



UNIVERSIDAD CÉSAR VALLEJO

FACULTAD DE INGENIERÍA Y ARQUITECTURA ESCUELA
PROFESIONAL DE INGENIERÍA DE SISTEMAS

TÍTULO

“Documentación Examen Parcial 02 Ejercicio2”

Experiencia Curricular
PROGRAMACIÓN DE APLICACIONES MÓVILES

Autor

Tokumoto Mora, Jhon Jorge Hideaki

ASESOR

MARTÍN GUSTAVO SALCEDO QUIÑONES

SECCIÓN

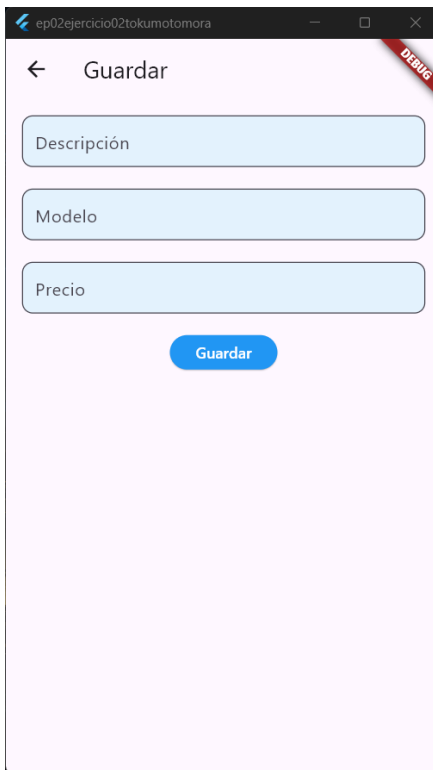
B1

TRUJILLO- PERÚ

(2024-01)

Ejercicio 02

Elabora una aplicación que haga uso de sqlite con Android Studio o Flutter que permita gestionar los datos de artefactos eléctricos (descripción, modelo y precio). Debe permitir registrar, modificar, eliminar y listar los artefactos. El diseño de las interfaces queda al criterio del estudiante.



ep02ejercicio02tokumotomora

← Guardar

Descripción

Modelo

Precio

Guardar

This screenshot shows the 'Guardar' (Save) screen. It features three light blue input fields for 'Descripción', 'Modelo', and 'Precio'. A blue 'Guardar' button is positioned below the fields. A red 'Drag' label is visible in the top right corner.



ep02ejercicio02tokumotomora

← Artefactos Electrónicos

Televisor Led Smart TV FHD 43

TELEVISOR LG 65 4K UHD THINQ AI SMART

+

This screenshot shows the 'Artefactos Electrónicos' list screen. It displays two items: 'Televisor Led Smart TV FHD 43' and 'TELEVISOR LG 65 4K UHD THINQ AI SMART'. Each item has a grey edit icon to its right. A blue '+' button is at the bottom right. A red 'Drag' label is in the top right corner.



ep02ejercicio02tokumotomora

← Artefactos Electrónicos

Televisor Led Smart TV FHD 43

TELEVISOR LG 65 4K UHD THINQ AI SMART

+

This screenshot shows the 'Artefactos Electrónicos' list screen, similar to the previous one but with a red delete icon (trash can) to the left of the second item, 'TELEVISOR LG 65 4K UHD THINQ AI SMART'. A blue '+' button is at the bottom right. A red 'Drag' label is in the top right corner.

Primero creamos nuestro dart llamado artefacto para poder especificar que datos y de que tipo son los que vamos a usar

```
1 class Artefacto{
2   int id;
3   String descripcion;
4   String modelo;
5   String precio;
6
7   Artefacto(
8     {required this.id,
9     required this.descripcion,
10    required this.modelo,
11    required this.precio});
12
13   Map<String, dynamic> tomap() {
14     return {'id': id, 'descripcion': descripcion, 'modelo': modelo, 'precio': precio};
15   }
16
17   Map<String, dynamic> tomapI() {
18     return {'descripcion': descripcion, 'modelo': modelo, 'precio': precio};
19   }
20 }
```

