|  |  |
| --- | --- |
|  |  |
| Trần Khắc Hải | **BỘ CÔNG THƯƠNG**  **TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**  **---------------------------------------** |
|  |
| ĐỒ ÁN TỐT NGHIỆP |
| NGÀNH HỆ THỐNG THÔNG TIN |
| **XÂY DỰNG WEBSITE ĐẶT VÉ XEM FILM BHD STAR** |
|  |
|  |
| **CBHD: TS.Nguyễn Văn Tỉnh** |
| HỆ THỐNG THÔNG TIN | **Sinh viên: Trần Khắc Hải** |
| **Mã số sinh viên: 2019600859** |
|  |
|  |
|  |
|  |
|  |
|  | Hà Nội – Năm 2023 |

**Lời mở đầu**

**1. Tên đề tài**

Xây dựng website đặt vé xem film BHD Star

**2. Mục tiêu của đề tài**

* Tên đề tài: “Xây dựng website đặt vé xem film BHD Star”
* Đề tài đáp ứng được những mục tiêu:
* Nắm bắt được những kiến thức cần để thiết kế, xây dựng một website, kiến thức phân tích thiết kế hệ thống, cơ sở dữ liệu.
* Có hiểu biết về các kiến thức cơ bản như C#, VueJS Framework, SQL server,... cùng với các kiến thức, công cụ liên quan.
* Giúp khách hàng đặt vé xem film một cách đơn giản, nhanh chóng và thuận tiện chỉ cần có thiết bị kết nối internet.
* Website đặt vé là website đặt vé trực tuyến. Khách hàng không cần phải đến tận rạp để đặt vé mà chỉ cần chọn film yêu thích, chọn thời gian và địa điểm rạp, hệ thống sẽ trả về thông tin vé. Khách hàng chỉ cần chụp màn hình thông tin vé và mang đến quầy tiếp tân tại rap để nhận được vé.

**3. Đối tượng và phạm vi**

Bất kì ai có nhu cầu đặt vé xem film trực tuyến trên nền tảng Internet.

**4. Kết quả dự án đạt được**

Website gồm những tính năng chính:

* Khách hàng đăng ký, đăng nhập tài khoản cá nhân.
* Thay đổi mật khẩu, cập nhật thông tin cá nhân.
* Quên mật khẩu của tài khoản.
* Phân quyền người quản trị, khách hàng.
* Chức năng quản trị: quản lý tài khoản, quản lý danh sách film, quản lý danh sách quảng cáo, quản lý vé.
* Chức năng của website: xem danh sách film, xem hệ thống rạp, xem khuyến mãi / sự kiện, đặt vé và thanh toán.

**CHƯƠNG 1: GIỚI THIỆU HỆ THỐNG ĐẶT VÉ, CÔNG CỤ LỰA CHỌN, NGÔN NGỮ SỬ DỤNG**

1. **Giới thiệu hệ thống**
2. **Bài toán xây dựng website**

* Ngày nay, với sự bùng nổ của cuộc cách mạng 4.0, công nghệ thông tin đã có sức ảnh hưởng sâu sắc đến đời sống, kinh tế và xã hội. Để tồn tại và phát triển, con người cần phải học tập và tiếp cận nhanh chóng với công nghệ mới, theo kịp sự phát triển của thời đại. Theo sự phát triển của thời đại, chất lượng đời sống ngày càng được cải thiện thì nhu cầu giải trí đối với mỗi người là không thể thiếu. Tùy thuộc vào sở thích cá nhân của mỗi người thì cũng có các hình thức giải trí khác nhau như chơi game, đọc sách, nghe nhạc, xem film,... Trong đó nhu cầu xem film là vô cùng phổ biến và được nhiều người lựa chọn hơn cả.
* Do như cầu xem film ngày càng tăng cao, việc đặt vé film truyền thống thường gặp rất nhiều khó khăn như phải xếp hàng lâu, không biết được chỗ ngồi còn trống hay đã hết,... Để tăng trải nghiệm người dùng, các rạp chiếu film đã phát triển các website đặt vé xem film trực tuyến để mang lại nhiều lợi ích hơn cho người dùng, như tiết kiệm thời gian mua vé, chủ động chọn được suất chiếu phù hợp với thời gian cá nhân, chọn được vị trí ngồi phù hợp, trải nghiệm người dùng tốt hơn,... Đồ án này cũng là một cơ hội để em áp dụng các kiến thức đã học về thiết kế web, lập trình web, cơ sở dữ liệu,... vào các dự án thực tế.
* Từ các yêu cầu kể trên, em sẽ đưa ra phân tích hệ thống, yêu cầu chức năng và phi chức năng của website đặt vé:

1. Phân tích hệ thống

* Vai trò người dùng:
  + Hệ thống chia ra 2 nhóm người dùng chính:
    - Quản trị viên (Admin)
    - Khách hàng (User)
  + Mô tả người dùng

|  |  |
| --- | --- |
| Vai trò | Mô tả |
| Quản trị viên | Là người có vai trò cao nhất trên toàn bộ hệ thống. Người quản trị có toàn quyền quản lý các thao tác: thêm mới, sửa, xóa, cập nhật toàn bộ thông tin trong hệ thống |
| Khách hàng | Là người thực hiện các thao tác vụ cơ bản được hiển thị trên giao diện website như xem danh sách film, thông tin rạp, xem thông tin sự kiện/khuyến mãi, đặt vé xem film, thanh toán, quản lý thông tin cá nhân,... |

**Bảng 1: Mô tả vai trò nhóm người dùng**

1. Yêu cầu chức năng

|  |  |
| --- | --- |
| Vai trò | Yêu cầu chức năng |
| Quản trị viên | - Quản lý toàn bộ thông tin film  - Quản lý thông tin các chương trình khuyến mãi/sự kiện  - Quản lý thông tin khách hàng  - Quản lý thông tin đơn hàng  - Theo đõi được mức độ tăng trưởng của rap thông qua các chỉ số: số người đăng ký mới qua từng tháng, số vé được đặt qua từng tháng,... |
| Người dùng | - Xem danh sách film  - Xem thông tin film  - Tạo tài khoản, thiết lập thông tin cá nhân  - Đặt vé xem film |

**Bảng 2: Bảng mô tả yêu cầu chức năng hệ thống**

1. Yêu cầu phi chức năng

* Khả năng tiếp cận (Accessibility)
* Hệ thống có khả năng truy cập từ bất cứ thiết bị nào có kết nối Internet như PC, laptop, máy tính bảng, Ipad,...
* Tính khả dụng (Availablility)
* Hệ thống có thể được truy cập vào bất cứ thời gian nào trong ngày và bất cứ ngày nào trong tuần
* Tính bảo mật (Security)
* Để thực hiện các thao tác đặc thù của hệ thống, người dùng cần phải đăng nhập tài khoản, hệ thống sẽ tự phát hiện ra đó là tài khoản đó có vai trò người dùng hay quản trị viên để từ đó chuyển đến các trang cho phù hợp
* Phân quyền rõ ràng giữa quản trị viên và khách hàng
* Mọi thông tin của khách hàng sẽ được bảo mật
* Hiệu năng (Performance)
* Tốc độ phản hồi nhanh
* Điều hướng mượt, độ trễ thấp
* Khả năng mở rộng (Scalability)
* Hệ thống có thể được mở rộng theo yêu cầu và hoạt động tốt.
* Front-end và Back-end có thể dễ dàng mở rộng linh hoạt

1. **Công cụ lựa chọn**

* Visual Studio Code: Công cụ sử dụng để lập trình HTML, CSS, JavaScript
* Visual Studio: Công cụ sử dụng để lập trình C#
* SQL server: Hệ thống quản lý cơ sở dữ liệu quan hệ dùng để lưu trữ dữ liệu
* Case Studio: Công cụ thiết kế mô hình thực thể liên kết
* Rational Rose: Công cụ thiết kế hệ thống

1. **Ngôn ngữ sử dụng**
   1. **Giới thiệu về SQL Server**
      1. Khái niệm

Là một hệ quản trị dữ liệu quan hệ sử dụng câu lệnh SQL để trao đổi dữ liệu giữa máy cái SQL server và máy client. Một Relational Database Management System – RDBMS gồm có: databases, datase engine và các chương trình ứng dụng dùng để quản lý các bộ phận trong RDBMS và những dữ liệu khác.

* + 1. Lịch sử ra đời và các phiên bản của SQL Server
* Lịch sử ra đời:
* Năm 1989, phiên bản đầu tiên của SQL Server 1.0 ra đời được dùng cho các hệ điều hành 16 bit và được phát triển cho tới ngày nay.
* Cho tới khi SQL Server ra phiên bản 6.5 thì được thị trường chấp nhận rộng rãi. Một đột phá cải tiến cho SQL Server 7.0 khi được Microsoft viết lại một engine hoàn toàn mới. Đến khi SQL Server từ phiên bản 7.0 cải tiến lên 8.0 chủ yếu phát triển về tính năng [**thiết kế website**](https://monamedia.co/dich-vu/thiet-ke-website/).
* Cho đến ngày nay thì phiên bản mới nhất đó là SQL Server 2016 hỗ trợ bộ vi xử lý 64 bit ra đời vào ngày 1 tháng 6 năm 2016.
* Các phiên bản của SQL server
* Enterprise: là một ấn bản chứa tất cả các đặc điểm nổ bật của SQL Server như: các công cụ cho tạo và quản lý phân cụm SQL Server, nhân bộ máy cơ sở dữ liệu và một số dịch vụ đi kèm. Nó có thể đánh địa chỉ 12 terabytes và quản lý cơ sở dữ liệu lên tới 524 petabytes.
* Standard: Ấn bản này có thể chạy tốt trên hệ thống lên tới 4 CPU và 2 GB RAM rất thích hợp cho các dịch vụ thiết kế web vừa và nhỏ.
* Developer: Ấn bản này giới hạn số lượng người kết nối với server nhưng có đầy đủ các tính năng của Enterprise Edition. Đây là phiên bản được sử dụng cho kiểm tra và phát triển ứng dụng phù hợp cho các cá nhân trong lĩnh vực web như: [freelancer Việt Nam](http://freelancervietnam.vn/), [website spa](http://websitespa.vn/),…
* Workgroup: ấn bản SQL Server này có các chức năng lõi cơ sở dữ liệu nhưng không đi kèm các dịch vụ. Ở phiên bản 2012 không có ấn bản này.
* Express: Ấn bản này dễ dàng sử dụng và quản trị cơ sở dữ liệu đơn giản.
* Các thành phần cơ bản trong SQL server

Các thành cơ bản trong SQL Server gồm có: Reporting Services, Database Engine, Integration Services, Notification Services, Full Text Search Service, … Tất cả kết hợp với nhau tạo thành một giải pháp hoàn chỉnh giúp cho việc phân tích và lưu trữ dữ liệu trở nên dễ dàng hơn.

