```
import 'package:flutter/material.dart';

void main() => runApp(MyApp());

class Product {
  String name;
  double purchasePrice;
  double salePrice;
  int quantity;
  String description;
  String category;
  String imageUrl;
  bool isActive;
  bool onSale;
  double discount;

  Product({
    required this.name,
    required this.purchasePrice,
    required this.salePrice,
    required this.quantity,
    required this.description,
    required this.category,
    required this.imageUrl,
    required this.isActive,
    required this.onSale,
    required this.discount,
  });
```

```
}

class MyApp extends StatefulWidget {
  @override
  State<MyApp> createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
  final List<Product> _products = [];

  void _addProduct(Product product) {
    setState(() {
      _products.add(product);
    });
  }

  void _editProduct(Product updatedProduct, int index) {
    setState(() {
      _products[index] = updatedProduct;
    });
  }

  void _removeProduct(int index) {
    setState(() {
      _products.removeAt(index);
    });
  }
```

```
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Cadastro de Produtos',
      home: ProductListScreen(
        products: _products,
        onAdd: _addProduct,
        onEdit: _editProduct,
        onRemove: _removeProduct,
      ),
    );
  }
}

class ProductForm extends StatefulWidget {
  final void Function(Product product) onSubmit;
  final Product? product;
  final int? index;

  ProductForm({required this.onSubmit, this.product, this.index});

  @override
  State<ProductForm> createState() => _ProductFormState();
}

class _ProductFormState extends State<ProductForm> {
  final _formKey = GlobalKey<FormState>();
```

```
late String _name;
double _purchasePrice = 0.0;
double _salePrice = 0.0;
int _quantity = 1;
late String _description;
String _category = 'Roupas';
String _imageUrl = '';
bool _isActive = true;
bool _onSale = false;
double _discount = 0.0;

@override
void initState() {
  super.initState();
  if (widget.product != null) {
    final p = widget.product!;
    _name = p.name;
    _purchasePrice = p.purchasePrice;
    _salePrice = p.salePrice;
    _quantity = p.quantity;
    _description = p.description;
    _category = p.category;
    _imageUrl = p.imageUrl;
    _isActive = p.isActive;
    _onSale = p.onSale;
    _discount = p.discount;
  } else {
```

```
    _name = '';
    _description = '';
  }
}

void _submit() {
  if (_formKey.currentState!.validate()) {
    _formKey.currentState!.save();
    final product = Product(
      name: _name,
      purchasePrice: _purchasePrice,
      salePrice: _salePrice,
      quantity: _quantity,
      description: _description,
      category: _category,
      imageUrl: _imageUrl,
      isActive: _isActive,
      onSale: _onSale,
      discount: _discount,
    );
    widget.onSubmit(product);
    Navigator.pop(context);
  }
}

@override
Widget build(BuildContext context) {
  return Scaffold(
```

```
appBar: AppBar(
  title: Text(
    'Cadastro de Produto',
    style: TextStyle(fontSize: 17),
  ),
  backgroundColor: Color.fromARGB(255, 102, 59, 81),
),
body: Padding(
  padding: const EdgeInsets.all(16.0),
  child: Form(
    key: _formKey,
    child: ListView(
      children: [
        Center(
          child: Text(
            'Informações do Produto',
            style: TextStyle(
              fontSize: 17,
              fontWeight: FontWeight.bold,
              color: Color.fromARGB(255, 102, 59, 81),
            ),
          ),
        ),
        SizedBox(height: 10),
        TextFormField(
          initialValue: _name,
          decoration: InputDecoration(labelText: 'Nome'),
          validator: (value) => value!.isEmpty ? 'Informe o nome' : null,
```

```
  onSaved: (value) => _name = value!,
),
TextFormField(
  initialValue: _purchasePrice.toString(),
  decoration: InputDecoration(labelText: 'Preço de compra'),
  keyboardType: TextInputType.number,
  validator: (value) => value!.isEmpty ? 'Informe o preço' : null,
  onSaved: (value) => _purchasePrice = double.tryParse(value!) ?? 0.0,
),
TextFormField(
  initialValue: _salePrice.toString(),
  decoration: InputDecoration(labelText: 'Preço de venda'),
  keyboardType: TextInputType.number,
  validator: (value) => value!.isEmpty ? 'Informe o preço' : null,
  onSaved: (value) => _salePrice = double.tryParse(value!) ?? 0.0,
),
TextFormField(
  initialValue: _quantity.toString(),
  decoration: InputDecoration(labelText: 'Quantidade'),
  keyboardType: TextInputType.number,
  validator: (value) => value!.isEmpty ? 'Informe a quantidade' : null,
  onSaved: (value) => _quantity = int.tryParse(value!) ?? 0,
),
TextFormField(
  initialValue: _description,
  decoration: InputDecoration(labelText: 'Descrição'),
  validator: (value) => value!.isEmpty ? 'Informe a descrição' : null,
  onSaved: (value) => _description = value!,
```

```
          ),
          DropdownButtonFormField<String>(
           value: _category,
           decoration: InputDecoration(labelText: 'Categoria'),
           items: [
            DropdownMenuItem(
             child: Row(children: [Icon(Icons.checkroom), SizedBox(width: 5),
Text('Roupas')]),
             value: 'Roupas',
            ),
            DropdownMenuItem(
             child: Row(children: [Icon(Icons.directions_run), SizedBox(width: 5),
Text('Tênis')]),
