**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**Giáo viên hướng dẫn: Gv. Phạm Văn Tiệp**

**Sinh viên thực hiện:**

**TRƯỜNG ĐẠI HỌC ĐẠI NAM**

Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**TÊN HỌC PHẦN: LẬP TRÌNH MOBILE**

**ĐỀ TÀI: XÂY DỰNG APP SMART CALENDAR - LỊCH THÔNG MINH**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Mã Sinh Viên** | **Họ và Tên** | **Ngày Sinh** | **Lớp** |
| 1 | 1671020136 | Chu Văn Huy | 20/01/2004 | CNTT 16-05 |

***Hà Nội, năm 2025***

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC ĐẠI NAM**

Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**TÊN HỌC PHẦN: LẬP TRÌNH MOBILE**

**ĐỀ TÀI: XÂY DỰNG APP SMART CALENDAR - LỊCH THÔNG MINH**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Mã Sinh Viên** | **Họ và Tên** | **Ngày Sinh** | **Điểm** | |
| **Bằng Số** | **Bằng Chữ** |
| **1** | **1671020136** | Chu Văn Huy | 20/01/2004 |  |  |

**CÁN BỘ CHẤM THI**

***Hà Nội, năm 2025***

# **LỜI NÓI ĐẦU**

Trong bối cảnh công nghệ ngày càng phát triển và nhu cầu quản lý thời gian của con người ngày một gia tăng, việc xây dựng một ứng dụng lịch thông minh để hỗ trợ người dùng trong việc lên kế hoạch và theo dõi hoạt động hàng ngày trở nên vô cùng cần thiết. Đề tài "Xây Dựng App Smart Calendar - Lịch Thông Minh" được thực hiện với mong muốn cung cấp một giải pháp bài bản và hiệu quả giúp người dùng dễ dàng lên lịch, nhắc nhở và quản lý thời gian của mình một cách tối ưu.

Chúng em quyết định sử dụng ngôn ngữ lập trình Flutter, một công nghệ hiện đại cho phép phát triển ứng dụng đa nền tảng với hiệu suất cao và giao diện người dùng trực quan. Dự án sẽ bao gồm các dependencies quan trọng như **lunar** để tính toán lịch âm, **shared\_preferences** để lưu trữ thông tin người dùng, **flutter\_local\_notifications** để gửi thông báo nhắc nhở, và nhiều thư viện khác như **timezone**, **audioplayers**, **intl** nhằm tăng cường tính năng cho ứng dụng.

Mục tiêu chính của đề tài không chỉ là phát triển một ứng dụng lịch thông minh có nhiều tính năng hữu ích mà còn tạo ra một công cụ tiện lợi giúp người dùng tối ưu hóa thời gian và nâng cao hiệu suất công việc của họ. Em hy vọng rằng sản phẩm cuối cùng sẽ mang lại giá trị thực tiễn cho người dùng và góp phần nâng cao trải nghiệm cá nhân trong việc quản lý thời gian.

Em xin gửi lời cảm ơn chân thành đến thầy Phạm Văn Tiệp đã cung cấp sự hướng dẫn và hỗ trợ quý báu trong suốt quá trình thực hiện dự án này. Đề tài không chỉ mang tính học thuật mà còn là cơ hội để em ứng dụng lý thuyết vào thực tiễn, thúc đẩy nhận thức về tầm quan trọng của việc quản lý thời gian và ứng dụng công nghệ trong cuộc sống hàng ngày.

# **MỤC LỤC**

[**LỜI NÓI ĐẦU** 3](#_Toc193798669)

[**MỤC LỤC** 4](#_Toc193798670)

[**DANH MỤC HÌNH ẢNH** 6](#_Toc193798671)

[**CHƯƠNG I. TỔNG QUAN VỀ ĐỀ TÀI** 7](#_Toc193798672)

[**1.1. Giới thiệu về đề tài** 7](#_Toc193798673)

[1.1.1 Giới thiệu 7](#_Toc193798674)

[1.1.2. Công nghệ sử dụng 7](#_Toc193798675)

[1.1.3. Tính năng chính 8](#_Toc193798676)

[1.1.4. Lợi ích của ứng dụng 9](#_Toc193798677)

[**1.2. Mục tiêu của đề tài** 10](#_Toc193798678)

[**1.3. Phạm vi của đề tài** 12](#_Toc193798679)

[**CHƯƠNG II : CƠ SỞ LÝ THUYẾT** 14](#_Toc193798680)

[**2.1. Tổng quan về Dart** 14](#_Toc193798681)

[**2.2. Giới thiệu về Flutter** 18](#_Toc193798682)

[**2.3. Visual Studio Code** 22](#_Toc193798683)

[**2.4. SharedPreferences** 26](#_Toc193798684)

[**CHƯƠNG III. THIẾT KẾ ỨNG DỤNG** 29](#_Toc193798685)

[**3.1. Ý tưởng thiết kế** 29](#_Toc193798686)

[3.1.1. Mục Tiêu của Ứng Dụng 29](#_Toc193798687)

[3.1.2. Công Nghệ Sử Dụng 29](#_Toc193798688)

[3.1.3. Tính Năng Chính 30](#_Toc193798689)

[3.1.4. Lợi Ích của Ứng Dụng 31](#_Toc193798690)

[**3.2. Sơ đồ ứng dụng** 32](#_Toc193798691)

[**3.3. Giao diện các trang chính** 33](#_Toc193798692)

[3.1.1. Trang chủ 33](#_Toc193798693)

[3.1.2. Trang lưu ngày 34](#_Toc193798694)

[3.1.3. Hẹn ngày báo 38](#_Toc193798695)

[3.1.4. Hành trình yêu 43](#_Toc193798696)

[3.1.5. Dò tìm cung hoàng đạo 47](#_Toc193798697)

[3.1.6. Tính ngày & giờ 49](#_Toc193798698)

[3.1.7. Giới thiệu 52](#_Toc193798699)

[**3.4. Code của một số trang chính** 54](#_Toc193798700)

[3.4.1. Thư viện và package hỗ trợ 54](#_Toc193798701)

[3.4.2. Code trang chủ 54](#_Toc193798702)

[3.4.3. Code ngày đã lưu 56](#_Toc193798703)

[3.4.4. Code hẹn ngày báo 59](#_Toc193798704)

[3.4.5. Code hành trình yêu 60](#_Toc193798705)

[3.4.6. Code tính Ngày & Giờ 61](#_Toc193798706)

[3.4.7. Code dò tìm cung hoàng đạo 62](#_Toc193798707)

[**KẾT LUẬN** 64](#_Toc193798708)

[**LỜI CẢM ƠN** 65](#_Toc193798709)

[**DANH MỤC TÀI LIỆU THAM KHẢO** 66](#_Toc193798710)

# **DANH MỤC HÌNH ẢNH**

Hình 1: Dart

Hình 2: Flutter

Hình 3: Visual Studio Code

Hình 4: Shared Preferences

Hình 5: Biểu đồ use case

Hình 6: sơ đồ ứng dụng

Hình 7: Trang chủ

Hình 8: Thêm ngày vào danh sách

Hình 9: Danh sách ngày đã lưu

Hình 10: Sửa ngày đã lưu

Hình 11: Sóa ngày đã lưu

Hình 12: Thêm ngày báo

Hình 13: Sửa ngày báo

Hình 14: Danh sách ngày báo

Hình 15: Đến ngày báo

Hình 16: Giao diện hành trình yêu

Hình 17: Thêm thông tin cặp đôi

Hình 18: Kết quả

Hình 19: Thêm ảnh cặp đôi

Hình 20: Reset (chia tay)

Hình 21: Dò tìm cung hoàng đạo

Hình 22: Tính số ngày

Hình 23: Kết quả tính số ngày

Hình 24: Tính số giờ

Hình 25: Kết quả tính số giờ

Hình 26: Kết quả tính giờ ra phút

Hình 27: Giới thiệu

**CHƯƠNG I. TỔNG QUAN VỀ ĐỀ TÀI**

## **1.1. Giới thiệu về đề tài**

### 1.1.1 Giới thiệu

Trong thời đại công nghệ số phát triển mạnh mẽ, việc quản lý thời gian hiệu quả là nhu cầu thiết yếu của mỗi cá nhân. Ứng dụng **Smart Calendar – Lịch Thông Minh** ra đời nhằm hỗ trợ người dùng trong việc sắp xếp, nhắc nhở và tối ưu hóa lịch trình cá nhân, giúp nâng cao hiệu suất làm việc và cân bằng cuộc sống.

Ứng dụng được phát triển trên nền tảng **Flutter**, đảm bảo tính linh hoạt, hiệu suất cao và khả năng tương thích với nhiều hệ điều hành như Android và iOS. Smart Calendar tích hợp nhiều tính năng thông minh như đồng bộ hóa dữ liệu, nhắc nhở theo múi giờ, quản lý sự kiện cá nhân, hỗ trợ lịch âm và nhiều tiện ích khác. Ngoài ra, ứng dụng còn hỗ trợ người dùng đính kèm tài liệu, hình ảnh và tùy chỉnh giao diện theo sở thích cá nhân.

### 1.1.2. Công nghệ sử dụng

* **Ngôn ngữ lập trình**: Dart
* **Framework**: Flutter
* **Cơ sở dữ liệu**: Shared Preferences
* **Thư viện và package hỗ trợ**:
  + **lunar**: Hỗ trợ hiển thị lịch âm, phù hợp với nhu cầu sử dụng của người Việt Nam.
  + **shared\_preferences**: Lưu trữ dữ liệu cục bộ để giữ lại thông tin người dùng.
  + **path\_provider**: Hỗ trợ xác định vị trí lưu trữ dữ liệu trong thiết bị.
  + **flutter\_local\_notifications**: Cung cấp chức năng hiển thị thông báo nhắc nhở sự kiện theo lịch trình.
  + **timezone**: Xử lý múi giờ, đảm bảo thông báo chính xác theo địa phương.
  + **audioplayers**: Hỗ trợ phát âm thanh thông báo cho các sự kiện quan trọng.
  + **intl**: Hỗ trợ định dạng ngày tháng theo nhiều ngôn ngữ khác nhau.
  + **image\_picker**: Cho phép người dùng tải ảnh lên làm hình đại diện sự kiện.
  + **file\_picker**: Hỗ trợ chọn và đính kèm tài liệu vào sự kiện.

### 1.1.3. Tính năng chính

**1. Lịch âm – dương**

* Hiển thị lịch theo cả hai hệ thống âm lịch và dương lịch.
* Người dùng có thể dễ dàng chuyển đổi giữa hai chế độ lịch để theo dõi các ngày quan trọng như ngày lễ, Tết.
* Hỗ trợ hiển thị thông tin về các ngày đặc biệt theo phong tục Việt Nam.

**2. Lưu ngày (Quản lý sự kiện cá nhân)**

* Cho phép người dùng thêm, sửa, xóa các sự kiện theo lịch trình cá nhân.
* Hỗ trợ phân loại sự kiện theo chủ đề (công việc, học tập, cá nhân, sức khỏe,...).
* Người dùng có thể đặt mức độ ưu tiên cho từng sự kiện để dễ dàng theo dõi.
* Hỗ trợ tính năng tìm kiếm và lọc sự kiện theo ngày, tháng, năm.

**3. Hẹn ngày báo**

* Cài đặt thông báo nhắc nhở sự kiện theo thời gian mong muốn.
* Hỗ trợ âm thanh và rung để cảnh báo các sự kiện quan trọng.
* Tích hợp hiển thị thông báo trên màn hình khóa.
* Nhắc nhở có thể được tùy chỉnh theo múi giờ để phù hợp với lịch trình cá nhân.

**4. Hành trình yêu**

* Cho phép người dung thêm ảnh và đếm ngày yêu
* Giao diện màu dễ chịu lãng mạng
* Giúp dễ dàng theo dõi số ngày yêu

**5. Dò tìm cung hoàng đạo**

* Xác định cung hoàng đạo theo ngày tháng năm sinh.
* Đưa ra thông tin về cung hoàng đạo đó.

