# **CHƯƠNG I: KHẢO SÁT HỆ THỐNG**

## **Mô tả về môi trường hoạt động**

Ứng dụng máy tính trên Android được phát triển trong môi trường gồm bốn giai đoạn chính: phát triển, kiểm thử, triển khai, và sản xuất.

Trong **môi trường phát triển**, các ngôn ngữ lập trình chính là Java và Kotlin. Java mang lại sự ổn định và tương thích cao, trong khi Kotlin cung cấp nhiều tính năng hiện đại, giúp giảm thiểu lỗi và cải thiện hiệu suất. Framework Android SDK được sử dụng để phát triển ứng dụng với các công cụ hỗ trợ mạnh mẽ như Android Studio và Gradle. Phần giao diện được thiết kế bằng XML, sử dụng Material Design Framework để đảm bảo giao diện hiện đại và thân thiện với người dùng.

Trong **môi trường kiểm thử**, các công cụ như Espresso được sử dụng để kiểm tra các tương tác giao diện người dùng. JUnit hỗ trợ kiểm thử đơn vị để đảm bảo logic tính toán chính xác. Firebase Test Lab được sử dụng để kiểm tra tự động trên nhiều thiết bị và phiên bản Android khác nhau.

Khi triển khai, ứng dụng được đóng gói dưới dạng APK hoặc AAB và ký số để đảm bảo tính xác thực khi phát hành trên Google Play Store. Trước khi phát hành, ứng dụng được tối ưu hóa kích thước và hiệu suất để đảm bảo hoạt động tốt trên các thiết bị di động.

Trong môi trường sản xuất, ứng dụng hỗ trợ các thiết bị Android từ phiên bản 5.0 (Lollipop) trở lên. Firebase Crashlytics được tích hợp để ghi nhận lỗi và Firebase Analytics theo dõi hành vi người dùng nhằm cải thiện ứng dụng dựa trên dữ liệu thực tế.

Môi trường hoạt động của ứng dụng được thiết kế cẩn thận để tối ưu hóa từ giai đoạn phát triển đến sản xuất, đảm bảo ứng dụng hoạt động hiệu quả, ổn định và dễ bảo trì.

## **Khảo sát bài toán**

1. Giới thiệu về bài toán:

Ứng dụng máy tính Android được thiết kế để cung cấp công cụ tính toán nhanh chóng, chính xác và dễ sử dụng. Mục tiêu của ứng dụng là đáp ứng nhu cầu tính toán hàng ngày của người dùng, từ các phép tính cơ bản đến nâng cao. Ứng dụng phải đảm bảo hiệu suất cao, giao diện thân thiện và có khả năng mở rộng linh hoạt để tích hợp các tính năng mới khi cần.

1. Yêu cầu của người dùng:

* **Chính xác và nhanh chóng**: Người dùng cần thực hiện các phép tính không sai sót, kết quả trả về ngay lập tức.
* **Giao diện trực quan, dễ thao tác**: Giao diện đơn giản, các phím bấm rõ ràng, dễ dàng nhận biết.
* **Tích hợp lịch sử tính toán**: Người dùng có thể xem lại các phép tính trước đó.
* **Chế độ tối ưu trải nghiệm**: Chế độ tùy chỉnh sáng/tối để phù hợp với môi trường ánh sáng xung quanh.
* **Hỗ trợ đa ngôn ngữ**: Đặc biệt ưu tiên Tiếng Việt, giúp người dùng Việt Nam dễ dàng sử dụng.

1. Phân tích chức năng cần có:

* **Chức năng tính toán cơ bản:** Hỗ trợ cộng, trừ, nhân, chia. Xử lý ưu tiên toán tử dựa trên quy tắc **BODMAS** (ngoặc, số mũ, nhân/chia, cộng/trừ).
* **Chức năng tính toán nâng cao**: Hàm lượng giác: sin, cos, tan. Tính căn bậc hai, lũy thừa. Hỗ trợ số âm và số thập phân.
* **Quản lý lịch sử tính toán**: Tự động lưu kết quả mỗi phép tính. Cho phép người dùng xóa lịch sử.
* **Tùy chỉnh giao diện**: Chế độ tối/sáng. Cho phép thay đổi kích thước hoặc vị trí các phím số/toán tử.
* **Tính năng mở rộng**: Hỗ trợ nhập dữ liệu từ file hoặc sao chép kết quả ra clipboard.

