Experiment no:3

Aim: To include icons, images, fonts in Flutter app

Theory:
Introduction

Flutter is a powerful UI development framework that allows developers to create visually appealing applications using a wide range of UI elements. In modern applications, icons, images, and custom fonts play a crucial role in enhancing user experience by improving aesthetics, readability, and branding. Flutter provides built-in support for adding these assets efficiently, ensuring a smooth and responsive UI.

This experiment focuses on understanding how to incorporate icons, images, and fonts into a Flutter application to create a more engaging and customized interface.

2. Objective

The main objectives of this experiment are:

- To learn how to use icons in a Flutter app.
- To understand how to include and display images.
- To explore the process of adding custom fonts for better UI styling.

3. Importance of Icons, Images, and Fonts in UI Design

Icons, images, and fonts enhance the usability and aesthetic appeal of a mobile application.

- Icons provide a visual representation of actions and functionalities, improving navigation and user experience.
- Images enhance the visual storytelling of an app, making it more engaging and informative.
- Custom Fonts help in defining the app's branding and improve text readability, ensuring consistency in UI design.

4. Including Icons in Flutter

Flutter supports two types of icons:

1. Material Icons – Built-in icons provided by Flutter's Material Design framework (Icons class).
2. Custom Icons – Custom icon sets imported as image assets or vector files (SVGs).

5. Adding Images in Flutter

Flutter allows developers to include images in three different ways:

1. Asset Images – Images stored in the project's assets folder.
2. Network Images – Images loaded from an external URL.
3. Memory Images – Images dynamically generated or loaded at runtime.

6. Using Custom Fonts in Flutter

Flutter allows the integration of custom fonts to enhance text styling. Custom fonts provide a unique appearance to the app and align with branding guidelines. To use custom fonts:

1. Font files are placed in the assets/fonts directory.
2. The pubspec.yaml file is updated to register the font.
3. The font is applied using the TextStyle property in Flutter widgets.

Code:

Colors.dart:

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

const backgrounndColor = Colors.black;

const kwhiteColor = Colors.white;

final kButtonColorBlue = Colors.blueAccent[700];

const kButtonColorWhite = Colors.white;

const kBlackColor = Colors.black;

