W3-S2 *PRACTICE*

MANIPULATE BASIC WIDGETS

Learning objectives

- ✓ Start from an **empty Flutter project**
- ✓ **Hot Reload**
- ✓ Use Flutter **doctor**, **update**, create, run
- ✓ Use Flutter **Documentation**

- ✓ Manipulate **Scaffold**, Text, TextStyle, Radius
- ✓ Manipulate Colors palettes
- ✓ Manipulate **Container**, BoxDecoration, Center, EdgeInsets, Column
- ✓ Create a **Custom stateless** widget

No AI tools allowed to solve this practice

How to submit?

- ✓ **Push** your final code on **your GitHub repository**
- ✓ Then **attach the GitHub path** to the MS Team assignment and **turn it in**

Are you lost?

Read the following documentation to be ready for this practice:

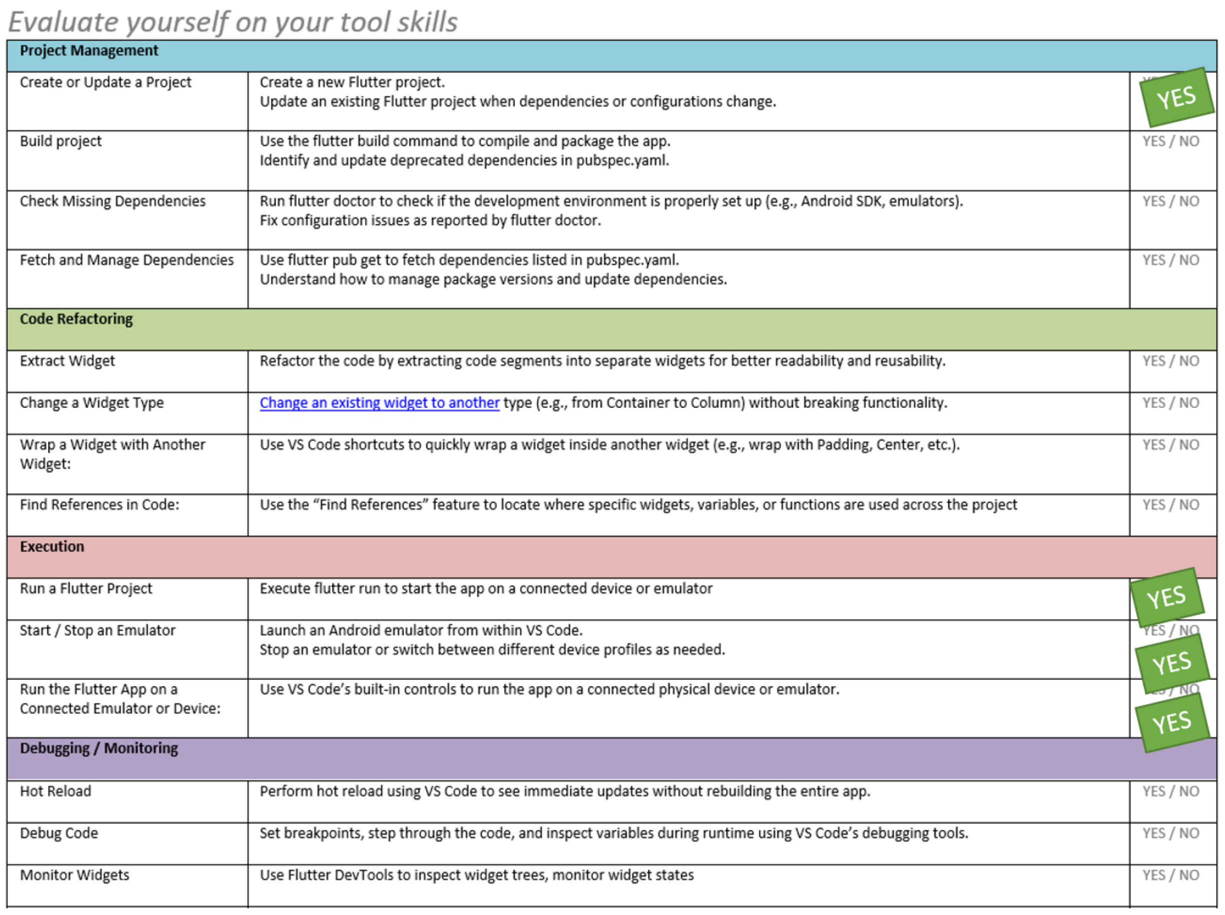
https://api.flutter.dev/flutter/material/Scaffold-class.html
https://docs.flutter.dev/ui/widgets/text
https://api.flutter.dev/flutter/widgets/Center-class.html
https://api.flutter.dev/flutter/painting/EdgeInsets-class.html
https://api.flutter.dev/flutter/widgets/Column-class.html
https://api.flutter.dev/flutter/widgets/Container-class.html
https://api.flutter.dev/flutter/painting/BoxDecoration-class.html

BEFORE THIS PRACTICE

Where are you in your Tools skills?

