Vivekanand Education Society's Institute of Technology

An Autonomous Institute Affiliated to University of Mumbai
Hashu Advani Memorial Complex, Collector Colony, Chembur East, Mumbai - 400074.

Department of Information Technology

CERTIFICATE

This is to certify that **MOKSHA JITENDRA GAWADA** of **D15A** semester **VI,** have successfully completed necessary experiments in the **MAD & PWA Lab** under my supervision in **VES Institute of Technology** during the academic year **2023-2024.**

Lab Assistant

Subject Teacher

Mrs. Kajal Joseph

Principal

Head of Department

Dr. Mrs. Shalu Chopra

Name of the Course : MAD & PWA Lab **Course Code :** ITL604

Year/Sem/Class : D15A/D15B **A.Y.:** 23-24

Faculty Incharge : Mrs. Kajal Joseph.

Lab Teachers : Mrs. Kajal Jewani.

Email : kajal.jewani@ves.ac.in

Programme Outcomes: The graduate will be able to:

PO1) Basic Engineering knowledge: An ability to apply the fundamental knowledge in mathematics, science and engineering to solve problems in Computer engineering.

PO2) Problem Analysis: Identify, formulate, research literature and analyze computer engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences and computer engineering and sciences.

PO3) Design/ Development of Solutions: Design solutions for complex computer engineering problems and design system components or processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal and environmental considerations.

PO4) Conduct investigations of complex engineering problems using research-based knowledge and research methods including design of experiments, analysis and interpretation of data and synthesis of information to provide valid conclusions.

PO5) Modern Tool Usage: Create, select and apply appropriate techniques, resources and modern computer engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6) The Engineer and Society: Apply reasoning informed by contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to computer engineering practice.

PO7) Environment and Sustainability: Understand the impact of professional computer engineering solutions in societal and environmental contexts and demonstrate knowledge of and need for sustainable development.

PO8) Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of computer engineering practice.

PO9) Individual and Team Work: Function effectively as an individual, and as a member or leader in diverse teams and in multidisciplinary settings.

PO10) Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations and give and receive clear instructions.

PO11) Project Management and Finance: Demonstrate knowledge and understanding of computer engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12) Life-long Learning: Recognize the need for and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

Program specific Outcomes

PSO1) An ability to manage and analyze data / information effectively for making better decisions.

PSO2) Demonstrate the ability to use state of the art technologies and tools including Free and Open Source Software (FOSS) tools in developing software.

Project Title:Flutter:SkypeClone / PWA:Fuzzy App **Roll No. 18**

Lab Objectives:

Sr. No.	Lab Objectives
The Lab experiments aims:	
1	Learn the basics of the Flutter framework.
2	Develop the App UI by incorporating widgets, layouts, gestures and animation
3	Create a production ready Flutter App by including files and firebase backend service.
4	Learn the Essential technologies, and Concepts of PWAs to get started as quickly and efficiently as possible
5	Develop responsive web applications by combining AJAX development techniques with the jQuery JavaScript library.
6	Understand how service workers operate and also learn to Test and Deploy PWA.

Lab Outcomes:

Sr. No.	Lab Outcomes	Cognitive levels of attainment as per Bloom's Taxonomy
On Completion of the course the learner/student should be able to:		
1	Understand cross platform mobile application development using Flutter framework	L1, L2
2	Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation	L3
3	Analyze and Build production ready Flutter App by incorporating backend services and deploying on Android / iOS	L3, L4
4	Understand various PWA frameworks and their requirements	L1, L2
5	Design and Develop a responsive User Interface by applying PWA Design techniques	L3
6	Develop and Analyse PWA Features and deploy it over app hosting solutions	L3, L4

Index

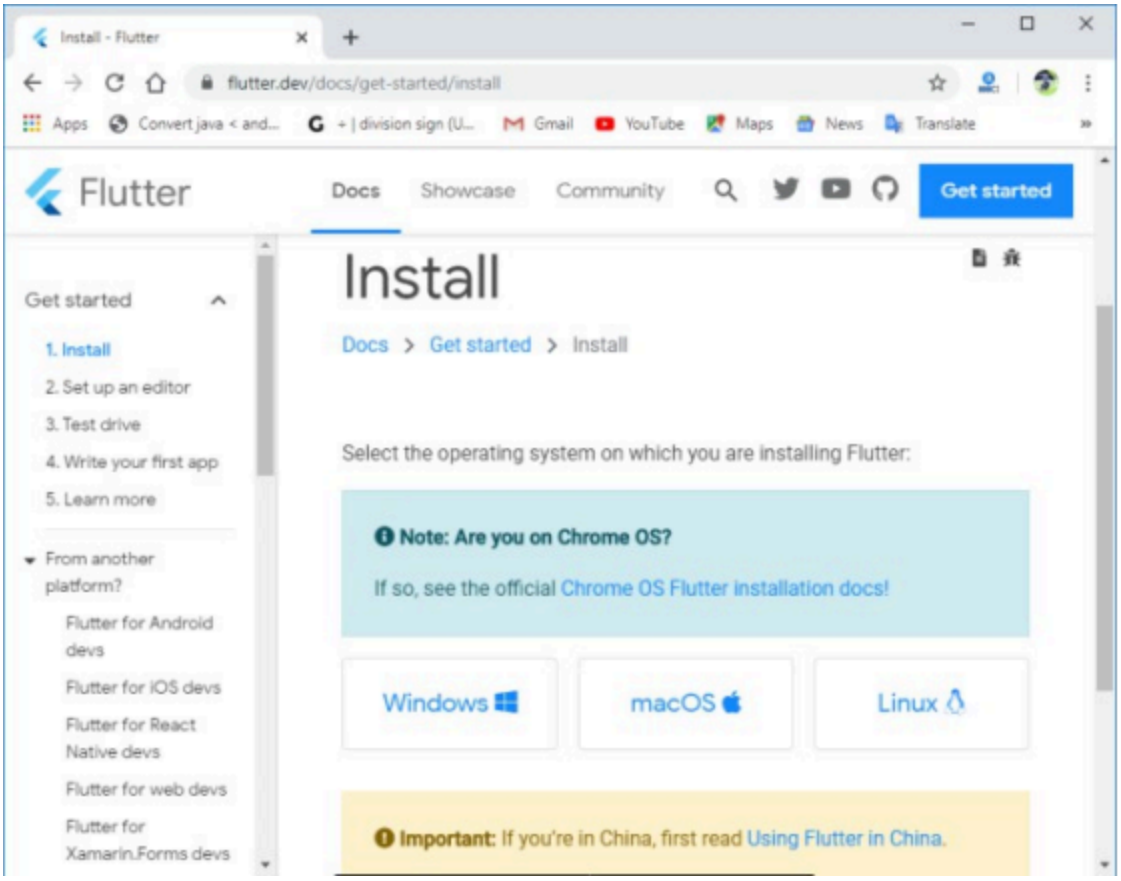
Sr. No	Experiment Title	LO	DOP	DOS	Grade
1.	To install and configure the Flutter Environment	LO1	16/1	23/1	15
2.	To design Flutter UI by including common widgets.	LO2	23/1	30/1	15
3.	To include icons, images, fonts in Flutter app	LO2	30/1	6/2	15
4.	To create an interactive Form using form widget	LO2	6/2	13/2	10
5.	To apply navigation, routing and gestures in Flutter App	LO2	13/2	20/2	15
6.	To Connect Flutter UI with fireBase database	LO3	20/2	5/3	15
7.	To write meta data of your Ecommerce PWA in a Web app manifest file to enable “add to homescreen feature”.	LO4	5/3	12/3	15
8.	To code and register a service worker, and complete the install and activation process for a new service worker for the E-commerce PWA	LO5	12/3	19/3	15
9.	To implement Service worker events like fetch, sync and push for E-commerce PWA	LO5	19/3	26/3	15
10.	To study and implement deployment of Ecommerce PWA to GitHub Pages.	LO5	26/3	2/4	15
11.	To use google Lighthouse PWA Analysis Tool to test the PWA functioning.	LO6	5/3	12/3	15
12.	Assignment-1	LO1,LO2 ,LO3	2/2	5/2	5
13.	Assignment-2	LO4,LO5 ,LO6	19/3	21/3	4

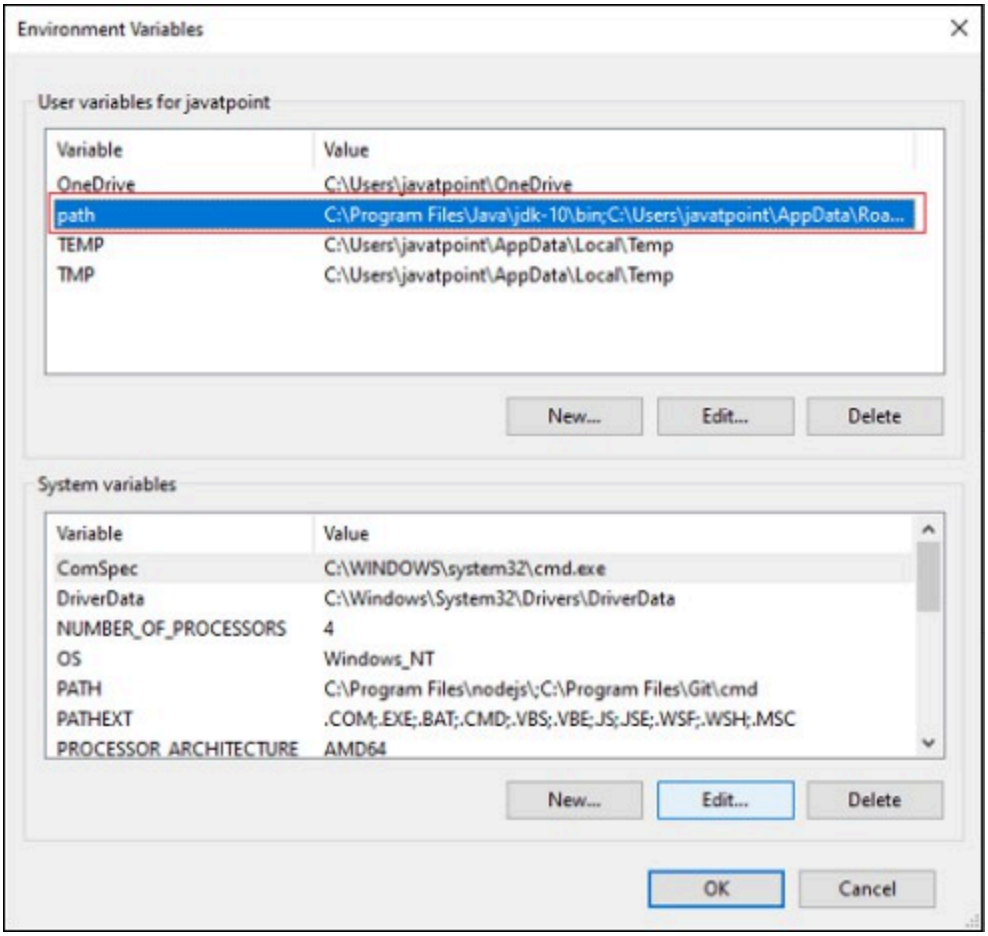
MAD & PWA Lab
Journal

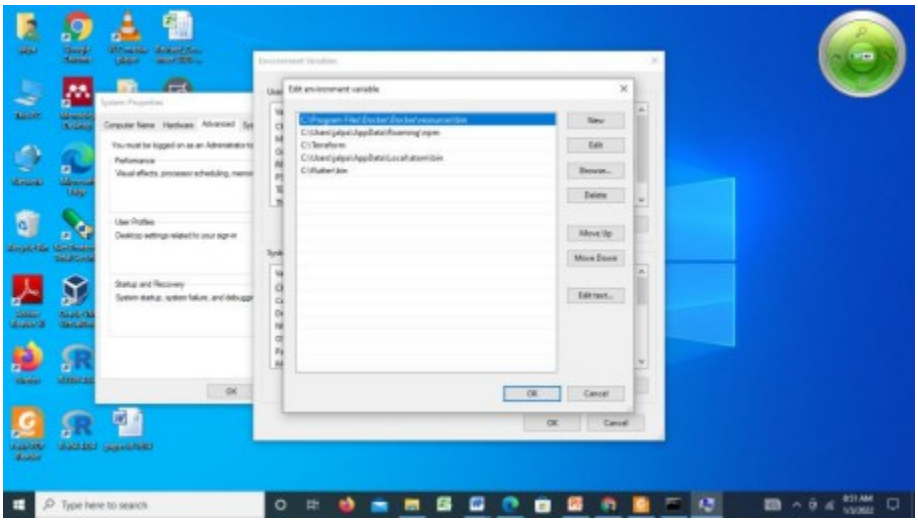
Experiment No.	01
Experiment Title.	To install and configure the Flutter Environment
Roll No.	18
Name	Moksha Gawada
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO1: Understand cross platform mobile application development using Flutter framework
Grade:	15

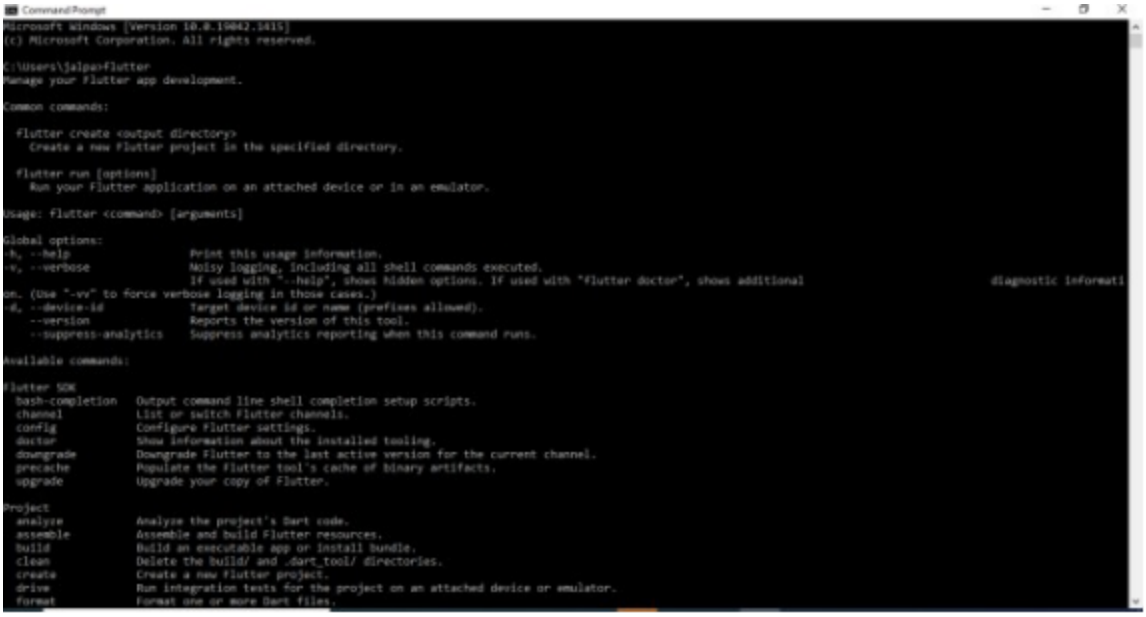
MAD PWA EXPERIMENT-01

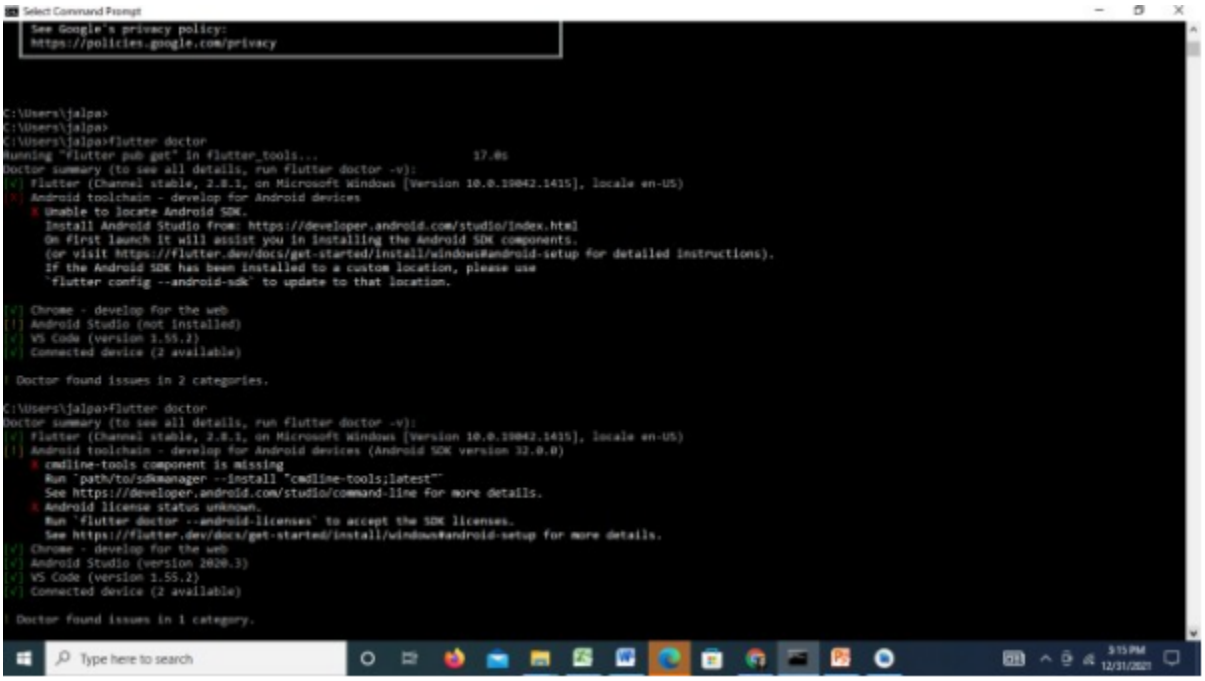
AIM:Installation and Configuration of Flutter Environment.


Install - Flutter
flutter.dev/docs/get-started/install
Apps
Convert java < and...
+ | division sign (U...
Gmail
YouTube
Maps
News
Translate
Flutter
Docs
Showcase
Community
Get started
Get started
1. Install
2. Set up an editor
3. Test drive
4. Write your first app
5. Learn more
From another platform?
Flutter for Android devs
Flutter for iOS devs
Flutter for React Native devs
Flutter for web devs
Flutter for Xamarin.Forms devs
Install
Docs > Get started > Install
Select the operating system on which you are installing Flutter:
Note: Are you on Chrome OS?
If so, see the official Chrome OS Flutter installation docs!
Windows
macOS
Linux
Important: If you're in China, first read Using Flutter in China.

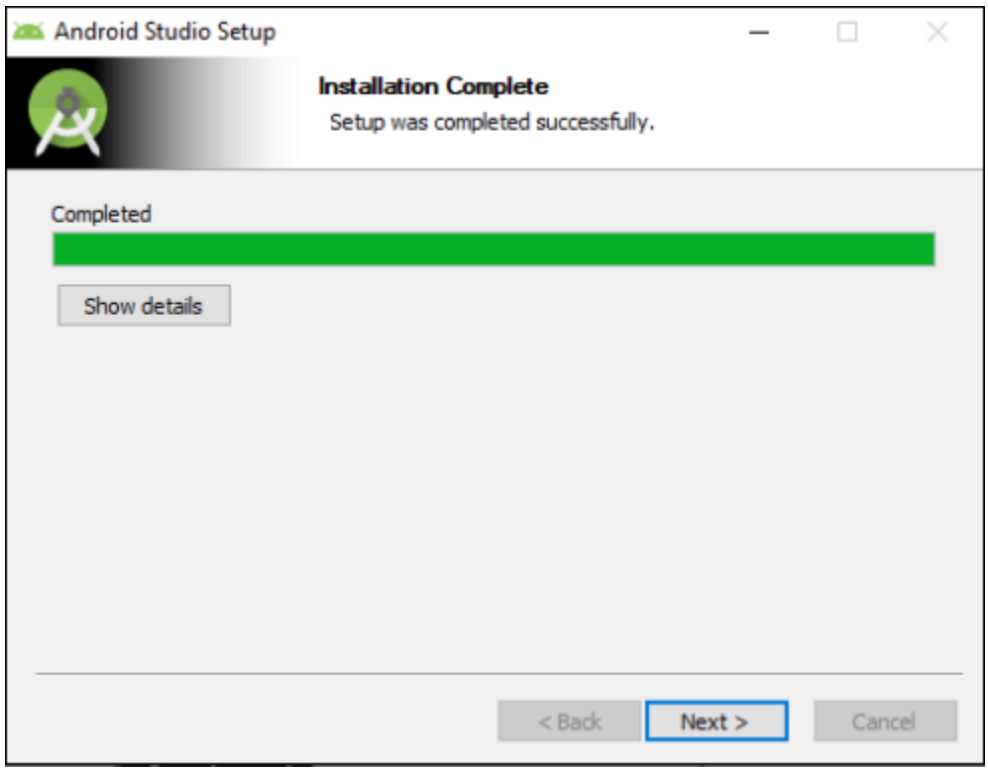

Environment Variables
User variables for javatpoint
Variable
Value
OneDrive
C:\Users\javatpoint\OneDrive
path
C:\Program Files\Java\jdk-10\bin;C:\Users\javatpoint\AppData\Roa...
TEMP
C:\Users\javatpoint\AppData\Local\Temp
TMP
C:\Users\javatpoint\AppData\Local\Temp
New...
Edit...
Delete
System variables
Variable
Value
ComSpec
C:\WINDOWS\system32\cmd.exe
DriverData
C:\Windows\System32\Drivers\DriverData
NUMBER_OF_PROCESSORS
4
OS
Windows_NT
PATH
C:\Program Files\nodejs\;C:\Program Files\Git\cmd
PATHEXT
.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE
AMD64
New...
Edit...
Delete
OK
Cancel

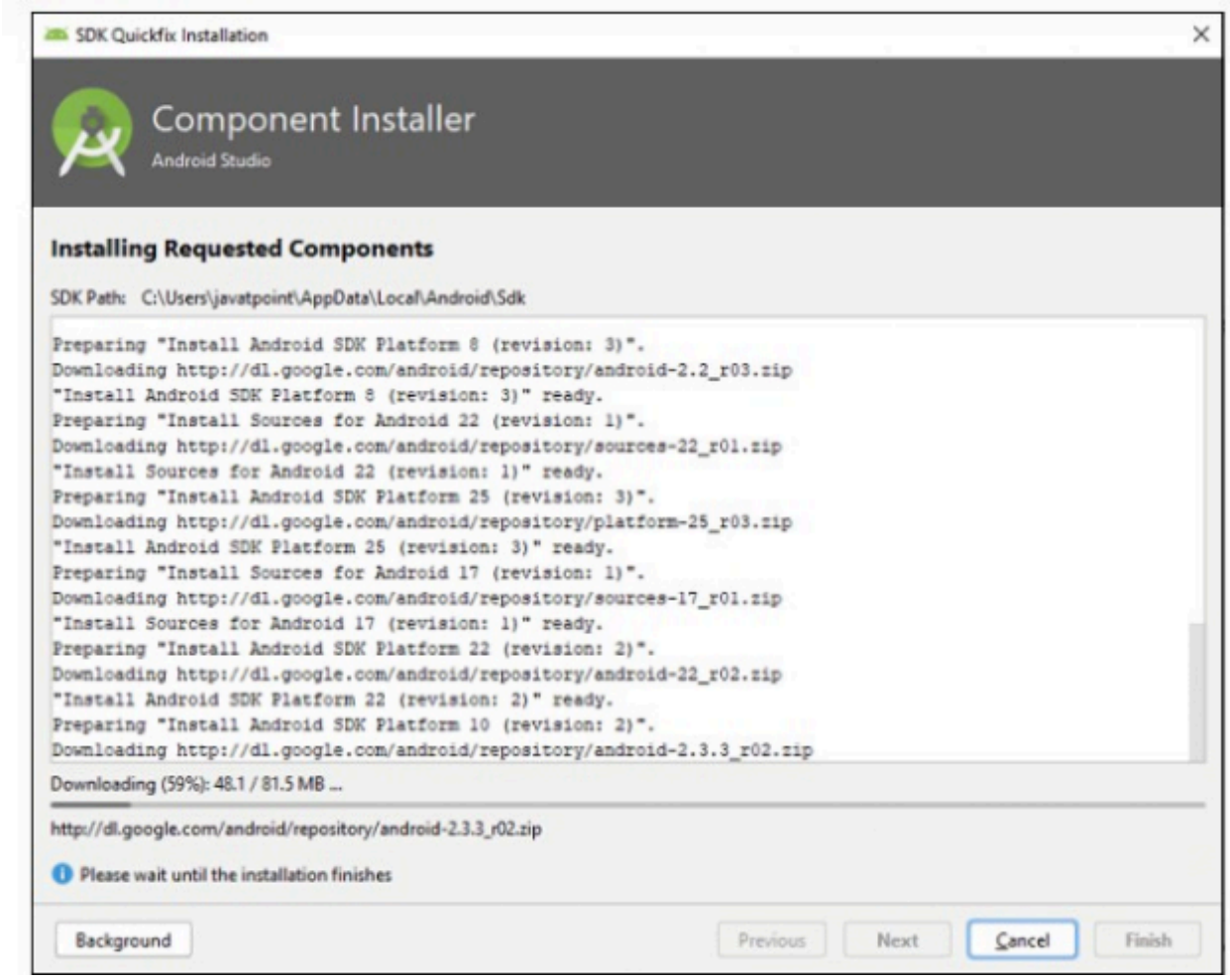


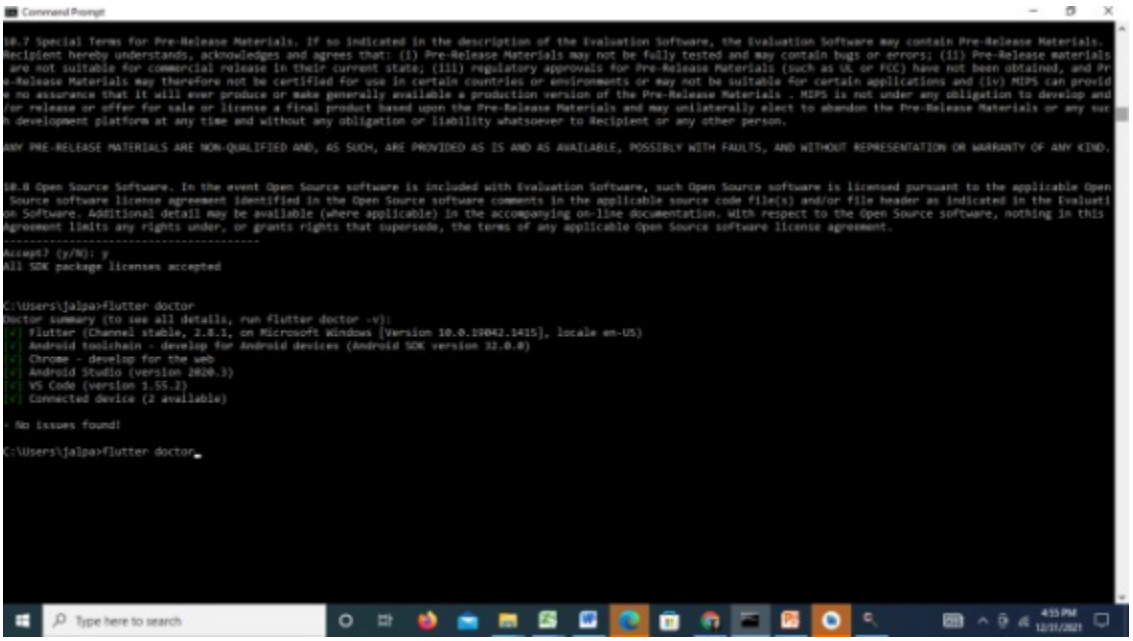

Command Prompt
Microsoft Windows [Version 10.0.19042.1415]
(c) Microsoft Corporation. All rights reserved.
C:\Users\jalpa>flutter
Manage your Flutter app development.
Common commands:
flutter create <output directory>
Create a new Flutter project in the specified directory.
flutter run [options]
Run your Flutter application on an attached device or in an emulator.
Usage: flutter <command> [arguments]
Global options:
-h, --help Print this usage information.
-v, --verbose Noisy logging, including all shell commands executed.
If used with "--help", shows hidden options. If used with "flutter doctor", shows additional diagnostic informati
on. (Use "-vv" to force verbose logging in those cases.)
-d, --device-id Target device id or name (prefixes allowed).
--version Reports the version of this tool.
--suppress-analytics Suppress analytics reporting when this command runs.
Available commands:
Flutter SDK
bash-completion Output command line shell completion setup scripts.
channel List or switch Flutter channels.
config Configure Flutter settings.
doctor Show information about the installed tooling.
downgrade Downgrade Flutter to the last active version for the current channel.
precache Populate the Flutter tool's cache of binary artifacts.
upgrade Upgrade your copy of Flutter.
Project
analyze Analyze the project's Dart code.
assemble Assemble and build Flutter resources.
build Build an executable app or install bundle.
clean Delete the build/ and .dart_tool/ directories.
create Create a new Flutter project.
drive Run integration tests for the project on an attached device or emulator.
format Format one or more Dart files.


