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

Theory

1. Icons in Flutter

Icons are essential elements in modern mobile apps as they visually represent actions, features, or categories. In Flutter, icons are used to provide an intuitive and clean user interface.

● Material Icons: Flutter comes with a built-in set of Material Design icons that can be used in any app. These icons are simple and designed to work well across various devices and screen sizes.

● Custom Icons: You can also integrate your own set of icons into the app. These custom icons can be added as image assets and used where required in the UI.

Benefits of Using Icons:

● Enhance usability by providing visual cues.

● Improve app aesthetics by creating a cohesive and modern look.

● Consistent and recognizable symbols for common actions (e.g., home, search, settings).

2. Images in Flutter

Images help to improve the look and feel of an app, offering visual appeal. In Flutter, you can display images from different sources such as local assets, network URLs, or even files stored on the device.

● Asset Images: These are images stored locally within the app's project directory. They can be bundled with the app and accessed efficiently.

● Network Images: Images fetched from the internet, typically hosted on a server or cloud service. Network images provide flexibility to display dynamic content.

● File Images: Images that reside on the device’s local storage, often from user uploads or device-specific data.

Benefits of Using Images:

● Add context or visual interest to the app's UI.

● Help users understand the app’s content or features quickly.

● Allow for brand identity by displaying logos, banners, or promotional graphics.

3. Fonts in Flutter

Fonts are key to the visual identity of any app, and Flutter allows you to use both default fonts as well as custom fonts to match your app's theme or brand.

● Custom Fonts: You can add fonts in various formats (e.g., TrueType Font .ttf or OpenType Font .otf) and define them in the app’s configuration. This allows you to maintain consistency with your brand’s typography.

● Default Fonts: Flutter also provides a set of default fonts which are used if no custom fonts are specified.

Benefits of Using Custom Fonts:

- Improve the app's aesthetic by using unique typography.
- Maintain brand consistency by incorporating brand-specific font
    - Allow for flexibility in styling text elements (headings, body text, etc.) throughout the app.

Code:

