**BÁO CÁO TUẦN 5**

**Họ và tên:** Nguyễn Kỳ Anh  
**MSSV:** 20225793  
 **GR1**  
**Giáo viên hướng dẫn:** Nguyễn Đức Toàn

# AppBar

* Vị trí: AppBar thường nằm ở đầu của một Scaffold widget, đóng vai trò là thanh điều hướng chính của màn hình.
* Chức năng:
* Hiển thị tiêu đề (title) của ứng dụng hoặc màn hình.
* Cung cấp các nút hành động (action buttons) như tìm kiếm, thêm, menu ,…
* Hỗ trợ nút dẫn ngược ( leading icon/button), thường là nút quay lại hoặc menu hamburger.
* Tùy chỉnh giao diện như màu sắc, chiều cao hoặc đổ bóng
* Các thuộc tính cơ bản của AppBar:
* Title: Widget hiển thị tiêu đề chính thường là dạng Text()
* Leading: Widget hiển thị ở bên trái, thường là các nút (menu, back). Nếu không được chỉ định Flutter sẽ tự động thêm nút quay lại nếu có thể.
* Actions: Danh sách các Widget (thường là các IconButton) ở bên phải AppBar.
* backgroundColor: Màu nền của AppBar.
* Elevation: độ cao của đổ bóng(shadow).
* automaticallylmplyLeading: nếu true (mặc định), tự động thêm nút quay lại khi cần.
* flexibleSpace: Cho phép thêm nội dung tùy chỉnh phía sau AppBar.
* Bottom: Widget hiển thị dưới AppBar, thường dung cho TabBar hoặc thanh tìm kiếm.

VD:

AppBar(

title: Text('Tiêu đề'),

leading: IconButton(

icon: Icon(Icons.menu),

onPressed: () {},

),

actions: [

IconButton(

icon: Icon(Icons.search),

onPressed: () {},

),

IconButton(

icon: Icon(Icons.more\_vert),

onPressed: () {},

),

],

backgroundColor: Colors.blue,

elevation: 4.0,

)

# InkWell

* Chức năng:
* Cung cấp hiệu ứng gợn sóng( ripple effect) khi người dung chạm, giữ hoặc kéo.
* Cho phép xử lý các sự kiện cảm ứng như nhấn (tap), nhân đúp (double tap), nhấn lâu (long press),…
* Thường được sử dụng để làm cho một Widget trở nên “Nhấn được” với phản hồi trực quan.
* Vị trí: InkWell thường được bao bọc một Material Widget đó có khả năng tương tác.
* Yêu cầu: InkWell cần được một Material Widget để hiện thị hiệu ứng gợn sóng đúng cách, vì hiệu ứng này dựa trên Material Design.
* Các thuộc tính của InkWell:
* onTap: Hàm gọi khi người dung nhấn vào InkWell
* onDoubleTap: Hàm gọi khi người dung ấn đúp.
* onLongPress: Hàm gọi khi người dung ấn và giữ lâu.
* Child: Widget con mà InkWell bao bọc, xác định khu vực có thể ấn.
* splashColor: Màu nền của hiệu ứng gợn sóng.
* highlightColor: Màu nền khi khu vực được nhấn hoặc giữ.
* Radius: Bán kính tối đa của hiệu ứng gợn sóng.
* customBorder: Tùy chỉnh dạng của vùng hiệu ứng(Vd: hình tròn, hình chữ nhật,..).
* onHover: Hàm gọi khi con trỏ chuột di chuyển qua vùng InkWell.

