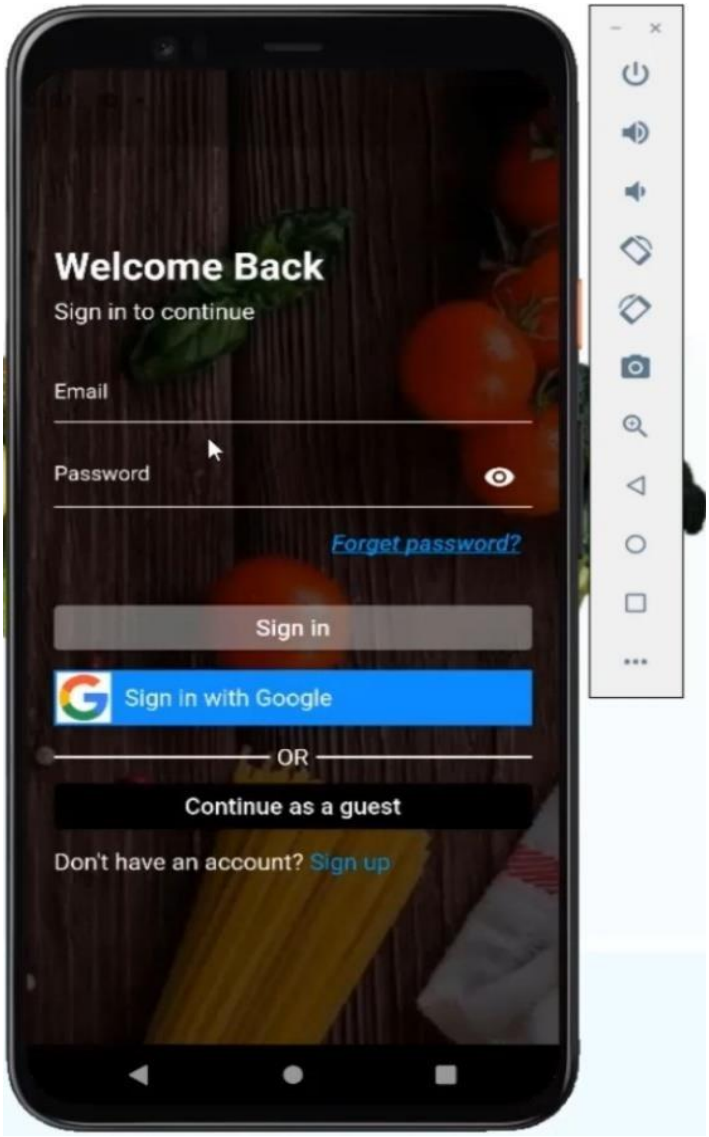
Experiment No.3
MAD & PWA LAB

- **Aim:** To include icons, images, fonts in Flutter app.

- **Theory:**

An Icon is a graphic image representing an application or any specific entity containing meaning for the user. It can be selectable and unselectable. Flutter provides an Icon widget to create icons in our applications. We can create icons in Flutter, either using inbuilt icons or with the custom icons. Flutter provides the list of all icons in the Icons class. To use this class, make sure you set uses-material-design: true in your project's pubspec.yaml file in the flutter section. This ensures that the Material Icons font is included in your application. This font is used to display the icons.



1. Icon:

- Description: Icons in Flutter are used to display graphical symbols representing an action, state, or concept. Flutter provides a wide range of built-in icons representing common actions, such as navigation, communication, and more.

- Usage: Icons can be used within various Flutter widgets, including AppBar, IconButton, ListTile, and more. They can be easily customized with properties like size, color, and opacity.

- Custom Icons: Besides built-in icons, Flutter also allows developers to use custom icons by importing them into the project and referencing them using the IconData class.

- Scalability: Icons in Flutter are vector graphics, which means they can scale without losing quality, making them suitable for different screen sizes and resolutions.

2. Image:

- Description: Images in Flutter are used to display visual content, such as pictures, photos, and graphics, within the app's user interface. Flutter supports various image formats, including JPEG, PNG, GIF, WebP, and animated WebP.

- Image Loading: Images can be loaded from different sources, such as assets (images bundled with the app), the internet (via network requests), and memory (decoded from raw byte data).

- Asset Images: Asset images are images bundled with the app and can be loaded using the AssetImage class. These images should be placed in the project's "assets" folder and referenced in the pubspec.yaml file.

- Network Images : Network images are loaded from URLs using the NetworkImage class. Flutter automatically handles caching and network requests for efficient image loading.

