**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA ĐIỆN TỬ VIỄN THÔNG**

**BÁO CÁO**

**LẬP TRÌNH ĐA NỀN TẢNG**

**SharedPreferences trong Flutter – Lưu trữ dữ liệu đơn giản**

**Sinh viên thực hiện:**

**01. Trịnh Minh Việt Lớp: 22KTMT1**   **MSSV:** 106220241

**02. Trần Lê Long Vũ**  **Lớp: 22KTMT1**  **MSSV:** 106220242

**Người hướng dẫn:**

**TS. Nguyễn Duy Nhật Viễn**

**Đà Nẵng, 2025.**

**THUYẾT MINH**

**BÁO CÁO**

**LẬP TRÌNH ĐA NỀN TẢNG**

**SharedPreferences trong Flutter – Lưu trữ dữ liệu đơn giản**

BẢNG PHÂN CÔNG CÔNG VIỆC TRONG NHÓM

|  |  |  |  |
| --- | --- | --- | --- |
| STT | HỌ VÀ TÊN | NHIỆM VỤ | KHỐI LƯỢNG |
| 01 | Trịnh Minh Việt | Tìm hiểu về so sánh các phương thức lưu trữ khác,xử lý lỗi, code demo và chỉnh sửa | 50% |
| 02 | Trần Lê Long Vũ | Tìm hiểu nguyên lí hoạt động, best practice, code demo và chỉnh sửa | 50% |

 Link code github: <https://github.com/TrMnhViet/Laptrinhdanentang_Viet/tree/main>

Mục Lục

[**1.SharedPreferences:** 4](#_Toc212413325)

[**1.1. Định nghĩa** 4](#_Toc212413326)

[**1.2. Nguyên lí hoạt động:** 5](#_Toc212413327)

[**2. Code Demo:** 6](#_Toc212413328)

[**3**. **So sánh với các phương thức lưu trữ khác** 10](#_Toc212413329)

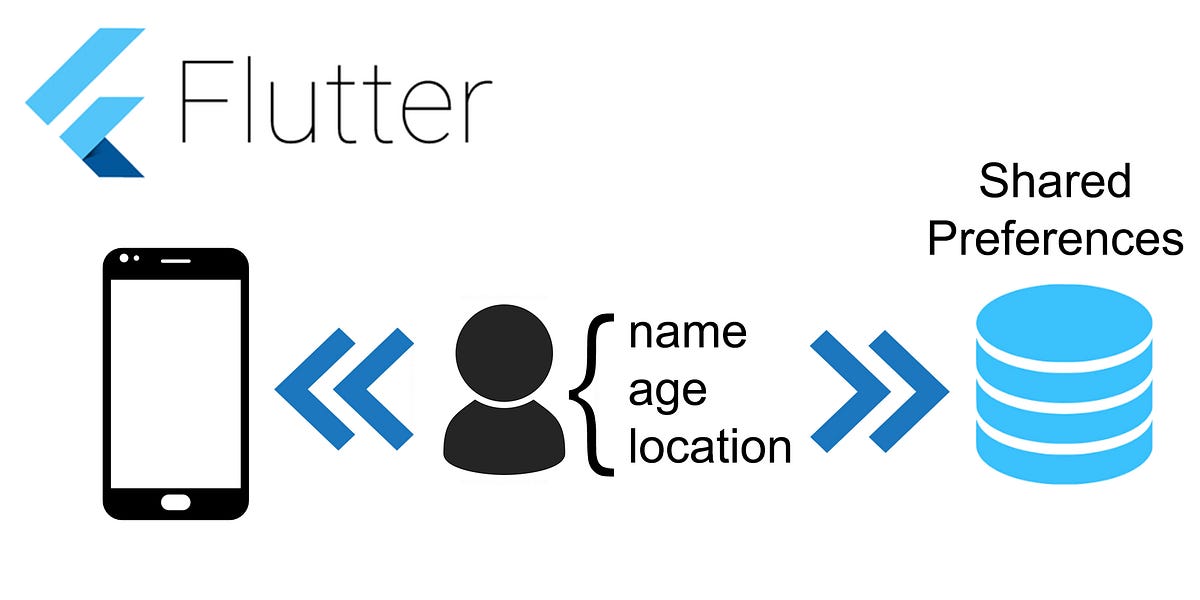
[**4.Xử lý lỗi và Best Practice** 10](#_Toc212413330)