Select Command Prompt
See Google's privacy policy:
https://policies.google.com/privacy
C:\Users\jalpa>
C:\Users\jalpa>
C:\Users\jalpa>flutter doctor
Running "flutter pub get" in flutter_tools... 17.0s
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, 2.8.1, on Microsoft Windows [Version 10.0.19042.1415], locale en-US)
[!] Android toolchain - develop for Android devices
✗ Unable to locate Android SDK.
Install Android Studio from: https://developer.android.com/studio/index.html
On first launch it will assist you in installing the Android SDK components.
(or visit https://flutter.dev/docs/get-started/install/windows#android-setup for detailed instructions).
If the Android SDK has been installed to a custom location, please use
`flutter config --android-sdk` to update to that location.
[√] Chrome - develop for the web
[!] Android Studio (not installed)
[√] VS Code (version 1.55.2)
[√] Connected device (2 available)
! Doctor found issues in 2 categories.
C:\Users\jalpa>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, 2.8.1, on Microsoft Windows [Version 10.0.19042.1415], locale en-US)
[!] Android toolchain - develop for Android devices (Android SDK version 32.0.0)
✗ cmdline-tools component is missing
Run `path/to/sdkmanager --install "cmdline-tools;latest"`
See https://developer.android.com/studio/command-line for more details.
✗ Android license status unknown.
Run `flutter doctor --android-licenses` to accept the SDK licenses.
See https://flutter.dev/docs/get-started/install/windows#android-setup for more details.
[√] Chrome - develop for the web
[√] Android Studio (version 2020.3)
[√] VS Code (version 1.55.2)
[√] Connected device (2 available)
! Doctor found issues in 1 category.
Type here to search
9:15 PM
12/31/2021

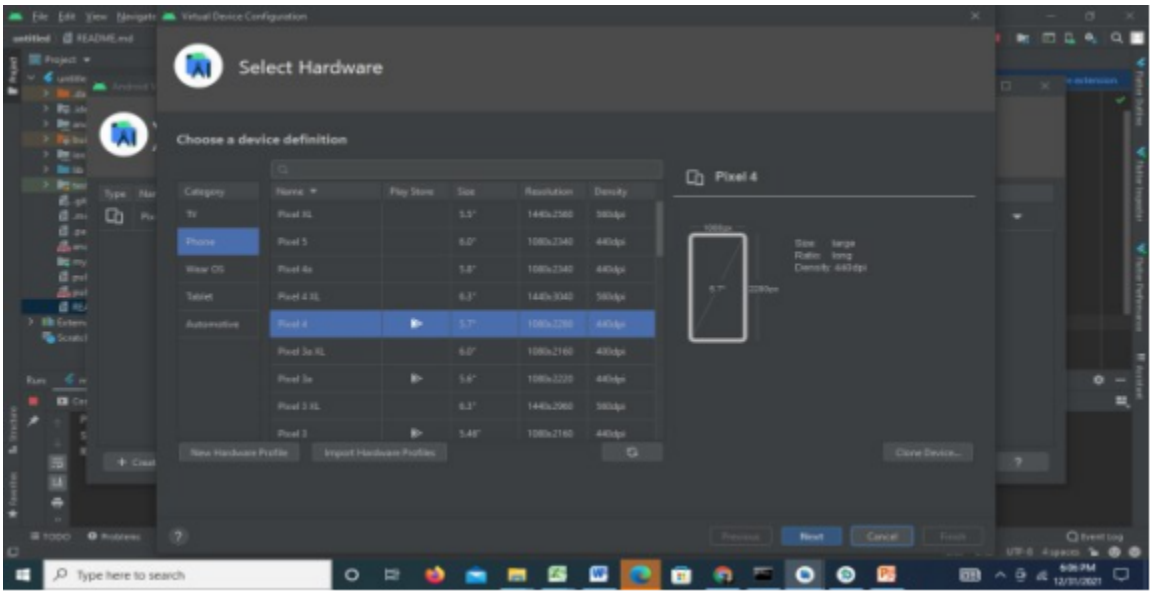

Android Studio Setup
Welcome to Android Studio Setup
Setup will guide you through the installation of Android Studio.
It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.
Click Next to continue.
Android Studio
< Back
Next >
Cancel


Android Studio Setup
Installation Complete
Setup was completed successfully.
Completed
Show details
< Back
Next >
Cancel

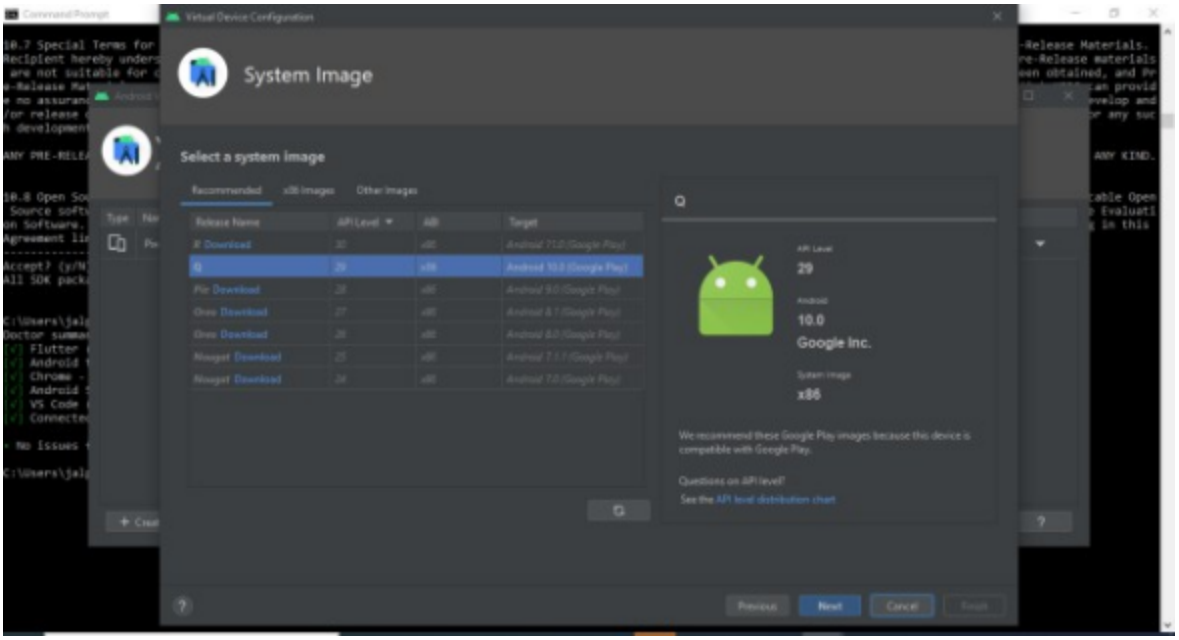

SDK Quickfix Installation
Component Installer
Android Studio
Installing Requested Components
SDK Path: C:\Users\javatpoint\AppData\Local\Android\Sdk
Preparing "Install Android SDK Platform 8 (revision: 3)".
Downloading http://dl.google.com/android/repository/android-2.2_r03.zip
"Install Android SDK Platform 8 (revision: 3)" ready.
Preparing "Install Sources for Android 22 (revision: 1)".
Downloading http://dl.google.com/android/repository/sources-22_r01.zip
"Install Sources for Android 22 (revision: 1)" ready.
Preparing "Install Android SDK Platform 25 (revision: 3)".
Downloading http://dl.google.com/android/repository/platform-25_r03.zip
"Install Android SDK Platform 25 (revision: 3)" ready.
Preparing "Install Sources for Android 17 (revision: 1)".
Downloading http://dl.google.com/android/repository/sources-17_r01.zip
"Install Sources for Android 17 (revision: 1)" ready.
Preparing "Install Android SDK Platform 22 (revision: 2)".
Downloading http://dl.google.com/android/repository/android-22_r02.zip
"Install Android SDK Platform 22 (revision: 2)" ready.
Preparing "Install Android SDK Platform 10 (revision: 2)".
Downloading http://dl.google.com/android/repository/android-2.3.3_r02.zip
Downloading (59%): 48.1 / 81.5 MB ...
http://dl.google.com/android/repository/android-2.3.3_r02.zip
Please wait until the installation finishes
Background
Previous
Next
Cancel
Finish

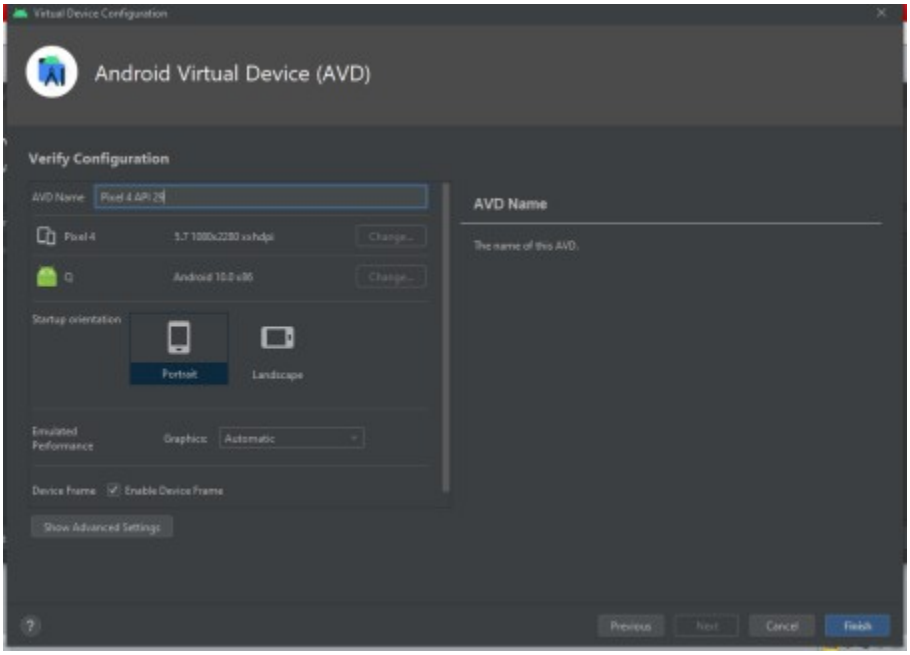

Command Prompt
Accept? (y/N): y
All SDK package licenses accepted
C:\Users\jalpa>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, 2.8.1, on Microsoft Windows [Version 10.0.19042.1415], locale en-US)
[√] Android toolchain - develop for Android devices (Android SDK version 32.0.0)
[√] Chrome - develop for the web
[√] Android Studio (version 2020.3)
[√] VS Code (version 1.55.2)
[√] Connected device (2 available)
• No issues found!
C:\Users\jalpa>flutter doctor
Type here to search


Android Virtual Device Manager
Your Virtual Devices
Android Studio
Virtual devices allow you to test your application without having to own the physical devices.
Create Virtual Device...
To prioritize which devices to test your application on, visit the Android Dashboards, where you can get up-to-date information on which devices are active in the Android and Google Play ecosystem.

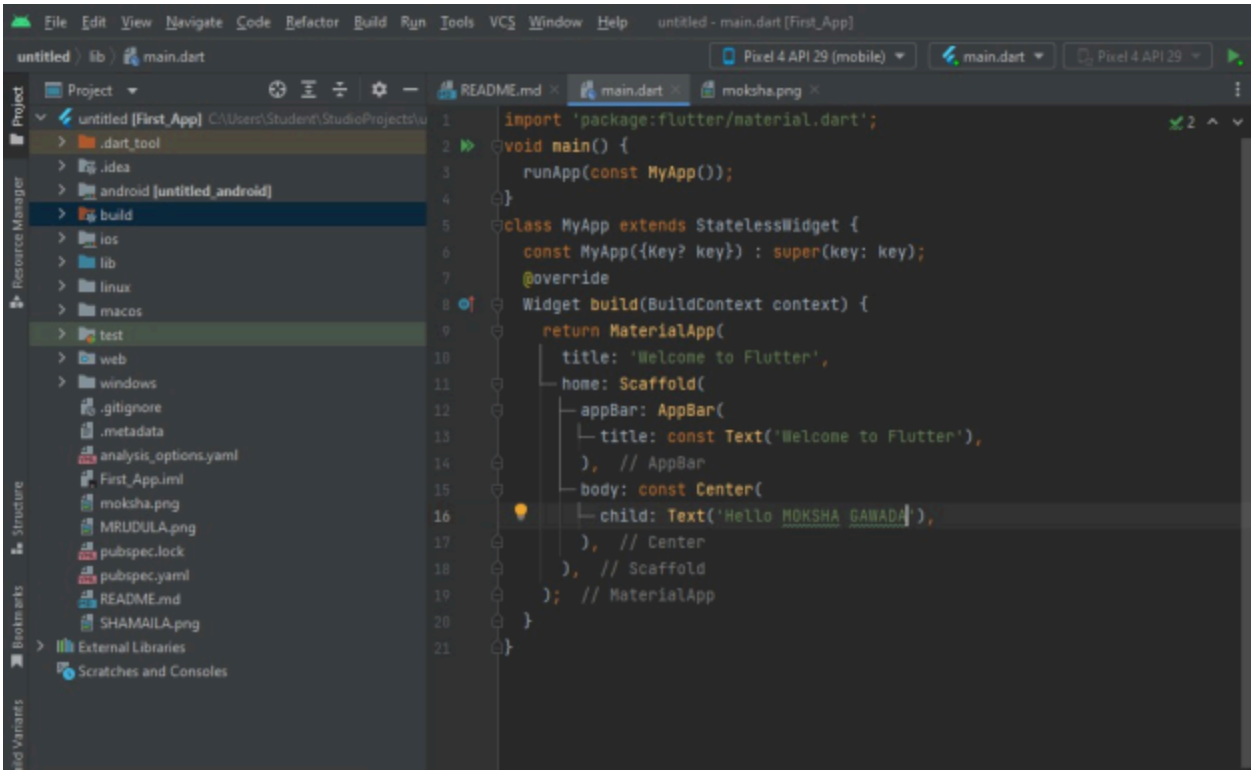

Select Hardware
Choose a device definition
Pixel 4

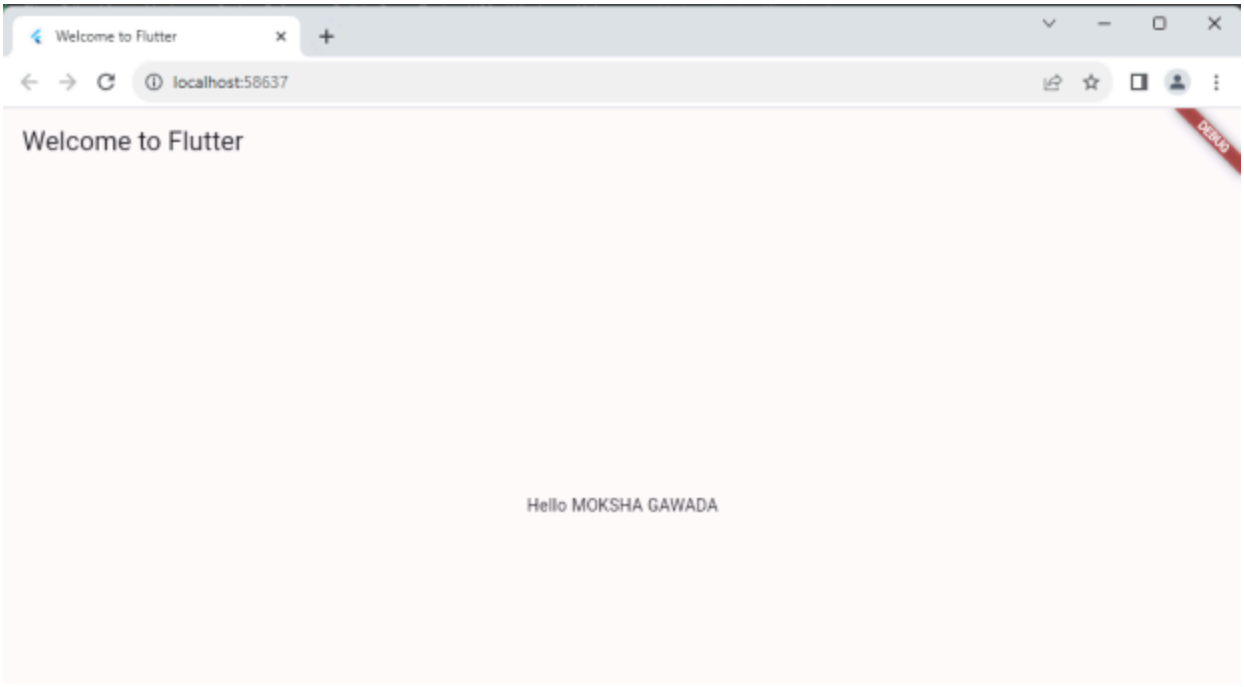
Project Title:Flutter:SkypeClone / PWA:Fuzzy App **Roll No. 18**

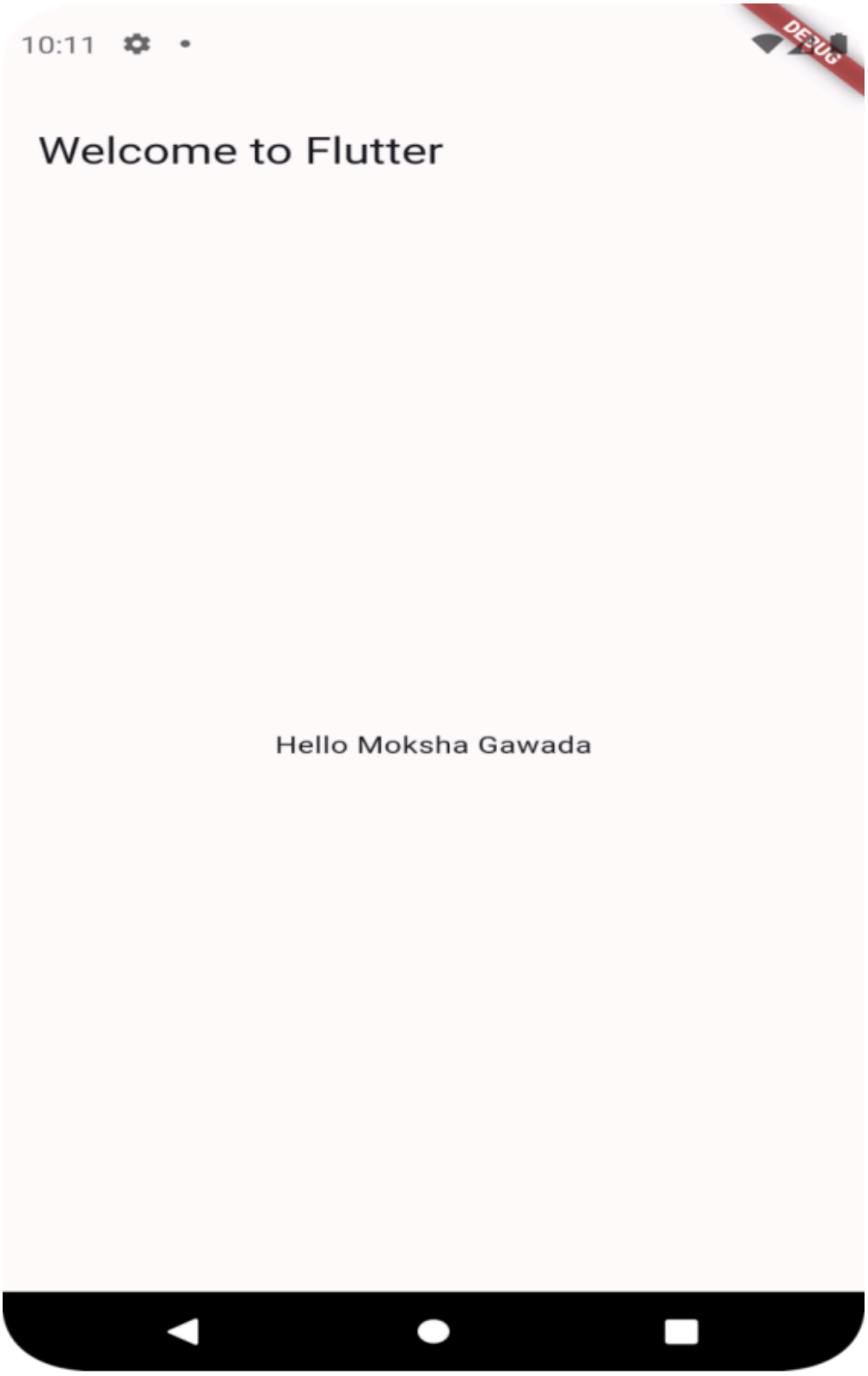




Project Title:Flutter:SkypeClone / PWA:Fuzzy App **Roll No. 18**




Welcome to Flutter
localhost:58637
Welcome to Flutter
DEBUG
Hello MOKSHA GAWADA



CONCLUSION:Basic of flutter was studied.

MAD & PWA Lab
Journal

Experiment No.	02
Experiment Title.	To design Flutter UI by including common widgets.
Roll No.	18
Name	Moksha Gawada
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	10

MAD PWA EXPERIMENT-02

AIM:To design flutter UI by adding common widgets CODE:

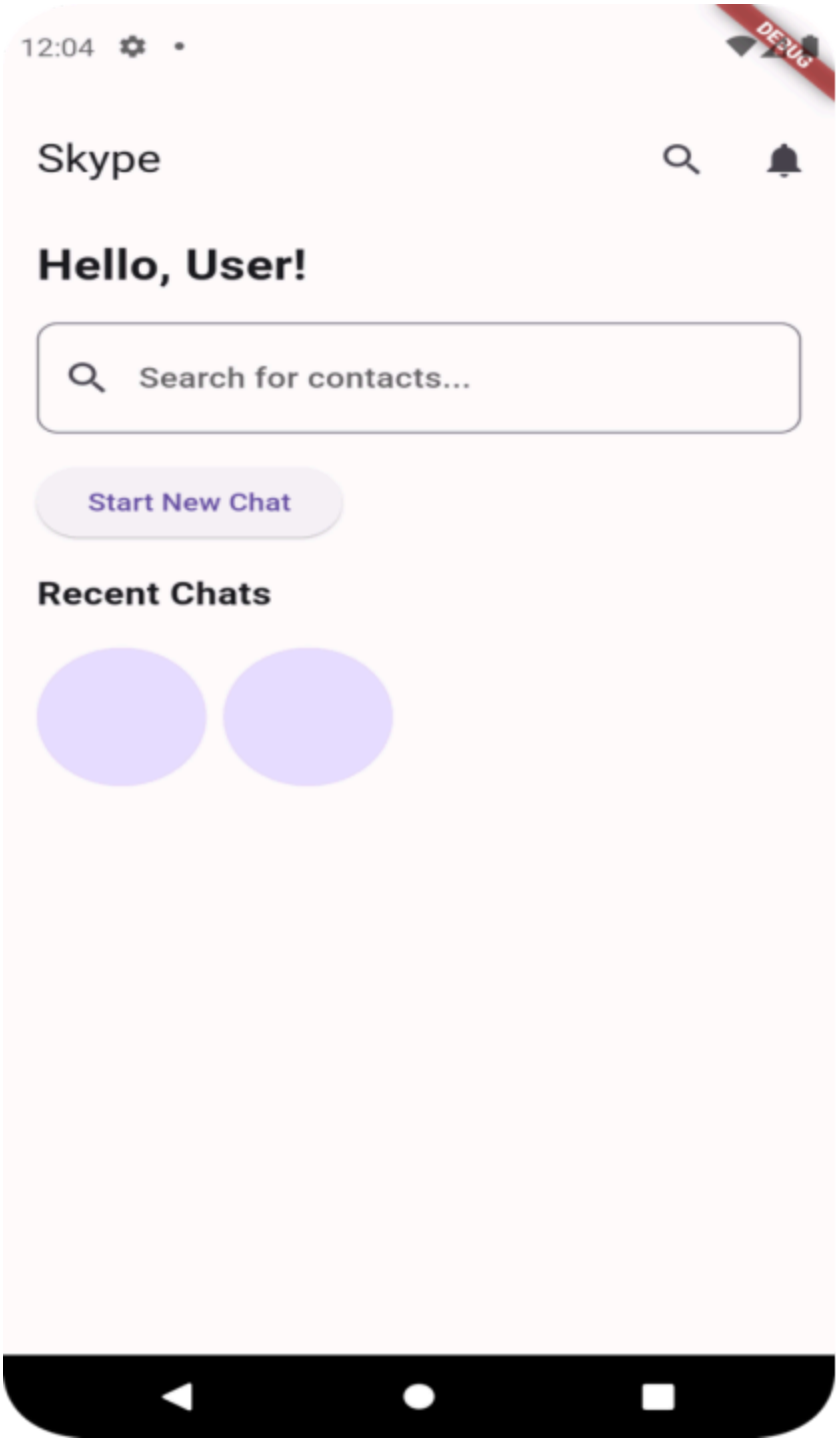
```
import 'package:flutter/material.dart'; void main() {
runApp(MySkypeApp());

}

class MySkypeApp extends StatelessWidget { @override

  Widget build(BuildContext context) { return MaterialApp(

    home: MySkypeHomePage(),

  );

MOKSHA GAWADA

            D15A-18


}

}

class MySkypeHomePage extends StatelessWidget {
 @override

  Widget build(BuildContext context) {
    return Scaffold(

      appBar: AppBar(
        title:
        Text('Skype'),
        actions: [
        IconButton(

icon: Icon(Icons.search),
onPressed: () {

          // Implement search functionality

},

        ),

        IconButton(
```

```
icon: Icon(Icons.notifications), onPressed: ()
{

        // Implement notifications functionality

},

      ),

    ],

  ),

  body: Padding(

    padding: const EdgeInsets.all(16.0),
    child: Column(

      crossAxisAlignment: CrossAxisAlignment.start,
      children: <Widget>[

Text(


        'Hello, User!',

        style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),

),

SizedBox(height: 16),
TextField(

          decoration: InputDecoration(
            hintText: 'Search for contacts...',
            prefixIcon: Icon(Icons.search),
            border: OutlineInputBorder(

            borderRadius: BorderRadius.circular(10.0),

          ),

        ),

),
```

```
SizedBox(height: 16),
ElevatedButton( onPressed: ()
{

            // Implement button functionality

          },

          child: Text('Start New Chat'),

),

SizedBox(height: 16), Text(

          'Recent Chats',

          style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),

),

SizedBox(height: 8),
Container(

          height: 100,
          child:
          ListView(

            scrollDirection:
            Axis.horizontal, children: [

              UserProfileImage(imageUrl: 'assets/user1.jpg'),
              UserProfileImage(imageUrl: 'assets/user2.jpg'),

              // Add more user profiles as needed

            ],

          ),

),

        ],

      ),

    ),

  );
```

```
  }
 }


 class UserProfileImage extends StatelessWidget {
  final String imageUrl;

 const UserProfileImage({required this.imageUrl}); @override

   Widget build(BuildContext context) {
    return Padding(

    padding: const EdgeInsets.only(right: 8.0), child:
    CircleAvatar(

      radius: 40,

      backgroundImage: AssetImage(imageUrl),

    ),

   );

  }

 }
```

OUTPUT:



CONCLUSION:Flutter ui using common widgets was executed.

MAD & PWA Lab
Journal

Experiment No.	03
Experiment Title.	To include icons, images, fonts in Flutter app
Roll No.	18
Name	Moksha Gawada
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	15

MAD PWA EXPERIMENT-03

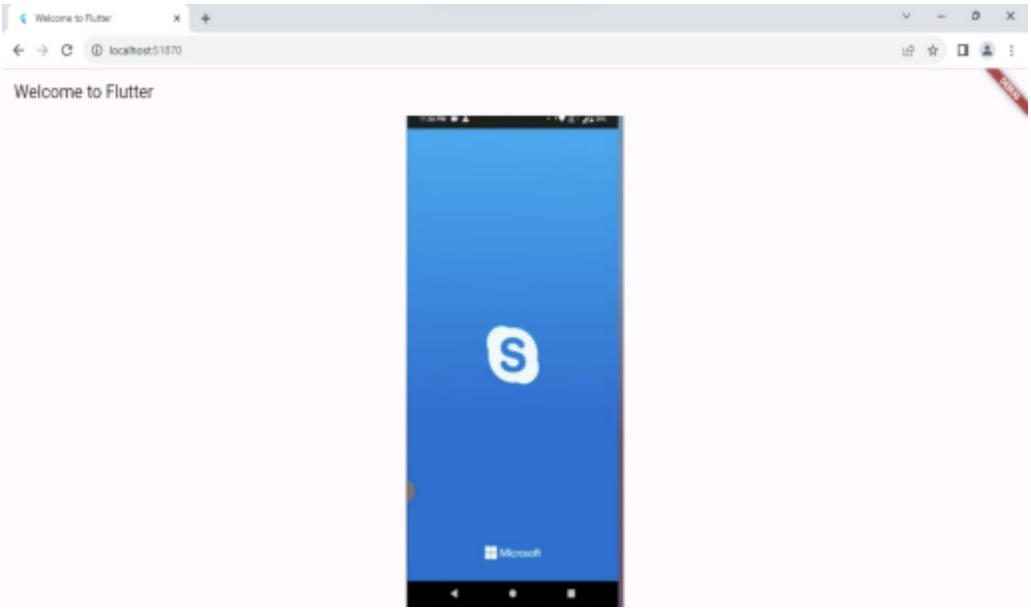
AIM:To include icons image in flutter app

CODE:

```
import 'package:flutter/material.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: 'Welcome to Flutter', home: Scaffold(

        appBar: AppBar(

          title: const Text('Welcome to Flutter'),

        ),

        body: Center(

          child: Image.asset('assets/comp.JPG'),

        ),

      ),

    );

  }

}
```

Project Title:Flutter:SkypeClone / PWA:Fuzzy App **Roll No. 18**

OUTPUT:



CONCLUSION:Including images was executed.

