Experiment No:03

Aim : To explore Flutter Widgets like image,icon and to use custom fonts.

Theory :

In the below code we use the following widgets:

1.MaterialApp: This widget represents a Flutter application that uses material design.
2.Scaffold: Scaffold is a layout structure widget from the Material library that provides a default layout structure for the app. It includes an app bar, a body, and other structural elements.
3.AppBar: AppBar is a Material Design app bar that displays the title and other actions above the app's main content.
4.SingleChildScrollView: This widget enables scrolling when the content is too large to fit within the visible area. It allows the child widget to be scrolled in one direction.
5.Column: Column is a layout widget that arranges its children vertically, one after another.
6.Image: The Image widget displays an image. In this code, it's used to display the login page's logo.
7.SizedBox: SizedBox is a widget that creates a fixed-size box. It's used here to create space between widgets vertically.
8.TextField: TextField is a widget that allows users to enter text. It's used here for the email and password input fields.
9.ElevatedButton: ElevatedButton is a button widget with a raised appearance, typically used for primary actions.
10.TextButton: TextButton is a button widget with only text, suitable for secondary actions.
11.Text: Text is a widget that displays a string of text. It's used to provide labels and button text in this code.

These widgets are used to create a simple login page with email and password input fields, along with login and forgot password buttons.

Code :

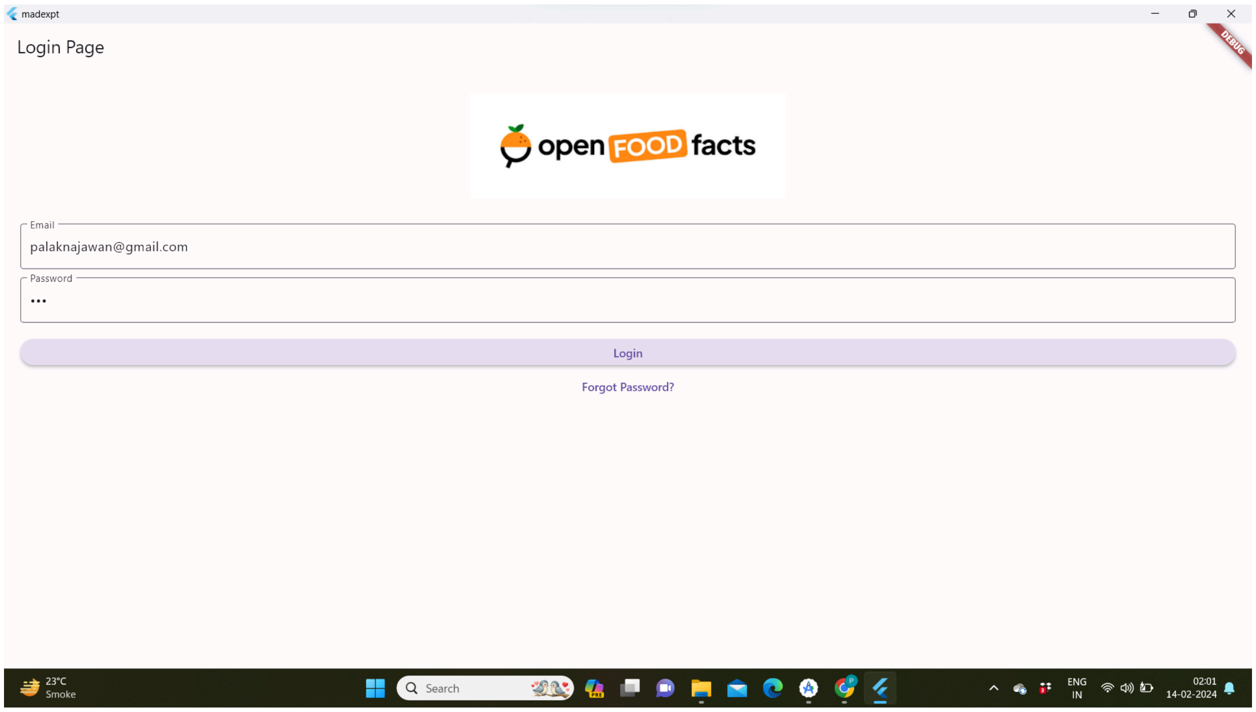
```
import 'package:flutter/material.dart';

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

class LoginPage extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return MaterialApp(
   home: Scaffold(
    appBar: AppBar(
     title: Text('Login Page'),
    ),
    body: SingleChildScrollView(
     padding: EdgeInsets.all(20.0),
     child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      crossAxisAlignment: CrossAxisAlignment.stretch,
      children: <Widget>[
       Image(
        image: AssetImage('images/download.png'),
        height: 150.0,
        width: 150.0,
       ),
       SizedBox(height: 20.0),
       TextField(
        decoration: InputDecoration(
         labelText: 'Email',
         border: OutlineInputBorder(),
        ),
       ),
       SizedBox(height: 10.0),
       TextField(
```

```dart
                obscureText: true,
                decoration: InputDecoration(
                  labelText: 'Password',
                  border: OutlineInputBorder(),
                ),
              ),
              SizedBox(height: 20.0),
              ElevatedButton(
                onPressed: () {
                  // Add login logic here
                },
                child: Text('Login'),
              ),
              SizedBox(height: 10.0),
              TextButton(
                onPressed: () {
                  // Add forgot password logic here
                },
                child: Text('Forgot Password?'),
              ),
            ],
          ),
        ),
      ),
    );
  }
}
```