**4.1. Xử lí lỗi**................................................................................................................10

[**4.2. Best Practices:** 12](#_Toc212413332)

**NỘI DUNG**

# **1.SharedPreferences:**



Hình 1. Tổng quan về SharedPreferences

## **1.1. Định nghĩa**

* Là một package Flutter cho phép chúng ta lưu trữ các cặp Key-Value đơn giản trên thiết bị.Thường dùng để lưu trữ
* **Cài đặt người dùng:** Chế độ sáng/tối, ngôn ngữ, tuỳ chọn thông báo
* **Trạng thái ứng dụng:** Đã đăng nhập, hoặc lần đầu mở ứng dụng
* **Dữ liệu nhỏ, không phức tạp:** Điểm số cao nhất gần đây, tên người dùng gần đây
* Các API lưu trữ nguyên bản gồm có NSUserDefaults (IOS) và SharedPreferences (Android).
* Khi nào nên dùng và khi nào không nên dùng:
* Nên dùng:
* Cần lưu trữ dữ liệu đơn giản và kích thước nhỏ
* Lưu trữ dữ liệu dạng cặp Key-Value
* Không có tính bảo mật cao
* Lưu trữ cấu hình hoặc giao diện người dùng
* Không nên dùng:
* Lưu dữ liệu kích thước lớn như ảnh, video
* Dữ liệu có độ bảo mật cao
* Dữ liệu truy vấn phức tạp

## **1.2. Nguyên lí hoạt động:**

A diagram of a diagram

Description automatically generated

Hình 2. Nguyên lí hoạt động

* Android Application: đại diện cho ứng dụng của bạn (ví dụ app Flutter hoặc Android).
* SharedPreferences: vùng lưu trữ dữ liệu key–value trong bộ nhớ cục bộ.
* Quy trình trong hình:

1. Ứng dụng lưu dữ liệu bằng cách gửi cặp Key–Value (ví dụ: Key = “UserName”, Value = “Carl”) vào SharedPreferences.
2. Dữ liệu được ghi vào bộ nhớ (file XML) để lưu lại.
3. Khi cần đọc lại, ứng dụng gửi yêu cầu get(key) → SharedPreferences trả về giá trị tương ứng (“Carl”).