Before the practice we expect you to **be able to run the default Flutter Project** using an android Device

If you cannot perform it, ask your group teammate to support you before the practice day!



Evaluate yourself on your tool skills

Project Management		
Create or Update a Project	Create a new Flutter project. Update an existing Flutter project when dependencies or configurations change.	[✓ YES]
Build project	Use the flutter build command to compile and package the app. Identify and update deprecated dependencies in pubspec.yaml.	YES / NO
Check Missing Dependencies	Run flutter doctor to check if the development environment is properly set up (e.g., Android SDK, emulators). Fix configuration issues as reported by flutter doctor.	YES / NO
Fetch and Manage Dependencies	Use flutter pub get to fetch dependencies listed in pubspec.yaml. Understand how to manage package versions and update dependencies.	YES / NO
Code Refactoring		
Extract Widget	Refactor the code by extracting code segments into separate widgets for better readability and reusability.	YES / NO
Change a Widget Type	Change an existing widget to another type (e.g., from Container to Column) without breaking functionality.	YES / NO
Wrap a Widget with Another Widget:	Use VS Code shortcuts to quickly wrap a widget inside another widget (e.g., wrap with Padding, Center, etc.).	YES / NO
Find References in Code:	Use the "Find References" feature to locate where specific widgets, variables, or functions are used across the project	YES / NO
Execution		
Run a Flutter Project	Execute flutter run to start the app on a connected device or emulator	[✓ YES]
Start / Stop an Emulator	Launch an Android emulator from within VS Code. Stop an emulator or switch between different device profiles as needed.	[✓ YES] / NO
Run the Flutter App on a Connected Emulator or Device:	Use VS Code's built-in controls to run the app on a connected physical device or emulator.	[✓ YES] / NO
Debugging / Monitoring		
Hot Reload	Perform hot reload using VS Code to see immediate updates without rebuilding the entire app.	YES / NO
Debug Code	Set breakpoints, step through the code, and inspect variables during runtime using VS Code's debugging tools.	YES / NO
Monitor Widgets	Use Flutter DevTools to inspect widget trees, monitor widget states	YES / NO

EX 1 – Scaffold, Text

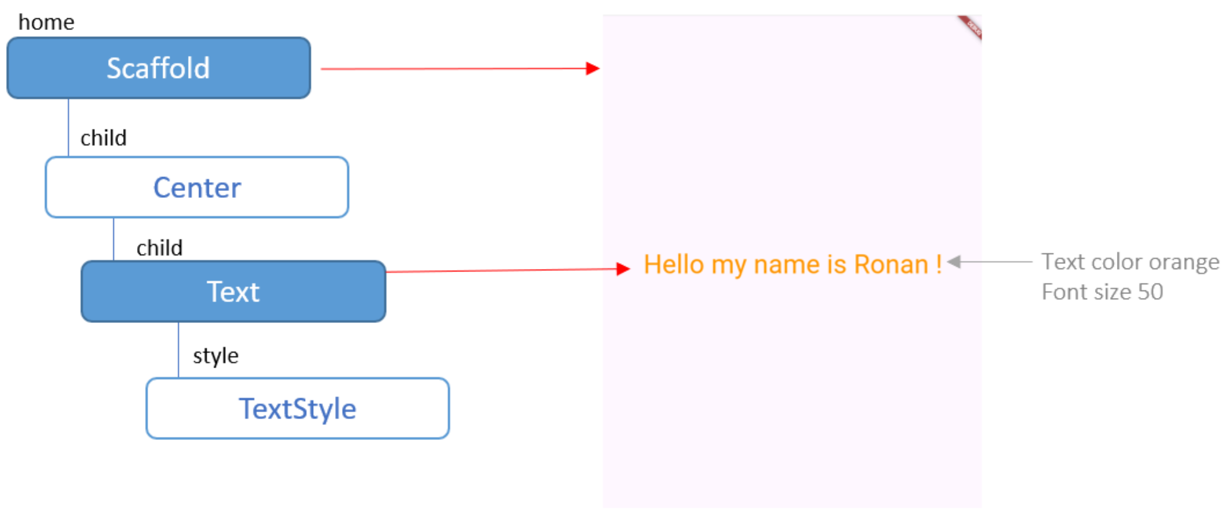
In this exercise, you will be working with the following widgets

- **Scaffold:** Provides a basic structure for UI, like app bar, drawer, etc.
- **Center**: Aligns a child widget to the center
- **Text**: Displays text.