             value: 'Tênis',
            ),
            DropdownMenuItem(
             child: Row(children: [Icon(Icons.laptop), SizedBox(width: 5),
Text('Eletrônicos')]),
             value: 'Eletrônicos',
            ),
           ],
           onChanged: (value) => setState(() => _category = value!),
          ),
          TextFormField(
           initialValue: _imageUrl,
           decoration: InputDecoration(labelText: 'URL da Imagem'),
           onSaved: (value) => _imageUrl = value ?? '',
          ),
          SwitchListTile(
```

```
 title: Text('Produto Ativo'),
 value: _isActive,
 activeColor: Color.fromARGB(255, 102, 59, 81),
 onChanged: (value) => setState(() => _isActive = value),
),
CheckboxListTile(
 title: Text('Em Promoção'),
 value: _onSale,
 activeColor: Color.fromARGB(255, 102, 59, 81),
 onChanged: (value) => setState(() => _onSale = value!),
),
Column(
 crossAxisAlignment: CrossAxisAlignment.start,
 children: [
  Text('Desconto: ${_discount.round()}%'),
  Slider(
   value: _discount,
   onChanged: (value) => setState(() => _discount = value),
   min: 0,
   max: 100,
   divisions: 20,
   label: '${_discount.round()}%',
   activeColor: Color.fromARGB(255, 102, 59, 81),
  ),
 ],
),
SizedBox(height: 40),
ElevatedButton(
```

```
            style: ElevatedButton.styleFrom(
              backgroundColor: Color.fromARGB(255, 102, 59, 81),
            ),
            onPressed: _submit,
            child: Text('Cadastrar Produto'),
          ),
        ],
      ),
    ),
  ),
 );
 }
}

class ProductListScreen extends StatelessWidget {
 final List<Product> products;
 final void Function(Product) onAdd;
 final void Function(Product, int) onEdit;
 final void Function(int) onRemove;

 ProductListScreen({
  required this.products,
  required this.onAdd,
  required this.onEdit,
  required this.onRemove,
 });

 @override
```

```
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      centerTitle: true,
      backgroundColor: Color.fromARGB(255, 102, 59, 81),
      title: Text(
        'Lista e Cadastro de Produtos (${products.length})',
        style: TextStyle(fontSize: 16), // Texto com tamanho reduzido
      ),
    ),
    body: ListView.builder(
      itemCount: products.length,
      itemBuilder: (ctx, i) {
        final p = products[i];
        return Card(
          shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(20)),
          margin: EdgeInsets.all(10),
          child: ListTile(
            leading: ClipOval(
              child: Image.network(
                p.imageUrl,
                width: 50,
                height: 50,
                fit: BoxFit.cover,
                errorBuilder: (context, error, stackTrace) => Icon(Icons.image_not_supported),
              ),
            ),
            title: Text(p.name),
```

```
        subtitle: Text('R\$ ${p.salePrice.toStringAsFixed(2)}\nQtd: ${p.quantity}'),
        isThreeLine: true,
        onTap: () {
          Navigator.push(
            context,
            MaterialPageRoute(
              builder: (_) => ProductDetailsScreen(
                product: p,
                onEdit: (updated) => onEdit(updated, i),
                onDelete: () {
                  onRemove(i);
                  Navigator.pop(context);
                },
              ),
            ),
          );
        },
      ),
    );
  },
),
floatingActionButton: FloatingActionButton(
  onPressed: () => Navigator.push(
    context,
    MaterialPageRoute(
      builder: (_) => ProductForm(onSubmit: onAdd),
    ),
  ),
```

```
      child: Icon(Icons.add),
      backgroundColor: Color.fromARGB(255, 102, 59, 81),
    ),
   );
  }
}

class ProductDetailsScreen extends StatelessWidget {
 final Product product;
 final void Function(Product) onEdit;
 final VoidCallback onDelete;

 ProductDetailsScreen({
  required this.product,
  required this.onEdit,
  required this.onDelete,
 });

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   backgroundColor: Color.fromARGB(255, 243, 232, 248),
   appBar: AppBar(
    backgroundColor: Color.fromARGB(255, 102, 59, 81),
    title: Text(product.name),
    actions: [
     IconButton(
      icon: Icon(Icons.edit),
```

```
      onPressed: () => Navigator.push(
        context,
        MaterialPageRoute(
          builder: (_) => ProductForm(
            product: product,
            onSubmit: onEdit,
          ),
        ),
      ),
    ),
    IconButton(
      icon: Icon(Icons.delete),
      onPressed: onDelete,
    ),
  ],
),
body: Padding(
  padding: const EdgeInsets.all(16.0),
  child: ListView(
    children: [
      Image.network(
        product.imageUrl,
        height: 200,
        errorBuilder: (context, error, stackTrace) =>
            Icon(Icons.image_not_supported, size: 100),
      ),
      SizedBox(height: 16),
      Text('Nome: ${product.name}'),
```

```
          Text('Preço de Compra: R\$ ${product.purchasePrice.toStringAsFixed(2)}'),
          Text('Preço de Venda: R\$ ${product.salePrice.toStringAsFixed(2)}'),
          Text('Quantidade: ${product.quantity}'),
          Text('Descrição: ${product.description}'),
          Text('Categoria: ${product.category}'),
          Text('Ativo: ${product.isActive ? "Sim" : "Não"}'),
          Text('Em Promoção: ${product.onSale ? "Sim" : "Não"}'),
          Text('Desconto: ${product.discount.round()}%'),
        ],
      ),
    ),
  );
 }
}
```