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 **Hertika Batra** 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

Project Title: Goggle-drive clone/ PWA:fuzzy **Roll No.5**

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

Faculty Incharge : Mrs. Kajal Joseph.

Lab Teachers **:** Mrs. Kajal Joseph

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: Goggle-drive clone/ PWA:fuzzy **Roll No.5**

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	15
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

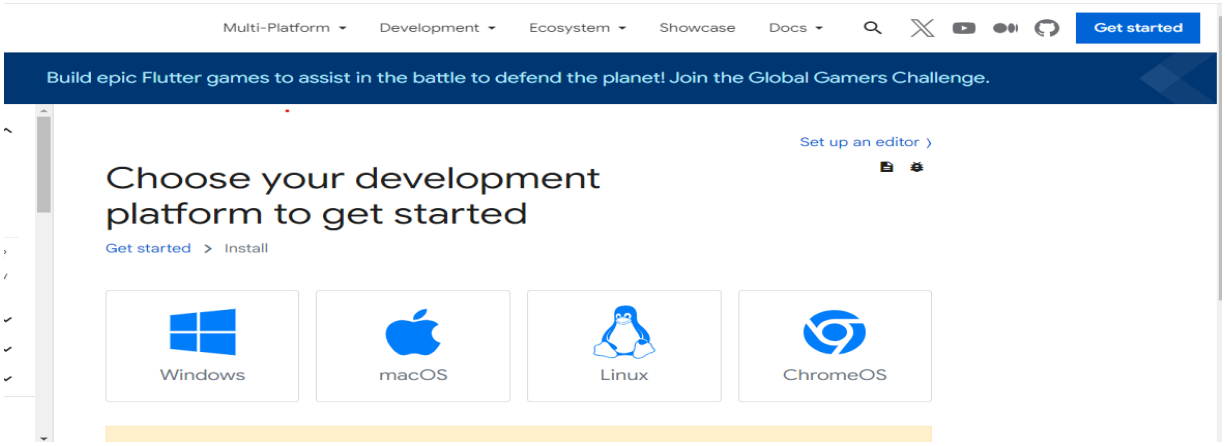
MAD & PWA Lab

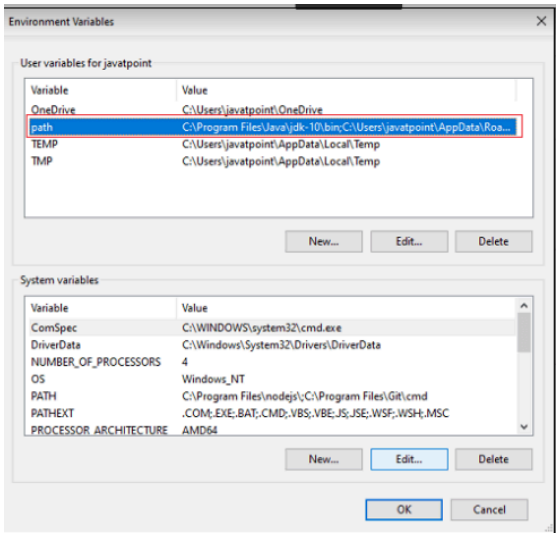
Journal

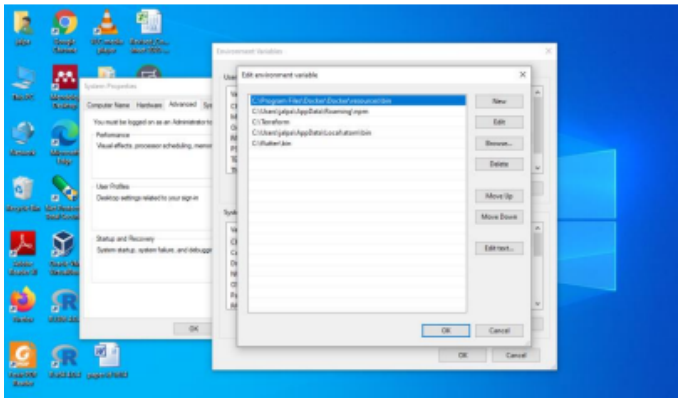
Experiment No.	01
Experiment Title.	To install and configure the Flutter Environment
Roll No.	5
Name	Hertika Batra
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO1: Understand cross platform mobile application development using Flutter framework
Grade:	15

Name :Hertika Batra
Roll:5
Batch : A

EXP 1: Installation and Configuration of Flutter Environment.





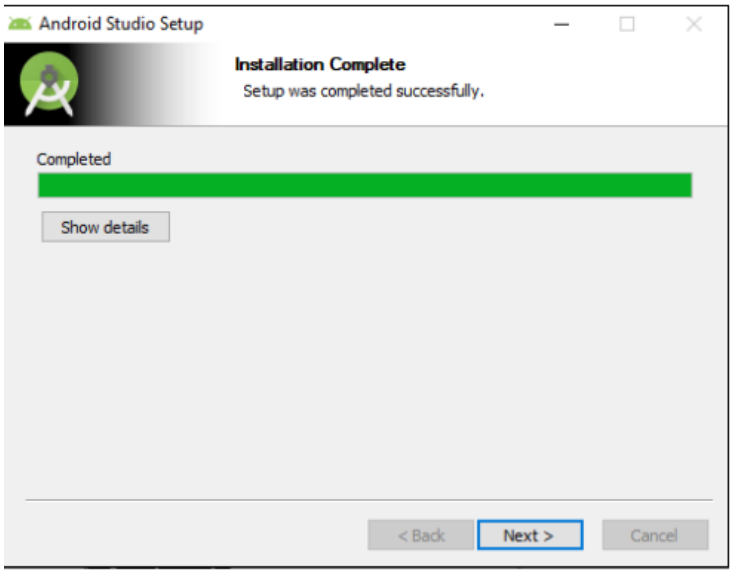


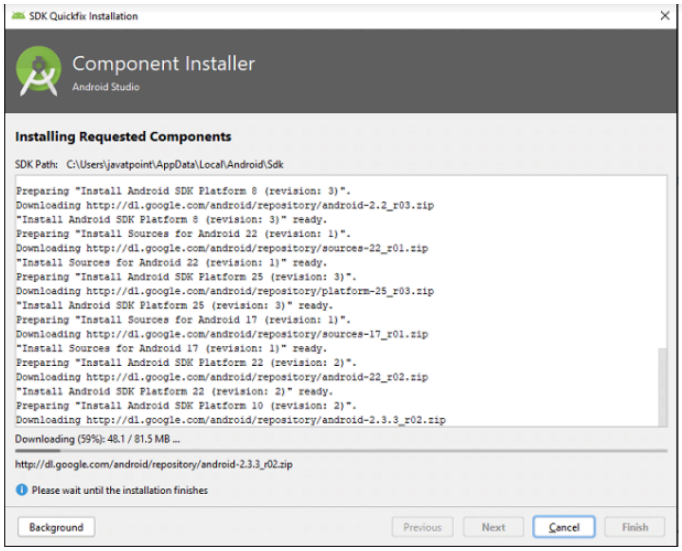
```
Microsoft Windows [Version 10.0.19042.1415]
(c) Microsoft Corporation. All rights reserved.

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.
```

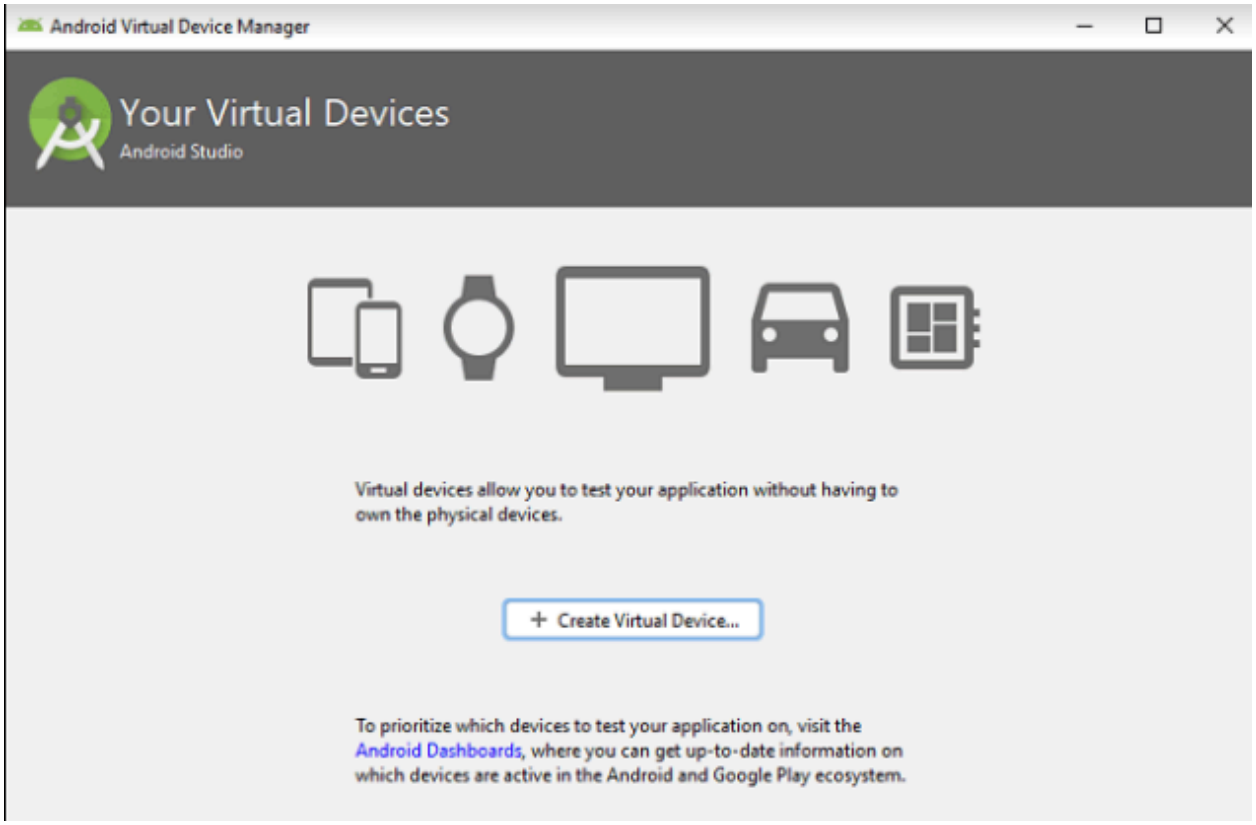
Select Command Prompt

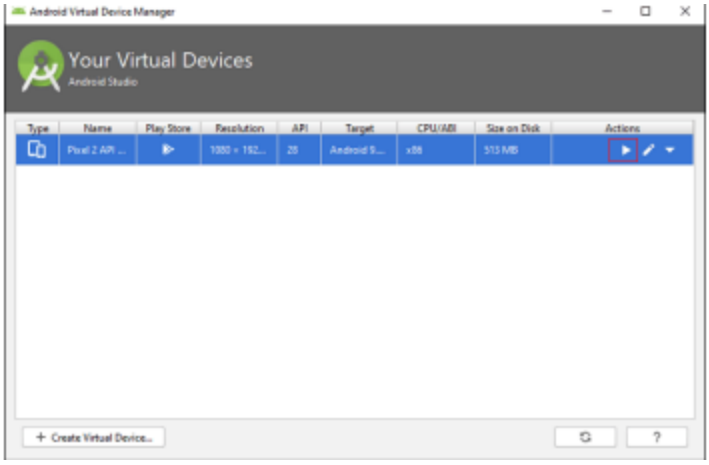
```
  See Google's privacy policy:
  https://policies.google.com/privacy



:\Users\jalpa>
:\Users\jalpa>
:\Users\jalpa>flutter doctor
unning "flutter pub get" in flutter_tools...                    17.0s
octor 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)
X] Android toolchain - develop for Android devices
    X 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.

:\Users\jalpa>flutter doctor
octor 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)
    X cmdline-tools component is missing
      Run `path/to/sdkmanager --install "cmdline-tools;latest"`
      See https://developer.android.com/studio/command-line for more details.
    X 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.
```

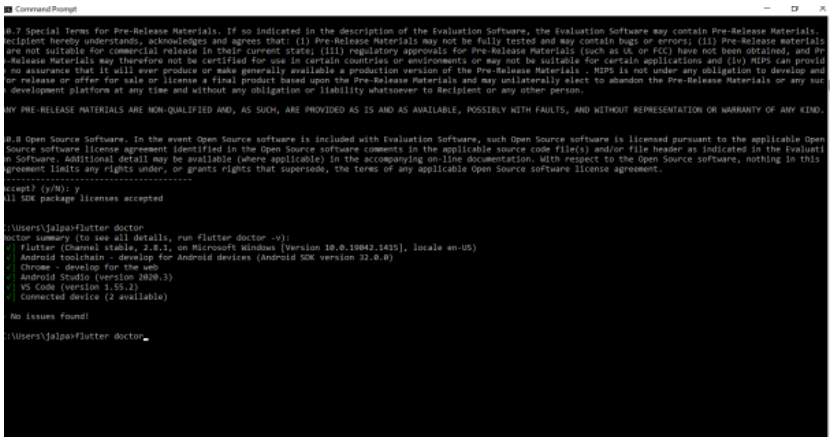

Android Studio Setup
Welcome to Android Studio Setup
Setup will guide you through the installation of Android Studio.
It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.
Click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

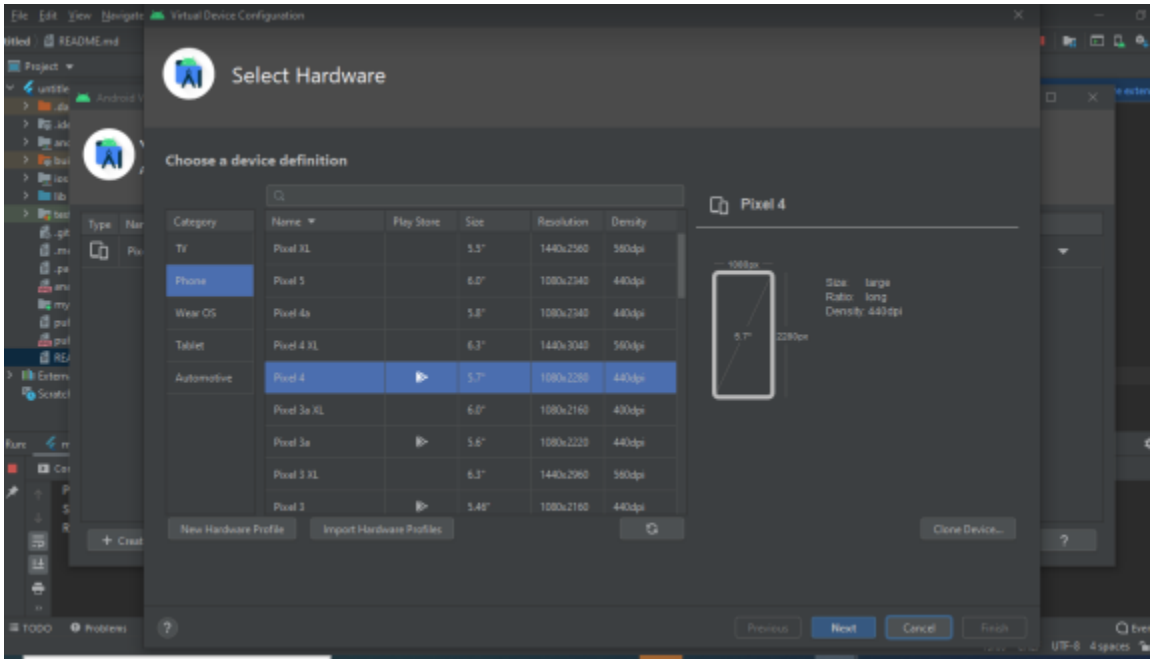

Android Virtual Device Manager
Your Virtual Devices
Android Studio
Virtual devices allow you to test your application without having to own the physical devices.
+ Create Virtual Device...
To prioritize which devices to test your application on, visit the Android Dashboards, where you can get up-to-date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

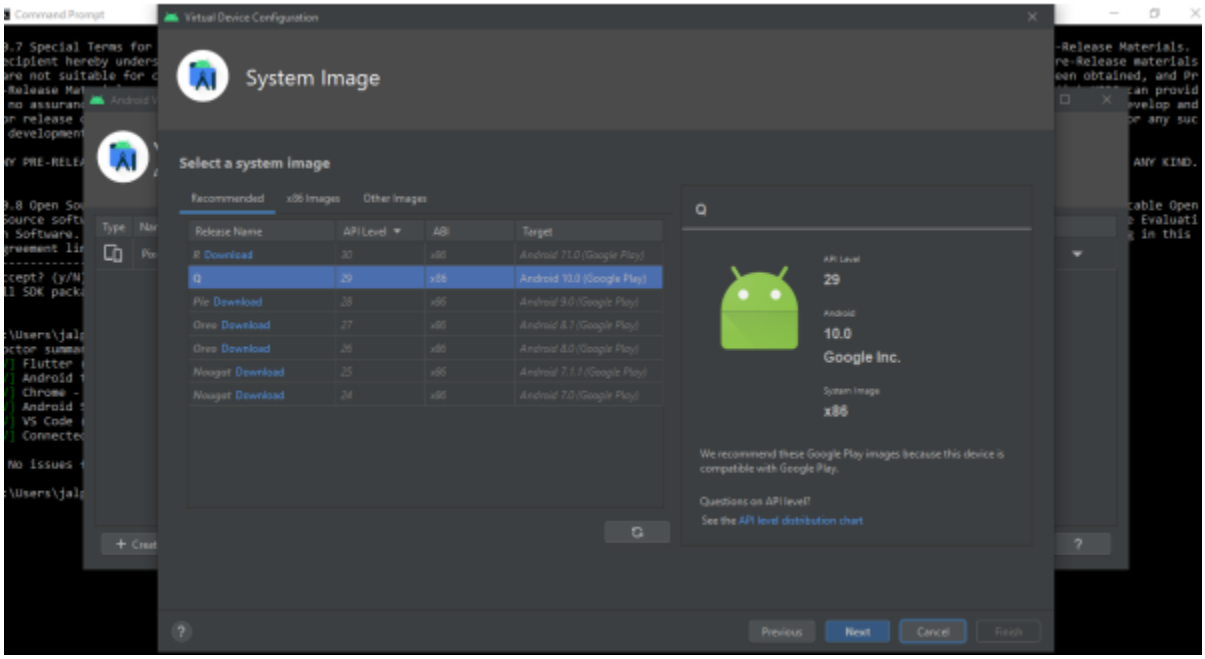

Android Virtual Device Manager
Your Virtual Devices
Android Studio
Type
Name
Play Store
Resolution
API
Target
CPU/ABI
Size on Disk
Actions
Pixel 2 API ...
1080 × 192...
28
Android 9...
x86
513 MB
Create Virtual Device...

