TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN &

TRUYỀN THÔNG VIỆT HÀN

**Khoa Khoa Học Máy Tính**



ĐỒ ÁN CHUYÊN NGÀNH 1

**ỨNG DỤNG QUẢN LÝ GHI CHÚ**

**ĐA NỀN TẢNG**

Sinh viên thực hiện: **TRẦN THỊ HẠNH HÒA**

Lớp: **19IT3**

Giảng viên hướng dẫn: TS. NGUYỄN SĨ THÌN

Đà Nẵng, tháng 12 năm 2022

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN &

TRUYỀN THÔNG VIỆT HÀN

**Khoa Khoa Học Máy Tính**

ĐỒ ÁN CHUYÊN NGÀNH 1

**ỨNG DỤNG** **QUẢN LÝ GHI CHÚ**

**ĐA NỀN TẢNG**

Sinh viên thực hiện: **TRẦN THỊ HẠNH HÒA**

Lớp: **19IT3**

Giảng viên hướng dẫn: TS. NGUYỄN SĨ THÌN

Đà Nẵng, tháng 12 năm 2022

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

# 

# 

# 

# 

# 

# 

# 

LỜI CẢM ƠN

Em xin gửi lời cảm ơn chân thành đến TS. NGUYỄN SĨ THÌN đã dẫn dắt, góp ý trong suốt quá trình thực hiện đề tài. Tuy nhiên trong quá trình xây dựng ứng dụng không tránh khỏi sai sót kính mong các thầy cô và các bạn đóng góp ý kiến xây dựng để em hoàn thiện hơn.

*Sinh viên*

Trần Thị Hạnh Hoà

MỤC LỤC

[MỞ ĐẦU 1](#_Toc121088716)

[Chương 1. TỔNG QUAN VỀ ỨNG DỤNG QUẢN LÝ GHI CHÚ ĐA NỀN TẢNG 2](#_Toc121088717)

[**1.1. Lý do chọn đề tài 2**](#_Toc121088718)

[**1.2. Mục tiêu và nhiệm vụ 2**](#_Toc121088719)

[**1.3. Đối tượng nghiên cứu và phạm vi hệ thống 2**](#_Toc121088720)

[**1.4. Phương pháp nghiên cứu 2**](#_Toc121088721)

[**2.1 Giới thiệu .NET MAUI 3**](#_Toc121088722)

[**2.1.1 .NET MAUI là gì? 3**](#_Toc121088723)

[**2.1.2 .NET MAUI và hệ sinh thái .NET 4**](#_Toc121088724)

[**2.1.3 Mô hình phát triển .NET MAUI 6**](#_Toc121088725)

[**2.2 Tổng quan về Oracle 8**](#_Toc121088726)

[**2.2.1 Orancle là gì? 8**](#_Toc121088727)

[**2.2.2 Cấu trúc của cơ sở dữ liệu Oracle 8**](#_Toc121088728)

[**2.2.3 Tính năng và tuỳ chọn tiêu biểu của Orancle 8**](#_Toc121088729)

[CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 10](#_Toc121088730)

[**3.1 Mô tả về chương trình 10**](#_Toc121088731)

[**3.1.1 Yêu cầu chức năng đối với hệ thống 10**](#_Toc121088732)

[**3.1.2 Yêu cầu phi chức năng đối với hệ thống: 11**](#_Toc121088733)

[**3.2 . Thiết kế hệ thống 11**](#_Toc121088734)

[**3.2.1 Xác định các tác nhân của hệ thống 11**](#_Toc121088735)

[**3.2.2 Danh sách các Use case 12**](#_Toc121088736)

[**3.2.3 Biểu đồ tuần tự cho các UC (Sequence Diagram) 14**](#_Toc121088737)

[**3.2.3 Sơ đồ trạng thái (State Machine Diagram) 18**](#_Toc121088738)

[**3.2.4 Thiết kế cơ sở dữ liệu 19**](#_Toc121088739)

[**3.2.5 Biểu đồ lớp chi tiết 20**](#_Toc121088740)

[CHƯƠNG 4: XÂY DỰNG ỨNG DỤNG 21](#_Toc121088741)

[**4.1. Giao diện Desktop 21**](#_Toc121088742)

[**4.1. Giao diện Mobile 26**](#_Toc121088743)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 31](#_Toc121088744)

[Tài Liệu Tham Khảo 31](#_Toc121088745)

Danh sách hình ảnh

[*Hình 1 Kiến trúc của ứng dụng .NET MAUI 3*](#_Toc121086736)

[*Hình 2 Kiến trúc của ứng dụng .NET MAUI 5*](#_Toc121086737)

[*Hình 3 Mẫu MVVM. Mô hình đại diện cho vấn đề mà ứng dụng của các nhà phát triển giải quyết và bao gồm các thực thể và dịch vụ 6*](#_Toc121086738)

[*Hình 4 Mẫu MVU, Model đại diện cho trạng thái toàn bộ ứng dụng 7*](#_Toc121086739)

[*Hình 5 Biểu đồ tuần tự đăng ký tài khoản 14*](#_Toc121086740)

[*Hình 6 Biểu đồ tuần tự đăng nhập hệ thống 14*](#_Toc121086741)

[*Hình 7 Biểu đồ tuần tự quản lý tập ghi chú 15*](#_Toc121086742)

[*Hình 8 Biểu đồ tuần tự quản lý ghi chú 16*](#_Toc121086743)

[*Hình 9 Biểu đồ tuần tự quản lý TodoList 17*](#_Toc121086744)

[*Hình 10 Biểu đồ tuần tự lập lịch Schedule 18*](#_Toc121086745)

[*Hình 11 Biểu đồ lớp chi tiết 20*](#_Toc121086746)

[*Hình 12 Giao diện màn hình đăng ký tài khoản 21*](#_Toc121086747)

[*Hình 13 Giao diện màn hình đăng nhập hệ thống 21*](#_Toc121086748)

[*Hình 14 Giao diện màn hình màn hình trang chủ 22*](#_Toc121086749)

[*Hình 15 Giao diện màn hình thông tin tài khoản 22*](#_Toc121086750)

[*Hình 16 Giao diện màn hình quản lý tập ghi chú 23*](#_Toc121086751)

[*Hình 17 Giao diện màn hình quản lý ghi chú 23*](#_Toc121086752)

[*Hình 18 Giao diện màn hình ghi chú yêu thích 24*](#_Toc121086753)

[*Hình 19 Giao diện màn hình ghi chú yêu thích 24*](#_Toc121086754)

[*Hình 20 Giao diện màn hình lập Schedule 25*](#_Toc121086755)

[*Hình 21 Giao diện màn hình trang giới thiệu 26*](#_Toc121086756)

[*Hình 22 Giao diện màn hình đăng ký tài khoản 26*](#_Toc121086757)

[*Hình 23 Giao diện màn hình đăng nhập hệ thống 27*](#_Toc121086758)

[*Hình 24 Giao diện màn hình trang chủ 27*](#_Toc121086759)

[*Hình 25 Giao diện màn hình thông tin ứng dụng 28*](#_Toc121086760)

[*Hình 26 Giao diện màn hình quản lý tập ghi chú 28*](#_Toc121086761)

[*Hình 27 Giao diện màn hình ghi chú yêu thích 29*](#_Toc121086762)

[*Hình 28 Giao diện màn hình quản lý ghi chú 29*](#_Toc121086763)

[*Hình 29 Giao diện màn hình lập Schedule 30*](#_Toc121086764)

**Danh sáchBảng**

[*Bảng 1 Bảng cơ sở dữ liệu cá nhân User 19*](#_Toc121088136)

[*Bảng 2 Bảng cơ sở dữ liệu tài khoản User 19*](#_Toc121088137)

[*Bảng 3 Bảng cơ sở dữ liệu notebook 19*](#_Toc121088138)

[*Bảng 4 Bảng cơ sở dữ liệu note 19*](#_Toc121088139)

[*Bảng 5 Bảng cơ sở dữ liệu TodoList 19*](#_Toc121088140)

[*Bảng 6 Bảng cơ sở dữ liệu Schechule 20*](#_Toc121088141)

[*Bảng 7 Bảng cơ sở dữ liệu NoteWithImg 20*](#_Toc121088142)

[*Bảng 8 Bảng cơ sở dữ liệu Img 20*](#_Toc121088143)

# **MỞ ĐẦU**

1. Giới thiệu

Trong cuộc sống bận rộng của thế kỷ 21 có thể nói việc note lại các công việc cần làm là đều không thể thiếu trong mỗi con người. Với sự phát triển của công nghệ hiện nay thì hầu như smart phone và laptop đã thay thế giấy bút. Các ứng dụng ra đời nhằm phục vụ cho việc sắp xếp công việc, ghi chú lại những công việc cần làm, những điều quan trọng, ..dần dần ra đời. Một trong số đó phải kể đến ghi chú online. Chính vì vậy em đã chọn đề tài “Ứng dụng ghi chú đa nền tảng” để giúp việc ghi chú trở nên thuận lợi hơn với đa nền tảng sử dụng.

2. Bố cục báo cáo

Sau phần *Mở đầu*, báo cáo được trình bày trong ba chương, cụ thể như sau:

Chương 1. *Tổng quan về ứng dụng quản lý ghi chú đa nền tảng*

Chương 2. *Cơ sở lý thuyết*

Chương 2. *Phân tích và thiết kế hệ thống*

Chương 3. *Xây dựng ứng dụng*

Cuối cùng là *Kết luận*, *Tài liệu tham khảo* liên quan đến đề tài.

# 

# **Chương 1. TỔNG QUAN VỀ ỨNG DỤNG QUẢN LÝ GHI CHÚ ĐA NỀN TẢNG**

* 1. **Lý do chọn đề tài**

Khi mà thời đại công nghệ phát triển để bắt kịp thời đại ở Việt Nam hiện nay. Công nghệ Thông tin cũng đã được nghiên cứu và ứng dụng rộng rãi trong nhiều lĩnh vực, đặc biệt là trong lĩnh vực phần mềm tay lập trình ứng dụng, thì học đã bỏ thành nối công cụ không thể thiếu.

Sau quá trình tìm hiểu em nhận thấy có thể phát triển một chương trình giúp người dùng lập kế hoạch và quản lí nó một cách có hiệu quả và chất lượng. Xuất phát từ nhận định này em sẽ xây dựng ứng dụng Mynote đa nền tảng. Hy vọng đem tới một ứng dụng thuận tiện, hữu ích cho mục đích quản lý kế hoạch, lịch trình, thời gian biểu cũng như sắp xếp công việc hàng ngày.

## 1.2. Mục tiêu và nhiệm vụ

**Mục tiêu:**

* Hiểu hơn về framework .NET MAUI.
* Sử dụng ngôn ngữ C# để hoàn thành đề tài.

**Nhiệm vụ:**

* Tìm đọc các tài liệu, thư viện liên quan.
* Lắng nghe những gợi ý, hướng dẫn của giảng viên để thực hiện chính xác yêu cầu.

## 1.3. Đối tượng nghiên cứu và phạm vi hệ thống

**Đối tượng:**

* Ngôn ngữ C#.
* Visual Studio.

