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 **Vrushabh Pundlik Ghuse_** of **D15A** semester **VI,** have successfully completed necessary experiments in the **MAD & PWA Lab** under my supervision in **VES Institute of Technology** during the academic year **2023-2024.**

Lab Assistant

Subject Teacher
Mrs. Kajal Joseph

Principal

Head of Department
Dr. Mrs. Shalu Chopra

Name of the Course : MAD & PWA Lab
Course Code : ITL604
Year/Sem/Class : D15A/D15B
A.Y.: 23-24

Faculty Incharge : Mrs. Kajal Joseph.

Lab Teachers **:** Mrs. Kajal Jewani.

Email **:** kajal.jewani@ves.ac.in

Programme Outcomes: The graduate will be able to:

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

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

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

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

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

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

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

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

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

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

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

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

Program specific Outcomes

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

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

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	113/26/1	20/2	13 M
2.	To design Flutter UI by including common widgets.	LO2	23/1	30/1	15 M
3.	To include icons, images, fonts in Flutter app	LO2	30/1	6/2	15 M
4.	To create an interactive Form using form widget	LO2	6/2	13/2	15 M
5.	To apply navigation, routing and gestures in Flutter App	LO2	13/2	20/2	15 M
6.	To Connect Flutter UI with fireBase database	LO3	20/2	5/3	15 M
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 M
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 M
9.	To implement Service worker events like fetch, sync and push for E-commerce PWA	LO5	19/3	26/3	15 M
10.	To study and implement deployment of Ecommerce PWA to GitHub Pages.	LO5	26/3	2/4	15 M
11.	To use google Lighthouse PWA Analysis Tool to test the PWA functioning.	LO6	5/3	12/3	15 M
12.	Assignment-1	LO1,LO2,LO3	2/2	5/2	5 M
13.	Assignment-2	LO4,LO5,LO6	19/3	21/3	4 M

MAD & PWA Lab

Journal

Experiment No.	01
Experiment Title.	To install and configure the Flutter Environment
Roll No.	04
Name	Vrushabh Ghuse
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO1: Understand cross platform mobile application development using Flutter framework
Grade:	15 M

Name : Vrushabh Ghuse **Roll no : 22** **D15A**

Experiment No : 1

Aim : To install and configure the Flutter Environment

Theory:

Flutter is an open-source UI software development toolkit created by Google. It is used for building natively compiled applications for mobile, web, and desktop from a single codebase. Flutter uses the Dart programming language and follows a reactive programming paradigm.

Key Concepts and Theoretical Aspects:

Widget Tree:

In Flutter, everything is a widget. Widgets are the basic building blocks of the user interface.
The entire UI is represented as a tree of widgets. Each widget can be a structural element (like a button or text) or a layout (like a row or column). Declarative UI:

Flutter follows a declarative approach, where the UI is described in terms of what it should look like based on the current state. Developers declare the desired state, and Flutter takes care of updating the UI to reflect that state. Hot Reload:

One of Flutter's key features is Hot Reload, which allows developers to instantly see the impact of the code changes without restarting the entire application. This accelerates the development process and enhances productivity. State Management:

Flutter provides various options for managing the state of an application. Understanding and implementing effective state management is crucial for building scalable and maintainable Flutter applications. Dart Language:

Flutter uses Dart as its programming language. Understanding the Dart language and its features is essential for efficient Flutter development. Material Design and Cupertino Widgets:

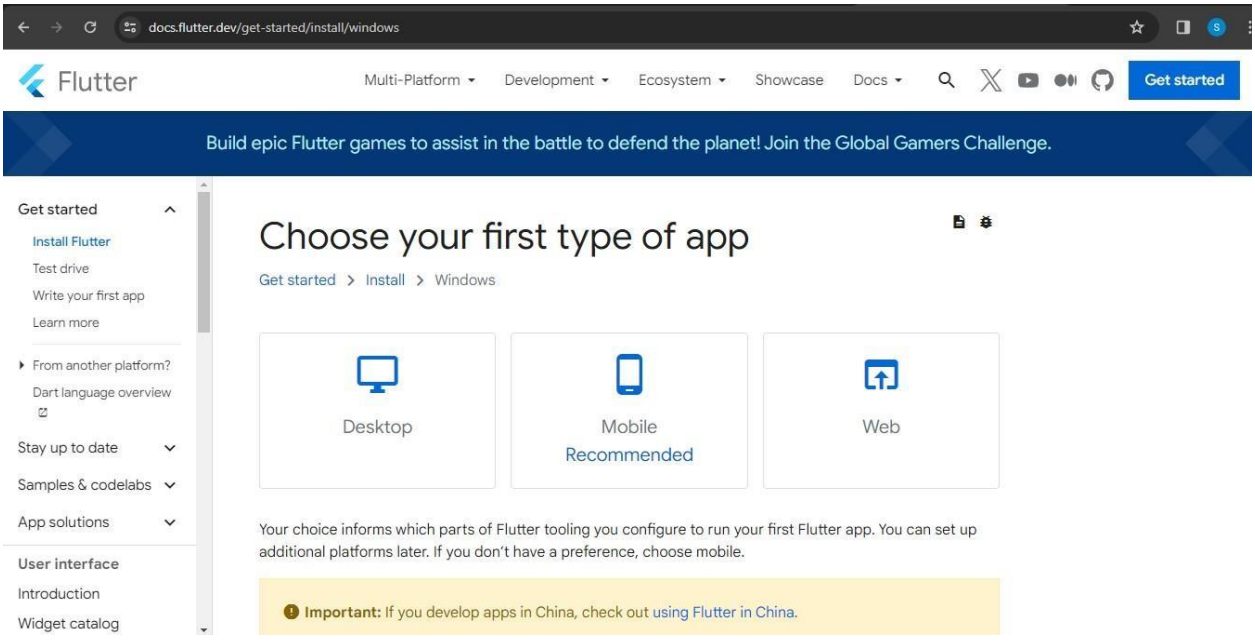
Flutter provides a set of widgets that implement the Material Design guidelines for Android and iOS-style widgets for iOS through the Cupertino library. Understanding

how to use these widgets helps in creating platform-specific user interfaces.
Asynchronous Programming:

Dart is designed to support asynchronous programming, which is important for handling tasks like network requests without blocking the UI thread. Understanding asynchronous programming concepts is crucial for developing responsive Flutter applications.

Installation of Flutter:

Step 1: Download the installation bundle of the Flutter Software Development Kit for windows.
To download Flutter SDK, Go to its official website https://docs.flutter.dev/get-started/install ,
you will get the following screen.



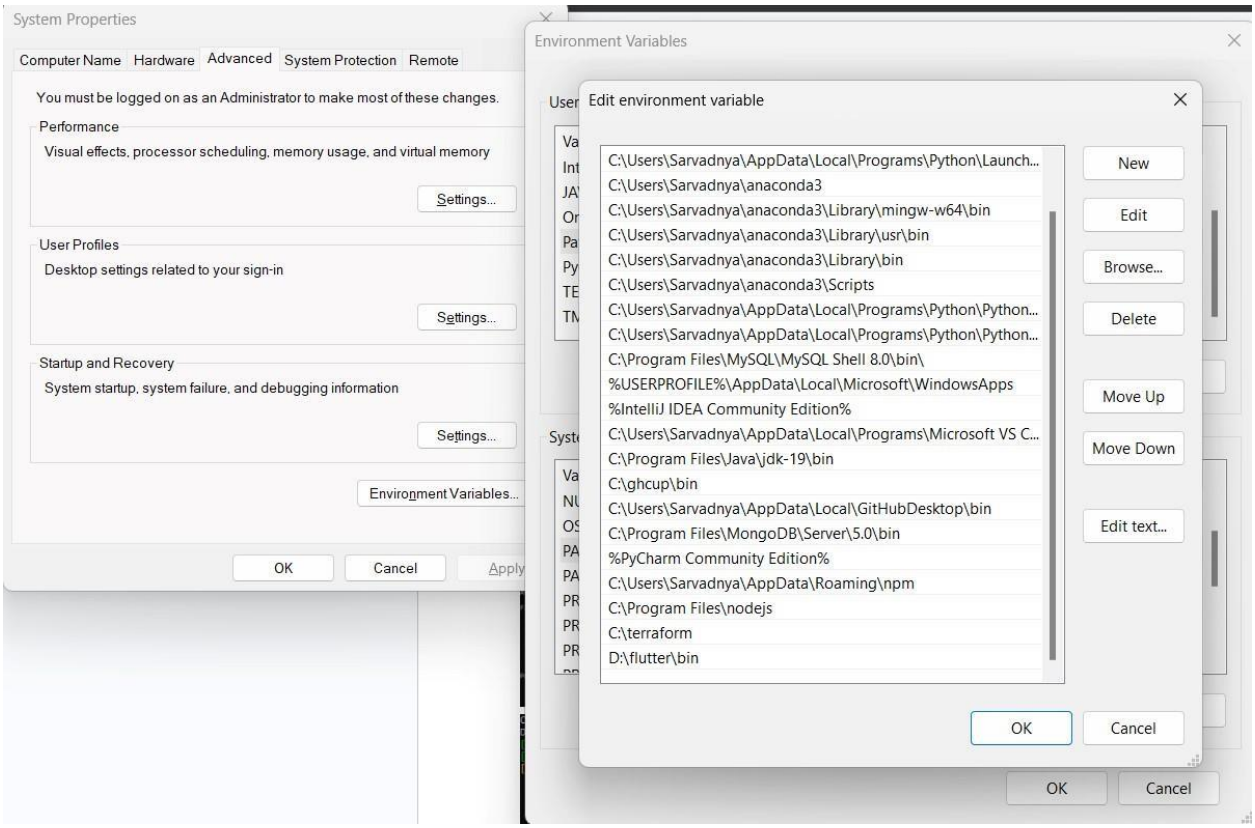
Step 2: Next, to download the latest Flutter SDK, click on the Windows icon. Here, you will find the download link for SDK.

Step 3: When your download is complete, extract the zip file and place it in the desired installation folder or location, for example, C: /Flutter.

Step 4: To run the Flutter command in regular windows console, you need to update the system path to include the flutter bin directory. The following steps are required to do this:

Step 4.1: Go to MyComputer properties -> advanced tab -> environment variables. You will get the following screen.

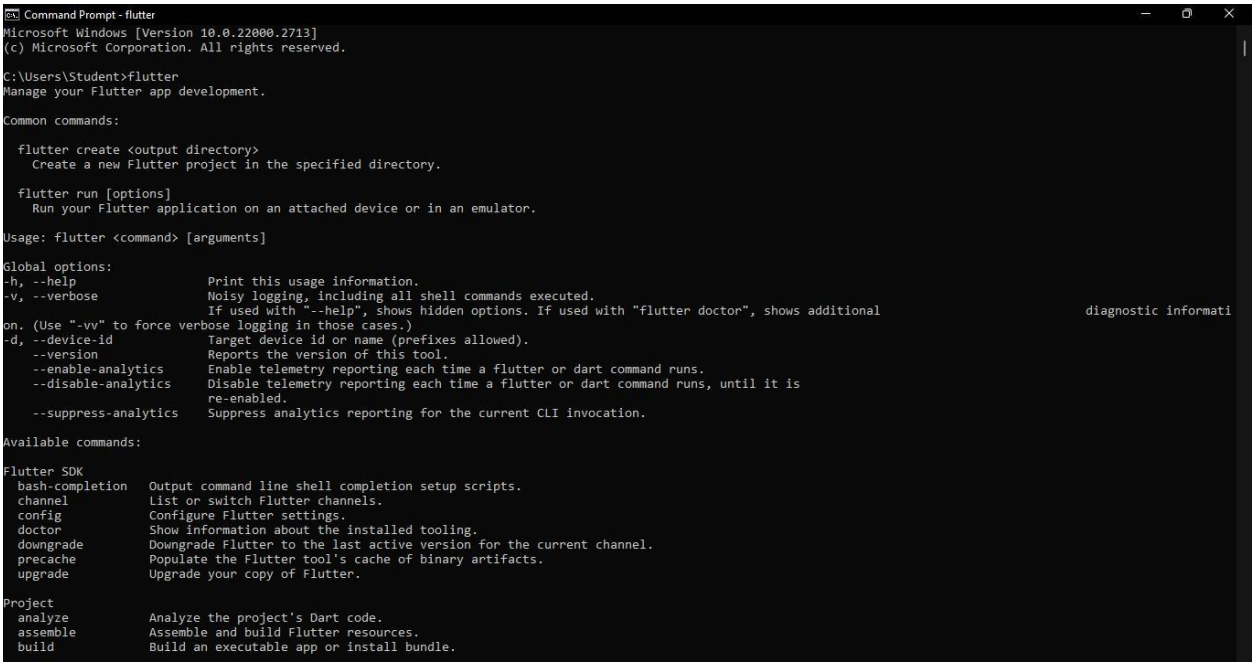
Editing Environment Variables:

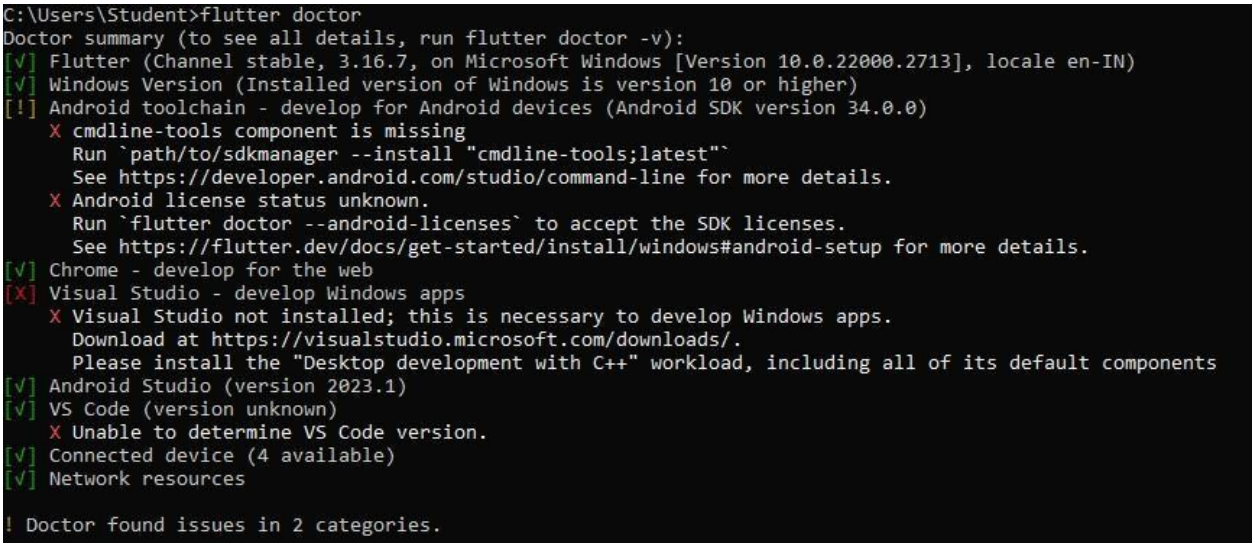


Step 4.3: In the above window, click on New->write path of Flutter bin folder in variable value -

> ok -> ok -> ok.

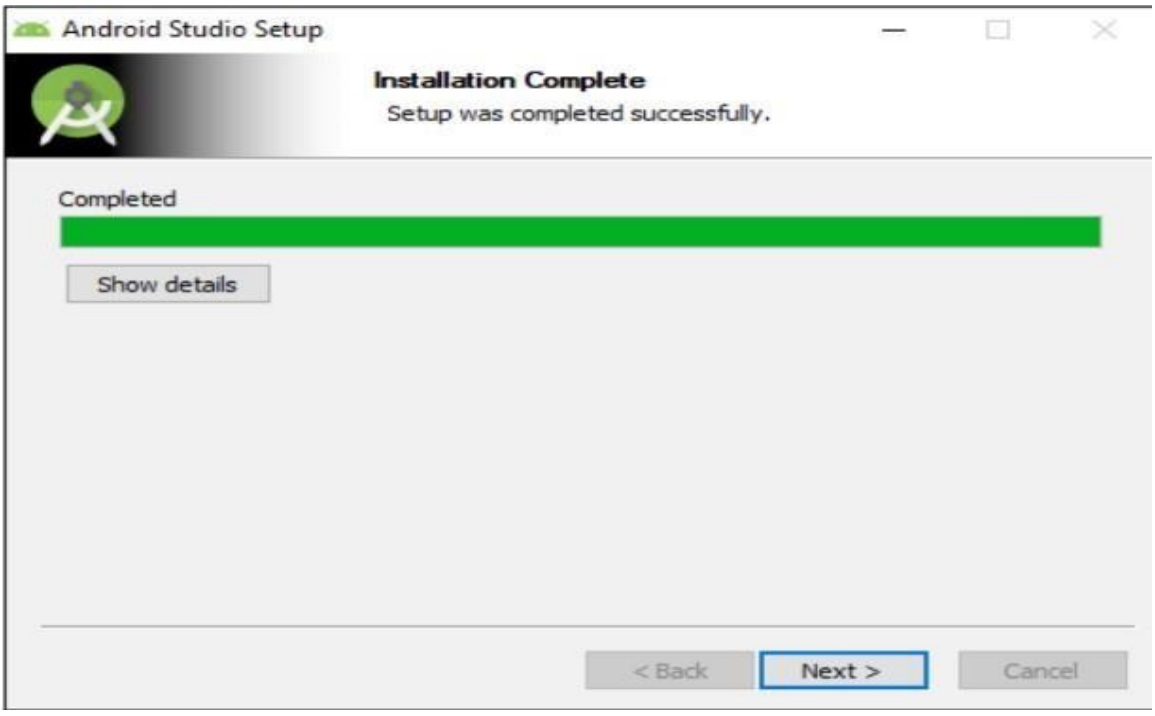
Step 5: Now, run the $ flutter command in command prompt.



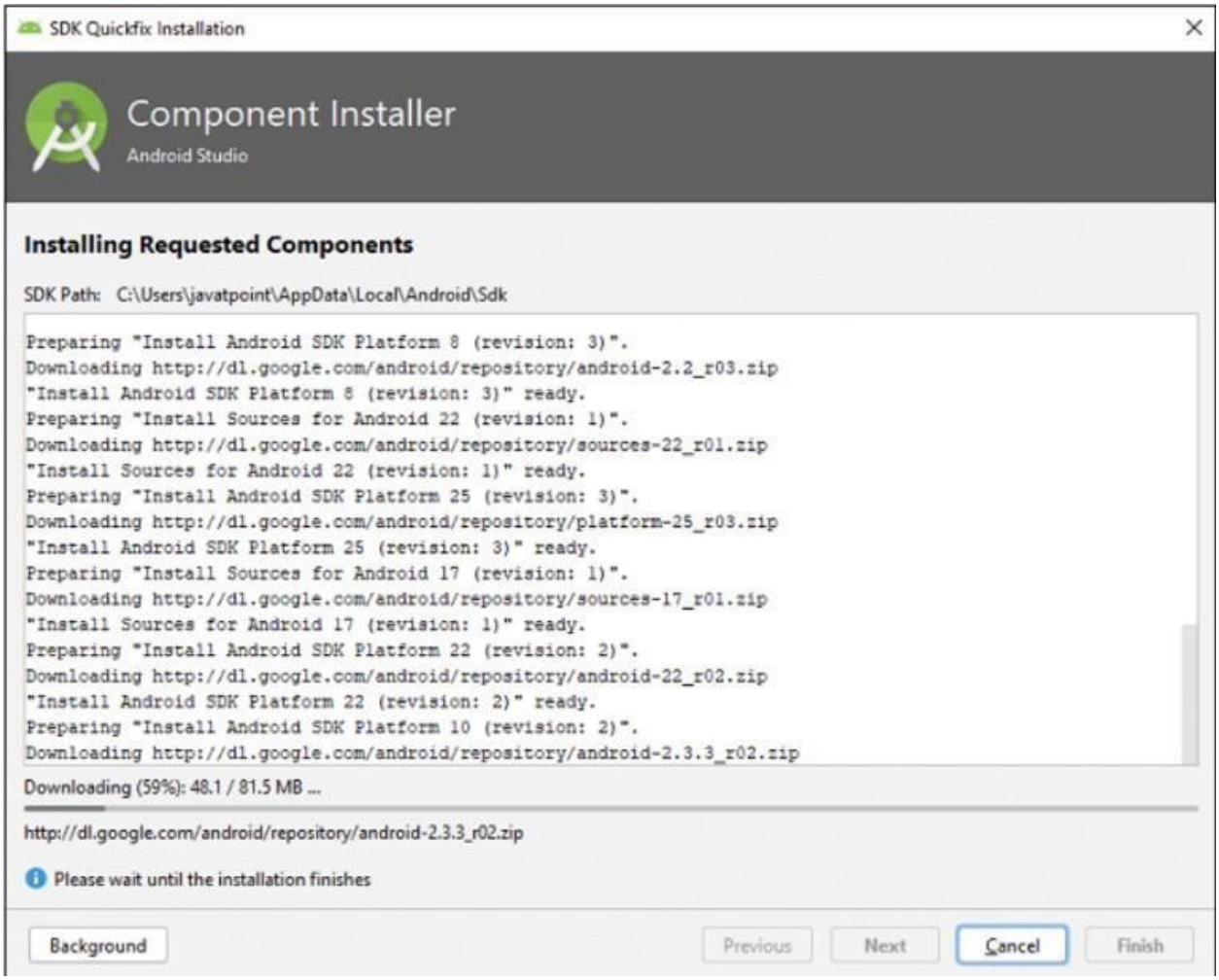


Installation of Android Studio:

Step : Follow the steps of the installation wizard. Once the installation wizard completes, you will get the following screen.

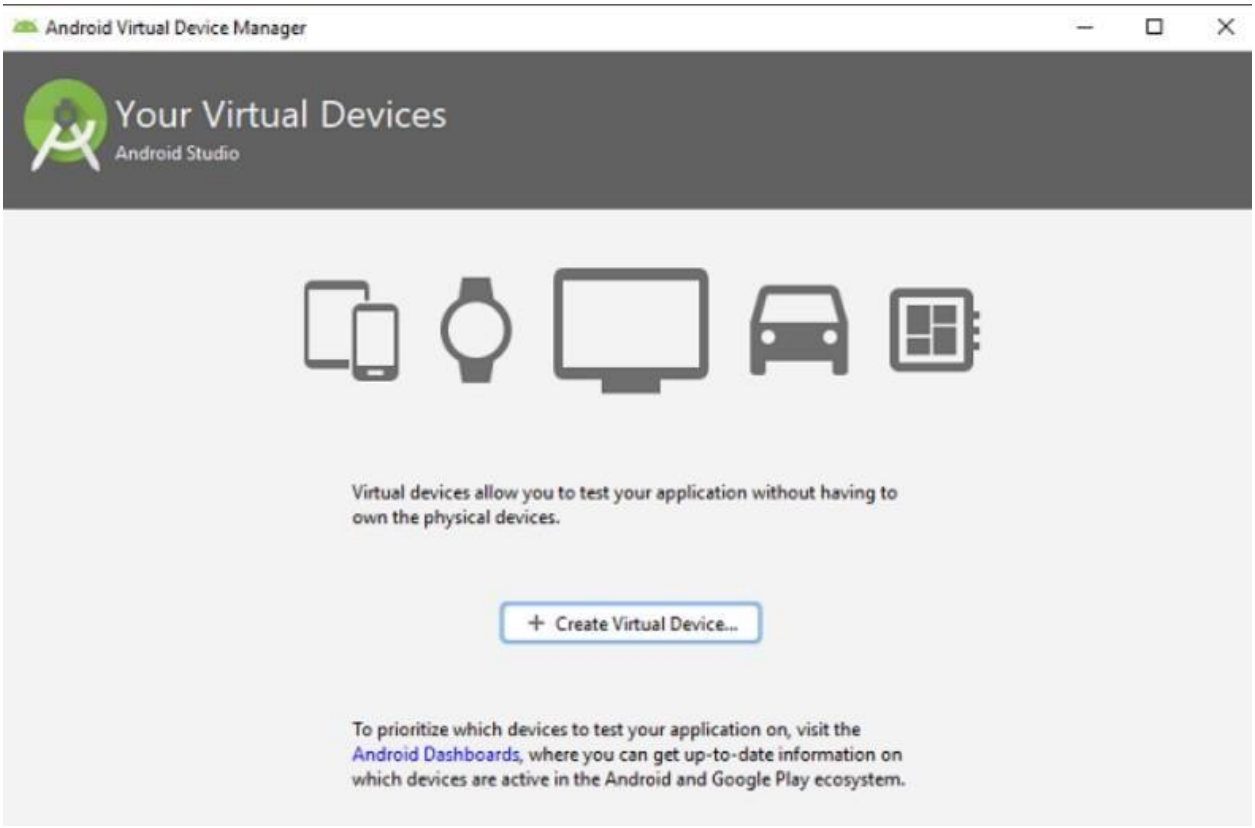


Step : In the above screen, click Next-> Finish. Once the Finish button is clicked, you need to choose the 'Don't import Settings option' and click OK. It will start the Android Studio.

