VIDYA JYOTHI INSTITUTE OF TECHNOLOGY

(An Autonomous Institution)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

B.Tech(CSE-DS) III Year / I Semester (R22)

LAB MANUAL

Name of the Faculty	**KISHORE K**
Department	**CSE(Data Science)**
Year & Semester	**B.Tech-III & I Sem**
Regulation	**R22**
Lab Name	**UI DESIGN-FLUTTER LAB (SKILL DEVELOPMENT COURSE)**

DEPARTMENT OF CSE (Data Science)

VIDYA JYOTHI INSTITUTE OF TECHNOLOGY

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

An Autonomous Institution

AZIZ NAGAR, C B POST, HYDERABAD-500075

2024-2025

INDEX

Week 1. Installation of Android Studio & Flutter.

Flutter Installation

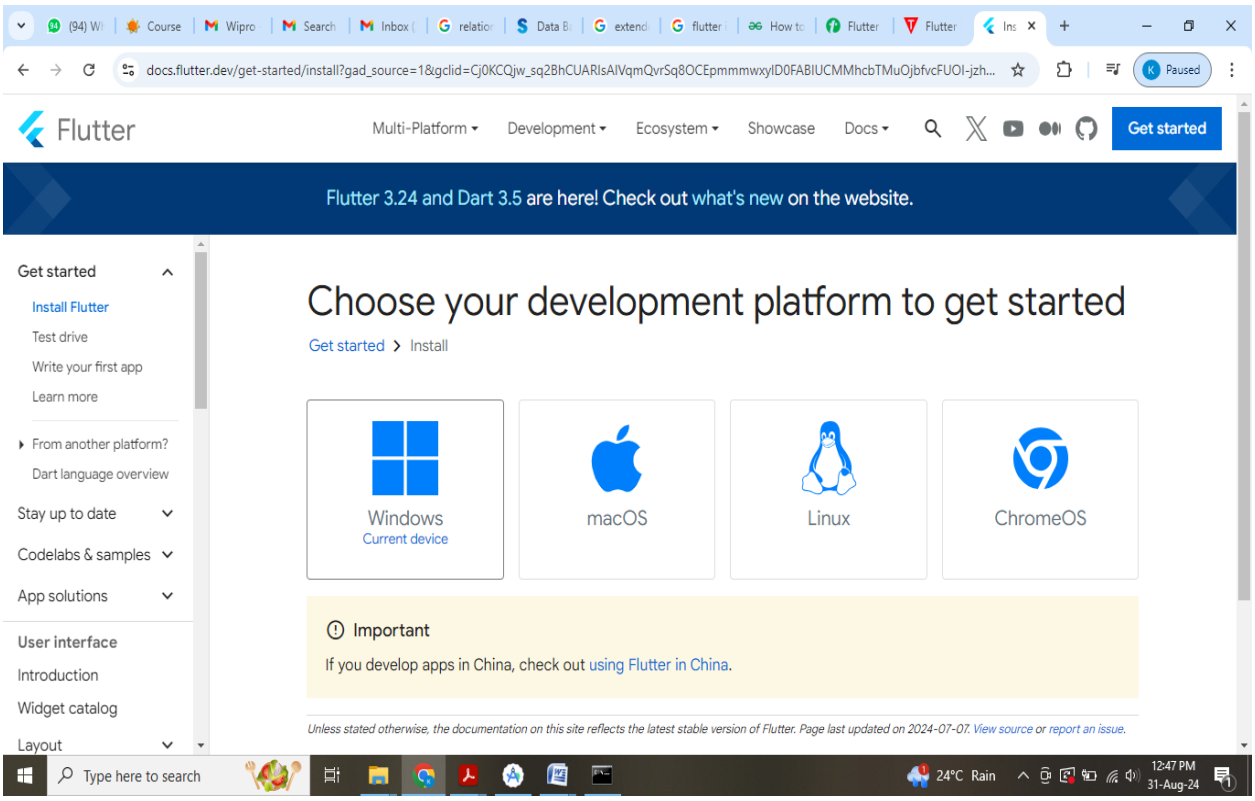
Flutter is an open-source portable UI framework for mobile, desktop, and web. It is developed and managed by Google, Flutter is used for creating a high-quality, beautiful, and fast native interface for android and iOS from a single codebase. The interface of a flutter app is composed of various widgets. Flutter has a quite rich widget library so that developers can create an effective interface for android and iOS as well without any hurdle.

Key Features of Flutter:

- **Hot Reload:** This feature lets users see every change in milliseconds in the output screen(Emulator or Android Device).
- **Rich Widget library:** Flutter contains a huge library of widgets, making an interface in flutter become quite easy and less time-consuming when we have access to the large library of widgets.
- **Expressive and Flexible UI:** An interface of a flutter app works as a layered architecture that allows full customization, which results in an extremely fast, expressive, and flexible UI.
- **Native performance:** When working with flutter app development, we have access to the widgets that are based on the platform such as android and iOS, so we get to use widgets that can integrate native functionality such as icons, navigation, scrolling, and many more.

Install the Flutter SDK

Step 1: Download the installation bundle of the Flutter Software Development Kit for windows. To download Flutter SDK, Go to its official website, click on **Get started** button, you will get the following screen.

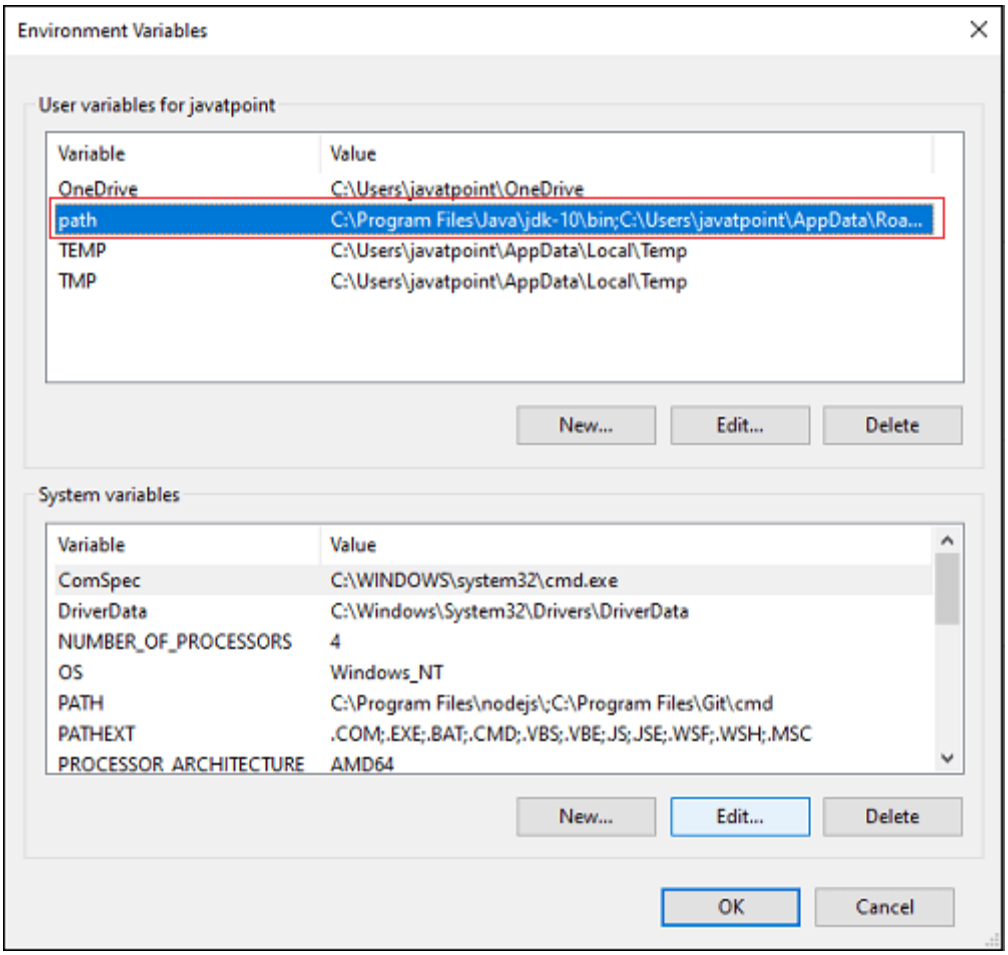


Step 2: Next, to download the latest Flutter SDK, click on the Windows **icon**. Here, you will find the download link for SDK.

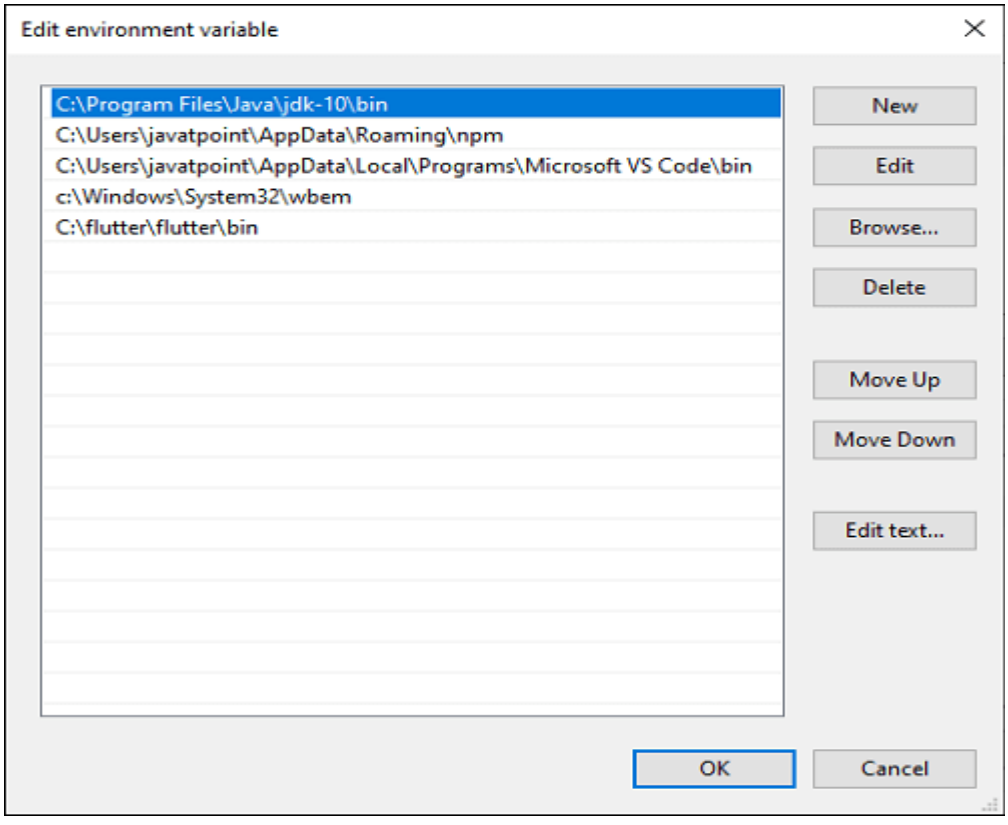
Step 3: When your download is complete, extract the **zip** file and place it in the desired installation folder or location, for example, D: /Flutter.

Step 4: To run the Flutter command in regular windows console, you need to update the system path to include the flutter bin directory. The following steps are required to do this:

Step 4.1: Go to MyComputer properties -> advanced tab -> environment variables. You will get the following screen.



Step 4.2: Now, select path -> click on edit. The following screen appears.

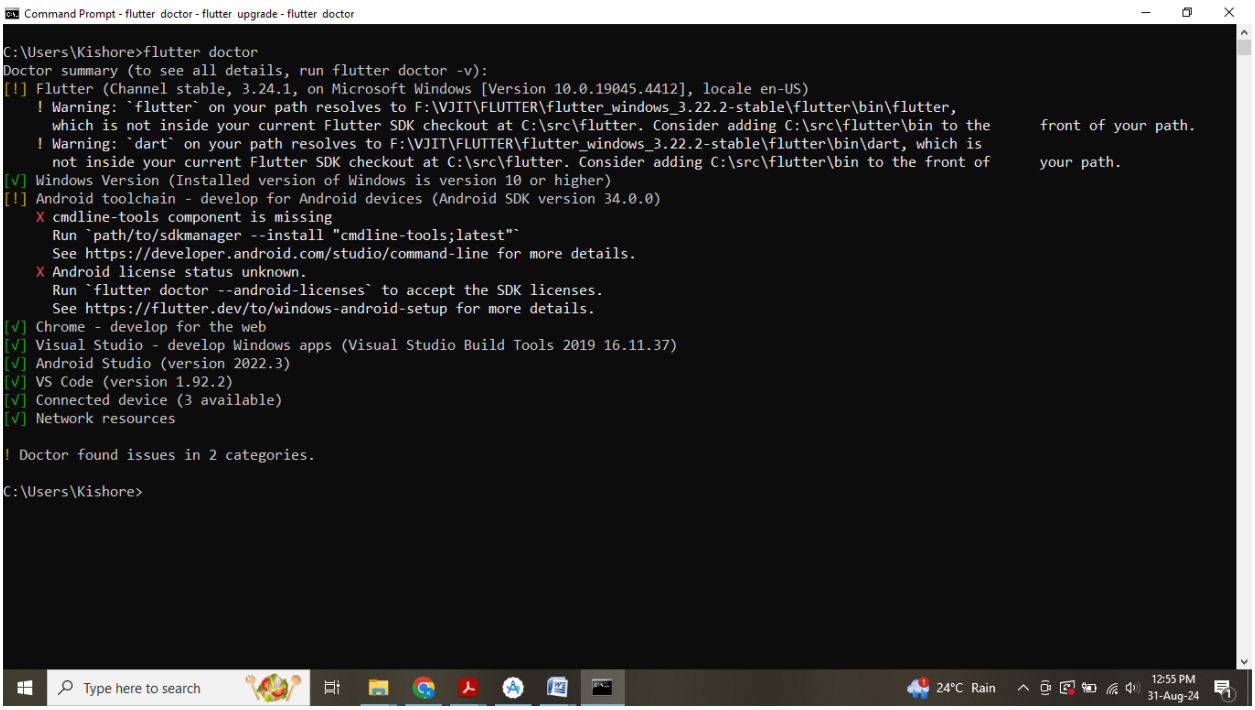


Step 4.3: In the above window, click on New->write path of Flutter bin folder in variable value -> ok -> ok -> ok.

Step 5: Now, run the $ **flutter doctor** command. This command checks for all the requirements of Flutter app development and displays a report of the status of your Flutter installation.

1. $ flutter doctor

Step 6: When you run the above command, it will analyze the system and show its report, as shown in the below image. Here, you will find the details of all missing tools, which required to run Flutter as well as the development tools that are available but not connected with the device.



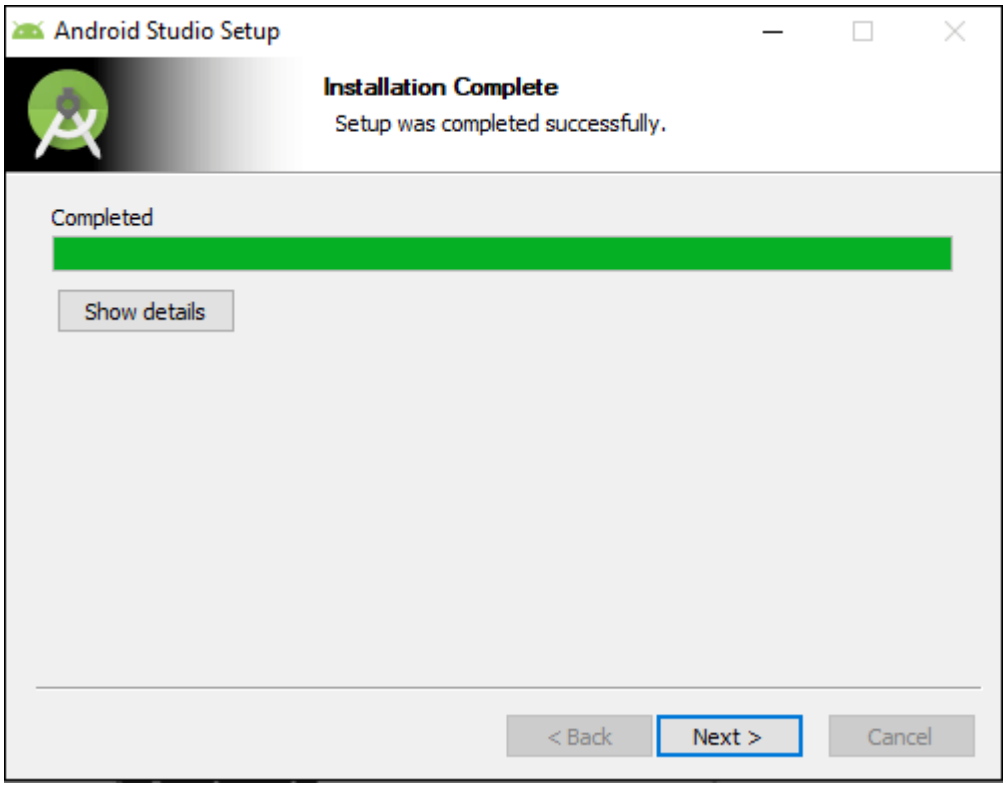
Install the Flutter SDK on Android Studio

Step 7: Install the Android SDK. If the flutter doctor command does not find the Android SDK tool in your system, then you need first to install the Android Studio IDE. To install Android Studio IDE, do the following steps.

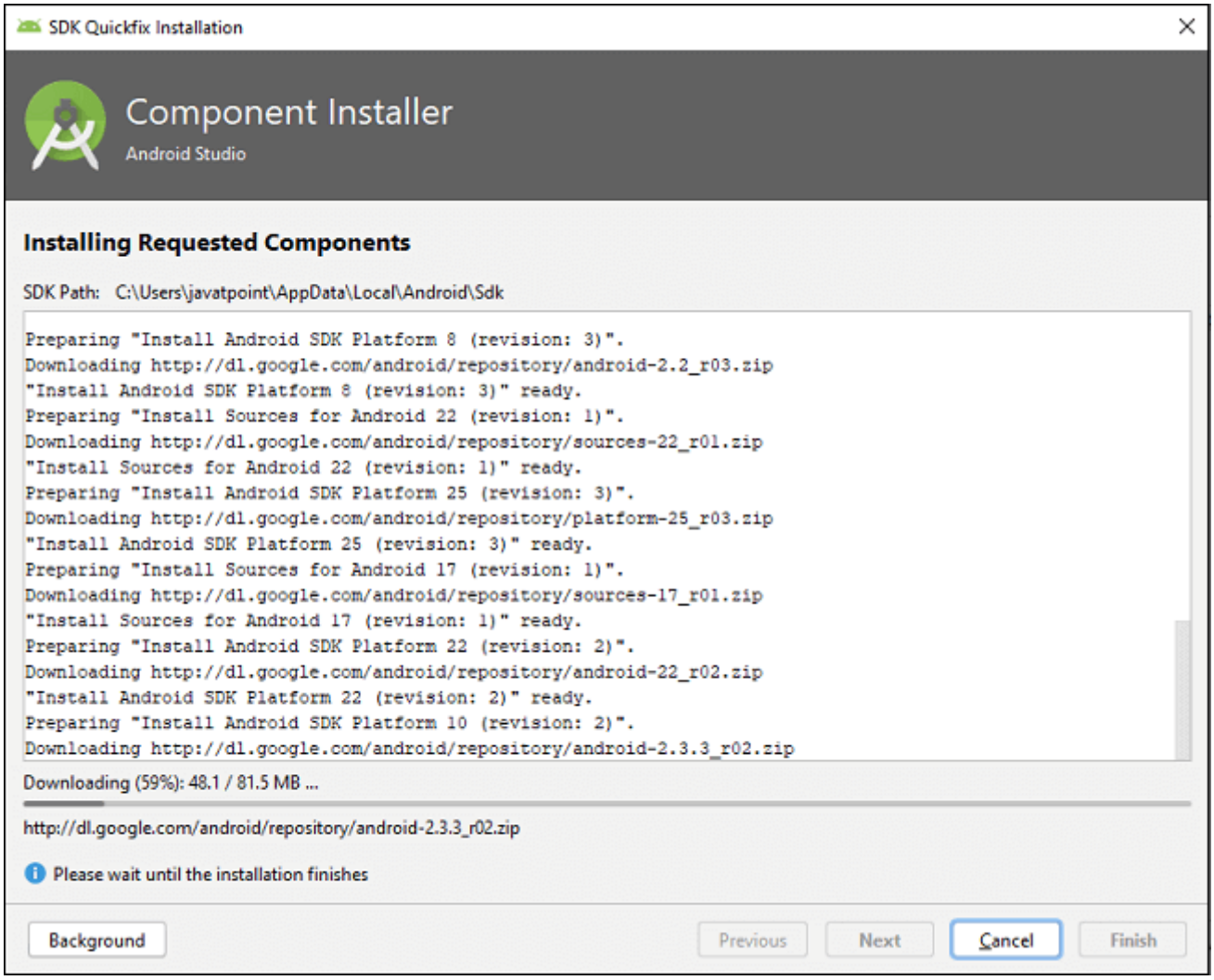
Step 7.1: Download the latest Android Studio executable or zip file from the official site.