**6. Tính ngày & giờ**

* Tính ngày và giờ cho người dùng
* Hỗ trợ chuyển số giờ ra phút

### 1.1.4. Lợi ích của ứng dụng

* Giúp người dùng tổ chức công việc một cách khoa học, giảm thiểu tình trạng quên lịch hẹn.
* Hỗ trợ làm việc nhóm thông qua việc chia sẻ sự kiện (có thể phát triển thêm trong tương lai).
* Tích hợp công nghệ thông báo thông minh giúp người dùng không bỏ lỡ sự kiện quan trọng.
* Mang lại trải nghiệm cá nhân hóa thông qua tùy chỉnh giao diện và cài đặt thông báo linh hoạt.

**Kết luận**

Với sự phát triển của công nghệ và nhu cầu quản lý thời gian ngày càng cao, **Smart Calendar – Lịch Thông Minh** là một giải pháp hữu ích, giúp người dùng tối ưu hóa công việc và cuộc sống hàng ngày. Ứng dụng được xây dựng trên nền tảng Flutter, tích hợp nhiều tính năng hiện đại, hứa hẹn mang đến trải nghiệm tốt nhất cho người dùng.

Ứng dụng sẽ không chỉ dừng lại ở việc quản lý lịch trình cá nhân mà còn có thể mở rộng để hỗ trợ làm việc nhóm, chia sẻ sự kiện và đồng bộ dữ liệu qua nền tảng đám mây, mang đến trải nghiệm quản lý thời gian toàn diện hơn.

## **1.2. Mục tiêu của đề tài**

**Mục tiêu cụ thể của ứng dụng**

1. **Tối ưu hóa quản lý thời gian:**
   * **Lập kế hoạch cụ thể:** Ứng dụng Smart Calendar sẽ cho phép người dùng tạo và quản lý lịch trình theo từng khoảng thời gian, từ hàng ngày, hàng tuần đến hàng tháng. Với khả năng phân loại và ưu tiên sự kiện, người dùng có thể dễ dàng nhận diện điều gì cần được hoàn thành trước.
   * **Giảm thiểu thời gian lãng phí:** Các tính năng như nhắc nhở thông minh và thông báo kịp thời sẽ giúp người dùng không bỏ lỡ các cuộc hẹn hay nhiệm vụ quan trọng, từ đó tối ưu hóa thời gian làm việc và nghỉ ngơi.
   * **Phân tích hiệu suất:** Ứng dụng có thể tích hợp các báo cáo và thống kê nhằm giúp người dùng theo dõi hiệu suất làm việc của mình, từ đó tìm ra phương pháp cải tiến thông qua các biện pháp như giảm thiểu các hoạt động không cần thiết.
2. **Cải thiện trải nghiệm người dùng:**
   * **Thiết kế giao diện thân thiện:** Dựa trên nguyên tắc thiết kế hiện đại, ứng dụng sẽ có một giao diện trực quan, dễ dàng điều hướng, giúp người dùng nhanh chóng làm quen và sử dụng hiệu quả mà không cần quá nhiều thời gian tìm hiểu.
   * **Tương tác mượt mà:** Các tính năng được tối ưu hóa để mang lại trải nghiệm người dùng liền mạch, giúp thao tác như thêm, sửa, xóa sự kiện trở nên đơn giản và nhanh chóng.
   * **Phản hồi tức thì:** Ứng dụng cung cấp phản hồi ngay lập tức sau mỗi hành động của người dùng, chẳng hạn như thông báo khi sự kiện được thêm vào thành công hoặc thông báo khi cần thực hiện các nhiệm vụ đã lên kế hoạch.
3. **Khuyến khích thói quen lên kế hoạch:**
   * **Nhắc nhở thông minh:** Ứng dụng cung cấp các chức năng nhắc nhở tự động cho các sự kiện quan trọng, giúp người dùng hình thành thói quen kiểm tra lịch trình và lập kế hoạch cho các hoạt động trong ngày hoặc tuần.
   * **Gợi ý kế hoạch:** Ứng dụng có thể đưa ra các gợi ý về việc sắp xếp các sự kiện dựa trên thói quen và lịch sử sử dụng của người dùng, từ đó khuyến khích họ lên kế hoạch một cách chủ động hơn.
   * **Chia sẻ và hợp tác:** Tính năng chia sẻ sự kiện với bạn bè, đồng nghiệp sẽ kích thích người dùng tạo dựng thói quen làm việc theo nhóm và tương tác xã hội, đồng thời tạo ra một môi trường làm việc tích cực.
4. **Khả năng mở rộng:**
   * **Tích hợp nhiều tính năng:** Trong tương lai, ứng dụng có thể mở rộng bằng cách tích hợp các tính năng mới như đồng bộ hóa với các ứng dụng khác (gMail, Google Calendar, etc.), giúp người dùng có một cái nhìn tổng quát hơn về lịch trình.
   * **Phát triển API:** Cung cấp các API cho phép các nhà phát triển bên ngoài có thể tích hợp dịch vụ của mình vào ứng dụng, từ đó nâng cao giá trị sử dụng và trải nghiệm của người dùng.
   * **Thích ứng theo nhu cầu người dùng:** Dựa trên phản hồi từ người dùng, ứng dụng sẽ được cập nhật định kỳ để cải thiện các tính năng hiện tại và triển khai tính năng mới cập nhật theo xu hướng và nhu cầu thực tế của thị trường.

**Kết luận**

Mục tiêu cụ thể của ứng dụng Smart Calendar không chỉ dừng lại ở việc quản lý thời gian mà còn bao gồm việc cải thiện trải nghiệm người dùng, khuyến khích thói quen lên kế hoạch, và cung cấp khả năng mở rộng để đáp ứng nhu cầu ngày càng đa dạng và phức tạp của người dùng. Điều này sẽ đảm bảo rằng ứng dụng không chỉ là một công cụ tiện ích mà còn là một phần không thể thiếu trong cuộc sống hàng ngày của người dùng.

## **1.3. Phạm vi của đề tài**

Phạm vi của đề tài "Xây dựng App Smart Calendar - Lịch Thông Minh" được định rõ dưới đây:

1. **Đối tượng sử dụng:**
   * **Người dùng cá nhân:** Ứng dụng sẽ phục vụ cho các cá nhân muốn tổ chức và quản lý thời gian hiệu quả, bao gồm học sinh, sinh viên, nhân viên văn phòng, và những người có nhu cầu theo dõi lịch trình cá nhân.
   * **Tổ chức và nhóm công việc:** Hỗ trợ các nhóm trong tổ chức hoặc tại nơi làm việc trong việc lập kế hoạch và chia sẻ sự kiện. Ứng dụng có thể mở rộng để phục vụ cho việc làm việc nhóm và hợp tác hiệu quả giữa các thành viên.
2. **Nền tảng phát triển:**
   * Ứng dụng sẽ được phát triển chủ yếu cho các hệ điều hành di động như **Android** và **iOS** thông qua việc sử dụng Flutter. Điều này sẽ giúp tăng cường khả năng tiếp cận và sử dụng trên nhiều thiết bị khác nhau.
3. **Tính năng chính:**
   * Phạm vi nghiên cứu và phát triển sẽ bao gồm các tính năng đã trình bày trong mục 1.1.3, bao gồm lịch âm-dương, quản lý sự kiện cá nhân, nhắc nhở thông minh, hành trình yêu, dò tìm cung hoàng đạo, và tính ngày giờ. Các tính năng sẽ được phát triển và tối ưu hóa dựa trên phản hồi từ người dùng.
   * Không bao gồm các tính năng không phổ biến hoặc ít ảnh hưởng đến trọng tâm của việc quản lý thời gian, chẳng hạn như các tính năng phức tạp liên quan đến tài chính hay quản lý dự án chuyên sâu.
4. **Thời gian phát triển:**
   * Dự kiến, thời gian phát triển của ứng dụng sẽ kéo dài từ 6 đến 12 tháng, trong đó bao gồm các giai đoạn nghiên cứu, thiết kế giao diện, phát triển, kiểm thử, và hoàn thiện ứng dụng. Nếu có khả năng, ứng dụng sẽ được phát triển theo phương pháp Agile để có thể điều chỉnh linh hoạt theo phản hồi từ người dùng trong suốt quá trình phát triển.
5. **Giới hạn về dữ liệu:**
   * Ứng dụng sẽ không thu thập hoặc xử lý các dữ liệu nhạy cảm hoặc cá nhân của người dùng ngoài các thông tin cần thiết để vận hành, như tên, ngày tháng năm sinh, và sự kiện cá nhân. Tất cả dữ liệu sẽ được bảo vệ và lưu trữ an toàn theo quy định của GDPR và các luật lệ bảo mật dữ liệu khác.
6. **Hỗ trợ và bảo trì:**
   * Sau khi ra mắt, ứng dụng sẽ tiếp tục nhận được hỗ trợ và bảo trì định kỳ để đảm bảo tính ổn định và hiệu suất sử dụng. Các bản cập nhật sẽ được phát hành theo định kỳ để thêm tính năng mới, cải thiện hiệu suất, và khắc phục lỗi.

**Kết luận**

Phạm vi của đề tài "Xây dựng App Smart Calendar Lịch Thông Minh" được đặt ra rõ ràng nhằm xác định mục tiêu cụ thể và đảm bảo rằng quá trình phát triển ứng dụng sẽ tập trung vào những vấn đề thiết yếu nhất liên quan đến quản lý thời gian và trải nghiệm người dùng. Qua đó, ứng dụng sẽ trở thành một phần không thể thiếu trong việc hỗ trợ người dùng trong việc tổ chức cuộc sống hàng ngày của họ.

# **CHƯƠNG II : CƠ SỞ LÝ THUYẾT**

## **2.1. Tổng quan về Dart**



*Hình 1: Dart*

**Dart** là một ngôn ngữ lập trình hướng đối tượng, được phát triển bởi Google, với mục tiêu chính là phục vụ cho việc xây dựng ứng dụng web, di động và máy tính để bàn. Với sự phát triển không ngừng và cộng đồng ngày càng lớn mạnh, Dart đã trở thành một phần không thể thiếu trong hệ sinh thái lập trình hiện đại, đặc biệt với sự nổi bật của Flutter.

**1. Tính năng nổi bật của Dart**

* **Lập trình hướng đối tượng:**
  + Dart hoàn toàn hỗ trợ lập trình hướng đối tượng với các khái niệm như lớp (class), đối tượng (object), kế thừa (inheritance), và đa hình (polymorphism). Điều này giúp lập trình viên phát triển ứng dụng một cách có tổ chức và dễ quản lý.
* **Kiểu dữ liệu tĩnh (Static Type) và động (Dynamic Type):**
  + Dart hỗ trợ cả kiểu dữ liệu tĩnh, cho phép lập trình viên định nghĩa kiểu dữ liệu khi khai báo biến, cũng như kiểu động, giúp linh hoạt hơn trong phần lớn các tình huống. Việc sử dụng kiểu tĩnh giúp phát hiện lỗi sớm tại thời điểm biên dịch.
* **Hệ thống null-safety:**
  + Dart hỗ trợ null-safety, giúp các lập trình viên tránh được các lỗi phổ biến liên quan đến giá trị null. Kiểu dữ liệu có thể được khai báo là không cho phép null bằng cách sử dụng dấu "?", từ đó đảm bảo an toàn hơn cho mã nguồn.
* **Tính năng bất đồng bộ (Asynchronous):**
  + Dart hỗ trợ lập trình bất đồng bộ thông qua async, await, và Future, giúp xử lý các tác vụ như gọi API mà không làm treo giao diện, nâng cao trải nghiệm người dùng.