- Memory Images: Memory images are created from raw byte data using the MemoryImage class, which is useful for dynamically generated images.

- Image Widgets: Flutter provides several widgets for displaying images, including Image, AssetImage, NetworkImage, and MemoryImage. These widgets support various properties for customization, such as width, height, fit, alignment, and loading placeholders.

3. Font :

- Description: Fonts in Flutter are used to define the visual appearance of text within the app. Flutter supports various font formats, including TrueType (TTF) and OpenType (OTF), allowing developers to choose from a wide range of typefaces and styles.

- Font Loading: Fonts can be loaded from different sources, such as assets (font files bundled with the app) and the internet (via network requests).

- Asset Fonts: Asset fonts are font files bundled with the app and can be loaded using the TextStyle class with the fontFamily property. These font files should be placed in the project's "assets/fonts" folder and referenced in the pubspec.yaml file.

- Google Fonts: Flutter provides seamless integration with Google Fonts, allowing developers to use custom fonts hosted on Google's servers. The google_fonts package simplifies the process of loading and applying Google Fonts in Flutter apps.

- Custom Fonts: Besides built-in and Google Fonts, Flutter also supports custom fonts loaded from other sources, such as locally stored font files or custom font services.

- Text Styling: Flutter provides extensive support for text styling, including properties for font size, font weight, font style, letter spacing, line height, text alignment, text direction, and text decoration. These properties can be applied using the TextStyle class to customize the visual appearance of text widgets.

Code:

```
import 'package:card_swiper/card_swiper.dart'; import
'package:firebase_auth/firebase_auth.dart'; import
'package:flutter/gestures.dart'; import
'package:flutter/material.dart'; import
'package:grocery_app/screens/auth/forget_pass.dart'; import
'package:grocery_app/screens/auth/register.dart'; import
'package:grocery_app/screens/btm_bar.dart'; import
'package:grocery_app/screens/loading_manager.dart'; import
'package:grocery_app/services/global_methods.dart';

import '../../consts/contss.dart'; import
'../../consts/firebase_consts.dart'; import
'../../fetch_screen.dart'; import
'../../widgets/auth_button.dart'; import
```

```
'../../widgets/google_button.dart';
import '../../widgets/text_widget.dart';

class LoginScreen extends StatefulWidget {
static const routeName = '/LoginScreen';
  const LoginScreen({Key? key}) : super(key: key);

  @override
  State<LoginScreen> createState() => _LoginScreenState();
}

class _LoginScreenState extends State<LoginScreen> {
final _emailTextController = TextEditingController();
final _passTextController = TextEditingController();
final _passFocusNode = FocusNode();   final
_formKey = GlobalKey<FormState>();
  var _obscureText = true;
@override   void
dispose() {
    _emailTextController.dispose();
    _passTextController.dispose();
_passFocusNode.dispose();     super.dispose();
  }

  bool _isLoading = false;   void
_submitFormOnLogin() async {     final isValid =
_formKey.currentState!.validate();
    FocusScope.of(context).unfocus();

    if (isValid) {
      _formKey.currentState!.save();
      setState(() {
_isLoading = true;
      });
try {
        await authInstance.signInWithEmailAndPassword(
email: _emailTextController.text.toLowerCase().trim(),
password: _passTextController.text.trim());
        Navigator.of(context).pushReplacement(
          MaterialPageRoute(
            builder: (context) => const FetchScreen(),
          ),
```

```
      );
      print('Succefully logged in');      } on
FirebaseException    catch    (error)    {
GlobalMethods.errorDialog(
          subtitle: '${error.message}', context: context);
setState(() {          _isLoading = false;
      });
    } catch (error) {
      GlobalMethods.errorDialog(subtitle: '$error', context: context);
setState(() {          _isLoading = false;
      });
    } finally {
setState(() {
_isLoading = false;
      });
    }
  }
 }

 @override
 Widget build(BuildContext context) {
  return Scaffold(      body:
LoadingManager(
isLoading: _isLoading,
child: Stack(children: [
Swiper(           duration:
800,           autoplayDelay:
8000,
          itemBuilder: (BuildContext context, int index) {
            return Image.asset(
              Constss.authImagesPaths[index],
              fit: BoxFit.cover,
            );           },
autoplay: true,
          itemCount: Constss.authImagesPaths.length,
        ),
        Container(
          color: Colors.black.withOpacity(0.7),
        ),
        SingleChildScrollView(
child: Padding(
```