Step 7.2: When the download is complete, open the **.exe** file and run it. You will get the following dialog box.

Step 7.3: Follow the steps of the installation wizard. Once the installation wizard completes, you will get the following screen.

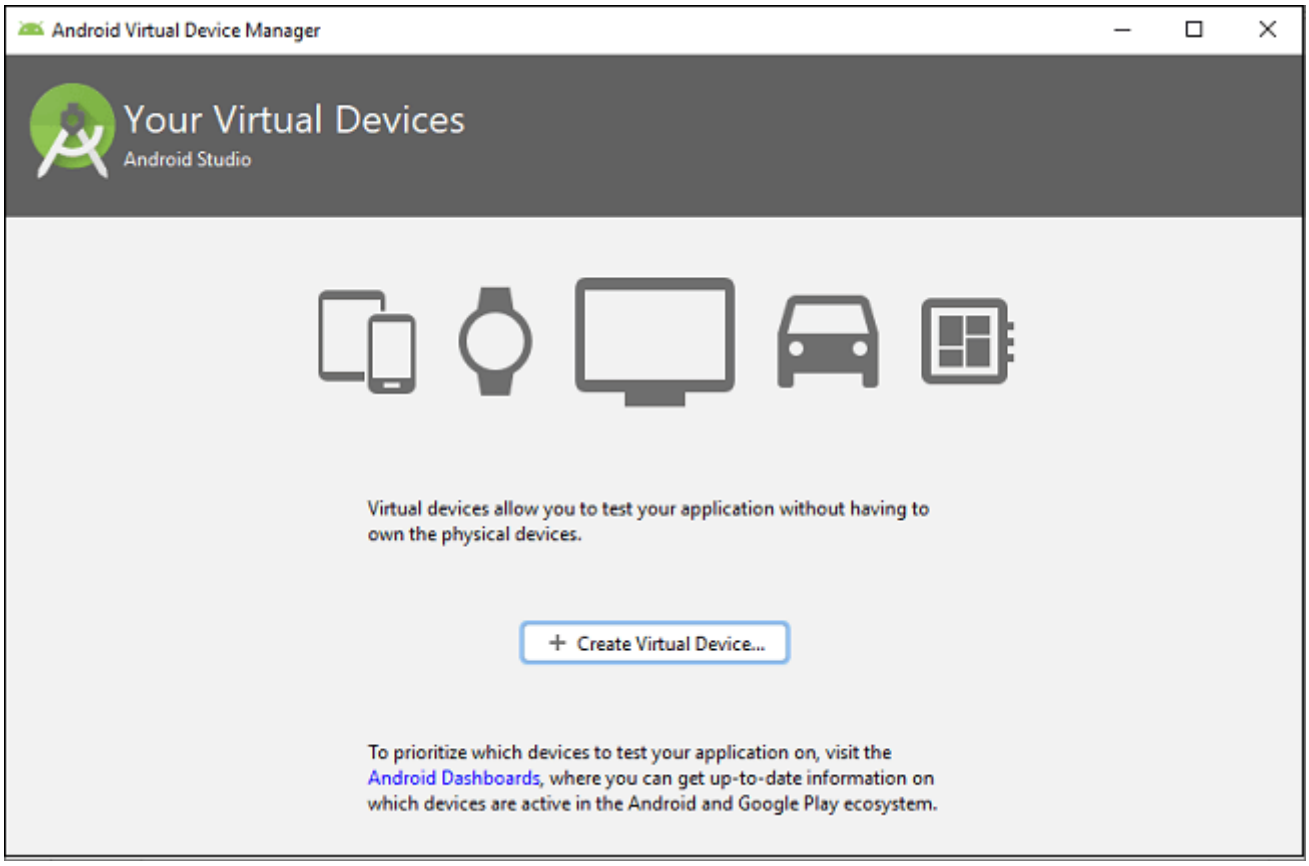


Step 7.4: In the above screen, click Next-> Finish. Once the Finish button is clicked, you need to choose the 'Don't import Settings option' and click OK. It will start the Android Studio.



Step 8: Next, you need to set up an Android emulator. It is responsible for running and testing the Flutter application.

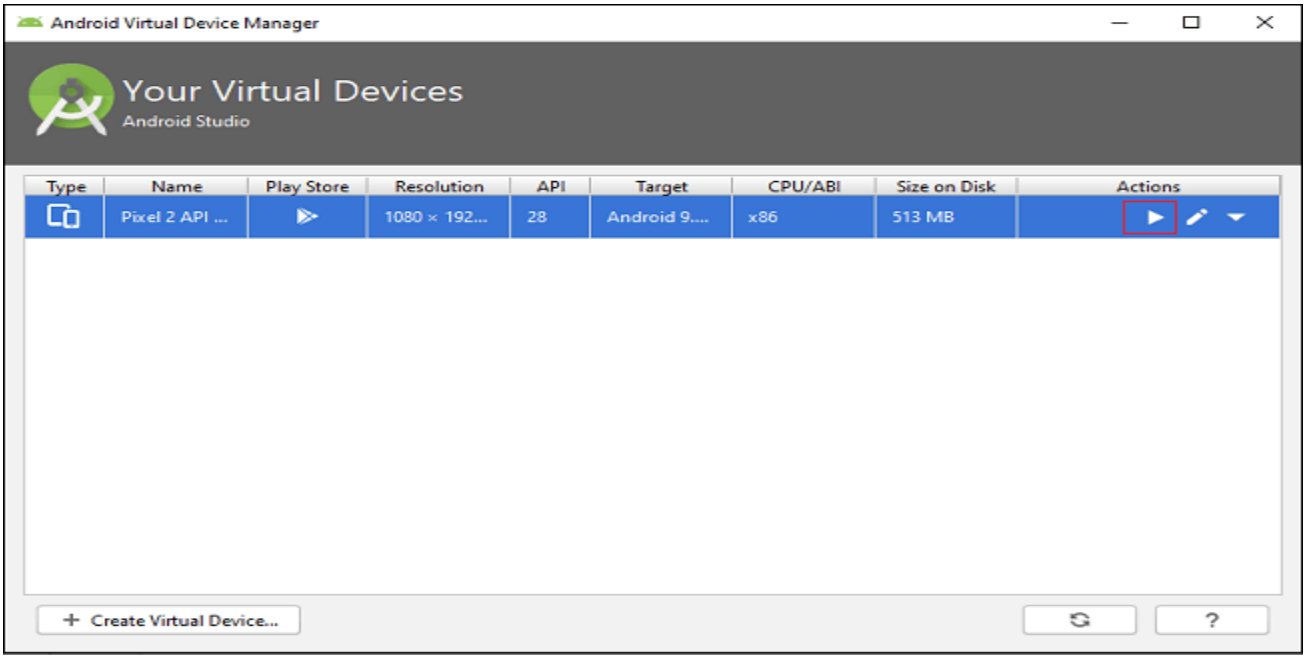
Step 8.1: To set an Android emulator, go to Android Studio > Tools > Android > AVD Manager and select Create Virtual Device. Or, go to Help->Find Action->Type Emulator in the search box. You will get the following screen.



Step 8.2: Choose your device definition and click on Next.

Step 8.3: Select the system image for the latest Android version and click on Next.

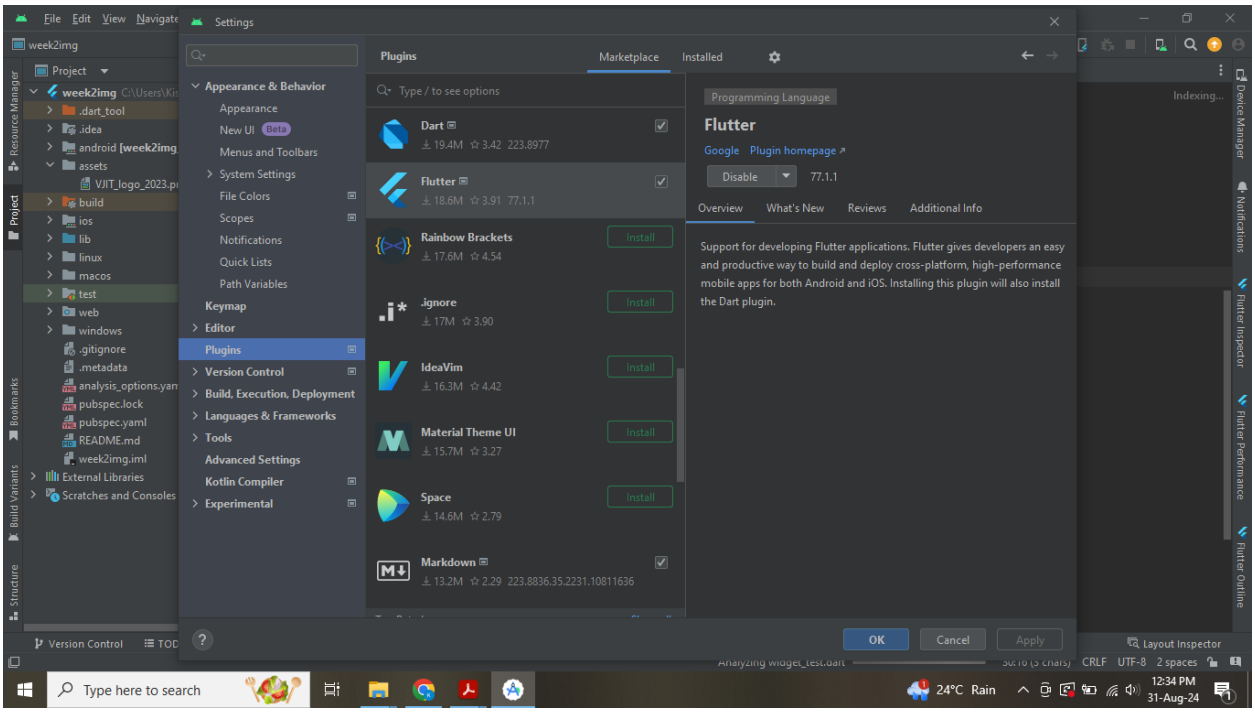
Step 8.4: Now, verify the all AVD configuration. If it is correct, click on Finish. The following screen appears.



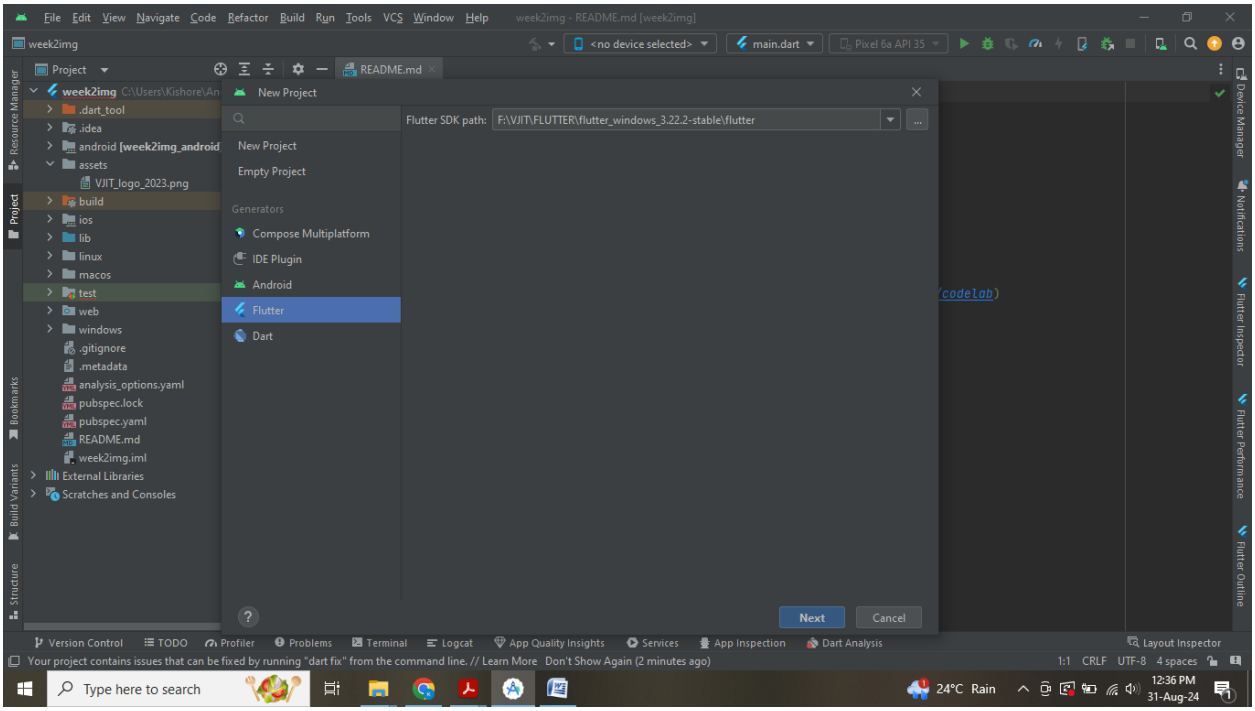
Step 8.5: Last, click on the icon pointed into the red color rectangle. The Android emulator displayed as below screen.

Step 9: Now, install Flutter and Dart plugin for building Flutter application in Android Studio. These plugins provide a template to create a Flutter application, give an option to run and debug Flutter application in the Android Studio itself. Do the following steps to install these plugins.

Step 9.1: Open the Android Studio and then go to File->Settings->Plugins.



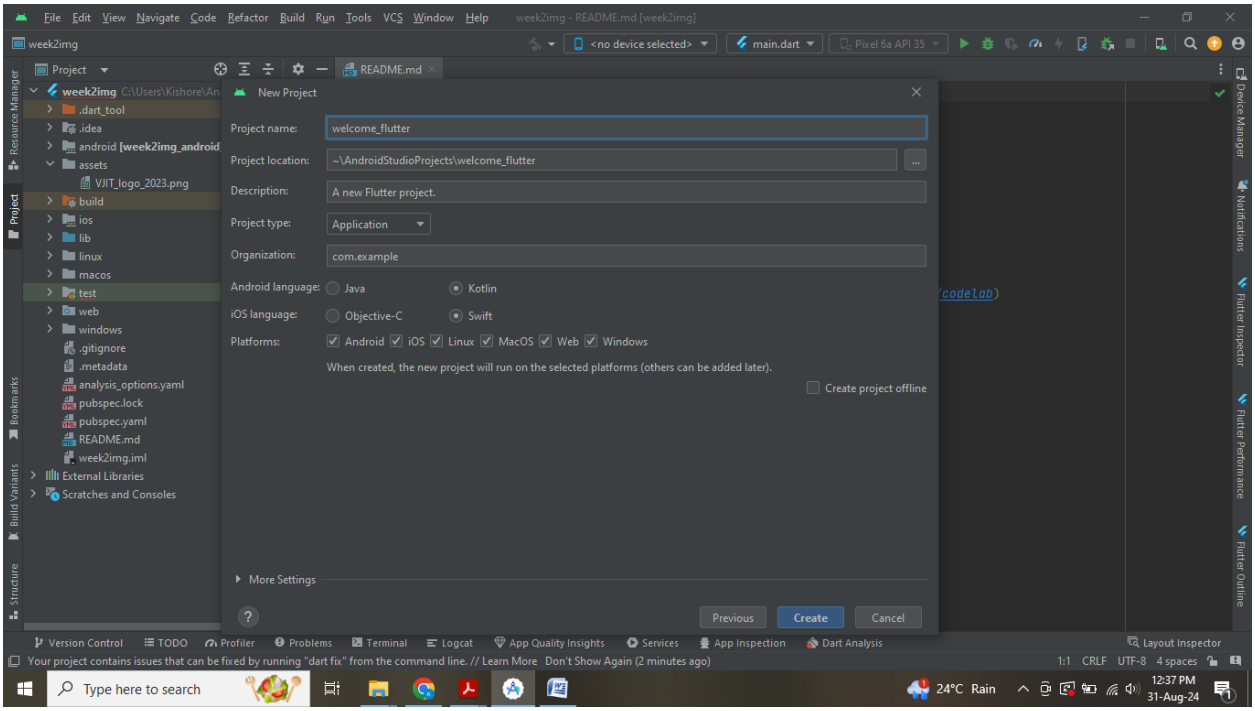
Step 9.2: Now, search the Flutter plugin. If found, select Flutter plugin and click install. When you click on install, it will ask you to install Dart plugin as below screen. Click yes to proceed.



Step 9.3: Restart the Android Studio.

