****

**BÁO CÁO --- MÔN SOF304**

**HỆ THỐNG ONLINE ENTERTAINMENT**

**ASSIGNMENT**

**GVHD:** thầy Lê Anh Tú

**SVTH :** Nguyễn Minh Hoàng

**MSSV :** PS26822

**TPHCM - 2023**

Contents

[1. PHÂN TÍCH 4](#_Toc152956125)

[1.1. HIỆN TRẠNG 4](#_Toc152956126)

[1.2. YÊU CẦU CỦA KHÁCH HÀNG 4](#_Toc152956127)

[1.2.1. PHẦN 1: DÀNH CHO KHÁCH HÀNG 4](#_Toc152956128)

[1.2.2. PHẦN 2: DÀNH CHO NHÂN VIÊN QUẢN TRỊ: 5](#_Toc152956129)

[1.3. USE CASE DIAGRAM 5](#_Toc152956130)

[2. THIẾT KẾ 6](#_Toc152956131)

[2.1. THIẾT KẾ CƠ SỞ DỮ LIỆU 6](#_Toc152956132)

[2.1.1. Sơ đồ ERD 6](#_Toc152956133)

[2.1.2. Database Diagram 6](#_Toc152956134)

[2.1.3. Thiết kế chi tiết các thực thể 7](#_Toc152956135)

[2.2. THIẾT KẾ GIAO DIỆN NGƯỜI DÙNG 9](#_Toc152956136)

[2.2.1. Layout khách hàng 9](#_Toc152956137)

[2.2.2. Trang Video 10](#_Toc152956138)

[2.2.3. Trang chi tiết 11](#_Toc152956139)

[2.2.4. Trang chia sẻ 12](#_Toc152956140)

[2.2.5. Trang yêu thích 12](#_Toc152956141)

[2.2.6. Trang đăng nhập 13](#_Toc152956142)

[2.2.7. Đăng ký 13](#_Toc152956143)

[2.2.8. Quên mật khẩu 14](#_Toc152956144)

[2.2.9. Đổi mật khẩu 14](#_Toc152956145)

[2.2.10. Câp nhật tài khoản 15](#_Toc152956146)

[2.3. THIẾT KẾ GIAO DIỆN AMIN 15](#_Toc152956147)

[2.3.1. Layout Admin 15](#_Toc152956148)

[2.3.2. Quản lý Video 16](#_Toc152956149)

[2.3.3. Quản lý User 17](#_Toc152956150)

[2.3.4. Thống kê số người yêu thích từng video 19](#_Toc152956151)

[2.3.5. Lọc người yêu thích từng video 19](#_Toc152956152)

[2.3.6. Lọc người gửi và người nhận từng video được share 20](#_Toc152956153)

[3. KIỂM THỬ UNIT TEST 20](#_Toc152956154)

[3.1. TEST LOGIN 20](#_Toc152956155)

[3.1.1. Test Case 20](#_Toc152956156)

[3.1.2. Junit Test Code 22](#_Toc152956157)

[3.1.3. Test Result 25](#_Toc152956158)

[3.2. TEST CREATE USER 25](#_Toc152956159)

[3.2.1. Test Case 25](#_Toc152956160)

[3.2.2. Junit Test Code 29](#_Toc152956161)

[3.2.3. Test Result 32](#_Toc152956162)

[3.3. TEST UPDATE USER 32](#_Toc152956163)

[3.3.1. Test Case 32](#_Toc152956164)

[3.3.2. Junit Test Code 37](#_Toc152956165)

[3.3.3. Test Result 43](#_Toc152956166)

[3.4. TEST DELETE USER 43](#_Toc152956167)

[3.4.1. Test Case 43](#_Toc152956168)

[3.4.2. Junit Test Code 44](#_Toc152956169)

[3.4.3. Test Result 46](#_Toc152956170)

[4. KIỂM THỬ AUTOMATION TEST 46](#_Toc152956171)

[4.1. TEST LOGIN 46](#_Toc152956172)

[4.1.1. Test Data 46](#_Toc152956173)

[4.1.2. TestNG Automation Test Code 46](#_Toc152956174)

[4.1.3. Test Result 49](#_Toc152956175)

[4.2. TEST CREATE VIDEO 49](#_Toc152956176)

[4.2.1. Test Data 49](#_Toc152956177)

[4.2.2. TestNG Automation Test Code 49](#_Toc152956178)

[4.2.3. Test Result 52](#_Toc152956179)

[4.3. TEST UPDATE VIDEO 52](#_Toc152956180)

[4.3.1. Test Data 52](#_Toc152956181)

[4.3.2. TestNG Automation Test Code 52](#_Toc152956182)

[4.3.3. Test Result 55](#_Toc152956183)

[4.4. TEST DELETE VIDEO 55](#_Toc152956184)

[4.4.1. Test Data 55](#_Toc152956185)

[4.4.2. TestNG Automation Test code 55](#_Toc152956186)

[4.4.3. Test Result 58](#_Toc152956187)

[5. Tổng kết 58](#_Toc152956188)

[5.1. KHÓ KHĂN 58](#_Toc152956189)

[5.2. THUẬN LỢI 58](#_Toc152956190)

# PHÂN TÍCH

## HIỆN TRẠNG

Công ty OE chuyên cung cấp dịch vụ giải trí cho các doanh nghiệp. Khi một doanh nghiệp có sự kiện họ tìm đến OE để đặt các tiểu phẩm hài phục vụ quan khách trong sự kiện. OE mong muốn có một website giới thiệu các trích đoạn tiểu phẩm (video) của mình nhằm phục vụ miễn phí cho cộng đồng, đồng thời qua đó tìm kiếm khách hàng.

## YÊU CẦU CỦA KHÁCH HÀNG

Các chức năng thiết yếu mà OE mong muốn gồm 2 phần, 1 dành cho người sử dụng và phần còn lại là cho nhân viên quản trị của công ty:

### PHẦN 1: DÀNH CHO KHÁCH HÀNG

#### Trang chủ

* Hiển thị 6 tiểu phẩm cùng với các nút điều hướng (đầu, cuối, trước, sau) để xem thêm. Các tiểu phẩm được sắp xếp giảm dần theo số lượt xem.
* Thông tin hiển thị cho mỗi tiểu phẩm gồm:
  + Poster (hình đại diện)
  + Title (tên tiêu đề)
  + 2 nút tương tác Like và Share
* Xử lý tương tác
  + [Poster.Click]: Chuyển sang trang chi tiết (mô tả bên dưới) đồng thời ghi nhận tiểu phẩm này trở thành tiểu phẩm đã xem của người sử dụng đó.
  + [Like.Click]: Ghi nhận tiểu phẩm trở thành yêu thích của người xem
  + [Share.Click]: Gửi liên kết của tiểu phẩm đến người thân, bạn bè thông qua email. Từ email bạn bè có thể click vào liên kết đó để xem trích đoạn tiểu phẩm.

#### Trang chi tiết

Hiển thị thông tin chi tiết của tiểu phẩm và các tiểu phẩm mà người dùng đã xem (để người dùng có thể xem lại). Thông tin chi tiết của mỗi tiểu phẩm gồm:

* Video (được lấy từ youtube)
* Title
* Description (mô tả tiểu phẩm)
* 2 nút Like và Share

#### Trang yêu thích của khách hàng

Hiển thị tất cả các tiểu phẩm yêu thích của người sử dụng này. Mỗi tiểu phẩm trình bày tương tự trên trang chủ tuy nhiên thay nút Like bằng nút Unlike để khi người sử dụng nhấp vào nút này thì sẽ gỡ tiểu phẩm khỏi danh sách yêu thích của họ.

#### Yêu cầu bảo mật

* Phải đăng nhập trước khi thực hiện các tương tác như Like, Share, Unlike và xem tiểu phẩm yêu thích của mình.
* Để có tài khoản đăng nhập, người sử dụng phải đăng ký trước. Website cũng cung cấp các trang web để thực hiện các chức năng liên quan đến tài khoản như: đổi mật khẩu, lấy lại mật khẩu đã quên, cậpnhật thông tin tài khoản và đăng xuất.
  + Sau khi đăng ký thành công cần gửi email chào mừng.
  + Sau khi tìm thấy mật khẩu cần gửi email qua email đã đăng ký để đảm bảo an toàn.

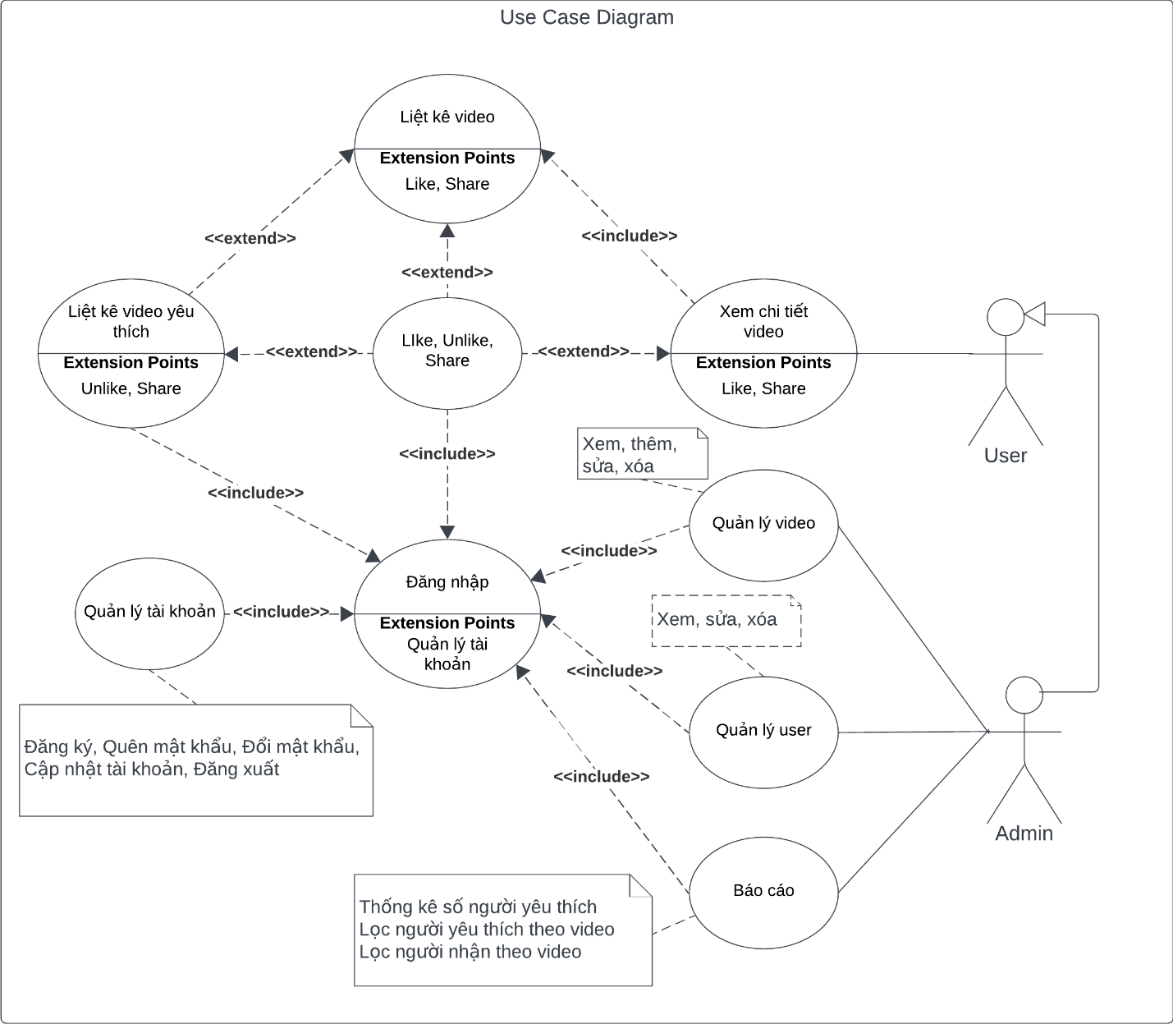
### PHẦN 2: DÀNH CHO NHÂN VIÊN QUẢN TRỊ:

