Aim: - To Include icons , images & fonts in the flutter app

Theory:

Flutter provides robust support for customizing the UI using **icons**, **images**, and **custom fonts**. These elements enhance the visual appeal and user experience of the application. Below is a detailed explanation of each component:

1. Icons in Flutter:

Icons are graphical representations of actions, files, devices, or app features. Flutter uses Material Design Icons by default, but also allows adding custom icons.

Types of Icons in Flutter:

- Built-in Icons: Provided by Flutter in the Icons class (Material Design).
- Custom Icons: You can use .ttf icon fonts (like Font Awesome) by adding them in the pubspec.yaml and referring via IconData.

How to Use Icons:

- Add icons using the Icon widget.
- Change size, color, and action by setting properties.
- Combine with buttons like IconButton for interactivity.

2. Images in Flutter:

Flutter supports both **network** and **local images** to display pictures, logos, banners, and illustrations.

Types of Images:

- **Asset Images**: Stored in the app's assets folder.
- **Network Images**: Fetched from a URL at runtime.

How to Add Asset Images:

1. Create an assets/ folder in the root of the project.
2. Place image files inside this folder (e.g., logo.png).
3. Declare the image in pubspec.yaml:
4. Use the image in code:

Customization:

- Set width, height, fit (BoxFit.cover, BoxFit.contain, etc.

- Apply rounded corners or shadows using ClipRRect or Container.

3. Fonts in Flutter:

Flutter allows the use of **custom fonts** for branding and better typography.

How to Add Custom Fonts:

1. Create a fonts/ folder and place .ttf or .otf files.
2. Declare fonts in pubspec.yaml:
3. Use the custom font in your app:

Font Properties You Can Customize:

- fontSize
- fontWeight (e.g., FontWeight.bold)
- fontStyle (e.g., FontStyle.italic)
- letterSpacing and wordSpacing

Best Practices:

- Keep asset names in lowercase and avoid spaces.
- Use appropriate image resolutions for better performance.
- Use responsive sizing for images and text for multi-device compatibility.
- Group similar assets (e.g., images/icons/fonts) in separate folders.

Code snippets:

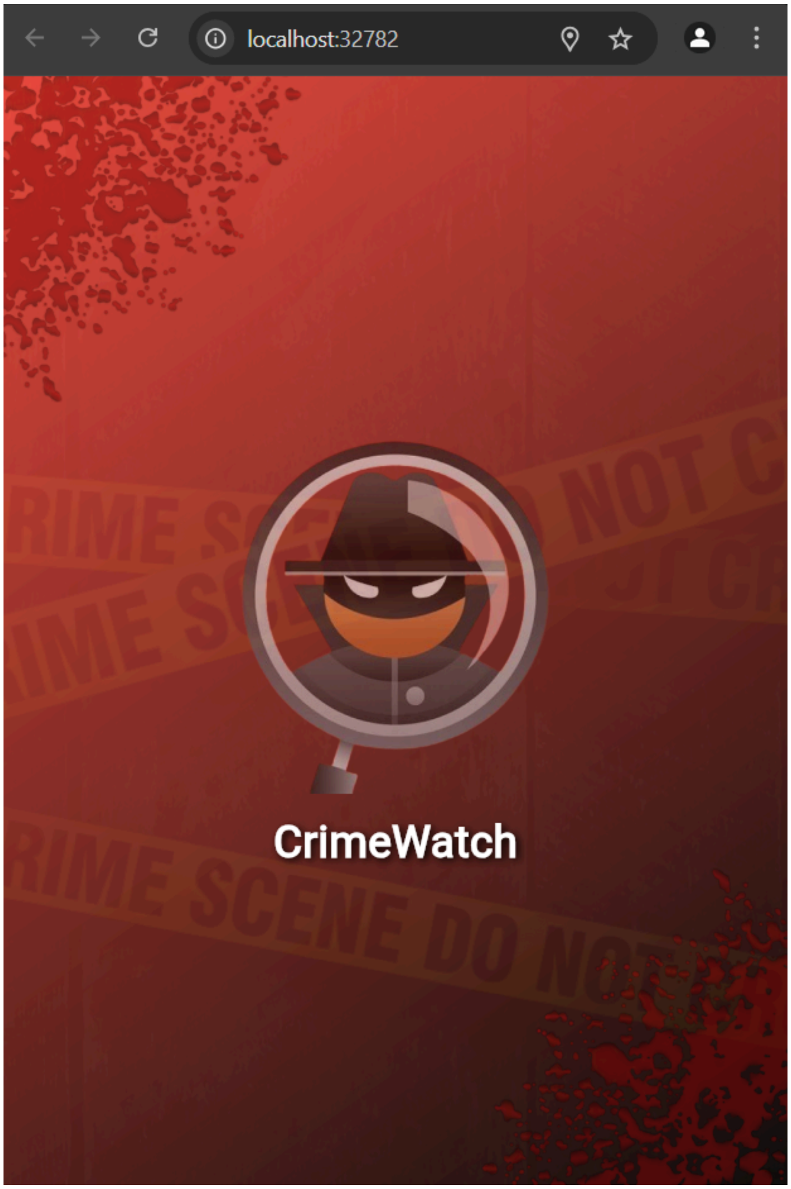
a. **pubspec.yaml**

```
assets:
  - assets/logo.png

fonts:
  - family: Poppins
    fonts:
      - asset: fonts/Comic Sans-Regular.ttf
```