* **Database Engine**: Đây là một engine có khả năng chứa dữ liệu ở các quy mô dưới dạng support và table. Ngoài ra, nó còn có khả năng tự điều chỉnh ví dụ: trả lại tài nguyên cho ệ điều hành khi một user log off và sử dụng thêm các tài nguyên của máy khi cần.
* **Integration Services**: là tập hợp các đối tượng lập trình và các công cụ đồ họa cho việc sao chép, di chuyển và chuyển đổi dữ liệu.  Khi bạn làm việc trong một công ty lớn thì dữ liệu được lưu trữ ở nhiều nơi khác nhau như được chứa trong: Oracle, SQL Server, DB2, Microsoft Access, … và bạn chắc chắn sẽ có nhu cầu di chuyển dữ liệu giữa các server này. Ngoài ra, bạn còn muốn định dạng dữ liệu trước khi lưu vào database. Chắc chắn Integration Services sẽ giúp bạn giải quyết được công việc này dễ dàng.
* **Analysis Services**: Đây là một dịch vụ phân tích dữ liệu rất hay của Microsoft. Dữ liệu khi được lưu trữ vào trong database mà bạn không thể lấy được những thông tin bổ ích thì coi như không có ý nghĩa gì. Chính vì thế, công cụ này ra đời giúp bạn trong việc phân tích dữ liệu một cách hiệu quả và dễ dàng bằng cách dùng kỹ thuật khai thác dữ liệu – datamining và khái niệm hình khối nhiều chiều – multi dimendion cubes.
* **Notification Services**: Dịch vụ thông báo này là nền tảng cho sự phát triển và triển khai các ứng dụng soạn và gửi thông báo. Ngoài ra, dịch vụ này còn có chức năng gửi thông báo theo dịch thời đến hàng ngàn người dăng ký sử dụng trên nhiều loại thiết bị khác nhau.
* **Reporting Services**: là một công cụ tạo, quản lý và triển khai báo cáo bao gồm: server và client. Ngoài ra, nó còn là nền tảng cho việc phát triển và xây dựng các ứng dụng báo cáo.
* **Full Text Search Service**: là một thành phần đặc biệt trong việc truy vấn và đánh chỉ mục dữ liệu văn bản không cấu trúc được lưu trữ trong các cơ sở dữ liệu SQL Server.
* **Service Broker**: là một môi trường lập trình cho việc tạo ra các ứng dụng trong việc nhảy qua các Instance.

Tham khảo: https://sqladvice.com/tong-quan-ve-sql-server/

* 1. **Giới thiệu về VueJS Framework**
     1. **VueJS Framework là gì?**
* Là một framework [mã nguồn mở](https://glints.com/vn/blog/open-source-la-gi/) của JavaScript được sử dụng để phát triển các giao diện web tương tác. Nó là một trong những framework nổi tiếng được sử dụng để đơn giản hóa việc phát triển web. VueJS tập trung vào view layer. Nó có thể dễ dàng tích hợp vào các dự án lớn để phát triển front-end mà không gặp bất kỳ sự cố nào.
* Việc cài đặt VueJS rất dễ dàng. Bất kỳ nhà phát triển nào cũng có thể hiểu và xây dựng giao diện web tương tác trong thời gian ngắn. VueJS được tạo ra bởi [Evan You](https://vuejs.org/about/team.html), một cựu nhân viên và[lập trình viên](https://glints.com/vn/blog/lap-trinh-vien-hoc-nganh-gi/) của Google. Phiên bản đầu tiên của VueJS được phát hành vào tháng 2 năm 2014. Gần đây, nó đã đạt 64.828 sao trên GitHub, khiến nó trở nên rất phổ biến.
  + 1. **Ưu và nhược điểm của VueJS**
* Ưu điểm:
* **Kích thước nhỏ**: Tệp zip được tải xuống của framework này chỉ nặng 18 KB. Điều này khiến nó không chỉ cài đặt nhanh mà còn tác động tích cực đến SEO và UX của bạn.
* **Kết xuất và hiệu suất DOM ảo:**Mô hình đối tượng tài liệu (DOM) là thứ bạn có thể gặp phải khi kết xuất các trang web. DOM đại diện cho một trang [HTML](https://glints.com/vn/blog/html-la-gi/) với các kiểu, thành phần và nội dung dưới dạng cấu trúc cây của các đối tượng (nút). Các đối tượng cây DOM lưu trữ dưới dạng cây và được tạo bởi trình duyệt khi tải trang.

vuejs là gì


**Hình 1:DOM ảo**

* Khi người dùng tương tác với trang, các đối tượng sẽ thay đổi trạng thái của chúng, do đó trình duyệt sẽ phải cập nhật thông tin và hiển thị trên màn hình. Tuy nhiên, việc cập nhật toàn bộ DOM rất phức tạp. Ưu tiên tốc độ load, VueJS sử dụng DOM ảo. Hãy coi đây là một bản sao của DOM gốc giúp tìm ra những phần tử cần cập nhật mà không cần kết xuất lại toàn bộ cây nút. Cách tiếp cận này giúp hiển thị trang khá nhanh và cải thiện hiệu suất ứng dụng.
* **Hệ thống phản ứng và các tùy chọn ràng buộc dữ liệu**: Liên kết dữ liệu là kết nối giữa mô hình dữ liệu (nguồn dữ liệu) và mẫu DOM hoặc HTML của chế độ xem. Liên kết dữ liệu một chiều cho phép thông tin truyền theo một hướng, từ mô hình sang chế độ xem hoặc ngược lại. Trong trường hợp đầu tiên, các thay đổi đối với nguồn sẽ tự động cập nhật DOM, nhưng nó không hoạt động ngược lại vì DOM có quyền truy cập chỉ đọc vào mô hình.
* Liên kết dữ liệu hai chiều cho phép trao đổi dữ liệu giữa mô hình và chế độ xem theo cả hai hướng. Nói cách khác, mô hình cũng lắng nghe các sự kiện trên DOM và bất kỳ cập nhật nào ở một bên sẽ phản ánh ngay lập tức ở bên kia. Cách tiếp cận này loại bỏ mã soạn sẵn và đơn giản hóa việc phát triển ứng dụng. Tuy nhiên, việc khó Debug và dễ xảy ra lỗi khiến luồng hai chiều không phù hợp cho các dự án lớn.
* Nhược điểm:
* **Thiếu hỗ trợ cho các dự án quy mô lớn**: Quy mô nhóm phát triển và cộng đồng của VueJS vẫn không thể so sánh với Angular hay React. Framework này cũng không được hỗ trợ tài chính từ các doanh nghiệp lớn. Để được áp dụng trong các dự án quy mô lớn, công nghệ phải ổn định và được hỗ trợ mạnh mẽ để các vấn đề có thể được giải quyết nhanh chóng. Mặc dù VueJS không gặp nhiều vấn đề và thậm chí còn có nhu cầu đến từ các doanh nghiệp như IBM và Adobe, nhưng nó chủ yếu được sử dụng trong các dự án tương đối nhỏ.
* **Nguy cơ đến từ việc quá linh hoạt**: Tính linh hoạt là một đặc tính gây tranh cãi của một dự án lớn. Cung cấp cho nhóm phát triển của bạn quá nhiều tùy chọn có thể dẫn đến các cách tiếp cận lập trình khác nhau trong một nhóm. Và kết quả là, nó trở thành một công cụ vô hiệu hóa cuối cùng thay vì một phần mềm hoạt động.
* **Nguồn tài nguyên giới hạn**: Mặc dù hệ sinh thái khá rộng và có tất cả các công cụ cần thiết để bắt đầu phát triển với VueJS, nhưng framework này không lớn bằng React hay Angular. Nói chính xác hơn, chỉ cần so sánh số lượng plugin có sẵn cho React và Vue.js, sự khác biệt là ở hàng trăm đơn vị. Các plugin hiện có có thể được sử dụng với các framework khác cũng thường không được hỗ trợ.
* **Rào cản ngôn ngữ**: Việc áp dụng VueJS của các doanh nghiệp như Xiaomi và Alibaba đã giúp phổ biến framework và tạo ra nhu cầu trên thị trường lao động. Với việc Vue.js ngày càng phổ biến ở Trung Quốc, một phần quan trọng trong nội dung và các cuộc thảo luận của nó đều là tiếng Trung.
  + 1. **Ứng dụng**
* **Xử lý các nguyên mẫu**
* Trước hết, Vue.js được thiết kế để tạo nguyên mẫu. Với liên kết dữ liệu phù hợp, thật tuyệt vời khi framework này còn có thể xử lý nhiều hoạt ảnh, yếu tố tương tác và đồ họa. Tìm hiểu giao diện người dùng của bạn, cài đặt Vue CLI và bạn có thể sử dụng các nguyên mẫu có thể nhấp được.
* **Giữ sự tập trung vào giao diện người dùng**
* Vue.js tập trung vững chắc vào UI, vì nó chỉ có yêu cầu HTML, [CSS](https://glints.com/vn/blog/css-la-gi/) và JS để hoạt động với nó mà không cần quá nhiều thứ dành riêng cho Vue. Ví dụ: IBM đã sử dụng VueJS làm khung giao diện người dùng cho Đám mây lai của mình, vì đường cong học tập nhẹ nhàng và sự phụ thuộc vào HTML, CSS và JS.