Step 9.4: Create New Flutter Project

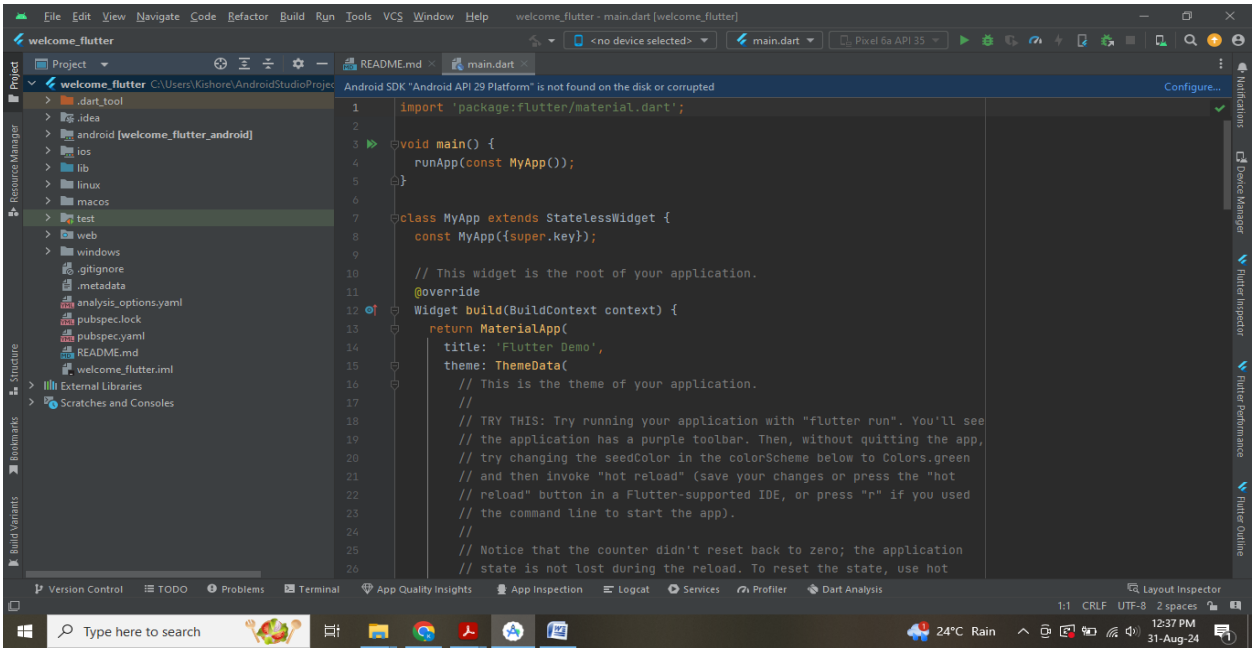
Goto file->New->New Flutter Project



Step 9.5: Assign Project Name

Welcome_flutter

Note: ***Project name must be in lowercase letters along with _ symbol***

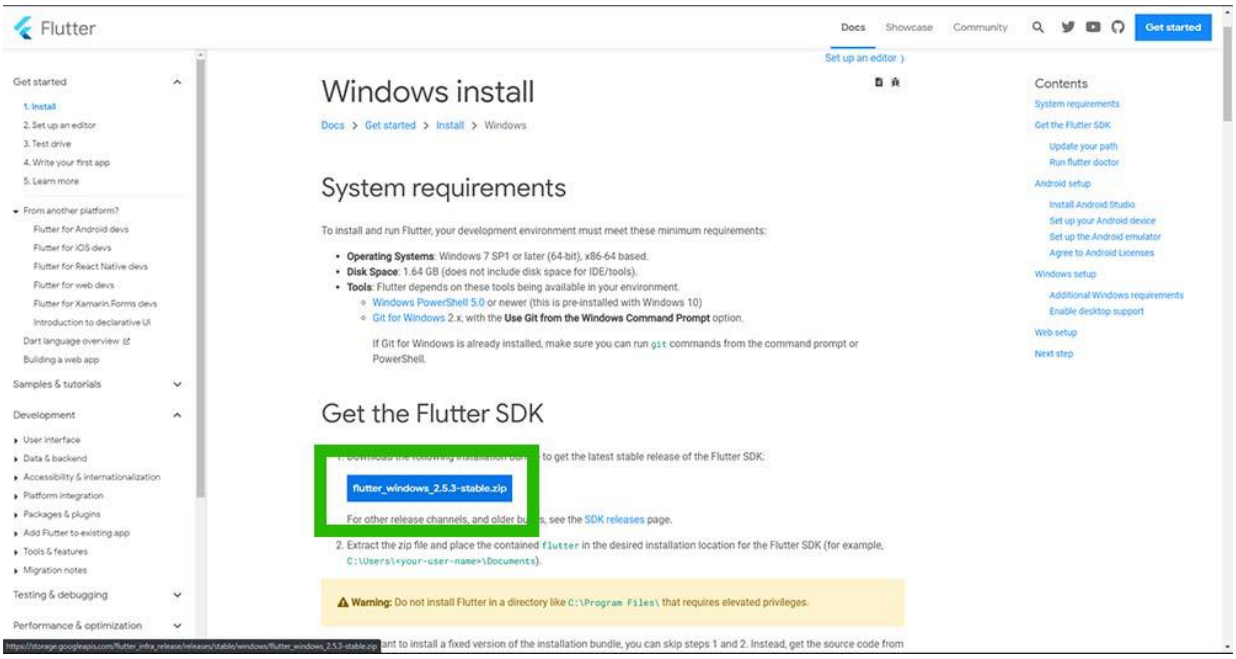


Flutter Installation on Visual Studio Code

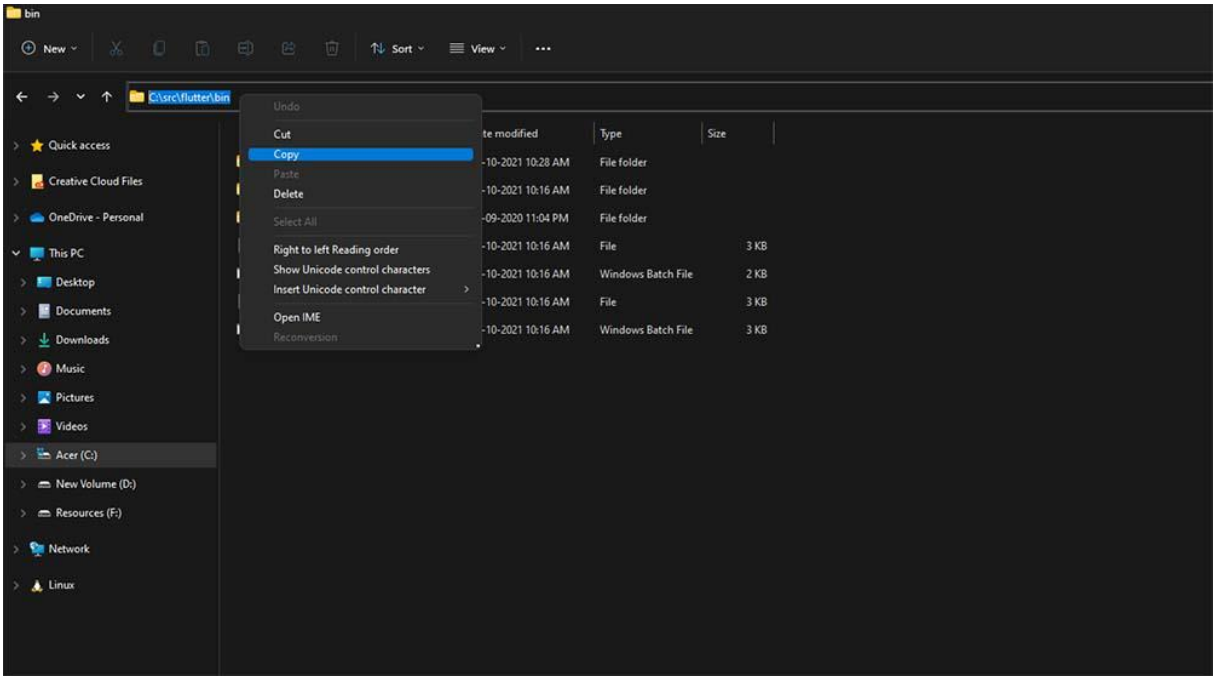
Installing Flutter in Visual Studio Code:

Follow the below steps to install Flutter in VS Code:

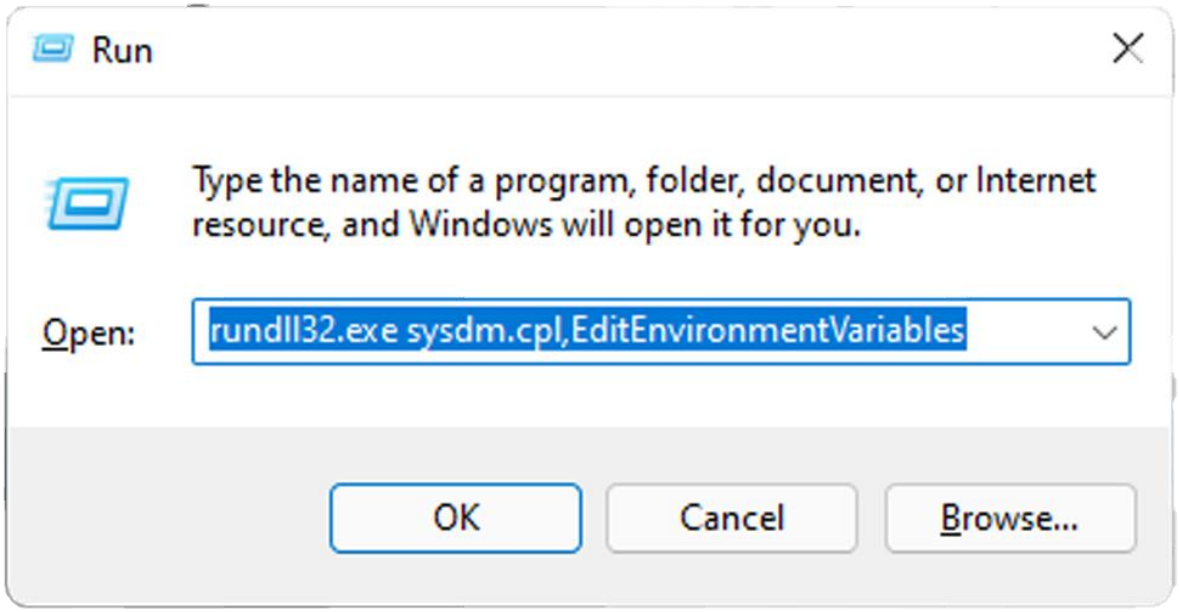
Step 1: Download the Flutter SDK. We'll have to download the Flutter SDK file in order to work with flutter. We can easily download it from the official website of Flutter.



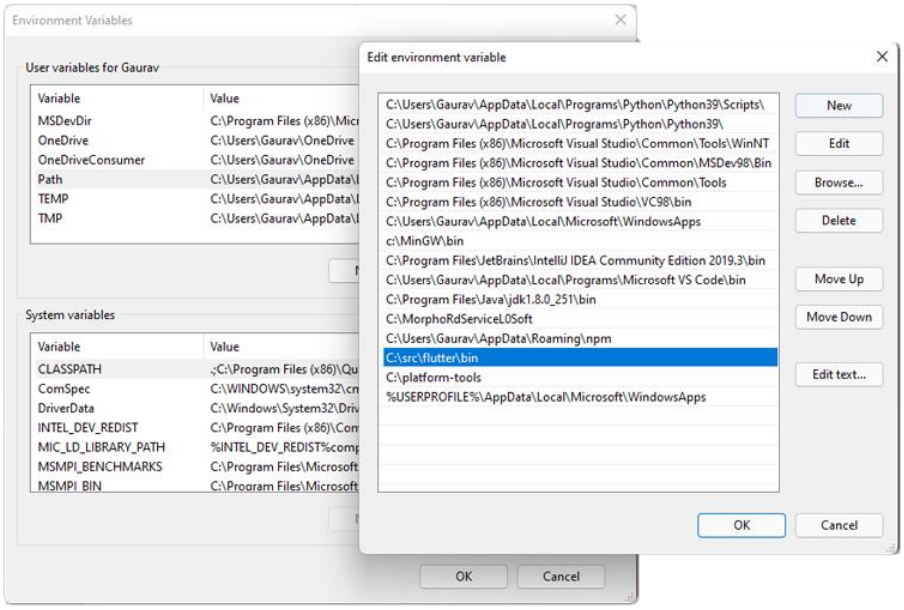
Step 2: Set Environment variable path. After downloading Flutter SDK, extract the file and copy the path of the bin folder.



Press WIN + R and paste the following: ***rundll32.exe sysdm.cpl,EditEnvironmentVariables***

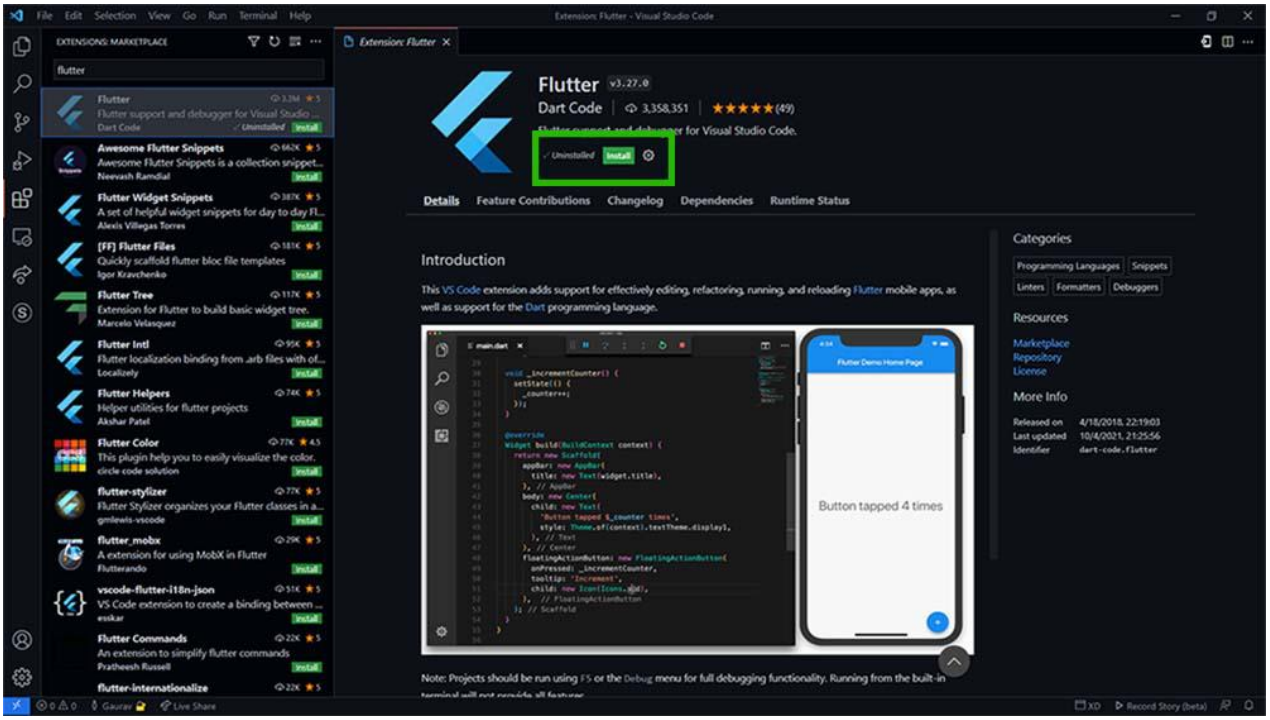


Now click on new and paste the path that was copied earlier and save.

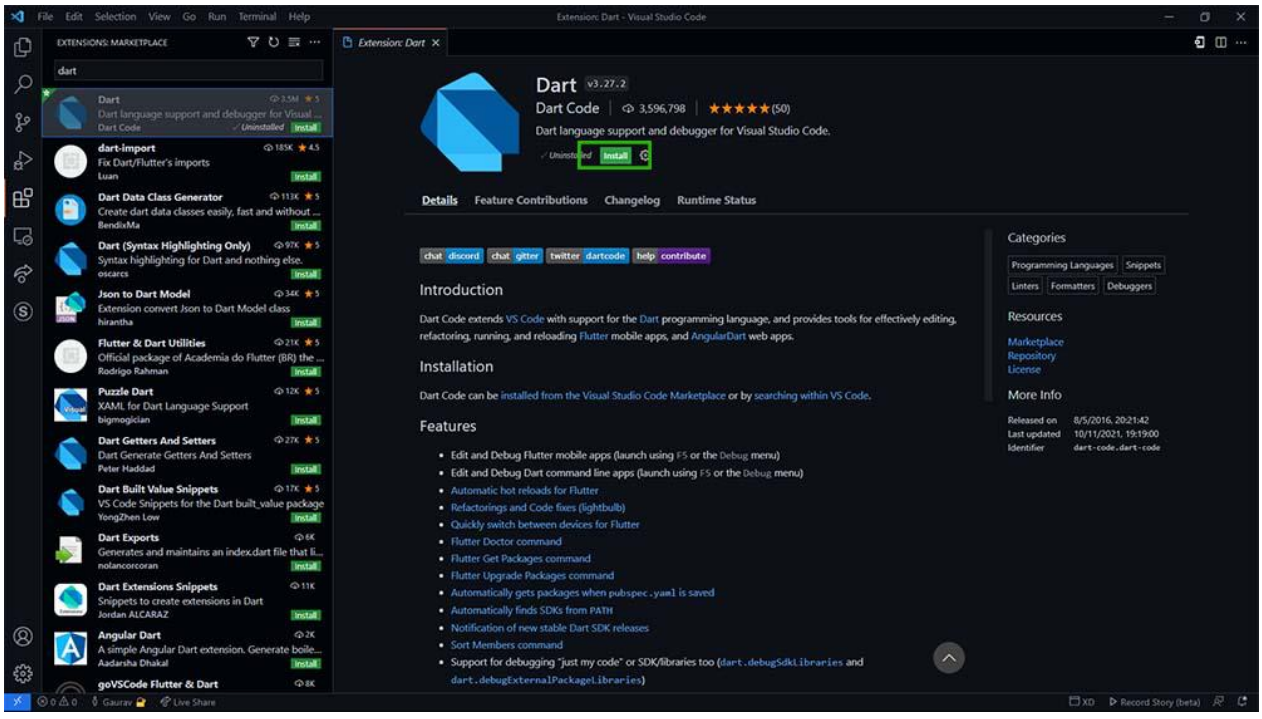


Now We have to set up the Visual Studio Code for the Flutter. We have to install two extensions in order to use flutter. The very first extension is Flutter and 2nd is Dart. Note that Dart is the programming language that is used in flutter for the application development for both and android and iOS.

Step 3: Installing Flutter in Visual Studio Code

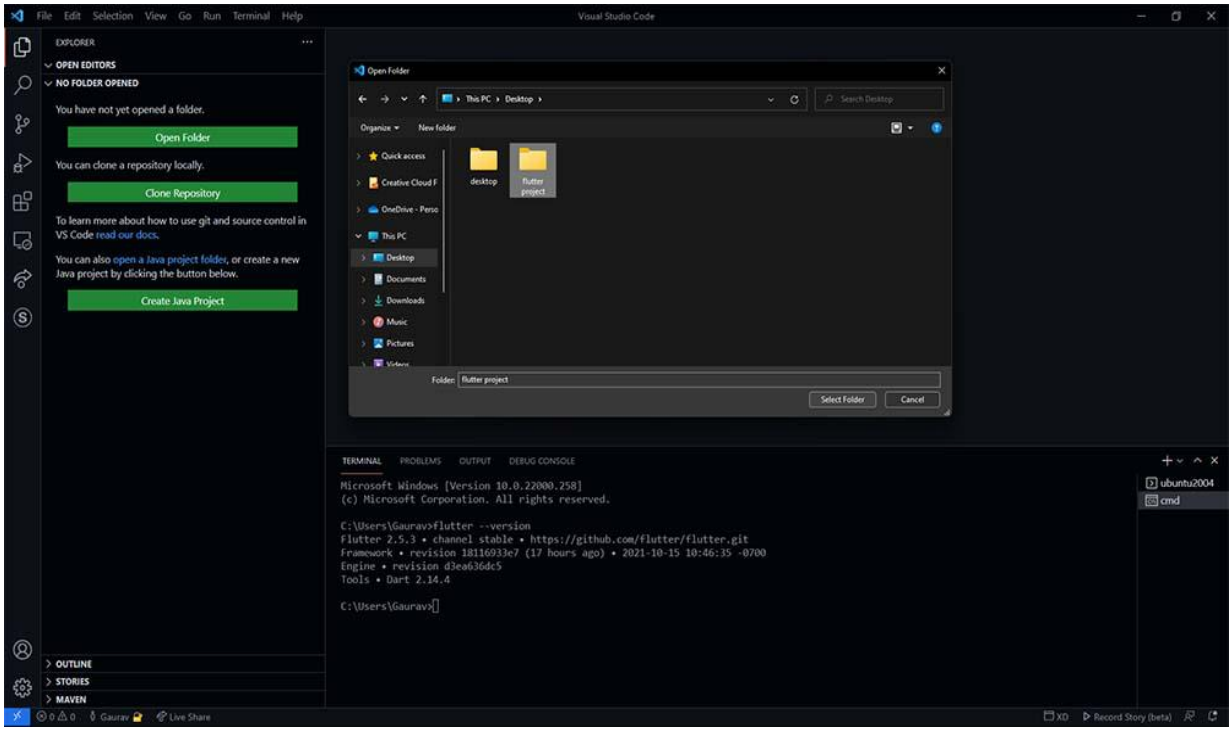


Step 4: Now install Dart in Visual Studio Code



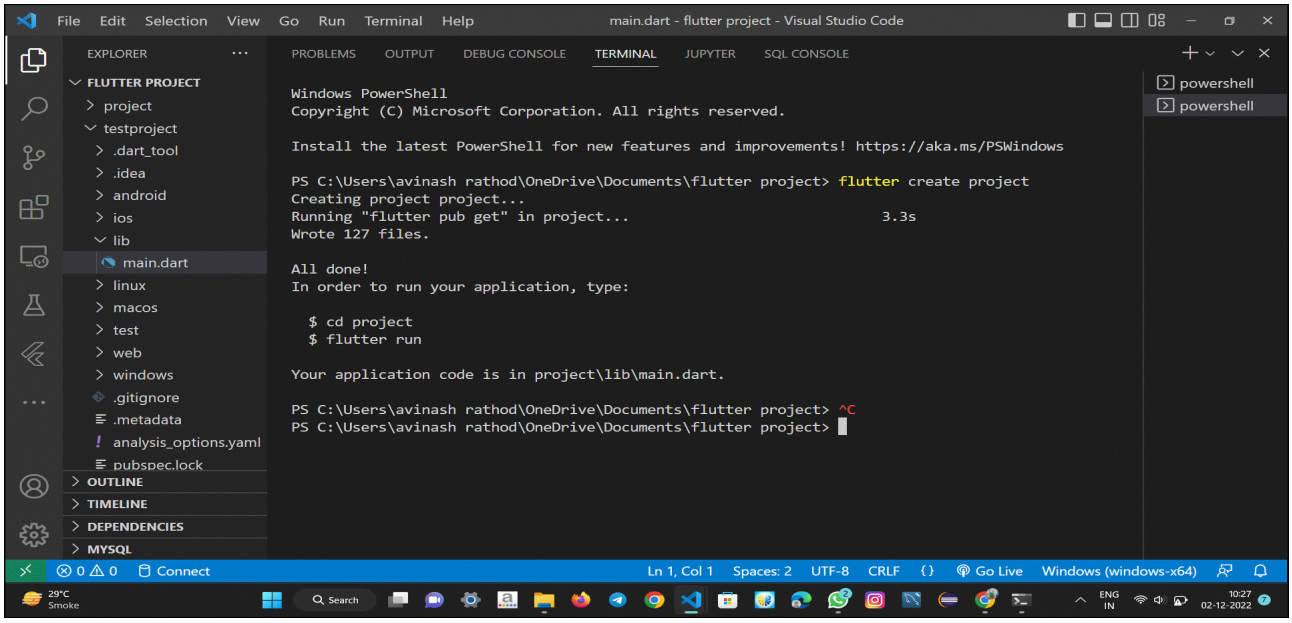
Step 5: Now we have successfully added Flutter and Dart to the Visual studio code, now let's check if flutter is installed or not. For this we will open a new terminal in Visual Studio Code and type the following "***flutter –version",*** if everything is fine then it will normally show the version of the installed flutter.

Now we are ready to create a new flutter project, for that we have to select a directory in which we are going to create the project. Click on the green button of the Open folder and then choose a preferred location.