**Phạm vi:** Ứng dụng đáp ứng được một số chức năng:

## 1.4. Phương pháp nghiên cứu

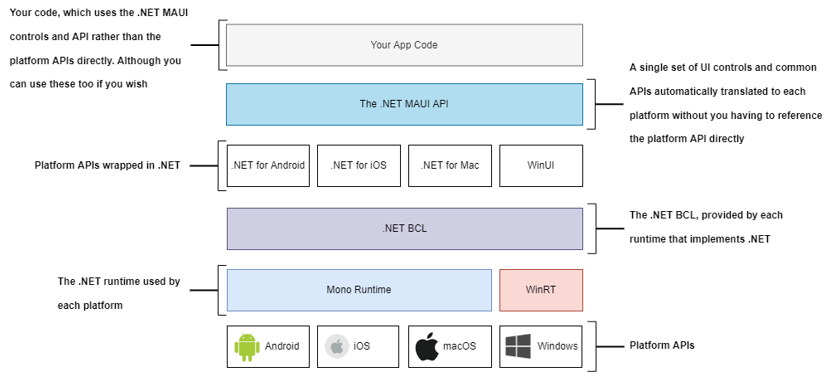
Đọc sách và các tài liệu tham khảo tìm được

**Chương 2. CƠ SỞ LÝ THUYẾT**

## Giới thiệu .NET MAUI

* + 1. **.NET MAUI là gì?**

.NET Multiplatform App UI (MAUI) là một framework mới của Microsoft để xây dựng các ứng dụng giao diện người dùng đa nền tảng nhắm mục tiêu Windows, macOS, iOS và Android.Với .NET MAUI, ta có thể xây dựng một ứng dụng giao diện người dùng gốc phong phú, tương tác, chạy trên bất kỳ nền tảng nào. Với một cơ sở mã duy nhất, có thể xây dựng một ứng dụng hỗ trợ tất cả các nền tảng này và chia sẻ 100% mã giữa chúng. Tóm lại, khi viết một ứng dụng bằng ngôn ngữ .NET và nó chạy mà không có bất kỳ thay đổi nào trên bất kỳ nền tảng đích nào. Tất cả logic đều có thể được viết bằng ngôn ngữ .NET và giao diện người dùng cũng có thể được xác định bằng XAML hoặc bằng ngôn ngữ .NET mà ta chọn.



*Hình 1 Kiến trúc của ứng dụng .NET MAUI*

NET MAUI được xây dựng từ dưới lên; mỗi nền tảng cung cấp API và có một thời gian chạy .NET cho mỗi nền tảng (WinRT trên Windows, Mono trên mọi thứ khác) được xây dựng trên các API này. Mỗi lớp cung cấp các API sử dụng để xây dựng các API trong lớp trên. Trong khi đó, mã của chúng ta được viết từ trên xuống; khi viết một ứng dụng .NET MAUI và kiến ​​trúc sẽ đảm nhiệm việc đóng gói nó cho các lớp thấp hơn.

Khi nhìn vào các lớp trong sơ đồ này. Ở lớp dưới cùng, chúng ta có các hệ điều hành mục tiêu (Android, iOS, macOS và Windows). Lớp tiếp theo hiển thị thời gian chạy .NET sẽ thực thi ứng dụng .NET MAUI của chúng ta trên mỗi hệ điều hành mục tiêu. Đối với Android, iOS và macOS, thời gian chạy này là Mono và đối với Windows là WinRT.

Lớp .NET (BCL). là abstraction đầu tiên - thư viện lớp cơ sở. BCL cung cấp quyền truy cập vào tất cả các tính năng ngôn ngữ phổ biến mà chúng ta mong đợi, chẳng hạn như danh sách và khái quát, không tạo thành một phần nguyên thủy của .NET. Từ .NET 5 trở đi, .NET (không có Core hoặc Framework) đã trở thành tiêu chuẩn mới, thay thế cả .NET Standard.

Lớp tiếp theo, nằm trên của lớp BCL, cung cấp quyền truy cập vào phần tóm tắt cho các API cụ thể của nền tảng. .NET cho Android và .NET cho iOS lần lượt là các phiên bản tiếp theo của Xamarin.Android và Xamarin.iOS. Đây là các liên kết với API nền tảng, sử dụng cùng loại và không gian tên được sử dụng bởi các nhà phát triển Objective-C, Swift, Java hoặc Kotlin. .NET dành cho Mac là mới nhưng hoạt động theo cùng một cách và đối với Windows, API WinUI được sử dụng.

Abstraction cuối cùng là .NET MAUI. Đây là một API hợp nhất cung cấp các phần tử giao diện người dùng chung cho tất cả các nền tảng được hỗ trợ. Thông qua không gian tên Essentials (trong Xamarin, phần này được bắt đầu dưới dạng một gói riêng biệt) các nhà phát triển cũng có quyền truy cập vào các tính năng phần cứng phổ biến, chẳng hạn như Bluetooth, dịch vụ định vị và bộ nhớ thiết bị.

Trong khi chúng ta xem xét các lớp từ dưới lên, triết lý xây dựng ứng dụng .NET MAUI được xem xét rất nhiều từ trên xuống: xây dựng một ứng dụng đa nền tảng bằng cách viết mã .NET MAUI (chứ không phải mã iOS hoặc Android). Nếu muốn, developer vẫn có thể viết mã dành riêng cho nền tảng hoặc hệ điều hành trong ứng dụng của mình, nhưng ta không cần phải làm vậy.

.NET MAUI lấy mã của các nhà phải triển và biên dịch nó cho nền tảng đích. Hiểu cách .NET MAUI xây dựng ứng dụng của mình cho các nền tảng khác nhau là không cần thiết để xây dựng ứng dụng .NET MAUI, mặc dù hiểu rõ về các nền tảng này là có lợi.

.NET MAUI không chỉ là phiên bản tiếp theo của Xamarin.Forms. Trong khi Xamarin là bộ công cụ phát triển phần mềm (SDK) mà developer đã cài đặt độc lập với .NET, thì MAUI là khối lượng công việc , nghĩa là nó là một phần của .NET, giống như phát triển ứng dụng ASP hoặc bảng điều khiển. Cách tiếp cận này cung cấp một số lợi ích, nhưng quan trọng nhất là nó thể hiện cam kết của Microsoft đối với tương lai của .NET MAUI và việc đưa nó vào làm một phần cốt lõi của .NET.

* + 1. **.NET MAUI và hệ sinh thái .NET**

Viết một ứng dụng đa nền tảng trong .NET MAUI là một lựa chọn tuyệt vời cho các nhà phát triển .NET. Các nhà phát triển có quyền truy cập vào tất cả các tài nguyên giống như khi sử dụng trong các dự án .NET khác, bao gồm các mạng, gói và mẫu hỗ trợ hiện có của mình (mặc dù cũng có một số mẫu mới cần tìm hiểu để phát triển thiết bị di động). Khi làm một dự án .NET, nhiều gói NuGet yêu thích có sẵn để sử dụng trong các ứng dụng .NET cũng như nhiều gói cụ thể được điều chỉnh để phát triển giao diện người dùng trên thiết bị di động và đa nền tảng.

Diagram

Description automatically generated

*Hình 2 Kiến trúc của ứng dụng .NET MAUI*

Nếu xây dựng một giải pháp full-stack, developer sẽ nhận được lợi ích có thể chia sẻ mã giữa các class khác nhau. Điều này có thể không phải lúc nào cũng áp dụng được vì các class khác nhau thường có những trách nhiệm rất khác nhau, Ví dụ: nếu đang xây dựng ứng dụng trò chuyện với ASP.NET WebApi với SignalR chạy trong Azure, giao diện người dùng web Blazor và giao diện người dùng máy tính và thiết bị di động .NET MAUI, ta có thể chia sẻ logic và kết nối liên kết giao diện người dùng với WebApi trên cả ứng dụng Blazor và .NET MAUI của mình và trong một số trường hợp, có thể sử dụng các gói NuGet giống nhau trong cả ứng dụng máy khách và máy chủ của mình. Nếu các nhà phát triển thực hiện các thay đổi đối với API của mình, có thể cập nhật mã máy khách một lần và thay đổi sẽ tự động được phản ánh trên tất cả các ứng dụng giao diện người dùng client.

Như đã đề cập trong phần 2.1.1, với .NET MAUI hiện là một phần cốt lõi của .NET và khối lượng công việc thay vì SDK, ta cũng có thể sử dụng tất cả các công cụ phát triển quen thuộc của mình. Nếu nó hoạt động với .NET, thì nó hoạt động với .NET MAUI. Điều này bao gồm các công cụ dành cho nhà phát triển thường dung như: Visual Studio (Mac hoặc Windows), VS Code và .NET CLI (mặc dù Visual Studio sẽ cung cấp trải nghiệm hạng nhất); bất kỳ công cụ xây dựng hoặc DevOps nào; nuget packages; và bất cứ điều gì khác mà ta có thể nghĩ đến. Các thành phần khác nhau tạo nên trải nghiệm nhà phát triển .NET của nhà phát triển, toàn bộ hệ sinh thái .NET, được tùy ý sử dụng với tư cách là nhà phát triển .NET MAUI.

.NET MAUI không phải là tiện ích bổ sung, mà là .NET. Điều đó có nghĩa là nếu nó hoạt động với .NET, nó sẽ hoạt động với .NET MAUI – và điều này ảnh hưởng đến kỹ năng của các nhà phát triển nhiều như bất kỳ công cụ nào khác.

* + 1. **Mô hình phát triển .NET MAUI**

XAML (eXtensible Application Markup Language) là lựa chọn thực tế để xây dựng các ứng dụng trong .NET MAUI. Như đã đề cập trong phần giới thiệu, Microsoft đã tạo XAML cho WPF, nhưng nó cũng đã được sử dụng cho Silverlight (trên thực tế, Silverlight đặc biệt là một plugin trình kết xuất XAML cho trình duyệt web), Windows Phone, UWP và Xamarin.Forms, và bây giờ là .NET MAUI cũng vậy.

XAML là một lựa chọn tốt cho hầu hết mọi người. Đó là ngôn ngữ đánh dấu dựa trên XML để xác định giao diện người dùng, giống như HTML hoặc XML thuần túy được sử dụng trong Kotlin để xây dựng giao diện người dùng Android. Có những khác biệt tinh tế giữa WPF, Xamarin.Forms và .NET MAUI XAML, nhưng những khác biệt này học rất nhanh và XAML Intellisense được chỉnh sửa xuất sắc mà các nhà phát triển có được với Visual Studio 2022 làm cho nó thậm chí còn dễ dàng hơn.

Bổ sung cho XAML là mẫu Model-View-View-Model (MVVM). Với MVVM, giao diện người dùng do XAML xác định được gọi là Chế độ xem. Chế độ xem bao gồm mọi thứ mà ứng dụng yêu cầu để hiển thị mọi thứ trên màn hình theo thiết kế, bao gồm các điều khiển giao diện người dùng cũng như bất kỳ mã nào được yêu cầu để thay đổi cách chúng được hiển thị. ViewModel đại diện cho trạng thái View và chứa logic để tương tác với Model. ViewModel phản hồi các sự kiện trong View và chuyển dữ liệu tới Model, đồng thời thay đổi trạng thái của View khi Model cung cấp thông tin yêu cầu trạng thái View thay đổi.