Nhân viên quản trị cũng hoạt động như khách hàng, ngoài ra có quyền thực hiện các chức năng sau:

* Quản lý tiểu phẩm: Thực hiện các công việc xem, thêm, sửa và xóa các tiểu phẩm.
* Quản lý khách hàng: Được phép xem cập nhật và xóa khách hàng.
* Báo cáo – Thống kê
  + Thống kê số người yêu thích của từng tiểu phẩm
  + Lọc người yêu thích theo tiểu phẩm
  + Lọc người nhận theo tiểu phẩm đã gửi

## USE CASE DIAGRAM

Căn cứ vào yêu cầu của khách hàng, chúng ta phác thảo sơ đồ use case (mô tả các tình huống sử dụng) của hệ thống website như sau:

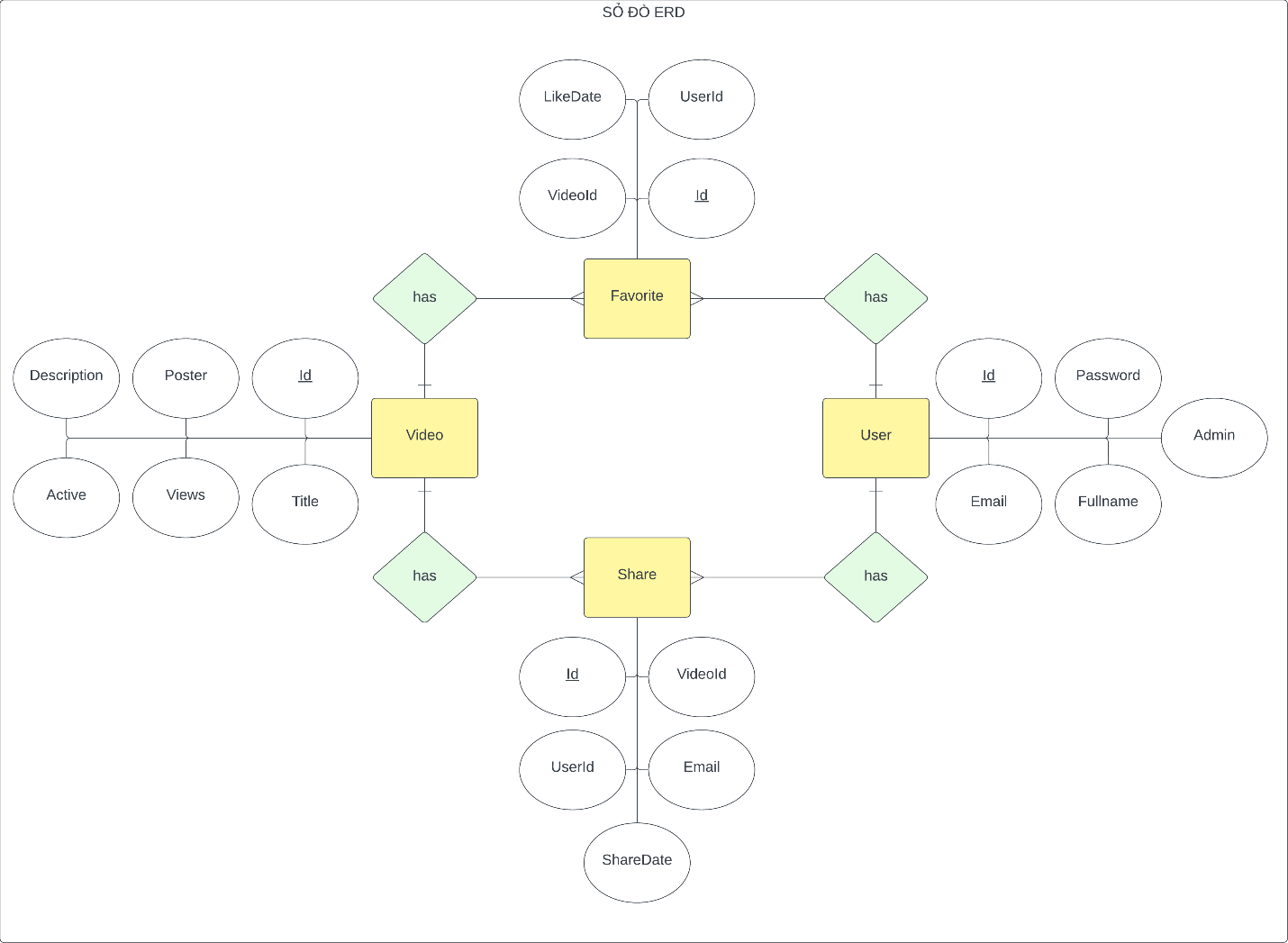


# THIẾT KẾ

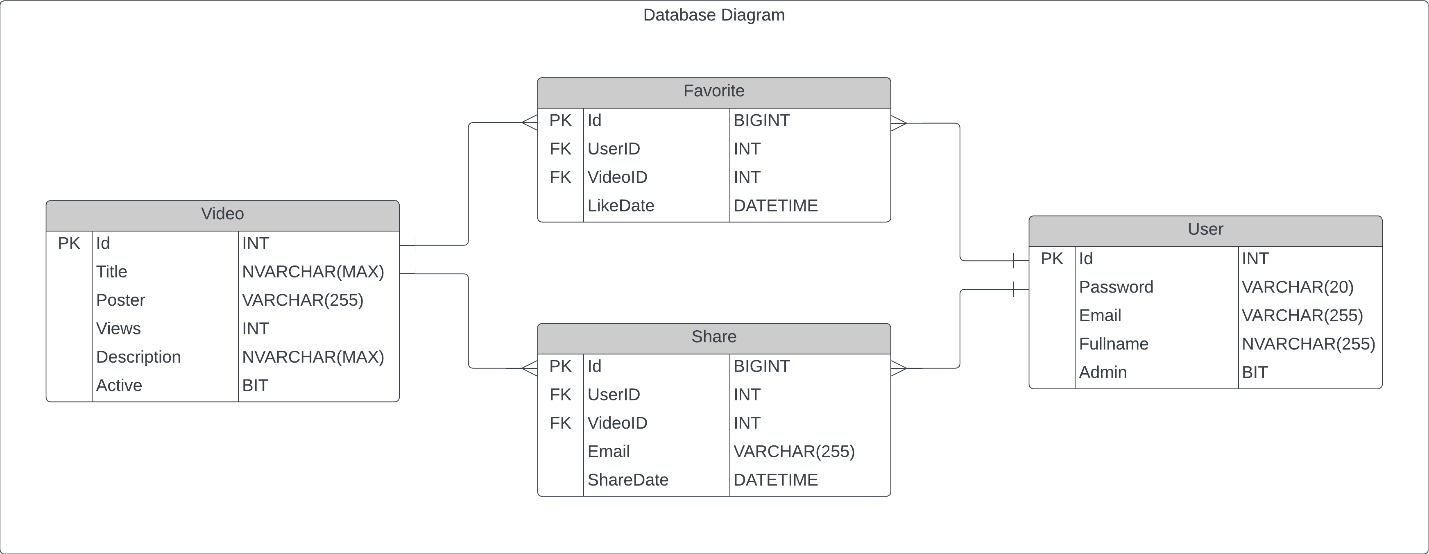
## THIẾT KẾ CƠ SỞ DỮ LIỆU

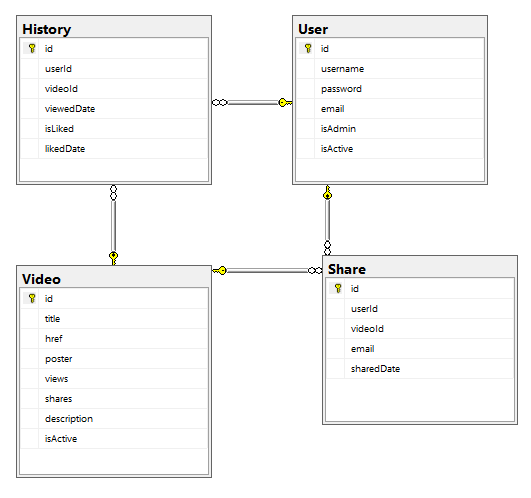
### Sơ đồ ERD

Phân tích ở góc độ dữ liệu phục vụ cho các hoạt động nghiệp vụ được mô tả từ yêu cầu khách hàng, chúng ta phác thảo sơ đồ ERD (mô tả mối liên hệ giữa các thực thể) như sau:



### Database Diagram





### Thiết kế chi tiết các thực thể

#### User

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu** | **Ràng buộc** | **Ghi chú** |
| Id | INT | PK, NOT NULL | Mã Id User |
| Password | VARCHAR(20) | NOT NULL | Mật khẩu đăng nhập |
| Email | VARCHAR(255) | NOT NULL | Email |
| Fullname | NVARCHAR(255) | NOT NULL | Họ và tên |
| Admin | BIT | DEFAULT 0 | Vai trò: - 0 là User - 1 là Admin |

#### Video

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu** | **Ràng buộc** | **Ghi chú** |
| Id | INT | PK, NOT NULL | Mã Id Video |
| Title | NVARCHAR(MAX) | NOT NULL | Tiêu đề video |
| Poster | VARCHAR(255) | NULL | Hình poster video |
| Views | INT | DEFAULT 0 | Số lượt xem |
| Decsription | NVARCHAR(MAX) | NULL | Mô tả video |
| Active | BIT | DEFAULT 1 | Trạng thái: - 0: ngừng hoạt động - 1: hoạt động |

#### Favorite

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu** | **Ràng buộc** | **Ghi chú** |
| Id | BIGINT | PK, NOT NULL | Mã Id |
| UserId | INT | FK, NOT NULL | Mã Id của User |
| VideoId | INT | FK, NOT NULL | Mã Id của Video |
| LikeDate | DATETIME | NULL | Ngày được nhấn LIke |

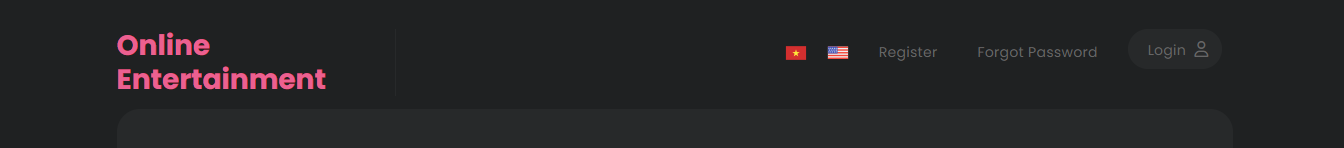
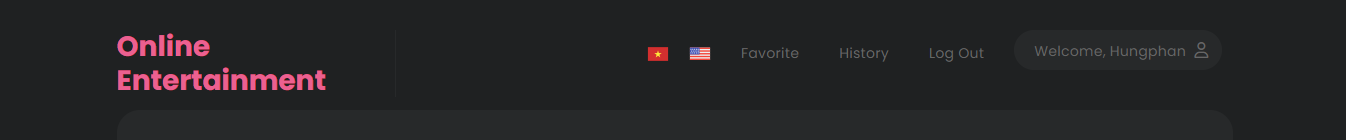
#### Share

