**Objective:-** To design a layout of the Flutter App using layout widgets. Create a flutter app to manage Todos with the add , remove , complete and list of todo.

**Code :-**

* **lib**
* **constants**
* **colors.dart**

import 'package:flutter/material.dart';

// Define a constant color for red, typically used for errors or alerts

const Color tdRed = Color(0xFFDA4040);

// Define a constant color for blue, can be used for primary buttons or highlights

const Color tdBlue = Color(0xFF5F52EE);

// Define a constant color for black, useful for text or dark backgrounds

const Color tdBlack = Color(0xFF3A3A3A);

// Define a constant color for grey, often used for disabled elements or secondary text

const Color tdGrey = Color(0xFF717171);

// Define a constant background color, providing a light theme to the app

const Color tdBGColor = Color(0xFFEEEFF5);

* **controller**
* **todo\_item.dart**

import 'package:flutter/material.dart';

import '../model/todo.dart'; // Importing the ToDo model.

import '../constants/colors.dart'; // Importing custom color constants.

class ToDoItem extends StatelessWidget {

final ToDo todo; // The ToDo item to be displayed.

// Callback function when the ToDo item is changed (completed or not).

// ignore: prefer\_typing\_uninitialized\_variables

final onToDoChanged;

// Callback function when the delete icon is pressed.

// ignore: prefer\_typing\_uninitialized\_variables

final onDeleteItem;

const ToDoItem({

super.key,

required this.todo, // Required parameter for the ToDo item.

required this.onToDoChanged, // Required callback for item change.

required this.onDeleteItem, // Required callback for item deletion.

});

@override

Widget build(BuildContext context) {

return Container(

margin: const EdgeInsets.only(bottom: 20), // Space between items.

child: ListTile(

onTap: () {

// This triggers when the ToDo item is tapped.

// print('Clicked on Todo Item.'); // Uncomment for debugging.

onToDoChanged(todo); // Call the callback to update ToDo status.

},

shape: RoundedRectangleBorder(

borderRadius:

BorderRadius.circular(20), // Rounded corners for the tile.

),

contentPadding: const EdgeInsets.symmetric(

horizontal: 20, vertical: 5), // Padding inside the tile.

tileColor: Colors.white, // Background color of the tile.

leading: Icon(

// Icon showing whether the ToDo is completed or not.

todo.isDone ? Icons.check\_box : Icons.check\_box\_outline\_blank,

color: tdBlue, // Color for the checkbox icon.

),

title: Text(

todo.todoText!, // Display the text of the ToDo item.

style: TextStyle(

fontSize: 16, // Font size for the ToDo text.

color: tdBlack, // Text color.

decoration: todo.isDone

? TextDecoration.lineThrough

: null, // Strikethrough if done.

),

),

trailing: Container(

padding: const EdgeInsets.all(

0), // No padding inside the trailing container.

margin: const EdgeInsets.symmetric(

vertical: 12), // Vertical margin for the container.

height: 35, // Fixed height for the delete button.

width: 35, // Fixed width for the delete button.

decoration: BoxDecoration(

color: tdRed, // Background color for the delete button.

borderRadius: BorderRadius.circular(

5), // Rounded corners for the delete button.

),

child: IconButton(

color: Colors.white, // Icon color.

iconSize: 18, // Size of the delete icon.

icon: const Icon(Icons.delete), // Delete icon.

onPressed: () {

// This triggers when the delete icon is pressed.

// print('Clicked on delete icon'); // Uncomment for debugging.

onDeleteItem(

todo.id); // Call the callback to delete the ToDo item.