MAD & PWA Lab
Journal

Experiment No.	04
Experiment Title.	To create an interactive Form using form widget
Roll No.	18
Name	Moksha Gawada
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	15

MAD PWA EXPERIMENT-04

AIM:To create an interactive form using form widget

CODE:

Sign_up page.dart

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

class SignupPage extends StatelessWidget { @override

  Widget build(BuildContext context) { return Scaffold(

    appBar: AppBar( title: Text('Signup'),

   ),

   body: SignupForm(),

  );

}

}


class SignupForm extends StatefulWidget
 { @override

_SignupFormState createState() => _SignupFormState();

}


class _SignupFormState extends
 State<SignupForm> { final _formKey =
 GlobalKey<FormState>();

String _email = '';
String _password
= ''; String
_username = '';
```

```
void _submitForm() {
  if (_formKey.currentState!.validate()) {
    // Form is valid, submit data to authentication service
    // Example: Firebase Authentication
    // authService.signup(email, password, username);
    // After signup success, navigate to the home
    page Navigator.pushReplacementNamed(context,
    '/home');
  }
}

@override
  Widget build(BuildContext context)
   { return Padding(
    padding: EdgeInsets.all(16.0),

      child:
       Form(
       key:
       _formKey,
       child:
       Column(
       crossAxisAlignment:
       CrossAxisAlignment.stretch, children:
       <Widget>[
         TextFormField(
           decoration: InputDecoration(labelText:
           'Email'), validator: (value) {
             if (value!.isEmpty) {
```

```
        return 'Please enter your email';
      }
      return null;
    },
    onChanged: (value) => _email = value,
  ),
  TextFormField(
    decoration: InputDecoration(labelText:
    'Password'), obscureText: true,
    validator: (value) {
      if (value!.isEmpty) {
        return 'Please enter your password';
      }
      return null;
    },
    onChanged: (value) => _password = value,
  ),
  TextFormField(
    decoration: InputDecoration(labelText: 'Username'),
    validator: (value) {
      if (value!.isEmpty) {
        return 'Please enter your username';
      }
      return null;
    },
```

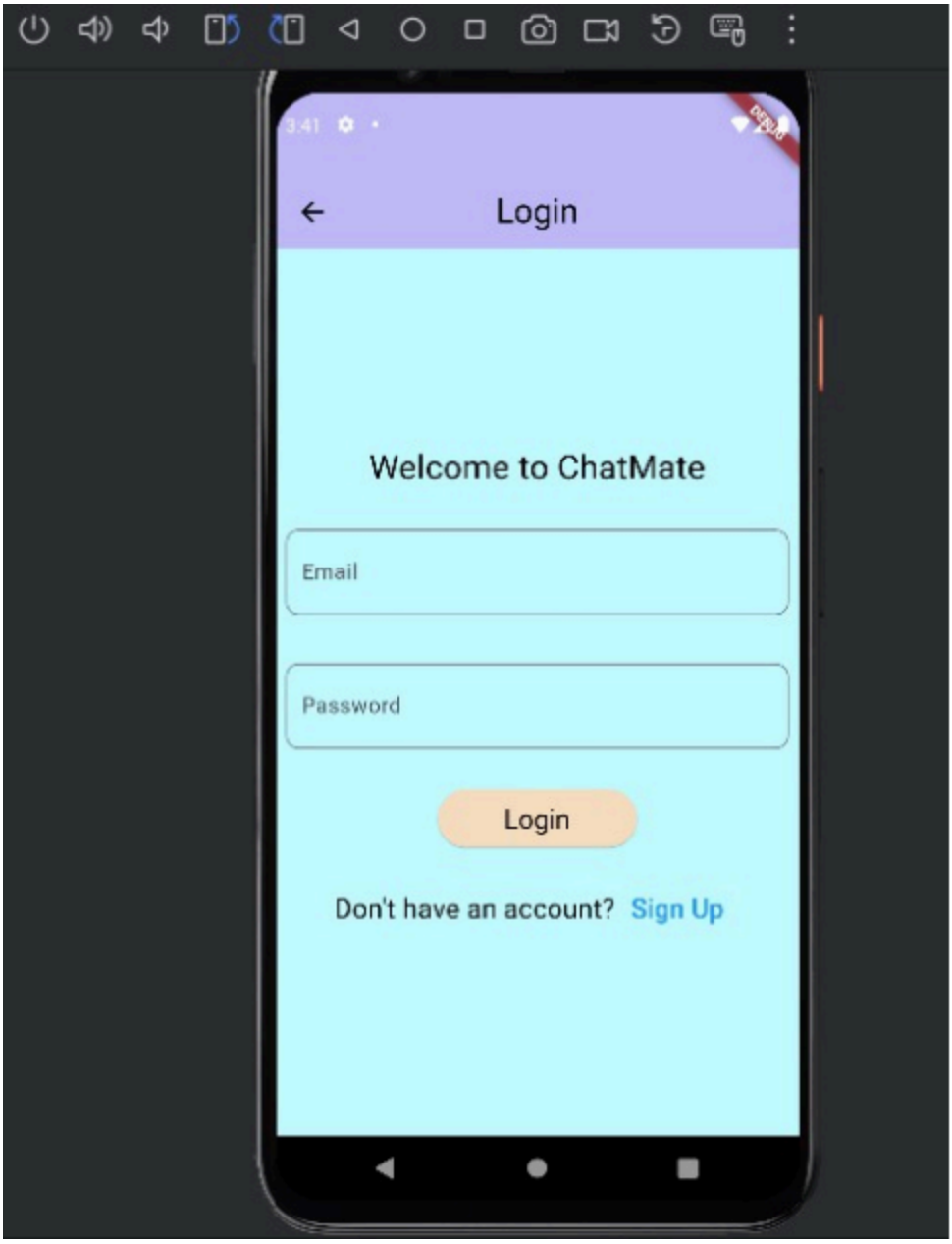
```
            onChanged: (value) => _username = value,

          ),

          SizedBox(height:
          16.0), ElevatedButton(
          onPressed:
          _submitForm, child:
          Text('Signup'),

          ),

        ],

      ),

    ),


  );

 }

}
```

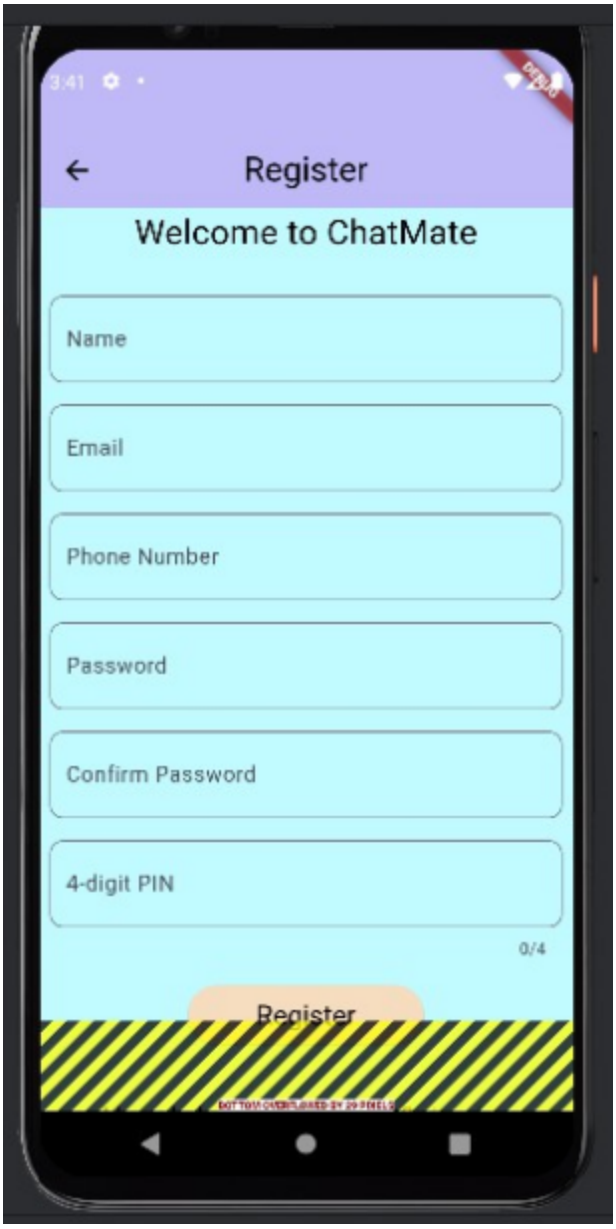
Main.dart

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


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

}


 class MyApp extends StatelessWidget {
  @override
```

```
Widget build(BuildContext context)
 { return MaterialApp(

 title: 'Skype',

  theme: ThemeData(
    primarySwatch:
    Colors.blue,

  visualDensity: VisualDensity.adaptivePlatformDensity,

 ),

 home: LoginPage(),

 );

}

}
```

OUTPUT:


Login
Welcome to ChatMate
Email
Password
Login
Don't have an account? Sign Up



CONCLUSION:Interactive forms using form widget was done.

MAD & PWA Lab
Journal

Experiment No.	05
Experiment Title.	To apply navigation, routing and gestures in Flutter App
Roll No.	18
Name	Moksha Gawada
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	15

MAD PWA LAB EXP-05

AIM:To apply navigation, routing and gestures in Flutter App CODE:

```
import 'package:chatmate/Screens/login.dart'; import
'package:flutter/material.dart';

import 'package:firebase_core/firebase_core.dart'; import
'firebase_options.dart';

 void main()async { WidgetsFlutterBinding.ensureInitialized(); await
  Firebase.initializeApp(

options: DefaultFirebaseOptions.currentPlatform,

 );

 runApp(const MyApp());

}


 class MyApp extends StatelessWidget { const MyApp({super.key});


 @override

Widget build(BuildContext context) { return const MaterialApp(

    title: 'Flutter Demo', home: Login(),

  );

MOKSHA GAWADA

            D15A-18


 }

}

// File generated by FlutterFire CLI.
```

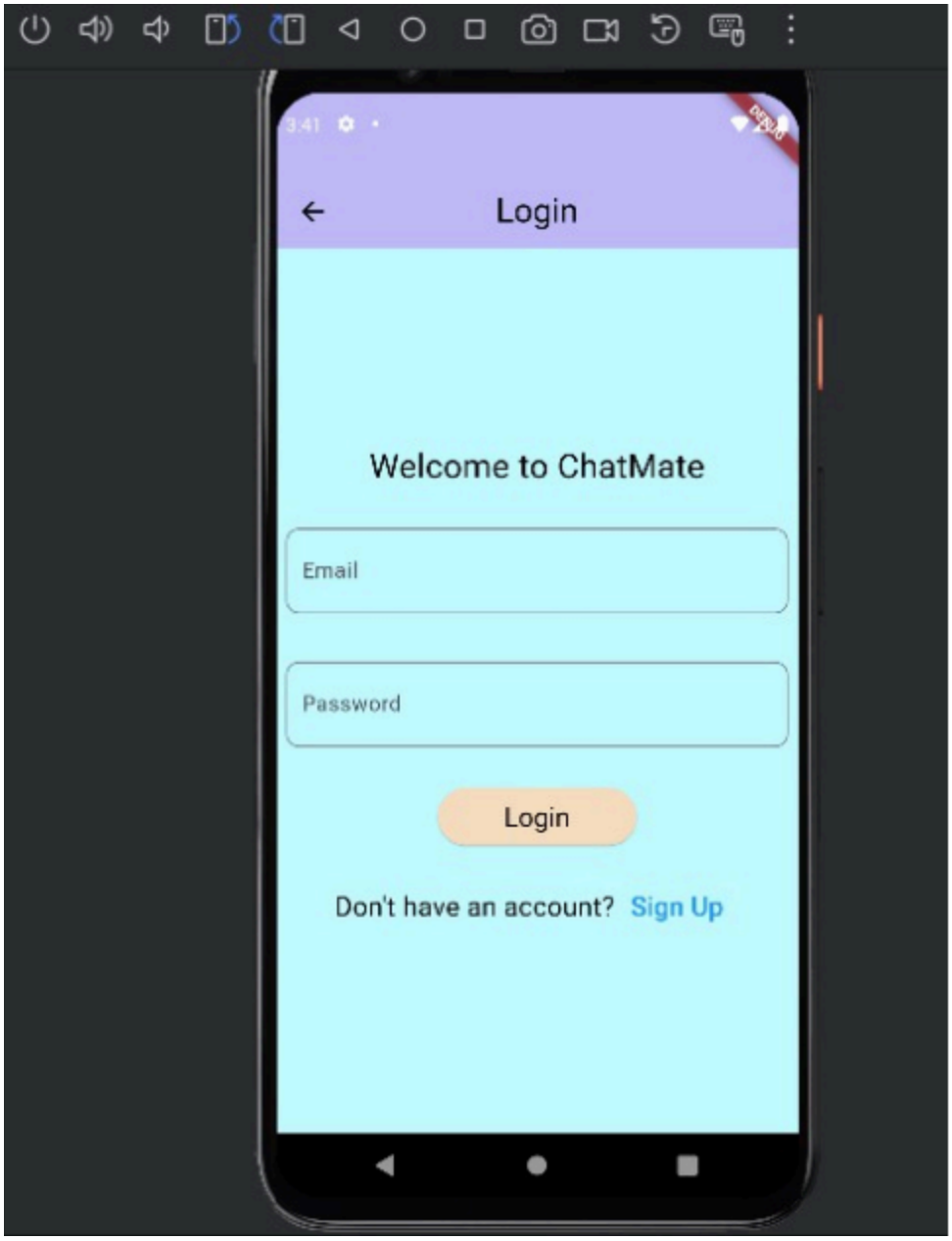
```
// ignore_for_file: lines_longer_than_80_chars,
avoid_classes_with_only_static_members import
'package:firebase_core/firebase_core.dart' show FirebaseOptions;

import 'package:flutter/foundation.dart'

show defaultTargetPlatform, kIsWeb, TargetPlatform;


/// Default [FirebaseOptions] for use with your Firebase apps.

///

/// Example:

/// ```dart

/// import 'firebase_options.dart';

/// // ...

/// await Firebase.initializeApp(

///   options: DefaultFirebaseOptions.currentPlatform,

/// );


/// ```

class DefaultFirebaseOptions {

static FirebaseOptions get
  currentPlatform { if (kIsWeb) {

    return web;

  }

    switch
      (defaultTargetPlatform) {
      case
      TargetPlatform.android:
```

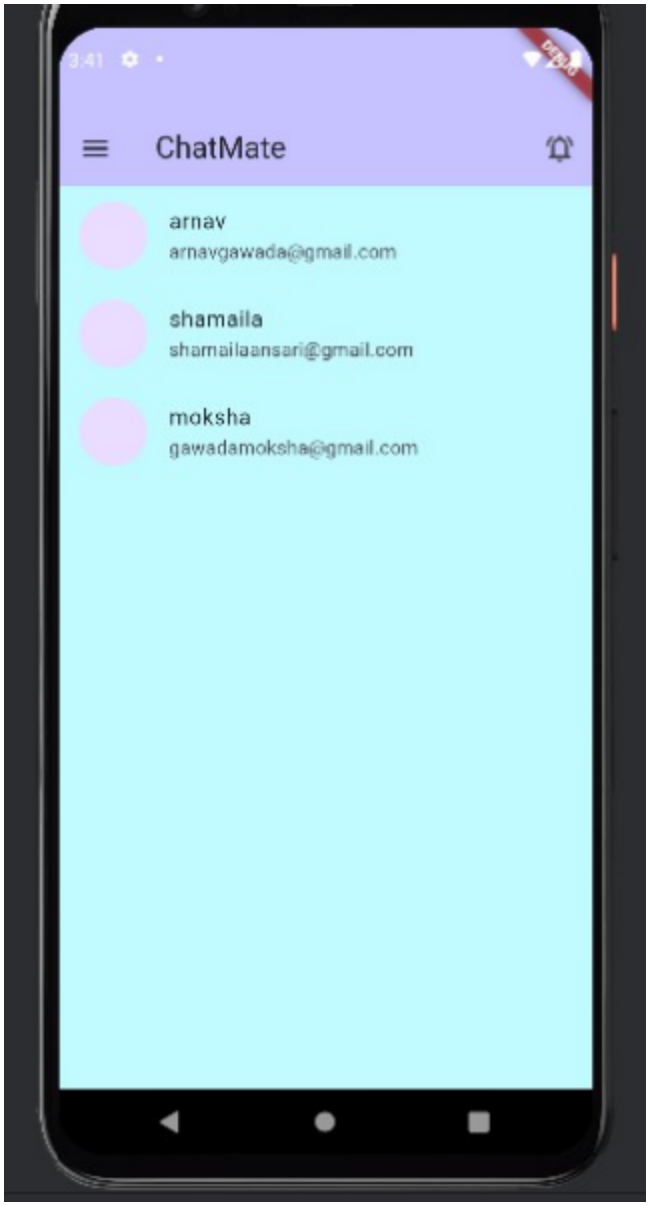
```
      return android;
    case TargetPlatform.iOS:
      return ios;
    case TargetPlatform.macOS:
      return macos;
    case TargetPlatform.windows:
      throw UnsupportedError(
        'DefaultFirebaseOptions have not been configured for windows
        - ' 'you can reconfigure this by running the FlutterFire CLI
        again.',
      );
    case TargetPlatform.linux:
      throw UnsupportedError(
        'DefaultFirebaseOptions have not been configured for linux
        - ' 'you can reconfigure this by running the FlutterFire CLI
        again.',
      );
    default:
      throw UnsupportedError(
        'DefaultFirebaseOptions are not supported for this platform.',
      );
  }
}

static const FirebaseOptions web = FirebaseOptions(
  apiKey:
```

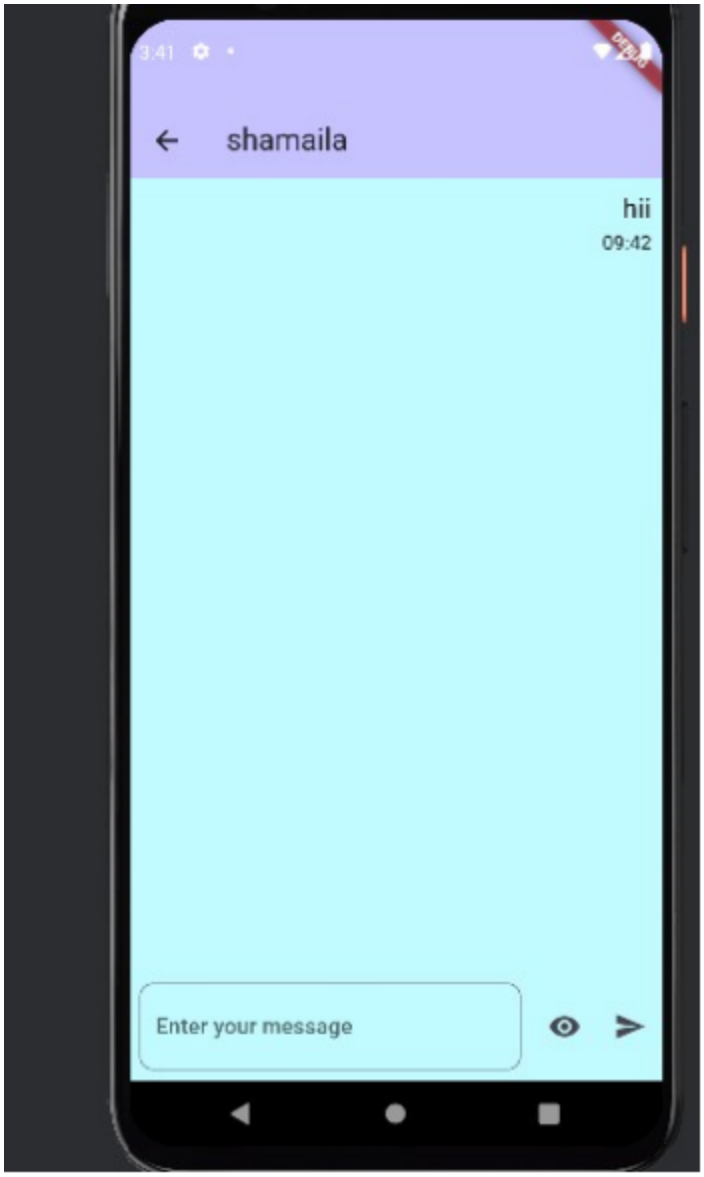
```
    'AIzaSyCA5cUyF7BlJo19k3iD_2GbukMlojc8iVA',
    appId:
    '1:202279690454:web:ec56a8a317e72e444f6977',

messagingSenderId: '202279690454',
projectId: 'chatmate-7761d',

authDomain:
'chatmate-7761d.firebaseapp.com',
storageBucket: 'chatmate-7761d.appspot.com',
measurementId: 'G-V5PGJ3QCN9',

  );


  static const FirebaseOptions android = FirebaseOptions(
    apiKey:
    'AIzaSyDxDXmyQ8SF3V5afBpdXLlng6kQdKNjnvE',
    appId:
    '1:202279690454:android:fd99ddf3bffb3df44f6977',
    messagingSenderId: '202279690454',


projectId: 'chatmate-7761d',

storageBucket: 'chatmate-7761d.appspot.com',

  );


  static const FirebaseOptions ios = FirebaseOptions(
    apiKey:
    'AIzaSyDToNik9eqMAikekr6_oNG17DiCi6FHTog',
    appId:
    '1:202279690454:ios:9507f6b3b233f1434f6977',

messagingSenderId: '202279690454',
projectId: 'chatmate-7761d',
```

```
storageBucket:
'chatmate-7761d.appspot.com', iosBundleId:
'com.example.chatmate',

  );


 static const FirebaseOptions macos =
   FirebaseOptions( apiKey:
   'AIzaSyDToNik9eqMAikekr6_oNG17DiCi6FHTog',
   appId:
   '1:202279690454:ios:55dedd222f6a01614f6977',

messagingSenderId: '202279690454',
projectId: 'chatmate-7761d',

storageBucket: 'chatmate-7761d.appspot.com',
iosBundleId: 'com.example.chatmate.RunnerTests',

  );

 }
```

OUTPUT:


Login
Welcome to ChatMate
Email
Password
Login
Don't have an account? Sign Up


ChatMate
arnav
arnavgawada@gmail.com
shamaila
shamailaansari@gmail.com
moksha
gawadamoksha@gmail.com



CONCLUSION:Navigation,Routing,Gestures was executed and studied.

MAD & PWA Lab
Journal

Experiment No.	06
Experiment Title.	To Connect Flutter UI with fireBase database
Roll No.	18
Name	Moksha Gawada
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO3: Analyze and Build production ready Flutter App by incorporating backend services and deploying on Android / iOS
Grade:	15

MAD PWA LAB EXP-06

AIM:Connect flutter ui with firebase database CODE:

```
import 'package:chatmate/Screens/login.dart'; import
'package:flutter/material.dart';

import 'package:firebase_core/firebase_core.dart'; import 'firebase_options.dart';

  void main()async { WidgetsFlutterBinding.ensureInitialized(); await
   Firebase.initializeApp(

    options: DefaultFirebaseOptions.currentPlatform,

  );

  runApp(const MyApp());

 }


  class MyApp extends StatelessWidget { const MyApp({super.key});


  @override

    Widget build(BuildContext context) { return const MaterialApp(

      title: 'Flutter Demo', home: Login(),

    );
```

MOKSHA GAWADA

D15A-18

```
  }

 }

// File generated by FlutterFire CLI.
```

```
// ignore_for_file: lines_longer_than_80_chars, avoid_classes_with_only_static_members
import 'package:firebase_core/firebase_core.dart' show FirebaseOptions;

import 'package:flutter/foundation.dart'

    show defaultTargetPlatform, kIsWeb, TargetPlatform;


/// Default [FirebaseOptions] for use with your Firebase apps.

  ///

/// Example:

/// ```dart

/// import 'firebase_options.dart';

  /// // ...

/// await Firebase.initializeApp(

///  options: DefaultFirebaseOptions.currentPlatform,

  /// );


  /// ```

class DefaultFirebaseOptions {

    static FirebaseOptions get
      currentPlatform { if (kIsWeb) {

      return web;

    }

      switch
        (defaultTargetPlatform) {
        case
        TargetPlatform.android:

        return android;
```

```
    case TargetPlatform.iOS:
      return ios;
    case TargetPlatform.macOS:
      return macos;
    case TargetPlatform.windows:
      throw UnsupportedError(
        'DefaultFirebaseOptions have not been configured for
        windows - ' 'you can reconfigure this by running the
        FlutterFire CLI again.',
      );
    case TargetPlatform.linux:
      throw UnsupportedError(
        'DefaultFirebaseOptions have not been configured for
        linux - ' 'you can reconfigure this by running the
        FlutterFire CLI again.',
      );
    default:
      throw UnsupportedError(
        'DefaultFirebaseOptions are not supported for this platform.',
      );
  }
}


  static const FirebaseOptions web =
    FirebaseOptions( apiKey:
    'AIzaSyCA5cUyF7BlJo19k3iD_2GbukMlojc8iVA',
```

```
    appId:
    '1:202279690454:web:ec56a8a317e72e444f6977',

  messagingSenderId:
  '202279690454', projectId:
  'chatmate-7761d',

  authDomain:
  'chatmate-7761d.firebaseapp.com',
  storageBucket:
  'chatmate-7761d.appspot.com',
  measurementId: 'G-V5PGJ3QCN9',

);



  static const FirebaseOptions android =
    FirebaseOptions( apiKey:
    'AIzaSyDxDXmyQ8SF3V5afBpdXLlng6kQdKNjnvE
    ', appId:
    '1:202279690454:android:fd99ddf3bffb3df44f6977',
    messagingSenderId: '202279690454',



  projectId: 'chatmate-7761d',

  storageBucket: 'chatmate-7761d.appspot.com',

);



  static const FirebaseOptions ios = FirebaseOptions(
    apiKey:
    'AIzaSyDToNik9eqMAikekr6_oNG17DiCi6FHTog'
    , appId:
    '1:202279690454:ios:9507f6b3b233f1434f6977',

  messagingSenderId:
  '202279690454', projectId:
  'chatmate-7761d',
```

```
    storageBucket:
    'chatmate-7761d.appspot.com',
    iosBundleId: 'com.example.chatmate',

  );



    static const FirebaseOptions macos =
      FirebaseOptions( apiKey:
      'AIzaSyDToNik9eqMAikekr6_oNG17DiCi6FHTog'
      , appId:
      '1:202279690454:ios:55dedd222f6a01614f6977',

    messagingSenderId:
    '202279690454', projectId:
    'chatmate-7761d',

    storageBucket: 'chatmate-7761d.appspot.com',
    iosBundleId: 'com.example.chatmate.RunnerTests',

  );

 }

import
'package:chatmate/Screens/chatroom.dart';
import 'package:chatmate/Screens/login.dart';
import 'package:chatmate/Screens/profile.dart';
import
'package:cloud_firestore/cloud_firestore.dart';
import
'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';



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



  @override
```

```
State<HomePage> createState() => _HomePageState();

}


class _HomePageState extends State<HomePage> {
 Color newBlue = const Color(0xFFC1FAFF);

Color newPurple = const
Color(0xFFC6C1FF); Color newOrange =
const Color(0xFFFFE5C1); final
FirebaseAuth _auth = FirebaseAuth.instance;

final FirebaseFirestore _firestore = FirebaseFirestore.instance;
@override

Widget build(BuildContext context) {


  return Scaffold(
    backgroundColor:
    newBlue, appBar:
    AppBar(
    backgroundColor:
    newPurple, title: const
    Text('ChatMate'), actions:
    [

        IconButton(
          onPressed: ()
          {},

        icon: const Icon(Icons.notifications_active_outlined))

    ],

  ),

    drawer:
      Drawer(
      child:
      ListView(
```

```
padding:
EdgeInsets.zero,
children: <Widget>[
DrawerHeader(

    decoration:
      BoxDecoration(
      color: newBlue,

  ),

    child: const
      Column(
      children: [
      CircleAvatar(

        radius: 50,

        // backgroundImage: AssetImage('assets/images/profile.png'),

      ),

        Text(
          'Username
          ', style:
          TextStyle(

          color:
          Colors.black,
          fontSize: 20,

        ),

      ),

    ],

  ),

),

ListTile(
```