vuejs là gì


**Hình 2: HTML vs VueJS**

* **Nhu cầu hội nhập**
* Nếu bạn có một ứng dụng và bạn muốn thêm một số tính tương tác vào ứng dụng đó, VueJS có thể trợ giúp điều này. Vì nó dựa trên JavaScript nên nó có thể dễ dàng tích hợp vào bất kỳ dự án nào bằng JS. Hơn nữa, nó tương thích với nhiều công nghệ back-end và framework như Laravel, Express, Rails và Django.
* Ngay cả khi xem xét nhược điểm của Vue.js, nó vẫn có thể được sử dụng trong các dự án lớn. Trong số những người chơi toàn cầu sử dụng VueJS để xây dựng trang web của họ có Grammarly, Upwork, Gitlab, Trivago, Nintendo và thậm chí cả Google. Danh sách này còn lâu mới hoàn thành và rõ ràng là sẽ được tiếp tục.
  + 1. **Một vài tính năng của VueJS**
* **DOM ảo**
* VueJS sử dụng DOM ảo, cũng được sử dụng bởi các framework khác như [React](https://glints.com/vn/blog/react-native-la-gi/), Ember, v.v. Các thay đổi không được thực hiện đối với DOM, thay vào đó, một bản sao của DOM được tạo dưới dạng cấu trúc dữ liệu JavaScript . Khi nào có bất kỳ thay đổi nào, chúng sẽ được thực hiện đối với cấu trúc dữ liệu JavaScript. Cấu trúc sau được so sánh với cấu trúc dữ liệu ban đầu. Những thay đổi cuối cùng sau đó được cập nhật vào DOM thực, người dùng sẽ thấy thay đổi này. Điều này tốt về mặt tối ưu hóa, ít tốn kém hơn và các thay đổi có thể được thực hiện với tốc độ nhanh hơn.
* **Ràng buộc dữ liệu**
* Tính năng ràng buộc dữ liệu giúp thao tác hoặc gán giá trị cho các thuộc tính HTML, thay đổi kiểu, gán lớp với sự trợ giúp của chỉ thị ràng buộc có tên v-bind có sẵn trong VueJS.
* **Component**
* Component là một trong những tính năng quan trọng của VueJS giúp tạo các phần tử tùy chỉnh và có thể tái sử dụng trong HTML.
* **Xử lý sự kiện**
* v-on là thuộc tính được thêm vào các phần tử DOM để lắng nghe các sự kiện trong VueJS.
* **Hoạt ảnh / Chuyển tiếp**
* VueJS cung cấp nhiều cách khác nhau để áp dụng chuyển đổi sang các phần tử HTML khi chúng được thêm, cập nhật hoặc xóa khỏi DOM. VueJS có một thành phần chuyển tiếp tích hợp cần được bao quanh phần tử để tạo hiệu ứng chuyển tiếp. Người dùng có thể dễ dàng thêm các thư viện hoạt hình của bên thứ ba và cũng có thể thêm nhiều tương tác hơn vào giao diện.
* **Thuộc tính được tính toán**
* Đây là một trong những tính năng quan trọng của VueJS. Nó giúp lắng nghe những thay đổi được thực hiện đối với các thành phần giao diện người dùng và thực hiện các tính toán cần thiết. Không cần mã hóa bổ sung cho việc này.
* **Mẫu**
* VueJS cung cấp các mẫu dựa trên HTML liên kết DOM với dữ liệu đối tượng Vue. VueJS biên dịch các mẫu thành các hàm Kết xuất DOM ảo. Chúng ta có thể sử dụng mẫu của các chức năng kết xuất và ngược lại.
* **Chỉ thị**
* VueJS có các chỉ thị tích hợp như v-if, v-else, v-show, v-on, v-bind và v-model, được sử dụng để thực hiện các hành động khác nhau trên giao diện người dùng.

Tham khảo: [VueJS là gì? Ứng dụng và một vài tính năng nổi trội của VueJS (glints.com)](https://glints.com/vn/blog/vuejs-la-gi/)

* 1. **Giới thiệu về ngôn ngữ C#**
     1. **C# là gì?**
* C# (hay C sharp) là một ngôn ngữ lập trình đơn giản, được phát triển bởi đội ngũ kỹ sư của Microsoft vào năm 2000, trong đó người dẫn đầu là Anders Hejlsberg và Scott Wiltamuth.
* C# là ngôn ngữ lập trình hiện đại, hướng đối tượng và nó được xây dựng trên nền tảng của hai ngôn ngữ mạnh nhất là C++ và Java.
* C# được thiết kế cho Common Language Infrastructure (CLI), mà gồm Executable Code và Runtime Environment, cho phép chúng ta sử dụng các ngôn ngữ high-level đa dạng trên các nền tảng và cấu trúc máy tính khác nhau.
* C# với sự hỗ trợ mạnh mẽ của .NET Framework giúp cho việc tạo một ứng dụng Windows Forms hay WPF (Windows Presentation Foundation), . . . trở nên rất dễ dàng.
  + 1. **Đặc trưng của C#**
* C# là ngôn ngữ đơn giản
* Như ta đã biết thì ngôn ngữ C# dựng trên nền tảng C++ và Java nên ngôn ngữ C# khá đơn giản. Nếu chúng ta thân thiện với C và C++ hoậc thậm chí là Java, chúng ta sẽ thấy C# khá giống về diện mạo, cú pháp, biểu thức, toán tử và những chức năng khác được lấy trực tiếp từ ngôn ngữ C và C++, nhưng nó đã được cải tiến để làm cho đơn giản hơn. Một vài trong các sự cải tiến là loại bỏ các dư thừa, hay là thêm vào những cú pháp thay đổi.
* C# là ngôn ngữ hiện đại
* Một vài khái niệm khá mới mẻ khá mơ hồ với các bạn vừa mới học lập trình, như xử lý ngoại lệ, những kiểu dữ liệu mở rộng, bảo mật mã nguồn ...v..v... Đây là những đặc tính được cho là của một ngôn ngữ hiện đại cần có. Và C# chứa tất cả các đặt tính ta vừa nêu trên. Các bạn sẽ dần tìm hiểu được các đặt tính trên qua các bài học trong series này.
* C# là một ngôn ngữ lập trình thuần hướng đối tượng
* Lập trình hướng đối tượng (tiếng Anh: Object-oriented programming, viết tắt: OOP) là một phương pháp lập trình có 4 tính chất. Đó là tính trừu tượng (*abstraction*), tính đóng gói (encapsulation), tính đa hình (polymorphism) và tính kế thừa (inheritance). C# hỗ trợ cho chúng ta tất cả những đặc tính trên. Và để hiểu rõ hơn thì chúng ta sẽ có một chương trình bày về phần này.
* C# là một ngôn ngữ ít từ khóa
* C được sử dụng để mô tả thôn# là ngôn ngữ sử dụng giới hạn những từ khóa (gồm khoảng 80 từ khóa và mười mấy kiểu dữ liệu xây dựng sẵn). Nếu bạn nghĩ rằng ngôn ngữ có càng nhiều từ khóa thì sẽ càng mạnh mẽ hơn. Điều này không phải sự thật, lấy ví dụ ngôn ngữ C# làm điển hình nhé. Nếu bạn học sâu về C# bạn sẽ thấy rằng ngôn ngữ này có thể được sử dụng để làm bất cứ nhiệm vụ nào.
* Ngoài những đặc điểm trên thì còn một số ưu điểm nổi bật của C#:
  + C# có cấu trúc khá gần gũi với các ngôn ngữ lập trình truyền thống, nên cũng khá dể dàng tiếp cận và học nhanh với C#.
  + C# có thể biên dịch trên nhiều nền tảng máy tính khác nhau.
  + C# được xây dựng trên nền tảng của C++ và Java nên nó được thừa hưởng những ưu điểm của ngôn ngữ đó.
  + C# là một phần của .NET Framework nên được sự chống lưng khá lớn đến từ bộ phận này.
  + C# có IDE Visual Studio cùng nhiều plug-in vô cùng mạnh mẽ.

Tham khảo: freetuts.net/c-sharp-la-gi-tong-quan-ve-c-sharp-1045.html

**CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ HỆ THỐNG**

1. **Mô hình hóa use case**
   1. Xây dựng biểu đồ use case



**Hình 3: Biểu đồ use case tổng quát Front-end**

****

**Hình 4: Biểu đồ use case tổng quát Back-end**



**Hình 5: Biểu đồ use case Đăng ký tài khoản khách hàng**



**Hình 6: Biểu đồ use case Đăng nhập phía front-end**



**Hình 7: Biểu đồ use case Xem danh sách rạp**



**Hình 8: Biều đồ use case Xem chi tiết rạp**



**Hình 9: Biều đồ use case Xem danh sách film**



**Hình 10: Biểu đồ use case Xem chi tiết film**



**Hình 11: Biểu đồ use case Quản lý hồ sơ khách hàng**



**Hình 12: Biểu đồ use case Xem thẻ thành viên**



**Hình 13: Biểu đồ use case Xem lịch sử giao dịch**



**Hình 14: Biểu đồ use case Đặt vé xem film**

* 1. Đặc tả
     1. Use case Đăng ký tài khoản khách hàng
* Use case này cho phép người dùng đăng ký tài khoản để đặt vé xem film
* Luồng sự kiện
* Luồng cơ bản:

1. Use case này bắt đầu khi khách hàng kích vào nút “Đăng nhập” trên thanh menu của website. Hệ thống sẽ hiển thị màn hình gồm 2 form: đăng nhập và đăng ký.
2. Khách hàng kích vào nút “Đăng ký ngay” để chuyển sang form đăng ký. Hệ thống sẽ hiển thị form đăng ký tài khoản gồm các thông tin: tên, số điện thoại, email, ngày tháng năm sinh, mật khẩu và nhập lại mật khẩu.
3. Khách hàng nhập các thông tin cần thiết vào form và kích vào nút “Đăng ký”. Hệ thống sẽ lưu cách thông tin của khách hàng vừa nhập vào bảng Accounts trong cơ sở dữ liệu và hiển thị thông báo “Đăng ký thành công!”
4. Use case kết thúc

* Luồng rẽ nhánh:

1. Tại bước 3 trong luồng cơ bản, nếu khách hàng nhập phần “nhập lại mật khẩu” không trùng với phần “mật khẩu” thì hệ thống sẽ hiển thị thông báo lỗi vào yêu cầu nhập lại.
2. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi vào use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

* Hậu điều kiện:

Không có.

* Điểm mở rộng:

Không có.

* + 1. Use case Đăng nhập
* Use case này cho phép khách hàng đăng nhập vào hệ thống
* Luồng sự kiện:
* Luồng cơ bản:

1. Use case này bắt đầu khi khách hàng kích vào nút “Đăng nhập” trên thành menu. Hệ thống sẽ hiển thị form đăng nhập gồm: email, mật khẩu.
2. Khách hàng nhập thông tin vào form đăng nhập và kích vào nút “Đăng nhập”. Hệ thống sẽ kiểm tra thông tin trong bảng Accounts từ cơ sở dữ liệu, nếu chính xác và quyền của tài khoản USER, hệ thống sẽ chuyển sang trang chủ của website.
3. Use case kết thúc.

* Luồng rẽ nhánh:

1. Tại bước 2 trong luồng cơ bản, nếu khách hàng nhập không chính xác thông tin đã được đăng ký, hệ thống sẽ hiển thị thông báo lỗi vào yêu cầu khách hàng nhập lại.
2. Tại bước 2 của luồng cơ bản, nếu khách hàng không nhớ mật khẩu của tài khoản, khách hàng có thể chọn mục “Quên mật khẩu”. Hệ thống sẽ chuyển sang form “Lấy lại mật khẩu” để cấp lại mật khẩu cho khách hàng.
3. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi vào use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Khách hàng đã đăng ký tài khoản thành công.