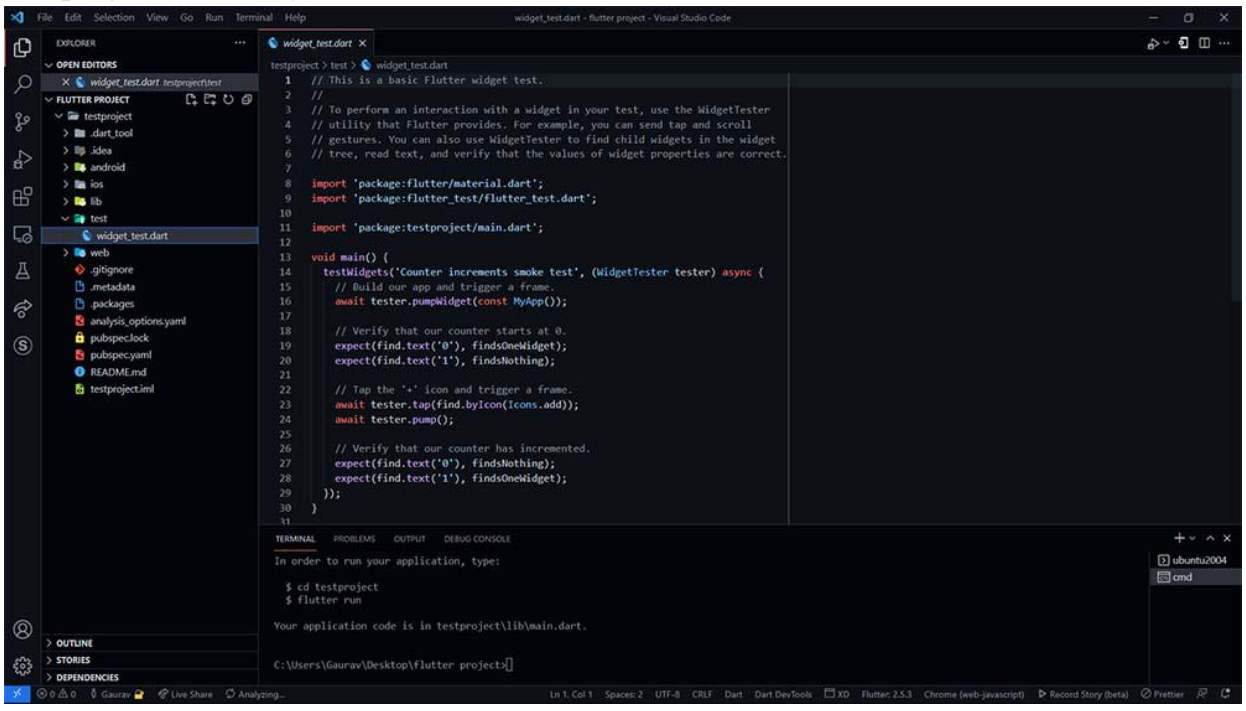


Step 6: open terminal in visual studio menu bar and write the following command

flutter create testproject

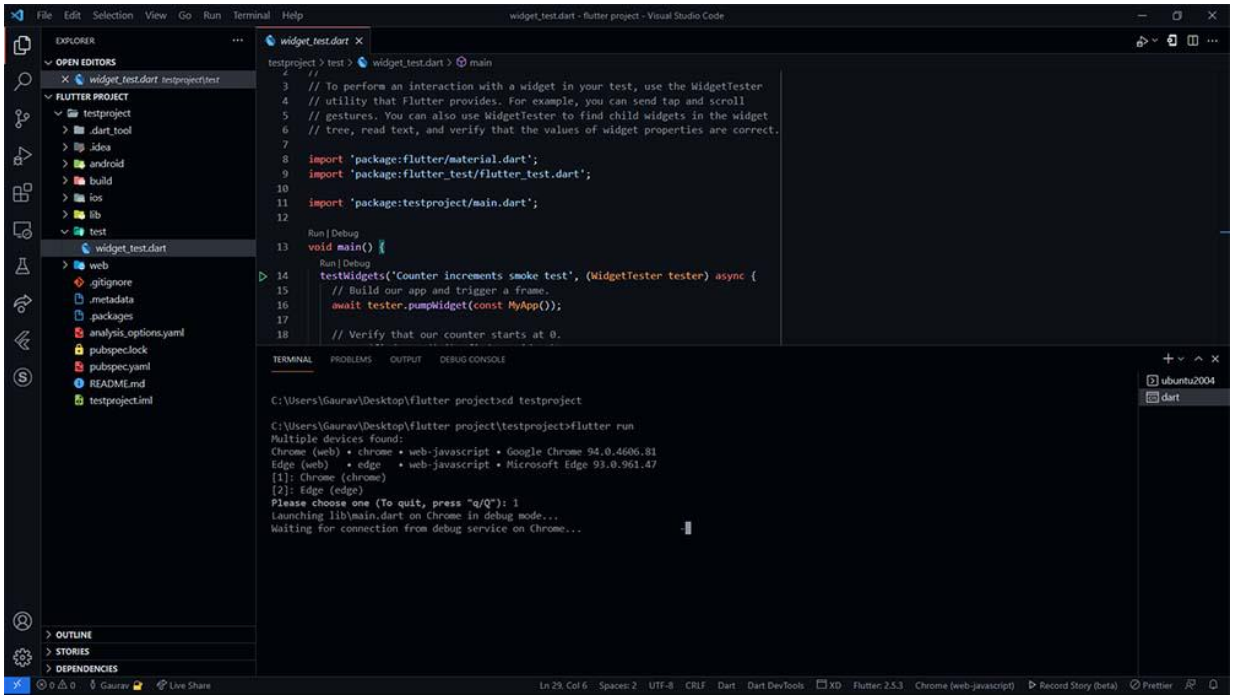


Step 7: Now for creating a new flutter project write the following in the visual studio Code terminal, "***flutter create testproject".*** After that project will be created inside the test project directory.

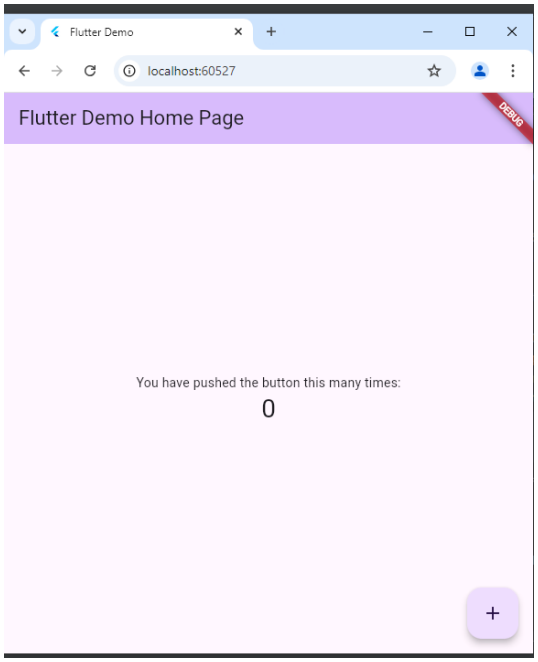


Flutter project is created now we have to run this program in order to check that if it is working or not. Here we need to understand that how a flutter project will show the output. We can run a flutter program in android emulator or we can run this in our browser as well. For running in the android emulator we must have the android studio installed in our system. For this article, we are going to test this program in our browser.

Now before running this program, we have to change our directory to the testproject for that type ***cd testproject,*** and now run the program by typing ***flutter run,*** after this command, it will ask where would you like to see the output, now choose for the desired browser by typing 1 or 2.



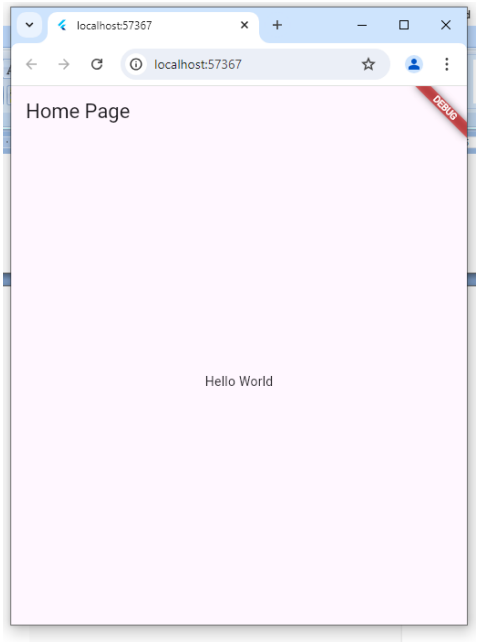
Output: Now we are ready to see the output, by default it has a program in which there is a button by clicking on the that a counter will be displayed in the center.



Week 2: Create an application using Flutter to print hello world.

```
/*  Flutter hello world app */
import 'package:flutter/material.dart';
void main()
{
 runApp(const MyApp());
}

class MyApp extends StatelessWidget
{
 const MyApp({Key? key}) : super(key: key);
 @override
 Widget build(BuildContext context)
 {
  // Material App
  return MaterialApp(
   // Scaffold Widget
    home: Scaffold(
     appBar: AppBar(
      // AppBar takes a Text Widget in it's title parameter
      title: const Text('Home Page'),
     ),
     body: const Center(child: Text('Hello World')),
    ));
 }
}
```

OUTPUT:


localhost:57367
localhost:57367
Home Page
DEBUG
Hello World

Week 3:

Create an application to implement Decision making and loops using Dart.

Decision Making and Loops

The **decision-making** is a feature that allows you to evaluate a condition before the instructions are executed. The Dart language supports the following types of decision-making statements:

- If statement
- If-else statement
- Switch statement

Loops are used to execute a block of code repeatedly until a specified condition becomes true. Dart language supports the following types of loop statements:

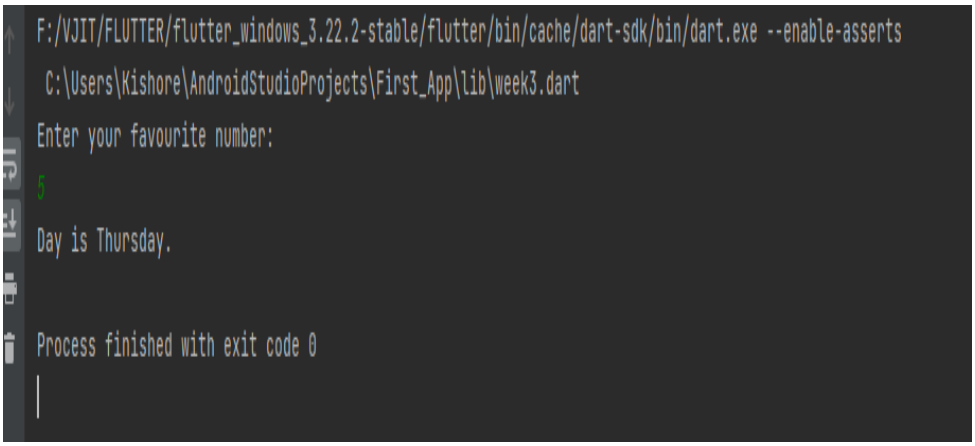
- for
- for..in
- while
- do..while

1. If..else statement

```
import 'dart:io';
void main()
{
   print("Enter your favourite number:");
  int? dayOfWeek = int.parse(stdin.readLineSync()!);
  if (dayOfWeek == 1)
{
   print("Day is Sunday.");
  }
  else if (dayOfWeek == 2)
 {
   print("Day is Monday.");
  }
  else if (dayOfWeek == 3)
{
   print("Day is Tuesday.");
  }
  else if (dayOfWeek == 4)
{
   print("Day is Wednesday.");
  }
```

```
  else if (dayOfWeek == 5)
 {
    print("Day is Thursday.");
  }
  else if (dayOfWeek == 6)
 {
    print("Day is Friday.");
  }
  else if (dayOfWeek == 7)
 {
    print("Day is Saturday.");
  }else
 {
    print("Invalid Weekday.");
  }
 }
```

OUTPUT:



2. Switch case example

```
import 'dart:io';
void main()
{
  print("Enter your favourite number:");
  int? dayOfWeek = int.parse(stdin.readLineSync()!);
  switch (dayOfWeek)
 {
    case 1:
      print("Day is Sunday.");
      break;
    case 2:
      print("Day is Monday.");
```

```
      break;
    case 3:
      print("Day is Tuesday.");
      break;
    case 4:
      print("Day is Wednesday.");
      break;
    case 5:
      print("Day is Thursday.");
      break;
    case 6:
      print("Day is Friday.");
      break;
    case 7:
      print("Day is Saturday.");
      break;
    default:
      print("Out of range");
      break;
  }
}
```

OUTPUT:

Loops

1) For loop

```
import 'dart:io';

 void main()
 {
  print("Enter your favourite number:");
  int? n = int.parse(stdin.readLineSync()!);
  print("Multiplication Table of: $n");
  for(int i=1;i<=10;i++)
  {
   print(n.toString()+'*'+i.toString()+'='+(i*n).toString());
  }
}
```

OUTPUT:

2. For each loop

```
void main()
 {
  var list1 = [10,20,30,40,50];
  // create an integer variable
  int sum = 0;
  print("Dart for..in loop Example");

  for(var i in list1)
  {
    // Each element of iterator and added to sum variable.
    sum = i+ sum;
```

```
  }
  print("The sum is : ${sum}");
 }
```

OUTPUT:

3) While loop

```
void main()
{

  int total = 0;
  int n = 10; // change as per required
  int i =1;
print("Display sum of $n natural numbers");
  while(i<=n)
{
    total = total + i;
    i++;
  }
  print("Total is $total");
}
```

OUTPUT:

4) Do..while loop

```
void main()
{
  int total = 0;
  int n = 10; // change as per required
  int i =1;
print("Display sum of $n natural numbers");
  do
{
    total = total + i;
    i++;
  }while(i<=n);
  print("Total is $total");
}
```

OUTPUT:

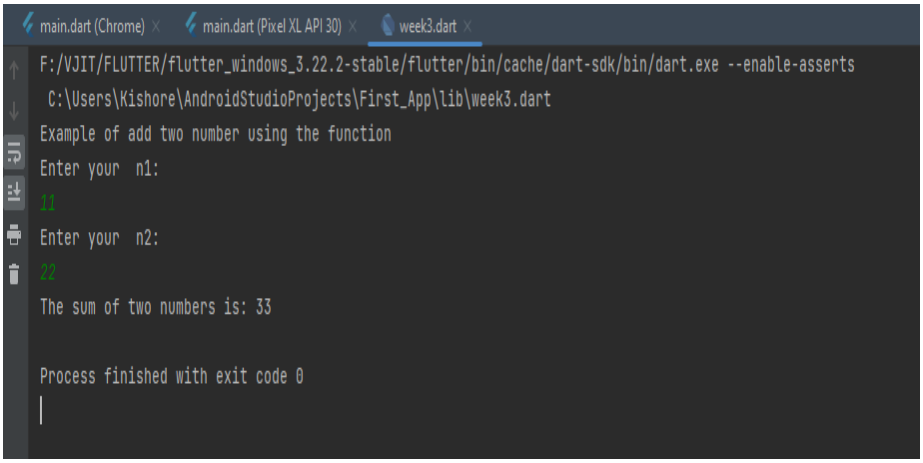
Week 4:

Create an application to demonstrate user defined functions using Dart.

Function Example

```
import 'dart:io';
// Creating a Function
int sum(int a, int b)
{
  // function Body
  int result;
  result = a+b;
  return result;
}
void main()
{
  print("Example of add two number using the function");
  print("Enter your  n1:");
  int? n1 = int.parse(stdin.readLineSync()!);
  print("Enter your  n2:");
  int? n2 = int.parse(stdin.readLineSync()!);

  // We are calling a function and storing a result in variable c
  var c = sum(n1,n2);
  print("The sum of two numbers is: $c");

}
```

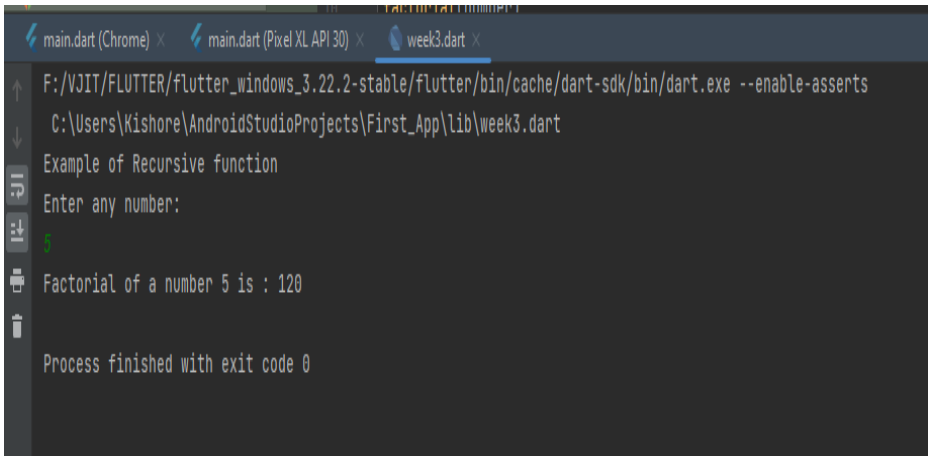
OUTPUT:



Recursive Function Example

```
import 'dart:io';
void main()
{
  print("Example of Recursive function");
  print("Enter any number:");
  int? n = int.parse(stdin.readLineSync()!);
  var c=factorial(n);
  print("Factorial of a number $n is : $c");
}
factorial(number)
{
  if (number <= 0)
{
    // termination case
    return 1;
  }
else
{
    return (number * factorial(number - 1)); // function calls itself

      }
}
```

OUTPUT:



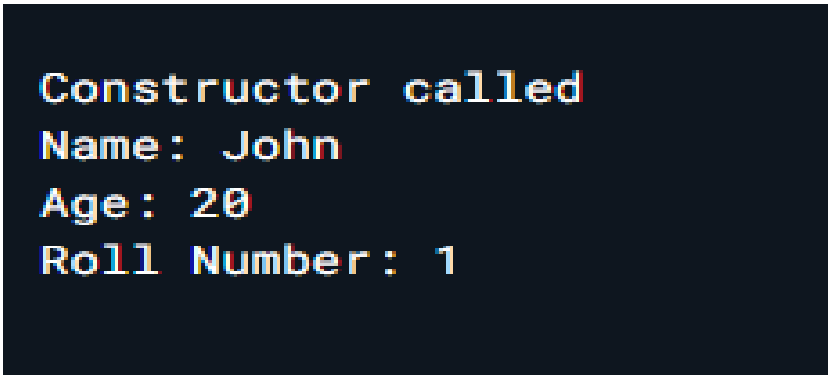
Week 5:

Create an application to implement object oriented programming using Dart

//Constuctors Example

```
class Student
{
  String? name;
  int? age;
  int? rollNumber;

  // Constructor
  Student(String name, int age, int rollNumber)
 {
   // this is for checking the constructor is called or not.
    print( "Constructor called");
   this.name = name;
    this.age = age;
    this.rollNumber = rollNumber;
  }
}
void main()
{
  // Here student is object of class Student.
  Student student = Student("John", 20, 1);
  print("Name: ${student.name}");
  print("Age: ${student.age}");
  print("Roll Number: ${student.rollNumber}");
}
```

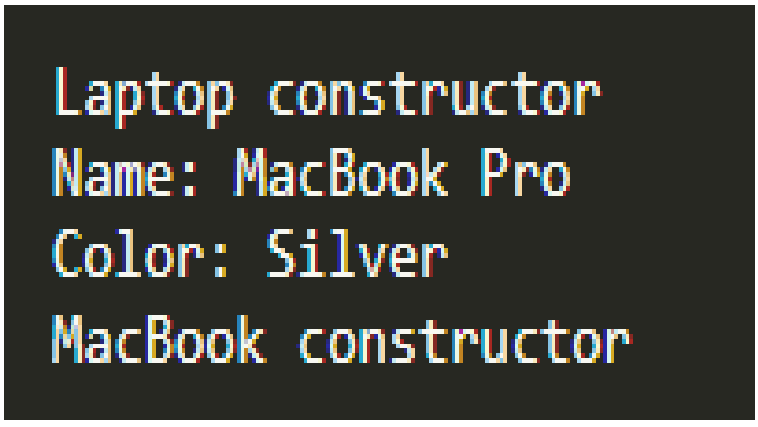
OUTPUT:



Inheritance example

```
class Laptop
{
  // Constructor
  Laptop(String name, String color)
 {
    print("Laptop constructor");
    print("Name: $name");
    print("Color: $color");
  }
}

class MacBook extends Laptop
{
  // Constructor
  MacBook(String name, String color) : super(name, color)
{
    print("MacBook constructor");
  }
}

void main()
{
  var macbook = MacBook("MacBook Pro", "Silver");
}
```

OUTPUT:



Interface Example

```
import 'dart:io';
// abstract class as interface
abstract class Person
{
  // properties
  String? name;
  // abstract method
  void run();
  void walk();
}
class Student implements Person
 {
  // properties
  String? name;
  // implementation of run()
  @override
  void run()
{
    print('Student is running');
  }
  // implementation of walk()
  @override
  void walk()
{
    print('Student is walking');
  }
}
void main()
{
  var student = Student();
  print("Enter your name?");
  // Reading name from keyboard
  String? name = stdin.readLineSync(); // null safety in name string
  student.name = name;
  print('Student Name:${student.name}');
  student.run();
  student.walk();
}
```