And the following classes
- **TextStyle**: Styles text appearance.

You need to produce the following mockup and widget structure:

You are free to customize it!



<u>To start</u>
Open `W3-S2/EX-1/main.dart`

```
void main() {
  runApp(
    const MaterialApp( ),
  );
}
```

EX 2 – Container, Insets, BoxDecoration

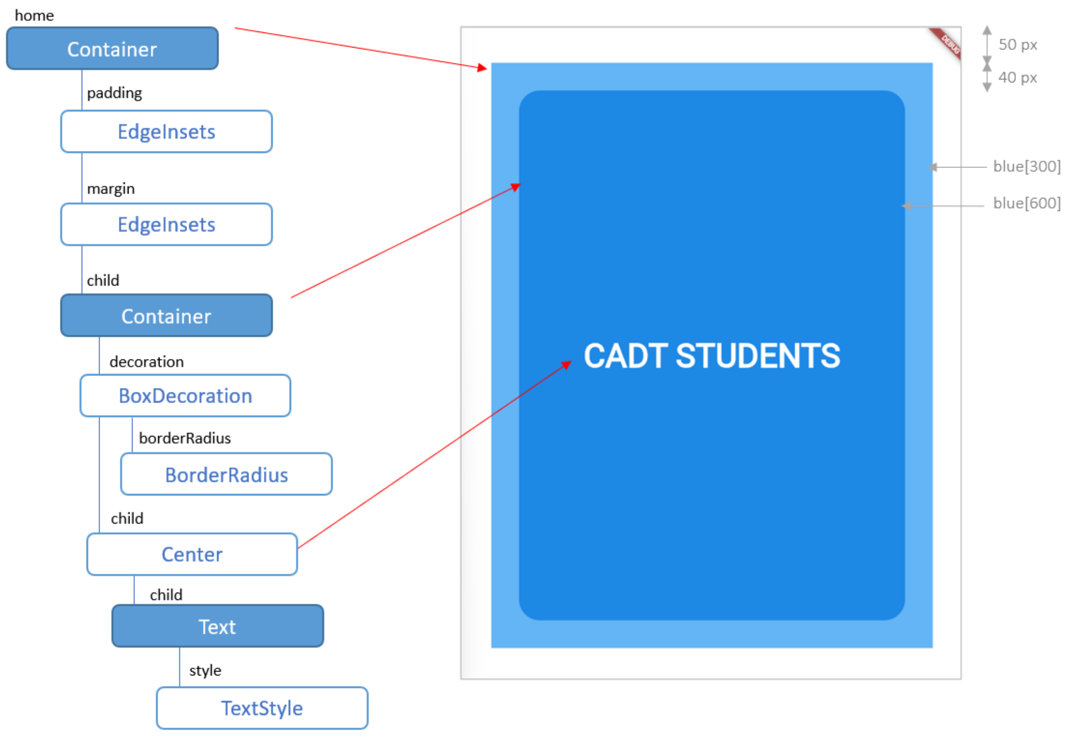
In this exercise, you will be working with the following widgets

- **Container**: A customizable box for layout and styling.
- **Text**: Displays text.
- **Center**: Aligns a child widget to the center

And the following classes

- **EdgeInsets**: Sets padding or margin.
- **BoxDecoration**: Styles a container's background, border, etc.
- **BorderRadius**: Rounds container corners.
- **TextStyle**: Styles text appearance.