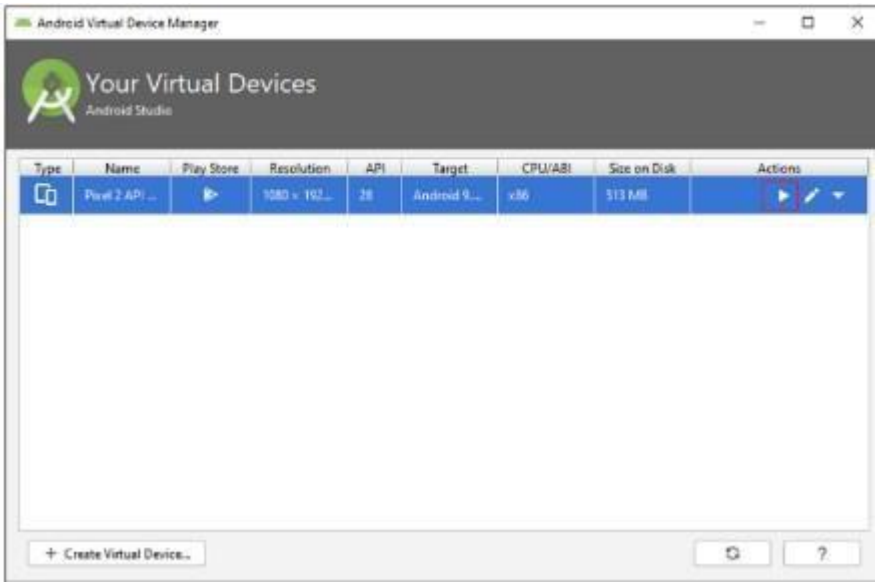


Step 8: Next, you need to set up an Android emulator. It is responsible for running and testing the Flutter application.

Step 8..2: To set an Android emulator, go to Android Studio > Tools > Android > AVD Managerand select Create Virtual Device. Or, go to Help->Find Action->Type Emulator in the search box. You will get the following screen.



Step 8.2: Choose your device definition and click on Next.
Step 8.3: Select the system image for the latest Android version and click on Next. Step 8.4: Now, verify the all AVD configuration. If it is correct, click on Finish. The following screen appears.

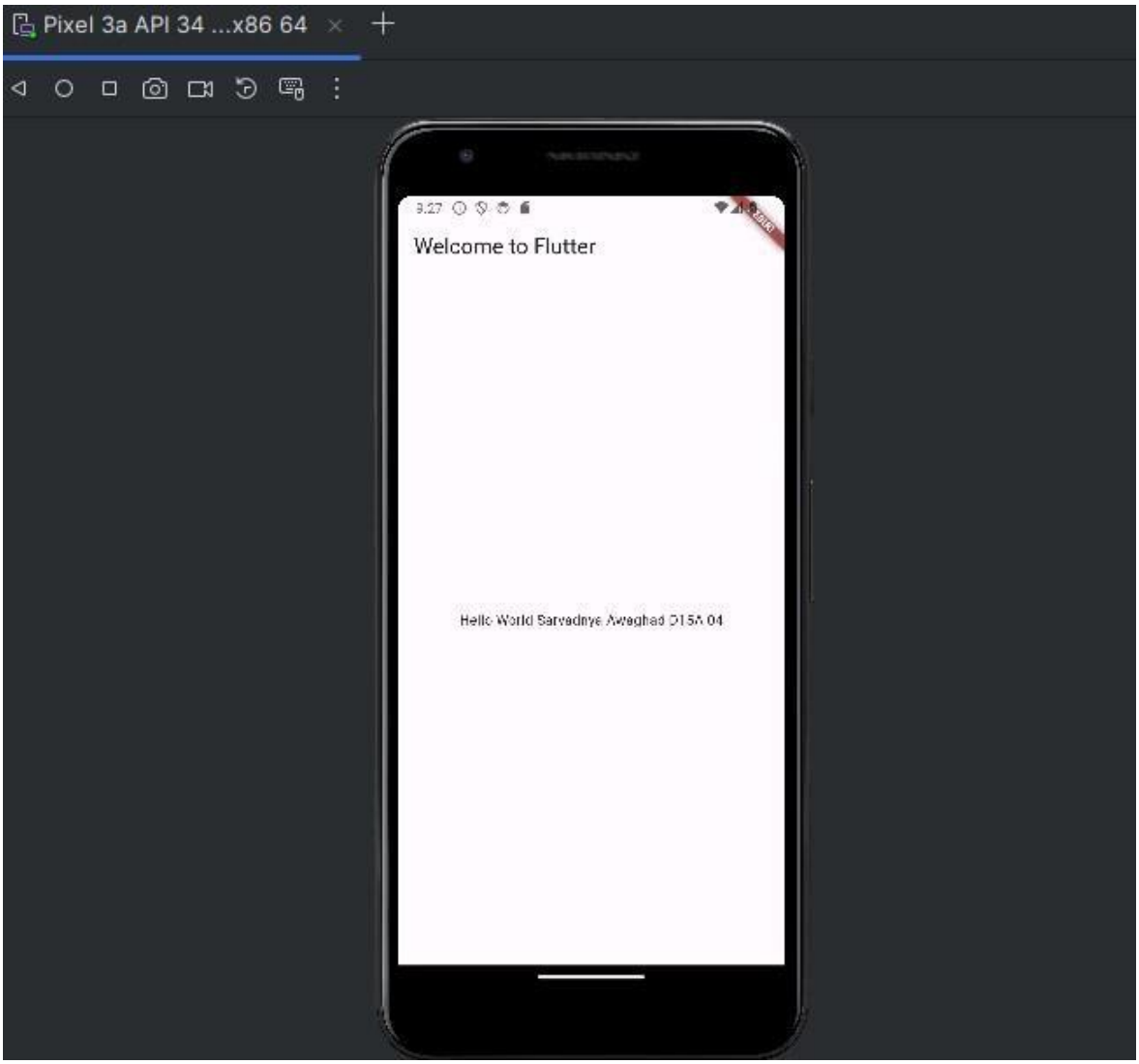


Step 8.5: Last, click on the icon pointed into the red color rectangle. The Android emulator displayed as below screen.

Flutter Hello World App :

```
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 World \n Sarvadnya
Awaghad D15A 04'), ),
      ),
    );
  }
}
```



Conclusion : In the above experiment , we have successfully configured and installed flutter and android studio in our desktop. AVD emulator is installed and flutter app is executed successfully.

MAD & PWA Lab

Journal

Experiment No.	**02**
Experiment Title.	**To design Flutter UI by including common widgets.**
Roll No.	**22**
Name	**Vrushabh Ghuse**
Class	**D15A**
Subject	**MAD & PWA Lab**
Lab Outcome	**LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation**
Grade:	**15 M**

MAD and PWA Lab

Name: Vrushabh Ghuse **Class: D15A** **Roll no:22**

Aim: To design Flutter UI by including common widgets.

Theory:

Flutter widgets are the building blocks of the user interface, representing visual and interactive elements. Widgets can be categorized into two main types: Stateless and Stateful. Stateless widgets are immutable and do not store any internal state, while Stateful widgets can change and maintain their state over time.

Built-in and Custom Widgets:
Flutter comes with a rich set of built-in widgets that cover a wide range of UI elements and interactions. Additionally, developers can create their custom widgets to tailor the user interface to specific requirements. This section explores the anatomy of custom widgets and their integration into the Flutter framework.

Layout and Constraints:
Flutter's layout system is based on the concept of constraints, allowing widgets to adapt to various screen sizes and orientations. Understanding how widgets handle layout constraints is essential for creating responsive and adaptive user interfaces.

Animation with Widgets:
Flutter provides a robust animation framework that seamlessly integrates with widgets, enabling the creation of fluid and engaging user experiences. This section explores how animations can be incorporated into Flutter widgets to enhance the overall user interface.

Testing and Debugging Widgets:
As with any software development, testing and debugging are integral parts of the process. This section discusses strategies for testing and debugging Flutter widgets, ensuring the reliability and quality of the UI components.
Container:

The Container widget is a basic building block that can contain other widgets and is often used to define the dimensions, padding, margin, and decoration of a UI element.

Text:
The Text widget is used to display a paragraph or a line of text. It supports various styling options such as font size, color, and alignment.

ListView:
The ListView widget is used to create a scrollable list of widgets. It can display a large number of children efficiently.

Row and Column:
Row and Column widgets are used to arrange children in a horizontal or vertical line, respectively.

Stack:
The Stack widget allows you to overlay multiple widgets on top of each other. It's often used for complex layouts.

AppBar:
The AppBar widget represents the top app bar that usually contains the app's title, icons, and actions.

Code: home_view.dart :

```
import 'package:dotted_dashed_line/dotted_dashed_line.dart';
import 'package:fitness/common_widget/round_button.dart';
import 'package:fitness/common_widget/workout_row.dart';
import 'package:fl_chart/fl_chart.dart';
import 'package:flutter/material.dart';
import
'package:simple_animation_progress_bar/simple_animation_progress_bar.dart';
import 'package:simple_circular_progress_bar/simple_circular_progress_bar.dart';
import '../../common/colo_extension.dart'; import 'activity_tracker_view.dart'; import
'finished_workout_view.dart'; import 'notification_view.dart';

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

  @override
  State<HomeView> createState() => _HomeViewState();
}
```

```
class _HomeViewState extends State<HomeView> {
  List lastWorkoutArr = [
    {
      "name": "Full Body Workout",
      "image": "assets/img/Workout1.png",
      "kcal": "180",
      "time": "20",
      "progress": 0.3
    },
    {
      "name": "Lower Body Workout",
      "image": "assets/img/Workout2.png",
      "kcal": "200",
      "time": "30",
      "progress": 0.4
    },
    {
      "name": "Ab Workout",
      "image": "assets/img/Workout3.png",
      "kcal": "300",
      "time": "40",
      "progress": 0.7
    },
  ];
  List<int> showingTooltipOnSpots = [21];

  List<FlSpot> get allSpots => const [
        FlSpot(0, 20),
        FlSpot(1, 25),
        FlSpot(2, 40),
        FlSpot(3, 50),
        FlSpot(4, 35),
        FlSpot(5, 40),
        FlSpot(6, 30),
        FlSpot(7, 20),
        FlSpot(8, 25),
        FlSpot(9, 40),
        FlSpot(10, 50),
        FlSpot(11, 35),
        FlSpot(12, 50),
        FlSpot(13, 60),
        FlSpot(14, 40),
        FlSpot(15, 50),
        FlSpot(16, 20),
        FlSpot(17, 25),
        FlSpot(18, 40),
        FlSpot(19, 50),
        FlSpot(20, 35),
        FlSpot(21, 80),
        FlSpot(22, 30),
        FlSpot(23, 20),
        FlSpot(24, 25),
```

```
      FlSpot(25, 40),
      FlSpot(26, 50),
      FlSpot(27, 35),
      FlSpot(28, 50),
      FlSpot(29, 60),
      FlSpot(30, 40)
    ];

 List waterArr = [
   {"title": "6am - 8am", "subtitle": "600ml"},
   {"title": "9am - 11am", "subtitle": "500ml"},
   {"title": "11am - 2pm", "subtitle": "1000ml"},
   {"title": "2pm - 4pm", "subtitle": "700ml"},
   {"title": "4pm - now", "subtitle": "900ml"},
 ];

 @override
 Widget build(BuildContext context) {
var media = MediaQuery.of(context).size;

   final lineBarsData = [
LineChartBarData(
      showingIndicators:
showingTooltipOnSpots,        spots: allSpots,
isCurved: false,        barWidth: 3,
belowBarData: BarAreaData(          show: true,
gradient: LinearGradient(colors: [
TColor.primaryColor2.withOpacity(0.4),
          TColor.primaryColor1.withOpacity(0.1),
        ], begin: Alignment.topCenter, end: Alignment.bottomCenter),
      ),
      dotData: FlDotData(show: false),
gradient: LinearGradient(
        colors: TColor.primaryG,
      ),
    ),
   ];

   final tooltipsOnBar = lineBarsData[0];

   return Scaffold(
backgroundColor: TColor.white,
body: SingleChildScrollView(
child: SafeArea(          child:
Padding(
          padding: const EdgeInsets.symmetric(horizontal: 15),
child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
Row(
              mainAxisAlignment:
MainAxisAlignment.spaceBetween,                children: [
Column(
```

```
                crossAxisAlignment: CrossAxisAlignment.start,
children: [
                  Text(
                    "Welcome Back,",
                    style: TextStyle(color: TColor.gray, fontSize: 12),
                  ),
                  Text(
                    "Vrushabh Ghuse",
style: TextStyle(
color: TColor.black,
                      fontSize: 20,
                      fontWeight: FontWeight.w700),
                  ),
                ],
              ),
              IconButton(
onPressed: () {
Navigator.push(
context,
                  MaterialPageRoute(
                    builder: (context) => const NotificationView(),
                  ),
                );
},
                  icon: Image.asset(
                    "assets/img/notification_active.png",
                    width: 25,
height: 25,                    fit:
BoxFit.fitHeight,
                  ))
            ],
          ),
          SizedBox(
            height: media.width * 0.05,
          ),
          Container(                height: media.width * 0.4,
decoration: BoxDecoration(                    gradient:
LinearGradient(colors: TColor.primaryG),                    borderRadius:
BorderRadius.circular(media.width * 0.075)),                child:
Stack(alignment: Alignment.center, children: [
              Image.asset(
                "assets/img/bg_dots.png",
height: media.width * 0.4,
width: double.maxFinite,
fit: BoxFit.fitHeight,
              ),
              Padding(
                padding: const
EdgeInsets.symmetric(                    vertical:
25, horizontal: 25),                child: Row(
                mainAxisAlignment:
MainAxisAlignment.spaceBetween,                    children: [
Column(
```

```
                    mainAxisAlignment: MainAxisAlignment.center,
crossAxisAlignment: CrossAxisAlignment.start,
children: [
                    Text(
                      "BMI (Body Mass
Index)",                          style: TextStyle(
color: TColor.white,
fontSize: 14,
                          fontWeight: FontWeight.w700),
                    ),
                    Text(
                      "You have a normal weight",
                      style: TextStyle(
                        color: TColor.white.withOpacity(0.7),
                        fontSize: 12),
                    ),
                    SizedBox(
                      height: media.width * 0.05,
                    ),
SizedBox(                             width:
120,                          height: 35,
child: RoundButton(
title: "View More",
                          type: RoundButtonType.bgSGradient,
                          fontSize: 12,
                          fontWeight: FontWeight.w400,
                          onPressed: () {}))
                  ],
                ),
                AspectRatio(
aspectRatio: 1,                        child: PieChart(
PieChartData(                            pieTouchData:
PieTouchData(
touchCallback:
                          (FlTouchEvent event, pieTouchResponse) {},
                      ),
                      startDegreeOffset: 250,
borderData: FlBorderData(
                        show: false,
                      ),
sectionsSpace: 1,
centerSpaceRadius: 0,
                      sections: showingSections(),
                    ),
                  ),
                ),
              ],
            ),
          )
        ]),
      ),
      SizedBox(
        height: media.width * 0.05,
```

```
            ),
            Container(
padding:
                const EdgeInsets.symmetric(vertical: 15, horizontal: 15),
decoration: BoxDecoration(
                color: TColor.primaryColor2.withOpacity(0.3),
                borderRadius: BorderRadius.circular(15),
              ),
child: Row(
                mainAxisAlignment: MainAxisAlignment.spaceBetween,
children: [
                Text(
"Today Target",                     style:
TextStyle(                        color:
TColor.black,
                    fontSize: 14,
                    fontWeight: FontWeight.w700),
                ),
                SizedBox(
width: 70,                    height:
25,                   child:
RoundButton(
                    title: "Check",
                    type:
RoundButtonType.bgGradient,
fontSize: 12,                       fontWeight:
FontWeight.w400,
                    onPressed: () {
Navigator.push(
context,
                      MaterialPageRoute(
builder: (context) =>
                          const ActivityTrackerView(),
                      ),
                    );
                  },
                ),
              )
            ],
          ),
        ),
        SizedBox(
          height: media.width * 0.05,
        ),
        Text(
          "Activity Status",
style: TextStyle(
color: TColor.black,
                fontSize: 16,
                fontWeight: FontWeight.w700),
        ),
        SizedBox(
          height: media.width * 0.02,
```

```
            ),
            ClipRRect(
              borderRadius: BorderRadius.circular(25),
              child: Container(                    height:
media.width * 0.4,                  width: double.maxFinite,
decoration: BoxDecoration(                    color:
TColor.primaryColor2.withOpacity(0.3),
borderRadius: BorderRadius.circular(25),
                ),                  child: Stack(
alignment: Alignment.topLeft,
                  children: [
Padding(
                    padding: const EdgeInsets.symmetric(
vertical: 20, horizontal: 20),                      child: Column(
                      mainAxisAlignment: MainAxisAlignment.start,
mainAxisSize: MainAxisSize.min,
crossAxisAlignment: CrossAxisAlignment.start,
                      children: [
                        Text(
"Heart Rate",                          style:
TextStyle(                          color:
TColor.black,
                            fontSize: 16,
                            fontWeight: FontWeight.w700),
                        ),
                        ShaderMask(
blendMode: BlendMode.srcIn,
shaderCallback: (bounds) {
return LinearGradient(
colors: TColor.primaryG,
begin: Alignment.centerLeft,
end: Alignment.centerRight)
.createShader(Rect.fromLTRB(
                              0, 0, bounds.width, bounds.height));
                          },                          child: Text(
"78 BPM",                          style: TextStyle(
color: TColor.white.withOpacity(0.7),
fontWeight: FontWeight.w700,
                              fontSize: 18),
                        ),
                      ),
                    ],
                  ),
                ),
                LineChart(                    LineChartData(
showingTooltipIndicators:
showingTooltipOnSpots.map((index) {
return ShowingTooltipIndicators([
LineBarSpot(                          tooltipsOnBar,
                      lineBarsData.indexOf(tooltipsOnBar),
tooltipsOnBar.spots[index],
                        ),
                      ]);
                    }).toList(),
```

```
                    lineTouchData: LineTouchData(
enabled: true,
handleBuiltInTouches: false,
touchCallback: (FlTouchEvent event,
LineTouchResponse? response) {                          if
(response == null ||
response.lineBarSpots == null) {
                        return;
                      }
                      if (event is FlTapUpEvent) {
final spotIndex =
response.lineBarSpots!.first.spotIndex;
showingTooltipOnSpots.clear();
                        setState(() {
                          showingTooltipOnSpots.add(spotIndex);
                        });
                      }
                    },
                    mouseCursorResolver: (FlTouchEvent
event,                      LineTouchResponse? response) {
if (response == null ||
response.lineBarSpots == null) {                          return
SystemMouseCursors.basic;
                      }
                      return SystemMouseCursors.click;
                    },
                    getTouchedSpotIndicator:
(LineChartBarData barData,
List<int> spotIndexes) {                          return
spotIndexes.map((index) {
                        return
TouchedSpotIndicatorData(
FlLine(                          color: Colors.red,
                          ),
                          FlDotData(
show: true,
getDotPainter:
                              (spot, percent, barData, index)
=>                              FlDotCirclePainter(
radius: 3,                          color: Colors.white,
strokeWidth: 3,
                            strokeColor: TColor.secondaryColor1,
                          ),
                        ),
                      );
                    }).toList();
                    },
                    touchTooltipData: LineTouchTooltipData(
tooltipBgColor: TColor.secondaryColor1,
tooltipRoundedRadius: 20,
getTooltipItems:
                        (List<LineBarSpot> lineBarsSpot) {
return lineBarsSpot.map((lineBarSpot) {
return LineTooltipItem(
```

```
                            "${lineBarSpot.x.toInt()} mins ago",
                            const TextStyle(
color: Colors.white,
fontSize: 10,
                              fontWeight: FontWeight.bold,
                            ),
                          );
                        }).toList();
                      },
                    ),
                  ),
                  lineBarsData: lineBarsData,
minY: 0,
                  maxY: 130,
titlesData: FlTitlesData(
show: false,
                  ),
                  gridData: FlGridData(show: false),
borderData: FlBorderData(
                    show: true,
border: Border.all(
color: Colors.transparent,
                    ),
                  ),
                ),
              )
            ],
          ),
        ),
      ),
      SizedBox(
        height: media.width * 0.05,
      ),
      Row(              children: [
Expanded(                 child: Container(
height: media.width * 0.95,                   padding:
const EdgeInsets.symmetric(
vertical: 25, horizontal: 20),
decoration: BoxDecoration(                      color:
Colors.white,
                  borderRadius: BorderRadius.circular(25),
                  boxShadow: const [
                    BoxShadow(color: Colors.black12, blurRadius: 2)
                  ]),
child: Row(
                children: [
                  SimpleAnimationProgressBar(
height: media.width * 0.85,                       width:
media.width * 0.07,                     backgroundColor:
Colors.grey.shade100,
                    foregrondColor: Colors.purple,
                    ratio: 0.5,                       direction:
Axis.vertical,                   curve:
```

```
Curves.fastLinearToSlowEaseIn,
duration: const Duration(seconds: 3),
borderRadius: BorderRadius.circular(15),
gradientColor: LinearGradient(
colors: TColor.primaryG,                    begin:
Alignment.bottomCenter,
                    end: Alignment.topCenter),
                ),
const SizedBox(
width: 10,                    ),
                Expanded(
                  child: Column(
                   crossAxisAlignment: CrossAxisAlignment.start,
children: [
                    Text(
                     "Water Intake",
style: TextStyle(
color: TColor.black,
                        fontSize: 12,
                        fontWeight: FontWeight.w700),
                    ),
                    ShaderMask(
blendMode: BlendMode.srcIn,
shaderCallback: (bounds) {
return LinearGradient(
colors: TColor.primaryG,
begin: Alignment.centerLeft,
end: Alignment.centerRight)
.createShader(Rect.fromLTRB(
                            0, 0, bounds.width, bounds.height));
                     },                          child: Text(
"4 Liters",                    style: TextStyle(
color: TColor.white.withOpacity(0.7),
fontWeight: FontWeight.w700,
                        fontSize: 14),
                    ),
                    ),
const SizedBox(
height: 10,
                    ),
                    Text(
                     "Real time updates",
style: TextStyle(
color: TColor.gray,
fontSize: 12,
                    ),
                    ),
                    Column(
                     crossAxisAlignment:
CrossAxisAlignment.start,                    children:
waterArr.map((wObj) {                    var isLast = wObj ==
waterArr.last;
```

```
                        return Row(
crossAxisAlignment:
CrossAxisAlignment.start,
                          children: [
Column(
mainAxisAlignment:
MainAxisAlignment.start,
crossAxisAlignment:
CrossAxisAlignment.center,
                            children: [
Container(
margin:
                                const EdgeInsets.symmetric(
                                  vertical: 4),
width: 10,
height: 10,
                              decoration: BoxDecoration(
color: TColor.secondaryColor1
.withOpacity(0.5),
borderRadius:
                                  BorderRadius.circular(5),
                              ),
),                             if (!isLast)
DottedDashedLine(                                 height:
media.width * 0.078,
                                width: 0,
dashColor: TColor
.secondaryColor1
.withOpacity(0.5),
axis: Axis.vertical)
                            ],                                ),
const SizedBox(                             width: 10,
),                           Column(
mainAxisAlignment:
MainAxisAlignment.start,
crossAxisAlignment:
CrossAxisAlignment.start,
                            children: [
Text(
wObj["title"].toString(),
style: TextStyle(
color: TColor.gray,
fontSize: 10,
                                ),
                              ),
                              ShaderMask(
blendMode: BlendMode.srcIn,
shaderCallback: (bounds) {
return LinearGradient(
colors:

TColor.secondaryG,
begin: Alignment
.centerLeft,                                  end:
```

```
Alignment
.centerRight)
                                    .createShader(Rect.fromLTRB(
                                       0,
0,
bounds.width,
bounds.height));
                                 },
child: Text(
wObj["subtitle"].toString(),
style: TextStyle(                                    color:
TColor.white
.withOpacity(0.7),
fontSize: 12),
                                 ),
                               ),
                             ],
                           )
                         ],
                        );
                       }).toList(),
                     )
                    ],
                   ))
                  ],
                 ),
                ),
               ),
               SizedBox(
                 width: media.width * 0.05,
               ),
               Expanded(
                  child: Column(
                  mainAxisAlignment:
MainAxisAlignment.start,                mainAxisSize:
MainAxisSize.min,                 children: [
Container(                    width: double.maxFinite,
height: media.width * 0.45,                      padding: const
EdgeInsets.symmetric(                         vertical: 25,
horizontal: 20),                    decoration: BoxDecoration(
color: Colors.white,
                    borderRadius: BorderRadius.circular(25),
                    boxShadow: const [
                      BoxShadow(color: Colors.black12, blurRadius: 2)
                    ]),
child: Column(
                    crossAxisAlignment:
CrossAxisAlignment.start,                      children: [
Text(                       "Sleep",                      style:
TextStyle(                          color: TColor.black,
                        fontSize: 12,
                        fontWeight: FontWeight.w700),
                      ),
```

```
                        ShaderMask(
blendMode: BlendMode.srcIn,
shaderCallback: (bounds) {
return LinearGradient(
colors: TColor.primaryG,
begin: Alignment.centerLeft,
end: Alignment.centerRight)
.createShader(Rect.fromLTRB(
                                0, 0, bounds.width, bounds.height));
                            },
child: Text(                                "8h
20m",                               style:
TextStyle(
                                color: TColor.white.withOpacity(0.7),
fontWeight: FontWeight.w700,
                                fontSize: 14),
                            ),
                          ),
                          const Spacer(),
                          Image.asset("assets/img/sleep_grap.png",
                             width: double.maxFinite,
fit: BoxFit.fitWidth)
                        ]),
                    ),
                    SizedBox(
                      height: media.width * 0.05,
                    ),                      Container(
width: double.maxFinite,                            height:
media.width * 0.45,                         padding: const
EdgeInsets.symmetric(                               vertical:
25, horizontal: 20),                        decoration:
BoxDecoration(                              color:
Colors.white,
                        borderRadius: BorderRadius.circular(25),
                        boxShadow: const [
                          BoxShadow(color: Colors.black12, blurRadius: 2)
                        ]),
child: Column(
                        crossAxisAlignment:
CrossAxisAlignment.start,                           children: [
Text(                               "Calories",                             style:
TextStyle(                                  color: TColor.black,
                                fontSize: 12,
                                fontWeight: FontWeight.w700),
                          ),
                          ShaderMask(
blendMode: BlendMode.srcIn,
shaderCallback: (bounds) {
return LinearGradient(
colors: TColor.primaryG,
begin: Alignment.centerLeft,
end: Alignment.centerRight)
.createShader(Rect.fromLTRB(
                                0, 0, bounds.width, bounds.height));
```

```
                        },
child: Text(
"760 kCal",
style: TextStyle(
                            color: TColor.white.withOpacity(0.7),
fontWeight: FontWeight.w700,
                            fontSize: 14),
                        ),
),
                    const Spacer(),
Container(
                     alignment:
Alignment.center,                           child:
SizedBox(                           width:
media.width * 0.2,                          height:
media.width * 0.2,                          child:
Stack(
                        alignment: Alignment.center,
children: [
                            Container(
width: media.width * 0.15,                              height:
media.width * 0.15,                             alignment:
Alignment.center,                               decoration:
BoxDecoration(                                  gradient:
LinearGradient(                                 colors:
TColor.primaryG),                               borderRadius:
BorderRadius.circular(
                                    media.width * 0.075),
                                ),
child: FittedBox(                               child:
Text(                               "230kCal\nleft",
textAlign: TextAlign.center,
style: TextStyle(                               color:
TColor.white,                               fontSize:
11),
                                ),
                              ),
                            ),
                            SimpleCircularProgressBar(
progressStrokeWidth: 10,
backStrokeWidth: 10,
progressColors: TColor.primaryG,
backColor: Colors.grey.shade100,
valueNotifier: ValueNotifier(50),
                                startAngle: -180,
                              ),
                            ],
                          ),
                        ),
                      )
                    ]),
                ),
              ],
            ))
```

```
            ],
          ),
          SizedBox(
            height: media.width * 0.1,
          ),
          Row(
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
children: [
              Text(
                "Workout Progress",
style: TextStyle(
color: TColor.black,
                    fontSize: 16,
                    fontWeight: FontWeight.w700),
              ),
              Container(
height: 30,
                padding: const EdgeInsets.symmetric(horizontal: 8),
decoration: BoxDecoration(
                  gradient: LinearGradient(colors: TColor.primaryG),
borderRadius: BorderRadius.circular(15),
                ),
                child: DropdownButtonHideUnderline(
child: DropdownButton(                        items:
["Weekly", "Monthly"]                          .map((name)
=> DropdownMenuItem(                                value:
name,                                child: Text(
name,                                  style: TextStyle(
                                color: TColor.gray, fontSize: 14),
                            ),
                          ))
                      .toList(),
onChanged: (value) {},
                  icon: Icon(Icons.expand_more, color:
TColor.white),                        hint: Text(
"Weekly",                        textAlign: TextAlign.center,
style:
                        TextStyle(color: TColor.white, fontSize: 12),
                  ),
                ),
              )),
            ],
          ),
          SizedBox(
            height: media.width * 0.05,
          ),
          Container(
            padding: const EdgeInsets.only(left:
15),                height: media.width * 0.5,
width: double.maxFinite,                child:
LineChart(                LineChartData(
showingTooltipIndicators:
```

```
                    showingTooltipOnSpots.map((index)
{                    return ShowingTooltipIndicators([
LineBarSpot(                        tooltipsOnBar,
                     lineBarsData.indexOf(tooltipsOnBar),
tooltipsOnBar.spots[index],
                    ),
                   ]);
                  }).toList(),
                  lineTouchData: LineTouchData(
enabled: true,
handleBuiltInTouches: false,
touchCallback: (FlTouchEvent event,
LineTouchResponse? response) {                    if
(response == null ||
response.lineBarSpots == null) {
                     return;
                    }
                    if (event is FlTapUpEvent) {
final spotIndex =
                       response.lineBarSpots!.first.spotIndex;
showingTooltipOnSpots.clear();                    setState(() {
                      showingTooltipOnSpots.add(spotIndex);
                     });
                    }
                  },
                  mouseCursorResolver: (FlTouchEvent
event,                    LineTouchResponse? response) {
if (response == null ||
response.lineBarSpots == null) {
                     return SystemMouseCursors.basic;
                    }
                    return SystemMouseCursors.click;
                  },
                  getTouchedSpotIndicator: (LineChartBarData
barData,                    List<int> spotIndexes) {
return spotIndexes.map((index) {
                     return TouchedSpotIndicatorData(
FlLine(
                       color: Colors.transparent,
                      ),
                      FlDotData(
show: true,
getDotPainter:
                          (spot, percent, barData, index) =>

FlDotCirclePainter(
radius: 3,                          color:
Colors.white,
strokeWidth: 3,
                       strokeColor: TColor.secondaryColor1,
                      ),
                     ),
                    );
                  }).toList();
```

```
                    },
                    touchTooltipData: LineTouchTooltipData(
tooltipBgColor: TColor.secondaryColor1,
tooltipRoundedRadius: 20,
                      getTooltipItems: (List<LineBarSpot> lineBarsSpot) {
return lineBarsSpot.map((lineBarSpot) {                              return
LineTooltipItem(
                        "${lineBarSpot.x.toInt()} mins ago",
                        const TextStyle(
color: Colors.white,
fontSize: 10,
                          fontWeight: FontWeight.bold,
                        ),
                      );
                    }).toList();
                  },
                ),
              ),
              lineBarsData: lineBarsData1,
              minY: -0.5,
maxY: 110,                    titlesData:
FlTitlesData(                       show:
true,                     leftTitles:
AxisTitles(),
topTitles: AxisTitles(),
bottomTitles: AxisTitles(
                    sideTitles: bottomTitles,
                  ),
                  rightTitles: AxisTitles(
sideTitles: rightTitles,
                  )),
              gridData: FlGridData(
show: true,                       drawHorizontalLine:
true,                     horizontalInterval: 25,
drawVerticalLine: false,
getDrawingHorizontalLine: (value) {
return FlLine(
                    color: TColor.gray.withOpacity(0.15),
                    strokeWidth: 2,
                  );
                },
              ),
              borderData: FlBorderData(
                show: true,
border: Border.all(
                    color: Colors.transparent,
                  ),
                ),
              ),
            )),
        SizedBox(
          height: media.width * 0.05,
        ),
        Row(
```

