# Madflix

Minimalist application built with ASP.NET Core Web Application. Using Project templates for creating ASP.NET applications.

### Live example

https://madflix.azurewebsites.net/

### How to start

Just start and enjoy via https://madflix.azurewebsites.net/

```

Project has separated database in Azure SQL Database. Check out \Madflix\Web.config

### Example of web.config file

name="DefaultConnection"

connectionString="Data Source=tcp:madflixdbserver.database.windows.net,1433;Initial Catalog=Madflix\_db;User Id=\*@\*;Password=\*\*\*"

providerName="System.Data.SqlClient"

You can save this project and open .sln file with your Visual Studio.

Project uses this dlls and modules:

"AutoMapper" version="4.1.0"

"bootbox" version="4.3.0"

"bootstrap" version="3.4.1"

"jquery.datatables" version="1.10.11"

"toastr" version="2.1.1"

"Twitter.Typeahead" version="0.11.1"

"toastr" version="2.1.1"

```

### License

This project is licensed under the LLStudio

## Description

Application is used for rental store. There are three main modules. Customers, Movies, Rentals. You can log in as an admin or as a quest, credential details are provided in login page. If you log in as an admin you can manage movies database, you are able to add, update and delete movies. If you log in as a quest you can only view movies from database.

### Customers

In Models, there is a class 'Customer' linked to the Customers table in db, there is also 'CustomerDto' data transfer class that is mapped using Mapper on 'Customer' class. 'CustomerDto' is used in all defined actions in controller.

There are two types of controllers, regular controller 'CustomersController' and Api controller 'CustomersController'. Each of mentioned controller's actions are used for different functionality.

There are three views using Customers module.

1. Index view

-This view is listing customers from db using Customer object for it's @model

-There is a link in Customer column to "Edit" action on "Customers" controller. It is opening 'CustomerForm' view with selected customers details

-Data are rendered using jquery.datatables

-There is an ajax call to api action for delete method

-There is a "bootbox" confirmation message when deleting item

2. CustomerForm view

-this view is used for adding and edditing customers

-is using 'CustomerFormViewModel' object for its's @model

-CustomerFormViewModel is combining Customer class and IEnumerable of MemberhipType class

-there is a custom validation 'Min18YearsIfAMember' when adding a new customer

-fields are render using Html form

-there is enabled server side validation

-Save button connected to Save action in Customers controller

3. Details view

-this view is used to preview details

-is using 'Customer' object as it's @model

### Movies

In Models, there is a class 'Movie' linked to the Movies table in db, there is also 'MovieDto' data transfer class that is mapped using Mapper on 'Movie' class. 'MovieDto' is used in all defined actions in controller.

There are two types of controllers, reqular controller 'MoviesController' and Api controller 'MoviesController'. Each of mentioned controller's actions are used for different functionality.

There are four views using Movies module.

1. List view

-This view is listing movies from db and is using Movie object for it's @model

-There is a link in Movie column with to "Edit" action on "Movies" controller. It is opening 'MovieForm' view with selected movie details

-Data are rendered using jquery.datatables

-There is a "bootbox" confirmation message when deleting item

-There is an ajax call to api action for delete method

2. MovieForm view

-this view is used for adding and editing movies

-is using 'MovieFormViewModel' object for its's @model

-MovieFormViewModel is combining selected properties from Movie class, IEnumerable of MemberhipType class and two constructors

-fields are render using Html form

-there is enabled server side validation

-Save button connected to Save action in Movies controller

3. Details view

-this view is used to preview details

-is using 'Movie' object as it's @model

4. ReadOnlyList viw

-this is read only view for listing movies from database

-there is a link in Movie column redirecting to 'Details' view

### Rentals

In Models, there is a 'Rental' class linked to the Rentals table in db. There is a 'NewRentalDto' data transfer class used for creation of a new rental. This class has CustomerId and list of MovieIds properties. This is used in 'NewRentalsController' api controller.

There is a one view using Rental module

1. New view

-there is a "Twitter.Typeahead" function in both fields that is quering over customers and movies tables with ajax api calls

-there is a "toastr" confirmation message

-when rental is created availability of particular movie in database is decreased