# 📚 Guía Definitiva para Crear una App de Inventario con Flutter y Google Sheets (UI Moderna)

Esta guía detalla paso a paso cómo implementar una app de inventario con una interfaz moderna usando Flutter y Google Sheets a través de Apps Script.

## 📖 Requisitos del Sistema

### Flutter SDK y herramientas necesarias:

* Flutter: 3.22.0
* Dart SDK: 3.2.0
* Android SDK: 34
* NDK: 29.0.13599879
* Android Studio: Hedgehog | Electric Eel o superior

## 🔄 Paso 1: Crear la Hoja de Google Sheets

1. Abre [Google Sheets](https://sheets.google.com).
2. Crea una hoja nueva llamada Inventario.
3. En la primera hoja (renómbrala a Inventario), agrega los siguientes encabezados en la fila 1:

ID | Modelo | Marca | Color | Cantidad

## ✨ Paso 2: Crear el Backend con Apps Script

1. En tu hoja de Google Sheets, ve a Extensiones > Apps Script.
2. Borra el contenido existente y pega el siguiente código, reemplazando “TU\_ID” por la Id de la hoja de Google (el valor que aparece en la URL entre /d/ y /edit):

function doPost(e) {  
 try {  
 const ss = SpreadsheetApp.openById("TU\_ID");  
 const sheet = ss.getSheetByName("Inventario");  
 const data = JSON.parse(e.postData.contents);  
 const action = data.action;  
  
 if (!sheet) {  
 return ContentService.createTextOutput(JSON.stringify({ status: "error", message: "Hoja no encontrada" }))  
 .setMimeType(ContentService.MimeType.JSON);  
 }  
  
 if (action === "addItem") {  
 const { modelo, marca, color } = data;  
 const id = (modelo[0] + marca[0] + color[0]).toUpperCase();  
 sheet.appendRow([id, modelo, marca, color, 1]);  
  
 return ContentService.createTextOutput(JSON.stringify({ status: "ok", message: "✅ Item agregado: " + id }))  
 .setMimeType(ContentService.MimeType.JSON);  
 }  
  
 if (action === "add" || action === "remove") {  
 const code = data.code;  
 const values = sheet.getDataRange().getValues();  
  
 for (let i = 1; i < values.length; i++) {  
 if (values[i][0] === code) {  
 const qty = values[i][4];  
 const newQty = action === "add" ? qty + 1 : Math.max(0, qty - 1);  
 sheet.getRange(i + 1, 5).setValue(newQty);  
  
 return ContentService.createTextOutput(JSON.stringify({  
 status: "ok",  
 message: `${action === "add" ? "📥 Ingreso" : "📤 Salida"} registrada para ${code}. Nueva cantidad: ${newQty}`  
 })).setMimeType(ContentService.MimeType.JSON);  
 }  
 }  
  
 return ContentService.createTextOutput(JSON.stringify({ status: "error", message: "⚠️ Código no encontrado" }))  
 .setMimeType(ContentService.MimeType.JSON);  
 }  
  
 return ContentService.createTextOutput(JSON.stringify({ status: "error", message: "❓ Acción no reconocida" }))  
 .setMimeType(ContentService.MimeType.JSON);  
  
 } catch (error) {  
 return ContentService.createTextOutput(JSON.stringify({ status: "error", message: "❌ Error interno" }))  
 .setMimeType(ContentService.MimeType.JSON);  
 }  
}

1. Haz clic en Deploy > Manage deployments > New deployment.
   * Tipo: Web app
   * Ejecutar como: Me
   * Quién tiene acceso: Anyone
2. Copia la URL generada. Es tu scriptURL para Flutter.