const kGray = Colors.grey;
```

Constants.dart:

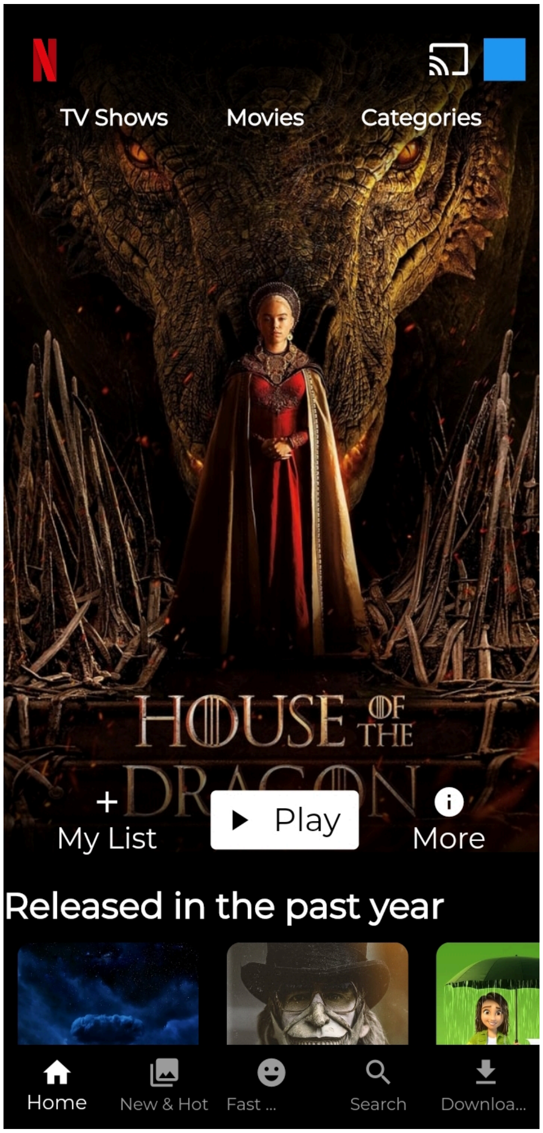
```
import 'package:flutter/material.dart';
const kWidth = SizedBox(
  width: 10,
);
const kHeight = SizedBox(
  height: 10,
);
const kHeight20 = SizedBox(
  height: 20,
);
const kHeight50 = SizedBox(
  height: 50,
);
// border radius
final BorderRadius kRadius10 = BorderRadius.circular(10);
final BorderRadius kRadius30 = BorderRadius.circular(30);
// image
const mainImage =
    'https://www.themoviedb.org/t/p/w600_and_h900_bestv2/A69pKujkYeLp7uO4LrUqZvxEAze.jpg';
const imageUrltemp1 =
    'https://www.themoviedb.org/t/p/w533_and_h300_bestv2/iQlJyRecJeGGzQGT2rEcyAgz89F.jpg';
const imageUrltemp2 =
    "https://www.themoviedb.org/t/p/w533_and_h300_bestv2/8TUb2U9GN3PonbXAQ1FBcJ4XeXu.jpg";
// Textstyle
```

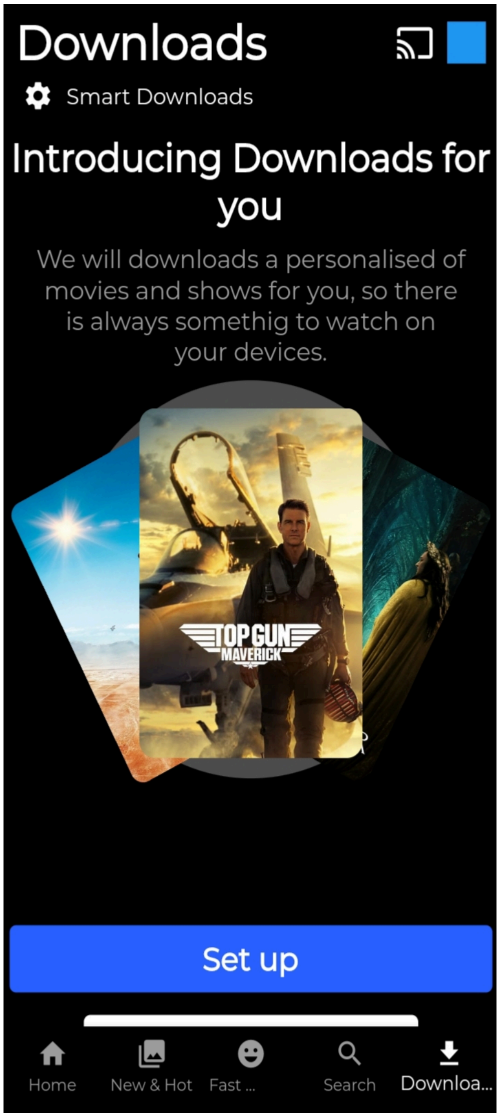
```
const TextStyle kHomeTitleText =

    TextStyle(fontSize: 14, fontWeight: FontWeight.bold);

const imageAppendUrl = "https://image.tmdb.org/t/p/w500";

const mainScrenImage =
'https://www.themoviedb.org/t/p/w600_and_h900_bestv2/z2yahl2uefxDCl0nogcRBstwruJ.jpg';
```

Strings.dart:

```
const kBaseUrl = "https://api.themoviedb.org/3";
```

Output:





Introducing Downloads for you

We will downloads a personalised of movies and shows for you, so there is always somethig to watch on your devices.

Conclusion:

Icons, images, and fonts are essential components of a well-designed Flutter application. They contribute to a more polished and professional look, improving both aesthetics and user experience. By understanding how to include and manage these assets, developers can build visually appealing and highly functional Flutter applications.