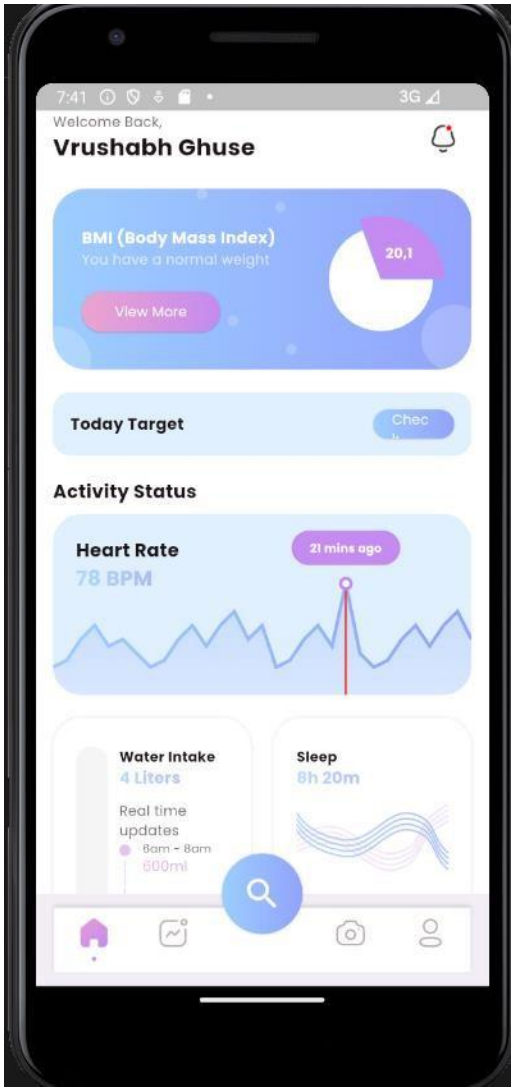
```
                mainAxisAlignment: MainAxisAlignment.spaceBetween,
children: [
                  Text(
                    "Latest Workout",
style: TextStyle(
color: TColor.black,
                      fontSize: 16,
                      fontWeight: FontWeight.w700),
                  ),
                  TextButton(
onPressed: () {},
child: Text(                        "See
More",                        style:
TextStyle(                          color:
TColor.gray,
                        fontSize: 14,
                        fontWeight: FontWeight.w700),
                    ),
                  )
                ],
              ),
              ListView.builder(
padding: EdgeInsets.zero,
                  physics: const NeverScrollableScrollPhysics(),
                  shrinkWrap:                         true,
itemCount:                    lastWorkoutArr.length,
itemBuilder: (context, index) {                    var
wObj = lastWorkoutArr[index] as Map? ?? {};
                    return InkWell(                        onTap:
() {                          Navigator.push(
context,                              MaterialPageRoute(
builder: (context) =>                                 const
FinishedWorkoutView(),
                            ),
                          );
                        },
                        child: WorkoutRow(wObj: wObj));
                  }),
              SizedBox(
                height: media.width * 0.1,
              ),
            ],
          ),
        ),
      ),
    ),
  );
 }

 List<PieChartSectionData> showingSections() {
   return List.generate(
     2,
     (i) {
```

```
      var color0 = TColor.secondaryColor1;

      switch (i) {
case 0:
          return
PieChartSectionData(
color: color0,                value: 33,
            title: '',
radius: 55,
            titlePositionPercentageOffset: 0.55,
            badgeWidget: const Text(
              "20,1",
style: TextStyle(
color: Colors.white,
                  fontSize: 12,
                  fontWeight: FontWeight.w700),
            ));
case 1:
          return
PieChartSectionData(            color:
Colors.white,              value: 75,
            title: '',
radius: 45,
            titlePositionPercentageOffset: 0.55,
          );
default:
          throw Error();
      }
    },
  );
 }

 LineTouchData get lineTouchData1 => LineTouchData(
      handleBuiltInTouches: true,        touchTooltipData:
LineTouchTooltipData(          tooltipBgColor:
Colors.blueGrey.withOpacity(0.8),
      ),
    );

 List<LineChartBarData> get lineBarsData1 => [
      lineChartBarData1_1,
lineChartBarData1_2,
    ];

 LineChartBarData get lineChartBarData1_1 => LineChartBarData(
isCurved: true,
      gradient: LinearGradient(colors: [
TColor.primaryColor2.withOpacity(0.5),
        TColor.primaryColor1.withOpacity(0.5),
      ]),
      barWidth: 4,
isStrokeCapRound: true,
dotData: FlDotData(show: false),
```

```
      belowBarData: BarAreaData(show:
false),        spots: const [          FlSpot(1, 35),
        FlSpot(2, 70),
        FlSpot(3, 40),
        FlSpot(4, 80),
        FlSpot(5, 25),
        FlSpot(6, 70),
        FlSpot(7, 35),
      ],
    );

  LineChartBarData get lineChartBarData1_2 => LineChartBarData(
isCurved: true,
      gradient: LinearGradient(colors: [
TColor.secondaryColor2.withOpacity(0.5),
        TColor.secondaryColor1.withOpacity(0.5),
      ]),
      barWidth: 2,
isStrokeCapRound: true,
dotData: FlDotData(show: false),
belowBarData: BarAreaData(
show: false,
      ),
      spots: const [
FlSpot(1, 80),
        FlSpot(2, 50),
        FlSpot(3, 90),
        FlSpot(4, 40),
        FlSpot(5, 80),
        FlSpot(6, 35),
        FlSpot(7, 60),
      ],
    );

  SideTitles get rightTitles => SideTitles(
getTitlesWidget: rightTitleWidgets,
      showTitles: true,
interval: 20,
reservedSize: 40,
    );

  Widget rightTitleWidgets(double value, TitleMeta meta) {
    String text;    switch
(value.toInt()) {      case
0:        text = '0%';
break;      case 20:
text = '20%';        break;
case 40:        text =
'40%';        break;
case 60:        text =
'60%';        break;
case 80:        text =
'80%';        break;
case 100:        text =
```

```
'100%';         break;
default:         return
Container();
    }

    return Text(text,
style: TextStyle(
color: TColor.gray,
          fontSize: 12,
        ),
        textAlign: TextAlign.center);
  }

  SideTitles get bottomTitles => SideTitles(
        showTitles: true,
reservedSize: 32,
interval: 1,
        getTitlesWidget: bottomTitleWidgets,
      );

  Widget bottomTitleWidgets(double value, TitleMeta meta) {
    var style = TextStyle(
color: TColor.gray,
      fontSize: 12,
    );
    Widget text;
    switch (value.toInt()) {       case
1:          text = Text('Sun', style:
style);
        break;       case 2:          text =
Text('Mon', style: style);
        break;       case 3:          text =
Text('Tue', style: style);
        break;       case 4:          text =
Text('Wed', style: style);
        break;       case 5:          text =
Text('Thu', style: style);
        break;       case 6:          text =
Text('Fri', style: style);
break;       case 7:          text =
Text('Sat', style: style);
break;       default:          text =
const Text('');         break;
    }

    return SideTitleWidget(
axisSide: meta.axisSide,
space: 10,       child: text,
    );
  }
}
```

Output:



Conclusion:

Flutter’s widget architecture offers great flexibility for building complex UIs.
Understanding key widgets and concepts is essential for effective Flutter development.

MAD & PWA Lab

Journal

Experiment No.	03
Experiment Title.	To include icons, images, fonts in Flutter app
Roll No.	04
Name	Vrushabh Ghuse
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	15 M

Exp 3.

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

Theory:

1. Icons:

Flutter Icons:

Flutter comes with its set of built-in icons, and you can easily use them in your app. These icons are part of the Icons class, and you can directly reference them in your code.

Custom Icons:

For custom icons, you can use Flutter's flutter_launcher_icons package or create your custom icons using tools like FlutterIcon.

To use custom icons, add the image to your project and reference it in your code:

2. Images:

Asset Images:

Include images in your Flutter app by placing them in the assets directory. To use these images, add them to the pubspec.yaml file:

yaml Copy

code

flutter:

assets:

- assets/images/image1.png
- assets/images/image2.jpg

Then, use Image.asset to display them: dart

Image.asset('assets/images/image1.png');

Network Images:

Load images from the network using the Image.network widget:

dart

Copy code

Image.network('https://example.com/image.jpg');

3. Fonts:

Custom Fonts:

To include custom fonts, place the font files (e.g., .ttf or .otf) in a fonts directory. Update the pubspec.yaml file:

yaml

Then, use the TextStyle with your custom font:

dart Copy code

```
Text(
'Custom Font Text', style:
TextStyle( fontFamily:
'CustomFont',
),
);
```

4. Using Packages:

Icon Packages:

Consider using icon packages like flutter_icons or font_awesome_flutter for a wide range of icons.

Image Packages:

For managing and loading images efficiently, you can use packages like cached_network_image for network images or image_picker for accessing device images.

Font Packages:

Explore font packages like google_fonts to easily use Google Fonts in your app.

5. Accessibility:

Ensure your app is accessible by providing alternative text for images using Semantics or ExcludeSemantics widgets.

Code:

```dart
```

```
import 'package:fitness/common/colo_extension.dart';
import 'package:fitness/common_widget/icon_title_next_row.dart';
import 'package:fitness/common_widget/round_button.dart';
import 'package:fitness/view/workout_tracker/exercises_stpe_details.dart';
import 'package:fitness/view/workout_tracker/add_schedule_view.dart';
import 'package:flutter/material.dart';
import '../../common_widget/exercises_set_section.dart';

class WorkoutDetailView extends StatefulWidget {
  final Map dObj;
  const WorkoutDetailView({Key? key, required this.dObj}) : super(key: key);

  @override
  State<WorkoutDetailView> createState() => _WorkoutDetailViewState();
}

class _WorkoutDetailViewState extends State<WorkoutDetailView> {
  List latestArr = [
    {"image": "assets/img/Workout1.png", "title": "Fullbody Workout", "time": "Today, 03:00pm"},
    {"image": "assets/img/Workout2.png", "title": "Upperbody Workout", "time": "June 05, 02:00pm"},
  ];

  List youArr = [
    {"image": "assets/img/barbell.png", "title": "Barbell"},
    {"image": "assets/img/skipping_rope.png", "title": "Skipping Rope"},
    {"image": "assets/img/bottle.png", "title": "Bottle 1 Liters"},
  ];

  List exercisesArr = [
    {
      "name": "Set 1",
      "set": [
        {"image": "assets/img/img_1.png", "title": "Warm Up", "value": "05:00"},
        {"image": "assets/img/img_2.png", "title": "Jumping Jack", "value": "12x"},
        {"image": "assets/img/img_1.png", "title": "Skipping", "value": "15x"},
        {"image": "assets/img/img_2.png", "title": "Squats", "value": "20x"},
```

```
      {"image": "assets/img/img_1.png", "title": "Arm Raises", "value": "00:53"},
      {"image": "assets/img/img_2.png", "title": "Rest and Drink", "value": "02:00"},
    ],
  },
  {
    "name": "Set 2",
    "set": [
      {"image": "assets/img/img_1.png", "title": "Warm Up", "value": "05:00"},
      {"image": "assets/img/img_2.png", "title": "Jumping Jack", "value": "12x"},
      {"image": "assets/img/img_1.png", "title": "Skipping", "value": "15x"},
      {"image": "assets/img/img_2.png", "title": "Squats", "value": "20x"},
      {"image": "assets/img/img_1.png", "title": "Arm Raises", "value": "00:53"},
      {"image": "assets/img/img_2.png", "title": "Rest and Drink", "value": "02:00"},
    ],
  }
];
late DateTime selectedDate = DateTime.now();
late String selectedTime = "5/27, 09:00 AM"; // Default time

late TimeOfDay selectedTimeOfDay = TimeOfDay.now(); late
String chosenWorkout = "Upperbody";
late String difficulty = "Beginner";
late String customRepetitions = "";
late String customWeights = "";

void updateState(Map updatedDetails) {
  setState(() {
    selectedTime = updatedDetails['selectedTime'];
    chosenWorkout = updatedDetails['chosenWorkout']; difficulty
    = updatedDetails['difficulty']; customRepetitions =
    updatedDetails['customRepetitions']; customWeights =
    updatedDetails['customWeights'];
  });
}

@override
```

```
Widget build(BuildContext context) { var
 media = MediaQuery.of(context).size;
 return Container(
  decoration: BoxDecoration(gradient: LinearGradient(colors: TColor.primaryG)), child:
  NestedScrollView(
   headerSliverBuilder: (context, innerBoxIsScrolled) {
    return [
     SliverAppBar( backgroundColor:
      Colors.transparent, centerTitle: true,
      elevation: 0, leading: InkWell( onTap: ()
      {
        Navigator.pop(context);
       },
       child: Container(
        margin: const EdgeInsets.all(8),
        height: 40, width: 40, alignment:
        Alignment.center, decoration:
        BoxDecoration(
          color: TColor.lightGray,
          borderRadius: BorderRadius.circular(10)),
        child: Image.asset(
        "assets/img/black_btn.png",
         width: 15, height: 15, fit:
         BoxFit.contain,
        ),
       ),
      ),
      actions: [
       InkWell(
       onTap: () {},
        child: Container(
         margin: const EdgeInsets.all(8),
         height: 40, width: 40, alignment:
         Alignment.center, decoration:
         BoxDecoration(
           color: TColor.lightGray, borderRadius:
           BorderRadius.circular(10)),
         child: Image.asset(
```

```
            "assets/img/more_btn.png",
              width: 15, height: 15, fit:
              BoxFit.contain,
            ),
          ),
        )
      ],
    ),
    SliverAppBar( backgroundColor:
      Colors.transparent, centerTitle: true,
      elevation: 0, leadingWidth: 0, leading:
      Container(), expandedHeight:
      media.width * 0.5, flexibleSpace: Align(
      alignment: Alignment.center, child:
      Image.asset(
      "assets/img/detail_top.png", width:
      media.width * 0.75, height: media.width
      * 0.8,
        fit: BoxFit.contain,
      ),
    ),
  ),
 ];
},
body: Container(
 padding: const EdgeInsets.symmetric(horizontal: 15), decoration:
 BoxDecoration(
    color: TColor.white, borderRadius: const
    BorderRadius.only(
       topLeft: Radius.circular(25), topRight: Radius.circular(25))),
 child: Scaffold( backgroundColor:
  Colors.transparent, body: Stack(
    children: [
     SingleChildScrollView(
     child: Column(
        children: [ const
         SizedBox(
           height: 10,
```

```
                ),
                Container( width: 50, height:
                 4,                  decoration:
                 BoxDecoration(
                    color: TColor.gray.withOpacity(0.3), borderRadius:
                    BorderRadius.circular(3)),
                ),
                SizedBox( height:
                 media.width * 0.05, ),
                Row(                                  mainAxisAlignment:
                 MainAxisAlignment.spaceBetween, children: [
                  Expanded( child:
                   Column(
                    crossAxisAlignment: CrossAxisAlignment.start, children:
                    [
                     Text( widget.dObj["title"].toString(),
                      style:       TextStyle(       color:
                      TColor.black,     fontSize:     16,
                      fontWeight: FontWeight.w700),
                     ),
                     Text(
                          "${widget.dObj["exercises"].toString()} | ${widget.dObj["time"].toString()}
| 320 Calories Burn",
                      style: TextStyle( color: TColor.gray,
                         fontSize: 12),
                     ),
                    ],
                   ),
                  ),
                  TextButton(
                   onPressed: () {}, child:


Image.asset(
                     "assets/img/fav.png",
        width: 15,      height: 15,
                    fit: BoxFit.contain,
                   ),
                  )
```

