Web xem phim

***Note****: Ý nghĩa màu chữ*

***Đen****: cần làm*

***Xám****: đã làm xong*

***Đỏ****: không làm được*

# To Do

## **Chia việc:**

* **Kiên**:
  + WebClient: Update Profile (Client)
    - Nhận vào UserID từ URL
      * Demo:
        + 
        + 
    - Call API -> Get User by ID -> Lấy dữ liệu về User -> Gán lên các <input> để người dùng sửa
      * ko dc phép sửa Role
      * Password: (DB: đã dc mã hóa)
        + -> Khi người dùng Update Password thì phải mã hóa lại trước khi Update vào DB
    - -> Submit form -> call API: Update User -> Show Modal: thông báo
      * Thành công -> Nhấn OK -> Về index.html
      * Thất bại -> Nhấn OK -> Tắt Modal (vẫn ở trang này)
    - Thêm 1 button Back -> Nhấn vào thì về index.html
* **Thông**:
  + WebClient: Admin
    - Quản lý Movie: CRUD Movie (AJAX – như đã học)
      * List:
        + Hiển thị ảnh (thẻ <img>) thay vì image Path
        + Ko cần hiển thị VideoPath
        + Paging
      * Add / Update:
        + 1 nút Back để quay về List: nếu ko muốn add hay update nữa thì bấm nút đó để hủy và quay về
        + nếu thành công thì hiện Modal thông báo, bấm ok thì quay về List
        + nếu fail thì hiện Modal thông báo, bấm ok thì tắt Modal
      * Delete:
        + Show Modal -> Confirm Delete
      * Search
        + Nếu Search không ra kết quả thì hiện “No data” thay vì để trống table
* **Huyền**:
  + Style Front-End:
    - MovieManagement
      * List Movie
        + Paging
        + Search No Data
      * Add Movie
      * Update Movie
    - UserManagement
      * List User
        + Paging
        + Search No Data
      * Add User
      * Update User
  + Fix WatchMovie - AutoPlay
* **An**:
  + Authentication: Login with Facebook
* **Giang**:
  + WebClient: User:
    - Home Page
      * <https://youtu.be/AOlkcLtyXkw>
      * Button Update Profile ở Navbar
    - Movie Detail
      * Là 1 Modal trong index
    - Search Movie By Name:
      * 1 trang HTML riêng
      * searchName truyền qua URL Param
      * hiển thị 1 list các kết quả tìm dc như Netflix
  + Admin Dashboard
    - Navbar: Các Link đến:
      * Dashboard
      * User Management
      * Movie Management
    - Thống kê:
      * Số lượng ng dùng active (đăng nhập lần cuối trong 30 vòng 30 ngày qua) trên tổng số ng dùng
        + => Database: Table User: thêm 1 trg last login date
  + Fix Project Structure
  + Add API: Get movies by…
  + Database Seeding
* Tính năng:
  + Authen – authorize
    - Login
      * Login with FB / Google
    - Signup
  + List phim
    - Mở 1 phim để xem
    - Hover -> mở to hơn -> Movie Detail
    - Thêm vào danh sách phát
  + Admin:
    - CRUD phim
    - CRUD account:
      * Create
      * List
      * Delete
  + Search
  + Find by category
  + Display các loại gợi ý:
    - Movie theo các Category
    - …
  + Phim (optional):
    - Chia ra 2 kiểu: phim lẻ, phim bộ

# Done

## 21/06

*Tìm hiểu: Cách gửi video (media) qua API*

* <https://www.c-sharpcorner.com/article/asynchronous-videos-live-streaming-with-asp-net-web-apis-2-0/>
  + Tạo Project ASP.NET Framework – Web
    - Version 4.7.2
    - MVC
  + => Thất bại (Version cũ)
* <https://www.strathweb.com/2013/01/asynchronously-streaming-video-with-asp-net-web-api/>
  + => Thất bại (Version cũ)
* <https://docs.microsoft.com/en-us/answers/questions/726990/serving-video-file-stream-from-asp-net-core-6-mini.html>
  + => Thất bại (Version 6.0)