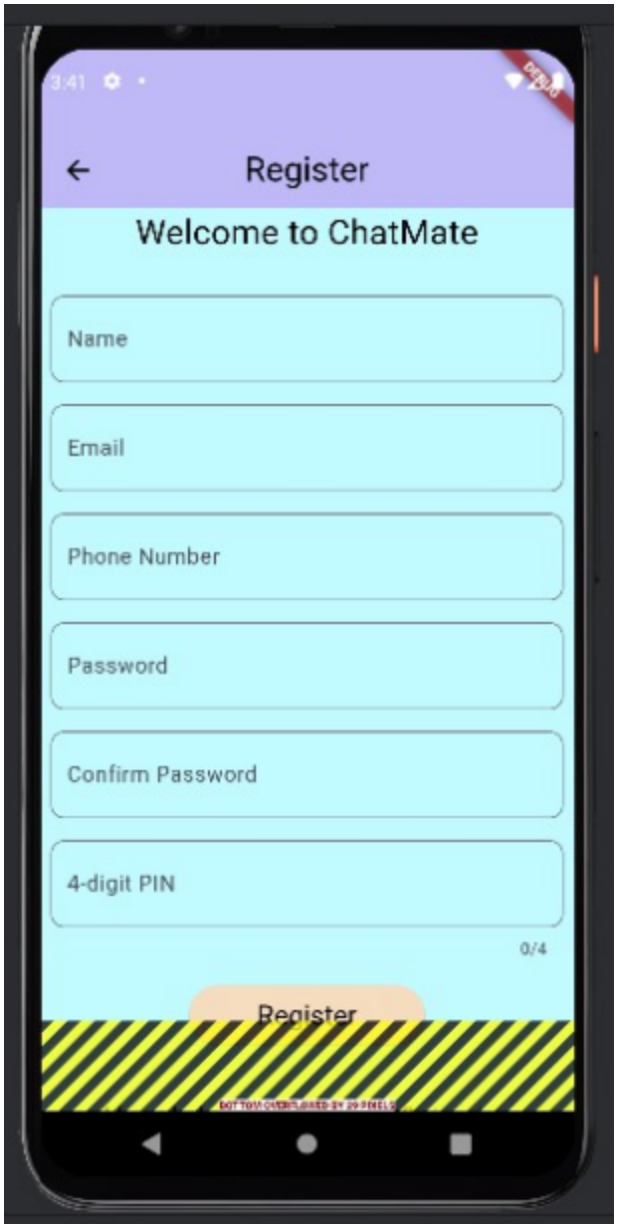
```
leading: const
Icon(Icons.person), title: const
Text('Profile'),

onTap: () {

  Navigator.push(context, MaterialPageRoute(builder:
    (context) { return const Profile();

  }));

},

),


ListTile(

leading: const
Icon(Icons.settings), title: const
Text('Setting'),

  onTap: () {
    Navigator.pop(conte
    xt);

},

),

ListTile(

leading: const
Icon(Icons.logout), title: const
Text('Logout'),

onTap: () {

  _auth.signOut();
  Navigator.pop(context);

    Navigator.push(context, MaterialPageRoute(builder:
      (context) { return const Login();
```

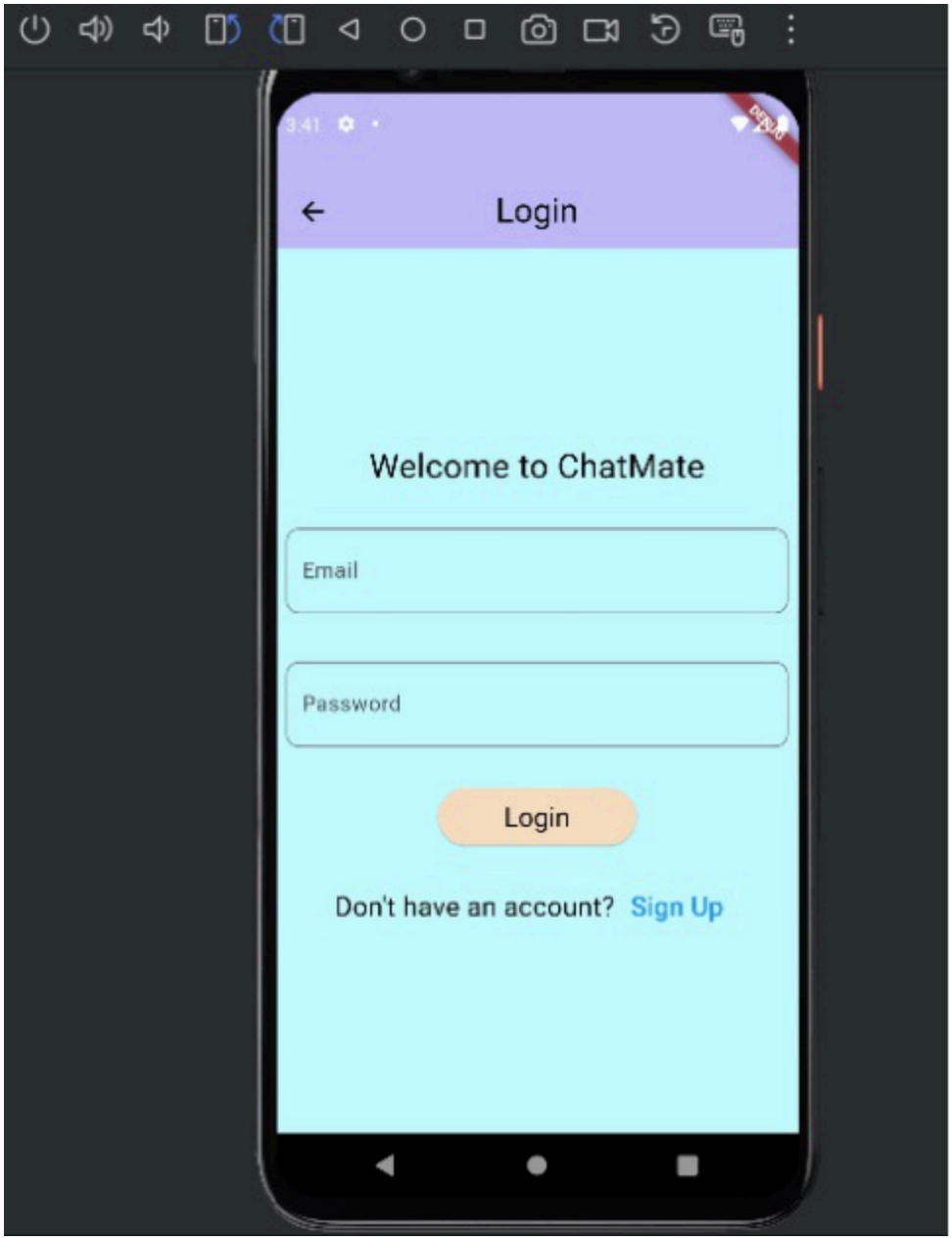
```
        }));
      },
    ),
  ],
 ),
),
body: Center(
  child:
 StreamBuilder(
 stream: _firestore
    .collection('users')
    // .where('email', isEqualTo: _auth.currentUser!.email)
    .snapshots(),
 builder: (context, snapshot) {
    if (snapshot.connectionState ==
      ConnectionState.waiting) { return const Center(
      child: CircularProgressIndicator(),
    );
  }
  if (snapshot.hasData) {
    var doc =
    snapshot.data!.docs; return
    ListView.builder(
    itemCount: doc.length,
    itemBuilder: (context,
    index) {
```

```
if (doc[index]['email'] ==
  _auth.currentUser!.email) { return
  Container();

}

return
  GestureDetector(
  onTap: (){

  Navigator.push(context, MaterialPageRoute(builder: (context){


    return ChatRoom(

      otherUserEmail:
      doc[index]['email'],
      otherName:
      doc[index]['name']

    );

  }));

},

child: ListTile(

    leading: const
      CircleAvatar( radius: 25,

  ),

  title:
  Text(doc[index]['name']),
  subtitle:
  Text(doc[index]['email']),

),

);

},
```

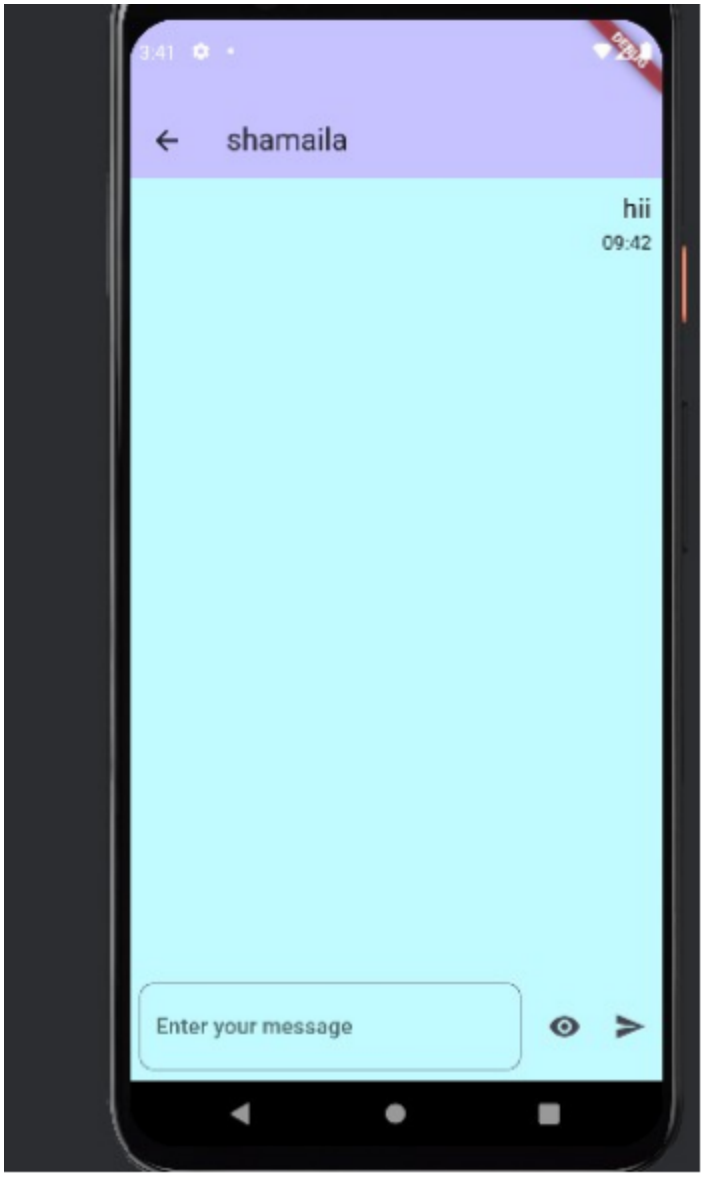
```
          );
        }
        return Container();
      },
    ),
  );
 }
}
```

OUTPUT:

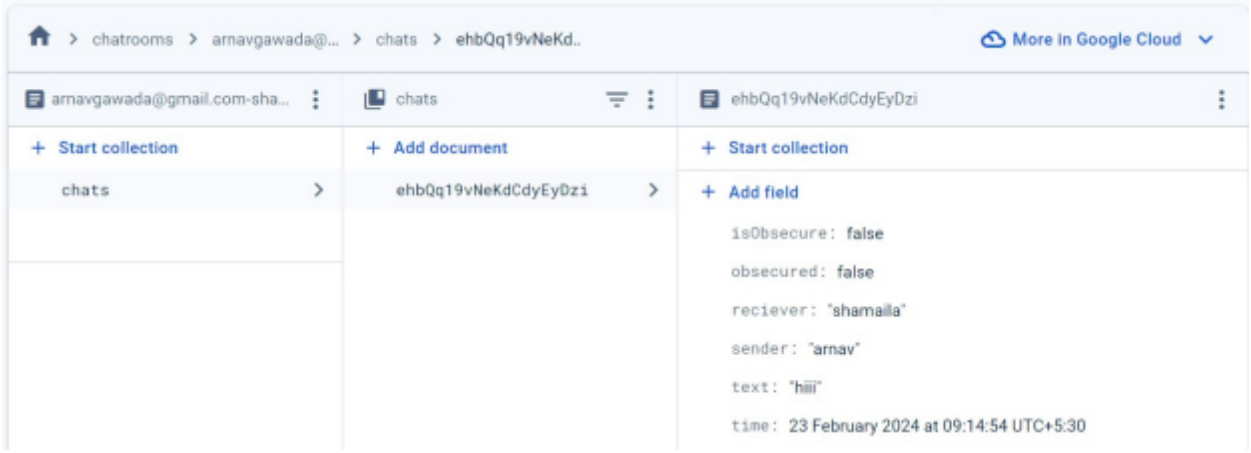

Register
Welcome to ChatMate
Name
Email
Phone Number
Password
Confirm Password
4-digit PIN
0/4
Register

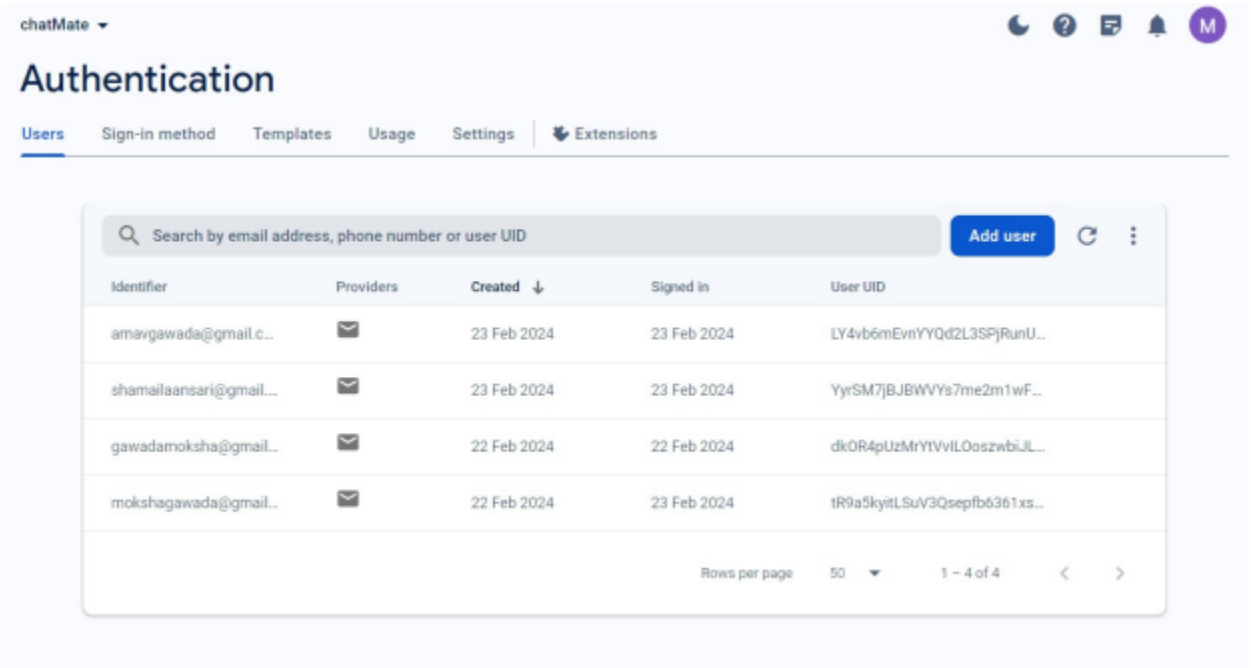

Login
Welcome to ChatMate
Email
Password
Login
Don't have an account? Sign Up

Project Title:Flutter:SkypeClone / PWA:Fuzzy App **Roll No. 18**



FIREBASE:





CONCLUSION: Connectivity With firebase was done and studied.

MAD & PWA Lab
Journal

Experiment No.	07
Experiment Title.	To write meta data of your Ecommerce PWA in a Web app manifest file to enable “add to homescreen feature”.
Roll No.	18
Name	Moksha Gawada
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO4: Understand various PWA frameworks and their requirements
Grade:	15

EXPERIMENT NO: 07

Aim: To write meta data of your Ecommerce PWA in a Web app manifest file to enable add to homescreen feature

Theory :

Regular Web App

A regular web app is a website that is designed to be accessible on all mobile devices such that the content gets fit as per the device screen. It is designed using a web technology stack (HTML, CSS, JavaScript, Ruby, etc.) and operates via a browser. They offer various native-device features and functionalities. However, it entirely depends on the browser the user is using. In other words, it might be possible that you can access a native-device feature on Chrome but not on Safari or Mozilla Firefox because the browsers are incompatible with that feature.

Progressive Web App

Progressive Web App (PWA) is a regular web app, but some extras enable it to deliver an excellent user experience. It is a perfect blend of desktop and mobile application experience to give both platforms to the end-users.

Difference between PWAs vs. Regular Web Apps:

A Progressive Web is different and better than a Regular Web app with features like:

1. Native Experience

Though a PWA runs on web technologies (HTML, CSS, JavaScript) like a Regular web app, it gives user experience like a native mobile application. It can use most native device features, including push notifications, without relying on the browser or any other entity. It offers a seamless and integrated user experience that it is quite tough for one to differentiate between a PWA and a Native application by considering its look and feel.

2. Ease of Access

Unlike other mobile apps, PWAs do not demand longer download time and make memory space available for installing the applications. The PWAs can be shared and installed by a link, which cuts down the number of steps to install and use. These applications can easily keep an app icon on the user's home screen, making the app easily accessible to the users and helps the brands remain in the users' minds, and improving the chances of interaction.

3. Faster Services

PWAs can cache the data and serve the user with text stylesheets, images, and other web content even before the page loads completely. This lowers the waiting time for the end-users and helps the brands improve the user engagement and retention rate, which eventually adds value to their business.

4. Engaging Approach

As already shared, the PWAs can employ push notifications and other native device features more efficiently. Their interaction does not depend on the browser user uses. This eventually improves the chances of notifying the user regarding your services, offers, and other options related to your brand and keeping them hooked to your brand. In simpler words, PWAs let you maintain the user engagement and retention rate.

5. Updated Real-Time Data Access

Another plus point of PWAs is that these apps get updated on their own. They do not demand the end-users to go to the App Store or other such platforms to download the update and wait until installed.

In this app type, the web app developers can push the live update from the server, which reaches the apps residing on the user's devices automatically. Therefore, it is easier for the mobile app developer to provide the best of the updated functionalities and services to the end-users without forcing them to update their app.

6. Discoverable

PWAs reside in web browsers. This implies higher chances of optimizing them as per the Search Engine Optimization (SEO) criteria and improving the Google rankings like that in websites and other web apps.

7. Lower Development Cost

Progressive web apps can be installed on the user device like a native device, but it does not demand submission on an App Store. This makes it far more cost-effective than native mobile applications while offering the same set of functionalities.

Pros and cons of the Progressive Web App The main features are:

Progressive — They work for every user, regardless of the browser chosen because they are built at the base with progressive improvement principles.

Responsive — They adapt to the various screen sizes: desktop, mobile, tablet, or dimensions that can later become available.

App-like — They behave with the user as if they were native apps, in terms of interaction and navigation.

Updated — Information is always up-to-date thanks to the data update process offered by service workers.

Secure — Exposed over HTTPS protocol to prevent the connection from displaying information or altering the contents.

Searchable — They are identified as “applications” and are indexed by search engines.

Reactivable — Make it easy to reactivate the application thanks to capabilities such as web notifications.

Installable — They allow the user to “save” the apps that he considers most useful with the corresponding icon on the screen of his mobile terminal (home screen) without having to face all the steps and problems related to the use of the app store.

Linkable — Easily shared via URL without complex installations.

Offline — Once more it is about putting the user before everything, avoiding the usual error message in case of weak or no connection. The PWA are based on two particularities: first of all the 'skeleton' of the app, which recalls the page structure, even if its contents do not respond and its elements include the header, the page layout, as well as an illustration that signals that the page is loading.

Weaknesses refer to:

IOS support from version 11.3
onwards; Greater use of the device
battery;

Not all devices support the full range of PWA features (same speech for iOS and Android operating systems);

It is not possible to establish a strong re-engagement for iOS users (URL scheme, standard web notifications);

Support for offline execution is however limited;

Lack of presence on the stores (there is no possibility to acquire traffic from that channel); There is no "body" of control (like the stores) and an approval process;

Limited access to some hardware components of the devices;

Little flexibility regarding "special" content for users (eg loyalty programs, loyalty, etc.)

CODE:

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
```

```
        <meta name="description" content="fuzzy" />

        <meta name="keywords" content="fuzzy" />

        <meta name="author" content="fuzzy" />

        <link rel="manifest" href="manifest.json" />

        <link rel="icon" href="assets/images/logo/favicon.png" type="image/x-icon" />

        <title>fuzzy</title>

        <link rel="apple-touch-icon"
href="assets/images/logo/favicon.png" />

        <meta name="theme-color" content="#122636" />

        <meta name="apple-mobile-web-app-capable" content="yes"
/>

        <meta name="apple-mobile-web-app-status-bar-style"
content="black" />

        <meta name="apple-mobile-web-app-title" content="fuzzy"
/>

        <meta name="msapplication-TileImage"
content="assets/images/logo/favicon.png" />


        <meta name="msapplication-TileColor" content="#FFFFFF"
/>

        <meta http-equiv="X-UA-Compatible" content="IE=edge" />


        <!--Google font-->
```

```
    <link rel="preconnect"
href="https://fonts.googleapis.com/" />

    <link rel="preconnect" href="https://fonts.gstatic.com/"
crossorigin />

    <link href="https://fonts.googleapis.com/css2?family=Poppins:wght
@100;200;300;400;500;600;700;800;900&display=swap"

rel="stylesheet" />



    <!-- iconsax css -->

    <link rel="stylesheet" type="text/css"
href="assets/css/vendors/iconsax.css" />



    <!-- bootstrap css -->

    <link rel="stylesheet" id="rtl-link" type="text/css"
href="assets/css/vendors/bootstrap.min.css" />



    <!-- swiper css -->

    <link rel="stylesheet" type="text/css"
href="assets/css/vendors/swiper-bundle.min.css" />



    <!-- Theme css -->

    <link rel="stylesheet" id="change-link" type="text/css" href="assets/css/style.css" />

  </head>



  <body class="auth-body dark">
```

```
<!-- onboarding section start -->

<section class="section-b-space">

    <div class="swiper intro slider-1">


        <div class="swiper-wrapper">

            <div class="swiper-slide">

        <div class="theme-logo pb-3">

            <img class="img-fluid logo-img"
src="assets/images/logo/logo.png" alt="logo" />

        </div>

        <div class="onboarding-design">

            <img class="img-fluid design-img"
src="assets/images/onboarding/design1.png" alt="bg-design"

/>



            <img class="img-fluid slider-img1"
src="assets/images/onboarding/1.png" alt="slider-1" />



            <img class="img-fluid vector1"
src="assets/images/onboarding/vector1.png" alt="v1" />

            <img class="img-fluid vector2"
src="assets/images/onboarding/vector2.png" alt="v2" />

            <img class="img-fluid vector3"
src="assets/images/onboarding/vector3.png" alt="v3" />

        </div>

            <div class="product-details">
```

```
                    <h1>Office Furniture</h1>

                    <span></span>

                    <p>The best products that suits your lifestyle with
visually appealing designs.</p>

                    <div class="redirate-btn">

                        <a href="login.html" class="next-arrow">

                            <i class="iconsax right-arrow" data-
icon="arrow-right"></i>

                        </a>

                    </div>

                </div>

            </div>


            <div class="swiper-slide">

                <div class="theme-logo pb-3">

                    <img class="img-fluid logo-img"
src="assets/images/logo/logo.png" alt="logo" />

                </div>

                <div class="onboarding-design">

                    <img class="img-fluid design-img"
src="assets/images/onboarding/design1.png" alt="bg-design"

/>



                    <img class="img-fluid slider-img2"
src="assets/images/onboarding/2.png" alt="slider-2" />
```

```
                    <img class="img-fluid vector1"
src="assets/images/onboarding/vector1.png" alt="v1" />

                    <img class="img-fluid vector2"
src="assets/images/onboarding/vector2.png" alt="v2" />

                    <img class="img-fluid vector3"
src="assets/images/onboarding/vector3.png" alt="v3" />

                </div>

                <div class="product-details">

                    <h1>Relaxing Furniture</h1>

                    <span></span>

                    <p>The best funiture that fits your House and workspace.</p>

                    <div class="redirate-btn">

                        <a href="login.html" class="next-arrow">

                            <i class="iconsax right-arrow" data-
icon="arrow-right"></i>

                        </a>

                    </div>

                </div>

            </div>


            <div class="swiper-slide">


                <div class="theme-logo pb-3">
```

```
                    <img class="img-fluid logo-img"
src="assets/images/logo/logo.png" alt="logo" />

                </div>

                <div class="onboarding-design">

                    <img class="img-fluid design-img"
src="assets/images/onboarding/design1.png" alt="bg-design"

/>



                    <img class="img-fluid slider-img3"
src="assets/images/onboarding/3.png" alt="slider-3" />



                    <img class="img-fluid vector1"
src="assets/images/onboarding/vector1.png" alt="v1" />

                    <img class="img-fluid vector2"
src="assets/images/onboarding/vector2.png" alt="v2" />

                    <img class="img-fluid vector3"
src="assets/images/onboarding/vector3.png" alt="v3" />

                    <div class="product-details">

                        <h1>Home Decor</h1>

                        <span></span>

                        <p>The best payment method connects your money to
friends, family, brands, and experiences.</p>

                        <div class="redirate-btn">

                            <a href="login.html" class="next-arrow">

                                <i class="iconsax right-arrow" data-
icon="arrow-right"></i>
```

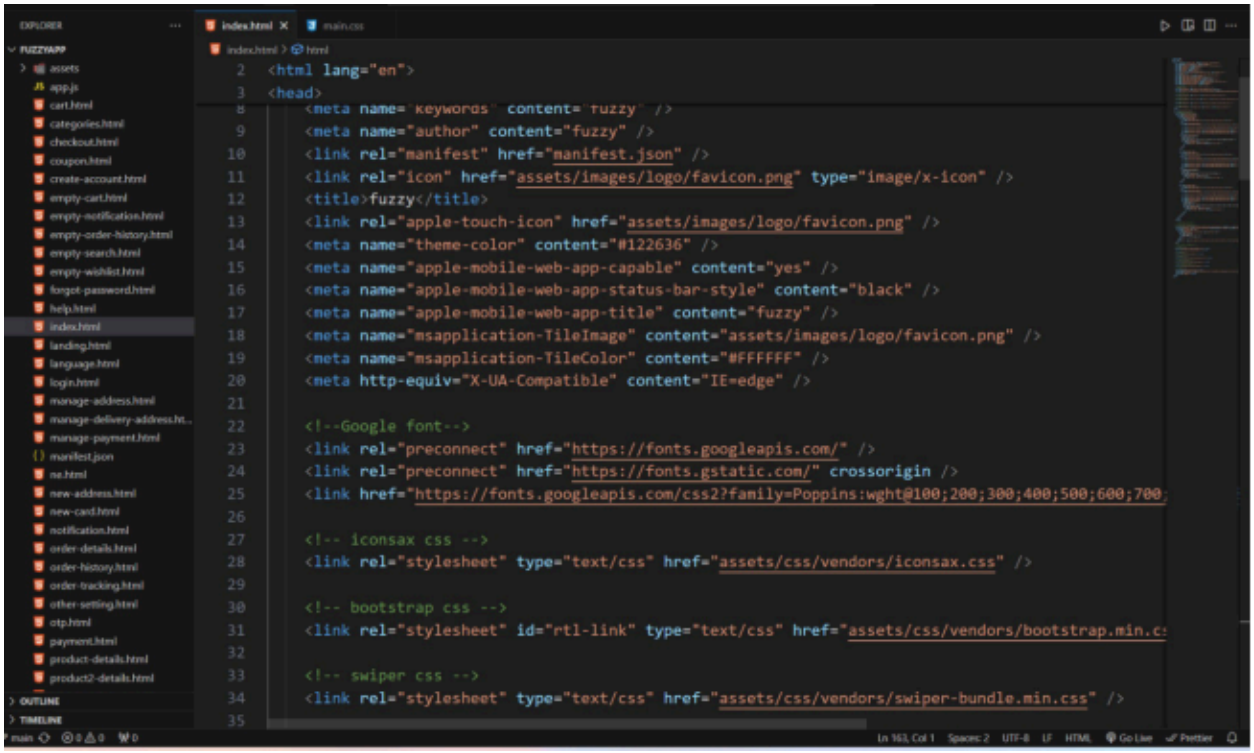
```
                    </a>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </section
            >



    <!-- onboarding section end -->


    <!-- pwa install app popup start -->
    <div class="offcanvas offcanvas-bottom addtohome-popup theme-offcanvas"
tabindex="-1" id="offcanvas">
        <button type="button" class="btn-close text-reset"
data-bs-dismiss="offcanvas" aria-label="Close"></button>
        <div class="offcanvas-body small">
            <div class="app-info">
                <img src="assets/images/logo/48.png" class="img- fluid" alt="" />
                <div class="content">
                    <h4>fuzzy app</h4>
                    <a href="#">www.fuzzy-app.com</a>
```

```
                </div>
            </div>
            <a href="#!" class="btn theme-btn install-app btn- inline home-screen-btn m-0" id="installapp">Add to Home Screen</a>
        </div>
    </div>
    <!-- pwa install app popup start -->

    <!-- swiper js -->
    <script src="assets/js/swiper-bundle.min.js"></script>
    <script src="assets/js/custom-swiper.js"></script>