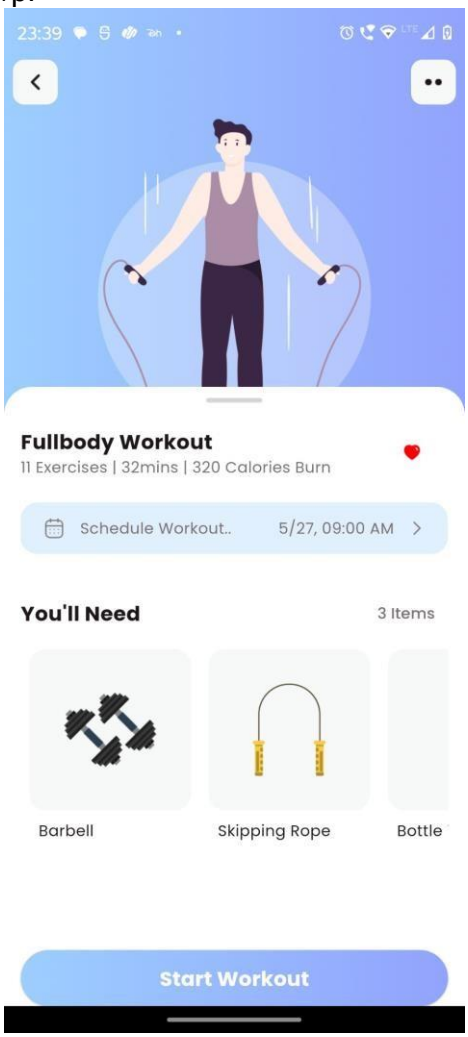
```
 ],
),
SizedBox(                height:
 media.width * 0.05,
),
IconTitleNextRow( icon:
 "assets/img/time.png", title: "Schedule
 Workout..", time: selectedTime, color:
 TColor.primaryColor2.withOpacity(0.3),
 onPressed: () async {
  var updatedDetails = await Navigator.push(
   context,
   MaterialPageRoute( builder: (context) =>
    AddScheduleView(
     date:  selectedDate,  onUpdate:
     updateState,
    ),
   ),
  );

  if (updatedDetails != null) {
   updateState(updatedDetails);
  }
 },
),
SizedBox(                height:
 media.width * 0.02,
),
SizedBox(                height:
 media.width * 0.05,
),
Row(                                    mainAxisAlignment:
 MainAxisAlignment.spaceBetween, children: [
  Text(        "You'll
   Need",      style:
   TextStyle(
    color: TColor.black,
```

```
            fontSize:      16,      fontWeight:
            FontWeight.w700),
      ),
      TextButton( onPressed: () {},
       child:                    Text(
       "${youArr.length}    Items",
       style:
            TextStyle(color: TColor.gray, fontSize: 12),
       ),
      )
     ],
    ),
    SizedBox(              height:
     media.width  *  0.5,  child:
     ListView.builder(
       padding: EdgeInsets.zero,
       scrollDirection: Axis.horizontal,
       shrinkWrap: true, itemCount:
       youArr.length, itemBuilder:
       (context, index) {
        var yObj = youArr[index] as Map? ?? {};
        return    Container(    margin:    const
        EdgeInsets.all(8),     child:     Column(
        crossAxisAlignment:
              CrossAxisAlignment.start,  children:
           [
            Container( height:
              media.width * 0.35, width:
              media.width * 0.35,
              decoration:  BoxDecoration(
                color:     TColor.lightGray,
                borderRadius:
                   BorderRadius.circular(15)),
              alignment:     Alignment.center,
              child:                Image.asset(
              yObj["image"].toString(),  width:
              media.width   *   0.2,   height:
```

```
                        media.width * 0.2, fit:
                        BoxFit.contain,
                        ),
                       ),
                       Padding( padding: const
                        EdgeInsets.all(8.0), child: Text(
                         yObj["title"].toString(),
                         style: TextStyle( color:
                         TColor.black, fontSize:
                         12),
                        ),
                       )
                      ],
                     ));
                }), ),
            SizedBox( height:
              media.width * 0.05,
            ),
            SizedBox( height:
              media.width * 0.1,
            ),
           ],
          ),
         ),
         SafeArea( child:
          Column(
           mainAxisSize: MainAxisSize.max,
           mainAxisAlignment: MainAxisAlignment.end,
           children: [
            RoundButton(title: "Start Workout", onPressed: () {})
           ],
          ),
         )
        ],
       ),
      ),
     ),
    ),
```

```
    );
  }
}
```

o/p:



Conclusion:

In summary, integrating icons, images, and fonts in Flutter enriches the app's visual experience, fostering design flexibility and user engagement. Leveraging packages further enhances development efficiency and diversity, while prioritizing accessibility ensures inclusivity for all users.

MAD & PWA Lab

Journal

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

Experiment 4

Aim: Creating an Interactive Form Using Form Widget in Flutter

Guidelines:

Introduction:

Flutter is a UI toolkit that enables the development of natively compiled applications for mobile, web, and desktop from a single codebase. To create an interactive form using a form widget in Flutter, we can utilize the `Form` widget along with various form-related widgets provided by Flutter.

Guidelines for Creating an Interactive Form:

1. Import Flutter Packages:
   Start by importing the necessary Flutter packages. The `flutter/material.dart` package is essential for building the user interface.

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

2. Create a StatefulWidget:
   Use the `StatefulWidget` to create a stateful widget that will contain the form and handle its state changes.

```dart
class InteractiveForm extends StatefulWidget {
  @override
  _InteractiveFormState createState() => _InteractiveFormState();
}
```

3. Create State Class:
   Inside the state class, define the variables and controllers for form elements.

```dart
class _InteractiveFormState extends State<InteractiveForm> {
  final _nameController = TextEditingController(); final
  _emailController = TextEditingController(); // Add
  more controllers for other form elements
}
```

4. Build Form Widget:

Use the `Form` widget to wrap the form elements. Utilize various form-related widgets like `TextFormField`, `Checkbox`, `Radio`, `DropdownButton`, etc.

```dart
@override
Widget build(BuildContext context) {
  return Form( child: Column( children:
  [
      TextFormField( controller: _nameController, decoration:
        InputDecoration(labelText:
        'Name'),
      ),
      TextFormField( controller: _emailController, decoration:
        InputDecoration(labelText:
        'Email'),
      ),
      // Add more form elements
      ElevatedButton( onPressed:
        () {
          // Handle form submission
        },
        child: Text('Submit'),
      ),
    ],
  ),
  );
}
```

5. Form Validation:

Implement validation logic using the `validator` property of the `TextFormField` widget or custom validation functions.

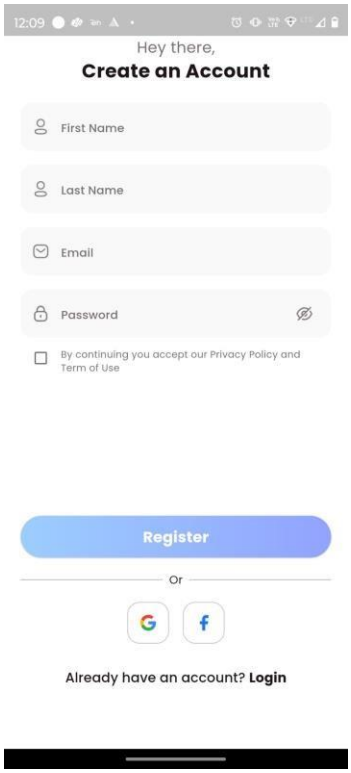
```dart
TextFormField( controller: _emailController, decoration:
  InputDecoration(labelText: 'Email'), validator: (value) {
    if (value.isEmpty || !value.contains('@')) {
      return 'Invalid email format';
    }
    return null;
  },
),
```

6. Handle Form Submission:

Add a callback function to handle the form submission when the submit button is pressed.

```dart
ElevatedButton( onPressed:
  () {
    if (Form.of(context).validate()) {
      // Form is valid, handle submission
      // Access form field values using controllers (e.g., _nameController.text)
    }
  },
  child: Text('Submit'),
),
```

Screenshot:



Conclusion:

Creating an interactive form using the form widget in Flutter involves utilizing various formrelated widgets, handling form validation, and implementing a submission mechanism. This experiment provides a practical guide to building interactive forms in Flutter, facilitating user input and enhancing the user experience in Flutter applications.

MAD & PWA Lab

Journal

Experiment No.	05
Experiment Title.	To apply navigation, routing and gestures in Flutter App
Roll No.	04
Name	Vrushabh Ghuse
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	15 M

MAD and PWA Lab

Name: Vrushabh Ghuse **Class: D15A** **Roll no:22**

Experiment - 5

Aim: To apply navigation, routing and gestures in Flutter App

Theory:

Navigation:
Navigation is a fundamental aspect of mobile app development that involves transitioning between different screens or pages. In Flutter, the Navigator class plays a central role in managing the navigation stack, allowing developers to push and pop routes as users move through the app.

Navigator Class:
The Navigator class handles the navigation stack, maintaining a history of routes. The push method adds a new route to the stack, typically triggered by user actions. The pop method removes the current route from the stack, enabling backward navigation.
Routing:
Routing in Flutter involves defining and organizing the paths or routes within the application. Routes represent different screens or pages, and their effective use is crucial for structuring the app's architecture.

MaterialPageRoute and CupertinoPageRoute:

MaterialPageRoute is employed in apps following Material Design principles, providing a standard Android-style transition between screens.
CupertinoPageRoute is used for iOS-style designs, ensuring a consistent and native user experience. Named Routes:

Named routes offer a more organized and readable approach to navigation.

By assigning names to routes, such as '/details', developers can easily navigate to specific screens using Navigator.pushNamed.

Gestures in Flutter:
Gestures enhance user interaction by allowing the app to respond to touch or mouse inputs. In Flutter, the GestureDetector widget is instrumental in recognizing and handling various gestures.

GestureDetector Widget:

The GestureDetector widget is used to detect a variety of gestures, including taps, drags, and long presses.
By wrapping UI components with GestureDetector, developers can specify callback functions to execute when specific gestures are detected. Gesture Recognizers:

Flutter provides gesture recognizers for more complex gestures, such as panning, pinching, and swiping.
These recognizers, when combined with a GestureDetector, enable the app to respond to a broader range of user inputs. InkWell and InkWell:

The InkWell and InkResponse widgets bring a material ripple effect to touchable UI components.
Integrating these widgets enhances the visual feedback during user interactions, contributing to a more polished and intuitive user experience.

Code: profile_screen.dart

```
import 'package:flutter/material.dart'; import
'../../common/colo_extension.dart'; import
'../../common_widget/round_button.dart'; import
'../../common_widget/setting_row.dart'; import
'../../common_widget/title_subtitle_cell.dart';
import 'package:animated_toggle_switch/animated_toggle_switch.dart';

class ProfileView extends StatefulWidget {
const ProfileView({super.key});   @override
  State<ProfileView> createState() => _ProfileViewState();
```

```
}

class _ProfileViewState extends State<ProfileView> {
bool positive = false;

  List accountArr = [
    {"image": "assets/img/p_personal.png", "name": "Personal Data", "tag": "1"},
    {"image": "assets/img/p_achi.png", "name": "Achievement", "tag": "2"},
    {
      "image": "assets/img/p_activity.png",
      "name": "Activity History",
      "tag": "3"
    },
    {
      "image": "assets/img/p_workout.png",
      "name": "Workout Progress",
      "tag": "4"
    }
];

  List otherArr = [
    {"image": "assets/img/p_contact.png", "name": "Contact Us", "tag": "5"},
    {"image": "assets/img/p_privacy.png", "name": "Privacy Policy", "tag": "6"},
    {"image": "assets/img/p_setting.png", "name": "Setting", "tag": "7"},
  ];
  @override
  Widget build(BuildContext context) {
return Scaffold(         appBar: AppBar(
backgroundColor:            TColor.white,
centerTitle:  true,               elevation:  0,
leadingWidth:  0,                 title:  Text(
"Profile",           style: TextStyle(
            color: TColor.black, fontSize: 16, fontWeight: FontWeight.w700),
        ),         actions: [           InkWell(
onTap: () {},             child: Container(
margin: const EdgeInsets.all(8),
height: 40,               width: 40,
            alignment:                        Alignment.center,
decoration:  BoxDecoration(                          color:
TColor.lightGray,                              borderRadius:
BorderRadius.circular(10)),           child: Image.asset(
"assets/img/more_btn.png",                        width:  15,
height: 15,                fit: BoxFit.contain,
              ),
            ),
          )
```

```
      ],
    ),
    backgroundColor: TColor.white,      body: SingleChildScrollView(
child: Container(          padding: const EdgeInsets.symmetric(vertical:
15, horizontal: 25),          child: Column(            crossAxisAlignment:
CrossAxisAlignment.stretch,            children: [              Row(
children: [                ClipRRect(                  borderRadius:
BorderRadius.circular(30),                  child: Image.asset(
"assets/img/u2.png",                  width: 50,                  height: 50,
fit: BoxFit.cover,
                ),
              ),
              const SizedBox(
width: 15,
              ),
              Expanded(
child: Column(
                  crossAxisAlignment: CrossAxisAlignment.start,
children: [
                    Text(
                      "Vrushabh Ghuse",
style: TextStyle(                        color:
TColor.black,                        fontSize: 14,
fontWeight: FontWeight.w500,
                    ),
                  ),
                  Text(
                    "Lose a Fat Program",
style: TextStyle(
color: TColor.gray,
fontSize: 12,
                    ),
                  )
                ],
              ),
            ),
            SizedBox(
width: 70,                height:
25,                child:
RoundButton(
title: "Edit",
                type: RoundButtonType.bgGradient,
fontSize: 12,                fontWeight:
FontWeight.w400,                onPressed: () {
                  // Navigator.push(
                  //   context,
                  //   MaterialPageRoute(
                  //     builder: (context) => const ActivityTrackerView(),
```

```
                //   ),
                // );
              },
            ),
          )
        ],
      ),
      const SizedBox(
height: 15,
      ),
      const Row(
children: [ Expanded(
child: TitleSubtitleCell(
title: "180cm",
subtitle: "Height",
            ),
          ),
          SizedBox(
width: 15,
          ),
          Expanded(
child: TitleSubtitleCell(
title: "65kg", subtitle:
"Weight",
            ),
          ),
          SizedBox(
width: 15,
          ),
          Expanded(
child: TitleSubtitleCell(
title: "22yo", subtitle:
"Age",
            ),
          ),
        ],
      ),
      const SizedBox(
height: 25,
      ),
      Container(
padding:
          const EdgeInsets.symmetric(vertical: 10, horizontal: 15),
decoration: BoxDecoration( color: TColor.white,
borderRadius: BorderRadius.circular(15), boxShadow:
const [
            BoxShadow(color: Colors.black12, blurRadius: 2)
```

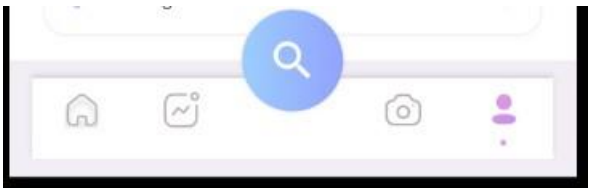
```
                ]),
            child: Column(                  crossAxisAlignment:
CrossAxisAlignment.start,                  children: [
Text(                   "Account",                   style:
TextStyle(                   color: TColor.black,
fontSize: 16,
                    fontWeight: FontWeight.w700,
                  ),
                ),
                const SizedBox(
height: 8,
                ),
                ListView.builder(                   physics: const
NeverScrollableScrollPhysics(),                   shrinkWrap:
true,                   itemCount: accountArr.length,
itemBuilder: (context, index) {                   var iObj =
accountArr[index] as Map? ?? {};                   return
SettingRow(                   icon: iObj["image"].toString(),
title: iObj["name"].toString(),                   onPressed: ()
{},
                  );
                },
                )
              ],
            ),
          ),
          const SizedBox(
height: 25,
          ),
          Container(
padding:
              const EdgeInsets.symmetric(vertical: 10, horizontal: 15),
decoration: BoxDecoration(
              color:                   TColor.white,
borderRadius:                BorderRadius.circular(15),
boxShadow: const [
                BoxShadow(color: Colors.black12, blurRadius: 2)
              ]),
            child: Column(                  crossAxisAlignment:
CrossAxisAlignment.start,
              children: [
                Text(
"Notification",                   style:
TextStyle(                   color:
TColor.black,                fontSize:
16,
                  fontWeight: FontWeight.w700,
```

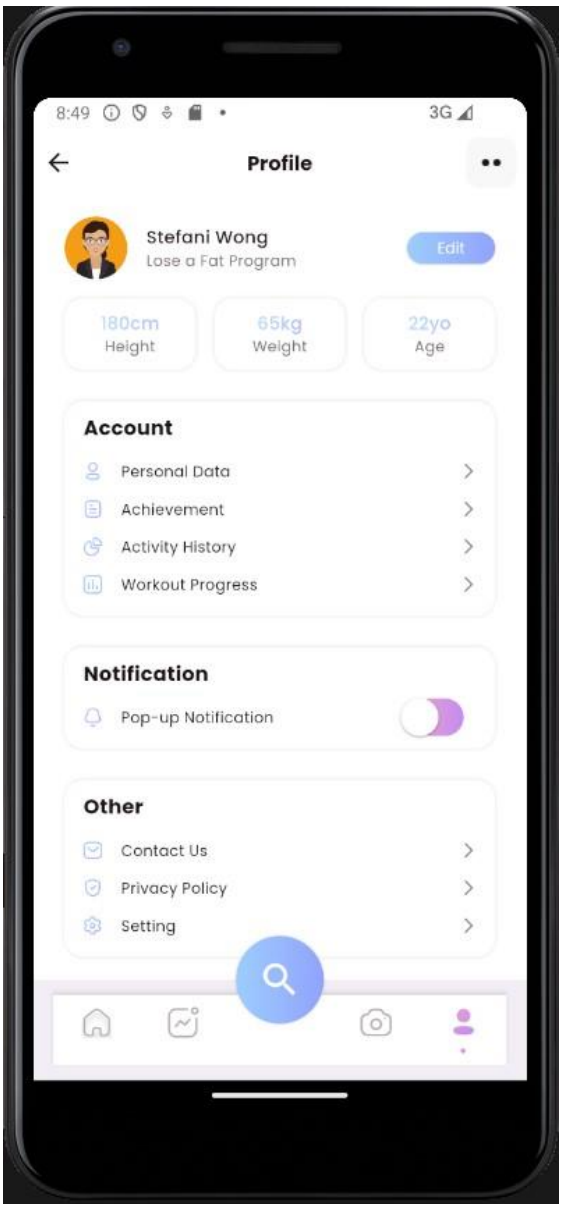
```
                ),
              ),
              const SizedBox(
height: 8,                ),
              SizedBox(                height: 30,
child: Row(                crossAxisAlignment:
CrossAxisAlignment.center,                children: [
                Image.asset("assets/img/p_notification.png",
height: 15, width: 15, fit: BoxFit.contain),
const SizedBox(                width: 15,
),
                Expanded(
child: Text(
                  "Pop-up Notification",
style: TextStyle(                color:
TColor.black,                fontSize:
12,
                  ),
                ),
                CustomAnimatedToggleSwitch<bool>(
current: positive,                values: [false, true],
dif: 0.0,                indicatorSize:
Size.square(30.0),                animationDuration:
                  const Duration(milliseconds: 200),
animationCurve: Curves.linear,                onChanged:
(b) => setState(() => positive = b),
iconBuilder: (context, local, global) {                return
const SizedBox();
                },
                defaultCursor: SystemMouseCursors.click,
onTap: () => setState(() => positive = !positive),
iconsTappable: false,                wrapperBuilder: (context, global, child)
{                return Stack(                alignment:
Alignment.center,                children: [
Positioned(                left: 10.0,                right:
10.0,
                                height:
30.0,                child: DecoratedBox(
decoration: BoxDecoration(
gradient: LinearGradient(
colors: TColor.secondaryG),
borderRadius:
                                const BorderRadius.all(
Radius.circular(50.0)),
                              ),
)),
child,
                  ],
```

```
                    );
                  },
                  foregroundIndicatorBuilder: (context, global) {
return SizedBox.fromSize(                    size: const
Size(10, 10),                    child: DecoratedBox(
decoration: BoxDecoration(                    color:
TColor.white,
                    borderRadius: const BorderRadius.all(
Radius.circular(50.0)),                    boxShadow:
const [                    BoxShadow(
color: Colors.black38,
spreadRadius: 0.05,                    blurRadius:
1.1,                    offset: Offset(0.0, 0.8))
                    ],
                  ),
                ),
              );
            },
          ),
        ]),
      )
    ],
  ),
),
const SizedBox(
height: 25,
),
Container(
padding:
    const EdgeInsets.symmetric(vertical: 10, horizontal: 15),
decoration: BoxDecoration(    color: TColor.white,
borderRadius: BorderRadius.circular(15),    boxShadow:
const [
    BoxShadow(color: Colors.black12, blurRadius: 2)
  ]),
  child: Column(    crossAxisAlignment:
CrossAxisAlignment.start,    children: [
Text(    "Other",    style: TextStyle(
color: TColor.black,    fontSize: 16,
    fontWeight: FontWeight.w700,
  ),
),
const SizedBox(
height: 8,
),
ListView.builder(    physics: const
NeverScrollableScrollPhysics(),    padding:
EdgeInsets.zero,    shrinkWrap: true,
```

```
itemCount: otherArr.length,                    itemBuilder:
(context, index) {                    var iObj = otherArr[index]
as Map? ?? {};                    return SettingRow(
icon: iObj["image"].toString(),                    title:
iObj["name"].toString(),                    onPressed: () {},
                    );
                  },
                )
              ],
            ),
          )
        ],
      ),
    ),
  ),
 );
 }
}
```

Output:





Conclusion:

In this experiment, we have successfully created routing in the bottom navigation bar and connected all pages successfully using Navigator class and implemented it successfully.

MAD & PWA Lab
Journal

Experiment No.	06
Experiment Title.	To Connect Flutter UI with fireBase database
Roll No.	04
Name	Vrushabh Ghuse
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 M

MAD and PWA Lab

Name: Vrushabh Ghuse **Class: D15A** **Roll no:22**

Experiment - 6

Aim: To Connect Flutter UI with FireBase database

Theory:

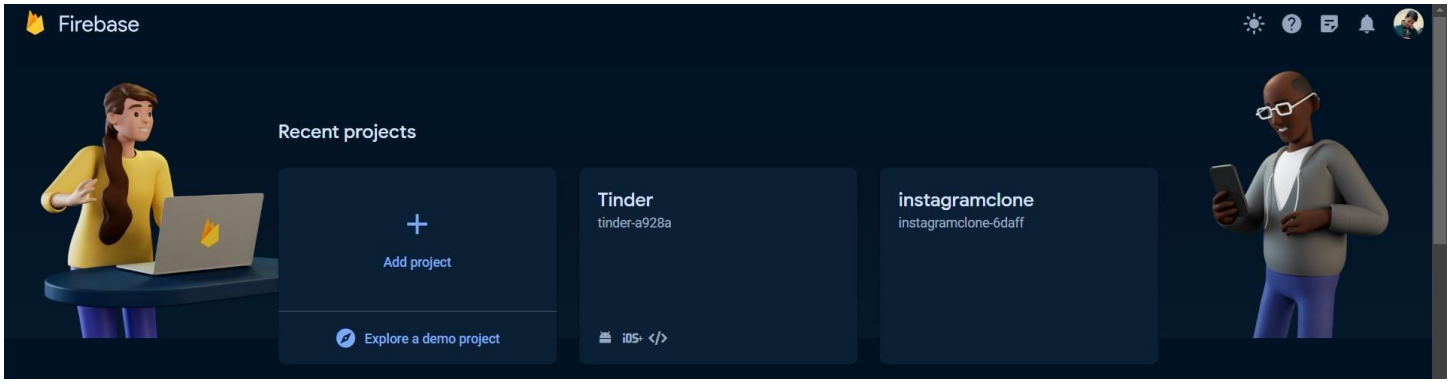
Prerequisites

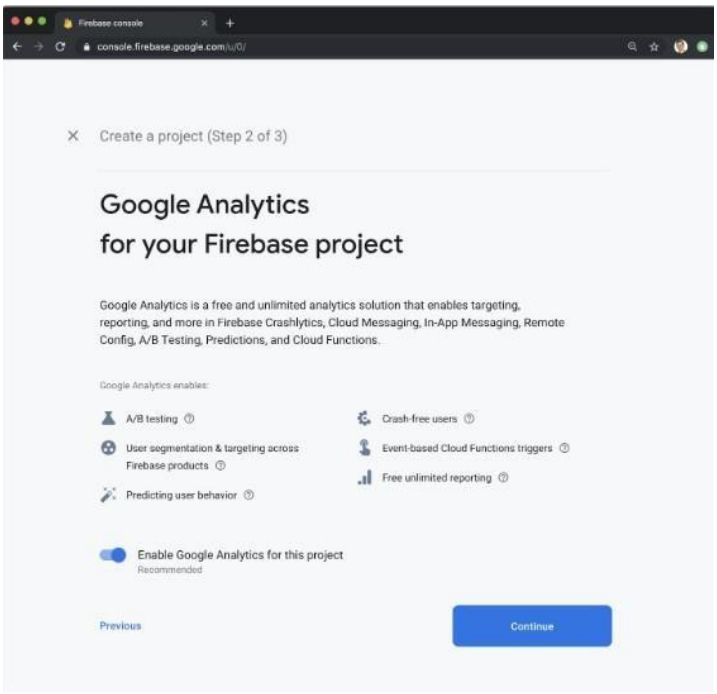
To complete this tutorial, you will need:

- A Google account to use Firebase.
- Developing for iOS will require XCode.
- To download and install Flutter.
- To download and install Android Studio and Visual Studio Code.
- It is recommended to install plugins for your code editor:
  - Flutter and Dart plugins installed for Android Studio.
  - Flutter extension installed for Visual Studio Code.