## 27/06

* Optional:
  + Gửi Link Video -> Client phát Video (Localhost)
  + Gửi link Youtube -> Client embed video youtube
* <https://www.thecodehubs.com/asynchronous-live-video-streaming-web-api-in-net-core-3-1/>
  + => Thành công
  + Sử dụng .NET 5

## 27/6: Họp

* Công nghệ:
  + Backend: ASP.Net Core Web API
  + Front End: AJAX + jQuery (HTML – CSS – JS)

## 29/6: Database Design

* Database:
  + User
    - UserId: int
    - Email: string
    - Username: string
    - Password: string
    - RoleId: int
  + Role
    - RoleId: int
    - RoleName: string
  + Movie
    - MovieId: int
    - MovieName: string
    - VideoPath: string (đường dẫn đến video của phim)
    - Description: string
    - Duration: int (thời lượng phim – tính theo phút)
    - Rated: int (độ tuổi xem)
    - PublishedYear: int
    - Country: string
    - ImagePath: string (đường dẫn đến ảnh poster của phim)
  + Category
    - CategoryId: string
    - CategoryName: name
  + Category-Movie (tự động gen trong DB, ko có trong Models) (mỗi Movie cx có nhiều Category)
    - CategoryId
    - MovieId
  + Actor
    - ActorId: int
    - ActorName: string
  + Actor-Movie (tự động gen trong DB, ko có trong Models)
    - ActorId
    - MovieId

## 30/6: Hỏi thầy

* Cấu trúc Project chuẩn: Theo như Lab 1 nhưng DAO nếu làm static nó lỗi thì có thể làm Singleton
  + => Mình làm kiểu Static
* DTO: ko bắt buộc phải luôn sử dụng, nếu dùng Entity Model cx đủ thì cứ dùng
* **Project nên làm Code-First!**

## Công việc from 27/6:

* **Giang + Huyền**:
  + Database
  + Xem phim
* **Kiên**: Admin:
  + CRUD phim
    - List: Paging (10 phim)
  + CRUD User
    - Add
    - delete
    - list
* **Thông**:
  + List phim:
    - Danh sách các phim (thêm cả danh sách các Category)
    - Paging (5 phim)
    - Khi bấm vào 1 phim: API Move Detail:
      * Các thông tin chi tiết của movie
      * Nút xem phim
* **An**: nghiên cứu làm authen:
  + DB: Code-first
    - User
    - Role
      * Admin
      * Customer
  + Code authen: Login, Signup – API
    - Nếu có thời gian: thêm phần authen = FB/Google

## Chia việc 27/6:

**API trc – Code-First (xem Lab)**

## **Chia việc:**

* **An**:
  + WebAPI: Tiếp tục làm phần Authentication
  + WebClient:
    - Login
    - Sign up
* **Huyền**:
  + Design:
    - Chọn màu Theme: Đỏ + đen
    - Vẽ Logo trang Web (đơn giản như Netflix cx dc):
      * Logo to cho Navbar
      * 1 cái Logo nhỏ để làm Favicon
  + WebClient: User: Page Xem phim
    - 
    - Mở chế độ toàn màn hình
    - Pause
    - Tua
    - Tắt âm
    - Quay lại
  + Style Front-End:
    - Login
    - Sign up
* **Kiên**:
  + WebClient: Admin
    - Quản lý User: CRUD User (AJAX – như đã học) + Paging
      * Hiển thị Role Name thay vì RoleID ở List và Add/Edit
        + Phải sửa API:

Sửa DTO

Sửa Controller: khi lấy dữ liệu thì lấy cả Role rồi map sang DTO

* + - Sau khi add hoặc edit
      * nếu thành công thì hiện alert, bấm ok thì quay về home
      * nếu fail thì hiện alert, bấm ok thì tắt alert
    - Optional
      * Search
        + Nếu Search không ra kết quả thì hiện “No data” thay vì để trống table
      * Hiển thị Bootstrap Modal thay cho alert
  + WebAPI: Client
    - Actor:
      * Get all Actors
      * Get Actors by MovieID
    - Category:
      * Get Categories by MovieID
  + [API] Role:
    - Get List Role
    - Get Role by ID