},

),

),

),

);

}

}

* **model**
* **todo.dart**

class ToDo {

*// Unique identifier for the ToDo item*

  String? id;

*// The text description of the ToDo item*

  String? todoText;

*// Flag to indicate whether the ToDo item is completed or not*

  bool isDone;

*// Constructor for the ToDo class, with 'id' and 'todoText' being required*

*// 'isDone' is optional and defaults to false*

  ToDo({

    required *this*.id,

    required *this*.todoText,

*this*.isDone = false,

  });

*// Static method to return an empty list of ToDo items*

  static List<ToDo> todoList() {

    return []; *// Returns an empty list of ToDo objects*

  }

}

* **view**
* **home.dart**

import 'package:flutter/material.dart';

import 'package:flutter/services.dart';

import '../model/todo.dart'; // Importing the ToDo model

import '../constants/colors.dart'; // Importing color constants

import '../controller/todo\_item.dart'; // Importing the ToDoItem widget

// The main Home widget, a StatefulWidget to manage state

class Home extends StatefulWidget {

const Home({super.key});

@override

State<Home> createState() => \_HomeState();

}

// State class for Home

class \_HomeState extends State<Home> {

// List to hold all the ToDo items

final todosList =

ToDo.todoList(); // Initializing the list with predefined items

List<ToDo> \_foundToDo = []; // List to hold filtered ToDo items

final \_todoController =

TextEditingController(); // Controller for the TextField

@override

void initState() {

\_foundToDo = todosList; // Initialize \_foundToDo with all todos

super.initState();

}

@override

Widget build(BuildContext context) {

return Scaffold(

backgroundColor: tdBGColor, // Setting background color

appBar: \_buildAppBar(), // Building the app bar

body: Stack(

children: [

Container(

padding: const EdgeInsets.symmetric(

horizontal: 20,

vertical: 15,

),

child: Column(

children: [

searchBox(), // Search box for filtering todos

Expanded(

child: ListView(

children: [

Container(

margin: const EdgeInsets.only(

top: 50,

bottom: 20,

),

child: const Text(

'All ToDos', // Title for the ToDo list

style: TextStyle(

fontSize: 30,

fontWeight: FontWeight.w500,

),

),

),

// Displaying each ToDo item in reverse order

for (ToDo todoo in \_foundToDo.reversed)

ToDoItem(

todo: todoo, // Passing the current ToDo item

onToDoChanged:

\_handleToDoChange, // Callback for when ToDo is changed

onDeleteItem:

\_deleteToDoItem, // Callback for deleting a ToDo item

),

],

),

)

],

),

),

// Bottom input field and button for adding new ToDo items

Align(

alignment: Alignment.bottomCenter,

child: Row(children: [

Expanded(

child: Container(

margin: const EdgeInsets.only(

bottom: 20,

right: 20,

left: 20,

),

padding: const EdgeInsets.symmetric(

horizontal: 20,

vertical: 5,

),

decoration: BoxDecoration(

color: Colors.white, // White background for input field

boxShadow: const [

BoxShadow(

color: Colors.grey, // Shadow effect

offset: Offset(0.0, 0.0),

blurRadius: 10.0,

spreadRadius: 0.0,

),

],

borderRadius: BorderRadius.circular(10), // Rounded corners

),

child: TextField(

controller: \_todoController, // TextField controller

decoration: const InputDecoration(

hintText: 'Add a new todo item', // Placeholder text

border: InputBorder.none), // No border

),

),

),

Container(

margin: const EdgeInsets.only(

bottom: 20,

right: 20,

),

child: ElevatedButton(

onPressed: () {

\_addToDoItem(\_todoController.text); // Adding new ToDo item

},

style: ElevatedButton.styleFrom(

backgroundColor: tdBlue, // Button background color

minimumSize: const Size(60, 60), // Button size

elevation: 10, // Button shadow

),

child: const Text(

'+', // Button text

style: TextStyle(

fontSize: 40, color: Colors.white), // Text style

),

),

),

]),

),

],

),

);

}

// Handle change of ToDo item status (completed or not)

void \_handleToDoChange(ToDo todo) {

setState(() {

todo.isDone = !todo.isDone; // Toggle the isDone status

});

}

// Delete ToDo item from the list by its ID

void \_deleteToDoItem(String id) {

setState(() {

todosList

.removeWhere((item) => item.id == id); // Remove item from the list

});

}

// Add new ToDo item to the list

void \_addToDoItem(String toDo) {

setState(() {

todosList.add(ToDo(

id: DateTime.now()