VD:

InkWell(

onTap: () {

print('Đã nhấn vào nút!');

},

onLongPress: () {

print('Đã nhấn lâu!');

},

splashColor: Colors.blueAccent,

highlightColor: Colors.blue.withOpacity(0.2),

borderRadius: BorderRadius.circular(8.0),

child: Container(

padding: EdgeInsets.all(16.0),

decoration: BoxDecoration(

border: Border.all(color: Colors.grey),

borderRadius: BorderRadius.circular(8.0),

),

child: Text(

'Nhấn vào tôi',

style: TextStyle(fontSize: 18),

),

),

# GestureDetetor

* Chức năng:
* Phát hiện các cử chỉ của người dung trên một khu vực cụ thể của giao diện.
* Cho phép xử lý nhiều loại sự kiện như nhấn, kéo, phóng to/ thu nhỏ, hoặc di chuyển.
* Không cung cấp hiệu ứng trực quan (như gợn sóng) mặc định, nhưng có thể kết hợp với các Widget khác để tạo hiệu ứng.
* Vị trí: GestureDetector bao bọc một Widget con (child) để làm cho Widget đó có khả năng phản hồi các tứ chi.
* Tính linh hoạt: GestureDetector hỗ trợ nhiều loại cử chỉ hơn so với các Widget khác như InkWell, phù hợp cho các tương tác phức tạp.
* Các thuộc tính của GestureDetector:
* onTap: Gọi khi người dùng nhấn và thả ngón tay.
* onDoubleTap: Gọi khi người dùng nhấn đúp.
* onLongPress: Gọi khi người dùng nhấn và giữ lâu.
* onPanStart/onPanUpdate/onPanEnd: Xử lý cử chỉ kéo (drag) với thông tin về vị trí và tốc độ.
* onScaleStart/onScaleUpdate/onScaleEnd: Xử lý cử chỉ phóng to/thu nhỏ (pinch/zoom).
* onHorizontalDragStart/Update/End: Phát hiện kéo ngang.
* onVerticalDragStart/Update/End: Phát hiện kéo dọc.
* child: Widget con mà GestureDetector bao bọc, xác định khu vực tương tác.
* behavior: Xác định cách GestureDetector xử lý các sự kiện (ví dụ: HitTestBehavior.opaque để mở rộng vùng nhận diện)

VD:

GestureDetector(

onTap: () {

print('Đã nhấn!');

},

onDoubleTap: () {

print('Đã nhấn đúp!');

},

onLongPress: () {

print('Đã nhấn lâu!');

},

onPanUpdate: (details) {

print('Đang kéo: ${details.localPosition}');

},

child: Container(

width: 200,

height: 200,

color: Colors.teal,

child: Center(

child: Text(

'Tương tác với tôi',

style: TextStyle(color: Colors.white, fontSize: 18),

),

),

),

),

# Text Field

* Chức năng:
* Cung cấp một khu vực nhập văn bản có thể chỉnh sửa
* Hỗ trợ các tính năng như gợi ý, kiểm tra đầu vào, và xử lý sự kiện khi nhập.
* Tùy chỉnh giao diện như viền, nhãn, biểu tượng và màu sắc.
* Vị trí: TextField thường được gọi trong Scaffold hoặc các Widget bố cục như Column, Row hoặc Form.
* Tương tác: TextField tự động hiển thị bàn phím khi được nhấn và hỗ trợ các kiểu bàn phím khác nhau ( văn bản, số, email,…).
* Các thuộc tính chính của TextField:
* Controller: Một TextEditingController để kiểm soát nội dung văn bản, lấy giá trị hoặc đặt giá trị ban đầu.
* Decoration: Tùy chỉnh giao diện của TextField thông qua InputDecoration, bao gồm:
* lableText: Nhãn hiển thị.
* hintText: Văn bản gợi ý khi TextField chống.
* border: Kiểu viền (VD: OutlineInputBorder, UnderlineInputBorder).
* prefixIcon/suffixIcon: Biểu tượng Icon ở đầu hoặc ở cuối TextField.
* onChanged: Hàm gọi mỗi khi nội dung TextField thay đổi.
* onSumitted: Hàm gọi khi người dung nhấn nút “Done” hoặc “Enter” trên bàn phím.
* keyboardType: Xác định loại bàn phím ze (VD:TextInputType.number,TextInputType.emailAddress).
* obscureText: Nếu true, ẩn văn bản( dung cho mật khẩu).
* maxLines: Số dòng tối đa (mặc định là 1, đặt null để hỗ trợ nhiều dòng).
* maxLength: giới hạn số ký tự tối đa.
* Enabled: Nếu false, TextField không thể chỉnh sửa.
* inputFormatters: Danh sách các bộ lọc để định dạng hoặc giới hạn đầu vào(VD: chỉ cho phép số).
* VD:

Padding(

padding: EdgeInsets.all(16.0),

child: Column(

children: [

TextField(

controller: \_controller,

decoration: InputDecoration(

labelText: 'Nhập tên của bạn',

hintText: 'Ví dụ: Nguyễn Văn A',

border: OutlineInputBorder(),

prefixIcon: Icon(Icons.person),

),

keyboardType: TextInputType.name,

onChanged: (value) {

print('Tên: $value');

},

onSubmitted: (value) {

print('Đã gửi: $value');

},

),

SizedBox(height: 20),

TextField(

decoration: InputDecoration(

labelText: 'Mật khẩu',

border: OutlineInputBorder(),

suffixIcon: Icon(Icons.visibility),

),

obscureText: true, // Ẩn văn bản (mật khẩu)

),

SizedBox(height: 20),

ElevatedButton(

onPressed: () {

print('Tên đã nhập: ${\_controller.text}');

},

child: Text('Lấy dữ liệu'),

),

],

),

),

# UI Đăng nhập

1. Mô tả tổng quan

* AppBar: Thanh tiêu đề với gradient màu và tiêu đề là “Đăng Nhập”.
* Body: Giao diện chính với hình nền với background màu đỏ đã được làm mờ, hai trường nhập liệu (TextField) cho email/SĐT và mật khẩu, nút checkbox “Nhớ mật khẩu”, và các nút hành động như (“Đăng nhập”, “ Quên mật khẩu”, Đăng ký tài khoản mới”).

1. Cấu trúc của mã nguồn

* AppBar
* Chức năng: Hiển thị tiêu đề “ĐĂNG NHẬP” với gradient màu từ đỏ sang xanh.
* Cấu hình:
* toolbarHeight: 40(chiều cao tùy chỉnh)
* centerTittle: Căn giữa tiêu đề.
* flexibleSpace: Sử dụng Container với LinearGradient để tạo hiệu ứng gradient.
* Body( UI Widget)
* Cấu trúc:
* Container: Widget giúp bao bọc toàn bộ nội dung với hình nền (AssetImage).
* BackdropFliter: Tạo hiệu ứng mờ cho hình nền bằng ImageFilter.blur.
* SingleChildScrollView: Cho phép cuộn khi nội dung vượt quá màn hình.
* Column: Sắp xếp các Widget theo chiều dọc, căn giữa với MainAxisAlignment.center.
* Các thành phần con:
* Logo: Một Container với biểu tượng Icons.restaurant\_menu và nền đỏ.
* Text: Hiển thị tiêu đề “Eats and Treats” tên của App.
* Card: Bao bọc các trường nhập liệu và nút hành động, với viền bo góc.
* TextField:

Trường email/SĐT: Có biểu tượng CupertinoIcons.person\_circle và viền bo góc.

Trường mật khẩu: Hỗ trợ ẩn/ hiện mật khẩu bằng (obscureText) với nút IconButton để chuyển trạng thái.

* Checkbox: Quản lý trạng thái “Nhớ mật khẩu” bằng biến \_ghinho được khởi tạo bằng false khi chưa ấn.
* ElevatedButton: Ba nút cho “Đăng nhập”, “Quên mật khẩu”, và “Đăng ký tài khoản mới”, với các kiểu dáng tùy chỉnh (màu nền, bo góc, khích thước).
* Các biến trạng thái:

\_ghinho: Quản lý trạng thái checkbox “Nhớ mật khẩu” ban đầu được khởi tạo là false sau khi người dung ấn tích nó sẽ thành true sẽ được setState khởi tạo lại và gắn cho giá trị true.

\_obscureText: Quản lý trạng thái ẩn/ hiện mật khẩu được khởi tạo bằng giá trị true khi người dung chưa ấn Icon sau khi người dung ấn widget setState sẽ cập nhật cho biến \_obscureText= false khi đó mật khẩu sẽ được hiện ra màn hình.