Create a Firebase Project:

First, log in with your Google account to manage your Firebase projects. From within the Firebase dashboard, select the Create new project button and give it a name:





Go to the Firebase Console and create a new project. Add your Flutter app to the Firebase project:

Register your app in the Firebase project, and follow the instructions to download the configuration files (google-services.json for Android, GoogleService-Info.plist for iOS).

The most important thing here is to match up the Android package name that you choose here with the one inside of our application.

The structure consists of at least two segments. A common pattern is to use a domain name, a company name, and the application name:

com.example.flutterfirebaseexample

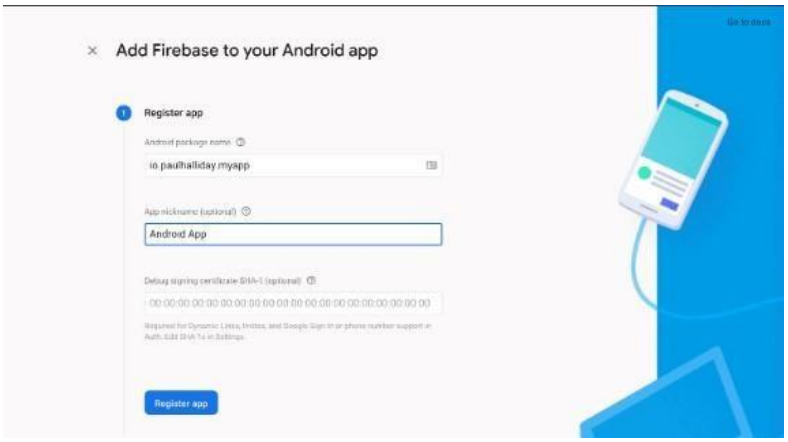
Once you've decided on a name, open android/app/build.gradle in your code editor and update the applicationId to match the Android package name: android/app/build.gradle

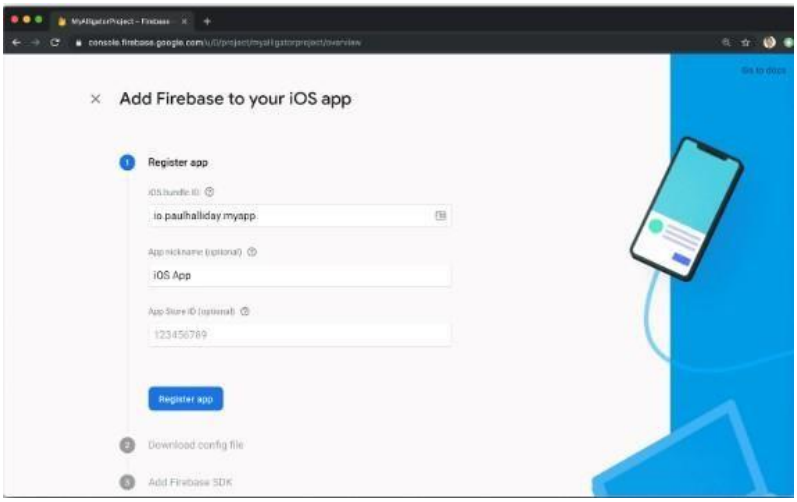
```
...
defaultConfig {
    // TODO: Specify your own unique Application ID (https://developer.android.com/studio/build/application-id.html).
    applicationId 'com.example.flutterfirebaseexample'
    ...
}
...
```

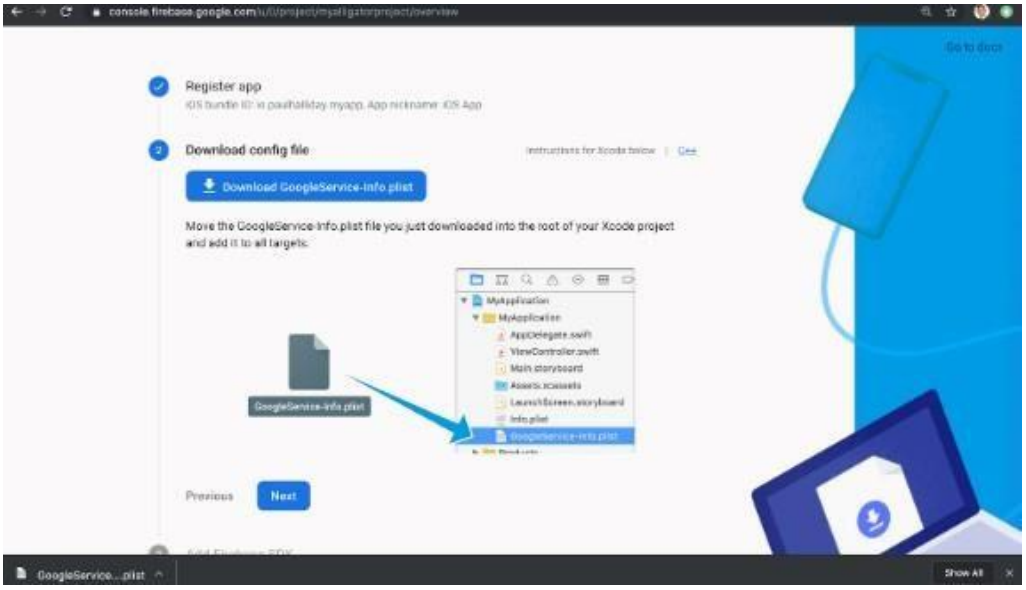
Downloading the Config File

The next step is to add the Firebase configuration file into our Flutter project. This is important as it contains the API keys and other critical information for Firebase to use.

Select Download `google-services.json` from this page:







2. Add Firebase to your Flutter project: Add Dependencies:

Open your pubspec.yaml file and add the necessary dependencies: yaml

```yaml
dependencies:
  flutter:
    sdk: flutter


  # The following adds the Cupertino Icons
  # Use with the CupertinoIcons class for
  cupertino_icons: ^1.0.2
  carousel_slider: ^4.2.1
  fl_chart: ^0.62.0
  simple_animation_progress_bar: ^1.6.0
  dotted_dashed_line: ^0.0.3
  simple_circular_progress_bar: ^1.0.2
  animated_toggle_switch: ^0.6.2
  readmore: ^2.1.0
  calendar_agenda:
    path: dev_lib/calendar_agenda/
  intl: ^0.18.1
  firebase_core: ^2.25.4
  firebase_auth: ^4.17.4

dependency_overrides:
  intl: ^0.18.1

dev_dependencies:
  flutter_test:
    sdk: flutter
```

Code: login_view.dart

```
import 'package:firebase_auth/firebase_auth.dart'; import
'package:fitness/common/colo_extension.dart'; import
'package:fitness/common_widget/round_button.dart'; import
'package:fitness/common_widget/round_textfield.dart'; import
'package:fitness/view/home/home_view.dart'; import
'package:fitness/view/login/complete_profile_view.dart';
import 'package:flutter/material.dart';

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

  @override
  State<LoginView> createState() => _LoginViewState();
```

```
}

class _LoginViewState extends State<LoginView> {
  TextEditingController emailCOntroller = TextEditingController();
TextEditingController passwordcontroller = TextEditingController();


  void login() async {
    String email = emailCOntroller.text.trim();
String password = passwordcontroller.text.trim();


    if (email == "" || password == "") {
print('please fill all details');
    } else {
try {
        UserCredential userCredential = await FirebaseAuth.instance
.signInWithEmailAndPassword(email: email, password: password);


        if (userCredential != null) {
Navigator.push(              context,
          MaterialPageRoute(builder: (context) => HomeView()),
         );
         // _firestore.collection('users').doc(userCredential.user!.uid).set({
         //   'uid': userCredential.user!.uid,
         //   'email': email,
         // }, SetOptions(merge: true));
        }
      } on FirebaseAuthException catch (ex) {
print(ex.code.toString());
      }
    }
}
```

```
 bool isCheck = false;
 @override
 Widget build(BuildContext context) {    var media =
MediaQuery.of(context).size;    return Scaffold(
backgroundColor: TColor.white,      body:
SingleChildScrollView(        child: SafeArea(          child:
Container(            height: media.height * 0.9,
padding: const EdgeInsets.symmetric(horizontal: 20),
child: Column(              crossAxisAlignment:
CrossAxisAlignment.center,              children: [
            Text(
              "Hey there,",              style: TextStyle(color:
TColor.gray, fontSize: 16),
            ),
            Text(
              "Welcome Back",              style:
TextStyle(                  color: TColor.black,
fontSize: 20,                    fontWeight:
FontWeight.w700),
            ),
            SizedBox(
height: media.width * 0.05,
            ),
            SizedBox(
height: media.width * 0.04,
            ),
            // const RoundTextField(
            //   hitText: "Email",
            //   icon: "assets/img/email.png",
            //   keyboardType: TextInputType.emailAddress,
            // ),
```

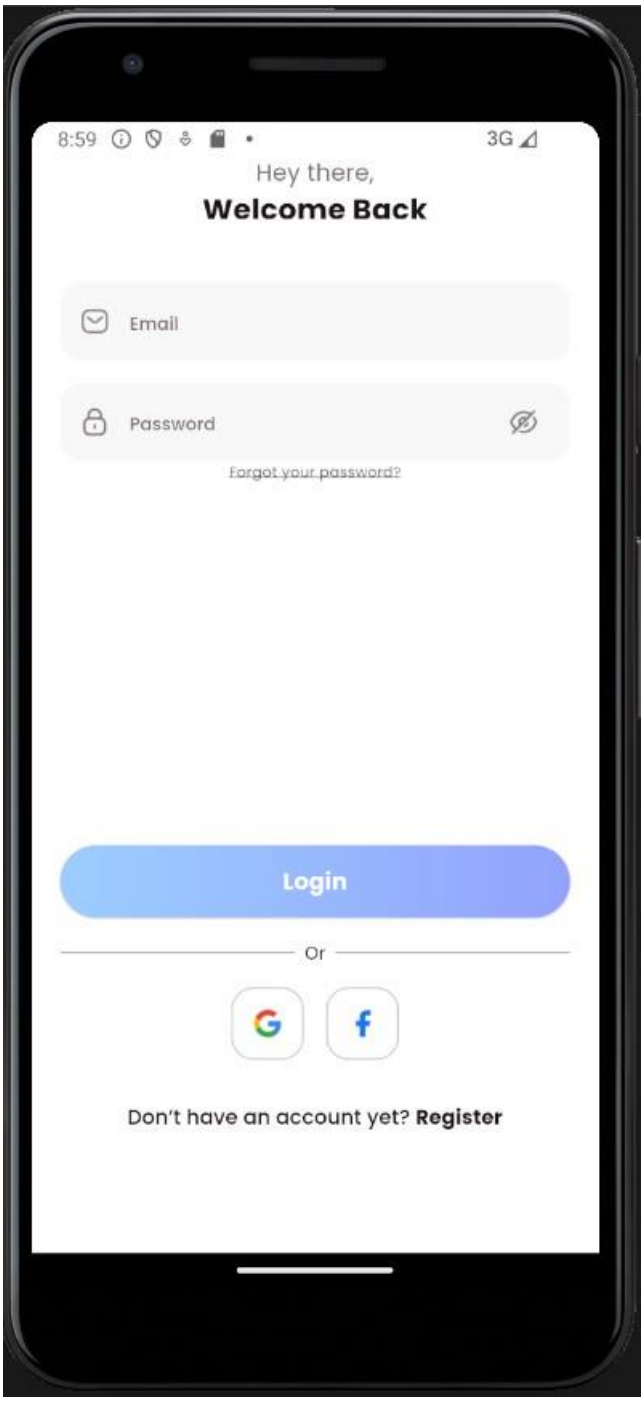
```
            TextField(                  controller:
emailCOntroller,                  decoration:
InputDecoration(labelText: 'Email'),
            ),
            SizedBox(
height: media.width * 0.04,
            ),
            // RoundTextField(
            //   hitText: "Password",
            //   icon: "assets/img/lock.png",
            //   obscureText: true,
            //   rigtIcon: TextButton(
            //       onPressed: () {},
            //       child: Container(
            //           alignment: Alignment.center,
            //           width: 20,
            //           height: 20,
            //           child: Image.asset(
            //             "assets/img/show_password.png",
            //             width: 20,
            //             height: 20,
            //             fit: BoxFit.contain,
            //             color: TColor.gray,
            //           ))),
            // ),
            TextField(                  controller:
passwordcontroller,                  decoration:
InputDecoration(labelText: 'Password'),
            ),
            Row(
              mainAxisAlignment: MainAxisAlignment.center,                  children: [
```

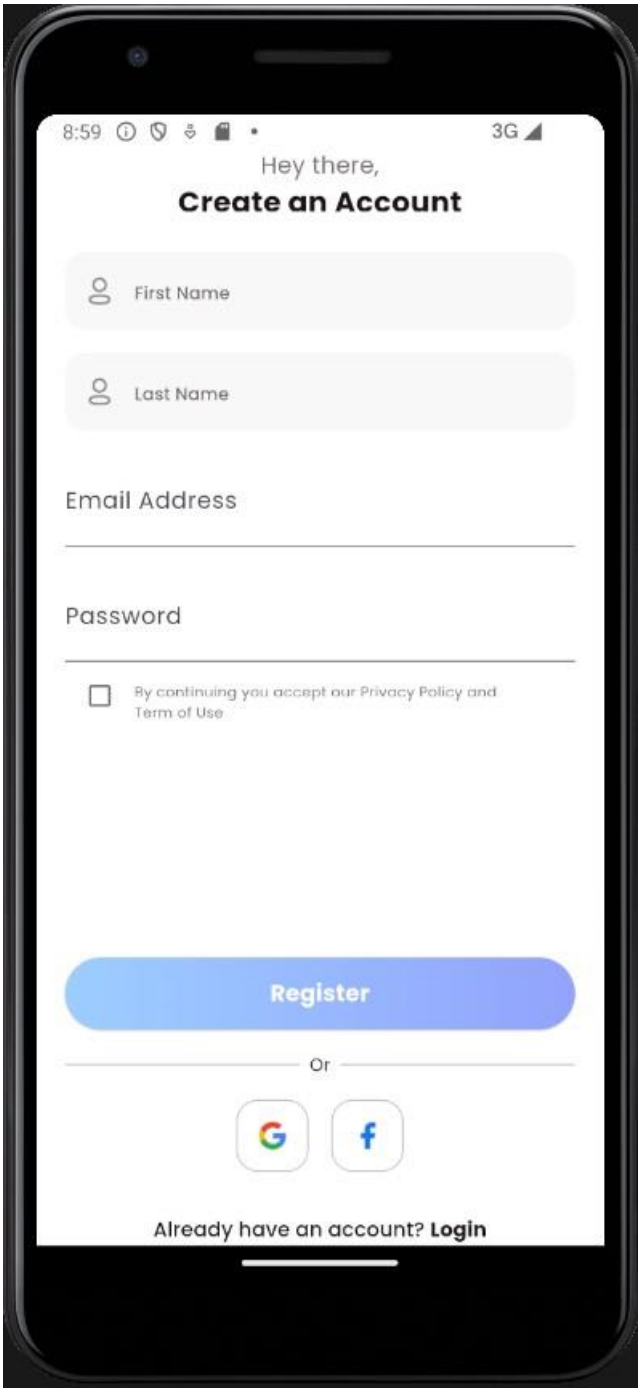
```
                Text(
                  "Forgot your password?",
style: TextStyle( color: TColor.gray,
fontSize: 10, decoration:
TextDecoration.underline),
                ),
              ],
            ),
            const Spacer(),
RoundButton(
title: "Login",
onPressed: () {
login();
                // Navigator.push(
                //     context,
                //     MaterialPageRoute(
                //         builder: (context) =>
                //             const CompleteProfileView()));
              }),
            SizedBox(
height: media.width * 0.04,
            ),
            Row(
              // crossAxisAlignment: CrossAxisAlignment.,
children: [ Expanded( child:
Container( height: 1, color:
TColor.gray.withOpacity(0.5),
              )),
              Text(
                "  Or  ", style: TextStyle(color:
TColor.black, fontSize: 12), ),
```

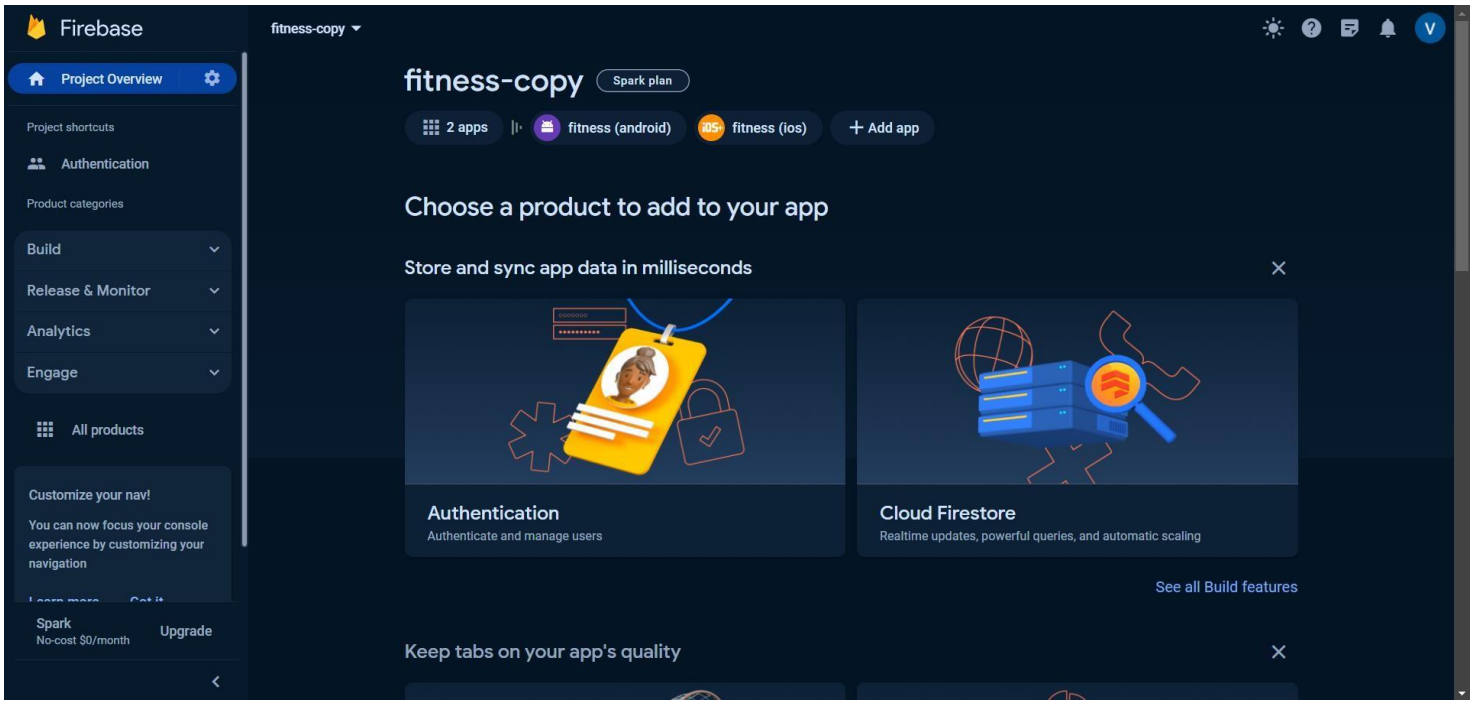
```
                Expanded(                    child:
Container(                height: 1,                color:
TColor.gray.withOpacity(0.5),
                )),
              ],
            ),
            SizedBox(
height: media.width * 0.04,
            ),
            Row(
              mainAxisAlignment: MainAxisAlignment.center,
children: [                GestureDetector(                onTap:
() {},                child: Container(                width:
50,                height: 50,                alignment:
Alignment.center,                decoration: BoxDecoration(
color: TColor.white,                border: Border.all(
width: 1,                color: TColor.gray.withOpacity(0.4),
                  ),
                  borderRadius: BorderRadius.circular(15),
                ),                child:
Image.asset(
"assets/img/google.png",
width: 20,                height: 20,
                  ),
                ),
              ),
              SizedBox(
                width: media.width * 0.04,
              ),
              GestureDetector(                onTap: ()
{},                child: Container(                width:
```

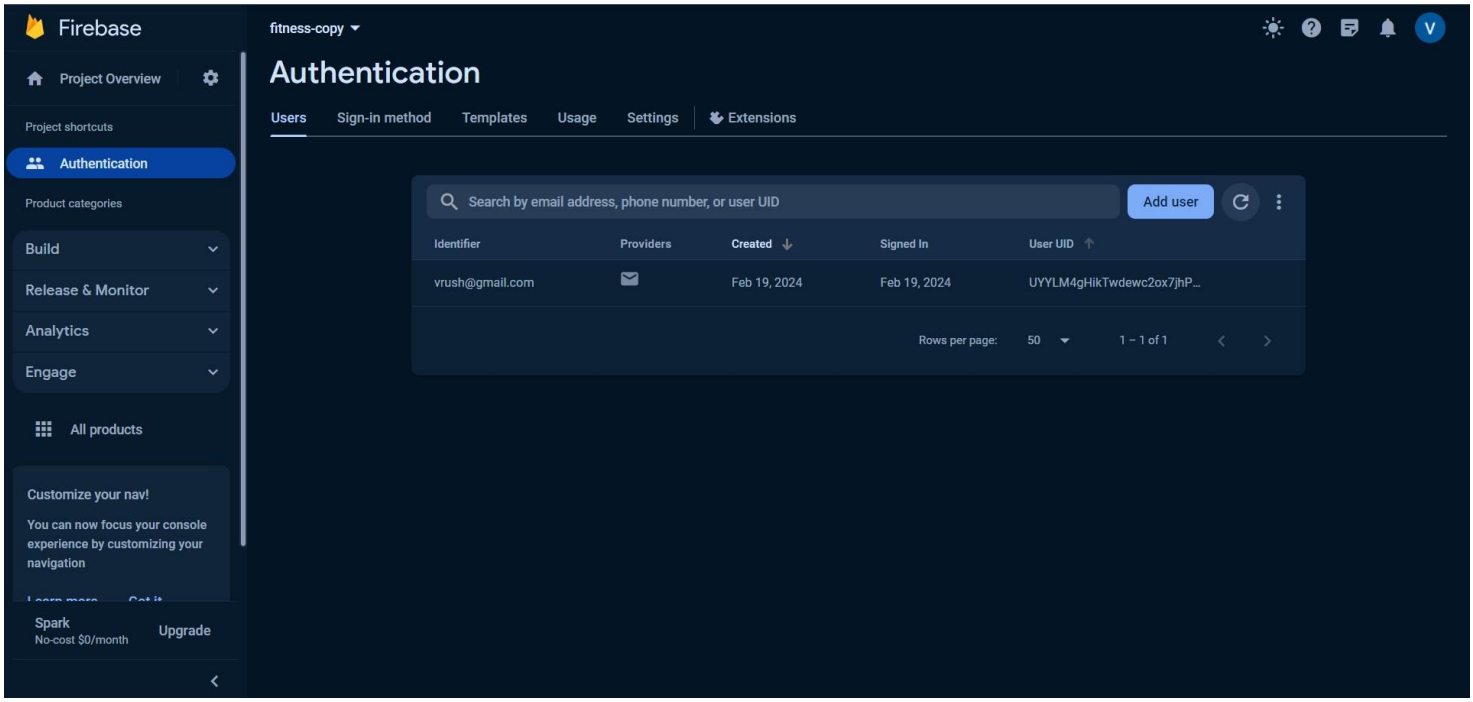
```
50,                height: 50,                alignment:
Alignment.center,                                decoration:
BoxDecoration(                color: TColor.white,
border: Border.all(                                width: 1,
color: TColor.gray.withOpacity(0.4),
                    ),
                    borderRadius: BorderRadius.circular(15),
                  ),                child:
Image.asset(
"assets/img/facebook.png",
width: 20,                height: 20,
                  ),
                ),
              )
            ],
          ),
          SizedBox(
height: media.width * 0.04,
          ),
          TextButton(
onPressed: () {
              Navigator.pop(context);
            },                child: Row(
mainAxisSize: MainAxisSize.min, children: [
              Text(
                "Don’t have an account yet? ",
style: TextStyle(                color:
TColor.black,                fontSize: 14,
                ),
              ),
```

```
                    Text(                    "Register",
style: TextStyle(                    color:
TColor.black,                    fontSize: 14,
fontWeight: FontWeight.w700),
                      )
                    ],
                  ),
                ),
                SizedBox(
height: media.width * 0.04,
                ),
              ],
            ),
          ),
        ),
      ),
    );
  }
}Output:
```


8:59
3G
Hey there,
Welcome Back
Email
Password
Forgot your password?
Login
Or
Don't have an account yet? Register


8:59
3G
Hey there,
Create an Account
First Name
Last Name
Email Address
Password
By continuing you accept our Privacy Policy and Term of Use
Register
Or
Already have an account? Login





Conclusion:

In this experiment, we have successfully connected firebase database and authenticated using google signin and email and password with out flutter application successfully.