* Hậu điều kiện:

Khách hàng có thể thực hiện thao tác đặt vé.

* Điểm mở rộng:

Không có.

* + 1. Use case Xem danh sách rạp
* Use case này cho phép khách hàng xem danh sách rạp trên website
* Luồng sự kiện:
* Luồng cơ bản:

1. Use case này bắt đầu khi khách hàng kích vào nút “Hệ thống rạp” trên thành menu. Hệ thống sẽ lấy thông tin các rạp gồm: hình ảnh, tên rạp từ bảng Cinemas trong cơ sở dữ liệu và hiển thị lên màn hình.
2. Use case kết thúc.

* Luồng rẽ nhánh:

1. Tại bước 1 trong luồng cơ bản nếu không có bản ghi nào trong bảng Cinema thì hệ thống sẽ hiển thị thông báo “Hệ thống rạp đang được cập nhật” và use case kết thúc.
2. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

* Hậu điều kiện:

Không có.

* Điểm mở rộng:
  + 1. Use case Xem chi tiết rạp
* Use case này cho phép khách hàng xem thông tin chi tiết của rạp.
* Luồng sự kiện:
* Luồng cơ bản:

1. Use case này bắt đầu khi khách hàng kích vào “Hình ảnh” hoặc “Tên rạp” mình muốn. Hệ thống sẽ lấy thông tin rạp gồm: tên rạp, địa chỉ, số điện thoại, email liên hệ, thông tin phòng chiếu, hình ảnh từ bảng Cinemas trong cơ sở dữ liệu và hiển thị lên màn hình.
2. Use case kết thúc

* Luồng rẽ nhánh

1. Tại bước 1 trong luồng cơ bản nếu không có bản ghi nào trong bảng Cinemas thì hệ thống sẽ hiển thị thông báo “Hệ thống rạp đang được cập nhật” và use case kết thúc.
2. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

* Hậu điều kiện:

Không có.

* Điểm mở rộng:

Không có.

* + 1. Use case Xem danh sách film
* Use case này cho phép khách hàng xem danh sách film trên website
* Luồng sự kiện:
* Luồng cơ bản:

1. Use case này bắt đầu khi khách hàng kích vào nút “Lịch chiếu” trên thành menu. Hệ thống sẽ lấy thông tin các film gồm: hình ảnh, tên film từ bảng Films trong cơ sở dữ liệu và hiển thị lên màn hình.
2. Use case kết thúc.

* Luồng rẽ nhánh:

1. Tại bước 1 trong luồng cơ bản nếu không có bản ghi nào trong bảng Films thì hệ thống sẽ hiển thị thông báo “Film đang được cập nhật” và use case kết thúc.
2. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

* Hậu điều kiện:

Không có.

* Điểm mở rộng:

Không có.

* + 1. Use case Xem chi tiết film
* Use case này cho phép khách hàng xem thông tin chi tiết của film.
* Luồng sự kiện:
* Luồng cơ bản:

1. Use case này bắt đầu khi khách hàng kích vào “Hình ảnh” hoặc “Tên film” mình muốn. Hệ thống sẽ lấy thông tin film gồm: hình ảnh, tên film, chi tiết film, đạo diễn, khởi chiếu, thời lượng, độ tuổi yêu cầu từ bảng Films trong cơ sở dữ liệu và hiển thị lên màn hình.
2. Use case kết thúc

* Luồng rẽ nhánh

1. Tại bước 1 trong luồng cơ bản nếu không có bản ghi nào trong bảng Films thì hệ thống sẽ hiển thị thông báo “Film đang được cập nhật” và use case kết thúc.
2. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

* Hậu điều kiện:

Không có.

* Điểm mở rộng:

Không có.

* + 1. Use case Quản lý hồ sơ khách hàng
* Use case này cho phép khách hàng thay đổi thông tin cá nhân trên hệ thống
* Luồng sự kiện
* Luồng cơ bản:

1. Use case này bắt đầu khi khách hàng kích vào tên của mình ở trên thành menu khi đã đăng nhập. Hệ thống sẽ hiển thị một cửa sổ các danh mục lên màn hình gồm các mục: thông tin tài khoản, thẻ thành viên, lịch sử giao dịch, đăng xuất.
2. Khách hàng kích chọn “Thông tin tài khoản”. Hệ thống lấy các thông tin gồm: tên, email, số điện thoại, ngày tháng năm sinh, password từ bảng Accounts trong cơ sở dữ liệu và hiển thị lên màn hình.
3. Khách hàng nhập các thông tin cần thay đổi (trừ date) và kích nút “Cập nhật”. Hệ thống sẽ cập nhật thông tin hồ sơ vào bảng Accounts và hiển thị thông báo cập nhật thành công.
4. Use case kết thúc.

* Luồng rẽ nhánh

1. Tại bất kỳ thời điểm nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Khách hàng phải đăng nhập thành công vào website.

* Hậu điều kiện:

Không có.

* Điểm mở rộng:

Không có.

* + 1. Use case Xem thẻ thành viên
* Use case này cho phép khách hàng xem hạng thành viên của mình để hưởng các ưu đãi của rạp đối với từng loại thành viên.
* Luồng sự kiện:
* Luồng cơ bản:

1. Use case này bắt đầu khi khách hàng kích vào tên của mình ở trên thành menu khi đã đăng nhập. Hệ thống sẽ hiển thị một cửa sổ các danh mục lên màn hình gồm các mục: thông tin tài khoản, thẻ thành viên, lịch sử giao dịch, đăng xuất.
2. Khách hàng kích chọn “Thẻ thành viên”. Hệ thống sẽ lấy thông tin gồm: email từ bảng Accounts; cardNumber, loại thành viên, ngày đăng ký từ bảng Card\_Membership; tổng chi tiêu được tính từ dữ liệu được lấy từ bảng Bills trong cơ sở dữ liệu và hiển thị lên màn hình.
3. Use case kết thúc.

* Luồng rẽ nhành

1. Tại bất kỳ thời điểm nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Khách hàng phải đăng nhập thành công vào website.

* Hậu điều kiện:

Không có.

* Điểm mở rộng:

Không có.

* + 1. Use case Xem lịch sử giao dịch
* Use case này cho phép khách hàng xem lịch sử đặt vé tại website.
* Luồng sự kiện
* Luồng cơ bản:

1. Use case này bắt đầu khi khách hàng kích vào tên của mình ở trên thanh menu khi đã đăng nhập. Hệ thống sẽ hiển thị một cửa sổ các danh mục lên màn hình gồm các mục: thông tin tài khoản, thẻ thành viên, lịch sử giao dịch, đăng xuất.
2. Khách hàng kích chọn “Lịch sử giao dịch”. Hệ thống sẽ lấy thông tin gồm: thời gian giao dịch, số lượng từ bảng Bills; tên film từ bảng Films; tên vé, giá vé từ bảng Tickets, tên đồ ăn, giá đồ ăn từ bảng Foods; tên rạp từ bảng Cinemas trong cơ sở dữ liệu và hiển thị lên màn hình.
3. Use case kết thúc.

* Luồng rẽ nhánh

1. Tại bất kỳ thời điểm nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Khách hàng phải đăng nhập thành công vào website.

* Hậu điều kiện:

Không có.

* Điểm mở rộng:

Không có.

* + 1. Use case Đặt vé
* Use case này cho phép khách hàng đặt vé xem film trực tuyến.
* Luồng sự kiện
* Luồng cơ bản:

1. Use case bắt đầu khi khách hàng kích vào nút “Mua vé ngay” sau khi xem chi tiết film. Hệ thống lấy tên rạp, địa điểm từ bảng Cinemas; khung giờ và hiển thị lên màn hình.
2. Khách hàng kích chọn khung giờ mình muốn tương ứng với rạp mình muốn xem. Hệ thống sẽ chuyển khách hàng đến trang đặt vé.
3. Hệ thống lấy tên vé, giá vé từ bảng bảng Room\_Cinemas trong cơ sở dữ liệu và hiển thị lên màn hình. Khách hàng chọn số lượng của từng loại vé và bấm nút “Chọn ghế”.
4. Hệ thống sẽ hiển thị bảng ghế để cho khách hàng chọn. Khách hàng kích chọn số ghế tương ứng với từng hàng ghế, sau đó kích nút “Chọn đồ ăn”.
5. Hệ thống lấy thông tin gồm: Hình ảnh đồ ăn, tên đồ ăn, giá đồ ăn từ bẳng Foods trong cơ sở dữ liệu và hiển thị lên màn hình. Khách hàng kích chọn loại đồ ăn mình mong muốn và sau đó kích nút “Xác nhận”.
6. Hệ thống sẽ hiển thị các thông tin gồm: Tên film, địa điểm rạp, ngày xem, suất chiếu, rạp số, ghế xem, đồ ăn (nếu có), tổng tiền; 1 bảng gồm thông tin vé và thông tin đồ ăn (nếu có) lên màn hình. Khách hàng kiểm tra lại thông tin và chọn phương thức thanh toán.
7. Khách hàng kích nút “Thanh toán”. Sau khi thanh toán thành công, hệ thống sẽ hiển thị thông báo “Thanh toán thành công!”; hệ thống sẽ truy vấn bảng Accounts, Films, Tickets, Foods, Cinemas, Room\_Cinema để hiển thị các thông tin gồm: tên người đặt, số điện thoại người đặt, email người đặt, tên film, địa điểm rạp, ngày xem, suất chiếu, rạp số bao nhiêu, ngồi ghế nào, 1 bảng chứa các thông tin gồm loại vé, số lượng vé, giá vé, loại đồ ăn, số lượng đồ ăn, giá đồ ăn lên màn hình; sau đó hệ thống cập nhật các thông tin đặt vé vào bảng Bills.
8. Use case kết thúc.

* Luồng rẽ nhánh

1. Tại bước 1 của luồng cơ bản, nếu khách hàng chưa đăng nhập vào hệ thống, hệ thống sẽ chuyển khách hàng đến trang đăng nhập để khách hàng đăng nhập vào hệ thống.
2. Tại bước 1 của luồng cơ bản, nếu film được chọn chưa được chiếu, hệ thống sẽ hiển thị thông báo và use case kết thúc.
3. Tại bước 3 của luồng cơ bản, nếu khách hàng chưa kích chọn số lượng của từng loại vé mà đã bấm nút “Chọn ghế” thì hệ thống sẽ hiển thị thông báo “Khách hàng chưa chọn loại vé” và yêu cầu khách hàng chọn lại.
4. Tại bước 4 của luồng cơ bản, nếu khách hàng chưa chọn ghế mà đã bấm nút “Chọn đồ ăn” thì hệ thống sẽ hiển thị thông báo “Khách hàng chưa chọn ghế!” và yêu cầu phải chọn lại.
5. Tại bước 6 của luồng cơ bản, nếu thông tin không chính xác, khách hàng có thể kích vào nút “Hủy đặt vé” và use case kết thúc.
6. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Khách hàng phải đăng nhập vào hệ thống.

