Tienda online de bolsos - Practica 2

Descripción general del proyecto

Esta aplicación permite a los usuarios explorar diferentes productos en una interfaz simple. La aplicación se ha modificado para cumplir los siguientes requisitos:

1. **Vista de GridView en 2 columnas:** Cambio en el diseño para que la lista de productos se visualicen en una cuadrícula de 2 columnas por fila.
2. **Mostrar precio original y actual:** Se muestra el precio original tachado junto al precio al que se encuentra actualmente el producto.
3. **Mostrar porcentaje de descuento:** Se muestra el descuento que se ha aplicado para cada producto.
4. **Etiqueta para productos recomendados:** Los productos recomendados muestran una etiqueta adecuada para ello.
5. **Estrellas de valoración:** Para cada producto se muestran estrellas de valoración de 0 a 5

Explicación de funciones implementadas

1. **Cambio a GridView**

Se ha cambiado el ListView que habia en el ejercicio por un GridView. A continuación se muestra el codigo implementado.

```
Unset
GridView.builder(gridDelegate:
SliverGridDelegateWithFixedCrossAxisCount(crossAxisCount: 2),
          itemCount: dummyProducts.length,
          itemBuilder: (ctx, i) =>
ProductItem(dummyProducts[i]),
          )
```

2. **Mostrar precio original y actual**

   Se ha añadido la etiqueta de precio original (tachado) junto al precio actual. Para ello se ha creado un widget personalizado

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

class PricesBadge extends StatelessWidget {
  final double original_price;
  final double price;
  final double fontSize;

  PricesBadge(this.original_price, this.price, this.fontSize);

  @override
  Widget build(BuildContext context) {
    if(original_price == price) return SizedBox.shrink();

    return Text(
        '\$${original_price}',
        style: TextStyle(color: Colors.grey,
        fontSize: fontSize, decoration: TextDecoration.lineThrough)
    );
  }
}

//Para ejeutar el widget se hace de la siguiente manera
PricesBadge(product.original_price, product.price, 20)
```

3. **Mostrar descuento**

   Para mostrar el descuento aplicado se ha hecho servir una condición para verificar que el precio original y el precio actual no sean el mismo. En caso de que sean el mismo, la etiqueta no se muestra. También se ha utilizado la fórmula que se muestra a continuación.

```
Unset
/*Formula for the discount*/
double discountPercentage = 0;
    if (product.price < product.original_price) {
      discountPercentage =
          ((product.original_price - product.price) /
product.original_price) * 100;
    }

/*DETAIL SCREEN*/
if(discountPercentage > 0)
               Container(
```

```
                padding: EdgeInsets.all(2),
                color: Colors.red,
                child: Text(
                  '-${discountPercentage.toStringAsFixed(0)}%',
                  style: TextStyle(
                    color: Colors.white,
                    fontWeight: FontWeight.bold,
                    fontSize: 17,
                  ),
                ),
              )

//MAIN SCREEN
if (discountPercentage > 0)
                  Positioned(
                    child: Container(
                      padding: EdgeInsets.all(2),
                      color: Colors.red,
                      child: Text(
                        '-${discountPercentage.toStringAsFixed(0)}%',
                        style: TextStyle(
                          color: Colors.white,
                          fontWeight: FontWeight.bold,
                          fontSize: 12,
                        ),
                      ),
                    ),
                  )
```

4. **Mostrar etiqueta de recomendado**

Se ha añadido una variable booleana (true o false) para saber si un producto es recomendado o no. En caso de ser recomendado, saldrá una etiqueta acorde a ello. Este proceso se ha hecho en un widget aparte.

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

class RecommendedBadge extends StatelessWidget {
  final bool isRecommended;
  final double fontSize;

  RecommendedBadge(this.isRecommended, this.fontSize);
```

```
  @override
  Widget build(BuildContext context) {
    if(!isRecommended) return SizedBox.shrink();

    return Container(
      padding: EdgeInsets.all(4),
      decoration: BoxDecoration(
          color: Colors.lightGreen,
          borderRadius: BorderRadius.circular(20)
      ),
      child: Text(
        'Recommended',
        style: TextStyle(
          color: Colors.green[900],
          fontWeight: FontWeight.bold,
          fontSize: fontSize,
        ),
      ),
    );
  }
}

//Uso del widget
RecommendedBadge(product.isRecommended, 20)
```

5. **Mostrar valoración**

Para la valoración, se han utilizado estrellas por defecto de flutter y el texto de valoracion que se ha añadido a la clase de producto. Por lo tanto hay dos elementos a añadir. Este apartado se ha generado en un widget aparte. Este apartado solo aparece a nivel de detalle del producto

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

class buildStars extends StatelessWidget {

  final double rating;

  buildStars(this.rating);
```

```
  @override
  Widget build(BuildContext context) {
    int fullStars = rating.floor();

    bool halfStar = (rating - fullStars) >= 0.5;

    int maxStars = 5;

    return Row(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
      Row(
        crossAxisAlignment: CrossAxisAlignment.center,
        children: List.generate(maxStars, (index) {
          if (index < fullStars) {
            return Icon(Icons.star, color: Colors.amber, size: 30,);
          }
          else if (index == fullStars && halfStar) {
            return Icon(Icons.star_half, color: Colors.amber, size: 30);
          }
          else {
            return Icon(Icons.star_border, color: Colors.amber, size: 30);
          }
        }),

      ),
        if(rating.truncate() == rating)
        Text('${rating.truncate()}', style: TextStyle(fontSize: 20),)
        else
          Text('${rating}', style: TextStyle(fontSize: 20),)
      ],
    );


  }
}

//Ejemplo de su uso
buildStars(product.rating)
```