Diagram

Description automatically generated

*Hình 3 Mẫu MVVM. Mô hình đại diện cho vấn đề mà ứng dụng của các nhà phát triển giải quyết và bao gồm các thực thể và dịch vụ*

Nhưng XAML không phải là lựa chọn duy nhất. Ta cũng có thể khai báo giao diện người dùng của mình bằng mã thay vì markup. Điều này có nghĩa là khai báo một thể hiện của lớp đại diện cho điều khiển giao diện người dùng hoặc dạng xem mà developer muốn hiển thị trên màn hình và chỉ định các thuộc tính của nó. Thông thường các nhà phát triển xác định giao diện người dùng của mình trong markup hơn; có thể là do cách tiếp cận mã gợi lên ký ức về việc vẽ các nút trên màn hình bằng cách chỉ định tọa độ để vẽ một đường màu đen cho đường viền trên cùng và bên trái và một đường màu trắng cho đường viền dưới cùng và bên phải (sau đó đảo ngược chúng khi người dùng nhấp vào nút) . Lúc đó các nhà phát triển sẽ không có quyền truy cập vào thư viện giao diện người dùng và phải vẽ tất cả các điều khiển của mình bằng tay.

Diagram, schematic

Description automatically generated

*Hình 4 Mẫu MVU, Model đại diện cho trạng thái toàn bộ ứng dụng*

Ngoài việc khai báo giao diện người dùng trong mã, cũng có thể sử dụng mô hình Model-View-Update (MVU) thay vì MVVM. MVU, còn được gọi là kiến trúc Elm, khác với MVVM ở hai điểm chính. Đây là những mô hình không thay đổi và dữ liệu đó chỉ theo một hướng. Điều này có nghĩa là ta không thể thay đổi mô hình để đáp ứng với các thay đổi về giao diện người dùng (đầu vào của người dùng), vì điều này sẽ vi phạm cả hai quy tắc này. Thay vào đó, các thay đổi trong luồng chế độ xem thành update, tạo ra một new model. Sau đó, chế độ view thay đổi để đáp ứng với new model.

 .NET MAUI còn cung cấp cho ta một tùy chọn khác hoàn toàn, đó là xây dựng giao diện người dùng bằng Blazor. Nó cho phép developer viết ứng dụng web trong khung ứng dụng một trang (SPA) (Blazor) và sử dụng trình bao bọc (.NET MAUI) để gói ứng dụng web đó thành tệp thực thi nhị phân có thể cài đặt được nhắm mục tiêu trên nhiều nền tảng. Tuy nhiên, điểm khác biệt chính với .NET MAUI Blazor là ở chỗ nó sử dụng chế độ xem web để hiển thị giao diện người dùng, giống như những thứ như Electron, Ionic hoặc Cordova, mã C# viết trong ứng dụng .NET MAUI Blazor được chạy dưới dạng mã được quản lý .NET, giống như trong ứng dụng XAML Blazor, thay vì được chạy bởi công cụ tạo tập lệnh trong chế độ xem web, đây là những gì các nhà phát triển nhận được với trình bao bọc web như Cordova, Ionic hoặc Electron.

## Tổng quan về Oracle

* + 1. **Orancle là gì?**

Cơ sở dữ liệu Oracle (Oracle Database) là loại cơ sở dữ liệu quan hệ, trong đó toàn bộ dữ liệu được lưu trong các bảng 2 chiều bao gồm các hàng và cột, cho phép bạn lưu trữ, cập nhật và truy xuất dữ liệu một cách hiệu quả.

Cơ sở dữ liệu Oracle bao gồm các cấu trúc vật lý và luận lý được dùng để lưu trữ thông tin hệ thống, thông tin người dùng và thông tin điều khiển. Bởi vì hai loại cấu trúc này tách biệt nhau, nên việc lưu trữ dữ liệu vật lý có thể được quản lý mà không ảnh hưởng đến việc truy xuất các cấu trúc luận lý.

Oracle cung cấp một phần mềm dùng để quản lý cơ sở dữ liệu này, gọi là Oracle Database Server.Oracle Database Server và Oracle Database được gọi chung là Hệ Cơ sở dữ liệu Oracle (Oracle Database System). Tuy nhiên, để cho gọn, chúng ta vẫn thường dùng từ “Oracle Database” để chỉ Oracle Database System.

* + 1. **Cấu trúc của cơ sở dữ liệu Oracle**

Oracle sử dụng nền tảng SQL. Đây là một ngôn ngữ nổi tiếng trong lập trình, được nhiều developer sử dụng. SQL có tác dụng chuẩn hóa, giúp việc quản lý dữ liệu chính xác hơn. SQL cũng được nhiều đơn vị sử dụng để quản lý và truy vấn dữ liệu trong quá trình lưu trữ. Oracle cũng được gắn liền với PL/SQL. Đây là một phần mềm có chức năng bổ trợ, giúp bổ sung extension cho hệ thống. Bên cạnh đó, Oracle cung hỗ trợ lập trình thông qua Java.

Về mặt kiến trúc, Oracle sẽ sử dụng máy chủ cơ sở dữ liệu. Máy chủ này gồm có database – có chức năng lưu trữ dữ liệu. Tùy từng máy chủ mà có thể có 1 hoặc nhiều database. Database cũng giúp quản lý các tệp dữ liệu, kết hợp giữa logic và vật lý. Trong đó:

* Cấu trúc vật lý của hệ thống bao gồm các tệp dữ liệu chứa metadata và tệp nhật ký online. Metadata có chức năng điều khiển dữ liệu.
* Cấu trúc Logic gồm các khối dữ liệu, các nhóm dữ liệu như Extents và các phần mở rộng phân đoạn, không gian bảng….

### Tính năng và tuỳ chọn tiêu biểu của Orancle

Cơ sở dữ liệu Oracle cung cấp nhiều tính năng hữu ích cho người dùng để đáp ứng yêu cầu của Database Management như:

* **Khả năng mở rộng và hiệu suất**: Các tính năng như Real Application Clustering và Portability làm cho Oracle database có thể mở rộng tùy theo cách sử dụng. Trong cơ sở dữ liệu đa người dùng, nó yêu cầu kiểm soát tính nhất quán và đồng thời của dữ liệu được Oracle dự tính.
* **Tính khả dụng**: Các ứng dụng real-time yêu cầu tính khả dụng của dữ liệu cao. Môi trường máy tính hiệu suất cao được định cấu hình để cung cấp dữ liệu luôn sẵn sàng. Dữ liệu luôn có sẵn trong Downtime theo kế hoạch hoặc ngoài kế hoạch.
* **Backup và Recovery**: Layout của nó hoàn chỉnh các tính năng recovery để khôi phục dữ liệu từ hầu hết các loại lỗi. Trong trường hợp bị lỗi, database cần được phục hồi ngay lập tức để có tính khả dụng cao. Các phần dữ liệu không bị ảnh hưởng vẫn có sẵn trong khi phần dữ liệu bị ảnh hưởng vẫn đang được phục hồi.
* **Bảo mật**: Bảo mật dữ liệu luôn được ưu tiên hàng đầu. Oracle cung cấp các cơ chết để kiểm soát việc truy cập và sử dụng dữ liệu. Việc thực hiện ủy quyền và chỉnh sửa hành động của người dùng có thể ngăn chặn truy cập trái phép và cho phép người dùng truy cập riêng biệt.

# **CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG**

* 1. **Mô tả về chương trình**

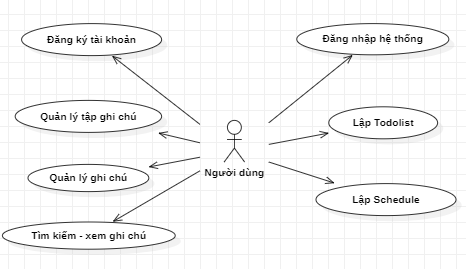
Khi làm việc, sẽ có rất nhiều lúc ta cần ghi lại những nội dung quan trọng, như lịch hẹn gặp mặt khách hàng, họp với team, công việc cần hoàn thành gấp...Tuy nhiên nếu thường xuyên làm việc với máy tính thì thay vì giấy nhớ (sticky note), ta có thể sử dụng các phần mềm quản lý ghi chú. Dễ dàng sử dụng, không mất một khoản phí nào và có thể đồng bộ hóa giữa máy tính và thiết bị di động (như smartphone hay tablet).

Từ bài toán đó ta có thể xây dựng một phần mềm giúp người dùng tạo ra các bản ghi chú trên máy tính hoặc thiết bị di động có cài ứng dụng. Các ghi chú này được lưu trữ trên server của ứng dụng nên người dùng chỉ cần đăng nhập vào tài khoản là có thể xem trên bất kỳ thiết bị nào. Ứng dụng tương thích với tất cả các hệ điều hành Windows, Mac và điện thoại Android, iOS, Windows Phone.

* + 1. **Yêu cầu chức năng đối với hệ thống**
* Cùng với đó nó còn phải có khả năng trợ giúp cho người dùng. Một mặt nó không chỉ dễ thao tác với những người thành thạo về tin học mà mặt khác còn phải dễ thao tác với người ít hiểu biết về tin học.
* Tự động đồng bộ hóa ghi chú và sổ tay trên máy tính, điện thoại hoặc máy tính bảng thông qua đăng nhập từ tài khoản đã đăng ký.
* Thấy ngay mọi thông tin cần thiết trên bảng điều khiển quản lý tập ghi chú.
* Thêm nhiều loại nội dung vào ghi chú, chẳng hạn như văn bản, tài liệu, xuất tệp PDF, bản phác thảo, hình ảnh, âm thanh, bản trích trang web và nhiều nội dung khác.
* Tạo todo list để sắp xếp nhiệm vụ cần làm trong ngày, gạch tên nhiệm vụ đã hoàn thành
* Quản lý danh sách việc cần làm bằng tính năng lập lịch. Đặt ngày đến hạn và lời nhắc để không bị trễ hạn..
* Nhanh chóng tìm được mọi thứ – tính năng tìm kiếm mạnh mẽ của ứng dụng có thể tìm được nội dung có trong hình ảnh và ghi chú viết tay.
* Bắt đầu công việc trên một thiết bị và tiếp tục làm việc trên một thiết bị khác mà không hề bị ngắt quãng.
  + 1. **Yêu cầu phi chức năng đối với hệ thống:**
* Về mặt giao diện tất cả các button chức năng đều hiển thị trên màn hình, ghi chú sẵn có hiển thị theo list ở các cột để tạo ghi chú mới mà người dùng ngoài việc viết như một ghi chú thông thường còn có thể chọn cỡ chữ, font chữ, thêm bảng biểu, căn chỉnh lề...như sử dụng một chương trình soạn văn bản đơn giản.
* Cùng với đó nó còn phải có khả năng trợ giúp cho người dùng. Một mặt nó không chỉ dễ thao tác với những người thành thạo về tin học mà mặt khác còn phải dễ thao tác với người ít hiểu biết về tin học.
  1. **. Thiết kế hệ thống**
     1. **Xác định các tác nhân của hệ thống**

Dựa trên văn bản mô tả bài toán và việc phân tích của tác nhân người dùng, ta xác định được các ca sử dụng như sau:

* Đăng nhập hệ thống.
* Đăng ký tài khoản.
* Quản lý tập ghi chú.
* Quản lý ghi chú.
* Lập Todolist.
* Lập lịch (schedule).
* Tìm kiếm, xem ghi chú.



***Sơ đồ 1. Use case tổng thể***

* + 1. **Danh sách các Use case**

1. **Đăng ký tài khoản**

* Đối tượng: Người dùng.
* Use case này cho phép người dùng tạo tài khoản cho ứng dụng.
* Các bước thực hiện:
  + Người dùng tiến hành nhập email, mật khẩu, nhập lại mật khẩu.
  + Nhấn nút đăng ký. Hệ thống sẽ hiển thị giao diện đăng nhập.
  + Nhấn đăng nhập và kết thúc hoạt động hoặc tiếp tục sử dụng các chức năng khác.