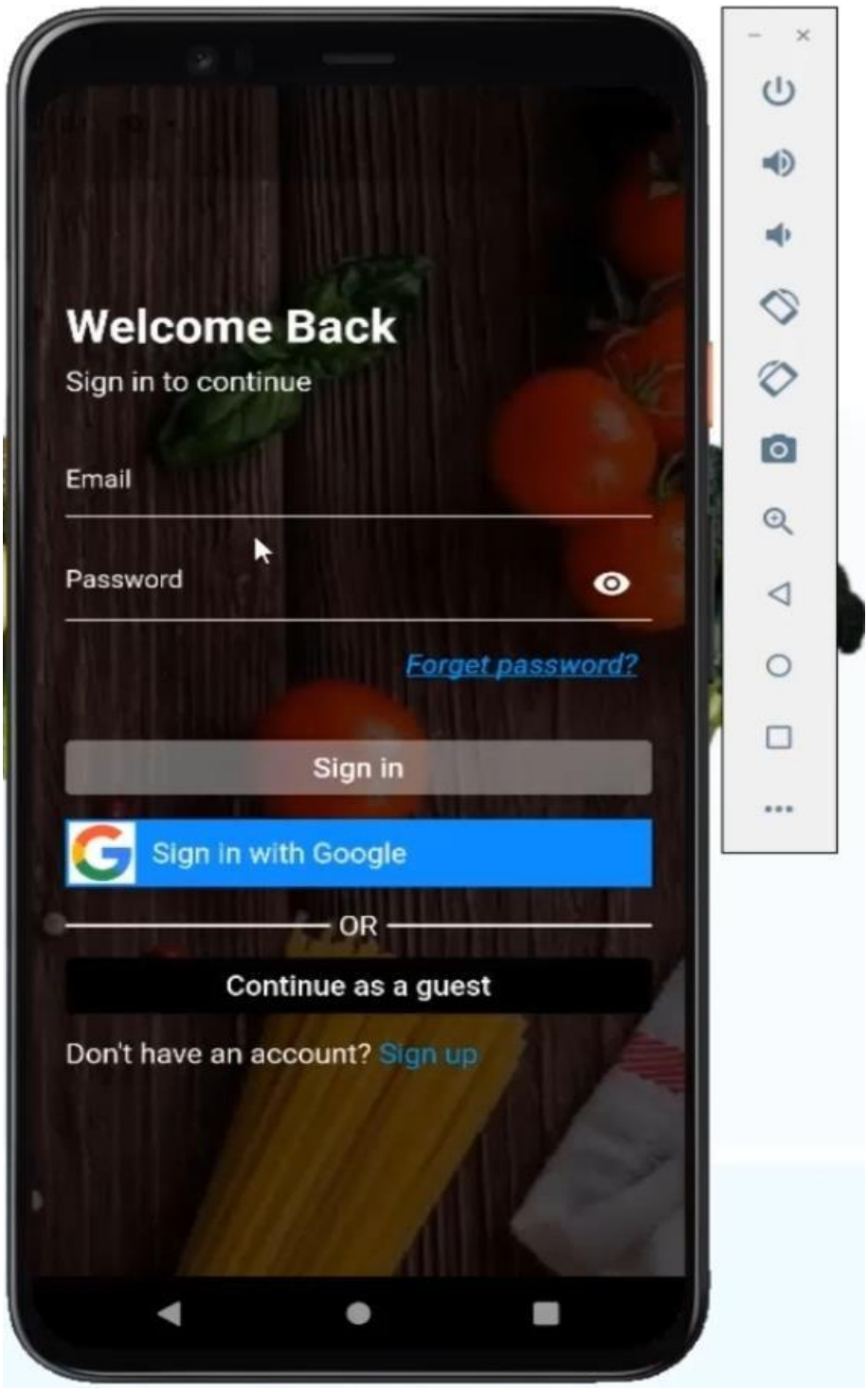
```
padding: const EdgeInsets.all(20.0),
child: Column(
mainAxisAlignment: MainAxisAlignment.center,
crossAxisAlignment: CrossAxisAlignment.start, mainAxisSize:
MainAxisSize.max,
children: [
const SizedBox(
height: 120.0,
),
TextWidget(
text: 'Welcome Back',
color: Colors.white,
textSize: 30, isTitle:
true,
),
const SizedBox(
height: 8,
),
TextWidget(
text: "Sign in to continue",
color: Colors.white,
textSize: 18, isTitle: false,
),
const SizedBox(
height: 30.0,
), Form(
key: _formKey,
child: Column(
children: [
TextFormField(
textInputAction: TextInputAction.next,
onEditingComplete: () => FocusScope.of(context)
.requestFocus(_passFocusNode),
controller: _emailTextController,
keyboardType: TextInputType.emailAddress,
validator: (value) {
if (value!.isEmpty || !value.contains('@')) {
return 'Please enter a valid email address';
} else {
return null;
}
},
```

```
                    style: const TextStyle(color: Colors.white),
decoration: const InputDecoration(
                     hintText: 'Email',                    hintStyle:
TextStyle(color:                                  Colors.white),
enabledBorder: UnderlineInputBorder(
                       borderSide: BorderSide(color: Colors.white),
                     ),
                     focusedBorder: UnderlineInputBorder(
borderSide: BorderSide(color: Colors.white),
                     ),
                    ),
                   ),
                   SizedBox(
height: 12,                    ),
                   //Password

                   TextFormField(
                    textInputAction:  TextInputAction.done,
onEditingComplete:                ()                {
_submitFormOnLogin();
                    },
                    controller: _passTextController,
focusNode: _passFocusNode,                    obscureText:
_obscureText,
                    keyboardType: TextInputType.visiblePassword,
                    validator: (value) {                    if
(value!.isEmpty || value.length < 7) {
return 'Please enter a valid password';
                     } else {
return null;
                     }
                    },
                    style: const TextStyle(color: Colors.white),
                    decoration: InputDecoration(
                     suffixIcon: GestureDetector(
                        onTap: () {
setState(() {
                          _obscureText = !_obscureText;
                         });
},                        child: Icon(
_obscureText
```

```
                            ?       Icons.visibility
: Icons.visibility_off,                         color:
Colors.white,
                          )),
                        hintText: 'Password',                   hintStyle:
const                     TextStyle(color:                  Colors.white),
enabledBorder: const UnderlineInputBorder(
                          borderSide: BorderSide(color: Colors.white),
                        ),
                        focusedBorder: const UnderlineInputBorder(