## **So sánh giữa hệ thống đã có trên thị trường và hệ thống mới**

1. Giao diện người dùng và Trải nghiệm người dùng (UX/UI)

* Ứng dụng máy tính hiện tại:
* **Ưu điểm:** Giao diện quen thuộc, dễ sử dụng với người dùng ở mọi lứa tuổi. Tính năng được bố trí rõ ràng, phù hợp với các tác vụ cơ bản.
* **Nhược điểm:** Thiết kế giao diện có phần lỗi thời, thiếu tính hiện đại và chưa tối ưu hóa cho các thiết bị di động. Thiếu các tùy chỉnh giao diện cá nhân hóa theo sở thích người dùng.
* Ứng dụng máy tính mới:
* **Ưu điểm:**Giao diện hiện đại, tối giản, nhấn mạng trải nghiểm người dùng với bố cục rõ ràng và dễ đọc. Hỗ trợ tùy chỉnh giao diện theo sở thích và được tối ưu hóa cho cả máy tính lẫn smartphone.
* **Nhược điểm:** Vì là ứng dụng mới, người dùng cần thời gian để làm quen với giao diện và các tính năng mới. Quá trình cải thiện UX/UI dựa trên phản hồi của người dùng có thể mất thời gian.

1. Quản lý nội dung:

* Ứng dụng máy tính hiện tại:
* **Ưu điểm:** Các phép tính cơ bản (cộng, trừ, nhân, chia) được xử lý nhanh và chính xác. Giao diện đơn giản, dễ thao tác, phù hợp với các nhu cầu sử dụng hàng ngày.
* **Nhược điểm**: Chủ yếu chỉ hỗ trợ các phép tính cơ bản, chưa tích hợp nhiều tính năng nâng cao như tính toán khoa học hoặc xử lý biểu thức phức tạp. Thiếu tự động hóa và gợi ý thông minh trong việc sử dụng.
* Ứng dụng máy tính mới:
* **Ưu điểm:** Sử dụng công nghệ AI để hỗ trợ các phép tính phức tạp và tối ưu hóa quy trình xử lý. Có khả năng tự động gợi ý công thức, lưu trữ lịch sử phép tính, và hỗ trợ người dùng học các khái niệm toán học.
* **Nhược điểm:** Công nghệ AI cần thời gian để học và điều chỉnh theo nhu cầu thực tế của người dùng. Đòi hỏi phần cứng và tài nguyên xử lý tốt hơn so với các ứng dụng truyền thống.

1. Tính năng và Công nghệ

* Ứng dụng máy tính hiện tại:
* **Ưu điểm:** Ổn định, dễ sử dụng với các tính năng cơ bản đã được chứng minh qua thời gian. Không yêu cầu cấu hình phần cứng cao.
* **Nhược điểm:** Công nghệ cũ, thiếu các tính năng hiện đại như lưu trữ dữ liệu đám mây, cá nhân hóa, hoặc hỗ trợ đa nền tảng. Chưa tích hợp các công cụ tính toán nâng cao hoặc khả năng hiểu ngữ cảnh trong biểu thức.
* Ứng dụng máy tính mới:
* **Ưu điểm:** Tích hợp công nghệ AI, hỗ trợ các phép tính nâng cao và phân tích ngữ cảnh của biểu thức. Hỗ trợ tìm kiếm và lưu trữ các công thức, giúp người dùng học tập và làm việc hiệu quả hơn. Đồng bộ dữ liệu trên nhiều thiết bị, cho phép sử dụng linh hoạt.
* **Nhược điểm:** Việc triển khai công nghệ mới yêu cầu đầu tư về thời gian và chi phí. Các lỗi kỹ thuật ban đầu có thể ảnh hưởng đến trải nghiệm người dùng.

