**BÁO CÁO TUẦN 8**

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

# Form

## Khái niệm

* Form là một container widget bao bọc các widget nhập liệu (như TextFormField).
* Nó yêu cầu một key (thường là GlobalKey<FormState>) để quản lý trạng thái của form, bao gồm việc xác thực và lưu dữ liệu.

## Các thành phần chính

* TextFormFiled: Widget phổ biến nhất trong Form, dung để nhập văn bản. Nó hỗ trợ xác thực qua thuộc tính validator.
* Validator: Hàm kiểm tra tính hợp lệ của dữ liệu nhập. Nếu dữ liệu không hợp lệ, nó trả về một chuỗi lỗi để hiển thị cho người dùng.
* onSaved: Hàm được gọi để lưu giá trị của trường khi gọi FormState.save().

## Quản lý trạng thái

* GlobalKey <FormState> cho phép truy cập vào các trạng thái của Form để thực hiện các thao tác như:
* Validate(): Kiểm tra tính hợp lệ của tất cả các trường trong Form.
* Save(): Lưu giá trị của các trường vào biến hoặc xử lý dữ liệu.
* Resrt(): Đặt lại Form về trạng thái ban đầu.

VD:

class LoginForm extends StatelessWidget {

final \_formKey = GlobalKey<FormState>();

@override

Widget build(BuildContext context) {

return Form(

key: \_formKey,

child: Column(

children: [

TextFormField(

decoration: InputDecoration(labelText: 'Email'),

validator: (value) {

if (value == null || !value.contains('@')) {

return 'Email không hợp lệ';

}

return null;

},

),

TextFormField(

decoration: InputDecoration(labelText: 'Mật khẩu'),

obscureText: true,

validator: (value) {

if (value == null || value.length < 6) {

return 'Mật khẩu phải dài hơn 6 ký tự';

}

return null;

},

),

ElevatedButton(

onPressed: () {

if (\_formKey.currentState!.validate()) {

ScaffoldMessenger.of(context).showSnackBar(

SnackBar(content: Text('Đăng nhập thành công')),

);

}

},

child: Text('Đăng nhập'),

),

],

),

);

}

}

# TextFormField

## Chức năng

* Cho phép người dung nhập và chỉnh sử văn bản.
* Tích hợp các tính năng như xác thực dữ liệu, hiển thị thông báo lỗi, và lưu giá trị nhập liệu.
* Thường được sử dụng trong Form để thu thập thông tin như email, mật khẩu, v.v.

## Các thuộc tính quan trọng

* Decoration: Tùy chỉnh giao diện của trường nhập liệu, bao gồm:
* labelText: nhãn hiển thị
* hintText: Gợi ý văn bản.
* icon,prefixIcon, suffixIcon: Thêm biểu tượng.
* boder: Tùy chỉnh viền (vd: OutlineInputBoder).
* Validator: hàm kiểm tra tính hợp lệ của giữ liệu. Trả về một chuỗi nếu dữ liệu không hợp lệ, hoặc null nếu hợp lệ.
* onSaved: Hàm được gọi khi FormState.save() chạy, dung để lưu giá trị nhập.
* controller: một TextEditingController để theo dõi và điều kiển nội dung văn bản.
* keyboardType: xác định loại bán phím (vd: TextInputType.emailAddress,TextInputType.number).
* obscureText: ẩn văn bản khi nhập (dung cho mật khẩu).
* onChanged: Hàm gọi mỗi khi nội dung trường thay đổi.

VD:

TextFormField(

decoration: InputDecoration(

labelText: 'Mật khẩu',

border: OutlineInputBorder(),

),

obscureText: true,

validator: (value) {

if (value == null || value.length < 6) {

return 'Mật khẩu phải dài hơn 6 ký tự';

}

return null;

},

onSaved: (value) {

// Lưu mật khẩu vào biến hoặc gửi đến server

},

)

# DropDownList

## Chức năng

* Hiển thị một dánh sách thả xuống (dropdown) chứa các mục mà người dùng có thể chọn.
* Khi người dung chọn một mục, giá trị của DropdownButton sẽ được cập nhật và có thể được xử lý thông qua hàm onChanged.