# **2. Code Demo:**



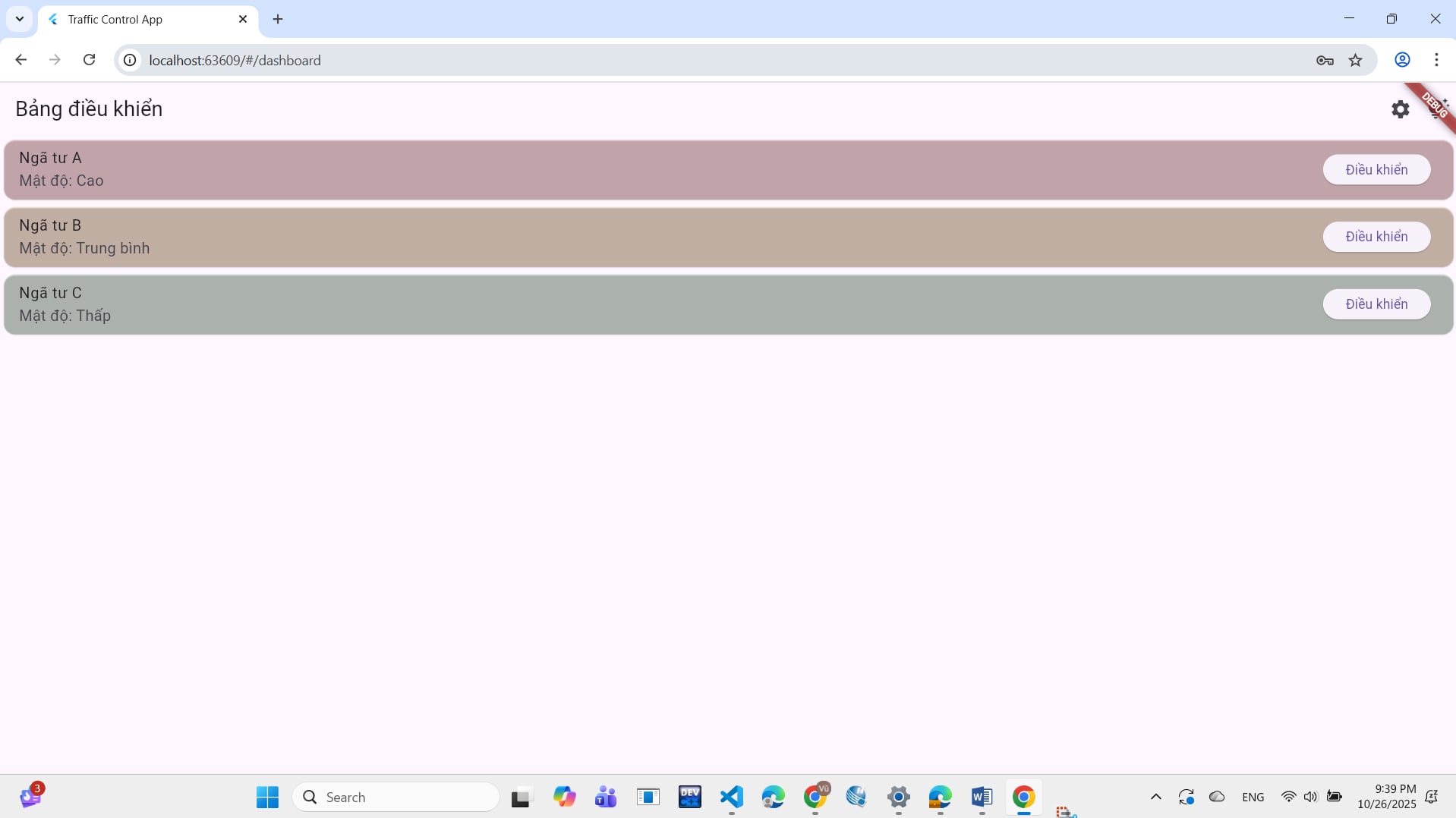
Hình 3. Code Demo

A screenshot of a computer program

Description automatically generated

Hình 4. Code Demo

Hình 5 bên dưới là màn hình giao diện của trang chủ khi Run Code



Hình 5 . Kết quả khi Run Code

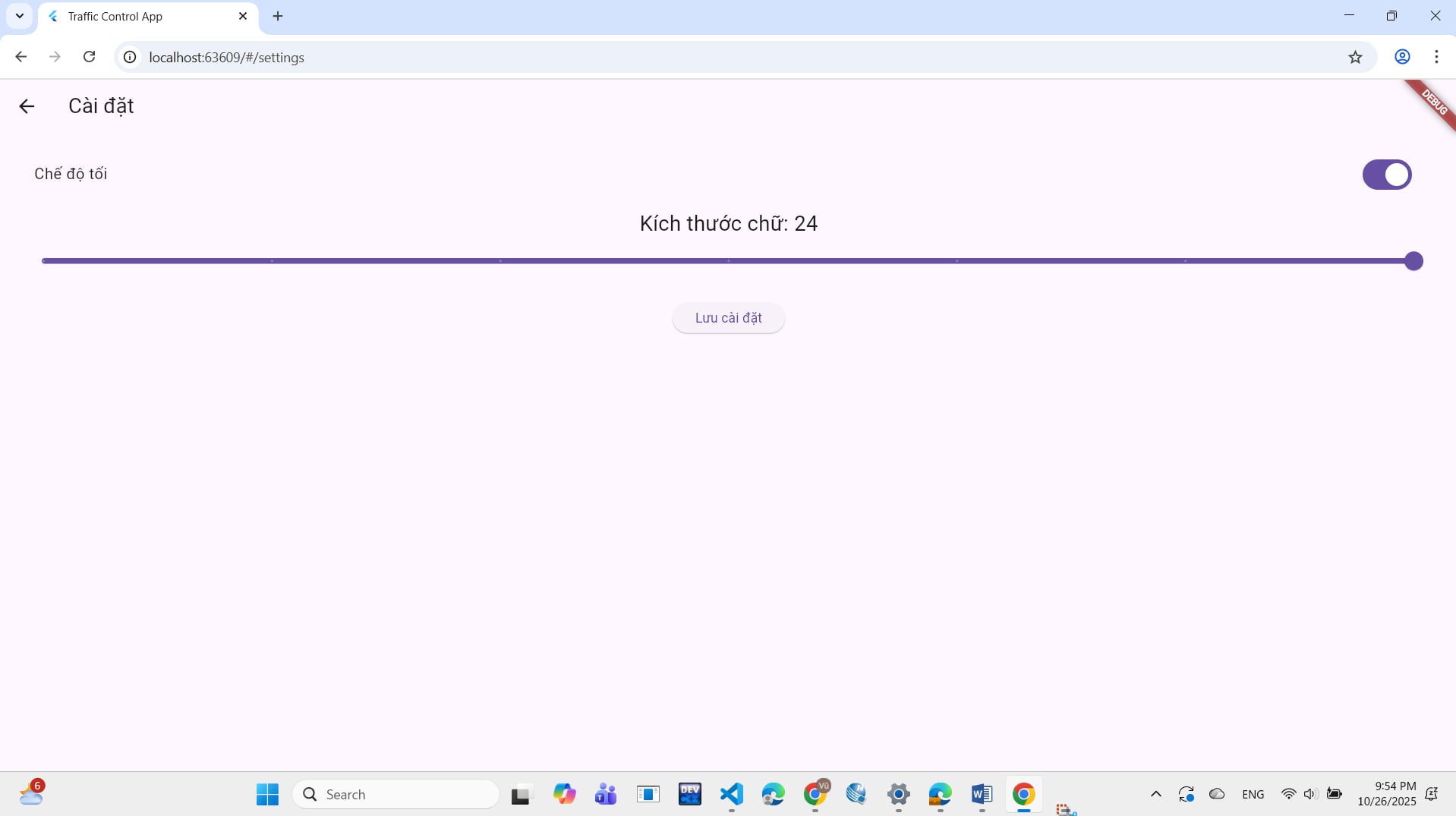
Tiếp theo, nháy chuột vào biểu tượng răng cưa phía trên bên phải ta sẽ được giao diện cài đặt như hình 6 bên dưới.