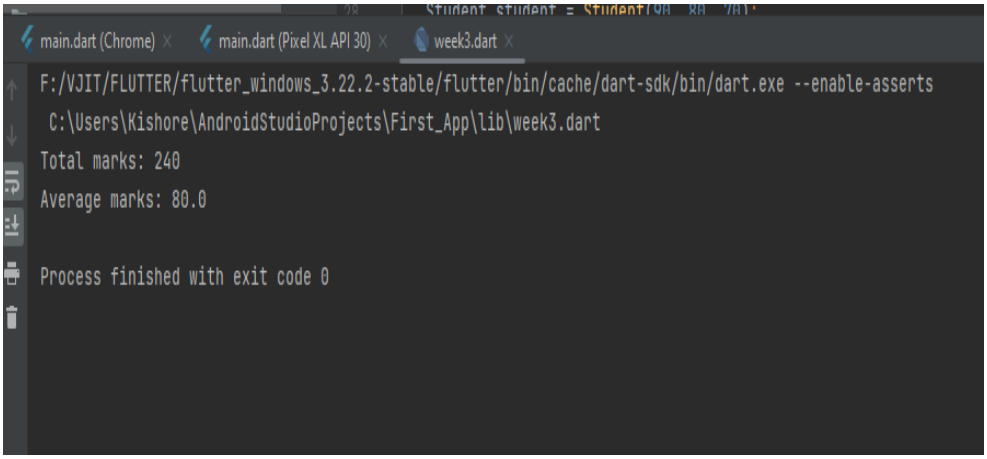
OUTPUT:

Multiple Interfaces Example

```
// abstract class as interface
abstract class CalculateTotal
 {
  int total();
}
// abstract class as interface
abstract class CalculateAverage
 {
  double average();
}
// implements multiple interfaces
class Student implements CalculateTotal, CalculateAverage
{
// properties
  int marks1, marks2, marks3;
// constructor
  Student(this.marks1, this.marks2, this.marks3);
// implementation of average()
  @override
  double average()
 {
    return total() / 3;
  }
// implementation of total()
  @override
  int total()
{
    return marks1 + marks2 + marks3;
  }
}
```

```
void main()
{
  Student student = Student(90, 80, 70);
  print('Total marks: ${student.total()}');
  print('Average marks: ${student.average()}');
}
```

OUTPUT:

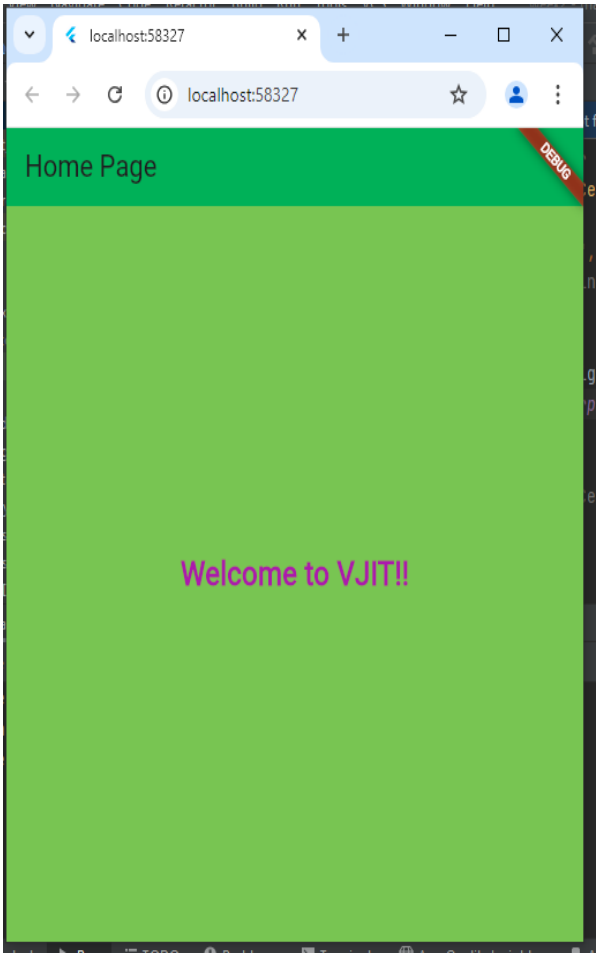


Week 6: Create an application for platform basic widgets (Text, Image, and Icon).

/* Flutter Text Widget app */

```
import 'package:flutter/material.dart';
void main()
{
 runApp(const MyApp());
}

class MyApp extends StatelessWidget
{
 const MyApp({Key? key}) : super(key: key);
 @override
 Widget build(BuildContext context)
 {
  // Material App
  return MaterialApp(
   // Scaffold Widget
    home: Scaffold(
     backgroundColor: Colors.lightGreen,
     appBar: AppBar(
      backgroundColor: Colors.green,
      // AppBar takes a Text Widget in it's title parameter
      title: const Text('Home Page'),
     ),
     body: const Center(// Display a centered text widget
  child: Text(
  "Welcome to VJIT!!",
  // Apply text styling
  style: TextStyle(
  fontSize: 24,         // Set font size
  fontWeight: FontWeight.bold, // Set font weight
   color: Colors.purple,
  ),
  ),
       ),  ));
 }
}
```

OUTPUT:


localhost:58327
localhost:58327
Home Page
DEBUG
Welcome to VJIT!!

Display an image in Flutter

To display an image in Flutter, do the following steps:

Step 1: First, we need to create a new **folder** inside the root of the Flutter project and named it assets. We can also give it any other name if you want.

Step 2: Next, inside this folder, add one image manually.

Step3: Update the **pubspec.yaml** file. Suppose the image name is VJIT_logo_2023.png', then

pubspec.yaml file is:

```
assets:
  - assets/VJIT_logo_2023.png'
```

```
import 'package:flutter/material.dart';

void main() => runApp(MyApp());

class MyApp extends StatelessWidget
{
  @override
  Widget build(BuildContext context)
 {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Flutter Image Demo'),
        ),
        body: Center(
          child: Column(
            children: <Widget>[
              Image.asset('assets/VJIT_logo_2023.png'),
              Text(
                'Welcome to VJIT::Hyd',
                style: TextStyle(fontSize: 20.0),
              )
            ],
          ),
        ),
      ),
    );
  }
}
```

OUTPUT:


localhost:61019
localhost:61019
Flutter Image Demo
DEBUG
VIDYA JYOTHI
INSTITUTE OF TECHNOLOGY
AN AUTONOMOUS INSTITUTION
Welcome to VJIT::Hyd

Display an Icons in Flutter

An icon is a **graphic image** representing an application or any specific entity containing meaning for the user. It can be selectable and non-selectable. **For example**, the company's logo is non-selectable. Sometimes it also contains a **hyperlink** to go to another page. It also acts as a sign in place of a detailed explanation of the actual entity.

Flutter provides an **Icon Widget** to create icons in our applications. We can create icons in Flutter, either using inbuilt icons or with the custom icons. Flutter provides the list of all icons in the **Icons class**. In this article, we are going to learn how to use Flutter icons in the application.

Icon Widget Properties

Flutter icons widget has different properties for customizing the icons. These properties are explained below:

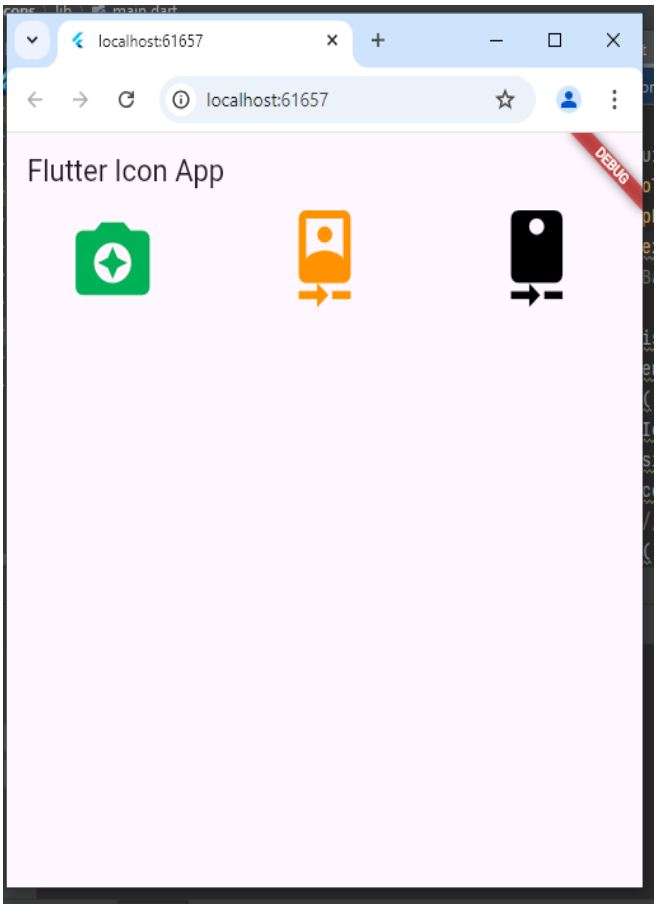
Property	Descriptions
icon	It is used to specify the icon name to display in the application. Generally, Flutter uses material design icons that are symbols for common actions and items.
color	It is used to specify the color of the icon.
size	It is used to specify the size of the icon in pixels. Usually, icons have equal height and width.
textDirection	It is used to specify to which direction the icon will be rendered.

Program:

```
import 'package:flutter/material.dart';

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      home: MyIconPage(),
    );
```

```
  }
}

class MyIconPage extends StatefulWidget
 {
  @override
  _MyIconPageState createState() => _MyIconPageState();
}

class _MyIconPageState extends State<MyIconPage>
{
  @override
  Widget build(BuildContext context)
{
    return Scaffold(
      appBar: AppBar(
        title: Text('Flutter Icon App'),
      ),
      body: Row(
          mainAxisAlignment: MainAxisAlignment.spaceAround,
          children: <Widget>[
            Icon(
                Icons.camera_enhance,
                size: 70,
                color:Colors.green
            ),
            Icon(
                Icons.camera_front,
                size: 70,
                color:Colors.orange
            ),
            Icon(
                Icons.camera_rear,
                size: 70,
                color:Colors.black
            ),
          ]),
    );
  }
}
```

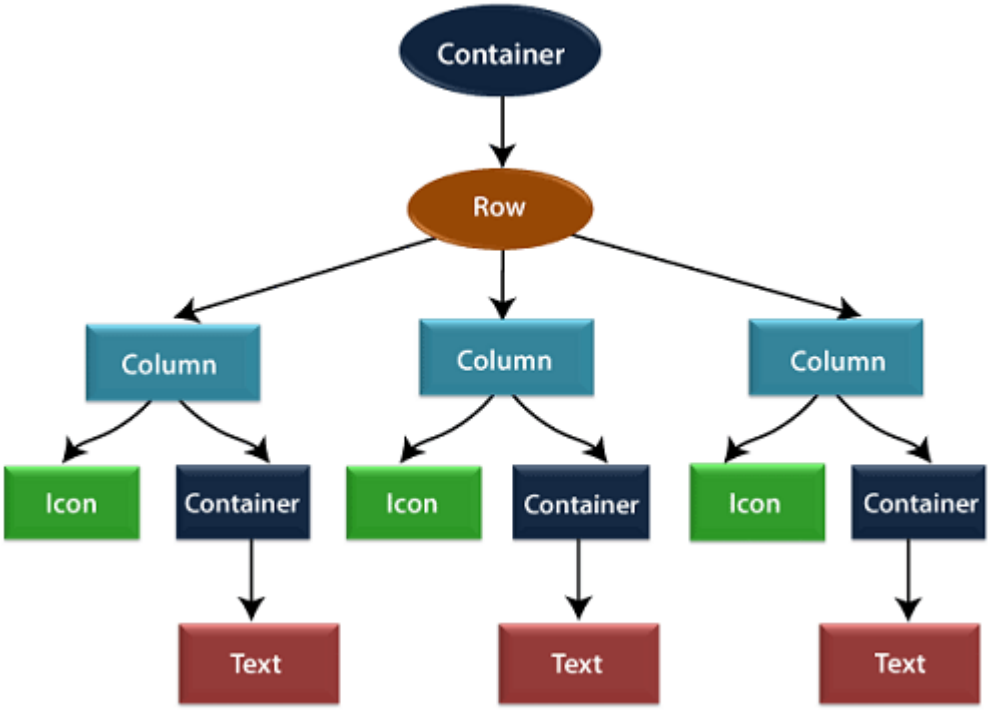
OUTPUT:



Week 7:

Create an application for Layout widgets (Single child, Multiple Child).

Flutter Layouts



Layout a widget

Let us learn how we can create and display a simple widget.

The following steps show how to layout a widget:

Step 1: First, you need to select a Layout widget.

Step 2: Next, create a visible widget.

Step 3: Then, add the visible widget to the layout widget.

Step 4: Finally, add the layout widget to the page where you want to display.

Types of Layout Widgets

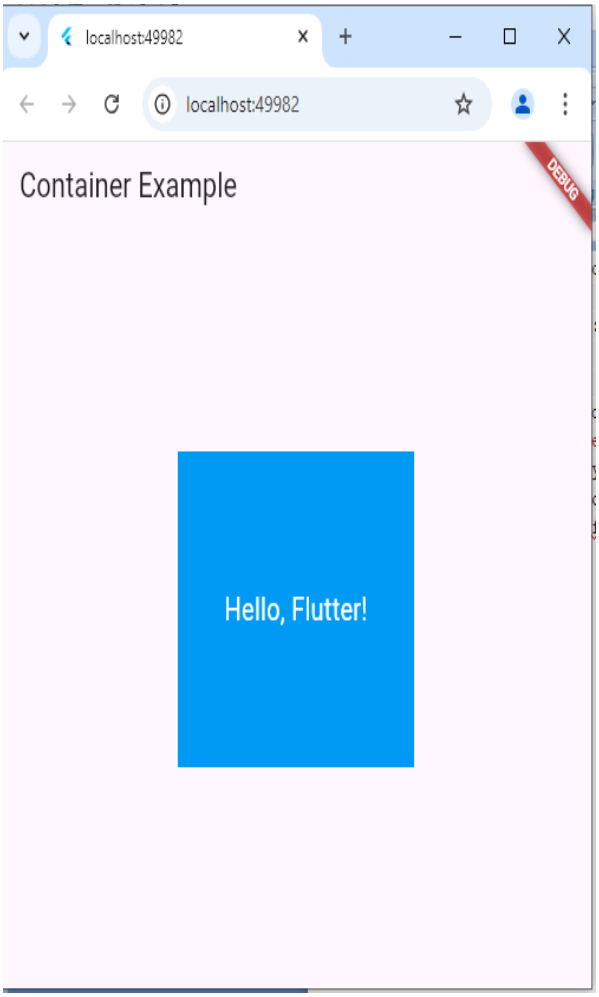
We can categories the layout widget into two types:

1. Single Child Widget
2. Multiple Child Widget

Display Container Widget as Single Child

```
import 'package:flutter/material.dart';

void main()
{
  runApp(MyApp());
}

class MyApp extends StatelessWidget
 {
  @override
  Widget build(BuildContext context)
 {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Container Example'),
        ),
        body: Center(
          child: Container(
            width: 200,
            height: 200,
            color: Colors.blue,
            child: Center(
              child: Text(
                'Hello, Flutter!',
                style: TextStyle(
                  color: Colors.white,
                  fontSize: 20,
                ),
              ),
            ),
          ),
        ),
      ),
    );
  }
}
```

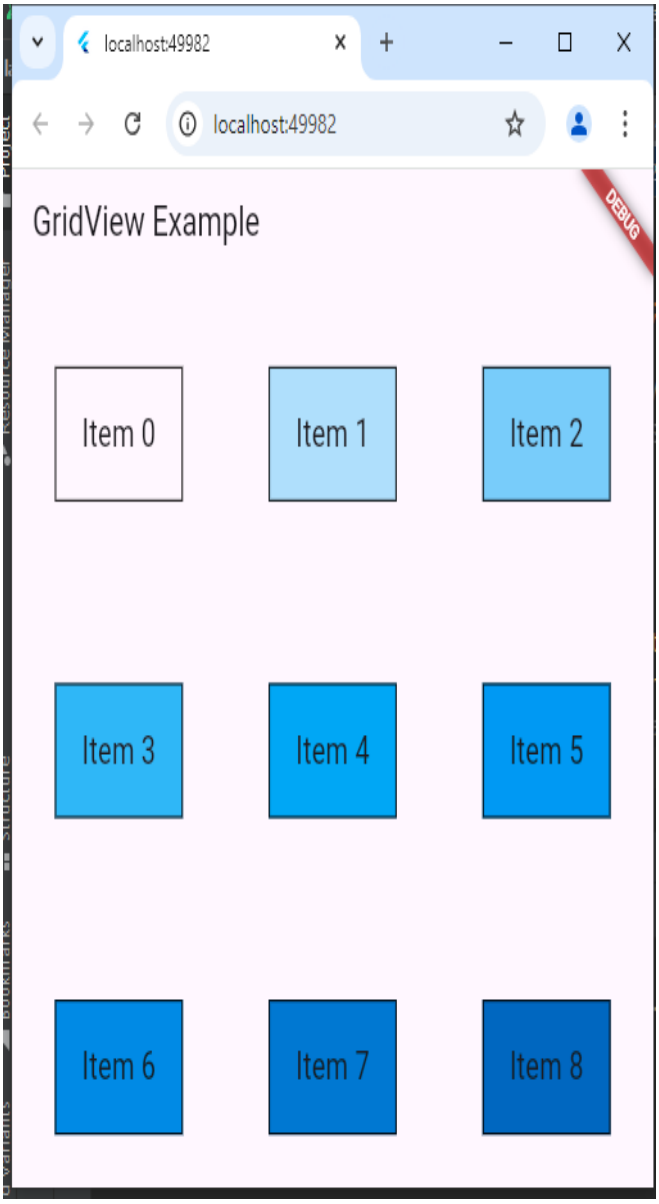
OUTPUT:


localhost:49982
localhost:49982
Container Example
DEBUG
Hello, Flutter!