1. Khả năng mở rộng và phát triển

* Ứng dụng máy tính hiện tại:
* **Ưu điểm:** Đã có một lượng lớn người dùng trung thành với thói quen sử dụng ổn định. Cơ sở hạ tầng ổn định, ít yêu cầu nâng cấp thường xuyên.
* **Nhược điểm:** Thiếu khả năng mở rộng để tích hợp các tính năng mới hoặc công nghệ tiên tiến. Gặp khó khăn trong việc đổi mới do các ràng buộc về thiết kế và công nghệ.
* Ứng dụng máy tính mới:
* **Ưu điểm:** Khả năng mở rộng linh hoạt, dễ dàng tích hợp các công nghệ mới như AI, học máy, và lưu trữ đám mây. Có thể phát triển thêm các mô-đun hoặc tính năng tùy chỉnh theo nhu cầu của người dùng.
* **Nhược điểm:** Thách thức lớn nhất là xây dựng và duy trì lượng người dùng, đặc biệt trong giai đoạn đầu. Yêu cầu đầu tư tài nguyên đáng kể để đảm bảo hệ thống vận hành ổn định khi mở rộng.

## **Yêu cầu của đề tài**

1. Yêu cầu chức năng:

* Đăng nhập và quản lý người dùng:
* Admin có quyền đăng nhập để quản lý toàn bộ ứng dụng. Là người dùng có phân quyền cao nhất. Quản lý quyền hạn của các người dùng khác, ví dụ: phân quyền tác giả hoặc quản trị nội dung.
* Người dùng có thể đăng nhập để lưu trữ lịch sử tính toán cá nhân. Hỗ trợ tính năng cá nhân hóa giao diện và cài đặt.
* Các tính năng tính toán:
* Phép tính cơ bản: Cộng, trừ, nhân, chia. Hỗ trợ xử lý số thập phân và số âm.
* Phép tính nâng cao (Scientific): Lũy thừa, căn bậc hai, logarit, hàm lượng giác (sin, cos, tan). Hỗ trợ tính toán biểu thức phức tạp theo thứ tự ưu tiên toán học.
* Quản lý lịch sử tính toán: Ghi lại toàn bộ phép tính đã thực hiện. Cho phép người dùng tìm kiếm hoặc xóa lịch sử theo thời gian.
* Tìm kiếm và gợi ý thông minh: Gợi ý cách giải hoặc tính toán thông qua nhập liệu. Hỗ trợ tìm kiếm thông minh theo từ khóa liên quan (ví dụ: "làm thế nào để tính logarit?").
* Quản lý giao diện và tính năng cá nhân hóa:
* Thiết kế giao diện: Giao diện tối giản, hiện đại, phù hợp với mọi đối tượng người dùng. Chế độ "Dark Mode" và "Light Mode".
* Cá nhân hóa: Cho phép người dùng tự điều chỉnh bố cục giao diện theo sở thích. Lưu trữ và đồng bộ hóa cài đặt cá nhân qua tài khoản.
* Chia sẻ và tương tác người dùng:
* Chia sẻ kết quả: Người dùng có thể chia sẻ kết quả tính toán qua các ứng dụng khác như email, mạng xã hội, hoặc tin nhắn.
* Đánh giá ứng dụng: Cung cấp tính năng gửi phản hồi và đánh giá từ người dùng về trải nghiệm ứng dụng.

1. Yêu cầu về giao diện người dùng:

* Thân thiện và trực quan:
* Giao diện thiết kế đơn giản, dễ thao tác, không gây rối mắt.
* Hiển thị nút rõ ràng, tối ưu hóa cho cả màn hình dọc và ngang.
* Tương thích trên nhiều thiết bị:
* Ứng dụng tương thích với mọi kích thước màn hình (điện thoại, máy tính bảng).
* Đáp ứng nhanh (Responsive Design) với thao tác cảm ứng.
* Tương tác dễ dàng:
* Phản hồi trực tiếp khi người dùng thực hiện thao tác, như nhấn nút hoặc nhập biểu thức.
* Cung cấp thông báo lỗi rõ ràng khi người dùng nhập sai biểu thức.

## **Công cụ lập trình**

* **Ngôn ngữ**: Java hoặc Kotlin.
* **Framework**: Android SDK.
* **IDE**: Android Studio.
* **Thiết kế giao diện**: XML.
* **Thư viện hỗ trợ**: Material Design, Room Database (quản lý lịch sử tính toán).