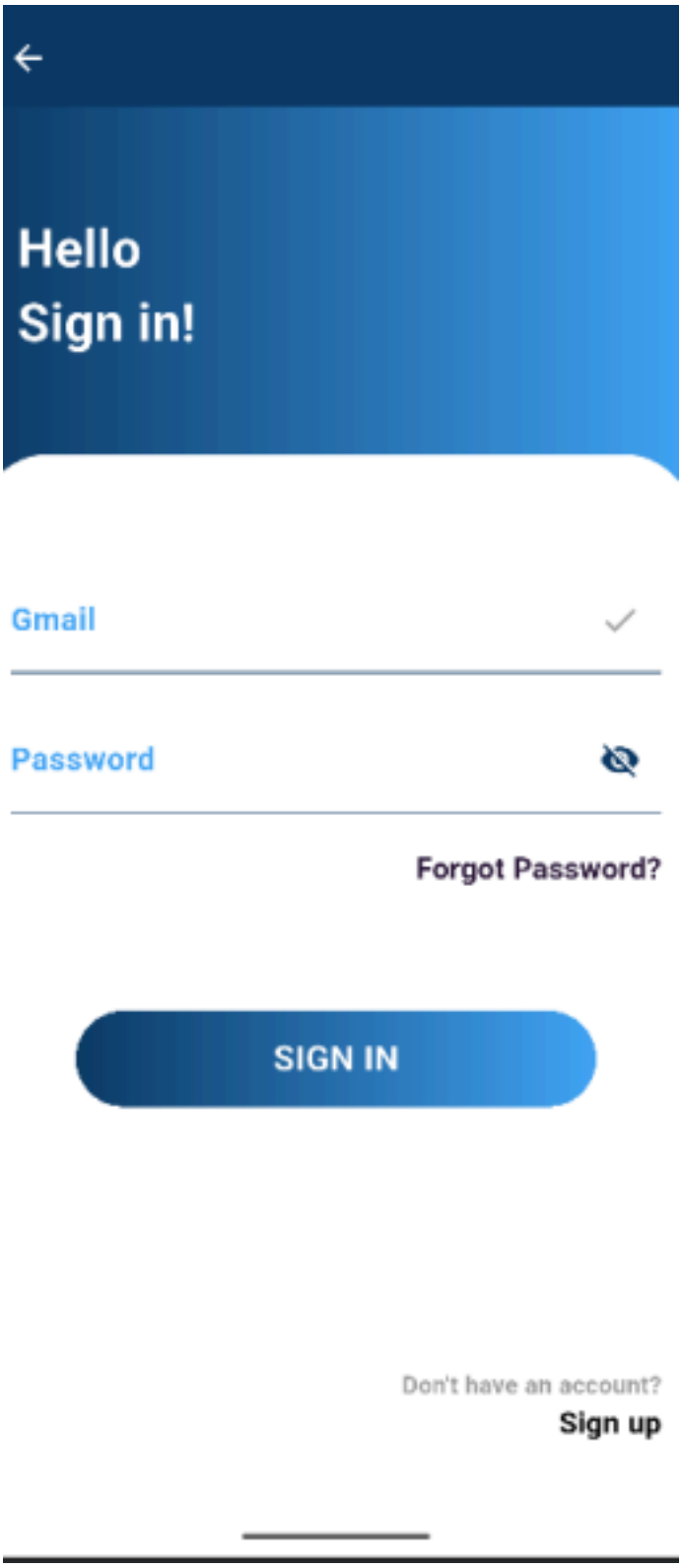
```
import 'package:crimetrack/validation/validator.dart';
import 'package:flutter/material.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:fluttertoast/fluttertoast.dart'; // Import FlutterToast
import 'verify_page.dart'; // Import VerifyEmailScreen
import '../app_colors.dart'; // Import AppColors

class RegScreen extends StatefulWidget {
  const RegScreen({Key? key}) : super(key: key);

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

class _RegScreenState extends State<RegScreen> {
  final _formKey = GlobalKey<FormState>();
  final TextEditingController _nameController = TextEditingController();
  final TextEditingController _emailController = TextEditingController();
  final TextEditingController _passwordController = TextEditingController();
  final TextEditingController _confirmPasswordController =
TextEditingController();

  bool _isPasswordVisible = false;
  bool _isConfirmPasswordVisible = false;
  bool _isLoading = false;

  FocusNode _nameFocusNode = FocusNode();
  FocusNode _emailFocusNode = FocusNode();
  FocusNode _passwordFocusNode = FocusNode();
  FocusNode _confirmPasswordFocusNode = FocusNode();

  @override
  void dispose() {
    _nameController.dispose();
    _emailController.dispose();
    _passwordController.dispose();
    _confirmPasswordController.dispose();
    _nameFocusNode.dispose();
    _emailFocusNode.dispose();
    _passwordFocusNode.dispose();
    _confirmPasswordFocusNode.dispose();
    super.dispose();
  }

  Future<void> _registerUser() async {
    if (_formKey.currentState?.validate() ?? false) {
      setState(() => _isLoading = true);
      try {
```