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu** | **Ràng buộc** | **Ghi chú** |
| Id | BIGINT | PK, NOT NULL | Mã Id |
| UserId | INT | FK, NOT NULL | Mã Id của User |
| VideoId | INT | FK, NOT NULL | Mã Id của Video |
| Email | VARCHAR(255) | NOT NULL | Email người nhận |
| ShareDate | DATETIME | NULL | Ngày video được Share |

## THIẾT KẾ GIAO DIỆN NGƯỜI DÙNG

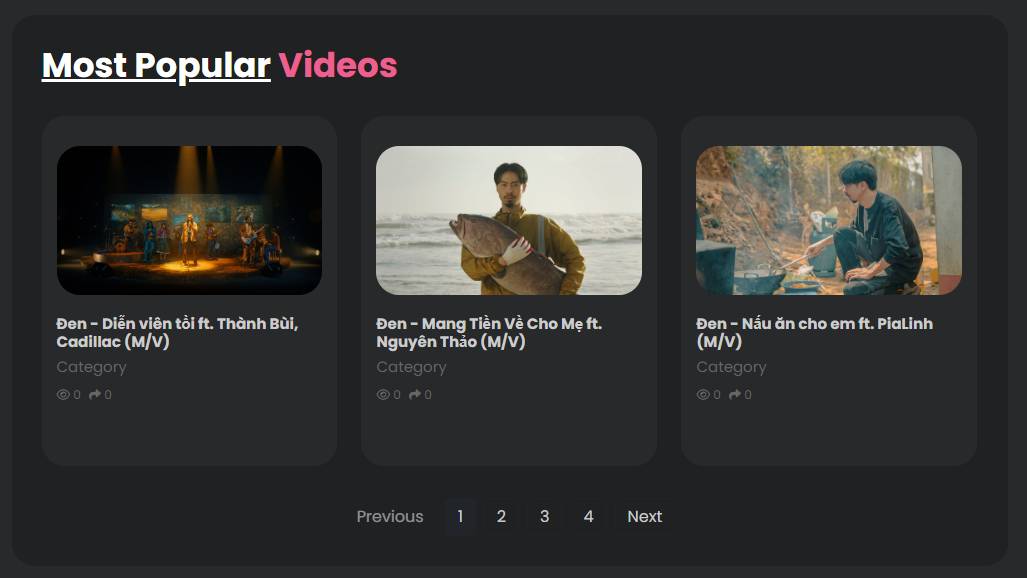
Căn cứ vào yêu cầu của khách hàng, các sơ đồ Use Case và ERD chúng ta phác thảo giao diện các chức năng phục vụ cho người sử dụng như sau:

### Layout khách hàng

Tất cả các trang web phục vụ người dùng phải chứa thanh menu sau đây

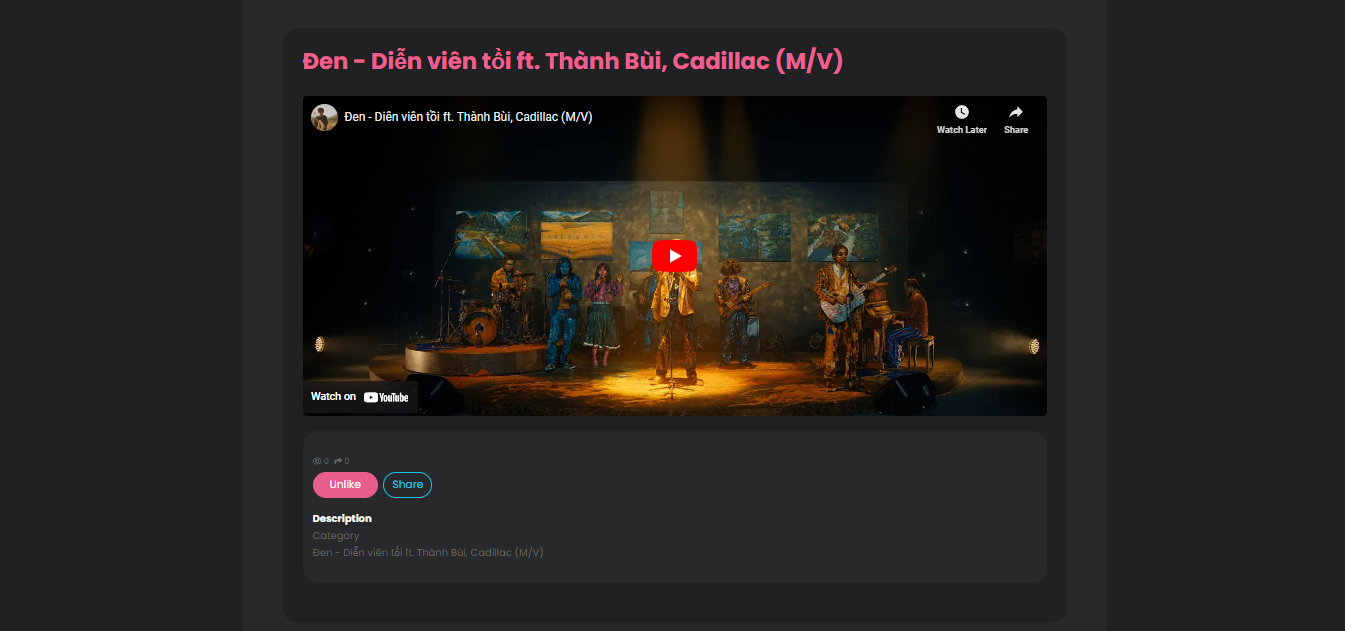
|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| [Online Entertainment].Click | Trang chủ |  |
| [My Favorites].Click | Trang chứa các video yêu thích | Login |
| [Login].Click | Trang đăng nhập |  |
| [Forgot Password].Click | Trang đổi mật khẩu | Login |
| [Registration].Click | Trang đăng ký |  |
| [Logoff].Click | Đăng xuất, về trang chủ | Login |
| [Change Password].Click | Trang đổi mật khẩu | Login |
| [Profile].Click | Trang cập nhật tài khoản | Login |

### Trang Video



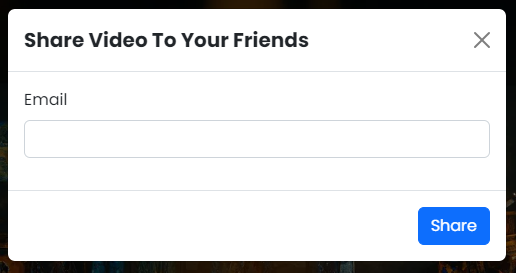
|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| Khởi đầu | Hiển thị tiểu phẩm, giảm dần theo số lượt xem |  |
| [Poster].Click | • Trang chi tiết • Ghi nhận tiểu phẩm đã xem bằng cookie • Tăng số lược xem của tiểu phẩm |  |
| [1] [2] [3] | Số trang |  |
| [Previous] | Trang trước |  |
| [Next] | Trang sau |  |

### Trang chi tiết



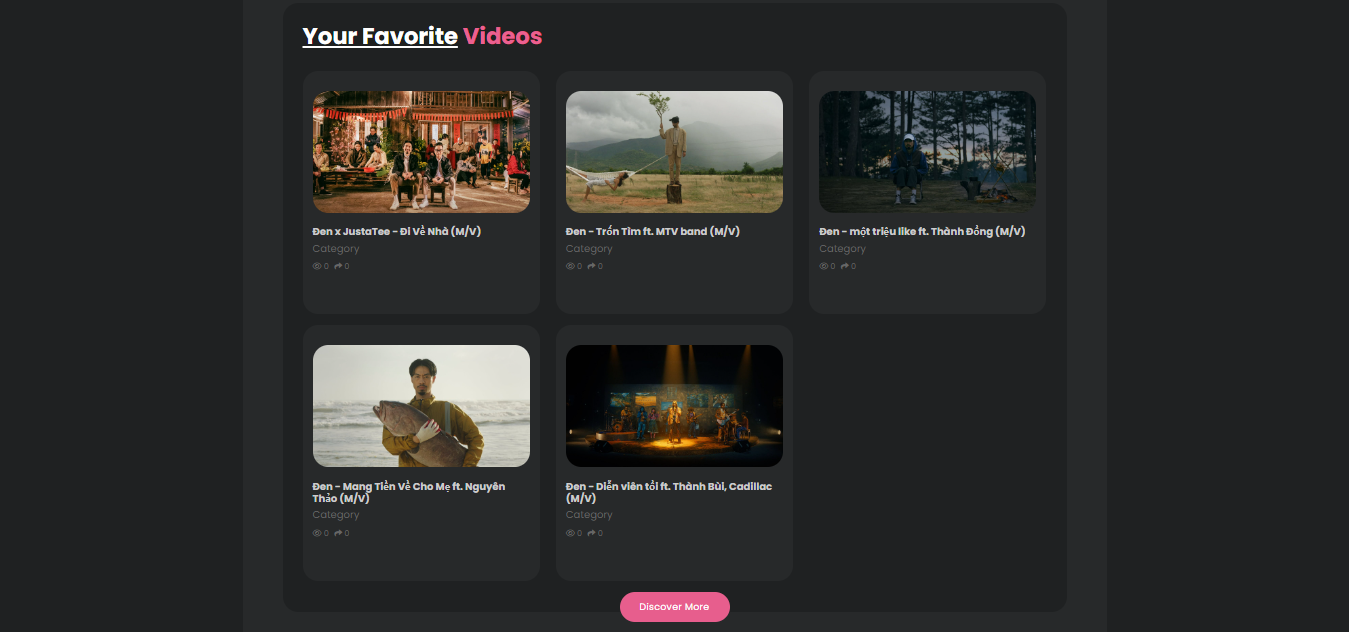
|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| Khởi đầu | Hiển thị thông tin chi tiết (video, title, description) Hiển thị các tiểu phẩm đã xem (lấy từ cookie) |  |
| [Poster].Click [Title].Click | Trang chi tiết |  |
| [Like].Click | Ghi nhận tiểu phẩm yêu thích của người sử dụng | Login |
| [Share].Click | Trang chia sẻ tiểu phẩm | Login |

### Trang chia sẻ



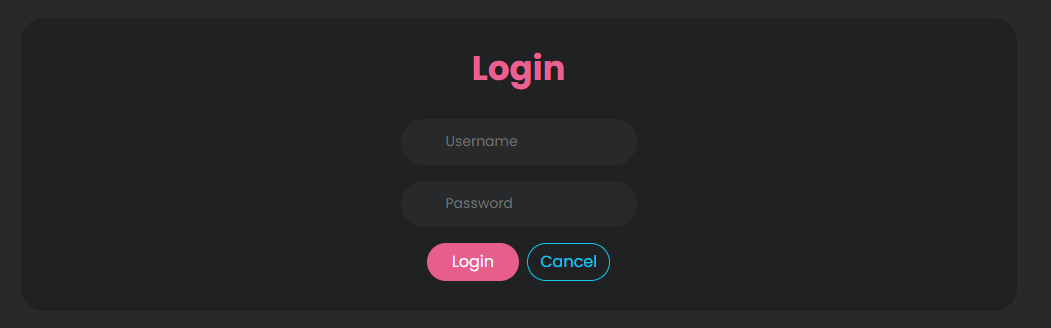
|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| [Send].Click | Gửi link của tiểu phẩm được chọn đến các email nhập vào (các email cách nhau dấu phẩy hoặc chấm phẩy) | Login |