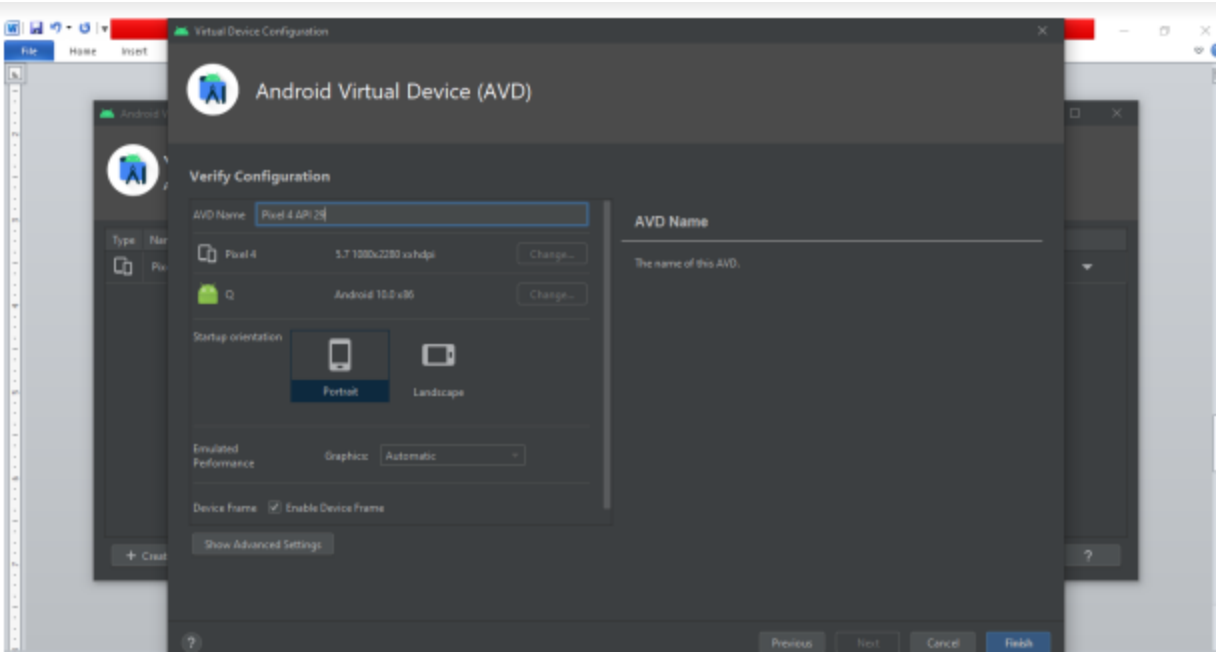

Command Prompt
0.7 Special Terms for Pre-Release Materials. If so indicated in the description of the Evaluation Software, the Evaluation Software may contain Pre-Release Materials.
ecipient hereby understands, acknowledges and agrees that: (i) Pre-Release Materials may not be fully tested and may contain bugs or errors; (ii) Pre-Release materials
are not suitable for commercial release in their current state; (iii) regulatory approvals for Pre-Release Materials (such as UL or FCC) have not been obtained, and Pr
-Release Materials may therefore not be certified for use in certain countries or environments or may not be suitable for certain applications and (iv) MIPS can provid
no assurance that it will ever produce or make generally available a production version of the Pre-Release Materials . MIPS is not under any obligation to develop and
or release or offer for sale or license a final product based upon the Pre-Release Materials and may unilaterally elect to abandon the Pre-Release Materials or any suc
development platform at any time and without any obligation or liability whatsoever to Recipient or any other person.
NY PRE-RELEASE MATERIALS ARE NON-QUALIFIED AND, AS SUCH, ARE PROVIDED AS IS AND AS AVAILABLE, POSSIBLY WITH FAULTS, AND WITHOUT REPRESENTATION OR WARRANTY OF ANY KIND.
0.8 Open Source Software. In the event Open Source software is included with Evaluation Software, such Open Source software is licensed pursuant to the applicable Open
Source software license agreement identified in the Open Source software comments in the applicable source code file(s) and/or file header as indicated in the Evaluati
n Software. Additional detail may be available (where applicable) in the accompanying on-line documentation. With respect to the Open Source software, nothing in this
agreement limits any rights under, or grants rights that supersede, the terms of any applicable Open Source software license agreement.
ccept? (y/N): y
ll SDK package licenses accepted
:\Users\jalpa>flutter doctor
octor 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!
:\Users\jalpa>flutter doctor

Name :Hertika Batra
Roll:5
Batch : A

Experiment 2: Create a ‘Hello World App’ using Flutter



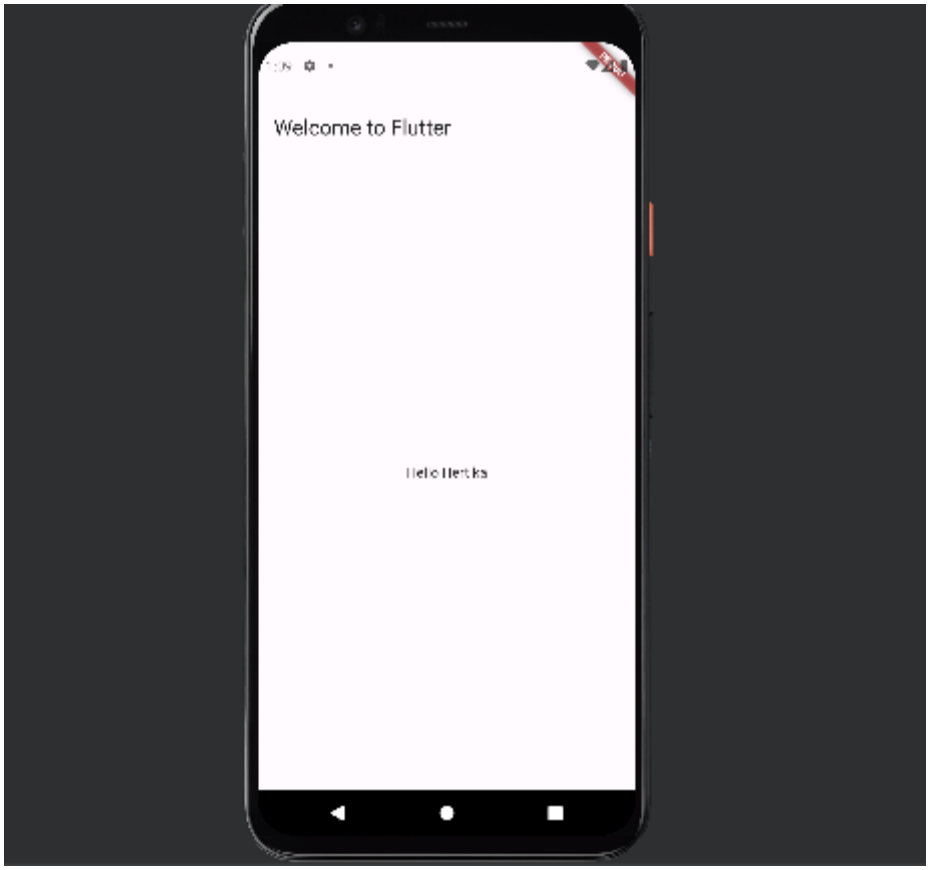




code:

```
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: const Center(
child: Text('Hello Hertika'),
),
),
);
}
}
```

output:-


Welcome to Flutter
Hello Nerdks


1:08
DEBUG
Welcome to Flutter
Hello Hertika

MAD & PWA Lab

Journal

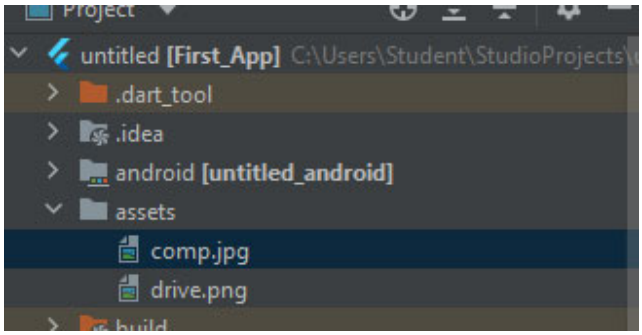
Experiment No.	02
Experiment Title.	To design Flutter UI by including common widgets.
Roll No.	5
Name	Hertika Batra
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	15

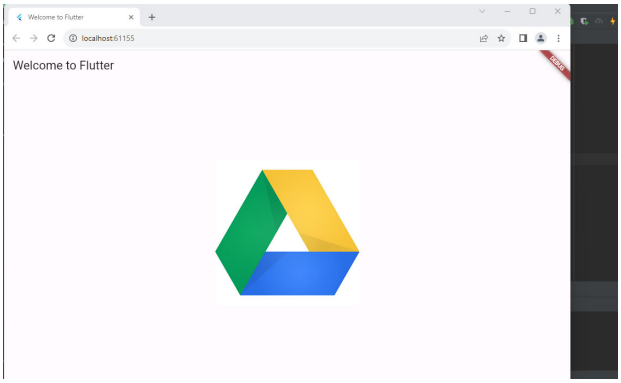
Name:-Hertika Batra
Roll:-5
Batch A

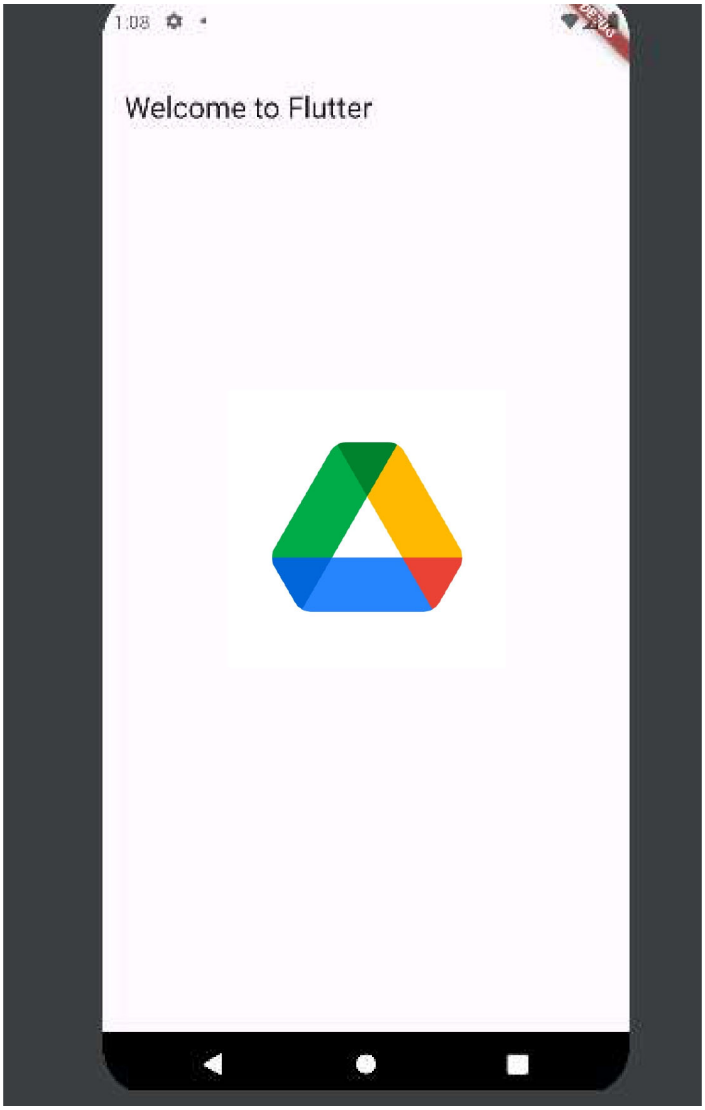
Experiment:2

main.dart

```
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/drive.png'),
      ),
    ),
  );
 }
}
```




Welcome to Flutter


Welcome to Flutter

MAD & PWA Lab

Journal

Experiment No.	03
Experiment Title.	To include icons, images, fonts in Flutter app
Roll No.	5
Name	Hertika Batra
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	15

Name :- Hertika Batra
Roll:-5
Batch:-A

Experiment:3

Aim:- To include Icon, Images and fonts in flutter App

Theory:-

In Flutter, icons, images, and fonts are like the ingredients in a recipe for making a beautiful and user-friendly app.

Icons are small pictures that show actions or things in the app, like a magnifying glass for search or a heart for liking something. They should be easy to understand and look nice.

Images are like the pictures you see in an app, such as photos or illustrations. They make the app more interesting and help to tell a story. Flutter helps to handle images well, so they look good on different devices.

Fonts are the different styles of writing you see in the app, like the size and type of letters. Picking the right font makes the app easier to read and gives it a special style. Flutter lets developers use different fonts and styles to make the app look unique and professional.

So, in Flutter, choosing the right icons, images, and fonts is important for making an app that people enjoy using and looks great on their phones or tablets.

Code:-

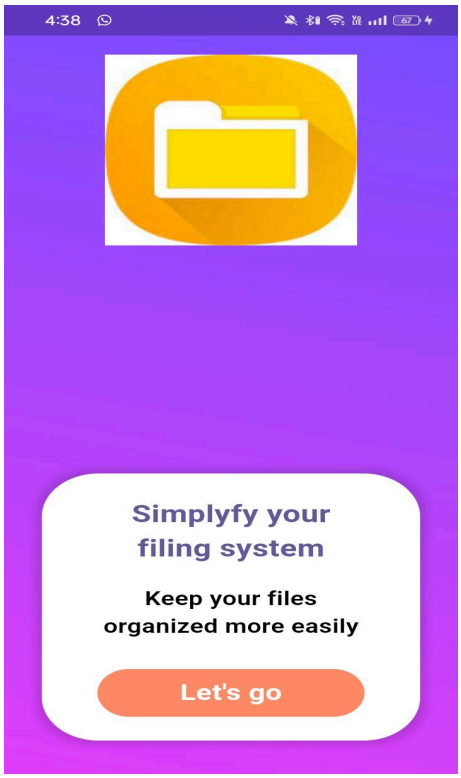
```
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:driveclone/controllers/authentication_controller.dart';
import 'package:driveclone/utils.dart';

class SigninScreen extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return Container(
   decoration: const BoxDecoration(
    gradient: LinearGradient(
     begin: Alignment.topCenter,
     end: Alignment.bottomLeft,
     colors: [
      Colors.deepPurpleAccent,
      Colors.purpleAccent,
     ],
    ),
   ),
   child: Scaffold(
    backgroundColor: Colors.transparent,
    body: Column(
     children: [
      Padding(
       padding: EdgeInsets.only(
        top: MediaQuery.of(context).viewInsets.top + 52,
       ),
       child: const Image(
```

```
    width: 200,
    height: 200,
    image: AssetImage('assets/filemanager.jpg'),
    fit: BoxFit.cover,
   ),
),
const Spacer(),
Container(
  width: MediaQuery.of(context).size.width,
  margin: const EdgeInsets.only(left: 30, right: 30, bottom: 35),
  decoration: BoxDecoration(
    borderRadius: BorderRadius.circular(45),
    color: Colors.white,
    boxShadow: const [
      BoxShadow(
        color: Colors.black12,
        spreadRadius: 5,
        blurRadius: 5,
      ),
    ],
  ),
  child: Padding(
    padding: const EdgeInsets.only(top: 25, bottom: 25),
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Text("Simplyfy your",
            style: textStyle(25, const Color(0xff635c9B), FontWeight.w700)),
        Text("filing system",
            style: textStyle(25, const Color(0xff635c9B), FontWeight.w700)),
```

```
            const SizedBox(height: 20),
            Text("Keep your files", style: textStyle(20, Colors.black, FontWeight.w600)),
            Text("organized more easily",
              style: textStyle(20, Colors.black, FontWeight.w600)),
            const SizedBox(height: 30),
            InkWell(
             onTap: () => Get.find<AuthController>().signin(),
             child: Container(
              width: MediaQuery.of(context).size.width / 1.7,
              height: 50,
              decoration: BoxDecoration(
               borderRadius: BorderRadius.circular(30),
               color: Colors.deepOrangeAccent.withOpacity(0.8),
              ),
              child: Center(
                child: Text("Let's go",
                  style: textStyle(23, Colors.white, FontWeight.w700))),
             ),
            ),
           ],
          ),
         ),
        ),
       ],
      ),
     ),
    );
}

TextStyle textStyle(double size, Color color, FontWeight weight) {
```

```
    return TextStyle(
      fontSize: size,
      color: color,
      fontWeight: weight,
    );
  }
}

// void main() {
//   runApp(MaterialApp(
//     home: SigninScreen(),
//   ));
// }
```

Output:-



MAD & PWA Lab

Journal

Experiment No.	04
Experiment Title.	To create an interactive Form using form widget
Roll No.	5
Name	Hertika Batra
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	15

Name :- Hertika Batra
Roll:-5
Batch:-A

Experiment:4

Aim:- To include Icon, Images and fonts in flutter App

Theory:-

In Flutter, icons, images, and fonts are like the ingredients in a recipe for making a beautiful and user-friendly app.

Icons are small pictures that show actions or things in the app, like a magnifying glass for search or a heart for liking something. They should be easy to understand and look nice.

Images are like the pictures you see in an app, such as photos or illustrations. They make the app more interesting and help to tell a story. Flutter helps to handle images well, so they look good on different devices.

Fonts are the different styles of writing you see in the app, like the size and type of letters. Picking the right font makes the app easier to read and gives it a special style. Flutter lets developers use different fonts and styles to make the app look unique and professional.

So, in Flutter, choosing the right icons, images, and fonts is important for making an app that people enjoy using and looks great on their phones or tablets.

Code:-

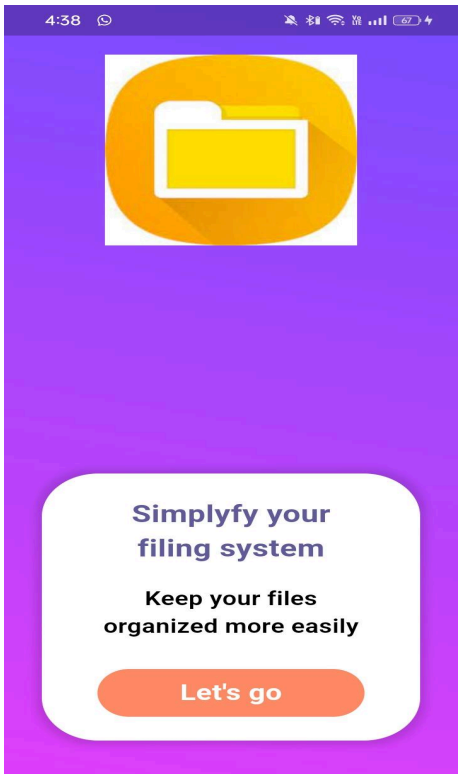
```
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:driveclone/controllers/authentication_controller.dart';
import 'package:driveclone/utils.dart';

class SigninScreen extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return Container(
   decoration: const BoxDecoration(
    gradient: LinearGradient(
     begin: Alignment.topCenter,
     end: Alignment.bottomLeft,
     colors: [
      Colors.deepPurpleAccent,
      Colors.purpleAccent,
     ],
    ),
   ),
   child: Scaffold(
    backgroundColor: Colors.transparent,
    body: Column(
     children: [
      Padding(
       padding: EdgeInsets.only(
        top: MediaQuery.of(context).viewInsets.top + 52,
       ),
       child: const Image(
```

```
    width: 200,
    height: 200,
    image: AssetImage('assets/filemanager.jpg'),
    fit: BoxFit.cover,
   ),
),
const Spacer(),
Container(
  width: MediaQuery.of(context).size.width,
  margin: const EdgeInsets.only(left: 30, right: 30, bottom: 35),
  decoration: BoxDecoration(
    borderRadius: BorderRadius.circular(45),
    color: Colors.white,
    boxShadow: const [
      BoxShadow(
        color: Colors.black12,
        spreadRadius: 5,
        blurRadius: 5,
      ),
    ],
  ),
  child: Padding(
    padding: const EdgeInsets.only(top: 25, bottom: 25),
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Text("Simplyfy your",
            style: textStyle(25, const Color(0xff635c9B), FontWeight.w700)),
        Text("filing system",
            style: textStyle(25, const Color(0xff635c9B), FontWeight.w700)),
```

```
            const SizedBox(height: 20),
            Text("Keep your files", style: textStyle(20, Colors.black, FontWeight.w600)),
            Text("organized more easily",
              style: textStyle(20, Colors.black, FontWeight.w600)),
            const SizedBox(height: 30),
            InkWell(
             onTap: () => Get.find<AuthController>().signin(),
             child: Container(
              width: MediaQuery.of(context).size.width / 1.7,
              height: 50,
              decoration: BoxDecoration(
               borderRadius: BorderRadius.circular(30),
               color: Colors.deepOrangeAccent.withOpacity(0.8),
              ),
              child: Center(
                child: Text("Let's go",
                  style: textStyle(23, Colors.white, FontWeight.w700))),
             ),
            ),
           ],
          ),
         ),
        ),
       ],
      ),
     ),
    );
}

TextStyle textStyle(double size, Color color, FontWeight weight) {
```

```
    return TextStyle(
      fontSize: size,
      color: color,
      fontWeight: weight,
    );
  }
}

// void main() {
//   runApp(MaterialApp(
//     home: SigninScreen(),
//   ));
// }
```

Output:-



MAD & PWA Lab

Journal

Experiment No.	05
Experiment Title.	To apply navigation, routing and gestures in Flutter App
Roll No.	5
Name	Hertika Batra
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	15

Name :- Hertika Batra
Roll:-5
Batch:-A

Experiment:5

Aim:- To Apply Navigation ,Routing and gestures in Flutter App

Theory:-

In Flutter, navigation, routing, and gestures are like the directions and actions that help users explore and interact with an app easily. Navigation is about moving between different parts of the app, like switching from the homepage to a settings page. Routing is the behind-the-scenes logic that decides which page to show next when a user taps a button or performs an action. Flutter provides tools to manage this process smoothly, so users can navigate through the app without confusion. Gestures, on the other hand, are the touch-based movements users make on the screen, like swiping or tapping. Flutter gives developers ways to recognize these gestures and respond accordingly, making the app feel more interactive and fun to use. So, by using Flutter's navigation, routing, and gesture features, developers can create apps that are easy to explore and enjoyable to interact with.