**2. Cú pháp và cấu trúc ngôn ngữ**

* **Cú pháp dễ đọc và dễ bảo trì:**
  + Cú pháp của Dart được thiết kế để dễ đọc và gần gũi với những lập trình viên đã quen thuộc với các ngôn ngữ như Java, JavaScript và C#. Ví dụ, bạn có thể khai báo biến bằng cú pháp sau:

int age = 25;

String name = "John";

* **Các cấu trúc điều khiển:**
  + Dart hỗ trợ đầy đủ các cấu trúc điều khiển như điều kiện (if, else), vòng lặp (for, while), và các cấu trúc điều kiện khác như switch.
* **Hàm (Functions):**
  + Hàm là một phần quan trọng trong Dart. Bạn có thể định nghĩa hàm đơn giản hoặc hàm ẩn danh (anonymous function), và truyền tham số một cách linh hoạt.

void greet(String name) {

print("Hello, $name!");

}

**3. Thư viện và công cụ phát triển**

* **Thư viện tích hợp:**
  + Dart đi kèm với một bộ thư viện phong phú, với các chức năng đa dạng như:
    - **Dart Standard Library:** Bao gồm các thư viện cho xử lý chuỗi, danh sách, kết nối mạng, xử lý JSON, và nhiều hơn nữa.
    - **Gói (Package):** Dart có một hệ thống gói (Pub) giúp lập trình viên dễ dàng cài đặt và sử dụng thư viện của bên thứ ba.
* **Công cụ phát triển:**
  + **IDE và Editor:** Dart có thể được phát triển trong nhiều công cụ khác nhau như Visual Studio Code, Android Studio và IntelliJ IDEA. Những công cụ này cung cấp tính năng tự động hoàn thành mã, phân tích mã và gỡ lỗi tiện lợi.
  + **DartPad:** Đây là một công cụ trực tuyến cho phép lập trình viên chỉ cần trình duyệt web để viết, chạy và chia sẻ mã Dart. DartPad là một cách tuyệt vời để thử nghiệm các tính năng và khái niệm mới mà không cần cài đặt phần mềm.

**4. Tích hợp với Flutter**

* Flutter là một framework phát triển ứng dụng di động, máy tính để bàn và web cũng được phát triển bởi Google, sử dụng Dart làm ngôn ngữ chính. Lợi ích của việc sử dụng Dart với Flutter bao gồm:
  + **Phát triển nhanh chóng:** Thời gian lấy lại tức thì (hot reload) giúp lập trình viên thử nghiệm và kiểm tra giao diện người dùng nhanh chóng.
  + **Giao diện người dùng tùy chỉnh:** Flutter cho phép tạo ra giao diện người dùng tùy chỉnh và đồ họa đẹp mắt nhanh chóng, với khả năng chạy trên nhiều nền tảng.
  + **Hệ sinh thái mạnh mẽ:** Dart hỗ trợ nhiều gói Flutter, cho phép lập trình viên dễ dàng tích hợp các tính năng như xác thực người dùng, lưu trữ dữ liệu và hơn thế nữa.

**5. Cộng đồng và sự phát triển**

* **Sự phát triển nhanh chóng:** Dart đang thu hút một cộng đồng lập trình viên lớn mạnh và tích cực. Sự gia tăng sử dụng Flutter trên toàn thế giới đã góp phần làm gia tăng lượng người sử dụng Dart.
* **Tài liệu và hỗ trợ:** Google cung cấp tài liệu đầy đủ và chi tiết cho Dart tại [trang web chính thức của Dart](https://dart.dev/), bao gồm cả hướng dẫn, tutorial và các ví dụ minh họa.
* **Sự kiện và hội thảo:** Các sự kiện lập trình và hội thảo như DartConf thường xuyên được tổ chức, tạo cơ hội cho cộng đồng lập trình viên tương tác và chia sẻ kiến thức.

**6. Ứng dụng thực tiễn**

* **Sử dụng rộng rãi trong doanh nghiệp:** Dart được sử dụng để phát triển nhiều ứng dụng doanh nghiệp và sản phẩm tiêu dùng. Không chỉ dừng lại ở các ứng dụng di động, Dart được áp dụng trong việc phát triển ứng dụng web và máy tính để bàn.
* **Ví dụ thực tiễn:**
  + **Google Ads:** Ứng dụng quản lý quảng cáo sử dụng Dart để tạo ra các trải nghiệm người dùng mượt mà.
  + **Alibaba:** Nền tảng thương mại điện tử nổi tiếng này sử dụng Dart và Flutter để phát triển ứng dụng di động của họ, giúp nâng cao trải nghiệm người dùng và tối ưu hóa hiệu suất.

**Kết luận**

Dart là một ngôn ngữ lập trình mạnh mẽ, linh hoạt, và hiện đại, phù hợp với nhu cầu phát triển ứng dụng đa nền tảng trong thời đại công nghệ hiện nay. Sự kết hợp giữa Dart và Flutter tạo ra một môi trường phát triển lý tưởng cho các lập trình viên, nhấn mạnh khả năng mở rộng và hiệu suất cao trong công việc phát triển ứng dụng. Với cộng đồng đang phát triển và tài liệu phong phú, Dart là một lựa chọn hấp dẫn cho bất kỳ lập trình viên nào muốn tìm kiếm một ngôn ngữ lập trình đa năng và hiệu quả.

## **2.2. Giới thiệu về Flutter**

 *Hình 2: Flutter*

**Flutter** là một framework mã nguồn mở do Google phát triển, được sử dụng để xây dựng ứng dụng di động, web và máy tính để bàn từ một cơ sở mã duy nhất. Kể từ khi ra mắt, Flutter đã nhanh chóng trở thành một trong những công cụ phát triển ứng dụng phổ biến nhất, nhờ vào khả năng tạo ra các ứng dụng có giao diện hấp dẫn và hiệu suất cao.

**1. Đặc điểm nổi bật của Flutter**

* **Mã nguồn mở:**
  + Flutter là mã nguồn mở, cho phép lập trình viên tùy chỉnh và đóng góp vào dự án. Điều này cũng giúp tăng cường sự hỗ trợ từ cộng đồng cùng những cải tiến liên tục.
* **Giao diện người dùng (UI) tuyệt đẹp:**
  + Flutter cho phép lập trình viên tạo ra các giao diện người dùng đẹp mắt và tùy chỉnh cao với widget (tiện ích). Các widget này cung cấp khả năng tái sử dụng mã và dễ dàng kết hợp với nhau.
* **Hiệu suất cao:**
  + Flutter biên dịch trực tiếp mã nguồn thành mã máy thông qua biên dịch Ahead-Of-Time (AOT), giúp tối ưu hóa tốc độ thực thi và khả năng phản hồi của ứng dụng.
* **Phát triển nhanh chóng với Hot Reload:**
  + Tính năng Hot Reload cho phép lập trình viên xem ngay lập tức những thay đổi mà họ thực hiện trong mã mà không cần khởi động lại ứng dụng, giúp nâng cao khả năng phát triển và thử nghiệm.
* **Hỗ trợ nhiều nền tảng:**
  + Flutter cho phép phát triển ứng dụng cho iOS, Android, web và máy tính để bàn (Windows, macOS, Linux) từ cùng một mã nguồn duy nhất. Điều này tiết kiệm thời gian và công sức phát triển.

**2. Cấu trúc của Flutter**

* **Widget:**
  + Widget là các khối xây dựng cơ bản của một ứng dụng Flutter. Mỗi thành phần của giao diện người dùng như nút bấm, hình ảnh, hay danh sách đều được xây dựng từ widget. Flutter cung cấp một thư viện widget phong phú cho nhiều mục đích khác nhau.
* **Chế độ lập trình phản ứng:**
  + Flutter sử dụng mô hình lập trình phản ứng, nơi mà giao diện người dùng tự động cập nhật khi trạng thái của ứng dụng thay đổi. Điều này giúp đơn giản hóa việc quản lý trạng thái và cải thiện trải nghiệm người dùng.
* **Cấu trúc dự án:**
  + Dự án Flutter thường bao gồm các thư mục như lib (chứa mã nguồn), pubspec.yaml (quản lý phụ thuộc và cấu hình ứng dụng), và test (chứa bài kiểm tra).

**3. Hệ sinh thái và công cụ phát triển**

* **Dart:**
  + Flutter được viết bằng ngôn ngữ lập trình Dart, cũng do Google phát triển. Dart cung cấp tính năng lập trình hiện đại, hỗ trợ lập trình hướng đối tượng và mô hình lập trình bất đồng bộ.
* **Công cụ phát triển:**
  + Flutter có thể phát triển bằng nhiều công cụ và IDE như Android Studio, Visual Studio Code, và IntelliJ IDEA, với các plugin hỗ trợ cho Flutter.
* **Thư viện và gói Flutter:**
  + Flutter hỗ trợ một hệ thống gói mạnh mẽ để thư viện bên thứ ba, giúp lập trình viên dễ dàng tích hợp các tính năng như xác thực người dùng, lưu trữ dữ liệu, hay tích hợp API.
  + **pub.dev:** Đây là kho lưu trữ chính thức cho các gói Dart và Flutter, bao gồm hàng nghìn gói mã nguồn mở có sẵn để người dùng sử dụng.

**4. Khám phá các đặc điểm kỹ thuật**

* **Khả năng tái sử dụng mã:**
  + Flutter cho phép lập trình viên sử dụng một cơ sở mã duy nhất để phát triển ứng dụng cho nhiều nền tảng, từ di động đến web và máy tính để bàn, giúp tiết kiệm thời gian phát triển và tài nguyên.
* **Thời gian tải cực nhanh:**
  + Khả năng biên dịch nhanh chóng giúp ứng dụng trên Flutter khởi động và chạy rất nhanh, mang đến trải nghiệm thú vị cho người dùng.
* **Tùy chỉnh UI mạnh mẽ:**
  + Flutter cho phép tạo ra các giao diện người dùng phức tạp, từ những thành phần đơn giản đến các bố cục phức tạp mà không gặp phải vấn đề về tính nhất quán giữa các nền tảng.

**5. Ứng dụng thực tiễn và thành công nổi bật**

* **Ứng dụng di động:**
  + Flutter được sử dụng để phát triển nhiều ứng dụng di động nổi tiếng, như:
    - **Google Ads:** Ứng dụng cho phép người dùng quản lý quảng cáo trên điện thoại một cách dễ dàng.
    - **Alibaba:** Nền tảng thương mại điện tử lớn sử dụng Flutter để tối ưu hóa trải nghiệm người dùng và hiệu suất.
* **Ứng dụng web và máy tính để bàn:**
  + Flutter đã mở rộng khả năng đến phát triển ứng dụng web, với nhiều dự án đang trong giai đoạn thử nghiệm, cho phép các lập trình viên sử dụng Flutter để tạo ra các ứng dụng web hiện đại.

**6. Cộng đồng và hỗ trợ**

* **Cộng đồng phát triển mạnh mẽ:**
  + Flutter có một cộng đồng lớn và tích cực với nhiều diễn đàn, nhóm thảo luận và tài liệu hỗ trợ. Cộng đồng thường xuyên tổ chức các buổi hội thảo, meetup và hackathon.
* **Tài liệu phong phú:**
  + Google cung cấp tài liệu chi tiết và hướng dẫn cho lập trình viên làm quen với Flutter tại trang web chính thức [flutter.dev](https://flutter.dev/), bao gồm cả bài học, ví dụ và tài liệu tham khảo.

**Kết luận**

Flutter đã chứng minh mình là một framework hiện đại và mạnh mẽ cho phát triển ứng dụng, nhờ vào thiết kế thân thiện, khả năng tùy chỉnh cao và hiệu suất tuyệt vời. Với sự hỗ trợ mạnh mẽ từ Google và cộng đồng lập trình viên, Flutter sẽ tiếp tục phát triển và tạo ra nhiều ứng dụng hấp dẫn trong tương lai. Với khả năng cung cấp trải nghiệm người dùng tốt và giảm thiểu thời gian phát triển, Flutter thực sự là một công cụ không thể thiếu cho các lập trình viên trong ngành công nghiệp công nghệ hiện đại.