1. **Đăng nhập hệ thống**

* Đối tượng: Người dùng.
* Use case này cho phép người dùng đăng nhập vào hệ thống.
* Các bước thực hiện:
  + Người dùng nhập email và mật khẩu và nhấn nút đang nhập đang hiện trên màn hình. Hệ thống sẽ thông báo đăng nhập thành công trên màn hình và vào trang chủ.
  + Nhấn nút quên mật khẩu nếu không nhớ mật khẩu. Hệ thống sẽ hiển thị yêu cầu nhập email sau đó nhận lại mật khẩu mới.
  + Nhấn Lưu để đăng nhập lại và kết thúc hoạt động hoặc tiếp tục sử dụng các chức năng khác.

1. **Quản lý tập ghi chú**

* Đối tượng: Người dùng.
* Use case này cho phép người dùng quản lý tập ghi chú.
* Các bước thực hiện:
  + Người dùng tạo, xoá, chia sẻ, đổi tên tập ghi chú. Hệ thống sẽ cập nhật thông tin và hiển thị thay đổi trên màn hình trang chủ.
  + Nhấn tạo, xoá ghi chú thuộc một tập ghi chú.
  + Nhấn tạo, xoá todo-list thuộc một tập ghi chú.
  + Nhấn Lưu để xác nhận thay đổi lại và kết thúc hoạt động hoặc tiếp tục sử dụng các chức năng khác. Hệ thống sẽ cập nhật và hiển thị thay đổi trong tập ghi chú.

1. **Quản lý ghi chú**

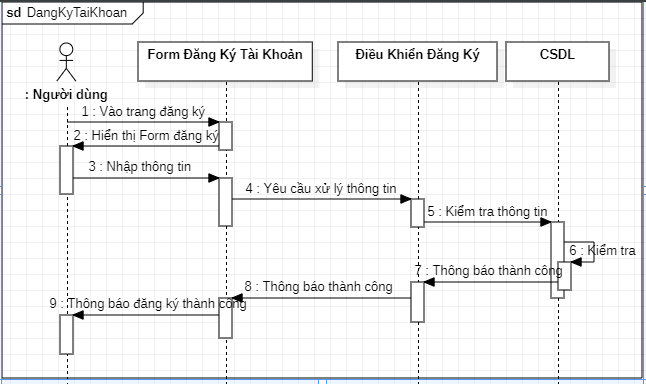
* Đối tượng: Người dùng.
* Use case này cho phép người dùng quản lý ghi chú.
* Các bước thực hiện:
  + Người dùng thêm, sửa, xoá, chia sẻ ghi chú.
  + Nhấn thêm, xoá ảnh trong ghi chú.
  + Người dùng tiến hành xuất ghi chú qua file pdf nếu có yêu cầu.
  + Nhấn Lưu để xác nhận thay đổi lại và kết thúc hoạt động hoặc tiếp tục sử dụng các chức năng khác. Hệ thống sẽ cập nhật và hiển thị thay đổi ghi chú.

1. **Quản lý Todo-List**

* Đối tượng: Người dùng.
* Use case này cho phép người dùng quản lý Todo-List.
* Các bước thực hiện:
  + Người dùng thêm, sửa, xoá Todo-List.
  + Người dùng tiến hành đánh dấu vào nhiệm vụ đã hoàn thành. Hệ thống sẽ cập nhật và gạch tên nhiệm vụ đã hoàn thành.
  + Nhấn Lưu để xác nhận thay đổi lại và kết thúc hoạt động hoặc tiếp tục sử dụng các chức năng khác. Hệ thống sẽ cập nhật và hiển thị thay đổi ghi chú.

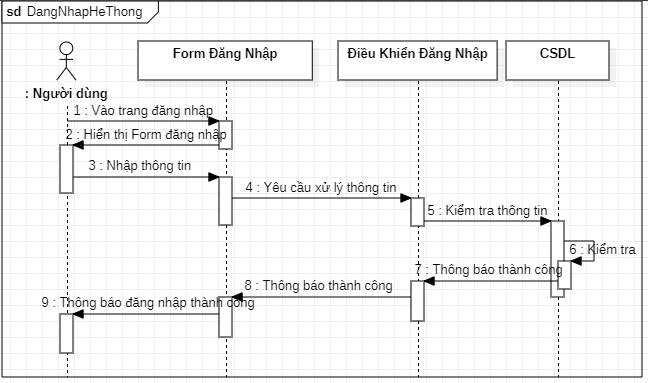
1. **Lập lịch Schedule**

* Đối tượng: Người dùng.
* Use case này cho phép người dùng lập lịch Schedule.
* Các bước thực hiện:
  + Người dùng thêm, sửa, xoá nhiệm vụ, sự kiện trong ngày giờ cụ thể. Hệ thống sẽ cập nhật và nhắc nhở công việc trước thời gian thực hiện 1 ngày, 1 giờ.
  + Nhấn Lưu để xác nhận thay đổi lại và kết thúc hoạt động hoặc tiếp tục sử dụng các chức năng khác. Hệ thống sẽ cập nhật và hiển thị thay đổi ghi chú.
    1. **Biểu đồ tuần tự cho các UC (Sequence Diagram)**
* **UC đăng ký tài khoản**



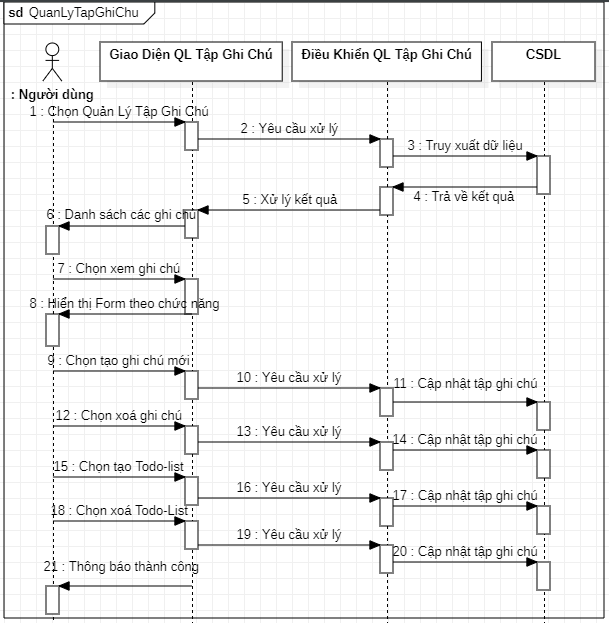
*Hình 5 Biểu đồ tuần tự đăng ký tài khoản*

* **UC đăng nhập hệ thống**



*Hình 6 Biểu đồ tuần tự đăng nhập hệ thống*

* **UC quản lý tập ghi chú**



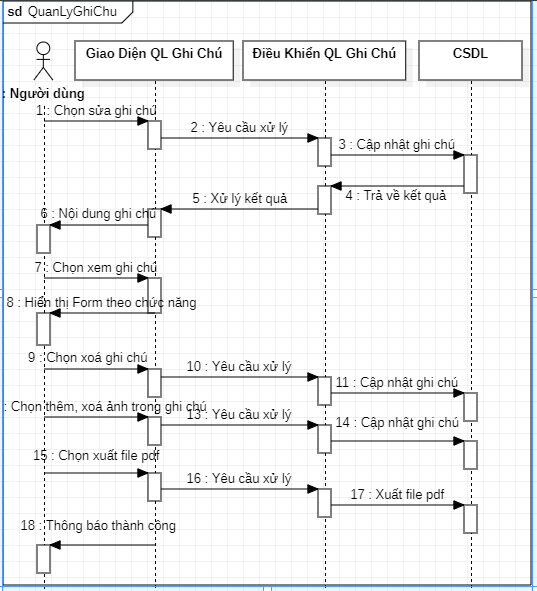
*Hình 7 Biểu đồ tuần tự quản lý tập ghi chú*

*x*

*x*

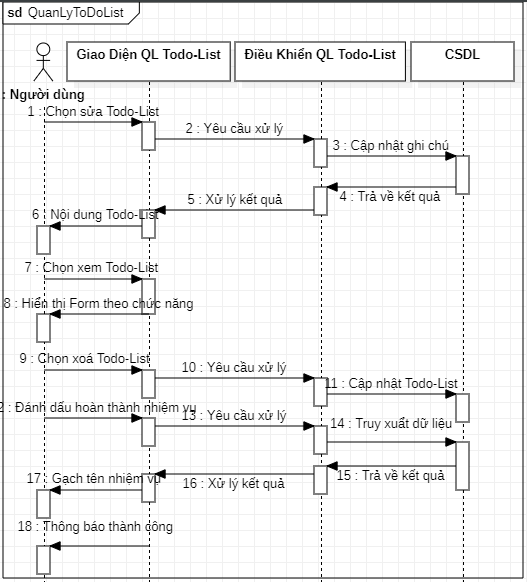
*x*

* **UC quản lý ghi chú**



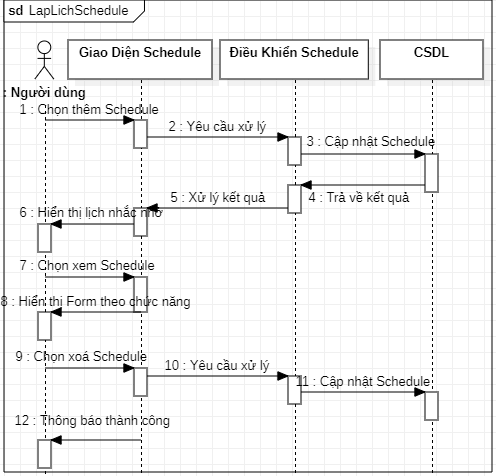
*Hình 8 Biểu đồ tuần tự quản lý ghi chú*

* **UC quản lý TodoList**



*Hình 9 Biểu đồ tuần tự quản lý TodoList*

* **UC Lập lịch Schedule**



*Hình 10 Biểu đồ tuần tự lập lịch Schedule*

**3.2.3 Sơ đồ trạng thái (State Machine Diagram)**

Diagram

Description automatically generated

***Sơ đồ 2. Sơ đồ trạng thái tổng quan ứng dụng***

* + 1. **Thiết kế cơ sở dữ liệu**

Bảng userDetail: lưu trữ thông tin cá nhân user

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Mô tả** |
| UserId | int | Khóa chính |
| fullName | nvarchar2(50) | Tên người dùng |
| Email | nvarchar2(50) | Email |
| DateOfBirth | date | Ngày sinh |
| Sex | int | Giới tính |

*Bảng 1 Bảng cơ sở dữ liệu cá nhân User*

Bảng user: lưu trữ thông tin tài khoản user

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Mô tả** |
| username | nvarchar2(50) | Tên người dùng |
| password | nvarchar2(50) | Mật khẩu |

*Bảng 2 Bảng cơ sở dữ liệu tài khoản User*

Bảng Notebook: lưu trữ thông tin tập ghi chú

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Mô tả** |
| UserId | int | Khóa chính |
| NotebookName | varchar2(50) | Tên tập ghi chú |
| DateCreate | Date | Ngày tạo tập ghi chú |

