



ESCUELA UNIVERSITARIA TECNOLÓGICO DE ESTUDIOS SUPERIORES DE ECATEPEC

SUAREZ HERNANDEZ ALEXIS EDUARDO

NOMBRE DEL PROFESOR:

Leonardo Miguel Moreno Villalba

GRUPO: 15-701

*MATERIA: DESARROLLO DE
APLICACIONES MÓVILES*

PRIMER PARCIAL

1. ¿Cómo organizaste la navegación para que el Hub, el índice de prácticas y el proyecto se integraran sin duplicar código?

Estrategia general:

Para evitar duplicación de código y mantener la navegación ordenada, se estructuró la app en **capas modulares**, separando cada sección en widgets independientes:

- **Hub principal:** pantalla de bienvenida con acceso al índice de prácticas y al proyecto final. Funciona como “punto central” de la app.
- **Índice de prácticas:** una **lista dinámica** generada a partir de un arreglo de datos (`List<Map<String, dynamic>>`), donde cada elemento tiene: título de la práctica, descripción breve y la ruta de navegación correspondiente.
- **Proyecto final:** widget autónomo que contiene la lógica específica de la aplicación final (ej. calculadora de IMC o aplicación integrada).

Decisiones concretas:

- Uso de **rutas nombradas** (`MaterialApp.routes`):
- `routes: {`
- `'/hub': (context) => HubScreen(),`
- `'/practical': (context) => PracticalScreen(),`

- `'/proyecto': (context) => ProyectoScreen(),`
- `}`

Esto centraliza la navegación y permite cambiar rutas sin modificar cada botón individual.

- **Reutilización de widgets:**
 - El índice de prácticas usa un `ListView.builder`, pasando los datos de cada práctica como parámetro.
 - Los botones de navegación dentro de cada práctica llaman al `Navigator.pushNamed` con la ruta específica.
- **Drawer y menú flotante:**
 - Se añadió un `Drawer` con enlaces rápidos a todas las secciones, evitando repetir botones en cada práctica.
 - Esto asegura que el usuario pueda cambiar de sección sin reescribir la lógica de navegación.

Beneficio: cualquier cambio en la ruta, en los títulos o en la estructura de prácticas se hace en un solo archivo, reduciendo errores y duplicación.

2. ¿Qué decisiones de UI tomaste para mantener consistencia entre prácticas y proyecto?

Enfoque visual:

El objetivo fue que todas las prácticas y el proyecto final compartieran una experiencia **coherente y profesional**:

- **Tema global (`ThemeData`):**
 - Colores: principal, secundario y de acento uniformes en toda la app.
 - Tipografía: misma fuente y tamaño base para títulos, subtítulos y texto de formularios.
 - Botones: mismos estilos (`ElevatedButton.styleFrom`) para mantener consistencia en interacción.
- **Componentes reutilizables:**
 - `CustomCard` para mostrar información de cada práctica.
 - `CustomButton` para acciones comunes, como enviar datos o navegar.
 - Formularios: `CustomTextField` con validación integrada.
- **Espaciado y layout:**
 - Uso de `Padding` y `SizedBox` consistentes entre prácticas y proyecto.
 - Mantener alineación de elementos (títulos arriba, formulario en el centro, botón al final).

Beneficio: el usuario percibe una **experiencia uniforme**, sin saltos visuales ni inconsistencias entre prácticas y proyecto.

3. ¿Qué límites encontraste al trabajar solo con estado en memoria y cómo los sorteaste?

Problema:

- Los datos se mantienen solo mientras la app está abierta.
- Al navegar entre pantallas o cerrar la app, todo se pierde.
- Compartir información entre prácticas o con el proyecto final requiere pasar parámetros explícitamente, lo que aumenta la complejidad si hay muchas prácticas.