Display Multi Child Widgets using Grid View

```
import 'package:flutter/material.dart';
class GridViewExample extends StatelessWidget
 {
  @override
  Widget build(BuildContext context)
{
    return Scaffold(
      appBar: AppBar(
        title: Text('GridView Example'),
      ),
      body: GridView.count(
        crossAxisCount: 3, // Number of columns in the grid
        children: List.generate(9, (index) {
          // Generate 9 grid items
          return Center(
            child: Container(
              decoration: BoxDecoration(
                color: Colors.blue[100 * (index % 9)],
                border: Border.all(),
              ),
              padding: EdgeInsets.all(20.0),
              child: Text(
                'Item $index',
                style: TextStyle(fontSize: 20.0),
              ),
            ),
          );
        }),
      ),
    );
  }
}
void main()
{
  runApp(MaterialApp(
    home: GridViewExample(),
  ));}
```

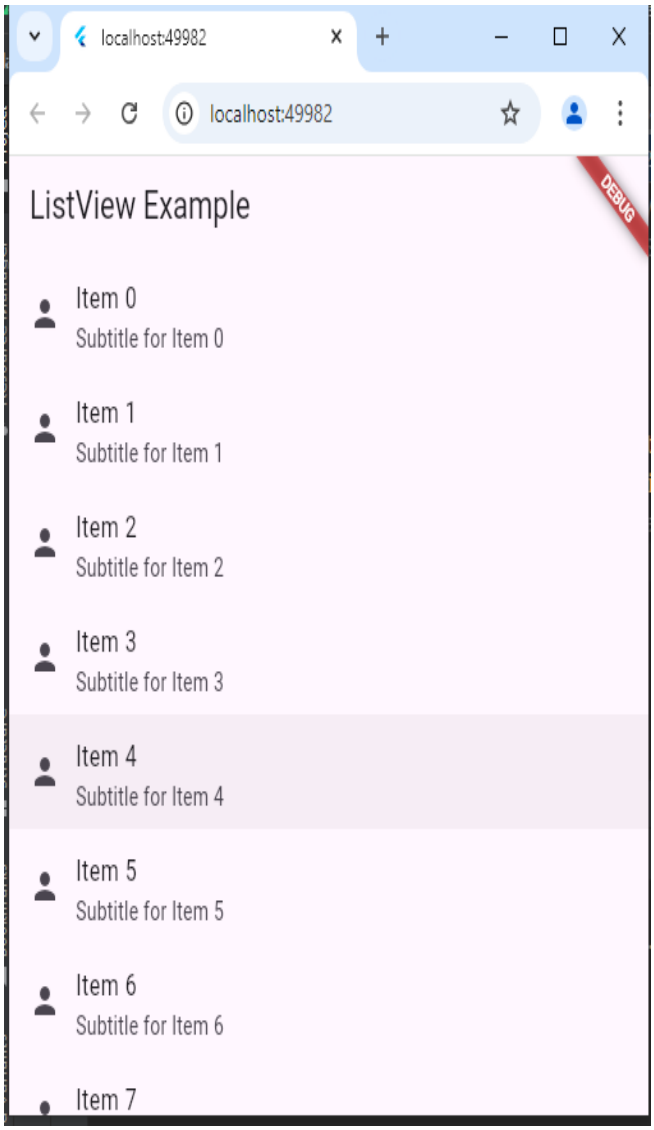
OUTPUT


localhost:49982
localhost:49982
GridView Example
DEBUG
Item 0
Item 1
Item 2
Item 3
Item 4
Item 5
Item 6
Item 7
Item 8

Display Multi Child Widgets using List View

```
import 'package:flutter/material.dart';
class ListViewExample extends StatelessWidget
{
  @override
  Widget build(BuildContext context)
 {
    return Scaffold(
      appBar: AppBar(
        title: Text('ListView Example'),
      ),
      body: ListView(
        children: List.generate(20, (index) {
          // Generate 20 list items
          return ListTile(
            leading: Icon(Icons.person),
            title: Text('Item $index'),
            subtitle: Text('Subtitle for Item $index'),
            onTap: () {
              // Action when item is tapped
              print('Item $index tapped');
            },
          );
        }),
      ),
    );
  }
}
void main()
{
  runApp(MaterialApp(
    home: ListViewExample(),
  ));
}
```

OUTPUT


localhost:49982
localhost:49982
ListView Example
DEBUG
Item 0
Subtitle for Item 0
Item 1
Subtitle for Item 1
Item 2
Subtitle for Item 2
Item 3
Subtitle for Item 3
Item 4
Subtitle for Item 4
Item 5
Subtitle for Item 5
Item 6
Subtitle for Item 6
Item 7

Week 8:

Create an application to demonstrate Gesture Detector.

main.dart

```
import 'package:flutter/material.dart';
import 'package:gesture2/home.dart';

void main()
 {
  runApp(MyApp());
}

class MyApp extends StatelessWidget
 {
  // This widget is the root of your application.
  @override
  Widget build(BuildContext context)
 {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Gesture Detector',
      home: GestureDetectorScreen(),
      // home: MyHomePage(),
    );
  }
}
```

home.dart

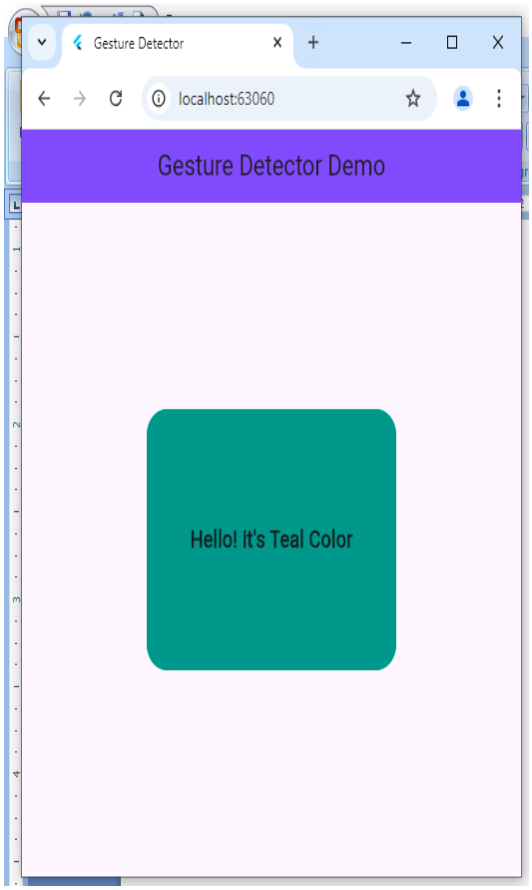
```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';

class GestureDetectorScreen extends StatefulWidget
 {

  @override
  _GestureDetectorScreenState createState() => _GestureDetectorScreenState();
}

class _GestureDetectorScreenState extends State<GestureDetectorScreen>
```

```
{
  Color color = Colors.teal;
  Text text = Text("Hello! it's Teal Color",
    style: TextStyle(
        fontSize: 18,
        fontWeight: FontWeight.bold
    ),);

  @override
  Widget build(BuildContext context)
 {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Colors.deepPurpleAccent,
        centerTitle: true,
        title: Text('Gesture Detector Demo'),
      ),
      body: Center(
        child: GestureDetector(
          onTap: ()
          {
            setState(() {
              color = Colors.amber;
              text = Text("Hello! it's Amber Color",
                style: TextStyle(
                    fontSize: 18,
                    fontWeight: FontWeight.bold
                ),);
            });
          },

          onDoubleTap: ()
          {
            setState(() {
              color = Colors.blueAccent;
              text = Text("Hello! it's Blue Color",
                style: TextStyle(
                    fontSize: 18,
                    fontWeight: FontWeight.bold
```

```
            ),);
          });
        },

        onLongPress: ()
        {
          setState(() {
            color = Colors.deepOrangeAccent;
            text = Text("Hello! it's Orange Color",
              style: TextStyle(
                  fontSize: 18,
                  fontWeight: FontWeight.bold
              ),);
          });
        },


        child: Container(
            decoration: BoxDecoration(
                color: color,
                borderRadius: BorderRadius.circular(20)
            ),
            width: 250,
            height: 200,
            child: Center(
              child: text,
            )),
      ),
    ),
  );
 }
}
```

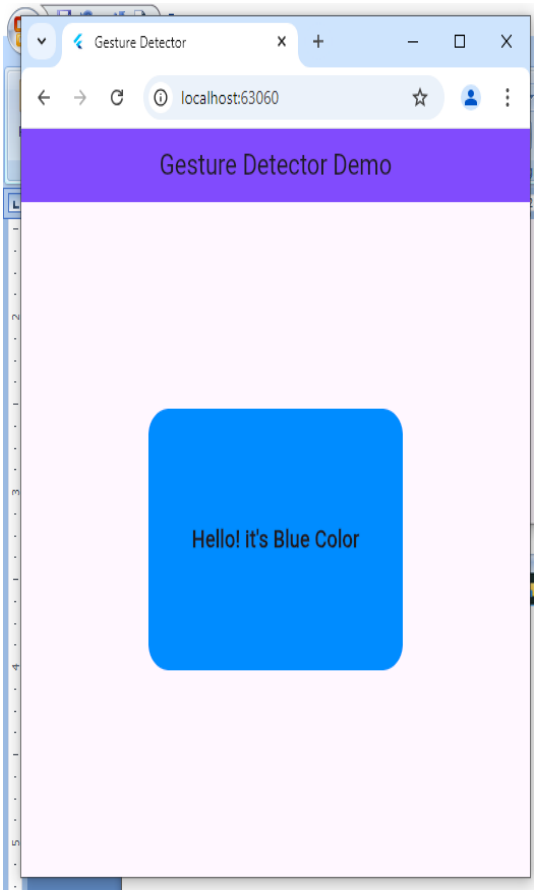
OUTPUT:

Application Loading

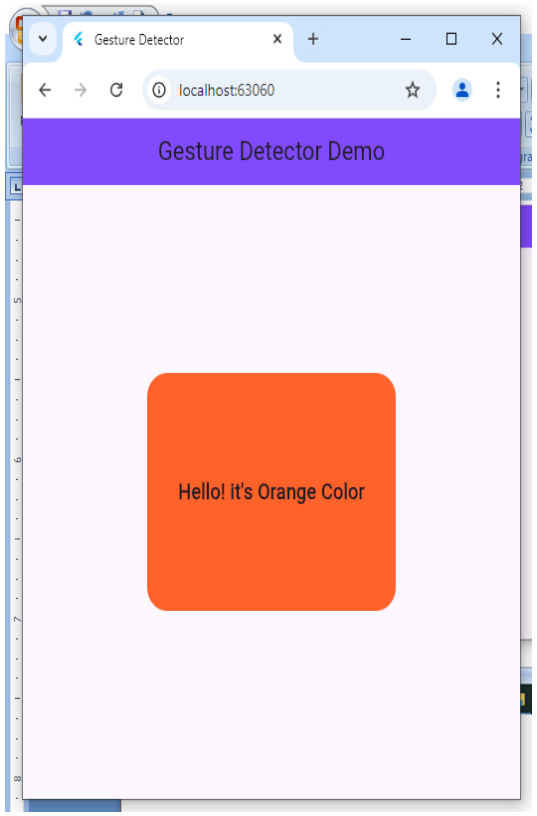


When apply Single Click

When apply Double Click



When apply long press



Week 9 & 10:

Create an application for Registration form.

Creating Form

Flutter provides a **Form widget** to create a form. The form widget acts as a container, which allows us to group and validate the multiple form fields. When you create a form, it is necessary to provide the **GlobalKey**. This key uniquely identifies the form and allows you to do any validation in the form fields.

The form widget uses child widget **TextFormField** to provide the users to enter the text field. This widget renders a material design text field and also allows us to display validation errors when they occur.

Let us create a form. First, create a Flutter project. Inside the class, we define a global key as **_formKey**. This key holds a **FormState** and can use to retrieve the form widget. Inside the **build** method of this class, we have added some custom style and use the TextFormField widget to provide the form fields such as name, Email Id, Password fields. Inside the TextFormField, we have used **InputDecoration** that provides the look and feel of your form properties such as borders, labels, icons, hint, styles, etc. Finally, we have added a **button** to submit the form.

main.dart

```
import 'package:flutter/material.dart';
import 'login_screen.dart';
void main()
{
  var app=MaterialApp(
     home:LoginScreen(),
     title:"flutter form"

  );
  runApp(app);
}
```

registration.dart

```
import 'package:flutter/material.dart';
import 'login.dart';
class LoginScreen extends StatefulWidget
 {
  const LoginScreen({super.key});

  @override
  State<LoginScreen> createState() => _LoginScreenState();
}
class _LoginScreenState extends State<LoginScreen>
{
  final _formKey = GlobalKey<FormState>(); // Step 1
  bool isObsecureText = true;
  TextEditingController _nameController = TextEditingController();
  TextEditingController _emailControlle = TextEditingController();
  TextEditingController _passwordController = TextEditingController();
  @override
  Widget build(BuildContext context) {
       return Scaffold(
      appBar: AppBar(
        title: Text("User Registration"),
        backgroundColor: Colors.lightBlue,
      ),
      body: Container(
        margin: EdgeInsets.only(left: 20, right: 20, top: 20),
        width: MediaQuery.of(context).size.width,
        height: MediaQuery.of(context).size.height,
        child: Form(
          key: _formKey,
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            crossAxisAlignment: CrossAxisAlignment.center,
            children: [
              Image.asset("images/login.png"),
              Text("Register Here!",
                  style: TextStyle(fontSize: 32, fontWeight: FontWeight.w600)),
              Column(
```

```
  crossAxisAlignment: CrossAxisAlignment.start,
  children: [
   Text(" Full Name"),
   SizedBox(
    height: 8,
   ),
   TextFormField(
    controller: _nameController,
    keyboardType: TextInputType.name,
    validator: (value) {
     print("Name text $value");
     final inputValue = value ?? "";
     if (inputValue.isEmpty) {
      return "Please Enter Name";
     }  else {
      return null;
     }
    },
    decoration: InputDecoration(
       hintText: "Please Enter Full Name",
       border: OutlineInputBorder(
          borderRadius: BorderRadius.circular(12),
          borderSide: BorderSide(color: Colors.grey))),
   ),
  ],
),
SizedBox(
  height: 10,
),
Column(
  crossAxisAlignment: CrossAxisAlignment.start,
  children: [
   Text(" Email ID"),
   SizedBox(
    height: 8,
   ),
   TextFormField(
    controller: _emailControlle,
    keyboardType: TextInputType.emailAddress,
```

```
      validator: (value) {
        print("email text $value");
        final inputValue = value ?? "";
        if (inputValue.isEmpty) {
          return "Please Enter Email";
        } else if (!checkEmailValidation(inputValue)) {
          return "Please Enter Valid Email";
        } else {
          return null;
        }
      },
      decoration: InputDecoration(
          hintText: "Please Enter Email",
          border: OutlineInputBorder(
              borderRadius: BorderRadius.circular(12),
              borderSide: BorderSide(color: Colors.grey))),
    ),
  ],
),
SizedBox(
  height: 10,
),
Column(
  crossAxisAlignment: CrossAxisAlignment.start,
  children: [
    Text("Password"),
    SizedBox(
      height: 8,
    ),
    TextFormField(
      controller: _passwordController,
      validator: (value) {
        print("password text $value");
        final inputValue = value ?? "";
        if (inputValue.isEmpty) {
          return "Please Enter Password";
        } else if (!validatePassword(inputValue)) {
          return "Please Enter valid Password";
        } else {
```

```
            return null;
          }
        },
        obscureText: isObsecureText,
        obscuringCharacter: "*",
        maxLength: 10,
        decoration: InputDecoration(
          suffixIcon: isObsecureText
              ? GestureDetector(
              onTap: (){
                setState(() {
                  isObsecureText = false;
                });
                print(isObsecureText);
              },
              child: Icon(Icons.visibility_off))
              : GestureDetector(
              onTap: (){
                setState(() {
                  isObsecureText = true;
                });
                print(isObsecureText);
              },
              child: Icon(Icons.visibility)),
          hintText: "Please Enter Password",
          border: OutlineInputBorder(
              borderRadius: BorderRadius.circular(12),
              borderSide: BorderSide(color: Colors.grey))),
      ),
    ],
  ),
  Row(
    mainAxisAlignment: MainAxisAlignment.spaceEvenly,
    children: [
      Expanded(
        child: ElevatedButton(

          onPressed: () {
            _nameController.clear();
```

```
                  _emailControlle.clear();
                  _passwordController.clear();
                },
                style: ButtonStyle(
                  backgroundColor: WidgetStateProperty.all<Color>(Colors.blue),
                   foregroundColor: WidgetStateProperty.all<Color>(Colors.white)
                ),
                child: Text('Reset'),
              ),
            ),
            Expanded(
              child: ElevatedButton(

                onPressed: () {
                  final result = _formKey.currentState!.validate();
                  if(result) {
                    Navigator.push(
                        context,
                        MaterialPageRoute(
                            builder: (context) =>
                                NextPage()));
                  }
                } ,
                style: ButtonStyle(
                  backgroundColor: WidgetStateProperty.all<Color>(Colors.blue), //
Change button color
                    foregroundColor: WidgetStateProperty.all<Color>(Colors.white)
                ),
                child: Text('Register'),
              ),
            ),
            // Add more buttons as needed
          ],
        ),
],        ),
      ),
    ),
  );
 }
```

```
 bool checkEmailValidation(String email)
{
   return RegExp(
      r"^[a-zA-Z0-9.a-zA-Z0-9.!#$%&'*+-/=?^_`{|}~]+@[a-zA-Z0-9]+\.[a-zA-Z]+")
      .hasMatch(email);
 }

 bool validatePassword(String password)
{
   // Regular expression for validating the password
   final RegExp passwordRegExp = RegExp(
      r'^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[@$!%*?&])[A-Za-z\d@$!%*?&]{8,}$');
   // Check if the password matches the regular expression
   return passwordRegExp.hasMatch(password);
 }
 }

class NextPage extends StatelessWidget
{
 const NextPage({super.key});
 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(title: Text('Registration Page'),
    backgroundColor: Colors.lightBlue,
   ),
   body: Center(
    child: Column(
     mainAxisAlignment: MainAxisAlignment.center,
     children: <Widget>[
      Text('Thank you for your Registration with us!!',
       style: TextStyle(
        fontSize: 18,
        fontWeight: FontWeight.bold,
        color: Colors.red,
       ),
      ),
      ElevatedButton(
       onPressed: ()
```

```
            {
              Navigator.push(
                  context,
                  MaterialPageRoute(
                      builder: (context) =>
                          Login()));
            },
            style: ButtonStyle(
               backgroundColor: WidgetStateProperty.all<Color>(Colors.blue),
               foregroundColor: WidgetStateProperty.all<Color>(Colors.white)
            ),
            child: const Text('Login Here!'),
          ),
        ],
      ),
    ),
  );
 }
}
```

login.dart

```
import 'package:flutter/material.dart';

class Login extends StatefulWidget
 {
  const Login({super.key});

  @override
  State<Login> createState() => _LoginState();
}

class _LoginState extends State<Login>
{
  final _formKey = GlobalKey<FormState>(); // Step 1
  bool isObsecureText = true;
  TextEditingController _emailControlle = TextEditingController();
  TextEditingController _passwordController = TextEditingController();
  @override
  Widget build(BuildContext context) {
```

```
return Scaffold(
  appBar: AppBar(
    title: Text("User Login"),
    backgroundColor: Colors.lightBlue,
  ),
  body: Container(
    margin: EdgeInsets.only(left: 20, right: 20, top: 20),
    width: MediaQuery.of(context).size.width,
    height: MediaQuery.of(context).size.height,
    child: Form(
      key: _formKey,
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        crossAxisAlignment: CrossAxisAlignment.center,
        children: [
          Image.asset("images/login.png"),
          Text("Login Here!",
              style: TextStyle(fontSize: 32, fontWeight: FontWeight.w600)),
          SizedBox(
            height: 30,
          ),
          Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              Text(" Email ID"),
              SizedBox(
                height: 8,
              ),
              TextFormField(
                controller: _emailControlle,
                keyboardType: TextInputType.emailAddress,
                validator: (value) {
                  print("email text $value");
                  final inputValue = value ?? "";
                  if (inputValue.isEmpty) {
                    return "Please Enter Email";
                  } else if (!checkEmailValidation(inputValue)) {
                    return "Please Enter Valid Email";
                  } else {
```

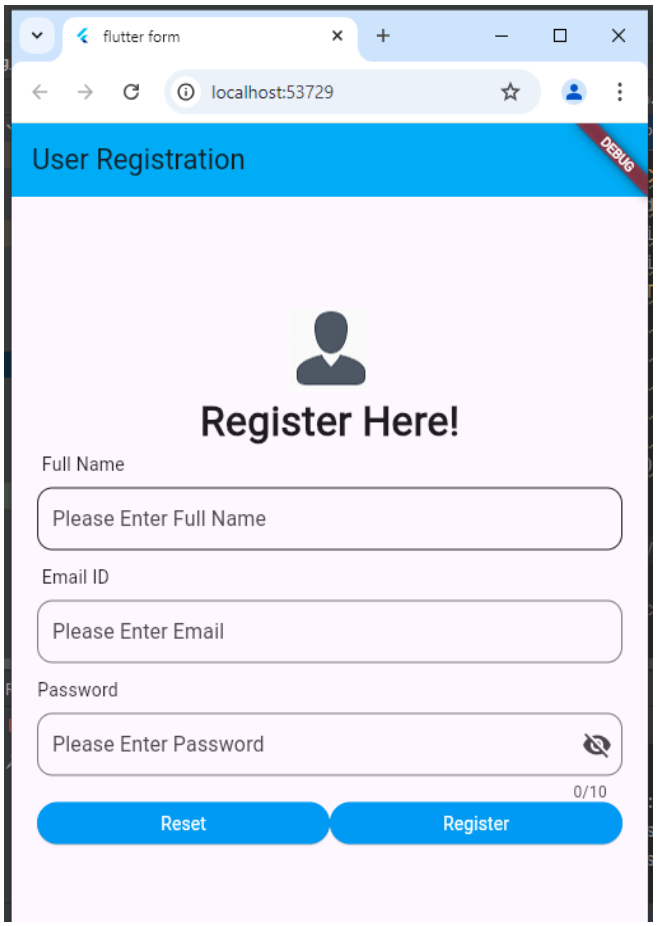
```
            return null;
          }
        },
        decoration: InputDecoration(
            hintText: "Please Enter Email",
            border: OutlineInputBorder(
                borderRadius: BorderRadius.circular(12),
                borderSide: BorderSide(color: Colors.grey))),
      ),
    ],
  ),
  SizedBox(
    height: 30,
  ),
  Column(
    crossAxisAlignment: CrossAxisAlignment.start,
    children: [
      Text("Password"),
      SizedBox(
        height: 8,
      ),
      TextFormField(
        controller: _passwordController,
        validator: (value) {
          print("password text $value");
          final inputValue = value ?? "";
          if (inputValue.isEmpty) {
            return "Please Enter Password";
          } else if (!validatePassword(inputValue)) {
            return "Please Enter valid Password";
          } else {
            return null;
          }
        },
        obscureText: isObsecureText,
        obscuringCharacter: "*",
        maxLength: 10,
        decoration: InputDecoration(
            suffixIcon: isObsecureText
```