## **2.3. Visual Studio Code**

 *Hình 3: Visual Studio Code*

**Visual Studio Code (VS Code)** là một trình soạn thảo mã nguồn mở và miễn phí do Microsoft phát triển. Ra mắt lần đầu vào tháng 4 năm 2015, VS Code đã nhanh chóng trở thành một trong những công cụ phát triển phổ biến nhất trên toàn thế giới nhờ vào tính linh hoạt, khả năng mở rộng, và mạnh mẽ.

**1. Tính năng nổi bật của VS Code**

* **Giao diện người dùng thân thiện:**
  + VS Code có giao diện người dùng trực quan, dễ sử dụng với bố cục linh hoạt cho phép lập trình viên tùy chỉnh theo sở thích cá nhân. Các tab và cửa sổ có thể được sắp xếp lại dễ dàng, giúp cải thiện trải nghiệm làm việc.
* **Hỗ trợ nhiều ngôn ngữ lập trình:**
  + VS Code hỗ trợ nhiều ngôn ngữ lập trình phổ biến như JavaScript, TypeScript, Python, Java, C++, PHP, Go, Ruby và nhiều khác thông qua các tiện ích mở rộng (extensions). Điều này cho phép lập trình viên làm việc trên nhiều dự án khác nhau mà không cần chuyển đổi giữa các công cụ.
* **Tính năng IntelliSense:**
  + IntelliSense cung cấp tính năng gợi ý mã, tự động hoàn thành và hiển thị thông tin về cú pháp, giúp lập trình viên viết mã nhanh chóng và chính xác hơn. Tính năng này giúp giảm thiểu lỗi và tăng tốc quá trình phát triển.
* **Hệ thống quản lý gói tích hợp:**
  + VS Code tích hợp với Git, cho phép quản lý mã nguồn dễ dàng ngay trong trình soạn thảo. Bạn có thể thực hiện các thao tác như commit, push, pull, kiểm tra lịch sử, và tạo nhánh mà không cần rời khỏi môi trường làm việc.
* **Tính năng Debug mạnh mẽ:**
  + VS Code hỗ trợ gỡ lỗi (debugging) ngay trong trình soạn thảo. Bạn có thể đặt breakpoint, theo dõi biến và kiểm tra các giá trị trong quá trình thực thi mã. Điều này giúp lập trình viên phát hiện lỗi và sửa chữa một cách hiệu quả.
* **Tính năng tìm kiếm và thay thế nâng cao:**
  + Tính năng tìm kiếm cho phép bạn tìm kiếm mã trong toàn bộ dự án, và bạn có thể thay thế nhanh chóng các đoạn mã hay cụm từ cụ thể.

**2. Tiện ích mở rộng (Extensions)**

* **Marketplace phong phú:**
  + VS Code có một Marketplace rộng lớn, nơi bạn có thể tìm thấy hàng ngàn tiện ích mở rộng cho nhiều mục đích khác nhau, từ hỗ trợ ngôn ngữ, công cụ phát triển, đến các trình biên dịch và định dạng mã.
* **Cài đặt và quản lý tiện ích mở rộng:**
  + Việc cài đặt tiện ích mở rộng rất đơn giản, chỉ cần vài cú nhấp chuột để cài đặt và kích hoạt. Bạn cũng có thể dễ dàng quản lý, cập nhật hay gỡ bỏ các tiện ích mở rộng theo nhu cầu.

**3. Hỗ trợ nền tảng và tích hợp**

* **Tương thích đa nền tảng:**
  + VS Code có thể chạy trên nhiều hệ điều hành như Windows, macOS, và Linux, giúp lập trình viên không bị giới hạn trong môi trường phát triển.
* **Tích hợp với các dịch vụ đám mây:**
  + Bạn có thể tích hợp VS Code với các dịch vụ đám mây như GitHub, Azure DevOps, và Docker, giúp tối ưu hóa quy trình phát triển và triển khai ứng dụng.

**4. Tùy chỉnh và cấu hình**

* **Cấu hình cá nhân hóa:**
  + VS Code cho phép lập trình viên tùy chỉnh nhiều khía cạnh của trình soạn thảo, từ màu sắc giao diện, font chữ, đến các phím tắt và bố cục. Bạn có thể dễ dàng chỉnh sửa tệp cấu hình settings.json để cá nhân hóa môi trường làm việc.
* **Themes và icon:**
  + Có nhiều chủ đề và biểu tượng khác nhau để thay đổi giao diện của VS Code, giúp bạn tạo ra không gian làm việc phù hợp với phong cách cá nhân.

**5. Quản lý dự án**

* **Sắp xếp dự án:**
  + VS Code cho phép bạn dễ dàng mở một thư mục dự án cụ thể, và quản lý các tệp tin trong đó một cách dễ dàng thông qua thanh bên (sidebar). Bạn có thể nhanh chóng mở, tạo, hoặc xóa tệp từ thanh bên.
* **Tính năng Workspace:**
  + Bạn có thể tạo workspace tùy chỉnh để quản lý nhiều dự án khác nhau trong cùng một cửa sổ. Điều này giúp tiết kiệm thời gian và tổ chức công việc tốt hơn.

**6. Cộng đồng và hỗ trợ**

* **Cộng đồng người dùng:**
  + VS Code có một cộng đồng người dùng rất lớn và tích cực. Nhiều diễn đàn, nhóm phát triển, và tài liệu hướng dẫn sẵn có để giải đáp thắc mắc và chia sẻ kiến thức.
* **Tài liệu hướng dẫn:**
  + Microsoft cung cấp tài liệu hướng dẫn chi tiết cho VS Code, bao gồm cả các bài học, ví dụ và hướng dẫn sử dụng cụ thể cho từng tính năng. Bạn có thể truy cập tài liệu này tại [Visual Studio Code Docs](https://code.visualstudio.com/docs).

**7. Ứng dụng thực tế**

* **Sử dụng trong phát triển phần mềm:**
  + VS Code được sử dụng rộng rãi trong các dự án phát triển phần mềm, từ các ứng dụng web đến ứng dụng di động và máy tính để bàn. Nhiều công ty công nghệ lớn cũng đã chọn VS Code làm công cụ chính để phát triển.
* **Hỗ trợ giáo dục:**
  + Nhiều trường học và tổ chức giáo dục cũng sử dụng VS Code làm công cụ giảng dạy, nhờ vào tính dễ sử dụng và khả năng hỗ trợ nhiều ngôn ngữ.

**Kết luận**

Visual Studio Code là một công cụ phát triển mạnh mẽ và linh hoạt, thích hợp cho cả lập trình viên mới vào nghề và chuyên gia. Với những tính năng nổi bật, khả năng mở rộng qua các tiện ích và hỗ trợ đa nền tảng, VS Code không chỉ giúp tăng năng suất làm việc mà còn mang lại trải nghiệm làm việc thú vị. Với sự hỗ trợ từ cộng đồng và tài liệu phong phú, Visual Studio Code xứng đáng là một trong những lựa chọn hàng đầu cho các lập trình viên trên toàn thế giới.

## **2.4. SharedPreferences**

 *Hình 4: Shared Preferences*

**SharedPreferences** là một API trong Android được sử dụng để lưu trữ và quản lý dữ liệu đơn giản dưới dạng cặp khóa-giá trị. Đây là một giải pháp lý tưởng để lưu trữ các thông tin cấu hình, thiết lập người dùng, hoặc dữ liệu không lớn mà không cần sử dụng tới cơ sở dữ liệu phức tạp như SQLite. Dưới đây là một giới thiệu chi tiết về SharedPreferences.

Các Đặc Điểm Chính của SharedPreferences

1. **Lưu Trữ Dữ Liệu Đơn Giản**: SharedPreferences cho phép bạn lưu trữ các kiểu dữ liệu cơ bản như String, int, float, long, và boolean. Thích hợp cho các dữ liệu cấu hình, trạng thái người dùng.
2. **Dễ Sử Dụng**: API của SharedPreferences rất dễ hiểu và sử dụng. Bạn có thể dễ dàng lưu trữ và truy xuất dữ liệu chỉ với một vài dòng mã.
3. **Lưu Trữ Không Có Cấu Trúc**: SharedPreferences không phải là một cơ sở dữ liệu, do đó bạn không cần phải thiết kế bảng hay cấu trúc phức tạp.
4. **Quản Lý Cấu Hình**: Thích hợp để quản lý các thiết lập người dùng như chủ đề, ngôn ngữ, và các tùy chọn tùy chỉnh khác.
5. **Tự Động Quản Lý**: Dữ liệu được lưu trữ sẽ tự động chuyển đổi giữa các phiên làm việc. Khi người dùng mở lại ứng dụng, tất cả dữ liệu trước đó vẫn có thể truy cập được.

Cách Sử Dụng SharedPreferences

Dưới đây là hướng dẫn từng bước về cách sử dụng SharedPreferences trong ứng dụng Android:

**1. Khởi Tạo SharedPreferences**

Bạn có thể khởi tạo SharedPreferences bằng phương thức getSharedPreferences():

SharedPreferences sharedPreferences = getSharedPreferences("MyPrefs", Context.MODE\_PRIVATE);

* MyPrefs: Tên của file lưu trữ.
* Context.MODE\_PRIVATE: Chế độ truy cập, chỉ ứng dụng của bạn có thể đọc/ghi.

**2. Lưu Dữ Liệu vào SharedPreferences**

Để lưu dữ liệu, bạn cần tạo một Editor từ SharedPreferences:

SharedPreferences.Editor editor = sharedPreferences.edit();

editor.putString("key\_name", "value\_name");

editor.putInt("key\_age", 25);

editor.putBoolean("key\_isActive", true);

editor.apply(); *// hoặc editor.commit();*

* apply(): Lưu thay đổi bất đồng bộ.
* commit(): Lưu thay đổi đồng bộ (chậm hơn và không nên sử dụng khi không cần thiết).

**3. Đọc Dữ Liệu từ SharedPreferences**

Để lấy dữ liệu từ SharedPreferences, bạn sử dụng các phương thức tương ứng:

String name = sharedPreferences.getString("key\_name", "default\_value");

int age = sharedPreferences.getInt("key\_age", 0);

boolean isActive = sharedPreferences.getBoolean("key\_isActive", false);

* Giá trị mặc định sẽ được sử dụng nếu khóa không tồn tại.

**4. Xóa Dữ Liệu**

Nếu bạn muốn xóa một mục, bạn có thể sử dụng phương thức remove():

editor.remove("key\_name").apply(); *// Xóa mục với khóa là "key\_name"*

Hoặc, để xóa toàn bộ dữ liệu trong SharedPreferences, bạn có thể sử dụng:

editor.clear().apply();

**Kết Luận**

SharedPreferences là một công cụ hiệu quả và mạnh mẽ để lưu trữ dữ liệu đơn giản trong ứng dụng Android. Với cú pháp dễ hiểu và khả năng lưu trữ nhanh chóng, nó rất thích hợp cho việc lưu trữ các thông tin cấu hình và thiết lập của người dùng. Tuy nhiên, nó không thích hợp cho những khối lượng dữ liệu lớn, nơi mà cơ sở dữ liệu sẽ là lựa chọn tối ưu hơn.

# **CHƯƠNG III. THIẾT KẾ ỨNG DỤNG**

## **3.1. Ý tưởng thiết kế**

**Giới thiệu**

**Smart Calendar** là một ứng dụng lịch thông minh được thiết kế để giúp người dùng lên kế hoạch, quản lý thời gian và tối ưu hóa các hoạt động hàng ngày. Ứng dụng sẽ tích hợp công nghệ AI để phân tích thói quen và sở thích của người dùng, từ đó gợi ý các kế hoạch và lịch trình phù hợp nhất.

