LABORATORIO	10 – MENÚ DE NAVEGACIÓN INFERIOR
OBJETIVO GENERAL	Elabora menús de navegación
INSTRUCCIONES	Elabora tu propia versión de un navegador personalizado
NOMBRE ESTUDIANTE	Itsai Zempoaltecatl Mejia
FECHA	24/marzo/2022
GRUPO	8SA

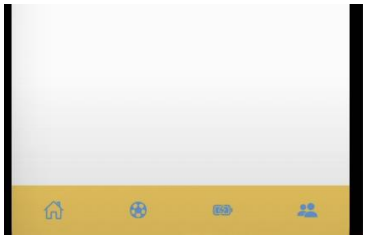
RECURSOS

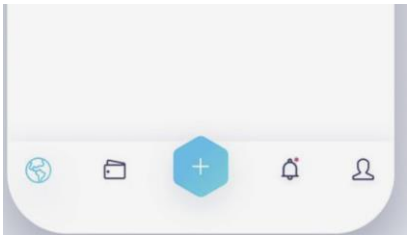
Puedes utilizar el recurso que te comparto para la creación de tu cronómetro

- https://docs.flutter.dev/cookbook/navigation
- https://youtu.be/_RY1ibJjAG0

Puedes usar cualquier otro medio para el desarrollo de tu práctica

Al finalizar tu aplicación deberás hacer énfazis en la explicación de la navegación por rutas y su configuración dentro de las aplicaciones móviles.





Demostrar que puedes navegar en cada pantalla (puedes usar un color para distinguir la navegación)

Puntos a entregar:

- Entregar video de 5 minutos como máximo con **formato MP4 (si es otro formato no se revisa)**, donde expliques como implementaste tu aplicación **detalladamente**, deberás **mostrar el código durante la explicación así como la aplicación desarrollada.**
- La aplicación deberá correr en **un emulador o dispositivo físico**, la grabación se debe apreciar correctamente
- Entregar reporte escrito en formato PDF

Elementos a calificar

- Reporte 10 pts
- Funcionamiento app 10 pts
- Diseño de la app 10pts
- Explicación del video 10pts, [la explicación deberá ser detallada, en caso de no dominar la explicación, la aplicación no se tomará en cuenta]

Te resta puntos 👇🏻

- Usar widgets que no explicas en tu video-10pts
- Usar widgets deprecate (desactualizados) -10pts

- Si la aplicación presenta al menos 1 warning -10pts
  - 

Ilustración 1 Ejemplo de warning

INTRODUCCIÓN

Para navegar entre pantallas tenemos la opción de Navigator.push(…), esto es útil cuando un botón único mostrara una única pantalla, pero en ocasiones necesitamos crear diferentes botones que manden a llamar a diferentes pantallas, para esto podemos usar algo llamado routes (rutas), este es una propiedad de MaterialApp, esta nos define las rutas con nombres disponibles y los widgets para construir al navegar a estas rutas.

```
routes: {
  // When navigating to the "/" route, build the FirstScreen widget.
  '/': (context) => const FirstScreen(),
  // When navigating to the "/second" route, build the SecondScreen widget.
  '/second': (context) => const SecondScreen(),
},
```

Ejemplo

Una vez definidas estas rutas, para acceder a ellas usamos Navigator.pushNamed(context, routeName), donde routeName debe ser el nombre de la ruta.

```
// Navigate to the second screen using a named route.
Navigator.pushNamed(context, '/second');
```

Tomando el ejemplo anterior, si queremos acceder a SecondScreen, basta con poner su nombre de ruta ‘/second’ y de esta manera nos construirá SecondScreen()