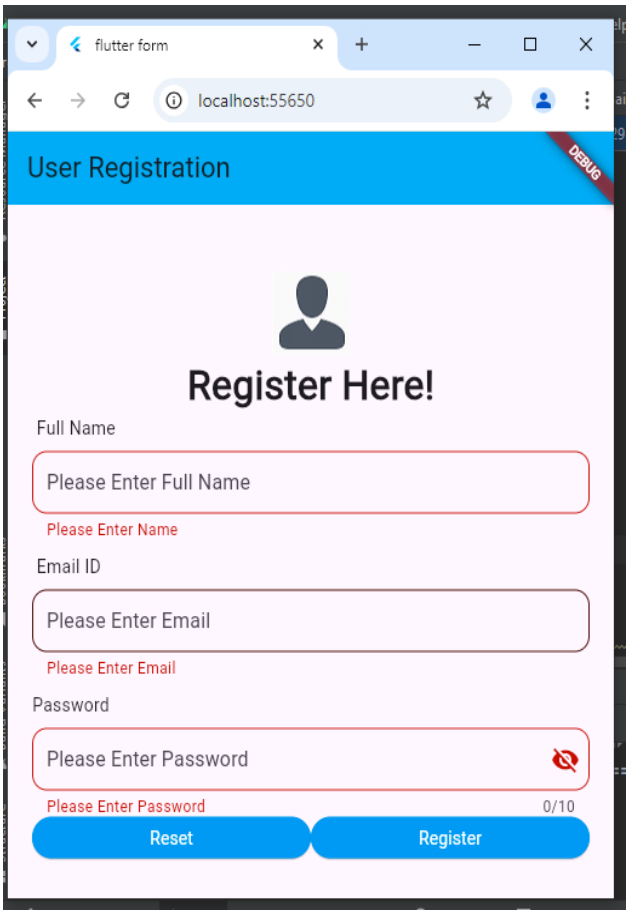
```
                ? GestureDetector(
                onTap: (){
                  setState(() {
                    isObsecureText = false;
                  });
                  print(isObsecureText);
                },
                child: Icon(Icons.visibility_off))
                : GestureDetector(
                onTap: (){
                  setState(() {
                    isObsecureText = true;
                  });

                  print(isObsecureText);
                },
                child: Icon(Icons.visibility)),
            hintText: "Please Enter Password",
            border: OutlineInputBorder(
                borderRadius: BorderRadius.circular(12),
                borderSide: BorderSide(color: Colors.grey))),
      ),
    ],
  ),
  SizedBox(
    width: MediaQuery.of(context).size.width * 0.8,
    child: TextButton(
        style: TextButton.styleFrom(
          backgroundColor: Color.fromRGBO(69, 182, 74, 1),
          shape: RoundedRectangleBorder(
            borderRadius:
            BorderRadius.circular(8), // Rounded corners
          ),
        ),
        onPressed: () {
          final result = _formKey.currentState!.validate();
         if(result) {
           Navigator.push(
               context,
```

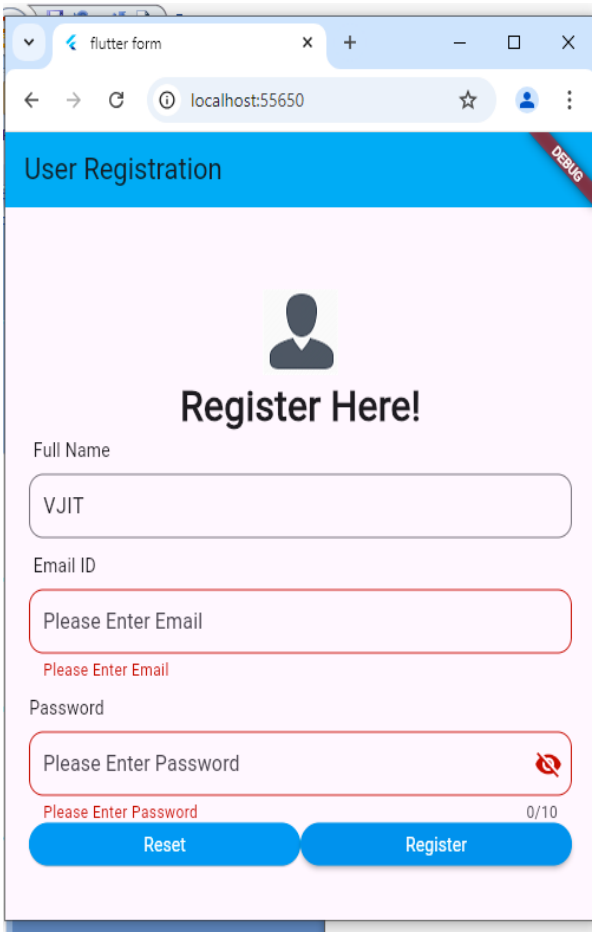
```
                    MaterialPageRoute(
                        builder: (context) =>
                            NextPage()));
                  }
                },
                child: Text(
                  "Login",
                  style: TextStyle(
                      fontSize: 18,
                      fontWeight: FontWeight.w600,
                      color: Colors.white),
                )),
            )
          ],
        ),
      ),
    ),
  );
 }

 bool checkEmailValidation(String email)
{
   return RegExp(
       r"^[a-zA-Z0-9.a-zA-Z0-9.!#$%&'*+-/=?^_`{|}~]+@[a-zA-Z0-9]+\.[a-zA-Z]+")
       .hasMatch(email);
 }

 bool validatePassword(String password)
{
   // Regular expression for validating the password
   final RegExp passwordRegExp = RegExp(
       r'^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[@$!%*?&])[A-Za-z\d@$!%*?&]{8,}$');

   // Check if the password matches the regular expression
   return passwordRegExp.hasMatch(password);
 }
}
class NextPage extends StatelessWidget
{
```

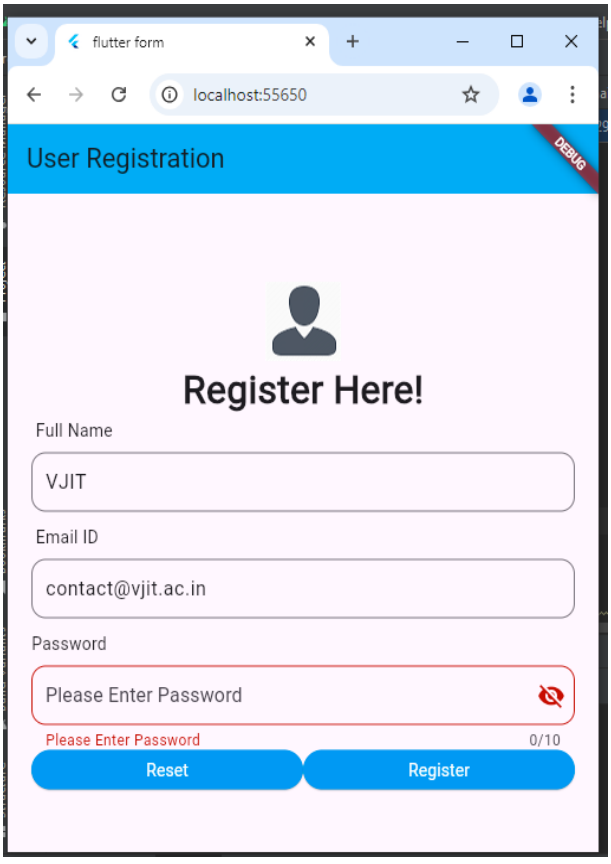
```
  const NextPage({super.key});
  @override
  Widget build(BuildContext context) {
   return Scaffold(
     appBar: AppBar(title: Text('Login Page'),
      backgroundColor: Colors.lightBlue,
     ),

     body: Center(
      child: Column(
       mainAxisAlignment: MainAxisAlignment.center,
       children: <Widget>[
        Text('Thank You !!',
         style: TextStyle(
          fontSize: 18,
          fontWeight: FontWeight.bold,
          color: Colors.red,
         ),
        ),

       ],
      ),
     ),
    );
   }
}
```

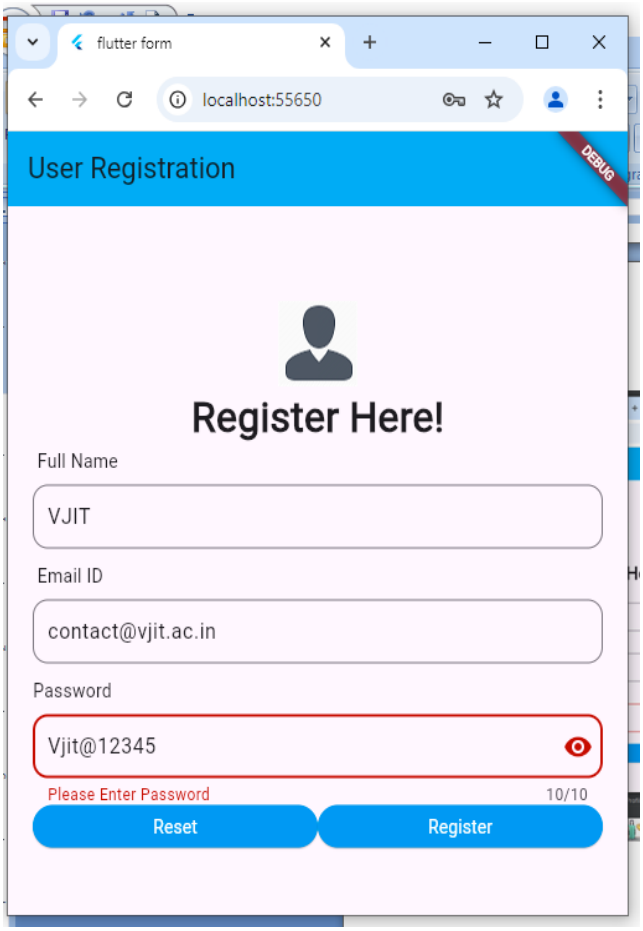
OUTPUT:

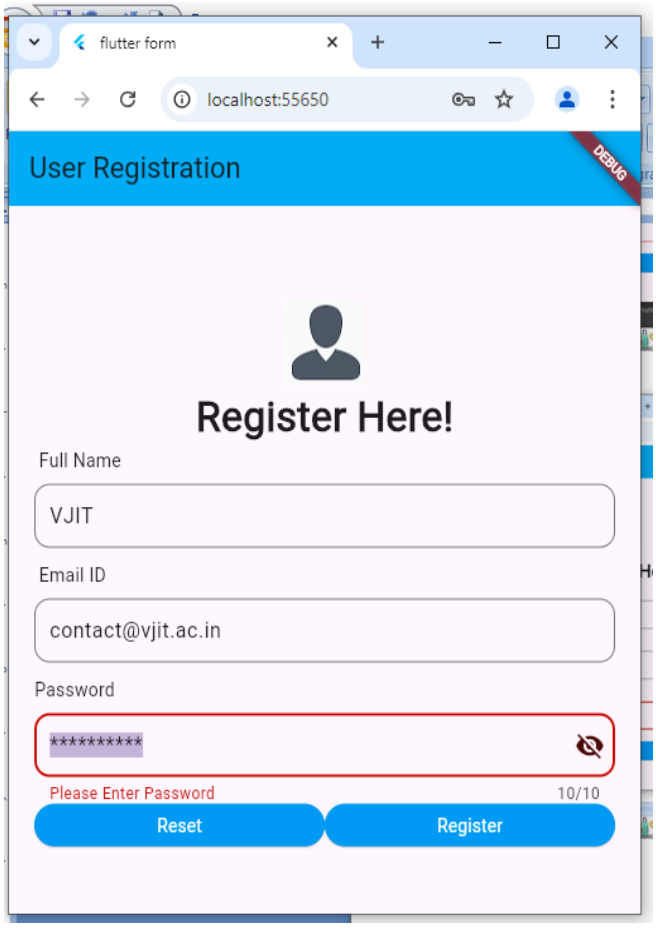

flutter form
localhost:53729
User Registration
DEBUG
Register Here!
Full Name
Please Enter Full Name
Email ID
Please Enter Email
Password
Please Enter Password
0/10
Reset
Register

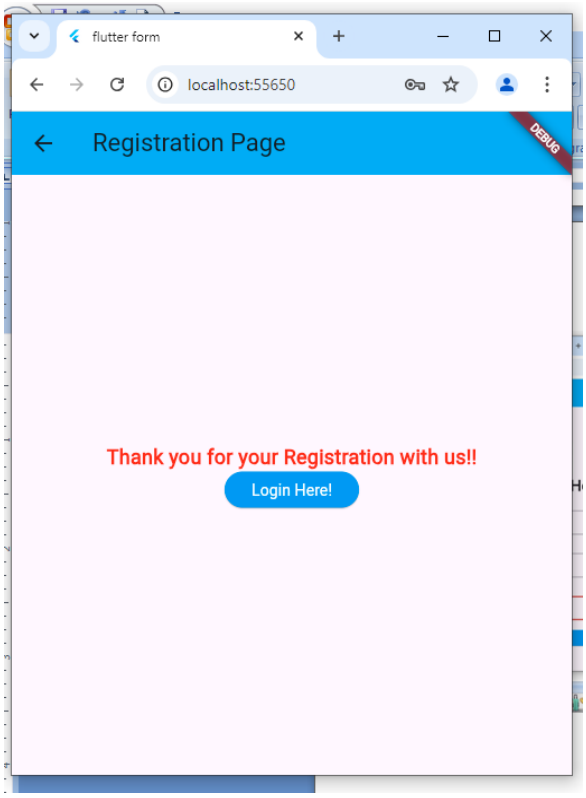

flutter form
localhost:55650
User Registration
DEBUG
Register Here!
Full Name
Please Enter Full Name
Please Enter Name
Email ID
Please Enter Email
Please Enter Email
Password
Please Enter Password
Please Enter Password
0/10
Reset
Register

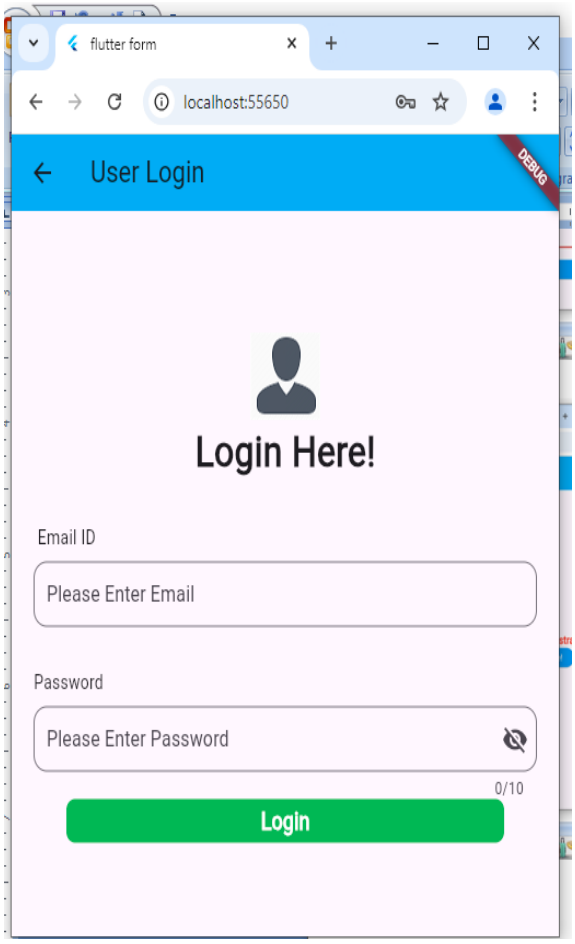

flutter form
localhost:55650
User Registration
DEBUG
Register Here!
Full Name
VJIT
Email ID
Please Enter Email
Please Enter Email
Password
Please Enter Password
Please Enter Password
0/10
Reset
Register

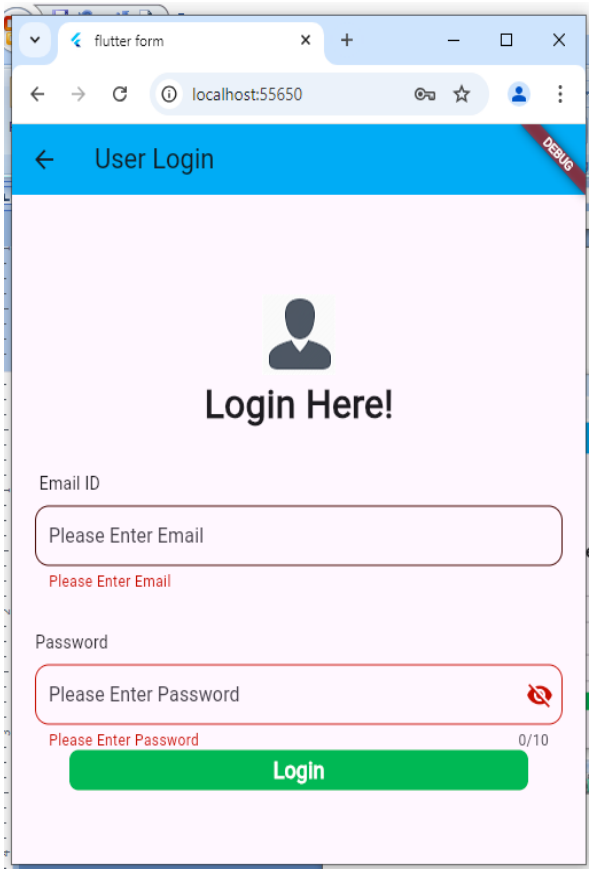

flutter form
localhost:55650
User Registration
DEBUG
Register Here!
Full Name
VJIT
Email ID
contact@vjit.ac.in
Password
Please Enter Password
Please Enter Password
0/10
Reset
Register

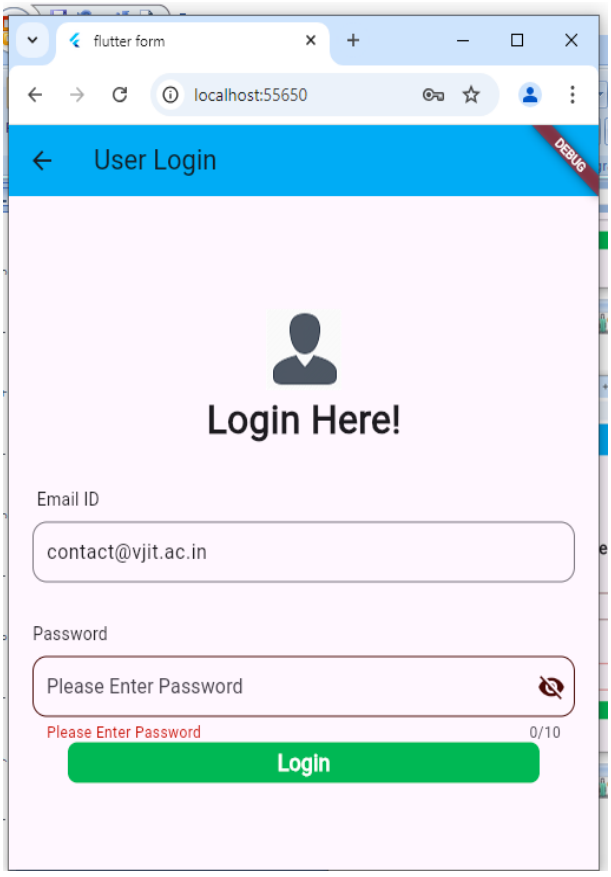

flutter form
localhost:55650
User Registration
DEBUG
Register Here!
Full Name
VJIT
Email ID
contact@vjit.ac.in
Password
Vjit@12345
Please Enter Password
10/10
Reset
Register

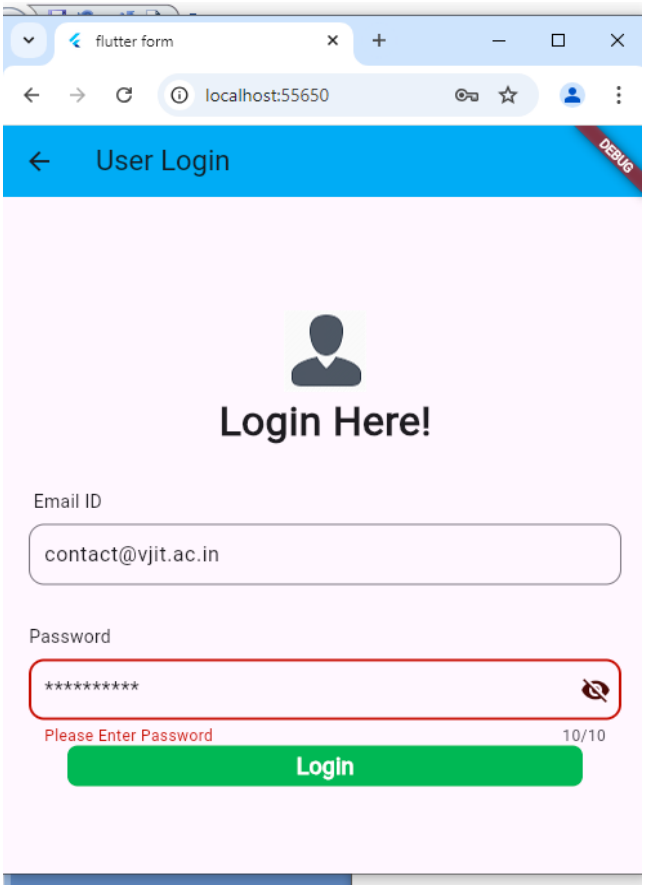

flutter form
localhost:55650
User Registration
DEBUG
Register Here!
Full Name
VJIT
Email ID
contact@vjit.ac.in
Password
****
Please Enter Password
10/10
Reset
Register

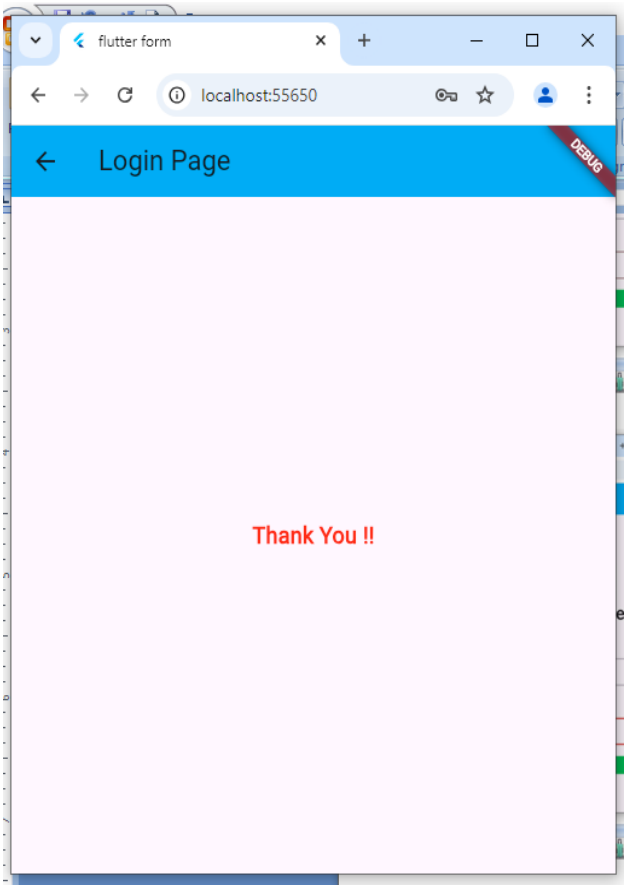

flutter form
localhost:55650
Registration Page
DEBUG
Thank you for your Registration with us!!
Login Here!


flutter form
localhost:55650
User Login
DEBUG
Login Here!
Email ID
Please Enter Email
Password
Please Enter Password
0/10
Login


flutter form
localhost:55650
User Login
DEBUG
Login Here!
Email ID
Please Enter Email
Please Enter Email
Password
Please Enter Password
Please Enter Password
0/10
Login


flutter form
localhost:55650
User Login
DEBUG
Login Here!
Email ID
contact@vjit.ac.in
Password
Please Enter Password
Please Enter Password
0/10
Login


flutter form
localhost:55650
User Login
DEBUG
Login Here!
Email ID
contact@vjit.ac.in
Password
****
Please Enter Password
10/10
Login


flutter form
localhost:55650
Login Page
DEBUG
Thank You !!