Code:-

Recent files:-

```
import 'package:driveclone/controllers/files_screen_controller.dart';
import 'package:driveclone/utils.dart';
import 'package:flutter/material.dart';
import 'package:get/get_state_manager/src/rx_flutter/rx_getx_widget.dart';
import 'package:driveclone/widgets/storage_container.dart';

class RecentFiles extends StatelessWidget {
  @override
```

```
Widget build(BuildContext context) {
 return Column(
  children: [
   Align(
    alignment: Alignment.centerLeft,
    child: Text(
     "Recent Files",
     style: textStyle(20, textColor, FontWeight.bold),
    ),
   ),
   const SizedBox(height: 15),
   GetX<FilesScreenController>(
    builder: (FilesScreenController controller) {
     return Container(
      height: 100,
      child: ListView.builder(
       scrollDirection: Axis.horizontal,
       itemCount: controller.recentfileList.length,
       itemBuilder: (context, index) {
        return Padding(
         padding: const EdgeInsets.only(right: 13.0),
         child: Container(
          height: 65,
          child: Column(
           crossAxisAlignment: CrossAxisAlignment.start,
           children: [
            controller.recentfileList[index].fileType == "image"
              ? ClipRRect(
                borderRadius: BorderRadius.circular(18),
                child: Image(
```

```
          width: 65,
          height: 60,
          image: NetworkImage(
              controller.recentfileList[index].url),
          fit: BoxFit.cover,
        ),
      )
    : Container(
        width: 65,
        height: 60,
        decoration: BoxDecoration(
          border: Border.all(
              color: Colors.grey, width: 0.15),
          borderRadius: BorderRadius.circular(14)),
        child: Center(
         child: Image(
          width: 42,
          height: 42,
          image: AssetImage(
              'assets/${controller.recentfileList[index].fileExtension}.png'),
         ),
        ),
      ),
  const SizedBox(height: 5),
  Text(
   controller.recentfileList[index].name,
   style: textStyle(13, textColor, FontWeight.w500),
   overflow: TextOverflow.ellipsis,
  )
],
```

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

Folder Section:-

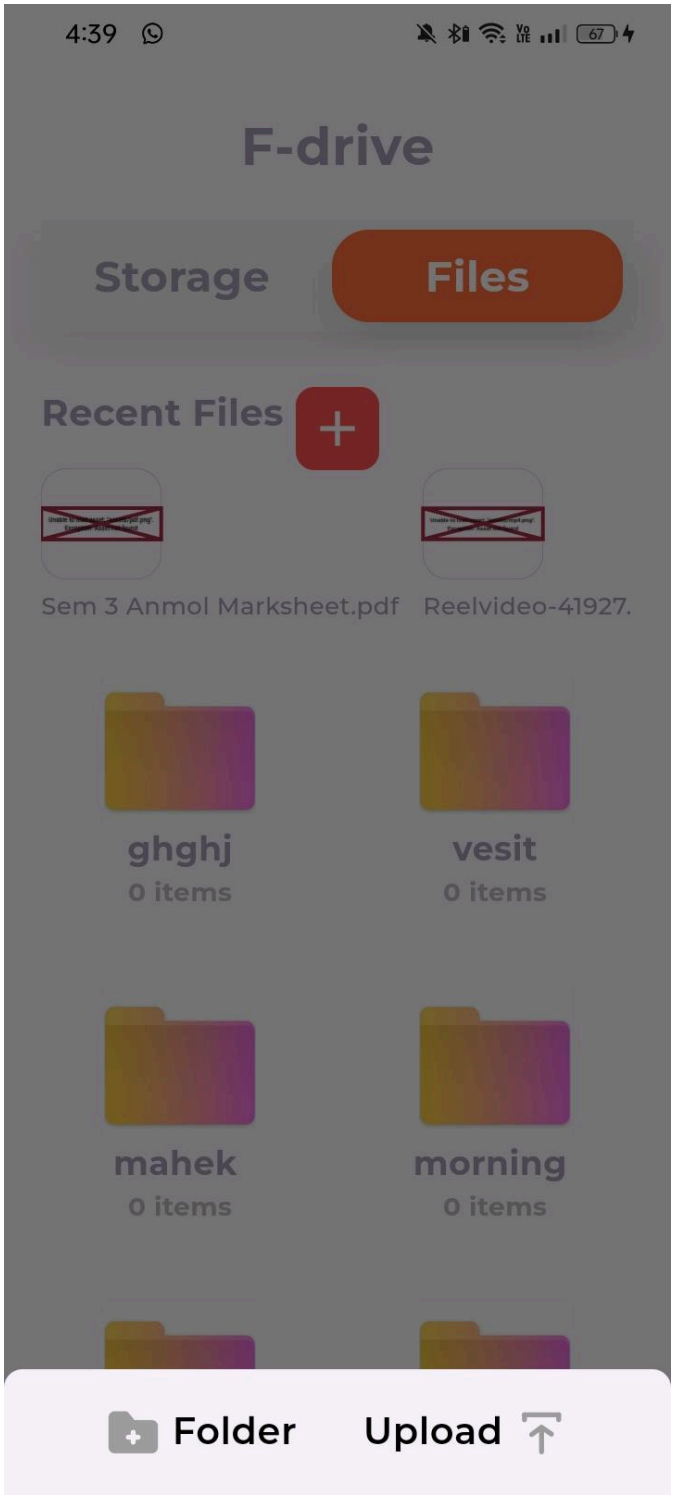
```
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:driveclone/screens/display_files_screen.dart';
import 'package:driveclone/utils.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:driveclone/controllers/files_screen_controller.dart';
// import 'package:driveclone/controllers/files_controller.dart;
import 'package:driveclone/widgets/storage_container.dart';

class FoldersSection extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return GetX<FilesScreenController>(
   builder: (FilesScreenController folderController) {
    if (folderController != null && folderController.foldersList != null) {
```

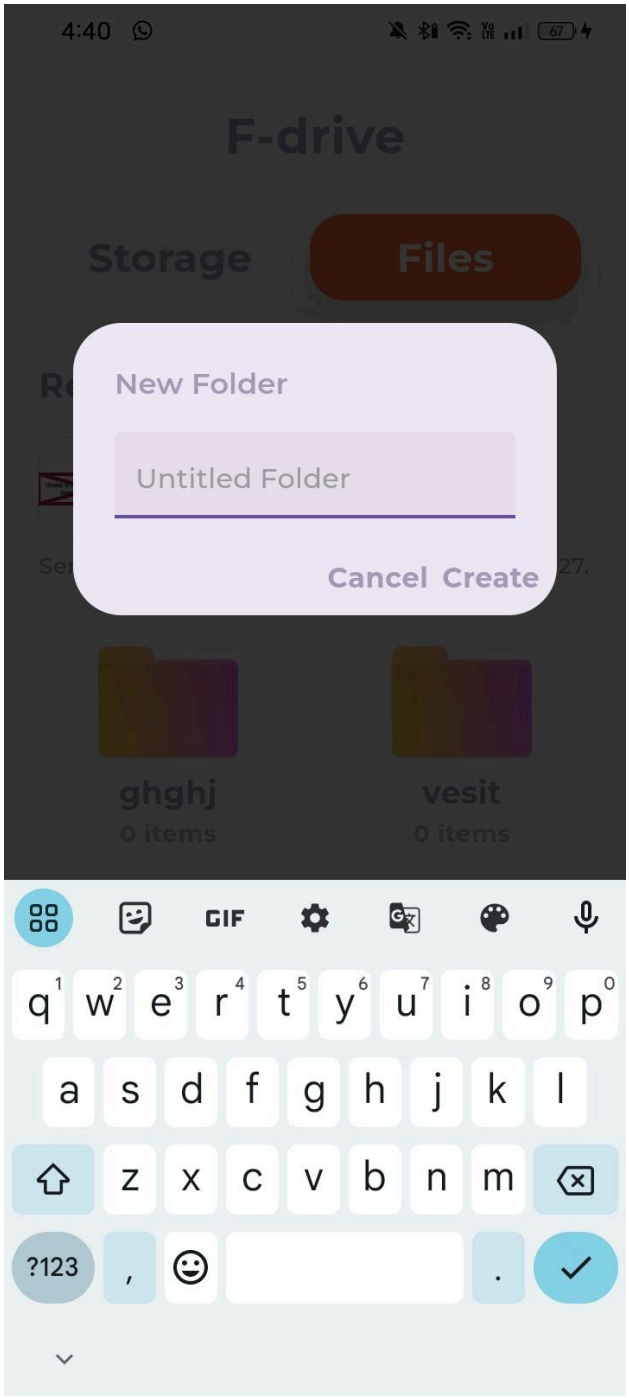
```
return GridView.builder(
 shrinkWrap: true,
 physics: const NeverScrollableScrollPhysics(),
 itemCount: folderController.foldersList.length,
 gridDelegate: const SliverGridDelegateWithFixedCrossAxisCount(
  crossAxisCount: 2,
  crossAxisSpacing: 20,
  mainAxisSpacing: 20,
 ),
 itemBuilder: (context, index) {
  return InkWell(
   onTap: () {
    if (FirebaseAuth.instance.currentUser != null) {
     Get.to(DisplayFilesScreen(
      folderController.foldersList[index].name, "folder"
     ));
    } else {
     // Handle case when user is not authenticated
     // You can show a snackbar or navigate to a login screen
     ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text('Please sign in to access folders'))
     );
    }
   },
   child: Container(
    decoration: BoxDecoration(
     borderRadius: BorderRadius.circular(10),
     boxShadow: [
      BoxShadow(
       color: Colors.grey.withOpacity(0.00001),
```

```
      offset: const Offset(10, 10),
      blurRadius: 5,
    )
  ],
),
child: Column(
  mainAxisAlignment: MainAxisAlignment.center,
  children: [
    Image.asset(
      'assets/folder.jpg',
      width: 82,
      height: 82,
      fit: BoxFit.cover,
    ),
    Text(
      folderController.foldersList[index].name,
      style: textStyle(18, textColor, FontWeight.bold),
    ),
    StreamBuilder<QuerySnapshot>(
      stream: FirebaseFirestore.instance
          .collection('users')
          .doc(FirebaseAuth.instance.currentUser!.uid)
          .collection('files')
          .where('folder', isEqualTo: folderController.foldersList[index].name)
          .snapshots(),
      builder: (context, snapshot) {
        if (snapshot.connectionState == ConnectionState.waiting) {
          return CircularProgressIndicator();
        }
        if (snapshot.hasError) {
```

```
                  return Text('Error: ${snapshot.error}');
                }
                return Text(
                 "${snapshot.data!.docs.length} items",
                 style: textStyle(14, Colors.grey[400]!, FontWeight.bold),
                );
               },
              )
             ],
            ),
           ),
          );
         },
        );
       } else {
        return Center(
         child: CircularProgressIndicator(),
        );
       }
      },
    );
  }
}
```

Output:-


4:39
F-drive
Storage
Files
Recent Files
Sem 3 Anmol Marksheet.pdf
Reelvideo-41927.
ghghj
0 items
vesit
0 items
mahek
0 items
morning
0 items
Folder
Upload



Conclusion:- In this experiment I learnt about navigation and routing in flutter and successfully implemented it in my project.

MAD & PWA Lab

Journal

Experiment No.	06
Experiment Title.	To Connect Flutter UI with fireBase database
Roll No.	5
Name	Hertika Batra
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

Name :- Hertika Batra
Roll:-5
Batch:- A

Experiment:-6

Aim:- To Connect UI with flutter with Firebase
Theory:-
Connecting Flutter UI with Firebase database involves integrating the frontend of your Flutter application with the backend database hosted on Firebase. Firebase provides a cloud-based NoSQL database solution that enables real-time data synchronization and secure storage. To connect the Flutter UI with Firebase, developers typically use the Firebase SDK for Flutter, which offers plugins and libraries to interact with Firebase services. This integration allows developers to perform operations like reading, writing, updating, and deleting data from the Firebase database directly from the Flutter application. By establishing this connection, developers can create dynamic and responsive apps that fetch and display real-time data from Firebase, providing users with a seamless and engaging experience while ensuring data integrity and security throughout the application lifecycle.
Code:-

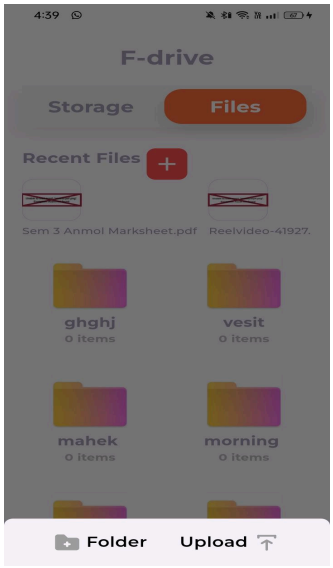
```
import 'dart:io';

import 'package:driveclone/utils.dart';
import 'package:file_picker/file_picker.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_storage/firebase_storage.dart';
import 'package:flutter_image_compress/flutter_image_compress.dart';
import 'package:get/get.dart';
import 'package:mime/mime.dart';
import 'package:path_provider/path_provider.dart';
import 'package:uuid/uuid.dart';
```

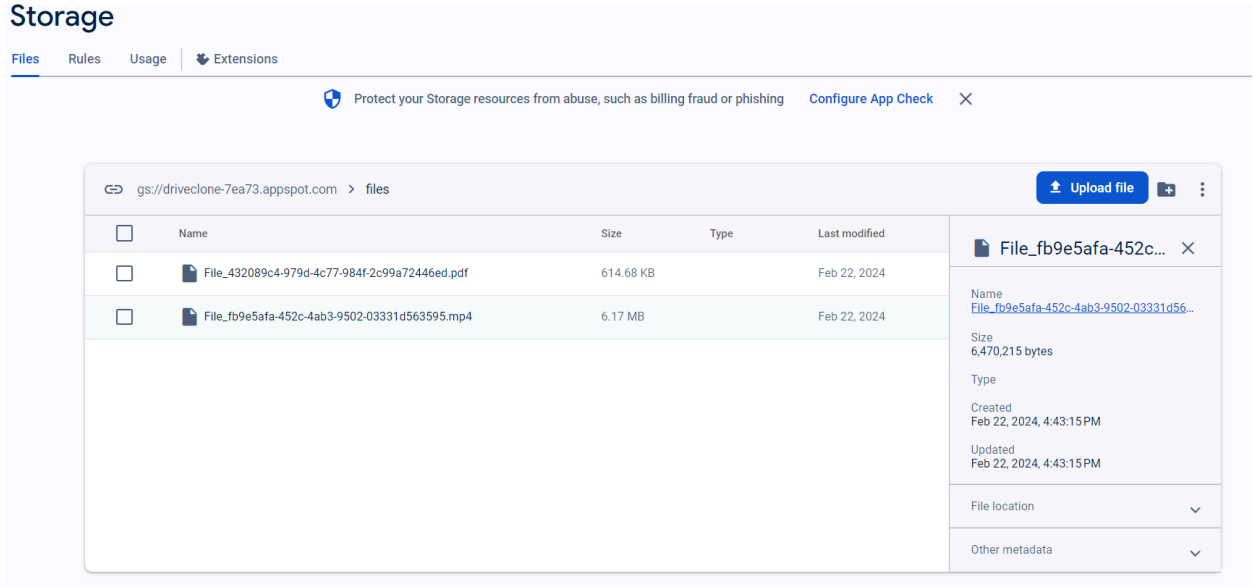
```
import 'package:video_compress/video_compress.dart';

class FirebaseService {
  Uuid uuid = Uuid();

  Future<Object?> compressFile(File file, String fileType) async {
    if (fileType.startsWith("image")) {
      Directory directory = await getTemporaryDirectory();
      String targetpath = directory.path + "/${uuid.v4()}.jpg";
      XFile? result = await FlutterImageCompress.compressAndGetFile(
        file.path,
        targetpath,
        quality: 75,
      );
      return result;
    } else if (fileType.startsWith("video")) {
      MediaInfo? info = await VideoCompress.compressVideo(
        file.path,
        quality: VideoQuality.MediumQuality,
        deleteOrigin: false,
        includeAudio: true,
      );
      if (info != null) {
        print(info.file);
        return File(info.path!);
      }
    }
    return file;
  }
```

