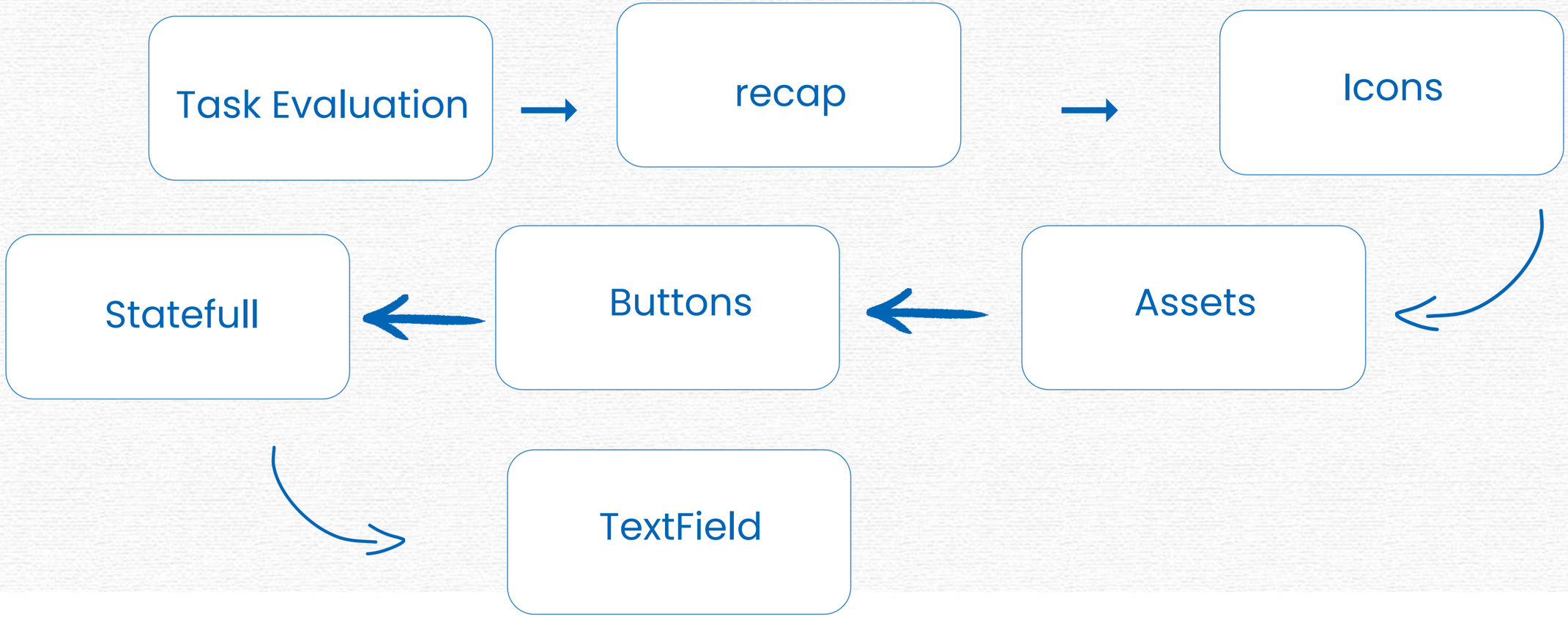
Mobile App

SESSION 7

Table of contenet



Task Evaluation

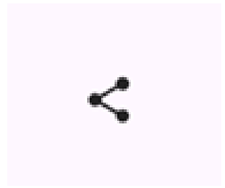
Recap

Quick recap

Icons

1.Basic Icon:

```
return MaterialApp(
  home: Scaffold(
    body: Center(child: Icon(Icons.share)),
  ), // Scaffold
); // MaterialApp
```



Icons

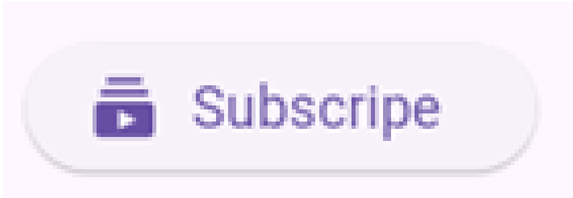
2.Icon Button

```
return MaterialApp(
    home: Scaffold(
        body: Center(
  child: IconButton(
      onPressed: () {},
      icon: Icon(
        Icons.add,
        size: 25,
        color: Colors.brown,
      )), // Icon // IconButton
))); // Center // Scaffold // MaterialApp
```

icons

3-Combining Icon and Text

```
return MaterialApp(
    home: Scaffold(
  body: Center(
    child: ElevatedButton.icon(
      onPressed: () {},
      icon: Icon(Icons.subscriptions),
      label: Text("Subscripe"),
    ), // ElevatedButton.icon
  ), // Center
)); // Scaffold // MaterialApp
```



Buttons

ElevatedButton:

A material design button that elevates when pressed. It's often used for important actions, like submitting forms or navigating to another screen.

```
return MaterialApp(
    home: Scaffold(
  body: Center(
    child: ElevatedButton(onPressed: () {}, child: Text("Send")),
  ), // Center
)); // Scaffold // MaterialApp
```

Buttons

TextButton:

A flat button with no elevation, typically used for secondary actions like “Cancel” or “Learn More”.

```
return MaterialApp(
    home: Scaffold(
  body: Center(
    child: TextButton(
        onPressed: () {},
        child: Text(
          "Cancel",
          style: TextStyle(color: Colors.black),
        )), // Text // TextButton
```

Buttons

OutlinedButton:

Similar to TextButton, but with an outline border. It's useful for secondary actions when you want a more defined boundary.