También tenemos nuestra dart llamado db en donde guardaremos todos nuestros datos que agregaremos

```
1 import 'package:ep02ejercicio02tokumotomora/artefacto.dart';
2 import 'package:sqflite_common_ffi/sqflite_ffi.dart';
3 import 'package:path/path.dart';
4
5 class DB {
6   static Future<Database> _openDB() async {
7     sqfliteFfiInit(); // Inicializar sqflite_common_ffi
8     databaseFactory = databaseFactoryFfi; // Establecer la fábrica de la base de datos
9
10    return openDatabase(join(await getDatabasesPath(), 'artefacto.db'),
11      onCreate: (db, version) {
12        return db.execute(
13          "Create Table artefacto (id INTEGER PRIMARY KEY AUTOINCREMENT, descripcion TEXT, modelo Text, precio TEXT)",
14        );
15      }, version: 1);
16  }
17
18  static Future<int> insert(Artefacto artefacto) async {
19    Database database = await _openDB();
20
21    return database.insert("artefacto", artefacto.tomapI());
22  }
23
24  static Future<int> delete(Artefacto artefacto) async {
25    Database database = await _openDB();
26
27    return database
28      .delete("artefacto", where: "id = ?", whereArgs: [artefacto.id]);
29  }
30
31  static Future<int> update(Artefacto artefacto) async {
32    Database database = await _openDB();
33
34    return database.update("artefacto", artefacto.tomap(),
35      where: "id= ?", whereArgs: [artefacto.id]);
36  }
37
38  static Future<List<Artefacto>> artefacto() async {
39    Database database = await _openDB();
40    final List<Map<String, dynamic>> artefactoMap =
41      await database.query("artefacto");
42
43    return List.generate(
44      artefactoMap.length,
45      (i) => Artefacto(
46        id: artefactoMap[i]['id'],
47        descripcion: artefactoMap[i]['descripcion'],
48        modelo: artefactoMap[i]['modelo'],
49        precio: artefactoMap[i]['precio']));
50  }
51 }
52
```

← Agregar

← Eliminar

← Actualizar

← Listado

Luego continuaremos con nuestro dart llamado editar en el cual funciona también como un agregar en donde nos permitiría con la ayuda de nuestro db.dart actualizar y agregar nuevos artefactos

```
1 import 'package:ep02ejercicio02tokumotomora/db.dart';
2 import 'package:ep02ejercicio02tokumotomora/artefacto.dart';
3 import 'package:flutter/material.dart';
4
5 class Editar extends StatelessWidget {
6   final _formkey = GlobalKey<FormState>();
7   final descripcionController = TextEditingController();
8   final modeloController = TextEditingController();
9   final precioController = TextEditingController();
10
11   @override
12   Widget build(BuildContext context) {
13     final producto = ModalRoute.of(context)!.settings.arguments as Artefacto;
14     descripcionController.text = producto.descripcion;
15     modeloController.text = producto.modelo;
16     precioController.text = producto.precio;
17
18     return Scaffold(
19       appBar: AppBar(title: Text("Guardar")),
20       body: Container(
21         child: Padding(
22           child: Form(
23             key: _formkey,
24             child: Column(
25               children: <Widget>[
26                 TextFormField(
27                   controller: descripcionController,
28                   validator: (value){
29                     if(value.toString().isEmpty)
30                       return "La descripción es obligatoria";
31                     return null;
32                   },
33                   decoration: InputDecoration(
34                     labelText: "Descripción",
35                     filled: true,
36                     fillColor: Colors.blue.shade50, // Fondo azul claro
37                     border: OutlineInputBorder(
38                       borderRadius: BorderRadius.circular(10.0),
39                     ),
40                   ),
41                 ),
42                 SizedBox(
43                   height: 20,
44                 ),
45                 TextFormField(
46                   controller: modeloController,
47                   validator: (value){
48                     if (value.toString().isEmpty)
49                       return "El modelo es obligatorio";
50                     return null;
51                   },
52                   decoration: InputDecoration(
53                     labelText: "Modelo",
54                     filled: true,
55                     fillColor: Colors.blue.shade50, // Fondo azul claro
56                     border: OutlineInputBorder(
57                       borderRadius: BorderRadius.circular(10.0),
58                     ),
59                   ),
60                 ),
61                 SizedBox(
62                   height: 20,
63                 ),
64                 TextFormField(
65                   controller: precioController,
66                   validator: (value){
67                     if (value.toString().isEmpty)
68                       return "El precio es obligatorio";
69                     return null;
70                   },
71                   decoration: InputDecoration(
72                     labelText: "Precio",
73                     filled: true,
74                     fillColor: Colors.blue.shade50, // Fondo azul claro
75                     border: OutlineInputBorder(
76                       borderRadius: BorderRadius.circular(10.0),
77                     ),
78                   ),
79                 ),
80                 SizedBox(
81                   height: 20,
82                 ),
83                 ElevatedButton(
84                   onPressed: (){
85                     if(_formkey.currentState!.validate()){
86                       if(producto.id > 0){
87                         producto.descripcion = descripcionController.text;
88                         producto.modelo = modeloController.text;
89                         producto.precio = precioController.text;
90                         DB.update(producto);
91                       } else {
92                         DB.insert(Artefacto(
93                           id: producto.id,
94                           descripcion: descripcionController.text,
95                           modelo: modeloController.text,
96                           precio: precioController.text));
97                       }
98                       Navigator.pushNamed(context, "/");
99                     }
100                   },
101                   style: ButtonStyle(
102                     backgroundColor: MaterialStateProperty.all<Color>(Colors.blue), // Fondo azul
103                     foregroundColor: MaterialStateProperty.all<Color>(Colors.white), // Texto blanco
104                   ),
105                   child: Text("Guardar"),
106                 ),
107               ],
108             ),
109           ),
110           padding: EdgeInsets.all(15),
111         ),
112       ),
113     );
114   }
115 }
116
```