* Hậu điều kiện:

Không có.

* Điểm mở rộng:

Không có.

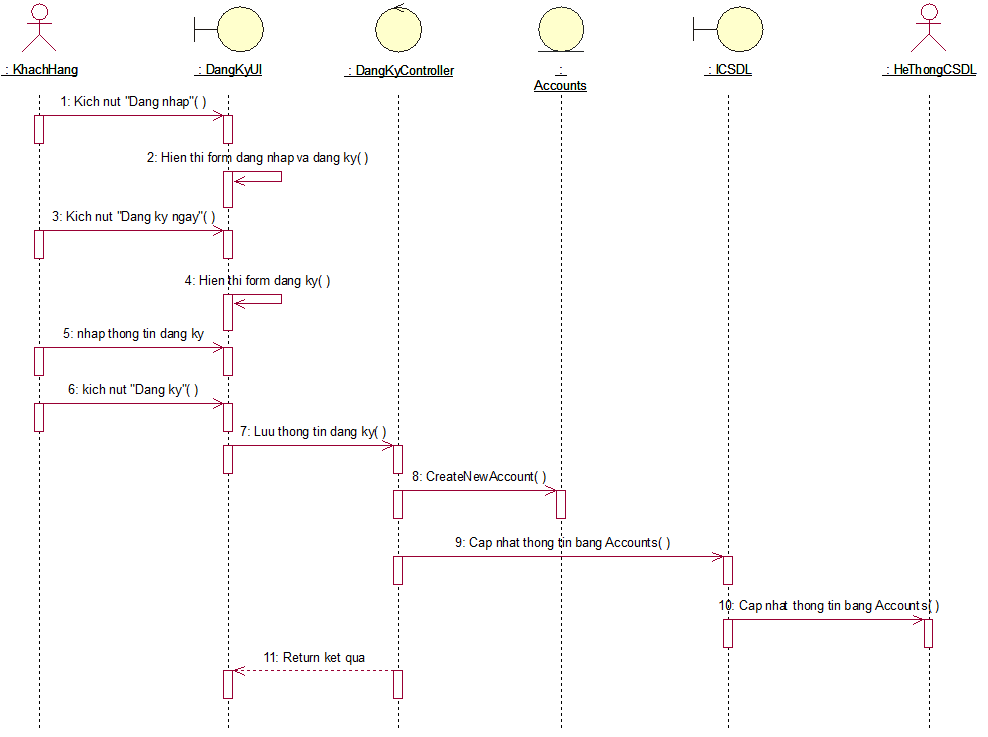
1. **Phân tích use case**
   1. **Phân tích use case Đăng ký tài khoản khách hàng**

* Biểu đồ VOPC

****

**Hình 15: Biểu đồ VOPC use case Đăng ký tài khoản khách hàng**

* Biểu đồ trình tự

****

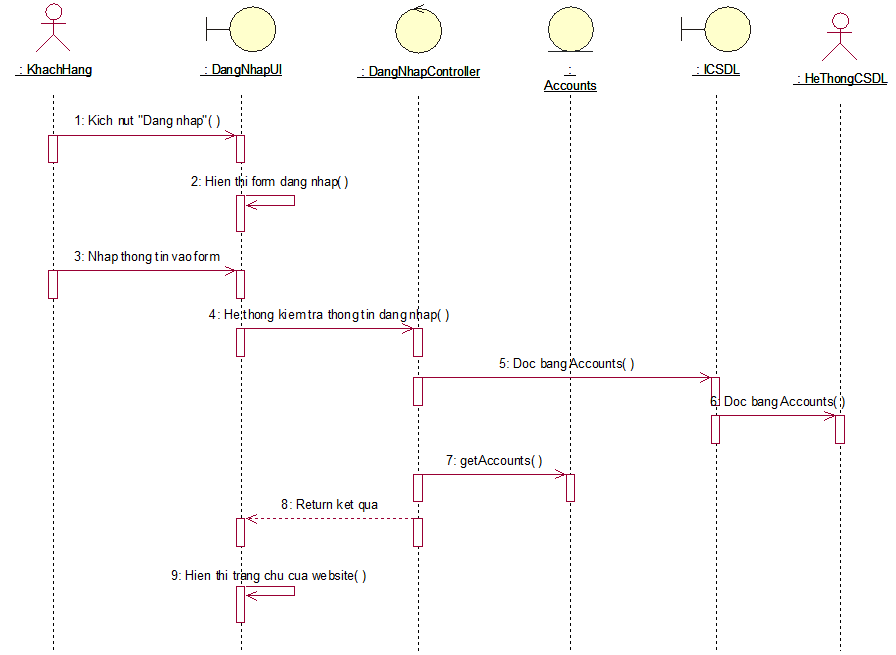
**Hình 16: Biểu đồ trình tự use case Đăng ký tài khoản khách hàng**