    <!-- iconsax js -->
    <script src="assets/js/iconsax.js"></script>

    <!-- bootstrap js -->
    <script src="assets/js/bootstrap.bundle.min.js"></script>

    <!-- homescreen popup js -->
    <script src="assets/js/homescreen-popup.js"></script>

    <!-- PWA offcanvas popup js -->
    <script src="assets/js/offcanvas-popup.js"></script>
```

```
    <!-- script js -->

    <script src="assets/js/script.js"></script>

        <script async src="https://pagead2.googlesyndication.com/pagead/js/adsbyg
oogle.js?client=ca-pub-5645451029544050"

        crossorigin="anonymous"></script>

        <script src="/app.js"></script>

  </body>

</html>
```

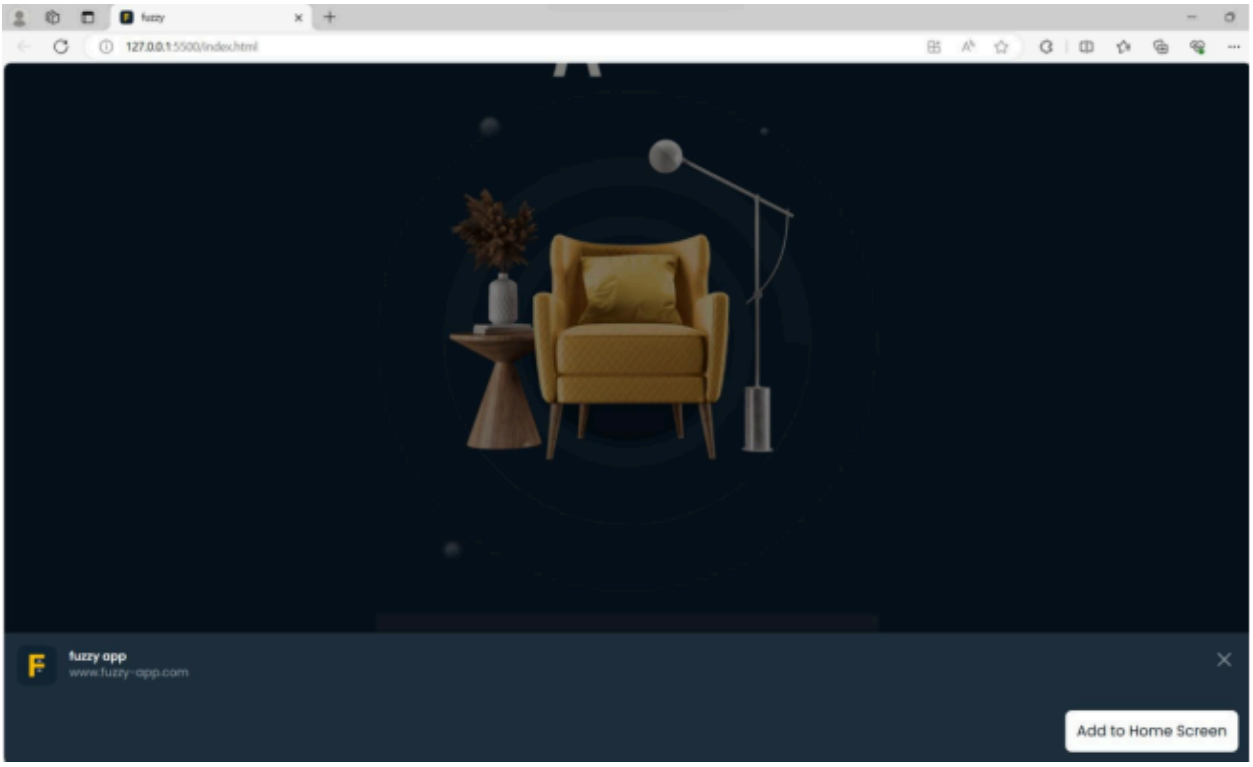
OUTPUT:

Open folder in VS code and click go live at bottom right corner



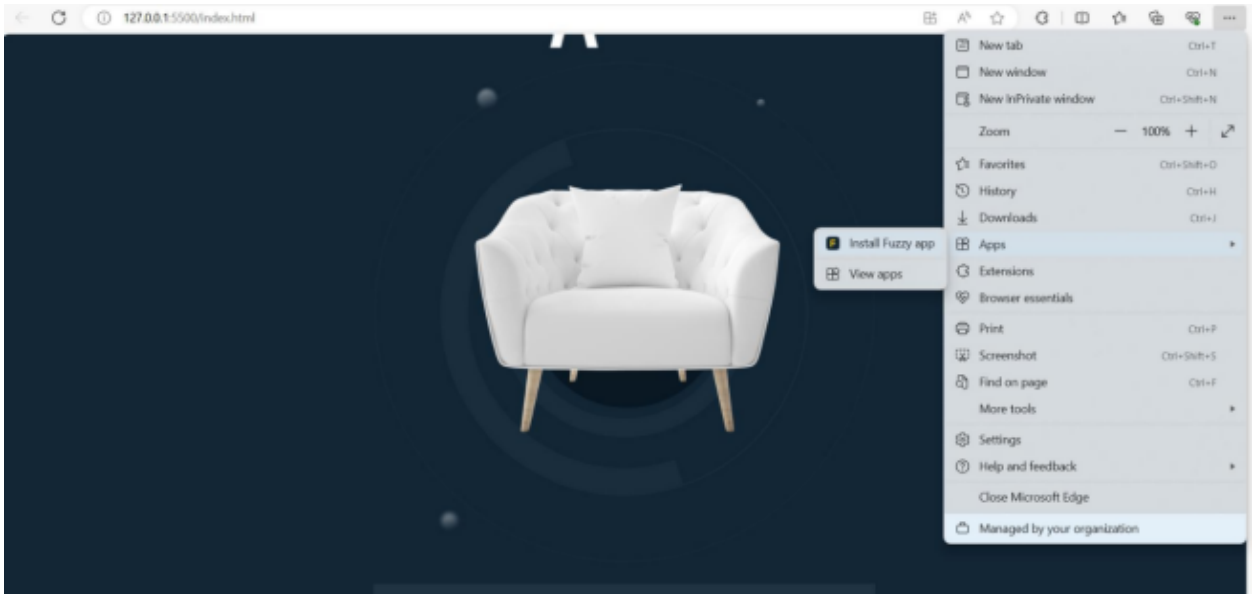
Open your hosted site on Microsoft Edge

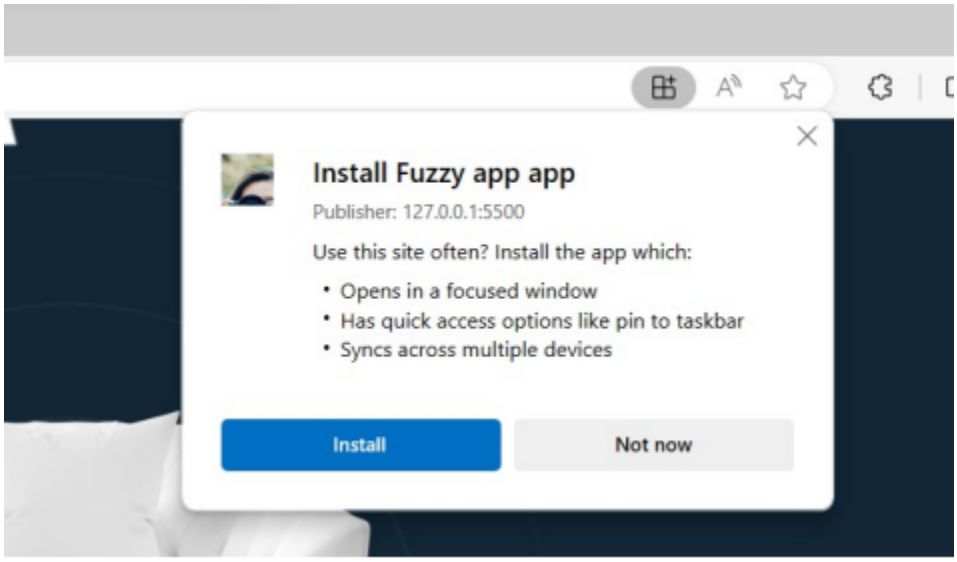
Project Title:Flutter:SkypeClone / PWA:Fuzzy App **Roll No. 18**



Click on 3 dots
Click on Apps
Select install app option

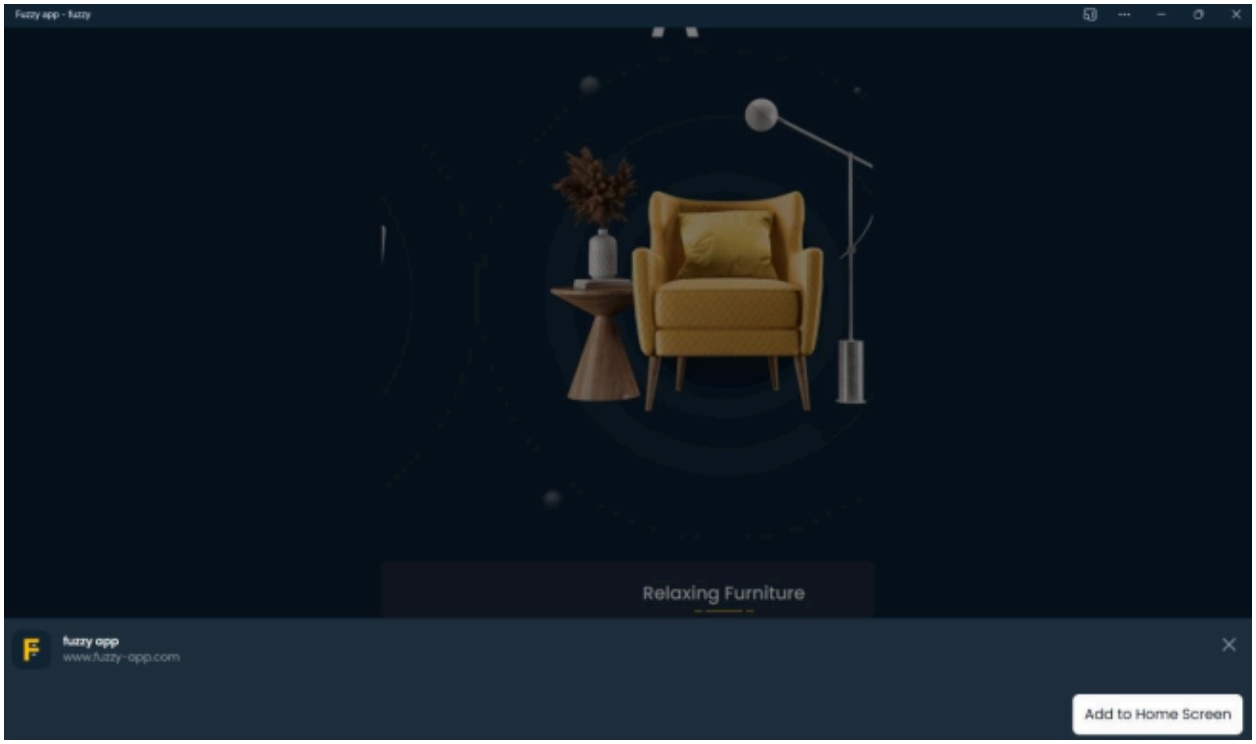
Project Title:Flutter:SkypeClone / PWA:Fuzzy App **Roll No. 18**

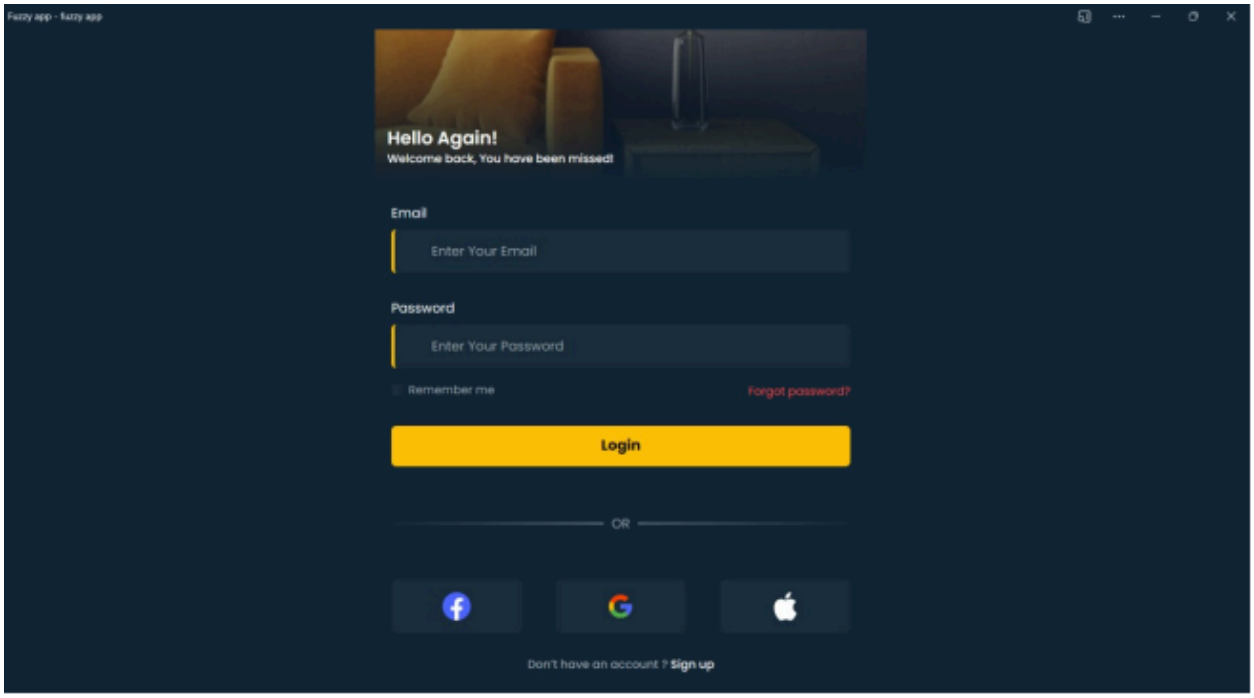




Desktop App created Successfully

Open Desktop App





Conclusion: Here, We have Successfully created a basic progressive web app of our web page and installed it inour desktop successfully.

MAD & PWA Lab
Journal

Experiment No.	08
Experiment Title.	To code and register a service worker, and complete the install and activation process for a new service worker for the E-commerce PWA
Roll No.	18
Name	Moksha Gawada
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	15

Experiment No:08

Aim: To code and register a service worker, and complete the install and activation process for a new service worker for the E-commerce PWA.

Theory:

Service Worker

Service Worker is a script that works on browser background without user interaction independently. Also, It resembles a proxy that works on the user side. With this script, you can track network traffic of the page, manage push notifications and develop “offline first” web applications with Cache API.

Things to note about Service Worker:

● A service worker is a programmable network proxy that lets you control how network requests from your page are handled.
● Service workers only run over HTTPS. Because service workers can intercept network requests and modify responses, "man-in-the-middle" attacks could be very bad.
● The service worker becomes idle when not in use and restarts when it's next needed. You cannot rely on a global state persisting between events. If there is information that you need to persist and reuse across restarts, you can use IndexedDB databases.

What can we do with Service Workers?

● You can dominate **Network Traffic**

You can manage all network traffic of the page and do any manipulations. For example, when the page requests a CSS file, you can send plain text as a response or when the page requests an HTML file, you can send a png file as a response. You can also send a true response too.

● You can **Cache**

You can cache any request/response pair with Service Worker and Cache API and you can access these offline content anytime.

● You can manage **Push Notifications**

You can manage push notifications with Service Worker and show any information message to the user.

● You can **Continue**

Although Internet connection is broken, you can start any process with Background Sync of Service Worker.

What can't we do with Service Workers?

● You can't access the **Window**

You can't access the window, therefore, You can't manipulate DOM elements. But, you can communicate to the window through post Message and manage processes that you want.

● You can't work it on **80 Port**

Service Worker just can work on HTTPS protocol. But you can work on localhost during development.

A service worker goes through three steps in its life cycle:

● Registration
● Installation
● Activation

Registration

To install a service worker, you need to register it in your main JavaScript code. Registration tells the browser where your service worker is located, and to start installing it in the background. Let's look at an example:

main.js

```
if
  ('serviceWorke
```

```
r' in navigator)
{
navigator.servi
ceWorker.regis
ter('/ser
vice-worker.js'
)

.then(function(registration) {

  console.log('Registration successful, scope is:', registration.scope);

})

.catch(function(error) {

  console.log('Service worker registration failed, error:', error);

});

}
```

This code starts by checking for browser support by examining **navigator.serviceWorker**. The service worker is then registered with navigator.serviceWorker.register, which returns a promise that resolves when the service worker has been successfully registered. The scope of the service worker is then logged with registration.scope. If the service worker is already installed, navigator.serviceWorker.register returns the registration object of the currently active service worker.

The scope of the service worker determines which files the service worker controls, in other words, from which path the service worker will intercept requests. The default scope is the location of the service worker file, and extends to all directories below. So if service-worker.js is located in the root directory, the service worker will control requests from all files at this domain.

You can also set an arbitrary scope by passing in an additional parameter when

registering. For example: main.js

```
navigator.serviceWorker.register('/ser
  vice-worker.js', { scope: '/app/'

});
```

In this case we are setting the scope of the service worker to /app/, which means the service worker will control requests from pages like /app/, /app/lower/ and /app/lower/lower, but not from pages like /app or /, which are higher.

If you want the service worker to control higher pages e.g. /app (without the trailing slash) you can indeed change the scope option, but you'll also need to set the Service-Worker-Allowed HTTP Header in your server config for the request serving the service worker script.

CODE:

Index.html

manifest.json M | app.js U | service-worker.js U | index.html M

index.html > html > body.auth-body.dark > script

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="description" content="fuzzy" />
    <meta name="keywords" content="fuzzy" />
    <meta name="author" content="fuzzy" />
    <link rel="manifest" href="manifest.json" />
    <link rel="icon" href="assets/images/logo/favicon.png" type="image/x-icon" />
    <title>fuzzy</title>
    <link rel="apple-touch-icon" href="assets/images/logo/favicon.png" />
    <meta name="theme-color" content="#122636" />
    <meta name="apple-mobile-web-app-capable" content="yes" />
    <meta name="apple-mobile-web-app-status-bar-style" content="black" />
    <meta name="apple-mobile-web-app-title" content="fuzzy" />
    <meta name="msapplication-TileImage" content="assets/images/logo/favicon.png" />
    <meta name="msapplication-TileColor" content="#FFFFFF" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
```

```
</head>

<body class="auth-body dark">
  <!-- onboarding section start -->
  <section class="section-b-space">
    <div class="swiper intro slider-1">
      <div class="swiper-wrapper">
        <div class="swiper-slide">
          <div class="theme-logo pb-3">
            <img class="img-fluid logo-img" src="assets/images/logo/logo.png" alt="logo" />
          </div>
          <div class="onboarding-design">
            <img class="img-fluid design-img" src="assets/images/onboarding/design1.png" alt="b

            <img class="img-fluid slider-img1" src="assets/images/onboarding/1.png" alt="slider

            <img class="img-fluid vector1" src="assets/images/onboarding/vector1.png" alt="v1"
            <img class="img-fluid vector2" src="assets/images/onboarding/vector2.png" alt="v2"
            <img class="img-fluid vector3" src="assets/images/onboarding/vector3.png" alt="v3"
          </div>
          <div class="product-details">
            <h1>Office Furniture</h1>
            <span></span>
```

```
<!-- pwa install app popup start -->
<div class="offcanvas offcanvas-bottom addtohome-popup theme-offcanvas" tabindex="-1" id="offc
  <button type="button" class="btn-close text-reset" data-bs-dismiss="offcanvas" aria-label="C
  <div class="offcanvas-body small">
    <div class="app-info">
      <img src="assets/images/logo/48.png" class="img-fluid" alt="" />
      <div class="content">
        <h4>fuzzy app</h4>
        <a href="#">www.fuzzy-app.com</a>
      </div>
    </div>
    <a href="#!" class="btn theme-btn install-app btn-inline home-screen-btn m-0" id="installa
  </div>
</div>
<!-- pwa install app popup start -->

<!-- swiper js -->
<script src="assets/js/swiper-bundle.min.js"></script>
<script src="assets/js/custom-swiper.js"></script>

<!-- iconsax js -->
<script src="assets/js/iconsax.js"></script>
```

App.js

manifest.json M | app.js U | service-worker.js U | index.html M | style.css 5

app.js > ...

```
1  if ('serviceWorker' in navigator) {
2      window.addEventListener('load', () => {
3      navigator.serviceWorker.register('/service-worker.js')
4      .then(registration => {
5      console.log('Service Worker registered with scope:', registration.scope);
6      })
7      .catch(error => {
8      console.error('Service Worker registration failed:', error);
9      });
10     });
11     }
```

Service Worker.js

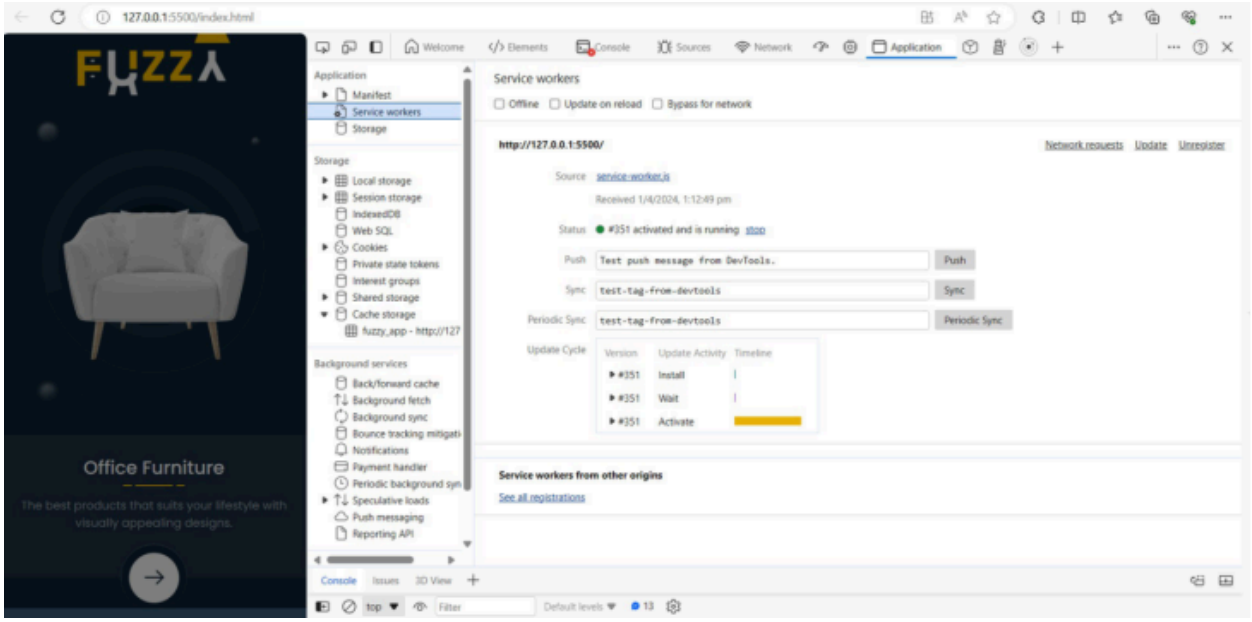
manifest.json M | app.js U | service-worker.js U | index.html M

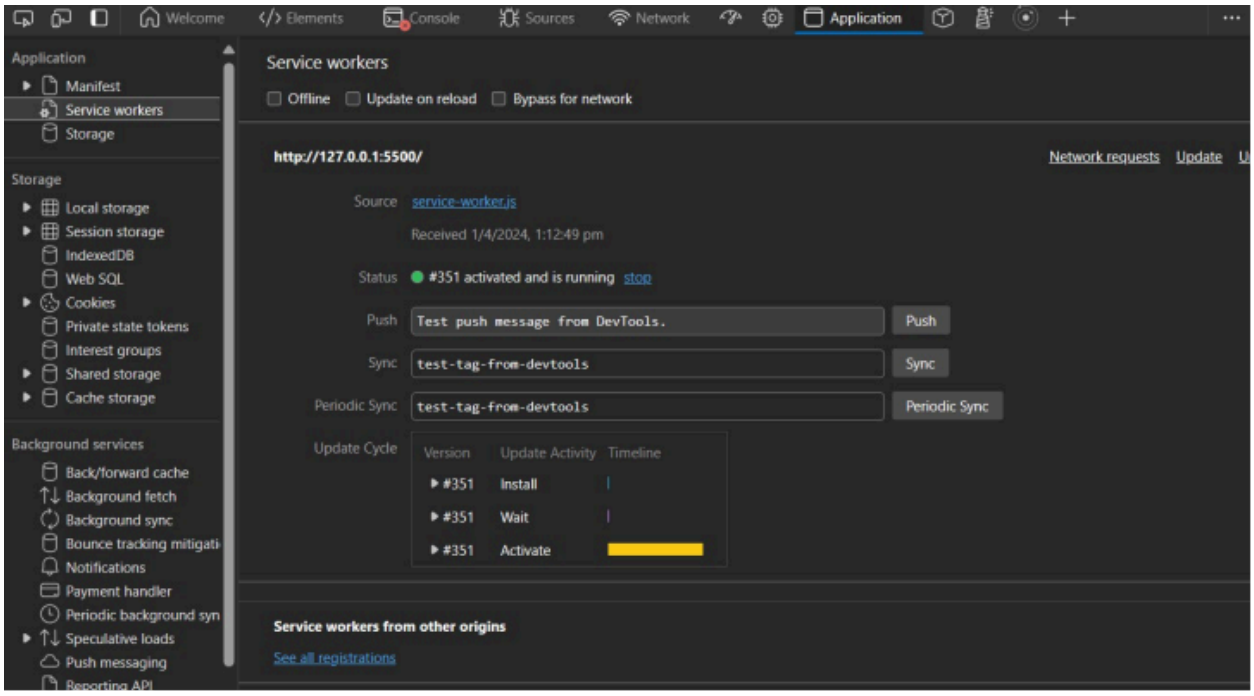
service-worker.js > assetsToCache

```
const cacheName = 'fuzzy_app';
const assetsToCache = [
'/',
'/index.html',
'/assets/css/style.css',
'/app.js'
]
self.addEventListener('install', event => {
event.waitUntil(
caches.open(cacheName)
.then(cache => {
return cache.addAll(assetsToCache);
})
);
});
self.addEventListener('activate', event => {
event.waitUntil(
caches.keys().then(cacheNames => {
return Promise.all(
cacheNames.filter(name => {
return name !== cacheName;
}).map(name => {
return caches.delete(name);
})
);
})
```

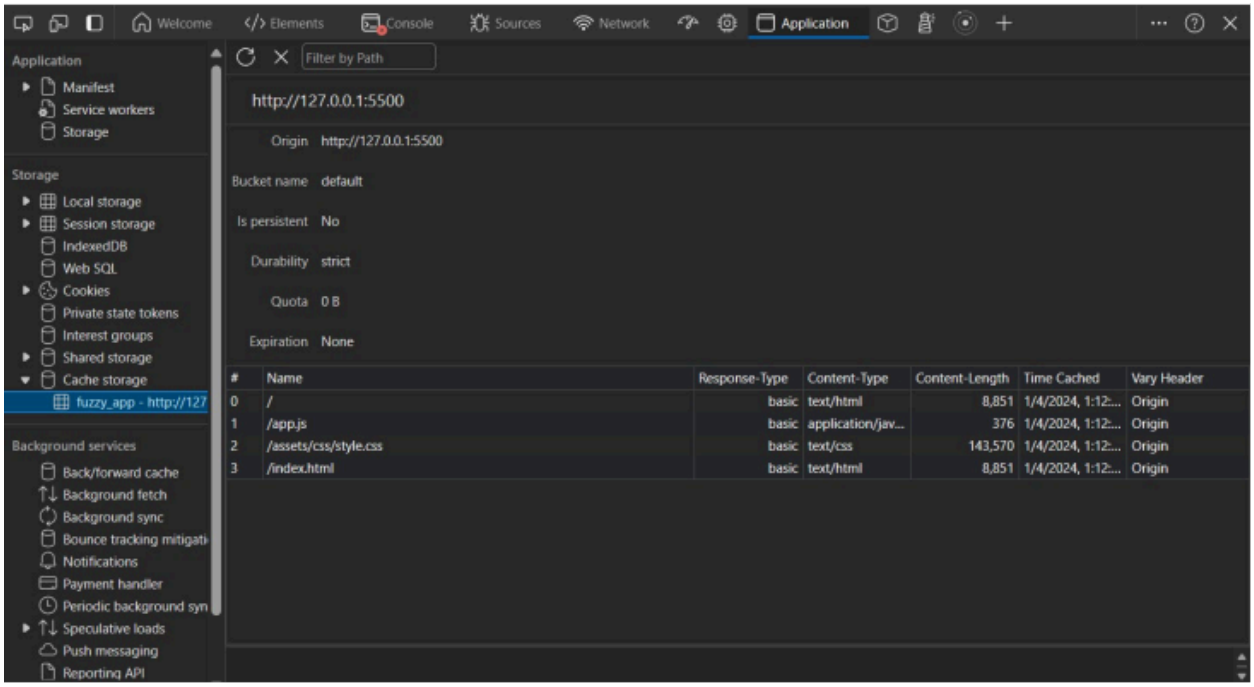
OUTPUT:

Project Title:Flutter:SkypeClone / PWA:Fuzzy App **Roll No. 18**





Cache Storage



Conclusion: In this experiment, we have registered a service worker, and completed the install and activation process for a new service worker for the E-commerce PW

MAD & PWA Lab
Journal

Experiment No.	09
Experiment Title.	To implement Service worker events like fetch, sync and push for E-commerce PWA
Roll No.	18
Name	Moksha Gawada
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	

Experiment No:09

Aim: To implement Service worker events like fetch, sync and push for E-commerce PWA.