El agregado de la descripción del artefacto junto con su condición en caso este campo este vacío

El agregado del modelo del artefacto junto con su condición en caso este campo este vacío

El agregado del precio del artefacto junto con su condición en caso este campo este vacío

La acción del Botón Guardar

Luego tenemos el dar llamado listado el cual como su propio nombre lo dice se trata de listar nuestros datos

```
1 import 'package:ep02ejercicio02tokumotomora/db.dart';
2 import 'package:ep02ejercicio02tokumotomora/artefacto.dart';
3 import 'package:flutter/material.dart';
4
5 class Listado extends StatelessWidget {
6   @override
7   Widget build(BuildContext context) {
8     return Scaffold(
9       appBar: AppBar(
10        title: Text(
11          "Artefactos Electrónicos",
12          style: TextStyle(color: Colors.white), // Texto blanco
13        ),
14        backgroundColor: Colors.blue, // Color de fondo azul
15      ),
16      floatingActionButton: FloatingActionButton(
17        child: Icon(Icons.add),
18        onPressed: () {
19          Navigator.pushNamed(
20            context,
21            "/editar",
22            arguments: Artefacto(id: 0, descripcion: "", modelo: "", precio: ""),
23          );
24        },
25        backgroundColor: Colors.blue, // Color de fondo azul
26      ),
27      body: Container(
28        child: Lista(),
29      ),
30    );
31  }
32 }
33
34 class Lista extends StatefulWidget {
35   @override
36   _MiLista createState() => _MiLista();
37 }
38
39 class _MiLista extends State<Lista> {
40   List<Artefacto> artefacto = [];
41
42   @override
43   void initState() {
44     cargarArtefactos();
45     super.initState();
46   }
47
48   cargarArtefactos() async {
49     List<Artefacto> auxArtefacto = await DB.artefacto();
50     setState(() {
51       artefacto = auxArtefacto;
52     });
53   }
54
55   @override
56   Widget build(BuildContext context) {
57     return ListView.builder(
58       itemCount: artefacto.length,
59       itemBuilder: (context, i) => Dismissible(
60         key: Key(i.toString()),
61         direction: DismissDirection.startToEnd,
62         background: Container(
63           color: Colors.red,
64           padding: EdgeInsets.only(left: 5),
65           child: Align(
66             alignment: Alignment.centerLeft,
67             child: Icon(Icons.delete, color: Colors.white),
68           ),
69         ),
70         onDismissed: (direction) {
71           DB.delete(artefacto[i]);
72         },
73         child: ListTile(
74           title: Text(
75             artefacto[i].descripcion,
76             style: TextStyle(color: Colors.black), // Texto negro
77           ),
78           trailing: MaterialButton(
79             onPressed: () {
80               Navigator.pushNamed(context, "/editar", arguments: artefacto[i]);
81             },
82             child: Icon(Icons.edit),
83             color: Colors.grey[400], // Fondo gris claro
84           ),
85         ),
86       ),
87     );
88   }
89 }
90
```

Titulo

Botón Agregar

Visualización de la lista artefacto |

Botón editar

Finalmente tenemos el main.dart en donde colocaremos nuestra lista como visualización predeterminada cada vez que abrimos el aplicativo

```
1 import 'package:ep02ejercicio02tokumotomora/editar.dart';
2 import 'package:ep02ejercicio02tokumotomora/listado.dart';
3 import 'package:flutter/material.dart';
4
5 void main() {
6   runApp(MyApp());
7 }
8
9 class MyApp extends StatelessWidget {
10   @override
11   Widget build(BuildContext context) {
12     return MaterialApp(
13       title: 'SQLite Demoo',
14       theme: ThemeData(
15         primarySwatch: Colors.blue,
16       ),
17       home: MiApp(),
18     );
19   }
20 }
21
22 class MiApp extends StatelessWidget {
23   @override
24   Widget build(BuildContext context) {
25     return MaterialApp(initialRoute: "/", routes: {
26       "/": (context) => Listado(),
27       "/editar": (context) => Editar()
28     });
29   }
30 }
```

La primera ruta
que se abrirá
será la del listado

Finalmente mostraremos el siguiente ejemplo

Descripción: Televisor Led Smart TV FHD 43
Modelo : LT-43KB315
Precio : S/ 1079

Descripción: TELEVISOR LG 65 4K UHD THINQ AI SMART 65
Modelo : UR7300PSA
Precio : S/ 1729

ep02ejercicio02tokumotomora

← Guardar

Back

Descripción

Televisor Led Smart TV FHD 43

Modelo

LT-43KB315

Precio

1079

Guardar

ep02ejercicio02tokumotomora

← Guardar

Descripción

TELEVISOR LG 65 4K UHD THINQ AI SMART 65

Modelo

UR7300PSA

Precio

1729

Guardar

ep02ejercicio02tokumotomora

← Artefactos Electrónicos

Televisor Led Smart TV FHD 43

TELEVISOR LG 65 4K UHD THINQ AI SMART 65

+