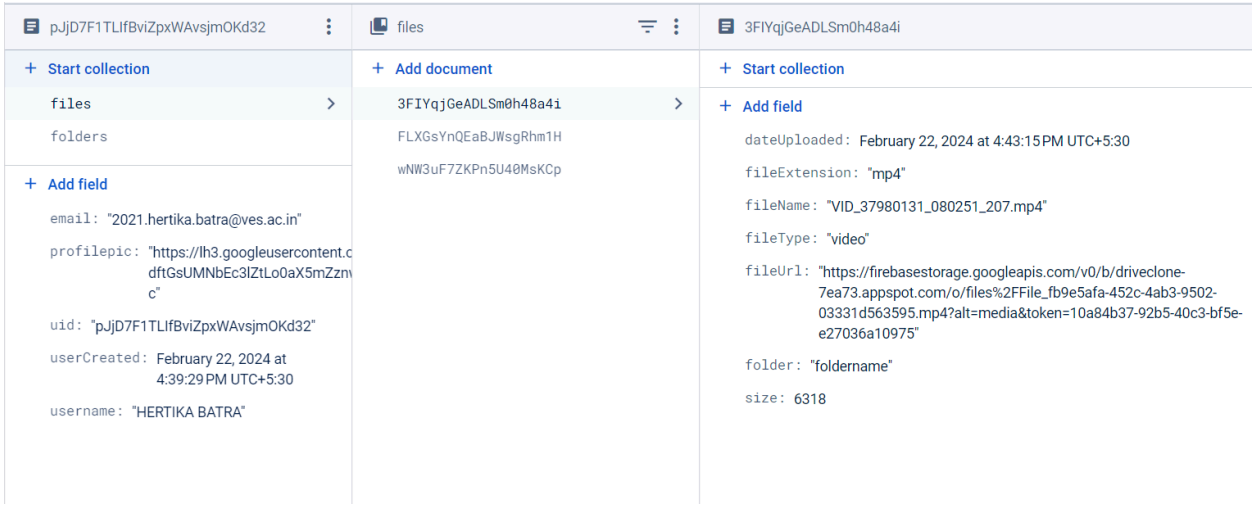
```
Future<void> uploadFile(String foldername) async {
 FilePickerResult? result =
    await FilePicker.platform.pickFiles(allowMultiple: true);

 if (result != null) {
  List<File> files = result.paths.map((path) => File(path!)).toList();

  for (File file in files) {
   String? fileType = lookupMimeType(file.path);
   print(fileType);
   if (fileType != null) {
    String filteredFileType = fileType.split('/')[0];
    String fileName = file.path.split('/').last;
    String fileExtension = fileName.substring(fileName.lastIndexOf('.') + 1);
    print(fileExtension);

    Object? compressedFile =
       await compressFile(file, filteredFileType);

    UploadTask uploadTask = FirebaseStorage.instance
       .ref()
       .child('files')
       .child('File_${uuid.v4()}.$fileExtension')
       .putFile(compressedFile as File);

    TaskSnapshot snapshot = await uploadTask.whenComplete(() {});
    String fileUrl = await snapshot.ref.getDownloadURL();

    await userCollection
       .doc(FirebaseAuth.instance.currentUser!.uid)
```

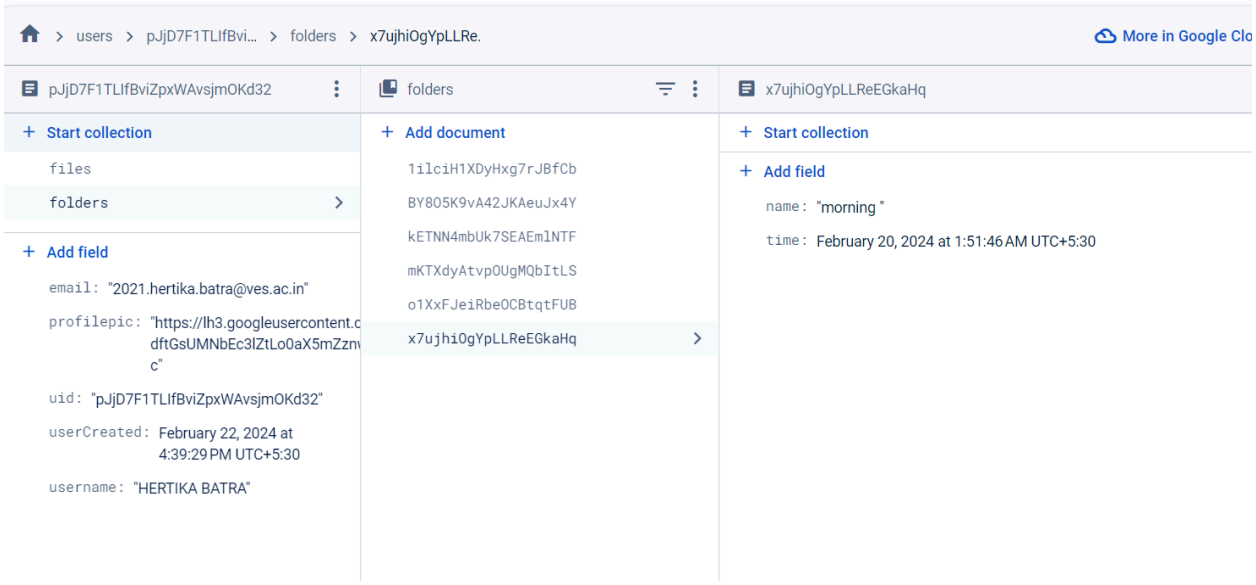
```
            .collection('files')
            .add({
          "fileName": fileName,
          "fileUrl": fileUrl,
          "fileType": filteredFileType,
          "fileExtension": fileExtension,
          "folder": foldername,
          "size": (compressedFile as File).readAsBytesSync().lengthInBytes ~/ 1024,
          "dateUploaded": DateTime.now(),
        });
      }
    }
    if (foldername == '') {
      Get.back();
    }
   } else {
    print("Cancelled");
   }
  }
}
```

Output:-


4:39
F-drive
Storage
Files
Recent Files
Sem 3 Anmol Marksheet.pdf
ReelVideo-41927.
ghghj
0 items
vesit
0 items
mahek
0 items
morning
0 items
Folder
Upload


Storage
Files
Rules
Usage
Extensions
Protect your Storage resources from abuse, such as billing fraud or phishing
Configure App Check
gs://driveclone-7ea73.appspot.com
files
Upload file
Name
Size
Type
Last modified
File_432089c4-979d-4c77-984f-2c99a72446ed.pdf
614.68 KB
Feb 22, 2024
File_fb9e5afa-452c-4ab3-9502-03331d563595.mp4
6.17 MB
Feb 22, 2024
File_fb9e5afa-452c...
Name
File_fb9e5afa-452c-4ab3-9502-03331d56...
Size
6,470,215 bytes
Type
Created
Feb 22, 2024, 4:43:15 PM
Updated
Feb 22, 2024, 4:43:15 PM
File location
Other metadata





Conclusion:- Here,successfully connected flutter with firebase

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.	5
Name	Hertika Batra
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO4: Understand various PWA frameworks and their requirements
Grade:	15

Name:-Hertika Batra **Roll:-5** **Batch:-A**

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&amp;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">
```

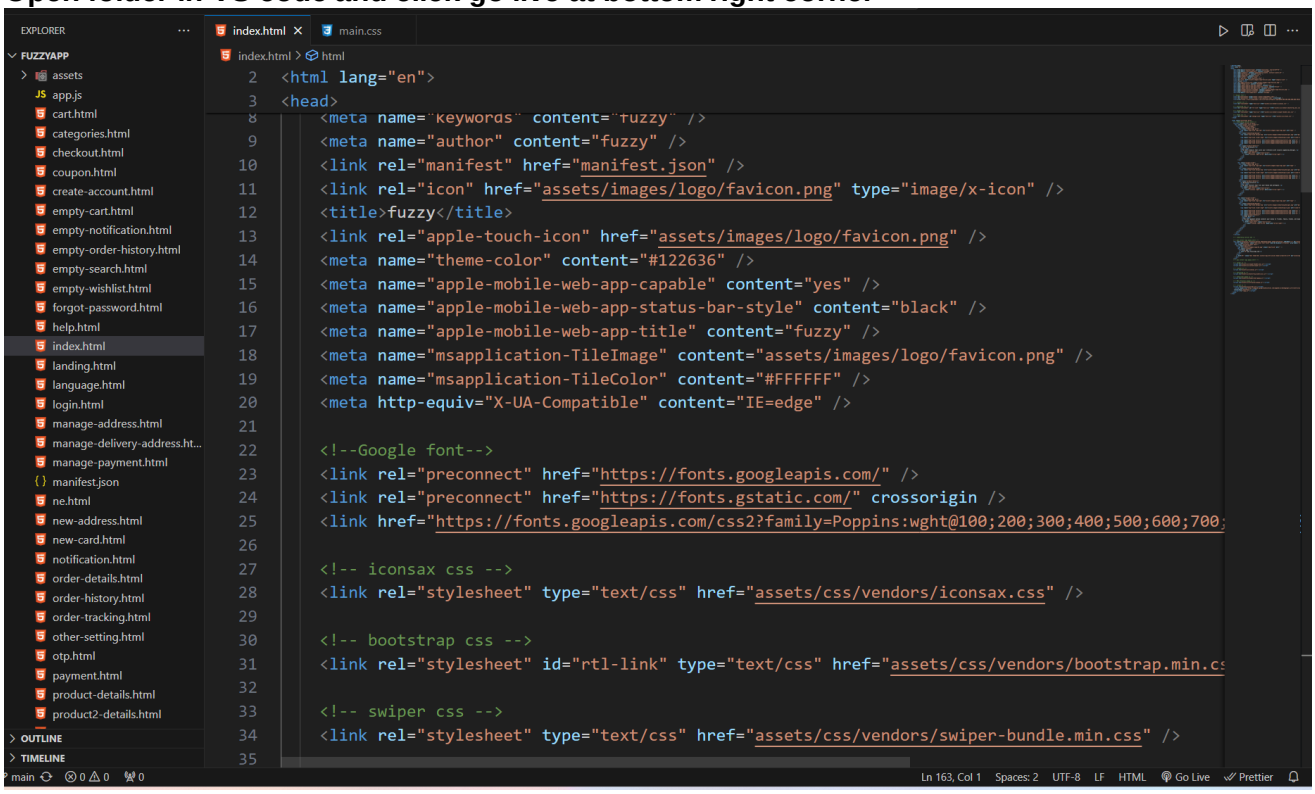
```
            <div class="theme-logo pb-3">
              <img class="img-fluid logo-img"
src="assets/images/logo/logo.png" alt="logo" />
            </div>
            <div class="onboarding-design">
              <img class="img-fluid design-img"
src="assets/images/onboarding/design1.png" alt="bg-design"
/>

              <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>
      </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>
```

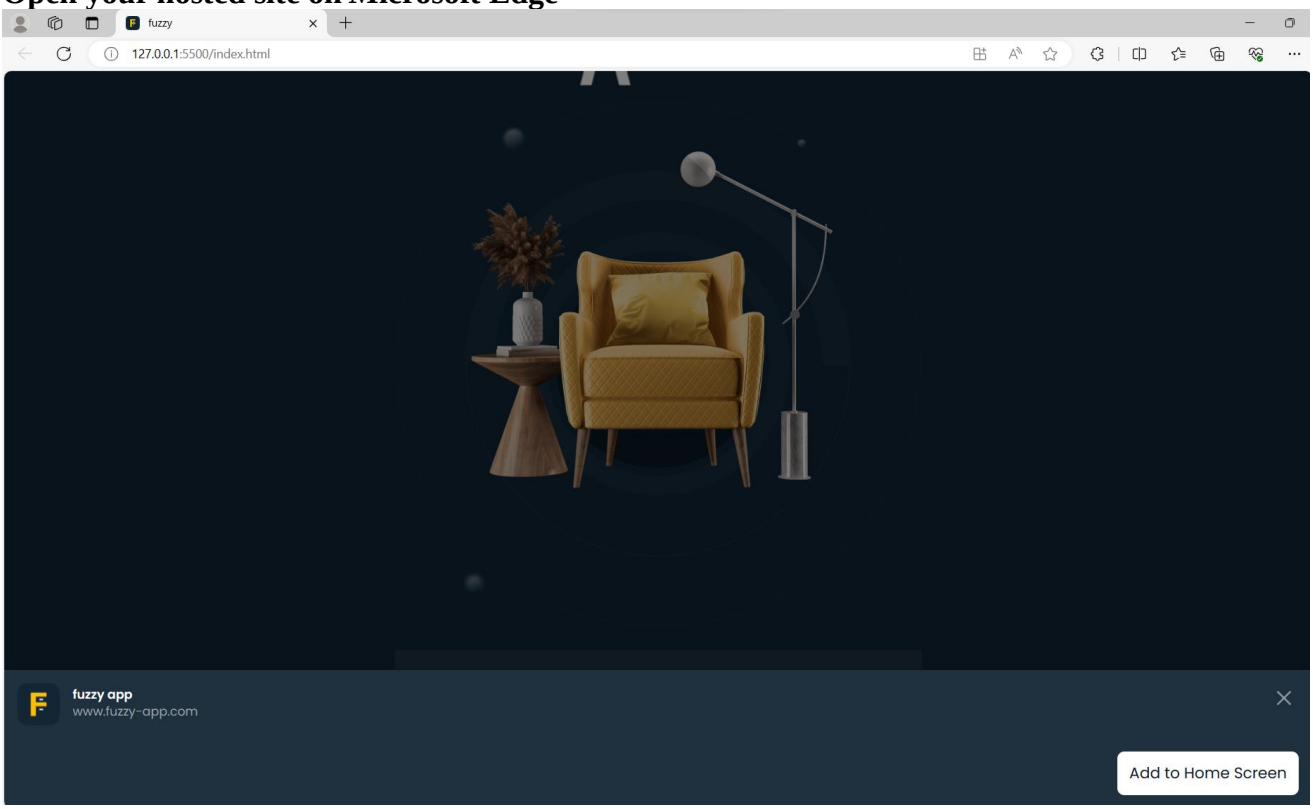
```
    <!-- homescreen popup js -->
    <script src="assets/js/homescreen-popup.js"></script>

    <!-- 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:

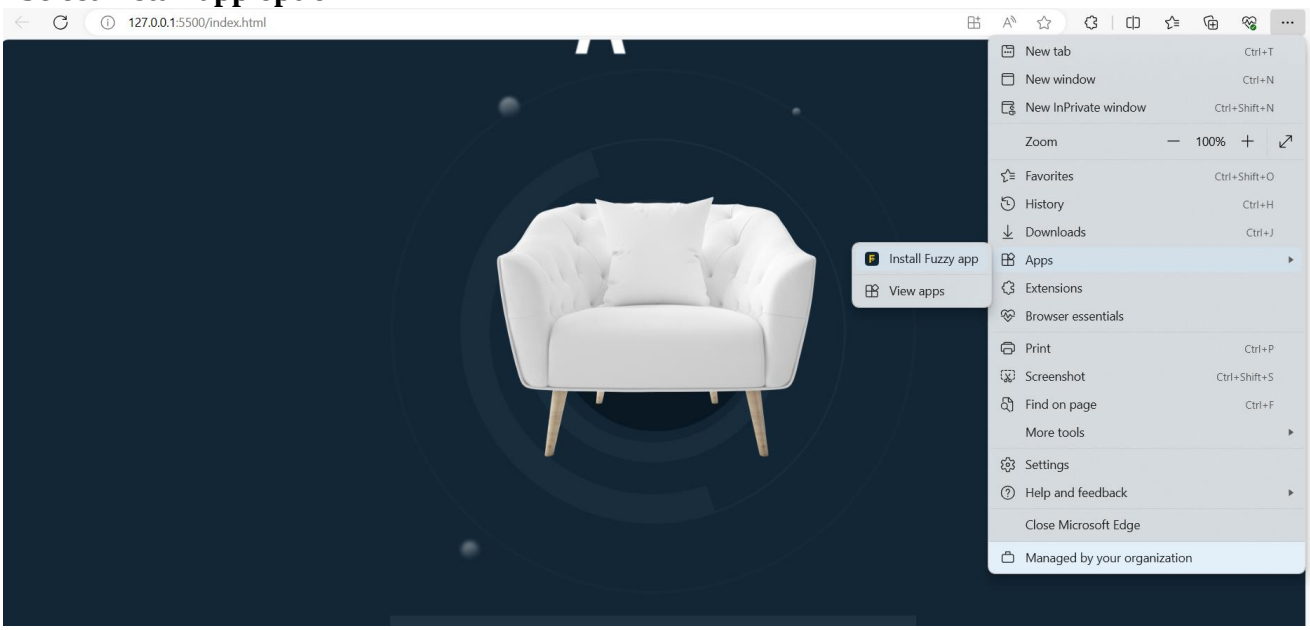
Open folder in VS code and click go live at bottom right corner

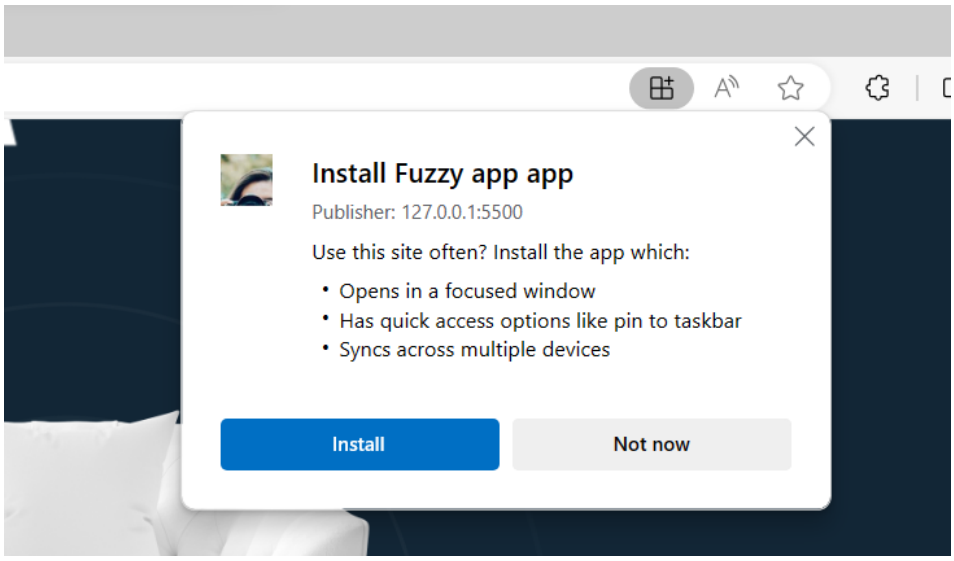


Open your hosted site on Microsoft Edge



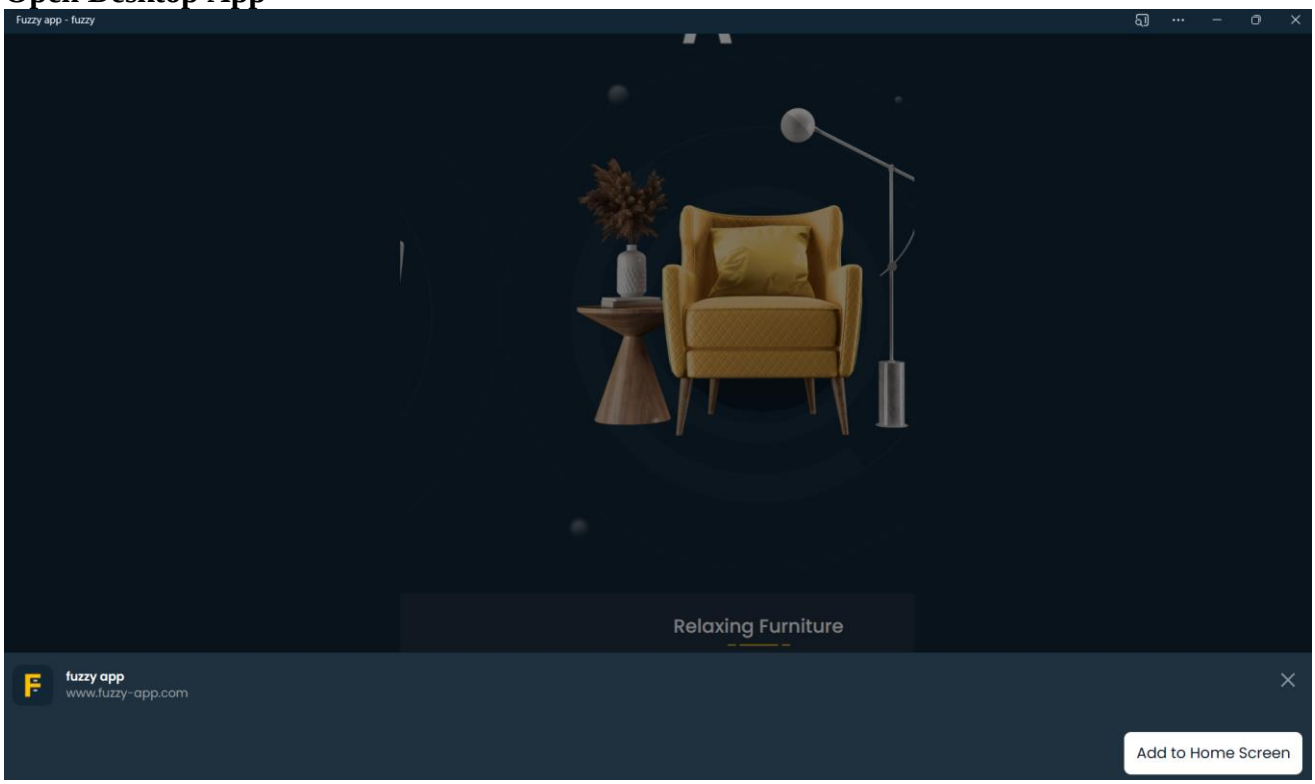
Click on 3 dots
Click on Apps
Select install app option