Soluciones implementadas:

1. **Propagación de parámetros y callbacks:**
 - Cada práctica recibe y devuelve datos mediante constructores y funciones callback.
 - Ejemplo: pasar el resultado de una práctica al proyecto final.
2. **StatefulWidgets y setState para actualizar UI:**
 - Se mantiene la coherencia visual mientras la app está abierta.
 - Permite cambios dinámicos sin recargar la pantalla completa.
3. **Uso opcional de ChangeNotifier para estados globales pequeños:**
 - Aunque todo sigue en memoria, centraliza cambios de estado, por ejemplo, seleccionar prácticas completadas.

Límite principal: sin persistencia, la continuidad educativa o profesional del usuario queda limitada.

4. ¿Qué validaciones aplicaste en la calculadora de IMC y por qué?

Objetivo: garantizar resultados precisos y evitar errores de entrada.

- **Campos vacíos:** no se permiten, se alerta al usuario mediante `SnackBar`.
- **Datos no numéricos:** se validan con `double.tryParse` para asegurar que el cálculo funcione.
- **Rangos lógicos:**
 - Peso: >0 y <500 kg
 - Altura: >0 y <3 metros
 - Evita valores extremos que distorsionen el IMC.

Implementación:

```
if (peso <= 0 || peso > 500 || altura <= 0 || altura > 3) {  
  ScaffoldMessenger.of(context).showSnackBar(  
    SnackBar(content: Text('Ingresa valores válidos'))  
  );  
}
```

Razón: asegura **confianza y profesionalismo** en la app, evitando resultados absurdos.

5. ¿Cómo garantizaste retroalimentación inmediata al usuario (SnackBar/Diálogos) sin recargar la interfaz?

Cómo se implementó:

- **SnackBar:** para errores de validación o confirmaciones rápidas.
- **AlertDialogs:** para mostrar resultados importantes o decisiones críticas.
- **No se recarga la UI:** gracias a `setState` y al manejo local del estado, la pantalla no se reinicia.
- **Ejemplo:** al calcular IMC, el resultado se muestra en un diálogo modal y el formulario sigue activo para nuevas entradas.

Beneficio: interacción fluida y sin interrupciones, mejorando la experiencia del usuario y la percepción de profesionalismo.

6. Si pudieras persistir datos, ¿qué mejorarías primero y con qué enfoque (sin implementarlo)?

Si se implementara almacenamiento:

- **Prioridad:**
 1. Historial de cálculos de IMC
 2. Prácticas completadas y progreso
 3. Configuración del usuario (tema, preferencias)
- **Enfoque recomendado:**
 - Datos simples: `SharedPreferences`
 - Datos complejos o estructurados: `Hive` o `SQLite`
 - Mantener separación de lógica: un repositorio de datos independiente de la UI para futuras escalas.

Beneficio esperado:

- Continuidad educativa: el usuario puede retomar prácticas y revisar resultados históricos.
- Mejora de experiencia: la app pasa de ser un prototipo a una herramienta profesional y persistente.



Hub Principal



Prácticas



Proyecto







Ajustes



Acerca

7:00

Menú de Navegación

-  Inicio
-  Índice de Prácticas
-  Proyecto (Kit Offline)
-  Ajustes / Acerca de

7:02



Índice de Prácticas

1

Práctica 1



2

Práctica 2



3

Práctica 4 (Registro)



4

Juego: RPS



7:02



Proyecto - Kit Offline



Notas rápidas

Agregar notas en memoria



Calculadora IMC

Form con validaciones



Galería local

Ver imágenes de assets



Juego: Par o Impar

Marcador en memoria

Android Emulator - Medium_Phone_API_36.0:5554

7:02



← Galería local





Par o Impar


Elige Par o Impar:

Elige un número (0-5):

Marcador — Tú: 2 CPU: 1

Reiniciar marcador

Tu 2 + CPU 0 = 2 → Par • Ganaste

7:03  



Ajustes

Tema oscuro

Alterna entre claro/oscuro (solo memoria)



Acerca de

Versión 1.0 • Autor: Alexis Eduardo Suarez
Hernandez