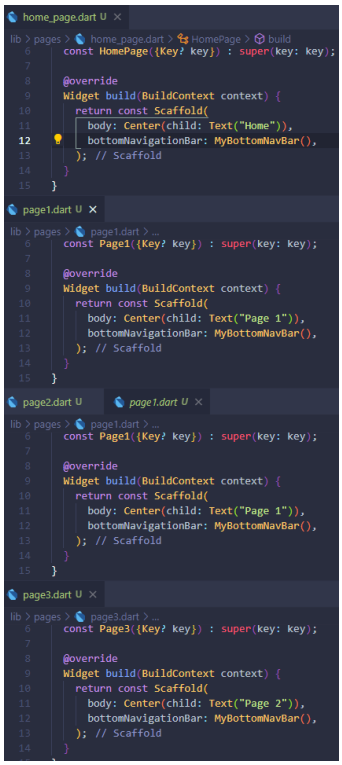
Scaffold tiene una propiedad llamada bottomNavigationBar, que como su nombre indica, nos provee de un espacio para la creación de un NavigationBar en la parte inferior de la pantalla. Un NavigationBar es un componente que consta de diferentes opciones para cambiar entre pantallas en diferentes tipos de aplicaciones. Para este ejemplo usaremos este widget.

DESARROLLO

```dart
Widget build(BuildContext context) {
  return MaterialApp(
    routes: {
      '/': (context) => const HomePage(),
      '/one': (context) => const Page1(),
      '/two': (context) => const Page2(),
      '/three': (context) => const Page3()
    },
  ); // MaterialApp
}
```

Main

Defino los nombres de las rutas y las pantallas a construir de las mismas en mi MaterialApp (‘/’ se considera como mi propiedad home)



HomePage, Page1, Page2, Page3

Cada una de estas pantallas no cuentan con gran contenido, pero una propiedad de Scaffold si es de suma importancia; bottomNavigationBar que contiene un widget llamado MyBottomNavBar()

```dart
Widget build(BuildContext context) {
  return Container(
    padding: const EdgeInsets.only(top: 10, bottom: 10),
    color: Colors.blue[100],
    child: SafeArea(
      child: Row(
        children: const [
          MyBottomNavBarItem(
            iconData: Icons.home_filled,
            label: "Home",
            routeName: '/',
          ), // MyBottomNavBarItem
          MyBottomNavBarItem(
            iconData: Icons.looks_one_outlined,
            label: "Page 1",
            routeName: '/one',
          ), // MyBottomNavBarItem
          MyBottomNavBarItem(
            iconData: Icons.looks_two_outlined,
            label: "Page 2",
            routeName: '/two',
          ), // MyBottomNavBarItem
          MyBottomNavBarItem(
            iconData: Icons.theater_comedy_outlined,
            label: "Page 3",
            routeName: '/three',
          ), // MyBottomNavBarItem
        ],
      ), // Row
    ), // SafeArea
  ); // Container
```

MyBottomNavBar

En este widget se encuentras las diferentes opciones que contiene mi bottomNavigationBar, cada componente está contenido en un Row para que el acomodo sea de manera horizontal y de izquierda a derecha. Como podemos ver los widgets que contiene son MyBottomNavBarItem el cual contiene 3 propiedades.