                          borderSide: BorderSide(color: Colors.white),
                        ),
                      ),
                    ),
                  ],
                )),
            const SizedBox(
height: 10,
            ),
Align(
              alignment: Alignment.topRight,
              child: TextButton(
onPressed: () {
                GlobalMethods.navigateTo(
ctx: context,
                    routeName: ForgetPasswordScreen.routeName);
                },                  child:
const Text(                     'Forget
password?',                 maxLines:
1,
                  style: TextStyle(
                      color: Colors.lightBlue,
                      fontSize: 18,
                      decoration: TextDecoration.underline,
                      fontStyle: FontStyle.italic),
                ),
              ),
            ),
            const SizedBox(
height: 10,
            ),
```

```
            AuthButton(                fct:
_submitFormOnLogin,
buttonText: 'Login',
            ),
            const SizedBox(
height: 10,
            ),
            GoogleButton(),
const SizedBox(
height: 10,
            ),
            Row(
children: [                const
Expanded(                  child:
Divider(                   color:
Colors.white,
thickness: 2,
                ),
              ),
              const SizedBox(
width: 5,                ),
              TextWidget(
text: 'OR',                      color:
Colors.white,              textSize:
18,
              ),
              const SizedBox(
width: 5,
              ),
              const Expanded(
child: Divider(
color: Colors.white,
thickness: 2,
                ),
              ),
            ],
            ),
            const SizedBox(
height: 10,
            ),
            AuthButton(
              fct: () {
```

```
                Navigator.of(context).push(
MaterialPageRoute(
                  builder: (context) => const FetchScreen(),
                ),
               );
              },
               buttonText: 'Continue as a guest',
primary: Colors.black,
             ),
             const SizedBox(
height: 10,
             ),
             RichText(                    text: TextSpan(
text: 'Don\'t have an account?',                    style:
const TextStyle(                    color: Colors.white,
fontSize: 18),
                  children: [
TextSpan(                    text: '  Sign
up',                    style: const
TextStyle(                    color:
Colors.lightBlue,
fontSize: 18,
                      fontWeight: FontWeight.w600),
recognizer: TapGestureRecognizer()
                    ..onTap = () {
                      GlobalMethods.navigateTo(
                          ctx: context,
                          routeName: RegisterScreen.routeName);
                    }),
                ]))
            ],
          ),
        ),
      )
]),
    ),
  );
}

}
```



- **Conclusion:**

  Hence, we understood how to include icons, images, fonts that have been used in making the Login Screen of our application (Grocery App).