## 📲 Paso 3: Crear la App en Flutter

### 3.1 Agrega las dependencias en pubspec.yaml

Ruta: tu\_proyecto/pubspec.yaml

dependencies:  
 flutter:  
 sdk: flutter  
 http: ^0.13.5  
 mobile\_scanner: ^3.2.0

### 3.2 Agrega permisos de cámara

Ruta: android/app/src/main/AndroidManifest.xml

Agrega dentro de <manifest>:

<uses-permission android:name="android.permission.CAMERA" />

<uses-feature android:name="android.hardware.camera" android:required="true"/>

### 3.2 Especifica la version de NDK

Ruta: android/app/build.gradle.kts

Para no tener conflictos con las librerías hay que usar otra version, especificamos laque está instalada:

ndkVersion = “29.0.13599879”

## 📂 Paso 4: Crea el archivo main.dart

Ruta: lib/main.dart

Recuerda colocarla URL generada por AppsScripts en “TU\_URL”.

import 'dart:convert';  
import 'package:flutter/material.dart';  
import 'package:http/http.dart' as http;  
import 'package:mobile\_scanner/mobile\_scanner.dart';  
  
void main() {  
 runApp(MaterialApp(  
 debugShowCheckedModeBanner: false,  
 home: InventoryApp(),  
 theme: ThemeData.dark(useMaterial3: true).copyWith(  
 colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),  
 ),  
 ));  
}  
  
class InventoryApp extends StatefulWidget {  
 @override  
 \_InventoryAppState createState() => \_InventoryAppState();  
}  
  
class \_InventoryAppState extends State<InventoryApp> {  
 final TextEditingController \_modeloController = TextEditingController();  
 final TextEditingController \_marcaController = TextEditingController();  
 final TextEditingController \_colorController = TextEditingController();  
 String \_selectedAction = "add";  
  
 final String scriptURL = "TU\_URL";  
  
 Future<void> addItem() async {  
 final modelo = \_modeloController.text.trim();  
 final marca = \_marcaController.text.trim();  
 final color = \_colorController.text.trim();  
  
 final body = jsonEncode({  
 "action": "addItem",  
 "modelo": modelo,  
 "marca": marca,  
 "color": color,  
 });  
  
 try {  
 final response = await http.post(  
 Uri.parse(scriptURL),  
 headers: {"Content-Type": "application/json"},  
 body: body,  
 );  
  
 final bodyText = response.body.contains("Moved Temporarily")  
 ? "🔁 Redireccionamiento activado"  
 : response.body;  
  
 print("📅 Respuesta final:\n\$bodyText");  
  
 if (mounted) {  
 ScaffoldMessenger.of(context).showSnackBar(  
 SnackBar(content: Text("✅ Item agregado")),  
 );  
 }  
 } catch (e) {  
 print("❌ Error al enviar datos: \$e");  
 }  
 }  
  
 Future<void> scanCodeAndSend(String action) async {  
 await Navigator.push(  
 context,  
 MaterialPageRoute(  
 builder: (context) => QRScannerScreen(  
 scriptURL: scriptURL,  
 action: action,  
 ),  
 ),  
 );  
 }  
  