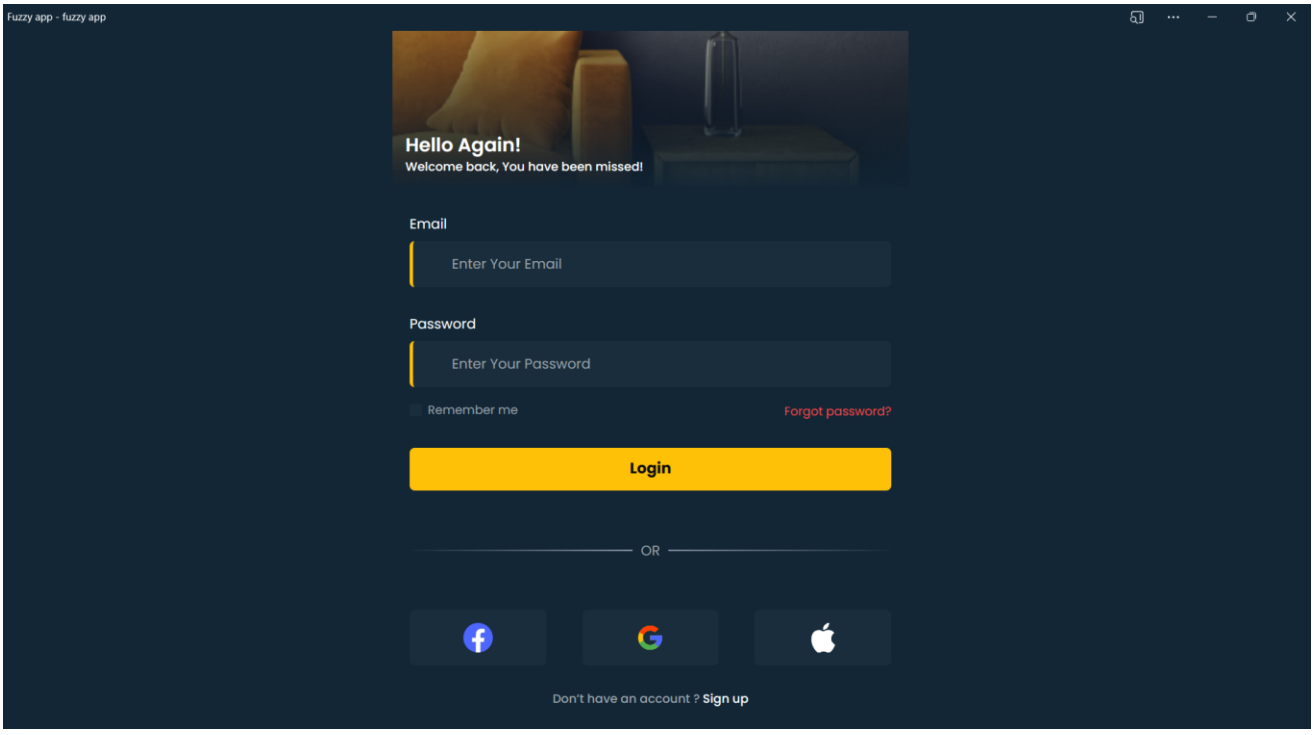




Desktop App created Successfully

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.	5
Name	Hertika Batra
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	15

Name: Hertika Batra **Roll:-5** **Batch:A**

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 ('serviceWorker' in navigator) {
  navigator.serviceWorker.register('/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('/service-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>
```

Style.css

```
@-webkit-keyframes fireworkLine {
  0% {
    right: 20%;
    -webkit-transform: scale(0, 0);
            transform: scale(0, 0);
  }
  25% {
    right: 20%;
    width: 6px;
    -webkit-transform: scale(1, 1);
            transform: scale(1, 1);
  }
  35% {
    right: 0;
    width: 35%;
  }
  70% {
    right: 0;
    width: 4px;
    -webkit-transform: scale(1, 1);
            transform: scale(1, 1);
  }
  100% {
    right: 0;
    -webkit-transform: scale(0, 0);
            transform: scale(0, 0);
```

```
32  @keyframes fireworkLine {
33    0% {
34      right: 20%;
35      -webkit-transform: scale(0, 0);
36              transform: scale(0, 0);
37    }
38    25% {
39      right: 20%;
40      width: 6px;
41      -webkit-transform: scale(1, 1);
42              transform: scale(1, 1);
43    }
44    35% {
45      right: 0;
46      width: 35%;
47    }
48    70% {
49      right: 0;
50      width: 4px;
51      -webkit-transform: scale(1, 1);
52              transform: scale(1, 1);
53    }
54    100% {
55      right: 0;
56      -webkit-transform: scale(0, 0);
```

App.js

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

app.js > ...

```
if ('serviceWorker' in navigator) {
    window.addEventListener('load', () => {
    navigator.serviceWorker.register('/service-worker.js')
    .then(registration => {
    console.log('Service Worker registered with scope:', registration.scope);
    })
    .catch(error => {
    console.error('Service Worker registration failed:', error);
    });
    });
    }
```

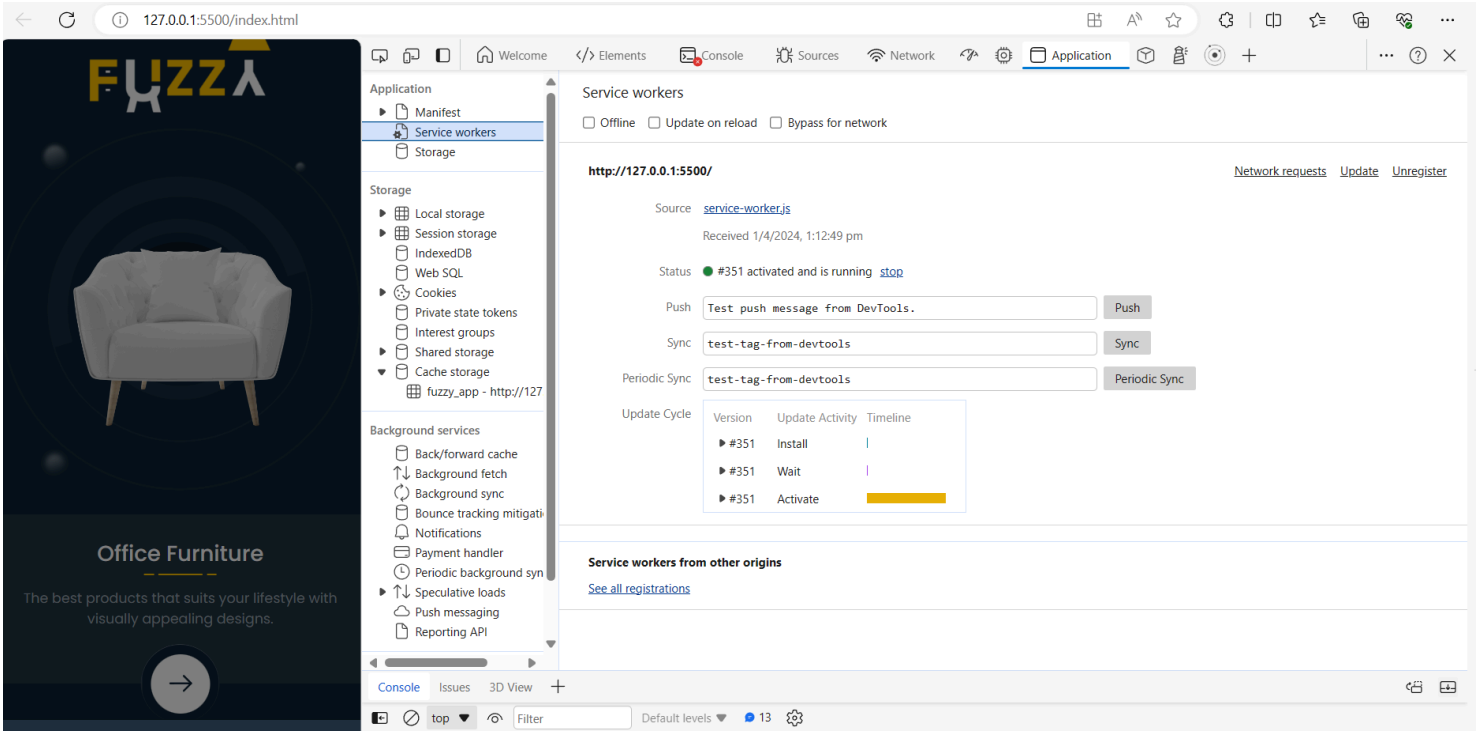
Service-Worker.js

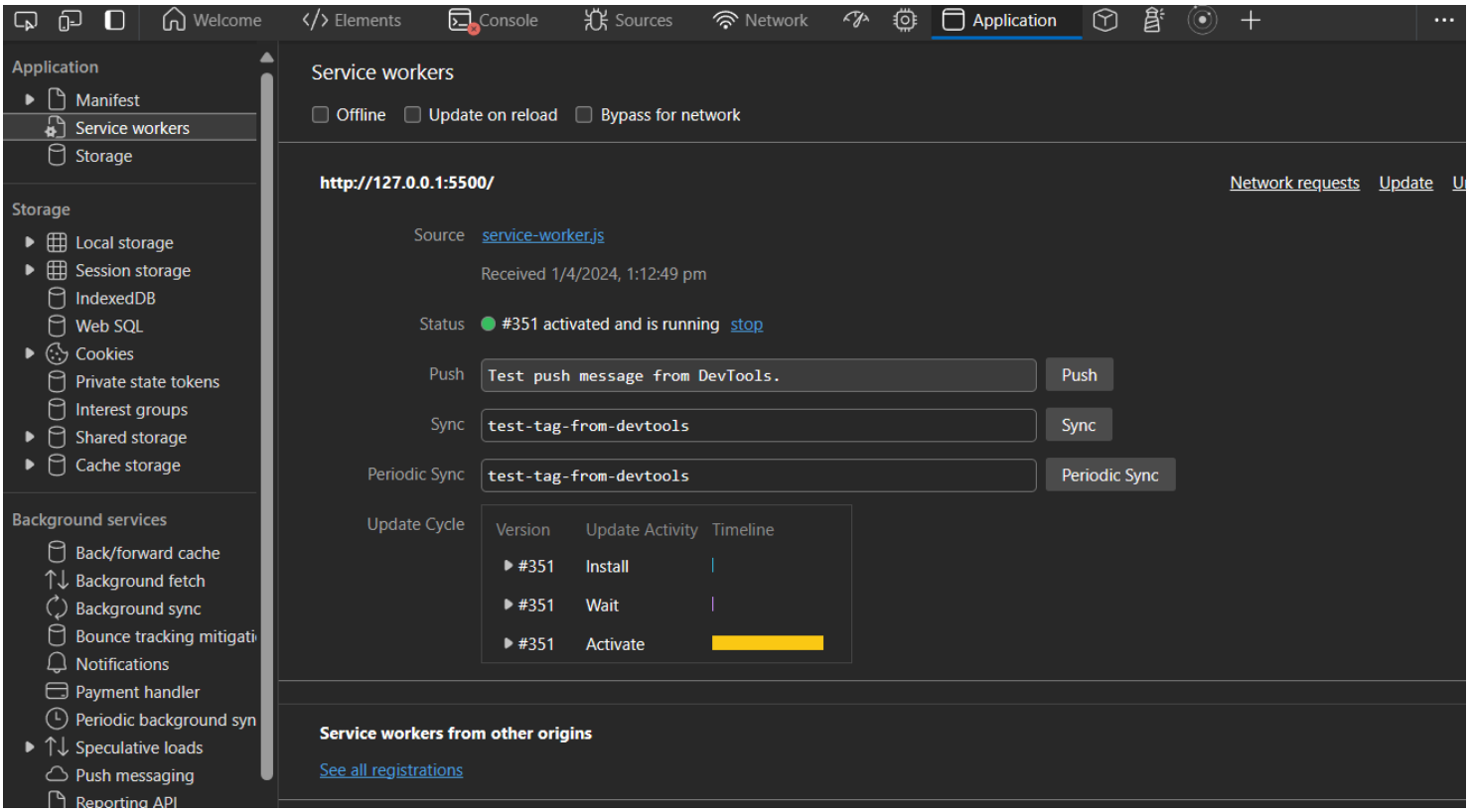
manifest.json M | app.js U | service-worker.js U | index.html M

service-worker.js > assetsToCache

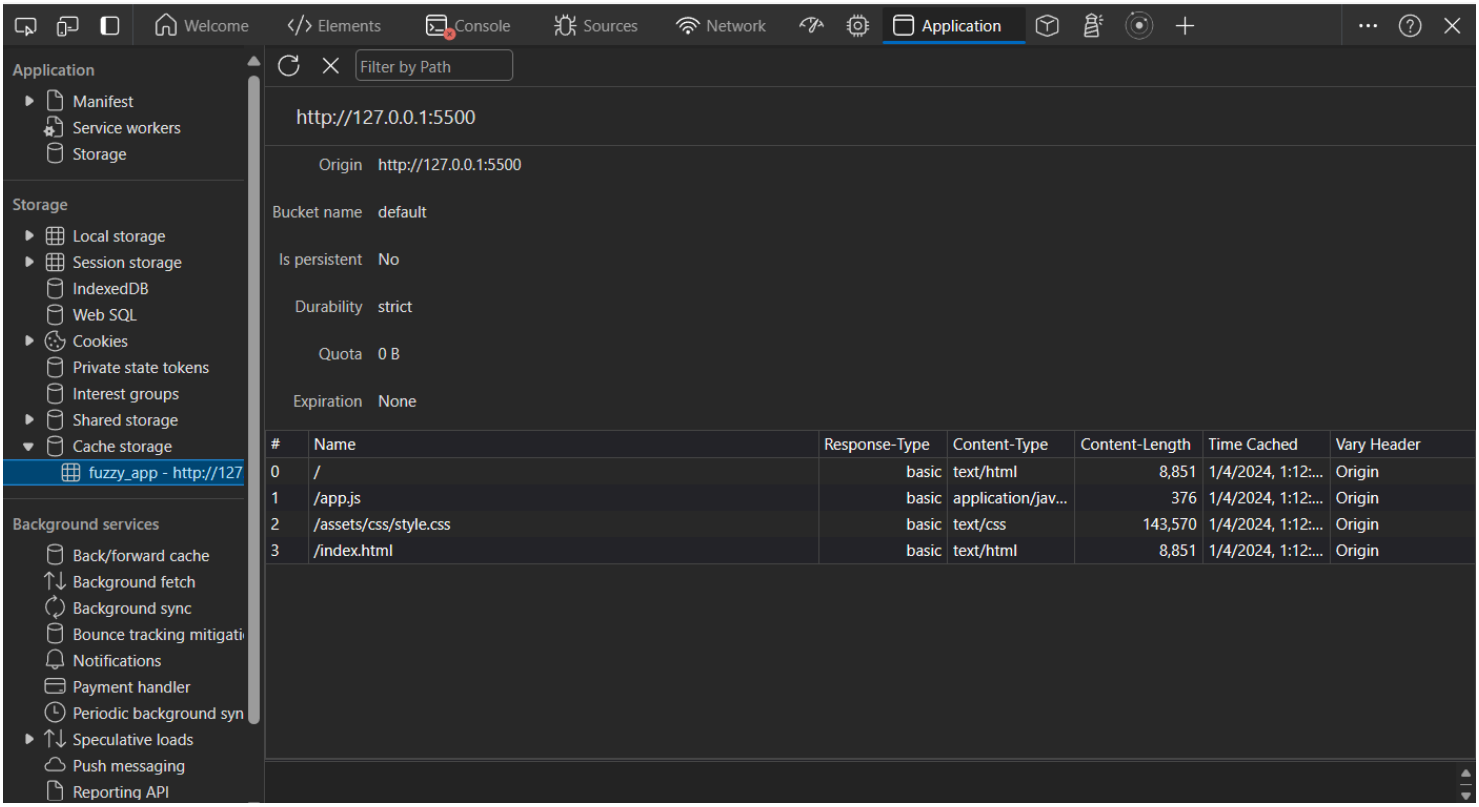
```
const cacheName = 'fuzzy_app';
const assetsToCache = [
'/',
'/index.html',
'/assets/css/style.css',
'/app.js'
]
self.addEventListener('install', event => {
event.waitUntil(
caches.open(cacheName)
.then(cache => {
return cache.addAll(assetsToCache);
})
);
});
self.addEventListener('activate', event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

OUTPUT:


127.0.0.1:5500/index.html
FUZZY
Office Furniture
The best products that suits your lifestyle with visually appealing designs.
Welcome
Elements
Console
Sources
Network
Application
Application
Manifest
Service workers
Storage
Storage
Local storage
Session storage
IndexedDB
Web SQL
Cookies
Private state tokens
Interest groups
Shared storage
Cache storage
fuzzy_app - http://127
Background services
Back/forward cache
Background fetch
Background sync
Bounce tracking mitigati
Notifications
Payment handler
Periodic background syn
Speculative loads
Push messaging
Reporting API
Service workers
Offline
Update on reload
Bypass for network
http://127.0.0.1:5500/
Network requests
Update
Unregister
Source
service-worker.js
Received 1/4/2024, 1:12:49 pm
Status
#351 activated and is running
stop
Push
Test push message from DevTools.
Push
Sync
test-tag-from-devtools
Sync
Periodic Sync
test-tag-from-devtools
Periodic Sync
Update Cycle
Version
Update Activity
Timeline
#351
Install
#351
Wait
#351
Activate
Service workers from other origins
See all registrations
Console
Issues
3D View
top
Filter
Default levels
13


Welcome
Elements
Console
Sources
Network
Application
Application
Manifest
Service workers
Storage
Storage
Local storage
Session storage
IndexedDB
Web SQL
Cookies
Private state tokens
Interest groups
Shared storage
Cache storage
Background services
Back/forward cache
Background fetch
Background sync
Bounce tracking mitigati
Notifications
Payment handler
Periodic background syn
Speculative loads
Push messaging
Reporting API
Service workers
Offline
Update on reload
Bypass for network
http://127.0.0.1:5500/
Network requests
Update
Source
service-worker.js
Received 1/4/2024, 1:12:49 pm
Status
#351 activated and is running
stop
Push
Test push message from DevTools.
Push
Sync
test-tag-from-devtools
Sync
Periodic Sync
test-tag-from-devtools
Periodic Sync
Update Cycle
Version
Update Activity
Timeline
#351
Install
#351
Wait
#351
Activate
Service workers from other origins
See all registrations