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.	04
Name	Vrushabh Ghuse
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO4: Understand various PWA frameworks and their requirements
Grade:	15 M

MAD and PWA Lab

Name: Vrushabh Ghuse **Class: D15A** **Roll no:22**

Experiment - 7

Aim: To write meta data of your Ecommerce PWA in a Web app manifest file to enable add to home screen 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 endusers 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.

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.

Code:

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>MbaKaro-Distance Learning Guide</title>
```

```
    <link rel="stylesheet" href="../styles/style.css" />
    <link
href="https://fonts.googleapis.com/css2?family=Raleway:ital,wght@0,100;0,200;0,300;0,400;0,500;0,600;
0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;1,600;1,700;1,800;1,900&display=swap"
rel="stylesheet" />
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.2/css/all.min.css" />
<link rel="manifest" href="../manifest.json">
    <link rel="service-worker" href="../service-worker.js">
    <script>        if ('serviceWorker' in navigator) {            window.addEventListener('load',
function () {                navigator.serviceWorker.register('/service-worker.js', { scope: '/pages/'
}).then(function
(registration) {                    console.log('ServiceWorker registration successful with scope: ',
registration.scope);
            }, function (err) {
console.log('ServiceWorker registration failed: ', err);
            });
        });
    }
    function triggerPushNotification() {    // Check if
PushManager is supported    if ('serviceWorker' in navigator &&
'PushManager' in window) {
    // Check if permission has been granted already
if (Notification.permission === 'granted') {
        // Register the service worker
navigator.serviceWorker.ready.then(function(registration) {
            // Convert data to JSON format
var data = {
                "method": "pushMessage",
                "message": "Abhishek here"
            };
            // Trigger push event with the specified data
registration.showNotification("Abhishek's MBAKaro", {
data: JSON.stringify(data)
            });
        });
    } else {
        // Permission hasn't been granted yet, request it
```

```
        Notification.requestPermission().then(function(permission) {
if (permission === 'granted') {
                // Permission granted, trigger push notification
triggerPushNotification();
            } else {                console.error('Notification
permission denied');
            }
        });
    }
}
function requestNotificationPermission() {
    Notification.requestPermission().then(function(permission) {
if (permission === 'granted') {            console.log('Notification
permission granted');
    } else {            console.error('Notification
permission denied');
    }
    });
}



    </script>

</head>

<body>
    <section class="header">
        <nav>
            <a href="index.html"><img src="" alt="" /></a>
            <div class="nav-links" id="nav-links">
                <i class="fa fa-times" onclick="hideMenu()"></i>
                <ul>
                    <li><a href="">Home</a></li>
                    <li><a href="/pages/aboutUs.html">About</a></li>
                    <li><a href="#courses" data-scroll>Courses</a></li>
                    <li><a href="/pages/contactUs.html">Contact Us</a></li>
```

```
                <li><i id="dark-mode-toggle" onclick="toggleDarkMode()">
                        <i id="dark-mode-icon" class="fas fa-sun"></i>

                    </i></li>
            </ul>
        </div>
        <i class="fa fa-bars" onclick="showMenu()"></i>
    </nav>
    <!-- Add this button inside the body of your HTML -->


    <div class="text-box">
        <h1>Unlock your potential with MbaKaro</h1>
        <p>
            Here we provide consultancy and guidance for your successful career
        </p>
        <a href="./aboutUs.html" class="hero-btn">Visit us to know more</a>
    </div>
</section>

<!-- ------------------course----------- -->
<section class="University">
    <h1>University</h1>
    <p>Checkout the new courses and know more ...</p>

    <div class="row">
        <div class="uni-col">
            <img src="../images/nmims-university-logo.png" alt="" />
            <h3>NMIMS UNIVERSITY</h3>
            <p>
                NMIMS has an A+ accreditation from NAAC, It has also achieved a
remarkable position in the top 100 ranking by NIRF, emphasizing its
recognition and stature among the best educational institutions in
                India.<br /><b>Course Duration – 2 years</b><br /><b>Course fees – 1,44,000</b>
            </p>
            <a href="" class="hero-btn2">Download</a>
        </div>
        <div class="uni-col">
```

```
            <img src="../images/dpu.png" alt="" />
            <h3>D.Y. PATIL UNIVERSITY</h3>
            <p>
                DY Patil has achieved a top 50 ranking among universities. It offers
                UGC-approved programs, its programs are approved by AICTE & A++ NAAC
reflects the university’s commitment to excellence in education.<br /><b>Course Duration – 2
years</b><br /><b>Course fees – 1,40,200/-</b>
            </p>
            <a href="" class="hero-btn2">Download</a>
        </div>
        <div class="uni-col">
            <img src="../images/logo.png.webp" alt="" />
            <h3>MANIPAL UNIVERSITY</h3>
            <p>
                Manipal University has been Accredited with an A+ grade by NAAC and
entitled by UGC, it ensures quality education at an affordable cost. Manipal
prepares students for the future by offering courses designed to meet industry
demands.<br /><b>Course Duration – 2 years</b><br /><b>Course fees –
                    1,66,000/-</b>
            </p>
            <a href="" class="hero-btn2">Download</a>
        </div>
    </div>
    <div class="row">
        <div class="uni-col">
            <img src="../images/upgrad.png" alt="" />
            <h3>UPGRADE INSTITUTION</h3>
            <p>
                The upgrade provides a comprehensive distance learning MBA program. It offers
the flexibility to study at your own pace and from anywhere in the world with expert faculty
and interactive online resources.<br /><b>Course Duration – 11 months</b><br
/><b>Course fees – 2,50,000/-</b>
            </p>
            <a href="" class="hero-btn2">Download</a>
        </div>
        <div class="uni-col">
            <img src="../images/logo (3).png" alt="" />
```

```
            <h3>AMITY UNIVERSITY</h3>
            <p>
                Amity’s Online MBA program has achieved a remarkable ranking of #37 worldwide. Amity’s Online MBA offers a flexible learning experience, empowering students to pursue their career goals with a highly regarded degree.<br /><b>Course Duration – 2 years</b><br /><b>Course fees – 1,75,000/</b>
            </p>
            <a href="" class="hero-btn2">Download</a>
        </div>
        <div class="uni-col">
            <img src="../images/jain-logo.png" alt="" />
            <h3>JAIN UNIVERSITY</h3>
            <p>
                Jain University offers UGC-entitled online degree programs, These programs provide students with the flexibility to pursue their education remotely, enabling them to balance their studies with other commitments.<br /><b>Course Duration – 2 years</b><br /><b>Course fees – 75,000/</b>
            </p>
            <a href="" class="hero-btn2">Download</a>
        </div>
    </div>
    <div class="row">
        <div class="uni-col">
            <img src="../images/logo (1).png" alt="" />
            <h3>PARUL UNIVERSITY</h3>
            <p>
                A multidisciplinary destination of learning and innovation, propelling quality in higher education with a record of being India’s youngest private university to receive NAAC A++ accreditation.Situated in Vadodara, Gujarat, Parul University.<br /><b>Course Duration – 2 years</b><br /><b>Course fees – 90,000/-</b>
            </p>
            <a href="" class="hero-btn2">Download</a>
        </div>
        <div class="uni-col">
            <img src="../images/gyan.png" alt="" />
            <h3>SGV UNIVERSITY</h3>
            <p>
```

Suresh Gyan Vihar has received an ‘A’ grade accreditation from NAAC. The online degrees awarded by SGVU are considered valid as the university has been approved by UGC, DEB, AICTE, and other relevant authorities.
Course Duration – 2 years
Course fees – 58,000/-

Download

<h3>WELINGKAR UNIVERSITY</h3>

SP Mandal Welingkar University is accredited by the National Board of Accreditation (NBA), ensuring the quality of its programs. the university is approved by the All India Council for Technical

Education (AICTE).
Course Duration – 2 years
Course fees – 81,000/

Download

</section>

<section class="courses" id="courses">

<h1>Our Courses</h1>

Lorem ipsum dolor sit amet consectetur adipisicing elit.

<h3>Marketing Management</h3>

<!--

 -->

<h3>Human Resource Management</h3>

<!--

```
    </div> -->
    </div>
    <div class="courses-col">
        <img src="../images/courses/finance.png" alt="" />
        <h3>Finance Management</h3>
        <!-- <div class="layer">

    </div> -->
    </div>
    <div class="courses-col">
        <img src="../images/courses/it.png" alt="" />
        <h3>IT Management</h3>
        <!-- <div class="layer">

    </div> -->
    </div>
</div>
<div class="row">
    <div class="courses-col">
        <img src="../images/courses/analytics.png" alt="" />                <h3>Project Management</h3>
        <!-- <div class="layer">

  </div> -->
    </div>
    <div class="courses-col">
        <img src="../images/courses/hr.png" alt="" />
        <h3>Operation Management</h3>
        <!-- <div class="layer">

    </div> -->
    </div>
    <div class="courses-col">
        <img src="../images/courses/hospital.png" alt="" />
        <h3>Health Care Management</h3>
        <!-- <div class="layer">
```

```
    </div> -->
    </div>
    <div class="courses-col">
        <img src="../images/courses/inter business.png" alt="" />
        <h3>International Business Management</h3>
        <!-- <div class="layer">

    </div> -->
    </div>
</div>
<div class="row">
    <div class="courses-col">
        <img src="../images/courses/fintech.png" alt="" />
        <h3>Fintech Management</h3>
        <!-- <div class="layer">

  </div> -->
    </div>
    <div class="courses-col">
        <img src="../images/courses/business.png" alt="" />
        <h3>Business Management</h3>
        <!-- <div class="layer">

    </div> -->
    </div>
    <div class="courses-col">
        <img src="../images/courses/logistic.png" alt="" />
        <h3>AI & ML Management</h3>
        <!-- <div class="layer">

    </div> -->
    </div>
    <div class="courses-col">
        <img src="../images/courses/scm.png" alt="" />
        <h3>Logistic & Supply Chain Management</h3>
        <!-- <div class="layer">
```

```
        </div> -->
        </div>
      </div>
    </section>

    <section class="facility">
      <h1>Perks of MbaKaro</h1>
      <p>
        Lorem ipsum dolor sit amet consectetur adipisicing elit. Sapiente ullam,
nostrum iste aliquid provident aliquam autem repellat pariatur incidunt quas
molestias earum enim corporis quod expedita vitae ducimus amet dolore.
      </p>
      <div class="row">
        <div class="facility-col">
          <img src="../images/perks/consultancy.png" alt="" />
          <h3>Complimentary Consultations</h3>
          <p>
            At MbaKaro, we are committed to supporting individuals on their journey to
mental well-being. We offer a range of high-quality counseling services, including
complimentary
            counseling sessions. Our team of experienced professionals is dedicated to providing
accessible and
            compassionate care to help you navigate life's challenges. Take the first step towards a
healthier, happier you with our no-cost counseling services. </p>
        </div>
        <div class="facility-col">
          <img src="../images/perks/faculty.png" alt="" />
          <h3>Skilled Mentors</h3>
          <p>
            Meet our accomplished faculty—an assembly of seasoned educators dedicated to nurturing
academic
            excellence. Our veteran instructors bring a wealth of experience, expertise, and passion to
the
            learning environment. With a commitment to fostering growth and knowledge, our skilled
mentors guide
            students on a transformative educational journey. Experience the difference of learning from
a team of accomplished educators who are committed to your success.
          </p>
        </div>
```

```
        <div class="facility-col">
            <img src="../images/perks/support.png" alt="" />
            <h3>Specialized Assistance</h3>
            <p>
                Experience the difference with our personalized support services. At MbaKaro, we
understand that individual needs vary, and our dedicated team is here to provide tailored assistance
                to help you thrive. Whether you're a student, client, or customer, our commitment to
personalized
                support ensures that your unique requirements are met with care and attention. Discover a
customized
                approach to [your service or product] that puts your satisfaction and success at the forefront
of                 everything we do.
            </p>
        </div>
    </div>
</section>
<section class="testimonials">
    <h1>Whats does people say about us</h1>
    <p>
        Lorem ipsum dolor sit amet consectetur adipisicing elit. Accusantium
ipsum laudantium, illum dolores voluptas enim blanditiis assumenda odio
et nesciunt eius labore. Assumenda at illo, accusantium laborum sit
eveniet excepturi?
    </p>

    <div class="row">
        <div class="test-col">
            <img src="../images/Work.png" alt="" />
            <div>
                <p>
                    Lorem ipsum dolor sit amet consectetur adipisicing elit. Sit,
veniam nisi fugiat rem, magnam praesentium non voluptatum unde, ex
atque assumenda accusamus vitae? Consectetur harum non cupiditate
vero tempore magnam.
                </p>
                <h3>User Name</h3>
                <i class="fa fa-star"></i>
                <i class="fa fa-star"></i>
```

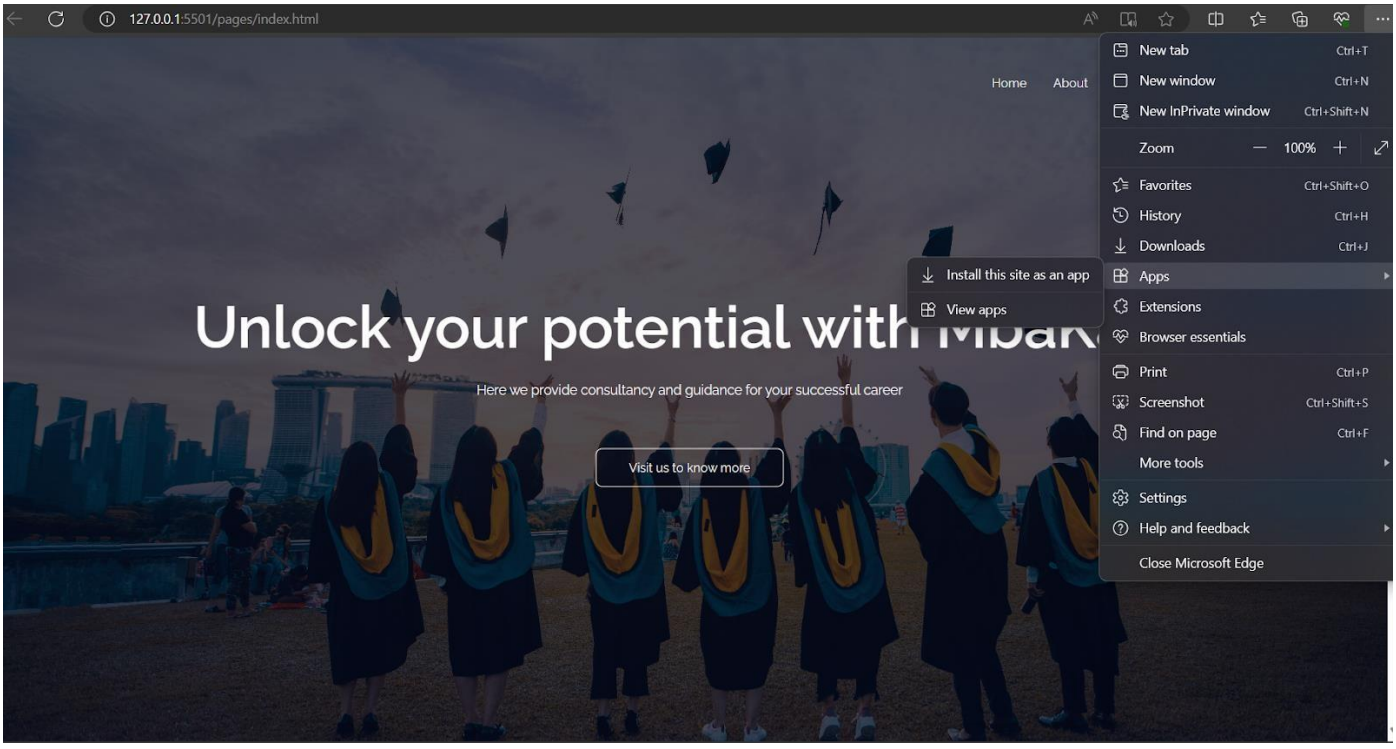
```
                <i class="fa fa-star"></i>
                <i class="fa fa-star"></i>
                <i class="fa fa-star-o"></i>
            </div>
        </div>
        <div class="test-col">
            <img src="../images/Work.png" alt="" />
            <div>
                <p>
                    Lorem ipsum dolor sit amet consectetur adipisicing elit. Sit,
veniam nisi fugiat rem, magnam praesentium non voluptatum unde, ex
atque assumenda accusamus vitae? Consectetur harum non cupiditate
vero tempore magnam.
                </p>
                <h3>User Name</h3>
                <i class="fa fa-star"></i>
                <i class="fa fa-star"></i>
                <i class="fa fa-star"></i>
                <i class="fa fa-star-half-o"></i>
            </div>
        </div>
    </div>
</section>

<section class="cta">
    <h1>Enroll course and know more</h1>
    <!-- Add button to trigger push notification -->
    <button onclick="triggerPushNotification()" class="hero-btn">Trigger Push Notification</button>
    <button onclick="requestNotificationPermission()" class="hero-btn">Request Notification
Permission</button>

</section>
<section class="footer">
    <h4>About Us</h4>
    <p>
        Lorem ipsum dolor sit amet consectetur adipisicing elit. Distinctio,
quisquam cum. Quia dolore maxime error, <br />eum eaque ex tenetur
```

```
architecto optio repellendus culpa, aliquid dignissimos possimus alias             porro
iure hic!
        </p>
        <div class="icons">
            <i class="fab fa-facebook"></i>
            <i class="fab fa-twitter"></i>
            <i class="fab fa-instagram"></i>
            <i class="fab fa-linkedin"></i>
        </div>
    </section>     <script>         var navlinks =
document.getElementById("nav-links");         function
showMenu() {             navlinks.style.width = "200px";
navlinks.style.transitionDuration = "0.5s";
        }
        function hideMenu() {
navlinks.style.width = "0px";
        }
        function toggleDarkMode() {
            // Toggle a class on the body to switch between light and dark styles
document.body.classList.toggle("dark-mode");

            // Save the user's preference in localStorage             const
isDarkMode = document.body.classList.contains("dark-mode");
localStorage.setItem("dark-mode", isDarkMode);
        }

        // Function to set initial dark mode state based on user preference
function setInitialDarkMode() {             const isDarkMode =
localStorage.getItem("dark-mode") === "true";
document.body.classList.toggle("dark-mode", isDarkMode);
        }

        // Call the function to set initial dark mode state when the page loads
window.onload = setInitialDarkMode;
    </script>
</body>