## Các thuộc tính quan trọng

* Value: giá trị hiện tại của DropDownButton (phải nằm trong danh sách items).
* Items: Danh sách các DropdownMenuItem đại diện cho các tùy chọn. Mỗi DropdownMenuItem cần có value(giá trị thực) và child (nội dung hiển thị).
* onChanged: Hàm được gọi khi người dung chọn một giá trị mới. Thường dùng để cập nhật trạng thái.
* Hint: Widget hiển thị khi chưa có giá trị nào được chọn.
* dropdownColor: màu nền của danh sách thả xuống.
* style: Tùy chỉnh kiểu chữ của các mục.
* isExpanded: nếu true, DropdownButton sẽ chiếm toàn bộ chiều rộng khả dụng.

VD:

class MyDropdown extends StatefulWidget {

@override

\_MyDropdownState createState() => \_MyDropdownState();

}

class \_MyDropdownState extends State<MyDropdown> {

String? selectedCategory;

@override

Widget build(BuildContext context) {

return DropdownButton<String>(

value: selectedCategory,

isExpanded: true,

hint: Text('Chọn danh mục'),

items: <String>['Điện tử', 'Thời trang', 'Gia dụng']

.map((String value) {

return DropdownMenuItem<String>(

value: value,

child: Text(value),

);

}).toList(),

onChanged: (String? newValue) {

setState(() {

selectedCategory = newValue;

});

},

);

}

}

# Đăng nhập UI(bổ sung)

## Chức năng chính

* Màn hình đăng nhập (DangNhap) cho phép người dùng nhập email/số điện thoại và mật khẩu.
* Hỗ trợ xác thực dữ liệu (validation) cho email/số điện thoại và mật khẩu.
* Tích hợp tính năng "Nhớ mật khẩu" bằng checkbox.
* Điều hướng đến màn hình đăng ký (DangKy) và màn hình quản lý (QuanLy) thông qua các nút.
* Giao diện được thiết kế với ảnh nền, hiệu ứng làm mờ, và các thành phần UI như TextFormField, ElevatedButton, và Checkbox.

## Các thành phần chính:

* Widget chính: MaterialApp với hỗ trợ đa ngôn ngữ (tiếng Việt là mặc định).
* Màn hình: DangNhap (StatefulWidget) với Form chứa hai TextFormField (email/số điện thoại và mật khẩu).
* Điều hướng: Sử dụng Navigator.pushNamed để chuyển giữa các màn hình (DangKy, QuanLy).
* Thẩm mỹ: Sử dụng ảnh nền, hiệu ứng làm mờ (BackdropFilter), gradient trên AppBar, và các thành phần được bo góc.

## Cấu trúc mã

* Hàm main() khởi tạo ứng dụng với MaterialApp. Home:DangnHAP() dễ thuận tiện để chuyển qua các giao diện khác vd: DangKy(),QuanLy(),…
* Định nghĩa các route(/DangNhap,./DangKy,/QuanLy) và thiết lập ngỗn ngữ mặc định tiếng việt.
* UI gồm Scaffold với AppBar và body
* Body chứa Form bên trong Card.
* Form gồm các widget như TextFormField, ElevetedButtom là các widget chính để tạo lên giao diện DangNhap.

# DangKy

## Các chức năng

* Xây dựng một màn hình đăng ký để người dùng nhập thông tin như email/số điện thoại, mật khẩu, xác nhận mật khẩu, ngày sinh, và chọn thành phố.
* Đảm bảo giao diện đẹp, dễ dùng, và có kiểm tra dữ liệu nhập (validation).
* Cho phép quay lại màn hình đăng nhập (DangNhap) khi cần.