 InputDecoration \_customInputDecoration(String label) {  
 return InputDecoration(  
 labelText: label,  
 filled: true,  
 fillColor: Colors.deepPurple.withOpacity(0.1),  
 border: OutlineInputBorder(borderRadius: BorderRadius.circular(12)),  
 enabledBorder: OutlineInputBorder(  
 borderSide: BorderSide(color: Colors.deepPurple.shade200),  
 borderRadius: BorderRadius.circular(12),  
 ),  
 );  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 title: Text("📦 Control de Inventario"),  
 centerTitle: true,  
 ),  
 body: SingleChildScrollView(  
 padding: const EdgeInsets.all(20),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 Text("📝 Registrar un producto", style: Theme.of(context).textTheme.headlineSmall),  
 const SizedBox(height: 16),  
 TextField(  
 controller: \_modeloController,  
 decoration: \_customInputDecoration("Modelo"),  
 ),  
 const SizedBox(height: 12),  
 TextField(  
 controller: \_marcaController,  
 decoration: \_customInputDecoration("Marca"),  
 ),  
 const SizedBox(height: 12),  
 TextField(  
 controller: \_colorController,  
 decoration: \_customInputDecoration("Color"),  
 ),  
 const SizedBox(height: 16),  
 Text("¿Qué acción deseas realizar?", style: Theme.of(context).textTheme.labelLarge),  
 const SizedBox(height: 8),  
 Container(  
 padding: EdgeInsets.symmetric(horizontal: 12),  
 decoration: BoxDecoration(  
 color: Colors.deepPurple.withOpacity(0.1),  
 borderRadius: BorderRadius.circular(12),  
 border: Border.all(color: Colors.deepPurple.shade200),  
 ),  
 child: DropdownButtonHideUnderline(  
 child: DropdownButton<String>(  
 isExpanded: true,  
 value: \_selectedAction,  
 items: const [  
 DropdownMenuItem(value: "add", child: Text("📥 Ingresar")),  
 DropdownMenuItem(value: "remove", child: Text("📤 Retirar")),  
 ],  
 onChanged: (value) {  
 if (value != null) {  
 setState(() {  
 \_selectedAction = value;  
 });  
 }  
 },  
 ),  
 ),  
 ),  
 const SizedBox(height: 24),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.spaceEvenly,  
 children: [  
 ElevatedButton.icon(  
 style: ElevatedButton.styleFrom(  
 padding: const EdgeInsets.symmetric(horizontal: 20, vertical: 12),  
 backgroundColor: Colors.deepPurple,  
 shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(12)),  
 ),  
 onPressed: addItem,  
 icon: Icon(Icons.add\_box),  
 label: Text("Agregar"),  
 ),  
 ElevatedButton.icon(  
 style: ElevatedButton.styleFrom(  
 padding: const EdgeInsets.symmetric(horizontal: 20, vertical: 12),  
 backgroundColor: Colors.purple.shade700,  
 shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(12)),  
 ),  
 onPressed: () => scanCodeAndSend(\_selectedAction),  
 icon: Icon(Icons.qr\_code\_scanner),  
 label: Text("Escanear"),  
 ),  
 ],  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}  
  
class QRScannerScreen extends StatefulWidget {  
 final String scriptURL;  
 final String action;  
  
 QRScannerScreen({required this.scriptURL, required this.action});  
  
 @override  
 State<QRScannerScreen> createState() => \_QRScannerScreenState();  
}  
  
class \_QRScannerScreenState extends State<QRScannerScreen> {  
 final MobileScannerController controller = MobileScannerController();  
 bool \_scanned = false;  
  
 @override  
 void dispose() {  
 controller.dispose();  
 super.dispose();  
 }  
  
 void \_handleScan(String code) async {  
 if (\_scanned) return;  
 \_scanned = true;  
  
 await controller.stop();  
  
 final body = jsonEncode({  
 "action": widget.action,  
 "code": code,  
 });  
  
 try {  
 final response = await http.post(  
 Uri.parse(widget.scriptURL),  
 headers: {"Content-Type": "application/json"},  
 body: body,  
 );  
  
 final isHtml = response.body.trim().toLowerCase().startsWith("<!doctype html") ||  
 response.body.trim().toLowerCase().startsWith("<html");  
  
 final mensaje = isHtml ? "🔁 Redireccionamiento activado" : response.body;  
  
 if (mounted) {  
 print("🔄 Código de estado: \${response.statusCode}");  
 print("📅 Respuesta final:\n\$mensaje");  
 Navigator.pop(context);  
 }  
 } catch (e) {  
 if (mounted) {  
 print("❌ Error: \$e");  
 Navigator.pop(context);  
 }  
 }  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(title: Text("Escanear QR")),  
 body: MobileScanner(  
 controller: controller,  
 onDetect: (capture) {  
 final barcode = capture.barcodes.first;  
 final String? code = barcode.rawValue;  
 if (code != null) {  
 \_handleScan(code);  
 }  
 },  
 ),  
 );  
 }  
}

## 🚀 Resultado Final

* UI moderna, intuitiva y responsiva.
* Agregado de productos con feedback visual.
* Escaneo de QR para ingreso o salida.
* Datos almacenados de forma confiable en Google Sheets.
* Recuerda que a veces hay que reiniciar varias veces el lanzamiento de la primera app o la implementación de Apps Script.

🌟 Proyecto listo para ser usado o expandido. ¿Te gustaría añadir una pantalla de historial o resumen de inventario? Puedo ayudarte.