### Trang yêu thích



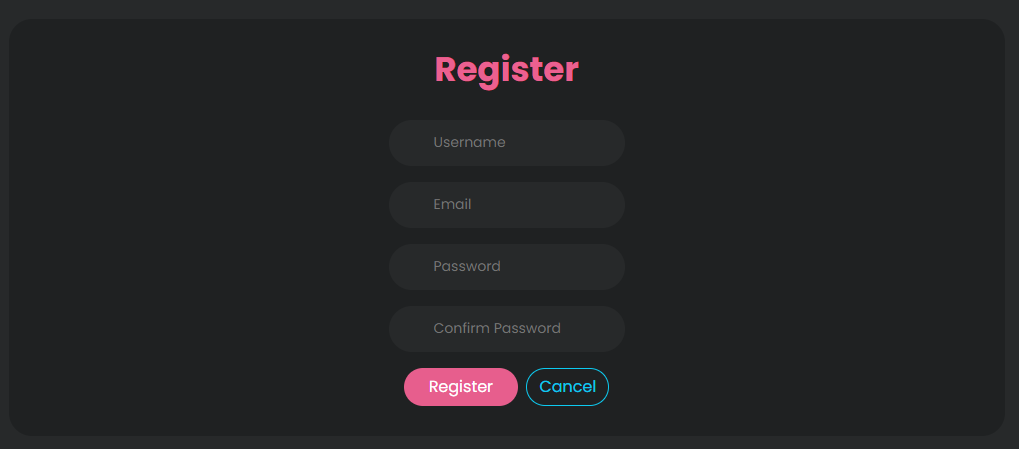
|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| Khởi đầu | Hiển thị tất cả các tiểu phẩm yêu thích của người sử dụng | Login |
| [Poster].Click | Trang chi tiết |  |

### Trang đăng nhập



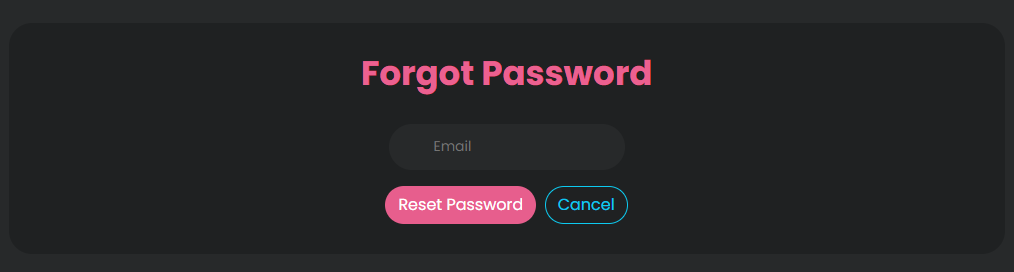
|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| Khởi đầu | Hiển thị username và password đã ghi nhớ lên form |  |
| [Send].Click | • Kiểm tra dữ liệu form • Kiểm tra sự tồn tại của user • Duy trì thông tin user vào session • Ghi nhớ tài khoản (bằng cookie) nếu có check Remember me • Xóa ghi nhớ tài khoản nếu không check Remember me • Trở về trang được yêu cầu bảo mật trước đó (nếu có) |  |

### Đăng ký



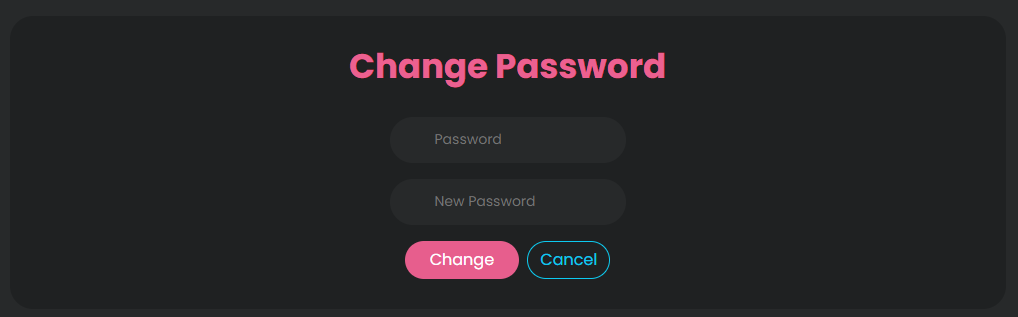
|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| [Sign Up].Click | • Kiểm tra dữ liệu form • Thêm mới một người sử dụng • Gửi email chào |  |

### Quên mật khẩu



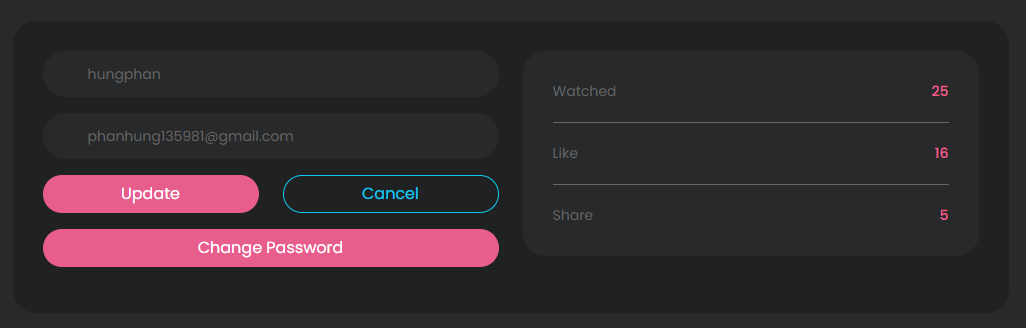
|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| [Reset Password].Click | • Kiểm tra dữ liệu nhập vào form • Gửi mật khẩu qua email nếu thông tin chính xác |  |

### Đổi mật khẩu



|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| [Change].Click | • Kiểm tra dữ liệu nhập vào form • Cập nhật mật khẩu • Cập nhật lại thông tin user trong session | Login |

### Câp nhật tài khoản



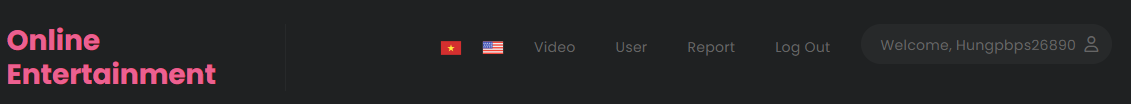
|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| Khởi đầu | Hiển thị thông tin user lấy từ session lên form | Login |
| [Update].Click | • Kiểm tra dữ liệu nhập vào form • Cập nhật thông tin tài khoản • Cập nhật lại thông tin user trong session | Login |

## THIẾT KẾ GIAO DIỆN AMIN

Căn cứ vào yêu cầu của khách hàng, các sơ đồ Use Case và ERD chúng ta phác thảo giao diện các chức năng phục vụ cho nhân viên quản trị như sau:

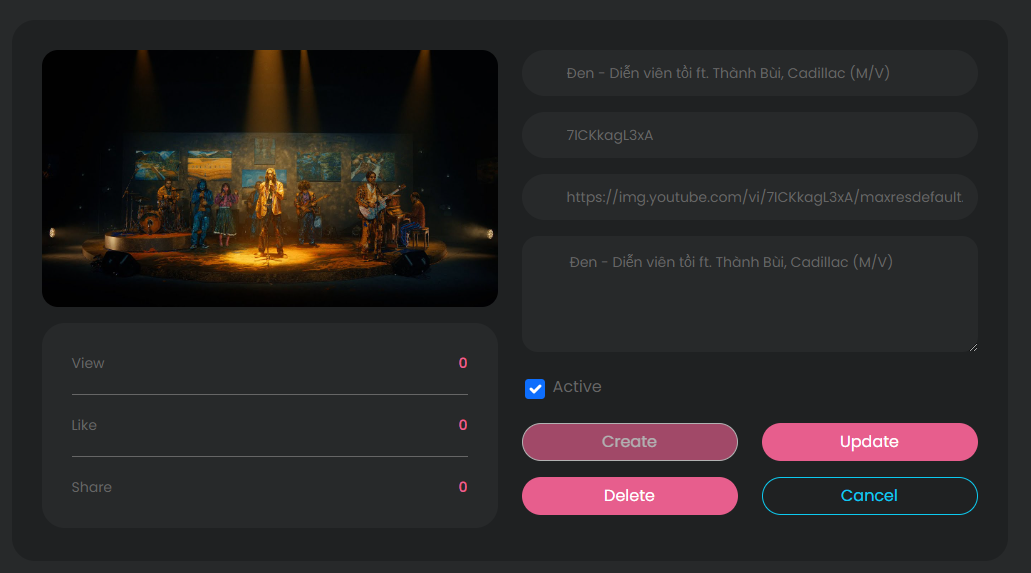
### Layout Admin

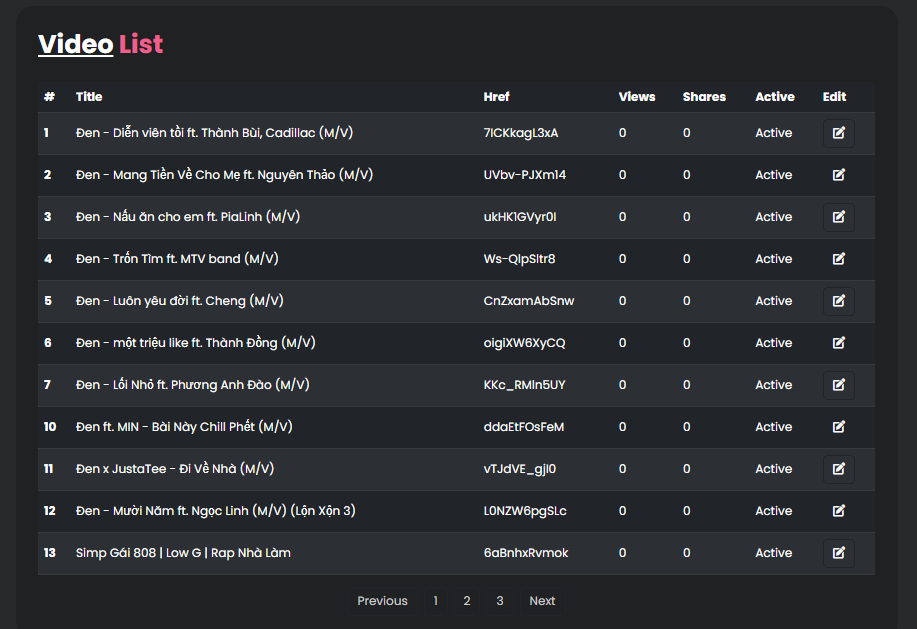
Tất cả các trang dành cho nhân viên đều phải được đăng nhập với vai trò là admin và phải bao gồm menu sau đây



|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| [Home].Click | Trang chủ |  |
| [Videos].Click | Trang quản lý tiểu phẩm | Admin |
| [Users].Click | Trang quản lý người sử dụng | Admin |
| [Reports].Click | Trang báo cáo, thống kê | Admin |