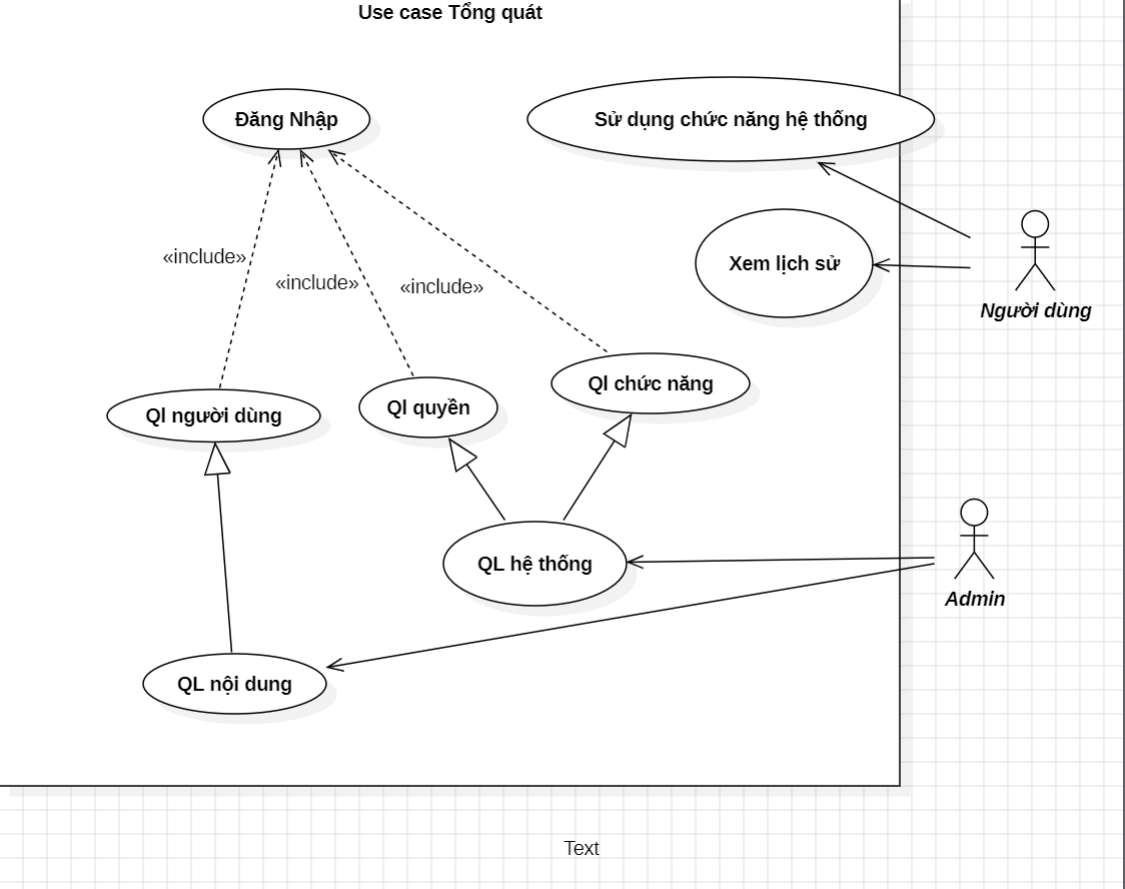
# **CHƯƠNG II: PHÂN TÍCH HỆ THỐNG**

## **Tổng quan chức năng**

### **Xây dựng biểu đồ usecase tổng quát**

* Danh sách Tác nhân người sử dụng hệ thống:

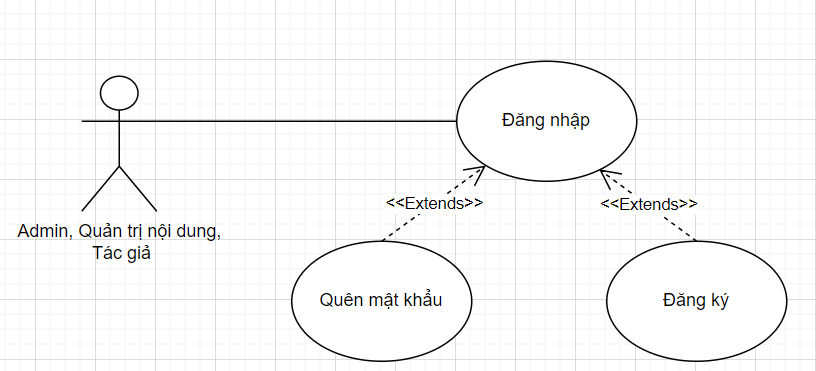
|  |  |  |
| --- | --- | --- |
| **Ký hiệu** | **Tác nhân** | **Mô tả chức năng** |
|  | **Admin** | * Quyền đăng nhập, truy cập: * Có quyền truy cập cao nhất để quản lý và duy trì hệ thống. * Quản lý nội dung: * Quản lý thể loại * Quản lý danh mục * Quản lý hệ thống: * Quản lý người dùng * Quản lý vai trò * Quản lý quyền |
| **Quản trị nội dung** | * Đăng nhập, đăng ký * Quản lý nội dung: * Quản lý danh mục |
| **Tác giả** | * Đăng nhập, đăng ký * Quản lý nội dung |
| Người dùng | * Tính toán |



Hình 2.1. Xây dựng biểu đồ usecase tổng quát

### **Xây dựng biểu đồ usecase phân rã**

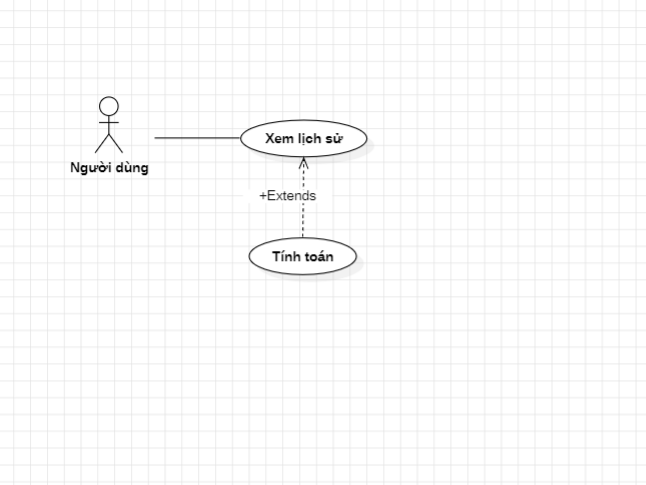
#### **Biểu đồ usecase phân rã Đăng nhập:**