*Bảng 3 Bảng cơ sở dữ liệu notebook*

Bảng Note: lưu trữ thông tin ghi chú

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Mô tả** |
| NotebookId | int | Khóa chính |
| UserId | int | Khóa chính |
| NoteName | nvarchar2(50) | Tên ghi chú |
| NoteDetail | clob | Chi tiết ghi chú |
| DateCreate | Date | Ngày tạo ghi chú |
| IsFavorite | int | Ghi chú yêu thích |

*Bảng 4 Bảng cơ sở dữ liệu note*

Bảng TodoList: lưu trữ thông tin TodoList

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Mô tả** |
| NotebookId | int | Khóa chính |
| UserId | int | Khóa chính |
| Title | nvarchar(100) | Tiêu đề TodoList |
| Name | clob | Tên TodoList |
| Description | clob | Nội dung TodoList |
| IsDone | int | Nhiệm vụ đã hoàn thành |

*Bảng 5 Bảng cơ sở dữ liệu TodoList*

Bảng Schedule: lưu trữ thông tin Schedule

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Mô tả** |
| UserId | int | Khóa chính |
| StartDateTime | Date | Thời gian bắt đầu sự kiện |
| EndDateTime | Date | Thời gian kết thúc sự kiện |
| Title | nvarchar2(100) | Tiêu đề Schedule |
| Description | clob | Nội dung Schedule |

*Bảng 6 Bảng cơ sở dữ liệu Schechule*

Bảng NoteWithImg: lưu trữ thông tin ghi chú của hình ảnh

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Mô tả** |
| NotebookId | int | Khóa chính |
| UserId | int | Khóa chính |
| Title | nvarchar2(100) | Tiêu đề ảnh |

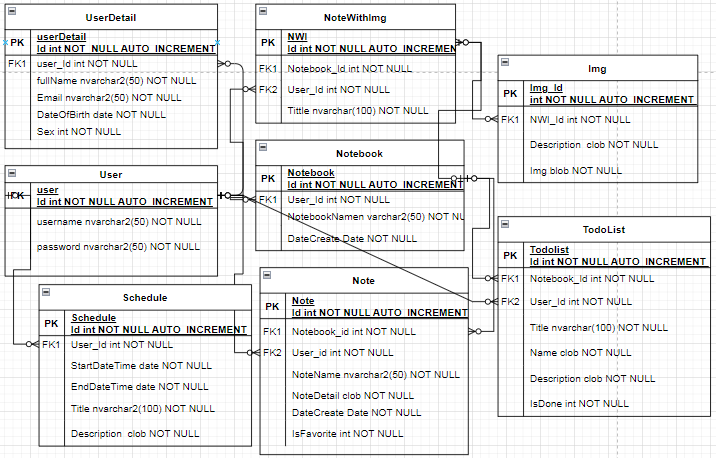
*Bảng 7 Bảng cơ sở dữ liệu NoteWithImg*

Bảng Img: lưu trữ hình ảnh

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Mô tả** |
| NWIId | int | Khóa chính |
| Description | clob | Nội dung hình ảnh |
| Img | clob | Hình ảnh |