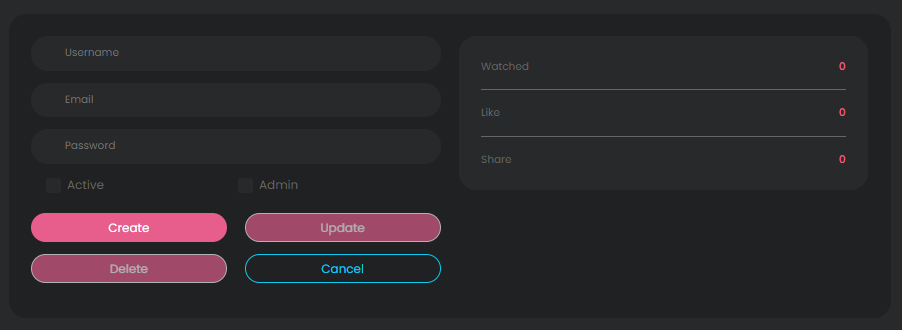
### Quản lý Video

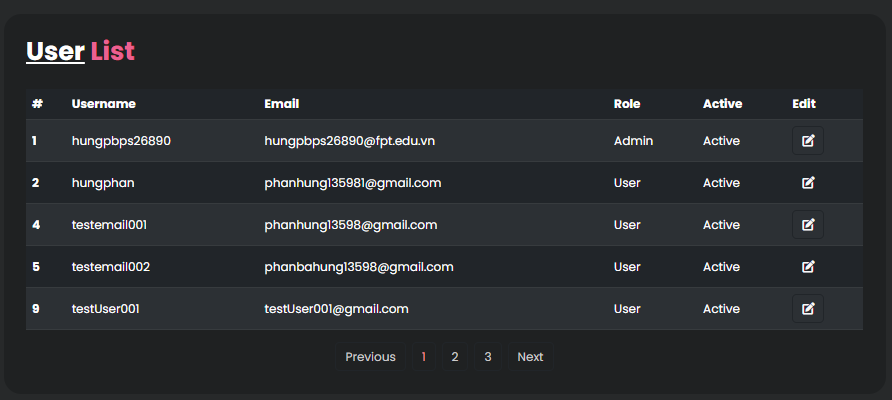




|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| Khởi đầu, [Reset].Click | • Hiển thị form trống • Hiển thị 10 tiểu phẩm • Vô hiệu hóa [Update] và [Delete] | Admin |
| [Edit].Click | • Hiển thị tiểu phẩm được chọn lên form • Vô hiệu hóa [Create] | Admin |
| [Create].Click | • Kiểm tra dữ liệu form • Thêm mới tiểu phẩm | Admin |
| [Update].Click | • Kiểm tra dữ liệu form • Cập nhật thông tin tiểu phẩm • Chuyển về trạng thái [Edit] | Admin |
| [Delete].Click | • Xóa tiểu phẩm đang xem • Chuyển về trạng thái [khởi đầu] | Admin |
| [|<,1, 2, 3,>|].Click | • Chuyển đến trang phù hợp • Chuyển về trạng thái [khởi đầu] | Admin |

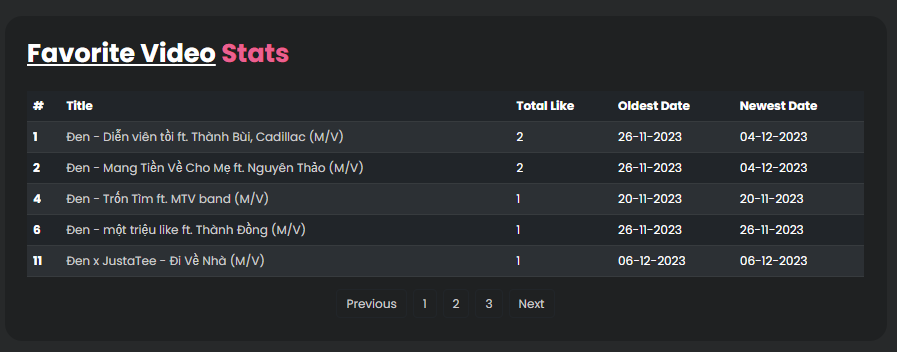
### Quản lý User





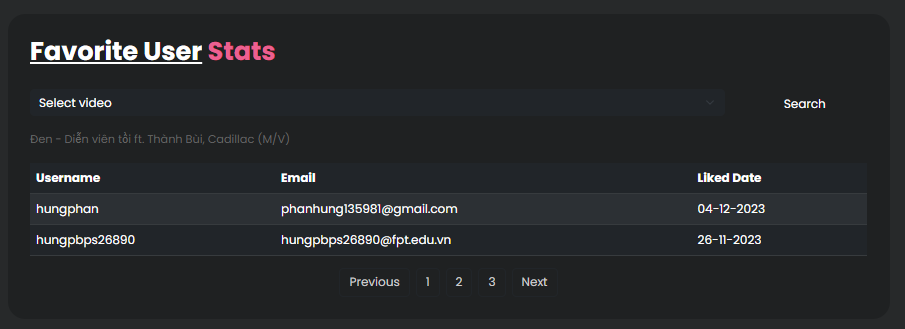
|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| Khởi đầu | • Hiển thị form trống • Hiển thị 10 người sử dụng • Vô hiệu hóa [Update] và [Delete] | Admin |
| [Edit].Click | • Hiển thị người sử dụng được chọn lên form | Admin |
| [Update].Click | • Kiểm tra dữ liệu form • Cập nhật thông tin người sử dụng • Chuyển về trạng thái [Edit] | Admin |
| [Delete].Click | • Xóa người sử dụng đang xem | Admin |
| [|<,1, 2, 3,>|].Click | • Chuyển đến trang phù hợp • Chuyển về trạng thái [khởi đầu] | Admin |

### Thống kê số người yêu thích từng video



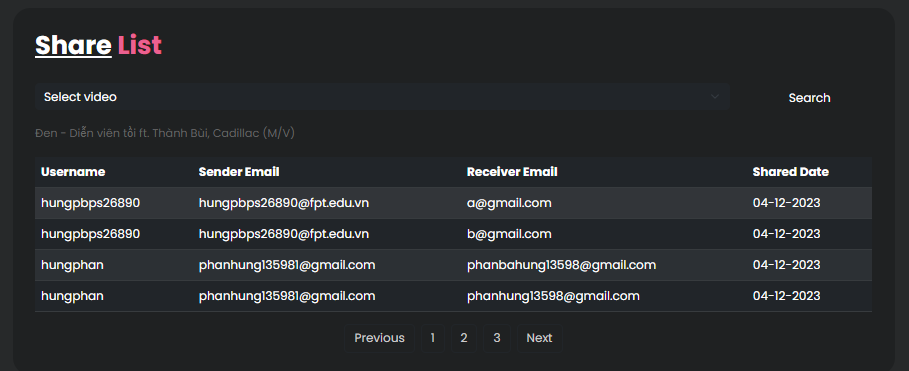
|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| Khởi đầu | • Tổng hợp dữ liệu từ Favorite • Hiển thị thông tin tổng hợp lên lưới | Admin |

### Lọc người yêu thích từng video



|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| Khởi đầu | • Đỗ các video có yêu thích vào select box | Admin |
| [Video Title].Select | • Lọc các Favorite theo video được chọn • Hiển thị các user của các Favorite đã lọc lên bảng | Admin |

### Lọc người gửi và người nhận từng video được share



|  |  |  |
| --- | --- | --- |
| **SỰ KIỆN** | **HÀNH ĐỘNG** | **BẢO MẬT** |
| Khởi đầu | • Đỗ các video có chia sẻ vào select box | Admin |
| [Video Title].Select | • Lọc các Share theo video được chọn • Hiển thị các user và share đã lọc lên bảng | Admin |

# KIỂM THỬ UNIT TEST

## TEST LOGIN

### Test Case

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **TEST CASE** | **TEST STEP** | **TEST DATA** | **EXPECTED** | **ACTUAL** | **STATUS** |
| testLogin001 | Kiểm thử phương thức login() | 1. Nhập username và password 2. Gọi login(username, password) 3. Chạy Junit Test | String username = "hungphan";  String password = "123456"; | Đăng nhập thành công | Đăng nhập thành công | PASS |
| testLogin002 | Kiểm thử phương thức login() | 1. Nhập username và password 2. Gọi login(username, password) 3. Chạy Junit Test | String username = "";  String password = ""; | Đăng nhập thất bại | Đăng nhập thất bại | PASS |
| testLogin003 | Kiểm thử phương thức login() | 1. Nhập username và password 2. Gọi login(username, password) 3. Chạy Junit Test | String username = "hungphan";  String password = ""; | Đăng nhập thất bại | Đăng nhập thất bại | PASS |
| testLogin004 | Kiểm thử phương thức login() | 1. Nhập username và password 2. Gọi login(username, password) 3. Chạy Junit Test | String username = "";  String password = "123456"; | Đăng nhập thất bại | Đăng nhập thất bại | PASS |
| testLogin005 | Kiểm thử phương thức login() | 1. Nhập username và password 2. Gọi login(username, password) 3. Chạy Junit Test | String username = "hungphan";  String password = "123"; | Đăng nhập thất bại | Đăng nhập thất bại | PASS |
| testLogin006 | Kiểm thử phương thức login() | 1. Nhập username và password 2. Gọi login(username, password) 3. Chạy Junit Test | String username = "hung";  String password = "123456"; | Đăng nhập thất bại | Đăng nhập thất bại | PASS |

### Junit Test Code

**public** **class** LoginTestTest {

**private** UserService userService;

@Before

**public** **void** beforeTest() {

userService = **new** UserServiceImpl();

System.***out***.println("Before Test");

}

@After

**public** **void** afterTest() {

System.***out***.println("After Test");

System.***out***.println("--------------------------------------");

}

@Test

**public** **void** testLogin001() {

User user = **new** User();

String username = "hungphan";

String password = "123456";

String expected = "Đăng nhập thành công";

String actual = "";

user = userService.login(username, password);

**if** (user != **null**) {

actual = "Đăng nhập thành công";

} **else** {

actual = "Đăng nhập thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testLogin002() {

User user = **new** User();

String username = "";

String password = "";

String expected = "Đăng nhập thất bại";

String actual = "";

user = userService.login(username, password);

**if** (user != **null**) {

actual = "Đăng nhập thành công";

} **else** {

actual = "Đăng nhập thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testLogin003() {

User user = **new** User();

String username = "hungphan";

String password = "";

String expected = "Đăng nhập thất bại";

String actual = "";

user = userService.login(username, password);

**if** (user != **null**) {

actual = "Đăng nhập thành công";

} **else** {

actual = "Đăng nhập thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testLogin004() {

User user = **new** User();

String username = "";

String password = "123456";

String expected = "Đăng nhập thất bại";

String actual = "";

user = userService.login(username, password);

**if** (user != **null**) {

actual = "Đăng nhập thành công";

} **else** {

actual = "Đăng nhập thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testLogin005() {

User user = **new** User();

String username = "hungphan";

String password = "123";

String expected = "Đăng nhập thất bại";

String actual = "";

user = userService.login(username, password);

**if** (user != **null**) {

actual = "Đăng nhập thành công";

} **else** {

actual = "Đăng nhập thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testLogin006() {

User user = **new** User();

String username = "hung";

String password = "123456";

String expected = "Đăng nhập thất bại";

String actual = "";

user = userService.login(username, password);

**if** (user != **null**) {

actual = "Đăng nhập thành công";

} **else** {

actual = "Đăng nhập thất bại";

}

System.***out***.println("Expected: " + expected);