Theory:

Service Worker

Service Worker is a script that works on browser background without user interaction independently. Also, It resembles a proxy that works on the user side. With this script, you can track network traffic of the page, manage push notifications and develop “offline first” web applications with Cache API.

Things to note about Service Worker:

- A service worker is a programmable network proxy that lets you control how network requests from your page are handled.
- Service workers only run over HTTPS. Because service workers can intercept network requests and modify responses, "man-in-the-middle" attacks could be very bad.
- The service worker becomes idle when not in use and restarts when it's next needed. You cannot rely on a global state persisting between events. If there is information that you need to persist and reuse across restarts, you can use IndexedDB databases.
- Service workers make extensive use of promises, so if you're new to promises, then you should stop reading this and check out Promises, an introduction.

Fetch Event

You can track and manage page network traffic with this event. You can check existing cache, manage “cache first” and “network first” requests and return a response that you want.

Of course, you can use many different methods but you can find in the following example a “cache first” and “network first” approach. In this example, if the request’s and current

location's origin are the same (Static content is requested.), this is called "cacheFirst" but if you request a targeted external URL, this is called "networkFirst".

- **CacheFirst** - In this function, if the received request has cached before, the cached response is returned to the page. But if not, a new response requested from the network.
- **NetworkFirst** - In this function, firstly we can try getting an updated response from the network, if this process completed successfully, the new response will be cached and returned. But if this process fails, we check whether the request has been cached before or not. If a cache exists, it is returned to the page, but if not, this is up to you. You can return dummy content or information messages to the page.

Code:

Service-Worker.js

```
self.addEventListener("install", function
   (event) { event.waitUntil(preLoad());

});

self.addEventListener("fetch", function
(event) { event.respondWith(
checkResponse(event.request).catch(functio
n () { console.log("Fetch from cache
successful!"); return
returnFromCache(event.request);

})

);

console.log("Fetch successful!");
event.waitUntil(addToCache(event.requesst
));

});

self.addEventListener("sync", (event)
=> { if (event.tag ===
```

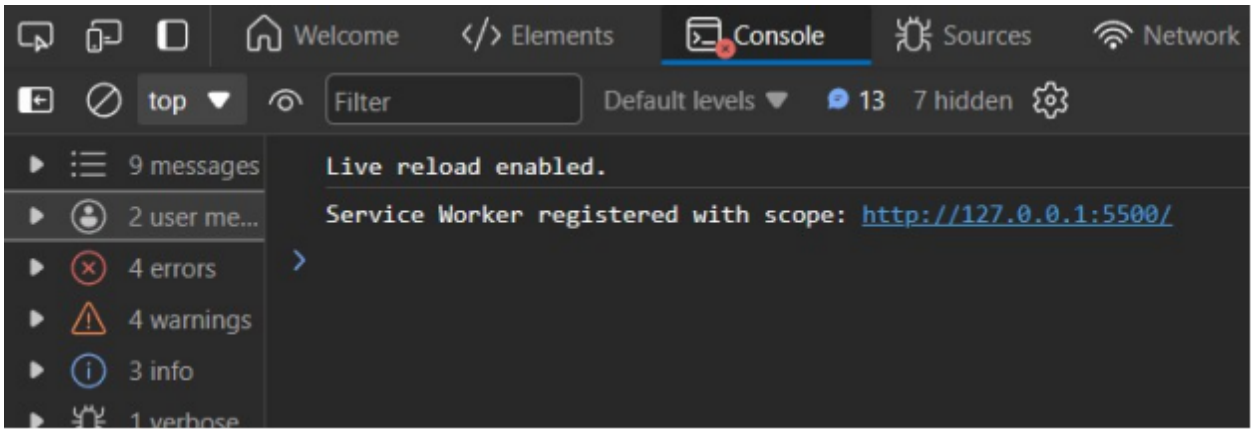
```
"syncMessage") { console.log("Sync
successful!");

}

});

self.addEventListener("push", function
(event) { if (event && event.data) {

try {

var data = event.data.json();

if (data && data.method === "pushMessage") {
console.log("Push notification sent");
self.registration.showNotification("Ecommerce website", {
body: data.message,

});

}

} catch (error) {

console.error("Error parsing push data:", error);

}

}

});

var preLoad = function () {

return caches.open("offline").then(function (cache) {


// caching index and important
routes return cache.addAll([

"/",

"/index.html",
"/landing.html",
```

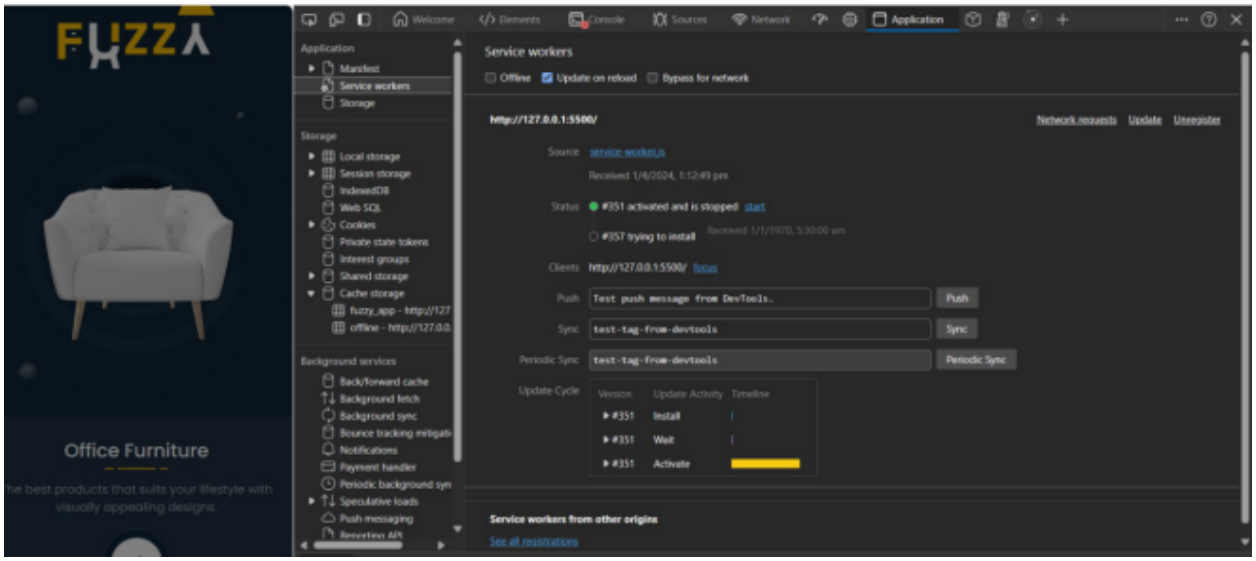
```
"/profile.html",
"/shop.html",

"/cart.html",
"/css/main.css",

]);

});

};

var checkResponse = function (request) {
return new Promise(function (fulfill,
reject) { fetch(request)

.then(function (response) {

if (response.status !== 404) {
fulfill(response);

} else {

reject(new Error("Response not found"));

}

})

.catch(function
(error) {
reject(error);

});

});

};

var returnFromCache = function (request) {

return caches.open("offline").then(function (cache)
{ return cache.match(request).then(function
(matching) { if (!matching || matching.status ==
404) {
```

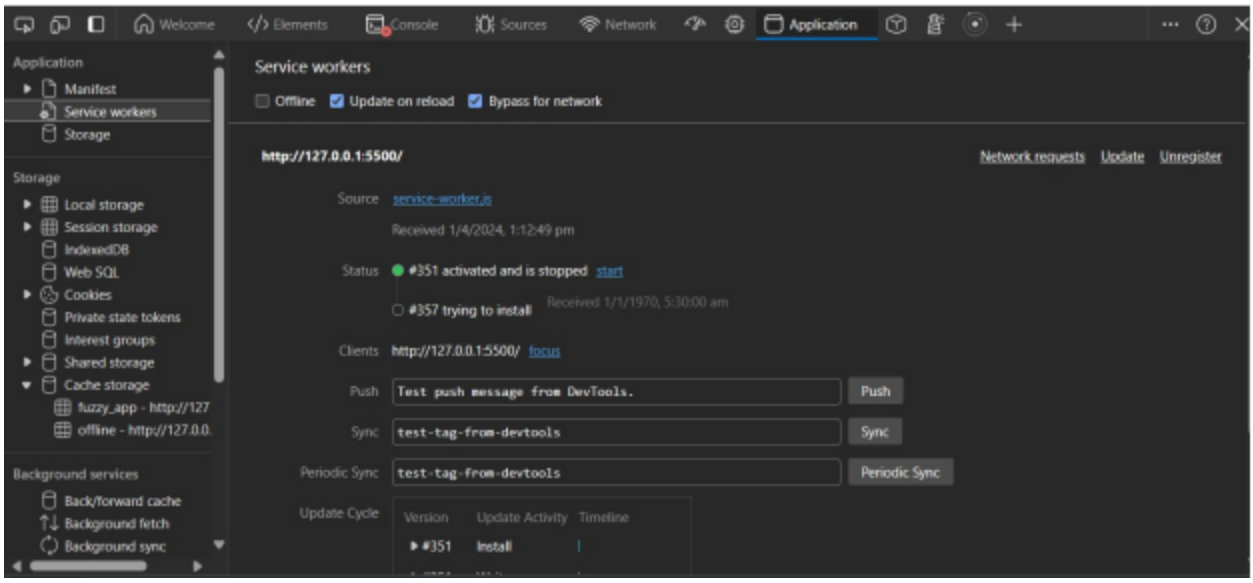
```
    return cache.match("offline.html");

    } else {

    return matching;

    }

    });

    });


    };

    var addToCache = function (request) {

    return caches.open("offline").then(function (cache)
    { return fetch(request).then(function (response) {

    return cache.put(request, response.clone()).then(function ()
    { return response;

}
)
;

}
)
;

}
)
```

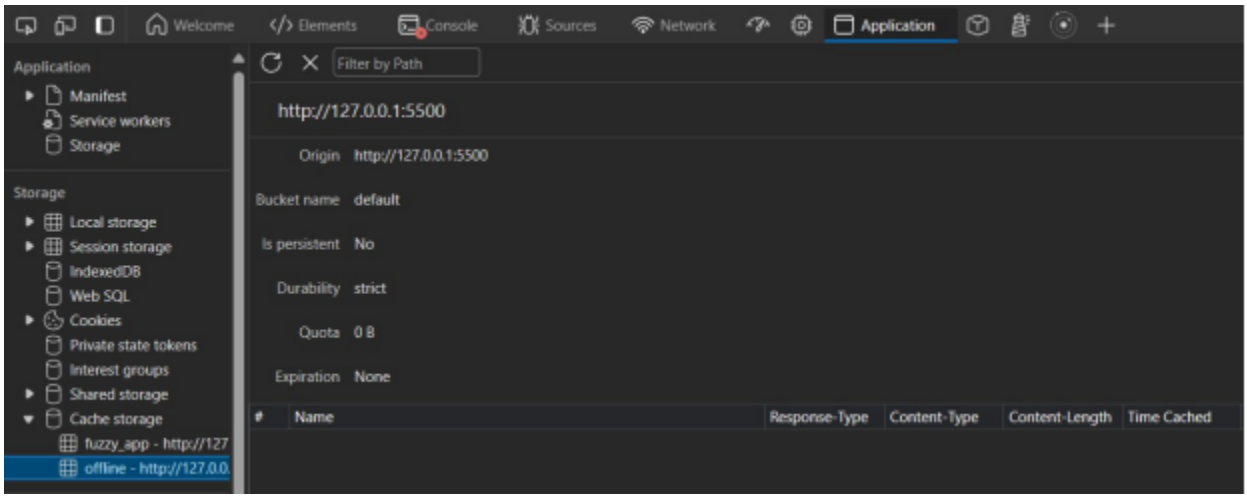
OUTPUT:

Project Title:Flutter:SkypeClone / PWA:Fuzzy App **Roll No. 18**









Conclusion : In this experiment, we have successfully implemented service worker events like fetch, sync and push for E-commerce PWA and found out output for above implementation

MAD & PWA Lab
Journal

Experiment No.	10
Experiment Title.	To study and implement deployment of Ecommerce PWA to GitHub Pages.
Roll No.	18
Name	Moksha Gawada
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	

Experiment No. 10

Aim:To study and implement deployment of Ecommerce PWA to GitHub Pages.

Theory:

GitHub Pages

Public web pages are freely hosted and easily published. Public webpages hosted directly from your GitHub repository. Just edit, push, and your changes are live.

GitHub Pages provides the following key features:

1. Blogging with Jekyll
2. Custom URL
3. Automatic Page Generator

Reasons for favoring this over Firebase:

1. Free to use
2. Right out of github
3. Quick to set up

GitHub Pages is used by Lyft, CircleCI, and HubSpot.

GitHub Pages is listed in 775 company stacks and 4401 developer stacks.

Pros

1. Very familiar interface if you are already using GitHub for your projects.
2. Easy to set up. Just push your static website to the gh-pages branch and your website is ready.
3. Supports Jekyll out of the box.
4. Supports custom domains. Just add a file called CNAME to the root of your site, add an A record in the site's DNS configuration, and you are done.

Cons

1. The code of your website will be public, unless you pay for a private repository.
2. Currently, there is no support for HTTPS for custom domains. It's probably coming soon though.
3. Although Jekyll is supported, plug-in support is rather spotty.

Firebase

The Realtime App Platform. Firebase is a cloud service designed to power real-time, collaborative applications. Simply add the Firebase library to your application to gain access to a shared data structure; any changes you make to that data are automatically synchronized with the Firebase cloud and with other clients within milliseconds.

Some of the features offered by Firebase are:

1. Add the Firebase library to your app and get access to a shared data structure. Any changes made to that data are automatically synchronized with the Firebase cloud and with other clients within milliseconds.
2. Firebase apps can be written entirely with client-side code, update in real-time out-of-the-box, interoperate well with existing services, scale automatically, and provide strong data security.
3. Data Accessibility- Data is stored as JSON in Firebase. Every piece of data has its own URL which can be used in Firebase's client libraries and as a REST endpoint. These URLs can also be entered into a browser to view the data and watch it update in real-time.

Reasons for favoring over GitHub Pages:

1. Realtime backend made easy
2. Fast and responsive

Instacart, 9GAG, and Twitch are some of the popular companies that use Firebase
Firebase has a broader approval, being mentioned in 1215 company stacks & 4651 developers stacks

Pros

1. Hosted by Google. Enough said.
2. Authentication, Cloud Messaging, and a whole lot of other handy services will be available to you.
3. A real-time database will be available to you, which can store 1 GB of data.
4. You'll also have access to a blob store, which can store another 1 GB of data.
5. Support for HTTPS. A free certificate will be provisioned for your custom domain within 24 hours.

Cons

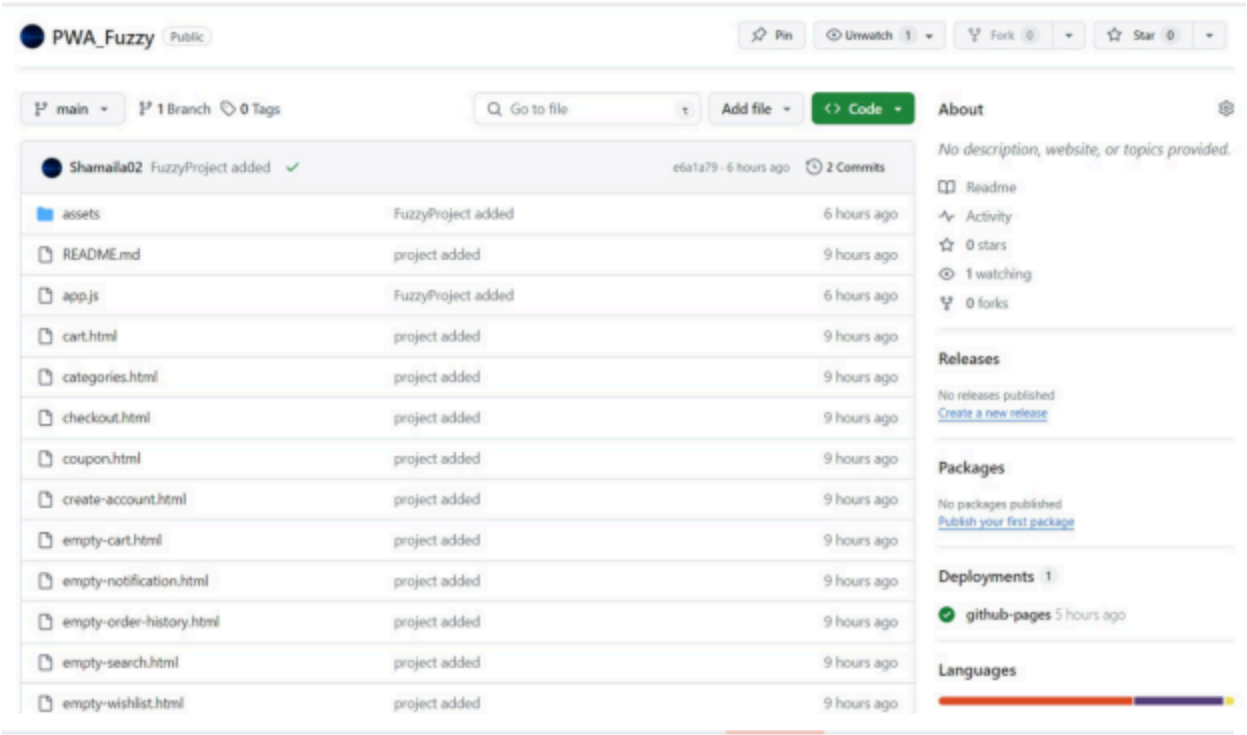
1. Only 10 GB of data transfer is allowed per month. But this is not really a big problem, if you use a CDN or AMP.
2. Command-line interface only.
3. No in-built support for any static site generator.

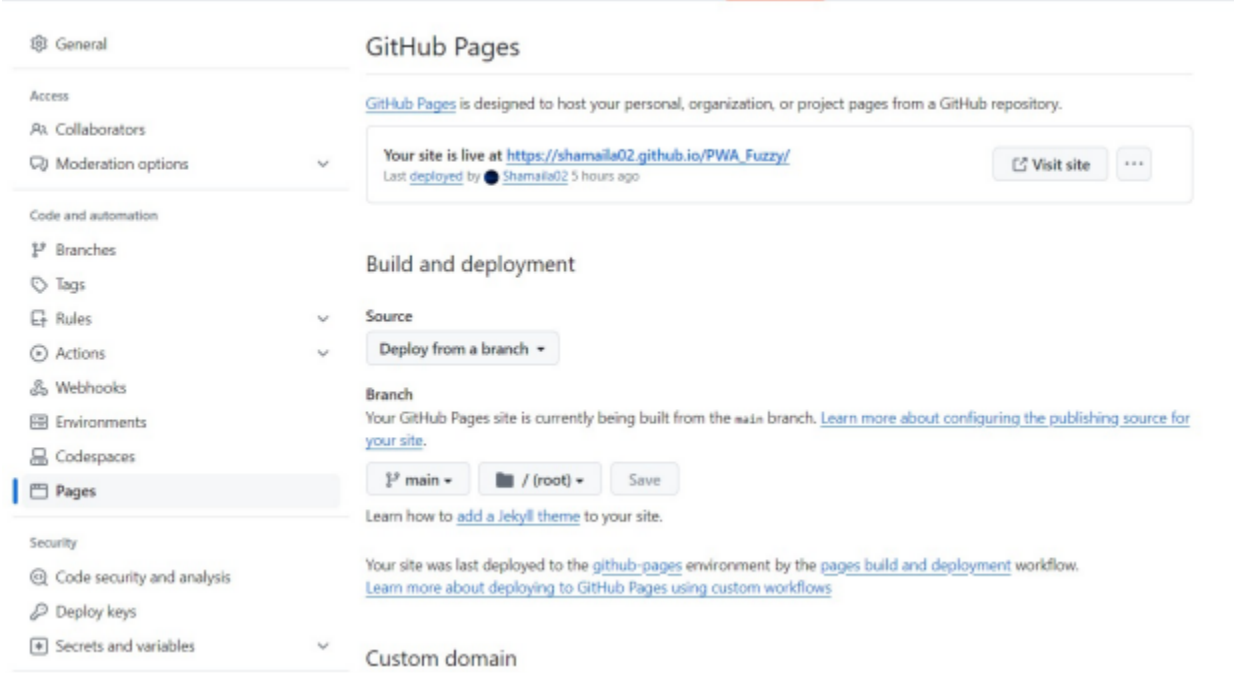
Link to our GitHub repository: https://github.com/Shamaila02/PWA_Fuzzy

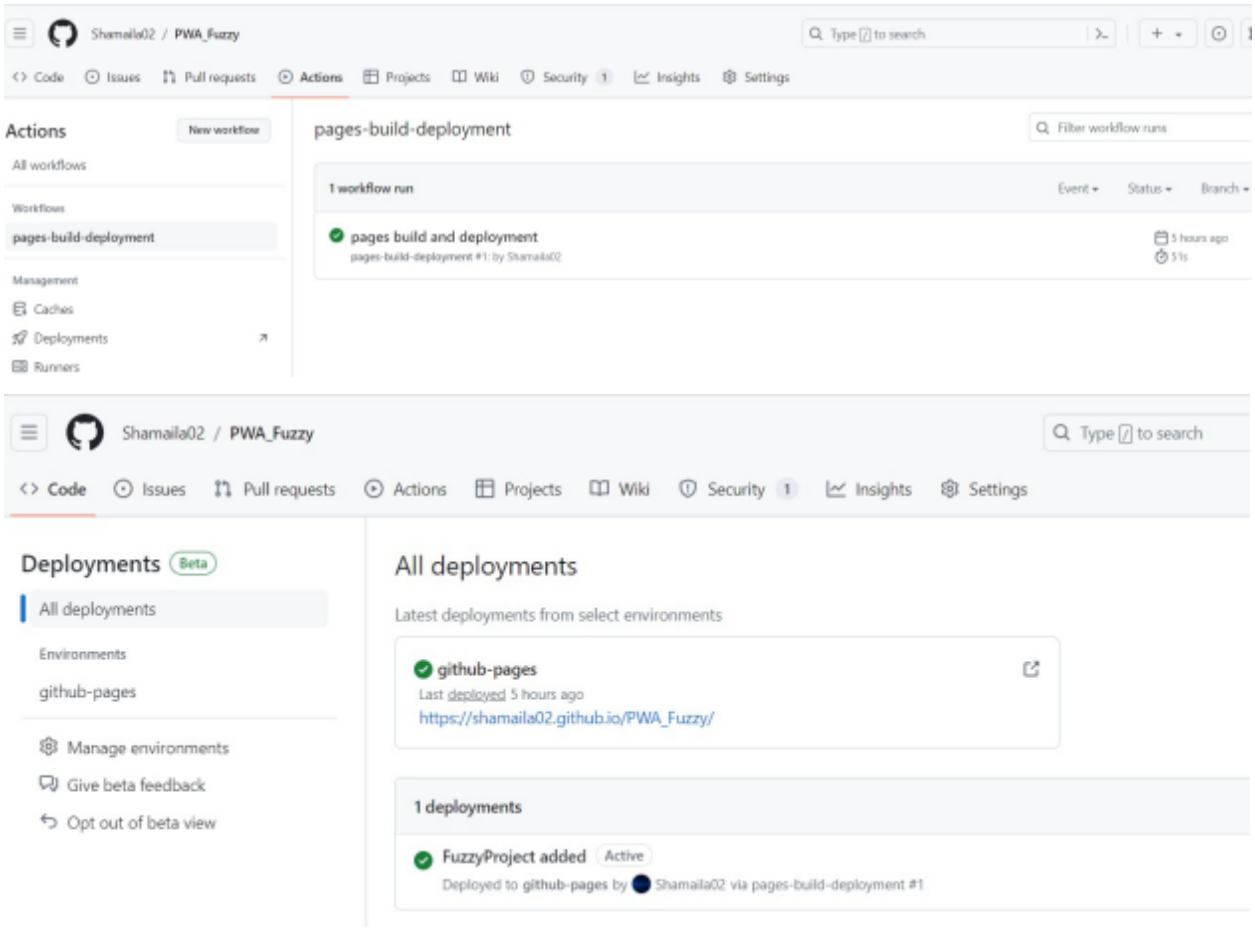
Hosted Link: https://shamaila02.github.io/PWA_Fuzzy/

Github Screenshot:

Project Title:Flutter:SkypeClone / PWA:Fuzzy App **Roll No. 18**







Conclusion: In this experiment we have Successfully deployed the Ecommerce PWA to GitHub Pages.

MAD & PWA Lab
Journal

Experiment No.	11
Experiment Title.	To use google Lighthouse PWA Analysis Tool to test the PWA functioning.
Roll No.	18
Name	Moksha Gawada
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO6: Develop and Analyze PWA Features and deploy it over app hosting solution
Grade:	15

Experiment 11

Aim : To use google Lighthouse PWA Analysis Tool to test the PWA functioning.

Theory : Reference : https://www.semrush.com/blog/google-lighthouse/

Google Lighthouse :

Google Lighthouse is a tool that lets you audit your web application based on a number of parameters including (but not limited to) performance, based on a number of metrics, mobile compatibility, Progressive Web App (PWA) implementations, etc. All you have to do is run it on a page or pass it a URL, sit back for a couple of minutes and get a very elaborate report, not much short of one that a professional auditor would have compiled in about a week.

The best part is that you have to set up almost nothing to get started. Let's begin by looking at some of the top features and audit criteria used by Lighthouse.

Key Features and Audit Metrics

Google Lighthouse has the option of running the Audit for Desktop as well as mobile version of your page(s). The top metrics that will be measured in the Audit are:

1. **Performance:** This score is an aggregation of how the page fared in aspects such as (but not limited to) loading speed, time taken for loading for basic frame(s), displaying meaningful content to the user, etc. To a layman, this score is indicative of how decently the

site performs, with a score of 100 meaning that you figure in the 98th percentile, 50 meaning that you figure in the 75th percentile and so on.

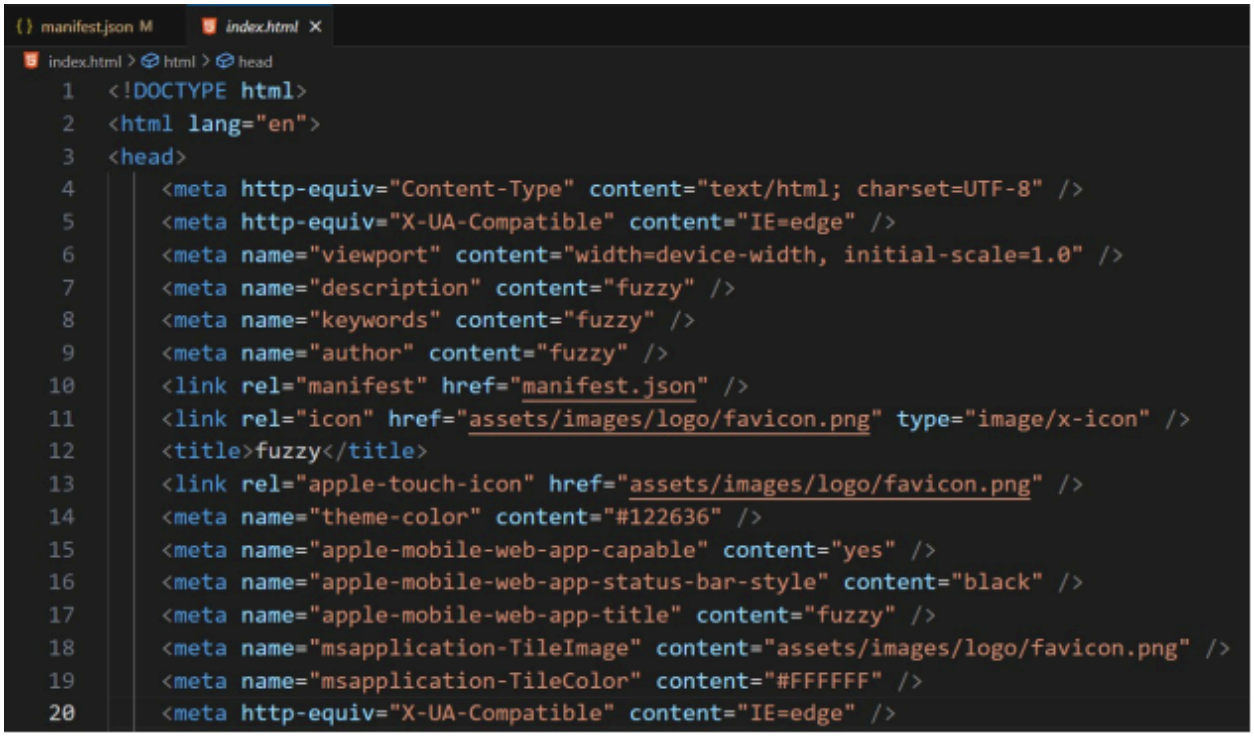
2. **PWA Score (Mobile):** Thanks to the rise of Service Workers, app manifests, etc., a lot of modern web applications are moving towards the PWA paradigm, where the objective is to make the application behave as close as possible to native mobile applications. Scoringpoints are based on the Baseline PWA checklist laid down by

Google which includes Service Worker implementation(s), viewport handling, offline functionality, performance in script-disabledenvironments, etc.

