MAD PWA Exp 3

Aim: To include icons, images and fonts in flutter app

Theory:

Flutter, Google's open-source UI toolkit, empowers developers to craft visually appealing and highly performant applications. A fundamental aspect of creating a compelling user interface lies in the seamless integration of icons, images, and custom fonts.

Icons

Icons serve as intuitive visual cues, aiding users in navigation and comprehension. Flutter provides an extensive set of built-in icons, and developers can easily integrate custom icons. Icons contribute to a consistent design language and enhance the overall user experience.
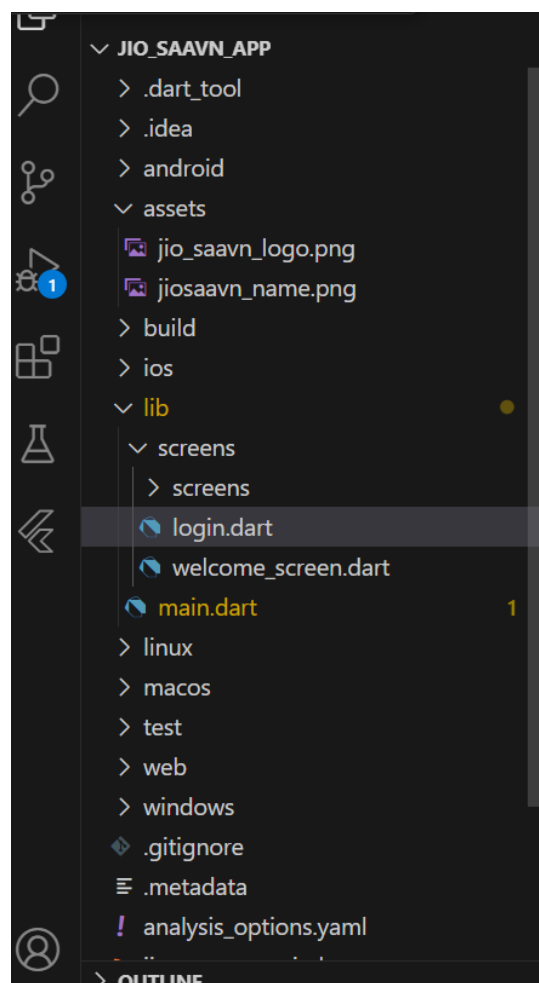
Principles:

1. Selecting Icon Packages: Developers often leverage existing icon packages, such as "material_icons" or "font_awesome_flutter," to access a diverse range of symbols. This streamlines the integration process and ensures a cohesive visual language.
2. Installation and Implementation: The chosen icon package is added to the project's dependencies. Icons can then be effortlessly implemented using the Icon widget, specifying the desired icon type, size, and color.

Images

Images play a pivotal role in conveying information, setting the mood, and creating an immersive user experience. Flutter supports various image formats and provides mechanisms for both local and network-based image integration.

Principles:

1. Organizing Image Assets: Structuring image assets within the 'assets' folder and updating the pubspec.yaml file facilitates proper asset management. This step ensures clarity and ease of access for images within the Flutter project.
2. Loading Images: Utilizing the Image.asset widget for local images and Image.network for network images simplifies the integration process. This allows developers to incorporate visuals seamlessly into their application interfaces.

Fonts

Custom fonts contribute to brand identity and enhance the overall aesthetics of an application. Flutter enables the integration of custom fonts, allowing developers to align the typography with the app's design principles.

Principles:

1. Adding Font Files: Placing font files within a 'fonts' folder and updating the pubspec.yaml file establishes a clear structure for font assets. This practice simplifies the inclusion of custom fonts in Flutter projects.
2. Loading and Applying Fonts: Integrating custom fonts involves specifying the font family and style using the TextStyle widget. This ensures consistent and visually appealing text elements throughout the application.

File structure:

Code:

```dart
//main.dart

import 'package:flutter/material.dart';
import 'screens/login.dart';
import 'screens/welcome_screen.dart';

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

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'JioSaavn App',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      home: const WelcomePage(),
    );
  }
}
```

```dart
//login.dart

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: LoginScreen(),
    );
  }
}

class LoginScreen extends StatelessWidget {
  @override
```

```dart
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Colors.black,
    body: Column(
      mainAxisAlignment: MainAxisAlignment.end,
      children: [
        Padding(
          padding: const EdgeInsets.all(25.0),
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              // Jio Saavn Logo
              Image.asset(
                'assets/jiosaavn_name.png', // Path to Jio Saavn logo
                height: 150,
                width: 200,
              ),
              SizedBox(height: 20),

              Text(
                'Login and enjoy more than 80 million songs',
                style: TextStyle(
                  fontSize: 14,
                  fontWeight: FontWeight.normal,
                  color: Color.fromARGB(198, 202, 189, 189),
                ),
              ),
              SizedBox(height: 20),

              // Login with Jio Button
              Container(
                width: double.infinity,
                margin: EdgeInsets.symmetric(horizontal: 35),
                decoration: BoxDecoration(
                  color: Color.fromRGBO(0, 255, 240, 1),
                  borderRadius: BorderRadius.circular(12),
                ),
                child: ElevatedButton(
                  onPressed: () {
                    // Handle Jio login
                  },
                  style: ElevatedButton.styleFrom(
                    primary: Colors
                        .transparent, // Transparent color for elevated
button
                    shape: RoundedRectangleBorder(
                      borderRadius: BorderRadius.circular(8),
                    ),
```
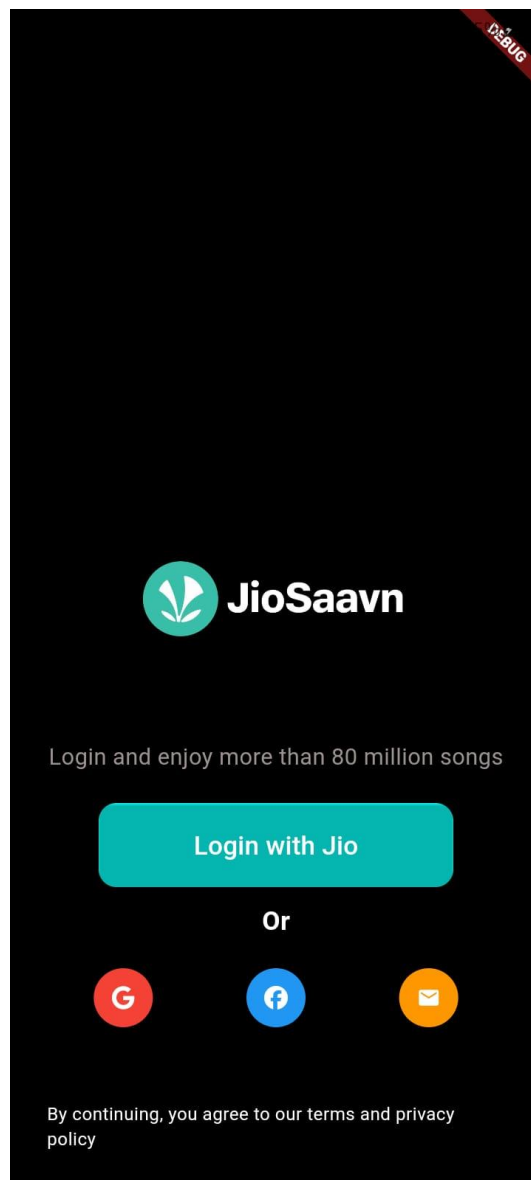
```dart
          ),
          child: Padding(
            padding: const EdgeInsets.all(16.0),
            child: Text(
              "Login with Jio",
              style: TextStyle(fontSize: 16, color: Colors.white),
            ),
          ),
        ),
      ),
      SizedBox(height: 10),

      Text(
        'Or',
        style: TextStyle(
          fontSize: 16,
          fontWeight: FontWeight.bold,
          color: Colors.white,
        ),
      ),
      SizedBox(height: 20),

      // Social Logins
      Row(
        mainAxisAlignment: MainAxisAlignment.spaceAround,
        children: [
          _buildSocialLoginButton(
              FontAwesomeIcons.google, "", Colors.red, () {
            // Handle Google login
          }),
          _buildSocialLoginButton(
              FontAwesomeIcons.facebook, "", Colors.blue, () {
            // Handle Facebook login
          }),
          _buildSocialLoginButton(Icons.mail, "", Colors.orange, ()
{

            // Handle Mail login
          }),
        ],
      ),
      SizedBox(height: 20),

      Text(
        'By continuing, you agree to our terms and privacy policy',
        style: TextStyle(
          fontSize: 11,
          fontWeight: FontWeight.normal,
          color: Colors.white,
```

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

  Widget _buildSocialLoginButton(
      IconData icon, String text, Color color, VoidCallback onPressed) {
    return InkWell(
      onTap: onPressed,
      child: Column(
        children: [
          CircleAvatar(
            radius: 20, // Reduced size
            backgroundColor: color,
            child: Icon(icon, size: 16, color: Colors.white),
          ),
          SizedBox(height: 8),
          Text(
            text,
            style: TextStyle(color: Colors.white),
          ),
        ],
      ),
    );
  }
}
```

Output:

Conclusion:
In conclusion, the integration of icons, images, and fonts in Flutter applications is foundational to creating visually stunning and engaging user interfaces. By adhering to best practices, developers can uphold design consistency, enhance user experience, and bring forth applications that resonate aesthetically with their target audience.