Output:



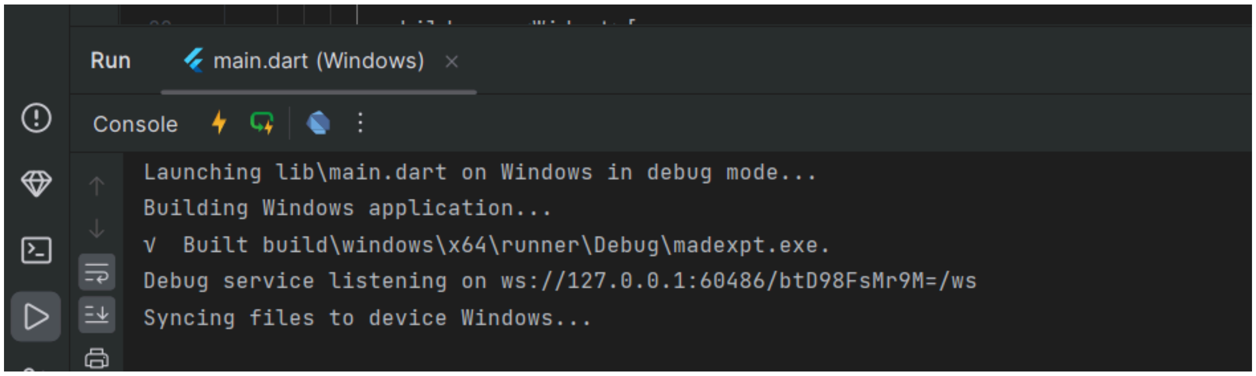
Login Page

Email
palaknajawan@gmail.com

Password
•••

Login

Forgot Password?



```
Launching lib\main.dart on Windows in debug mode...
Building Windows application...
√  Built build\windows\x64\runner\Debug\madexpt.exe.
Debug service listening on ws://127.0.0.1:60486/btD98FsMr9M=/ws
Syncing files to device Windows...
```

Conclusion:

Overall, mastering these widgets empowers developers to create visually appealing and cohesive user interfaces that align with design requirements and brand aesthetics. Additionally, Flutter's flexibility and ease of use make it straightforward to incorporate images, icons, and custom fonts into applications, enhancing the overall user experience and engagement.