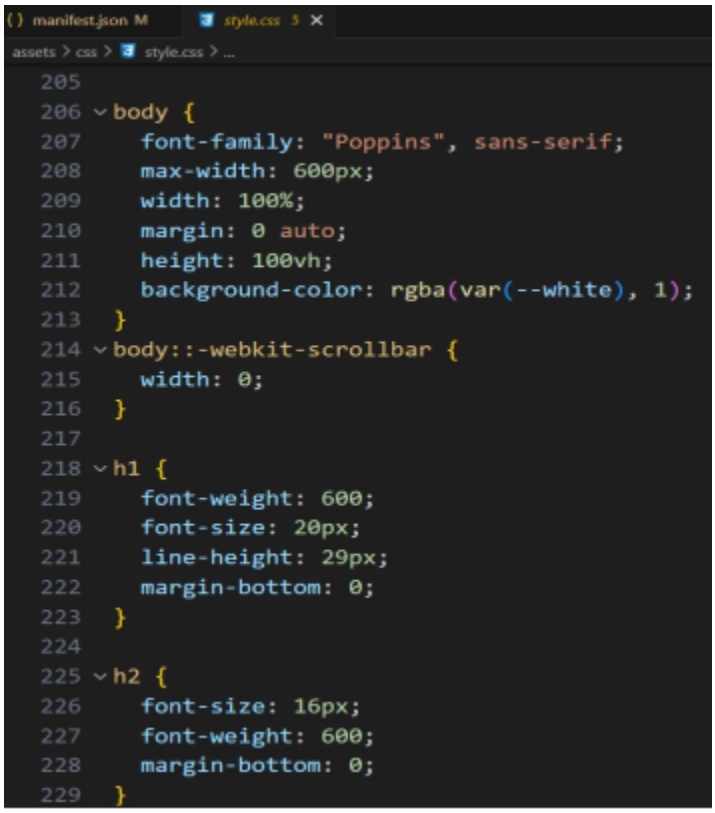
3. **Accessibility:** As you might have guessed, this metric is a measure of how accessible your website is, across a plethora of accessibility features that can be implemented in your page (such as the 'aria-'attributes like aria-required, audio captions, button names, etc.). Unlike the other metrics though, Accessibility metrics score on a pass/fail basis i.e. if all possible elements of the page are notscreen-reader friendly (HTML5 introduced features that would make pages easy to interpret for screen readers used by visuallychallenged people like tag names, tags such as <section>, <article>, etc.), you get a 0 on that score. The aggregate of these scores is your Accessibility metric score.
4. **Best Practices:** As any developer would know, there are a number of practices that have been deemed 'best' based on empirical data. This metric is an aggregation of many such points, including but not limited to:Use of HTTPSAvoiding the use of deprecated code elements like tags, directives, libraries, etc. Password input with paste-into disabled Geo-Location and cookie usage alerts on load, etc.

Code&Implementation:

Index.html



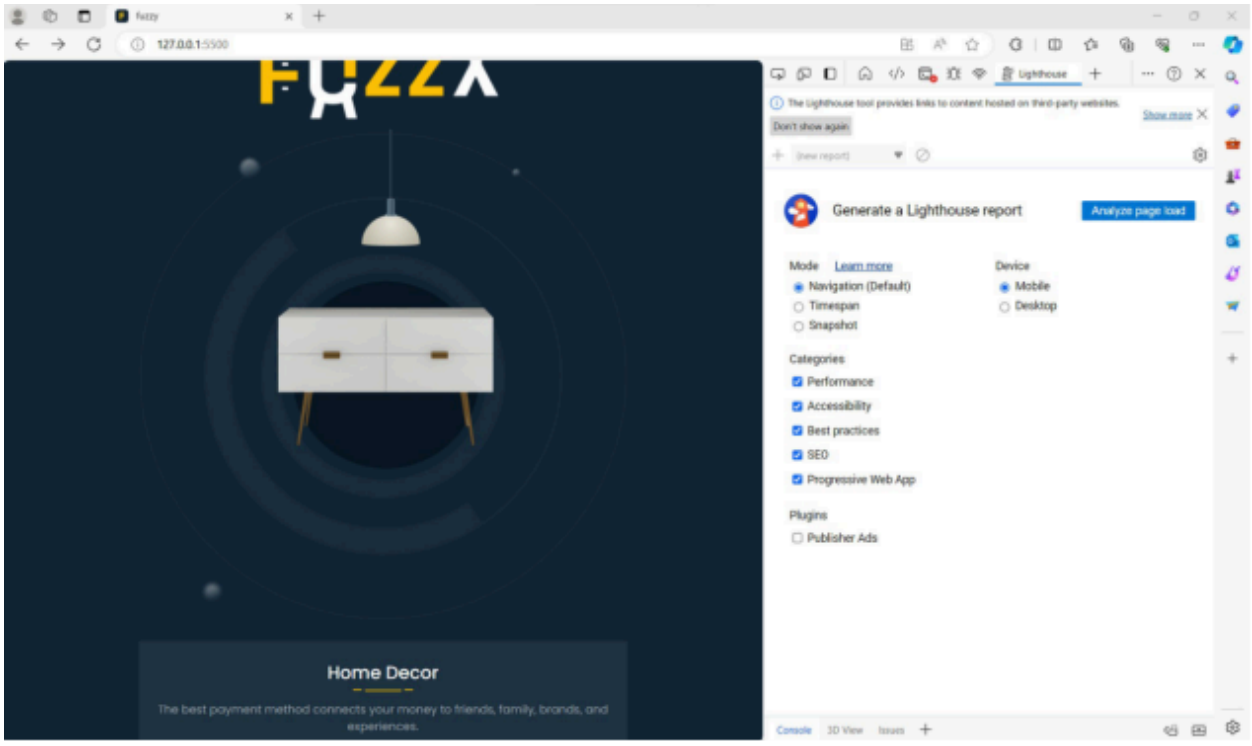
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="description" content="fuzzy" />
    <meta name="keywords" content="fuzzy" />
    <meta name="author" content="fuzzy" />
    <link rel="manifest" href="manifest.json" />
    <link rel="icon" href="assets/images/logo/favicon.png" type="image/x-icon" />
    <title>fuzzy</title>
    <link rel="apple-touch-icon" href="assets/images/logo/favicon.png" />
    <meta name="theme-color" content="#122636" />
    <meta name="apple-mobile-web-app-capable" content="yes" />
    <meta name="apple-mobile-web-app-status-bar-style" content="black" />
    <meta name="apple-mobile-web-app-title" content="fuzzy" />
    <meta name="msapplication-TileImage" content="assets/images/logo/favicon.png" />
    <meta name="msapplication-TileColor" content="#FFFFFF" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
```

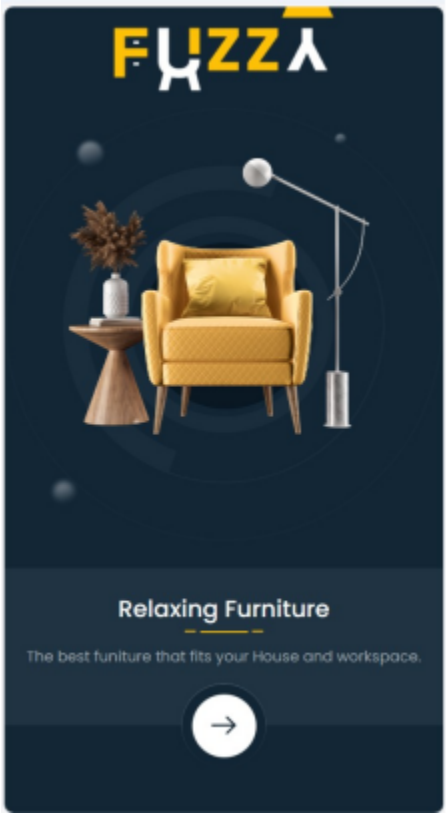


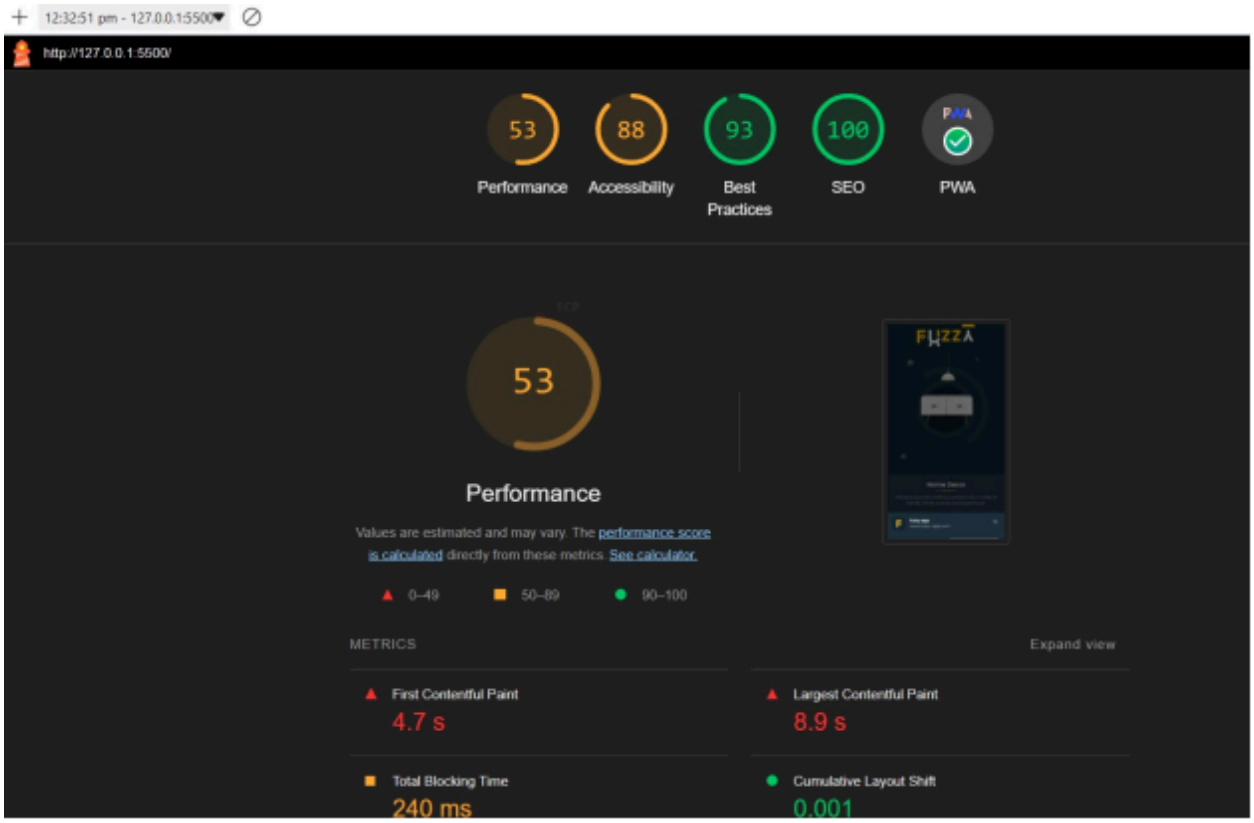
```
body {
    font-family: "Poppins", sans-serif;
    max-width: 600px;
    width: 100%;
    margin: 0 auto;
    height: 100vh;
    background-color: rgba(var(--white), 1);
}
body::-webkit-scrollbar {
    width: 0;
}

h1 {
    font-weight: 600;
    font-size: 20px;
    line-height: 29px;
    margin-bottom: 0;
}