## Nội dung code

* File: DangKy.dart
* Widget chính: DangKy là một StatefulWidget, với trạng thái được quản lý trang class DangKy1.
* Giao diện:
* AppBar: Có nút quay lại(back) để về màn hình đăng nhập và nút làm mới (refresh) chưa tích hợp chức năng.
* Body: Một Form trong Card, chứa:
* 4 TextFormField: Email/SĐT, mật khẩu, xác nhận mật khẩu, ngày sinh.
* 1 DropdownButtonFormField: chọn thành phố
* Tính năng:
* Kiểm tra dữ liệu nhập (email/SĐT hợp lệ, mật khẩu tối thiểu 8 ký tự, xác nhận mật khẩu khớp, ngày sinh bắt buộc).
* Hiển thị thông báo “Đăng ký thành công “ bằng SnackBar khi dữ liệu hợp lệ.
* Điều hướng về màn hình đăng nhập qua nút back hoặc liên kết “Đăng nhập”.
* Thư viện sử dụng:
* Flutter/material.dart: Các widget cơ bản.
* Google\_fonts: Dùng font Poppins cho tiều đề “Đăng Ký”.

## Phần chính của UI và Logic

* **Giao diện**
* Thiết kế giao diện với nền xám (Colors.grey[400]), đặt Form trong một Card bo góc để nổi bật.
* Tiêu đề "Đăng Ký" dùng font Poppins (từ google\_fonts), kèm biểu tượng tài khoản (Icons.account\_circle).
* Các ô nhập liệu (TextFormField) có viền bo góc, màu viền đổi thành xanh khi nhấn vào (focusedBorder), và có biểu tượng bên trái (prefixIcon) như tài khoản, khóa, lịch.
* Ô mật khẩu và xác nhận mật khẩu có nút mắt (Icons.visibility) để hiện/ẩn mật khẩu.
* Ô ngày sinh dùng showDatePicker để chọn ngày, hiển thị định dạng ngày/tháng/năm.
* Danh sách thành phố được tạo bằng DropdownButtonFormField với 9 thành phố em tự thêm (Hà Nội, TP.HCM, ...).
* **Logic**
* Kiểm tra dữ liệu:
  + Email/số điện thoại: Dùng biểu thức chính quy (RegExp) để kiểm tra định dạng email hoặc số điện thoại 10 chữ số bắt đầu bằng 0.
  + Mật khẩu: Phải có ít nhất 8 ký tự.
  + Xác nhận mật khẩu: Phải giống mật khẩu.
  + Ngày sinh: Bắt buộc chọn.
  + Thành phố: Chưa kiểm tra bắt buộc (em quên thêm validator).
* Quản lý trạng thái:
  + Dùng TextEditingController để lấy dữ liệu từ các ô nhập.
  + Dùng GlobalKey<FormState> để kiểm tra và lưu dữ liệu form.
  + Biến \_matKhau và \_xacNhan để bật/tắt ẩn mật khẩu.
  + Biến \_selectedDate để lưu ngày sinh.
* Điều hướng:
  + Nút back và liên kết "Đăng nhập" dùng Navigator.pushReplacementNamed để quay về màn hình đăng nhập.
* Quản lý bộ nhớ:
  + Em gọi dispose() cho tất cả controller để tránh rò rỉ bộ nhớ.

# QuanLy()

## Các chức năng

* Xây dựng một màn hình quản lý để hiển thị thông tin tổng quan của cửa hàng, bao gồm hóa đơn, bàn, doanh thu, và các chức năng quản lý khác.
* Tích hợp hai danh sách thả xuống (dropdown) để lọc dữ liệu theo thời gian và chi nhánh.
* Thiết kế giao diện đẹp, dễ sử dụng, với các thành phần như card, banner, và thanh điều hướng dưới cùng.