* 1. **Phân tích use case Đăng nhập**
* Biểu đồ VOPC

****

**Hình 17: Biểu đồ VOPC use case Đăng nhập**

* Biểu đồ trình tự

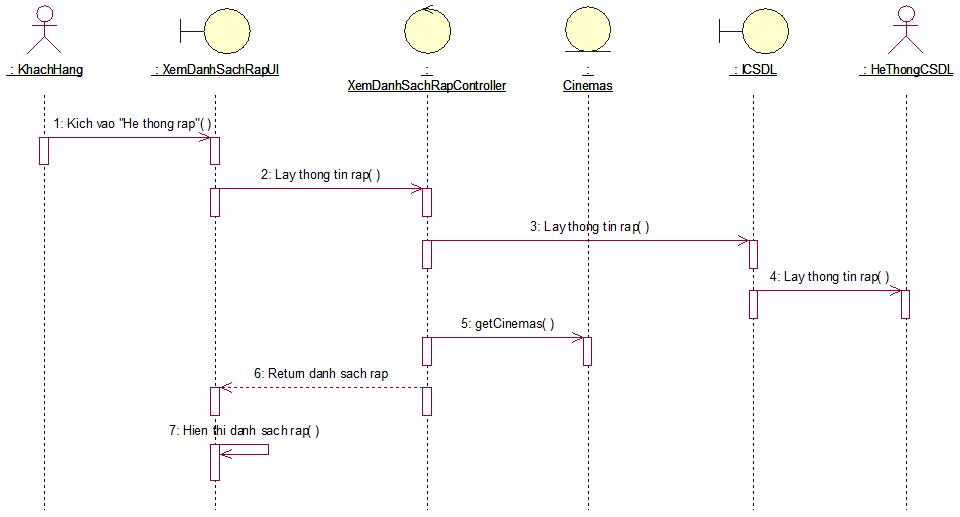
****

**Hình 18: Biểu đồ trình tự use case Đăng nhập**

* 1. **Phân tích use case Xem danh sách rạp**
* Biểu đồ VOPC

**Hình 19: Biểu đồ VOPC use case Xem danh sách rạp**

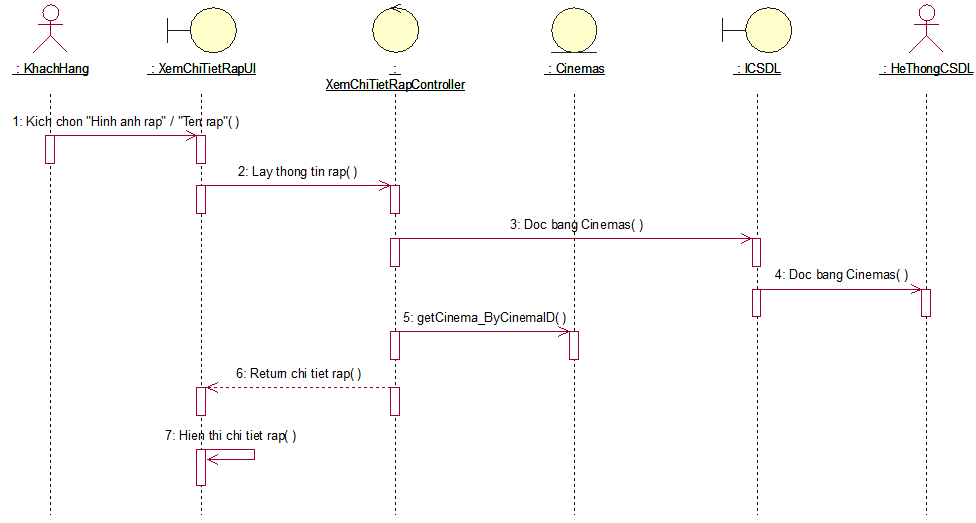
* Biểu đồ trình tự

**Hình 20: Biểu đồ trình tự use case Xem danh sách rạp**

* 1. **Phân tích use case Xem chi tiết rạp**
* Biểu đồ VOPC

**Hình 21: Biểu đồ VOPC use case Xem chi tiết rạp**

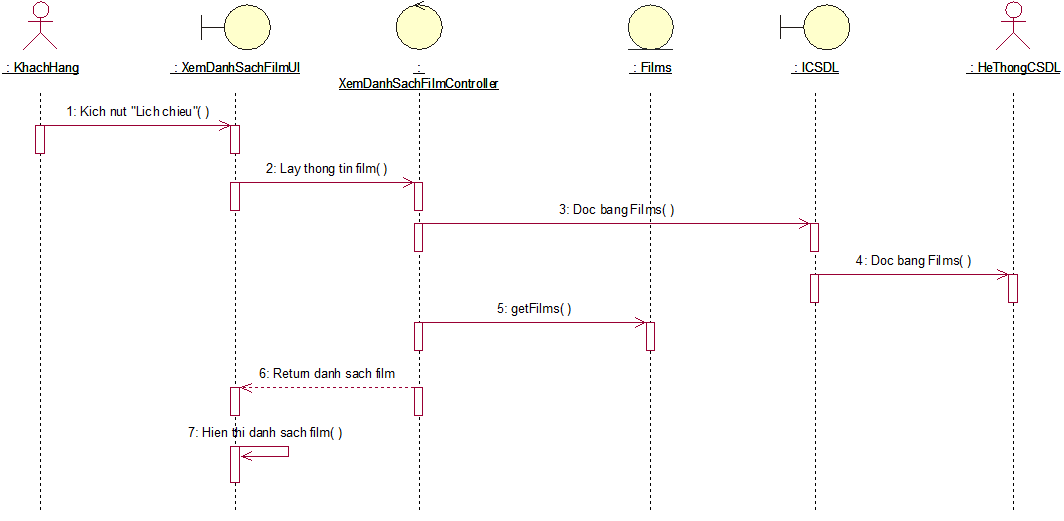
* Biểu đồ trình tự

**Hình 22: Biểu đồ trình tự use case Xem chi tiết rạp**

* 1. **Phân tích use case Xem danh sách film**
* Biểu đồ VOPC

**Hình 23: Biểu đồ VOPC use case Xem danh sách film**

* Biểu đồ trình tự

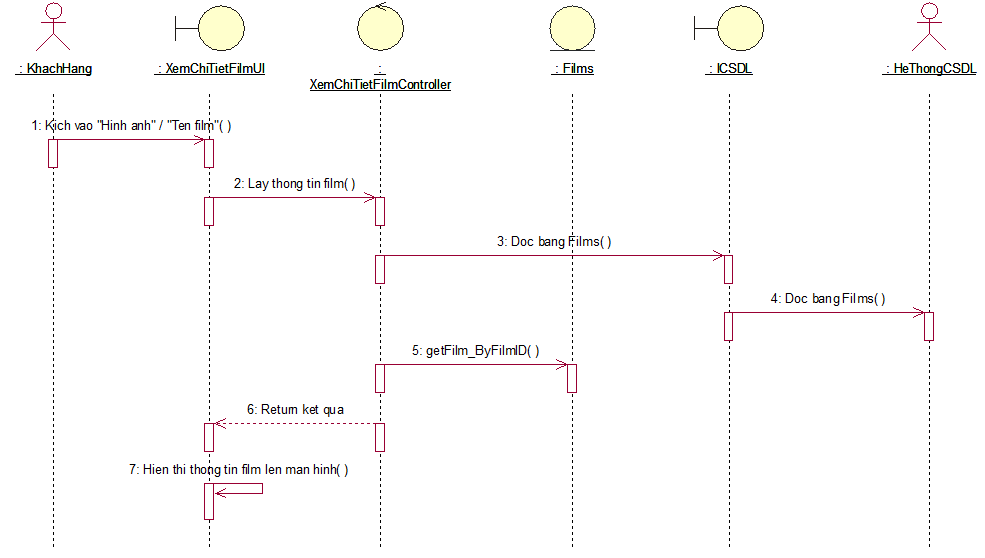
**Hình 24: Biểu đồ trình tự use case Xem danh sách film**

* 1. **Phân tích use case Xem chi tiết film**
* Biểu đồ VOPC

****

**Hình 25: Biểu đồ VOPC use case Xem chi tiết film**

* Biểu đồ trình tự

****

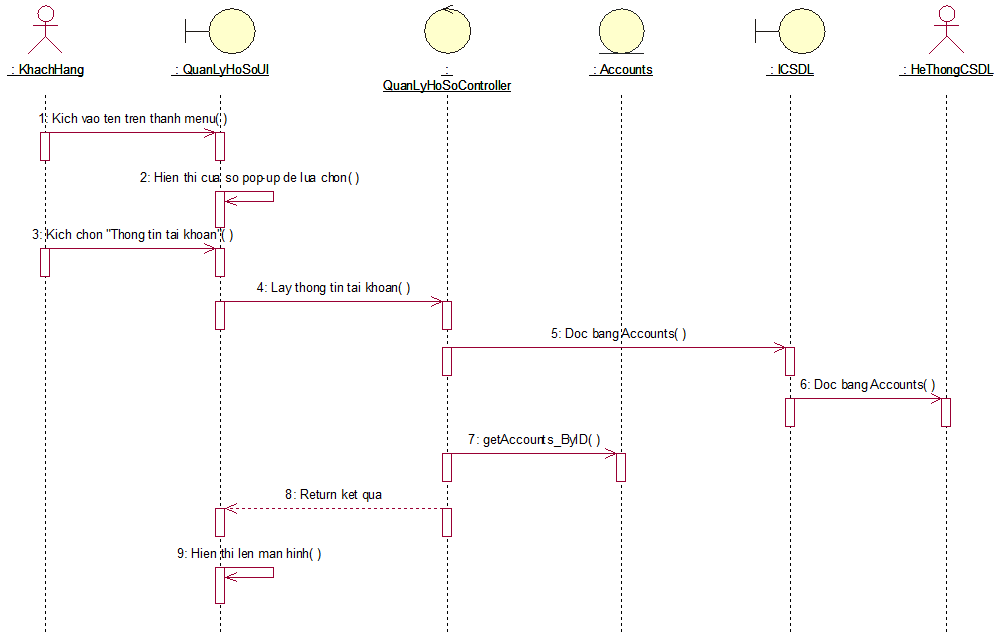
**Hình 26: Biểu đồ trình tự use case Xem chi tiết film**

* 1. **Phân tích use case Quản lý hồ sơ khách hàng**
* Biểu đồ VOPC

****

**Hình 27: Biểu đồ VOPC use case Quản lý hồ sơ khách hàng**

* Biểu đồ trình tự



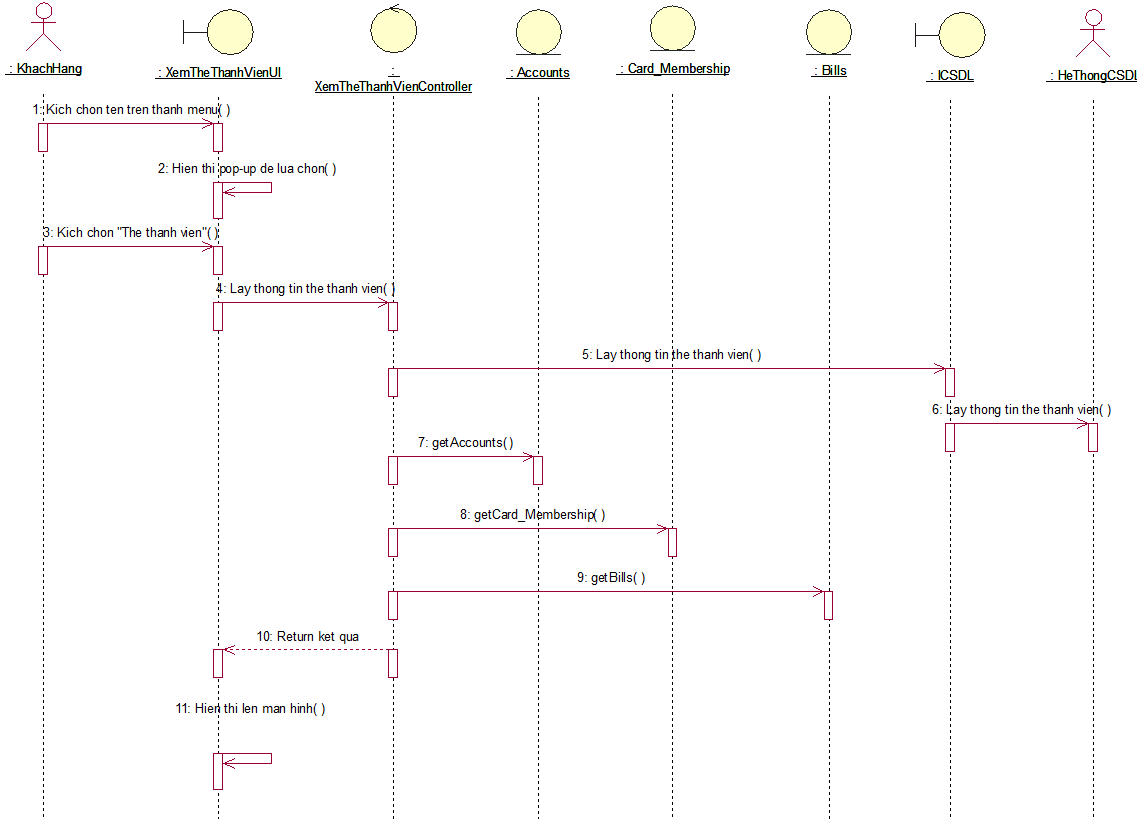
**Hình 28: Biểu đồ trình tự use case Quản lý hồ sơ khách hàng**

* 1. **Phân tích use case Xem thẻ thành viên**
* Biểu đồ VOPC



**Hình 29: Biểu đồ VOPC use case Xem thẻ thành viên**

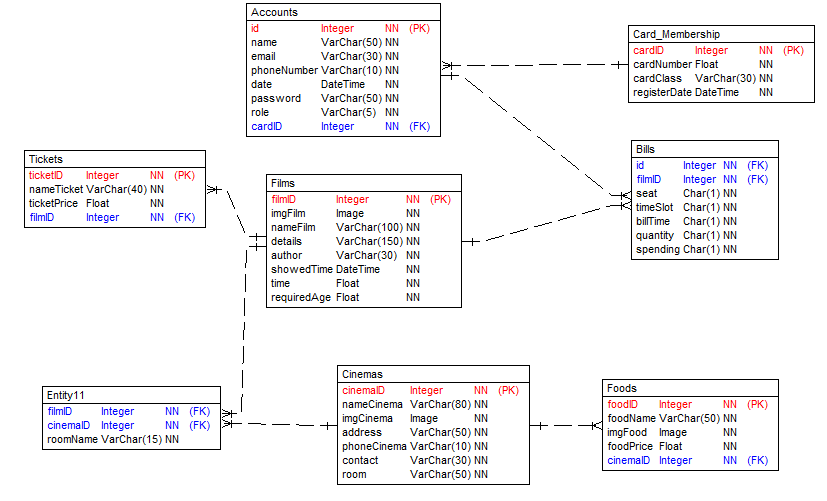
* Biểu đồ trình tự

****

**Hình 30: Biểu đồ trình tự use case Xem thẻ thành viên**

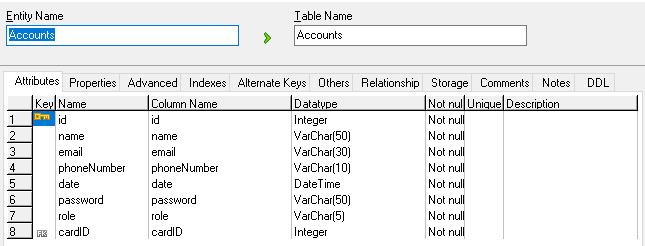
* 1. **Phân tích use case Xem lịch sử giao dịch**
* Biểu đồ VOPC
* Biểu đồ trình tự
  1. **Phân tích use case Đặt vé**
* Biểu đồ VOPC
* Biểu đồ trình tự

1. **Biểu đồ lớp phân tích**
2. **Thiết kế cơ sở dữ liệu**
   1. Biểu đồ mô tả cấu trúc cơ sở dữ liệu