A screenshot of a computer

Description automatically generated

Hình 6 . Giao diện cài đặt

Tiếp đến, Click chuột vào ô chế độ tối phía trên bên phải để chuyển sang chế độ tối và kéo thanh kích cỡ để chọn kích thước chữ mong muốn, sau đó nhấn lưu cài đặt sẽ thu được kết quả như hình 7.



Hình 7 . Sau khi kích hoạt chế độ tối và điều chỉnh cỡ chữ

Cuối cùng, thoát ra màn hình chính sẽ thu được kết quả như hình 8

A screenshot of a computer

Description automatically generated

Hình 8 . Giao diện trang chủ sao khi thay đổi

# 3. **So sánh với các phương thức lưu trữ khác**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Phương thức | Mục đích | Loại lưu trữ | Ưu điểm | Nhược điểm |
| SharedPreference | Lưu trữ dữ liệu nhỏ, không nhạy cảm | Key-Value thông thường | Đơn giản, nhanh | Không có tính bảo mật và không thể lưu dữ liệu truy vấn phức tạp |
| Hive | Dữ liệu có cấu trúc, đối tượng Dart | Key-value dạng đối tượng | Đơn giản, dễ dùng và có mã hoá | Không hỗ trợ truy vấn các quan hệ phức tạp |
| SQLite | Dữ liệu quan hệ, cấu trúc truy vấn phức tạp | Cấu trúc bảng (SQL) | Mạnh mẽ, toàn vẹn, truy vấn SQL | Cần nhiều kiến thức về SQL |
| Secure Storage | Dữ liệu nhạy cảm, bảo mật cao | Key-Value dạng String | Bảo mật nhất, mã hoá tự động | Không thể lưu dữ liệu quá lớn |

# **4.Xử lý lỗi và Best Practice**

## **4.1. Xử lý lỗi:**

Có 3 loại lỗi phổ biến hay gặp trong Flutter: Lỗi Flutter có thể bắt được, Lỗi Flutter không bắt được và Lỗi khi xảy ra trong quá trình build Widget.