.millisecondsSinceEpoch

.toString(), // Unique ID based on timestamp

todoText: toDo, // ToDo text

));

});

\_todoController.clear(); // Clear the text field after adding

}

// Filter ToDo items based on search input

void \_runFilter(String enteredKeyword) {

List<ToDo> results = [];

if (enteredKeyword.isEmpty) {

results = todosList; // If search input is empty, show all todos

} else {

results = todosList

.where((item) => item.todoText!.toLowerCase().contains(

enteredKeyword.toLowerCase())) // Filter based on search input

.toList();

}

setState(() {

\_foundToDo = results; // Update found ToDo list

});

}

// Widget for the search box

Widget searchBox() {

return Container(

padding: const EdgeInsets.symmetric(horizontal: 15),

decoration: BoxDecoration(

color: Colors.white, // White background for the search box

borderRadius: BorderRadius.circular(20), // Rounded corners

),

child: TextField(

onChanged: (value) => \_runFilter(value), // Run filter on text change

decoration: const InputDecoration(

contentPadding: EdgeInsets.all(0), // Padding for the input

prefixIcon: Icon(

Icons.search, // Search icon

color: tdBlack,

size: 20,

),

prefixIconConstraints: BoxConstraints(

maxHeight: 20,

minWidth: 25,

),

border: InputBorder.none, // No border for the input field

hintText: 'Search', // Placeholder text

hintStyle: TextStyle(color: tdGrey), // Hint text style

),

),

);

}

// Widget to build the app bar

AppBar \_buildAppBar() {

return AppBar(

backgroundColor: tdBGColor, // App bar background color

elevation: 0, // No shadow

title: const Row(

mainAxisAlignment: MainAxisAlignment.spaceBetween,

children: [

Icon(

Icons.menu, // Menu icon

color: tdBlack,

size: 30,

),

]),

);

}

}

// Main application widget

class MyApp extends StatelessWidget {

const MyApp({super.key});

// This widget is the root of your application.

@override

Widget build(BuildContext context) {

SystemChrome.setSystemUIOverlayStyle(const SystemUiOverlayStyle(

statusBarColor: Colors.transparent)); // Setting the status bar color

return const MaterialApp(

debugShowCheckedModeBanner: false, // Disable debug banner

title: 'ToDo App', // Application title

home: Home(), // Home widget as the main screen

);

}

}

* **main.dart**

import 'package:flutter/material.dart'; // Importing Flutter material package for UI components

import 'package:flutter/services.dart'; // Importing Flutter services package for system UI control

import './view/home.dart'; // Importing the Home widget from the view directory

void main() {

runApp(

const MyApp()); // Running the MyApp widget as the root of the application

}

class MyApp extends StatelessWidget {

const MyApp({super.key}); // Constructor for MyApp with a key parameter

// This widget is the root of your application.

@override

Widget build(BuildContext context) {

// Setting the system UI overlay style to make the status bar transparent

SystemChrome.setSystemUIOverlayStyle(

const SystemUiOverlayStyle(statusBarColor: Colors.transparent));

// Building the MaterialApp widget which serves as the main app structure

return const MaterialApp(

debugShowCheckedModeBanner: false, // Hiding the debug banner in the app

title: 'ToDo App', // Setting the title of the application

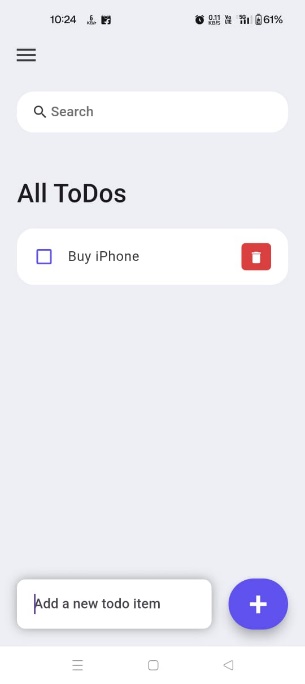
home: Home(), // Specifying the Home widget as the starting screen

);

}

**}**

**Output:-**

****