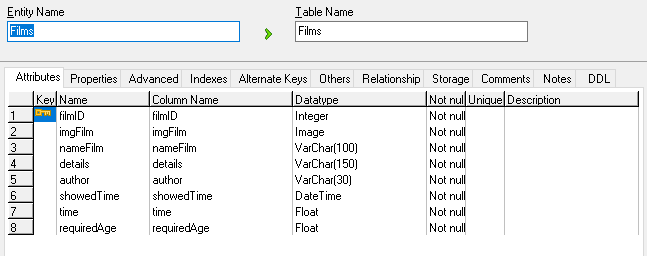
**Hình: Sơ đồ cấu trúc cơ sở dữ liệu hệ thống**

* 1. Mô tả cơ sở dữ liệu
* Bảng Account để lưu trữ thông tin tài khoản



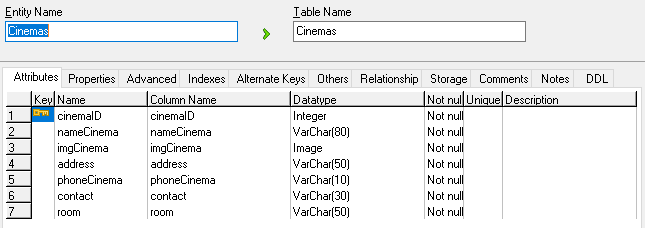
**Hình: Bảng Accounts**

* Bảng Films để lưu trữ thông tin film

****

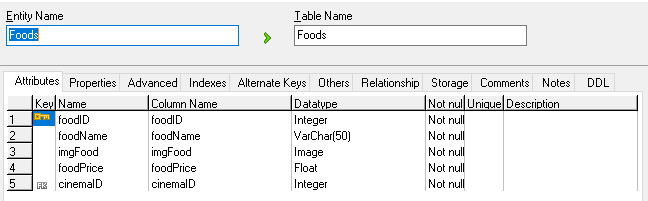
**Hình: Bảng Films**

* Bảng Cinemas để lưu trữ thông tin rạp chiếu film

****

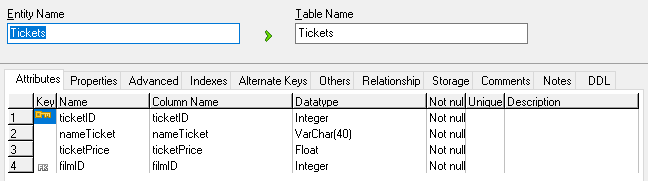
**Hình: Bảng Cinemas**

* Bảng Foods để lưu trữ thông tin đồ ăn được bán trong rạp

****

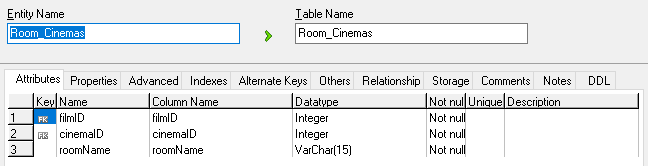
**Hình: Bảng Foods**

* Bảng Tickets để lưu trữ thông tin vé được bán trong rạp

****

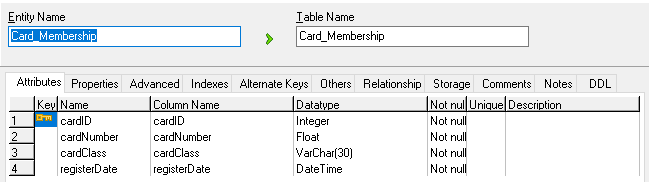
**Hình: Bảng Tickets**

* Bảng Room\_Cinemas để lưu trữ thông tin phòng chiếu trong rạp

****

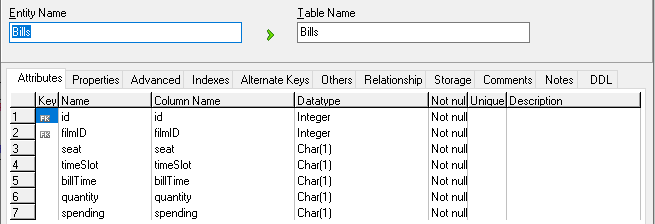
**Hình: Bảng Cinemas**

* Bảng Card\_Membership để lưu trữ thông tin thẻ thành viên



**Hình: Bảng Card\_Membership**

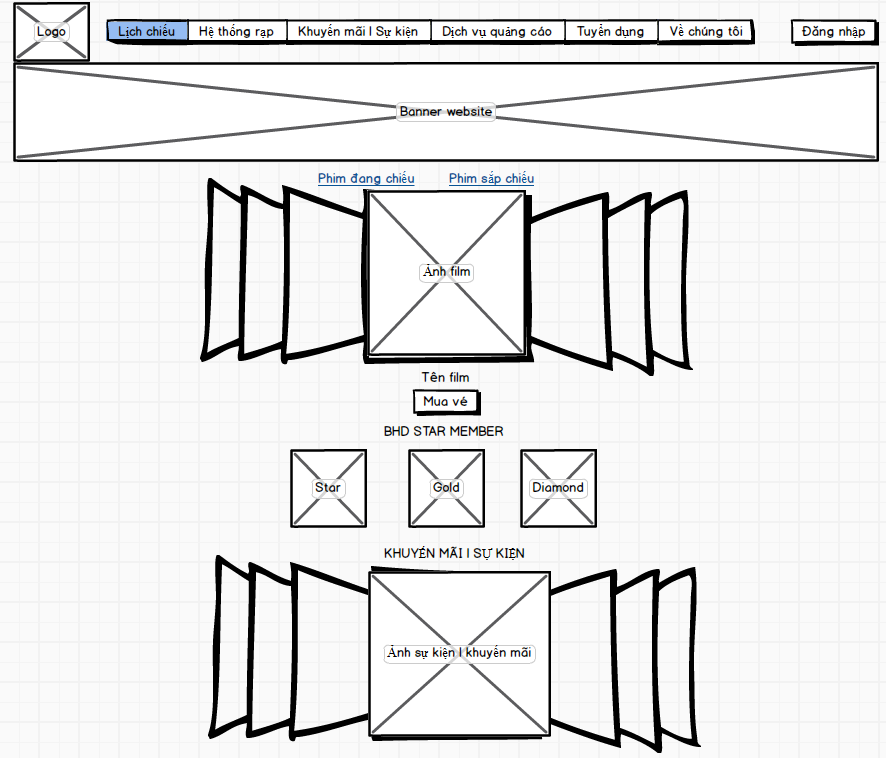
* Bảng Bills để lưu trữ thông tin hóa đơn sau khi khách hàng đặt vé

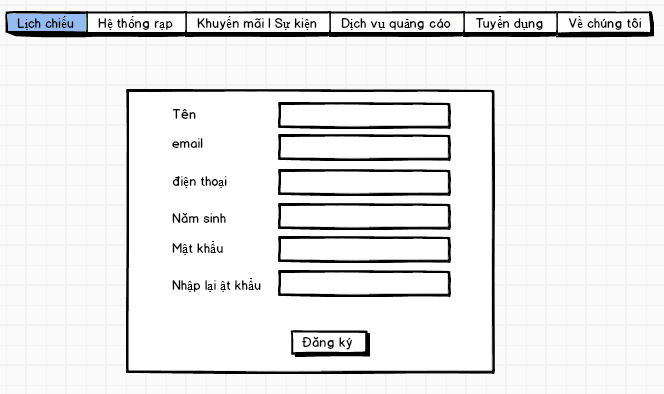
****

**Hình: Bảng Bills**

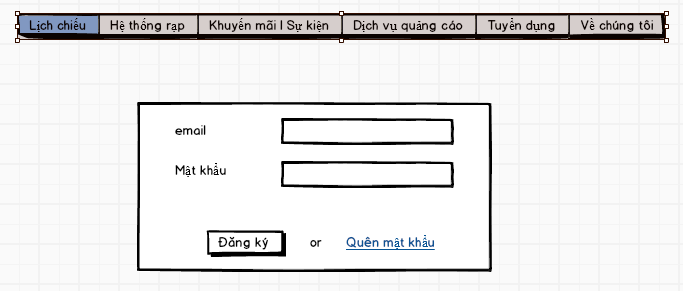
1. **Thiết kế giao diện**

Một số phác thảo giao diện website

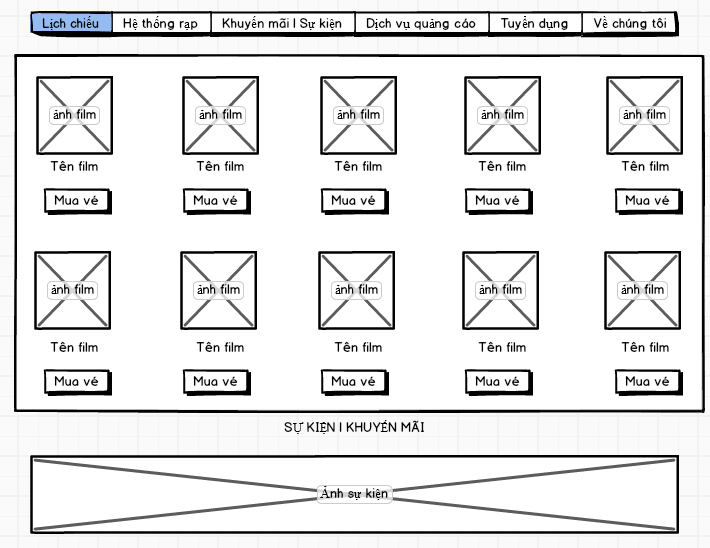
**Hình: Phác thảo giao diện trang chủ chính**



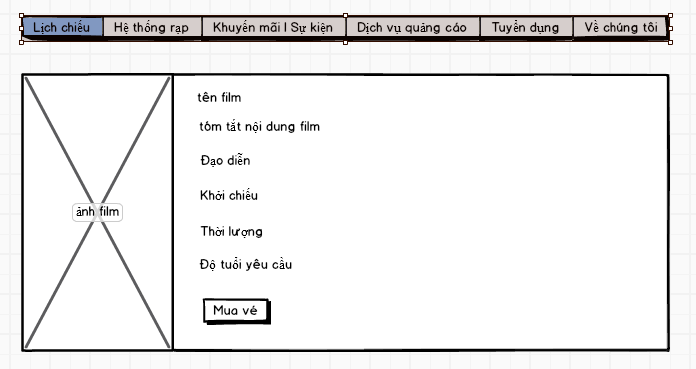
**Hình: Phác thảo giao diện trang đăng ký tài khoản**