### 3.1.1. Mục Tiêu của Ứng Dụng

Ứng dụng **Smart Calendar** được thiết kế nhằm giúp người dùng quản lý thời gian và sự kiện một cách hiệu quả hơn, với các tính năng thông minh như:

* Tạo và quản lý sự kiện cá nhân.
* Gợi ý lịch trình dựa trên thói quen người dùng.
* Nhắc nhở thông minh dùng AI.
* Đồng bộ hóa với các nền tảng khác (Google Calendar, Outlook, Zalo...).
* Hỗ trợ quản lý công việc nhóm.

### 3.1.2. Công Nghệ Sử Dụng

* **Ngôn ngữ lập trình**: Dart
* **Framework**: Flutter
* **Cơ sở dữ liệu**: Shared Preferences
* **Thư viện và package hỗ trợ**:
  + **lunar**: Hỗ trợ hiển thị lịch âm, phù hợp với nhu cầu sử dụng của người Việt Nam.
  + **shared\_preferences**: Lưu trữ dữ liệu cục bộ để giữ lại thông tin người dùng.
  + **path\_provider**: Hỗ trợ xác định vị trí lưu trữ dữ liệu trong thiết bị.
  + **flutter\_local\_notifications**: Cung cấp chức năng hiển thị thông báo nhắc nhở sự kiện theo lịch trình.
  + **timezone**: Xử lý múi giờ, đảm bảo thông báo chính xác theo địa phương.
  + **audioplayers**: Hỗ trợ phát âm thanh thông báo cho các sự kiện quan trọng.
  + **intl**: Hỗ trợ định dạng ngày tháng theo nhiều ngôn ngữ khác nhau.
  + **image\_picker**: Cho phép người dùng tải ảnh lên làm hình đại diện sự kiện.
  + **file\_picker**: Hỗ trợ chọn và đính kèm tài liệu vào sự kiện.

### 3.1.3. Tính Năng Chính

1. **Lịch âm – dương**
   * Hiển thị lịch theo cả hai hệ thống âm lịch và dương lịch.
   * Người dùng có thể dễ dàng chuyển đổi giữa hai chế độ lịch để theo dõi các ngày quan trọng như ngày lễ, Tết.
   * Hỗ trợ hiển thị thông tin về các ngày đặc biệt theo phong tục Việt Nam.
2. **Quản lý sự kiện cá nhân**
   * Cho phép người dùng thêm, sửa, xóa các sự kiện theo lịch trình cá nhân.
   * Hỗ trợ phân loại sự kiện theo chủ đề (công việc, học tập, cá nhân, sức khỏe,…).
   * Người dùng có thể đặt mức độ ưu tiên cho từng sự kiện để dễ dàng theo dõi.
   * Hỗ trợ tính năng tìm kiếm và lọc sự kiện theo ngày, tháng, năm.
3. **Hẹn ngày báo**
   * Cài đặt thông báo nhắc nhở sự kiện theo thời gian mong muốn.
   * Hỗ trợ âm thanh và rung để cảnh báo các sự kiện quan trọng.
   * Tích hợp hiển thị thông báo trên màn hình khóa.
   * Nhắc nhở có thể được tùy chỉnh theo múi giờ để phù hợp với lịch trình cá nhân.
4. **Hành trình yêu**
   * Cho phép người dùng thêm ảnh và đếm ngày yêu.
   * Giao diện màu sắc dễ chịu, lãng mạn.
   * Giúp dễ dàng theo dõi số ngày yêu.
5. **Dò tìm cung hoàng đạo**
   * Xác định cung hoàng đạo theo ngày tháng năm sinh.
   * Đưa ra thông tin về cung hoàng đạo đó.
6. **Tính ngày & giờ**
   * Tính ngày và giờ cho người dùng.
   * Hỗ trợ chuyển số giờ ra phút.

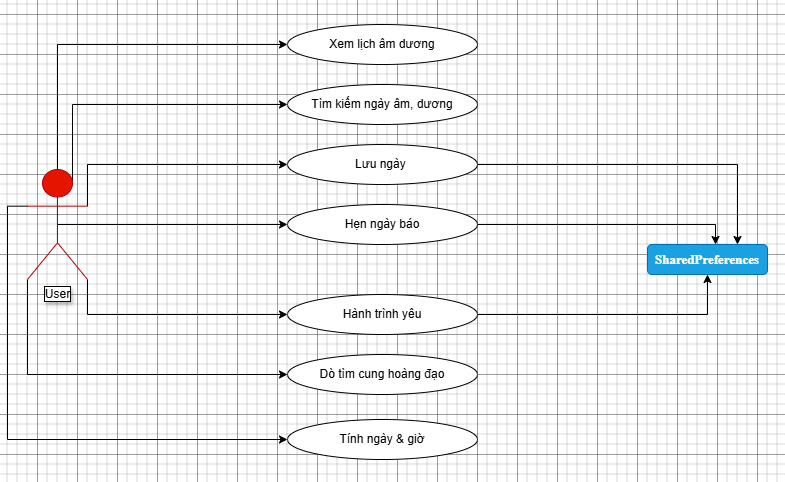
### 3.1.4. Lợi Ích của Ứng Dụng

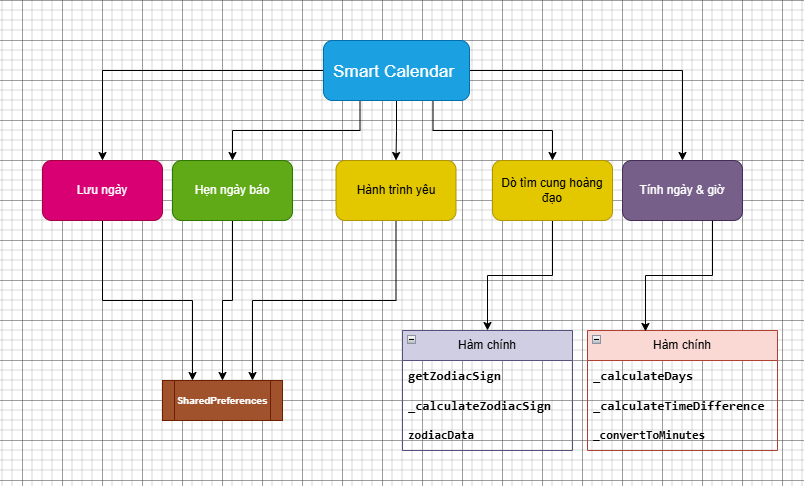
* Giúp người dùng tổ chức công việc một cách khoa học, giảm thiểu tình trạng quên lịch hẹn.
* Hỗ trợ làm việc nhóm thông qua việc chia sẻ sự kiện (có thể phát triển thêm trong tương lai).
* Tích hợp công nghệ thông báo thông minh giúp người dùng không bỏ lỡ sự kiện quan trọng.
* Mang lại trải nghiệm cá nhân hóa thông qua tùy chỉnh giao diện và cài đặt thông báo linh hoạt.

**Kết Luận**

Ứng dụng Smart Calendar - Lịch Thông Minh không chỉ cung cấp các tính năng quản lý thời gian, mà còn mang đến những tiện ích giá trị cho người dùng thông qua khả năng hiển thị lịch âm-dương và các chức năng hỗ trợ đặc biệt. Với thiết kế friendly và công nghệ hiện đại, ứng dụng này sẽ đáp ứng nhu cầu của người dùng tại Việt Nam, giúp họ tổ chức cuộc sống một cách hiệu quả và thuận tiện hơn.