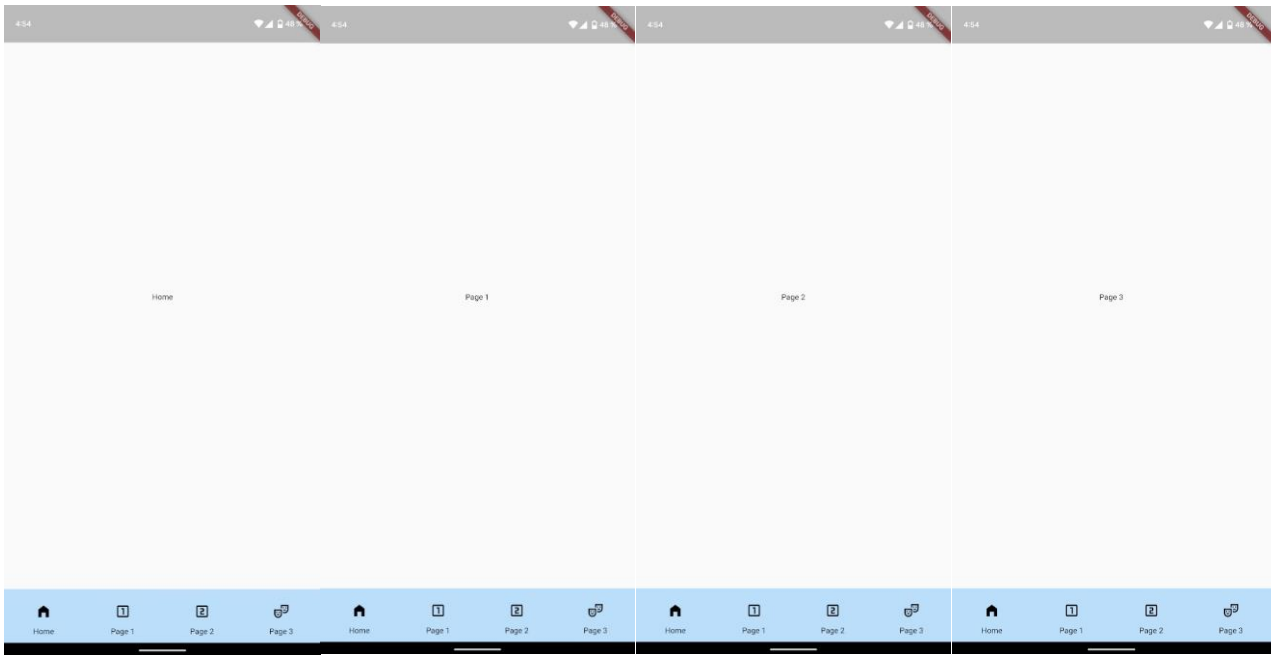
```dart
class MyBottomNavBarItem extends StatelessWidget {
  final IconData iconData;
  final String label;
  final String routeName;
  const MyBottomNavBarItem(
      {Key? key,
      required this.iconData,
      required this.label,
      required this.routeName})
      : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Expanded(
        child: Column(
      mainAxisSize: MainAxisSize.min,
      children: [
        TextButton(
            onPressed: () {
              Navigator.pop(context);
              Navigator.pushNamed(context, routeName);
            },
            child: Icon(
              iconData,
              color: Colors.black,
            )), // Icon // TextButton
        Text(label)
      ],
    )); // Column // Expanded
  }
}
```

MyBottomNavBarItem

Las propiedades de esta clase no son opcionales, ya que es necesario para la construcción del widget, como primer punto tenemos *iconData*, esta propiedad recibirá un Icon que será puesto en el widget para que represente visualmente su función, *label* es un string que indicara explícitamente hacia que pantalla nos dirigirá este widget y por ultimo *routeName,* esta propiedad es básicamente el nombre de la pantalla que definimos con anterioridad en routes, así que será utilizado el valor de esta para ejecutar Navigator.pushNamed.

El widget esta conformado por un Expanded para que use todo el espacio disponible, Column nos servirá para poner apilados los widgets TextButton que tiene un Icon el mismo que tomara el Icono de la propiedad iconData y Text que tomara el valor de label. Dentro de TextButton en su propiedad onPressed tenemos a Navigator.pop encargado de eliminar la pantalla previa (para no apilar pantallas) y Navigator.pushNamed(…), que construirá la pantalla de la ruta que recibió.

Resultado



Pantallas

CONCLUSIONES

La manera en la que se creó la aplicación deja con un resultado no tan dinámico, ya que Navigator.pushNamed crea pantallas nuevas, mas no las agrega en una misma, es por esto que da una percepción que la aplicación se esta creando una y otra vez al presionar los botones.

REFERENCIAS

Xenia Padilla. [Xenia Padilla] (2021). 20 FLUTTER BOTTONNAVIGATIONBAR [Video]. Youtube. https://www.youtube.com/watch?v=_RY1ibJjAG0&t=252s

Xenia Padilla. [Xenia Padilla] (2021). 21 FLUTTER CREANDO MIS WIDGETS PARTE 1 [Video]. Youtube. https://www.youtube.com/watch?v=NjhTKa5LF8Q

Xenia Padilla. [Xenia Padilla] (fecha). 22 FLUTTER CREANDO MIS WIDGETS PARTE 2 [Video]. Youtube. https://www.youtube.com/watch?v=_CouypzU_cQ&t=107s

Flutter. (2022). Navigate with named routes. Flutter. https://docs.flutter.dev/cookbook/navigation/named-routes

Flutter. (2022). BottomNavigationBar class. Flutter. https://api.flutter.dev/flutter/material/BottomNavigationBar-class.html