* **Lỗi Flutter bắt được:**
* Đây là những lỗi xảy ra trong các callback được kích hoạt bởi chính framework Flutter, bao gồm các lỗi gặp phải trong các pha *build*, *layout* và *paint***.**
* Tất cả các lỗi được Flutter bắt sẽ được định tuyến đến trình xử lý FlutterError.onError

A screenshot of a computer code

Description automatically generated

Hình 9. Lỗi Flutter bắt được

* **Lỗi Flutter không bắt được :**
* Đây là những lỗi không xảy ra trong các callback của Flutter trên stack gọi
* Những lỗi này được chuyển tiếp đến **PlatformDispatcher**

**A screen shot of a computer program

Description automatically generated**

Hình 10 . Lỗi Flutter không bắt được

* **Lỗi trong quá trình Build Widget:**
* Đây là trường hợp cụ thể khi lỗi xảy ra trong quá trình xây dựng (build phase) của một widget
* Khi lỗi build xảy ra, callback ErrorWidget.builder được gọi để xây dựng một widget thay thế cho widget bị lỗi

A screenshot of a computer program

Description automatically generated

Hình 11 . Lỗi trong quá trình Build Widget

## **4.2. Best Practices:**

1. **Kiểm soát Chi phí Xây dựng Widget (Control Widget Build Cost):**

* **Sử dụng const Constructors:** Áp dụng hàm tạo const cho widget càng nhiều càng tốt, vì điều này cho phép Flutter rút ngắn hầu hết công việc xây dựng lại (rebuild work)
* **Cục bộ hóa setState():** Giới hạn lệnh gọi setState() vào phần nhỏ nhất của cây con (subtree) mà UI thực sự cần thay đổi, tránh gọi setState() quá cao trong cây widget
* **Tách Widget:** Tránh các widget đơn lẻ quá lớn với hàm build() lớn. Hãy chia chúng thành các widget nhỏ hơn dựa trên tính đóng gói và cách chúng thay đổi

1. **Tối ưu hóa Thao tác Tốn Kém (Optimize Expensive Operations):**

* **Giảm thiểu saveLayer():** Sử dụng saveLayer() một cách thận trọng vì đây là một thao tác tốn kém (allocate offscreen buffer, gián đoạn thông lượng rendering trên GPU di động)
* **Tránh Opacity trong Animation:** Hạn chế sử dụng widget Opacity, đặc biệt là trong animation. Thay vào đó, hãy dùng AnimatedOpacity hoặc FadeInImage
* **Sử dụng StringBuffer:** Khi xây dựng chuỗi từ nhiều phần (đặc biệt trong vòng lặp), dùng StringBuffer hiệu quả hơn so với toán tử
* **Thận trọng với Clipping:** Mặc dù không tốn kém bằng Opacity, cắt xén (Clipping) vẫn tốn kém

1. **Tối ưu hóa Bố cục và Danh sách (Layout and Lists):**

* **Áp dụng "Lazy Builder Methods":** Khi xây dựng danh sách hoặc lưới lớn, hãy sử dụng các phương thức **builder lười biếng** (ví dụ: ListView.builder) để đảm bảo chỉ phần hiển thị trên màn hình được xây dựng lúc khởi động
* **Tránh Intrinsic Passes:** Giảm thiểu các lượt tính toán nội tại (intrinsic passes) tốn kém trong bố cục bằng cách đặt kích thước cố định cho các ô hoặc chọn một ô "neo" (anchor cell)

1. **Mục tiêu Hiệu suất Khung hình:**

* **Mục tiêu 16ms:** Đối với màn hình 60Hz, mục tiêu là xây dựng (build) và hiển thị (render) một khung hình trong **16ms hoặc ít hơn**. Duy trì thời gian render thấp giúp **cải thiện tuổi thọ pin** và các vấn đề nhiệt

**TÀI LIỆU THAM KHẢO**

[1]<https://www.topcoder.com/thrive/articles/shared-preferences-in-android>

[2] <https://docs.flutter.dev/testing/errors>

[3]<https://docs.flutter.dev/perf/best-practices>

***Chú ý:***

Phải tuân thủ format của tài liệu tham khảo.

Nếu có tham khảo các thuyết minh, phải trích dẫn và thể hiện cụ thể trong các tài liệu tham khảo. Code của khóa trước phải upload đầy đủ code cũ lẫn code mới, trong thuyết minh phải chỉ ra được cái mới của code nhóm thực hiện. Nếu phát hiện sao chép mà không trích dẫn, không có trong mục tài liệu tham khảo thì toàn nhóm sẽ bị điểm 0.