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.	5
Name	Hertika Batra
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	15

Name: Hertika Batra **Roll:-5** **Batch:A**

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:

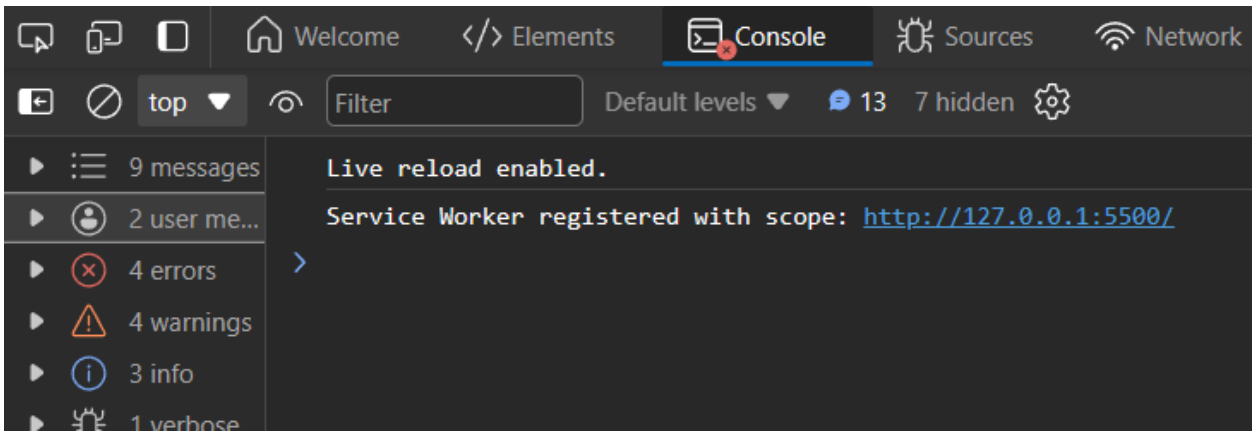
Service-Worker.js

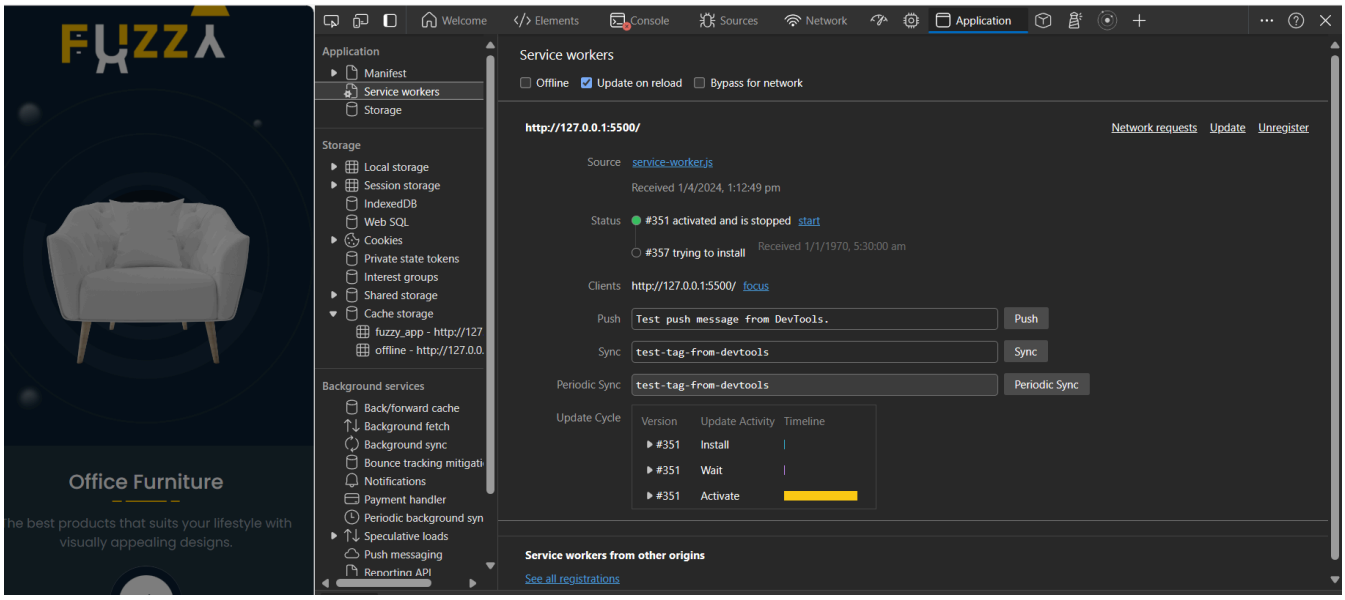
```
self.addEventListener("install", function (event) {
  event.waitUntil(preLoad());
  });
  self.addEventListener("fetch", function (event) {
  event.respondWith(
  checkResponse(event.request).catch(function () {
  console.log("Fetch from cache successful!");
  return returnFromCache(event.request);
  })
  );
  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) {
```

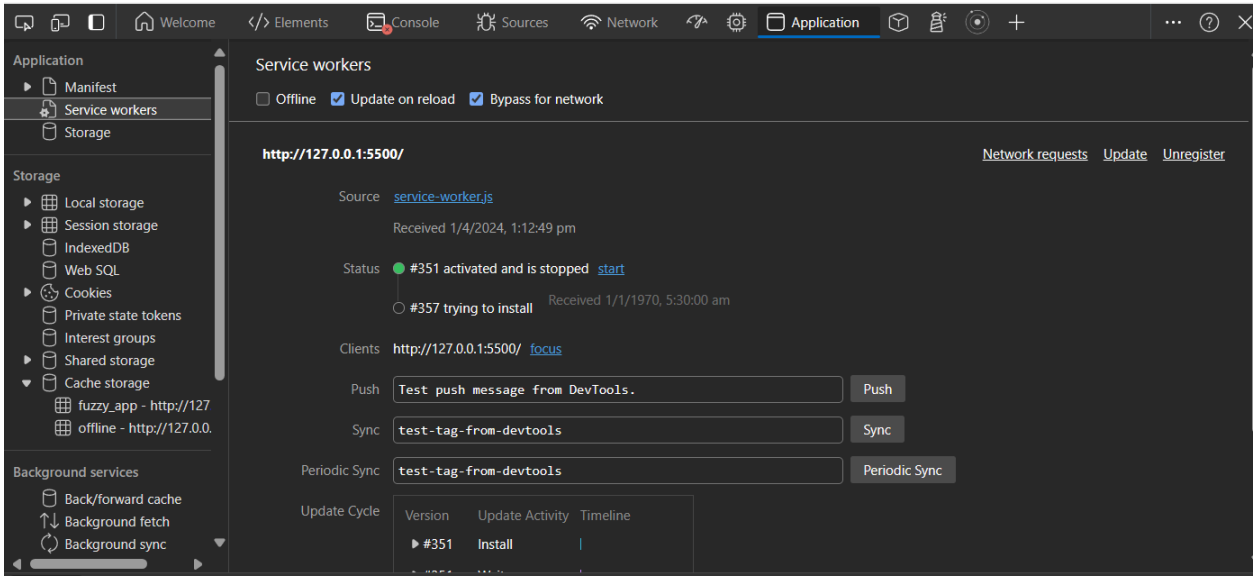
```
// caching index and important routes
return cache.addAll([
"/",
"/index.html",
"/landing.html",
"/profile.html",
"/shop.html",
"/cart.html",
"/css/main.css",
]);
});
};
var checkResponse = function (request) {
return new Promise(function (fulfill, reject) {
fetch(request)
.then(function (response) {
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;
});
});
});
};
```

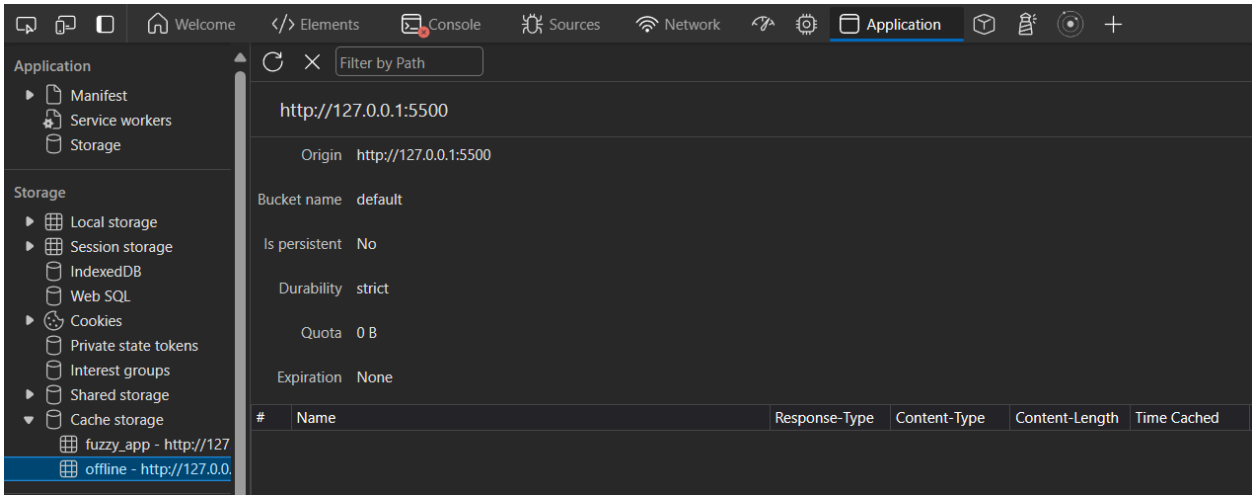
Output:







Offline



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.	5
Name	Hertika Batra
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	15

Name:- Hertika Roll:5 Batch:A

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

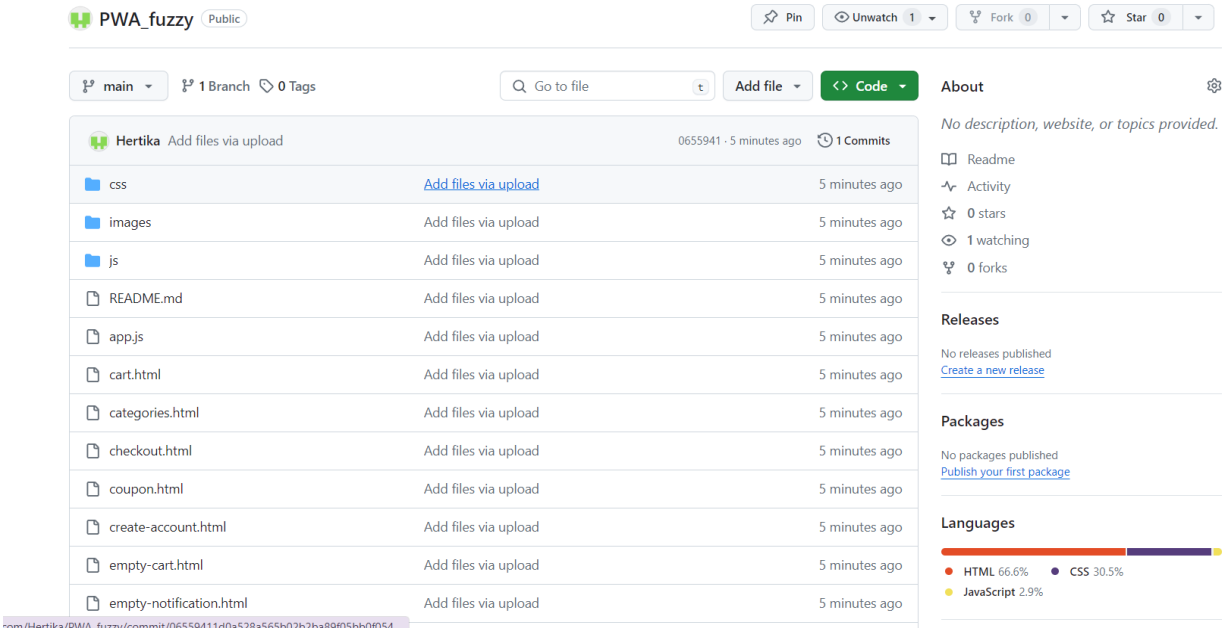
problem, if you use a CDN or AMP.

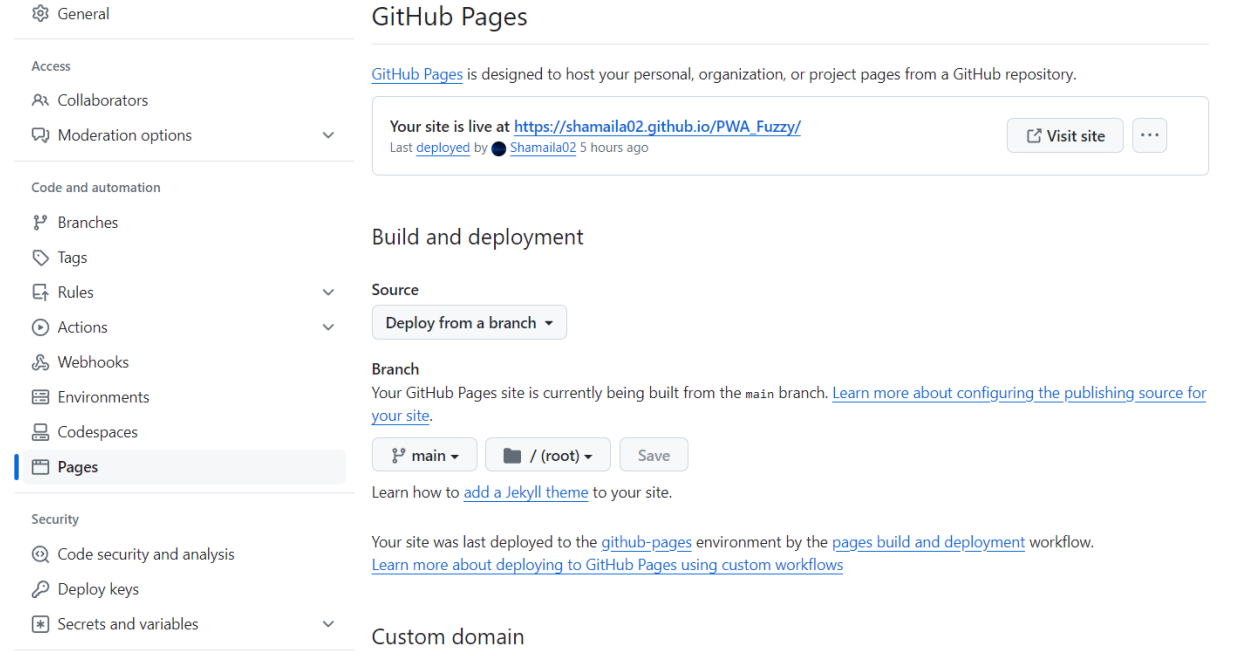
2. Command-line interface only.
3. No in-built support for any static site generator.

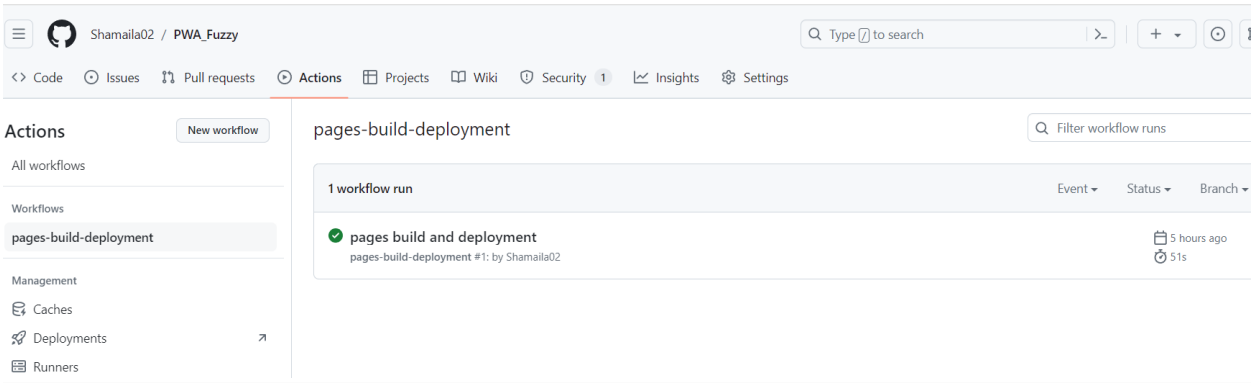
Link to our GitHub repository: https://github.com/Hertika/PWA_fuzzy

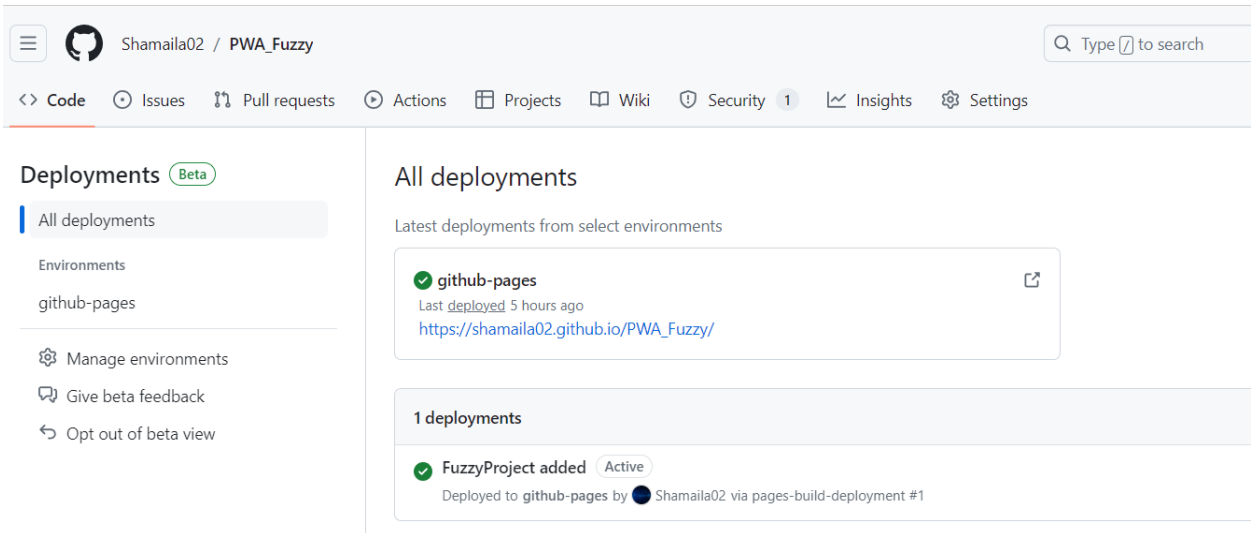
Hosted Link: https://shamaila02.github.io/PWA_Fuzzy/

Github Screenshot:




General
Access
Collaborators
Moderation options
Code and automation
Branches
Tags
Rules
Actions
Webhooks
Environments
Codespaces
Pages
Security
Code security and analysis
Deploy keys
Secrets and variables
GitHub Pages
GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.
Your site is live at https://shamaila02.github.io/PWA_Fuzzy/
Last deployed by Shamaila02 5 hours ago
Visit site
Build and deployment
Source
Deploy from a branch
Branch
Your GitHub Pages site is currently being built from the main branch. Learn more about configuring the publishing source for your site.
main
/ (root)
Save
Learn how to add a Jekyll theme to your site.
Your site was last deployed to the github-pages environment by the pages build and deployment workflow.
Learn more about deploying to GitHub Pages using custom workflows
Custom domain