*Bảng 8 Bảng cơ sở dữ liệu Img*

* + 1. **Biểu đồ lớp chi tiết**

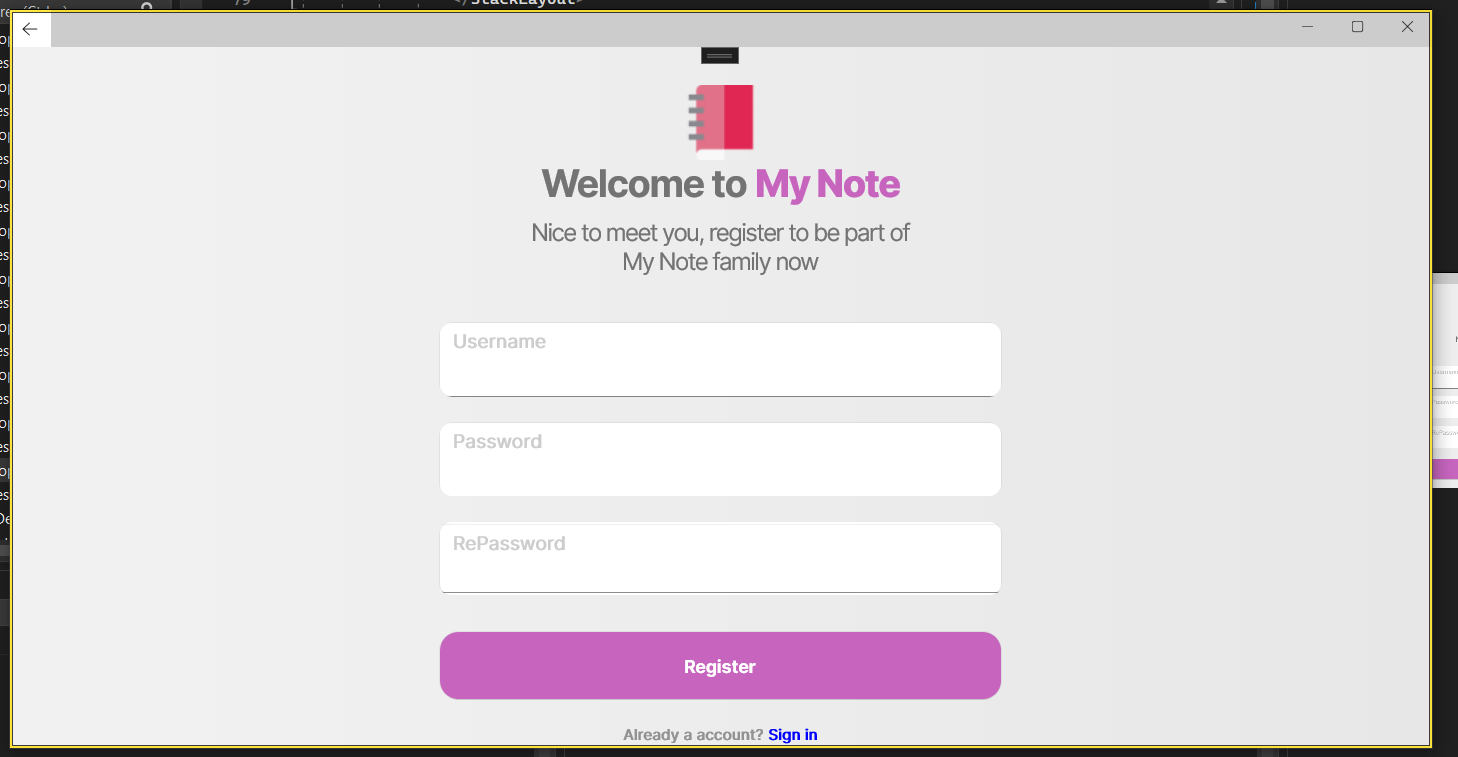


*Hình 11 Biểu đồ lớp chi tiết*

**CHƯƠNG 4: XÂY DỰNG ỨNG DỤNG**

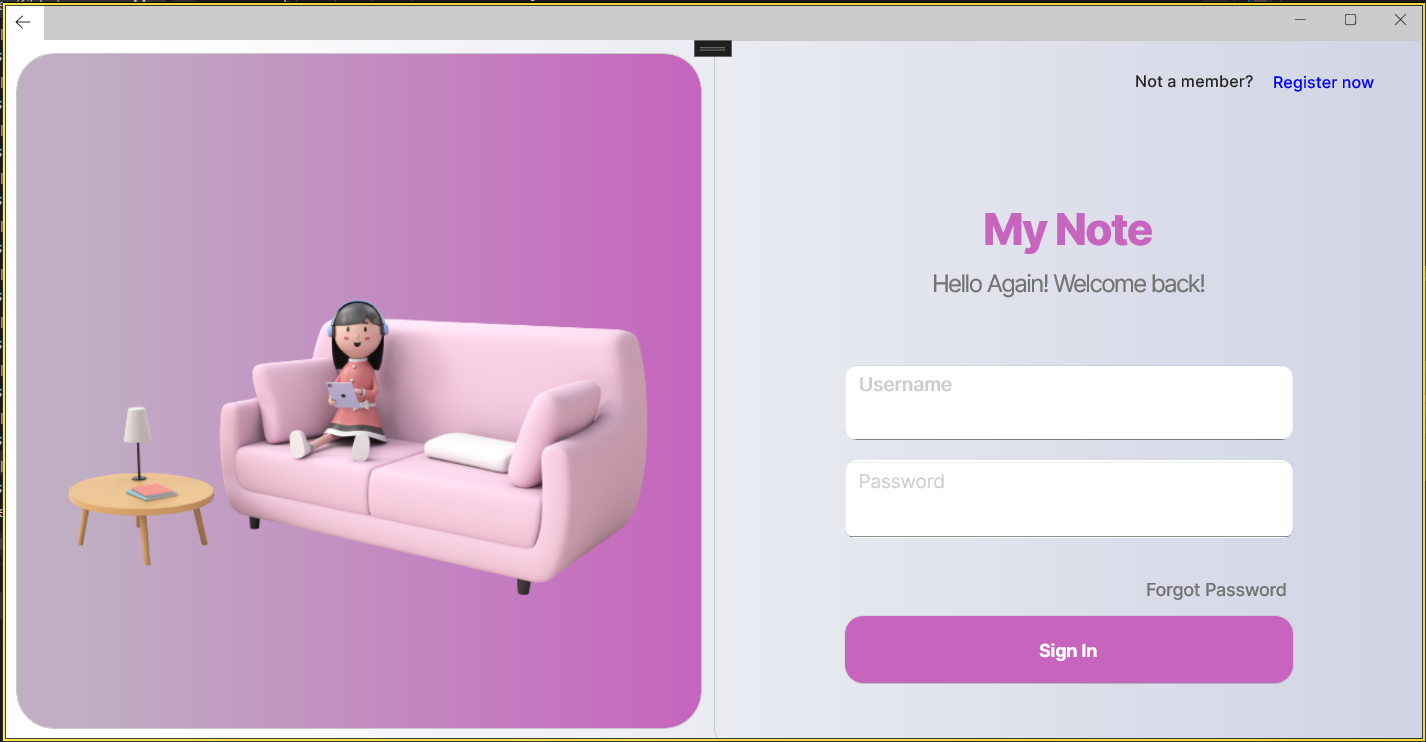
**4.1. Giao diện Desktop**

* **Giao diện màn hình đăng ký tài khoản**

****

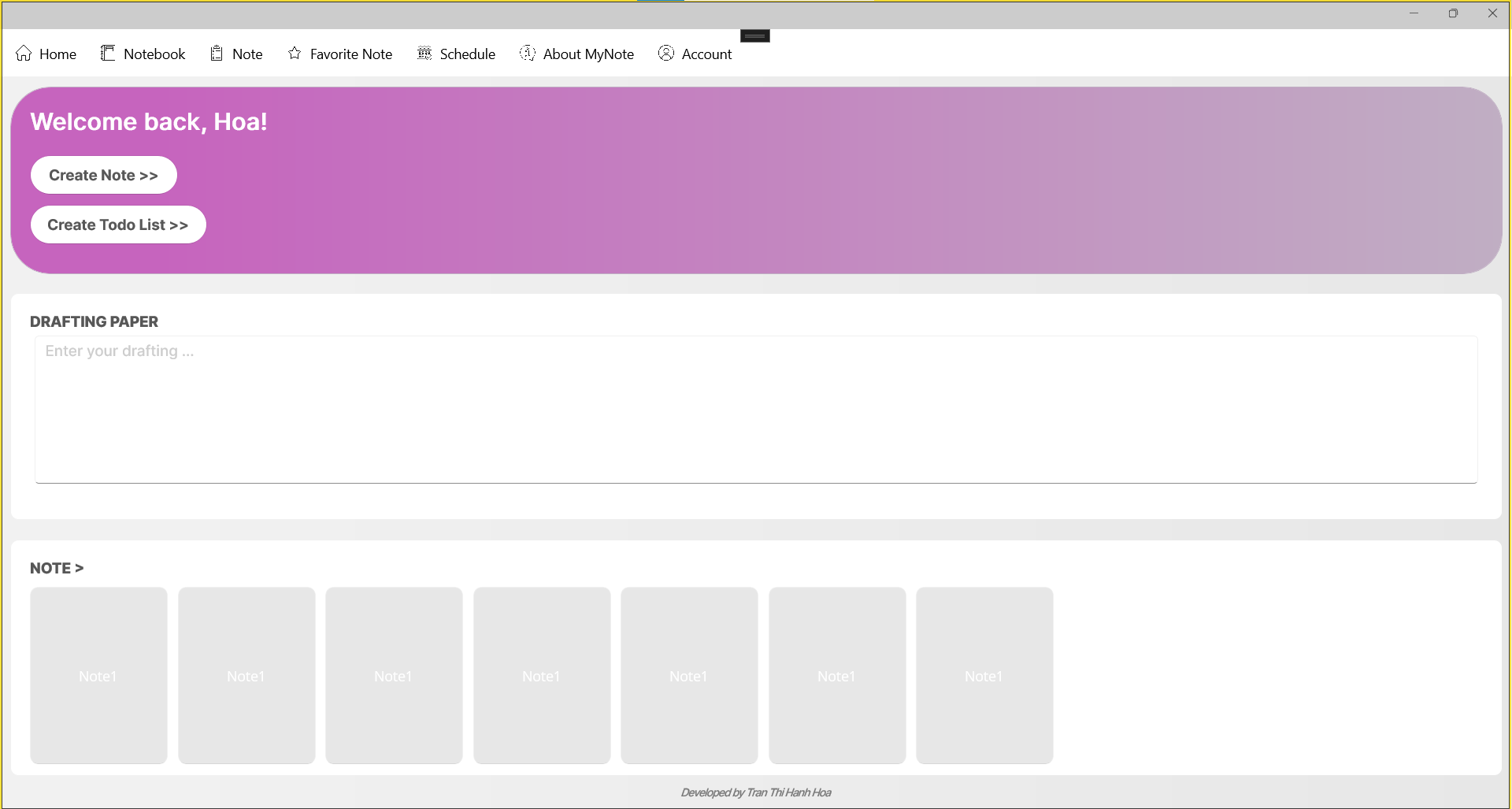
*Hình 12 Giao diện màn hình đăng ký tài khoản*

* **Giao diện màn hình đăng nhập hệ thống**

****

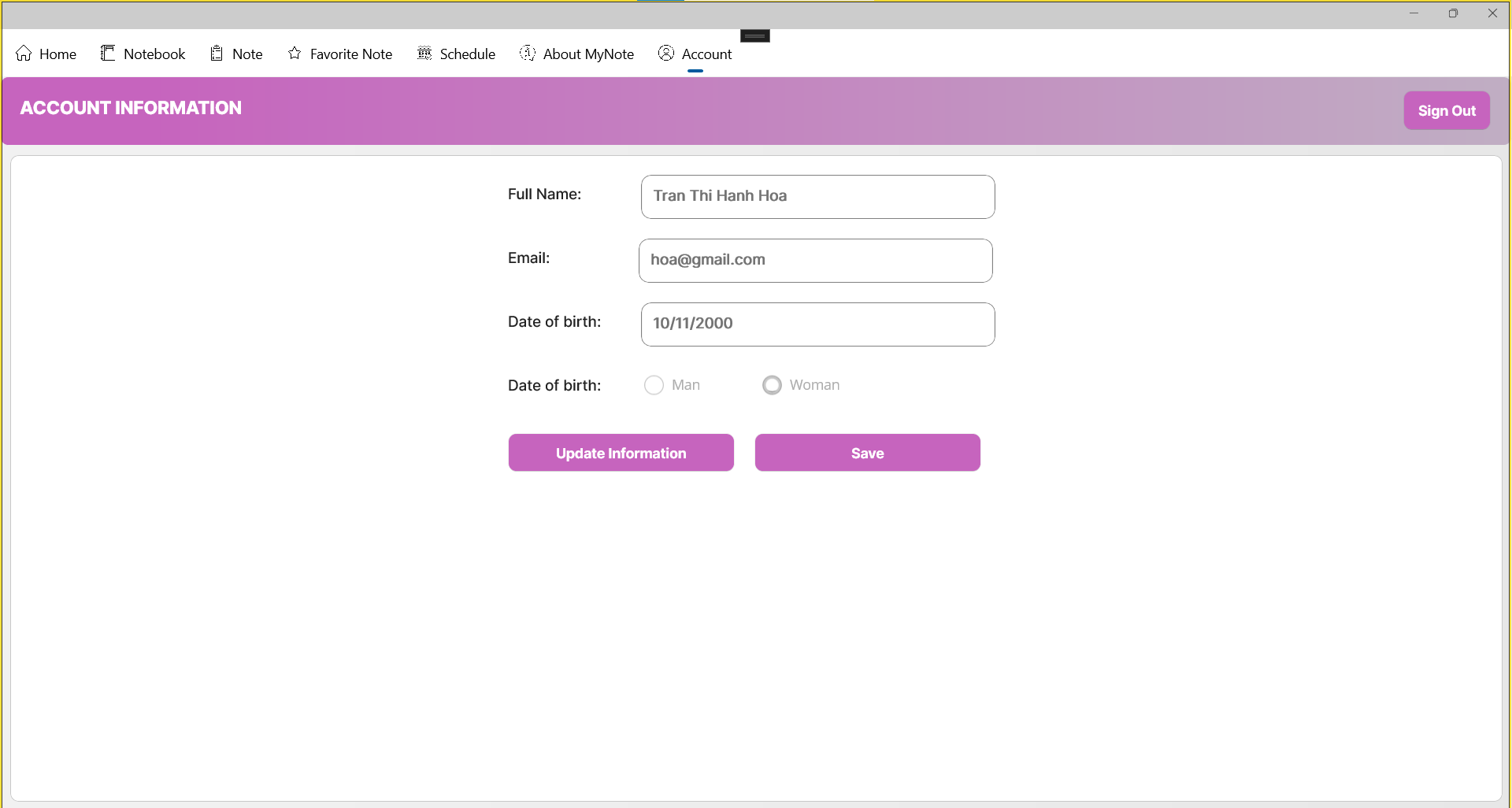
*Hình 13 Giao diện màn hình đăng nhập hệ thống*

* **Giao diện màn hình trang chủ**

****

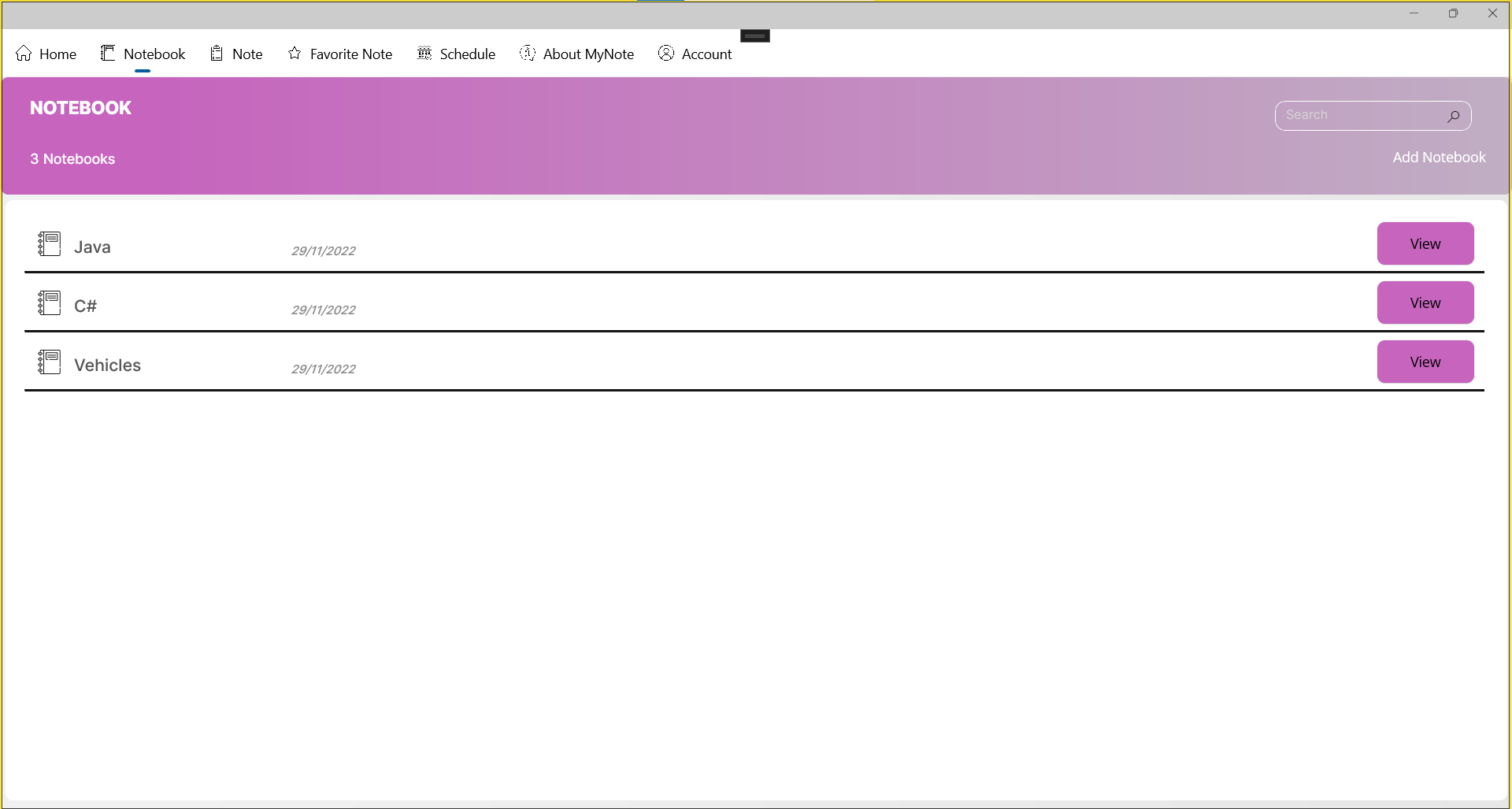
*Hình 14 Giao diện màn hình màn hình trang chủ*