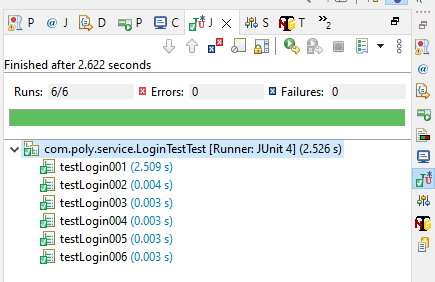
System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

}

### Test Result



## TEST CREATE USER

### Test Case

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **TEST CASE** | **TEST STEP** | **TEST DATA** | **EXPECTED** | **ACTUAL** | **STATUS** |
| testCreateUser001 | Kiểm thử phương thức create() đối tượng User | 1. Nhập username và password 2. Gọi create(username, password, email) 3. Chạy Junit Test | String username = "hungphan001";  String password = "123456";  String email = "hungphan001@gmail.com"; | Tạo mới thành công | Tạo mới thành công | PASS |
| testCreateUser002 | Kiểm thử phương thức create() đối tượng User | 1. Nhập username và password 2. Gọi create(username, password, email) 3. Chạy Junit Test | String username = "";  String password = "";  String email = ""; | Tạo mới thất bại | Tạo mới thành công | FAIL |
| testCreateUser003 | Kiểm thử phương thức create() đối tượng User | 1. Nhập username và password 2. Gọi create(username, password, email) 3. Chạy Junit Test | String username = "hungphan001";  String password = "";  String email = ""; | Tạo mới thất bại | Tạo mới thất bại | PASS |
| testCreateUser004 | Kiểm thử phương thức create() đối tượng User | 1. Nhập username và password 2. Gọi create(username, password, email) 3. Chạy Junit Test | String username = "";  String password = "123456";  String email = ""; | Tạo mới thất bại | Tạo mới thất bại | PASS |
| testCreateUser005 | Kiểm thử phương thức create() đối tượng User | 1. Nhập username và password 2. Gọi create(username, password, email) 3. Chạy Junit Test | String username = "";  String password = "";  String email = "hungphan001@gmail.com"; | Tạo mới thất bại | Tạo mới thất bại | PASS |
| testCreateUser006 | Kiểm thử phương thức create() đối tượng User | 1. Nhập username và password 2. Gọi create(username, password, email) 3. Chạy Junit Test | String username = "hungphan001";  String password = "123456";  String email = "hungphan001@gmail.com"; | Tạo mới thất bại | Tạo mới thất bại | PASS |

### Junit Test Code

**public** **class** CreateUserTest {

**private** UserService userService;

@Before

**public** **void** beforeTest() {

userService = **new** UserServiceImpl();

System.***out***.println("Before Test");

}

@After

**public** **void** afterTest() {

System.***out***.println("After Test");

System.***out***.println("--------------------------------------");

}

@Test

**public** **void** testCreateUser001() {

User user = **new** User();

String username = "hungphan001";

String password = "123456";

String email = "hungphan001@gmail.com";

String expected = "Tạo mới thành công";

String actual = "";

**try** {

user = userService.create(username, password, email);

**if** (user != **null**) {

actual = "Tạo mới thành công";

} **else** {

**throw** **new** Exception();

}

} **catch** (Exception e) {

actual = "Tạo mới thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testCreateUser002() {

User user = **new** User();

String username = "";

String password = "";

String email = "";

String expected = "Tạo mới thất bại";

String actual = "";

**try** {

user = userService.create(username, password, email);

**if** (user != **null**) {

actual = "Tạo mới thành công";

} **else** {

**throw** **new** Exception();

}

} **catch** (Exception e) {

actual = "Tạo mới thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testCreateUser003() **throws** Exception {

User user = **new** User();

String username = "hungphan001";

String password = "";

String email = "";

String expected = "Tạo mới thất bại";

String actual = "";

**try** {

user = userService.create(username, password, email);

**if** (user != **null**) {

actual = "Tạo mới thành công";

}

} **catch** (Exception e) {

actual = "Tạo mới thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testCreateUser004() **throws** Exception {

User user = **new** User();

String username = "";

String password = "123456";

String email = "";

String expected = "Tạo mới thất bại";

String actual = "";

**try** {

user = userService.create(username, password, email);

**if** (user != **null**) {

actual = "Tạo mới thành công";

}

} **catch** (Exception e) {

actual = "Tạo mới thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testCreateUser005() **throws** Exception {

User user = **new** User();

String username = "";

String password = "";

String email = "hungphan001@gmail.com";

String expected = "Tạo mới thất bại";

String actual = "";

**try** {

user = userService.create(username, password, email);

**if** (user != **null**) {

actual = "Tạo mới thành công";

}

} **catch** (Exception e) {

actual = "Tạo mới thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testCreateUser006() **throws** Exception {

User user = **new** User();

String username = "hungphan001";

String password = "123456";

String email = "hungphan001@gmail.com";

String expected = "Tạo mới thất bại";

String actual = "";

**try** {

user = userService.create(username, password, email);

**if** (user != **null**) {

actual = "Tạo mới thành công";

}

} **catch** (Exception e) {

actual = "Tạo mới thất bại";

}

System.***out***.println("Expected: " + expected);

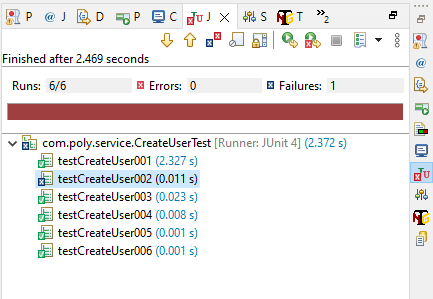
System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

}

### Test Result



## TEST UPDATE USER

### Test Case

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **TEST CASE** | **TEST STEP** | **TEST DATA** | **EXPECTED** | **ACTUAL** | **STATUS** |
| testUpdateUser001 | Kiểm thử phương thức update() đối tượng User | 1. Nhập username, password, email, isActive, isAdmin 2. Gọi findByUsername(username) 3. Gọi update(user) 4. Chạy Junit Test | String username = "hungphan001";  String password = "123456789";  String email = "hungphan001@gmail.com";  Boolean isAdmin = true;  Boolean isActive = true; | Cập nhật thành công | Cập nhật thành công | PASS |
| testUpdateUser002 | Kiểm thử phương thức update() đối tượng User | 1. Nhập username, password, email, isActive, isAdmin 2. Gọi findByUsername(username) 3. Gọi update(user) 4. Chạy Junit Test | String username = "";  String password = "";  String email = "";  Boolean isAdmin = null;  Boolean isActive = null; | Cập nhật thất bại | Cập nhật thất bại | PASS |
| testUpdateUser003 | Kiểm thử phương thức update() đối tượng User | 1. Nhập username, password, email, isActive, isAdmin 2. Gọi findByUsername(username) 3. Gọi update(user) 4. Chạy Junit Test | String username = "hungphan001";  String password = "";  String email = "";  Boolean isAdmin = null;  Boolean isActive = null; | Cập nhật thất bại | Cập nhật thất bại | PASS |
| testUpdateUser004 | Kiểm thử phương thức update() đối tượng User | 1. Nhập username, password, email, isActive, isAdmin 2. Gọi findByUsername(username) 3. Gọi update(user) 4. Chạy Junit Test | String username = "hungphan001";  String password = "123456789";  String email = "";  Boolean isAdmin = null;  Boolean isActive = null; | Cập nhật thất bại | Cập nhật thất bại | PASS |
| testUpdateUser005 | Kiểm thử phương thức update() đối tượng User | 1. Nhập username, password, email, isActive, isAdmin 2. Gọi findByUsername(username) 3. Gọi update(user) 4. Chạy Junit Test | String username = "hungphan001";  String password = "123456789";  String email = "hungphan001@gmail.com";  Boolean isAdmin = null;  Boolean isActive = null; | Cập nhật thất bại | Cập nhật thất bại | PASS |
| testUpdateUser006 | Kiểm thử phương thức update() đối tượng User | 1. Nhập username, password, email, isActive, isAdmin 2. Gọi findByUsername(username) 3. Gọi update(user) 4. Chạy Junit Test | String username = "hungphan001";  String password = "123456789";  String email = "hungphan001@gmail.com";  Boolean isAdmin = true;  Boolean isActive = null; | Cập nhật thất bại | Cập nhật thất bại | PASS |
| testUpdateUser007 | Kiểm thử phương thức update() đối tượng User | 1. Nhập username, password, email, isActive, isAdmin 2. Gọi findByUsername(username) 3. Gọi update(user) 4. Chạy Junit Test | String username = "hungphan002";  String password = "123456789";  String email = "hungphan001@gmail.com";  Boolean isAdmin = true;  Boolean isActive = true; | Cập nhật thất bại | Cập nhật thất bại | PASS |

### Junit Test Code

**public** **class** UpdateUserTest {

**private** UserService userService;

@Before

**public** **void** beforeTest() {

userService = **new** UserServiceImpl();

System.***out***.println("Before Test");

}

@After

**public** **void** afterTest() {

System.***out***.println("After Test");

System.***out***.println("--------------------------------------");

}

@Test

**public** **void** testUpdateUser001() **throws** Exception {

String expected = "Cập nhật thành công";

String actual = "";

**try** {

String username = "hungphan001";

String password = "123456789";

String email = "hungphan001@gmail.com";

Boolean isAdmin = **true**;

Boolean isActive = **true**;

User user = userService.findByUsername(username);

user.setUsername(username);

user.setPassword(password);

user.setEmail(email);

user.setAdmin(isAdmin);

user.setActive(isActive);

User updatedUser = userService.update(user);

**if** (updatedUser != **null**) {

actual = "Cập nhật thành công";

} **else** {

**throw** **new** Exception();

}

} **catch** (Exception e) {

actual = "Cập nhật thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testUpdateUser002() **throws** Exception {

String expected = "Cập nhật thất bại";

String actual = "";

**try** {

String username = "";

String password = "";

String email = "";

Boolean isAdmin = **null**;

Boolean isActive = **null**;

User user = userService.findByUsername(username);

user.setUsername(username);

user.setPassword(password);

user.setEmail(email);

user.setAdmin(isAdmin);

user.setActive(isActive);

User updatedUser = userService.update(user);

**if** (updatedUser != **null**) {

actual = "Cập nhật thành công";

} **else** {

**throw** **new** Exception();

}

} **catch** (Exception e) {

actual = "Cập nhật thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testUpdateUser003() **throws** Exception {

String expected = "Cập nhật thất bại";

String actual = "";

**try** {

String username = "hungphan001";

String password = "";

String email = "";

Boolean isAdmin = **null**;

Boolean isActive = **null**;

User user = userService.findByUsername(username);

user.setUsername(username);

user.setPassword(password);

user.setEmail(email);

user.setAdmin(isAdmin);

user.setActive(isActive);

User updatedUser = userService.update(user);

**if** (updatedUser != **null**) {

actual = "Cập nhật thành công";

} **else** {

**throw** **new** Exception();

}

} **catch** (Exception e) {

actual = "Cập nhật thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testUpdateUser004() **throws** Exception {

String expected = "Cập nhật thất bại";

String actual = "";

**try** {

String username = "hungphan001";

String password = "123456789";

String email = "";

Boolean isAdmin = **null**;

Boolean isActive = **null**;

User user = userService.findByUsername(username);

user.setUsername(username);

user.setPassword(password);

user.setEmail(email);

user.setAdmin(isAdmin);

user.setActive(isActive);