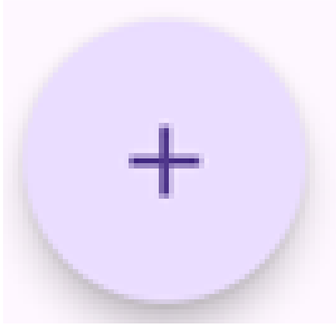
```
return MaterialApp(
    home: Scaffold(
  body: Center(
    child: OutlinedButton(
        onPressed: () {},
        child: Text(
          "Help",
          style: TextStyle(color: Colors.black),
        )), // Text // OutlinedButton
```

Buttons

FloatingActionButton:

A circular button that floats above content. It's often used for the most important action on a screen, such as adding or creating something.

```
return MaterialApp(
    home: Scaffold(
  floatingActionButton: FloatingActionButton(
    onPressed: () {},
    shape: CircleBorder(),
    child: Icon(Icons.add),
  ), // FloatingActionButton
  body: Center(child: Text("Floating Action Button Example!!")),
)); // Scaffold // MaterialApp
```



Assets

In Flutter, assets are files that are bundled and included with your app package. These can be images, fonts, JSON files, icons, or any other files your app needs to work properly or look the way you want.

Common Types of Assets in Flutter:

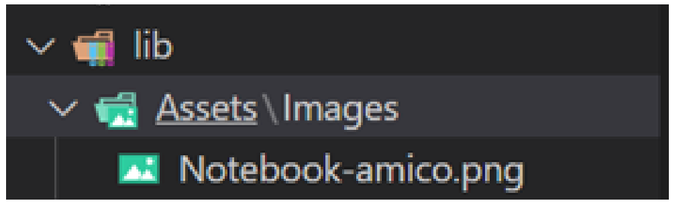
- Images
- Fonts
- Text files (e.g., .json, .txt)
- Audio or video files

Assets

How to Use Assets in Flutter

1. Add assets to your project:

Place your assets in a folder (commonly named assets) at the root of your Flutter project.



2. Declare assets in pubspec.yaml:

```
# To add assets to your application, add an assets section, like this:
assets:
  - lib\Assets\Images\Notebook-amico.png
```

Assets

Images

Assets

1.Asset Image

- Place your image inside the **assets/images/** folder.
- Add the path in pubspec.yaml file
- use **Image.Asset** Widget

```
Image.asset(
  "lib/Assets/Images/Notebook-amico.png",
  height: 350,
  width: 350,
), // Image.asset
```

Assets

2.Network Image

- use **Image.network** Widget

```
Image.network(
  "https://www.pinterest.com/pin/1618549863834736/",
  height: 350,
  width: 350,
), // Image.network
```

Assets

Fonts

Assets

1.Custom Font

```yaml
assets:
  - lib\Assets\Images\Notebook-amico.png

fonts:
  - family: vegan style
    fonts:
      - asset: lib/Assets/Fonts/VeganStylePersonalUse-5Y58.ttf
```

- Place your image inside the **assets/fonts/** folder.
- Add the path in pubspec.yaml file
- use Font family in **TextStyle** widget

```
      child: Text(
    "Custom Font",
    style: TextStyle(fontFamily: "vegan style", fontSize: 32),
  )), // Text // Center
)); // Scaffold // MaterialApp
```

Custom Font

Assets

2.Google Fonts

- Add the Google Fonts Package

```
google_fonts: ^6.2.1
```

- import 'package:google_fonts/google_fonts.dart';
- style: GoogleFonts.lobster(fontSize: 24, fontWeight: FontWeight.bold, color: Colors.blue,),

```
body: Center(
    child: Text(
  "Custom Font",
  style: GoogleFonts.abel(fontSize: 35),
)), // Text // Center
```

Custom Font

TextField

TextField

Allows user input in a form or any interactive UI element.

Key Properties:

- **controller:** Manages text input.
- **decoration:** Adds styling, like labels and borders.
- **onChanged:** Callback when the text changes.

```
child: TextField(
  controller: controller,
  decoration: InputDecoration(
      border: OutlineInputBorder(),
      hintText: "Enter ur name",
      labelText: "Name",
      prefixIcon: Icon(Icons.person),
      suffixIcon: IconButton(
          onPressed: () {
            controller.clear();
          },
          icon: Icon(Icons.clear))), // IconButton // InputDecoration
  onChanged: (value) {
    setState(() {
      inputValue = value;
    });
  },
  onSubmitted: (value) {
    setState(() {
      inputValue = "ur name sumitted Successfully";
    });
  },
), // TextField
```

Statefull

setState in Flutter:

Updates the state of a **StatefulWidget** and triggers a UI rebuild.

```
ElevatedButton(
    onPressed: () {
      setState(() {
        inputValue = controller.text;
      });
    },
    child: Text("send")), // ElevatedButton
Text(inputValue)
```

Note:

- Only works with StatefulWidget.
- Call setState inside the widget's State class.

Exercise

Simple Calculator App

tHANK YOU

SEE YOU NEXT TIME