* **Giao diện màn hình thông tin tài khoản**

****

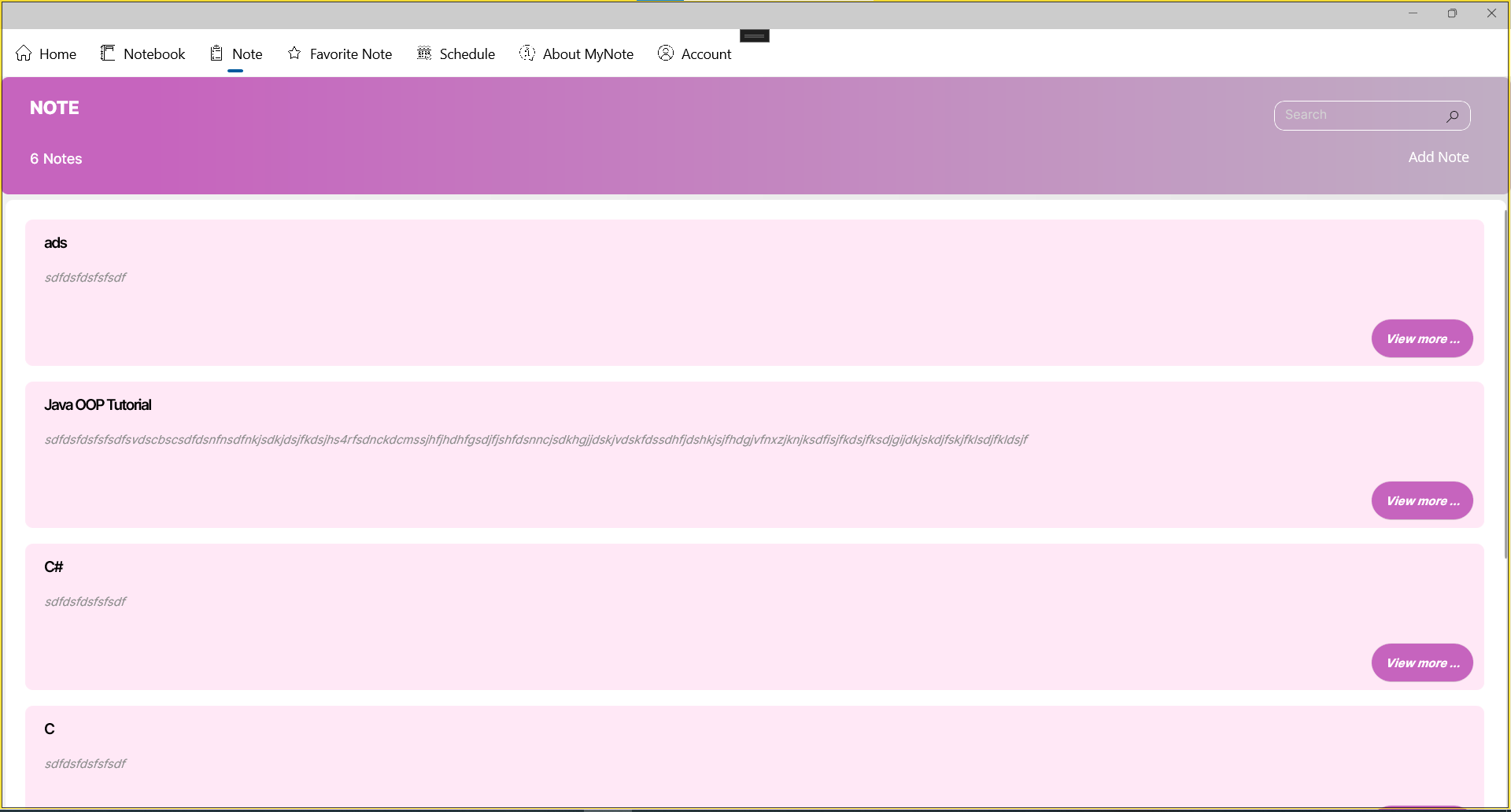
*Hình 15 Giao diện màn hình thông tin tài khoản*

* **Giao diện màn hình quản lý tập ghi chú**

****

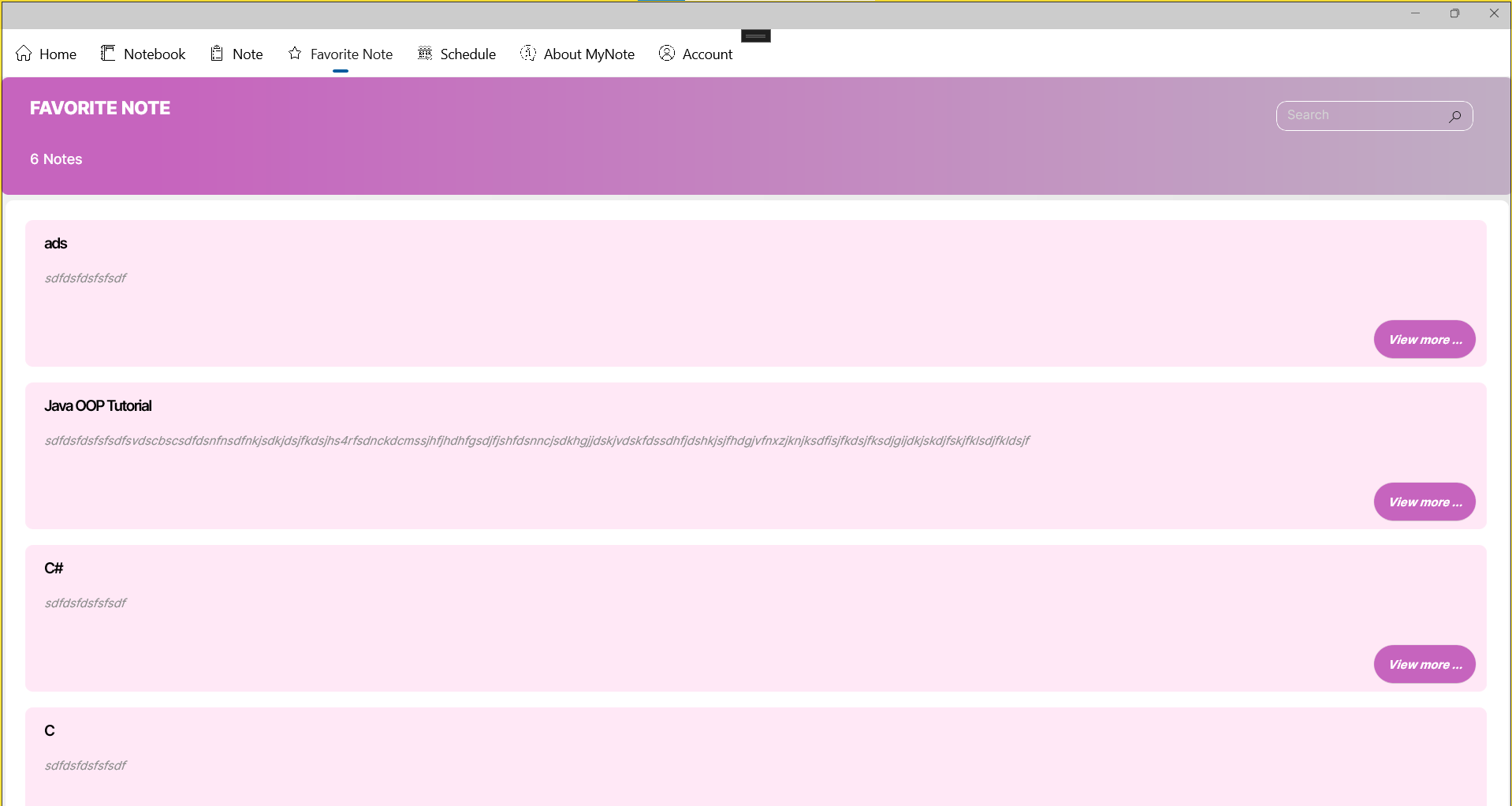
*Hình 16 Giao diện màn hình quản lý tập ghi chú*

* **Giao diện màn hình quản lý ghi chú**

****

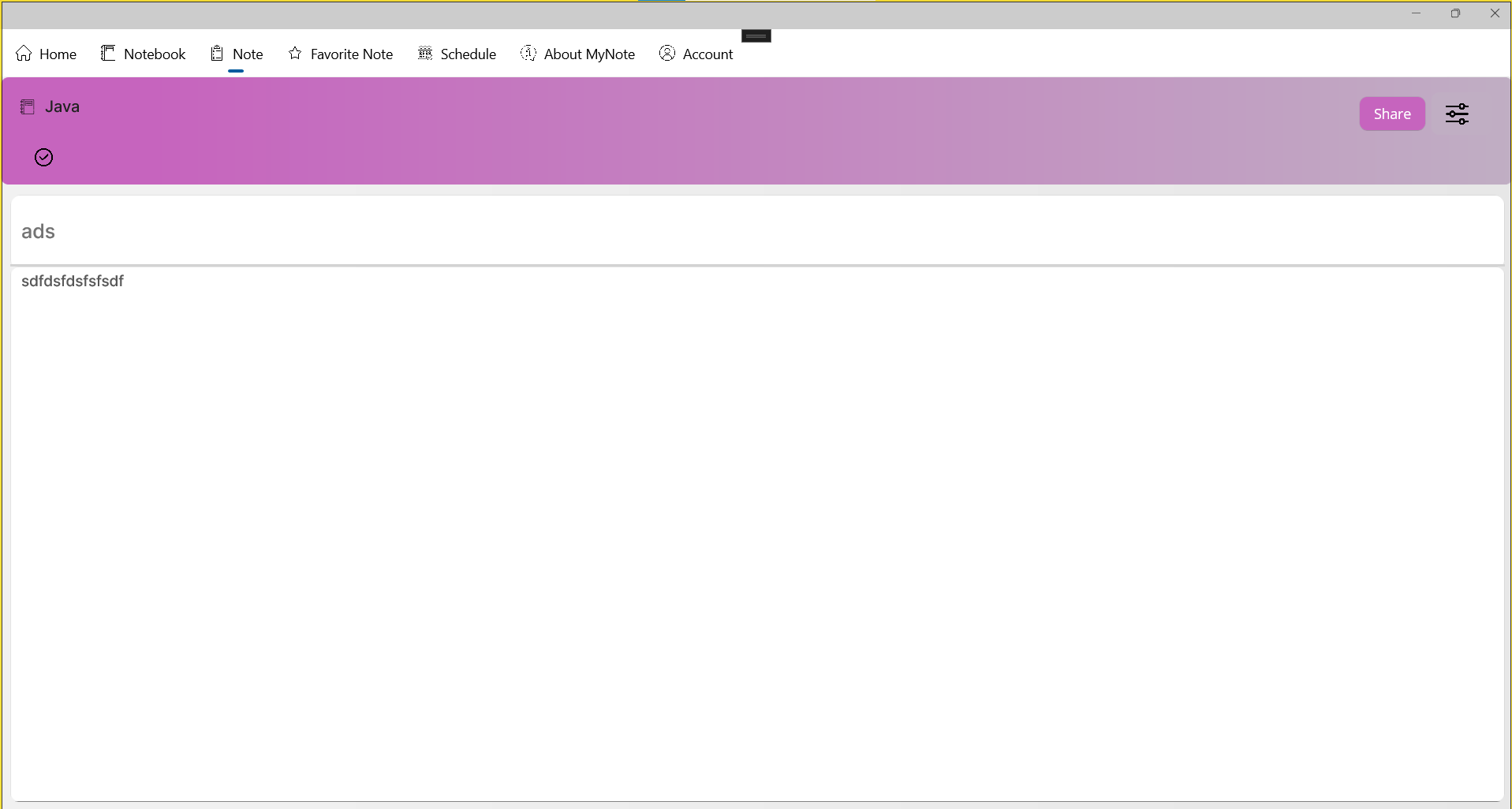
*Hình 17 Giao diện màn hình quản lý ghi chú*