User updatedUser = userService.update(user);

**if** (updatedUser != **null**) {

actual = "Cập nhật thành công";

} **else** {

**throw** **new** Exception();

}

} **catch** (Exception e) {

actual = "Cập nhật thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testUpdateUser005() **throws** Exception {

String expected = "Cập nhật thất bại";

String actual = "";

**try** {

String username = "hungphan001";

String password = "123456789";

String email = "hungphan001@gmail.com";

Boolean isAdmin = **null**;

Boolean isActive = **null**;

User user = userService.findByUsername(username);

user.setUsername(username);

user.setPassword(password);

user.setEmail(email);

user.setAdmin(isAdmin);

user.setActive(isActive);

User updatedUser = userService.update(user);

**if** (updatedUser != **null**) {

actual = "Cập nhật thành công";

} **else** {

**throw** **new** Exception();

}

} **catch** (Exception e) {

actual = "Cập nhật thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testUpdateUser006() **throws** Exception {

String expected = "Cập nhật thất bại";

String actual = "";

**try** {

String username = "hungphan001";

String password = "123456789";

String email = "hungphan001@gmail.com";

Boolean isAdmin = **true**;

Boolean isActive = **null**;

User user = userService.findByUsername(username);

user.setUsername(username);

user.setPassword(password);

user.setEmail(email);

user.setAdmin(isAdmin);

user.setActive(isActive);

User updatedUser = userService.update(user);

**if** (updatedUser != **null**) {

actual = "Cập nhật thành công";

} **else** {

**throw** **new** Exception();

}

} **catch** (Exception e) {

actual = "Cập nhật thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testUpdateUser007() **throws** Exception {

String expected = "Cập nhật thất bại";

String actual = "";

**try** {

String username = "hungphan002";

String password = "123456789";

String email = "hungphan001@gmail.com";

Boolean isAdmin = **true**;

Boolean isActive = **true**;

User user = userService.findByUsername(username);

user.setUsername(username);

user.setPassword(password);

user.setEmail(email);

user.setAdmin(isAdmin);

user.setActive(isActive);

User updatedUser = userService.update(user);

**if** (updatedUser != **null**) {

actual = "Cập nhật thành công";

} **else** {

**throw** **new** Exception();

}

} **catch** (Exception e) {

actual = "Cập nhật thất bại";

}

System.***out***.println("Expected: " + expected);

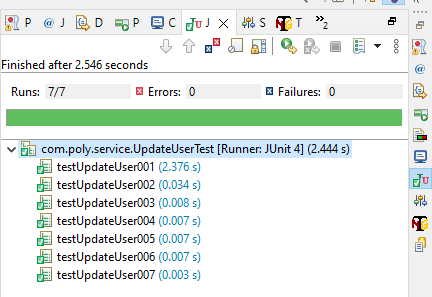
System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

}

### Test Result



## TEST DELETE USER

### Test Case

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **TEST CASE** | **TEST STEP** | **TEST DATA** | **EXPECTED** | **ACTUAL** | **STATUS** |
| testDeleteUser001 | Kiểm thử phương thức delete() đối tượng User | 1. Nhập username 2. Gọi delete(username) 3. Chạy Junit Test | String username = "hungphan001"; | Tạo mới thành công | Tạo mới thành công | PASS |
| testDeleteUser002 | Kiểm thử phương thức delete() đối tượng User | 1. Nhập username 2. Gọi delete(username) 3. Chạy Junit Test | String username = ""; | Xóa thất bại | Xóa thất bại | PASS |
| testDeleteUser003 | Kiểm thử phương thức delete() đối tượng User | 1. Nhập username 2. Gọi delete(username) 3. Chạy Junit Test | String username = "hungphan002"; | Xóa thất bại | Xóa thất bại | PASS |

### Junit Test Code

**public** **class** DeleteUserTest {

**private** UserService userService;

@Before

**public** **void** beforeTest() {

userService = **new** UserServiceImpl();

System.***out***.println("Before Test");

}

@After

**public** **void** afterTest() {

System.***out***.println("After Test");

System.***out***.println("--------------------------------------");

}

@Test

**public** **void** testDeleteUser001() **throws** Exception {

String expected = "Xóa thành công";

String actual = "";

**try** {

String username = "hungphan001";

User updatedUser = userService.delete(username);

**if** (updatedUser != **null**) {

actual = "Xóa thành công";

} **else** {

**throw** **new** Exception();

}

} **catch** (Exception e) {

actual = "Xóa thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testDeleteUser002() **throws** Exception {

String expected = "Xóa thất bại";

String actual = "";

**try** {

String username = "";

User updatedUser = userService.delete(username);

**if** (updatedUser != **null**) {

actual = "Xóa thành công";

} **else** {

**throw** **new** Exception();

}

} **catch** (Exception e) {

actual = "Xóa thất bại";

}

System.***out***.println("Expected: " + expected);

System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

@Test

**public** **void** testDeleteUser003() **throws** Exception {

String expected = "Xóa thất bại";

String actual = "";

**try** {

String username = "hungphan002";

User updatedUser = userService.delete(username);

**if** (updatedUser != **null**) {

actual = "Xóa thành công";

} **else** {

**throw** **new** Exception();

}

} **catch** (Exception e) {

actual = "Xóa thất bại";

}

System.***out***.println("Expected: " + expected);

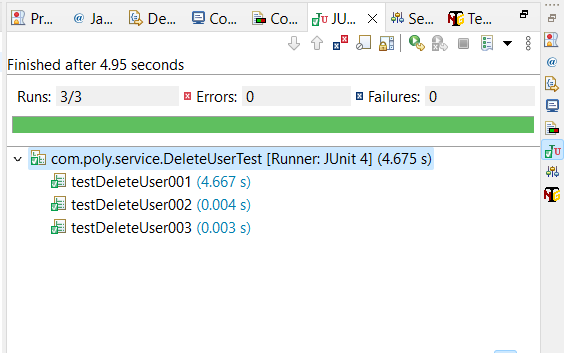
System.***out***.println("Actual: " + actual);

*assertEquals*(expected, actual);

}

}

### Test Result



# KIỂM THỬ AUTOMATION TEST

## TEST LOGIN

### Test Data

|  |  |  |
| --- | --- | --- |
| **Username** | **Password** | **Expected** |
| hungphan | 1234 | <http://localhost:8080/SOF3011_ASM/index> |
| hungphan | 123456 | <http://localhost:8080/SOF3011_ASM/index> |
| hung | 123456 | <http://localhost:8080/SOF3011_ASM/login> |
|  |  | <http://localhost:8080/SOF3011_ASM/login> |

### TestNG Automation Test Code

**public** **class** LoginTest {

**private** WebDriver driver;

**private** **final** String SRC = ExcelUtils.***DATA\_SRC*** + "LOGIN\_TEST.xlsx";

**private** Set<LoginData> logs;

**private** LoginData data;

@BeforeClass

**public** **void** init() {

System.*setProperty*("webdriver.chrome.driver", ExcelUtils.***CHROME\_DRIVER\_SRC***);

logs = **new** LinkedHashSet<>();

}

@BeforeMethod

**public** **void** setUp() {

driver = **new** ChromeDriver();

driver.get("http://localhost:8080/SOF3011\_ASM/login");

driver.manage().window().maximize();

data = **new** LoginData();

}

**private** **void** login(String username, String password) **throws** InterruptedException {

WebElement usernameInput = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div[2]/form/div[1]/input"));

usernameInput.sendKeys(username);

WebElement passwordInput = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div[2]/form/div[2]/input"));

passwordInput.sendKeys(password);

Thread.*sleep*(2000);

WebElement loginBtn = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div[2]/form/div[4]/button[1]"));

Actions actions = **new** Actions(driver).click(loginBtn);

actions.build().perform();

}

@Test(dataProvider = "loginData")

**public** **void** multipleLogin(String username, String password, String expected) **throws** InterruptedException {

login(username, password);

String currentURL = driver.getCurrentUrl();

data.setUsername(username);

data.setPassword(password);

data.setAction("Test login (authenticate) function");

data.setLogTime(**new** Date());

data.setExpected(expected);

data.setActual(currentURL);

*assertEquals*(currentURL, expected);

}

@AfterMethod

**public** **void** tearDown(ITestResult result) **throws** IOException {

data.setTestMethod(result.getName());

**switch** (result.getStatus()) {

**case** ITestResult.***SUCCESS***:

data.setStatus("PASS");

**break**;

**case** ITestResult.***FAILURE***:

data.setStatus("FAIL");

data.setException(result.getThrowable().getMessage());

String path = ExcelUtils.***IMAGES\_SRC*** + "failure-" + System.*currentTimeMillis*() + ".png";

ExcelUtils.*takeScreenShot*(driver, path);

data.setImage(path);

**break**;

**case** ITestResult.***SKIP***:

data.setStatus("SKIP");

**default**:

**break**;

}

logs.add(data);

driver.close();

}

@AfterClass

**public** **void** destroy() **throws** IOException {

data.writeLog(SRC, "RESULT\_TEST", logs);

}

@DataProvider(name = "loginData")

**public** Object[][] data() **throws** IOException {

XSSFWorkbook workbook = ExcelUtils.*getWorkbook*(SRC);

XSSFSheet sheet = workbook.getSheet("LOGIN\_DATA");

Object[][] data = ExcelUtils.*readSheetData*(sheet);

**return** data;

}

}

### Test Result



## TEST CREATE VIDEO

### Test Data

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **Href** | **Description** | **Expected** |
| Lừa Dối - Maya [LYRIC VIDEO] | 8DiPHu-DPV0 | Lừa Dối - Maya [LYRIC VIDEO] | Thêm mới thành công |
|  |  |  | Thêm mới thất bại |
| Lừa Dối - Maya [LYRIC VIDEO] | 8DiPHu-DPV0 | Lừa Dối - Maya [LYRIC VIDEO] | Thêm mới thành công |

### TestNG Automation Test Code

**public** **class** CreateVideoTest {

**private** WebDriver driver;

**private** **final** String SRC = ExcelUtils.***DATA\_SRC*** + "VIDEO\_TEST.xlsx";

**private** Set<CreateVideoData> logs;

**private** CreateVideoData data;

@BeforeClass

**public** **void** init() {

System.*setProperty*("webdriver.chrome.driver", ExcelUtils.***CHROME\_DRIVER\_SRC***);

logs = **new** LinkedHashSet<>();

}

**private** **void** login(String username, String password) {

WebElement usernameInput = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div[2]/form/div[1]/input"));

usernameInput.sendKeys(username);

WebElement passwordInput = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div[2]/form/div[2]/input"));

passwordInput.sendKeys(password);

WebElement loginBtn = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div[2]/form/div[4]/button[1]"));

Actions actions = **new** Actions(driver).click(loginBtn);

actions.build().perform();

}

@BeforeMethod

**public** **void** setUp() {

driver = **new** ChromeDriver();

driver.get("http://localhost:8080/SOF3011\_ASM/login");

driver.manage().window().maximize();

data = **new** CreateVideoData();

login("hungpbps26890", "123456");

WebElement videoNavLink = driver.findElement(By.*xpath*("/html/body/header/div/div/div/nav/ul/li[3]/a"));

videoNavLink.click();

}

@DataProvider(name = "createVideoData")

**public** Object[][] data() **throws** IOException {

XSSFWorkbook workbook = ExcelUtils.*getWorkbook*(SRC);

XSSFSheet sheet = workbook.getSheet("CREATE\_VIDEO\_DATA");