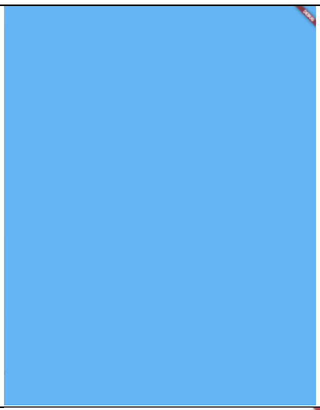
You need to produce the following mockup and widget structure:

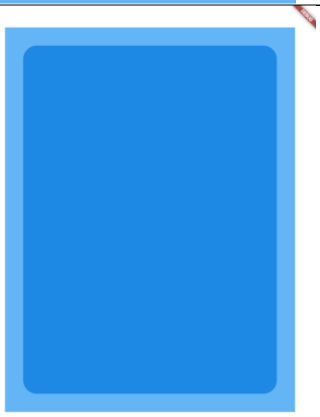


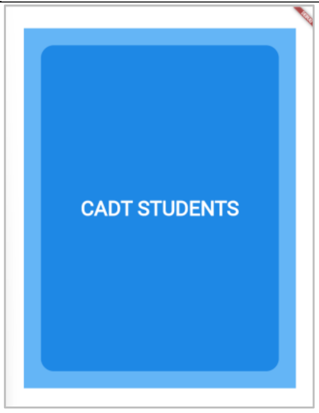
We recommend you to follow **the bellow steps** to be able to understand each widget in details. However, you are free to choose your favorite workflow!

To start
Open `W3-S2/EX-2/main.dart`

```
void main() {
  runApp(MaterialApp(
    home: Container( ),
  ));
}
```

STEP 1		Just a blue container for the home…
		
		`MaterialApp`
		`    home: Container`

STEP 2		Add an inner container and some padding margins and box decoration
		
		`MaterialApp`
		`    home: Container`
		`    margin: EdgeInsets`
		`    padding: EdgeInsets`
		`    child: Container`
		`          decoration: BoxDecoration`

STEP 3	 CADT STUDENTS	Add the Text, centered, and style it.
		
		`MaterialApp`
		`    home: Container`
		`    margin: EdgeInsets`
		`    padding: EdgeInsets`
		`    child: Container`
		`          decoration: BoxDecoration`
		`          child: Center`
		`               child: Text`
		`                style TextStyle`

In this exercise, we haven't used the Scaffold widget: **Why**? What is the **purpose of this widget**?

EX 3 – Column Gradient,

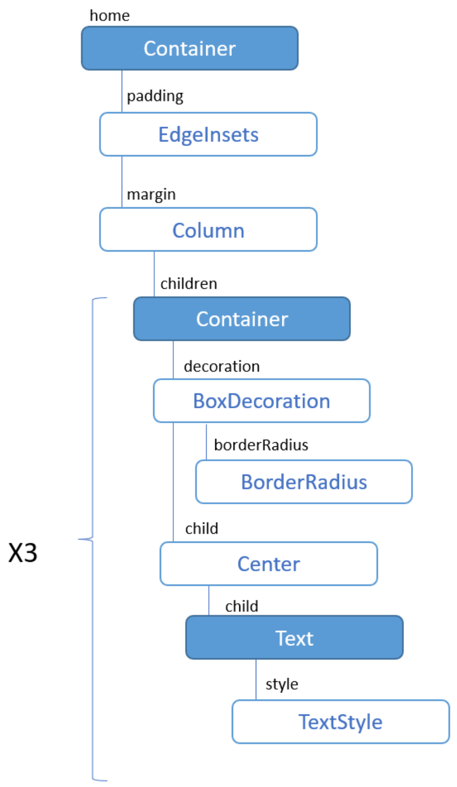
In this exercise, you will be working with the following widgets

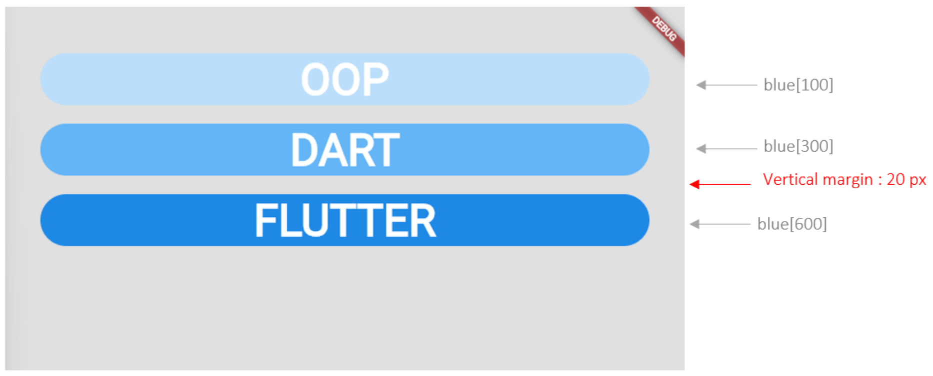
- **Container**: A customizable box for layout and styling.
- **Column** Arranges widgets vertically in a column.