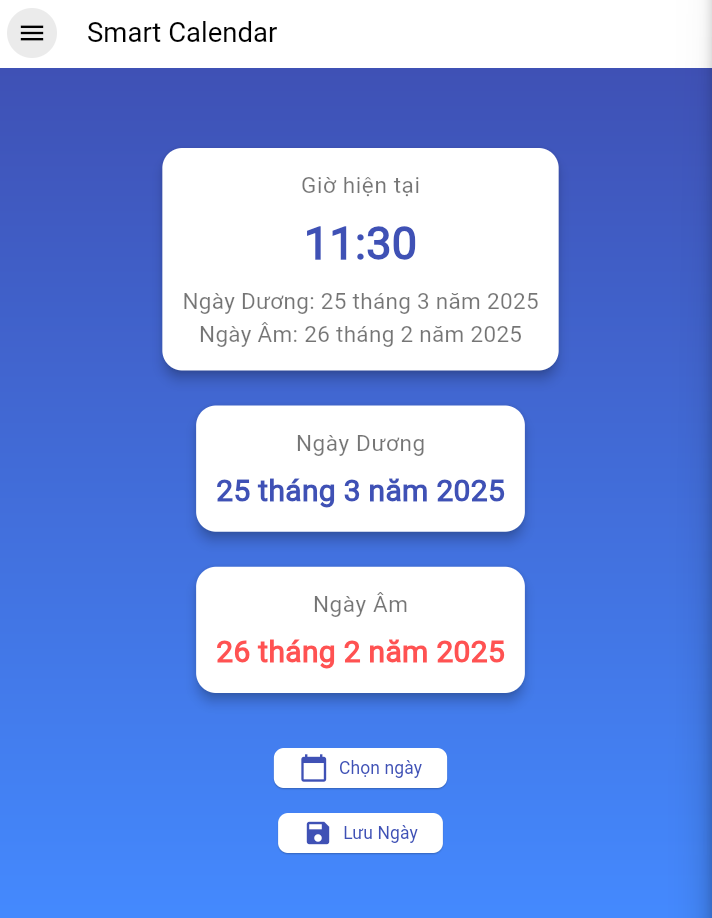
## **3.2. Sơ đồ ứng dụng**

 *Hình 5: Biểu đồ use case*

 *Hình 6: sơ đồ ứng dụng*

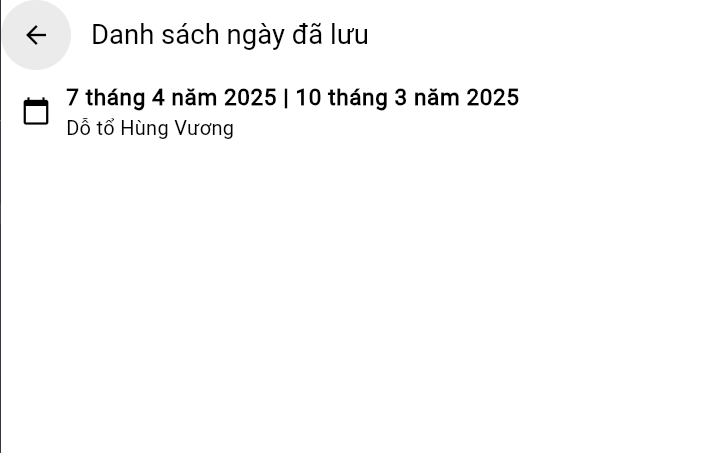
## **3.3. Giao diện các trang chính**

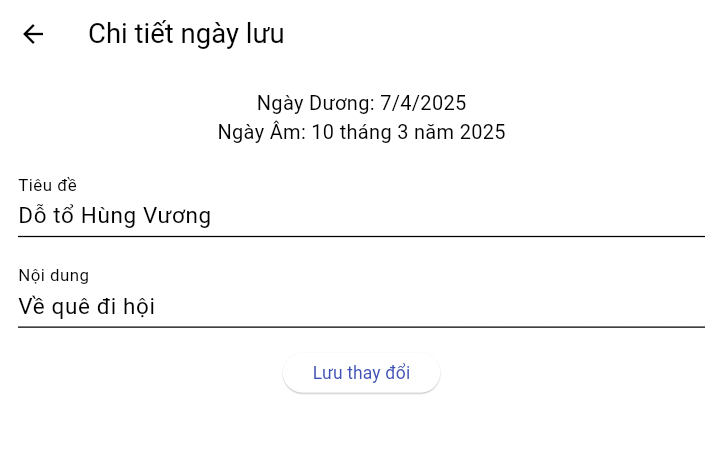
### 3.1.1. Trang chủ

*Hình 7: Trang chủ*

### 3.1.2. Trang lưu ngày

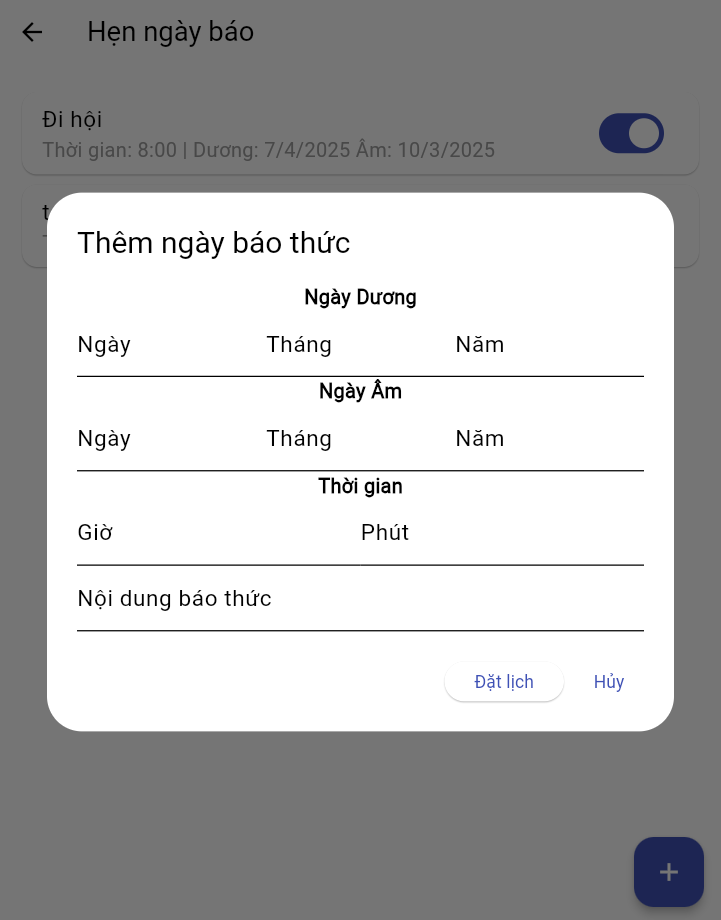
*Hình 8: Thêm ngày vào danh sách*

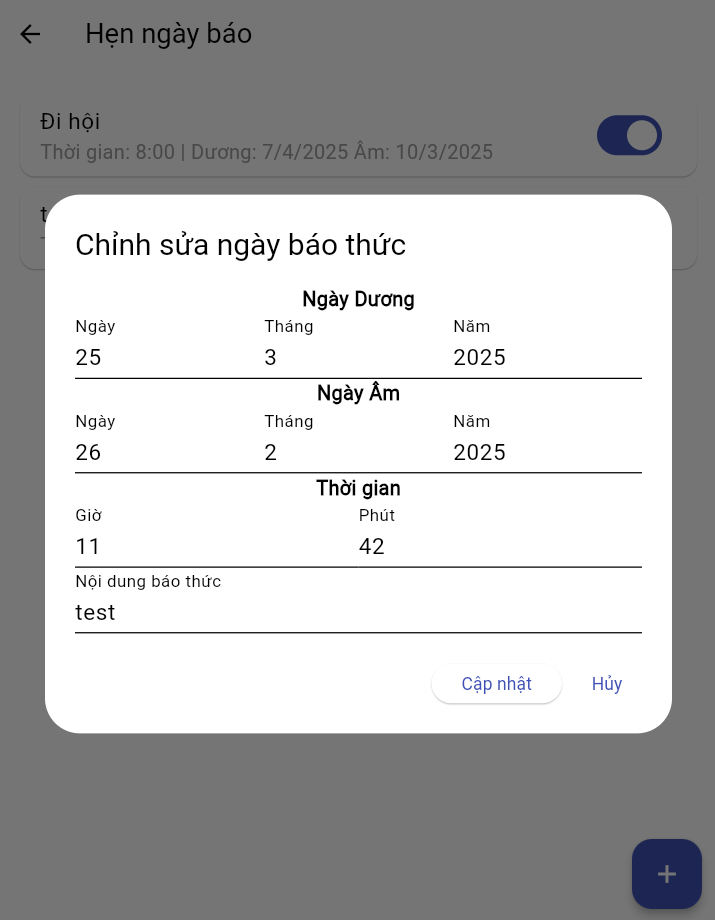
*Hình 9: Danh sách ngày đã lưu*

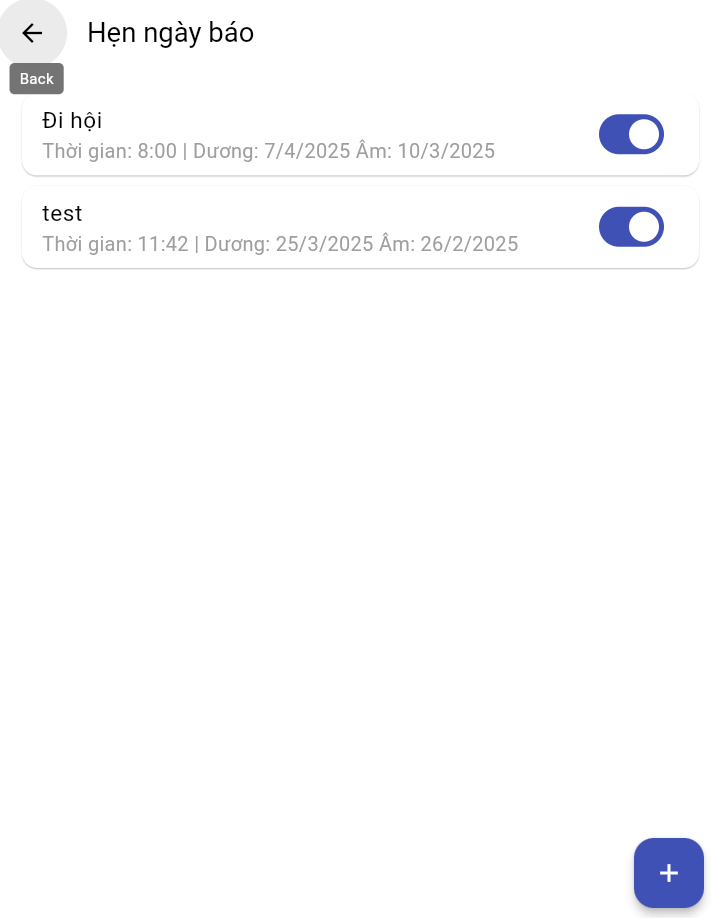
*Hình 10: Sửa ngày đã lưu*

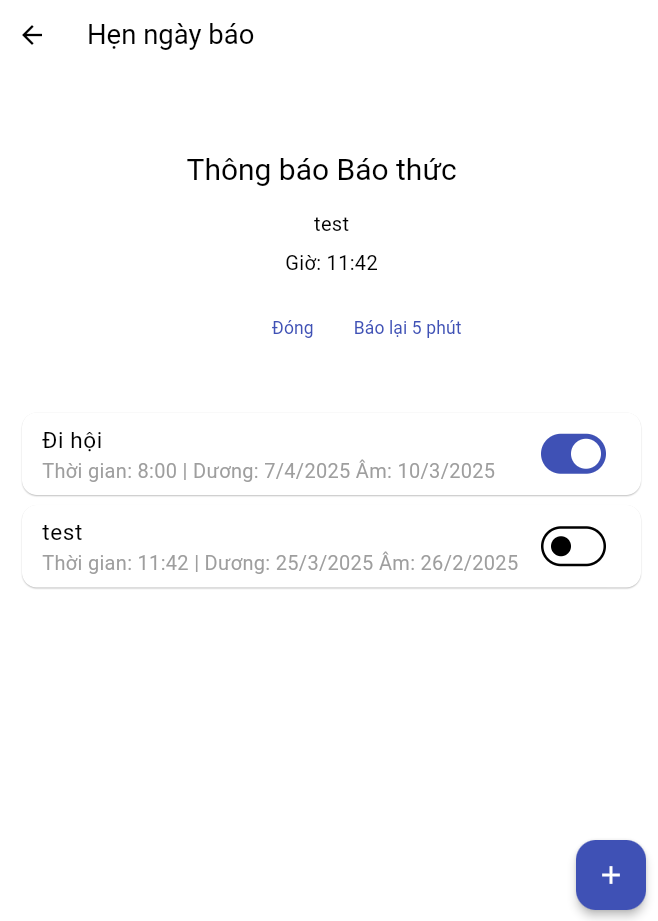
*Hình 11: Sóa ngày đã lưu*

### 3.1.3. Hẹn ngày báo

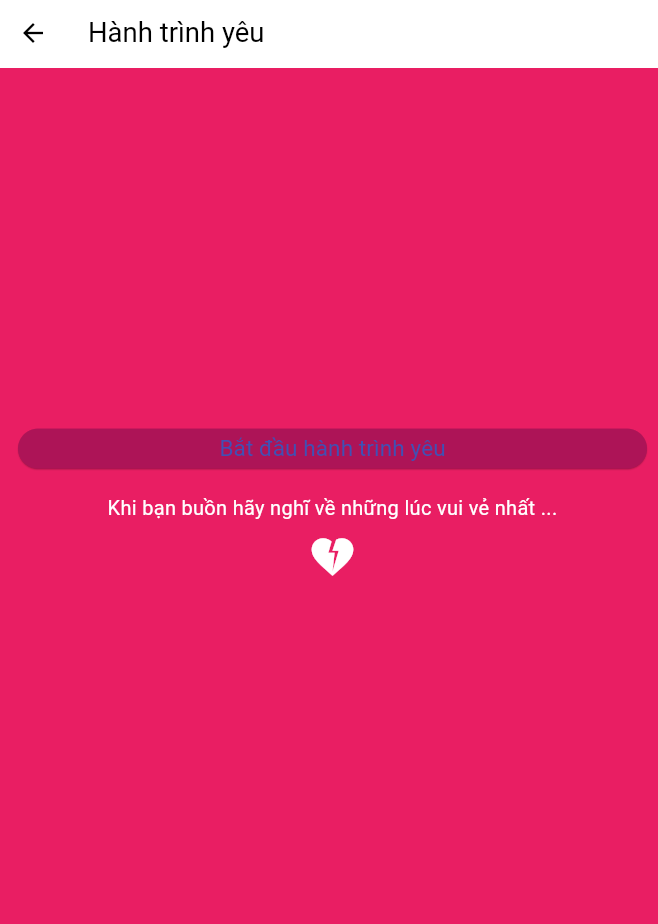
*Hình 12: Thêm ngày báo*

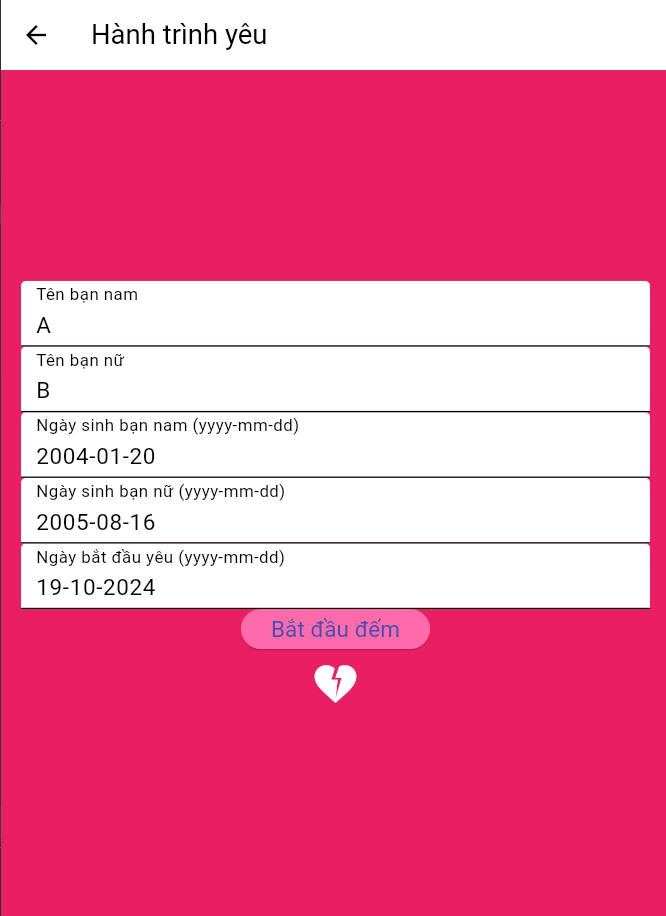
*Hình 13: Sửa ngày báo*

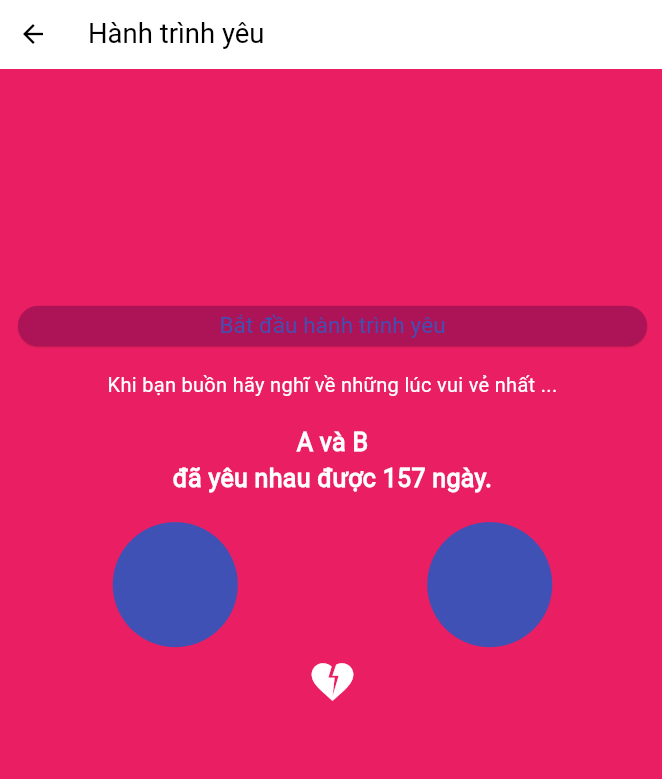
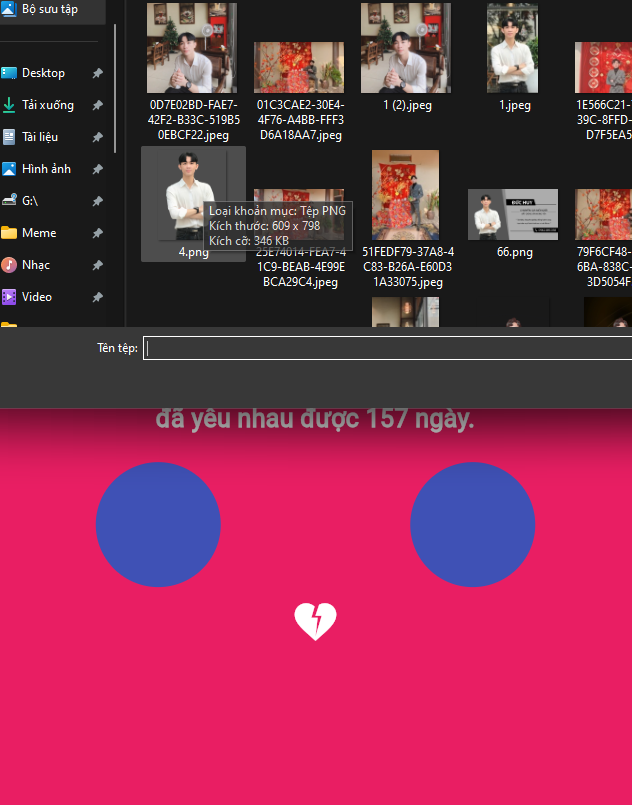
*Hình 14: Danh sách ngày báo*

*Hình 15: Đến ngày báo*

### 3.1.4. Hành trình yêu

 *Hình 16: Giao diện hành trình yêu*

 *Hình 17: Thêm thông tin cặp đôi*

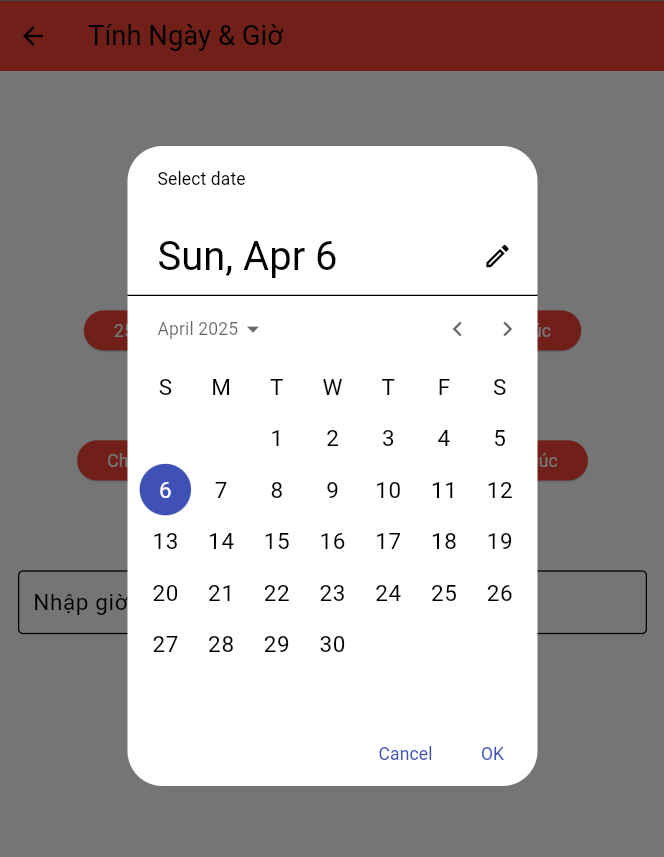
*Hình 18: Kết quả*   *Hình 19: Thêm ảnh cặp đôi*

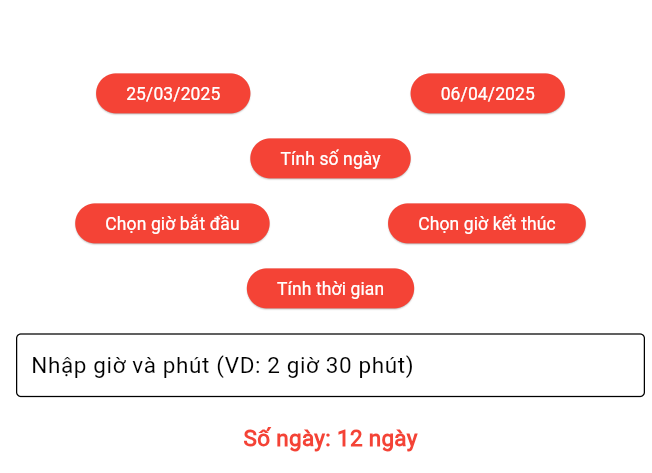
 *Hình 20: Reset (chia tay)*

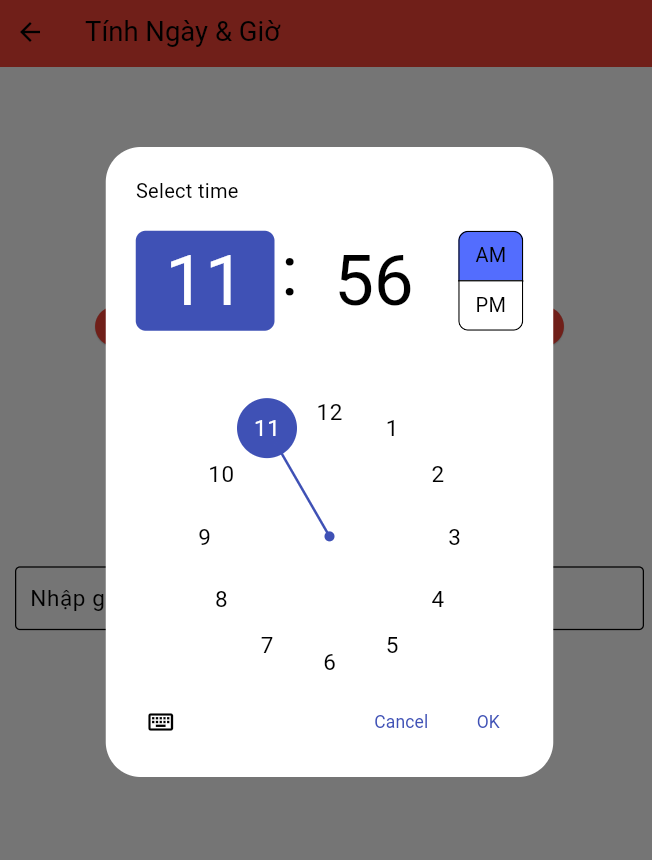
### 3.1.5. Dò tìm cung hoàng đạo

 *Hình 21: Dò tìm cung hoàng đạo*

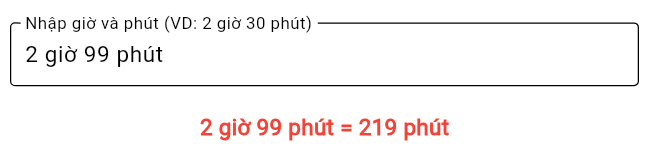
### 3.1.6. Tính ngày & giờ

*Hình 22: Tính số ngày*