## Nội dung code

* File: QuanLy.dart
* Widget chính: QuanLy là một StatefulWidget, với trạng thái được quản lý trong class \_QuanLyState.
* Giao diện:
* AppBar:
* Có logo (Image.asset), tiêu đề "GoodFood", và hai nút hành động (thông báo, trợ giúp).
* Dùng gradient nền xanh nhạt và trong suốt (backgroundColor: Colors.transparent).
* Có phần bottom chứa hai DropdownButton để chọn thời gian (Hôm nay, Hôm qua, Tuần này) và chi nhánh (Tất cả, Hà Nội, HCM).
* Body:
  + - * Nền gradient xanh nhạt, chứa SingleChildScrollView để cuộn nội dung.
      * Card thông tin: Hiển thị số hóa đơn, bàn đang sử dụng, và các nút chức năng (Vay vốn, Kiểm kho, ...).
      * Banner quảng cáo: Hiển thị thông tin hướng dẫn xuất hóa đơn với gradient xanh và hiệu ứng bóng.
      * Card doanh thu: Hiển thị trạng thái doanh thu (hiện tại là "Chưa có dữ liệu") với các nút lọc theo giờ, ngày, thứ.
* BottomNavigationBar: Thanh điều hướng dưới cùng với 5 tab (Tổng Quan, Hàng hóa, Hóa đơn, Sổ quỹ, Nhiều hơn).
* Thư viện sử dụng: flutter/material.dart: Cung cấp các widget cơ bản của Material Design.

## Phần chính của UI và logic

* **Giao diện**
* AppBar:
  + Em tách AppBar thành hàm \_buildAppBar() để code gọn hơn.
  + Thêm logo từ file ảnh (assets/images/Logo.jpg) và tiêu đề "GoodFood" với chữ đỏ đậm.
  + Dùng gradient xanh nhạt (Color(0xFF4FC3F7) đến Color(0xFFB2EBF2)) làm nền, đặt elevation: 0 để loại bỏ bóng.
  + Thêm hai DropdownButton trong bottom để chọn thời gian và chi nhánh, với viền dưới xám và icon mũi tên.
* Body:
  + Nền body dùng gradient giống AppBar, với extendBodyBehindAppBar: true để gradient kéo dài lên trên.
  + Dùng SingleChildScrollView để cuộn nội dung, thêm padding trên (top: 150.0) để tránh bị che bởi AppBar.
  + Card thông tin:
    - Hiển thị số liệu giả (0 hóa đơn, 0 bàn) trong hai cột, dùng Row và Column.
    - Thêm 5 nút chức năng (Vay vốn, Kiểm kho, ...) với icon và nhãn, dùng hàm \_buildIconButton.
    - Card có nền trắng trong suốt (Colors.white.withOpacity(0.9)), bo góc, và bóng nhẹ (elevation: 4).
  + Banner:
    - Tạo container với gradient xanh đậm (Color(0xFF1976D2) đến Color(0xFF42A5F5)), bo góc, và hiệu ứng bóng.
    - Hiển thị văn bản hướng dẫn xuất hóa đơn, căn giữa, chữ trắng, đậm.
  + Card doanh thu:
    - Hiển thị tiêu đề "Doanh thu" và ba nút lọc (Giờ, Ngày, Thứ).
    - Hiện thông báo "Chưa có dữ liệu" với ghi chú, căn giữa.
* BottomNavigationBar:
  + Thêm 5 tab với icon và nhãn, tab "Tổng Quan" được chọn mặc định (currentIndex: 0).
  + Màu xanh cho tab được chọn, xám cho tab không chọn.
* **Logic**
* Dropdown:
  + Quản lý trạng thái \_selectedValue (thời gian) và \_chiNhanh (chi nhánh) bằng setState.
  + Danh sách thời gian (\_ngay) và chi nhánh (\_allChiNhanh) được định nghĩa tĩnh.
* Hàm phụ:
  + \_buildIconButton: Tạo nút icon với nhãn, tái sử dụng cho các chức năng trong card thông tin.