Shamaila02 / PWA_Fuzzy
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security 1
Insights
Settings
Actions
New workflow
All workflows
Workflows
pages-build-deployment
Management
Caches
Deployments
Runners
pages-build-deployment
Filter workflow runs
1 workflow run
Event
Status
Branch
pages build and deployment
pages-build-deployment #1: by Shamaila02
5 hours ago
51s


Shamaila02 / PWA_Fuzzy
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security 1
Insights
Settings
Deployments Beta
All deployments
Environments
github-pages
Manage environments
Give beta feedback
Opt out of beta view
All deployments
Latest deployments from select environments
github-pages
Last deployed 5 hours ago
https://shamaila02.github.io/PWA_Fuzzy/
1 deployments
FuzzyProject added
Active
Deployed to github-pages by Shamaila02 via pages-build-deployment #1

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.	5
Name	Hertika Batra
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO6: Develop and Analyze PWA Features and deploy it over app hosting solution
Grade:	15

Name: Hertika Batra **Roll:5** **Batch:A**

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. Scoring

points are based on the Baseline PWA checklist laid down by Google which includes Service Worker implementation(s), viewport handling, offline functionality, performance in script-disabled environments, 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 not screen-reader friendly (HTML5 introduced features that would make pages easy to interpret for screen readers used by visually challenged 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 HTTPS

   Avoiding 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

manifest.json M | index.html

index.html > html > head

```
<!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" />
```

Style.css

manifest.json M | style.css 5

assets > css > style.css > ...

```
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;
}
```

Script.js

```
document.getElementById("loginForm").addEventListen
er("submit",(event)=>{
    event.preventDefault()
})

firebase.auth().onAuthStateChanged((user)=>{
    if(user){
        location.replace("/landing.html")
    }
})

function login(){
                            const       email       =
document.getElementById("email").value
                          const       password       =
document.getElementById("password").value

firebase.auth().signInWithEmailAndPassword(email,
password)
    .catch((error)=>{
          document.getElementById("error").innerHTML
= error.message
    })
}

function signUp(){
                        const      email_login      =
```

```
document.getElementById("email_login").value
                    const     password_login     =
document.getElementById("password_login").value

firebase.auth().createUserWithEmailAndPassword(emai
l, password)
    .catch((error) => {
          document.getElementById("error").innerHTML
= error.message
    });
}

function forgotPass(){
                            const       email       =
document.getElementById("email").value
    firebase.auth().sendPasswordResetEmail(email)
    .then(() => {
        alert("Reset link sent to your email id")
    })
    .catch((error) => {
          document.getElementById("error").innerHTML
= error.message
    });
}
```

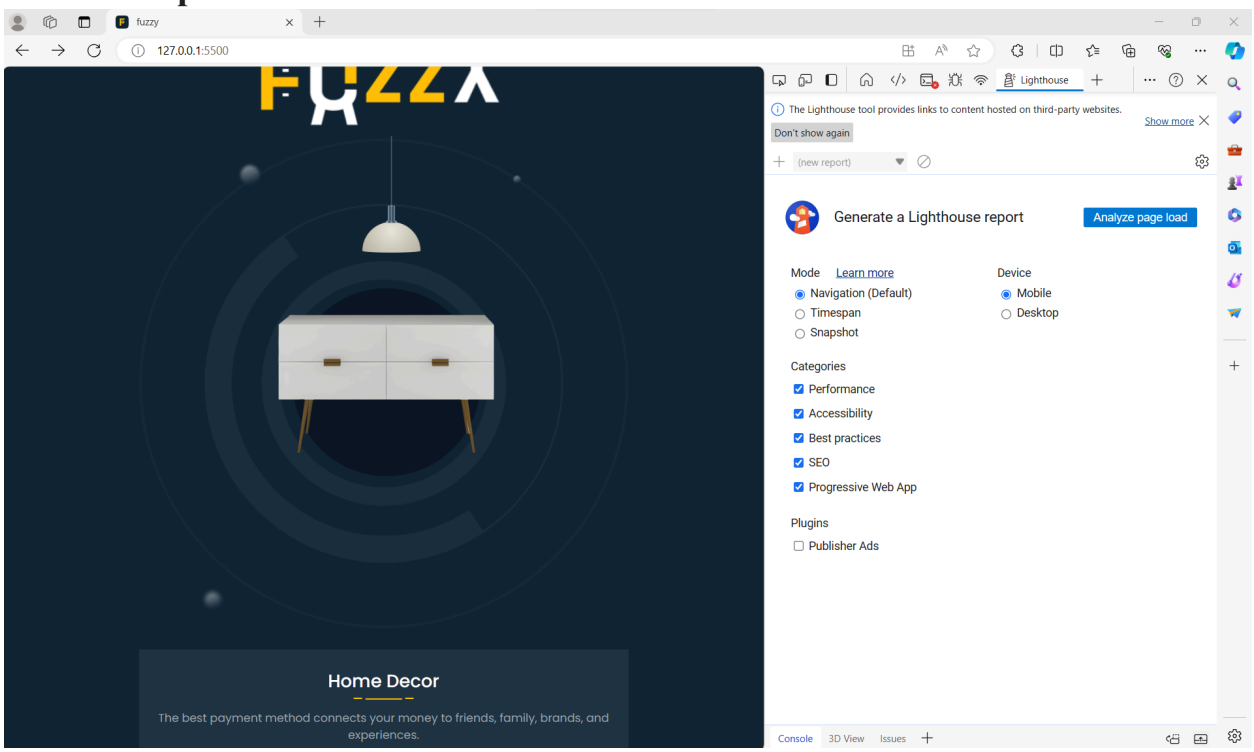
Manifest.json

```
{
  "name": "Fuzzy app",
  "short_name": "Fuzzy app",
```

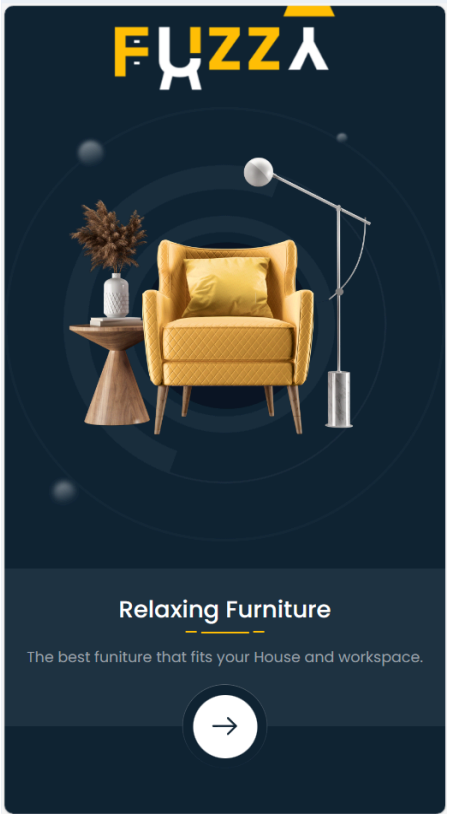
```
"lang": "en-US",
"start_url": ".",
"display": "standalone",
"background_color": "white",
"theme_color": "#122636",
"description": "fuzzy app",
"icons": [
  {
    "src": "assets/images/logo/512.png",
    "sizes": "144x144",
    "type": "image/png",
    "purpose": "any"
  },

  {
    "src": "assets/images/logo/512.png",
    "sizes": "512x512",
    "type": "image/png",
    "purpose": "any maskable"
  },
  {
    "src": "assets/images/icon.png",
    "sizes": "144x144",
    "type": "image/png",
    "purpose": "any maskable"
  }

]
```

Output:

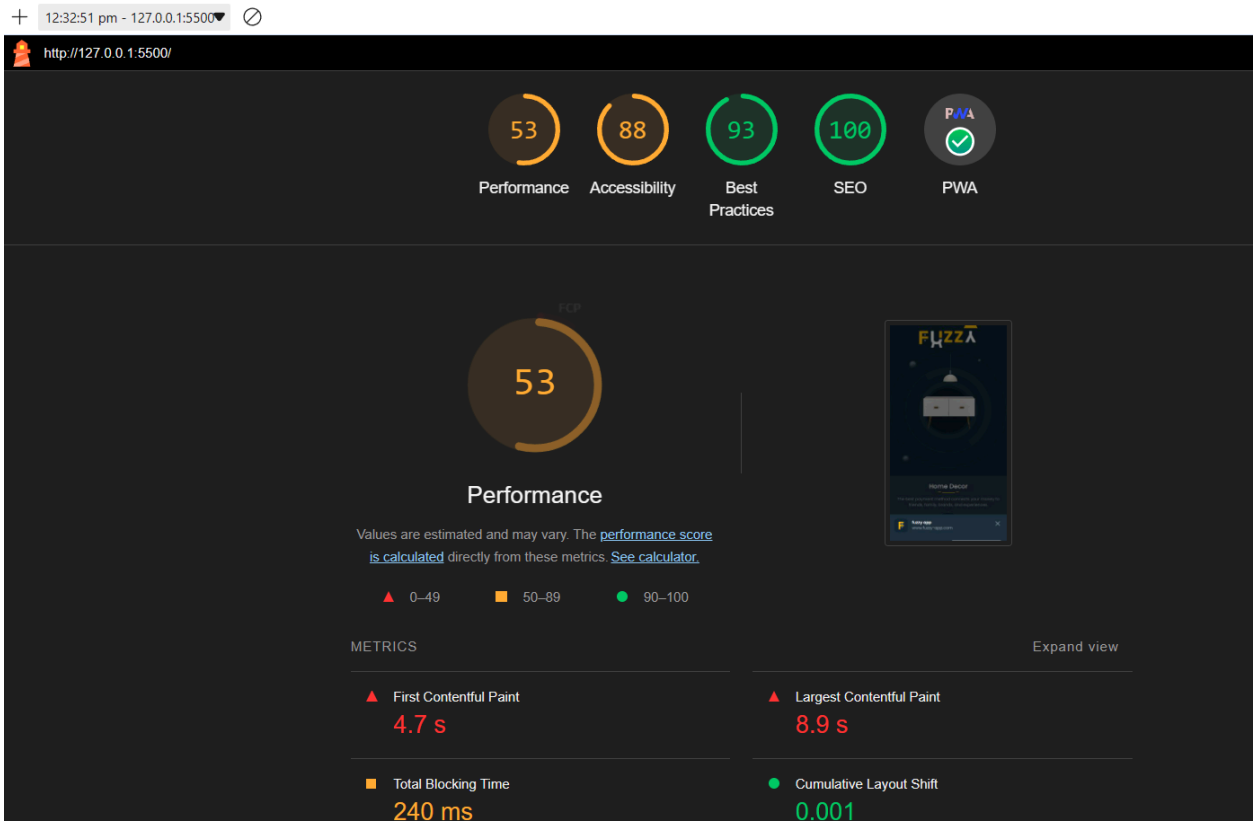
For Desktop Devices

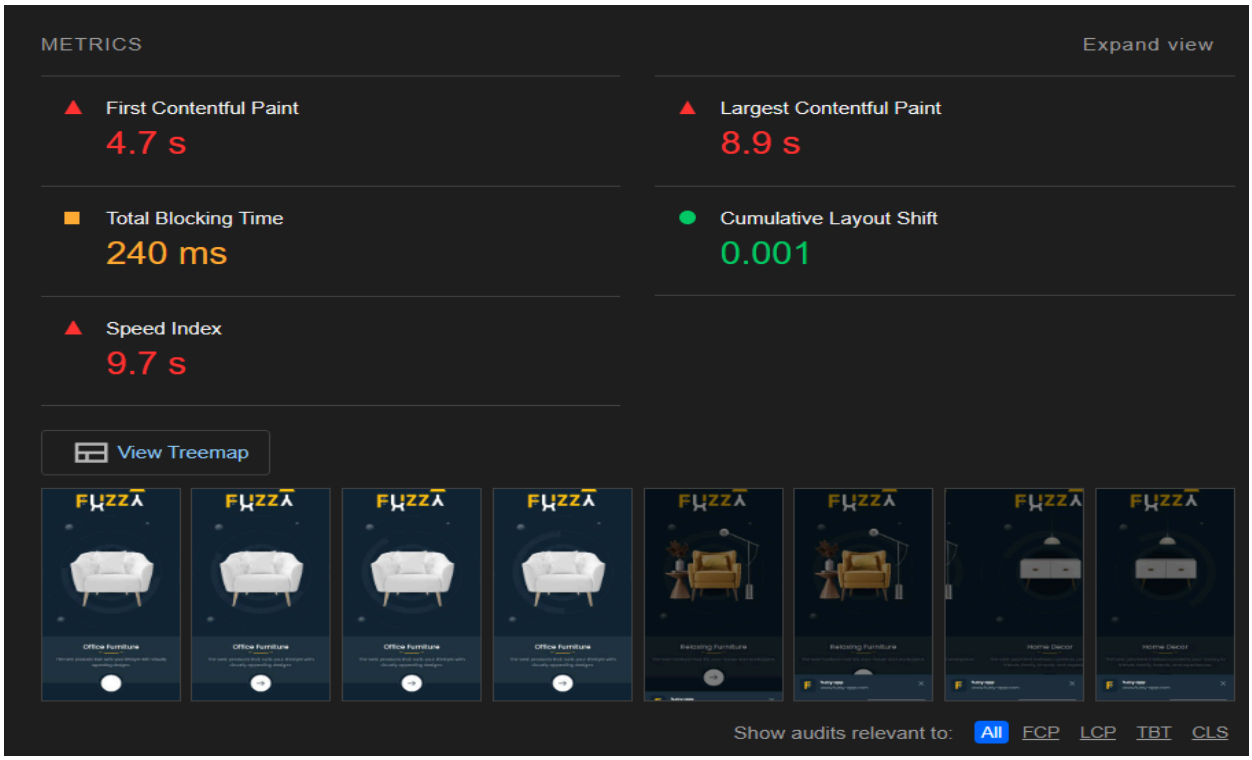


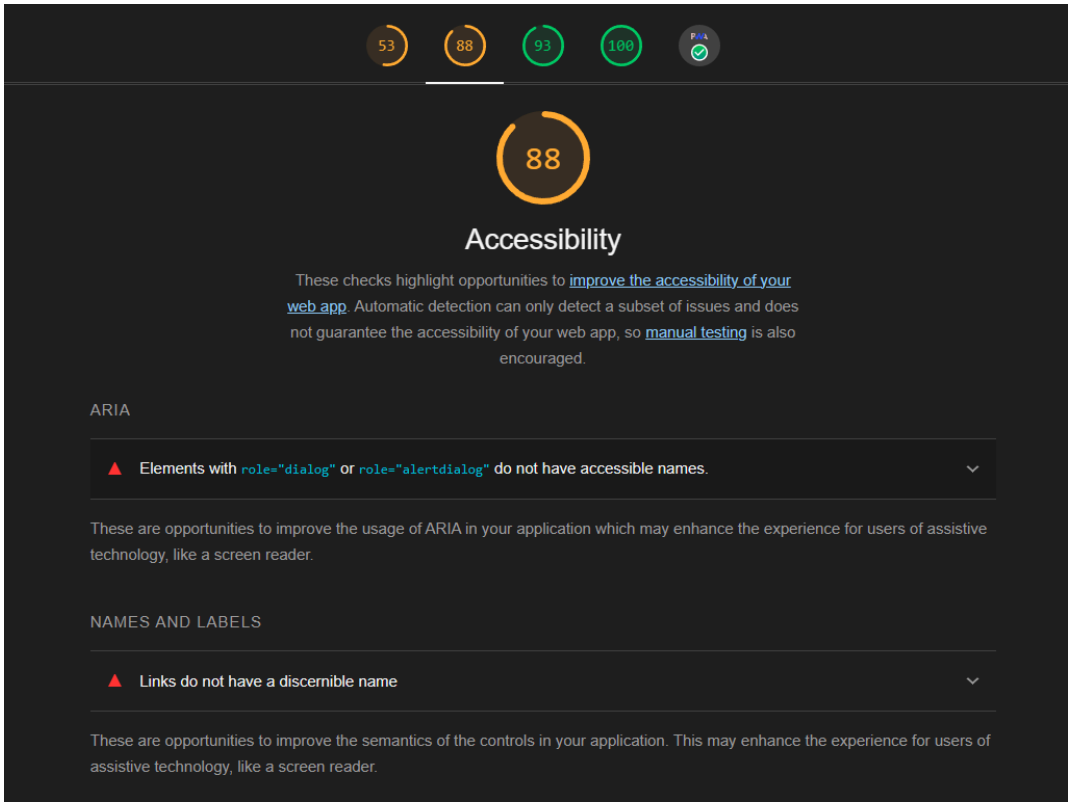
For Mobile Devices

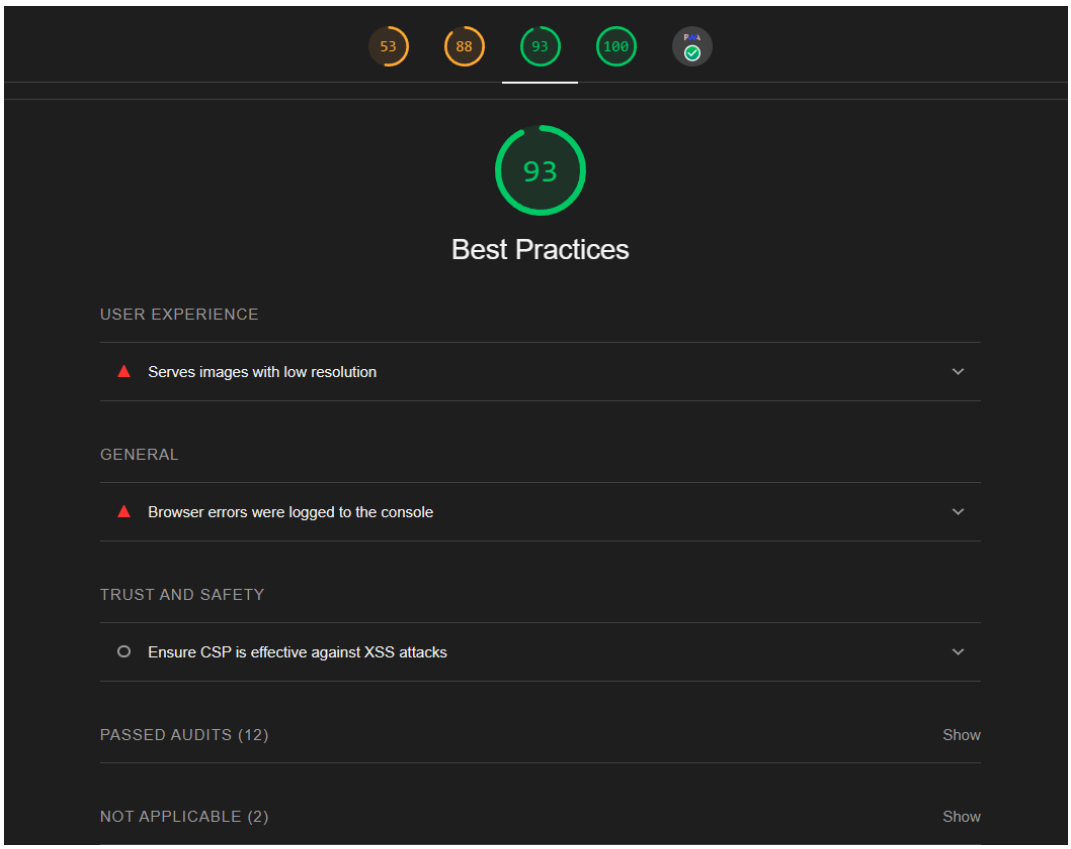


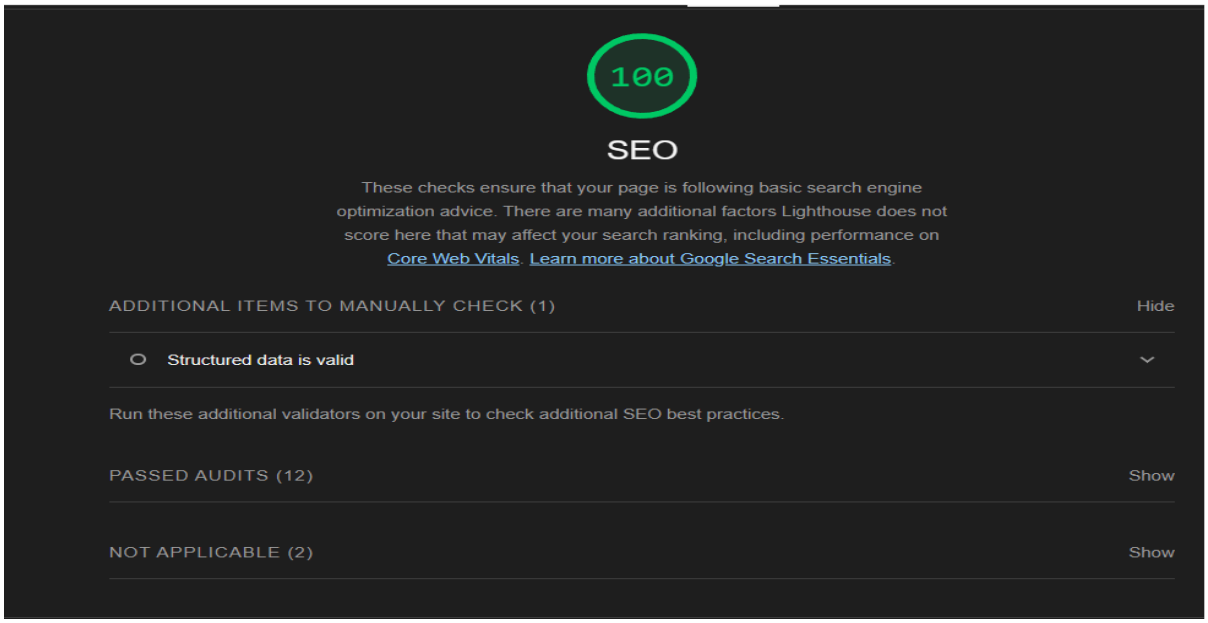
Report Generated

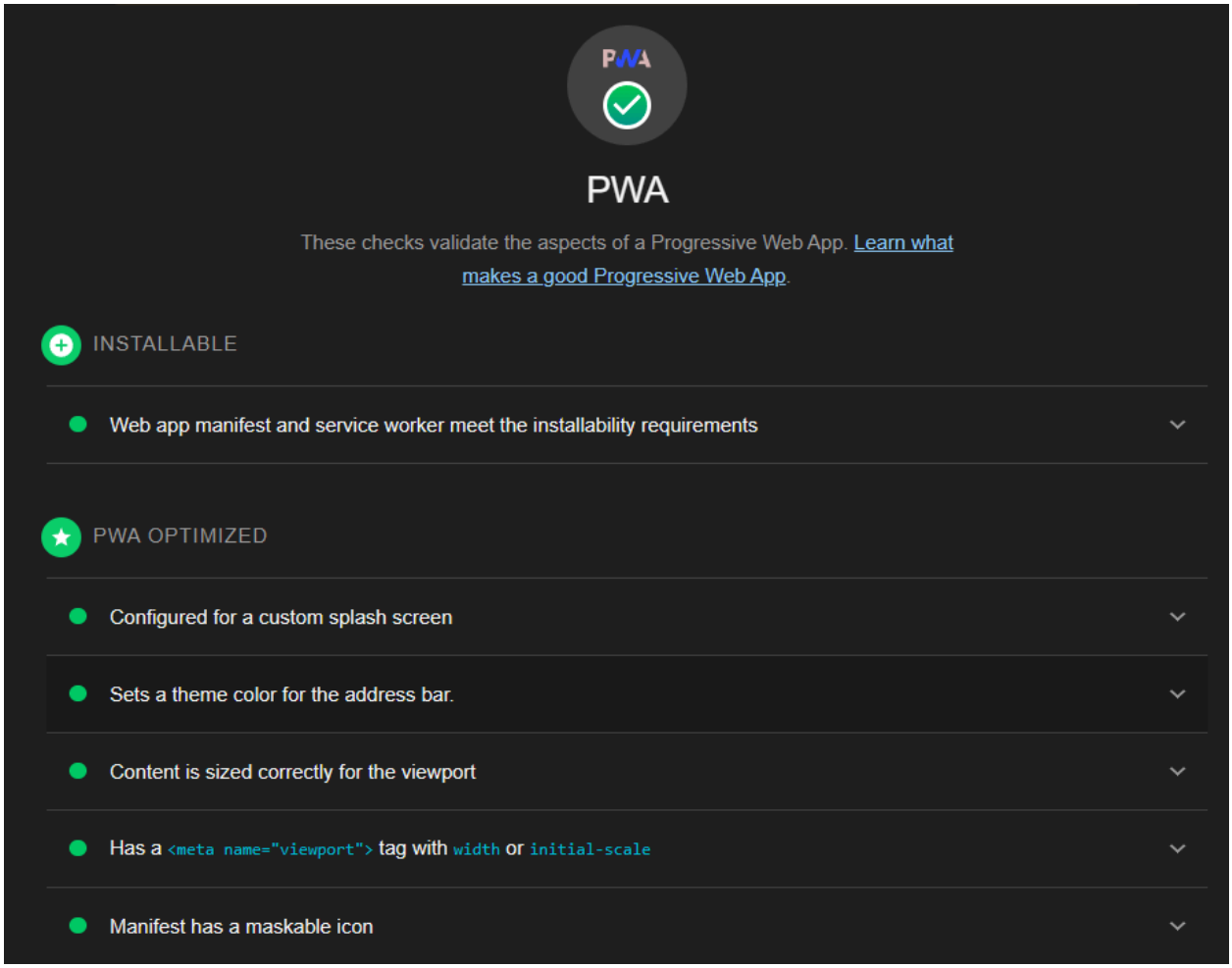


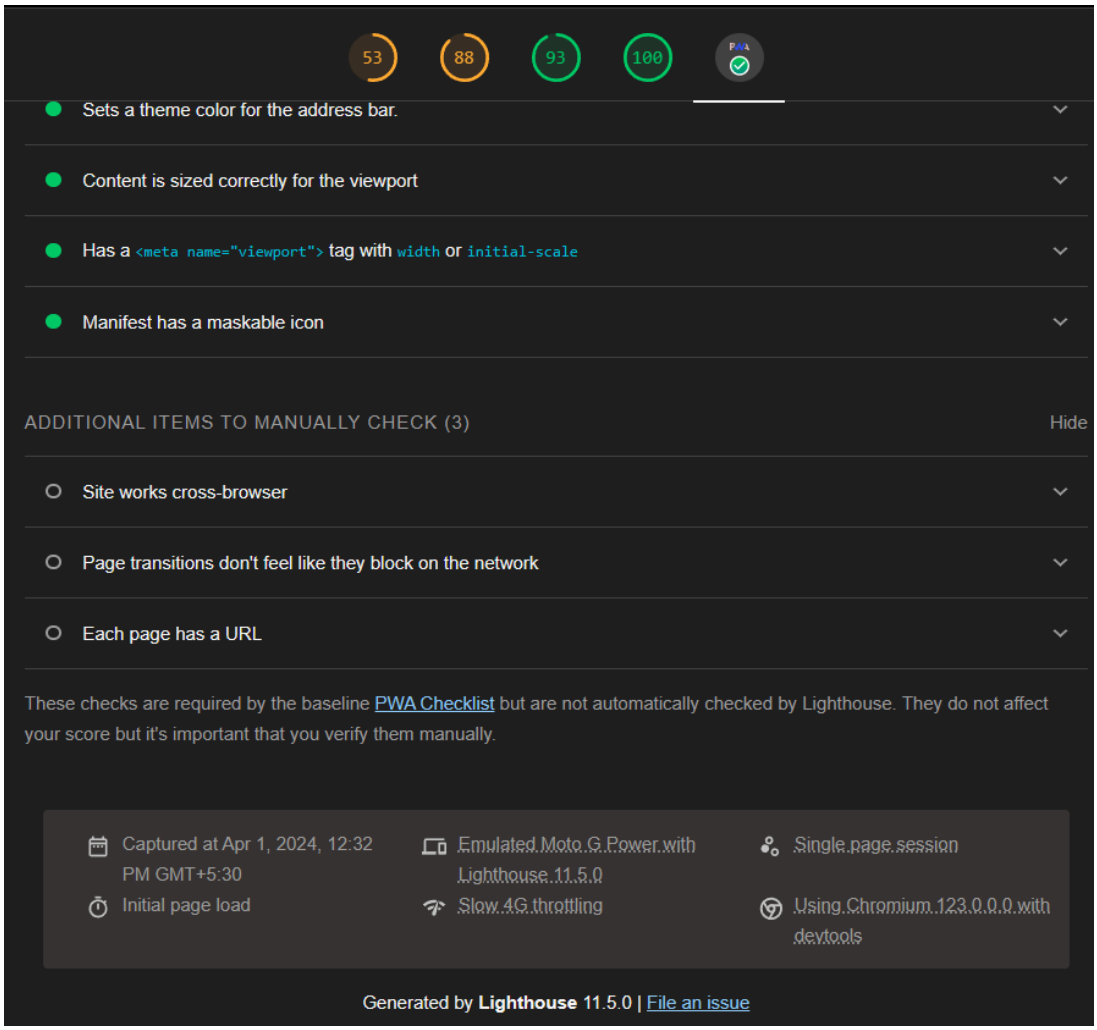



53
88
93
100
88
Accessibility
These checks highlight opportunities to improve the accessibility of your web app. Automatic detection can only detect a subset of issues and does not guarantee the accessibility of your web app, so manual testing is also encouraged.
ARIA
Elements with role="dialog" or role="alertdialog" do not have accessible names.
These are opportunities to improve the usage of ARIA in your application which may enhance the experience for users of assistive technology, like a screen reader.
NAMES AND LABELS
Links do not have a discernible name
These are opportunities to improve the semantics of the controls in your application. This may enhance the experience for users of assistive technology, like a screen reader.


53
88
93
100
93
Best Practices
USER EXPERIENCE
Serves images with low resolution
GENERAL
Browser errors were logged to the console
TRUST AND SAFETY
Ensure CSP is effective against XSS attacks
PASSED AUDITS (12)
Show
NOT APPLICABLE (2)
Show


100
SEO
These checks ensure that your page is following basic search engine optimization advice. There are many additional factors Lighthouse does not score here that may affect your search ranking, including performance on Core Web Vitals. Learn more about Google Search Essentials.
ADDITIONAL ITEMS TO MANUALLY CHECK (1)
Hide
Structured data is valid
Run these additional validators on your site to check additional SEO best practices.
PASSED AUDITS (12)
Show
NOT APPLICABLE (2)
Show


PWA
These checks validate the aspects of a Progressive Web App. Learn what makes a good Progressive Web App.
INSTALLABLE
Web app manifest and service worker meet the installability requirements
PWA OPTIMIZED
Configured for a custom splash screen
Sets a theme color for the address bar.
Content is sized correctly for the viewport
Has a <meta name="viewport"> tag with width or initial-scale
Manifest has a maskable icon



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.	5
Name	Hertika Batra
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

Name - Hemlika Bateja
ROU - 5 . Div:- D15A
Batch - A.

Assignment :- 1

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

=> 1) Key features of Flutter:
a) Single codebase for multiple platforms.
Flutter allows developers to write code & deploy it on both iOS & Android platform
b) Hot Reload -
This enables developers to instantly see the results of the code changes they make.
c) Expressive UI:-
Developer have the flexible to create Expressive & flexible UIs
d) Integration with other Tools:-
Flutter can easily Integrate with other popular development tools & frameworks.

2) Advantages of Flutter

a) Faster Development
uses single code base for multiple platforms.

5. Consistent UI Across platforms
Widgets provide a consistent look & feel across different platforms
6. Cost Efficiency:-
Developing & maintaining single codebase for both iOS & Android reduces development cost & resources.

3. Differ from Traditional Approach
(a) Traditional approach uses a hierarchical structure for UI components whereas Flutter uses a widget-based approach
(b) Flutter compiles to native ARM code providing performance comparable to native application.

Flutters popularity is driven by increased productivity a growing community flexibility in UI design, cross-platform development capabilities & adoption by major companies.

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

⇒ Widget tree

- The widget tree is a hierarchical structure of widgets that defines the user interface of an application.
- Every visual element from simple components to complex layouts is represented by a widget
- Widget can be categorized in 2 types.
  (a) Stateless widget
  It is immutable & cannot change over time eg images, text
  (b) Stateful widget
  widget that can change its state over time eg:- buttons, forms

Widget Composition

- Widget composition in flutter involves combining multiple simple widgets to create more complex & compound widgets
- This composability is a powerful concept that allows developers to build sophisticated

User Interfaces by nesting widgets within each other

Commonly used widgets:-

(a) Container
- A box model for padding, margin & decoration.

(b) Column & Row
- layout widgets for arranging children vertically or horizontally

(c) Stack
- overlaping widgets allowing them to be layered on the top of each other

(d) list view
- A scrollable list of widgets

(e) grid view
- A scrollable grid of widgets

(f) App Bar
- A material design app bar typically at top of screen

(g) Text field
- Input field for users to enter text

Q3 State management In flutter :-

=> - State management is crucial in flutter application because it involves managing the data that can change overtime.
- flutter is reactive meaning the UI rebuilds when the underlying data changes

Set state	provider	- Riverpod.
Built in flutter method	Internal package named ('provider')	External package
local state within a widget	Global state within widgettree	Global state with additional features
Limited scalability for large apps	suitable for medium sized apps	Designed for large & complex apps
may lead to code redundancy	Balances simplicity & readability	Emphasizes readability & clean syntax
Testing can be more challenging	Good testability support	Enhanced testing experience.

Scenario where each is applicable

1) setState
- For small to moderately complex applications
- when managing local state within a widget.

Eg:- Simple forms, UI components with local-UI-specific state.

2) provider
- For medium to large-sized applications
- When a centralized state is needed accessible by multiple widgets

Eg:- managing user authentication, theme changes or app-wide configuration

3) Riverpod
- for large & complex applications
- when testability & maintainability are top priorities.

Eg complex applications with multiple features dynamic UIs.

Q4 Firebase Integration in Flutter:-

=> Integration:

1) Go to Firebase console & create a new project
2) Add Firebase SDK by including dependencies in pubspec.yaml

dependencies:
  firebase_core: ^version
  firebase_auth: ^version
  cloud_firebase: ^version

3) Run flutter pub get
4) Initialise Firebase by calling
• Firebase.initializeApp() in main.

import 'package:firebase_core/firebase_core.dart';

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

Benefits of using firebase as backend

① Real-time Database
• Firebase offers a real-time NoSQL database.

② Authentication
- provides a secure & easy to implement solution for user authentication

③ Cloud Firestore
Firebase Cloud Firestore provides a secure & easy-to-implement scalable NoSQL database that allows you to store & sync data in real time.

④ Hosting
Firebase Hosting provides a simple & efficient way to deploy & host web applications

Data synchronization

① Real-time Database.
- When data changes on one client it triggers events that automatically update data on other clients.

② Cloud Firestore :- It notifies clients about changes allowing for seamless real-time updates

③ Authentication :- If user sign in or out on one device the authentication state is automatically reflected on other devices

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.	5
Name	Hertika Batra
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

DATE:

Name- Hemlina Bafna
Roll-5 , Div- D15A
Batch [A] , Sub:- PWA

Assignment-2

Q1) Define progressive web APP (PWA) & explain its significance in modern web development. Discuss the key characteristic that differentiate PWAs from traditional mobile apps.

A progressive web App (PWA) is a type of web Application that utilizes modern web technologies to provide a user experience similar to that of native mobile apps. PWA's are designed to work on any platform that uses a standard compliant browser. Such as smartphones, tablets & desktops. They offer several significant advantages in modern web development :-

1] Cross-platform compatibility :- PWA's are built with web technologies like HTML, CSS & Javascript making them capable across various platforms & devices.

2] Offline functionality :- one of the key features of PWA's is their ability to work offline or with limited connectivity.

DATE:

App-like Experience :- PWA's provide an app-like experience with features such as push notification, home screen icons & full screen mode blurring the line between web app & native mobile apps

Key features & functionalities that differ PWA from other MOBILE Apps.

1) Easy Installation : PWA can be installed directly from the browser without going through app store.

2) Offline functionality :- PWA can function offline or with limited connectivity by caching content & data

3) Security :- PWA's are served over HTTPS ensuring data security & preventing unauthorized access.

2] Define responsive web design & explain its importance in the context of progressive web Apps compare & contrast responsive fluid & adaptive web design approaches.

→ Responsive web design is an approach to web design aimed at creating websites that provide an optimal viewing experience across a wide range of devices & screen

DATE:

size from desktop computers to smartphones & tablets. It involves using flexible layouts, images & CSS media queries. In context of PWA's responsive web design is crucial because PWA's are designed to work seamlessly across various devices & platforms.

Q3) Describe the lifecycle of service worker including registration, installation & activation.

⇒ A key player in the PWA universe is the "service worker". The service worker is a JavaScript file that runs on a separate thread apart from the one in which your usual website JavaScript files run. The service worker is always up & listening to the incoming responses & the outgoing requests.

Web Apps The lifecycle of a service worker involves:-

1) registration
2) Installation
3) Activation

1) Registration:- The developers use 'navigator.serviceWorker.register' method to register the service worker. This method takes the path to the service worker script as its argument.

The Registeration phase begins when a Javascript file containing the service worker code is registered in the web page. This registration typically occurs in the main Javascript file of the web Application.

2) Installation:-

During Installation the service worker caches essential resources like HTML, CSS & JS files. The download the service worker initializes by listening for events such as fetch & push notifications. In the 'Install' event handler, developers can perform tasks such as caching static assets, performing other setup operations. If the Installation is useful the service worker moves on to the activation phase.

③ Activation :-

After the Installation is completed the service worker enters the activation phase. During activation the new service worker becomes active & takes control of any clients under its scope. If there is one is then replace by the new one. This process is known as the "updated" process.

DATE:

Q4) Explain the use of IndexedDB in the service workers for data storage.

=> IndexedDB is a browser-based database that allows web applications including progressive web apps to store & retrive large amount of structured data.

1] Initialization - PWA's typically initialize IndexedDB when the application loads or when certain conditions are met. This involves opening a connection to the database.

2] Data Storage :- PWA's can then store data in the IndexedDB by adding updating or deleting records within the defined object stores

3] Data Retrival : When needed, PWA's can request the IndexedDB to retrive data based on certain criteria queries can be performed using indexes for faster retrieval

4] Offline support :- One of the key benefits of using IndexedDB in PWA's is its support for offline data storage PWA's can store data locally in IndexedDB allowing users to access & interact with the app even when they are offline