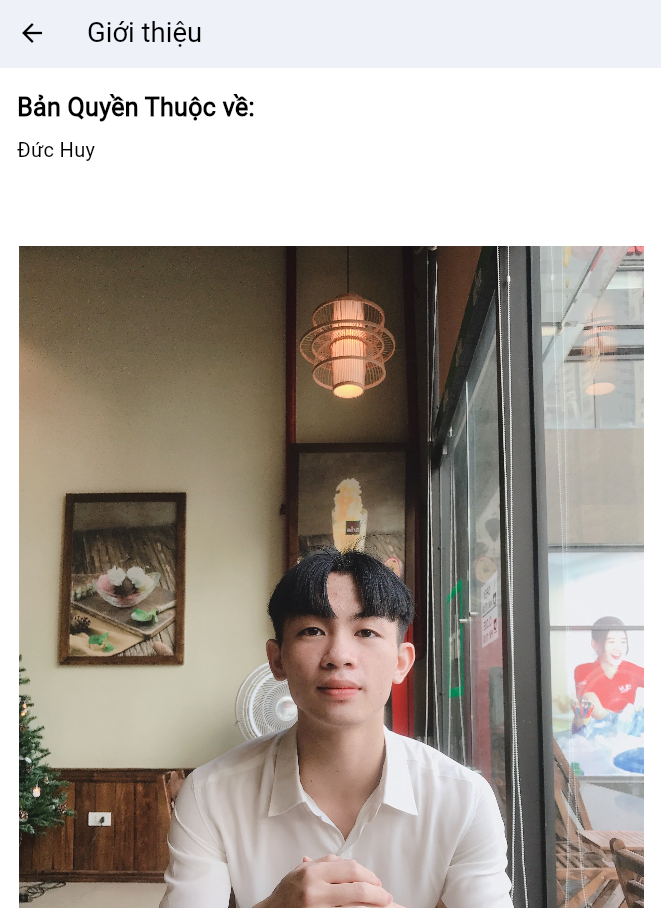
*Hình 23: Kết quả tính số ngày*

*Hình 24: Tính số giờ*

*Hình 25: Kết quả tính số giờ*

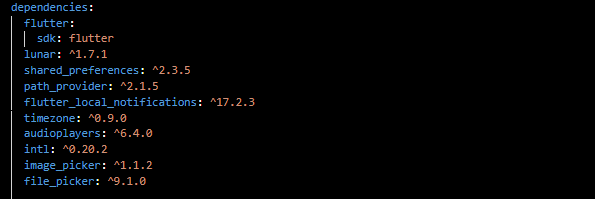
*Hình 26: Kết quả tính giờ ra phút*

### 3.1.7. Giới thiệu

 *Hình 27: Giới thiệu*

## **3.4. Code của một số trang chính**

### 3.4.1. Thư viện và package hỗ trợ



### 3.4.2. Code trang chủ

import 'package:flutter/material.dart';

import 'package:lunar/lunar.dart';

import 'package:shared\_preferences/shared\_preferences.dart';

import 'dart:convert';

import '../models/saved\_day.dart';

import 'saved\_days\_screen.dart';

import 'reminders\_screen.dart';

import 'love\_journey\_screen.dart';

import 'Zodiac.dart';

import 'about\_screen.dart';

import 'Calculatedatetime.dart';

class HomeScreen extends StatefulWidget {

const HomeScreen({Key? key}) : super(key: key);

@override

State<HomeScreen> createState() => \_HomeScreenState();

}

class \_HomeScreenState extends State<HomeScreen> with SingleTickerProviderStateMixin {

// Định nghĩa biến cần thiết

// ... (các biến và hàm khởi tạo như được trình bày trong đoạn code của bạn)

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(title: const Text('Lịch Âm Dương')),

// ... (các thành phần UI khác trong build)

);

}

void saveDay() {

showInputDialog();

}

// Hàm để lưu ngà

// ...