Week 11:

Create an application to implement flutter calendar.

Date Picker in Flutter

DatePicker is a material widget in a flutter that lets the user selects a date. Since there is no widget available for creating a date picker we will use showDatePicker() function.

It will display a Material Design date picker in a dialog by calling flutter's inbuilt function.

A date picker is a helpful addition to your UI that makes it easy for your app users to select dates from a calendar.

Add package

Install the package:

Add ***intl*** package to the ***pubspec.yaml*** file:

```
dependencies:
  flutter:
    sdk: flutter
  intl: ^0.17.0
```

Then configure it using pub get. You can also install the package from the command line by running the following command:

```
flutter pub add intl
```

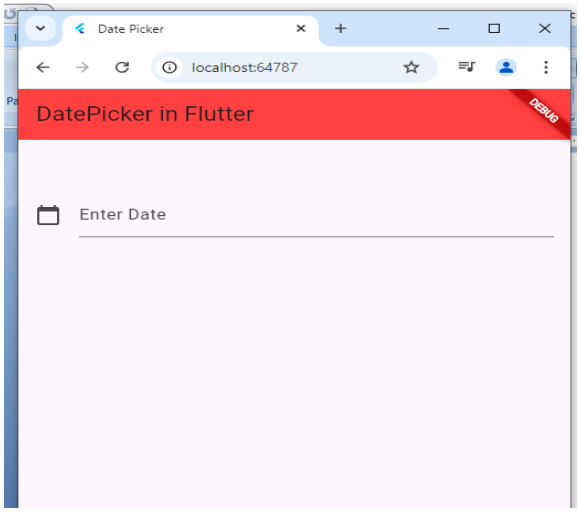
main.dart

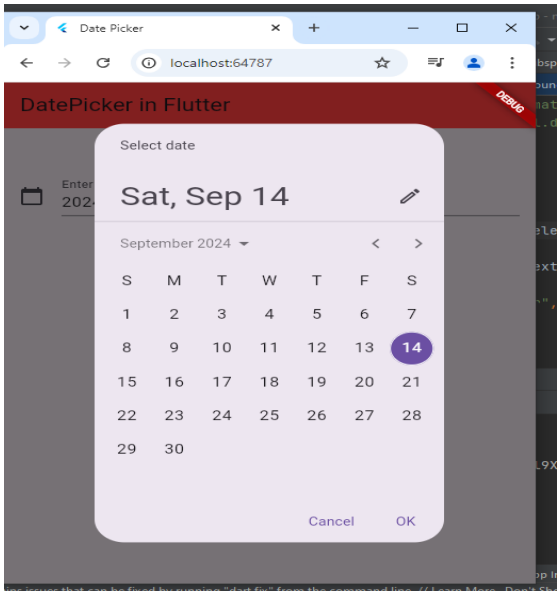
```
import 'package:flutter/material.dart';
import 'package:intl/intl.dart';

void main()
{
  runApp(MyApp());
}
class MyApp extends StatelessWidget
{
  @override
  Widget build(BuildContext context)
  {
```

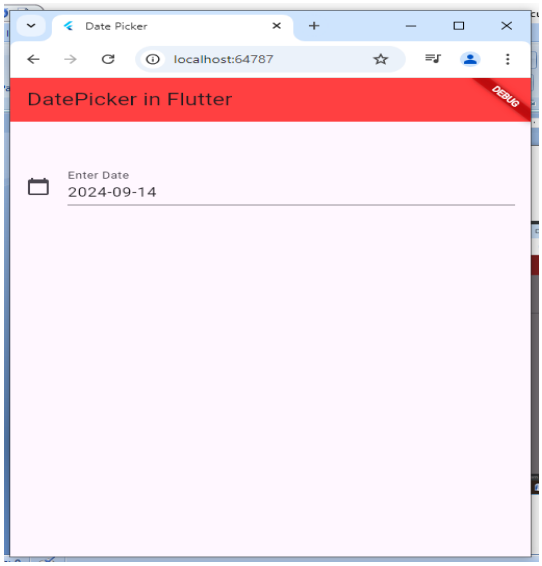
```
    return MaterialApp(
      title: "Date Picker",
      home: HomePage(),
    );
  }
}
class HomePage extends StatefulWidget
 {
  @override
  State<StatefulWidget> createState()
 {
    return _HomePage();
  }
}
class _HomePage extends State<HomePage>
 {
  TextEditingController dateInput = TextEditingController();
  @override
  void initState()
{
    dateInput.text = ""; //set the initial value of text field
    super.initState();
  }
  @override
  Widget build(BuildContext context)
{
    return Scaffold(
        appBar: AppBar(
          title: Text("DatePicker in Flutter"),
          backgroundColor: Colors.redAccent, //background color of app bar
        ),
        body: Container(
            padding: EdgeInsets.all(15),
            height: MediaQuery.of(context).size.width / 3,
            child: Center(
                child: TextField(
                  controller: dateInput,
                  //editing controller of this TextField
                  decoration: InputDecoration(
```

```
            icon: Icon(Icons.calendar_today), //icon of text field
            labelText: "Enter Date" //label text of field
          ),
          readOnly: true,
          //set it true, so that user will not able to edit text
          onTap: () async {
           DateTime? pickedDate = await showDatePicker(
              context: context,
              initialDate: DateTime.now(),
              firstDate: DateTime(1950),
              //DateTime.now() - not to allow to choose before today.
              lastDate: DateTime(2100));

           if (pickedDate != null) {
            print(
                pickedDate); //pickedDate output format => 2021-03-10 00:00:00.000
            String formattedDate =
            DateFormat('yyyy-MM-dd').format(pickedDate);
            print(
          formattedDate); //formatted date output using intl package =>  2021-03-16
            setState(() {
              dateInput.text =
                  formattedDate; //set output date to TextField value.
            });
           } else {}
          },
        ))));
  }
}
```

OUTPUT:


Date Picker
localhost:64787
DatePicker in Flutter
DEBUG
Enter Date


Date Picker
localhost:64787
DatePicker in Flutter
DEBUG
Select date
Sat, Sep 14
September 2024
S M T W T F S
1 2 3 4 5 6 7
8 9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30
Cancel
OK


Date Picker
localhost:64787
DatePicker in Flutter
DEBUG
Enter Date
2024-09-14

Week 12:

Create an application to implement Animated Text in Flutter.

Animated Text in Flutter

Animations make the UI more interactive and enhance the user experience. There is no limitation of creativity when it comes to animations or making the application more interactive. In Flutter, we got an easy way to display Animated text. In this article, we will see how we can animate texts using animated_text_kit.

Types of Text Animations:

Following are the types of text animations available with the animated_text_kit package:

1. RotatedAnimatedText()
2. ScaleAnimatedText()
3. FadeAnimatedText()
4. TyperAnimatedText()
5. WavyAnimatedText()
6. FlickerAnimatedText()

We can also create customized animated texts. Let us move to the implementation part of this package.

Install the package:

Add animated_text_kit package to the *pubspec.yaml* file:

```
dependencies:
  flutter:
    sdk: flutter


  # The following adds the Cupertino Icons font to your application.
  # Use with the CupertinoIcons class for iOS style icons.
  cupertino_icons: ^1.0.2
  animated_text_kit: ^4.2.1
```

Then configure it using pub get.

You can also install the package from the command line by running the following command:

```
flutter pub add animated_text_kit
```

Implementation:

We use AnimatedTextKit() widget to create animated texts. In the *animatedTexts* property, we can add as many texts and any type of animated texts like RotatedAnimatedText, FlickerAnimatedText, etc.

Some properties of AnimatedTextKit are –

- animatedTexts
- onTap
- totalRepeatCount
- repeatForever
- pause
- displayFullTextOnTap
- isRepeatingAnimation

main.dart

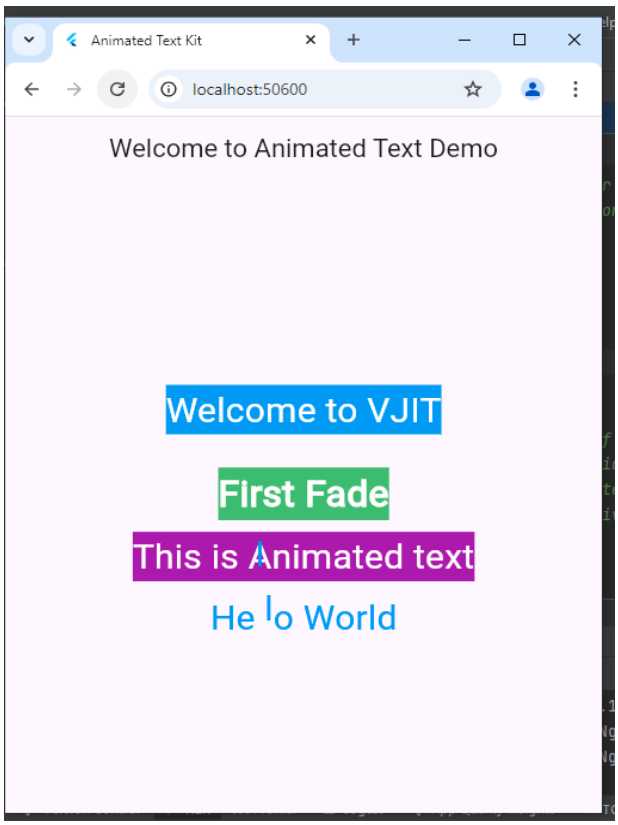
```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:animated_text_kit/animated_text_kit.dart';

void main() => runApp(MyApp());

class MyApp extends StatefulWidget
{
  @override
  _MyAppState createState() => _MyAppState();
}

class _MyAppState extends State<MyApp>
{
  @override
  Widget build(BuildContext context)
{
    return MaterialApp(
        title: 'Animated Text Kit',
        debugShowCheckedModeBanner: false,
        theme: ThemeData(primarySwatch: Colors.green),
        home: Scaffold(
          appBar: AppBar(
            title: const Text("Welcome to Animated Text Demo"),
            centerTitle: true,
```

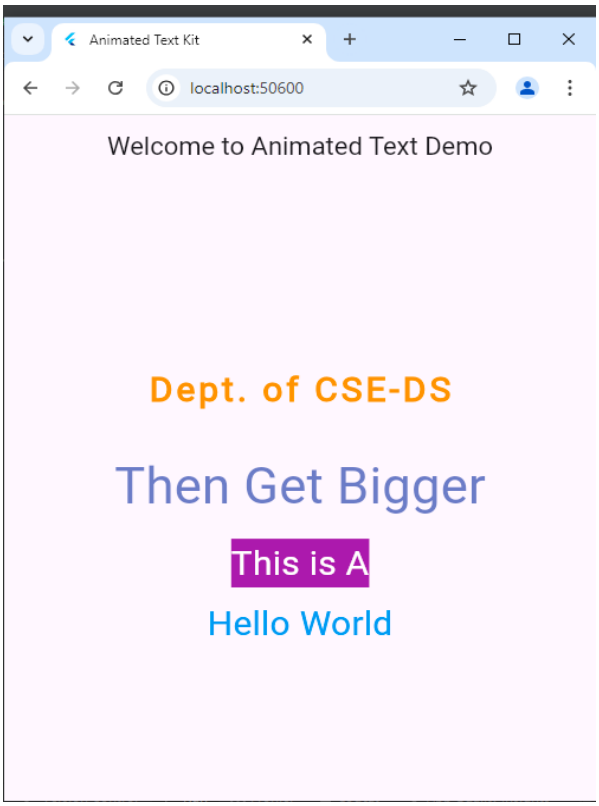
```
),
body: Column(
 mainAxisAlignment: MainAxisAlignment.center,
 children: <Widget>[
  AnimatedTextKit(
   animatedTexts: [
    RotateAnimatedText('Welcome to VJIT',
      textStyle: TextStyle(
        fontSize: 30,
        color: Colors.white,
        backgroundColor: Colors.blue)),
    RotateAnimatedText('Dept. of CSE-DS',
      textStyle: TextStyle(
        letterSpacing: 3,
        fontSize: 30,
        fontWeight: FontWeight.bold,
        color: Colors.orange)),
    RotateAnimatedText(
     'Hyderabad',
     textStyle: TextStyle(
      fontSize: 30,
      decoration: TextDecoration.underline,
     ),
    ),
   ],
   isRepeatingAnimation: true,
   totalRepeatCount: 10,
   pause: Duration(milliseconds: 1000),
  ),
  // SizedBox(height: 10),
  Center(
   child: AnimatedTextKit(
    totalRepeatCount: 40,
    animatedTexts: [
     FadeAnimatedText(
      'First Fade',
      textStyle: const TextStyle(
        backgroundColor: Colors.green,
        color: Colors.white,
```

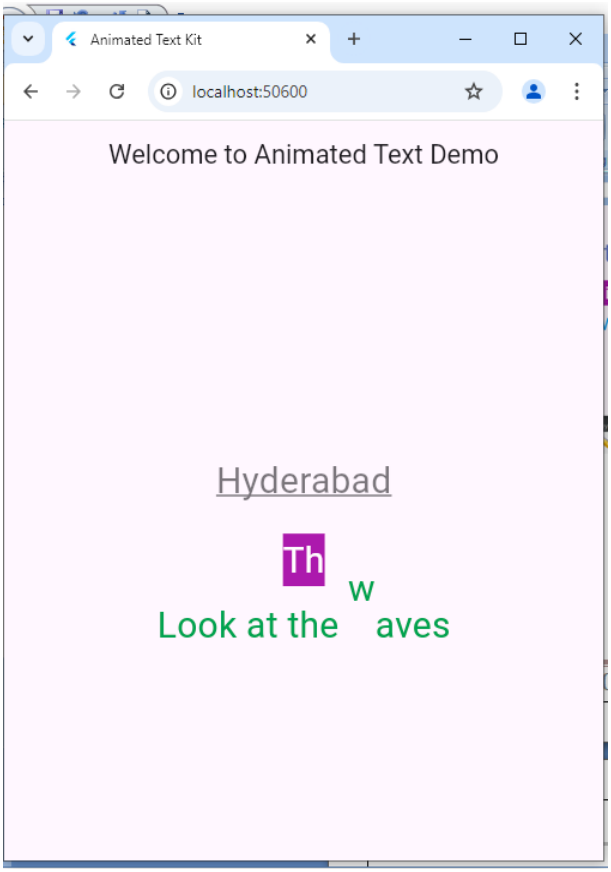
```
            fontSize: 32.0,
            fontWeight: FontWeight.bold),
      ),
      ScaleAnimatedText(
       'Then Get Bigger',
       duration: Duration(milliseconds: 4000),
       textStyle:
       const TextStyle(color: Colors.indigo, fontSize: 50.0),
      ),
    ],
  ),
),

SizedBox(height: 10),
AnimatedTextKit(
  animatedTexts: [
    TyperAnimatedText('This is Animated text,',
        textStyle: const TextStyle(
            color: Colors.white,
            fontSize: 30,
            backgroundColor: Colors.purple)),
    TyperAnimatedText('You are viewing it here.',
        textStyle: const TextStyle(
            fontSize: 20, fontWeight: FontWeight.bold)),
  ],
  onTap: () {
    print("I am executing");
  },
),

SizedBox(height: 10),
Center(
  child: AnimatedTextKit(
    animatedTexts: [
      WavyAnimatedText('Hello World',
          textStyle: TextStyle(
            color: Colors.blue,
            fontSize: 30,
          )),
```

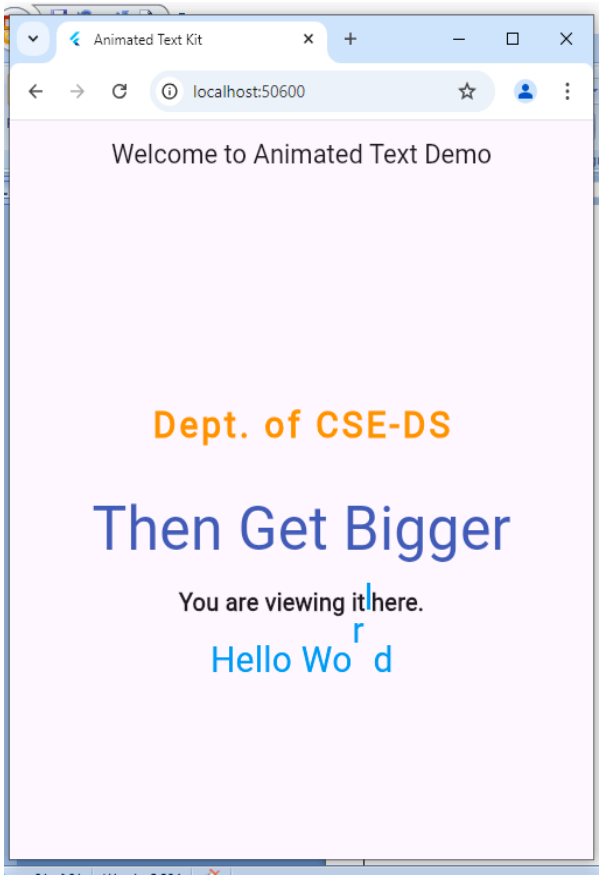
```
                WavyAnimatedText('Look at the waves',
                  textStyle: TextStyle(
                    color: Colors.green[600],
                    fontSize: 30,
                  )),
              ],
              repeatForever: true,
              onTap: () {
                print("Tap Event");
              },
            ),
          ),
        ],
      ),
    ));
  }
}
```

OUTPUT


Animated Text Kit
localhost:50600
Welcome to Animated Text Demo
Welcome to VJIT
First Fade
This is Animated text
He lo World


Animated Text Kit
localhost:50600
Welcome to Animated Text Demo
Dept. of CSE-DS
Then Get Bigger
This is A
Hello World


Animated Text Kit
localhost:50600
Welcome to Animated Text Demo
Hyderabad
Th
w
Look at the
aves


Animated Text Kit
localhost:50600
Welcome to Animated Text Demo
Dept. of CSE-DS
Then Get Bigger
You are viewing it here.
r
Hello Wo d