Object[][] data = ExcelUtils.*readSheetData*(sheet);

**return** data;

}

**private** **void** createVideo(String title, String href, String description) **throws** InterruptedException {

WebElement titleInput = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div/div[2]/div/div/form/div[1]/input"));

titleInput.sendKeys(title);

WebElement hrefInput = driver

.findElement(By.*xpath*("//\*[@id=\"href\"]"));

hrefInput.sendKeys(href);

WebElement descriptionInput = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div/div[2]/div/div/form/div[4]/textarea"));

descriptionInput.sendKeys(description);

Thread.*sleep*(2000);

WebElement createBtn = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div/div[2]/div/div/form/div[6]/div[1]/button"));

Actions actions = **new** Actions(driver).click(createBtn);

actions.build().perform();

}

@Test(dataProvider = "createVideoData")

**public** **void** createVideoTest(String title, String href, String description, String expected) **throws** InterruptedException {

createVideo(title, href, description);

WebElement messageInput = driver.findElement(By.*id*("message"));

String actual = messageInput.getAttribute("value");

data.setTitle(title);

data.setHref(href);

data.setDescription(description);

data.setAction("Test Create Video Function");

data.setLogTime(**new** Date());

data.setExpected(expected);

data.setActual(actual);

*assertEquals*(actual, expected);

}

@AfterMethod

**public** **void** tearDown(ITestResult result) **throws** IOException {

data.setTestMethod(result.getName());

**switch** (result.getStatus()) {

**case** ITestResult.***SUCCESS***:

data.setStatus("PASS");

**break**;

**case** ITestResult.***FAILURE***:

data.setStatus("FAIL");

data.setException(result.getThrowable().getMessage());

String path = ExcelUtils.***IMAGES\_SRC*** + "failure-" + System.*currentTimeMillis*() + ".png";

ExcelUtils.*takeScreenShot*(driver, path);

data.setImage(path);

**break**;

**case** ITestResult.***SKIP***:

data.setStatus("SKIP");

**default**:

**break**;

}

logs.add(data);

driver.close();

}

@AfterClass

**public** **void** destroy() **throws** IOException {

data.writeLog(SRC, "CREATE\_VIDEO\_RESULT", logs);

}

}

### Test Result



## TEST UPDATE VIDEO

### Test Data

|  |  |  |  |
| --- | --- | --- | --- |
| **VideoId** | **Title** | **Description** | **Expected** |
| 12 | Simp Gái 808 | Low G | Rap Nhà Làm Với Nhiều Tình Cảm | Simp Gái 808 | Low G | Rap Nhà Làm Với Nhiều Tình Cảm | Cập nhật thành công |
| 12 |  |  | Cập nhật thất bại |
| 12 | Simp Gái 808 | Low G | Rap Nhà Làm | Simp Gái 808 | Low G | Rap Nhà Làm | Cập nhật thất bại |

### TestNG Automation Test Code

**public** **class** UpdateVideoTest {

**private** WebDriver driver;

**private** **final** String SRC = ExcelUtils.***DATA\_SRC*** + "VIDEO\_TEST.xlsx";

**private** Set<UpdateVideoData> logs;

**private** UpdateVideoData data;

@BeforeClass

**public** **void** init() {

System.*setProperty*("webdriver.chrome.driver", ExcelUtils.***CHROME\_DRIVER\_SRC***);

logs = **new** LinkedHashSet<>();

}

**private** **void** login(String username, String password) {

WebElement usernameInput = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div[2]/form/div[1]/input"));

usernameInput.sendKeys(username);

WebElement passwordInput = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div[2]/form/div[2]/input"));

passwordInput.sendKeys(password);

WebElement loginBtn = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div[2]/form/div[4]/button[1]"));

Actions actions = **new** Actions(driver).click(loginBtn);

actions.build().perform();

}

@BeforeMethod

**public** **void** setUp() {

driver = **new** ChromeDriver();

driver.get("http://localhost:8080/SOF3011\_ASM/login");

driver.manage().window().maximize();

data = **new** UpdateVideoData();

login("hungpbps26890", "123456");

WebElement videoNavLink = driver.findElement(By.*xpath*("/html/body/header/div/div/div/nav/ul/li[3]/a"));

videoNavLink.click();

}

@DataProvider(name = "updateVideoData")

**public** Object[][] data() **throws** IOException {

XSSFWorkbook workbook = ExcelUtils.*getWorkbook*(SRC);

XSSFSheet sheet = workbook.getSheet("UPDATE\_VIDEO\_DATA");

Object[][] data = ExcelUtils.*readSheetData*(sheet);

**return** data;

}

**private** **void** updateVideo(String title, String description) **throws** InterruptedException {

WebElement titleInput = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div/div[2]/div/div/form/div[1]/input"));

titleInput.clear();

titleInput.sendKeys(title);

WebElement descriptionInput = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div/div[2]/div/div/form/div[4]/textarea"));

descriptionInput.clear();

descriptionInput.sendKeys(description);

Thread.*sleep*(2000);

WebElement updateBtn = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div/div[2]/div/div/form/div[6]/div[2]/button"));

Actions actions = **new** Actions(driver).click(updateBtn);

actions.build().perform();

}

@Test(dataProvider = "updateVideoData")

**public** **void** updateVideoTest(String videoId, String title, String description, String expected) **throws** InterruptedException {

String currentURL = driver.getCurrentUrl();

String editURL = currentURL + "/edit/" + videoId;

driver.get(editURL);

updateVideo(title, description);

WebElement messageInput = driver.findElement(By.*id*("message"));

String actual = messageInput.getAttribute("value");

data.setVideoId(videoId);

data.setTitle(title);

data.setDescription(description);

data.setAction("Test Update Video Function");

data.setLogTime(**new** Date());

data.setExpected(expected);

data.setActual(actual);

*assertEquals*(actual, expected);

}

@AfterMethod

**public** **void** tearDown(ITestResult result) **throws** IOException {

data.setTestMethod(result.getName());

**switch** (result.getStatus()) {

**case** ITestResult.***SUCCESS***:

data.setStatus("PASS");

**break**;

**case** ITestResult.***FAILURE***:

data.setStatus("FAIL");

data.setException(result.getThrowable().getMessage());

String path = ExcelUtils.***IMAGES\_SRC*** + "failure-" + System.*currentTimeMillis*() + ".png";

ExcelUtils.*takeScreenShot*(driver, path);

data.setImage(path);

**break**;

**case** ITestResult.***SKIP***:

data.setStatus("SKIP");

**default**:

**break**;

}

logs.add(data);

driver.close();

}

@AfterClass

**public** **void** destroy() **throws** IOException {

data.writeLog(SRC, "UPDATE\_VIDEO\_RESULT", logs);

}

}

### Test Result



## TEST DELETE VIDEO

### Test Data

|  |  |  |  |
| --- | --- | --- | --- |
| **VideoId** | **Title** | **Description** | **Expected** |
| 12 | Simp Gái 808 | Low G | Rap Nhà Làm | Simp Gái 808 | Low G | Rap Nhà Làm | Xóa thành công |
| 12 | Simp Gái 808 | Low G | Rap Nhà Làm | Simp Gái 808 | Low G | Rap Nhà Làm | Xóa thất bại |

### TestNG Automation Test code

**public** **class** DeleteVideoTest {

**private** WebDriver driver;

**private** **final** String SRC = ExcelUtils.***DATA\_SRC*** + "VIDEO\_TEST.xlsx";

**private** Set<UpdateVideoData> logs;

**private** UpdateVideoData data;

@BeforeClass

**public** **void** init() {

System.*setProperty*("webdriver.chrome.driver", ExcelUtils.***CHROME\_DRIVER\_SRC***);

logs = **new** LinkedHashSet<>();

}

**private** **void** login(String username, String password) {

WebElement usernameInput = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div[2]/form/div[1]/input"));

usernameInput.sendKeys(username);

WebElement passwordInput = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div[2]/form/div[2]/input"));

passwordInput.sendKeys(password);

WebElement loginBtn = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div[2]/form/div[4]/button[1]"));

Actions actions = **new** Actions(driver).click(loginBtn);

actions.build().perform();

}

@BeforeMethod

**public** **void** setUp() {

driver = **new** ChromeDriver();

driver.get("http://localhost:8080/SOF3011\_ASM/login");

driver.manage().window().maximize();

data = **new** UpdateVideoData();

login("hungpbps26890", "123456");

WebElement videoNavLink = driver.findElement(By.*xpath*("/html/body/header/div/div/div/nav/ul/li[3]/a"));

videoNavLink.click();

}

@DataProvider(name = "deleteVideoData")

**public** Object[][] data() **throws** IOException {

XSSFWorkbook workbook = ExcelUtils.*getWorkbook*(SRC);

XSSFSheet sheet = workbook.getSheet("DELETE\_VIDEO\_DATA");

Object[][] data = ExcelUtils.*readSheetData*(sheet);

**return** data;

}

**private** **void** deleteVideo() **throws** InterruptedException {

Thread.*sleep*(2000);

WebElement deletBtn = driver

.findElement(By.*xpath*("/html/body/div[2]/div/div/div/div[1]/div/div/div/div[2]/div/div/form/div[6]/div[3]/button"));

Actions actions = **new** Actions(driver).click(deletBtn);

actions.build().perform();

}

@Test(dataProvider = "deleteVideoData")

**public** **void** updateVideoTest(String videoId, String title, String description, String expected) **throws** InterruptedException {

String currentURL = driver.getCurrentUrl();

String editURL = currentURL + "/edit/" + videoId;

driver.get(editURL);

deleteVideo();

WebElement messageInput = driver.findElement(By.*id*("message"));

String actual = messageInput.getAttribute("value");

data.setVideoId(videoId);

data.setTitle(title);

data.setDescription(description);

data.setAction("Test Delete Video Function");

data.setLogTime(**new** Date());

data.setExpected(expected);

data.setActual(actual);

*assertEquals*(actual, expected);

}

@AfterMethod

**public** **void** tearDown(ITestResult result) **throws** IOException {

data.setTestMethod(result.getName());

**switch** (result.getStatus()) {

**case** ITestResult.***SUCCESS***:

data.setStatus("PASS");

**break**;

**case** ITestResult.***FAILURE***:

data.setStatus("FAIL");

data.setException(result.getThrowable().getMessage());

String path = ExcelUtils.***IMAGES\_SRC*** + "failure-" + System.*currentTimeMillis*() + ".png";

ExcelUtils.*takeScreenShot*(driver, path);

data.setImage(path);

**break**;

**case** ITestResult.***SKIP***:

data.setStatus("SKIP");

**default**:

**break**;

}

logs.add(data);

driver.close();

}

@AfterClass

**public** **void** destroy() **throws** IOException {

data.writeLog(SRC, "DELETE\_VIDEO\_RESULT", logs);

}

}

### Test Result



# Tổng kết

## KHÓ KHĂN

Trong quá trình thực hiện dự án còn trì trệ, nhóm thiếu tinh thần tự giác, chưa đồng quan điểm trong một vài vấn đề, một số thành viên còn lơ là không đầu tự thời gian, công sức để làm dự án.

## THUẬN LỢI

Mọi thành viên trong nhóm đều tranh luận với nhau, không ngừng nỗ lực để xây dựng tốt về dự án nhất.

Được sự hỗ trợ nhiệt tình từ thầy trong những buổi lên lớp, thầy đã phân tích, hướng dẫn, góp ý nên khiến mọi thứ dễ dàng hơn rất nhiều.