</html>
```

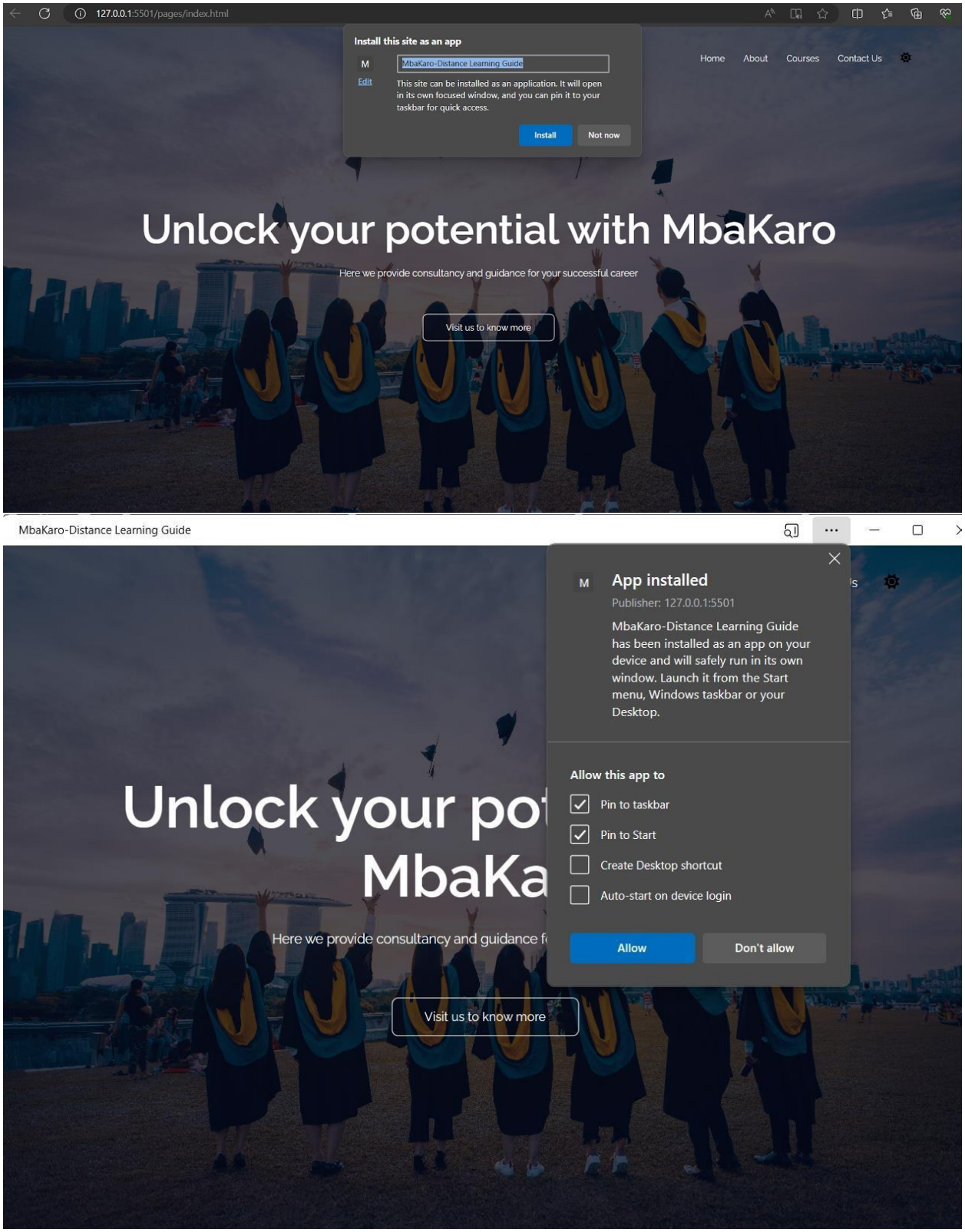
Open folder in VS code and click go live at bottom right corner

Open your hosted site on Microsoft Edge

Click on 3 dots click on
Apps
select install app option



Click on Install

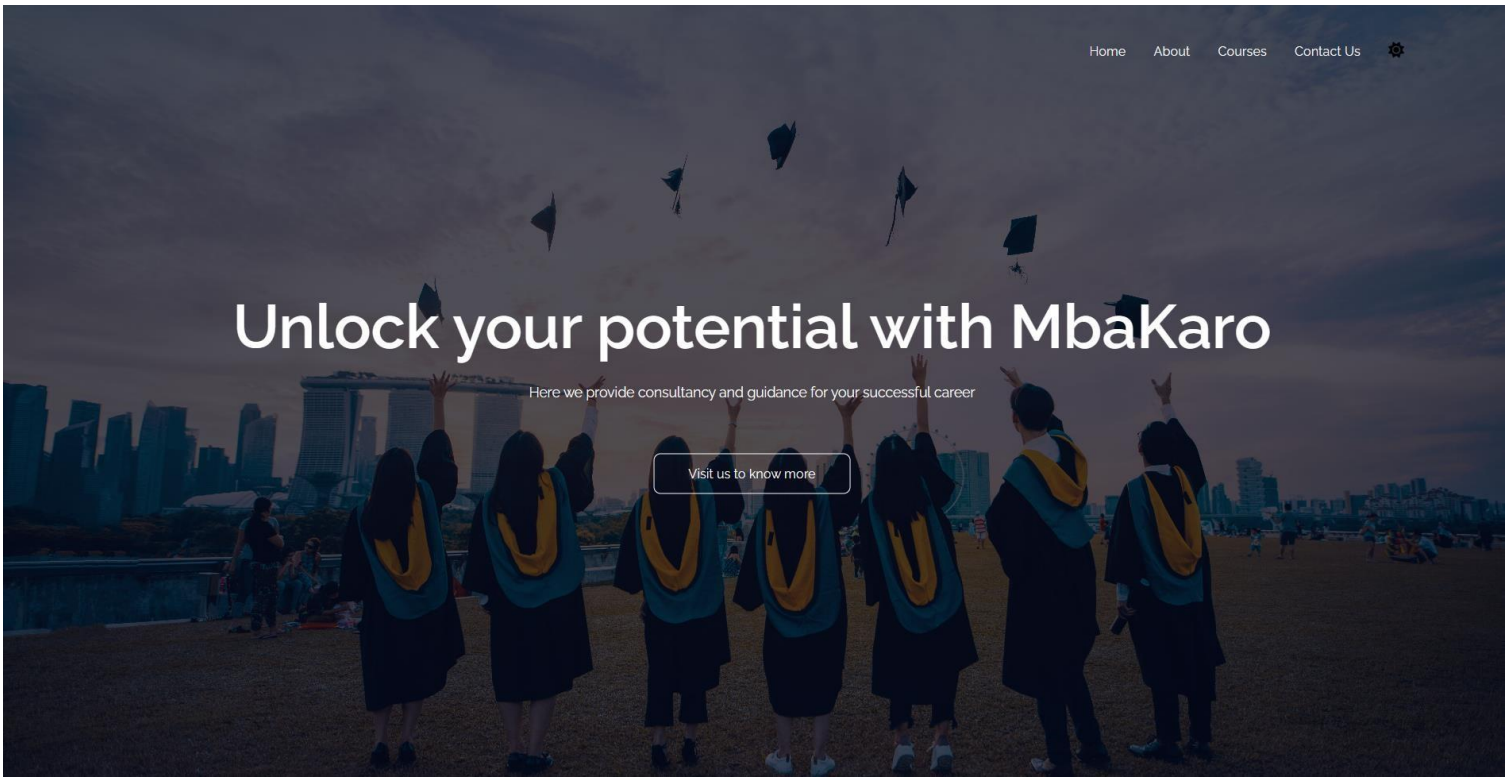


App icon will appear at the bottom

Output:

Desktop App Created Successfully.

Open Desktop App:



Conclusion:

In this experiment, we have successfully created a basic progressive web app of our web page and installed it in our 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.	04
Name	Vrushabh Ghuse
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	15 M

Experiment No: 08

Aim: To implement service worker registration and complete the install and activation process for a new service worker for an E-commerce Progressive Web App (PWA).

Theory:

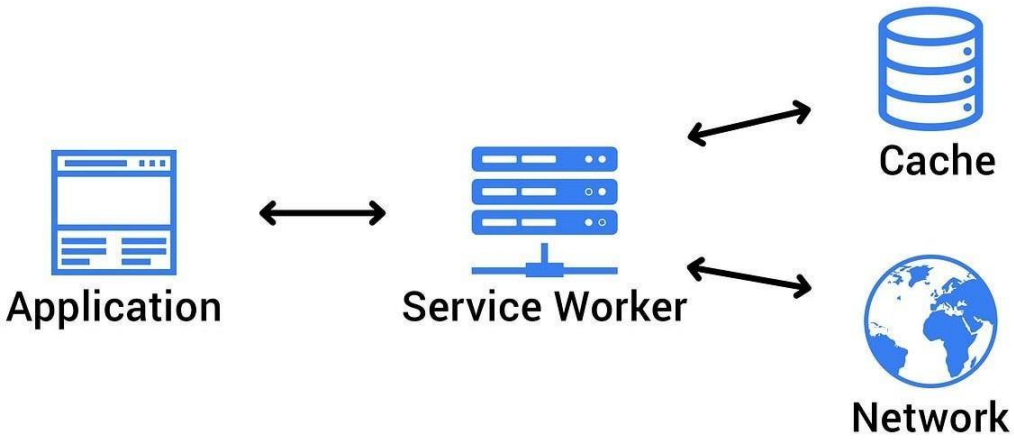
Service Worker:
A service worker is a script that operates in the background of a web browser independently of user interaction. It acts as a programmable network proxy, allowing you to control how network requests from your web page are handled. Service workers enable various functionalities such as tracking network traffic, managing push notifications, and developing "offline-first" web applications using the Cache API.

Key Points about Service Workers:
- Service workers operate as network proxies, allowing you to manipulate network requests and responses.
- They only run over HTTPS to prevent security vulnerabilities.
- Service workers have a life cycle consisting of registration, installation, and activation phases.
- They can cache resources for offline use, manage push notifications, and perform background sync operations.

What We Can Do with Service Workers:
- Control Network Traffic: Manipulate network requests and responses, cache resources, and manage offline content.
- Cache Resources: Store resources locally using the Cache API for offline access.
- Manage Push Notifications: Handle push notifications and display messages to users. - Background Sync: Continue processes even when the internet connection is interrupted.

What We Can't Do with Service Workers:
- Access the Window: Service workers cannot access the DOM directly but can communicate with the window using postMessage. - Operate on Port 80: Service workers require HTTPS and cannot work on Port 80.

Service Worker Cycle:
1. Registration: Register the service worker in your main JavaScript code to instruct the browser where to find it. This initiates the installation process.
2. Installation: During installation, the service worker caches essential resources for offline access and handles any precaching tasks.
3. Activation: After installation, the service worker enters the activation phase, where it takes control of pages within its scope. It cleans up outdated caches and prepares to handle network requests.



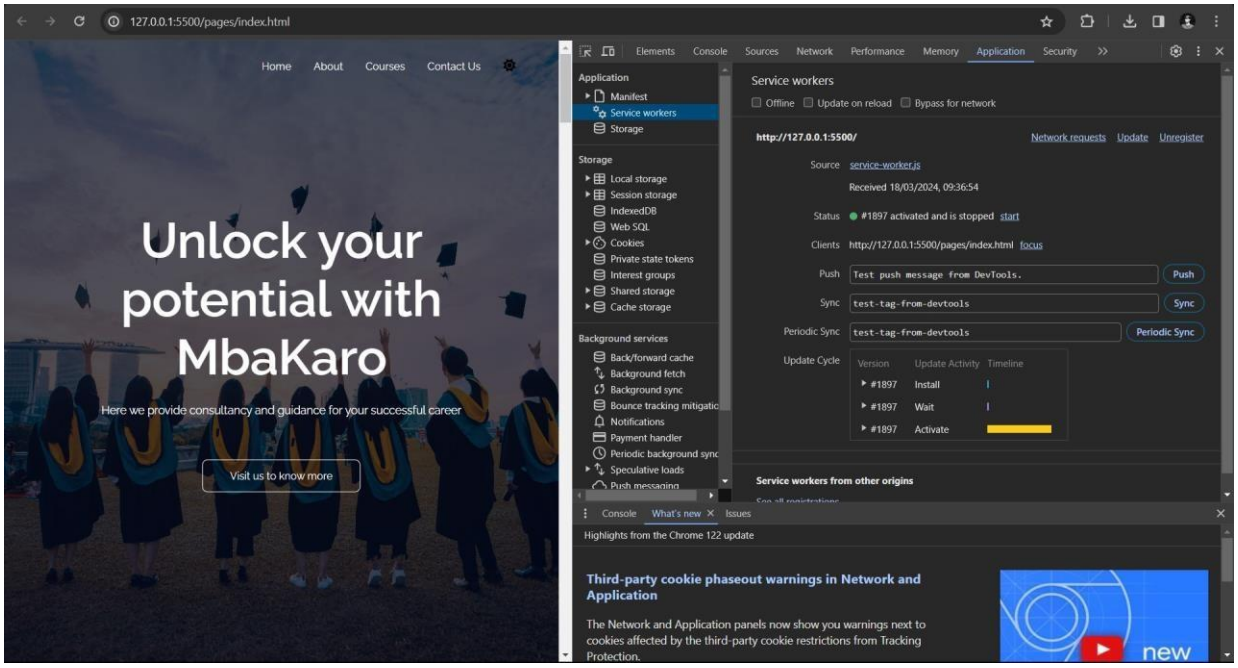
Code Example:

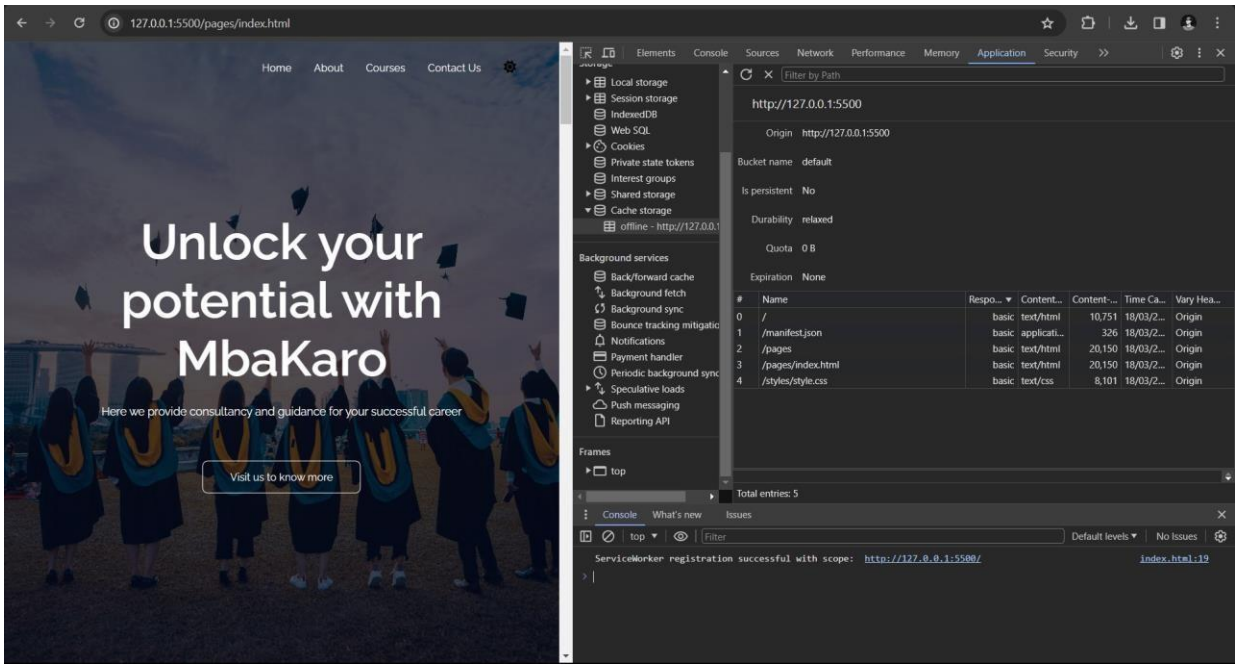
service-worker.js:

javascript

```
self.addEventListener("install", function (event) { event.waitUntil(preLoad());
});
var filesToCache = [
    '/',
    '/menu',
    '/contactUs',
    '/offline.html',
];
var preLoad = function () { return
caches.open("offline").then(function (cache) { return
cache.addAll(filesToCache);
    });
};

self.addEventListener("fetch", function (event) {
    event.respondWith(checkResponse(event.request).catch(function () {
        return returnFromCache(event.request);
    }));
    event.waitUntil(addToCache(event.request));
});

var checkResponse = function (request) {
    return new Promise(function (fulfill, reject) { fetch(request).then(function
        (response) {
```

```
            if (response.status !== 404) {
                fulfill(response);
            } else { reject();
            }
        }, reject);
    });
};


var addToCache = function (request) {
    return caches.open("offline").then(function (cache) {
        return fetch(request).then(function (response) {
            return cache.put(request, response);
        });
    }); };

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;
            }
        });
    });
};
```





Conclusion:

Service workers play a crucial role in enhancing web applications by enabling features such as offline access, push notifications, and background sync. Understanding their life cycle and capabilities is essential for developing robust and efficient Progressive Web Apps.

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.	22
Name	Vrushabh Ghuse
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	15 M

PWALab

PRACTICAL9

Name : Vrushabh Ghuse **Class:D15A** **Roll no: 22**

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 tra c of the page, manage push notifications and develop “o ine 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 tra c 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 di erent 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.

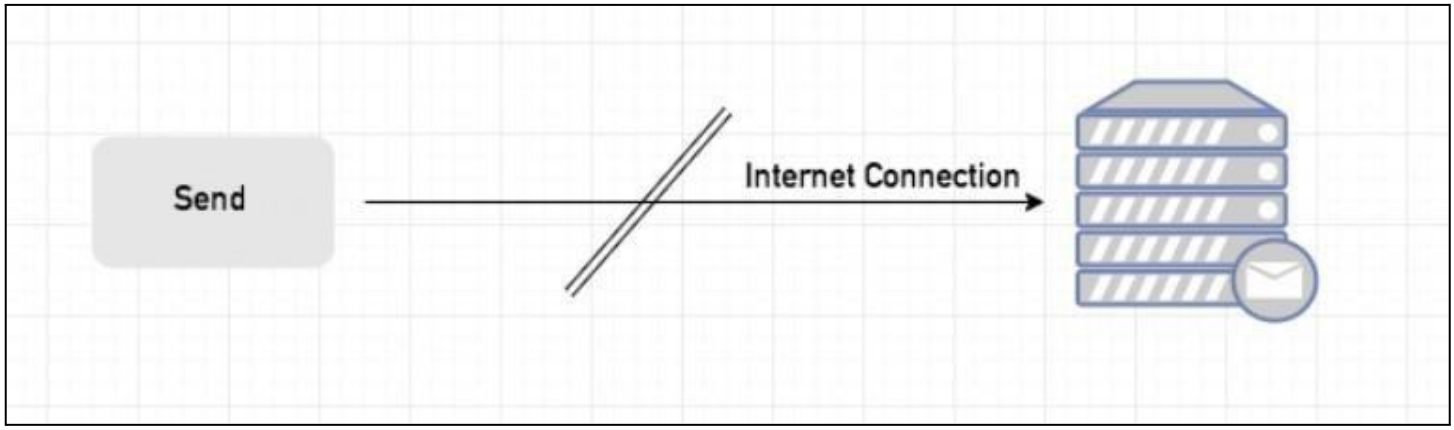
You can return dummy content or information messages to the page.

Sync Event

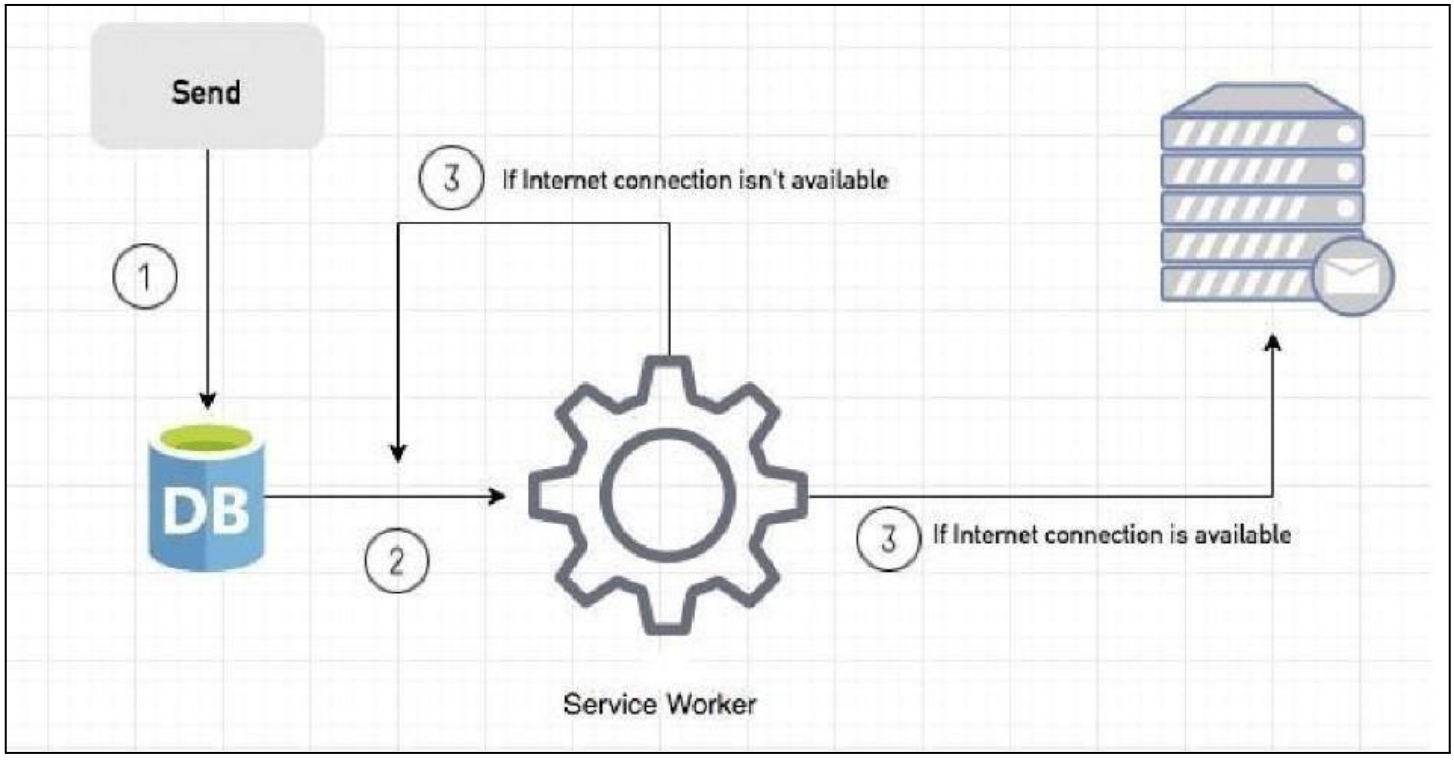
Background Sync is a Web API that is used to delay a process until the Internet connection is stable. We can adapt this definition to the real world; there is an e-mail client application that works on the browser and we want to send an email with this tool. Internet connection is broken while we are writing e-mail content and we didn't realize it. When completing the writing, we click the send button.

Here is a job for the Background Sync.

The following view shows the classical process of sending email to us. If the Internet Connection is broken, we can't send any content to Mail Server.



Here, you can create any scenario for yourself. A sample is in the following for this case.



1. When we click the "send" button, email content will be saved to IndexedDB.
2. Background Sync registration.
3. If the Internet connection is available, all email content will be read and sent to Mail Server.

If the Internet connection is unavailable, the service worker waits until the connection is available even though the window is closed. When it is available, email content will be sent to Mail Server.
You can see the working process within the following code block.

Event Listener for Background Sync Registration

Event Listener for sw.js

Push Event
This is the event that handles push notifications that are received from the server. You can apply any method with received data.

We can check in the following example.
“Notification.requestPermission();” is the necessary line to show notification to the user. If you don’t want to show any notification, you don’t need this line.

In the following code block is in sw.js file. You can handle push notifications with this event. In this example, I kept it simple. We send an object that has “method” and “message” properties. If the method value is “pushMessage”, we open the information notification with the “message” property.

CODE AND OUTPUT:

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("Vrushabh", {
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',
        '/style.css',
        '/app.js',
    ]);
  });
};

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"));
            }
        })
```

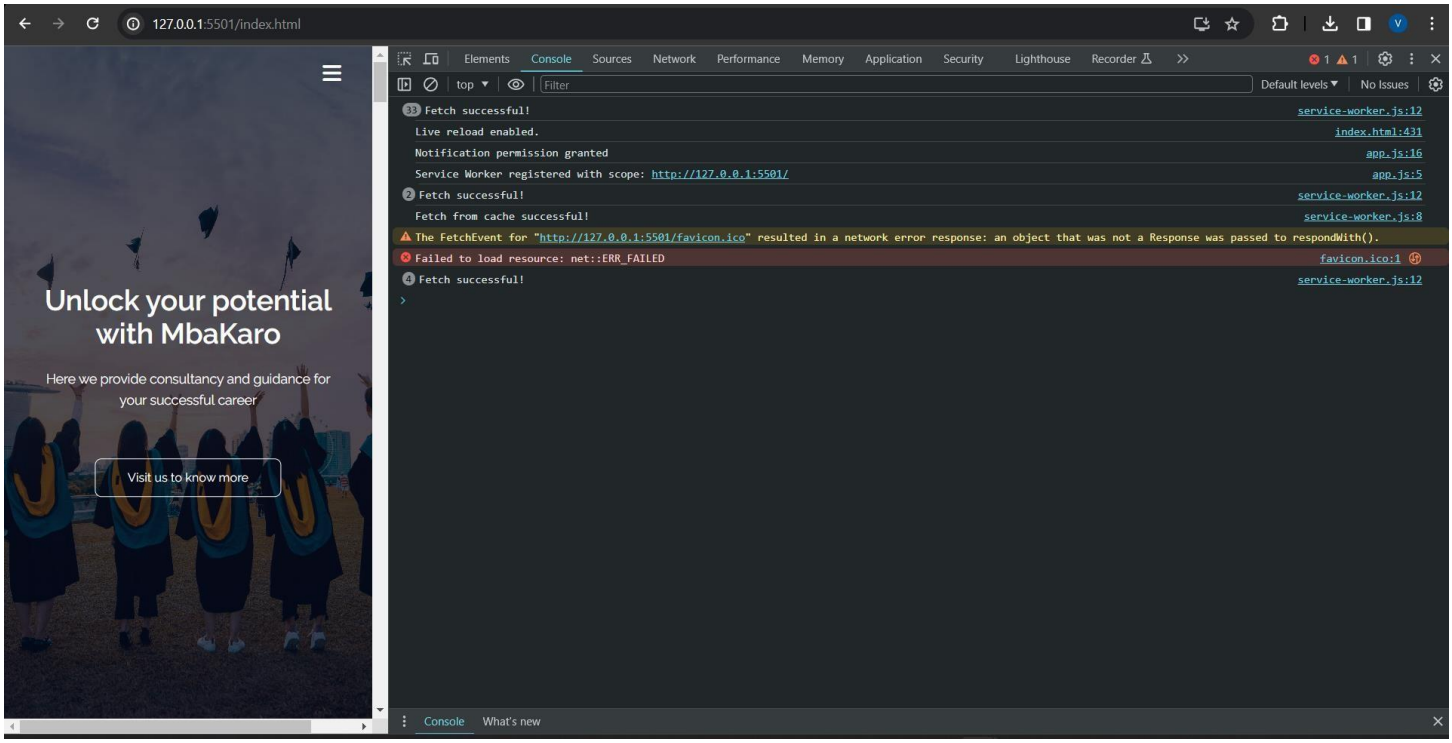
```
        .catch(function (error) {
            reject(error);
        });
  });
};

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;
        });
```

```
    });

  });

};
```