Hình 2.2. Xây dựng biểu đồ usecase phân rã Đăng nhập

|  |  |
| --- | --- |
| **Usecase** | * Đăng nhập hệ thống |
| **Actor** | * Admin, Quản trị nội dung, Tác giả |
| **Mô tả** | * Use case này cho phép người dùng đăng nhập vào hệ thống bằng cách nhập tên tài khoản và mật khẩu. |
| **Điều kiện đầu vào** | * Hệ thống ở trạng thái đăng nhập và có màn hình đăng nhập được hiển thị. * Người dùng hoặc quản trị viên phải có tài khoản hợp lệ trong hệ thống. |
| **Điều kiện hậu nghiệm** | * Nếu thông tin đăng nhập hợp lệ, người dùng sẽ được chuyển đến trang chính của hệ thống với quyền truy cập tương ứng. * Nếu thông tin đăng nhập không hợp lệ, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu người dùng nhập lại thông tin. |
| **Luồng sự kiện chính** | 1. **Bắt đầu**: Use case bắt đầu khi người dùng mở trang đăng nhập hệ thống hiển thị form đăng nhập. 2. Người dùng nhập tài khoản và mật khẩu vào form đăng nhập. 3. Người dùng nhấn nút "Đăng nhập". 4. Hệ thống kiểm tra tính hợp lệ của thông tin đăng nhập (tên tài khoản và mật khẩu). 5. Hệ thống xác nhận thông tin đăng nhập với cơ sở dữ liệu. 6. Nếu thông tin đăng nhập hợp lệ:  * Hệ thống xác định quyền truy cập của người dùng. * Hệ thống điều hướng người dùng đến trang chính với quyền truy cập tương ứng.  1. **Kết thúc**: Use case kết thúc khi người dùng được đăng nhập thành công và điều hướng đến trang chính. |
| **Luồng sự kiện phụ** | **Luồng sự kiện phụ 1: Actor nhập thiếu thông tin đăng nhập**   * Hệ thống sẽ thông báo lỗi (bạn chưa nhập đủ thông tin đăng nhập ) và quay lại bước 2   **Luồng sự kiện phụ 2: Actor nhập sai thông tin đăng nhập**   1. Nếu tài khoản không tồn tại hoặ **Điểm mở rộng** c mật khẩu sai:  * Hệ thống hiển thị thông báo lỗi "Tên tài khoản hoặc mật khẩu không đúng".   Use case quay lại bước 2 của luồng chính |
| **Điểm mở rộng** | **Mở rộng 1: Người dùng có thể đăng ký tài khoản khi chưa có tài khoản đăng nhập**   * Người dùng chọn yêu cầu đăng ký * Hệ thống hiển thị form đăng ký * Người dùng nhập đầy đủ thông tin vào form đăng ký và ấn nút “đăng ký”   **Mở rộng 2: Người dùng có thể ấn quên mật khẩu và yêu cầu tạo lại mật khẩu mới**   * Người dùng ấn vào mục quên mật khẩu * Hệ thống hiển thị form điền thông tin và mục điền mật khẩu mới   Người dùng nhập đầy đủ thông tin |
| **Điều kiện đầu ra** | **1. Đăng nhập thành công:**   * **Hành động hệ thống**: * Chuyển hướng người dùng đến trang chính   **2. Đăng nhập thất bại (Tên đăng nhập hoặc mật khẩu sai):**   * **Thông điệp**: "Tên tài khoản hoặc mật khẩu không đúng." * **Hành động hệ thống**: * Hiển thị thông báo lỗi trên trang đăng nhập.   Cho phép người dùng nhập lại thông tin đăng nhập. |

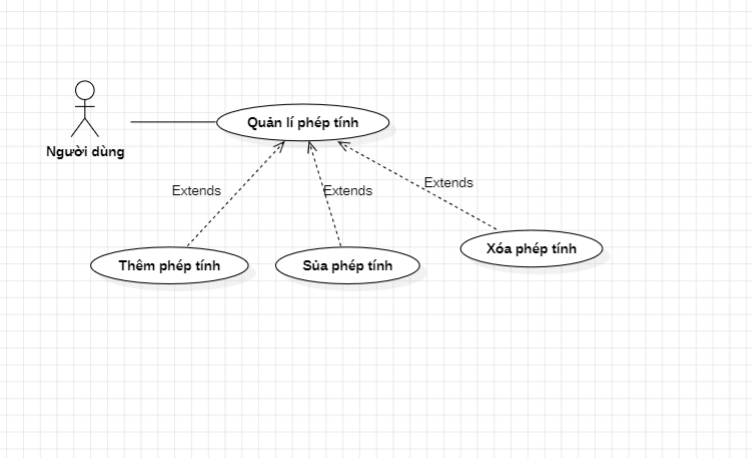
#### **Biểu đồ usecase phân rã Xem lịch sử tính:**



Hình 2.3. Biểu đồ phân rã usecase Xem lịch sử

|  |  |
| --- | --- |
| **Usecase** | * Xem lịch sử tính |
| **Actor** | * Người dùng |
| **Mô tả** | * Use case này cho phép người đọc truy cập và xem lịch sử các phép tính đã tính trên hệ thống |
| **Điều kiện đầu vào** | * Ca sử dụng bắt đầu khi người dùng bắt đầu truy cập vào ứng dụng * Hệ thống phải có sẵn các chức năng * Người dùng không cần thiết bị kết nối internet để truy cập ứng dụng |
| **Điều kiện hậu nghiệm** | * Người dung có thể xem lại được lịch sử các phép tính mà mình đã tính toán * Hệ thống cập nhật lịch sử tính toán của người dùng |
| **Luồng sự kiện chính** | 1. **Bắt đầu**: Use case bắt đầu khi người dùng mở ứng dụng 2. Hệ thống hiển thị các chức năng chính của ứng dụng. 3. Người dùng chọn một số chức năng để tính toán. 4. **Kết thúc**: Use case kết thúc khi người dùng đã tính toán xong |
| **Luồng sự kiện phụ** | **Luồng sự kiện phụ 2: Người dùng xem lại lịch sử tính**   1. Người dùng chọn chức năng xem lịch sử để hiển thị lịch sử tính 2. Hệ thống hiển thị danh sách lịch sử các phép tính |
| **Điều kiện đầu ra** | 1. **Điều kiện đầu ra thành công**:  * Người dùng xem được lịch sử tính  1. **Điều kiện đầu ra không thành công**:  * Nếu hệ thống gặp lỗi khi cho người dùng xem lịch sử tính: * **Thông điệp**: "Có lỗi xảy ra khi tải lịch sử.Vui lòng thử lại sau." * **Hành động hệ thống**: Hiển thị thông báo lỗi và cho phép người dùng thử lại. |