h2 {
    font-size: 16px;
    font-weight: 600;
    margin-bottom: 0;
}
```

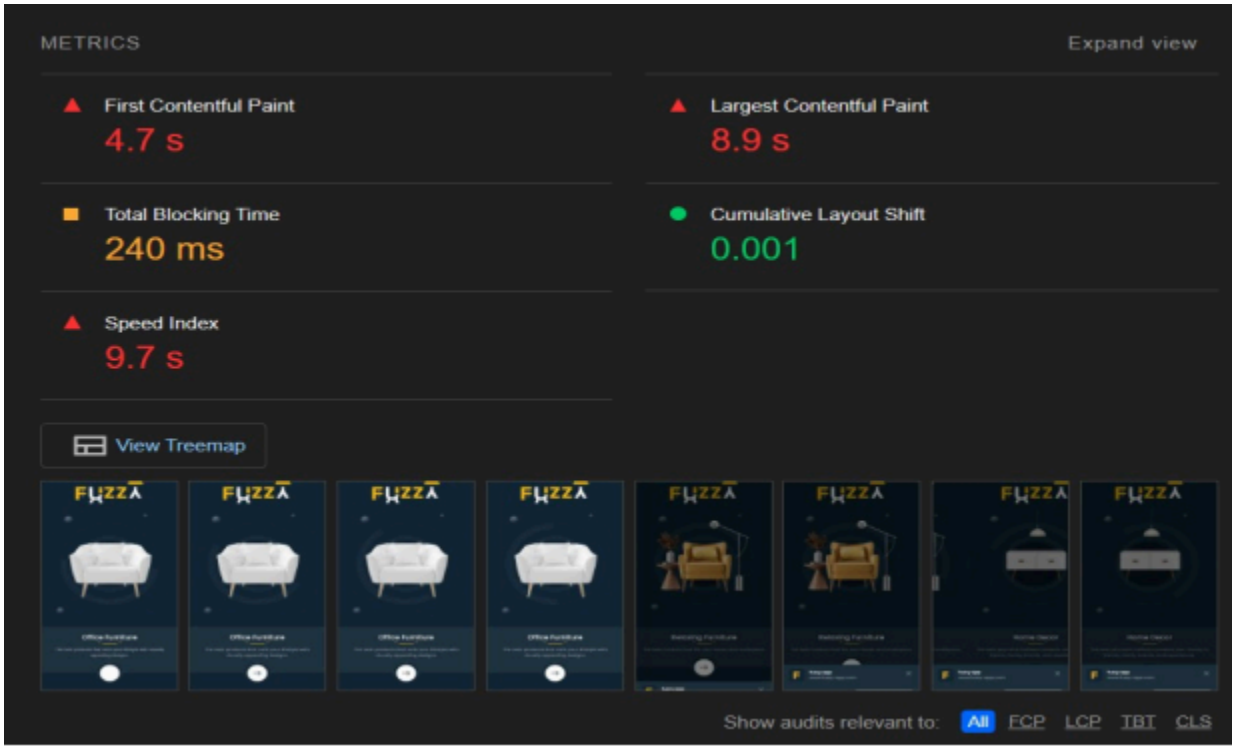
OUTPUT:

Project Title:Flutter:SkypeClone / PWA:Fuzzy App **Roll No. 18**

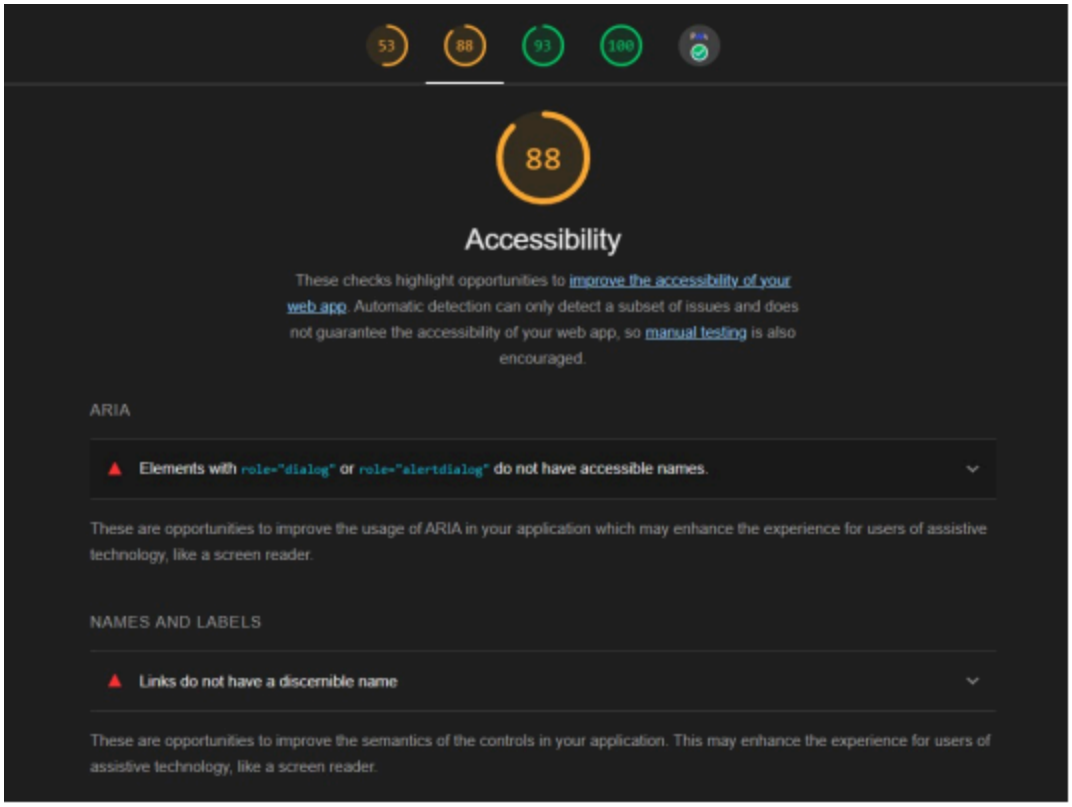




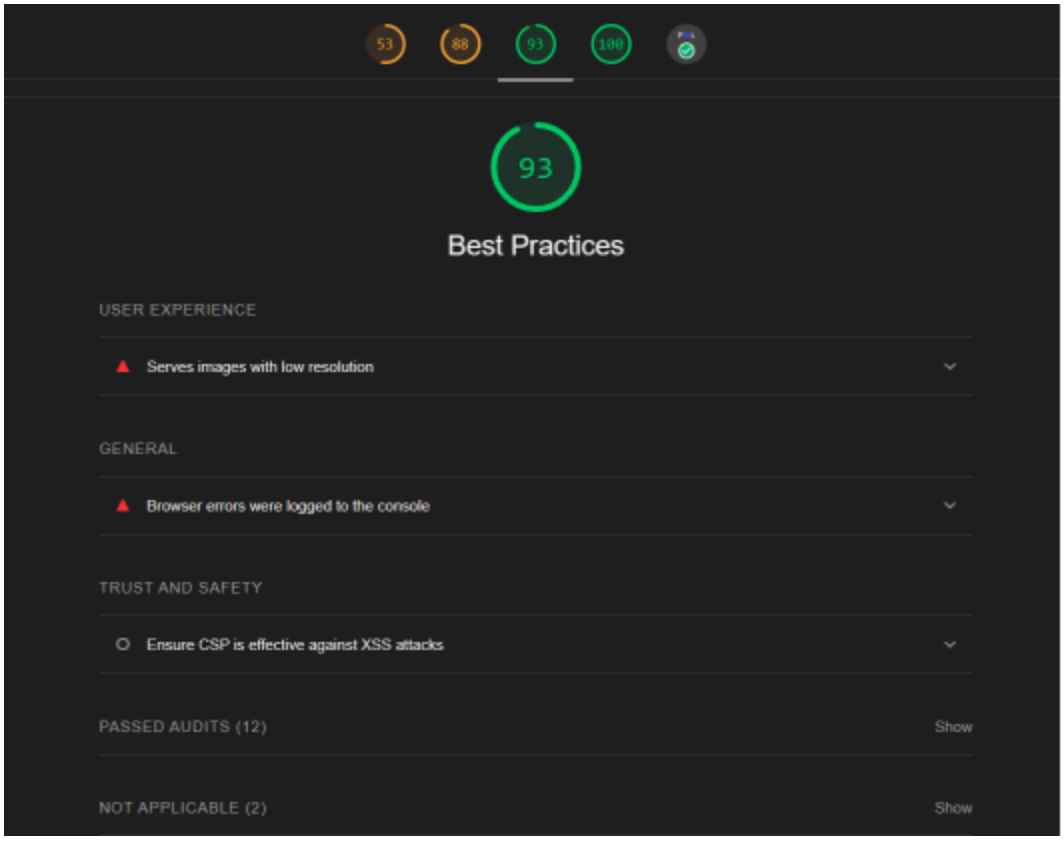

12:32:51 pm - 127.0.0.1:5500
http://127.0.0.1:5500/
53
Performance
88
Accessibility
93
Best Practices
100
SEO
PWA
53
Performance
Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.
0–49
50–89
90–100
METRICS
Expand view
First Contentful Paint
4.7 s
Largest Contentful Paint
8.9 s
Total Blocking Time
240 ms
Cumulative Layout Shift
0.001

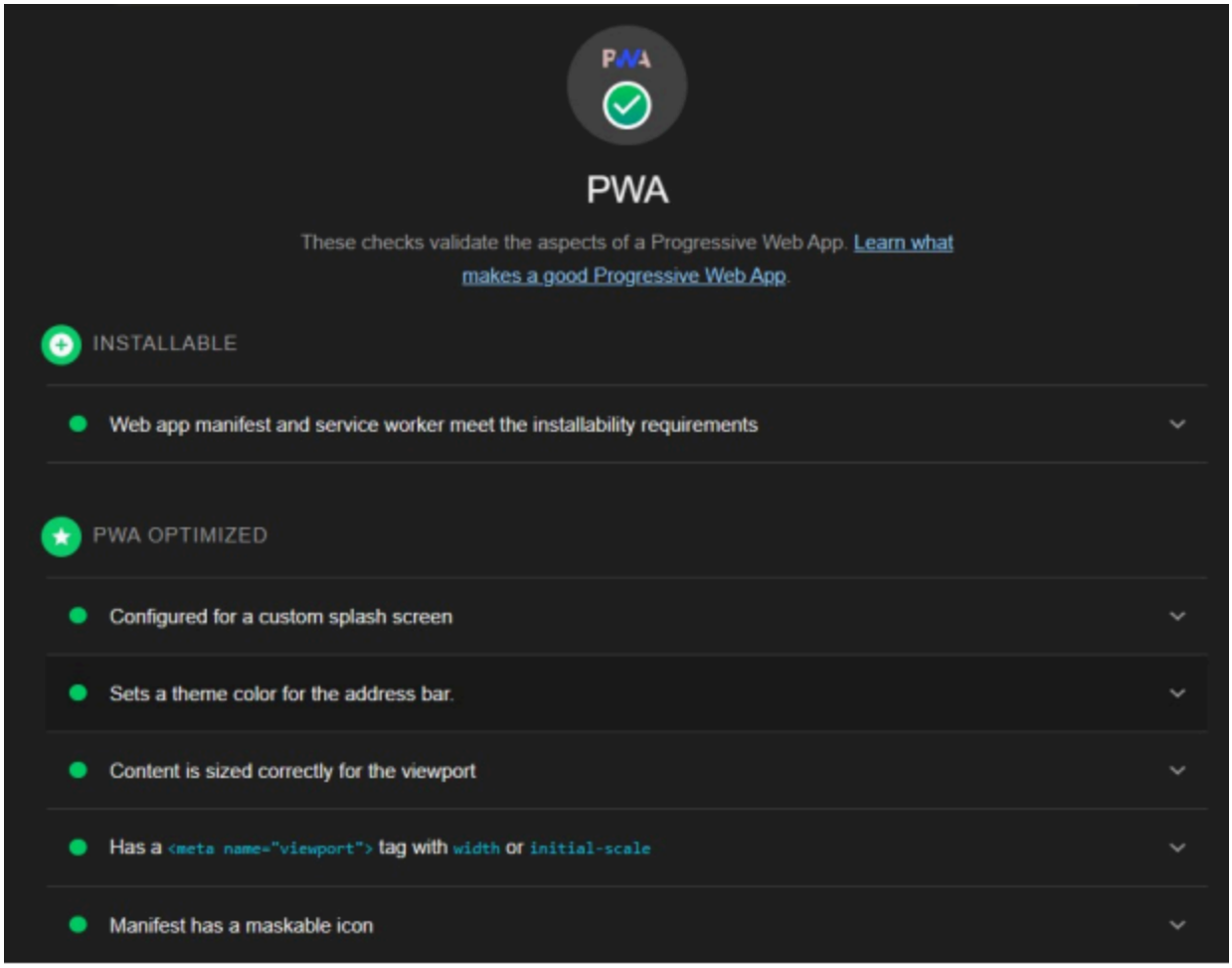

METRICS
Expand view
First Contentful Paint
4.7 s
Largest Contentful Paint
8.9 s
Total Blocking Time
240 ms
Cumulative Layout Shift
0.001
Speed Index
9.7 s
View Treemap
Show audits relevant to:
All
FCP
LCP
TBT
CLS

Project Title:Flutter:SkypeClone / PWA:Fuzzy App **Roll No. 18**



Project Title:Flutter:SkypeClone / PWA:Fuzzy App **Roll No. 18**

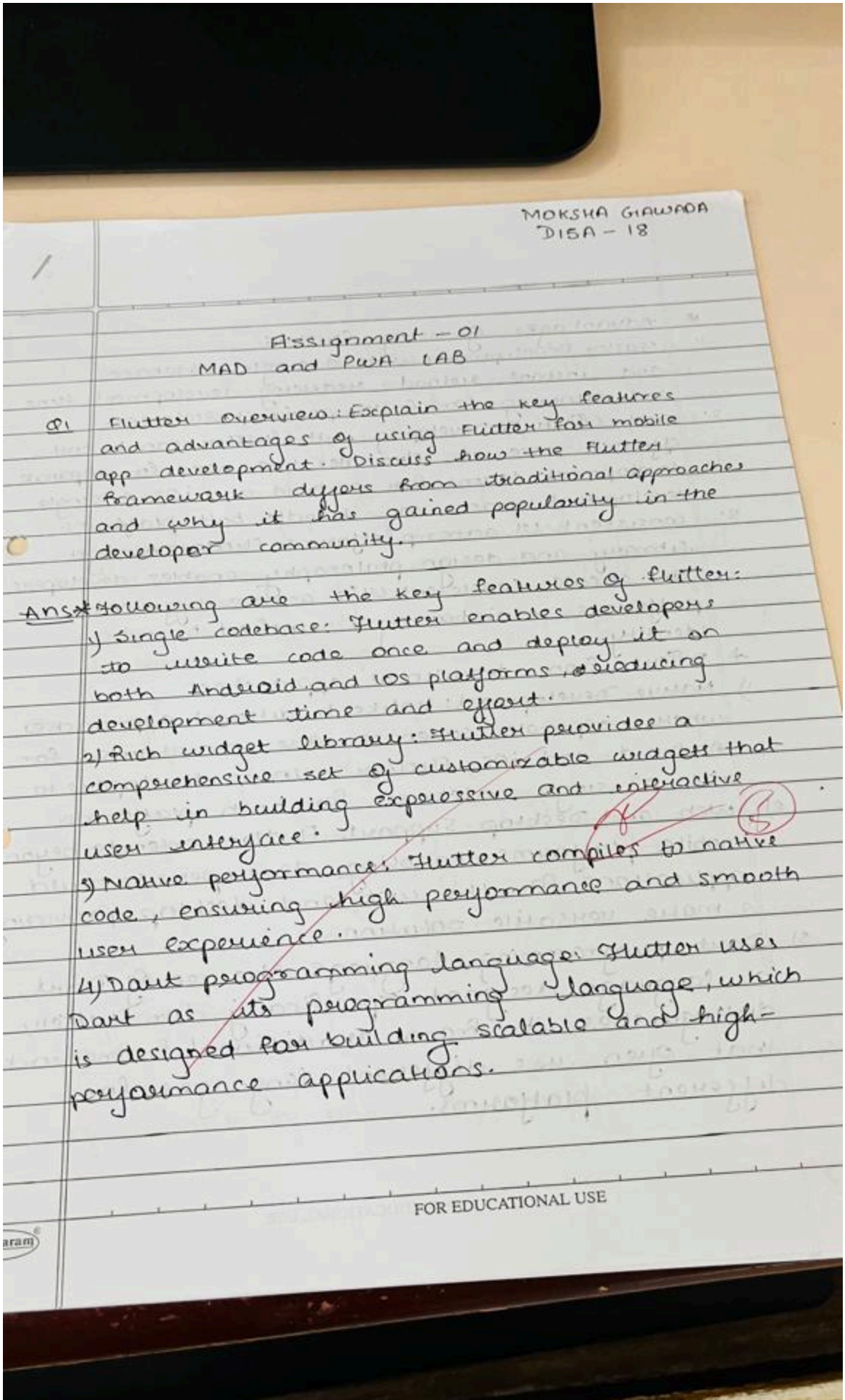




Conclusion: Thus we successfully used google Lighthouse PWA Analysis Tool for testing the PWA functioning.

MAD & PWA Lab
Journal

Experiment No.	Assignment-1
Assignment 1 Questions	1. Flutter Overview:Explain the key features and advantages of using Flutter for mobile app development. Discuss how the Flutter framework differs from traditional approaches and why it has gained popularity in the developer community. 2. Widget Tree and Composition: Describe the concept of the widget tree in Flutter. Explain how widget composition is used to build complex user interfaces. Provide examples of commonly used widgets and their roles in creating a widget tree. 3. State Management in Flutter: Discuss the importance of state management in Flutter applications. Compare and contrast the different state management approaches available in Flutter, such as setState, Provider, and Riverpod. Provide scenarios where each approach is suitable. 4. Firebase Integration in Flutter: Explain the process of integrating Firebase with a Flutter application. Discuss the benefits of using Firebase as a backend solution. Highlight the Firebase services commonly used in Flutter development and provide a brief overview of how data synchronization is achieved.
Roll No.	18
Name	Moksha Gawada
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO1: Understand cross platform mobile application development using Flutter framework LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation LO3: Analyze and Build production ready Flutter App by incorporating backend services and deploying on Android / iOS
Grade:	5



MOKSHA GIAWADA
D15A - 18

Assignment - 01
MAD and PWA LAB

Q1 Flutter Overview: Explain the key features and advantages of using Flutter for mobile app development. Discuss how the Flutter framework differs from traditional approaches and why it has gained popularity in the developer community.

Ans* Following are the key features of flutter:
1) Single codebase: Flutter enables developers to write code once and deploy it on both Android and iOS platforms, reducing development time and effort.
2) Rich widget library: Flutter provides a comprehensive set of customizable widgets that help in building expressive and interactive user interface.
3) Native performance: Flutter compiles to native code, ensuring high performance and smooth user experience.
4) Dart programming language: Flutter uses Dart as its programming language, which is designed for building scalable and high-performance applications.

FOR EDUCATIONAL USE

* Advantages of using flutter:
1. Faster Development: with a single codebase and instant reload, reducing development time and allowing for faster product releases.
2. Cost-Effective: Developing with flutter can be cost-effective since it eliminates the need for separate development teams for android and iOS. A single development team can handle both platforms.
3. Consistent UI across platforms: Flutters widget library and design philosophy enables developers to create consistent UI's across different platforms, maintaining a unified brand identity.

* Differs from traditional approaches:
1) Native Development: Unlike traditional approaches where separate codebases are maintained for Android and iOS, Flutter allows developers to use a single codebase for both platforms.
2) Web and Desktop Support: Flutter extends beyond mobile platforms, enabling developers to build applications for the web and desktop, providing a more versatile solution.
3) Dart programming language: The use of Dart, a language designed specifically for Flutter, distinguishes it from traditional frameworks that often use different languages for different platforms.

* Popularity in developer community:
1. Versatility: Fluttters ability to target multiple platforms from a single codebase appeals to developers looking for a versatile solution that extends beyond mobile app development.
2. Growing Ecosystem: The active and growing Flutter community, along with a rich ecosystem of packages and plugins, has contributed to the frameworks popularity.
3. Ease of Learning: Flutter's straightforward and expressive syntax, combined with comprehensive documentation, makes it relatively easy for developers to learn and adopt.

Q2 Widget tree and composition: Describe the concept of widget tree in Flutter. Explain how widget composition is used in Flutter to build complex user interfaces. Provide examples of commonly used widgets and their roles in creating a widget tree.

Ans 1) In Flutter, the widget tree is a hierarchical structure that represents the user interface of an application.
2) Each element in the tree is a widget, which is a lightweight, immutable descriptio of a part of the user interface.

3) Flutter follows a declarative approach, where the UI is expressed as a tree of widgets, and changes to the UI are made by creating a new widget tree.

* Widget tree concept:

1) Widgets:
a) Widgets are the basic building blocks in Flutter
b) They can represent structural elements (eg- containers, columns, rows) or visuals elements (e.g. Text, images, buttons).
c) Widgets are either StatelessWidget (immutable) or StatefulWidget (mutable).

2) Hierarchy:
a) The widget tree is a hierarchy where each widget has a parent and zero or more children.
b) The root of the tree is usually the top-level widget of the application, and
c) it branches out to represent the entire UI.

3) Composition:
a) Complex UI's are built by composing simple widgets into more complex ones.
b) This composition allows for a modular and reusable approach to UI development.

* Widget composition:

1) Container widget: The container widget is a basic building block that can contain other widgets. It is often used to define the size, padding, and decoration of a widget.
2) Row and column widgets: "Row" and "Column" widgets are used to arrange child widgets horizontally and vertically, respectively.
3) Stack widget: The 'Stack' widget allows widgets to be overlaid on top of each other. It's useful for creating complex layouts.
4) ListView and GridView widgets: 'ListView' and 'GridView' are used to create scrollable lists and grids of widgets.
5) TextField and Button widgets: "TextField" is used for user input, and various button widgets ('RaisedButton', 'IconButton') are used for user interaction.
6) Material Design widgets: Flutter provides a wide range of material design widgets, such as 'Card', 'Drawer', and 'BottomNavigationBar', to implement common design patterns.

By combining these basic widgets, developers can create feature-rich user interfaces. The widget tree allows for easy organization, reusability, and maintainance of UI components making it a fundamental concept in flutter development.

Q3 State Management in Flutter: Discuss the importance of state management in flutter application. Compare and contrast the different state management approaches in flutter, such as setState, Provider, and Riverpod. Provide scenarios where each approach is suitable.

Ans 1) State management is crucial in flutter applications to handle and update the state of user interface.

2) In flutter, UI components can be either stateful or stateless.

3) Stateful widgets have mutable state, that can change during the lifetime of the widget, such as user interactions or data fetching.

4) Proper state management ensures that changes in the state are reflected in the UI, providing a responsive and dynamic user experience.

* State Management approaches in Flutter:

1) setState:

→ Description: The 'setState' method is a basic and built-in mechanism for managing state in flutter. It belongs to 'StatefulWidget' class, and when called, it triggers a rebuild of the widget subtree.

→ Suitability : Suitable for small to medium-sized applications with a limited number of stateful widgets. Best for simple UI's with low complexity and minimal inter-widget dependencies.

2) Provider :

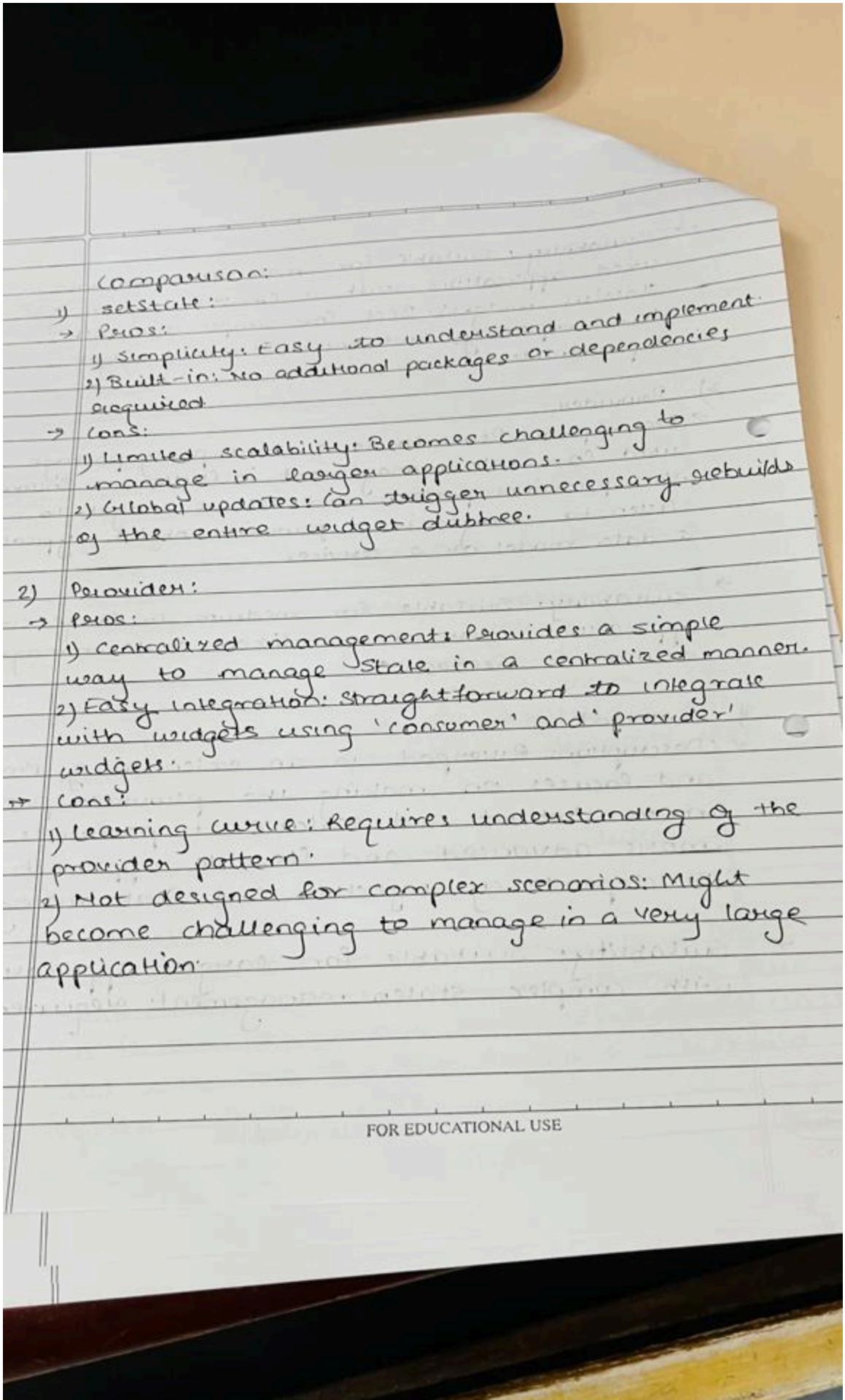
→ Description: Provider is a third-party package used for state management in flutter. It follows the provider pattern and allows widgets to listen to changes in a provided object, typically a data model or a service.

→ suitability: Suitable for medium to large-sized applications where a centralized state management solution is needed.

3) Riverpod:

→ Description: Riverpod is an extension of Provider and focuses on making the provider pattern more robust and scalable. It provides a more advanced and flexible approach to dependency injection and State managemen

Suitability: Suitable for larger applications with complex states management requirement



Comparison:

1) setState:

→ Pros:

1) Simplicity: Easy to understand and implement.
2) Built-in: No additional packages or dependencies required.

→ Cons:

1) Limited scalability: Becomes challenging to manage in larger applications.
2) Global updates: Can trigger unnecessary rebuilds of the entire widget subtree.

2) Provider:

→ Pros:

1) Centralized management: Provides a simple way to manage state in a centralized manner.
2) Easy integration: Straightforward to integrate with widgets using 'consumer' and 'provider' widgets.

→ Cons:

1) Learning curve: Requires understanding of the provider pattern.
2) Not designed for complex scenarios: Might become challenging to manage in a very large application.

3. Riverpod:

→ Pros:

1) Advanced features: Offers more advanced features such as asynchronous dependencies, lazy-loading, and read-only providers.

2) Scalability: Designed to handle complex state management scenarios in large applications.

→ Cons:

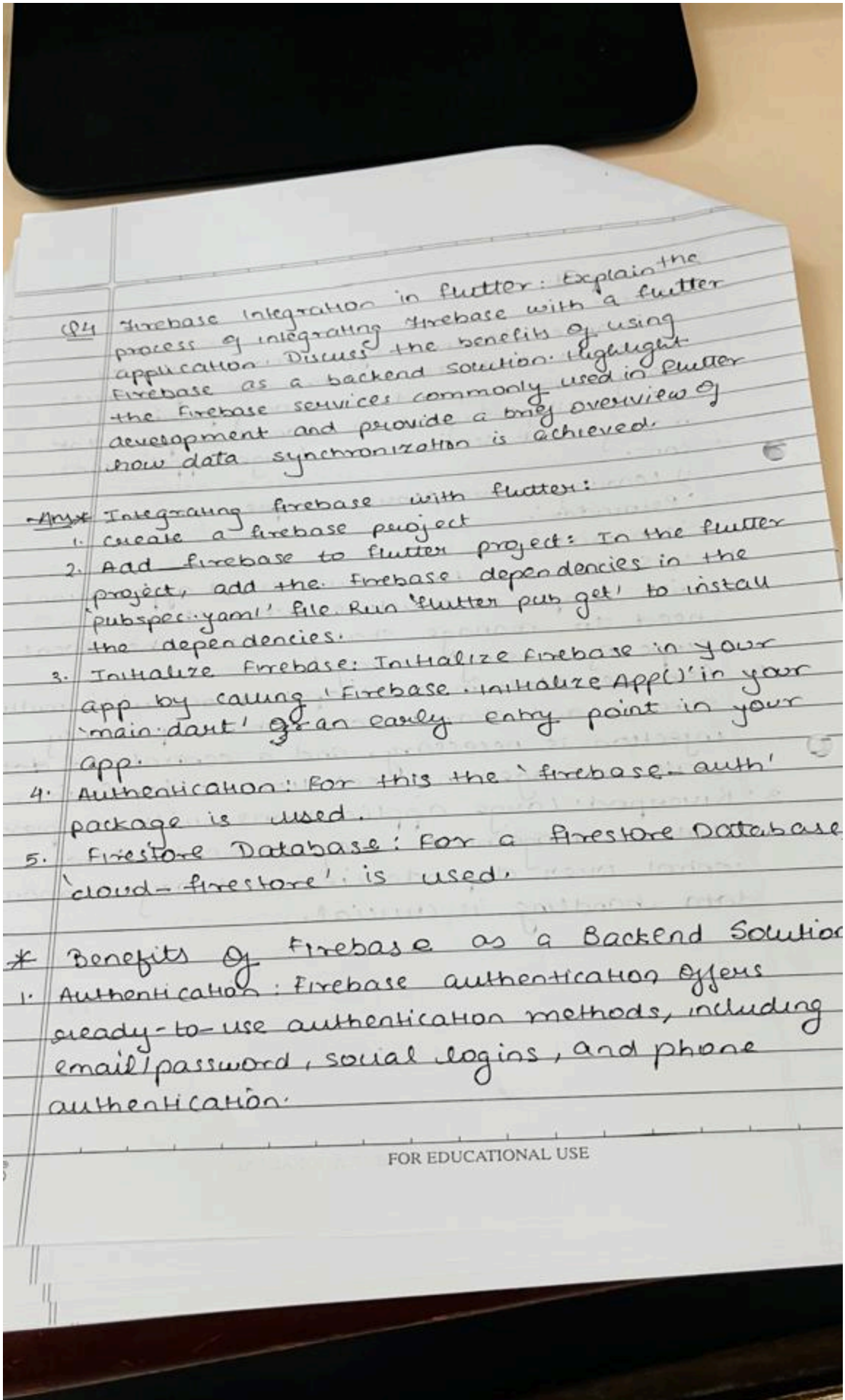
1) Learning curve: More complex than basic 'Provider'.

Scenarios:

1. setState: Small to medium-sized applications with simple UI structure. When there is a need to manage state within the local scope of a single widget.

2. Provider: Medium-sized applications with multiple screens and shared data. When dependency injection is necessary, and a centralized state management solution is beneficial.

3. Riverpod: Large applications with complex state management requirements. When control over dependencies and asynchronous data handling is crucial.



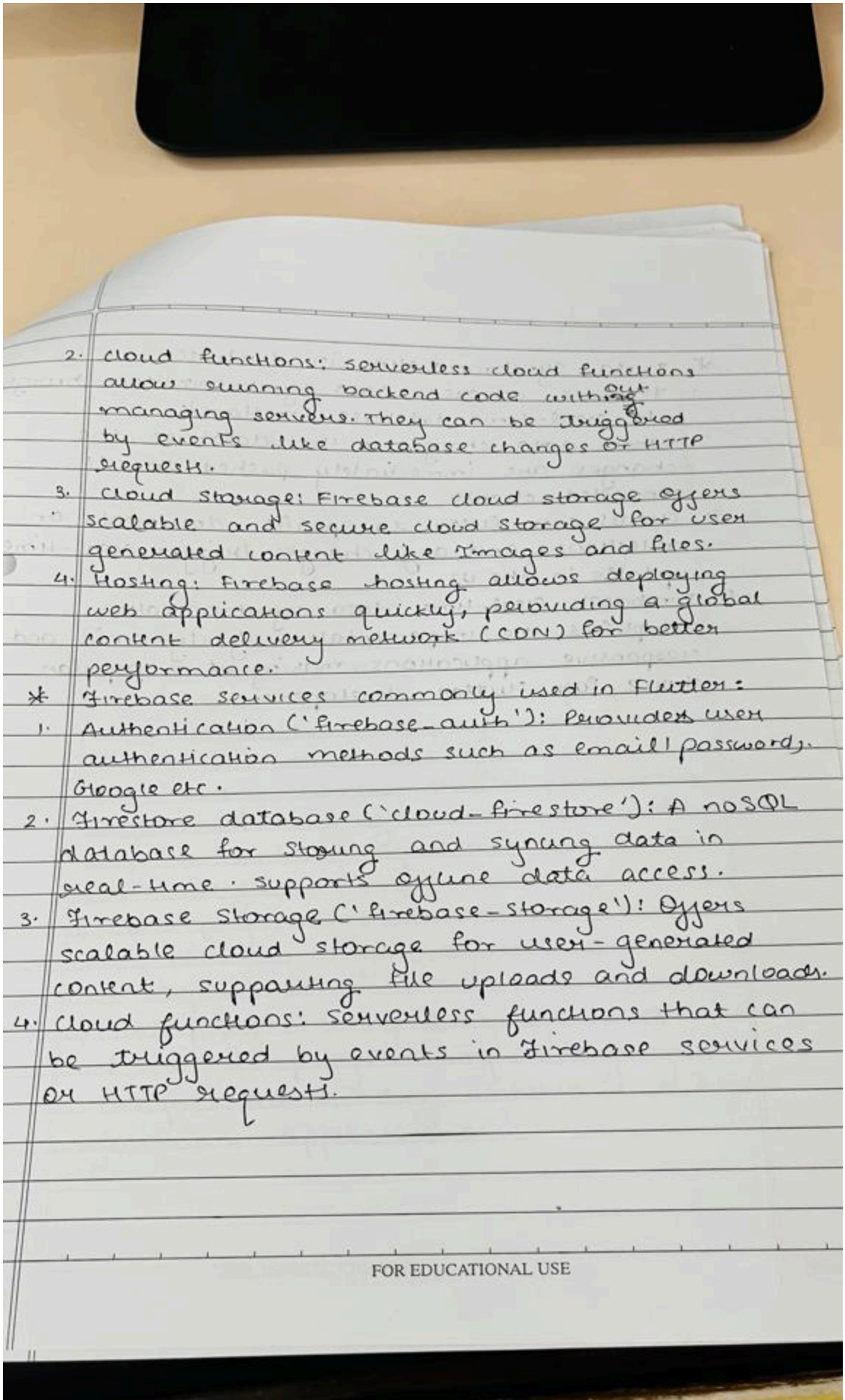
Q4 Firebase Integration in flutter: Explain the process of integrating Firebase with a flutter application. Discuss the benefits of using Firebase as a backend solution. Highlight the Firebase services commonly used in flutter development and provide a brief overview of how data synchronization is achieved.

Ans* Integrating firebase with flutter:

1. Create a firebase project
2. Add firebase to flutter project: In the flutter project, add the firebase dependencies in the 'pubspec.yaml' file. Run 'flutter pub get' to install the dependencies.
3. Initialize Firebase: Initialize Firebase in your app by calling 'Firebase.initializeApp()' in your 'main.dart' or an early entry point in your app.
4. Authentication: For this the 'firebase_auth' package is used.
5. Firestore Database: For a firestore Database 'cloud_firestore' is used.

* Benefits of Firebase as a Backend Solution

1. Authentication: Firebase authentication offers ready-to-use authentication methods, including email/password, social logins, and phone authentication.



2. Cloud functions: Serverless cloud functions allow running backend code without managing servers. They can be triggered by events like database changes or HTTP requests.
3. Cloud storage: Firebase cloud storage offers scalable and secure cloud storage for user generated content like images and files.
4. Hosting: Firebase hosting allows deploying web applications quickly, providing a global content delivery network (CDN) for better performance.

* Firebase services commonly used in Flutter:

1. Authentication ('firebase_auth'): Provides user authentication methods such as email/password, Google etc.
2. Firestore database ('cloud_firestore'): A noSQL database for storing and syncing data in real-time. Supports offline data access.
3. Firebase Storage ('firebase_storage'): Offers scalable cloud storage for user-generated content, supporting file uploads and downloads.
4. Cloud functions: Serverless functions that can be triggered by events in Firebase services or HTTP requests.

* Data synchronization in Firebase:
1. Firebase achieves data synchronization through its real-time database (Firestore).
2. When data changes in the database, the changes are immediately pushed to all connected clients.
3. Clients can listen to specific documents or collections, and any changes trigger real-time updates in the UI.
4. Firebase's real-time data synchronization simplifies the implementation of dynamic and responsive applications, making it a popular choice for Flutter developers.

MAD & PWA Lab
Journal

Experiment No.	Assignment-2
Assignment 2 Questions	1. Define Progressive Web App (PWA) and explain its significance in modern web development.Discuss the key characteristics that differentiate PWAs from traditional mobile apps 2. Define responsive web design and explain its importance in the context of Progressive Web Apps.Compare and contrast responsive, fluid, and adaptive web design approaches. 3. Describe the lifecycle of Service Workers, including registration, installation, and activation phases. 4. Explain the use of IndexedDB in the Service Worker for data storage.
Roll No.	18
Name	Moksha Gawada
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO4:Understand various PWA frameworks and their requirements LO5: Design and Develop a responsive User Interface by applying PWA Design techniques LO6:Develop and Analyze PWA Features and deploy it over app hosting solutions
Grade:	4

MOKSHA GAWADA
D15A-18

/

PWA Assignment - 02

Q1 Define Progressive web App (PWA) and explain its significance in modern web development. Discuss the key characteristics that differentiate PWAs from traditional mobile apps.

Ans A Progressive web App (PWA) is a type of web application that leverages modern web technologies to provide users with a native app-like experience directly through a web browser. PWA's are designed to be fast, reliable and engaging, offering features such as offline functionality, push notifications, and access to device hardware like camera and geolocation.

Following are the key characteristics

1. Cross-platform compatibility:- PWAs are built using standard web technologies (HTML, CSS, and JavaScript) making them compatible with various devices and platforms, including desktops, smartphones, and tablets. In contrast, traditional mobile apps often require separate development efforts for different platforms (e.g. iOS & android.)

2. Offline functionality: PWAs can work offline or with a limited internet connection, thanks to features like service workers, which cache resources and enable offline access to previously visited content. This capability allows users to continue using the app even when they're offline, enhancing user experience and accessibility.
3. Responsive Design: PWAs are designed to be responsive, adapting seamlessly to different screen sizes and orientations. This ensures a consistent user experience across various devices, whether users access the apps on a desktop computer, smartphone or tablet.
4. Fast Performance:- PWAs are optimized for speed and performance, allowing for quick loading times and smooth interactions. By minimizing network requests and efficiently caching resources, PWAs deliver a fast and responsive user experience, similar to native apps.
5. Engagement features: PWAs can engage ~~features~~ users through features like push notifications, which enable real-time communication and re-engage users even when the app is not actively open. this helps increase user retention and keeps users informed about new content or updates.

6. Installation and Discoverability: PWAs can be installed directly from the web browser without the need for app distribution. This simplifies the installation process for users and improves app discoverability.
7. Security: PWAs are served over HTTPS to ensure data security and integrity, protecting user information from unauthorized access or tampering. This commitment to security helps build trust with users and reinforces the reliability of PWAs as a viable alternative to native apps.

Q2 Define responsive web design and explain its importance in the context of Progressive Web App. Compare and contrast responsive, fluid and adaptive web design approaches.

Ans 1. Responsive web design is an approach to web development that aims to create websites that provide an optimal viewing and interaction experience across a wide range of devices and screen sizes.
2. This is achieved by using flexible layouts, fluid grids, and media queries to adapt the design and content of the website based on the characteristics of the device and browser being used.

3. The importance of responsive web design in the context of Progressive Web Apps (PWAs) lies in its ability to ensure that PWAs are accessible and usable on various devices, including desktop computers, smartphones, tablets, and other mobile devices.
4. By implementing responsive design principles, PWAs can dynamically adjust their layout, typography, and content to provide a consistent and user-friendly experience across different screen sizes and resolutions.
5. This helps enhance user engagement, improve accessibility, and increases the reach of PWAs to a broader audience.

Responsive web design	Fluid Web design	Adaptive web design
1. Adapts the layout and content dynamically based on the viewport size using flexible grids, fluid layouts, and media queries.	1. Utilizes flexible layouts and elements that scale smoothly with the viewport size, maintaining proportion and readability across different screen sizes.	1. Creates multiple versions of the website each optimized for specific devices or screen sizes, and serves the appropriate version based on device characteristics
Ensures PWAs are accessible and usable across various devices, improving user experience and engagement.	2. Contributes to a consistent and visually appealing user experience by ensuring that content flows and adjusts smoothly across different devices.	2. Allows for greater control over the user experience on different devices by tailoring the design and content to specific device types or screen sizes

Responsive Web design	Fluid Web design	Adaptive web design.
9. Utilizes flexible grids, fluid layouts, and media queries to adjust styling based on view-port size.	3. Uses flexible widths, percentages for layout and element sizes, and scalable images and media to ensure smooth scalability.	3. Involves creating multiple fixed-width layouts, server-side or client-side detection methods, and targeted optimizations for specific devices or screen sizes.
4. One design adapts to various viewport sizes through CSS media queries and flexible layouts.	4. Content and layout elements smoothly scale with the viewport size, maintaining proportions and readability.	4. Multiple versions of the website are created, each targeting specific devices or screen sizes and served accordingly
. Adapts layout and content based on view-port size, ensuring a consistent user experience across devices.	5. Maintains readability and usability by scaling elements smoothly with the viewport size.	5. Offers tailored experience for different devices or screen size, potentially providi a more optimized experience.

Q3 Describe the lyecycle of Service workers, including registration, installation, and activation phases.

Ans The lyecycle of Service workers involves three main phases: registration, installation and activation. Service workers are type of JavaScript code that runs in the background of a web application, enabling features like offline caching, push notifications, and background sync.

1. Registration :-

→ Definition: Registration is the process of telling the browser to start installing a Service worker for a specific URL scope.

→ Service worker registration typically occurs in the main JavaScript file of a web application. Developers use the `navigator.serviceWorker.register()` method to register the service worker file.

2. Installation:

→ Installation occurs when the browser fetches and installs the service worker script for the first time.

→ Installation is triggered immediately after registration if the service worker file is new or if there have been changes to the existing file.
→ During installation, the 'install' event is fired in the service worker script.

3. Activation:

→ Activation occurs when the installed service worker becomes active and starts controlling pages within its ~~so~~ scope.
→ After installation is complete, the service worker enters the 'waiting' state until all tabs using the old version of the service worker are closed
→ Once activated, the 'activate' event is fired. Developers can use the 'activate' event to clean up old ~~the~~ caches, update databases, or perform other tasks related to the activation of the service worker.

Q4. Explain the use of Indexed DB in the Service Worker for data storage.

Ans 1. Indexed DB is a client-side storage mechanism provided by modern web browsers, allowing web applications to store large amounts of structured data persistently.

2. In the context of Service Workers, Indexed DB is often used to cache data for offline access, manage background synchronization, and store other application data needed for offline functionality.

Following are the steps of how Indexed DB is used in the Service Worker for data storage:-

i. Offline Caching: i) Service Workers can intercept network requests and cache responses using Indexed DB.

ii) When a web application makes a request for a specific resource (eg. HTML, CSS, JS, images), the Service Worker can store the response data in Indexed DB for future offline access.

iii) This enables the web application to continue functioning even when the device is offline, serving cached resources from Indexed DB instead of making network requests.

2. Background sync:- i) Indexed DB can be used to store data temporarily when background sync events occur.
   ii) For example, if a web application needs to synchronize data with a server while the device is offline, the Service Workers can queue the data in IndexedDB until a network connection becomes available.
   iii) Once the connection is restored, the service worker can retrieve the queued data from IndexedDB and synchronize it with the server.
3. Data Storage:- i) Service workers can utilize IndexedDB to store application data needed for offline functionality or other purposes.
   ii) This data can include user preferences, or any other structured data required by the web application.
4. IndexedDB API: i) Service workers interact with IndexedDB using the IndexedDB API, which provides methods for opening databases, creating object stores, storing and retrieving data, and handling transactions.
   ii) Service workers can use asynchronous IndexedDB operations to perform database operations without blocking the main thread, ensuring smooth performance and responsiveness.