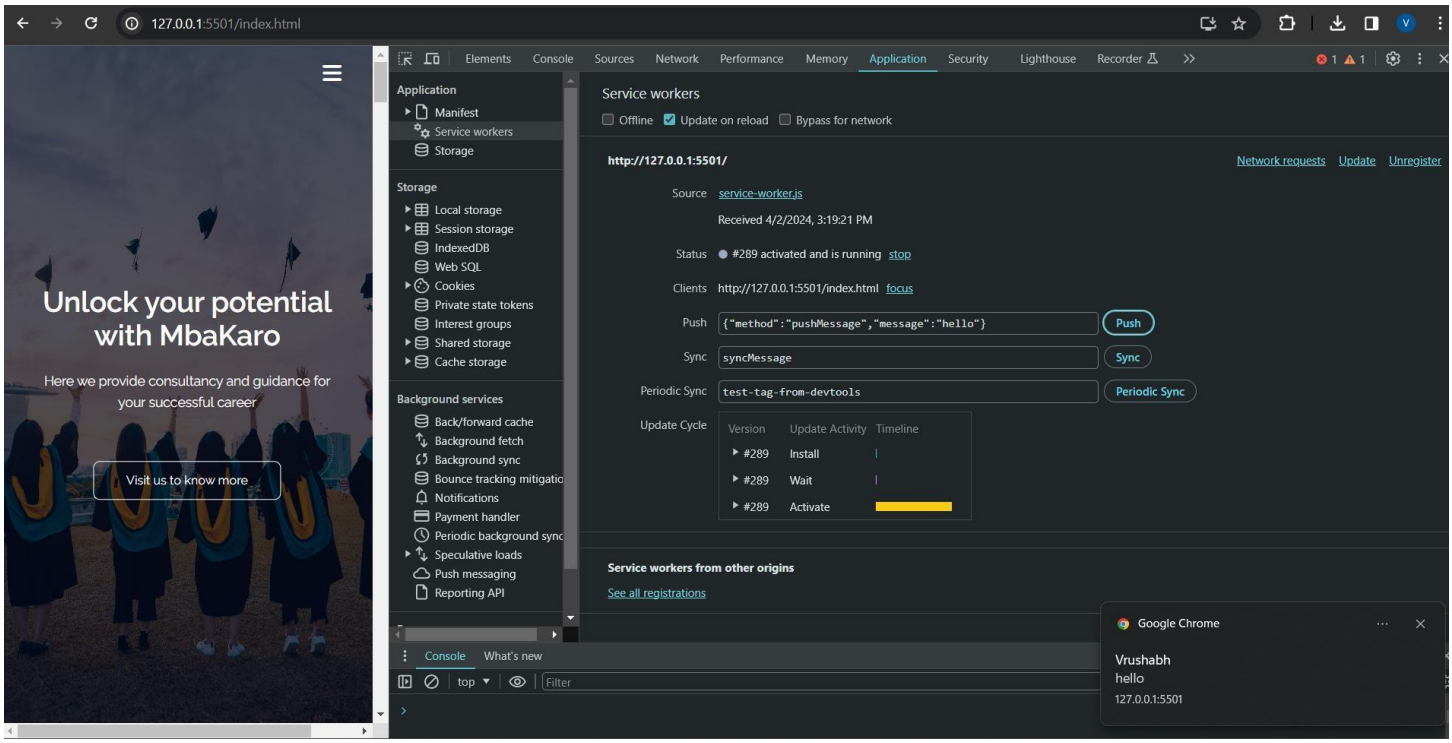
app.js

```
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);
   });
   });
   }

   if ('Notification' in window) {
   Notification.requestPermission().then(function (result) {
if (result === 'granted') {     console.log('Notification
permission granted');
   } else {
   console.warn('Notification permission denied');
   }
   });
   }
```

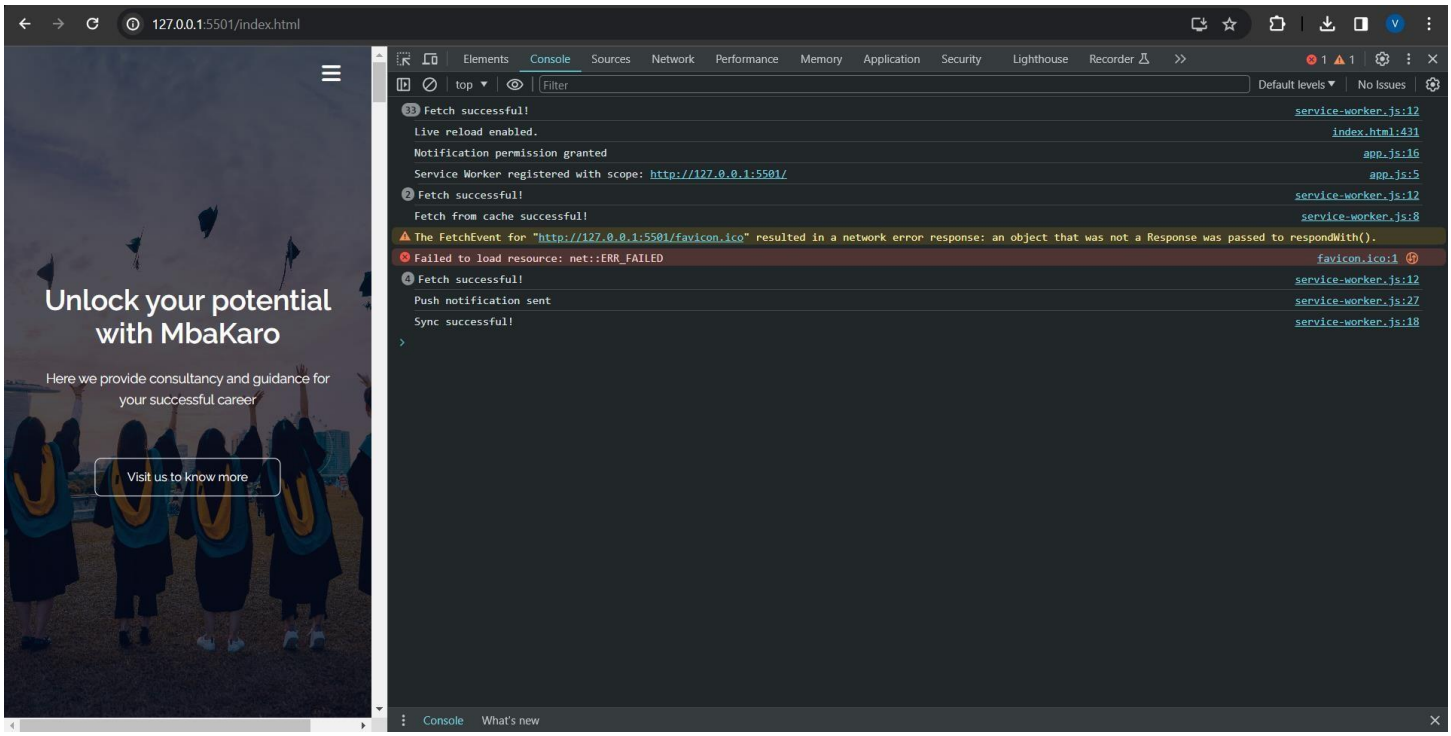
1) Fetch event



2) Push Event



2) Sync event



Conclusion:

In this experiment, we have successfully implemented service worker events like fetch, sync and push for my Twiggy 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.	22
Name	Vrushabh Ghuse
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	15M

Experiment No: - 10

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

Theory:-

GitHub is a web-based platform for version control using Git, primarily used for source code management and collaboration in software development projects. It allows developers to host their code repositories, manage changes, and coordinate work with collaborators.

GitHub Pages is a feature of GitHub that allows users to host static websites directly from their GitHub repositories. It's often used for personal or project websites, documentation, and blogs.

To host a website using GitHub Pages:

1. **Create a Repository:** Start by creating a new repository on GitHub. Name it in the format `<username>.github.io` if it's for a personal site (replace `<username>` with your GitHub username) or `<repositoryname>` if it's for a project site.

2. **Add Content:** Add your website's files (HTML, CSS, JavaScript, etc.) to the repository. Ensure that your main HTML file is named `index.html` for the site to be served correctly.

3. **Commit Changes:** Commit your changes to the repository using Git commands or GitHub's web interface.

4. **Enable GitHub Pages:** Go to your repository's settings tab on GitHub, scroll down to the GitHub Pages section, and choose the branch you want to serve your site from (usually the `main` or `master` branch). GitHub Pages will then build and deploy your site automatically.

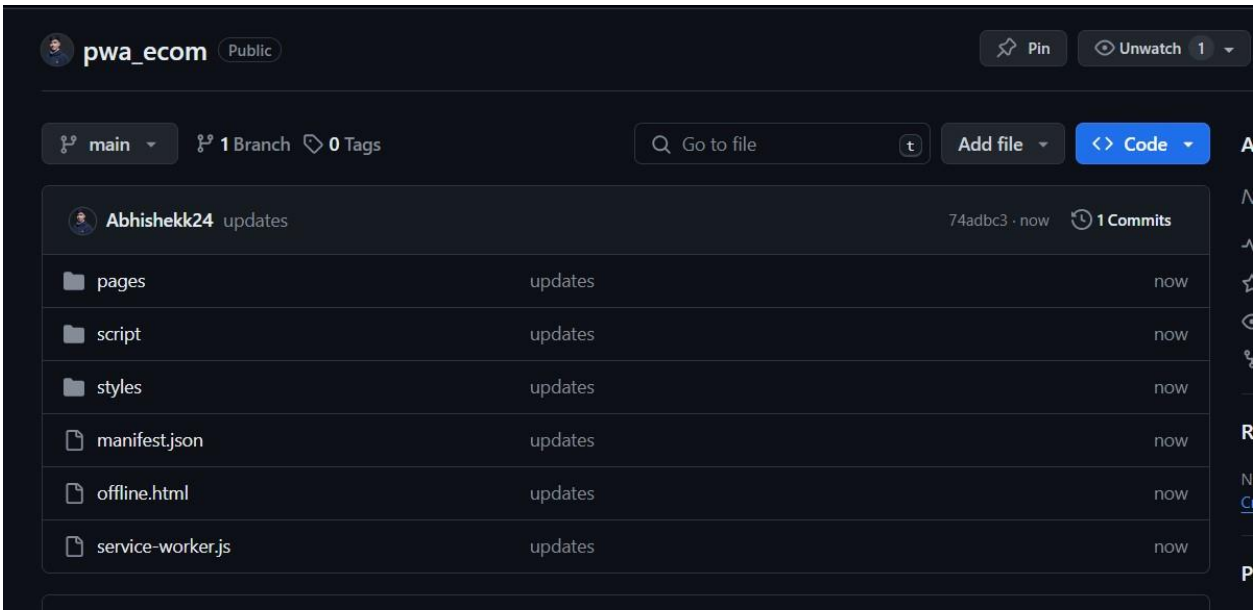
5. Custom Domain (Optional): If you have a custom domain, you can configure it to point to your GitHub Pages site by adding a CNAME file to your repository and setting up DNS records with your domain registrar.

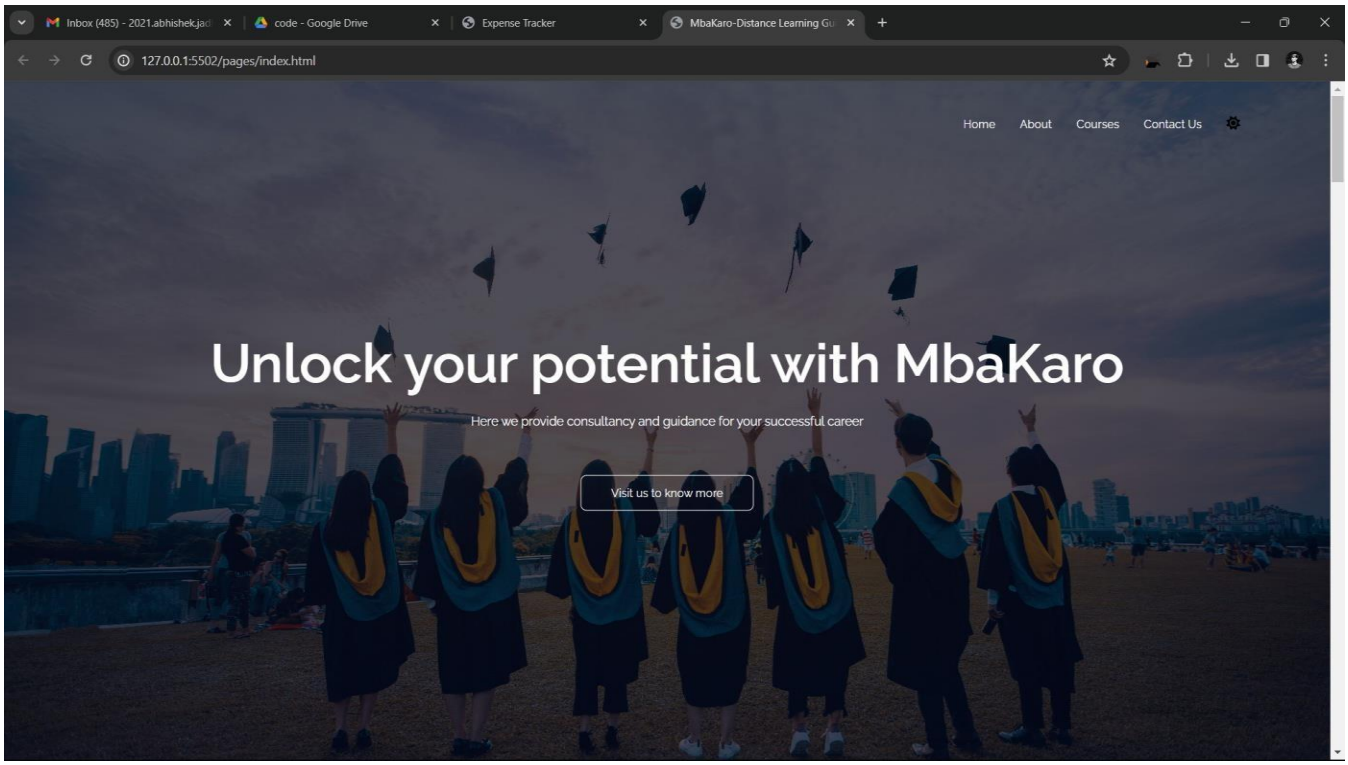
6. Access Your Site: Once GitHub Pages has deployed your site (usually just a few minutes), you can access it via `https://<username>.github.io` or `https://<repositoryname>.github.io`, depending on whether it's a personal or project site.

With these steps, you can easily host your website using GitHub Pages, leveraging GitHub's version control features and seamless deployment process.

Output:

Github Repo Link: https://github.com/Abhishekk24/pwa_ecom





Conclusion:

In conclusion, studying and implementing the deployment of an Ecommerce Progressive Web App (PWA) to GitHub Pages has provided valuable insights into the process of hosting web applications on a static hosting platform. Leveraging GitHub Pages offers a cost-effective and efficient solution for showcasing and distributing PWAs, ensuring wider accessibility and reach for users. This deployment method enhances the visibility and usability of the Ecommerce PWA, facilitating seamless access for potential customers.

MAD & PWA Lab

Journal

Experiment No.	11
Experiment Title.	To use google Lighthouse PWA Analysis Tool to test the PWA functioning.
Roll No.	22
Name	Vrushabh Ghuse
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO6: Develop and Analyze PWA Features and deploy it over app hosting solution
Grade:	15 M

PWALab

PRACTICAL11

Name : Vrushabh Ghuse **Class:D15A** **Roll no: 22**

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

Theory:

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:

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.

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, o ine functionality, performance in script-disabled environments, etc.

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 ariarequired, 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.

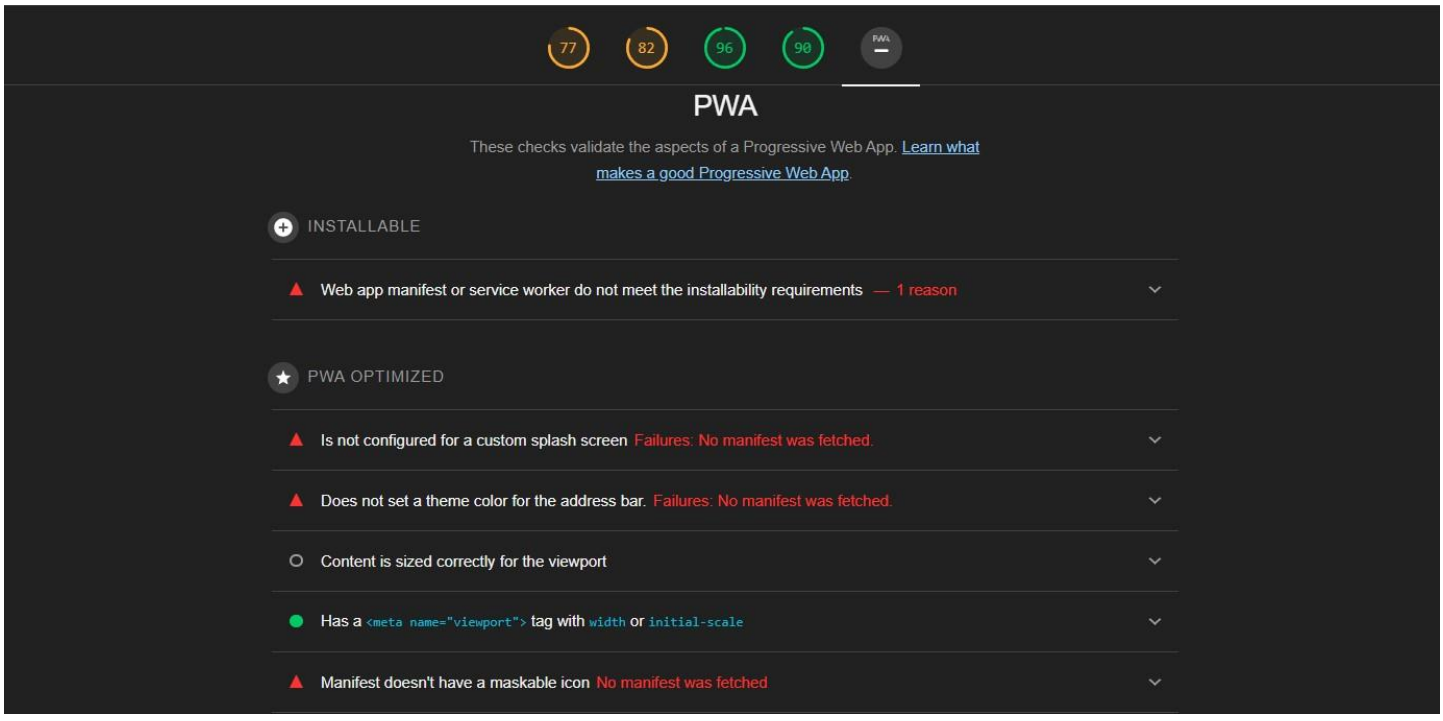
Best Practices: As any developer would know, there are a number of practices that have been deemed 'best' based on empirical data. This metric is an aggregation of many such points, including but not limited to:Use of

HTTPS

Avoiding the use of deprecated code elements like tags, directives, libraries, etc. Password input with pasteinto disabled Geo-Location and cookie usage alerts on load, etc.

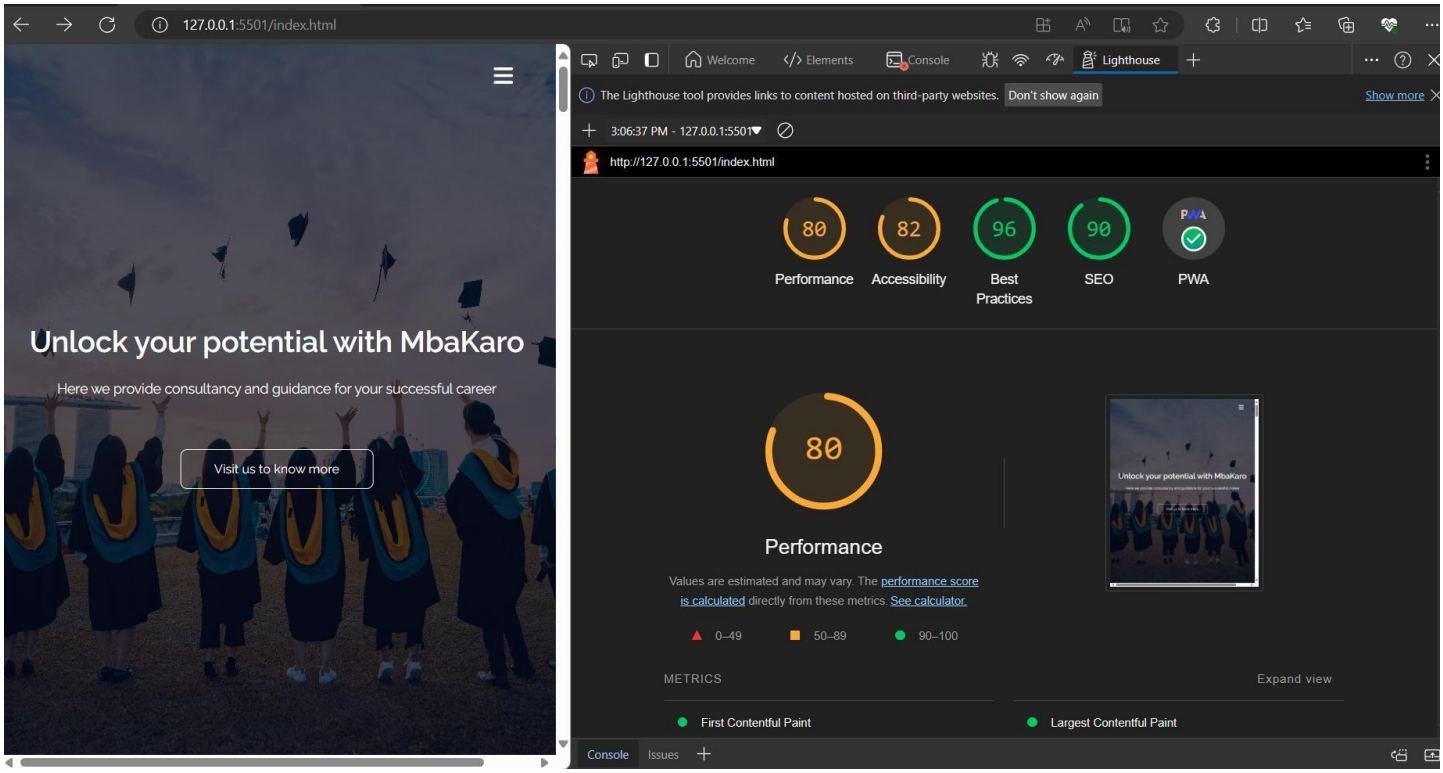
CODE AND OUTPUT: manifest.json

```
{
        "name":"Vrushabh",
        "start_url":"index.html",
        "display":"standalone",
        "background_color":"#5900b3",
        "theme_color":"black",
        "scope": ".",
        "icons":[
        {
        "src":"image/cropped.jpg",
        "sizes":"192x192",
        "type":"image/png"
        },
        {
        "src":"image/CompressJPEG.online_512x512_image.jpg",
        "sizes":"512x512",
        "type":"image/png"
        }
]
```



After changes made in manifest.json

```
{
 "name": "Vrushabh Ghuse",
 "start_url": "index.html",
 "display": "standalone",
 "background_color": "#5900b3",
 "theme_color": "black",
 "scope": ".",
 "description": "Ok",
 "icons": [
  {
   "src": "images/cropped.jpg",
   "sizes": "192x192",
   "type": "image/png",
   "purpose": "any maskable"
  },
  {
   "src": "images/CompressJPEG.online_512x512_image.jpg",
   "sizes": "512x512",
   "type": "image/png",
   "purpose": "any maskable"
  }
 ]
}
```

Conclusion:

- We analyzed the website with the help of lighthouse tool and found some issues with the website

- Hence we made changes in the manifest.json file we added "purpose":"any maskable" for maskable error face
- Also added a meta tag in index.html to resolve the theme error faced