#### **Biểu đồ usecase phân rã Quản lý phép tính:**

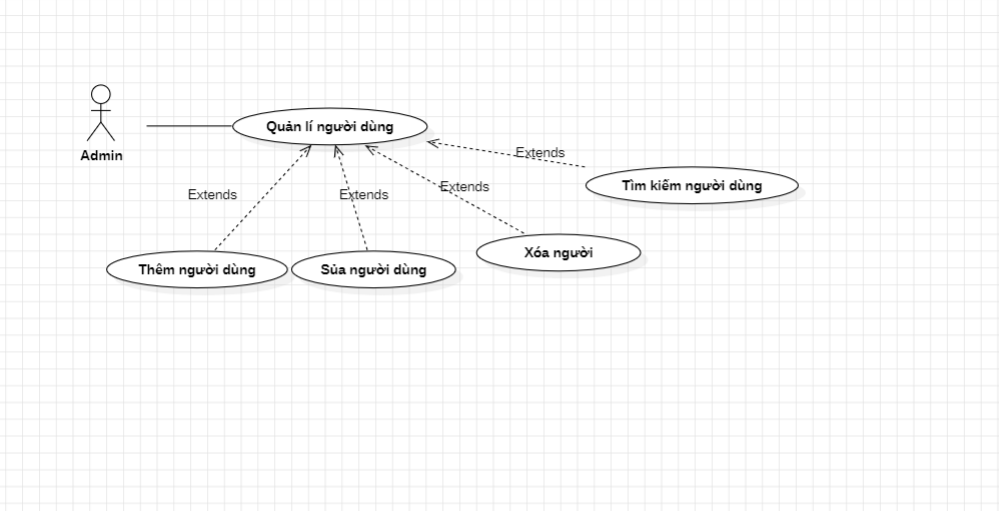


Hình 2.5. Biểu đồ phân rã usecase Quản lý phép tính

* Đặc tả use case Quản lý phép tính:

|  |  |
| --- | --- |
| **Usecase** | * Quản lý phép tính |
| **Actor** | * Người dùn |
| **Mô tả** | * Use case này cho phép người dùng quản lý phép tính * Người dùng có thể thêm xóa sủa phép tính |
| **Điều kiện đầu vào** | * Người dùng phải đăng nhập vào ứng dụng và có quyền truy cập để quản lí phép tính |
| **Điều kiện hậu nghiệm** | * Các phép tín phải được thêm vào lịch sử tính |
| **Luồng sự kiện chính** | 1. Bắt đầu: Use case bắt đầu khi Người dùng mở trang ứng dụng. 2. Hệ thống hiển thị danh sách các chức năng hệ thống hiện có. 3. Người dùng chọn thao tác muốn thực hiện:  * Thêm phép tính * Sửa phép tính * Xóa phép tính * Thêm phép tính mới:  1. Người dùng chọn “Thêm phép tính” 2. Hệ thống hiển phép tính mới  * Chỉnh sửa phép tính hiện có:  1. Người dùng chọn phép tính cần sửa. 2. Hệ thống hiển thị nút sửa. 3. Hệ thống tiến hành sủa phép tính  * Xóa phép tính:  1. Người dùng chọn phép tính cần xóa. 2. Hệ thống hiển thị nút xóa. 3. Hệ thống xóa phép tính khỏi ứng dụng. 4. Kết thúc: Use case kết thúc khi Actor hoàn thành thao tác quản lý phép tính. |
| **Điều kiện đầu ra** | 1. **Điều kiện đầu ra thành công**:  * Thêm phép tính: Phép tính mới được lưu lịch sử * Chỉnh sửa phép tính: Phép tính được cập nhật với thông tin mới và hiển thị trên ứng dụng. * Xóa bài đăng: Bài đăng được xóa khỏi cơ sở dữ liệu và không còn hiển thị trên lịch sử |