* **Giao diện màn hình ghi chú yêu thích**

****

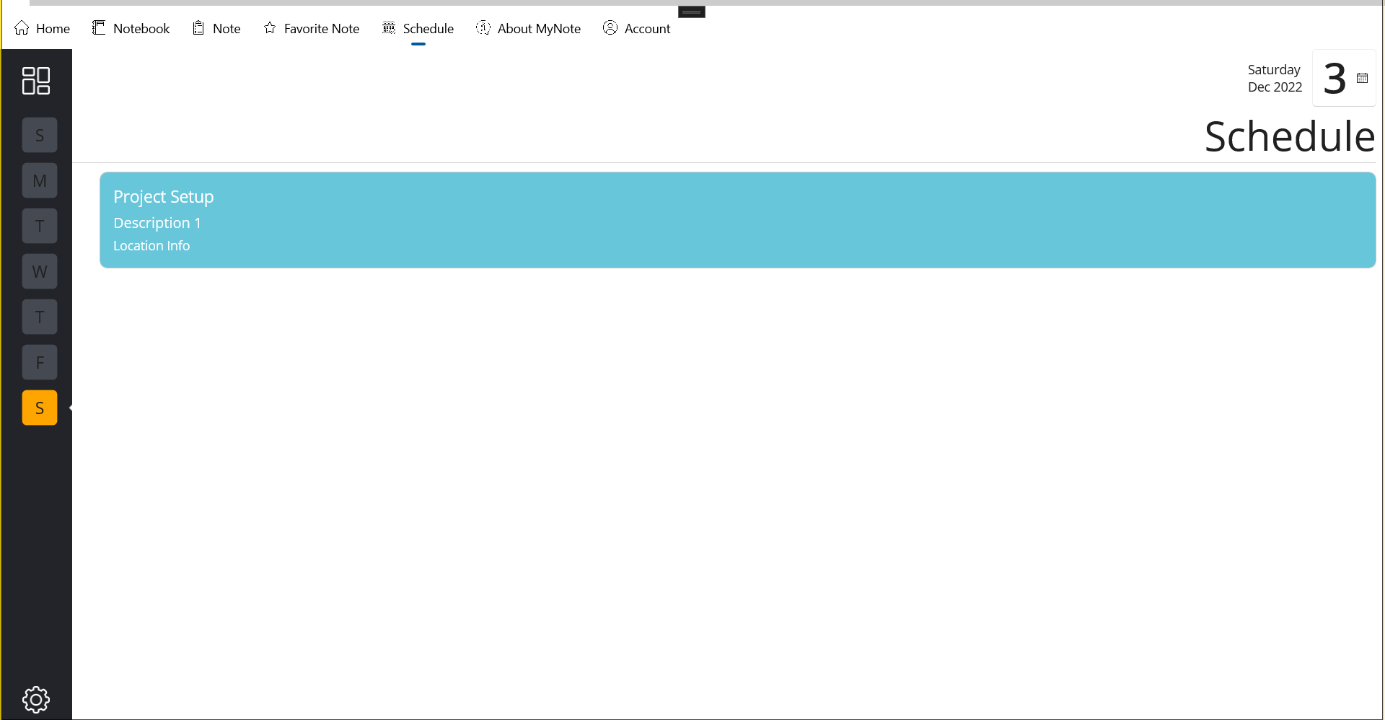
*Hình 18 Giao diện màn hình ghi chú yêu thích*

* **Giao diện màn hình ghi chú yêu thích**

****

*Hình 19 Giao diện màn hình ghi chú yêu thích*

* **Giao diện màn hình lập Schedule**

****

*Hình 20 Giao diện màn hình lập Schedule*

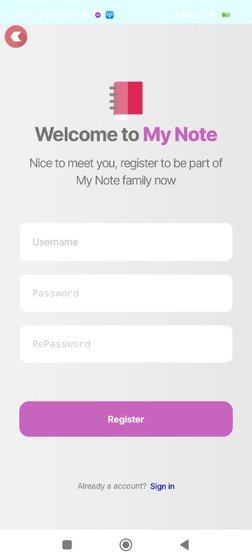
**4.1. Giao diện Mobile**

* **Giao diện màn hình trang giới thiệu**

****

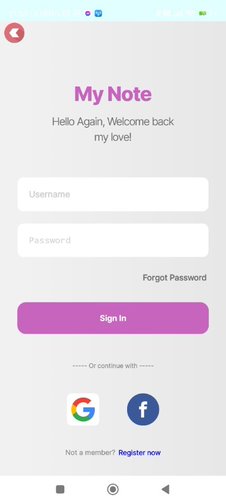
*Hình 21 Giao diện màn hình trang giới thiệu*

* **Giao diện màn hình đăng ký tài khoản**

****

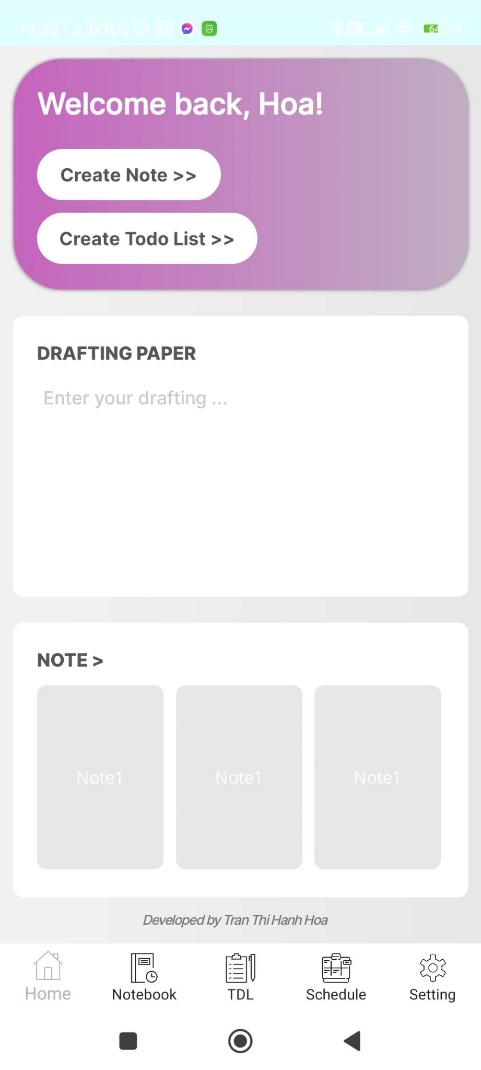
*Hình 22 Giao diện màn hình đăng ký tài khoản*

* **Giao diện màn hình đăng nhập hệ thống**

****

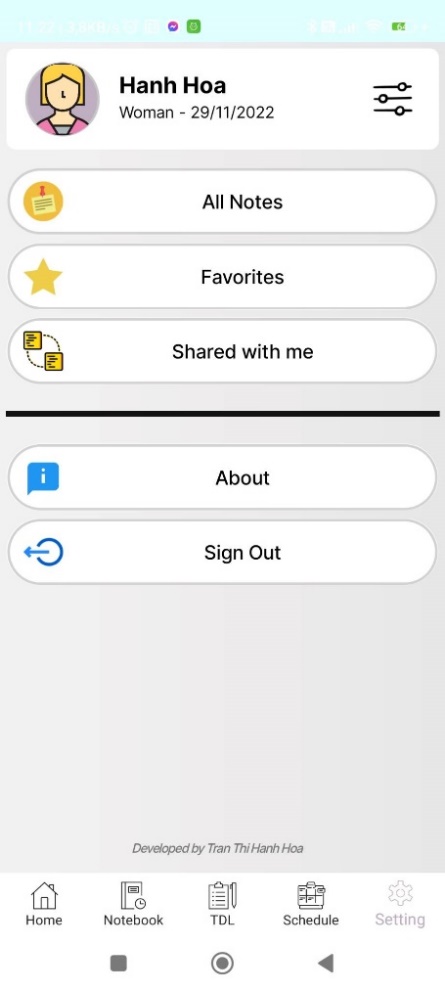
*Hình 23 Giao diện màn hình đăng nhập hệ thống*

* **Giao diện màn hình trang chủ**

****

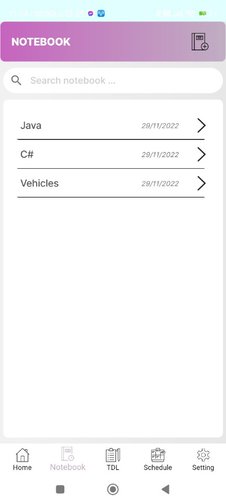
*Hình 24 Giao diện màn hìn*h *trang chủ*

* **Giao diện màn hình trang thông tin ứng dụng**

****

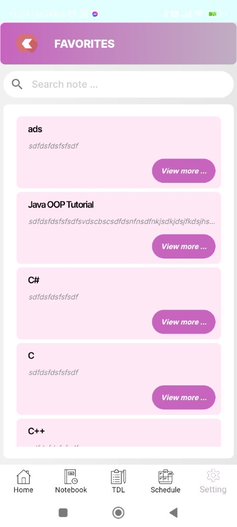
*Hình 25 Giao diện màn hình thông tin ứng dụng*

* **Giao diện màn hình quản lý tập ghi chú**

****

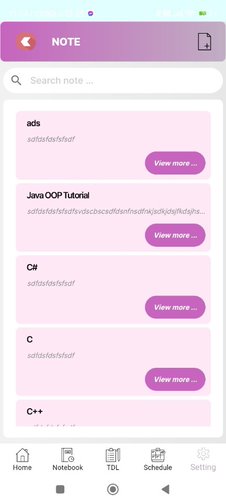
*Hình 26 Giao diện màn hình quản lý tập ghi chú*

* **Giao diện màn hình ghi chú yêu thích**

****

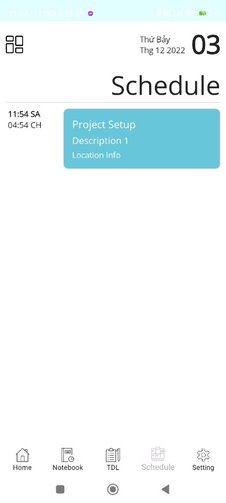
*Hình 27 Giao diện màn hình ghi chú yêu thích*

* **Giao diện màn hình quản lý ghi chú**

****

*Hình 28 Giao diện màn hình quản lý ghi chú*

* **Giao diện màn hình lập Schedule**

****

*Hình 29 Giao diện màn hình lập Schedule*

# **KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

**Kết luận**

**Ưu điểm**

* Chương trình có nhiều chức năng hỗ trợ đầy đủ cho các tác vụ ghi chú cơ bản.
* Chương trình hỗ trợ xử lý đa nền trên ghi chú.
* Chương trình nhỏ gọn, tương thích cao với nhiều phiên bản.

**Nhược điểm**

* Ứng dụng giao diện chưa bắt mắt

**Hướng phát triển**

* Mở rộng thêm nhiều chức năng hữu ích: gắn tab cho mỗi ghi chú để dễ dàng tìm kiếm hơn, hoàn thiện thêm giao diện.

# **Tài Liệu Tham Khảo**

[1] Book .NET Maui in Action by Matt Goldman 2021

**Website:**

[2].https://learn.microsoft.com/en-us/dotnet/maui/what-is-maui?view=net-maui-6.0

[3].https://learn.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/tutorials/