b. **home_screen.dart** : A splash screen was developed featuring multiple image assets (background, corners, and logo) layered using a Stack widget. The logo appears with a bounce animation, while the app name "CrimeWatch" is revealed letter-by-letter using a typewriter effect. A custom font ('Comic Sans MS') is applied to the text, contributing to a distinctive and engaging interface.

```
import 'package:flutter/material.dart';
import 'signup.dart';
import 'package:crime_alert/views/home/home_screen.dart';

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

  @override
  State<HomeScreen> createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen>
    with SingleTickerProviderStateMixin {
  late AnimationController _controller;
  late Animation<double> _bounceAnimation;
  String _displayText = '';
  final String _fullText = 'CrimeWatch';
  int _charIndex = 0;

  @override
  void initState() {
    super.initState();
    _controller = AnimationController(
      duration: const Duration(milliseconds: 1000),
      vsync: this,
    );

    _bounceAnimation = TweenSequence<double>([
      TweenSequenceItem(
        tween: Tween<double>(begin: 0.0, end: 1.2)
            .chain(CurveTween(curve: Curves.easeInOut)),
        weight: 40.0,
      ),
      TweenSequenceItem(
        tween: Tween<double>(begin: 1.2, end: 1.0)
            .chain(CurveTween(curve: Curves.bounceOut)),
        weight: 60.0,
      ),
    ]).animate(_controller);
    _controller.forward();

    _controller.addStatusListener((status) {
      if (status == AnimationStatus.completed) {
        _animateText();
      }
    });
  }

void _animateText() {
  if (_charIndex < _fullText.length) {
    Future.delayed(const Duration(milliseconds: 200), () {
      if (mounted) {
        setState(() {
          _displayText = _fullText.substring(0, _charIndex + 1);
          _charIndex++;
        });
        _animateText();  // Call itself recursively with delay
      }
    });
  } else {
    Future.delayed(const Duration(seconds: 1), () {
      if (mounted) {
        Navigator.pushReplacement(
          context,
          MaterialPageRoute(builder: (context) =>  Dashboard()),
        );
      }
    });
  }
}


  @override
  void dispose() {
    _controller.dispose();
    super.dispose();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Stack(
        fit: StackFit.expand,
        children: [
          Positioned.fill(
  child: Image.asset(
    'images/background.png',
    fit: BoxFit.cover,
  ),
),
Positioned.fill(
  child: Container(
    decoration: BoxDecoration(
      gradient: LinearGradient(
        begin: Alignment.topLeft,
        end: Alignment.bottomRight,
        colors: [
          Colors.red.withOpacity(0.9), // Mild Red
          Colors.black.withOpacity(0.9), // Mild Black
        ],
      ),
    ),
  ),
),
Positioned(
  top: 0,
  left: 0,
  child: Image.asset(
    '/images/splash_top_left.png',
    width: 200,
    height: 220,
    fit: BoxFit.cover,
  ),
),
Positioned(
  bottom: 0,
  right: 0,
  child: Image.asset(
    '/images/splash_bottom_right.png',
    width: 200,
    height: 220,
    fit: BoxFit.cover,
  ),
),
Center(
  child: Column(
    mainAxisSize: MainAxisSize.min,
    children: [
      ScaleTransition(
        scale: _bounceAnimation,
        child: Image.asset(
                    'images/logo.png',
                    width: 400,
                    height: 250,
                  ),
                ),
                const SizedBox(height: 10),
                Text(
                  _displayText,
                  style: const TextStyle(
                    fontFamily: 'Comic Sans MS',
                    fontSize: 28,
                    fontWeight: FontWeight.bold,
                    color: Colors.white,
                    shadows: [
                      Shadow(
                        color: Colors.black,
                        offset: Offset(2, 2),
                        blurRadius: 3,
                      ),
                    ],
                  ),
                ),
              ],
            ),
          ),
        ],
      ),
    );
  }
}
```

Output :



The icon is positioned at the center, while splash screen images are placed at the top and bottom edges, complemented by the 'Crimwatch' custom font for branding.

Conclusion : The use of animated logo transitions, background visuals, and stylized text significantly enhanced the app’s visual identity and user engagement. This validates the importance of UI elements in establishing a strong first impression and aligns well with the project's aim of delivering a visually appealing mobile application.