#### **Biểu đồ usecase phân rã Quản lý người dùng:**



Hình 2.8. Biểu đồ phân rã usecase Quản lý người dùng

* Đặc tả use case Quản lý người dùng:

|  |  |
| --- | --- |
| **Usecase** | * Quản lý người dùng |
| **Actor** | * Admin |
| **Mô tả** | * Use case này cho phép actor quản lý người dùng trên hệ thống ứng dụng. Các thao tác bao gồm xem và tìm kiếm người dùng, thêm người dùng, chỉnh sửa người dùng hiện có và xóa người dùng không cần thiết. |
| **Điều kiện đầu vào** | * Actor phải đăng nhập vào hệ thống và có quyền truy cập để quản lý người dùng. |
| **Điều kiện hậu nghiệm** | * Người dùng mới được thêm vào hệ thống. * người dùng hiện có được cập nhật với thông tin mới. * Người dùng không cần thiết được xóa khỏi hệ thống. |
| **Luồng sự kiện chính** | 1. **Bắt đầu**: Use case bắt đầu khi Actor mở trang quản lý người dùng. 2. Hệ thống hiển thị danh sách các người dùng hiện có. 3. Actor chọn thao tác muốn thực hiện:  * Thêm người dùng mới * Chỉnh sửa người dùng hiện có * Xóa người dùng * **Thêm người dùng** **mới**:  1. Actor chọn "Thêm người dùng mới". 2. Hệ thống hiển thị form để nhập thông tin người dùng mới. 3. Actor nhập thông tin người dùng (tên người dùng, tên tài khoản, mật khẩu, Xác nhận mật khẩu, vai trò, email, số điện thoại) 4. Actor nhấn "Lưu". 5. Hệ thống kiểm tra tính hợp lệ của thông tin và lưu người dùng mới vào cơ sở dữ liệu.  * **Chỉnh sửa người dùng hiện có**:  1. Actor chọn người dùng cần chỉnh sửa. 2. Hệ thống hiển thị thông tin chi tiết của người dùng. 3. Actor cập nhật thông tin cần thiết. 4. Actor nhấn "Lưu". 5. Hệ thống kiểm tra tính hợp lệ của thông tin và cập nhật người dùng trong cơ sở dữ liệu.  * **Xóa người dùng**:  1. Actor chọn người dùng cần xóa. 2. Hệ thống hiển thị thông báo xác nhận xóa. 3. Actor xác nhận xóa người dùng. 4. Hệ thống xóa người dùng khỏi cơ sở dữ liệu. 5. **Kết thúc**: Use case kết thúc Actor hoàn thành thao tác quản lý người dùng. |
| **Luồng sự kiện phụ** | **Luồng sự kiện phụ 1: Thông tin người dùng không hợp lệ khi thêm hoặc chỉnh sửa**   1. Hệ thống hiển thị thông báo lỗi tương ứng với trường thông tin có dữ liệu không hợp lệ |
| **Điều kiện đầu ra** | 1. **Điều kiện đầu ra thành công**:  * **Thêm người dùng mới**: người dùng mới được lưu vào cơ sở dữ liệu và hiển thị trên trang tin tức. * **Chỉnh sửa người dùng**: người dùng được cập nhật với thông tin mới và hiển thị trên trang tin tức. * **Xóa người dùng**: người dùng được xóa khỏi cơ sở dữ liệu và không còn hiển thị trên trang tin tức.  1. **Điều kiện đầu ra không thành công**:  * **Thông tin người dùng không hợp lệ**: Hệ thống hiển thị thông báo lỗi và yêu cầu tác nhân nhập lại thông tin. * **Lỗi hệ thống khi thêm, chỉnh sửa hoặc xóa thể loại**: * **Hành động hệ thống**: Hiển thị thông báo lỗi và không lưu thay đổi vào cơ sở dữ liệu. |