}

### 3.4.3. Code ngày đã lưu

**1. Chi tiết ngày đã lưu (SavedDaysScreen)**

import 'package:flutter/material.dart';

import '../models/saved\_day.dart';

import 'day\_details\_screen.dart';

class SavedDaysScreen extends StatefulWidget {

final List<SavedDay> savedDays;

final Function(List<SavedDay>) onUpdate;

const SavedDaysScreen({Key? key, required this.savedDays, required this.onUpdate}) : super(key: key);

@override

\_SavedDaysScreenState createState() => \_SavedDaysScreenState();

}

class \_SavedDaysScreenState extends State<SavedDaysScreen> {

// Định nghĩa các biến cần thiết và hàm build

// ...

void \_deleteSelectedDays() {

// Hàm để xóa các ngày đã chọn

}

}

**2. Chi tiết ngày đã lưu (DayDetailsScreen)**

import 'package:flutter/material.dart';

import '../models/saved\_day.dart';

class DayDetailsScreen extends StatefulWidget {

final SavedDay savedDay;

final Function(SavedDay) onUpdate;

const DayDetailsScreen({Key? key, required this.savedDay, required this.onUpdate}) : super(key: key);

@override

\_DayDetailsScreenState createState() => \_DayDetailsScreenState();

}

class \_DayDetailsScreenState extends State<DayDetailsScreen> {

// Định nghĩa các biến cần thiết

// ...

void updateSavedDay() {

// Cập nhật thông tin ngày lưu

}

}

**3. Lớp SavedDay cho mô hình ngày đã lưu**

class SavedDay {

DateTime solarDate; // Ngày Dương

String lunarDate; // Ngày Âm

String title; // Tiêu đề của ngày

String content; // Nội dung chi tiết

SavedDay({required this.solarDate, required this.lunarDate, required this.title, required this.content});

Map<String, dynamic> toJson() {

return {

'solarDate': solarDate.toIso8601String(),

'lunarDate': lunarDate,

'title': title,

'content': content,

};

}

static SavedDay fromJson(Map<String, dynamic> json) {

return SavedDay(

solarDate: DateTime.parse(json['solarDate']),

lunarDate: json['lunarDate'],

title: json['title'],

content: json['content'],

);

}

}

### 3.4.4. Code hẹn ngày báo

import 'package:flutter/material.dart';

import 'dart:async';

import 'package:flutter\_local\_notifications/flutter\_local\_notifications.dart';

import 'package:lunar/lunar.dart';

import 'package:shared\_preferences/shared\_preferences.dart';

import 'package:audioplayers/audioplayers.dart';

class RemindersScreen extends StatefulWidget {

const RemindersScreen({Key? key}) : super(key: key);

@override

\_RemindersScreenState createState() => \_RemindersScreenState();

}

class \_RemindersScreenState extends State<RemindersScreen> {

// Định nghĩa các biến cần thiết

// ...

void \_setReminder() {

// Hàm để thiết lập ngày báo

}

}

### 3.4.5. Code hành trình yêu

import 'package:flutter/material.dart';

import 'package:file\_picker/file\_picker.dart';

import 'package:shared\_preferences/shared\_preferences.dart';

class LoveJourneyScreen extends StatefulWidget {

const LoveJourneyScreen({Key? key}) : super(key: key);

@override

State<LoveJourneyScreen> createState() => \_LoveJourneyScreenState();

}

class \_LoveJourneyScreenState extends State<LoveJourneyScreen> {

// Định nghĩa các biến cần thiết

// ...

void pickImage(bool isBoy) async {

// Hàm để chọn hình ảnh

}

}

### 3.4.6. Code tính Ngày & Giờ

import 'package:flutter/material.dart';

import 'package:intl/intl.dart';

class Calculatedatetime extends StatefulWidget {

const Calculatedatetime({Key? key}) : super(key: key);

@override

\_Calculatedatetime createState() => \_Calculatedatetime();

}

class \_Calculatedatetime extends State<Calculatedatetime> {

// Định nghĩa các biến cần thiết

// ...

void \_calculateDays() {

// Hàm để tính số ngày giữa hai ngày

}

}

### 3.4.7. Code dò tìm cung hoàng đạo

import 'package:flutter/material.dart';

class Zodiac extends StatefulWidget {

const Zodiac({Key? key}) : super(key: key);

@override

\_ZodiacState createState() => \_ZodiacState();

}

class \_ZodiacState extends State<Zodiac> {

// Định nghĩa các biến cần thiết

// ...

void \_calculateZodiacSign() {

// Hàm để tính cung hoàng đạo

}

}

**KẾT LUẬN**

Ứng dụng **Smart Calendar – Lịch Thông Minh** đã đạt được nhiều kết quả đáng kể trong việc hỗ trợ người dùng quản lý thời gian một cách hiệu quả. Với các tính năng như hiển thị lịch âm – dương, gợi ý lịch trình, thông báo thông minh và lưu trữ dữ liệu đơn giản, ứng dụng mang lại trải nghiệm tiện lợi, đặc biệt phù hợp với nhu cầu của người Việt Nam. Ngoài ra, các tính năng độc đáo như **Hành trình yêu**, **Dò tìm cung hoàng đạo** và **Tính toán ngày & giờ** cũng góp phần tạo thêm sự thú vị và cá nhân hóa cho người dùng.

Tuy nhiên, ứng dụng vẫn tồn tại một số hạn chế như giới hạn nền tảng, khả năng mở rộng lưu trữ dữ liệu còn hạn chế và thuật toán gợi ý lịch trình cần được cải thiện hơn nữa. Bên cạnh đó, giao diện có thể được thiết kế trực quan và hấp dẫn hơn để nâng cao trải nghiệm người dùng.

Trong tương lai, việc **phát triển giao diện trực quan hơn, mở rộng tính năng cho nhóm làm việc, tối ưu hóa quy trình lưu trữ dữ liệu và tích hợp thêm trí tuệ nhân tạo** sẽ giúp ứng dụng hoàn thiện hơn và đáp ứng tốt hơn nhu cầu đa dạng của người dùng. Ngoài ra, việc bổ sung các tính năng hỗ trợ sức khỏe và mở rộng đa ngôn ngữ cũng sẽ là hướng đi tiềm năng, giúp ứng dụng tiếp cận được nhiều đối tượng hơn.

Tóm lại, **Smart Calendar** là một sản phẩm tiềm năng với nhiều tính năng hữu ích, và với những cải tiến trong tương lai, nó có thể trở thành một công cụ không thể thiếu trong việc quản lý thời gian, nâng cao hiệu suất làm việc và cải thiện chất lượng cuộc sống của người dùng.

# **LỜI CẢM ƠN**

Trong quá trình học tập và nghiên cứu môn Lập trình Mobile tại trường Đại học Đại Nam, em đã có cơ hội tích lũy nhiều kiến thức quý báu, đặc biệt là trong việc xây dựng ứng dụng di động. Đề tài "Xây Dựng App Smart Calendar - Lịch Thông Minh" mà em thực hiện là kết quả của thời gian học tập, tìm hiểu và thực hành dưới sự hướng dẫn của thầy Phạm Văn Tiệp.

Em nhận thức rằng, kiến thức về lập trình và công nghệ luôn không ngừng mở rộng, và bản thân em vẫn còn nhiều điều cần học hỏi. Trong quá trình thực hiện báo cáo này, em không thể tránh khỏi những thiếu sót. Vì vậy, em rất mong nhận được sự góp ý từ thầy để hoàn thiện hơn công trình của mình.

Em xin gửi lời cảm ơn chân thành đến thầy Phạm Văn Tiệp – người đã tận tâm giảng dạy, hỗ trợ và giải đáp mọi thắc mắc của em trong suốt học kỳ. Sự nhiệt huyết và kiên nhẫn của thầy không chỉ giúp em nâng cao hiểu biết về lập trình mobile, mà còn truyền cảm hứng để em tiếp tục theo đuổi con đường phát triển trong lĩnh vực công nghệ.

Cuối cùng, em xin chúc thầy sức khỏe, hạnh phúc và gặt hái nhiều thành công trong sự nghiệp giảng dạy và nghiên cứu. Em hy vọng những kiến thức mà em đã tiếp thu từ môn học này sẽ là nền tảng vững chắc cho sự phát triển trong nghề nghiệp tương lai của em.

**Sinh viên thực hiện:**

CHU VĂN HUY

**DANH MỤC TÀI LIỆU THAM KHẢO**

[1]. Đặng Văn Hải (2021), Giáo trình Lập trình Mobile, NXB Đại học Quốc gia Hà Nội.

[2]. Nguyễn Minh Tuấn (2019), Đề cương bài giảng Lập trình Mobile, Đại học Khoa học Tự nhiên.

[3]. Ray Wenderlich (2020), iOS Programming: The Big Nerd Ranch Guide, 6th Edition, Big Nerd Ranch.

[4]. David Flanagan & Yukihiro Matsumoto (2018), Programming Ruby: The Pragmatic Programmers' Guide, 4th Edition, Pragmatic Bookshelf.

[5]. Mark J. Price (2021), Android Programming for Beginners, Packt Publishing.

[6]. Dreyfus, S. E., & Littlestone, N. (2020), Professional Android Application Development, John Wiley & Sons.

[7]. Google Developers (2023), Android Developers Documentation, Retrieved from <https://developer.android.com/docs>.

[8]. Apple Developer (2023), Swift Language Documentation, Retrieved from <https://developer.apple.com/swift/>.

[9]. Allen B. Downey (2013), Think Stats: Exploratory Data Analysis, O'Reilly Media.

[10]. S. M. S. Rahman (2018), Mobile App Development with Flutter: A Practical Guide, Springer.

[11]. Lee, W., & Lee, N. (2022), Developing Smart Applications with Artificial Intelligence, IGI Global.

[12]. Pahl, C., & J. G. H. (2021), Smart Mobile Applications: Modern Mobile App Development, Springer.

[13]. González, J. C. (2020), Mobile and Web Programming: A Practical Approach, Springer.

[14]. Borsato, M., & Lenzi, A. (2021), Design and Develop Applications with Vue.js and Firebase, Packt Publishing.

[15]. Collins, J. (2019), React Native for Mobile Development: A Comprehensive Guide, CreateSpace Independent Publishing Platform.

[16]. Kottke, J. (2020), Building Smart Mobile Apps using AI: Concepts and Applications, Wiley.

[17]. Nascimento, T. S., & Ribeiro, F. (2023), Mobile Computing and Platforms: Theory and Applications, IGI Global.