****

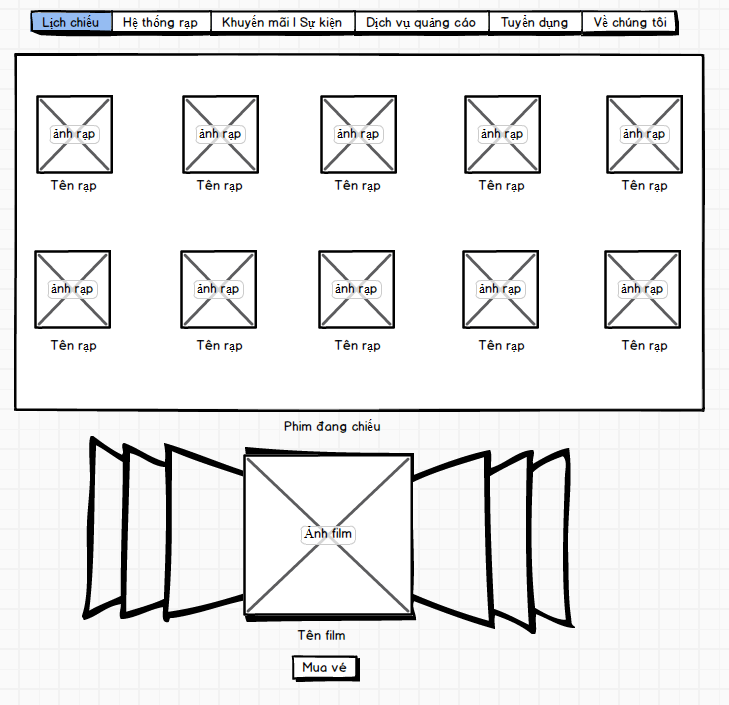
**Hình: Phác thảo giao diện trang đăng nhập tài khoản**

****

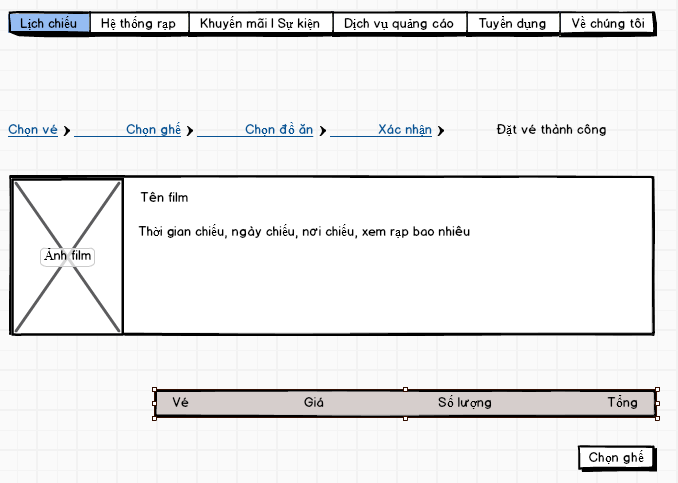
**Hình: Phác thảo giao diện trang danh sách film**

****

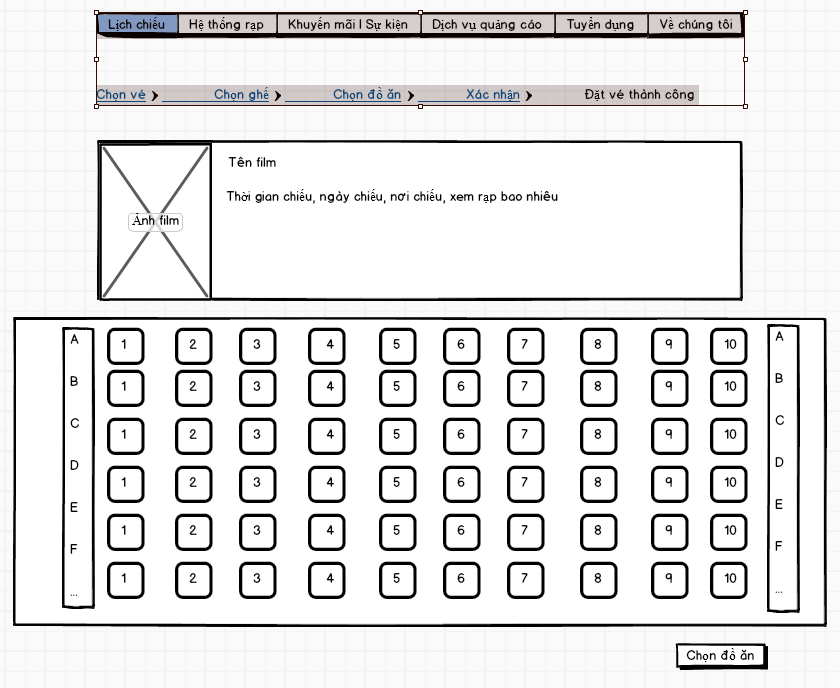
**Hình: Phác thảo giao diện trang chi tiết film**

****

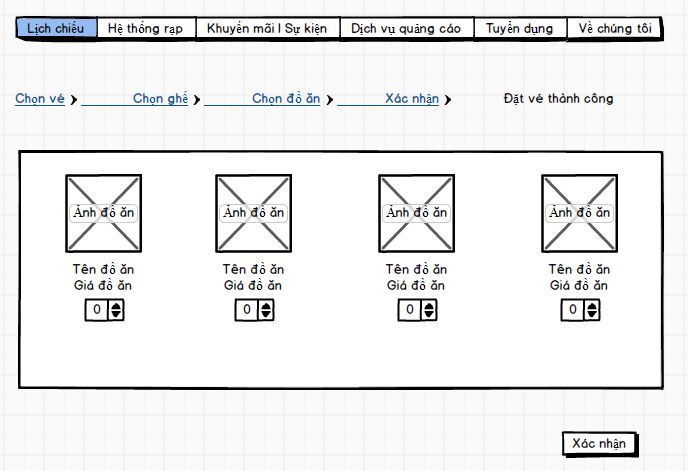
**Hình: Phác thảo giao diện trang danh sách rạp**

****

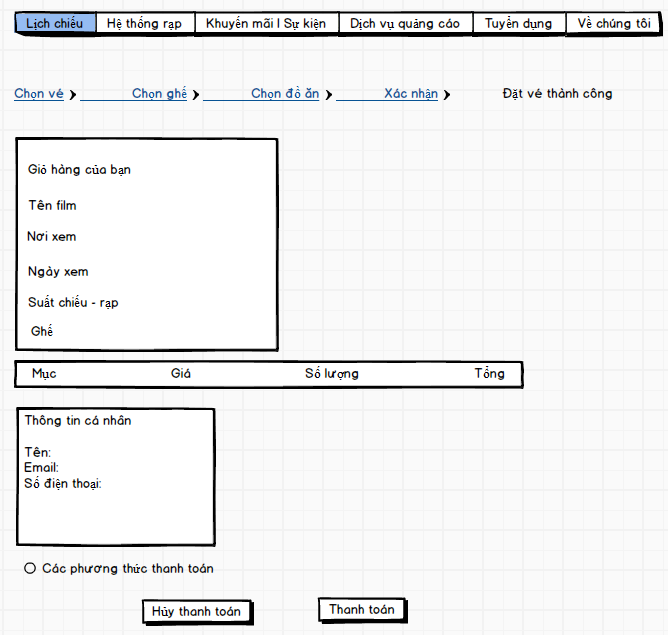
**Hình: Phác thảo giao diện trang chọn vé**

****

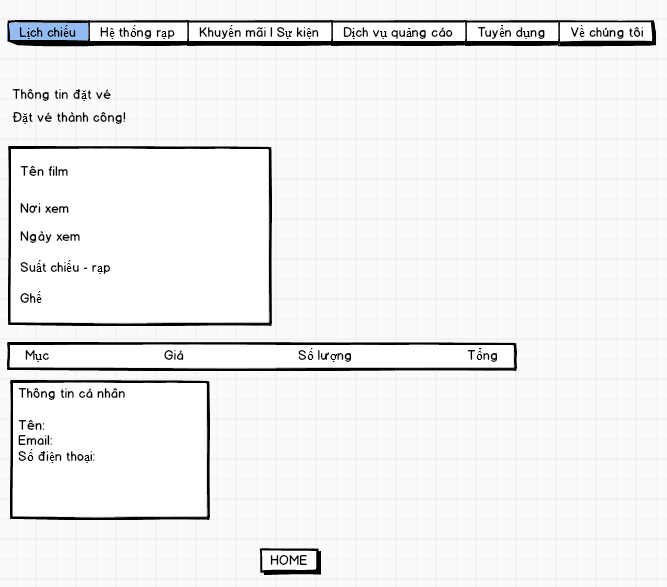
**Hình: Phác thảo giao diện trang chọn ghế**

****

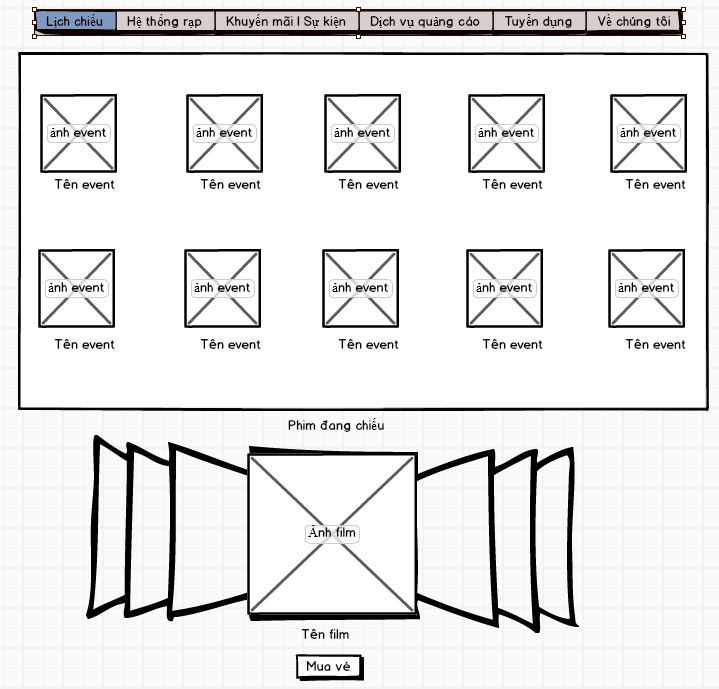
**Hình: Phác thảo giao diện trang chọn đồ ăn**

****

**Hình: Phác thảo giao diện trang xác nhận thành toán**

****

**Hình: Phác thảo giao diện trang thanh toán thành công**

****

**Hình: Phác thảo giao diện trang danh sách sự kiện**