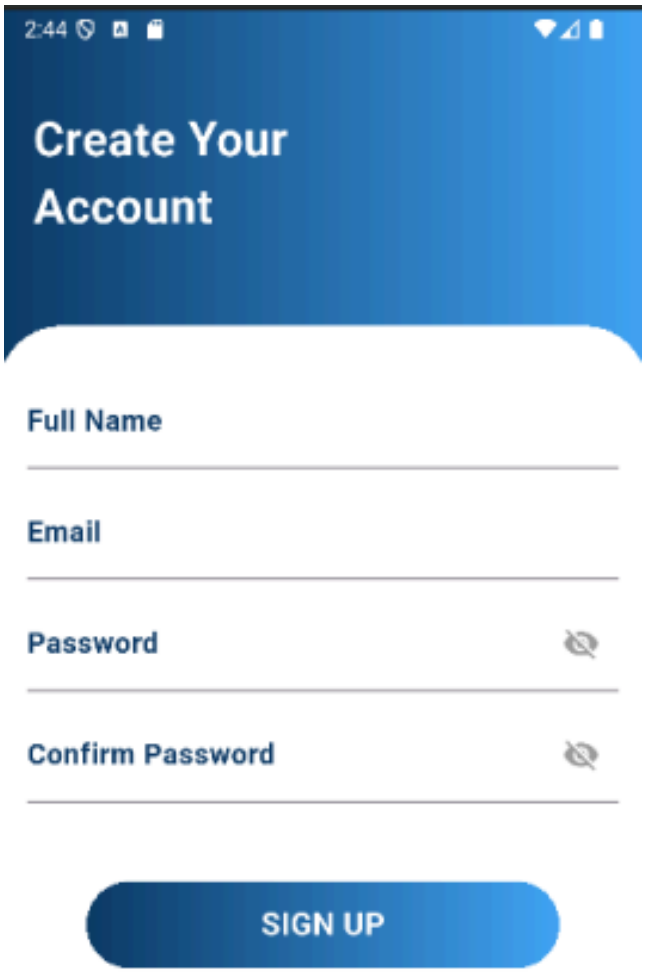
```
      UserCredential userCredential = await FirebaseAuth.instance
          .createUserWithEmailAndPassword(
          email: _emailController.text, password: _passwordController.text);

      // Set display name after user is created
      await userCredential.user?.updateDisplayName(_nameController.text);

      // Send email verification
      await userCredential.user?.sendEmailVerification();

      // Show success toast
      Fluttertoast.showToast(
        msg: "Registration Successful! Please verify your email.",
        toastLength: Toast.LENGTH_SHORT,
        gravity: ToastGravity.BOTTOM,
        timeInSecForIosWeb: 1,
        backgroundColor: AppColors.successColor,
        textColor: AppColors.textColor,
        fontSize: 16.0,
      );

      // Navigate to VerifyEmailScreen
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => VerifyEmailScreen()),
      );
    } catch (e) {
      // Show error toast
      Fluttertoast.showToast(
        msg: "Error: ${e.toString()}",
        toastLength: Toast.LENGTH_SHORT,
        gravity: ToastGravity.BOTTOM,
        timeInSecForIosWeb: 1,
        backgroundColor: AppColors.errorColor,
        textColor: AppColors.textColor,
        fontSize: 16.0,
      );
    } finally {
      setState(() => _isLoading = false);
    }
  }
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    body: Stack(
      children: [
        // Background Gradient
        Container(
```

```
          height: double.infinity,
          width: double.infinity,
          decoration: const BoxDecoration(
            gradient: LinearGradient(
              colors: [AppColors.primaryColor, AppColors.secondaryColor],
            ),
          ),
          child: const Padding(
            padding: EdgeInsets.only(top: 60.0, left: 22),
            child: Text(
              'Create Your\nAccount',
              style: TextStyle(fontSize: 30, color: Colors.white, fontWeight:
FontWeight.bold),
            ),
          ),
        ),
        Padding(
          padding: const EdgeInsets.only(top: 200.0),
          child: Container(
            decoration: const BoxDecoration(
              borderRadius: BorderRadius.only(topLeft: Radius.circular(40),
topRight: Radius.circular(40)),
              color: AppColors.backgroundColor,
            ),
            height: double.infinity,
            width: double.infinity,
            child: SingleChildScrollView(
              padding: const EdgeInsets.symmetric(horizontal: 18.0, vertical: 30),
              child: Form(
                key: _formKey,
                child: Column(
                  children: [
                    _buildTextField('Full Name', _nameController, false,
Validator.validateName, _nameFocusNode),
                    const SizedBox(height: 10),
                    _buildTextField('Email', _emailController, false,
Validator.validateEmail, _emailFocusNode),
                    const SizedBox(height: 10),
                    _buildTextField('Password', _passwordController, true,
Validator.validatePassword, _passwordFocusNode),
                    const SizedBox(height: 10),
                    _buildTextField('Confirm Password', _confirmPasswordController,
true, (value) {
                      return Validator.validateConfirmPassword(value ?? '',
_passwordController.text);
                    }, _confirmPasswordFocusNode),
                    const SizedBox(height: 50),
                    GestureDetector(
                      onTap: _isLoading ? null : _registerUser,
                      child: Container(
```

```
                    height: 55,
                    width: 300,
                    decoration: BoxDecoration(
                      borderRadius: BorderRadius.circular(30),
                      gradient: const LinearGradient(
                        colors: [AppColors.primaryColor, AppColors.secondaryColor],
                      ),
                    ),
                    child: Center(
                      child: _isLoading
                          ? const CircularProgressIndicator(color: Colors.white)
                          : const Text(
                        'SIGN UP',
                        style: TextStyle(fontWeight: FontWeight.bold, fontSize: 20,
color: AppColors.buttonTextColor),
                      ),
                    ),
                  ),
                ),
                const SizedBox(height: 50),
              ],
            ),
          ),
        ),
      ),
    ],
  ),
 );
}
```

Output:


Hello
Sign in!
Gmail
Password
Forgot Password?
SIGN IN
Don't have an account?
Sign up



Conclusion: We learned how to include icons, images, and custom fonts in a Flutter application to improve the visual quality and user experience.

- Icons help to visually communicate actions or features to users, enhancing the user interface.
- Images are a powerful tool for adding visual context, branding, and dynamic content.
- Custom fonts allow you to personalize the app's text style and ensure it aligns with your brand identity.

Mastering the integration of these elements is essential for developing visually appealing and user-friendly Flutter applications.