And he following classes

- **EdgeInsets**: Sets padding or margin.
- **BoxDecoration**: Styles a container's background, border, etc.

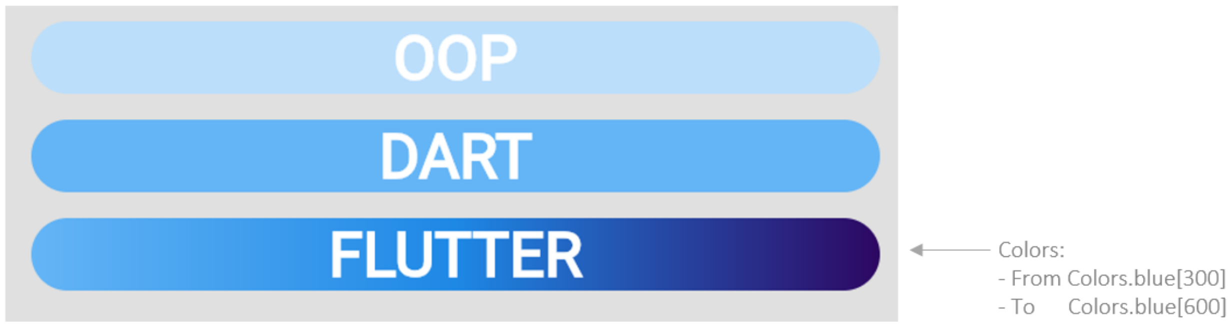
You need to produce the following mockup and widget structure:





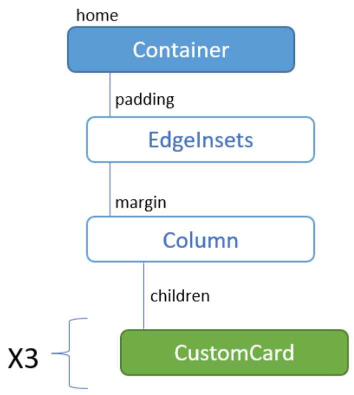
BONUS

In the last item, add a linear gradient as specified bellow



EX 4 – Extract widget to a Stateless Widget

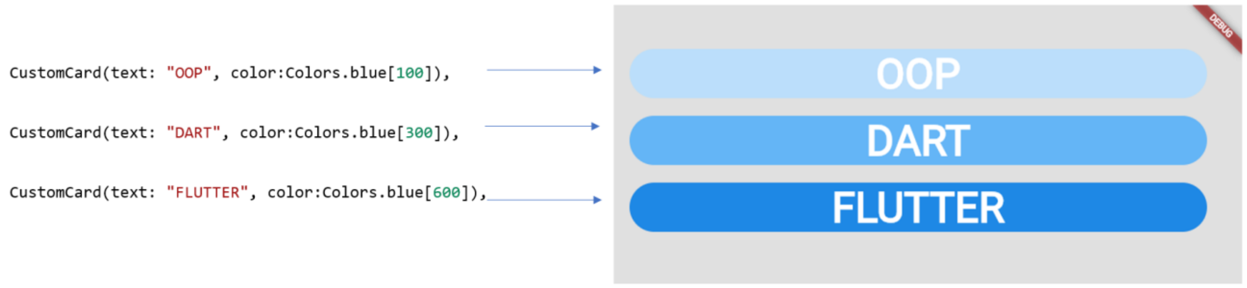
Startin from previous exercise, your You need to extract the repetitive card design into a custom Stateless Widget called `CustomCard`.



This custom widget will take 2 parameters:

Parameter name	Type	Is optional?	Default value
Text	String	no	No
Color	Color	yes	Blue

As example the previous code will be refactored as follow:

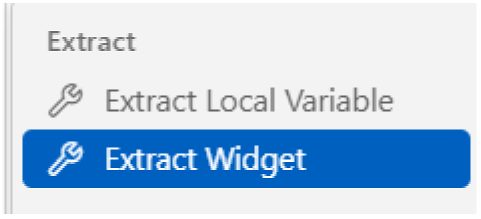


```
CustomCard(text: "OOP", color:Colors.blue[100]),

CustomCard(text: "DART", color:Colors.blue[300]),

CustomCard(text: "FLUTTER", color:Colors.blue[600]),
```

Q1 – Starting from previous exercise, extract the Container into a Stateless widget (CustomCard)



VSCode can help you to extract widget!

Q2– Add widget constructor parameters
Q3 – Update the main() to call this CustomCard widget

BONUS

If you were able to manage **linear gradient**, add another constructor parameter to display the card with gradient colors.