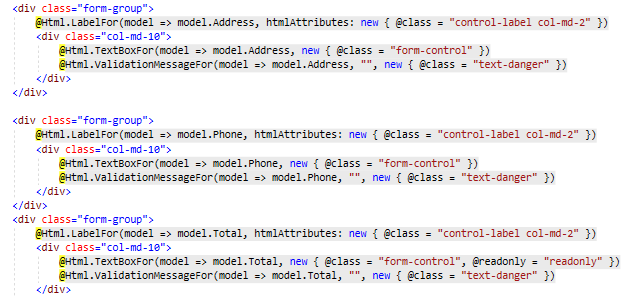




Fig: layout.cshtml(UI)







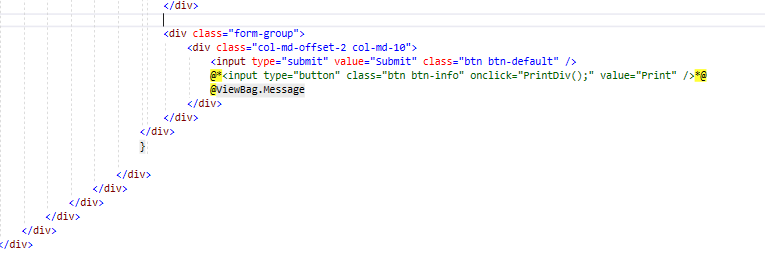
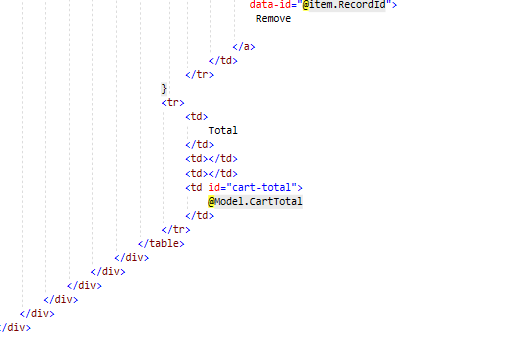


Fig: addressandpayment.cshtml







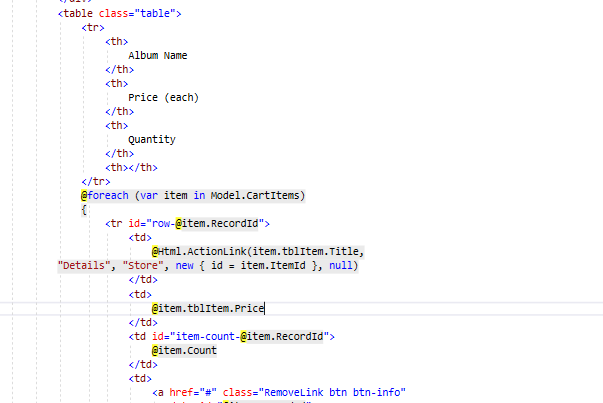


Fig: shopping cartlist.cshtml

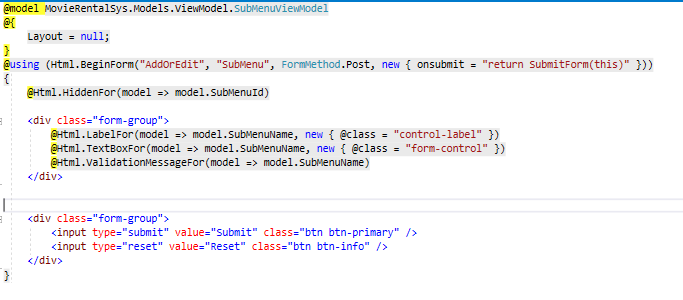


Fig: addoredit submenu.cshtml



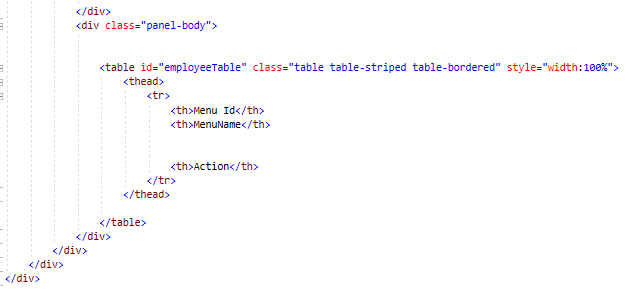
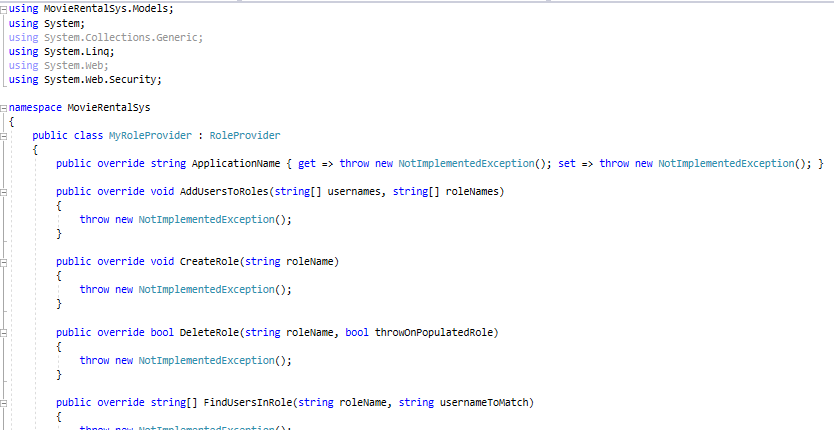
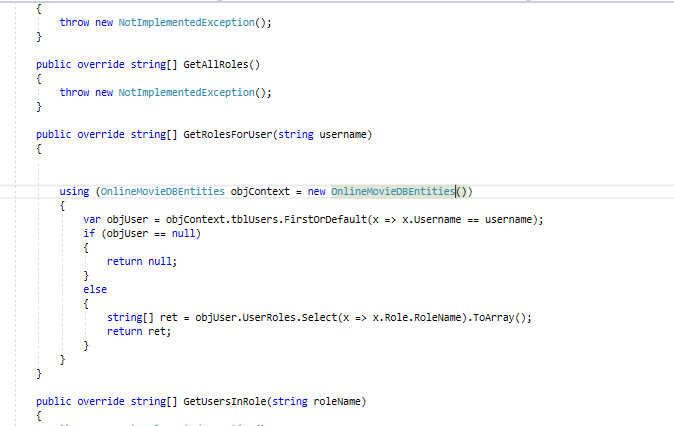


Fig: managesubmenu.cshtml





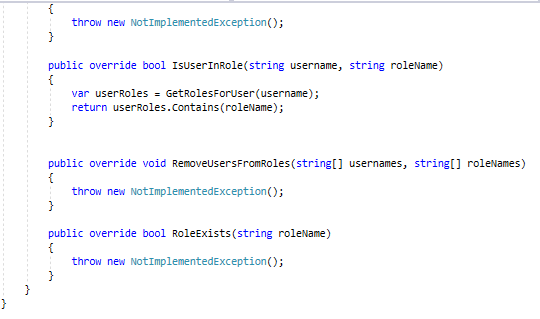


Fig: myroleprvider.cshtml